



Freescale Semiconductor, Inc.

CodeWarrior™ Development Tools IDE 5.1 SDK API Reference

Revised: 25-Aug-2003

For More Information: www.freescale.com



Freescale Semiconductor, Inc.

Metrowerks, the Metrowerks insignia, and CodeWarrior are registered trademarks of Metrowerks Corp. in the US and/or other countries. All other trade names, trademarks and registered trademarks are the property of their respective owners.

© Copyright 2002. Metrowerks Corp. ALL RIGHTS RESERVED.

Metrowerks reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Metrowerks does not assume any liability arising out of the application or use of any product described herein. Metrowerks software is not authorized for and has not been designed, tested, manufactured, or intended for use in developing applications where the failure, malfunction, or any inaccuracy of the application carries a risk of death, serious bodily injury, or damage to tangible property, including, but not limited to, use in factory control systems, medical devices or facilities, nuclear facilities, aircraft or automobile navigation or communication, emergency systems, or other applications with a similar degree of potential hazard.

Documentation stored on electronic media may be printed for personal use only. Except for the forgoing, no portion of this documentation may be reproduced or transmitted in any form or by any means, electronic or mechanical, without prior written permission from Metrowerks.

ALL SOFTWARE, DOCUMENTATION AND RELATED MATERIALS ARE SUBJECT TO THE METROWERKS END USER LICENSE AGREEMENT FOR SUCH PRODUCT.

How to Contact Metrowerks:

Corporate Headquarters	Metrowerks Corporation 7700 West Parmer Lane Austin, TX 78729 U.S.A.
World Wide Web	http://www.metrowerks.com
Ordering & Technical Support	Voice: (800) 377-5416 Fax: (512) 996-5300

For More Information: www.freescale.com

1 Introduction	29
2 Introduction to the Plug-in API	33
3 Plug-in API Reference	35
Routines for Plug-ins	35
Plug-in Context	35
Alphabetical Routine Index	35
Functional Routine Index	37
Request Handling	37
Memory Management	37
Plug-in Data	38
Preference Data	38
File Management	38
Directory Information	38
Project File Information	38
Target Information	39
Overlay Information	39
Segment Information	39
IDE Information	39
User Interaction	39
Error Handling	39
COM Object Interfaces	39
CWAddProjectEntry	40
CWAlert	41
CWAllocateMemory	42
CWAllocMemHandle	43
CWCreateNewTextDocument	44
CWDonePluginRequest	45
CWFindAndLoadFile	45
CWFreeMemHandle	47
CWFreeMemory	48
CWGetAccessPathInfo	49
CWGetAccessPathListInfo	50
CWGetAccessPathSubdirectory	51
CWGetAPIVersion	52
CWGetCallbackOSError	53

CWGetCOMApplicationInterface	53
CWGetCOMProjectInterface	54
CWGetCOMDesignInterface	54
CWGetCOMTargetInterface	55
CWGetFileInfo	55
CWGetFileText	56
CWGetFrameworkCount	58
CWGetFrameworkInfo	58
CWGetFrameworkSharedLibrary	59
CWGetIDEInfo	59
CWGetMemHandleSize	60
CWGetNamedPreferences	61
CWGetOutputFileDirectory	61
CWGetOverlay1FileInfo	62
CWGetOverlay1GroupInfo	63
CWGetOverlay1GroupsCount	64
CWGetOverlay1Info	64
CWGetPluginData	65
CWGetPluginRequest	66
CWGetProjectFile	67
CWGetProjectFileCount	67
CWGetSegmentInfo	68
CWGetTargetDataDirectory	68
CWGetTargetName	69
CWLockMemHandle	70
CWMacOSErrToCWResult	71
CWPostDialog	72
CWPostFileAction	72
CWPreDialog	73
CWPreFileAction	73
CWReleaseFileText	74
CWRemoveProjectEntry	75
CWReportMessage	75
CWResizeMemHandle	77
CWResolveRelativePath	77
CWSetModDate	78
CWSetPluginOSError	79

CWShowStatus	80
CWStorePluginData	81
CWUnlockMemHandle	82
CWUserBreak	83
Plug-In Entry Points	84
Main Plug-in Entry Point	84
CWPlugin_GetDropInFlags Entry Point	85
CWPlugin_GetDropInName Entry Point	86
CWPlugin_GetDisplayName Entry Point	86
CWPlugin_GetPluginInfo	87
Data Structures for Plug-ins	88
CWAccessPathInfo	89
CWAccessPathListInfo	90
CWAccessPathType.	92
CWAddr64	92
CWDataType	93
CWDependencyType	93
CWFileInfo	95
CWFileName	97
CWFileSpec	98
CWFileTime	100
CWFrameworkInfo.	100
CWIDEInfo	101
CWMemHandle	102
CWMessageRef	102
CWNewProjectEntryInfo	103
CWNewTextDocumentInfo	105
CWOSResult	106
CWOOverlay1FileInfo	106
CWOOverlay1GroupInfo	107
CWOOverlay1Info	107
CWPluginContext	108
CWPluginInfo	108
CWProjectFileInfo	109
CWProjectSegmentInfo	113
CWRelativePath	113
CWRelativePathFormat	115

CWRelativePathTypes	116
CWResult	117
DropInFlags	117
Constants for Plug-ins	119
CWDROPINCOMPILETYPE	120
CWDROPINLINKERTYPE	120
CWDROPINPREFSTYPE	121
CWDROPINPREFSTYPE_1	121
CWDROPINVCSTYPE	122
cwFileTypePrecompiledHeader	122
cwFileTypeText	123
cwFileTypeUnknown	123
cwSystemPath	123
cwUserPath	124
kCurrentCompiledFile	124
kDefaultLinkPosition	124
kTargetGlobalPluginData	125
messagetypeInfo	125
messagetypeWarning	125
messagetypeError	126
reqAbout	126
reqIdle	127
reqInitialize	127
reqPrefsChange	128
reqTerminate	128
Result Codes for Plug-ins	129
cwErrCantSetAttribute	129
cwErrFileNotFound	130
cwErrInvalidCallback	130
cwErrInvalidMPCallback	130
cwErrInvalidParameter	131
cwErrOSError	131
cwErrOutOfMemory	131
cwErrRequestFailed	131
cwErrSilent	132
cwErrStringBufferOverflow	132
cwErrUnknownFile	132

cwErrUserCanceled	133
cwNoErr	133

4 Compiler and Linker Plug-in Reference 135

Overview	135
Routines for Compiler and Linker Plug-ins.	135
Alphabetical Routine Index	135
Functional Routine Index	136
Current File Information	136
Request Handling	137
Managing Object Data	137
Managing Precompiled Headers	137
Managing Target Information	137
Managing Target Storage	137
Target Runtime Settings	137
User Interaction	138
CWCachePrecompiledHeader	138
CWDisplayLines	139
CWFreeObjectData	140
CWGetBrowseOptions	140
CWGetBuildSequenceNumber	141
CWGetCommandLineArgs	142
CWGetEnvironmentVariable.	143
CWGetEnvironmentVariableCount	144
CWGetMainFileID	145
CWGetMainFileNumber	146
CWGetMainFileSpec	147
CWGetMainFileText	147
CWGetModifiedFiles	148
CWGetPrecompiledHeaderSpec	149
CWGetResourceFile	150
CWGetStoredObjectFileSpec.	151
CWGetSuggestedObjectFileSpec	152
CWGetTargetInfo	153
CWGetTargetStorage	154
CWGetWorkingDirectory	156
CWIsAutoPrecompiling.	157

CWIsCachingPrecompiledHeaders	158
CWIsGeneratingDebugInfo	158
CWIsPrecompiling	159
CWIsPreprocessing	160
CWLoadObjectData	161
CWPutResourceFile	162
CWSetTargetInfo	163
CWSetTargetStorage	164
CWStoreObjectData	165
User Routines for Compiler and Linker Plug-ins	166
CWPlugin_GetDefaultMappingList Entry Point	167
CWPlugin_GetTargetList Entry Point	168
Helper_GetCompilerBrSymbols Entry Point	168
Helper_Unmangle Entry Point	171
Data Structures for Compiler and Linker Plug-ins	172
CWBrowseOptions	173
CWCompilerBrSymbol	174
CWCompilerBrSymbolInfo	175
CWCompilerBrSymbolList	175
CWDependencyInfo	176
CWExtensionMapping	177
CWExtMapList	178
CWObjectData	180
CWTargetInfo	183
CWTargetList	188
CWUnmangleInfo	190
Constants for Compiler and Linker Plug-ins	191
cantDisassemble	194
cwAccessAbsolute	195
cwAccessPathRelative	195
cwAccessFileName	195
cwAccessFileRelative	196
DROPINCOMPILERLINKERAPIVERSION	196
exelinkageFlat	196
exelinkageOverlay1	197
exelinkageSegmented	197
isPostLinker	198

isPreLinker	198
kCandisassemble.	198
kCanimport	198
kCanprecompile	199
kCanpreprocess	199
kCompAllowDupFileNames.	199
kCompAlwaysReload.	200
kCompEmitsOwnBrSymbols	200
kCompMultiTargAware	201
kCompReentrant	201
kCompRequiresFileListBuildStartedMsg	201
kCompRequiresProjectBuildStartedMsg.	201
kCompRequiresSubProjectBuildStartedMsg	202
kCompRequiresTargetBuildStartedMsg	202
kCompRequiresTargetCompileStartedMsg	202
kCompSavesDbgPreprocess	202
kCompUsesTargetStorage	203
kGeneratescode	203
kGeneratesrsrscs	203
kIgnored	203
kLaunchable.	204
kIsMPAAware.	204
kIsppascal	204
kPersistent	204
kPrecompile	205
kRsrcfile	205
linkAllowDupFileNames	205
linkAlwaysReload	205
linkMultiTargAware	206
linkOutputNone	206
linkOutputFile	206
linkOutputDirectory	207
linkRequiresFileListBuildStartedMsg	207
linkRequiresProjectBuildStartedMsg	207
linkRequiresSubProjectBuildStartedMsg.	208
linkRequiresTargetBuildStartedMsg	208
linkRequiresTargetLinkStartedMsg	208

linkerDisasmRequiresPreprocess	209
linkerGetTargetInfoThreadSafe.	209
linkerInitializeOnMainThread	209
linkerSuggestsNonRecursiveAccessPaths	209
linkerUnmangles.	210
linkerUsesCaseInsensitiveSymbols	210
linkerUsesFrameworks	210
linkerUsesTargetStorage.	211
linkerWantsPreRunRequest	211
magicCapLinker	211
reqCheckSyntax	211
reqCompile	211
reqCompDisassemble.	212
reqDisassemble	213
reqFileListBuildEnded	213
reqFileListBuildStarted	214
reqLink	214
reqMakeParse	214
reqPreprocessForDebugger	215
reqPreRun.	215
reqProjectBuildEnded.	215
reqProjectBuildStarted	215
reqSubProjectBuildEnded	216
reqSubProjectBuildStarted.	216
reqTargetBuildEnded	217
reqTargetBuildStarted.	217
reqTargetCompileEnded	218
reqTargetCompileStarted	218
reqTargetInfo	218
reqTargetLinkEnded	219
reqTargetLinkStarted	219
reqTargetLoaded	219
reqTargetPrefsChanged	220
reqTargetUnloaded	220
targetCPU68K	221
targetCPUAny	222
targetCPUEmbeddedPowerPC.	222

targetCPUi80x86	222
targetCPUMips	223
targetCPUNECv800	223
targetCPUPowerPC	224
targetOSAny.	224
targetOSEmbeddedABI	225
targetOSMacintosh	225
targetOSMagicCap	225
targetOSOS9.	226
targetOSWindows	226
Result Codes for Compiler and Linker Plug-ins	227
cwErrObjectFileNotStored.	227
cwErrUnknownSegment	228

5 Browser Reference

229

Browser Records	229
Browse Header	230
Browser Data Stream	231
Fields For All Records	232
Additional Fields For Functions	236
Additional Fields For Classes	237
Class Member List	239
Additional Field For Templates	242
Additional Field For Global Variables	242
Data Structures for the Browser.	243
BrowseHeader	243
Constants for the Browser	244
BROWSE_EARLIEST_COMPATIBLE_VERSION	244
BROWSE_HEADER	245
BROWSE_VERSION	245
EAccess	245
EBrowserItem	246
ELanguage	247
EMember	248
ETemplateType	249

6 Settings Panel Plug-in API Reference

251

Routines for Settings Panel Plug-ins	251
Organization of Function Reference.	251
Plug-in Context	252
Alphabetical Routine Index	252
Functional Routine Index	256
Manipulating Panel Controls	256
Getting and Setting Text Controls	256
Obtaining Settings Handles	257
Handling Panel Events (Windows)	257
Manipulating Combo Box Item Lists (Windows)	257
Obtaining IDE State (Windows)	257
Manipulating Controls (Mac OS).	257
Custom Controls (User Items) (Mac OS)	258
Manipulating Relative Paths	258
Reading Primitive XML Types	258
Writing Primitive XML Types	258
Reading and Writing XML Structures and Arrays	259
Reading XML Structure and Array Elements	259
Writing XML Structure and Array Elements	259
CWGetArraySettingElement.	259
CWGetArraySettingSize.	262
CWGetBooleanValue	262
CWGetFloatingPointValue.	263
CWGetIntegerValue	264
CWGetNamedSetting.	264
CWGetRelativePathValue	266
CWGetStringValue	266
CWGetStructureSettingField.	267
CWPanelActivateItem	270
CWPanelAppendItems	270
CWPanelChooseRelativePath	272
CWPanelDeleteListItem.	274
CWPanelEnableItem	274
CWPanelGetCurrentPrefs	275
CWPanelGetDebugFlag	276
CWPanelGetDialogItemHit	277

CWPanelGetFactoryPrefs	278
CWPanelGetItemData	279
CWPanelGetItemMaxLength	280
CWPanelGetItemText	280
CWPanelGetItemTextHandle	282
CWPanelGetItemValue	283
CWPanelGetListItemText	284
CWPanelGetNumBaseDialogItems	285
CWPanelGetOriginalPrefs	285
CWPanelGetPanelPrefs	287
CWPanelGetRelativePathString	288
CWPanelInsertListItem	290
CWPanelInvalItem	291
CWPanelSetFactoryFlag	291
CWPanelSetItemData	292
CWPanelSetItemMaxLength	293
CWPanelSetItemText	294
CWPanelSetItemTextHandle	295
CWPanelSetItemValue	296
CWPanelSetListItemText	297
CWPanelSetRecompileFlag	298
CWPanelSetRelinkFlag	299
CWPanelSetReparsingFlag	299
CWPanelSetResetPathsFlag	300
CWPanelSetRevertFlag	301
CWPanelShowItem	302
CWPanelValidItem	303
CWPanelActivateItem	303
CWPanelAppendItems	303
CWPanelChooseRelativePath	304
CWPanelDrawPanelBox	304
CWPanelDrawUserItemBox	305
CWPanelEnableItem	305
CWPanelGetArraySettingElement	305
CWPanelGetArraySettingSize	306
CWPanelGetBooleanValue	306
CWPanelGetFloatingPointValue	306

CWPanelGetIntegerValue	306
CWPanelGetItemControl	306
CWPanelGetItemData	307
CWPanelGetItemMaxLength	308
CWPanelGetItemRect	308
CWPanelGetItemText	308
CWPanelGetItemTextHandle	309
CWPanelGetItemValue.	309
CWPanelGetMacPort	309
CWPanelGetNamedSetting.	310
CWPanelGetPanelPrefs	310
CWPanelGetRelativePathString	310
CWPanelGetRelativePathValue	311
CWPanelGetStringValue	311
CWPanelGetStructureSettingField.	311
CWPanelInstallUserItem	311
CWPanelInvalItem	312
CWPanelReadBooleanSetting.	312
CWPanelReadFloatingPointSetting	312
CWPanelReadIntegerSetting	312
CWPanelReadRelativePathAEDesc	312
CWPanelReadRelativePathSetting.	313
CWPanelReadStringSetting.	313
CWPanelRemoveUserItem	314
CWPanelSetBooleanValue	314
CWPanelSetFloatingPointValue	314
CWPanelSetIntegerValue.	314
CWPanelSetItemData	315
CWPanelSetItemMaxLength	315
CWPanelSetItemText	315
CWPanelSetItemTextHandle	315
CWPanelSetItemValue	316
CWPanelSetRelativePathValue	316
CWPanelSetStringValue	316
CWPanelShowItem	316
CWPanelValidItem	316
CWPanelWriteBooleanSetting.	317

CWPanlWriteFloatingPointSetting	317
CWPanlWriteIntegerSetting	317
CWPanlWriteRelativePathAEDesc	317
CWPanlWriteRelativePathSetting	318
CWPanlWriteStringSetting	318
CWReadBooleanSetting	319
CWReadFloatingPointSetting	319
CWReadIntegerSetting	320
CWReadRelativePathSetting.	321
CWReadStringSetting.	322
CWSetBooleanValue	323
CWSetFloatingPointValue	324
CWSetIntegerValue.	325
CWSetRelativePathValue	326
CWSetStringValue	327
CWWriteBooleanSetting.	328
CWWriteFloatingPointSetting	329
CWWriteIntegerSetting	330
CWWriteRelativePathSetting	330
CWWriteStringSetting	331
User Routines for Settings Panel Plug-ins	332
CWPlugin_GetHelpInfo Entry Point	332
Data Structures for Settings Panel Plug-ins	333
CWDialog.	333
CWHelpInfo.	333
CWSettingID	334
PanelFlags.	335
PanelParamBlkPtr	336
PanelParameterBlock	336
Constants for Settings Panel Plug-ins	344
CW_STRICT_DIALOGS.	345
DROPINPANELAPIVERSION	346
kCurrentCWHelpInfoVersion	346
menu_Clear	346
menu_Copy	347
menu_Cut.	347
menu_Paste	348

menu_SelectAll	348
panelScopeGlobal	348
panelScopeProject	349
panelScopeTarget	349
reqActivateItem	350
reqAEGetPref	350
reqAESetPref	351
reqByteSwapData	351
reqDeactivateItem	352
reqDragDrop	352
reqDragEnter	353
reqDragExit	354
reqDragWithin	355
reqDrawCustomItem	356
reqFilter.	356
reqFindStatus	357
reqFirstLoad	357
reqGetData	358
reqGetFactory	358
reqHandleClick	359
reqHandleKey	360
reqInitDialog	360
reqInitPanel	361
reqItemHit	361
reqObeyCommand	363
reqPutData	363
reqReadSettings	364
reqRenameProject	365
reqSetupDebug	365
reqTermDialog.	366
reqTermPanel	366
reqUpdatePref	367
reqValidate	367
reqWriteSettings	368
supportsByteSwapping	369
supportsTextSettings	369
usesStrictAPI	370

Settings Panel Result Codes	370
kBadPrefVersion	371
kMissingPrefErr	371
kSettingNotFoundErr.	371
kSettingTypeMismatchErr.	372
kInvalidCallbackErr	372
kSettingOutOfRangeErr.	372

7 Version Control System Plug-in API

373

Plug-In Callbacks	373
Standard Callbacks	374
VCS Callbacks.	377
CWAllowV1Compatibility	379
CWGetComment	380
CWFileStateChanged.	380
CWVCSStateChanged	381
CWDoVisualDifference	384
CWCompletionRatio	385
CWGetProjectFileSpecifier	386
CWIAdvancedRequest.	386
CWIRecursiveRequest	386
CWICommandSupportedRequest.	387
CWSetCommandDescription	388
CWGetDatabaseConnectionInfo	389
CWGetCommandStatus	390
CWSetCommandStatus.	391
CWGetVCSItemCount	391
CWGetVCSItem	391
CWSetVCSItem	393
CWGetVCSPointerStorage	393
CWSetVCSPointerStorage.	394
VCS Commands	394
Determining the Mode	394
Negotiation Mode	395
Execution Mode	395
Command Types.	395
Advanced Options	396

Reporting Status	. 396
VCS Requests	. 397
Initialization and Termination	. 398
reqInitialize	. 398
reqTerminate	. 399
Database Connection and Disconnection	. 399
reqDatabaseConnect	. 399
reqDatabaseDisconnect	. 400
Queries	. 400
reqAbout	. 400
reqDatabaseVariables.	. 401
Information	. 401
reqIdle	. 401
reqPrefsChange	. 402
File Processing.	. 402
reqFileAdd	. 402
reqFileBranch	. 403
reqFileCheckin	. 403
reqFileCheckout	. 403
reqFileComment.	. 404
reqFileDelete	. 404
reqFileDestroy.	. 404
reqFileDifference.	. 405
reqFileGet	. 405
reqFileHistory.	. 405
reqFileLabel.	. 406
reqFileProperties.	. 406
reqFilePurge	. 406
reqFileRename	. 407
reqFileRollback	. 407
reqFileShare.	. 407
reqFileStatus	. 408
reqFileUndoCheckout	. 408
reqFileVersion.	. 408
reqFileView	. 409
VCS API Version 1 Compatibility	. 409
Version 1 VCS API	. 409

Installing on a Pre-Pro 4 CodeWarrior Installation	410
Behavioral Differences	410
Supported Callbacks	411
Unsupported Requests	412
Porting a Version 1 VCS Plug-in to Version 7+	412
V1 to V7 Callback Mappings	414
V1 Param Block Members to Accessor Mappings.	414
8 Introduction to the COM API	419
Requirements	419
What You Should Already Know	420
Starting Points	420
Services	420
Callbacks	420
9 Access Paths	423
Access Paths API Reference	423
ICodeWarriorAccessPath	424
Inherited Interfaces	424
Methods	424
ICodeWarriorAccessPaths	429
Inherited Interfaces	429
Methods	429
ICodeWarriorUserTree	436
Inherited Interfaces	436
Methods	436
Access Paths Data Types	440
EAccessPathLocation	440
EAccessPathType	441
EUserDefinedTree	441
10 Application	443
Application API Overview	443
Application API Reference	444
ICodeWarriorApp	445
Inherited Interfaces	445
Methods	445
ICodeWarriorAppEvents	475

Inherited Interfaces	475	
Methods	475	
ICodeWarriorCompare	479	
Inherited Interfaces	479	
Methods	479	
Application Data Types	482	
ECodeWarriorConvertOption	482	
ECodeWarriorRevertPanelOption	482	
ECodeWarriorProjectOption.	483	
ECodeWarriorSaveOption.	483	
ECodeWarriorVCSInteractionOption	484	
Standard Folder Names.	484	
11 Collections		487
Collections API Overview	487	
Using the Collections API	487	
Collections API Reference	488	
CodeWarrior Collections	489	
Inherited Interfaces	489	
Methods	489	
12 Commands		493
Commands API Overview	493	
Using the Commands API	494	
Creating a Command Group	494	
Assigning a Command Handler	495	
Registering a Command	495	
Displaying a Command Group	496	
Commands API Reference	498	
ICodeWarriorCommandHandler	499	
Inherited Interfaces	499	
Methods	499	
ICodeWarriorCommandRegistry	502	
Inherited Interfaces	502	
Methods	502	
ICodeWarriorDeferredAction	504	
Inherited Interfaces	504	

Methods	504
Commands Data Types	505
Command status.	505
Menu Commands	505
SRegisterCommandGroup	506
13 Components	509
Components API Overview	509
Components API Reference	509
ICodeWarriorComponent	510
Inherited Interfaces	510
Methods	510
ICodeWarriorComponentEvent	514
Inherited Interfaces	514
Methods	514
ICodeWarriorComponentEventSet	517
Inherited Interfaces	517
Methods	517
ICodeWarriorComponentProperty	519
Inherited Interfaces	519
Methods	519
14 Creatable Items	523
Creatable Items API Reference	523
ICodeWarriorCreatableItem	524
Inherited Interfaces	524
Methods	524
ICodeWarriorCreateFileItem	527
Inherited Interfaces	527
Methods	527
ICodeWarriorCreateObjectItem	530
Inherited Interfaces	530
Methods	530
ICodeWarriorCreateProjectItem	534
Inherited Interfaces	534
Methods	534
Creatable Items Data Types	537

Built-in Icon Index Values	538	
Creatable Item Category Constants.	538	
ECreateProjectType	538	
15 Designs		541
Designs API Overview	541	
Designs API Reference	541	
IcodeWarriorDesign	542	
Inherited Interfaces	542	
Methods	542	
IcodeWarriorDesignAttachment	553	
Inherited Interfaces	553	
Methods	553	
IcodeWarriorDesignEvents	555	
Inherited Interfaces	555	
Methods	555	
Data Types	557	
EcodeWarriorLinkFlags	557	
16 Dialog Services		559
Dialog Services API Overview	559	
Using the Dialog Services API	559	
Registering the Command	559	
Implementing the Command	561	
Dialog Services API Reference	563	
IcodeWarriorDialogServices.	563	
Inherited Interfaces	563	
Methods	563	
Dialog Services Data Types	569	
Dialog Box Types	569	
SaveDontSaveDialog Result Types	569	
17 Documents		571
Documents API Overview	571	
Documents API Reference	571	
IcodeWarriorDocument.	572	
Inherited Interfaces	572	
Methods	572	

ICodeWarriorProjectDocument.	580	
Inherited Interfaces	580	
Methods	580	
ICodeWarriorTextDocument.	583	
Inherited Interfaces	583	
Methods	583	
18 Error Info		589
Error Info API Overview.	589	
Error Info API Reference.	589	
ICodeWarriorErrorInfo	590	
Inherited Interfaces	590	
Methods	590	
19 Files		599
IFileSpec	599	
Inherited Interfaces	599	
Methods	599	
20 Menus		603
Menus API Overview	603	
Using the Menus API	603	
Menus API Reference	604	
ICodeWarriorMenu.	605	
Inherited Interfaces	605	
Methods	605	
ICodeWarriorMenuHandler	610	
Methods	610	
ICodeWarriorMenuManager.	612	
Methods	612	
21 Messages		615
Messages API Overview.	615	
Messages API Reference.	615	
ICodeWarriorBuildMessages.	616	
Inherited Interfaces	616	
Properties.	616	
ICodeWarriorMessage	621	

Inherited Interfaces	. 621	
Properties.	. 621	
Message Data Types	. 628	
EMsgType	. 628	
22 Projects		629
Projects API Overview	. 629	
Projects API Reference.	. 629	
IcodeWarriorProject	. 630	
Inherited Interfaces	. 630	
Methods	. 630	
IcodeWarriorProjectAssociation	. 651	
Inherited Interfaces	. 651	
Methods	. 651	
IcodeWarriorProjectEvents	. 653	
Inherited Interfaces	. 653	
Methods	. 653	
IcodeWarriorProjectFile.	. 659	
Inherited Interfaces	. 659	
Methods	. 659	
Project Data Types	. 662	
EcodeWarriorBuildOptions	. 662	
EcodeWarriorCompileChoice	. 662	
EcodeWarriorRunMode	. 662	
EpluginDataStorageLoc.	. 663	
23 Symbols		665
Symbols API Reference	. 665	
IcodeWarriorBaseClassInfo	. 666	
Inherited Interfaces	. 666	
Methods	. 666	
IcodeWarriorClass	. 668	
Inherited Interfaces	. 668	
Methods	. 668	
IcodeWarriorDataMember	. 675	
Inherited Interfaces	. 675	
Methods	. 675	

ICodeWarriorMethod	. 677
Inherited Interfaces	. 677
Methods	. 677
ICodeWarriorScope	. 683
Inherited Interfaces	. 683
Methods	. 683
ICodeWarriorSourceContext	. 686
Inherited Interfaces	. 686
Methods	. 686
ICodeWarriorSymbol	. 690
Inherited Interfaces	. 690
Methods	. 690
ICodeWarriorSymbolContainer	. 694
Inherited Interfaces	. 694
Methods	. 694
Symbols Data Types	. 703
ECodeWarriorAccess	. 703
ECodeWarriorFindSymbolsMode	. 703
ECodeWarriorShowSymbolLocation	. 704
ECodeWarriorSymbolType	. 704
24 Targets	705
Targets API Overview	. 705
Targets API Reference	. 705
ICodeWarriorTarget	. 706
Inherited Interfaces	. 706
Methods	. 706
ICodeWarriorTargetFile	. 743
Inherited Interfaces	. 743
Methods	. 743
ICodeWarriorTargetOutput	. 751
ICodeWarriorSubTarget	. 753
Inherited Interfaces	. 753
Methods	. 753
ICodeWarriorSubProjectTarget	. 755
Inherited Interfaces	. 755
Methods	. 755

ICodeWarriorSegment	.757	
Inherited Interfaces	.757	
Methods	.757	
Targets Data Types	.764	
ECodeWarriorBrowserGenerator.	.764	
ECodeWarriorTargetLinkOrderType	.764	
ECodeWarriorTargetOutputKind	.764	
ECodeWarriorWhichTargetOptions	.765	
25 Text		767
Text API Overview	.767	
Text API Reference	.767	
ICodeWarriorTextEngine	.768	
Inherited Interfaces	.768	
Methods	.768	
26 Toolbar		777
Toolbar API Overview.	.777	
Toolbar API Reference.	.777	
ICodeWarriorCustomToolBarItem	.779	
Inherited Interfaces	.779	
Methods	.779	
ICodeWarriorPopupMenuToolBarItem	.782	
Inherited Interfaces	.782	
Methods	.782	
ICodeWarriorToggleButtonToolBarItem	.787	
Inherited Interfaces	.787	
Methods	.787	
ICodeWarriorToolBar	.789	
Inherited Interfaces	.789	
Methods	.789	
ICodeWarriorToolBarInstanceCreationNotification	.796	
Inherited Interfaces	.796	
Methods	.796	
ICodeWarriorToolBarItemHelp.	.798	
Inherited Interfaces	.798	
ICodeWarriorToolBarItemRegistry	.799	

Inherited Interfaces	. 799	
Toolbar Data Types	. 801	
SPopupMenuToolbarItem	. 801	
CWToolbarItemID	. 801	
CWToolbarIconRegistryInfo	. 802	
Item Flags	. 802	
27 Version Control		803
Version Control API Reference	. 803	
ICodeWarriorVersionControl	. 804	
Inherited Interfaces	. 804	
ICodeWarriorVCSState	. 809	
Inherited Interfaces	. 809	
Methods	. 809	
ICodeWarriorVCSFileStateListener	. 811	
Inherited Interfaces	. 811	
Methods	. 811	
VCS Data Types	. 812	
ECodeWarriorVCSCKIDState	. 812	
ECodeWarriorVCSDBState	. 813	
ECodeWarriorVCSFileLockState	. 813	
28 Windows		815
Windows API Overview	. 815	
Using the Windows API	. 815	
Windows API Reference	. 816	
ICodeWarriorWindowManager	. 817	
Inherited Interfaces	. 817	
Methods	. 817	
ICodeWarriorWindow	. 821	
Inherited Interfaces	. 821	
Methods	. 821	
ICodeWarriorWindowEvents	. 833	
Inherited Interfaces	. 833	
Methods	. 833	
Windows Data Types	. 840	
CWNativeXWindowPart	. 840	

A CodeWarrior IDE Interface Definition Language (IDL)	841
B PowerPC Object Code (Mac OS)	897
PowerPC Object Code Structure	898
PowerPC Object Header	898
PowerPC Object Data Section	900
Preventing Dead-Stripping	902
PowerPC Simple Hunks.	902
PowerPC Regular Code Hunks.	902
PowerPC Data Hunks.	905
PowerPC Alternate Entry Point Hunks	907
PowerPC Cross-Reference Hunks	908
PowerPC PEF Import Hunks.	909
PowerPC Source File Specification Hunks	911
PowerPC Object Pascal Hunks	912
PowerPC Reserved Hunks.	914
PowerPC Symbolic Data Header	915
PowerPC Symbolic Function Data Section	916
PowerPC Symbolic Type Data	919
Other Types for PowerPC	920
PowerPC Pointer Type	921
PowerPC Array Type	922
PowerPC Structured Type	923
PowerPC Enumerated Type	924
PowerPC Pascal Array Type	925
PowerPC Pascal Subrange Type	926
PowerPC Pascal Set Type	927
PowerPC Pascal Enumerated Type	928
PowerPC Pascal String Type	929
PowerPC Name Table	929



Introduction

Welcome to the CodeWarrior™ IDE 5.1 SDK API Reference.

This book documents the various APIs that you can use to extend or replace the features of the CodeWarrior IDE.

This book consists of two sections:

- Section 1: Plug-in API
- Section 2: COM API



Freescal Semiconductor, Inc.

Introduction

Section 1: Plug-in API

This section includes the following chapters:

- Introduction to the Plug-in API
- Plug-in API Reference
- Compiler and Linker Plug-in Reference
- Browser Reference
- Settings Panel Plug-in API Reference
- Version Control System Plug-in API
- PowerPC Object Code (Mac OS)



Freescale Semiconductor, Inc.

Introduction to the Plug-in API

The plug-in API lets you extend the CodeWarrior IDE to include new features or to replace existing features. For example, you can write a plug-in to create a new preference panel. You can also write a plug-in that links the IDE to a different compiler or linker.

The IDE itself uses plug-ins to provide most of its services. For example, the IDE's standard compiler consists of a compiler plug-in with a small number of panel plug-ins to let users control its settings.



Freescale Semiconductor, Inc.

Introduction to the Plug-in API

Plug-in API Reference

This chapter describes the part of the plug-in SDK that all plug-ins can use. Later chapters describe the parts of the SDK that relate to specific kinds of plug-ins (such as compiler plug-ins).

This chapter covers the following topics:

- Routines for Plug-ins
- Plug-In Entry Points
- Data Structures for Plug-ins
- Constants for Plug-ins
- Result Codes for Plug-ins

Routines for Plug-ins

This section documents the SDK methods that all plug-ins can use.

Plug-in Context

All the routines provided by the CodeWarrior IDE require a value of type `CWPluginContext` to be passed as the first parameter. See “`CWPluginContext`” on page 108 and “Main Entry Point context Parameter” in the *IDE SDK Developer's Guide* for more information.

Alphabetical Routine Index

This section lists all common plug-in routines alphabetically.

- `CWAddProjectEntry`
- `CWAlert`
- `CWAllocateMemory`
- `CWAllocMemHandle`



Freescal Semiconductor, Inc.

Plug-in API Reference

Alphabetical Routine Index

- CWCreateNewTextDocument
- CWDonePluginRequest
- CWFindAndLoadFile
- CWFreeMemHandle
- CWFreeMemory
- CWGetAccessPathInfo
- CWGetAccessPathListInfo
- CWGetAccessPathSubdirectory
- CWGetAPIVersion
- CWGetCallbackOSError
- CWGetCOMApplicationInterface
- CWGetCOMProjectInterface
- CWGetCOMDesignInterface
- CWGetCOMTargetInterface
- CWGetFileInfo
- CWGetFileText
- CWGetFrameworkCount
- CWGetFrameworkInfo
- CWGetFrameworkSharedLibrary
- CWGetIDEInfo
- CWGetMemHandleSize
- CWGetNamedPreferences
- CWGetOutputFileDirectory
- CWGetOverlay1FileInfo
- CWGetOverlay1GroupsCount
- CWGetOverlay1GroupInfo
- CWGetOverlay1Info
- CWGetPluginData
- CWGetPluginRequest
- CWGetProjectFile
- CWGetProjectFileCount

- CWGetSegmentInfo
- CWGetTargetDataDirectory
- CWGetTargetName
- CWLockMemHandle
- CWMacOSErrToCWResult
- CWPostDialog
- CWPostFileAction
- CWPreFileAction
- CWPreDialog
- CWReleaseFileText
- CWRemoveProjectEntry
- CWReportMessage
- CWResizeMemHandle
- CWResolveRelativePath
- CWSetModDate
- CWSetPluginOSError
- CWShowStatus
- CWStorePluginData
- CWUnlockMemHandle
- CWUserBreak

Functional Routine Index

This section lists all common plug-in routines grouped by function.

Request Handling

- CWDonePluginRequest
- CWGetPluginRequest

Memory Management

- CWAllocateMemory
- CWAllocMemHandle
- CWFreeMemHandle



Freescale Semiconductor, Inc.

Plug-in API Reference

Functional Routine Index

- CFreeMemory
- CGetMemHandleSize
- CLockMemHandle
- CResizeMemHandle
- CUnlockMemHandle

Plug-in Data

- CGetPluginData
- CStorePluginData

Preference Data

- CGetNamedPreferences

File Management

- CFindAndLoadFile
- CGetFileInfo
- CGetFileText
- CPostFileAction
- CPreFileAction
- CReleaseFileText
- CSetModDate

Directory Information

- CGetAccessPathInfo
- CGetAccessPathListInfo
- CGetAccessPathSubdirectory
- CResolveRelativePath

Project File Information

- CAddProjectEntry
- CGetProjectFile
- CGetProjectFileCount

Target Information

- CWGetTargetDataDirectory
- CWGetTargetName
- CWGetOutputFileDirectory

Overlay Information

- CWGetOverlay1FileInfo
- CWGetOverlay1GroupInfo
- CWGetOverlay1GroupsCount
- CWGetOverlay1Info

Segment Information

- CWGetSegmentInfo

IDE Information

- CWGetAPIVersion
- CWGetIDEInfo

User Interaction

- CWAlert
- CWCreateNewTextDocument
- CWPostDialog
- CWPreDialog
- CWReportMessage
- CWShowStatus
- CWUserBreak

Error Handling

- CWGetCallbackOSError
- CWMacOSErrToCWResult
- CWSetPluginOSError

COM Object Interfaces

- CWGetCOMApplicationInterface



Plug-in API Reference
CWAddProjectEntry

- CWGetCOMProjectInterface
- CWGetCOMDesignInterface
- CWGetCOMTargetInterface

NOTE The COM routines are for use by COM plug-ins. COM routines do not work within compiler, linker, preference panel, or VCS plug-ins. See Section 2 of this book for more information.

CWAddProjectEntry

Description Adds a file to a project.

Prototype

```
#include <"DropinCompilerLinker.h">
CW_CALLBACK CWAddProjectEntry(
    CWPluginContext      context,
    const CWFileSpec*    fileSpec,
    Boolean               isGenerated,
    const CWNewProjectEntryInfo* projectEntryInfo,
    long*                 whichfile);
```

Parameters The parameters for this Method include:

context	Private, opaque IDE state.
filespec	Specifies the file to add to the project.
isGenerated	When true, specifies that the file has been created for the current request. For example, a compiler that generates a C source code file and adds it to the project would pass true in isGenerated.

projectEntryInfo	Specifies where to add the new file. Contains fields for flat, segmented, and overlayed linkers. The appropriate fields should be set for whichever model is used by the current linker. All position indexes are zero-based. If the pointer to this struct is NULL, the IDE will use default values for positioning the new file.
whichfile	Returns the index at which the file was added, in current target link order.

- Return See “Result Codes for Plug-ins” on page 129.
- Remarks If a plug-in calls CWAddProjectEntry during a make operation that is in progress, the IDE will restart the build operation.
- See Also “Adding a File to a Project” in the *IDE SDK Developer’s Guide*
- “CWFileSpec” on page 98
- “CWNewProjectEntryInfo” on page 103
- “CWSetModDate” on page 78
- “CWRemoveProjectEntry” on page 75

CWAlert

- Description Displays a message in the IDE’s message window.
- Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWAlert(
    CWPluginContext context,
    const char*      msg1,
    const char*      msg2,
    const char*      msg3,
    const char*      msg4);
```
- Parameters The parameters for this method include:

Plug-in API Reference

CWAAllocateMemory

context	Private, opaque IDE state.
msg1, msg2, msg3, msg4	Specifies the alert text, as C character strings. All four strings are concatenated in suffix order before display.

- Return See “Result Codes for Plug-ins” on page 129.
- Remarks To create a message in a modal dialog box, use CWReportMessage instead.
- See Also “CWReportMessage” on page 75

CWAAllocateMemory

Description Allocates a block of memory referred to by a pointer.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWAAllocateMemory(
    CWPluginContext context,
    long size,
    Boolean isPermanent,
    void** ptr);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
size	Specifies the size of the block to allocate, in bytes.
isPermanent	Specifies the life of the allocation. If isPermanent is true, the IDE preserves the memory between requests to the plug-in and even after the plug-in is unloaded from memory. Memory allocated with isPermanent set to true must be explicitly freed by the plug-in itself. If isPermanent is false, the IDE frees the memory when the plug-in finishes handling the current request.
ptr	Returns a pointer to the allocated memory. The pointer is valid only if CWAAllocateMemory returns cwNoErr.

- Return

See “Result Codes for Plug-ins” on page 129.
- Remarks

The pointer returned by this routine should be disposed of by calling `CWFreeMemory`.

The only reliable way to manage permanently allocated memory is to allocate it when the IDE makes a `reqTargetLoaded` call to the plug-in and to dispose of it during a `reqTargetUnloaded` call.
- See Also

“Managing Target Storage” in the *IDE SDK Developer’s Guide*

“`CWFreeMemory`” on page 48

“`reqTargetLoaded`” on page 219

“`reqTargetUnloaded`” on page 220

CWAllocMemHandle

- Description

Allocates a `CWMemHandle`, which refers to an indirectly referenced, resizable block of memory.
- Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWAllocMemHandle(
    CWPluginContext context,
    long size,
    Boolean useTempMemory,
    CWMemHandle* handle);
```
- Parameters

The parameters for this method include:

context	Private, opaque IDE state.
size	Specifies the size of the block to allocate, in bytes.
useTempMemory	Specifies whether the memory allocation should be from the temporary (process manager) heap or the application heap. Set this parameter to true to use temporary memory. This flag only affects Mac OS-hosted plug-ins.
handle	Returns the allocated memory in a <code>CWMemHandle</code> .

Plug-in API Reference
CWCreateNewTextDocument

Return	See “Result Codes for Plug-ins” on page 129.
Remarks	The handle returned by this routine should be disposed of by calling <code>CWFreeMemHandle</code>. Memory allocated by this routine can only be referenced indirectly, by first calling <code>CWLockMemHandle</code>. When access to a handle is no longer required, it should be unlocked using <code>CWUnlockMemHandle</code>. The IDE frees any memory allocated with <code>CWAllocMemHandle</code> when the plug-in finishes servicing a request. To allocate memory that persists between calls made by the IDE, use <code>CWAllocateMemory</code>.
See Also	“CWAllocateMemory” on page 42 “CWFreeMemHandle” on page 47 “CWGetMemHandleSize” on page 60 “CWLockMemHandle” on page 70 “CWMemHandle” on page 102 “CWResizeMemHandle” on page 77 “CWUnlockMemHandle” on page 82

CWCreateNewTextDocument

Description	Opens a new text editor window containing specified text.				
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWCreateNewTextDocument (CWPluginContext context, const CWNewTextDocumentInfo* docinfo);</pre>				
Parameters	The parameters for this method include: <table border="1"> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>docinfo</td><td>Specifies the name and contents of the new editor window.</td></tr> </table>	context	Private, opaque IDE state.	docinfo	Specifies the name and contents of the new editor window.
context	Private, opaque IDE state.				
docinfo	Specifies the name and contents of the new editor window.				
Return	See “Result Codes for Plug-ins” on page 129.				

Remarks Use CWCreateNewTextDocument to open a text editor window in which to present the textual result of an operation. A compiler or linker typically uses this routine to show disassembly and preprocessor text.

See Also "CWNewTextDocumentInfo" on page 105

CWDonePluginRequest

Description Tells the IDE that a plug-in has finished handling a request.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWDonePluginRequest(
    CWPluginContext context,
    CWResult resultCode);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
resultCode	A value indicating whether the plug-in was able to handle a request successfully. Use cwNoErr if the plug-in handled the request successfully. Use cwErrRequestFailed if the plug-in was not able to complete the request successfully.

Returns See "Result Codes for Plug-ins" on page 129.

Remarks When a plug-in has finished servicing a request, a plug-in must call CWDonePluginRequest to indicate success or failure to the IDE. In addition, the plug-in should pass the result code returned from CWDonePluginRequest back to the IDE when returning from its main() function.

See Also "cwNoErr" on page 133
"cwErrRequestFailed" on page 131

CWFindAndLoadFile

Description Locates and optionally loads the contents of a file and specifies a file dependency.

Prototype

```
#include "DropinCompilerLinker.h"
```



Freescale Semiconductor, Inc.

Plug-in API Reference
CWFindAndLoadFile

```
CW_CALLBACK CWFindAndLoadFile(  
    CWPluginContext context,  
    const char* filename,  
    CWFileInfo* fileinfo);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
filename	Specifies the name of the file to search for and load. The IDE searches within the project and the access paths specified by the project. If more than one file has the same name, the IDE loads the first file found.
fileinfo	Specifies the file to load and any dependency information. On return, contains information about the file loaded.

Return See “Result Codes for Plug-ins” on page 129.

Remarks Use CWFindAndLoadFile to search for a file in one of the access paths listed in the **Access Paths** panel of the current target. Use this method to load header and interface files that are not necessarily in the project. If CWFindAndLoadFile successfully finds the file and optionally loads it into memory, it returns cwNoErr.

The fileinfo object has a suppressload parameter, which is a boolean value that controls whether the IDE tries to load the file into memory. If suppressload is true, the IDE does not load the file into memory, regardless of its type. If suppressload is false, CWFindAndLoadFile attempts to load the requested file into memory. CWFindAndLoadFile loads a file into memory only if it is a text file or a cached and precompiled header file.

When CWFindAndLoadFile finds a file and loads it into memory, it sets the following fields in fileinfo:

filespec	The specification for the file
filedata	A pointer to the file’s contents in memory. NULL if the file is not loaded into memory.

filedatalength	The size of the file's contents, in bytes. 0 if the file is not loaded into memory.
filedatatype	cwFileTypeText if the file contains text, cwFileTypePrecompiledHeader if the file is a cached and precompiled header, or cwFileTypeUnknown for any other kind of file.

NOTE Every call to `CWFindAndLoadFile` that returns (non-NULL) `filedata` must be balanced by a single call to `CWReleaseFileText`.

The `isdependentoffile` and `dependencyType` fields of the `CWFileInfo` structure specify a source file dependency. If `dependencyType` is set to `cwNormalDependency` or `cwInterfaceDependency`, the IDE recompiles the file specified in the `isdependentoffile` parameter when the interface or content of the file specified by `filespec` changes. If `dependencyType` is set to `cwNoDependency`, the IDE assumes no dependency exists.

`isdependentoffile` specifies which file depends upon the file being searched for, by index. You can set the value of `isdependentoffile` to `kCurrentCompiledFile` to specify that the file currently being compiled is the dependent file.

- See Also
- "CWFileInfo" on page 95
 - "CWDependencyType" on page 93
 - "CWGetFileText" on page 56
 - "CWCachePrecompiledHeader" on page 138
 - "CWReleaseFileText" on page 74

CWFreeMemHandle

Description Releases handles allocated with `CWAllocMemHandle`.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWFreeMemHandle(
```

Plug-in API Reference

CWFreeMemory

	<pre> CWPluginContext context, CWMemHandle handle); </pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>handle</td><td>The CWMemHandle to deallocate.</td></tr> </table>	context	Private, opaque IDE state.	handle	The CWMemHandle to deallocate.
context	Private, opaque IDE state.				
handle	The CWMemHandle to deallocate.				
Return	See “Result Codes for Plug-ins” on page 129.				
Remarks	To free a CWMemHandle allocated with CWAllocMemHandle, use CWFreeMemHandle. A plug-in should free a handle only once. Do not use CWFreeMemHandle to free blocks that are not handles.				
NOTE	All handles are automatically freed when a plug-in returns to the IDE. However, you can increase performance by freeing large memory blocks as soon as you can.				
See Also	“CWAllocMemHandle” on page 43 “CWGetMemHandleSize” on page 60 “CWLLockMemHandle” on page 70 “CWMemHandle” on page 102 “CWResizeMemHandle” on page 77 “CWUnlockMemHandle” on page 82				

CWFreeMemory

Description	Releases a pointer created by CWAllocateMemory.
Prototype	<pre> #include "DropinCompilerLinker.h" CW_CALLBACK CWFreeMemory(CWPluginContext context, void* ptr, Boolean isPermanent); </pre>
Parameters	The parameters for this method include:

context	Private, opaque IDE state.
ptr	The pointer block to deallocate.
isPermanent	The value passed must match the value used when allocating the pointer with CWAllocateMemory.

Return See "Result Codes for Plug-ins" on page 129.

Remarks A plug-in should free a pointer only once. Do not use CWFFreeMemory to free objects that are not pointers.

NOTE A plug-in should balance every call to CWAllocateMemory with a single corresponding call to CWFFreeMemory for the same pointer.

See Also "Allocating and Releasing Handles" in the *IDE SDK Developer's Guide*

"CWAllocateMemory" on page 42

CWGetAccessPathInfo

Description Returns information about a single access path specified for the current project target.

Prototype

```
#include "CWPlugins.h"
CW_CALLBACK CWGetAccessPathInfo(
    CWPluginContext context,
    CWAccessPathType pathType,
    long whichPath,
    CWAccessPathInfo* pathInfo);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
pathType	Set to one of cwSystemPath or cwUserPath. Specifies whether to return a user path or system path.
whichPath	A zero-based index for the path to be returned.
pathInfo	Returns information about the specified path.

Plug-in API Reference

CWGetAccessPathListInfo

- Returns** See “Result Codes for Plug-ins” on page 129.
- Remarks** Plug-ins use `CWGetAccessPathInfo` to obtain information about the access paths specified in the **Access Paths** panel. Plug-ins can use `CWGetAccessPathListInfo` to get the number of user or system paths in the project. By using an index that ranges from 0 to `systemPathCount - 1` or `userPathCount - 1`, plug-ins can iterate all system or user paths in the current project.

The information returned in `pathinfo` includes the `CWFileSpec` for the access path, as well as a count of subdirectories and a flag indicating whether the IDE searches subdirectories of the path for project files or just the access path itself. If subdirectory searching is disabled, `CWGetAccessPathInfo` returns 0 for the subdirectory count, regardless of how many subdirectories actually exist in the access path.

To obtain information about the subdirectories of project access paths, use `CWGetAccessPathSubdirectory`.

- See Also** “`CWFileSpec`” on page 98
- “`CWGetAccessPathListInfo`” on page 50
- “`CWGetAccessPathSubdirectory`” on page 51
- “`CWAccessPathInfo`” on page 89

CWGetAccessPathListInfo

- Description** Returns information about the access paths specified for the current project.
- Prototype**

```
#include "CWPlugins.h"
CW_CALLBACK CWGetAccessPathListInfo(
    CWPluginContext context,
    CWAccessPathListInfo* pathListInfo);
```
- Parameters** The parameters for this method include:

context	Private, opaque IDE state.
pathListInfo	Returns information about the access paths specified for the current project target, including the number of user and system paths.

- Returns See “Result Codes for Plug-ins” on page 129.
- Remarks Plug-ins can determine the number of user and system paths in the current project target by calling CWGetAccessPathListInfo.
- See Also “CWAcessPathListInfo” on page 90

CWGetAccessPathSubdirectory

- Description Returns information about a subdirectory of a project path.
- Prototype


```
#include "CWPlugins.h"
CW_CALLBACK CWGetAccessPathSubdirectory(
    CWPluginContext context,
    CWAcessPathType pathType,
    long whichPath,
    long whichSubdirectory,
    CWFileSpec* subdirectory);
```
- Parameters The parameters for this method include:

context	Private, opaque IDE state.
pathType	Set to one of cwSystemPath or cwUserPath. Specifies whether to return information about a subdirectory within a user path or a system path.
whichPath	A zero-based index specifying a path of type pathType in the current target.
whichSubdirectory	A zero-based index specifying a subdirectory of whichPath.
subdirectory	Returns a file specification object for the requested access path subdirectory.

- Returns See “Result Codes for Plug-ins” on page 129.



Plug-in API Reference
CWGetAPIVersion

Remarks Use CWGetAccessPathSubdirectory to obtain the subdirectories of any project path for which recursive project path searching is enabled. Plug-ins can use this method to create a list of search paths to pass to command line tools.

NOTE CWGetAccessPathSubdirectory does not return subdirectories for access paths for which recursive path searching is disabled, regardless of how many actual subdirectories may exist. See the *IDE User's Guide* for how to change the recursive search setting.

To determine the range of valid indexes for whichPath, use CWGetAccessPathListInfo. To determine the range of indexes for whichSubdirectory, use CWGetAccessPathInfo.

See Also "CWGetAccessPathInfo" on page 49
"CWGetAccessPathListInfo" on page 50

CWGetAPIVersion

Description Returns the version of the plug-in API that the IDE uses interact with the plug-in.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetAPIVersion(
    CWPluginContext context,
    long* version);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
version	Returns the version number of the API the IDE uses interact with the plug-in.

Returns See "Result Codes for Plug-ins" on page 129.

Remarks Plug-ins can use CWGetAPIVersion at runtime to get the version of the plug-in API that the IDE uses to communicate with the plug-in.

NOTE The IDE does not always use the latest version of the API. The IDE uses an older version when the plug-in requires an earlier API



version. The plug-in tells the IDE that the plug-in needs an earlier version of the API by setting the value of the `newestAPIVersion` parameter in the `CWPlugin_GetDropInFlags` entry point.

See Also
 "CWGetIDEInfo" on page 59
 "DROPINCOMPILERLINKERAPIVERSION" on page 196

CWGetCallbackOSError

Description
 Returns the error the host operating system returned to the IDE the last time the plug-in called the IDE.

Prototype

```

#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetCallbackOSError(
    CWPluginContext context,
    CWOSResult* error);
            
```

Parameters
 The parameters for this method include:

context	Private, opaque IDE state.
error	Returns the last error code reported by the host operating system during a call to the IDE by the plug-in.

Return
 cwNoErr

Remarks
 Use this routine to get the error that the host operating system returned to the IDE when the IDE was last called by the plug-in. This lets a plug-in discover the nature of the error when the IDE reports a result of `cwErrOSError`.

See Also
 "CWMacOSErrToCWResult" on page 71

CWGetCOMApplicationInterface

Description
 Returns a reference to a COM interface for the IDE as an application object.

Prototype

```

#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetCOMApplicationInterface(
    CWPluginContext context,
    struct ICodeWarriorApp **app);
            
```

Plug-in API Reference

CWGetCOMProjectInterface

Parameters The parameters for this method include:

context	Private, opaque IDE state.
app	Returns a reference to the CodeWarrior IDE application's COM interface, which can be used to obtain other properties and services of the IDE.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Compilers, linkers, settings panels, or version control plug-ins cannot use this method.

CWGetCOMProjectInterface

Description Returns a reference to a COM interface for the current project.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetCOMProjectInterface(
    CWPluginContext context,
    struct ICodeWarriorProject **project);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
project	Returns a reference to the current project's COM interface, which can be used to obtain properties and services of the current project.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Compilers, linkers, settings panels, or version control plug-ins cannot use this method.

CWGetCOMDesignInterface

Description Returns a reference to a COM interface for the design associated with the current target.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetCOMDesignInterface(
    CWPluginContext context,
    struct ICodeWarriorDesign **design);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
design	Returns a reference to the current target design's COM interface, which can be used to manipulate user interface controls.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Compilers, linkers, settings panels, or version control plug-ins cannot use this method.

CWGetCOMTargetInterface

Description Returns a reference to a COM interface for the current project target.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetCOMTargetInterface(
    CWPluginContext context,
    struct ICodeWarriorTarget **target);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
target	Returns a reference to the current target's COM interface, which can be used to obtain properties and services of the current target.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Compilers, linkers, settings panels, or version control plug-ins cannot use this method.

CWGetFileInfo

Description Returns information about a file in the active project.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetFileInfo(
    CWPluginContext context,
    long whichfile,
    Boolean checkFileLocation,
    CWProjectFileInfo* fileinfo);
```

Plug-in API Reference

CWGetFileText

Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>whichfile</td><td>The number of the file in the active target for which to get information.</td></tr> <tr> <td>checkFileLocation</td><td>true to request that the IDE verify the location of the file returned in fileinfo. false to request the most recently cached location for the file.</td></tr> <tr> <td>fileinfo</td><td>Returns information about the file.</td></tr> </table>	context	Private, opaque IDE state.	whichfile	The number of the file in the active target for which to get information.	checkFileLocation	true to request that the IDE verify the location of the file returned in fileinfo. false to request the most recently cached location for the file.	fileinfo	Returns information about the file.
context	Private, opaque IDE state.								
whichfile	The number of the file in the active target for which to get information.								
checkFileLocation	true to request that the IDE verify the location of the file returned in fileinfo. false to request the most recently cached location for the file.								
fileinfo	Returns information about the file.								
Returns	See “Result Codes for Plug-ins” on page 129.								
Remarks	Use CWGetFileInfo to get information about a file in the active project, such as the file’s file specification and modification date. This method also returns a value indicating how the IDE processes the file during build operations. The whichfile argument specifies the index (starting at 0) of the file for which to return information, in the link order of the active project target. checkFileLocation is typically used when iterating over all the files in a project multiple times. Setting this flag to true tells the IDE to use cached locations for files, rather than verifying a file’s location with the host operating system. In some cases, this may improve performance. Usually, plug-ins set this parameter to false, which tells the IDE to verify a file’s true location on disk each time CWGetFileInfo is called.								
NOTE	The definition of the fileinfo structure returned by CWGetFileInfo varies by platform.								
See Also	“CWProjectFileInfo” on page 109 “CWGetProjectFileCount” on page 67								
	CWGetFileText								
Description	Loads the contents of a file.								

```

Prototype  #include "DropinCompilerLinker.h"
           CW_CALLBACK CWGetFileText(
               CWPluginContext context,
               const CWFileSpec* filespec,
               const char**      text,
               long*              textLength,
               short*             filedatatype);
    
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
filespec	The file specification of the file to load.
text	Returns a pointer to the contents of the file.
textLength	Returns the length of the file’s contents in bytes.
filedatatype	Returns the file’s type. The possible values for this argument include <code>cwFileTypePrecompiledHeader</code> , <code>cwFileTypeText</code> , and <code>cwFileTypeUnknown</code> .

Return See “Result Codes for Plug-ins” on page 129.

Remarks Use `CWGetFileText` to retrieve the contents of a file. If the file is a text file that is opened in an editor window, the IDE returns a pointer in `text` to the contents of the open editor window. If the file is on disk and is not open in an editor window, the IDE loads the contents of the file and returns a pointer to that data in `text`.

NOTE Every call to `CWGetFileText` that returns (non-NULL) `text` must be balanced by a single call to `CWReleaseFileText`.

`CWGetFileText` returns the text file as a constant. Plug-ins should not modify the text data returned in the `text` parameter.

See Also “`CWFindAndLoadFile`” on page 45
“`CWReleaseFileText`” on page 74

Plug-in API Reference

CWGetFrameworkCount

CWGetFrameworkCount

Description Gets the number of frameworks.

Prototype `CW_CALLBACK CWGetFrameworkCount (
 CWPluginContext context,
 long* frameworkCount);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
frameworkCount	Returns the number of frameworks.

Return See "Result Codes for Plug-ins" on page 129.

See Also "CWGetFrameworkInfo" on page 58

"CWGetFrameworkSharedLibrary" on page 59

CWGetFrameworkInfo

Description Gets information about a framework, as a CWFrameworkInfo structure.

Prototype `CW_CALLBACK CWGetFrameworkInfo (
 CWPluginContext context,
 long whichFramework,
 CWFrameworkInfo* frameworkInfo);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichFramework	The framework about which to get information.
frameworkInfo	Returns the framework information.

Return See "Result Codes for Plug-ins" on page 129.

See Also "CWGetFrameworkCount" on page 58

"CWGetFrameworkSharedLibrary" on page 59

"CWFrameworkInfo" on page 100

CWGetFrameworkSharedLibrary

Description Gets the shared library for a framework.

Prototype `CW_CALLBACK CWGetFrameworkSharedLibrary(
CWPluginContext context,
long whichFramework,
CWFileSpec* frameworkSharedLibrary);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichFramework	The framework about which to get information.
frameworkSharedLibrary	Returns the shared library used by the framework.

Return See “Result Codes for Plug-ins” on page 129.

See Also “CWGetFrameworkCount” on page 58
“CWGetFrameworkInfo” on page 58

CWGetIDEInfo

Description This returns version information about the IDE and which version of the dropin API it implements.

Prototype `#include "CWPlugins.h"
CW_CALLBACK CWGetIDEInfo(
CWPluginContext context,
CWIDEInfo* info);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
info	Returns information about the IDE, including major, minor, bug fix, and build version numbers, as well as the highest API version supported by the IDE.

Return See “Result Codes for Plug-ins” on page 129.

Plug-in API Reference

CWGetMemHandleSize

Remarks	Note that the version of the dropin API returned by this call may be different from the one returned by <code>CWGetAPIVersion</code> . <code>CWGetAPIVersion</code> returns the version of the API that the IDE is providing to a particular plug-in. This can be less than the IDE's latest supported dropin API version (reported by this call) if a plug-in is written to use an older API.
See Also	“ <code>CWGetAPIVersion</code> ” on page 52 “ <code>CWIDEInfo</code> ” on page 101

CWGetMemHandleSize

Description	Returns the size of a <code>CWMemHandle</code> .						
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWGetMemHandleSize(CWPluginContext context, CWMemHandle handle, long* size);</pre>						
Parameters	The parameters for this method include: <table> <tr> <td><code>context</code></td><td>Private, opaque IDE state.</td></tr> <tr> <td><code>handle</code></td><td>The handle whose size is to be determined.</td></tr> <tr> <td><code>size</code></td><td>Returns the size of the handle, in bytes.</td></tr> </table>	<code>context</code>	Private, opaque IDE state.	<code>handle</code>	The handle whose size is to be determined.	<code>size</code>	Returns the size of the handle, in bytes.
<code>context</code>	Private, opaque IDE state.						
<code>handle</code>	The handle whose size is to be determined.						
<code>size</code>	Returns the size of the handle, in bytes.						
Return	See “Result Codes for Plug-ins” on page 129.						
Remarks	To get the size of a block of memory allocated with <code>CWAllocMemHandle</code> or resized by <code>CWResizeMemHandle</code> , use <code>CWGetMemHandleSize</code> .						
See Also	“ <code>CWAllocMemHandle</code> ” on page 43 “ <code>CWFreeMemHandle</code> ” on page 47 “ <code>CWResizeMemHandle</code> ” on page 77 “ <code>CWLockMemHandle</code> ” on page 70 “ <code>CWUnlockMemHandle</code> ” on page 82						

CWGetNamedPreferences

Description Returns the settings data of a settings panel.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetNamedPreferences (
    CWPluginContext context,
    const char*      prefsname,
    CWMemHandle*     prefsdata);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
prefsname	The name of the settings panel whose preferences are to be loaded, specified as a C character string.
prefsdata	Returns the preference data in a handle.

Return See “Result Codes for Plug-ins” on page 129.

Remarks The IDE returns a CWMemHandle in the prefsdata parameter. This handle refers to the settings data for the settings panel specified in prefsname parameter.

The IDE cannot see the contents and format of the preference data. Only the plug-in and any associated settings panel plug-ins manage and interpret the contents of the settings data.

The returned handle becomes the plug-in’s property. Prior to accessing the handle’s contents, the plug-in should lock the handle with CWLockMemHandle. When the handle is no longer needed, the plug-in should free it with CWFreeMemHandle.

See Also “CWMemHandle” on page 102
 “CWFreeMemHandle” on page 47
 “CWLockMemHandle” on page 70

CWGetOutputFileDirectory

Description Returns the folder specified in the **Output Directory** field of the **Target Settings** panel.

Plug-in API Reference

CWGetOverlay1FileInfo

Prototype `#include "DropinCompilerLinker.h"`
 `CW_CALLBACK CWGetOutputFileDirectory(`
 `CWPluginContext    context,`
 `CWFileSpec*        outputFileDirectory);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
outputFileDirectory	Returns the file specification of the output directory.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Use CWGetOutputFileDirectory to determine the location where linker output should be stored, as specified by the user in the **Target Settings** panel.

See Also "CWGetProjectFile" on page 67

CWGetOverlay1FileInfo

Description Returns information about a file in an overlay.

Prototype `#include "DropinCompilerLinker.h"`
 `CW_CALLBACK CWGetOverlay1FileInfo(`
 `CWPluginContext       context,`
 `long                    whichgroup,`
 `long                    whichoverlay,`
 `long                    whichoverlayfile,`
 `CWOverlay1FileInfo* fileinfo);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichgroup	Specifies the overlay group number.
whichoverlay	Specifies the overlay number.
whichoverlayfile	Specifies the overlay file number.
fileinfo	Returns information about the file in the overlay.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Use CWGetOverlay1FileInfo to get an overlay file's position within a project. Each of the whichgroup, whichoverlay, and whichoverlayfile indexes is zero based and specifies the overlay component for which to return information. The fileinfo result specifies the position in the file list of the current target file.

NOTE Overlays are only used for certain embedded targets.

See Also "Getting Information About Overlays and Segments" in the *IDE SDK Developer's Guide*
"Specifying Project Components" in the *IDE SDK Developer's Guide*
"Project Structure" in the *IDE SDK Developer's Guide*
"CWGetOverlay1Info" on page 64

CWGetOverlay1GroupInfo

Description Returns information about an overlay group in the project.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetOverlay1GroupInfo(
    CWPluginContext context,
    long whichgroup,
    CWOOverlay1GroupInfo* groupinfo);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichgroup	The overlay group to get information for.
groupinfo	Returns information about the overlay group.

Return See "Result Codes for Plug-ins" on page 129.

Remarks Use CWGetOverlay1GroupInfo to get the name, the address, and the number of overlays in an overlay group.

Overlay groups are specified by index. The index of the first overlay group is 0. The index of the last is CWGetOverlay1GroupsCount - 1.

See Also "CWOOverlay1GroupInfo" on page 107

CWGetOverlay1GroupsCount

Description	Returns the number of overlay groups.				
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWGetOverlay1GroupsCount (CWPluginContext context, long* count);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>count</td><td>Returns the number of overlay groups in the project.</td></tr> </table>	context	Private, opaque IDE state.	count	Returns the number of overlay groups in the project.
context	Private, opaque IDE state.				
count	Returns the number of overlay groups in the project.				
Return	See “Result Codes for Plug-ins” on page 129.				
Remarks	Use CWGetOverlay1GroupsCount to get the number of overlay groups in the active project.				
See Also	“CWGetOverlay1GroupInfo” on page 63				

CWGetOverlay1Info

Description	Returns information about an overlay in a group.								
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWGetOverlay1Info (CWPluginContext context, long whichgroup, long whichoverlay, CWOOverlay1Info* overlayinfo);</pre>								
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>whichgroup</td><td>The overlay group number.</td></tr> <tr> <td>whichoverlay</td><td>The overlay number.</td></tr> <tr> <td>overlayinfo</td><td>Returns information about the overlay.</td></tr> </table>	context	Private, opaque IDE state.	whichgroup	The overlay group number.	whichoverlay	The overlay number.	overlayinfo	Returns information about the overlay.
context	Private, opaque IDE state.								
whichgroup	The overlay group number.								
whichoverlay	The overlay number.								
overlayinfo	Returns information about the overlay.								
Return	See “Result Codes for Plug-ins” on page 129.								

Remarks Use CWGetOverlay1Info to get the name and number of files in an overlay.

The IDE indexes overlays, from 0 to numoverlays - 1, where numoverlays is the value returned by CWGetOverlay1GroupInfo.

See Also “CWGetOverlay1FileInfo” on page 62

“CWGetOverlay1GroupInfo” on page 63

CWGetPluginData

Description Returns the plug-in data associated with a file in the active project.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetPluginData(
    CWPluginContext context,
    long whichfile,
    CWDataType type,
    CWMemHandle* pluginData);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the file to retrieve data for. To obtain global data (that is, data for the target, not a specific target file), specify kTargetGlobalPluginData.
type	Specifies which kind of data to retrieve.
pluginData	Returns a CWMemHandle containing the data associated with the file. Returns NULL if no data of type type exists.

Return See “Result Codes for Plug-ins” on page 129.

Remarks CWGetPluginData retrieves data in the project for a file. The whichfile parameter specifies the file name. The type parameter specifies the type of the file. The plug-in must maintain the format of this data, because the IDE never uses this data.

The IDE stores plug-in data until one of the following events occurs:

- The user chooses **Remove Object Code** from the **Project** menu.



Plug-in API Reference
CWGetPluginRequest

- The file is removed from the target.
- The user manually deletes the target data folder.

The returned handle becomes the plug-in's property. Prior to accessing the handle's contents, the plug-in should lock the handle with `CWLockMemHandle`. When the handle is no longer needed, the plug-in should free it with `CWFreeMemHandle`.

See Also "Storing and Retrieving Plug-in Data" in the *IDE SDK Developer's Guide*

"CWFreeMemHandle" on page 47

"CWGetTargetStorage" on page 154

"CWLockMemHandle" on page 70

"CWSetTargetStorage" on page 164

"CWStorePluginData" on page 81

CWGetPluginRequest

Description Returns a value indicating the task the IDE is currently asking the plug-in to perform.

Prototype `#include "DropinCompilerLinker.h"`
`CW_CALLBACK CWGetPluginRequest (`
 `CWPluginContext context,`
 `long* request);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
request	Returns a request code indicating the service the IDE is requesting from the plug-in.

Returns See "Result Codes for Plug-ins" on page 129.

Remarks When called by the IDE, a plug-in should call `CWGetPluginRequest` to determine what the IDE is asking the plug-in to do.

See Also "Basic Request Handling" in the *IDE SDK Developer's Guide*

“Handling Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

“Settings Panel Requests” in the *IDE SDK Developer’s Guide*

“reqInitialize” on page 127

“reqTerminate” on page 128

CWGetProjectFile

Description	Returns the file specification of the project file.				
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWGetProjectFile(CWPluginContext context, CWFileSpec* projectSpec);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>projectSpec</td><td>Returns the file specification for the project.</td></tr> </table>	context	Private, opaque IDE state.	projectSpec	Returns the file specification for the project.
context	Private, opaque IDE state.				
projectSpec	Returns the file specification for the project.				
Return	See “Result Codes for Plug-ins” on page 129.				
Remarks	Use CWGetProjectFile to get a file specification for the active project. Plug-ins can use this method to get information such as the volume or folder where the project is stored.				
See Also	“CWGetOutputFileDirectory” on page 61				

CWGetProjectFileCount

Description	Returns the number of files in the current project.				
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWGetProjectFileCount(CWPluginContext context, long* count);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>count</td><td>Returns the number of files in the active project.</td></tr> </table>	context	Private, opaque IDE state.	count	Returns the number of files in the active project.
context	Private, opaque IDE state.				
count	Returns the number of files in the active project.				

Plug-in API Reference

CWGetSegmentInfo

- Returns See "Result Codes for Plug-ins" on page 129.
- Remarks Use CWGetProjectFileCount to get the number of files in the active project target.
- See Also "CWGetFileInfo" on page 55

CWGetSegmentInfo

- Description Returns information about a segment in the project.
- Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetSegmentInfo(
    CWPluginContext context,
    long whichsegment,
    CWProjectSegmentInfo* segmentinfo);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichsegment	Specifies a segment by index. Indexes are zero based, and match the order specified in the Segments tab of a project.
segmentinfo	Returns information about the specified segment.

- Return See "Result Codes for Plug-ins" on page 129.
- Remarks Use CWGetSegmentInfo to get information about a segment in the active project. The returned information specifies the segment name and the segment's resource attributes.

NOTE Only Mac OS 68K targets use segments.

See Also "CWProjectSegmentInfo" on page 113

CWGetTargetDataDirectory

- Description Returns the directory where files in the current target are stored.
- Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWGetTargetDataDirectory(
    CWPluginContext context,
```

CWFileSpec* targetDataDirectorySpec);

Parameters The parameters for this method include:

context	Private, opaque IDE state.
targetDataDirectorySpec	Returns the directory where data for the current target should be stored.

Return See “Result Codes for Plug-ins” on page 129.

Remarks CWGetTargetDataDirectory returns the target data directory for the current target. If the plug-in needs to store temporary data files with the target, the plug-in should put the files in this folder.

Plug-ins that implement interfaces to command-line utilities generally use this method to store settings and data for the utility. Most plug-ins should use CWGetTargetStorage and CWSetTargetStorage to store per-target data.

NOTE The IDE deletes the files in this directory when the user chooses **Remove Object Code from the Project** menu.

See Also “CWGetProjectFile” on page 67

CWGetTargetName

Description Returns the name of the active target in the active project.

Prototype #include "DropinCompilerLinker.h"
CW_CALLBACK CWGetTargetName(
 CWPluginContext context,
 char* name,
 short maxLength);

Parameters The parameters for this method include:

context	Private, opaque IDE state.
name	Returns the target’s name as a C string.
maxLength	Specifies the space available in the name parameter for storing the target name, in bytes.

Return See “Result Codes for Plug-ins” on page 129.

Plug-in API Reference

CWLockMemHandle

- Remarks Use CWGetTargetName to get the name of the active target in the current project.
- See Also “CWGetProjectFile” on page 67

CWLockMemHandle

Description Gets a pointer to a memory handle that was allocated with CWMemHandle.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWLockMemHandle(
    CWPluginContext context,
    CWMemHandle handle,
    Boolean moveHi,
    void** ptr);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
handle	The handle for which to get a pointer.
moveHi	Specifies whether to move the handle to the top of the heap before locking it. If moveHi is true, the allocated memory is moved to the top of the heap, potentially reducing heap fragmentation. This flag works only on the Mac OS.
ptr	Returns a pointer to the handle’s memory. The pointer is valid only as long as the handle remains locked.

Return See “Result Codes for Plug-ins” on page 129.

Remarks To access the memory described by a CWMemHandle allocated with CWAllocMemHandle or resized with CWResizeMemHandle, use CWLockMemHandle.

When finished accessing a handle block, a plug-in should call CWUnlockMemHandle. On some platforms, locked handles may adversely affect memory management. Unlock handles as soon as possible.

Calls to CWLockMemHandle and CWUnlockMemHandle are counted. A handle remains locked as long as its lock count is greater

than zero. When initially allocated, handles are unlocked. CWLockMemHandle increases a handle’s lock count, and CWUnlockMemHandle decreases its lock count.

- See Also
- “CWAllocMemHandle” on page 43
 - “CWFreeMemHandle” on page 47
 - “CWGetMemHandleSize” on page 60
 - “CWResizeMemHandle” on page 77
 - “CWUnlockMemHandle” on page 82

CWMacOSErrToCWResult

Description Converts a Mac OS error code to a CWResult error code.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWMacOSErrToCWResult(
    CWPluginContext context,
    OSErr err);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
err	Specifies the Mac OS error code to convert to an IDE error code.

Return See “Result Codes for Plug-ins” on page 129.

Remarks This method converts a Mac OS error code to an IDE error code. This routine always succeeds. The result contains the IDE error code equivalent to the Mac OS error code passed in the err parameter.

NOTE Because this method can return error codes that do not fit the situation, you should write your own code to convert OS errors to IDE errors.

See Also “Result Codes for Plug-ins” on page 129

Plug-in API Reference

CWPostDialog

CWPostDialog

Description	Informs the IDE that a plug-in has finished displaying a dialog box.		
Prototype	<pre> #include "CWPlugins.h" CW_CALLBACK CWPostDialog(CWPluginContext context); </pre>		
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> </table>	context	Private, opaque IDE state.
context	Private, opaque IDE state.		
Returns	See “Result Codes for Plug-ins” on page 129.		
Remarks	Call CWPostDialog after displaying your own dialog box. This routine lets the IDE’s internal window manager operate properly with your dialogs. Call CWPreDialog before displaying the dialog.		
See Also	“Displaying Plug-in Dialog Boxes” in the <i>IDE SDK Developer’s Guide</i> “CWPreDialog” on page 73		

CWPostFileAction

Description	Tells the IDE that a plug-in has relinquished access to a file, allowing the IDE to reopen and reload the file, if necessary.				
Prototype	<pre> #include "CWPlugins.h" CW_CALLBACK CWPostFileAction (CWPluginContext context, const CWFileSpec *theFile); </pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>theFile</td><td>Specifies the file the plug-in has finished accessing.</td></tr> </table>	context	Private, opaque IDE state.	theFile	Specifies the file the plug-in has finished accessing.
context	Private, opaque IDE state.				
theFile	Specifies the file the plug-in has finished accessing.				
Returns	See “Result Codes for Plug-ins” on page 129.				
Remarks	A plug-in should call CWPostFileAction after opening, reading, writing, or otherwise modifying any file in the current project for which CWPreFileAction was previously called. This routine tells the IDE that the plug-in has relinquished control over the file and that the IDE can reopen the file if the user has the file open in the IDE. A plug-in should call CWPreFileAction before accessing the				

file. Plug-ins need to use these methods only when accessing files directly, rather than through the plug-in API.

NOTE Only version control plug-ins can use this method.

See Also “CWPReFileAction” on page 73

CWPReDialog

Description Informs the IDE that a plug-in is about to display a dialog box.

Prototype

```
#include "CWPlugins.h"
CW_CALLBACK CWPReDialog (CWPluginContext context);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
---------	----------------------------

Returns See “Result Codes for Plug-ins” on page 129.

Remarks Call CWPReDialog prior to display your own dialog in a plug-in. This routine lets the IDE’s internal window manager operate properly with your dialogs. Be sure to call CWPPostDialog after displaying the dialog box.

See Also “Displaying Plug-in Dialog Boxes” in the *IDE SDK Developer’s Guide*
“CWPPostDialog” on page 72

CWPReFileAction

Description Informs the IDE that a plug-in needs access to a file that the IDE may have open.

Prototype

```
#include "CWPlugins.h"
CW_CALLBACK CWPReFileAction
(CWPluginContext context, const CWFileSpec
*theFile);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
theFile	Specifies the file the plug-in would like to access.

Plug-in API Reference *CWReleaseFileText*

Returns	See “Result Codes for Plug-ins” on page 129.
Remarks	A plug-in should call <code>CWPreFileAction</code> when it needs to read or write to the file specified by <code>theFile</code> . This tells the IDE that the plug-in needs access to the specified file, and that it should be closed. A plug-in should also call <code>CWPostFileAction</code> after accessing the file.
NOTE	Currently, this routine is only available to version control plug-ins. This may change in the future.

See Also “CWPostFileAction” on page 72

CWReleaseFileText

Description	Releases the contents of a file retrieved with <code>CWGetFileText</code> or <code>CWFindAndLoadFile</code> .				
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWReleaseFileText(CWPluginContext context, const char* text);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>text</td><td>A pointer to the contents of a file previously loaded by <code>CWGetFileText</code> or <code>CWFindAndLoadFile</code>.</td></tr> </table>	context	Private, opaque IDE state.	text	A pointer to the contents of a file previously loaded by <code>CWGetFileText</code> or <code>CWFindAndLoadFile</code> .
context	Private, opaque IDE state.				
text	A pointer to the contents of a file previously loaded by <code>CWGetFileText</code> or <code>CWFindAndLoadFile</code> .				

Return	See “Result Codes for Plug-ins” on page 129.
Remarks	Use <code>CWReleaseFileText</code> to free the contents of a file retrieved using <code>CWGetFileText</code> or <code>CWFindAndLoadFile</code> .
NOTE	Every call to <code>CWGetFileText</code> or <code>CWFindAndLoadFile</code> that returns a non-NULL text or filedata pointer must be balanced by a corresponding call to <code>CWReleaseFileText</code> .

See Also “CWGetFileText” on page 56
“CWFindAndLoadFile” on page 45

CWRemoveProjectEntry

Description	Removes a file from the current target.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWRemoveProjectEntry(CWPluginContext context, const CWFileSpec* fileSpec)</pre>				
Parameters	The parameters for this method include: <table border="1"> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>fileSpec</td><td>The file that needs to be removed.</td></tr> </table>	context	Private, opaque IDE state.	fileSpec	The file that needs to be removed.
context	Private, opaque IDE state.				
fileSpec	The file that needs to be removed.				
Return	See “Result Codes for Plug-ins” on page 129.				
Remarks	Use CWRemoveProjectEntry to remove a file from the project.				
NOTE	If a plug-in calls CWRemoveProjectEntry during a make operation, the IDE restarts the build operation.				
See Also	“Removing a File from a Project” in the <i>IDE SDK Developer’s Guide</i> “CWFileSpec” on page 98				

CWReportMessage

Description	Displays an error, warning, or informational message in the IDE’s errors and warnings window (a modal dialog box).
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWReportMessage(CWPluginContext context, const CWMessageRef* msgRef, const char * line1, const char * line2, short errorlevel, long errorNumber);</pre>
Parameters	The parameters for this method include:

Plug-in API Reference *CWReportMessage*

context	Private, opaque IDE state.
msgRef	Specifies the source text that generated the message. Use NULL for informational messages.
line1, line2	Message text, specified as C character strings. Each string appears on a separate line in the message window.
errorlevel	Specifies the kind of message. Valid values are messageTypeInfo, messageTypeWarning, and messageTypeError.
errorNumber	Reserved value. Use NULL.

Return See “Result Codes for Plug-ins” on page 129.

Remarks To display an informational message using `CWReportMessage`, specify `messageTypeInfo` for `errorlevel` and pass `NULL` for `msgRef`. If you instead pass a pointer to a `CWMessageRef`, the message appears as a warning. The contents of `line1` and `line2` appear on separate lines.

To display an error or warning using `CWReportMessage`, specify `messageTypeError` or `messageTypeWarning` respectively, and fill in `msgRef`. Generally, `line1` contains an error description, and `line2` contains a portion of the source code that caused the error.

To have a message appear in the IDE’s message bar, use `CWAlert` instead.

See Also “`CWAlert`” on page 41

“`CWShowStatus`” on page 80

CWResizeMemHandle

Description Changes the size of a CWMemHandle.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWResizeMemHandle(
    CWPluginContext context,
    CWMemHandle handle,
    long newSize);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
handle	Specifies the handle to resize.
newSize	Specifies the new handle size, in bytes.

Return See “Result Codes for Plug-ins” on page 129.

Remarks Use CWResizeMemHandle to change the size of a handle created with CWAllocMemHandle. CWResizeMemHandle supports resizing a handle to a smaller or larger size. Zero is a valid handle size.

See Also “CWAllocMemHandle” on page 43
 “CWFreeMemHandle” on page 47
 “CWGetMemHandleSize” on page 60
 “CWLLockMemHandle” on page 70
 “CWUnlockMemHandle” on page 82

CWResolveRelativePath

Description Determines the absolute location of a path specified relative to another.

Prototype

```
#include "CWPlugins.h"
CW_CALLBACK CWResolveRelativePath(
    CWPluginContext context,
    const CWRelativePath* relativePath,
    CWFileSpec* fileSpec,
    Boolean create);
```

Plug-in API Reference

CWSetModDate

Parameters The parameters for this method include:

context	Private, opaque IDE state.
relativePath	A pointer to a <code>CWRelativePath</code> structure specifying the relative path to resolve.
fileSpec	Returns a <code>CWFileSpec</code> specifying the absolute location described by the relative path.
create	If the relative path specified by <code>relativePath</code> is not found, and this parameter is set to true, the path will be created.

Return See “Result Codes for Plug-ins” on page 129.

Remarks Use `CWResolveRelativePath` to resolve a relative path to a full path. Plug-ins use this method to determine the absolute path of a relative path stored in project preferences. Plug-ins can use `CWPanelChooseRelativePath` to get the relative path.

See Also “`CWRelativePath`” on page 113
 “`CWPanelChooseRelativePath`” on page 272

CWSetModDate

Description Sets the internal modification date of a file.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWSetModDate(
    CWPluginContext context,
    const CWFileSpec* filespec,
    CWFileTime* moddate,
    Boolean isGenerated);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
filespec	Specifies which file’s modification date to set.

moddate	Specifies the date and time that the file was modified.
isGenerated	Pass true in this argument to tell the IDE that the file has been created during the current request. For example, a compiler that generates a C source code as its output would pass true in isGenerated.

- Return See “Result Codes for Plug-ins” on page 129.
- Remarks Plug-ins should call CWSetModDate after modifying a file in the current project. This informs the IDE that a file has changed and allows it to update its internal dependency tracking information. CWSetModDate does not change a file’s modification date but rather changes the IDE’s internal understanding of when a file was modified.
- Generally, plug-ins should set the isGenerated flag to true. The flag should be set if the plug-in generated the contents of the file. isGenerated should be set false for a file used by the project but not generated by the plug-in.
- See Also “Adding a File to a Project” in the *IDE SDK Developer’s Guide*
- “CWAddProjectEntry” on page 40
- “CWFileSpec” on page 98
- “CWFileTime” on page 100
- “CWGetProjectFile” on page 67

CWSetPluginOSError

- Description Informs the IDE of an OS error code generated during the current plug-in request.
- Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWSetPluginOSError(
    CWPluginContext context,
    CWOSResult      result);
```
- Parameters The parameters for this method include:

Plug-in API Reference *CWShowStatus*

context	Private, opaque IDE state.
result	The error code generated by the host operating system.

- Return See “Result Codes for Plug-ins” on page 129.
- Remarks If a plug-in fails to act on a request because of an error returned by the host operating system, the plug-in may pass the error along to the IDE through `CWSetPluginOSError`.
- See Also “`CWMacOSErrToCWResult`” on page 71

CWShowStatus

- Description Presents progress information during a build operation.
- Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWShowStatus(
    CWPluginContext context,
    const char*      line1,
    const char*      line2);
```
- Parameters The parameters for this method include:

context	Private, opaque IDE state.
line1, line2	Pass text describing plug-in status in these arguments as C character strings.

- Return See “Result Codes for Plug-ins” on page 129.
- Remarks Use `CWShowStatus` to display status information during a long operation. The IDE displays `line1` and `line2` in the linker status window.
- NOTE** This method works only for linkers.
- See Also “`CWReportMessage`” on page 75
“`CWAlert`” on page 41

CWStorePluginData

Description Associates arbitrary, persistently stored plug-in data with a file in the active target.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWStorePluginData(
    CWPluginContext context,
    long whichfile,
    CWDataType type,
    CWMemHandle pluginData);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the project file with which to associate the data. To specify data for the entire target and not a specific file, use kTargetGlobalPluginData.
type	Specifies the kind of data. The plug-in assigns a four-character code for this value.
pluginData	A CWMemHandle containing the data to be stored with the file.

Return See “Result Codes for Plug-ins” on page 129.

Remarks The plug-in must maintain the format of this data, because the IDE never uses this data. Plug-ins may store any data they choose and can assign the data any unique four-character type code.

NOTE Metrowerks reserves lower-case type codes. Use a mixed-case or upper-case value.

By requesting the same type of data for the same file, a plug-in can retrieve data stored by another plug-in.

In the `whichfile` parameter specifies, use the index number of a file in the current target. The index number comes from the target’s link order. File numbers range from 0 to the count returned by `CWGetProjectFileCount - 1`. To store data with the current target (rather than a specific file), use `kTargetGlobalPluginData` in the `whichfile` parameter.

Plug-in API Reference
CWUnlockMemHandle

The data to be stored must reside in a `CWMemHandle` allocated by `CWAllocMemHandle`. The handle becomes the property of the IDE.

By using different data types in the `type` parameter, a plug-in can store more than one kind of data for a file in the project. Calling `CWStorePluginData` with a type of data that has already been stored will replace the original data.

Plug-in data associated with a file is kept in the project’s target data file and thus persists across invocations of the plug-in and the IDE. The IDE stores plug-in data until one of the following events occurs:

- the user chooses **Remove Object Code** from the **Project** menu
- the file is removed from the target
- the user manually deletes the target data folder

See Also “`CWGetPluginData`” on page 65
 “`CWAllocMemHandle`” on page 43

CWUnlockMemHandle

Description Tells the IDE that a plug-in no longer needs access to a `CWMemHandle`.

Prototype

```
#include "DropinCompilerLinker.h"
CW_CALLBACK CWUnlockMemHandle(
    CWPluginContext context,
    CWMemHandle handle);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
handle	The handle to unlock.

Return See “Result Codes for Plug-ins” on page 129.

Remarks After using a handle that has been locked and when finished using any pointers to the data contained in the handle, plug-ins should call `CWUnlockMemHandle` to unlock the handle.

The IDE does not guarantee that a handle’s address remains valid after it has been unlocked. For this reason, plug-ins should not use a

pointer to a memory handle after calling CWUnlockMemHandle on the handle.

The IDE counts calls to CWLockMemHandle and CWUnlockMemHandle. A handle remains locked as long as its lock count remains greater than zero. When initially allocated, handles are unlocked. CWLockMemHandle increases a handle's lock count, and CWUnlockMemHandle decreases its lock count.

- See Also
- “CWAllocMemHandle” on page 43
 - “CWFreeMemHandle” on page 47
 - “CWGetMemHandleSize” on page 60
 - “CWResizeMemHandle” on page 77
 - “CWLockMemHandle” on page 70

CWUserBreak

Description	Checks to see if the user has cancelled the current operation.		
Prototype	<pre>#include "DropinCompilerLinker.h" CW_CALLBACK CWUserBreak(CWPluginContext context);</pre>		
Parameters	The parameters for this method include:<table><tr><td>context</td><td>Private, opaque IDE state.</td></tr></table>	context	Private, opaque IDE state.
context	Private, opaque IDE state.		
Return	See “Result Codes for Plug-ins” on page 129.		
Remarks	Call this routine regularly while acting on a request that requires more than a few seconds of processing time. If CWUserBreak returns cwErrUserCanceled, the plug-in should stop processing, save its files, free its pointers and handles, and return to the IDE as quickly as possible.		
See Also	“CWReportMessage” on page 75		

Plug-In Entry Points

This section describes entry points common to all plug-ins:

- Main Plug-in Entry Point
- CWPlugin_GetDropInFlags Entry Point
- CWPlugin_GetDropInName Entry Point
- CWPlugin_GetDisplayName Entry Point
- CWPlugin_GetPluginInfo

Main Plug-in Entry Point

Description	Provides the main plug-in entry point for servicing requests from the IDE.		
Prototype	<code>CWPlugin_ENTRY (main) (CWPluginContext context)</code>		
Parameters	The parameters for this entry point include: <table border="1"> <tr> <td><code>CWPluginContext</code></td><td>Private, opaque IDE state.</td></tr> </table>	<code>CWPluginContext</code>	Private, opaque IDE state.
<code>CWPluginContext</code>	Private, opaque IDE state.		
Return	See “Result Codes for Plug-ins” on page 129.		
Remarks	The IDE uses to request services from the plug-in. All plug-ins must implement this method.		

To determine what service is being requested by the IDE, plug-ins call `CWGetPluginRequest`, which returns a value indicating the service to perform.

Prior to returning from its main entry point, a plug-in calls `CWDonePluginRequest` with an error code of type `CWResult`. Use `cwNoErr` if no error occurred or `cwErrRequestFailed` if the plug-in could not handle request. `CWDonePluginRequest` then returns an error code to the plug-in, which the plug-in should return to the IDE as its main function result.

The IDE passes the main entry point a single parameter of type `CWPluginContext`. This value holds the IDE’s private state value for the plug-in. This state assists the IDE in providing services to the plug-in. The plug-in must return the context value to the IDE in all calls.

See Also “CWGetPluginRequest” on page 66
 “CWDonePluginRequest” on page 45
 “CWPluginContext” on page 108
 “reqInitialize” on page 127
 “reqTerminate” on page 128

CWPlugin_GetDropInFlags Entry Point

Description	Provides an informational entry point to the IDE that the IDE calls to learn about plug-in features.				
Prototype	<pre>#include <CWPlugins.h> CWPlugin_ENTRY (CWPlugin_GetDropInFlags) (const DropInFlags**, long* flagsSize);</pre>				
Parameters	The parameters for this entry point include: <table border="1"> <tr> <td>flags</td><td>Returns a DropInFlags structure to the IDE. The structure specifies information about the plug-in’s features.</td></tr> <tr> <td>flagsSize</td><td>Specifies the size (in bytes) of the structure returned in flags.</td></tr> </table>	flags	Returns a DropInFlags structure to the IDE. The structure specifies information about the plug-in’s features.	flagsSize	Specifies the size (in bytes) of the structure returned in flags.
flags	Returns a DropInFlags structure to the IDE. The structure specifies information about the plug-in’s features.				
flagsSize	Specifies the size (in bytes) of the structure returned in flags.				
Return	Unless the plug-in could not process the request, the plug-in should return the <code>cwNoErr</code> result code. See “Result Codes for Plug-ins” on page 129.				
Remarks	This entry point returns information about a plug-in to the IDE, by filling in a DropInFlags structure. Plug-ins usually implement this method by returning a pointer to a constant DropInFlags structure. The flagsSize parameter should be set to the size of the structure being returned (normally obtained by applying the C sizeof operator to the DropInFlags type).				
NOTE	COM-only plug-ins do not need to specify flags. All other plug-ins must do so.				

Plug-in API Reference

CWPlugin_GetDropInName Entry Point

- Mac OS On Mac OS, this routine is optional. In its place, a plug-in may provide a 'Flag' resource, specifying the same information.
- See Also “Specifying Plug-In Capabilities” in the *IDE SDK Developer’s Guide*
“Compiler and Linker Drop-In Flags” in the *IDE SDK Developer’s Guide*
“DropInFlags” on page 117

CWPlugin_GetDropInName Entry Point

- Description Provides an informational entry point that the IDE uses to determine the plug-in’s dropin name.
 - Prototype

```
#include <CWPlugins.h>
CWPlugin_ENTRY (CWPlugin_GetDropInName)
(const char** dropinName)
```
 - Parameters The parameters for this entry point include:
 - dropinName Returns a C string containing the drop-in name of the plug-in to the IDE.
 - Return Usually, the plug-in should return the `cwNoErr` result code. See “Result Codes for Plug-ins” on page 129.
 - Remarks This entry point is used by the IDE to determine the plug-in’s dropin name, which is used to keep track of plug-ins in the project file. All plug-ins must implement this entry point.
-
- NOTE** The `CWPlugin_GetDropInName` entry is deprecated in favor of the new `CWPlugin_GetPluginInfo` entry. `GetDropInName` will be called if the dropin name is not set by `CWPlugin_GetPluginInfo`.
-
- See Also “Result Codes for Plug-ins” on page 129

CWPlugin_GetDisplayName Entry Point

- Description Provides an informational entry point that the IDE uses to determine the plug-in’s display name.
- Prototype

```
#include <CWPlugins.h>
CWPlugin_ENTRY (CWPlugin_GetDisplayName)
```

```
(const char** displayName);
```

Parameters	The parameters for this entry point include: displayName Returns a C string containing the display name of the plug-in to the IDE.
Return	Usually, the plug-in should return the <code>cwNoErr</code> result code. See “Result Codes for Plug-ins” on page 129.
Remarks	This entry point is used by the IDE to determine the plug-in’s display name, which is used when displaying the name of the plug-in to the user (when selecting a compiler or linker in Target Settings , for example).
NOTE	The <code>CWPlugin_GetDisplayName</code> entry is deprecated in favor of the new <code>CWPlugin_GetPluginInfo</code> entry. <code>GetDropInName</code> will be called if the dropin name is not set by <code>CWPlugin_GetPluginInfo</code> .
Mac OS	On Mac OS, this routine is optional. In its place, a plug-in may provide a 'STR ' resource, specifying the same information.
See Also	“Result Codes for Plug-ins” on page 129

CWPlugin_GetPluginInfo

Description	Provides an informational entry point that the IDE uses to determine the plug-in’s display name.
Prototype	<pre>#include <CWPlugins.h> CWPLUGIN_ENTRY (CWPlugin_GetPluginInfo) (const CWPluginInfo** pluginInfo);</pre>
Parameters	The parameters for this entry point include: CWPluginInfo A CWPluginInfo object containing information about the plug-in.
Return	Usually, the plug-in should return the <code>cwNoErr</code> result code. See “Result Codes for Plug-ins” on page 129.
Remarks	Use this function to get information about the plug-in, including the dropin name and the display name.

Plug-in API Reference

Data Structures for Plug-ins

NOTE The compiler checks for information from the CWPlugin_GetPluginInfo function. Then it looks for information from the CWPlugin_GetDropInName Entry Point and CWPlugin_GetDisplayName Entry Point functions. Finally, it looks for a resource file. For this reason, using CWPlugin_GetPluginInfo is more efficient than the other two ways of providing this information.

The CWPlugin_GetPluginInfo function supercedes the CWPlugin_GetDropInName Entry Point and CWPlugin_GetDisplayName Entry Point functions.

Mac OS On Mac OS, this routine is optional. In its place, a plug-in may provide a 'STR ' resource, specifying the same information.

See Also "Result Codes for Plug-ins" on page 129

Data Structures for Plug-ins

This section describes the data structures and data types available to all plug-ins.

The plug-in API's data structures are:

- CWAccessPathInfo
- CWAccessPathListInfo
- CWAccessPathType
- CWAddr64
- CWDataType
- CWDependencyType
- CWFileInfo
- CWFileName
- CWFileSpec
- CWFileTime
- CWFrameworkInfo
- CWIDEInfo
- CWMemHandle
- CWMessageRef



- CWNewProjectEntryInfo
- CWNewTextDocumentInfo
- CWOSResult
- CWOverlay1FileInfo
- CWOverlay1GroupInfo
- CWOverlay1Info
- CWPluginContext
- CWPluginInfo
- CWProjectFileInfo
- CWProjectSegmentInfo
- CWRelativePath
- CWRelativePathFormat
- CWRelativePathTypes
- CWResult
- DropInFlags

CWAcessPathInfo

Description Returns information about a single access path.

Definition

```
#include <CWPlugins.h>
typedef struct CWAcessPathInfo
{
    CWFileSpec pathSpec;
    Boolean     recursive;
    long        subdirectoryCount;
} CWAcessPathInfo;
```

Fields The fields in CWAcessPathInfo include:

pathSpec	Specifies a project path (directory).
----------	---------------------------------------

Plug-in API Reference *CWAccessPathListInfo*

recursive	Indicates whether the IDE examines subdirectories of pathSpec to find project files. If true, the IDE searches subdirectories.
subdirectoryCount	Returns the count of subdirectories in pathSpec. This field will be zero if recursive is false.

- Remarks
CWGetAccessPathInfo returns this structure to describe a file access path in the current project. pathSpec indicates the location of the access path. If recursive is true, the plug-in can get the subdirectories by calling CWGetAccessPathSubdirectory. If recursive is true, subdirectoryCount contains the number of subdirectories, which have an index ranging from 0 to subdirectoryCount - 1.
- See Also
“CWGetAccessPathInfo” on page 49
“CWGetAccessPathSubdirectory” on page 51

CWAccessPathListInfo

- Description
Returns properties and counts of the access paths in the current target.
- Definition

```
#include <CWPlugins.h>
typedef struct CWAccessPathListInfo
{
    long      systemPathCount;
    long      userPathCount;
    Boolean    alwaysSearchUserPaths;
    Boolean    convertPaths;
} CWAccessPathListInfo;
```
- Fields
The fields in CWAccessPathListInfo are:

systemPathCount	Returns the number of system paths specified in Edit > Target Settings , in the Access Paths panel.
userPathCount	Returns the number of user paths specified in Edit > Target Settings , in the Access Paths panel.
alwaysSearchUserPaths	Returns true if the IDE has been configured to always search for files in the user access paths (regardless of #include syntax).
convertPaths	On the Mac OS, indicates that the IDE attempts to interpret special characters appearing in file specifications (such as '\ ' and '/') as if they were DOS and Unix path separators. Not used on other host platforms.

Remarks CWGetAccessPathListInfo returns a CWAcessPathListInfo structure containing information about the access paths specified for the current target. systemPathCount and userPathCount indicate the total number of system and user access paths. Plug-ins typically call CWGetAccessPathListInfo to determine the number of user and system paths, prior to enumerating them with CWGetAccessPathInfo.

alwaysSearchUserPaths indicates whether the IDE searches user paths when looking for “system includes.” When set to true, the following statements have the same effect:

```
#include <headerfile.h>
#include "headerfile.h"
```

When alwaysSearchUserPaths is true, plug-ins interfacing with command line tools should include all user search paths on the command line as system search paths.

Mac OS convertPaths indicates whether a Mac OS hosted IDE will attempt to interpret file specifications, such as include file paths, as if they were DOS or Unix paths.

Plug-in API Reference *CWAccessPathType*

See Also “CWGetAccessPathListInfo” on page 50
 “CWGetAccessPathInfo” on page 49

CWAccessPathType

Description Specifies a kind of access path (user or system).

Definition

```
#include <CWPlugins.h>
typedef enum CWAccessPathType
{
    cwSystemPath,
    cwUserPath
} CWAccessPathType;
```

Fields The members of CWAccessPathType are:

cwSystemPath	Specifies a system access path.
cwUserPath	Specifies a user access path.

Remarks CWAccessPathType defines the values used to specify user or system paths when calling CWGetAccessPathInfo and CWGetAccessPathListInfo.

See Also “CWGetAccessPathInfo” on page 49
 “CWGetAccessPathListInfo” on page 50

CWAddr64

Description Stores a 64-bit address.

Definition

```
#include <CWPlugins.h>
typedef struct CWAddr64
{
    long    lo;
    long    hi;
} CWAddr64;
```

Fields The fields in CWAddr64 are:

lo	Holds the lower 32 bits of a 64-bit address.
hi	Holds the upper 32 bits of a 64-bit address.

- Remarks Use the CWAddr64 data structure to hold a 64-bit memory address. The CWOverlay1GroupInfo data structure incorporates this data structure.
- See Also “CWOverlay1GroupInfo” on page 107

CWDataType

- Description Uniquely identifies a category or type of data. Also used for specifying unique identifiers.
- Definition

```
#include <CWPlugins.h>
typedef unsigned long CWDataType;
```
- Remarks Used to identify the type of custom data associated by a plug-in with a project file or a target as a whole. Must be a four-character constant. This type is also used to hold unique, mnemonic 4-character identifiers or descriptors.

NOTE Metrowerks reserves lower-case type codes. Use a mixed-case or upper-case value.

- See Also “CWStorePluginData” on page 81
“CWGetPluginData” on page 65

CWDependencyType

- Description Specifies how one source file relies on the contents of another.
- Definition

```
#include <CWPlugins.h>
typedef enum CWDependencyType {
    cwNoDependency,
    cwNormalDependency,
    cwInterfaceDependency
} CWDependencyType;
```
- Remarks CWFileInfo uses this data type to indicate the type of source dependency to establish between files.
- cwNoDependency specifies that whatever method uses this data type should not establish a dependency. When using

Plug-in API Reference

CWDependencyType

`CWFindAndLoadFile`, you can use this value to load a file without establishing a dependency.

`cwNormalDependency` specifies that a dependent file depends upon the entire contents of the file. Whenever the depended-upon file is modified, the IDE marks the dependent file as dirty. A C/C++ compiler uses this type of dependency to specify dependencies on header files.

`cwInterfaceDependency` specifies that the dependent file depends on the external interface of the file. With this type of dependency, the IDE does not mark the dependent files as dirty when the depended-upon file has been modified. The IDE marks the dependent files as dirty only when the depended-upon file is compiled and the compiler indicates that the interface has changed. The compiler indicates that an interface has changed by setting the `interfaceChanged` flag in the `CWObjectData` data structure. The compiler must set the `interfaceChanged` flag before calling `CWStoreObjectData`. The plug-in determines what is exported through the external interface and when it has changed.

For `cwInterfaceDependency`, the plug-in usually generates the data describing the external interface and saves it using `CWStorePluginData`. On subsequent compilations, the compiler recomputes the interface data, retrieves the interface data from the last time the data was compiled, and checks to see if the interface data has changed. The compiler uses `CWGetPluginData` to determine when the plug-in data last changed. If the data has change, the compiler stores the new interface data and sets the `interfaceChanged` flag when it calls `CWStoreObjectData`.

See Also *"CWFileInfo"* on page 95

"CWFindAndLoadFile" on page 45

"CWStorePluginData" on page 81

"CWGetPluginData" on page 65

"CWStoreObjectData" on page 165

CWFileInfo

Description Holds information returned by CWFindAndLoadFile.

Definition

```
#include <CWPlugins.h>
typedef struct CWFileInfo {
    Boolean      fullsearch;
    char         dependencyType;
    long         isdependentoffile;
    Boolean      suppressload;
    Boolean      padding;
    const char*  filedata;
    long         filedatalength;
    short        filedatatype;
    short        fileID;
    CWFileSpec   filespec;
    Boolean      alreadyincluded;
    Boolean      recordbrowseinfo;
} CWFileInfo;
```

Fields The fields in CWFileInfo are:

fullsearch	true causes the IDE to search the user access paths and then the system paths for the file. false causes the IDE to search only the system paths.
dependencyType	Set this field to a value from CWDependencyType to specify the dependency between the file for which to search and the file specified in the isdependentoffile field.
isdependentoffile	Specifies the project file (by index) that depends on the file for which to search (specified by the filename parameter in a call to CWFindAndLoadFile). To specify the file that the plug-in is currently processing, set this value to kCurrentCompiledFile.

Plug-in API Reference

CWFileInfo

suppressload	true causes the IDE to find a file without loading it into memory. false causes the IDE to load the file into memory. The IDE only loads text files or precompiled header files.
padding	Reserved for use by the IDE. Do not modify the contents of this field.
filedata	Returns a pointer to the file data. The IDE returns NULL if it could not find the file or could not load the file into memory. If the file was found but not loaded into memory, the IDE stores a specification for the file in the filespec field.
filedatalength	Returns the size of the file data, in bytes, if the file was loaded into memory. Returns 0 if the file was not loaded into memory.
filedatatype	Returns the kind of data in the file, if found. Possible values for this field include cwFileTypePrecompiledHeader, cwFileTypeText, or cwFileTypeUnknown
fileID	Returns a unique number for the file, if the IDE found the file. This field is only valid when used by compiler plug-ins.
filespec	Returns the file specification of the file for which the IDE searched. The IDE stores a file specification in this field whether or not it loads the file's contents into memory. Only compiler plug-ins use this field.

alreadyincluded	On return, the IDE sets this value to true if the file has already been searched for in the current operation or false if not. Only compiler plug-ins use this field.
recordbrowseinfo	On return, the IDE sets this value to true to tell the plug-in to generate browser data for the file. Only compiler plug-ins use this field.

- Remarks** Plug-ins use this structure to specify how the IDE should search for files when using `CWFindAndLoadFile`.
- See Also** “`CWDependencyType`” on page 93
 “`CWFindAndLoadFile`” on page 45
 “`cwFileTypePrecompiledHeader`” on page 122
 “`cwFileTypeText`” on page 123
 “`cwFileTypeUnknown`” on page 123
 “`kCurrentCompiledFile`” on page 124

CWFileName

- Description** Specifies the name of a file.
- Definition**

```

#include <CWPlugins.h>
#if CWPlugin_API == CWPlugin_API_MACOS
    typedef char                CWFileName[32];
#elif CWPlugin_API == CWPlugin_API_WIN32
    typedef char                CWFileName[65];
#endif
        
```
- Remarks** Use this type to specify the name of a file. For example, `CWProjectFileInfo` uses this type to specify a file name.
- NOTE** The definition of this type varies by platform.
- See Also** “`CWProjectFileInfo`” on page 109

Plug-in API Reference

CWFileSpec

Description Represents the data type used by the host operating system to specify a file.

Definition

```
#include <CWPlugins.h>
#if CWPLUGIN_API == CWPLUGIN_API_MACOS

    #if CWPLUGIN_LONG_FILENAME_SUPPORT

        #define CWPLUGIN_FILENAME_LEN 256
        typedef struct CWFileSpec
        {
            FSRef          parentDirRef; /* parent
directory */
            HFSUniStr255 filename;      /* unicode file
name */
        } CWFileSpec;

        typedef char
CWFileName[CWPLUGIN_FILENAME_LEN];

    #else

        #define CWPLUGIN_FILENAME_LEN 32
        typedef FSSpec CWFileSpec;
        typedef char
CWFileName[CWPLUGIN_FILENAME_LEN];

    #endif

    typedef unsigned long CWFileTime;
    typedef OSErr          CWOSError;

#elif CWPLUGIN_API == CWPLUGIN_API_WIN32

    typedef unsigned char Boolean;
    typedef struct CWFileSpec { char
path[MAX_PATH]; } CWFileSpec;

    #if CWPLUGIN_LONG_FILENAME_SUPPORT

        #define CWPLUGIN_FILENAME_LEN 256
```

```
typedef char
CWFileName[CWPLUGIN_FILENAME_LEN];

#else

#define CWPLUGIN_FILENAME_LEN 65
typedef char
CWFileName[CWPLUGIN_FILENAME_LEN];

#endif

typedef FILETIME CWFileTime;
typedef DWORD CWOSResult;

#elif CWPLUGIN_API == CWPLUGIN_API_UNIX

#define MAX_PATH MAXPATHLEN
#define CWPLUGIN_FILENAME_LEN 65
#ifndef __MACTYPES__
typedef unsigned char Boolean;
#endif

#ifndef FALSE
#define FALSE 0
#endif

#ifndef TRUE
#define TRUE 1
#endif

typedef struct CWFileSpec { char
path[MAX_PATH]; } CWFileSpec;
typedef char
CWFileName[CWPLUGIN_FILENAME_LEN];
typedef time_t CWFileTime;
typedef int CWOSResult;

#endif
```

Remarks The plug-in API defines this data type to be equivalent to the data type used by the host operating system to specify a file. On Windows, this is a full path. On Mac OS, this is a FileSpec.

Plug-in API Reference

CWFileTime

NOTE The definition of this type varies by platform.

See Also “CWFileInfo” on page 95
“CWGetProjectFile” on page 67

CWFileTime

Description Specifies the modification date and time of a file.

Definition

```
#include <CWPlugins.h>
#if CWPlugin_API == CWPlugin_API_MACOS
    typedef unsigned long    CWFileTime;
#elif CWPlugin_API == CWPlugin_API_WIN32
    typedef FILETIME        CWFileTime;
#endif
```

Remarks The plug-in API defines this data type to be equivalent to the data type used by the host operating system to represent a date and time value.

NOTE The definition of this type varies by platform.

See Also “CWProjectFileInfo” on page 109
“CWSetModDate” on page 78

CWFrameworkInfo

Description Holds information about a framework.

Definition

```
#include <DropInCompilerLinker.h>
typedef struct CWFrameworkInfo {
    CWFileSpec    fileSpec;
    char          version[256];
} CWFrameworkInfo;
```

Fields The fields in CWIDEInfo include:

fileSpec	Location of ".framework" directory
version[256]	Which version directory to use

Remarks If the version string is empty, the IDE uses the current symbolic link.

See Also “CWGetFrameworkInfo” on page 58

CWideInfo

Description Holds information about the IDE.

Definition

```
#include <CWPlugins.h>
typedef struct CWideInfo
{
    unsigned short    majorVersion;
    unsigned short    minorVersion;
    unsigned short    bugFixVersion;
    unsigned short    buildVersion;
    unsigned short    dropinAPIVersion;
} CWideInfo;
```

Fields The fields in CWideInfo are:

majorVersion	Major IDE version number.
minorVersion	Minor IDE version number.
bugFixVersion	Revision number of the IDE.
buildVersion	Build number of the IDE.
dropinAPIVersion	Latest plug-in API version available with this IDE.

Remarks The IDE returns this structure in response to a CWGetIDEInfo call. It provides detailed IDE version information and the plug-in API version.

NOTE This structure contains the latest API version rather than the version the IDE uses to run the current plug-in. Use CWGetAPIVersion to get the version of the plug-in API the IDE uses for the current plug in.

See Also “CWGetIDEInfo” on page 59

“CWGetAPIVersion” on page 52

Plug-in API Reference

CWMemHandle

CWMemHandle

Description	Maintains a resizable memory block by indirect reference.
Definition	<pre>#include <CWPlugins.h> typedef struct CWMemHandlePrivateStruct* CWMemHandle;</pre>
Remarks	Variables of type <code>CWMemHandle</code> store blocks of memory allocated, freed, and referenced through methods exposed by the plug-in API. However, plug-ins cannot access the contents of <code>CWMemHandles</code> directly. Instead, plug-ins must call <code>CWLockMemHandle</code> to obtain a pointer to a handle's contents. When finished accessing a handle, a plug-in should call <code>CWUnlockMemHandle</code> as soon as possible, to let the IDE manage memory as efficiently as possible. Handles can be allocated with <code>CWAllocMemHandle</code>, freed with <code>CWFreeMemHandle</code>, and resized with <code>CWResizeMemHandle</code>. The current size of a handle can be determined with <code>CWGetMemHandleSize</code>.
See Also	"Handles" in the <i>IDE SDK Developer's Guide</i> "<code>CWAllocMemHandle</code>" on page 43 "<code>CWFreeMemHandle</code>" on page 47 "<code>CWResizeMemHandle</code>" on page 77 "<code>CWGetMemHandleSize</code>" on page 60 "<code>CWLockMemHandle</code>" on page 70 "<code>CWUnlockMemHandle</code>" on page 82

CWMessageRef

Description	Specifies the location of the source code that generated a compiler error or warning.
Definition	<pre>#include <CWPlugins.h> typedef struct CWMessageRef { CWFileSpec sourcefile;</pre>

```

    long    linenumber;
    short   tokenoffset;
    short   tokenlength;
    long    selectionoffset;
    long    selectionlength;
} CWMMessageRef;

```

Fields The fields in CWMMessageRef are:

sourcefile	Specifies the source file that caused the error or warning.
linenumber	Specifies the line of text where the error or warning occurred.
tokenoffset	Specifies the beginning of the item within the line of source code that caused the error or warning.
tokenlength	Specifies the length of the offending token. The IDE underlines the token when displaying it.
selectionoffset	Specifies the beginning of the text to highlight when the user selects an error or warning item in the message window. The offset must be relative to the start of the source file.
selectionlength	Specifies the length of the text to select when the user selects an error or warning item in the message window.

Remarks A CWMMessageRef structure describes the source file location at which an error or warning occurred. It provides sufficient information for the IDE to locate and display the source code associated with a message. A plug-in passes this structure to the IDE in a call to CWReportMessage.

See Also “CWReportMessage” on page 75

CWNewProjectEntryInfo

Description Specifies placement information and file flags for a file added to a project.

Plug-in API Reference
CWNewProjectEntryInfo

```

Definition  typedef struct CWNewProjectEntryInfo
              {
                long          position;
                long          segment;
                long          overlayGroup;
                long          overlay;
                const char *  groupPath;
                Boolean       mergeintooutput;
                Boolean       weakimport;
                Boolean       initbefore;
              } CWNewProjectEntryInfo;
  
```

Fields: The fields in CWNewProjectEntryInfo are:

position	Specifies the link order position.
segment	Specifies the segment number.
overlayGroup	Specifies the overlay group number.
overlay	Specifies the overlay number.
groupPath	Specifies the project group to which the file should be added. This C-style string specifies the full path to a group, using a colon (:) to delimit the groups. If the IDE cannot find the specified group, the IDE creates a new group to match the specification.
mergeintooutput	Specifies the value for the the “Merge Into Output” setting. Used only in Mac OS projects.
weakimport	Specifies the value for the the “Import Weak” setting. Used only in Mac OS projects.
initbefore	Specifies the value for the the “Initialize Before” setting. Used only in Mac OS projects.

Remarks This structure specifies where a file added to a project using CWAddProjectEntry appears in the various project file orderings (link, overlay, and segment). In addition, it provides values for several per-file target settings.

The API provides some optional indexes provided, which the IDE ignores if not relevant. For example, Windows projects do not have segments or overlays. Thus, the IDE ignores the segment, overlayGroup, and overlay fields when running on Windows.

Plug-ins can specify kDefaultLinkPosition for any of the position, segment, overlayGroup, and overlay fields. This setting tells the IDE to add the file in the default position in the corresponding project ordering.

If the specified file already exists in a project, the IDE ignores the mergeintooutput, weakimport, and initbefore flags. Also, the IDE ignores these flags if they are not relevant to the file being added to the project. Only projects containing Mac OS targets (which can exist on any platform) use these flags.

See Also “CWAddProjectEntry” on page 40

CWNewTextDocumentInfo

Description Specifies information about a newly created editor window

Definition

```
#include <CWPlugins.h>
typedef struct CWNewTextDocumentInfo
{
    const char*    documentname;
    CWMemHandle    text;
    Boolean        markDirty;
} CWNewTextDocumentInfo;
```

Fields The fields in CWNewTextDocumentInfo are:

documentname	Specifies the title of the document, as a C string.
text	Provides the text handle for the document.
markDirty	Specifies whether the IDE should regard the document as saved. Set this value to true to indicate that the document should be saved prior to closing. Set this value to false to discard the document’s content upon closing.

Remarks The plug-in uses this structure to specify the title and contents of an editor window created with CWCreateNewTextDocument. The

Plug-in API Reference

CWOSResult

text handle becomes the IDE's property. Plug-ins should not dispose of the handle.

See Also "CWCreateNewTextDocument" on page 44

CWOSResult

Description Represents the data type used by the host operating system to specify an error code.

Definition `#include <CWPlugins.h>`

Remarks The plug-in API defines this data type to be equivalent to the data type used by the host operating system to specify an error condition.

NOTE The definition of this type varies by platform.

See Also "CWGetCallbackOSError" on page 53

"CWSetPluginOSError" on page 79

CWOOverlay1FileInfo

Description Describes a file in an overlay.

Definition `#include <CWPlugins.h>`
`typedef struct CWOOverlay1FileInfo {`
`long                whichfile;`
`} CWOOverlay1FileInfo;`

Fields The fields in CWOOverlay1FileInfo are:

whichfile	On return, this value contains the index of the overlay file in the file list view of the current target.
-----------	---

Remarks A plug-in can use CWGetOverlay1FileInfo to iterate through files. Use whichfile in method calls that require a file number, such as CWLoadObjectData or CWGetFileInfo.

See Also "CWGetOverlay1FileInfo" on page 62

"CWLoadObjectData" on page 161

“CWGetFileInfo” on page 55

CWOOverlay1GroupInfo

Description Describes an overlay group.

Definition

```
#include <CWPlugins.h>
typedef struct CWOOverlay1GroupInfo {
    char          name[256];
    CWAddr64      address;
    long          numoverlays;
} CWOOverlay1GroupInfo;
```

Fields The fields in CWOOverlay1GroupInfo are:

name	Returns the name of the overlay, as a C string.
address	Returns the 64-bit absolute load address of the overlay.
numoverlays	Returns the number of overlays in this group.

Remarks Use CWGetOverlay1GroupInfo to iterate through overlay groups.

See Also “CWGetOverlay1GroupInfo” on page 63

CWOOverlay1Info

Description Returns information about one overlay.

Definition

```
#include <CWPlugins.h>
typedef struct CWOOverlay1Info {
    char          name[256];
    long          numfiles;
} CWOOverlay1Info;
```

Fields The fields in CWOOverlay1Info are:

name	Returns the name of the overlay, as a C string.
numfiles	Returns the number of files compiled into this overlay.

Remarks Use CWGetOverlay1Info to iterate through overlay groups.

Plug-in API Reference

CWPluginContext

See Also “CWGetOverlay1Info” on page 64

CWPluginContext

Description Opaque reference to IDE state.

Definition

```
#include <CWPlugins.h>
typedef struct
    CWPluginPrivateContext* CWPluginContext;
```

Remarks This data type maintains information for the IDE during calls to a plug-in. This information belongs to the IDE. Plug-ins should never try to modify this information.

The Main Plug-in Entry Point for a plug-in receives a parameter of this type. The plug-in must preserve this value and pass it back to the IDE in the context parameter of any services the plug-in uses.

The informational plug-in entry points do not receive such a parameter, because they simply report static information back to the IDE.

For more information, see “Main Entry Point context Parameter” in the *IDE SDK Developer’s Guide*.

CWPluginInfo

Description Contains information about a plug-in.

Definition

```
#include <CWPlugins.h>
typedef struct CWPluginInfo
{
    short                version;
    const char*          companyName;
    const char*          pluginName;
    const char*          pluginDisplayName;
    const char*          familyName;
    unsigned short       majorIDEVersion; // Required
    unsigned short       minorIDEVersion;
} CWPluginInfo;
```

Fields	The fields in CWPluginInfo are:	
	version	Optional.
	companyName	Optional. For example, Metrowerks.
	pluginName	Optional. The name of your plug-in module. If you do not provide this name, the compiler uses Dropin->GetName(). Otherwise, it looks for a resource file.
	pluginDisplayName	The name of your plug-in, as you would like users to see it in the IDE.
	familyName	For example, "Java".
	majorIDEVersion	This field is required.
	minorIDEVersion	Optional.

Remarks The CWPlugin_GetPluginInfo entry point passes CWPluginInfo to the IDE, to let the IDE identify the plug-in.

NOTE You can get better performance from the compiler by providing a value in the pluginName field.

CWProjectFileInfo

Description Contains information about a file in a project.

Definition

```
#include <CWPlugins.h>
typedef struct CWProjectFileInfo
{
    CWFileSpec      filespec;
    CWFileTime      moddate;
    short           segment;
    Boolean          hasobjectcode;
    Boolean          hasresources;
    Boolean          isresourcefile;
    Boolean          weakimport;
    Boolean          initbefore;
    Boolean          gendebg;
    CWFileTime      objmoddate;
    CWFileName      dropinname;
```

Plug-in API Reference

CWProjectFileInfo

```

short      fileID;
Boolean    recordbrowseinfo;
Boolean    reserved;
#if CWPlugin_HOST == CWPlugin_HOST_MACOS
OSType     filetype;
OSType     filecreator;
#endif
Boolean    hasunitdata;
Boolean    mergeintooutput;
unsigned long unitdatadependencytag;
} CWProjectFileInfo;

```

Fields The fields in CWProjectFileInfo include:

filespec	Returns the file specification for the file.
moddate	Returns the date and time that the file was last modified.
segment	Returns the number of the segment that into which the file is linked.
hasobjectcode	Indicates whether or not the file generates object code (not whether any actually exists).
hasresources	Indicates whether or not the file generates resource data (not whether any actually exists).
isresourcefile	Indicates whether or not the file is a binary resource file to link into the final executable.
weakimport	Indicates whether or not the file's Import Weak flag is set.
initbefore	Indicates whether or not the file's Init Before flag is set.
gendebug	Indicates whether or not debugging information should be generated for the file.

objmoddate	Indicates the date and time at which the file's object code was last modified.
dropinname	Returns the name of the plug-in that the IDE calls on to process the file.
fileID	Returns a unique number, used in browser records to identify source files.
recordbrowseinfo	Indicates whether or not browser information should be generated for the file.
reserved	This field is reserved by the IDE.
filetype	Returns the Mac OS file type of the file. This field is defined only for Mac OS-hosted plug-ins.
filecreator	Returns the Mac OS creator signature of the file. This field is defined only for Mac OS-hosted plug-ins.
hasunitdata	Indicates whether the file has associated unit data (useful only for Pascal compilers).
mergeintooutput	Returns true if the "Merge Into Output" checkbox is checked in the project inspector.
unitdatadependencytag	Returns a dependency tag (checksum) of unit data (useful only for Pascal compilers).

Remarks CWGetFileInfo uses this data structure to describe a file in the active project.

filespec returns the file specification for the specified file. The IDE returns the most recent known (cached) file location if the plug-in passes 0 for the checkFileLocation parameter when calling CWGetFileInfo. Passing 1 tells the IDE to verify the file's current location on disk.

Plug-in API Reference

CWProjectFileInfo

The `hasobjectcode`, `hasresources`, and `isresourcefile` fields have very precise and not entirely intuitive meanings:

- `hasobjectcode` means that the specified file is a source file and that compiling the source produces object data.
- `hasresources` means that the specified file is a source file and that compiling it produces resource data.
- `isresourcefile` means that the file itself is resource data, which should simply be copied into the target binary (this is typically only the case on the Mac OS).

NOTE Neither `hasobjectcode` nor `hasresources` imply that such data actually exists for the file. Plug-ins determine whether this data exists by loading the object data with `CWLoadObjectData` and examining the data. The plug-in should call `CWFreeObjectData` when done examining the data.

`dropinname` is the internal “plug-in name” of the plug-in used to process the file, not the plug-in’s display name.

Plug-ins use `fileID` when generating browse information. This file ID uniquely identifies a target file. Plug-ins should emit this value in browser record fields that specify the source file for a symbol. See “Browser Reference” on page 229 for more information.

`gendebug` reflects the debug setting in the main project window. Compilers and linkers should only generate debug code for files which have this flag set.

Mac OS The `segment` field applies only to Mac 68K targets, and its value is determined by the assignment of files to segments in the **Segments** tab of the project.

The `weakimport`, `initbefore`, and `mergeintooutput` flags apply only to library files included in Mac OS targets. The values of the flags match the values established in the project inspector window.

The `weakimport` flag indicates whether the corresponding checkbox is set in the project inspector window for this file. When set, the flag indicates that the target application wishes to run even

if it cannot bind to all imported library symbols (variables and routines) at launch time.

mergeintooutput indicates that the library should be copied into the final executable. This ensures that the library will be available at runtime (much like statically linking a dynamic library).

See Also “CWGetFileInfo” on page 55

CWProjectSegmentInfo

Description Contains information about a segment in the active project.

Definition

```
#include <CWPlugins.h>
typedef struct CWProjectSegmentInfo {
    char  name[32];
    short attributes;
} CWProjectSegmentInfo;
```

Fields The fields in CWProjectFileInfo include:

name	The name of the segment
attributes	The Resource Manager attributes for the segment.

Remarks CWGetSegmentInfo uses this data structure to describe a segment in the current project target, for Mac OS 68K targets only. Segments are blocks of compiled code, stored in ‘CODE’ resources in the Mac OS 68K runtime environment. The **Segment** tab of a 68K target specifies which files should be placed in each ‘CODE’ resource.

A CWProjectSegmentInfo structure specifies the name and resource manager attributes for a segment (any combination of preload, locked, purgeable, protected, system heap).

See Also “CWGetSegmentInfo” on page 68

CWRelativePath

Description Describes a directory specified relative to another.

Definition

```
#include <CWPlugins.h>
typedef struct CWRelativePath
{
    short          version;
```

Plug-in API Reference *CWRelativePath*

```

        unsigned char  pathType;
        unsigned char  pathFormat;
        char           userDefinedTree[256];
        char           pathString[512];
    } CWRelativePath;

```

Fields The fields in CWRelativePath include:

version	Specifies the version of the CWRelativePath record.
pathType	Specifies the reference directory to which the path is relative. This should be one of the CWRelativePathTypes enumeration values: type_Absolute, type_Project, type_Compiler, or type_System, type_UserDefined.
pathFormat	Specifies the OS path syntax used in pathString. This should be one of the CWRelativePathFormat enumeration values: format_Generic, format_Mac or format_Win, format_Unix.
userDefinedTree	Specifies the path to the reference directory when pathType is type_UserDefined.
pathString	Specifies a directory location, relative to the location specified by pathType and, optionally, userDefinedTree.

Remarks A CWRelativePath structure specifies a relative directory location. A relative path specifies a directory location using a sequence of subdirectories starting from another absolute reference location.

The reference location usually, specified by pathType, generally refers to a location that typically exists on all IDE hosts. For example, the directory containing the compiler and its support files exists on all IDE hosts.

The partial path string stored in pathString may be formatted for any supported platform.

Relative paths are especially useful when the reference path may have a different location on different host systems. Relative paths help to make directory specifications portable. They also eliminate the need to determine the locations of standard directories.

CWPanelChooseRelativePath returns CWRelativePath structures that may be resolved to full absolute paths with CWResolveRelativePath.

- See Also
- “CWResolveRelativePath” on page 77
 - “CWPanelChooseRelativePath” on page 272
 - “CWRelativePathFormat” on page 115
 - “CWRelativePathTypes” on page 116

CWRelativePathFormat

Description Specifies the operating system format of a relative path.

Definition

```
#include <CWPlugins.h>
typedef enum CWRelativePathFormat
{
    format_Generic = 0,
    format_Mac,
    format_Win,
    format_Unix
} CWRelativePathFormat;
```

Values These constants have the following meanings:

format_Generic	Indicates a simple file name, without any additional parts, such as drive or path specifiers.
format_Mac	Indicates Mac OS path syntax. Directory names are separated by colon (':') characters.
format_Win	Indicates Windows path syntax. Directory names are separated by backslash characters ('\').
format_Unix	Indicates Unix path syntax. Directory names are separated by slash characters ('/').



Plug-in API Reference
CWRelativePathTypes

- Remarks This type specifies the format of a relative path in a CWRelativePath structure returned by CWPannelChooseRelativePath.
- See Also “CWResolveRelativePath” on page 77
- “CWRelativePath” on page 113
- “CWRelativePathTypes” on page 116
- “CWPannelChooseRelativePath” on page 272

CWRelativePathTypes

- Description Describes an absolute location referred to by a relative path.
- Definition

```
#include <CWPlugins.h>
typedef enum CWRelativePathTypes
{
    type_Absolute= 0,
    type_Project,
    type_Compiler,
    type_System,
    type_UserDefined
} CWRelativePathTypes;
```
- Values These constants have the following meanings:

type_Absolute	Indicates that the relative path is not actually relative but instead specifies a fully qualified absolute path.
type_Project	Indicates that the path is specified relative to the directory containing the project.
type_Compiler	Indicates that the path is specified relative to the directory containing the compiler.

type_System	Indicates that the path is specified relative to the directory containing the system. On Windows, this refers to the Windows\System directory on the boot drive. On Mac OS, this is the System folder on the boot volume.
type_UserDefined	Indicates that the relative path refers to a directory specified by the plug-in.

- Remarks This enumeration specifies the starting point for a relative path. Relative paths constructed by `CWPanelChooseRelativePath` return a value of this type in the `pathType` field.
- See Also “CWResolveRelativePath” on page 77
“CWRelativePath” on page 113
“CWRelativePathFormat” on page 115
“CWPanelChooseRelativePath” on page 272

CWResult

- Description Specifies a result code returned by a call to the IDE.
- Definition

```
#include <CWPlugins.h>
typedef long CWResult;
```
- Remarks All routines provided by the plug-in API return a result of type `CWResult`. The IDE directly returns the most common errors. For OS-specific errors, the `CWResult` is `cwErrOSError`, and the OS-specific error can be obtained by calling `CWGetCallbackOSError`.
- See Also “cwErrOSError” on page 131
“CWGetCallbackOSError” on page 53

DropInFlags

- Description Describes a plug-in and its capabilities to the IDE.

Plug-in API Reference

DropInFlags

Definition	<pre>#include <CWPlugins.h> typedef struct DropInFlags { short rsrcversion; CWDDataType dropintype; unsigned short earliestCompatibleAPIVersion; unsigned long dropinflags; CWDDataType edit_language; unsigned short newestAPIVersion; } DropInFlags, **DropInFlagsHandle;</pre>	
Fields	The fields in DropInFlags are:	
rsrcversion		Indicates the version number of the DropInFlags structure. Usually, new plug ins use the current version of the structure. Use kCurrentDropInFlagsVersion to indicate the latest version.
dropintype		Specifies the plug-in type. Should be one of CWDROPINLINKERTYPE, CWDROPINCOMPILETYPE, CWDROPINPREFSTYPE, CWDROPINVCSTYPE, or CWDROPINCOMTYPE.
earliestCompatibleAPIVersion		Specifies the earliest API version with which the plug-in can operate properly.
dropinflags		Contains bit flags indicating plug-in capabilities. Flag meanings vary with each plug-in type.
edit_language		The source language of files accepted by the plug-in (where relevant).
newestAPIVersion		Specifies the most recent API version with which the plug-in can operate properly.

Remarks The DropInFlags structure is returned by compiler, linker, and version control plug-ins to the IDE via their CWPlugin_GetDropInFlags entry point. This structure provides the IDE with information about the plug-in and its capabilities.

The `dropintype` field consists of bit flags having different meanings for each plug-in type. See “Compiler Capability Flags,” “Linker Capability Flags,” and “VCS Plug-in Capability Flags” in the *IDE SDK Developer’s Guide* and for more information. Preference panels use a similar but different structure to report their capabilities. See “PanelFlags” on page 335 for more information.

The `newestAPIVersion` field should usually be set to `DROPINCOMPILERLINKERAPIVERSION` or `DROPINPANELAPIVERSION`, depending upon the type of the plug-in. Each update of the API headers sets this value to the latest API version.

For most plug-ins, `earliestCompatibleAPIVersion` should be set to the latest version of the API that was in effect when the plug-in was first created. Usually, this should be done using a specific version, rather than one of `DROPINCOMPILERLINKERAPIVERSION` or `DROPINPANELAPIVERSION` (since their values change with API releases, but the earliest API version supported by a plug-in usually does not).

See Also “Specifying Plugin Capabilities” in the *IDE SDK Developer’s Guide*
“Compiler Capability Flags” in the *IDE SDK Developer’s Guide*
“Linker Capability Flags” in the *IDE SDK Developer’s Guide*
“CWPlugin_GetDropInFlags Entry Point” on page 85

Constants for Plug-ins

This section describes the constants and defined values (created with `#define` statements) used by the CodeWarrior Plug-in API.

The following list shows the constants and defined values available in the Plug-in API:

- `CWDROPINCOMPILERTYPE`
- `CWDROPINLINKERTYPE`
- `CWDROPINPREFSTYPE`
- `CWDROPINPREFSTYPE_1`

Plug-in API Reference

CWDROPINCOMPILETYPE

- CWDROPINVCSTYPE
- cwFileTypePrecompiledHeader
- cwFileTypeText
- cwFileTypeUnknown
- kCurrentCompiledFile
- kDefaultLinkPosition
- kTargetGlobalPluginData
- messagetypeInfo
- messagetypeWarning
- messagetypeError
- reqAbout
- reqIdle
- reqInitialize
- reqPrefsChange
- reqTerminate

CWDROPINCOMPILETYPE

Description	Defines the file type of compiler plug-ins.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ CWDROPINCOMPILETYPE = 'Comp', /* ... */ };</pre>
Remarks	Use CWDROPINCOMPILETYPE in the dropintype field of a DropInFlags structure to specify that a plug-in is a compiler.
Mac OS	A plug-in should also use this value for the file type of a compiler plug-in.

CWDROPINLINKERTYPE

Description	Defines the file type of linker plug-ins.
-------------	---

Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ CWDROPINLINKERTYPE = 'Link', /* ... */ };</pre>
Remarks	Use CWDROPINLINKERTYPE in the dropintype field of a DropInFlags structure to specify that a plug-in is a linker.
Mac OS	A plug-in should also use this value for the file type of a linker plug-in.

CWDROPINPREFSTYPE

Description	Defines the file type of settings panel plug-ins.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ CWDROPINPREFSTYPE = 'PanL', /* ... */ };</pre>
Remarks	Use CWDROPINPREFSTYPE in the dropintype field of a DropInFlags structure to specify that a plug-in is a preference panel.
Mac OS	A plug-in should also use this value for the file type of a preference panel plug-in.

CWDROPINPREFSTYPE_1

Description	Defines the dropin type of settings panel plug-ins created for versions of the IDE before version 2.0.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ CWDROPINPREFSTYPE_1 = 'Panl', /* ... */ };</pre>

Plug-in API Reference

CWDROPINVCSTYPE

	<code>};</code>
Remarks	CWDROPINPREFSTYPE_1 was used in the <code>dropintype</code> field of a <code>DropInFlags</code> structure for 1.x versions of the IDE.
Mac OS	This value was also used for the file type of a version 1 preference panel plug-in.
NOTE	New plug-ins should not use this type.

CWDROPINVCSTYPE

Description	Defines the file type of version control system (VCS) plug-ins.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ CWDROPINVCSTYPE = 'VCS ' /* ... */ };</pre>
Remarks	Use CWDROPINVCSTYPE in the <code>dropintype</code> field of a <code>DropInFlags</code> structure to specify that a plug-in is a version control system.
Mac OS	A plug-in should also use this value for the file type of a VCS plug-in.

cwFileTypePrecompiledHeader

Description	Specifies that the IDE has found a cached precompiled header file.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ cwFileTypePrecompiledHeader /* ... */ };</pre>
Remarks	The IDE uses <code>cwFileTypePrecompiledHeader</code> in the <code>filedatatype</code> field of a <code>CWFileInfo</code> data structure to specify that it has found a precompiled header file that has been cached.

See Also "CWFileInfo" on page 95

cwFileTypeText

Description Specifies that the IDE has found a text file.

Definition

```
#include "DropinCompilerLinker.h"
enum {
    /* ... */
    cwFileTypeText,
    /* ... */
};
```

Remarks The IDE uses `cwFileTypeText` in the `filedatatype` field of a `CWFileInfo` data structure to specify that it has found a precompiled header file that has been cached.

See Also "CWFileInfo" on page 95

cwFileTypeUnknown

Description Specifies that the IDE has found a binary file.

Definition

```
#include "DropinCompilerLinker.h"
enum {
    /* ... */
    cwFileTypeUnknown,
    /* ... */
};
```

Remarks The IDE uses `cwFileTypeUnknown` in the `filedatatype` field of a `CWFileInfo` data structure to specify that it has found a file that contains unknown binary data.

See Also "CWFileInfo" on page 95

cwSystemPath

Description Specifies a system path as configured in the **Access Paths** panel.

Definition

```
#include <CWPlugins.h>
```

Remarks Specifies a system path in calls to `CWGetAccessPathInfo` and `CWGetAccessPathSubdirectory`. The IDE searches system after searching user paths.

Plug-in API Reference

cwUserPath

See Also “CWOVerlay1GroupInfo” on page 107

cwUserPath

Description Specifies a user path as configured in the **Access Paths** panel.

Definition `#include <CWPlugins.h>`

FieldRemarks Specifies a user path in calls to `CWGetAccessPathInfo` and `CWGetAccessPathSubdirectory`. The IDE searches user paths before searching system paths.

See Also “CWOVerlay1GroupInfo” on page 107

kCurrentCompiledFile

Description Specifies that a file is dependent on the file that is currently being compiled by the IDE.

Definition `#include "DropinCompilerLinker.h"`
`#define kCurrentCompiledFile -1L`

Remarks The IDE uses `kCurrentCompiledFile` in the `isdependentoffile` field of a `CWFileInfo` data structure to specify that a file depends on the file that is currently being compiled.

See Also “CWFileInfo” on page 95

kDefaultLinkPosition

Description Specifies the default position in an ordering of project files.

Definition `#include "DropinCompilerLinker.h"`
`#define kDefaultLinkPosition -1L`

Remarks Plug-ins use `kDefaultLinkPosition` in a `CWNewProjectEntryInfo` structure for any of the `position`, `segment`, `overlayGroup`, and `overlay` fields, to specify that the IDE should insert the new file in the default position with the respective file ordering.

See Also “CWNewProjectEntryInfo” on page 103

kTargetGlobalPluginData

Description	Specifies data that applies globally to a project's target rather than a single file.
Definition	<pre>#include "CWPlugins.h" #define kTargetGlobalPluginData -1L</pre>
Remarks	A plug-in passes kTargetGlobalPluginData in the whichfile parameters for CWStorePluginData and CWGetPluginData to specify that the IDE should store data globally for the active target rather than a single file in the target.
See Also	"CWStorePluginData" on page 81 "CWGetPluginData" on page 65

messagetypeInfo

Description	Specifies the nature of an item to appear in a message window.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ messagetypeInfo, /* ... */ };</pre>
Remarks	Use this constant in the errorlevel argument for CWReportMessage to specify that a message provides a piece of information that isn't an error or warning.
See Also	"CWReportMessage" on page 75

messagetypeWarning

Description	Specifies the nature of an item to appear in a message window.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ messagetypeWarning, /* ... */ };</pre>

Plug-in API Reference

messagetypeError

Remarks Use this constant in the `errorlevel` argument for `CWReportMessage` to specify that a message is a warning.

See Also “`CWReportMessage`” on page 75

messagetypeError

Description Specifies the nature of an item to appear in a message window.

Definition

```
#include "DropinCompilerLinker.h"
enum {
    /* ... */
    messagetypeError,
    /* ... */
};
```

Remarks Use this constant in the `errorlevel` argument for `CWReportMessage` to specify that a message is an error.

See Also “`CWReportMessage`” on page 75

reqAbout

Description Asks the plug-in to display its about box.

Definition

```
#include "DropinCompilerLinker.h"
enum {
    /* ... */
    reqAbout = -101
    /* ... */
};
```

Remarks This request is sent by the IDE to ask a plug-in to display its about box.

NOTE Currently, the IDE only sends this request for version control plug ins when the user selects **About** from the version control menu.

See Also “`CWGetPluginRequest`” on page 66

reqIdle

Description	Gives the plug-in a chance to perform a low priority background task.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ reqIdle = -100 /* ... */ };</pre>
Remarks	The IDE sends this request occasionally (roughly once every two minutes), allowing a plug-in to perform some low priority background task. Plug-ins should avoid making assumptions about the exact frequency of reqIdle requests.
NOTE	Currently, the IDE only sends this request for version control plug ins.
CAUTION	Plug-ins cannot make any other calls to the IDE when handling this request.
See Also	"CWGetPluginRequest" on page 66

reqInitialize

Description	Asks the plug-in to set itself to a known initial state.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ reqInitialize = -2 /* ... */ };</pre>
Remarks	After the IDE has loaded the plug-in into memory, the IDE uses this request to initialize the plug-in. While handling a reqInitialize request, a plug-in can call only CWGetPluginRequest and CWDonePluginRequest.
See Also	"CWGetPluginRequest" on page 66

Plug-in API Reference

reqPrefsChange

“reqInitPanel” on page 361

reqPrefsChange

Description	Informs the plug-in that its preferences have changed.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ reqInitialize = -102 /* ... */ };</pre>
Remarks	This message is sent when the user changes the preferences for a plug-in. Typically, a plug-in may wish to respond by reloading its preference data.

NOTE Currently, the IDE sends this request only for version control plug ins.

See Also “CWGetPluginRequest” on page 66

reqTerminate

Description	Asks the plug-in to clean up before it is unloaded.
Definition	<pre>#include "DropinCompilerLinker.h" enum { /* ... */ reqTerminate = -1 /* ... */ };</pre>
Remarks	The IDE issues this request before unloading the plug-in from memory. While handling a reqTerminate request, a plug-in can call only CWGetPluginRequest and CWDonePluginRequest.

See Also “CWGetPluginRequest” on page 66

“reqTermPanel” on page 366

Result Codes for Plug-ins

This section lists error codes that the IDE recognizes. A plug-in should return one of the values described in this section when handling a request from the IDE. The IDE returns a value from this section when a plug-in calls it.

The IDE sends and expects to receive the following result codes:

- `cwErrCantSetAttribute`
- `cwErrFileNotFound`
- `cwErrInvalidCallback`
- `cwErrInvalidMPCallback`
- `cwErrInvalidParameter`
- `cwErrOSError`
- `cwErrOutOfMemory`
- `cwErrRequestFailed`
- `cwErrSilent`
- `cwErrStringBufferOverflow`
- `cwErrUnknownFile`
- `cwErrUserCanceled`
- `cwNoErr`

NOTE The literal values listed in this section are for reference only and do not appear in the headers. Always use constants rather than the literal values in plug-in code.

`cwErrCantSetAttribute`

Description The plug-in requested inappropriate flags in `CWAddProjectEntry`.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrCantSetAttribute = 11,
```

Plug-in API Reference

cwErrFileNotFound

```
/* ... */
};
```

cwErrFileNotFound

Description A file was not found on disk.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrFileNotFound = 8,
    /* ... */
};
```

cwErrInvalidCallback

Description The IDE cannot provide a routine for a plug-in.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrInvalidCallback = 4,
    /* ... */
};
```

Remarks The IDE issues this error code when a plug-in tries to call an IDE routine that is not appropriate for the request the plug-in is acting on or the plug-in's type.

cwErrInvalidMPCallback

Description The plug-in cannot act on a request while executing as a multiprocessing thread.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrInvalidMPCallback = 5,
    /* ... */
};
```

cwErrInvalidParameter

Description An IDE routine cannot complete because it received a parameter value it cannot use.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrInvalidParameter = 3,
    /* ... */
};
```

cwErrOSError

Description The host operating system issued an error.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrOSError = 6,
    /* ... */
};
```

cwErrOutOfMemory

Description The IDE could not find enough memory to complete a memory allocation.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrOutOfMemory = 7,
    /* ... */
};
```

cwErrRequestFailed

Description The plug-in could not service a request.

Definition

```
#include <CWPluginErrors.h>
enum
```

Plug-in API Reference

cwErrSilent

```
{
    /* ... */
    cwErrRequestFailed = 2,
    /* ... */
};
```

cwErrSilent

Description An error code that a plug-in may pass back to the IDE when a request failed but the plug-in does not want the IDE to report the error to the user. This value lets the plug-in suppress the IDE's error reporting, so that the plug-in can report the error itself.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrSilent = 10,
    /* ... */
};
```

cwErrStringBufferOverflow

Description An output string buffer was too small.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrStringBufferOverflow = 12,
    /* ... */
};
```

cwErrUnknownFile

Description An invalid file number was used.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrUnknownFile = 9,
    /* ... */
};
```

```
};
```

cwErrUserCanceled

Description The user has issued a command to stop the current operation.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwErrUserCanceled = 1,
    /* ... */
};
```

cwNoErr

Description An operation was completed successfully.

Definition

```
#include <CWPluginErrors.h>
enum
{
    /* ... */
    cwNoErr = 0,
    /* ... */
};
```



Freescal Semiconductor, Inc.

Plug-in API Reference

cwNoErr

Compiler and Linker Plug-in Reference

This chapter describes the plug-in API services available to compilers, linkers, pre-linkers, and post-linkers.

Overview

This chapter covers the following topics:

- Routines for Compiler and Linker Plug-ins
- User Routines for Compiler and Linker Plug-ins
- Data Structures for Compiler and Linker Plug-ins
- Constants for Compiler and Linker Plug-ins
- Result Codes for Compiler and Linker Plug-ins

Routines for Compiler and Linker Plug-ins

This section lists the routines a compiler, linker, pre-linker, or post-linker may call.

Alphabetical Routine Index

This section lists all compiler and linker routines alphabetically.

- `CWCachePrecompiledHeader`
- `CWDisplayLines`
- `CWFreeObjectData`
- `CWGetBrowseOptions`
- `CWGetBuildSequenceNumber`



Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

Functional Routine Index

- CWGetCommandLineArgs
- CWGetEnvironmentVariable
- CWGetEnvironmentVariableCount
- CWGetMainFileID
- CWGetMainFileNumber
- CWGetMainFileSpec
- CWGetMainFileText
- CWGetModifiedFiles
- CWGetPrecompiledHeaderSpec
- CWGetResourceFile
- CWGetStoredObjectFileSpec
- CWGetSuggestedObjectFileSpec
- CWGetTargetInfo
- CWGetTargetStorage
- CWGetWorkingDirectory
- CWIsAutoPrecompiling
- CWIsCachingPrecompiledHeaders
- CWIsGeneratingDebugInfo
- CWIsPrecompiling
- CWIsPreprocessing
- CWLoadObjectData
- CWPutResourceFile
- CWSetTargetInfo
- CWSetTargetStorage
- CWStoreObjectData

Functional Routine Index

This section lists all routines grouped by function.

Current File Information

- CWGetMainFileID
- CWGetMainFileNumber

- CWGetMainFileSpec
- CWGetMainFileText

Request Handling

- CWIsAutoPrecompiling
- CWIsCachingPrecompiledHeaders
- CWIsGeneratingDebugInfo
- CWIsPrecompiling
- CWIsPreprocessing
- CWGetBrowseOptions
- CWGetBuildSequenceNumber

Managing Object Data

- CWFreeObjectData
- CWLoadObjectData
- CWStoreObjectData
- CWGetStoredObjectFileSpec
- CWGetSuggestedObjectFileSpec

Managing Precompiled Headers

- CWCachedPrecompiledHeader
- CWGetPrecompiledHeaderSpec

Managing Target Information

- CWGetTargetInfo
- CWSetTargetInfo
- CWGetModifiedFiles

Managing Target Storage

- CWGetTargetStorage
- CWSetTargetStorage

Target Runtime Settings

- CWGetCommandLineArgs

Compiler and Linker Plug-in Reference

CWCachedPrecompiledHeader

- CWGetEnvironmentVariable
- CWGetEnvironmentVariableCount
- CWGetWorkingDirectory

User Interaction

- CWDisplayLines
- CWGetResourceFile
- CWPutResourceFile

CWCachedPrecompiledHeader

Description Stores precompiled header data in a RAM cache for faster access.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWCachedPrecompiledHeader(
    CWPluginContext context,
    const CWFileSpec* filespec,
    CWMemHandle pchhandle);
```

Parameters The parameters for this Method include:

context	Private, opaque IDE state.
filespec	Specifies the location of the precompiled header file corresponding to the header data specified in pchhandle.
pchhandle	Contains the precompiled header data.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks pchhandle contains the precompiled header data. filespec contains the file specification for the corresponding precompiled header file. After a plug-in successfully caches a header, attempts to load the file with CWFindAndLoadFile return the cached data.

NOTE The IDE stores only two precompiled headers in the precompiled header cache.



Before calling `CWCachePrecompiledHeader`, a plug-in should call `CWIsCachingPrecompiledHeaders` to ensure that the IDE has sufficient memory to cache precompiled headers. If `CWIsCachingPrecompiledHeaders` returns true, the plug-in can call `CWCachePrecompiledHeader`. A true result does not guarantee that `CWCachePrecompiledHeader` succeeds. Plug-ins should still check for successful completion.

See Also “CWMemHandle” on page 102
“CWIsCachingPrecompiledHeaders” on page 158

CWDisplayLines

Description Reports progress information to the IDE for display during the current compilation.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWDisplayLines(
    CWPluginContext context,
    long nlines);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
nlines	Specifies the number of line processed so far by the plug-in.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks Call this routine regularly while compiling to update the IDE’s progress window. Plug-ins should call this routine often enough to provide good feedback but not so often as to adversely affect performance. A good rule of thumb is to call `CWDisplayLines` once per 50-100 lines of source code or a few times per second.

CAUTION Ensure that whatever logic you use to determine when to call `CWDisplayLines` is fast. Otherwise, simply determining when to use this method can affect performance.

See Also “CWShowStatus” on page 80

Compiler and Linker Plug-in Reference

CWFreeObjectData

CWFreeObjectData

Description Releases memory allocated by CWLoadObjectData.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWFreeObjectData(
    CWPluginContext context,
    long whichfile,
    CWMemHandle objectdata);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the file by index, in target link order, corresponding to the object data to release.
objectdata	Contains the object data to be disposed, returned by a previous call to CWLoadObjectData.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks Call CWFreeObjectData to free object data previously loaded by CWLoadObjectData. Be careful to pass the index in whichfile of the file corresponding to the data passed in objectdata.

See Also “CWLoadObjectData” on page 161

CWGetBrowseOptions

Description Tells a plug-in what types of browser symbols to generate.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetBrowseOptions(
    CWPluginContext context,
    CWBrowseOptions* browseOptions);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
browseOptions	Returns a CWBrowseOptions structure specifying the types of symbols the plug-in should generate.

Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129
Remarks	The IDE expects all compiler plug-ins that support generation of browser information to generate information for global variables and objects. A compiler plug-in should call <code>CWGetBrowseOptions</code> to determine which additional types of browser data should be generated.
See Also	“ <code>CWBrowseOptions</code> ” on page 173 “ <code>reqCompile</code> ” on page 211

CWGetBuildSequenceNumber

Description	Returns a unique ID for the current Make operation.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWGetBuildSequenceNumber(CWPluginContext context, long* sequenceNumber);</pre>				
Parameters	The parameters for this method include: <table border="1"> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>sequenceNumber</td><td>Returns the unique ID of the current Make operation.</td></tr> </table>	context	Private, opaque IDE state.	sequenceNumber	Returns the unique ID of the current Make operation.
context	Private, opaque IDE state.				
sequenceNumber	Returns the unique ID of the current Make operation.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	The IDE assigns each Make operation a unique ID number. By retrieving this number with <code>CWGetBuildSequenceNumber</code> and storing it in persistent memory (allocated with <code>CWAllocateMemory</code>), a plug-in can determine if the current request and previous requests apply to the same Make operation.				
See Also	“ <code>CWGetTargetStorage</code> ” on page 154 “ <code>CWSetTargetStorage</code> ” on page 164 “ <code>CWAllocateMemory</code> ” on page 42				

CWGetCommandLineArgs

Description	Returns the text entered by the user in the Program Arguments text box of the Runtime Settings panel.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWGetCommandLineArgs (CWPluginContext context, const char** commandLineArgs);</pre>				
Parameters	The parameters for this method include: <table border="1"> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>commandLineArgs</td><td>Returns a pointer to a constant C string containing the command line parameters entered in the Program Arguments field of the Runtime Settings panel.</td></tr> </table>	context	Private, opaque IDE state.	commandLineArgs	Returns a pointer to a constant C string containing the command line parameters entered in the Program Arguments field of the Runtime Settings panel.
context	Private, opaque IDE state.				
commandLineArgs	Returns a pointer to a constant C string containing the command line parameters entered in the Program Arguments field of the Runtime Settings panel.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	This routine returns any command line options that the user has specified in the Program Arguments field of the Runtime Settings panel. These command line arguments apply to the execution of the final target executable.				
Mac OS	On the Mac OS, this method returns information only when the current linker enables the Runtime Settings panel. Currently, this happens only for the Java linker.				
See Also	“CWGetEnvironmentVariable” on page 143 “CWGetEnvironmentVariableCount” on page 144 “CWGetWorkingDirectory” on page 156				

CWGetEnvironmentVariable

Description Returns the value of an environment variable as configured in the **Environment Settings** list in the **Runtime Settings** panel.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetEnvironmentVariable(
    CWPluginContext context,
    long index,
    const char** name,
    const char** value);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
index	Specifies which environment variable to return, by index.
name	Returns a pointer to a constant C string containing the name of the runtime environment variable specified by index.
value	Returns a pointer to a constant C string containing the value of the runtime environment variable specified by index.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks The values returned control runtime settings established prior to launching the target executable, not the current IDE environment variable settings. linker plug-ins use CWGetEnvironmentVariable to set up the proper runtime environment before running targets.

To obtain all runtime environment variable settings, call CWGetEnvironmentVariable repeatedly with an index value ranging from 1 to the count returned by CWGetEnvironmentVariableCount.

Mac OS On the Mac OS, this method returns information only when the current linker enables the **Runtime Settings** panel. Currently, this happens only for the Java linker.

See Also “CWGetCommandLineArgs” on page 142

“CWGetEnvironmentVariableCount” on page 144
“CWGetWorkingDirectory” on page 156

CWGetEnvironmentVariableCount

Description	Returns the number of runtime environment variables configured in the Environment Settings list box of the Runtime Settings panel.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWGetEnvironmentVariableCount (CWPluginContext context, long* count);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>count</td><td>Returns the count of environment variable settings established in the Environment Settings list box of the Runtime Settings panel.</td></tr> </table>	context	Private, opaque IDE state.	count	Returns the count of environment variable settings established in the Environment Settings list box of the Runtime Settings panel.
context	Private, opaque IDE state.				
count	Returns the count of environment variable settings established in the Environment Settings list box of the Runtime Settings panel.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	Plug-ins use CWGetEnvironmentVariableCount to determine the number of environment variable settings the user has specified for the current target in its Runtime Settings panel. Given this count, the plug-in can process all the runtime environment variable settings, usually in preparation for launching a final executable. Typically, linkers obtain all environment variables and pass them to a helper application, which sets up the runtime environment for the target application prior to launch.				
Mac OS	On the Mac OS, this method returns information only when the current linker enables the Runtime Settings panel. Currently, this happens only for the Java linker.				
See Also	“CWGetCommandLineArgs” on page 142 “CWGetEnvironmentVariable” on page 143 “CWGetWorkingDirectory” on page 156				

CWGetMainFileID

Description Gets the ID of the file currently being processed in the active project target.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetMainFileID(
    CWPluginContext context,
    short* fileID);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
fileID	Returns the file ID of the file to process.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks The IDE assigns a unique ID to each file in a project. Plug-ins use these IDs when creating browser information to specify source files.

A compiler plug-in calls `CWGetMainFileID` in response to a `reqCompile` request, to determine the browser ID of the file currently being compiled. A compiler then uses this ID in browser information to specify the current file as the source file in which a symbol originates.

File IDs can also be useful for tracking project files concisely. File IDs persist between instances of starting and terminating the IDE and opening and closing a project. To determine the file ID of any project file or to determine which file has a given ID, use `CWGetFileInfo`.

See Also “`CWGetMainFileNumber`” on page 146
 “`CWGetMainFileSpec`” on page 147
 “`CWGetMainFileText`” on page 147
 “`CWGetFileInfo`” on page 55
 “`CWProjectFileInfo`” on page 109
 “`CWStoreObjectData`” on page 165

“reqCompile” on page 211

CWGetMainFileNumber

Description Gets the index of the file currently being processed.

Prototype `#include <DropInCompilerLinker.h>`
`CW_CALLBACK CWGetMainFileNumber(`
`CWPluginContext context,`
`long*            fileNumber);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
fileNumber	Returns the index of the file to process, in target link order.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks In response to a reqCompile, reqCompDisassemble, or reqCheckSyntax request, a plug-in determines the index of the file being compiled by calling CWGetMainFileNumber. A plug-in can use the value returned by CWGetMainFileNumber to call methods such as CWLoadObjectData and CWStoreObjectData.

See Also “CWGetMainFileID” on page 145
 “CWGetMainFileSpec” on page 147
 “CWGetMainFileText” on page 147
 “CWLoadObjectData” on page 161
 “CWStoreObjectData” on page 165
 “reqCompile” on page 211
 “reqCompDisassemble” on page 212
 “reqCheckSyntax” on page 211

CWGetMainFileSpec

Description	Gets the file specification of the file currently being processed.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWGetMainFileSpec(CWPluginContext context, CWFileSpec* fileSpec);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>fileSpec</td><td>Returns the file specification of the file currently being processed.</td></tr> </table>	context	Private, opaque IDE state.	fileSpec	Returns the file specification of the file currently being processed.
context	Private, opaque IDE state.				
fileSpec	Returns the file specification of the file currently being processed.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	During a reqCompile, reqCompDisassemble, or reqCheckSyntax request, a plug-in determines the location of the file on which to operate by calling CWGetMainFileSpec. A plugin can use the value returned by CWGetMainFileSpec to retrieve information about the file. A plug-in can also get the file’s contents by calling CWGetFileText with the value returned by CWGetMainFileSpec. To load the text of the file currently being compiled, plug-ins can instead call CWGetMainFileText.				
See Also	“CWGetMainFileID” on page 145 “CWGetMainFileNumber” on page 146 “CWGetMainFileText” on page 147 “reqCompile” on page 211 “reqCompDisassemble” on page 212 “reqCheckSyntax” on page 211				

CWGetMainFileText

Description	Loads the contents of the file currently being processed.
Prototype	#include <DropInCompilerLinker.h>

Compiler and Linker Plug-in Reference

CWGetModifiedFiles

```
CW_CALLBACK CWGetMainFileText (
    CWPluginContext context,
    const char** text,
    long* textLength);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
text	Returns a pointer to memory containing the file's contents.
textLength	Returns the size in bytes of the file's contents.

Return An error code listed in "Result Codes for Compiler and Linker Plug-ins" on page 227 or "Result Codes for Plug-ins" on page 129

Remarks In response to a reqCompile, reqCompDisassemble, or reqCheckSyntax request, a plug-in retrieves the contents of the file on which to operate by calling CWGetMainFileText.

CAUTION Plug-ins should *not* release the text pointer returned by CWGetMainFileText.

NOTE CWGetFileText returns text that has been declared as constant. Plug-ins should not modify this text data.

See Also "CWGetMainFileID" on page 145
 "CWGetMainFileNumber" on page 146
 "CWGetMainFileSpec" on page 147
 "reqCompile" on page 211
 "reqCompDisassemble" on page 212
 "reqCheckSyntax" on page 211

CWGetModifiedFiles

Description Gets the count of and a list of source files that have been modified since the last time they were compiled.

Prototype `#include "DropinCompilerLinker.h"`
 `CW_CALLBACK CWGetModifiedFiles(`
 `CWPluginContext context,`
 `long* modifiedFileCount,`
 `const long** modifiedFiles);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
modifiedFileCount	Returns the count of modified files.
modifiedFiles	Returns an array listing the modified files by index.

Remarks This method exists to support Java but may be useful for other languages. It returns a list of modified or “touched” files, which require recompilation.

 This method gets a constant array of long integer indexes that refer to other files in the current target by index. Typically, plug-ins use the indexes returned in `modifiedFiles` to call `CWGetFileInfo`.

 The IDE allocates `modifiedFiles` on the plug-in’s behalf. The plug-in should not release this data (it will be released by the IDE at the end of the pending compilation).

NOTE The IDE creates the list of files to be returned only once during a plug-in request. Subsequent calls to `CWGetModifiedFiles` during the same request get the same list.

See Also “CWGetFileInfo” on page 55

CWGetPrecompiledHeaderSpec

Description Returns the recommended file specification for saving a precompiled header.

Prototype `#include <DropInCompilerLinker.h>`
 `CW_CALLBACK CWGetPrecompiledHeaderSpec(`
 `CWPluginContext context,`
 `CWFileSpec* pchspec,`
 `const char* target);`

Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

CWGetResourceFile

Parameters The parameters for this method include:

context	Private, opaque IDE state.
pchspec	Returns a file specification for a precompiled header file.
target	Specifies the desired name of the precompiled header file.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks Plug-ins should call CWGetPrecompiledHeaderSpec to determine where to save a precompiled header file. If target is NULL, the IDE prompts the user for a file specification and returns it in pchspec. Otherwise, the IDE creates a file specification using target and returns it in pchspec.

Plug-ins normally call CWGetPrecompiledHeaderSpec in response to a reqCompile request for which one of CWIsPrecompiling or CWIsAutoPrecompiling returns true.

See Also “Precompiling” in the *IDE SDK Developer’s Guide*

“CWIsPrecompiling” on page 159

“CWIsAutoPrecompiling” on page 157

CWGetResourceFile

Description Prompts the user to select an existing file containing resources.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetResourceFile(
    CWPluginContext context,
    CWFileSpec* filespec);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
filespec	Returns the file specification of the resource file chosen by the user.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks CWGetResourceFile displays a standard file selection dialog box showing resource files and waits for the user to select a file. If the user cancels, CWGetResourceFile returns `cwErrUserCanceled`.

CWGetResourceFile is currently used by the Mac OS 68K and PowerPC linkers when compiling code resources for targets with both the **Display Dialog** and **Merge to File** checkboxes set. However, other resource compilers can also use it.

See Also “CWPutResourceFile” on page 162

CWGetStoredObjectFileSpec

Description Gets the file specification most recently stored with a file in a call to `CWStoreObjectData`.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetStoredObjectFileSpec(
    CWPluginContext context,
    long whichfile,
    CWFileSpec* fileSpec);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the file for which to obtain a <code>CWFileSpec</code> , by index in target link order.
fileSpec	Returns the <code>CWFileSpec</code> associated with a file, if any was specified when a plug-in last called <code>CWStoreObjectData</code> .

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks `CWGetStoredObjectFileSpec` gets the `CWFileSpec` for any externally stored object data file associated with a file in the current project target. `whichfile` specifies the file for which to get the external object file specification, by index, in target link order.

Compiler and Linker Plug-in Reference

CWGetSuggestedObjectFileSpec

On return, `fileSpec` contains the file specification most recently stored by a plug-in when calling `CWStoreObjectData`. If the most recent call to `CWStoreObjectData` specified no file specification, `CWGetSuggestedObjectFileSpec` returns an error: `cwErrObjectFileNotStored`.

The IDE determines whether a file specification has been associated with the file by examining the `objectdata` field of the `CWObjectData` structure. If `objectdata` is `NULL`, the IDE assumes the `objectfile` file specification should be ignored.

- See Also
- “`CWStoreObjectData`” on page 165
 - “`CWObjectData`” on page 180
 - “`CWFileSpec`” on page 98
 - “`cwErrObjectFileNotStored`” on page 227

CWGetSuggestedObjectFileSpec

Description Gets the IDE’s preferred location for externally stored object files.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetSuggestedObjectFileSpec(
    CWPluginContext context,
    long whichfile,
    CWFileSpec* fileSpec);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the file for which to return a suggested file specification.
fileSpec	Returns the location of the file in which the IDE wants external object files stored.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks Plug-ins that need to store object data in files outside of the project should call `CWGetSuggestedObjectFileSpec` to determine where to store the object code files. Doing so ensures that the IDE

can properly manage the object code, including the following operations:

- Rebuilding the object code when it is missing
- Loading the object code when the plug-in calls `CWLoadObjectData`
- Removing the object code prior to recompilation or when the user selects **Remove Object Code** from the **Project** menu.

The returned file specification specifies the location of the directory in which the plug-in should store object files, not the name of an object code file. The plug-in can use any desired file name within this folder.

Compiler plug-ins often call `CWGetSuggestedObjectFileSpec` as part of the following tasks:

- Determine where to put output files.
- Tell command-line tools where to put output files.
- Specify a location in the `objectfile` field when calling `CWStoreObjectData`.

Linker plug-ins can determine the location and name of the object code file later by calling `CWGetStoredObjectFileSpec`.

See Also "CWLoadObjectData" on page 161

"CWStoreObjectData" on page 165

"CWGetStoredObjectFileSpec" on page 151

CWGetTargetInfo

Description Gets information about the current target.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWGetTargetInfo(
    CWPluginContext context,
    CWTargetInfo* targetInfo);
```

Compiler and Linker Plug-in Reference

CWGetTargetStorage

Parameters	The parameters for this method include:				
	<table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>targetInfo</td><td>Returns information about the current target.</td></tr> </table>	context	Private, opaque IDE state.	targetInfo	Returns information about the current target.
context	Private, opaque IDE state.				
targetInfo	Returns information about the current target.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129.				
Remarks	CWGetTargetInfo gets information about the current link target, including the target processor and OS, the final binary type, and target debugging information. Most of this information is determined by the linker currently selected in the Target Settings panel. The linker returns this information when handling the reqTargetInfo request, by calling CWSetTargetInfo. The IDE retains the returned information with the current target.				
NOTE	Post-linkers can also override target information.				
See Also	“Providing Target Information” in the <i>IDE SDK Developer’s Guide</i> “CWTargetInfo” on page 183 “CWSetTargetInfo” on page 163				

CWGetTargetStorage

Description	Retrieves global data for the active project target.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWGetTargetStorage(CWPluginContext context, void** storage);</pre>				
Parameters	The parameters for this method include:				
	<table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>storage</td><td>Returns a pointer to global storage retained for the plug-in with the current target by the IDE.</td></tr> </table>	context	Private, opaque IDE state.	storage	Returns a pointer to global storage retained for the plug-in with the current target by the IDE.
context	Private, opaque IDE state.				
storage	Returns a pointer to global storage retained for the plug-in with the current target by the IDE.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129.				

Remarks

Use CWGetTargetStorage to retrieve target data previously stored by using CWSetTargetStorage. Plug-ins use CWSetTargetStorage and CWGetTargetStorage to maintain global data with the active target. Target storage is maintained in memory for the duration of one invocation of the IDE but is not stored on disk.

NOTE

To associate data persistently with individual files in a project, use CWGetPluginData and CWStorePluginData.

Plug-ins can use CWGetTargetStorage to cache data for symbol unmangling requests. Since compilation and unmangling occur as separate requests, data must be stored globally to support unmangling.

A plug-in normally calls CWGetTargetStorage and CWSetTargetStorage when it receives a reqTargetLoaded or reqTargetUnloaded request. Plug-ins should allocate persistent target storage by calling CWAllocateMemory with the isPermanent parameter set to true. Plug-ins should release target memory by calling CWFreeMemory.

The IDE only provides target storage to plug-ins that set the kCompUsesTargetStorage or linkerUsesTargetStorage flags in their dropin flags. Target storage services fail if the appropriate flag is not set.

See Also

“Specifying Plug-in Capabilities” in the *IDE SDK Developer’s Guide*

“Compiler and Linker Drop-In Flags” in the *IDE SDK Developer’s Guide*

“CWSetTargetStorage” on page 164

“reqTargetLoaded” on page 219

“reqTargetUnloaded” on page 220

“CWGetPluginData” on page 65

“CWStorePluginData” on page 81

“CWAllocateMemory” on page 42

“CWFreeMemory” on page 48

CWGetWorkingDirectory

Description	Gets the location specified by the user in the Working Directory field of the Runtime Settings panel.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWGetWorkingDirectory(CWPluginContext context, CWFileSpec* workingDirectorySpec);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>workingDirectorySpec</td><td>Returns the user-specified location entered in the Working Directory field of the Runtime Settings panel.</td></tr> </table>	context	Private, opaque IDE state.	workingDirectorySpec	Returns the user-specified location entered in the Working Directory field of the Runtime Settings panel.
context	Private, opaque IDE state.				
workingDirectorySpec	Returns the user-specified location entered in the Working Directory field of the Runtime Settings panel.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	Linker plug-ins should call CWGetWorkingDirectory to determine the preferred working directory prior to launching a target executable. CWGetWorkingDirectory returns the location specified in the Working Directory field of the Runtime Settings panel.				
Mac OS	On the Mac OS, this method returns information only when the current linker enables the Runtime Settings panel. Currently, this happens only for the Java linker.				
See Also	“CWGetCommandLineArgs” on page 142 “CWGetEnvironmentVariable” on page 143 “CWGetEnvironmentVariableCount” on page 144				

CWIsAutoPrecompiling

Description	Indicates whether the current precompile operation was initiated by a build operation.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWIsAutoPrecompiling(CWPluginContext context, Boolean* isAutoPrecompiling);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>isAutoPrecompiling</td><td>Returns true if the file currently being compiled should be precompiled and if the compilation request resulted from a build operation.</td></tr> </table>	context	Private, opaque IDE state.	isAutoPrecompiling	Returns true if the file currently being compiled should be precompiled and if the compilation request resulted from a build operation.
context	Private, opaque IDE state.				
isAutoPrecompiling	Returns true if the file currently being compiled should be precompiled and if the compilation request resulted from a build operation.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	When responding to a reqCompile request, a compiler plug-in should call CWIsAutoPrecompiling. If CWIsAutoPrecompiling returns true, the plug-in should precompile the current source text obtained by calling CWGetMainFileText, or by loading the text of the file indicated by CWGetMainFileNumber. The resulting precompiled header should be stored in the location specified by CWGetPrecompiledHeaderSpec. CWIsAutoPrecompiling returns true when the user chooses Make or Bring Up To Date from the Project menu and a precompiled interface file must be updated. The IDE automatically precompiles only those file types for which the ‘precompiled’ flag is set in Project Settings > File Mappings.				
NOTE	CWIsAutoPrecompiling returns true <i>only</i> when precompiling interface files encountered during an automated build operation, rather than when manually precompiling a file.				
See Also	“Compiling” in the <i>IDE SDK Developer’s Guide</i>				

Compiler and Linker Plug-in Reference

CWIsCachingPrecompiledHeaders

- “reqCompile” on page 211
- “CWGetPrecompiledHeaderSpec” on page 149
- “CWIsPrecompiling” on page 159
- “CWGetMainFileID” on page 145
- “CWGetMainFileNumber” on page 146
- “CWGetMainFileSpec” on page 147
- “CWGetMainFileText” on page 147

CWIsCachingPrecompiledHeaders

Description	Indicates whether the IDE supports precompiled header caching.				
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWIsCachingPrecompiledHeaders (CWPluginContext context, Boolean* isCaching);</pre>				
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>isCaching</td><td>Returns true if the IDE can accept requests to cache precompiled headers.</td></tr> </table>	context	Private, opaque IDE state.	isCaching	Returns true if the IDE can accept requests to cache precompiled headers.
context	Private, opaque IDE state.				
isCaching	Returns true if the IDE can accept requests to cache precompiled headers.				
Return	An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129				
Remarks	Before calling <code>CWCachePrecompiledHeader</code> , a plug-in should call <code>CWIsCachingPrecompiledHeaders</code> to make sure the IDE is able to cache precompiled header data.				
See Also	“CWCachePrecompiledHeader” on page 138				

CWIsGeneratingDebugInfo

Description	Indicates whether a plug-in should generate debugging information.
Prototype	<pre>#include <DropInCompilerLinker.h></pre>

```
CW_CALLBACK CWIsGeneratingDebugInfo (
    CWPluginContext context,
    Boolean* isGenerating);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
isGenerating	Returns true if the plug-in should generate debugging information.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks Compiler plug-ins responding to reqCompile requests should call CWIsGeneratingDebugInfo to determine if the plug-in should generate debugging information. CWIsGeneratingDebugInfo returns true if the debugging flag (indicated by a bug icon next to a file’s name in the project window) is enabled for the file being compiled.

See Also “Generating Debugging Data” in the *IDE SDK Developer’s Guide*
“reqCompile” on page 211
“reqLink” on page 214

CWIsPrecompiling

Description Indicates whether a plug-in should precompile the current file.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWIsPrecompiling (
    CWPluginContext context,
    Boolean* isPrecompiling);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
isPrecompiling	Returns true if the file currently being compiled should be precompiled.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Compiler and Linker Plug-in Reference

CWIsPreprocessing

Remarks When responding to a reqCompile request, a compiler plug-in should call CWIsPrecompiling. If CWIsPrecompiling returns true, the plug-in should precompile the source code specified by CWGetMainFileText and store it in the file returned by CWGetPrecompiledHeaderSpec.

CWIsPrecompiling returns true when the user chooses **Precompile** from the **Project** menu or when the IDE automatically recompiles a precompiled header file. In the latter case, CWIsAutoPrecompiling also returns true.

See Also “Compiling” in the *IDE SDK Developer’s Guide*

“CWIsAutoPrecompiling” on page 157

“CWGetPrecompiledHeaderSpec” on page 149

“CWGetMainFileText” on page 147

“reqCompile” on page 211

CWIsPreprocessing

Description Indicates whether a plug-in should preprocess the current file.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWIsPreprocessing(
    CWPluginContext context,
    Boolean* isPreprocessing);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
isPreprocessing	Returns true if the file currently being compiled should be preprocessed.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks When responding to a reqCompile request, a compiler plug-in should call CWIsPreprocessing to determine if the source code obtained by calling CWGetMainFileText should be preprocessed.

Preprocessing usually involves textual substitution or source transformation. Plug-ins usually display the result of preprocessing in a window by calling `CWCreateNewTextDocument`.

- See Also
- “Compiling” in the *IDE SDK Developer’s Guide*
 - “CWGetMainFileText” on page 147
 - “CWGetMainFileSpec” on page 147
 - “CWIsPrecompiling” on page 159
 - “CWCreateNewTextDocument” on page 44
 - “reqCompile” on page 211

CWLoadObjectData

Description Retrieves object data stored with a file.

Prototype `#include <DropInCompilerLinker.h>`
`CW_CALLBACK CWLoadObjectData(`
 `CWPluginContext context,`
 `long whichfile,`
 `CWMemHandle* objectdata);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the index of the file for which to load object data, in target link order.
objectdata	Returns a <code>CWMemHandle</code> containing the object data.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks To retrieve object data associated with a file in the active target, use `CWLoadObjectData`. `CWLoadObjectData` returns the object data most recently stored for `whichfile` in a call to `CWStoreObjectData`.

The data returned in `objectdata` can be object code, resource data, or some combination of both. The IDE has no knowledge of

Compiler and Linker Plug-in Reference

CWPutResourceFile

and makes no assumptions about the content of the `objectdata` handle. The format of the data is determined by agreement between compiler and linker.

If a plug-in stores object data in a file external to the project, `CWLoadObjectData` loads the entire contents of the file into a `CWMemHandle` and returns it to the plug-in. A plug-in can store object data in an external file by specifying a value for the `objectfile` field of a `CWObjectData` structure when calling `CWStoreObjectData`,

Before accessing the data returned by `CWLoadObjectData`, lock it using `CWLockMemHandle`. When finished using the data, release it by calling `CWFreeObjectData`.

To specify the first file in the active target, use 0. To specify the last file, use the value returned by `CWGetProjectFileCount - 1`.

See Also “`CWStoreObjectData`” on page 165
 “`CWObjectData`” on page 180
 “`CWLockMemHandle`” on page 70
 “`CWFreeObjectData`” on page 140
 “`CWGetProjectFileCount`” on page 67

CWPutResourceFile

Description Displays a file save dialog box to prompt the user to select a file in which to store resources.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWPutResourceFile(
    CWPluginContext context,
    const char*      prompt,
    const char*      name,
    CWFileSpec*      filespec);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
prompt	A C string prompt.
name	A C string specifying the default name of the resource file.
filespec	Returns a file specification for the resource file chosen by the user.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks CWPutResourceFile displays a standard file dialog box and waits for the user to specify the name and location of a new resource file. If the user cancels, CWPutResourceFile returns `cwErrUserCanceled`.

The Mac OS 68K and PowerPC linkers use CWPutResourceFile when compiling code resources for targets with the **Display Dialog** checkbox set and the **Merge to File** checkbox cleared. However, other resource compilers can also use this method.

See Also “CWGetResourceFile” on page 150

CWSetTargetInfo

Description Sets properties of the current target.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWSetTargetInfo(
    CWPluginContext context,
    CWTargetInfo* targetInfo);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
targetInfo	Specifies properties of the current target.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks In response to a `reqTargetInfo` request, a linker plug-in should call CWSetTargetInfo. CWSetTargetInfo specifies many



properties of the current link target, including target processor and OS, the final binary type, and target debugging information. Plug-ins can obtain the current target settings using CWGetTargetInfo.

See Also “CWSetTargetInfo” on page 163
“CWGetTargetInfo” on page 153
“CWTargetInfo” on page 183
“reqTargetInfo” on page 218

CWSetTargetStorage

Description Stores global data with the current target.

Prototype

```
#include <DropInCompilerLinker.h>
CW_CALLBACK CWSetTargetInfo(
    CWPluginContext context,
    void* storage);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
storage	Specifies a pointer to global data to be retained by the IDE with the current target for the current plug-in.

Return An error code listed in “Result Codes for Compiler and Linker Plug-ins” on page 227 or “Result Codes for Plug-ins” on page 129

Remarks Use CWSetTargetStorage to store global data with the current target. Plug-ins use CWGetTargetStorage and CWSetTargetStorage to maintain global data with the active target. Target storage is maintained in memory for the duration of one invocation of the IDE but is not stored on disk.

NOTE To associate data persistently with individual files in a project, use CWGetPluginData and CWStorePluginData.

Plug-ins use CWSetTargetStorage to cache data for symbol unmangling requests. Since compilation and unmangling occur as

separate requests, data must be stored globally to support unmangling.

A plug-in normally calls CWGetTargetStorage and CWSetTargetStorage when it receives a reqTargetLoaded or reqTargetUnloaded request. Target storage should be allocated persistently by calling CWAllocateMemory with its isPermanent parameter set to true and released by calling CWFreeMemory.

The IDE only provides target storage to plug-ins that set the kCompUsesTargetStorage or linkerUsesTargetStorage flags in their dropin flags. Target storage services fail if the appropriate flag is not set.

- See Also

“Specifying Plug-In Capabilities” in the *IDE SDK Developer’s Guide*

“Compiler and Linker Drop-In Flags” in the *IDE SDK Developer’s Guide*

“CWGetTargetStorage” on page 154

“reqTargetLoaded” on page 219

“reqTargetUnloaded” on page 220

“CWGetPluginData” on page 65

“CWStorePluginData” on page 81

“CWAllocateMemory” on page 42

“CWFreeMemory” on page 48

CWStoreObjectData

Description	Stores object, resource, and browser data with a file in the current target and, optionally, specifies source dependencies.
Prototype	<pre>#include <DropInCompilerLinker.h> CW_CALLBACK CWStoreObjectData(CWPluginContext context, long whichfile, CWOBJECTDATA* objectdata);</pre>

Compiler and Linker Plug-in Reference

User Routines for Compiler and Linker Plug-ins

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichfile	Specifies the file for which the IDE should store object data and dependencies, by index in target link order.
objectdata	Specifies numerous properties of the object data to be stored, including the object code, browser data, and optionally dependencies.

Return An error code listed in "Result Codes for Compiler and Linker Plug-ins" on page 227 or "Result Codes for Plug-ins" on page 129

Remarks CWStoreObjectData stores data passed in objectdata with the current project target and associates it with the file in the whichfile parameter, specified by index.

Compilers normally call CWStoreObjectData when done responding to a reqCompile request, to save both object code and browser data.

CWStoreObjectData can also be used to establish dependencies among arbitrary files in a project. Normally, plug-ins establish file dependencies with CWFindAndLoadFile. When using CWFindAndLoadFile is inconvenient (such as when adapting command line tools), a plug-in can store a list of dependencies by using CWStoreObjectData.

See Also "CWOBJData" on page 180

User Routines for Compiler and Linker Plug-ins

In addition to the standard plug-in entry points, compiler and linker plug-ins provide two additional entry points for specifying associated file types, and for providing target information to the IDE. They may optionally export additional symbol unmangling and browser symbol entry points.

For more about plug-in entry points, including compiler- and linker-specific entry points, see "Compiler- and Linker-Specific Entry Points," "Informational Entry Points," and "Responding to IDE Requests" in the *IDE SDK Developer's Guide*.

The following entry points are specific to compilers and linkers:

- CWPlugin_GetDefaultMappingList Entry Point
- CWPlugin_GetTargetList Entry Point
- Helper_GetCompilerBrSymbols Entry Point
- Helper_Unmangle Entry Point

CWPlugin_GetDefaultMappingList Entry Point

Description	Provides an informational entry point that the IDE uses to determine the file types to associate with a plug-in.	
Prototype	CWPLUGIN_ENTRY (CWPlugin_GetDefaultMappingList) (const CWExtMapList** defaultMappingList)	
Parameters	The parameters for this entry point include:	
	defaultMappingList	Returns a CWExtMapList to the IDE containing an array of CWExtensionMapping structures specifying file type, extension, and flags.
Return	Usually, the plug-in should return the cwNoErr result code. See “Result Codes for Plug-ins” on page 129.	
Remarks	The IDE uses this entry point to obtain the default list of file mappings to associate with a plug-in. compiler and linker plug-ins must implement this entry point. Usually, plug-ins implement this entry point by returning a pointer to a constant CWExtMapList structure. File mappings appear in the File Mappings panel of the Project Settings dialog box. File mappings determine which types of files the IDE presents to a plug-in for compiling or linking. File mappings also control which files may be added to a project. The IDE adds file mappings to a project when it creates a new project completely from scratch (not from stationery) and when the user reverts to factory defaults.	
Mac OS	On Mac OS, this routine is optional. In its place, a plug-in may provide an 'EMap' resource, specifying the same information.	

Compiler and Linker Plug-in Reference

CWPlugin_GetTargetList Entry Point

See Also “Specifying File Mappings” in the *IDE SDK Developer’s Guide*

CWPlugin_GetTargetList Entry Point

Description Provides an informational entry point that the IDE uses to determine the target CPUs and operating systems supported by a plug-in.

Prototype CWPLUGIN_ENTRY (CWPlugin_GetTargetList)
(const CWTargetList** targetList)

Parameters The parameters for this entry point include:

targetList	Returns a CWTargetList to the IDE containing two arrays of 4-character codes listing the CPUs and operating systems supported by a plug-in.
------------	---

Return Usually, the plug-in should return the `cwNoErr` result code. See “Result Codes for Plug-ins” on page 129.

Remarks The IDE uses this entry point to obtain a list of the operating systems and CPUs supported by a plug-in. Compiler and linker plug-ins must implement this entry point. Usually, plug-ins implement this entry point returning a pointer to a constant CWTargetList structure.

The IDE uses this list to determine which plug-ins are enabled. When the user chooses a linker from the Linker popup in the **Target Settings** panel of the **Project Settings** window, the linker’s supported CPU and OS are used to find other matching compiler plug-ins, which are then enabled for use in the **File Mappings** panel. Also, the IDE adds the preference panels for all enabled plug-ins to the **Project Settings** dialog box.

Mac OS On the Mac OS, this routine is optional. In its place, a plug-in may provide a 'Targ' resource to specify the same information.

See Also “Specifying Target Platforms” in the *IDE SDK Developer’s Guide*

Helper_GetCompilerBrSymbols Entry Point

Description Provides an informational entry point that the IDE uses to obtain names of custom browser symbol types generated by a compiler plug-in.



Prototype `#include <DropinCompilerLinker.h>`
 `typedef CWPLUGIN_ENTRY`
 `(*Helper_GetCompilerBrSymbols)`
 `(CWCompilerBrSymbolInfo* BrowseSymbolInfo);`

Parameters The parameters for this entry point include:

BrowseSymbolInfo	Returns a list of browser symbol type names to the IDE.
------------------	---

Return An error value described in “Result Codes for Plug-ins” on page 129

Remarks The IDE calls a compiler’s browser symbol entry point to obtain the human-readable, localized names of custom browser symbol types from the plug-in. The IDE passes this entry point a structure containing the compiler’s target storage (if any has been allocated) and a field in which the plug-in returns a pointer to a list. The list associates internal and localized type names with the custom symbol type codes generated by the compiler.

The IDE only calls this entry point if a compiler sets the `kCompEmitsOwnBrSymbols` dropin flag. The IDE calls a compiler’s browser symbol entry point if the file mappings for a project enable the compiler, regardless of whether any files associated with the compiler appear in the active project. Only compiler plug-ins need to implement this entry point.

Compilers that generate custom browser symbols assign numeric values to symbols to indicate symbol type. The values must be assigned from the range reserved for plug-ins, starting with `browseCompSymbolStart` and increasing sequentially. The strings returned by this entry point provide type names for the custom symbol types (not individual symbols, which appear in the browse data).

For each custom symbol type, the `CWCompilerBrSymbolInfo` structure returned by this entry point provides two pieces of information:

- `symName`: The symbol types’s internal name, used to group symbols having functionally equivalent type. Plug-ins should not localize this name.

Compiler and Linker Plug-in Reference

Helper_GetCompilerBrSymbols Entry Point

- `symUIName`: The symbol type's localized, user-visible name, to be displayed in browser windows and used when selecting the type of symbol to view (function, global variable, class, etc.).

The IDE returns this information in an array of `CWCompilerBrSymbol` structures, with one entry for each custom type used by the plug-in when generating browser data. The first array entry corresponds to the first symbol type (`browseCompSymbolStart`). The second entry corresponds to the second symbol type (`browseCompSymbolStart + 1`). Subsequent entries continue to increment this value.

NOTE The IDE treats the `CWCompilerBrSymbolList` structure returned by reference as constant data.

If multiple compilers generate symbols with the same type name (`symName`), the IDE groups the symbols under the same heading in the IDE's browser catalog window.

Mac OS 68K-based non-CFM plug-ins implement this entry point by providing a code resource of type 'Dhlp', which must be named "Helper_GetCompilerBrSymbols". Plug-ins must declare the `main()` entry point for the code resource as shown above.

See also "Compiler Browser Symbol Entry Point" in the *IDE SDK Developer's Guide*

"Compiler and Linker Drop-In Flags" in the *IDE SDK Developer's Guide*

"'Dhlp' Resource" in the *IDE SDK Developer's Guide*

"'Flag' Resource" in the *IDE SDK Developer's Guide*

"CWCompilerBrSymbolInfo" on page 175

"CWCompilerBrSymbolList" on page 175

"CWCompilerBrSymbol" on page 174

"EBrowserItem" on page 246

Helper_Unmangle Entry Point

Description	Provides an informational entry point that the IDE uses to unmangle browser symbols generated by linker plug-ins.		
Prototype	<pre>#include <DropinCompilerLinker.h> typedef CWPLUGIN_ENTRY (*Helper_Unmangle) (CWUnmangleInfo*);</pre>		
Parameters	The parameters for this entry point include:<table><tr><td>CWUnmangleInfo</td><td>Provides information about the symbol to be unmangled and buffers for returning unmangled symbol names to the IDE.</td></tr></table>	CWUnmangleInfo	Provides information about the symbol to be unmangled and buffers for returning unmangled symbol names to the IDE.
CWUnmangleInfo	Provides information about the symbol to be unmangled and buffers for returning unmangled symbol names to the IDE.		
Return	An error value described in “Result Codes for Plug-ins” on page 129.		
Remarks	The IDE calls a linker’s symbol unmangling entry point to convert a mangled identifier name emitted in browser data into its human-readable form. The IDE calls this entry point only if the plug-in sets the <code>linkerUnmangles</code> flag in its dropin flags. Only linker plug-ins need to implement this entry point.		

The `CWUnmangleInfo` structure passed to the plug-in specifies the following:

- 1. `targetStorage`: Contains the plug-in’s target storage, often used to maintain additional information that is useful when unmangling symbols. The linker plug-in should allocate this storage in response to a previous `reqTargetLoaded` request. The linker plug-in should save information relevant to unmangling in target storage during compilation and linking.
- 2. `mangledName`: Provides the mangled symbol name.
- 3. `unmangleBuff`: Provides storage in which to return an unmangled name,
- 4. `unmangleBuffSize`: Specifies the size of the unmangled name to return, including the `NULL` terminator character. Do not exceed this size when unmangling.
- 5. `browserClassID`: Indicates the type of the symbol whose name is to be unmangled.

Compiler and Linker Plug-in Reference

Data Structures for Compiler and Linker Plug-ins

6. `browserLang`: The plug-in should return an `ELanguage` language code in this field, specifying the symbol source language.

The IDE calls this entry point to unmangle browser symbols when displaying browser windows.

Mac OS 68K-based non-CFM plug-ins implement this entry point by providing a code resource of type 'Dhlp', which must be named "Helper_Unmangle". Plug-ins must declare the `main()` entry point for the code resource as shown above.

See also "'Dhlp' Resource" in the *IDE SDK Developer's Guide*

"CWUnmangleInfo" on page 190

"reqTargetLoaded" on page 219

"ELanguage" on page 247

Data Structures for Compiler and Linker Plug-ins

This section describes the plug-in API data structures available for compilers, linkers, pre-linkers, and post-linkers.

The data structures for compilers and linkers are:

- `CWBrowseOptions`
- `CWCompilerBrSymbolInfo`
- `CWCompilerBrSymbolList`
- `CWDependencyInfo`
- `CWCompilerBrSymbol`
- `CWExtensionMapping`
- `CWExtMapList`
- `CWObjectData`
- `CWTargetInfo`
- `CWTargetList`
- `CWUnmangleInfo`

CWBrowseOptions

Description Specifies what browser information compilers should generate.

Definition

```
#include <DropInCompilerLinker.h>
typedef struct CWBrowseOptions {
    Boolean    recordClasses;
    Boolean    recordEnums;
    Boolean    recordMacros;
    Boolean    recordTypedefs;
    Boolean    recordConstants;
    Boolean    recordTemplates;
    Boolean    recordUndefinedFunctions;
    long       reserved1;
    long       reserved2;
} CWBrowseOptions;
```

Fields The fields in CWBrowseOptions include:

recordClasses	If true, classes and structured types should be recorded. Examples of classes are C++ classes and Pascal classes. Examples of structured type classes are C structs and unions and Pascal records.
recordEnums	If true, enumerated types should be recorded.
recordMacros	If true, macros should be recorded.
recordTypedefs	If true, user-defined types other than those recorded for recordClasses should be recorded.
recordConstants	If true, constant definitions should be recorded.
recordTemplates	If true, templates should be recorded.
recordUndefinedFunctions	
	If true, routines that are declared but not defined should be recorded.
reserved1, reserved2	These fields are reserved by the IDE. Do not modify their contents.

Compiler and Linker Plug-in Reference

CWCompilerBrSymbol

Remarks This structure specifies the source code elements for which a compiler should generate browser information. Plug-ins can get this structure by calling `CWGetBrowseOptions`.

NOTE The IDE assumes that compilers always record browser information for global variables and functions, in addition to any item types enabled by `CWGetBrowseOptions`. Thus, no flags for functions and global variables exist.

See Also “`CWGetBrowseOptions`” on page 140

CWCompilerBrSymbol

Description Describes a compiler-specific browser symbol type.

Definition

```
#include <DropInCompilerLinker.h>
typedef struct CWCompilerBrSymbol {
    char symName[32];
    char symUIName[32];
} CWCompilerBrSymbol;
```

Fields The fields in `CWBrowserBrSymbol` include:

symName	Specifies an internal name for the browser symbol type. This name should not be localized.
symUIName	Specifies the localized, human-readable name of the browser symbol type. The IDE presents this symbol to the user.

Remarks This data structure describes a compiler-specific browser symbol type. It appears in a `CWCompilerBrSymbolList` data structure. Compiler plug-ins return the `CWCompilerBrSymbolList` structure to the IDE through the `Helper_GetCompilerBrSymbols` Entry Point.

The IDE groups symbols in the browser window by `symName`. If multiple compilers generate symbols with the same `symName`, the symbols appear under the same heading in the IDE’s browser catalog window.

See Also “`CWCompilerBrSymbolList`” on page 175

“CWCompilerBrSymbolInfo” on page 175

“Helper_GetCompilerBrSymbols Entry Point” on page 168

CWCompilerBrSymbolInfo

Description	Gets the names of custom browser symbol types for the IDE.	
Definition	<pre>#include <DropInCompilerLinker.h> typedef struct CWCompilerBrSymbolInfo { void* targetStorage; CWCompilerBrSymbolList* symList; } CWCompilerBrSymbolInfo;</pre>	
Fields	The fields in CWBrowserBrSymbolInfo include:	
	targetStorage	Points to the plug-in’s target data, if it’s available. This should have been set up during a previous reqTargetLoaded request.
	symList	Returns a pointer to a list of browser symbol type names.
Remarks	The IDE passes this data structure to a compiler’s Helper_GetCompilerBrSymbols Entry Point. The plug-in is expected to return a pointer to a constant list of custom symbol type names, in symList. The IDE passes the plug-in’s target storage in targetStorage (but this is usually not required to return symbol type names).	
See Also	“Helper_GetCompilerBrSymbols Entry Point” on page 168	

CWCompilerBrSymbolList

Description	Contains a list of compiler-specific browser symbols.
Definition	<pre>#include <DropInCompilerLinker.h> typedef struct CWCompilerBrSymbolList { short count; CWCompilerBrSymbol items[1]; } CWCompilerBrSymbolList;</pre>

Compiler and Linker Plug-in Reference

CWDependencyInfo

Fields The fields in CWBrowserBrSymbolList include:

count	Specifies the number of symbols in the list.
items	Points to an array of browser symbol type names.

Remarks This data structure contains an array of compiler browser symbol type name structures. Each CWCompilerBrSymbol element in the array specifies the internal and user-visible names for a single custom type used by a compiler when generating browser data.

See Also “CWCompilerBrSymbol” on page 174
“Helper_GetCompilerBrSymbols Entry Point” on page 168

CWDependencyInfo

Description Specifies a file depended upon by the file for which object code is being stored in a call to CWStoreObjectData.

Definition

```
#include <DropInCompilerLinker.h>
typedef struct CWDependencyInfo {
    long      fileIndex;
    CWFileSpec fileSpec;
    short     fileSpecAccessType;
    short     dependencyType;
} CWDependencyInfo;
```

Fields The fields in CWDependencyInfo include:

fileIndex	Specifies a file depended upon by the file whose object data is being stored, by index in target link order. Use -1 to specify the depended-upon file using fileSpec instead.
fileSpec	When fileIndex is -1, this field specifies the depended-upon file by CWFileSpec instead of by index.

fileSpecAccessType	Specifies how the IDE should find the file after the user has chosen a command like Re-search for Files in the Project menu. Valid values are: <code>cwAccessPathRelative</code> , <code>cwAccessAbsolute</code> , <code>cwAccessFileName</code> , and <code>cwAccessFileRelative</code> . Ignored unless <code>fileIndex</code> is -1.
dependencyType	A <code>CWDependencyType</code> value that specifies how the dependent file depends on the depended-upon file.

Remarks A plug-in compiler or linker uses the `CWDependencyInfo` structure to store dependency information for a file when calling `CWStoreObjectData`. The `CWObjectData` structure's `dependencies` field points to an array of elements of `CWDependencyInfo` type, each of which specifies one file depended upon by the file whose object code is being stored.

To specify that the object data depends on the first file in the target, store 0 in `fileIndex`. To specify the last file, store `CWGetProjectFileCount(...) - 1`. Store -1 in `fileIndex` to specify the depended-upon file using the `fileSpec` field. Using a file specification lets your plug-in finds other depended-upon files by iterating through target files using `CWGetFileInfo`.

See Also “`CWDependencyType`” on page 93
“`CWGetProjectFileCount`” on page 67
“`CWObjectData`” on page 180
“`CWStoreObjectData`” on page 165
“`CWGetFileInfo`” on page 55

CWExtensionMapping

Description Provides flags indicating how the IDE should handle files having a specified type and extension.

Prototype `#include <DropInCompilerLinker.h>`

Compiler and Linker Plug-in Reference
CWExtMapList

```
typedef struct CWExtensionMapping {
    CWDataType          type;
    char                extension[32];
    CompilerMappingFlags flags;
} CWExtensionMapping;
```

Fields The fields in CWExtensionMapping are:

type	Specifies the type of the file. Relevant only on the Mac OS.
extension	Specifies the file extension, if any, as a C string.
flags	Specifies bit flags indicating how the IDE should handle the file. Valid values are kPrecompile, kLaunchable, kRsrcfile, and kIgnored. Set all other bits to zero.

Remarks Plug-ins use structures of type CWExtensionMapping to return file mappings to the IDE via the file mapping entry point. File mapping entry points return an array of CWExtensionMapping structures in a CWExtMapList object. Each array entry specifies one kind of project file, by extension and Mac OS file type (if relevant) and flags indicating how files of this type should be handled by the IDE.

See Also “CWExtMapList” on page 178

CWExtMapList

Description Returns a list of file mappings for a plug-in to the IDE.

Prototype

```
#include <DropInCompilerLinker.h>
typedef struct CWExtMapList {
    short                version;
    short                nMappings;
    CWExtensionMapping* mappings;
} CWExtMapList;
```



Fields	The fields in CWExtMapList include:	
	version	Specifies the version of the CWExtMapList structure. Use kCurrentCWExtMapListVersion for the latest version.
	nMappings	Specifies the number of CWExtensionMapping structures appearing in mappings.
	mappings	An array of CWExtensionMapping structures, each of which specifies a single file type, by file type and extension, and its associated flags.

Remarks	Compiler and linker plug-ins return a list of file mappings to the IDE via the CWPlugin_GetDefaultMappingList Entry Point. The IDE calls this entry point when adding file mappings to a newly created project or when the user reverts to factory defaults. The CWExtMapList structure specifies the number of file mappings returned and returns a pointer to an array of CWExtensionMapping structures. Each CWExtensionMapping structure specifies a file type handled by the plug-in, including flags to indicate how the IDE should handle files of that type.
See Also	“Specifying File Mappings” in the <i>IDE SDK Developer’s Guide</i> “CWExtensionMapping” on page 177

CWOBJECTDATA

Description Describes information about the object code generated by a compiler.

Definition

```
#include <DropInCompilerLinker.h>
typedef struct CWOBJECTDATA {
    CWMemHandle    objectdata;
    CWMemHandle    browsedata;
    long           reserved1;
    long           codesize;
    long           udatasize;
    long           idatasize;
    long           compiledlines;
    Boolean        interfacechanged;
    long           reserved2;
    void*          compilecontext;
    CWDependencyInfo* dependencies;
    short          dependencycount;
    CWFileSpec*    objectfile;
} CWOBJECTDATA;
```

Fields The fields in CWOBJECTDATA include:

objectdata	Contains the object code, resource data, or other data that was generated from the file. May be NULL to indicate that the data is stored in a file external to the project.
browsedata	Contains the browser records that were generated for the file. May be NULL if browser data is not supported.
reserved1	Reserved for future use. Plug-ins should store 0 in this field.
codesize	Indicates the size, in bytes, of the object data that was generated from the file.
udatasize	Indicates the size, in bytes, of uninitialized data in the object data.
idatasize	Indicates the size, in bytes, of initialized data in the object data.

compiledlines	Indicates the number of lines of source code processed while generating the object data.
interfaceChanged	Set this value to true if the external interface of the file that was compiled has changed. The IDE then recompiles any other target file with a dependency of type CWInterfaceDependency on that file.
compilecontext	Reserved for use by the IDE. Plug-ins should set this value to 0.
dependencies	Points to an optional array of files that rely on the object data. The array consists of elements of type CWDependencyInfo. Set this value to NULL to specify no dependencies.
dependencyCount	Indicates the number of elements in the dependencies array.
objectfile	Indicates the location of external object code to associate with the file. Ignored unless objectdata is NULL. Set this to NULL to specify no external object code file.

Remarks A plug-in compiler or linker uses the CWOBJECTDATA data structure to store object code, resource data, browser data, and other information generated from a source file. The plug-in passes this data structure to the IDE when it calls CWStoreObjectData.

objectdata stores the object code, resource data, and possibly debugging information generated from a source file or imported from a library file. If objectdata is NULL, then the IDE assumes that the data resides in an external object file, specified by objectfile.

browsedata stores the browser information generated for a file, which consists of symbolic information, such as variables and their types, functions and their parameteres, and classes and their methods and data members. The IDE does not require browsedata, and its value can be NULL.

Compiler and Linker Plug-in Reference

CWObjectData

codesize, udatasize, and idatasize indicate the sizes of any code, data, and initialized data emitted in the objectdata handle. Since the IDE knows nothing about the format of the objectdata handle, it cannot infer these sizes. Any of these values can be 0.

compiledlines indicates the number of lines of source compiled by the plug-in. These fields are simply displayed to the user in the project window and are provided for informational purposes only.

interfaceChanged indicates to the IDE whether the public interface of the compiled file has changed. If so, the IDE recompiles any dependent file having a dependency of type cwInterfaceDependency.

dependencies and dependencyCount specify a list of files that the compiled file depends upon. Note that setting this flag may cause files to be recompiled even when the content of objectdata does not change. The dependencies list can be NULL.

objectfile specifies the location of external object code associated with a file in place of data in objectdata. The IDE ignores objectfile unless objectdata contains a value of NULL.

See Also “CWStoreObjectData” on page 165
 “CWLoadObjectData” on page 161
 “CWDependencyInfo” on page 176
 “CWFindAndLoadFile” on page 45
 “CWDependencyType” on page 93



CWTargetInfo

Description	Specifies information about a project target.
Definition	<pre>#include <DropInCompilerLinker.h> typedef struct CWTargetInfo { short outputFile; CWFileSpec symfile; CWFileSpec runfile; short linkType; Boolean canRun; Boolean canDebug; CWDataType targetCPU; CWDataType targetOS; #if CWPLUGIN_HOST == CWPLUGIN_HOST_MACOS OSType outfileCreator; OSType outfileType; OSType debuggerCreator; OSType runHelperCreator; #endif #if CWPLUGIN_HOST == CWPLUGIN_HOST_WIN32 Boolean runHelperIsRegKey; Boolean debugHelperIsRegKey; char args[512]; char runHelperName[512]; Boolean runHelperRequiresURL; char reserved2; char debugHelperName[512]; #endif CWFileSpec linkAgainstFile; } CWTargetInfo;</pre>

Compiler and Linker Plug-in Reference

CWTargetInfo

Fields The fields in CWTargetInfo include:

Field	Description
outputType	Specifies what the linker creates: linkOutputNone, linkOutputFile, or linkOutputDirectory.
outfile	Specifies the target output file produced by the linker.
symfile	Specifies the file that contains the symbolic debugging information produced by the linker.
runfile	Specifies the file that should be launched or passed to the run helper when the user chooses Run from the Project menu. This specification can be the same file as outfile. If this file is an executable file, such as an application, the IDE launches it directly. Otherwise, the IDE asks the application specified by runHelperName or runHelperCreator to open the file specified by runFile.
linkType	Specifies the target linker model. Possible values include exelinkageFlat, exelinkageSegmented, and exelinkageOverlay1.
canRun	If true, the IDE activates the Run command.
canDebug	If true, the IDE activates the Debug command.
targetCPU	The processor for which the linker generates software. "Constants for Compiler and Linker Plug-ins" on page 191 shows possible values for this field.
targetOS	The operating system for which the linker generates software. "Constants for Compiler and Linker Plug-ins" on page 191 shows possible values for this field.

Field	Description
outfileCreator	If outputType is linkOutputFile, this field contains the creator of the linker's output file. Defined for Mac OS plug-ins only.
outfileType	If outputType is linkOutputFile, this field contains the file type of the linker's output file. Defined for Mac OS plug-ins only.
debuggerCreator	Specifies the signature of the application to be used to debug the linker's output file. Defined for Mac OS plug-ins only.
runHelperCreator	Specifies the signature of the application needed to run the linker's output file. Defined for Mac OS plug-ins only.
runHelperIsRegKey	If true, runHelperName holds a registry key. If false, the IDE treats runHelperName as a file name.
debugHelperIsRegKey	If true, debugHelperName holds a registry key. If false, the IDE treats debugHelperName as a file name.
args	A C string constructed by the plug-in, containing command line arguments, to be passed to the executable prior to launch. Defined for Windows plug-ins only.

Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference CWTargetInfo

Field	Description
runHelperName	If runHelperIsRegKey is false, this value specifies the name of any application required to assist in running the executable. If runHelperIsRegKey is true, this value specifies the name of a Windows registry key, that specifies the full path of the application to use. Defined for Windows plug-ins only.
runHelperRequiresURL	Set this flag if the run helper expects the specification of the file to run to be passed as a URL. Defined for Windows plug-ins only.
reserved2	Reserved. Do not use. Defined on Windows only.
debugHelperName	If debugHelperIsRegKey is false, this value specifies the name of any application required to assist in debugging the executable. If debugHelperIsRegKey is true, this value specifies the name of a Windows registry key, which specifies the full path of the application to use. Defined for Windows plug-ins only.
linkAgainstFile	Specifies a file within the project that includes this file should link into its final binary. The specified file can be the same as outfile.

Remarks A linker plug-in uses the CWTargetInfo data structure to store information about its target executable. CWGetTargetInfo and CWSetTargetInfo use this data structure.

outputType and outfile specify the type and location of the linker's output. symfile specifies the location of the corresponding



symbolics file, if any, used by a debugger. `runfile` specifies a file to be passed to the run helper application. If `runfile` matches `outfile`, the IDE launches the target executable. If `runfile` is specified, but no run helper exists, the IDE launches the target application and passes it the `runfile`.

`linkType` describes the organization and addressing model of code in the final executable produced by the linker. The IDE supports normal ("flat"), overlayed (used most commonly by embedded systems), and segmented (Mac 68K code resource) linkage models.

`canRun` specifies whether the IDE should enable the **Project > Run** command for this target. `canDebug` specifies whether the IDE should enable the **Project > Debug** command for this target.

`targetCPU` and `targetOS` specify the CPU and operating system on which the target runs. The IDE uses these fields to enable other plug-in types, as determined by the currently selected linker. `DropinCompilerLinker.h` defines the appropriate values for these fields.

Windows When true, `runHelperIsRegKey` and `debugHelperIsRegKey` indicate that `runHelperName` and `debugHelperName` are full paths specifying the run and debug helper applications, respectively. When false, `runHelperName` and `debugHelperName` are the names of registry keys specifying the full paths of the run and debug helper applications. `runHelperRequiresURL` indicates that the run helper wants its primary file argument (which refers to `runfile`) specified using URL syntax.

As an example, to invoke a run helper named 'javavm' on a `runfile` named `test.java` in directory `c:\testapp` with an args strings of '-a -b -c', with the `runfile` specified using URL syntax, the IDE would construct this command line:

```
javavm file://c:/testapp/test.java -a -b -c
```

`args` specifies command line arguments to be passed to the final executable when launched, either directly or through the run helper application. The plug-in specifies these arguments and returns them to the IDE. Linkers typically obtain the command line arguments

Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

CWTargetList

specified by the user in **Runtime Settings** using `CWGetCommandLineArgs`, add any necessary additional command line arguments, and return the complete string of arguments to the IDE.

`linkAgainstFile` specifies the file produced by this target, which should be linked into any parent projects that include the current project as a subproject. `linkAgainstFile` does not always match as `outfile`. For example, when compiling DLLs, the linker produces both a `.dll` file and a `.lib` file. The former contains the code, but the latter is used when linking projects that use the DLL library.

Mac OS `outfileType` and `outfileCreator` specify the file type (usually but not always 'APPL') and the signature, respectively, for Mac OS application targets.

`debuggerCreator` specifies the signature of the application to use to debug the target executable, when the user selects **Debug** from the **Project** menu.

`runHelperCreator` specifies the signature of the run helper application to use assist in running the target executable.

NOTE The definition of this structure varies by platform.

See Also "CWGetTargetInfo" on page 153
"CWSetTargetInfo" on page 163
"linkOutputNone" on page 206
"linkOutputFile" on page 206
"linkOutputDirectory" on page 207
"Constants for Compiler and Linker Plug-ins" on page 191

CWTargetList

Description Returns a list of supported CPUs and operating systems to the IDE.
Prototype `#include <DropInCompilerLinker.h>`

```
typedef struct CWTargetList {
    short      version;
    short      cpuCount;
    CWDataType* cpus;
    short      osCount;
    CWDataType* oss;
} CWTargetList;
```

Fields The fields in CWTargetList are:

version	Specifies the version of this structure (different versions contain different information). To use the latest version of this structure, specify kCurrentCWTargetListVersion for this field.
cpuCount	Specifies the number of CPU codes in the cpus array.
cpus	An array of CPU codes, each specifying one target CPU.
osCount	Specifies the number of OS codes in the oss array.
oss	An array of OS codes, each specifying one target operating system.

Remarks Plug-ins use a structure of type CWTargetList to return information about the CPUs and operating systems they support to the IDE. The IDE uses this information to match compilers and linkers and their associated preference panels with the currently selected target linker.

DropInCompilerLinker.h defines valid CPU and operating system codes.

NOTE The IDE lets only one of cpuCount and osCount be greater than one. That is, plug-ins can indicate support for multiple CPUs or for multiple operating systems, but not for both.

See Also “Specifying Target Platforms” in the IDE SDK Developer’s Guide
“CWPlugin_GetTargetList Entry Point” on page 168
“CWDataType” on page 93

CWUnmangleInfo

Description Contains information required to unmangle a single compiler or linker symbol.

Definition `#include <DropInCompilerLinker.h>`
 `typedef struct CWUnmangleInfo {`
 `void*                    targetStorage;`
 `const char*          mangledName;`
 `char*                 unmangleBuff;`
 `long                  unmangleBuffSize;`
 `unsigned short       browserClassID;`
 `unsigned char        browserLang;`
 `unsigned char        filler1;`
 `} CWUnmangleInfo;`

Fields The fields in CWUnmangleInfo include:

targetStorage	Specifies the plug-in's target data, if available. If required, this data should have been previously set up in response to a reqTargetLoaded request.
mangledName	Specifies the text containing the name to unmangle, formatted as a NULL-terminated C string.
unmangleBuff	Provides a buffer in which the plug-in should store the unmangled name as a NULL-terminated C string.
unmangleBuffSize	Specifies the size of unmangleBuff. The plug-in should not place text larger than this size in unmangleBuff, including the terminating NULL byte.
browserClassID	Specifies the EBrowserItem value (that is, the type) of the item to be unmangled.
browserLang	Returns the ELanguage value for the source language of the symbol. The plug-in should specify the same language that was used when generating the browser data.
filler1	Reserved. Do not use.

Remarks	The IDE uses this data structure when it calls a linker’s unmangling entry point to translate a compiler or linker’s internal symbol into its human-readable version. The linker should examine the mangled name provided in <code>mangledName</code> and any information about the mangling process that the compiler or linker has retained in <code>targetStorage</code>, and store an unmangled version of the symbol name in <code>unmangleBuff</code>. The length of the unmangled symbol should not exceed the size specified in <code>unmangleBuffSize</code>, including the terminating NULL byte. <code>browserClassID</code> specifies the type of the symbol being unmangled. <code>browserLang</code> specifies the source language of the symbol, as emitted by the compiler while generating browser symbols. <code>CompilerMapping.h</code> defines the possible values.
See Also	“Helper_Unmangle Entry Point” on page 171 “reqTargetLoaded” on page 219 “EBrowserItem” on page 246 “ELanguage” on page 247

Constants for Compiler and Linker Plug-ins

This section describes the constant and predefined values in the compiler/linker plug-in API for the CodeWarrior IDE on Mac OS.

These constants are:

- `cantDisassemble`
- `cwAccessAbsolute`
- `cwAccessPathRelative`
- `cwAccessFileName`
- `cwAccessFileRelative`
- `DROPINCOMPILERLINKERAPIVERSION`
- `exelinkageFlat`
- `exelinkageOverlay1`



Freescal Semiconductor, Inc.

Compiler and Linker Plug-in Reference

Constants for Compiler and Linker Plug-ins

- `exelinkageSegmented`
- `isPostLinker`
- `isPreLinker`
- `kCandisassemble`
- `kCanimport`
- `kCanprecompile`
- `kCanpreprocess`
- `kCompAllowDupFileNames`
- `kCompAlwaysReload`
- `kCompEmitsOwnBrSymbols`
- `kCompMultiTargAware`
- `kCompReentrant`
- `kCompRequiresFileListBuildStartedMsg`
- `kCompRequiresProjectBuildStartedMsg`
- `kCompRequiresSubProjectBuildStartedMsg`
- `kCompRequiresTargetBuildStartedMsg`
- `kCompRequiresTargetCompileStartedMsg`
- `kCompSavesDbgPreprocess`
- `kCompUsesTargetStorage`
- `kGeneratescode`
- `kGeneratesrsrsrcs`
- `kIgnored`
- `kIsMPAware`
- `kIspascal`
- `kLaunchable`
- `kPersistent`
- `kPrecompile`
- `kRsrcfile`
- `linkAllowDupFileNames`
- `linkAlwaysReload`
- `linkMultiTargAware`



Freescal Semiconductor, Inc.

Compiler and Linker Plug-in Reference

Constants for Compiler and Linker Plug-ins

- linkOutputDirectory
- linkOutputFile
- linkOutputNone
- linkRequiresFileListBuildStartedMsg
- linkRequiresProjectBuildStartedMsg
- linkRequiresSubProjectBuildStartedMsg
- linkRequiresTargetBuildStartedMsg
- linkRequiresTargetLinkStartedMsg
- linkerDisasmRequiresPreprocess
- linkerGetTargetInfoThreadSafe
- linkerInitializeOnMainThread
- linkerSuggestsNonRecursiveAccessPaths
- linkerUnmangles
- linkerUsesCaseInsensitiveSymbols
- linkerUsesFrameworks
- linkerUsesTargetStorage
- linkerWantsPreRunRequest
- magicCapLinker
- reqCheckSyntax
- reqCompile
- reqCompDisassemble
- reqDisassemble
- reqFileListBuildEnded
- reqFileListBuildStarted
- reqLink
- reqPreprocessForDebugger
- reqProjectBuildEnded
- reqProjectBuildStarted
- reqSubProjectBuildEnded
- reqSubProjectBuildStarted
- reqTargetBuildEnded

Compiler and Linker Plug-in Reference

cantDisassemble

- reqTargetBuildStarted
- reqTargetInfo
- reqTargetLinkEnded
- reqTargetLinkStarted
- reqTargetLoaded
- reqTargetPrefsChanged
- reqTargetUnloaded
- targetCPU68K
- targetCPUAny
- targetCPUEmbeddedPowerPC
- targetCPUI80x86
- targetCPUMips
- targetCPUNECv800
- targetCPUPowerPC
- targetOSAny
- targetOSEmbeddedABI
- targetOSMacintosh
- targetOSMagicCap
- targetOSOS9
- targetOSWindows

cantDisassemble

Description	Specifies that a linker does <i>not</i> support disassembly.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in a linker's <code>DropInFlags</code> structure to indicate that it does not support disassembly in response to a <code>reqDisassemble</code> request.

NOTE Unlike most dropin flags, setting this flag indicates that the plug-in does *not* support the corresponding feature.

CAUTION Do not confuse `cantDisassemble` with `kCandisassemble`, which applies to compilers and has a different numeric value.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

cwAccessAbsolute

Description Specifies that a depended-upon file should be stored using its absolute path.

Definition `#include <DropInCompilerLinker.h>`

Remarks Use `cwAccessPathAbsolute` in a `CWDependencyInfo` structure passed to `CWStoreObjectData` to specify that the IDE should store the dependency using a full path.

See Also “CWObjectData” on page 180
“CWStoreObjectData” on page 165

cwAccessPathRelative

Description Specifies that a depended-upon file should be stored using a file path relative to one of the active target’s access paths.

Definition `#include <DropInCompilerLinker.h>`

Remarks Use `cwAccessPathRelative` in a `CWDependencyInfo` structure passed to `CWStoreObjectData` to specify that the IDE should store the dependency using a path relative to one of the project paths.

See Also “CWObjectData” on page 180
“CWStoreObjectData” on page 165

cwAccessFileName

Description Specifies that a depended-upon file should be stored using only its file name, not its path.

Definition `#include <DropInCompilerLinker.h>`

Remarks Use `cwAccessPathRelative` in a `CWDependencyInfo` structure passed to `CWStoreObjectData` to specify that the IDE should

Compiler and Linker Plug-in Reference

cwAccessFileRelative

store the dependency using only a file's name, rather than its path. This method of locating a file often fails if a project contains more than one file with the same name.

cwAccessFileRelative

Description	Specifies that the IDE should search for a depended-upon file by using a path relative to the dependent file's path.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	Use <code>cwAccessFileRelative</code> in a <code>CWDependencyInfo</code> structure passed to <code>CWStoreObjectData</code> to specify that the IDE should store the dependency using a path relative to the dependent file.
See Also	"CWObjectData" on page 180 "CWStoreObjectData" on page 165

DROPINCOMPILERLINKERAPIVERSION

Description	Specifies the current version of the compiler and linker plug-in API at compile time.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This predefined symbol specifies the current version of the compiler plug-in API. Use this value in a <code>DropInFlags</code> structure to indicate that your plug-in supports the latest plug-in API version that was available at the time it was compiled.

NOTE Any plug-in that uses this value automatically specifies the latest API version when the plug-in API headers change and the plug-in is recompiled. Plug-ins that depend upon older API versions should use a different fixed constant to specify the supported API version.

exelinkageFlat

Description	Specifies that the plug-in generates an executable binary consisting of a single piece of binary code.
Definition	<code>#include <DropInCompilerLinker.h></code>

Remarks	This predefined symbol, used in the <code>linkType</code> field of the <code>CWTargetInfo</code> data structure, specifies that the linker generates executables composed of one contiguous piece of executable code, rather than multiple, independently loaded code segments or overlays. For targets specifying this linkage type, the IDE displays files in the link order project view in a single list, without any subgroups.
See Also	“CWTargetInfo” on page 183

exelinkageOverlay1

Description	Specifies that the plug-in generates an executable binary comprised of overlays.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This predefined symbol, used in the <code>linkType</code> field of the <code>CWTargetInfo</code> data structure, specifies that the linker organizes its executable file into overlays. Overlays consist of code images loaded into the same portion of physical memory under program control, to reduce memory requirements. For targets that specify this linkage type, the IDE displays files in the link order project view as overlays. This view lets files be grouped into overlays. Overlays can be grouped into overlay groups.
See Also	“CWTargetInfo” on page 183

exelinkageSegmented

Description	Specifies that the plug-in generates executable code consisting of separate code resources.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This predefined symbol, used in the <code>linkType</code> field of the <code>CWTargetInfo</code> data structure, specifies that the linker organizes an executable file into code segments. Only the Mac OS uses this setting. For targets that specify this linkage type, the IDE displays files grouped into segments.

Compiler and Linker Plug-in Reference

isPostLinker

See Also “CWTargetInfo” on page 183

isPostLinker

Description Specifies that a plug-in is a post-linker.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker’s `DropInFlags` to indicate that it is a post-linker that receives IDE requests after the target linker has been run.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

isPreLinker

Description Specifies that a linker plug-in is a pre-linker.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker’s `DropInFlags` to indicate that it is a pre-linker that receives IDE requests before the target linker runs.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

kCandisassemble

Description Specifies that a compiler supports code disassembly.

Definition `#include <CompilerMapping.h>`

Remarks This constant appears in a compiler’s `DropInFlags` to indicate that it supports code disassembly, in response to a `reqCompDisassemble` request.

CAUTION Do not confuse `kCandisassemble` with `cantDisassemble`, which applies to linkers and has a different numeric value.

See Also “Compiler Capability Flags” in the *IDE SDK Developer’s Guide*

kCanimport

Description Specifies that a compiler can import a compiled object code library.

Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that it imports compiled object code by adding it to the project in response to a <code>reqCompile</code> request. When this flag is set, the IDE enables the Weak Link checkbox in the project inspector for the file types associated with the compiler.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCanprecompile

Description	Specifies that a compiler can precompile one or more of the source file types it handles.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that the compiler supports precompilation of interface files, to accelerate the compilation process. To support precompilation, plug-ins must also mark the relevant file types as precompilable.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i> "Specifying File Mappings" in the <i>IDE SDK Developer's Guide</i>

kCanpreprocess

Description	Specifies that a compiler can preprocess one or more of the source file types it handles.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that the compiler performs textual substitution or expansion on one or more of its file types. The plug-in then presents the results to users in a window in response to a <code>reqPreprocess</code> request.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCompAllowDupFileNames

Description	Specifies that a compiler can properly handle the appearance of multiple files in one target having the same name.
-------------	--

Compiler and Linker Plug-in Reference

kCompAlwaysReload

Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that the compiler can handle project targets containing multiple files having the same name (but different paths). Most compilers should set this flag. In the past, some compilers used file names to construct unique symbol names, which caused problems when multiple files having the same name appeared in one target.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCompAlwaysReload

Description	Specifies that the IDE should reload the compiler before every request.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that the compiler needs to be reloaded between requests from the IDE (except <code>reqInitialize</code> and <code>reqTerminate</code>). This setting supports compilers that have complex global state that must be reinitialized. Plug-ins generally use this setting when porting existing (often command line) tools that make heavy use of global state.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCompEmitsOwnBrSymbols

Description	Specifies that a compiler emits browser symbols of specialized type.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that the browser symbol information generated by a compiler contains custom type codes, indicating the presence of nonstandard symbol types. When set, the IDE calls the compiler's browser symbol entry point, to obtain the names of the custom symbol types.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i> "Helper_GetCompilerBrSymbols Entry Point" on page 168

kCompMultiTargAware

Description Unused.

kCompReentrant

Description Specifies that the compiler is reentrant and permits the IDE to issue commands to multiple instances simultaneously.

Definition `#include <CompilerMapping.h>`

Remarks This flag enables concurrent compilation.

See Also “Compiler Capability Flags” in the *IDE SDK Developer’s Guide*

kCompRequiresFileListBuildStartedMsg

Description Specifies that a compiler requires the reqFileListBuildStarted and reqFileListBuildEnded requests.

Definition `#include <CompilerMapping.h>`

Remarks This constant appears in a compiler’s DropInFlags to indicate that a compiler wants to receive reqFileListBuildStarted and reqFileListBuildEnded requests.

See Also “Compiler Capability Flags” in the *IDE SDK Developer’s Guide*

kCompRequiresProjectBuildStartedMsg

Description Specifies that a compiler requires the reqProjectBuildStarted and reqProjectBuildEnded requests.

Definition `#include <CompilerMapping.h>`

Remarks This constant appears in a compiler’s DropInFlags to indicate that a compiler wants to receive reqProjectBuildStarted and reqProjectBuildEnded requests.

See Also “Compiler Capability Flags” in the *IDE SDK Developer’s Guide*



Freescal Semiconductor, Inc.

Compiler and Linker Plug-in Reference

kCompRequiresSubProjectBuildStartedMsg

kCompRequiresSubProjectBuildStartedMsg

Description	Specifies that a compiler requires the reqSubProjectBuildStarted and reqSubProjectBuildEnded requests.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's DropInFlags to indicate that a compiler wants to receive reqSubProjectBuildStarted and reqSubProjectBuildEnded requests.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCompRequiresTargetBuildStartedMsg

Description	Specifies that a compiler requires the reqTargetBuildStarted and reqTargetBuildEnded requests.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's DropInFlags to indicate that a compiler wants to receive reqTargetBuildStarted and reqTargetBuildEnded requests.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCompRequiresTargetCompileStartedMsg

Description	Indicates that the compiler can receive target compile started and ended messages.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	A compiler that recognizes started and ended messages can sometimes benefit from pre-processing or post-processing after all individual compilations have been completed.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kCompSavesDbgPreprocess

Description	Obsolete. Do not use.
-------------	-----------------------

kCompUsesTargetStorage

Description	Specifies that a compiler uses target storage services.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's DropInFlags to indicate that a compiler uses target storage facilities (CWGetTargetStorage and CWSetTargetStorage). Plug-ins use target storage to store global data that a plug-in wishes to preserve even when the IDE unloads the plug-in.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kGeneratescode

Description	Specifies that a compiler generates code.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's DropInFlags to indicate that a compiler generates object code in response to a reqCompile request and stores it in the project using CWStoreObjectData. The only compilers that should not set this flag are those that exclusively generate source code or resources.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kGeneratesrsrcs

Description	Specifies that a compiler generates resource data.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's DropInFlags to indicate that a compiler generates binary resource data in response to a reqCompile request and stores it in the project using CWStoreObjectData.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kIgnored

Description	Specifies that a file type should be ignored by the IDE during Make operations.
-------------	--

Compiler and Linker Plug-in Reference

kLaunchable

Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant indicates that, during build operations, the IDE should completely ignore the file type specified in a plug-in's file mappings.
See Also	"CWPlugin_GetDefaultMappingList Entry Point" on page 167

kLaunchable

Description	Specifies a file type that may be launched directly from the project window.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant indicates that, when the user double-clicks a file in the project window, the IDE should launch or open files specified in a plug-in's file mappings.
See Also	"CWPlugin_GetDefaultMappingList Entry Point" on page 167

kIsMPAware

Description	Unused.
-------------	---------

kIsPascal

Description	Reserved.
-------------	-----------

kPersistent

Description	Specifies that the IDE should avoid unloading the compiler.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a compiler's <code>DropInFlags</code> to indicate that the compiler needs to be loaded and unloaded as infrequently as possible. Setting this flag does <i>not</i> guarantee that a plug-in remains resident for the duration of an IDE run. This is most commonly used for plug-ins that perform complex initialization tasks, to reduce the time spent initializing plug-in state.
See Also	"Compiler Capability Flags" in the <i>IDE SDK Developer's Guide</i>

kPrecompile

Description	Specifies that a file type may be precompiled.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a plug-in's file mappings to indicate that the corresponding file type can be precompiled by the plug-in. Plug-ins must set this flag to receive reqPrecompile requests
See Also	"CWPlugin_GetDefaultMappingList Entry Point" on page 167

kRsrcfile

Description	Specifies that a file type is a binary resource file.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	This constant appears in a plug-in's file mappings to indicate that the corresponding file type is a binary resource file, to be included in final compiled executables. Generally, only the Mac OS uses this setting.
See Also	"CWPlugin_GetDefaultMappingList Entry Point" on page 167

linkAllowDupFileNames

Description	Specifies that a linker properly handles targets containing multiple files having the same name.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in a linker's DropInFlags to indicate that a linker supports projects containing multiple files with the same name. Most linkers should set this flag. An exception would be linkers that use a file's name to help construct unique symbol names.
See Also	"Linker Capability Flags" in the <i>IDE SDK Developer's Guide</i> "Specifying File Mappings" in the <i>IDE SDK Developer's Guide</i>

linkAlwaysReload

Description	Specifies that the IDE should reload the linker before every request.
-------------	---

Compiler and Linker Plug-in Reference

linkMultiTargAware

Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in a linker's <code>DropInFlags</code> to indicate that the linker needs to be reloaded between requests from the IDE (except <code>reqInitialize</code> and <code>reqTerminate</code>). This setting supports linkers that have complex global state that must be reinitialized. Plug-ins generally use this setting when porting existing (often command line) tools that make heavy use of global state.
See Also	"Linker Capability Flags" in the <i>IDE SDK Developer's Guide</i>

linkMultiTargAware

Description	Specifies that a linker wishes to be reloaded every time the project target changes.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in a linker's <code>DropInFlags</code> to indicate that a linker wants to receive <code>reqInitialize</code> and <code>reqTerminate</code> requests every time the current project target changes. When this flag is <i>clear</i> , the IDE sends the requests. When this flag is set, the IDE does <i>not</i> send the requests.
See Also	"Linker Capability Flags" in the <i>IDE SDK Developer's Guide</i>

linkOutputNone

Description	Specifies that the linker produces no final target binaries.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This predefined symbol, used in the <code>outputType</code> field of the <code>CWTargetInfo</code> data structure, specifies that the linker produces no binary output file.
See Also	"CWTargetInfo" on page 183

linkOutputFile

Description	Specifies that the linker produces a file.
Definition	<code>#include <DropInCompilerLinker.h></code>

Remarks This predefined symbol, used in the `outputType` field of the `CWTargetInfo` data structure, specifies that the linker produces a file.

See Also “`CWTargetInfo`” on page 183

linkOutputDirectory

Description Specifies that the linker produces a directory and one or more files within the directory.

Definition `#include <DropInCompilerLinker.h>`

Remarks This predefined symbol, used in the `outputType` field of the `CWTargetInfo` data structure, specifies that the linker produces a directory containing target files.

See Also “`CWTargetInfo`” on page 183

linkRequiresFileListBuildStartedMsg

Description Specifies that a linker requires `reqFileListBuildStarted` and `reqFileListBuildEnded` requests.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker’s `DropInFlags` to indicate that a linker wants to receive `reqFileListBuildStarted` and `reqFileListBuildEnded` requests.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

“Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

linkRequiresProjectBuildStartedMsg

Description Specifies that a linker requires `reqProjectBuildStarted` and `reqProjectBuildEnded` requests.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker’s `DropInFlags` to indicate that a linker wants to receive `reqProjectBuildStarted` and `reqProjectBuildEnded` requests.

Compiler and Linker Plug-in Reference

linkRequiresSubProjectBuildStartedMsg

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*
 “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

linkRequiresSubProjectBuildStartedMsg

Description Specifies that a linker requires reqSubProjectBuildStarted and reqSubProjectBuildEnded request.

Definition #include <DropInCompilerLinker.h>

Remarks This constant appears in a linker’s DropInFlags to indicate that a linker wants to receive reqSubProjectBuildStarted and reqSubProjectBuildEnded requests.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*
 “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

linkRequiresTargetBuildStartedMsg

Description Specifies that a linker requires reqTargetBuildStarted and reqTargetBuildEnded request.

Definition #include <DropInCompilerLinker.h>

Remarks This constant appears in a linker’s DropInFlags to indicate that a linker wants to receive reqTargetBuildStarted and reqTargetBuildEnded requests.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*
 “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

linkRequiresTargetLinkStartedMsg

Description Specifies that a linker requires reqTargetLinkStarted and reqTargetLinkEnded request.

Definition #include <DropInCompilerLinker.h>

Remarks This constant appears in a linker's `DropInFlags` to indicate that a linker wants to receive `reqTargetLinkStarted` and `reqTargetLinkEnded` requests.

See Also "Linker Capability Flags" in the *IDE SDK Developer's Guide*
"Additional Compiler and Linker Requests" in the *IDE SDK Developer's Guide*

linkerDisasmRequiresPreprocess

Description Obsolete. Do not use.

linkerGetTargetInfoThreadSafe

Description Specifies that a linker implements the `reqTargetInfo` request in a thread-safe manner.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker's `DropInFlags` to indicate whether the linker is thread safe.

See Also "Linker Capability Flags" in the *IDE SDK Developer's Guide*
"Providing Target Information" in the *IDE SDK Developer's Guide*

linkerInitializeOnMainThread

Description Specifies that a linker must initialize on the main thread.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker's `DropInFlags` to indicate that the linker must initialize on the main thread.

See Also "Linker Capability Flags" in the *IDE SDK Developer's Guide*

linkerSuggestsNonRecursiveAccessPaths

Description When off, the IDE creates a new access path. When on, it recurses access paths.

Definition `#include <CompilerMapping.h>`

Compiler and Linker Plug-in Reference

linkerUnmangles

Remarks On Mac and Windows, the default is for new paths to have the recursive flag on.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

linkerUnmangles

Description Specifies that a linker supports name unmangling.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in a linker’s `DropInFlags` to indicate that a linker exports a name unmangling entry point, for converting mangled identifiers back into human-readable form.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

“Linker Symbol Unmangling Entry Point” in the *IDE SDK Developer’s Guide*

linkerUsesCaseInsensitiveSymbols

Description If this flag is on, browser symbols do not consider case. If it is off, browser symbols consider case.

Definition `#include <CompilerMapping.h>`

Remarks Use this flag to specify whether to use case-sensitive or -insensitive browser symbols.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

linkerUsesFrameworks

Description Setting this flag enables IDE handling of frameworks in the MacOS X Mach-o target.

Definition `#include <CompilerMapping.h>`

Mac OS This flag only applies to the mach-o target.

Remarks Use this flag to specify whether the IDE should handle frameworks in the MacOS X Mach-o target.

See Also “Linker Capability Flags” in the *IDE SDK Developer’s Guide*

linkerUsesTargetStorage

Description	Specifies that a linker uses target storage.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in a linker's <code>DropInFlags</code> to indicate that the linker uses target storage, by calling <code>CWGetTargetStorage</code> and <code>CWSetTargetStorage</code> .
See Also	"Linker Capability Flags" in the <i>IDE SDK Developer's Guide</i> "CWGetTargetStorage" on page 154 "CWSetTargetStorage" on page 164 "Managing Target Storage" in the <i>IDE SDK Developer's Guide</i>

linkerWantsPreRunRequest

Description	Specifies that a linker wants to receive a <code>reqPreRun</code> request.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in a linker's <code>DropInFlags</code> to indicate that it want to receive the <code>reqPreRun</code> request, prior to launch of the target executable.
See Also	"Linker Capability Flags" in the <i>IDE SDK Developer's Guide</i>

magicCapLinker

Description	Obsolete. do not use.
-------------	-----------------------

reqCheckSyntax

Description	Reserved for use by Metrowerks.
-------------	---------------------------------

reqCompile

Description	Asks a compiler plug-in to compile a project file into object data, resource data, or source code text.
Definition	<code>#include <DropInCompilerLinker.h></code>

Compiler and Linker Plug-in Reference

reqCompDisassemble

Remarks The IDE sends this request when compiling a source file, as part of a **Compile, Make, Bring Up To Date, Check Syntax, Preprocess, or Precompile** operation. To determine which file should be compiled, use `CWGetMainFileID`, `CWGetMainFileSpec`, or `CWGetMainFileText`.

See Also “Compiling” in the *IDE SDK Developer’s Guide*
 “CWGetMainFileID” on page 145
 “CWGetMainFileSpec” on page 147
 “CWGetMainFileText” on page 147
 “CWStoreObjectData” on page 165
 “CWGetPluginRequest” on page 66
 “CWDonePluginRequest” on page 45

reqCompDisassemble

Description Asks a compiler plug-in to convert binary object code into human-readable text.

Definition `#include <DropInCompilerLinker.h>`

Remarks The IDE sends this request when the user chooses **Disassemble** from the **Project** menu. To determine which file should be disassembled, use `CWGetMainFileID` or `CWGetMainFileSpec`.

The IDE only sends this request if the compiler plug-in specifies that it handles disassembly in its 'Flag' resource by setting the `kCandisassemble` flag.

See Also “Disassembling” in the *IDE SDK Developer’s Guide*
 “'Flag' Resource” in the *IDE SDK Developer’s Guide*
 “CWGetMainFileID” on page 145
 “CWGetMainFileSpec” on page 147
 “CWGetPluginRequest” on page 66

“CWDonePluginRequest” on page 45

reqDisassemble

Description Asks a linker to generate human-readable text from object data.

Definition `#include <DropInCompilerLinker.h>`

The IDE sends this request when the user chooses **Disassemble** from the **Project** menu. To determine which file should be disassembled, use `CWGetMainFileID` or `CWGetMainFileSpec`.

The IDE only sends this request if the linker plug-in specifies that it handles disassembly in its 'Flag' resource by setting the `cantDisassemble` flag.

To determine which file must be disassembled, use `CWGetMainFileID`. To obtain the object code for the file to be disassembled, use `CWLoadObjectData`.

See Also “Disassembling” in the *IDE SDK Developer’s Guide*

“CWGetMainFileID” on page 145

“CWLoadObjectData” on page 161

“CWGetPluginRequest” on page 66

“CWDonePluginRequest” on page 45

reqFileListBuildEnded

Description The IDE sends this request at the end of a **Compile** operation.

Definition `#include <DropInCompilerLinker.h>`

Remarks The IDE sends this request when it has finished compiling one or more files manually compiled by the user (that is, files compiled using **Compile**, rather than **Make**).

The IDE sends this request only once, at the end of a **Compile** operation.

See Also “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

Compiler and Linker Plug-in Reference

reqFileListBuildStarted

“reqFileListBuildStarted” on page 214

“reqProjectBuildEnded” on page 215

reqFileListBuildStarted

Description	The IDE sends this request after the user selects the Compile command and before compiling any files.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request prior to compiling files when the user selects the Compile command. The IDE sends this request only once, at the start of a Compile operation.
See Also	“Additional Compiler and Linker Requests” in the <i>IDE SDK Developer’s Guide</i> “reqFileListBuildEnded” on page 213 “reqProjectBuildStarted” on page 215

reqLink

Description	Asks a linker plug-in to produce a final output file.
Definition	<code>#include <DropInCompilerLinker.h></code> The IDE sends this request when the user chooses Make from the Project menu. The IDE uses this request to link the active target’s object data into a final library or executable.
See Also	“Linking” in the <i>IDE SDK Developer’s Guide</i> “CWGetPluginRequest” on page 66 “CWDonePluginRequest” on page 45

reqMakeParse

Description	Reserved.
-------------	-----------

reqPreprocessForDebugger

Description Obsolete.

reqPreRun

Description Informs a linker or post-linker that the final target executable is about to be launched.

Definition `#include <DropInCompilerLinker.h>`

Remarks The IDE issues this request prior to executing the target binary or any specified helper applications.

See Also “CWTargetInfo” on page 183

“CWGetPluginRequest” on page 66

“CWDonePluginRequest” on page 45

reqProjectBuildEnded

Description The IDE sends this request after completing a build operation initiated by the **Make** command.

Definition `#include <DropInCompilerLinker.h>`

Remarks The IDE sends this request after compiling all files in a project, when the compilation occurred as a result of a **Make** command (rather than a **Compile** command).

The IDE sends this request only once at the end of a **Make** operation.

See Also “Additional Compiler and Linker Requests” in the *IDE SDK Developer's Guide*

“reqProjectBuildStarted” on page 215

“reqFileListBuildEnded” on page 213

reqProjectBuildStarted

Description The IDE sends this request prior to compiling any files when the user selects **Make**.



Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

reqSubProjectBuildEnded

Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request before compiling files in response to the user selecting Make . The IDE sends this request only once at the start of a Make operation.
See Also	“Additional Compiler and Linker Requests” in the <i>IDE SDK Developer’s Guide</i> “reqProjectBuildEnded” on page 215 “reqFileListBuildStarted” on page 214

reqSubProjectBuildEnded

Description	The IDE sends this request after completing a build of a subproject, when compilation was initiated by the Make command.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request at the completion of a subproject Make operation. The IDE sends one pair of <code>reqSubProjectBuildStarted</code> / <code>reqSubProjectBuildEnded</code> requests for each subproject encountered during a Make operation.
See Also	“Additional Compiler and Linker Requests” in the <i>IDE SDK Developer’s Guide</i> “reqSubProjectBuildStarted” on page 216

reqSubProjectBuildStarted

Description	The IDE sends this request prior to compiling a subproject when compilation was initiated by the Make command.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request when prior to compiling any files in a subproject during a Make operation.

The IDE sends one pair of `reqSubProjectBuildStarted` / `reqSubProjectBuildEnded` requests for each subproject encountered during a **Make** operation.

See Also “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

“`reqSubProjectBuildEnded`” on page 216

reqTargetBuildEnded

Description The IDE sends this request after compiling a target as part of a **Make** operation.

Definition `#include <DropInCompilerLinker.h>`

Remarks The IDE sends this request after compiling all files in a target during a **Make** operation.

The IDE sends one pair of `reqTargetBuildStarted` / `reqTargetBuildEnded` requests for each target encountered during a **Make** operation.

See Also “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

“`reqTargetBuildStarted`” on page 217

reqTargetBuildStarted

Description The IDE sends this request prior to compiling a target as part of a **Make** operation.

Definition `#include <DropInCompilerLinker.h>`

The IDE sends this request before compiling any files in a target during a **Make** operation.

The IDE sends one pair of `reqTargetBuildStarted` / `reqTargetBuildEnded` requests for each target encountered during a **Make** operation.

See Also “Additional Compiler and Linker Requests” in the *IDE SDK Developer’s Guide*

Compiler and Linker Plug-in Reference

reqTargetCompileEnded

“reqTargetBuildEnded” on page 217

reqTargetCompileEnded

Description	The IDE sends this request when a target’s compile phase ends.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	The IDE enables this request flag when it sets the <code>kCompRequiresTargetCompileStartedMsg</code> flag.
See Also	“reqTargetCompileStarted” on page 218

reqTargetCompileStarted

Description	The IDE sends this request when a target’s compile phase begins.
Definition	<code>#include <CompilerMapping.h></code>
Remarks	The IDE enables this request flag when it sets the <code>kCompRequiresTargetCompileStartedMsg</code> flag.
See Also	“reqTargetCompileEnded” on page 218

reqTargetInfo

Description	Asks a linker plug-in to describe the output it creates.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE uses this request to get information about the linker’s output. When it receives this request, the linker should fill out a <code>CWTargetInfo</code> structure and call <code>CWSetTargetInfo</code> .
See Also	“Providing Target Information” in the <i>IDE SDK Developer’s Guide</i> “CWSetTargetInfo” on page 163 “CWGetPluginRequest” on page 66 “CWDonePluginRequest” on page 45

reqTargetLinkEnded

Description	The IDE sends this request after linking a target as part of a Make operation.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request after linking a target during a Make operation. The IDE sends one pair of reqTargetLinkStarted / reqTargetLinkEnded requests for each target linked during a Make operation.
See Also	"Additional Compiler and Linker Requests" in the <i>IDE SDK Developer's Guide</i>

reqTargetLinkStarted

Description	The IDE sends this request prior to linking a target as part of a Make operation.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request prior to linking a target during a Make operation. The IDE sends one pair of reqTargetLinkStarted / reqTargetLinkEnded requests for each target linked during a Make operation.
See Also	"Additional Compiler and Linker Requests" in the <i>IDE SDK Developer's Guide</i>

reqTargetLoaded

Description	Tells the plug-in to allocate and set its target storage.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	The IDE sends this request when switching targets, to tell the plug-in to allocate its target storage. The IDE sends this request only if the plug-in has set the kCompUsesTargetStorage or linkerUsesTargetStorage flag in its 'Flag' resource.

Compiler and Linker Plug-in Reference

reqTargetPrefsChanged

A plug-in allocates target storage by calling `CWAllocateMemory`. A plug-in then calls `CWSetTargetStorage` to pass the pointer back to the IDE. During subsequent requests from the IDE, a plug-in may call `CWGetTargetStorage` to retrieve the stored target data.

See Also “`CWGetTargetStorage`” on page 154
 “`CWSetTargetStorage`” on page 164
 “`CWGetPluginRequest`” on page 66
 “`CWDonePluginRequest`” on page 45

reqTargetPrefsChanged

Description Tells the plug-in that the active target’s settings data has changed.

Definition `#include <DropInCompilerLinker.h>`

Remarks Upon receiving a `reqTargetPrefsChanged`, a plug-in retrieves its target-related data with `CWGetTargetStorage`. After updating its data, the plug-in should return the new data to the IDE by calling `CWSetTargetStorage`. If the plug-in’s target-related data does not depend on target settings, the plug-in can ignore this request.

The IDE sends this request only if the plug-in has set the `kCompUsesTargetStorage` or `linkerUsesTargetStorage` flag in its ‘Flag’ resource.

After completing a request, the plug-in should call `CWDonePluginRequest` before returning execution to the IDE.

See Also “‘Flag’ Resource” in the *IDE SDK Developer’s Guide*
 “`CWSetTargetStorage`” on page 164
 “`CWGetTargetStorage`” on page 154
 “`CWGetPluginRequest`” on page 66
 “`CWDonePluginRequest`” on page 45

reqTargetUnloaded

Description Tells the plug-in to release its target storage.



Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

targetCPU68K

Definition	#include <DropInCompilerLinker.h>
Remarks	The IDE sends this request when switching targets, to tell the plug-in to release its target storage. The IDE sends this request only if the plug-in has set the kCompUsesTargetStorage or linkerUsesTargetStorage flag in its 'Flag' resource. Upon receiving a reqTargetUnloaded request, a plug-in retrieves its target-related data by calling CWGetTargetStorage. If the target-related data was allocated using CWAllocateMemory, the plug-in should call CWFreeMemory to release it. The plug-in should then call CWSetTargetStorage with a NULL target storage pointer to ensure that the stale target storage pointer is not subsequently used by the plug-in.
See Also	“'Flag' Resource” in the <i>IDE SDK Developer's Guide</i> “CWGetTargetStorage” on page 154 “CWSetTargetStorage” on page 164 “CWGetPluginRequest” on page 66 “CWDonePluginRequest” on page 45

targetCPU68K

Description	Specifies that the linker generates software for MC680x0 processors.
Definition	#include <DropInCompilerLinker.h>
Remarks	This constant appears in the targetCPU field of a CWTargetInfo structure when calling CWGetTargetInfo or CWSetTargetInfo to specify that a linker generates software for a MC680x0 processor. It may also be returned in the CWTargetList structure by the CWPlugin_GetDefaultMappingList Entry Point.
See Also	“CWTargetInfo” on page 183 “CWGetTargetInfo” on page 153 “CWSetTargetInfo” on page 163

targetCPUAny

Description	Specifies that the linker generates output that is not processor-specific.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in the <code>targetCPU</code> field of a <code>CWTargetInfo</code> structure when calling <code>CWGetTargetInfo</code> or <code>CWSetTargetInfo</code> to specify that a linker generates output for any processor. It may also be returned in the <code>CWTargetList</code> structure by the <code>CWPlugin_GetDefaultMappingList</code> Entry Point.
See Also	“CWTargetInfo” on page 183 “CWGetTargetInfo” on page 153 “CWSetTargetInfo” on page 163

targetCPUEmbeddedPowerPC

Description	Specifies that the linker generates software for embedded PowerPC processors.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in the <code>targetCPU</code> field of a <code>CWTargetInfo</code> structure when calling <code>CWGetTargetInfo</code> or <code>CWSetTargetInfo</code> to specify that a linker generates software for embedded PowerPC processors. It may also be returned in the <code>CWTargetList</code> structure by the <code>CWPlugin_GetDefaultMappingList</code> Entry Point.
See Also	“CWTargetInfo” on page 183 “CWGetTargetInfo” on page 153 “CWSetTargetInfo” on page 163

targetCPUi80x86

Description	Specifies that the linker generates software for an Intel x86-compatible processor.
Definition	<code>#include <DropInCompilerLinker.h></code>

Remarks This constant appears in the `targetCPU` field of a `CWTargetInfo` structure when calling `CWGetTargetInfo` or `CWSetTargetInfo` to specify that a linker generates software for an Intel x86-compatible processor. It may also be returned in the `CWTargetList` structure by the `CWPlugin_GetDefaultMappingList` Entry Point.

See Also “`CWTargetInfo`” on page 183
“`CWGetTargetInfo`” on page 153
“`CWSetTargetInfo`” on page 163

targetCPUMips

Description Specifies that the linker generates software for a MIPS processor.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in the `targetCPU` field of a `CWTargetInfo` structure when calling `CWGetTargetInfo` or `CWSetTargetInfo` to specify that a linker generates software for a MIPS processor. It may also be returned in the `CWTargetList` structure by the `CWPlugin_GetDefaultMappingList` Entry Point.

See Also “`CWTargetInfo`” on page 183
“`CWGetTargetInfo`” on page 153
“`CWSetTargetInfo`” on page 163

targetCPUNECv800

Description Specifies that the linker generates software for a NEC v800 processor.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in the `targetCPU` field of a `CWTargetInfo` structure when calling `CWGetTargetInfo` or `CWSetTargetInfo` to specify that a linker generates software for a NEC v800 processor. It may also be returned in the `CWTargetList` structure by the `CWPlugin_GetDefaultMappingList` Entry Point.

See Also “`CWTargetInfo`” on page 183



Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

targetCPUPowerPC

“CWGetTargetInfo” on page 153

“CWSetTargetInfo” on page 163

targetCPUPowerPC

Description	Specifies that the linker generates software for a desktop PowerPC processor.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in the <code>targetCPU</code> field of a <code>CWTargetInfo</code> structure when calling <code>CWGetTargetInfo</code> or <code>CWSetTargetInfo</code> to specify that a linker generates software for a desktop PowerPC processor. It may also be returned in the <code>CWTargetList</code> structure by the <code>CWPlugin_GetDefaultMappingList</code> Entry Point.
See Also	“CWTargetInfo” on page 183 “CWGetTargetInfo” on page 153 “CWSetTargetInfo” on page 163

targetOSAny

Description	Specifies that the linker generates output that is not for a specific operating system.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in the <code>targetCPU</code> field of a <code>CWTargetInfo</code> structure when calling <code>CWGetTargetInfo</code> or <code>CWSetTargetInfo</code> to specify that a linker generates output that isn’t for a specific operating system. It may also be returned in the <code>CWTargetList</code> structure by the <code>CWPlugin_GetDefaultMappingList</code> Entry Point.
See Also	“CWTargetInfo” on page 183 “CWGetTargetInfo” on page 153 “CWSetTargetInfo” on page 163

targetOSEmbeddedABI

Description	Specifies that the linker generates software that conforms to an embedded system's application binary interface.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in the <code>targetCPU</code> field of a <code>CWTargetInfo</code> structure when calling <code>CWGetTargetInfo</code> or <code>CWSetTargetInfo</code> to specify that a linker generates software that conforms to an embedded system's application binary interface. It may also be returned in the <code>CWTargetList</code> structure by the <code>CWPlugin_GetDefaultMappingList</code> Entry Point.
See Also	"CWTargetInfo" on page 183 "CWGetTargetInfo" on page 153 "CWSetTargetInfo" on page 163

targetOSMacintosh

Description	Specifies that the linker generates software for the Mac OS.
Definition	<code>#include <DropInCompilerLinker.h></code>
Remarks	This constant appears in the <code>targetCPU</code> field of a <code>CWTargetInfo</code> structure when calling <code>CWGetTargetInfo</code> or <code>CWSetTargetInfo</code> to specify that a linker generates software for the Mac OS. It may also be returned in the <code>CWTargetList</code> structure by the <code>CWPlugin_GetDefaultMappingList</code> Entry Point.
See Also	"CWTargetInfo" on page 183 "CWGetTargetInfo" on page 153 "CWSetTargetInfo" on page 163

targetOSMagicCap

Description	Specifies that the linker generates software for General Magic's Magic Cap operating system.
Definition	<code>#include <DropInCompilerLinker.h></code>

Compiler and Linker Plug-in Reference

targetOSOS9

Remarks This constant appears in the `targetCPU` field of a `CWTargetInfo` structure when calling `CWGetTargetInfo` or `CWSetTargetInfo` to specify that a linker generates software for Magic Cap. It may also be returned in the `CWTargetList` structure by the `CWPlugin_GetDefaultMappingList` Entry Point.

See Also “`CWTargetInfo`” on page 183
 “`CWGetTargetInfo`” on page 153
 “`CWSetTargetInfo`” on page 163

targetOSOS9

Description Specifies that the linker generates software for Microware’s OS-9 operating system.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in the `targetCPU` field of a `CWTargetInfo` structure when calling `CWGetTargetInfo` or `CWSetTargetInfo` to specify that a linker generates software for Microware’s OS-9 operating system. It may also be returned in the `CWTargetList` structure by the `CWPlugin_GetDefaultMappingList` Entry Point.

See Also “`CWTargetInfo`” on page 183
 “`CWGetTargetInfo`” on page 153
 “`CWSetTargetInfo`” on page 163

targetOSWindows

Description Specifies that the linker generates software for the Microsoft Windows operating system.

Definition `#include <DropInCompilerLinker.h>`

Remarks This constant appears in the `targetCPU` field of a `CWTargetInfo` structure when calling `CWGetTargetInfo` or `CWSetTargetInfo` to specify that a linker generates software for the Microsoft Windows operating system. It may also be returned in the `CWTargetList` structure by the `CWPlugin_GetDefaultMappingList` Entry Point.



See Also “CWTargetInfo” on page 183
 “CWGetTargetInfo” on page 153
 “CWSetTargetInfo” on page 163

Result Codes for Compiler and Linker Plug-ins

There are two compiler and linker specific error codes:

- `cwErrObjectFileNotStored`
- `cwErrUnknownSegment`

NOTE This section provides literal values only for reference. These values do not appear in the headers. Always use constants rather than the literal values in plug-in code.

`cwErrObjectFileNotStored`

Description	No object code exists for this file.
Definition	<pre>#include <CWPluginErrors.h> enum { /* ... */ cwErrObjectFileNotStored = 515, /* ... */ };</pre>
Remarks	Returned by



Freescale Semiconductor, Inc.

Compiler and Linker Plug-in Reference

cwErrUnknownSegment

cwErrUnknownSegment

Description	The plug-in has specified a segment that does not exist.
Definition	<pre>#include <CWPluginErrors.h> enum { /* ... */ cwErrUnknownSegment = 513, /* ... */ };</pre>
Remarks	CWGetSegmentInfo returns this error when the index supplied is out of range.

Browser Reference

This section documents the data structures and predefined symbols used in the browser data for the CodeWarrior IDE class browser.

- Browser Records—describes the format of a browser record, the unit used to describe a browsable item in source code.
- Data Structures for the Browser—describes the data structures used by the API to format data for the class browser.
- Constants for the Browser—describes the constant values used by the API to support the class browser.

For information on how to store browser data, see the *IDE SDK Developer's Guide*.

TIP To examine a plug-in's browser data output for debugging purposes, use the **Dump internal browse information after compile** checkbox in the **Build Extras** settings panel.

Browser Records

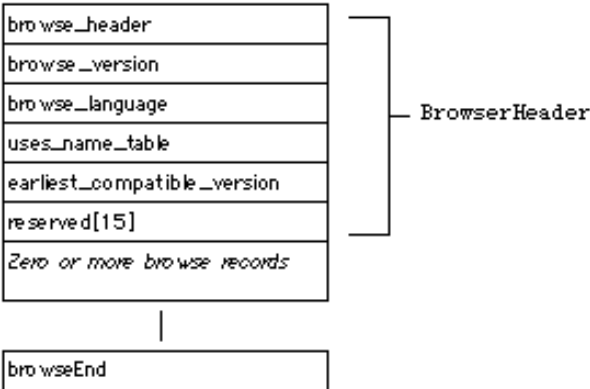
Compilers store browser data as a sequence of variable size records. This data is stored in the `browsedata` field of the `CWObjectData` structure passed to the IDE when calling `CWStoreObjectData`. Browser data describes the symbols found by a compiler during compilation of source text. The IDE displays this information in the **Browser Contents**, **Class Hierarchy**, and **Class Browser** windows, accessible under the **Windows** menu.

NOTE The IDE enables browser capabilities only when **Activate Browser** is checked in **Target Settings > Build Extras**.

Browse Header

The browser data stream starts with a `BrowseHeader`, which describes the browser data to the IDE.

Figure 5.1 Browser data organization



The following table describes the browser data fields:

Field	Description
<code>browse_header</code>	4 bytes—Contains the arbitrary value <code>BROWSE_HEADER</code> , which indicates to the IDE that the browser data stream is valid.
<code>browse_version</code>	4 bytes—Indicates the version of the browser data stream. This number changes as the browser data stream format is revised. Plug-ins should normally specify the latest stream format, using <code>BROWSE_VERSION</code> .
<code>browse_language</code>	2 bytes—Specifies the language of the source file from which the browser symbol information was derived. See <code>ELanguage</code> for valid values.
<code>uses_name_table</code>	2 bytes—Reserved for Metrowerks internal use. Plug-ins should always set this field to zero.
<code>earliest_compatible_version</code>	

Field	Description
	4 bytes—This field specifies the earliest browser stream format that this browser data is compatible with. Plug-ins should normally store the value <code>BROWSE_EARLIEST_COMPATIBLE_VERSION</code> in this field.
reserved	60 bytes—15 four-byte integers reserved for future use. Plug-ins should set all 15 to zero.

NOTE The value of the `earliest_compatible_version` field typically matches the value specified in `browse_version`.

The browser data stream ends with a single byte containing the value `browseEnd`. `browseEnd` is defined in the `EBrowserItem` enumeration.

Browser Data Stream

As illustrated in Figure 5.1, the header is followed by one variable-length record for each browsable item. Symbol records need not be presented in any particular order.

Browser records for individual symbols have variant definition, depending upon the symbol type. All symbol records start with common fields, with type-specific fields following.

NOTE Browser data consists of records of variable size, without padding. Thus, the alignment of browser information varies and need not be aligned on host CPU word boundaries.

The rest of this section discusses:

- Fields For All Records
- Additional Fields For Functions
- Additional Fields For Classes

Browser Reference
Fields For All Records

- Additional Field For Templates
- Additional Field For Global Variables

Fields For All Records

The common fields for all records describe the symbol’s type, its point of declaration, and its name.

Figure 5.2 A browser record

type
contribFile
srcFile
startOffset
endOffset
reserved
simple name length
simple name
qualified name length
qualified name
<i>Additional fields if type is browseFunction , browse- Class , browseTemplate , or browseGlobal</i>

Field	Description
type	1 byte—Every record begins with the <i>type</i> field. This field specifies the type of item being described. The valid types are described in “EBrowserItem” on page 246. This field can also be set to a custom type, if a plug-in generates symbol records for nonstandard types of symbols. Custom type values should be assigned consecutively beginning with <code>browseCompSymbolStart</code>. See the <i>IDE SDK Developer’s Guide</i> for more information.
contribFile	2 bytes—The file number of the file that should be considered the contributor of the item. For all record types except templates, this should be the same file as specified in <i>srcFile</i>. This must be a valid file number as returned by <code>CWFindAndLoadFile</code>. <code>contribFile</code> specifies the file contributing a fully-specified declaration of this syntactic entity. In the case of templates, this is the file in which the template was instantiated, rather than the header in which the generic template definition appears. The IDE uses this field when updating its internal browser information database. If <code>contribFile</code> is recompiled, the IDE will look for any items originally contributed by <code>contribFile</code> which don’t appear in the newly generated browser data stream. These items will be removed from the IDE’s database.
srcFile	2 bytes—The file number of the file containing the item’s declaration. This must be a valid file number returned by <code>CWFindAndLoadFile</code>.
startOffset	4 bytes—The zero-based offset of the start of the item’s location in <i>srcFile</i>. Use -1 to indicate an unknown or not-applicable offset.

Browser Reference

Fields For All Records

Field	Description
endOffset	4 bytes—The zero-based offset of the start of the item's location in <i>srcFile</i> . Use -1 to indicate an unknown or not-applicable offset.
reserved	4 bytes—Reserved for future use, must be zero.
simple name length	2 bytes—Length of following simple item name, excluding the terminating null character.
simple name	variable length—The simple, unqualified name of the item as a variable-length, null-terminated C string. The length of this string is specified in <i>simple name length</i> . The simple name of an item is the identifier by which it is normally specified in its parent language. It is the name a programmer would type to refer to the item.
qualified name length	2 bytes—Length of following qualified item name, excluding the terminating null character. Can be zero.
qualified name	variable length—The qualified name of the item as a variable-length, null-terminated C string. The length of this string is specified in <i>qualified name length</i> . This string is not stored if the <i>qualified name length</i> is zero.

For an example of how simple names relate to qualified names, consider the following declaration:

Listing 5.1 Simple name versus qualified name example

```

class CExampleClass
{
public:
    CExampleClass ();
    virtual ~CExampleClass ();

    int test (int A);
    int test (int A, int B);
};

```

The simple, unqualified name of both methods named `test` is `"test"`. As this example shows, simple names need not be unique.

The qualified name of an item specifies it uniquely, usually by including scope information, and any necessary distinguishing information within its scope. In listing 5.1, the first of the two overloaded `test` methods might have a qualified name of `"CExampleClass::test (int A)"`. The second `Foo` method might have a qualified name of `"CExampleClass::test (int A, int B)"`. The qualified names could also be "mangled" names.

The IDE places no restrictions on the content of a qualified name, but it should uniquely identify its associated symbol. The IDE uses the qualified name when searching for symbols in its internal browser symbol database, such as when adding and deleting symbols from the database. Qualified names for the same symbol should be consistent across compiles.

If type is `browseMacro`, `browseEnum`, `browseConstant`, `browseTypedef`, `browsePackage`, or a compiler-specific browser symbol, then no additional fields are required for the browser record.

If type is `browseFunction`, `browseClass`, `browseTemplate`, or `browseGlobal`, then the record requires additional data, described here.

- "Additional Fields For Functions" on page 236—Additional fields for the `browseFunction` type of browser record.
- "Additional Fields For Classes" on page 237—Additional fields for the `browseClass` type of browser record.
- "Additional Field For Templates" on page 242—Additional fields for the `browseTemplate` type of browser record.
- "Additional Field For Global Variables" on page 242—Additional fields for the `browseGlobal` type of browser record.

Additional Fields For Functions

Figure 5.3 A browse record for a function

Fields For All Records, type is set to browseFunction

flags
ID

For a list of fields required for all browser records, see “Fields For All Records” on page 232.

The following table describes additional data for browseFunction records:

Field	Description
flags	4 bytes—Specifies properties of the function. See listing 5.2 for the available flags. Flags may be combined.
ID	4 bytes—The member function ID for this function, so it can be matched with the class’ member function information. Store zero if this is not a member function.

Listing 5.2 Function flags listing

```
enum
{
    kStatic = 2,          /* Static routine          */
    kMember = 8,         /* Class member function */
    kInline = 0x80,      /* In line routine        */
    kPascal = 0x100,     /* Pascal routine         */
    kAsm    = 0x200,     /* Assembly routine       */
};
```

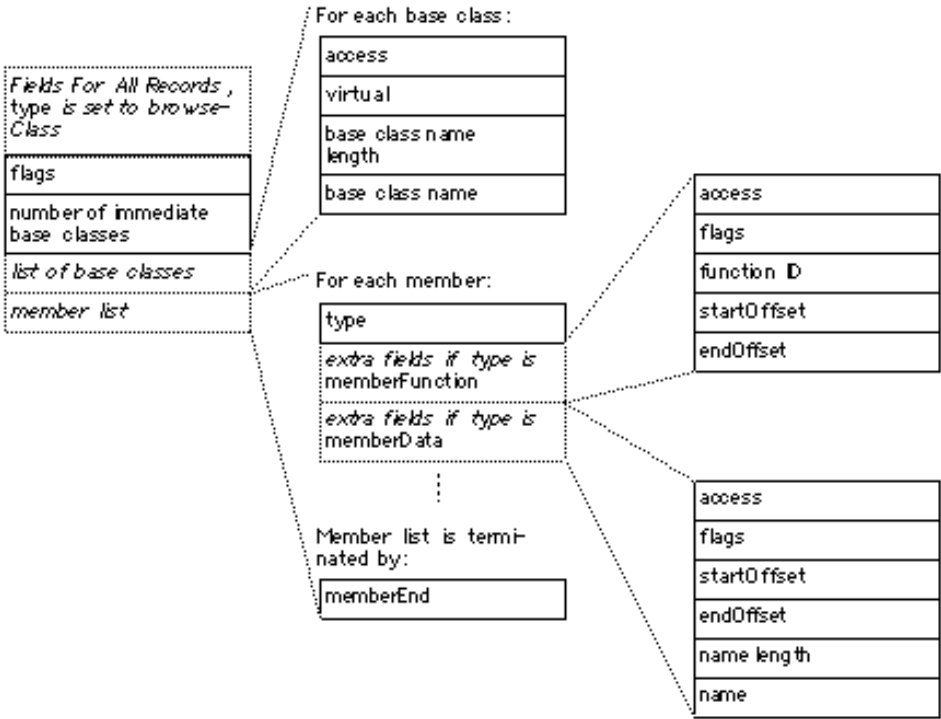
Member function IDs uniquely identify the member functions of classes and appear in the Additional Fields For Classes. Member function IDs should be assigned starting with 1 and incremented for each subsequent member function. However, the order does not matter, and the numbers may change from compilation to the next.

Additional Fields For Classes

Browser records for classes include additional information, including superclasses, data members, member functions, member access, and the source locations of member declarations.

As illustrated in Figure 5.4, the browse data for a class starts with the usual information common to all browser records. Subsequent portions of the data include fixed-size class information, a variable-length base class list, and a variable-length member list.

Figure 5.4 A browse record for a class



NOTE The count provided in the class information determines the base class list length, The presence of the `memberEnd` sentinel value determines the length of the member list.

Browser Reference

Additional Fields For Classes

For a list of fields required for all browser records, see “Fields For All Records” on page 232.

The additional data for browseClass records includes:

Field	Description
flags	4 bytes—Specifies properties of the class as a whole. See listing 5.3 for valid flag values. Flags can be combined.
number of immediate base classes	1 byte—Specifies the number of classes this class inherits from.

Listing 5.3 shows the enumeration that defines the class flags data:

Listing 5.3 Class flags listing

```
enum
{
    kAbstract    = 1,           /* Abstract class */
    kFinal       = 4,           /* Final class */
    kInterface   = 0x80,        /* Java interface */
    kPublic      = 0x100        /* Public Java class */
};
```

The count of base classes is immediately followed by one record for each class. Base class information should be omitted if there are no base classes. The following table shows the data contained in base class records:

Field	Description
access	1 byte—Specifies the scoping relationship between this base class and the class inheriting from it. For valid values, see “EAccess” on page 245.
virtual	1 byte—Specifies whether this base class is virtual. True if this is a virtual base class.

Field	Description
base class name length	2 bytes—Specifies the length of the following base class name, excluding the terminating null character.
base class name	variable length—Specifies the mangled or qualified name of this base class, terminated with a null character.

Class Member List

For each member of the class, one class member record should follow the base class records (if any). The class member record varies according to the member type (method or data member).

The member list for each class contains a variable-length list of class members. The member list should contain only the data members and methods declared by the current class. The member list should not include inherited members and methods.

Each member record contains the following:

Field	Description
type	1 byte—The type of class member. For valid values, see “EMember” on page 248. When type is <code>memberFunction</code> or <code>memberData</code> , additional data follows. A type value of <code>memberEnd</code> signals the end of the member list for the current class.
The additional data for <code>memberFunction</code> is:	
access	1 byte—Specifies the scope of this member function. For valid values, see “EAccess” on page 245.
flags	4 bytes—Specifies properties of this member function. See listing 5.4 for valid flag values. Flags may be combined.

Browser Reference

Additional Fields For Classes

Field	Description
function ID	4 bytes—The ID of this member function. The IDE uses member function IDs to match function records found elsewhere in the browser data stream with method entries in class records. Method entries in class information minimally describe a method, and “point” to additional information in a corresponding function record using the function ID. Function IDs must be unique and non-zero. Function IDs should be assigned to method functions – and their corresponding class method entries – starting with 1, incrementing sequentially. Function IDs can vary from compile to compile. Functions referred to by ID in a class member function entry need not appear in the browser stream before the class record.
startOffset	4 bytes—Zero-based offset from the start of the class declaration file (the <code>srcFile</code> specified for the class). The offset specifies the location of the member function’s declaration within its class declaration.
endOffset	4 bytes—Zero-based offset from the start of the class declaration file (<code>srcFile</code>) of the end of the method’s declaration. If the member function is defined inline in the class declaration, then the range includes the definition.
The additional data for <code>memberData</code> is:	
access	1 byte—Specifies the scope of this data member. For valid values, see “EAccess” on page 245.
flags	4 bytes—Specifies properties of this data member. See listing 5.5 for valid flag values. Flags may be combined
startOffset	4 bytes—Zero-based offset from start of the class declaration file (<code>srcFile</code>) to the beginning of the data member declaration.

Field	Description
endOffset	4 bytes—Zero-based offset from start of the class declaration file (srcFile) to the end of the data member declaration.
name length	2 bytes—Length of following name, not including the terminating null character.
name	Variable length—Name of data member, terminated with a null character.

Listing 5.4 shows the enumeration that defines member function flags.

Listing 5.4 Member function flags listing

```
enum
{
    kAbstract= 1,          /* Abstract/pure virtual */
    kStatic= 2,            /* Static member or function */
    kFinal= 4,             /* Final Java class/method/member */
    kVirtual= 0x400,       /* Virtual member function */
    kCtor= 0x800,          /* Is constructor */
    kDtor= 0x1000,         /* Is destructor */
    kNative= 0x2000,       /* Native Java method */
    kSynch= 0x4000,        /* Synchronized Java method */
};
```

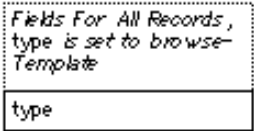
Listing 5.5 shows the enumeration that defines class flags.

Listing 5.5 Class flags listing

```
enum
{
    kStatic = 2,           /* static member */
    kFinal= 4,             /* final Java class/method/member */
    kTransient = 0x80,     /* transient Java member */
    kVolatile = 0x100,     /* volatile Java or C++ member */
};
```

Additional Field For Templates

Figure 5.5 A browse record for templates

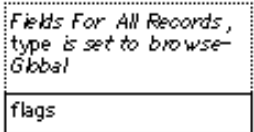


For a list of fields required for all browser records, see “Fields For All Records” on page 232.

The additional data for records of type `browseTemplate` consists of a single byte, called `type`. This value specifies the type of the template, as either a class or a function. For valid values, see “ETemplateType” on page 249.

Additional Field For Global Variables

Figure 5.6 A browse record for a global variable



For a list of fields required for all browser records, see “Fields For All Records” on page 232.

Global variables require one additional field, called `flags`. This field contains four bytes. The four bytes contain a bit flag that specify whether the global variable is static. See listing 5.6 for valid flag values.

Listing 5.6 Global variable types listing

```
enum
{
    kStatic = 2,                /* Static global */
};
```



Data Structures for the Browser

The plug-in API header 'MWBrowse.h' defines the BrowseHeader data structure. The IDE uses the BrowseHeader structure when generating browser data:

BrowseHeader

Description Describes a browser data stream.

Prototype

```
#include <MWBrowse.h>
typedef struct BrowseHeader {
    long        browse_header;
    long        browse_version;
    short       browse_language;
    short       uses_name_table;
    long        earliest_compatible_version;
    long        reserved[15];
} BrowseHeader;
```

Fields The following table describes the fields in BrowseHeader:

browse_header	Contains the value of BROWSE_HEADER. This field contains distinctive value used by the IDE to ensure that the browser data stream is valid.
browse_version	Indicates the format of the browser data stream. Currently, compilers should store BROWSE_VERSION in this field.
browse_language	Specifies the language of the source file for which the browser data was generated. "ELanguage" on page 247 lists valid language values.
uses_name_table	If the compiler stores true in this field, the compiler uses the CodeWarrior object code name table. Non-Metrowerks compilers should set this field to false.

Browser Reference

Constants for the Browser

<code>earliest_compatible_version</code>	Specifies the earliest version of the browser data stream format with which this data is compatible. Plug-ins should use <code>BROWSE_EARLIEST_COMPATIBLE_VERSION</code> for this field, which will usually be the same as <code>BROWSE_VERSION</code> .
<code>reserved[15]</code>	Reserved. Set this value to zero.

- Remarks All browser data streams must start with a single `BrowseHeader`. The header describes properties of the stream to the IDE.
- A valid browser data stream starts with a single `BrowseHeader`, followed by additional variable-length data records. Each record describes a single browsable item (class, function, global variable, or other browser-supported source element).
- See Also “Browser Records” on page 229
- “BrowseHeader” on page 243

Constants for the Browser

This section documents the constants and enumerations defined in `'MWBrowse.h'`:

- `BROWSE_EARLIEST_COMPATIBLE_VERSION`
- `BROWSE_HEADER`
- `BROWSE_VERSION`
- `EAccess`
- `EBrowserItem`
- `ELanguage`
- `EMember`
- `ETemplateType`

BROWSE_EARLIEST_COMPATIBLE_VERSION

- Description Specifies the earliest version of the browser stream format with which this data is compatible.

Prototype `#include <MWBrowse.h>`
Remarks A compiler stores this value in the `earliest_compatible_version` field in the `BrowseHeader` data structure. Usually, the value of this define matches the value of `BROWSE_VERSION`.

BROWSE_HEADER

Description An arbitrary 64-bit signature, marking the start of a browse data stream, used by the IDE to ensure the validity of browser data.
Prototype `#include <MWBrowse.h>`
Remarks A compiler stores this value used in the `browser_header` field in the `BrowseHeader` data structure.

BROWSE_VERSION

Description Specifies the version of the browser data at compile time.
Prototype `#include <MWBrowse.h>`
Remarks This predefined symbol specifies the current version of the browser API. At compile time use this symbol to determine which data structures and constants are available at compile time.

 Also, store this value in the `browse_version` field of a `BrowseHeader` data structure.

EAccess

Description Specifies the types of access to a class member.
Prototype `#include <MWBrowse.h>`
Values The following table describes the meanings of the enumerated values in `EAccess`:

<code>accessNone</code>	The item isn't accessible. Plug-ins rarely use this setting.
<code>accessPrivate</code>	The item is accessible only from other members in the same class, but not in its descendents.

Browser Reference

EBrowserItem

accessProtected	The item is accessible from other members in the same class and from members in derived classes.
accessPublic	The item is accessible from both inside and outside the class.
accessAll	The item can be accessed in any way (private, protected, and public). This value usually serves as a mask for the bits specifying member access, not as a valid access value.

See Also “Additional Fields For Classes” on page 237

EBrowserItem

Description Specifies the type of item described in a browser record.

Prototype `#include <MWBrowse.h>`

Values The following table describes the enumerated values in EBrowserItem:

browseFunction	The browser record describes a function, procedure, or class method.
browseGlobal	The browser record describes a global variable.
browseClass	The browser record describes a class, struct, union, or Java interface.
browseMacro	The browser record describes a macro.
browseEnum	The browser record describes an enumerated type.
browseTypedef	The browser record describes a user-defined type other than a class, struct, union, or Java interface (usually a scalar, pointer, or array type).

browseConstant	The browser record describes a constant definition.
browseTemplate	The browser record describes a C++ template.
browsePackage	The browser record describes a Java package.
browseCompSymbolStart	Use this as the first value to describe a compiler-specific browser item in a browser record. Subsequent types of compiler-specific browser items should use the values browseCompSymbolStart+1, browseCompSymbolStart+2, and so on. See the <i>IDE SDK Developer's Guide</i> for more information on generating compiler-specific browser symbols. Records for custom compiler-specific symbol types contain no additional fields beyond those described in "Fields For All Records" on page 232.
browseEnd	Use this value to mark the end of the list of browser records.

Remarks The `EBrowserItem` enumeration lists the kinds of items that can be described in a browser record. All browser records start with an `EBrowserItem` value, which specifies the layout of the data that follows.

See Also "Browser Records" on page 229

ELanguage

Description Used in the browse data header to specify source language type.

Prototype `#include <MWLangDefs.h>`

Values The following table describes the enumerated values in `ELanguage`:

Browser Reference

EMember

langUnknown	Use langUnknown when there is no other more appropriate value in the ELanguage enumeration.
langC	Specifies C source code.
langCPlus	Specifies C++ source code.
langPascal	Specifies Pascal source code.
langObjectPascal	Specifies Object Pascal source code.
langJava	Specifies Java source code.
langAssembler	Specifies assembly language source code (regardless of processor type).
langFortran	Specifies FORTRAN source code.
langRez	Specifies a textual source file for resources, such as a Mac OS Rez file.

Remarks The ELanguage enumeration lists the languages that the CodeWarrior IDE browser supports. Use a value from ELanguage in the browse_language field of a BrowseHeader structure.

If the IDE does not directly support the language of a compiler, use the nearest match.

EMember

Description Describes the type of a class member.

Prototype

```
#include <MWBrowse.h>
typedef unsigned char EMember;
```

Values The enumerated values in EAccess are:

memberFunction	The member is a routine or method.
memberData	The member is a data field.
memberEnd	A value used to mark the end of a member list, rather than to specify the type of a member record.

- Remarks The EMember enumeration specifies the type of a class member. Use this enumeration to describe records in a member list for a base class.
- See Also “The member list for each class contains a variable-length list of class members. The member list should contain only the data members and methods declared by the current class. The member list should not include inherited members and methods.” on page 239

ETemplateType

- Description Describes a C++ template type.
- Prototype

```
#include <MWBrowse.h>
typedef enum ETemplateType
```
- Values The enumerated values in ETemplateType are:

templateClass	The template describes a class.
templateFunction	The template describes a function.
- Remarks The ETemplateType enumeration lists the types of C++ templates. Browser records for templates include a single byte of template-specific information containing a value of type ETemplateType, which specifies the template type (class or function).
- See Also “Additional Field For Templates” on page 242



Freescale Semiconductor, Inc.

Browser Reference
ETemplateType

Settings Panel Plug-in API Reference

This chapter describes the plug-in API services available to settings panel plug-ins. The reference material in this chapter contains the following sections:

- Routines for Settings Panel Plug-ins
- User Routines for Settings Panel Plug-ins
- Data Structures for Settings Panel Plug-ins
- Constants for Settings Panel Plug-ins
- Settings Panel Result Codes

NOTE In many places, this chapter refers to the **Target Settings** dialog box. In fact, panel plug-ins can appear in any of three dialogs: **Preferences**, **Target Settings**, and **Version Control Settings** (all in the Edit menu). For convenience, this list refers only to **Target Settings**, because most panel plug-ins appear in this dialog box.

Routines for Settings Panel Plug-ins

This section describes all routines available to settings panel plug-ins.

Organization of Function Reference

Because the API routines for both Windows and Mac OS plug-ins differ in declaration but have similar function, this section of the reference is organized slightly differently from other reference sections.

Settings Panel Plug-in API Reference

Plug-in Context

The organizational differences directly reflect differences in the Windows and Mac OS versions of the settings panel plug-in API. These differences include the following:

1. Windows routines, except for XML routines, start with a "CWPanel" prefix. Windows XML routines start with a "CW" prefix. All Mac OS routines start with a "CWPan1" prefix.
2. The plug-in state parameter passed to all routines (context) is of type CWPluginContext on Windows and type PanelParamBlkPtr on the Mac OS.
3. Windows routines use handles of type CWMemHandle. Mac OS routines use Mac OS toolbox handles of type Handle.

Because similarly named Windows and Mac OS routines are functionally identical, the Mac OS routine entries simply refer to their Windows counterparts.

NOTE DropInPanel.h (which was for the Mac OS) and DropInPanelWin32.h have been deprecated. A cross-platform header file, CWDropInPanel.h, has superseded the header files. The deprecated header files remain valid, for backwards compatibility.

Plug-in Context

All settings panel routines provided by the CodeWarrior IDE require a value of type CWPluginContext (on Windows) or of type PanelParamBlkPtr (on Mac OS) to be passed as the first parameter.

Windows	See "CWPluginContext" on page 108 and "Main Entry Point" in the <i>IDE SDK Developer's Guide</i> for more information.
Mac OS	See "PanelParamBlkPtr" on page 336 and "PanelParameterBlock" on page 336 for more information.

Alphabetical Routine Index

- CWGetArraySettingElement
- CWGetArraySettingSize
- CWGetBooleanValue



Freescal Semiconductor, Inc.

Settings Panel Plug-in API Reference

Alphabetical Routine Index

-
- CWGetFloatingPointValue
 - CWGetIntegerValue
 - CWGetNamedSetting
 - CWGetRelativePathValue
 - CWGetStringValue
 - CWGetStructureSettingField
 - CWPanelActivateItem
 - CWPanelAppendItems
 - CWPanelChooseRelativePath
 - CWPanelDeleteListItem
 - CWPanelEnableItem
 - CWPanelGetCurrentPrefs
 - CWPanelGetDebugFlag
 - CWPanelGetDialogItemHit
 - CWPanelGetFactoryPrefs
 - CWPanelGetItemData
 - CWPanelGetItemMaxLength
 - CWPanelGetItemText
 - CWPanelGetItemTextHandle
 - CWPanelGetItemValue
 - CWPanelGetListItemText
 - CWPanelGetNumBaseDialogItems
 - CWPanelGetOriginalPrefs
 - CWPanelGetPanelPrefs
 - CWPanelGetRelativePathString
 - CWPanelInsertListItem
 - CWPanelInvalItem
 - CWPanelSetFactoryFlag
 - CWPanelSetItemData
 - CWPanelSetItemMaxLength
 - CWPanelSetItemText

Settings Panel Plug-in API Reference

Alphabetical Routine Index

- CWPanelSetItemTextHandle
- CWPanelSetItemValue
- CWPanelSetListItemText
- CWPanelSetRecompileFlag
- CWPanelSetRelinkFlag
- CWPanelSetReparsingFlag
- CWPanelSetResetPathsFlag
- CWPanelSetRevertFlag
- CWPanelShowItem
- CWPanelValidItem
- CWPanelActivateItem
- CWPanelAppendItems
- CWPanelChooseRelativePath
- CWPanelDrawPanelBox
- CWPanelDrawUserItemBox
- CWPanelEnableItem
- CWPanelGetArraySettingElement
- CWPanelGetArraySettingSize
- CWPanelGetBooleanValue
- CWPanelGetFloatingPointValue
- CWPanelGetIntegerValue
- CWPanelGetItemControl
- CWPanelGetItemData
- CWPanelGetItemMaxLength
- CWPanelGetItemRect
- CWPanelGetItemText
- CWPanelGetItemTextHandle
- CWPanelGetItemValue
- CWPanelGetMacPort
- CWPanelGetNamedSetting
- CWPanelGetPanelPrefs



Freescale Semiconductor, Inc.

Settings Panel Plug-in API Reference

Alphabetical Routine Index

-
- `CWPanlGetRelativePathString`
 - `CWPanlGetRelativePathValue`
 - `CWPanlGetStringValue`
 - `CWPanlGetStructureSettingField`
 - `CWPanlInstallUserItem`
 - `CWPanlInvalItem`
 - `CWPanlReadBooleanSetting`
 - `CWPanlReadFloatingPointSetting`
 - `CWPanlReadIntegerSetting`
 - `CWPanlReadRelativePathAEDesc`
 - `CWPanlReadRelativePathSetting`
 - `CWPanlReadStringSetting`
 - `CWPanlRemoveUserItem`
 - `CWPanlSetBooleanValue`
 - `CWPanlSetFloatingPointValue`
 - `CWPanlSetIntegerValue`
 - `CWPanlSetItemData`
 - `CWPanlSetItemMaxLength`
 - `CWPanlSetItemText`
 - `CWPanlSetItemTextHandle`
 - `CWPanlSetItemValue`
 - `CWPanlSetRelativePathValue`
 - `CWPanlSetStringValue`
 - `CWPanlShowItem`
 - `CWPanlValidItem`
 - `CWPanlWriteBooleanSetting`
 - `CWPanlWriteFloatingPointSetting`
 - `CWPanlWriteIntegerSetting`
 - `CWPanlWriteRelativePathAEDesc`
 - `CWPanlWriteRelativePathSetting`
 - `CWPanlWriteStringSetting`

Settings Panel Plug-in API Reference

Functional Routine Index

- `CWReadBooleanSetting`
- `CWReadFloatingPointSetting`
- `CWReadIntegerSetting`
- `CWReadRelativePathSetting`
- `CWReadStringSetting`
- `CWSetBooleanValue`
- `CWSetFloatingPointValue`
- `CWSetIntegerValue`
- `CWSetRelativePathValue`
- `CWSetStringValue`
- `CWWriteBooleanSetting`
- `CWWriteFloatingPointSetting`
- `CWWriteIntegerSetting`
- `CWWriteRelativePathSetting`
- `CWWriteStringSetting`

Functional Routine Index

Manipulating Panel Controls

- `CWPanelAppendItems`, `CWPanlAppendItems`
- `CWPanelActivateItem`, `CWPanlActivateItem`
- `CWPanelEnableItem`, `CWPanlEnableItem`
- `CWPanelGetItemValue`, `CWPanlGetItemValue`
- `CWPanelSetItemValue`, `CWPanlSetItemValue`
- `CWPanelShowItem`, `CWPanlShowItem`
- `CWPanelInvalItem`, `CWPanlInvalItem`
- `CWPanelValidItem`, `CWPanlValidItem`
- `CWPanelGetItemData`, `CWPanlGetItemData`
- `CWPanelSetItemData`, `CWPanlSetItemData`

Getting and Setting Text Controls

- `CWPanelGetItemText`, `CWPanlGetItemText`
- `CWPanelSetItemText`, `CWPanlSetItemText`

- `CWPanelGetItemTextHandle`,
`CWPanelGetItemTextHandle`
- `CWPanelSetItemTextHandle`,
`CWPanelSetItemTextHandle`
- `CWPanelGetItemMaxLength`, `CWPanelGetItemMaxLength`
- `CWPanelSetItemMaxLength`, `CWPanelSetItemMaxLength`

Obtaining Settings Handles

- `CWPanelGetCurrentPrefs`
- `CWPanelGetOriginalPrefs`
- `CWPanelGetFactoryPrefs`
- `CWPanelGetPanelPrefs`, `CWPanelGetPanelPrefs`

Handling Panel Events (Windows)

- `CWPanelGetDialogItemHit`
- `CWPanelGetNumBaseDialogItems`

Manipulating Combo Box Item Lists (Windows)

- `CWPanelDeleteListItem`
- `CWPanelInsertListItem`
- `CWPanelGetListItemText`
- `CWPanelSetListItemText`

Obtaining IDE State (Windows)

- `CWPanelGetDebugFlag`
- `CWPanelSetFactoryFlag`
- `CWPanelSetRecompileFlag`
- `CWPanelSetRelinkFlag`
- `CWPanelSetReparseFlag`
- `CWPanelSetResetPathsFlag`
- `CWPanelSetRevertFlag`

Manipulating Controls (Mac OS)

- `CWPanelGetItemControl`
- `CWPanelGetItemRect`

Settings Panel Plug-in API Reference

Functional Routine Index

- `CWPanlGetMacPort`

Custom Controls (User Items) (Mac OS)

- `CWPanlInstallUserItem`
- `CWPanlRemoveUserItem`
- `CWPanlDrawPanelBox`
- `CWPanlDrawUserItemBox`

Manipulating Relative Paths

- `CWPanelChooseRelativePath,`
`CWPanlChooseRelativePath`
- `CWPanelGetRelativePathString,`
`CWPanlGetRelativePathString`
- `CWPanlReadRelativePathAEDesc`
- `CWPanlWriteRelativePathAEDesc`

Reading Primitive XML Types

- `CWReadBooleanSetting,` `CWPanlReadBooleanSetting`
- `CWReadFloatingPointSetting,`
`CWPanlReadFloatingPointSetting`
- `CWReadIntegerSetting,` `CWPanlReadIntegerSetting`
- `CWReadRelativePathSetting,`
`CWPanlReadRelativePathSetting`
- `CWReadStringSetting,` `CWPanlReadStringSetting`

Writing Primitive XML Types

- `CWWriteBooleanSetting,`
`CWPanlWriteBooleanSetting`
- `CWWriteFloatingPointSetting,`
`CWPanlWriteFloatingPointSetting`
- `CWWriteIntegerSetting,`
`CWPanlWriteIntegerSetting`
- `CWWriteRelativePathSetting,`
`CWPanlWriteRelativePathSetting`
- `CWWriteStringSetting,` `CWPanlWriteStringSetting`



Reading and Writing XML Structures and Arrays

- CWGetNamedSetting, CWPanlGetNamedSetting
- CWGetStructureSettingField, CWPanlGetStructureSettingField
- CWGetArraySettingElement, CWPanlGetArraySettingElement
- CWGetArraySettingSize, CWPanlGetArraySettingSize

Reading XML Structure and Array Elements

- CWGetBooleanValue, CWPanlGetBooleanValue
- CWGetFloatingPointValue, CWPanlGetFloatingPointValue
- CWGetIntegerValue, CWPanlGetIntegerValue
- CWGetRelativePathValue, CWPanlGetRelativePathValue
- CWGetStringValue, CWPanlGetStringValue

Writing XML Structure and Array Elements

- CWSetBooleanValue, CWPanlSetBooleanValue
- CWSetFloatingPointValue, CWPanlSetFloatingPointValue
- CWSetIntegerValue, CWPanlSetIntegerValue
- CWSetRelativePathValue, CWPanlSetRelativePathValue
- CWSetStringValue, CWPanlSetStringValue

CWGetArraySettingElement

Description	Returns one element of an XML array, by ID.
Prototype	<pre>#include <CWDropInPanel.h> CW_CALLBACK CWGetArraySettingElement(CWPluginContext context, CWSettingID settingID, long index, CWSettingID* elementSettingID);</pre>
Parameters	The parameters for this Method include:

Settings Panel Plug-in API Reference

CWGetArraySettingElement

context	Private, opaque IDE state.
settingID	Specifies the XML array containing the element to be returned.
index	A zero-based index specifying the item in the array to be returned.
elementSettingID	Returns the ID of the indexed array element, which may be simple type, or a structured type.

CWGetArraySettingElement returns an ID for a field of an XML structure. *CWGetArraySettingElement* behaves differently depending upon when it is called.

During a *reqReadSettings* request, this routine returns an error if a field having the specified name is not found. If the named field is found, the returned ID can be used to call one of the following routines to obtain the value of the field:

- *CWGetBooleanValue*
- *CWGetFloatingPointValue*
- *CWGetIntegerValue*
- *CWGetRelativePathValue*
- *CWGetStringValue*

If the field is another array, the plug-in should call *CWGetArraySettingSize* and *CWGetArraySettingElement* to read the elements of the array. If the field is a structure, the plug-in should instead call *CWGetStructureSettingField* to obtain the fields of the structure.

During a *reqReadSettings* request, to get the ID of a top-level XML array setting (that is, a setting whose parent container is implicitly the plug-in's XML stream), a plug-in calls *CWGetNamedSetting*. The plug-in can use the returned ID to call *CWGetArraySettingElement*.

During a *reqWriteSettings* request, this routine creates a new XML setting with the specified name if one has not already been created and return its ID. The plug-in can use the returned ID to call one of the following routines to set the value of the field:

- `CWSetBooleanValue`
- `CWSetFloatingPointValue`
- `CWSetIntegerValue`
- `CWSetRelativePathValue`
- `CWSetStringValue`

If the field to be written is instead another structure or array, the plug-in should call `CWGetStructureSettingField` or `CWGetArraySettingElement` to create a new setting. Both routines return an ID, which can be used in further calls to any of the above routines.

A plug-in creates a new top-level XML structure setting during a `reqWriteSettings` request by calling `CWGetNamedSetting`. The ID returned can be used to call `CWGetArraySettingElement`.

See Also [“CWGetStructureSettingField” on page 267](#)

[“CWGetArraySettingSize” on page 262](#)

[“CWGetNamedSetting” on page 264](#)

[“CWGetBooleanValue” on page 262](#)

[“CWGetFloatingPointValue” on page 263](#)

[“CWGetIntegerValue” on page 264](#)

[“CWGetRelativePathValue” on page 266](#)

[“CWGetStringValue” on page 266](#)

[“CWSetBooleanValue” on page 323](#)

[“CWSetFloatingPointValue” on page 324](#)

[“CWSetIntegerValue” on page 325](#)

[“CWSetRelativePathValue” on page 326](#)

[“CWSetStringValue” on page 327](#)

Settings Panel Plug-in API Reference

CWGetArraySettingSize

CWGetArraySettingSize

Description Returns the number of elements in an XML array.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetArraySettingSize(
    CWPluginContext context,
    CWSettingID settingID,
    long* size);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
settingID	Specifies the XML array containing the element to be returned.
size	Returns the number of elements in the specified array.

Remarks CWGetArraySettingSize returns the number of elements in an XML array. Plug-ins can use this method to read an array of unknown size or to verify that the size of an array is correct prior to accessing its elements with CWGetArraySettingElement.

See Also "CWGetArraySettingElement" on page 259

CWGetBooleanValue

Description Returns the value of a boolean element of an XML array or structure.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetBooleanValue(
    CWPluginContext context,
    CWSettingID settingID,
    Boolean* value);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
settingID	Specifies the XML setting for which to return a value.
value	Returns the boolean XML setting value.



Remarks CWGetBooleanValue returns the value of a boolean XML setting, given the setting’s ID. This method returns an error if the given setting is not a boolean value.

Plug-ins usually use this method to read a structure field or array element obtained by calling CWGetStructureSettingField or CWGetArraySettingElement. To read the value of a top-level boolean setting, use CWReadBooleanSetting.

See Also “CWGetStructureSettingField” on page 267
“CWGetArraySettingElement” on page 259
“CWReadBooleanSetting” on page 319

CWGetFloatingPointValue

Description Returns the value of a floating point element of an XML array or structure.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetFloatingPointValue(
    CWPluginContext context,
    CWSettingID      settingID,
    double*          value);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
settingID	Specifies the XML setting for which to return a value.
value	Returns the double-precision floating point XML setting value.

Remarks CWGetFloatingPointValue returns the value of a floating point XML setting, given its ID. This method returns an error if the given setting is not a floating point number.

Plug-ins use this routine to read a structure field or array element obtained by calling CWGetStructureSettingField or CWGetArraySettingElement. To read the value of a top-level floating point setting, use CWReadFloatingPointSetting.

Settings Panel Plug-in API Reference

CWGetIntegerValue

See Also “CWGetStructureSettingField” on page 267
“CWGetArraySettingElement” on page 259
“CWReadFloatingPointSetting” on page 319

CWGetIntegerValue

Description Returns the value of an integer element of an XML array or structure.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetIntegerValue(
    CWPluginContext context,
    CWSettingID settingID,
    long* value);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
settingID	Specifies the XML setting for which to return a value.
value	Returns the long integer XML setting value.

Remarks CWGetIntegerValue returns the value of an integer XML setting given its ID. This method returns an error if the given setting is not an integer.

Plug-ins use this method to read a structure field or array element obtained by calling CWGetStructureSettingField or CWGetArraySettingElement. To read the value of a top-level integer setting, use CWReadIntegerSetting.

See Also “CWGetStructureSettingField” on page 267
“CWGetArraySettingElement” on page 259
“CWReadIntegerSetting” on page 320

CWGetNamedSetting

Description Returns the ID of a top-level XML setting specified by name.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetNamedSetting(
    CWPluginContext context,
    const char* name,
    CWSettingID* settingID);
```

Parameters

The parameters for this method include:

context	Private, opaque IDE state.
name	Specifies the name of an XML setting, as a C string.
settingID	Returns the ID of a newly-created or existing setting container.

Remarks

CWGetNamedSetting returns the ID of an XML setting contained in a panel’s XML data. Normally, the ID refers to a structured XML setting, such as an array or structure.

The returned ID can be used to read or write a nested structure, using CWGetStructureSettingField. To read or write a nested array, call CWGetArraySettingElement.

NOTE

Plug-ins can use CWGetNamedSetting to get or create an ID for any XML setting type, whether structured or not. However, it is usually easier to use the CWRead... and CWWrite... calls when reading non-structured types.

A setting returned (while reading) or created (during writing) by CWGetNamedSetting is always a top-level setting, meaning that the parent container is the plug-in’s XML stream (either input or output).

CWGetNamedSetting behaves differently during a read request than during a write request. During a reqReadSettings request, CWGetNamedSetting returns an error if the named setting is not found in the XML input stream. During a reqWriteSettings request, CWGetNamedSetting instead creates a top-level setting having the specified name.

See Also

“CWGetStructureSettingField” on page 267

“CWGetArraySettingElement” on page 259

Settings Panel Plug-in API Reference

CWGetRelativePathValue

“reqReadSettings” on page 364

“reqWriteSettings” on page 368

CWGetRelativePathValue

Description Returns the value of a relative path element of an XML array or structure.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetRelativePathValue(
    CWPluginContext context,
    CWSettingID settingID,
    CWRelativePath* value);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
settingID	Specifies the XML setting for which to return a value.
value	Returns the relative path XML setting value.

Remarks CWGetBooleanValue returns the value of a relative path XML setting given its ID. Returns an error if the given setting is not a relative path.

Plug-ins use this method read a structure field or array element obtained by calling CWGetStructureSettingField or CWGetArraySettingElement. To read the value of a top-level relative path setting, use CWReadRelativePathSetting.

See Also “CWGetStructureSettingField” on page 267

“CWGetArraySettingElement” on page 259

“CWReadRelativePathSetting” on page 321

CWGetStringValue

Description Returns the value of a string element of an XML array or structure.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetStringValue(
```

```
CWPluginContext context,
CWSettingID settingID,
const char** value);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
settingID	Specifies the XML setting for which to return a value.
value	Returns a C string containing the contents of an XML string setting. The string remains the property of the IDE.

Remarks CWGetBooleanValue returns the value of a string XML setting, given the setting’s ID. It returns an error if the given setting is not of string type.

CAUTION The returned `value` string remains the property of the IDE. The can release it at the end of the pending request. Therefore, to retain the text returned in `value`, to copy it into another string before returning to the IDE.

This routine is typically used to read a structure field or array element obtained by calling `CWGetStructureSettingField` or `CWGetArraySettingElement`. To read the value of a top-level string setting, use `CWReadStringSetting` instead.

See Also “CWGetStructureSettingField” on page 267
“CWGetArraySettingElement” on page 259
“CWReadStringSetting” on page 322

CWGetStructureSettingField

Description Returns one field of an XML structure, by ID.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWGetStructureSettingField(
    CWPluginContext context,
    CWSettingID settingID,
```

Settings Panel Plug-in API Reference

CWGetStructureSettingField

	<pre> const char* name, CWSettingID* fieldSettingID); </pre>								
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>settingID</td><td>Specifies the IDE of an XML structure.</td></tr> <tr> <td>name</td><td>Specifies the name of the structure field to return.</td></tr> <tr> <td>fieldSettingID</td><td>Returns the ID of the named field.</td></tr> </table>	context	Private, opaque IDE state.	settingID	Specifies the IDE of an XML structure.	name	Specifies the name of the structure field to return.	fieldSettingID	Returns the ID of the named field.
context	Private, opaque IDE state.								
settingID	Specifies the IDE of an XML structure.								
name	Specifies the name of the structure field to return.								
fieldSettingID	Returns the ID of the named field.								
Remarks	CWGetStructureSettingField returns an ID for a field within an XML structure. CWGetStructureSettingField behaves differently depending upon when it is called. During a reqReadSettings request, this routine returns an error if a field having the specified name is not found. If the named field is found, the plug-in can use the returned ID to call one of the following routines to obtain the value of the field: • CWGetBooleanValue • CWGetFloatingPointValue • CWGetIntegerValue • CWGetRelativePathValue • CWGetStringValue If the field is another structure, the plug-in should call CWGetStructureSettingField to obtain the fields of the structure. If the field is an array, the plug-in should call CWGetArraySettingSize and CWGetArraySettingElement to read the elements of the array. During a reqReadSettings request, to get the ID of a top-level XML structure setting (that is, a setting whose parent container is implicitly the plug-in's XML stream), the plug-in can call CWGetNamedSetting. The plug-in can use the returned ID to call CWGetStructureSettingField. During a reqWriteSettings request, this routine creates a new XML setting have the specified name if one has not already been								

created and return its ID. The plug-in can use the returned ID to call one of the following routines to set the value of the field:

- `CWSetBooleanValue`
- `CWSetFloatingPointValue`
- `CWSetIntegerValue`
- `CWSetRelativePathValue`
- `CWSetStringValue`

If the field to be written is another structure or array, the plug-in should call `CWGetStructureSettingField` or `CWGetArraySettingElement` to create a new setting. Both routines return an ID, which the plug-in can use in further calls to any of the above routines.

A plug-in creates a new top-level XML structure setting during a `reqWriteSettings` request by calling `CWGetNamedSetting`. The plug-in can use the returned ID to call `CWGetStructureSettingField`.

See Also “`CWGetArraySettingElement`” on page 259

“`CWGetArraySettingSize`” on page 262

“`CWGetNamedSetting`” on page 264

“`reqReadSettings`” on page 364

“`reqWriteSettings`” on page 368

“`CWGetBooleanValue`” on page 262

“`CWGetFloatingPointValue`” on page 263

“`CWGetIntegerValue`” on page 264

“`CWGetRelativePathValue`” on page 266

“`CWGetStringValue`” on page 266

“`CWSetBooleanValue`” on page 323

“`CWSetFloatingPointValue`” on page 324

Settings Panel Plug-in API Reference
CWPanelActivateItem

“CWSetIntegerValue” on page 325

“CWSetRelativePathValue” on page 326

“CWSetStringValue” on page 327

CWPanelActivateItem

Description Switches user input focus to the specified item.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelActivateItem(
    CWPluginContext context,
    long            whichItem);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichItem	The plug-in specifies the item to give the input focus.

Remarks A settings panel plug-in calls this routine to make the specified item the focus of user input. For example, to switch user input to a text edit box, a plug-in can call `CWPanelActivateItem` with the ID of the edit box.

TIP Generally, the user should be in control of input focus; use this routine only in special cases where appropriate.

See Also “CWPanelEnableItem” on page 274

CWPanelAppendItems

Description Adds the panel’s dialog items to the **Target Settings** dialog box.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelAppendItems(
    CWPluginContext context,
    short           ditlID);
```

Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>ditlID</td><td>The plug-in specifies the resource ID of the dialog resource (on Windows) or a 'PPob' resource (on Mac OS) to add to the Target Settings dialog box.</td></tr> </table>	context	Private, opaque IDE state.	ditlID	The plug-in specifies the resource ID of the dialog resource (on Windows) or a 'PPob' resource (on Mac OS) to add to the Target Settings dialog box.
context	Private, opaque IDE state.				
ditlID	The plug-in specifies the resource ID of the dialog resource (on Windows) or a 'PPob' resource (on Mac OS) to add to the Target Settings dialog box.				
Remarks	Plug-ins call CWPanelAppendItems to create their user interfaces. CWPanelAppendItems constructs the panel's user interface controls from a dialog resource specified by ID. The plug-in adds the panel's controls to the dialog box specified by the panelscope field of a plug-in's PanelFlags structure. For more information , see "PanelFlags Structure" in the <i>IDE SDK Developer's Guide</i>. The IDE expects to find a dialog resource among the currently accessible resources. On Windows, the dialog resource should be linked into the plug-in DLL. On Mac OS, the dialog ('PPob') resource should be copied into the plug-in's resource fork. A settings panel plug-in should call CWPanelAppendItems in response to a reqInitDialog request. After the panel's user interface has been created, the plug-in can make any additional changes needed to its interface.				
Windows	On Windows, plug-ins sometimes need to call CWPanelGetListItemText, CWPanelSetListItemText, CWPanelInsertListItem, and CWPanelDeleteListItem to set up the items in a combo-box list.				
See Also	"Handling a reqInitDialog Request" in the <i>IDE SDK Developer's Guide</i> "reqInitDialog" on page 360 "CWPanelGetListItemText" on page 284 "CWPanelSetListItemText" on page 297 "CWPanelDeleteListItem" on page 274 "CWPanelInsertListItem" on page 290				

CWPanelChooseRelativePath

Description	Prompts the user to select a folder, and returns the specified folder as a relative path.												
Prototype	<pre>#include <CWDropInPanel.h> CW_CALLBACK CWPanelChooseRelativePath(CWPluginContext context, CWRelativePath* ioPath, Boolean isFolder, short filterCount, void* filterList, char* prompt);</pre>												
Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>ioPath</td><td>Specifies the initial location for selecting a file or folder, and returns the file or folder chosen by the user, as a relative path.</td></tr> <tr> <td>isFolder</td><td>Specifies whether a file or folder is to be selected. Selects a folder if true.</td></tr> <tr> <td>filterCount</td><td>Specifies the number of types used to filter the file list, when displaying files. Used only on Mac OS.</td></tr> <tr> <td>filterList</td><td>On Windows, this specifies a file or directory filter string. On the Mac OS, this points to an <code>SFTypeList</code> (an array of 4-character <code>OSType</code> file type codes).</td></tr> <tr> <td>prompt</td><td>Unused and ignored.</td></tr> </table>	context	Private, opaque IDE state.	ioPath	Specifies the initial location for selecting a file or folder, and returns the file or folder chosen by the user, as a relative path.	isFolder	Specifies whether a file or folder is to be selected. Selects a folder if true.	filterCount	Specifies the number of types used to filter the file list, when displaying files. Used only on Mac OS.	filterList	On Windows, this specifies a file or directory filter string. On the Mac OS, this points to an <code>SFTypeList</code> (an array of 4-character <code>OSType</code> file type codes).	prompt	Unused and ignored.
context	Private, opaque IDE state.												
ioPath	Specifies the initial location for selecting a file or folder, and returns the file or folder chosen by the user, as a relative path.												
isFolder	Specifies whether a file or folder is to be selected. Selects a folder if true.												
filterCount	Specifies the number of types used to filter the file list, when displaying files. Used only on Mac OS.												
filterList	On Windows, this specifies a file or directory filter string. On the Mac OS, this points to an <code>SFTypeList</code> (an array of 4-character <code>OSType</code> file type codes).												
prompt	Unused and ignored.												
Remarks	Plug-ins call <code>CWPanelChooseRelativePath</code> to prompt the user to select a file or folder, specified relatively by a <code>CWRelativePath</code> structure. Relative path specifications let plug-ins keep track of files when moving project files. <code>CWPanelChooseRelativePath</code> prompts the user to choose a file or folder, according to the value of <code>isFolder</code>, starting in the location specified by <code>ioPath</code>. It returns a relative path specification for the file or folder chosen in <code>ioPath</code>. If the user cancels the												

operation, *CWPanelChooseRelativePath* returns *cwErrUserCanceled*.

Currently, the IDE ignores the *prompt* parameter.

CWPanelChooseRelativePath filters the list of files (when selecting files) using *filterList*. *filterList* controls which files appear for selection. The format of the parameter differs on Windows and the Mac OS.

Windows On Windows, the *filterList* parameter has the same format as the *lpstrFilter* field of an *OPENFILENAME* structure passed to the *GetOpenFileName* common dialog call is formatted. The IDE ignores the *filterCount* parameter on Windows.

Briefly, the filter string consists of pairs of null-terminated strings, concatenated (but with intervening *NULL* characters intact) into one large string, terminated by a final *NULL* character. Each pair of strings consists of a filter name followed by a pattern specification.

The pattern specification can include legal file name characters and the DOS "*" wildcard character and usually specifies the literal value of a file extension. Multiple file patterns can be included in one string by separating each with a semicolon (";") character.

For further details, see Microsoft's MSDN developer information.

Mac OS On Mac OS, the list of displayed files is filtered against a standard *SFTypeList* array of four-character codes, listing the file types to display. The *filterCount* parameter specifies the number of files in the type list. To display all files, specify a count of -1.

NOTE These comments also apply to the Mac OS version of this routine, named *CWPanelChooseRelativePath*.

See Also "CWPanelGetRelativePathString" on page 288
 "CWRelativePath" on page 113
 "CWRelativePathTypes" on page 116
 "CWResolveRelativePath" on page 77

CWPanelDeleteListItem

Description	Deletes an item from a combo box list.						
Prototype	<pre>#include <CWDropInPanel.h> CW_CALLBACK CWPanelDeleteListItem(CWPluginContext context, long dlgItemID, long index);</pre>						
Parameters	The parameters for this method include: <table border="1"> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>dlgItemID</td><td>Specifies the ID of the combo-box containing the item to delete.</td></tr> <tr> <td>index</td><td>A one-based index that specifies the item to delete from a combo-box list.</td></tr> </table>	context	Private, opaque IDE state.	dlgItemID	Specifies the ID of the combo-box containing the item to delete.	index	A one-based index that specifies the item to delete from a combo-box list.
context	Private, opaque IDE state.						
dlgItemID	Specifies the ID of the combo-box containing the item to delete.						
index	A one-based index that specifies the item to delete from a combo-box list.						
Remarks	<code>CWPanelDeleteListItem</code> removes one item from a combo-box list. Use <code>CWPanelDeleteListItem</code> to maintain the list of items appearing in a combo box. For example, if your settings panel maintains a fixed-length history of recent text entries in a combo box, you can use <code>CWPanelDeleteListItem</code> to delete old items from the history and <code>CWPanelInsertListItem</code> to add new ones to it.						
See Also	“ <code>CWPanelInsertListItem</code> ” on page 290						

CWPanelEnableItem

Description	Enables or disables a panel control.
Prototype	<pre>#include <CWDropInPanel.h> CW_CALLBACK CWPanelEnableItem(CWPluginContext context, long whichItem, Boolean enableIt);</pre>
Parameters	The parameters for this method include:

context	Private, opaque IDE state.
whichItem	Specifies the ID of the panel control to enable or disable.
enableIt	Specifies whether to enable the control.

Remarks A plug-in calls this routine to enable or disable a panel item. Enabled controls respond to events and are drawn normally. Disabled items are dimmed and do not respond to user actions.

Plug-ins use `CWPanelEnableItem` to enable or disable one control based upon the state of another. For example, a panel for a linker capable of producing relocatable executables and absolute address executables could use `CWPanelEnableItem` to enable and disable the absolute load address edit box, depending upon whether the selected executable type is relative or absolute.

See Also “`CWPanelActivateItem`” on page 270
“`CWPanelShowItem`” on page 302

CWPanelGetCurrentPrefs

Description Returns a handle containing the panel’s current settings data.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetCurrentPrefs(
    CWPluginContext context,
    CWMemHandle* currentPrefs);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
currentPrefs	Returns a handle containing the plug-in’s current settings data.

Remarks A plug-in calls `CWPanelGetCurrentPrefs` to obtain the IDE’s current copy of its settings data. The returned handle remains the IDE’s property. A plug-in directly modifies the handle’s contents to save its current settings data.

Settings Panel Plug-in API Reference

CWPanelGetDebugFlag

Before accessing the returned settings handle, a plug-in should call `CWLockMemHandle` to lock the handle. When done accessing the handle, a plug-in should call `CWUnlockMemHandle`. To determine the size of the handle, use `CWGetMemHandleSize`. To resize the handle, use `CWResizeMemHandle`.

Mac OS On the Mac OS, plug-ins should instead directly access the `factoryPrefs` field of the `PanelParameterBlock`. The handle should be locked, unlocked, and resized using the Mac OS toolbox `HLock`, `HUnlock`, and `GetHandleSize` and `SetHandleSize` calls.

See Also “Settings Panel Data” in the *IDE SDK Developer’s Guide*
 “CWPanelGetFactoryPrefs” on page 278
 “CWPanelGetOriginalPrefs” on page 285
 “CWPanelGetPanelPrefs” on page 287

CWPanelGetDebugFlag

Description Indicates whether debugging is enabled for the current target.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetDebugFlag (
    CWPluginContext context,
    Boolean* debugOn) ;
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
debugOn	Returns true if debugging is currently on, and false otherwise.

Remarks `CWPanelGetDebugFlag` returns true if the user has enabled debugging by selecting **Enable Debugger** from the **Project** menu.

NOTE Do not confuse this flag with the value returned by `CWIsGeneratingDebugInfo`. `CWPanelGetDebugFlag` indicates whether the user enabled debugging of the final executable. The



value returned by `CWIsGeneratingDebugInfo` indicates whether the user enabled debugging information for a particular project file.

Plug-ins typically call `CWPanelGetDebugFlag` in response to a `reqSetupDebug` request, to determine whether debugging has been enabled or disabled and to then make any necessary modifications to settings data.

See Also Handling a “`reqSetupDebugRequest`” in the *IDE SDK Developer’s Guide*
 “`reqSetupDebug`” on page 365

CWPanelGetDialogItemHit

Description

Returns the ID of the most-recently operated panel control.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetDialogItemHit(
    CWPluginContext context,
    short* itemHit);
```

Parameters

The parameters for this method include:

context	Private, opaque IDE state.
itemHit	Returns the ID of the item hit during a <code>reqItemHit</code> request.

Remarks

`CWPanelGetDialogItemHit` returns the ID of the control with which the user most recently interacted. In response to a `reqItemHit` request, plug-ins call `CWPanelGetDialogItemHit` to determine which item was hit.

The plug must adjust the item number returned by `CWPanelGetDialogItemHit` by subtracting the base number of items appearing in the dialog, prior to the addition of the panel’s controls. Use `CWPanelGetNumBaseDialogItems` to get this count.

Depending upon the control that was hit, plug-ins can take further action, such as enabling, disabling, and changing the values of other controls. For example, a preference panel might provide a button for



Freescale Semiconductor, Inc.

Settings Panel Plug-in API Reference

CWPanelGetFactoryPrefs

selecting a folder. When the plug-in receives a reqItemHit request, it checks to see if the ID of the item hit matches the ID of the button and, if so, calls CWPanelChooseRelativePath to select a folder.

Mac OS On the Mac OS, plug-ins instead examine the itemHit and baseItems fields of the PanelParameterBlock to determine the item hit and the base number of dialog items.

See Also “Handling User Selection in Dialog Boxes” in the *IDE SDK Developer’s Guide*

“Handling a reqItemHit Request” in the *IDE SDK Developer’s Guide*

“CWPanelGetNumBaseDialogItems” on page 285

“reqItemHit” on page 361

CWPanelGetFactoryPrefs

Description Returns a handle containing the panel’s factory default settings data.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetFactoryPrefs (
    CWPluginContext context,
    CWMemHandle*    factoryPrefs);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
factoryPrefs	Returns a handle containing the plug-in’s factory default settings data.

Remarks A plug-in calls CWPanelGetFactoryPrefs to obtain the IDE’s current copy of its settings data. The returned handle remains the IDE’s property. A plug-in directly modifies the handle’s contents to change its factory default settings data.

The IDE uses the factory default settings data when the user clicks the **Factory Settings** button in the **Target Settings** dialog box. Plug-ins should also compare their current settings to their current



factory defaults in response to a reqItemHit request, to determine how to set the factory flag using CWPanelSetFactoryFlag.

Before accessing the returned settings handle, a plug-in should call CWLockMemHandle to lock the handle. When finished accessing the handle, a plug-in should call CWUnlockMemHandle. To determine the size of the handle, use CWGetMemHandleSize. To resize the handle, use CWResizeMemHandle.

Mac OS On Mac OS, plug-ins should directly access the factoryPrefs field of the PanelParameterBlock. The handle should be locked, unlocked, and resized using the Mac OS toolbox HLock, HUnlock, and GetHandleSize and SetHandleSize calls.

See Also “Settings Panel Data” in the IDE SDK Developer’s Guide
“CWPanelGetCurrentPrefs” on page 275
“CWPanelGetOriginalPrefs” on page 285
“CWPanelGetPanelPrefs” on page 287

CWPanelGetItemData

Description Returns additional type-specific data for a panel control.

Prototype #include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetItemData(
 CWPluginContext context,
 long dlgItemID,
 void* outData,
 long* outDataLength);

Parameters The parameters for this method include:

context	Private, opaque IDE state.
dlgItemID	Specifies the ID of the item for which to get additional control data.
outData	Points to storage in which to return the additional control information.
outDataLength	Specifies the length of the data to be returned in outData.



Freescal Semiconductor, Inc.

Settings Panel Plug-in API Reference
CWPanelGetItemMaxLength

Remarks This routine provides no useful service for third-party plug-in developers at this time. The exact behavior of this routine is subject to change. Plug-ins should not use this routine.

See Also "CWPanelSetItemData" on page 292

CWPanelGetItemMaxLength

Description Returns the maximum allowed length for a text item.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetItemMaxLength(
    CWPluginContext context,
    long           dlgItemID,
    short*         outLength);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
dlgItemID	Specifies the ID of a static text or edit field about which to return length information.
outLength	Returns the maximum length, in characters, allowed when entering text into the item specified by dlgItemID.

Remarks To determine the maximum allowed length for text entered into a static text or edit text field, use CWPanelGetItemMaxLength. To set the maximum length, use CWPanelSetItemMaxLength.

See Also "Validating Input" in the IDE SDK Developer's Guide

"CWPanelSetItemMaxLength" on page 293

CWPanelGetItemText

Description Returns the text of an item in a dialog box.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetItemText(
    CWPluginContext context,
    long           whichItem,
    char*          str,
    short          maxLen);
```

Parameters	The parameters for this method include: <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>whichItem</td><td>Specifies the ID of the item for which to return text.</td></tr> <tr> <td>str</td><td>Provides space in which the IDE returns the item's text, as a C string on Windows, and a Pascal string on Mac OS.</td></tr> <tr> <td>maxLen</td><td>Specifies the maximum length of the text to be returned, including the length byte. If the text exceeds this length,</td></tr> </table>	context	Private, opaque IDE state.	whichItem	Specifies the ID of the item for which to return text.	str	Provides space in which the IDE returns the item's text, as a C string on Windows, and a Pascal string on Mac OS.	maxLen	Specifies the maximum length of the text to be returned, including the length byte. If the text exceeds this length,
context	Private, opaque IDE state.								
whichItem	Specifies the ID of the item for which to return text.								
str	Provides space in which the IDE returns the item's text, as a C string on Windows, and a Pascal string on Mac OS.								
maxLen	Specifies the maximum length of the text to be returned, including the length byte. If the text exceeds this length,								
Remarks	This routine returns the text from an editable text field or static text field. For other types of controls, it returns the control's title. Plug-ins usually call <code>CWPanelGetItemText</code> to extract the text typed into a panel by the user when handling a <code>reqGetData</code> request. <code>CWPanelGetItemText</code> also lets plug-ins perform character string validation. For example, in response to a <code>reqItemHit</code> request, a plug-in might ensure that the characters entered into a text field contain only legal file name characters. To extract the value of a field as a number, a plug-in calls <code>CWPanelGetItemValue</code> instead. If the text associated with the item is longer than <code>maxLen</code> characters, <code>CWPanelGetItemText</code> returns no error code, but returns as much text as will fit in <code>str</code>. To get the full text of an item, use <code>CWPanelGetItemTextHandle</code>. To determine the maximum possible length of text entered in a field, call <code>CWPanelGetItemMaxLength</code>. The value returned can be used to allocate space for text returned by <code>CWPanelGetItemText</code>.								
See Also	"Validating Input" in the <i>IDE SDK Developer's Guide</i> "<code>CWPanelGetItemMaxLength</code>" on page 280 "<code>CWPanelSetItemText</code>" on page 294 "<code>CWPanelGetItemTextHandle</code>" on page 282 "<code>CWPanelGetItemValue</code>" on page 283 "<code>CWPanelSetItemValue</code>" on page 296								



Freescale Semiconductor, Inc.

Settings Panel Plug-in API Reference

CWPanelGetItemTextHandle

“reqGetData” on page 358

“reqItemHit” on page 361

CWPanelGetItemTextHandle

Description Returns a handle containing the text of a panel control.

Prototype `#include <CWDropInPanel.h>`

```
CW_CALLBACK CWPanelGetItemTextHandle(  
    CWPluginContext context,  
    long            whichItem,  
    CWMemHandle*   text);
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichItem	Specifies the ID of the item for which to return a text handle.
text	Returns a handle containing the text of the item. The handle becomes the caller’s property.

Remarks CWPanelGetItemTextHandle returns a handle to the text of a static text or edit text item. For any other type of control, it returns a handle to the control’s title text. Plug-ins can use CWPanelGetItemTextHandle to get text from a control when that text is too large to fit in a Pascal string returned by CWPanelGetItemText.

The returned handle becomes the property of the plug-in and should be released when no longer needed. Alternatively, a plug-in can modify the handle’s text content and pass it back to the IDE using CWPanelSetItemTextHandle to set the item’s text. In this case, because CWPanelSetItemTextHandle assumes ownership of the handle passed to it, the handle need not be released.

The returned handle should be locked prior to access using CWLockMemHandle and unlocked using CWUnlockMemHandle. A plug-in can determine the size of the handle by calling CWGetMemHandleSize and change it by calling CWResizeMemHandle.

See Also “CWPanelSetItemTextHandle” on page 295
 “CWPanelGetItemText” on page 280
 “CWPanelGetItemValue” on page 283

CWPanelGetItemValue

Description Returns the value of a panel control as a long integer.

Prototype `#include <CWDropInPanel.h>`

 `CW_CALLBACK CWPanelGetItemValue (`
 `CWPluginContext context,`
 `long whichItem,`
 `long* value);`

Parameters The parameters for this method include:

context	Private, opaque IDE state.
whichItem	Specifies the ID of the item for which to return a value.
value	Returns the value of the specified control as a long integer. For text controls, this value is obtained by attempting to convert the item text to a number.

Remarks CWPanelGetItemValue returns the value of a panel control.
 CWPanelGetItemValue returns the following values for common control types:

Control type	Values returned
checkbox	0 when clear, 1 when set.
radio button	0 when clear, 1 when set.
popup menu	A number from 1 to the number of items in the popup, indicating the currently selected item.
static text	The value of the text converted to a number.
edit text	The value of the text converted to a number.

If the item is a text field, CWPanelGetItemValue assumes the text is a decimal number and attempts to convert it to a numeric value. Plug-ins can use this feature when validating numeric fields. If the

Settings Panel Plug-in API Reference
CWPanelGetListItemText

text of the specified field cannot be converted to a long integer, *CWPanelGetValue* returns an error.

- See Also “Handling a *reqItemHit* Request” in the *IDE SDK Developer’s Guide*
- “Validating Input” in the *IDE SDK Developer’s Guide*
- “*CWPanelGetItemText*” on page 280
- “*CWPanelSetItemValue*” on page 296

CWPanelGetListItemText

Description Get the text of a Windows combo box list item.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetListItemText (
    CWPluginContext context,
    long          dlgItemID,
    long          index,
    char*         str,
    short         maxLen) ;
```

Parameters The parameters for this method include:

context	Private, opaque IDE state.
dlgItemID	Specifies the ID of a combo box.
index	Specifies the one-based index of the combo box list item to return.
str	Provides space in which the IDE returns the text of the specified list item as a C string.
maxLen	Specifies the maximum number of characters to return in <i>str</i> , including null byte.

Remarks Panels use *CWPanelGetListItemText* to obtain the text of a Windows combo box list item. Plug-ins can use *CWPanelGetListItemText* to determine the selected item in a combo box list constructed at run time (for example, a list of fonts).

CWPanelGetListItemText returns at most *maxLen* characters of the text. If the item is longer than *maxLen*,



CWPanelGetListItemText returns as many characters as can fit in str.

- See Also
- “CWPanelSetListItemText” on page 297
 - “CWPanelInsertListItem” on page 290
 - “CWPanelDeleteListItem” on page 274

CWPanelGetNumBaseDialogItems

- Description
- Returns the number of controls in the CodeWarrior **Target Settings** dialog prior to installation of a panel’s controls.
- Prototype
- ```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetNumBaseDialogItems(
 CWPluginContext context,
 short* baseItems);
```
- Parameters
- The parameters for this method include:

|           |                                                                                                                      |
|-----------|----------------------------------------------------------------------------------------------------------------------|
| context   | Private, opaque IDE state.                                                                                           |
| baseItems | Returns the original number of items in the <b>Target Settings</b> dialog, prior to installation of a panel’s items. |

- Remarks
- Plug-ins call CWPanelGetNumBaseDialogItems when responding to a reqItemHit request. The value returned is used to adjust the item number returned by CWPanelGetDialogItemHit. To determine the true item number of the hit item, plug-ins should subtract the value returned by CWPanelGetNumBaseDialogItems from the item number returned by CWPanelGetDialogItemHit.
- See Also
- “Handling User Selection in Dialog Boxes” in the *IDE SDK Developer’s Guide*
  - “CWPanelGetDialogItemHit” on page 277
  - “reqItemHit” on page 361

CWPanelGetOriginalPrefs

- Description
- Returns a handle containing the panel’s original settings data.

## Settings Panel Plug-in API Reference

### CWPanelGetOriginalPrefs

**Prototype**    `#include <CWDropInPanel.h>`  
                   `CW_CALLBACK CWPanelGetOriginalPrefs(`  
                           `CWPluginContext context,`  
                           `CWMemHandle* originalPrefs);`

**Parameters**    The parameters for this method include:

|               |                                                                                  |
|---------------|----------------------------------------------------------------------------------|
| context       | Private, opaque IDE state.                                                       |
| originalPrefs | Returns a handle containing the plug-in's most recently committed settings data. |

**Remarks**    A plug-in calls `CWPanelGetFactoryPrefs` to obtain the IDE's most recently saved copy of the panel's settings data. The returned handle remains the IDE's property. Plug-ins should not modify the contents of this handle.

The returned handle contains the settings data values most recently saved by the user. If the user has not made any changes in the **Target Settings** dialog box, the contents of the original and current settings remain identical.

Before accessing the returned settings handle, a plug-in should call `CWLockMemHandle` to lock the handle. When finished accessing the handle, a plug-in should call `CWUnlockMemHandle`. To determine the size of the handle, use `CWGetMemHandleSize`. To resize the handle, use `CWResizeMemHandle`.

**Mac OS**    On the Mac OS, plug-ins should instead directly access the `factoryPrefs` field of the `PanelParameterBlock`. The handle should be locked, unlocked, read, and resized using the Mac OS toolbox calls `HLock`, `HUnlock`, `GetHandleSize`, and `SetHandleSize`.

**See Also**    "Settings Panel Data" in the *IDE SDK Developer's Guide*  
                   "Handling a `reqItemHit` Request" in the *IDE SDK Developer's Guide*  
                   "CWPanelGetCurrentPrefs" on page 275  
                   "CWPanelGetFactoryPrefs" on page 278  
                   "CWPanelGetPanelPrefs" on page 287  
                   "CWPanelSetFactoryFlag" on page 291

“CWPanelSetRevertFlag” on page 301

### CWPanelGetPanelPrefs

**Description** Returns a handle containing settings data for another panel, specified by name.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetPanelPrefs(
 CWPluginContext context,
 const char* panelName,
 CWMemHandle* prefs,
 Boolean* requiresByteSwap);
```

**Parameters** The parameters for this method include:

|                  |                                                                                                                                                      |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| context          | Private, opaque IDE state.                                                                                                                           |
| panelName        | Specifies the name of the panel whose settings data handle should be returned. Specified as a C string on Windows, and as a Pascal string on Mac OS. |
| prefs            | Returns a handle containing the named panel’s settings data.                                                                                         |
| requiresByteSwap | Returns true if the data in the prefs handle must have its byte order swapped prior to use.                                                          |

**Remarks** Plug-ins call `CWPanelGetPanelPrefs` to obtain the settings data of a different panel. Plug-ins use this method when setting up a compiler, linker, or version control system’s data requires multiple panels. Plug-ins also use this method when one panel constructs some of its data values based on settings configured in another panel.

A plug-in must know the format of another panel’s data to use it effectively. Normally, only closely related settings panels should share data this way.

If `requiresByteSwap` is true on return, the plug-in must swap the byte ordering of the data before using it.



Settings Panel Plug-in API Reference  
CWPanelGetRelativePathString

Plug-ins should not modify the contents of the data handle returned. The handle remains the property of the IDE and should *not* be released by the plug-in.

Before accessing the returned settings handle, a plug-in should call CWLockMemHandle to lock the handle. When finished accessing the handle, a plug-in should call CWUnlockMemHandle. To determine the size of the handle, use CWGetMemHandleSize. To resize the handle, use CWResizeMemHandle.

- See Also    “Settings Panel Data” in the *IDE SDK Developer’s Guide*
- “Handling a reqItemHit Request” in the *IDE SDK Developer’s Guide*
- “CWPanelGetCurrentPrefs” on page 275
- “CWPanelGetFactoryPrefs” on page 278
- “CWPanelGetOriginalPrefs” on page 285
- “CWPanelSetFactoryFlag” on page 291
- “CWPanelSetRevertFlag” on page 301

CWPanelGetRelativePathString

Description    Converts a relative path to a textual full path representation.

Prototype    `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWPanelGetRelativePathString(`  
              `CWPluginContext    context,`  
              `CWRelativePath*    inPath,`  
              `char*                pathString,`  
              `long*                maxLength);`

Parameters    The parameters for this method include:

|         |                                                 |
|---------|-------------------------------------------------|
| context | Private, opaque IDE state.                      |
| inPath  | Specifies the relative path to convert to text. |

|            |                                                                                                                                                                                                                                                                       |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pathString | Returns the full path equivalent of the relative path specified by inPath, as a C string on Windows, and as a Pascal string on Mac OS. The returned string includes special path prefixes, such as '{Project}' to indicate the reference location for relative paths. |
| maxLength  | Specifies the maximum length of the text to return in pathString, including length NULL byte.                                                                                                                                                                         |

Remarks CWPanelGetRelativePathString returns a path string for a given relative path in a form suitable for display to users. For example, the IDE uses CWPanelGetRelativePathString to display the **Output Directory** path in the **Target Settings** dialog.

CWRelativePath structures consist of a reference directory and a partial path. The partial path specifies the location of an item relative to the reference directory. Path strings returned by CWPanelGetRelativePathString start with prefixes that indicate the type of reference path. (CWResolveRelativePath, in contrast, always returns the full path.)

Depending upon the value of pathType for a CWRelativePath structure, CWPanelGetRelativePathString constructs path strings beginning with one of the following prefixes:

| Path type                          | Path string prefix                                                                      |
|------------------------------------|-----------------------------------------------------------------------------------------|
| type_Compiler                      | '{Compiler}'                                                                            |
| type_Project                       | '{Project}'                                                                             |
| type_System                        | '{System}'                                                                              |
| type_Absolute,<br>type_UserDefined | No prefix returned. In these cases, CWPanelGetRelativePathString returns the full path. |

For example, if a plug-in sets the output directory to a 'bin' subdirectory of the current project, CWPanelGetRelativePathString converts the relative path for the output directory to the following string:

{Project}\bin'



Settings Panel Plug-in API Reference  
*CWPanelInsertListItem*

---

Panels can call `CWPanelChooseRelativePath` to request selection of a relative path by the user. Plug-ins can also explicitly construct `CWRelativePath`.

See Also    `"CWPanelChooseRelativePath"` on page 272  
              `"CWRelativePath"` on page 113  
              `"CWRelativePathTypes"` on page 116  
              `"CWResolveRelativePath"` on page 77

**CWPanelInsertListItem**

Description    Inserts an item into a Windows combo box list.

Prototype    `#include <CWDropInPanel.h>`  
              `CWPanelInsertListItem(`  
                    `CWPluginContext context,`  
                    `long dlgItemID,`  
                    `long index,`  
                    `char* str);`

Parameters    The parameters for this method include:

|           |                                                                                                            |
|-----------|------------------------------------------------------------------------------------------------------------|
| context   | Private, opaque IDE state.                                                                                 |
| dlgItemID | Specifies the ID of the combo box.                                                                         |
| index     | Specifies the position at which to insert the text item in the combo box list. Item indexes are one-based. |
| str       | Specifies the text of the item to insert, as a C string.                                                   |

Remarks    Use `CWPanelInsertListItem` to insert items in a Windows combo box list. Plug-ins typically call `CWPanelInsertListItem` in response to a `reqInitDialog` request, to initialize the contents of a combo box list.

Plug-ins can also call `CWPanelInsertListItem` while a panel is running, to modify the options presented in a combo box (such as a history list). To delete items from a combo box list, call `CWPanelDeleteListItem`.

Items are inserted at the position specified by `index`, and all other items initially at the same index or higher are shuffled to higher indexes. For example, inserting “test” at position 2 in the list {“First Item”, “Last Item”} yields the list {“First Item”, “test”, “Last Item”}.

To insert items at the start of the list, use a value of 1 for `index`. To insert items at the end of the list, specify an `index` value one higher than the number of items in the list.

See Also “CWPanelDeleteListItem” on page 274

“reqInitDialog” on page 360

“CWPanelGetListItemText” on page 284

“CWPanelSetListItemText” on page 297

CWPanelInvalItem

Description Invalidates the dialog area occupied by a dialog item.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelInvalItem(
 CWPluginContext context,
 long whichItem);
```

Parameters The parameters for this method include:

|           |                                                   |
|-----------|---------------------------------------------------|
| context   | Private, opaque IDE state.                        |
| whichItem | Specifies the ID of the panel item to invalidate. |

Remarks Plug-ins call `CWPanelInvalItem` to force an item to be redrawn. The IDE redraws the item the next time the IDE receives a paint message. Plug-ins generally use this method for custom items. Most plug-ins do not need to call this routine.

See Also “Redrawing Panel Items” in the *IDE SDK Developer’s Guide*

“CWPanelValidItem” on page 303

CWPanelSetFactoryFlag

Description Sets the value of the factory settings flag.

## Settings Panel Plug-in API Reference

### CWPanelSetItemData

Prototype `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWPanelSetFactoryFlag(`  
`CWPluginContext context,`  
`Boolean canFactory);`

Parameters The parameters for this method include:

|            |                                                                                                                                            |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| context    | Private, opaque IDE state.                                                                                                                 |
| canFactory | Specifies the new value of the factory flag. Set this to <code>true</code> if the current preferences differ from the factory preferences. |

Remarks Plug-ins call `CWPanelSetFactoryFlag` to set the state of the factory settings flag. This flag indicates whether the current settings match the factory settings. A value of `true` means the current settings do not match factory settings. When `true`, the IDE enables the **Factory Settings** button in the **Target Settings** dialog, allowing the user to reset all settings to their factory defaults.

This flag should be set or cleared in response to a `reqItemHit` request. To obtain the plug-in's current settings data, call `CWPanelGetCurrentPrefs`. To obtain its current factory settings, call `CWPanelGetFactoryPrefs`.

See Also "Handling a `reqItemHit` Request" in the *IDE SDK Developer's Guide*  
 "reqItemHit" on page 361  
 "CWPanelSetRevertFlag" on page 301  
 "CWPanelGetCurrentPrefs" on page 275  
 "CWPanelGetFactoryPrefs" on page 278  
 "CWPanelGetOriginalPrefs" on page 285

### CWPanelSetItemData

Description Sets additional type-specific data for a panel control.

Prototype `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWPanelSetItemData(`  
`CWPluginContext context,`  
`long dlgItemID,`



## Settings Panel Plug-in API Reference

### CWPanelSetItemText

A plug-in typically calls CWPanelSetItemMaxLength when initializing its UI in response to a reqInitDialog request and occasionally when responding to a reqItemHit request.

See Also    “Limiting Text Input” in the *IDE SDK Developer’s Guide*

              “Validating Input” in the *IDE SDK Developer’s Guide*

              “CWPanelGetItemMaxLength” on page 280

              “CWPanelGetItemText” on page 280

              “CWPanelSetItemText” on page 294

### CWPanelSetItemText

Description    Sets the text of a dialog item.

Prototype      

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelSetItemText (
 CWPluginContext context,
 long whichItem,
 char* str);
```

Parameters    The parameters for this method include:

|           |                                                                                                     |
|-----------|-----------------------------------------------------------------------------------------------------|
| context   | Private, opaque IDE state.                                                                          |
| whichItem | Specifies the ID of the item for which to set the text.                                             |
| str       | Specifies the new text for the control, as a C string on Windows, and as a Pascal string on Mac OS. |

Remarks      This method changes the text for an editable text field, static text, or an item’s title.

                 A plug-in typically uses CWPanelSetItemText to install text in controls in response to a reqPutData request. Plug-ins can also call CWPanelSetItemText to modify the text of an item in response to a reqItemHit request.

                 For example, if the user enters invalid text in a field, the plug-in can extract the text from a field using CWPanelGetItemText, remove the invalid characters, and put the cleaned up text in the text control by using CWPanelSetItemText.

See Also “Handling a reqPutData Request” in the *IDE SDK Developer’s Guide*  
 “Handling a reqItemHit Request” in the *IDE SDK Developer’s Guide*  
 “CWPanelsItemText” on page 280  
 “CWPanelsItemValue” on page 283  
 “CWPanelsItemValue” on page 296  
 “reqPutData” on page 363  
 “reqItemHit” on page 361

### CWPanelsItemTextHandle

Description

Sets the text or title of a panel control to the text contained in a handle.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelsItemTextHandle(
 CWPluginContext context,
 long whichItem,
 CWMemHandle text);
```

Parameters

The parameters for this method include:

|           |                                                         |
|-----------|---------------------------------------------------------|
| context   | Private, opaque IDE state.                              |
| whichItem | Specifies the ID of the item for which to set the text. |
| text      | Contains the new text of the item.                      |

Remarks

Plug-ins can call CWPanelsItemTextHandle to change the text of a static text or edit text item or the title of any other control type.

Unlike CWPanelsItemText, which only provides access to the first 255 characters of an item’s text (Mac OS), CWPanelsItemTextHandle allows a plug-in to set the entire text of an item.

The text handle becomes the property of the IDE. Call CWPanelsItemTextHandle to get the text of an item in a handle. A plug-in can make changes to an item’s text by calling

## Settings Panel Plug-in API Reference

### CWPanelsItemValue

CWPanelsItemTextHandle, making changes to the text, and then calling CWPanelsItemTextHandle.

- See Also
- “CWPanelsItemTextHandle” on page 282
  - “CWPanelsItemText” on page 280
  - “CWPanelsItemText” on page 294
  - “CWPanelsItemValue” on page 283
  - “CWPanelsItemValue” on page 296

### CWPanelsItemValue

Description Sets the text of an static or edit text item to the text representation of a long integer.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelsItemValue(
 CWPanelsContext context,
 long whichItem,
 long value);
```

Parameters The parameters for this method include:

|           |                                                        |
|-----------|--------------------------------------------------------|
| context   | Private, opaque IDE state.                             |
| whichItem | Sets the ID of the item for which to change the value. |
| value     | Specifies the new value of the text item.              |

Remarks CWPanelsItemValue sets the value of a panel control. CWPanelsItemValue accepts the following values for common control types:

| Control type | Values to install in control                                                                 |
|--------------|----------------------------------------------------------------------------------------------|
| checkbox     | 0 to clear; 1 to set.                                                                        |
| radio button | 0 to clear; 1 to set.                                                                        |
| popup menu   | A number from 1 to the number of items in the popup, indicating the item currently selected. |

| Control type | Values to install in control                 |
|--------------|----------------------------------------------|
| static text  | The value of the text converted to a number. |
| edit text    | The value of the text converted to a number. |

If the dialog item is a text field, `CWPanelSetItemValue` sets the item to the text representation of a long integer. This is convenient when dealing with numeric entry fields.

For example, when responding to a `reqItemHit` request, a panel can obtain the value of a text field interpreted as a number by calling `CWPanelGetItemValue`. If the value returned is out of range, the plug-in can install a properly constrained value in the text field using `CWPanelSetItemValue`.

See Also    “`CWPanelGetItemValue`” on page 283  
               “`CWPanelGetItemText`” on page 280  
               “`CWPanelSetItemText`” on page 294

### CWPanelSetListItemText

Description    Sets the text of a Windows combo box list item.

Prototype    `#include <CWDropInPanel.h>`  
                  `CW_CALLBACK CWPanelSetListItemText (`  
                      `CWPluginContext    context,`  
                      `long                dlgItemID,`  
                      `long                index,`  
                      `char*              str);`

Parameters    The parameters for this method include:

|                        |                                                                                        |
|------------------------|----------------------------------------------------------------------------------------|
| <code>context</code>   | Private, opaque IDE state.                                                             |
| <code>dlgItemID</code> | Specifies the ID of the combo box item.                                                |
| <code>index</code>     | Specifies the one-based index of the combo box item for which to set the text content. |
| <code>str</code>       | Specifies the new text of the combo box list item, as a C string.                      |

## Settings Panel Plug-in API Reference

### CWPanelSetRecompileFlag

|          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remarks  | <p>A panel calls <code>CWPanelSetListItemText</code> to set the text of a Windows combo box list item. Combo box list indexes start at one. To obtain the text of an item, call <code>CWPanelGetListItemText</code>.</p> <p>Plug-ins call <code>CWPanelSetListItemText</code> to install items in a combo box during dialog initialization, in response to a <code>reqInitDialog</code> request. Plug-ins can also call <code>CWPanelSetListItemText</code> to change the text of combo box list items in response to user actions (typically during a <code>reqItemHit</code> request).</p> |
| See Also | <p>“<code>CWPanelGetListItemText</code>” on page 284</p> <p>“<code>CWPanelInsertListItem</code>” on page 290</p> <p>“<code>CWPanelDeleteListItem</code>” on page 274</p> <p>“<code>reqInitDialog</code>” on page 360</p> <p>“<code>reqItemHit</code>” on page 361</p>                                                                                                                                                                                                                                                                                                                        |

### CWPanelSetRecompileFlag

|             |                                                                                                                                                                                                                                                                                                                                                                           |         |                            |           |                                                |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----------------------------|-----------|------------------------------------------------|
| Description | Sets the value of the recompile flag.                                                                                                                                                                                                                                                                                                                                     |         |                            |           |                                                |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWPanelSetRecompileFlag(     CWPluginContext context,     Boolean recompile);</pre>                                                                                                                                                                                                                                     |         |                            |           |                                                |
| Parameters  | <p>The parameters for this method include:</p> <table border="1"> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>recompile</td><td>Specifies the new value of the recompile flag.</td></tr> </table>                                                                                                                                              | context | Private, opaque IDE state. | recompile | Specifies the new value of the recompile flag. |
| context     | Private, opaque IDE state.                                                                                                                                                                                                                                                                                                                                                |         |                            |           |                                                |
| recompile   | Specifies the new value of the recompile flag.                                                                                                                                                                                                                                                                                                                            |         |                            |           |                                                |
| Remarks     | <p>A plug-in calls <code>CWPanelSetRecompileFlag</code> when responding to a <code>reqValidate</code> request.</p> <p>A plug-in should set this flag if its current settings data necessitates a recompilation of the the current target’s source code. This might be necessary, for example, if the user of a panel changes the optimization level of compiled code.</p> |         |                            |           |                                                |
| See Also    | “Handling a <code>reqValidate</code> Request” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                                                                                                     |         |                            |           |                                                |

“reqValidate” on page 367

“CWPanelSetRelinkFlag” on page 299

“CWPanelSetReparseFlag” on page 299

“CWPanelSetResetPathsFlag” on page 300

### CWPanelSetRelinkFlag

Description

Sets the value of the relink flag.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelSetRelinkFlag(
 CWPluginContext context,
 Boolean relink);
```

Parameters

The parameters for this method include:

|         |                                             |
|---------|---------------------------------------------|
| context | Private, opaque IDE state.                  |
| relink  | Specifies the new value of the relink flag. |

Remarks

A plug-in calls CWPanelSetRelinkFlag when responding to a reqValidate request.

A plug-in should set this flag if its current settings data necessitates relinking the the current target’s compiled executable. Relinking might become necessary, for example, if the user of a panel changes the executable binary type of the current target.

See Also

“Handling a reqValidateRequest” in the *IDE SDK Developer’s Guide*

“reqValidate” on page 367

“CWPanelSetRecompileFlag” on page 298

“CWPanelSetReparseFlag” on page 299

“CWPanelSetResetPathsFlag” on page 300

### CWPanelSetReparseFlag

Description

Sets the value of the reparse flag.



Settings Panel Plug-in API Reference

CWPanelSetResetPathsFlag

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanelSetReparseFlag(`  
                  `CWPluginContext context,`  
                  `Boolean reparse);`

Parameters    The parameters for this method include:

|         |                                              |
|---------|----------------------------------------------|
| context | Private, opaque IDE state.                   |
| reparse | Specifies the new value of the reparse flag. |

Remarks    A plug-in calls `CWPanelSetReparseFlag` when responding to a `reqValidate` request. A panel should set this flag if changes to its settings affect the information maintained by the IDE's symbol browser.

**NOTE**    The IDE does not use this flag. Most plug-ins do not need to set this flag.

See Also    “Handling a `reqValidate` request” in the *IDE SDK Developer's Guide*  
              “`reqValidate`” on page 367  
              “`CWPanelSetRecompileFlag`” on page 298  
              “`CWPanelSetRelinkFlag`” on page 299  
              “`CWPanelSetResetPathsFlag`” on page 300

CWPanelSetResetPathsFlag

Description    Sets the value of the “reset paths” settings flag.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanelSetResetPathsFlag(`  
                  `CWPluginContext context,`  
                  `Boolean resetPaths);`

Parameters    The parameters for this method include:

|            |                                                    |
|------------|----------------------------------------------------|
| context    | Private, opaque IDE state.                         |
| resetPaths | Specifies the new value of the “reset paths” flag. |



Remarks     A plug-in calls CWPanelSetResetPathsFlag when responding to a reqValidate request.

              A plug-in should set this flag if its current settings data requires the IDE to rescan for project files. When a plug-in sets this flag, the IDE searches for all target files within the currently established access paths, after the user commits (saves) the changes made in the panel. This might be necessary, for example, if the user of a panel adds or removes project paths.

See Also     “Handling a reqValidate Request” in the *IDE SDK Developer’s Guide*

              “reqValidate” on page 367

              “CWPanelSetRecompileFlag” on page 298

              “CWPanelSetRelinkFlag” on page 299

              “CWPanelSetReparsingFlag” on page 299

CWPanelSetRevertFlag

Description     Sets the value of the revert flag.

Prototype     

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelSetRevertFlag(
 CWPluginContext context,
 Boolean canRevert);
```

Parameters     The parameters for this method include:

|           |                                             |
|-----------|---------------------------------------------|
| context   | Private, opaque IDE state.                  |
| canRevert | Specifies the new value of the revert flag. |

Remarks     A plug-in calls CWPanelSetRevertFlag when responding to a reqItemHit request.

              A plug-in should set this flag to indicate that the current settings data differs from the most recently saved settings data. When set, the IDE enables the **Revert** button in the **Target Settings** dialog.

See Also     “Handling a reqItemHit Request” in the *IDE SDK Developer’s Guide*

## Settings Panel Plug-in API Reference

### CWPanelShowItem

“CWPanelSetFactoryFlag” on page 291

“reqItemHit” on page 361

## CWPanelShowItem

**Description** Shows or hides an item in a settings panel.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelShowItem(
 CWPluginContext context,
 long whichItem,
 Boolean showIt);
```

**Parameters** The parameters for this method include:

|           |                                                                 |
|-----------|-----------------------------------------------------------------|
| context   | Private, opaque IDE state.                                      |
| whichItem | Specifies the ID of the item to show or hide.                   |
| showIt    | Specifies whether to show the item (nonzero) or hide it (zero). |

**Remarks** A plug-in calls this routine to show or hide a dialog item.

CWPanelShowItem may be useful for hiding controls which, based upon other target settings, have no relevance. Usually, however, controls should not be hidden from the user; in most cases, enabling and disabling items using CWPanelEnableItem is more appropriate.

**TIP** In general, if an item can ever be enabled during use of its panel, it should not be hidden.

When showing controls that were previously hidden, you may want to activate one of them using CWPanelEnableItem, to give it input focus.

**See Also** “Enabling and Disabling Items” in the *IDE SDK Developer’s Guide*

“CWPanelEnableItem” on page 274

“CWPanelActivateItem” on page 270

## CWPanlValidItem

|             |                                                                                                                                                                                                                                                                                                             |         |                            |           |                                           |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----------------------------|-----------|-------------------------------------------|
| Description | Validates a panel control.                                                                                                                                                                                                                                                                                  |         |                            |           |                                           |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWPanlValidItem(     CWPluginContext context,     long whichItem);</pre>                                                                                                                                                                                  |         |                            |           |                                           |
| Parameters  | <p>The parameters for this method include:</p> <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>whichItem</td><td>Specifies the ID of the item to validate.</td></tr> </table>                                                                                                | context | Private, opaque IDE state. | whichItem | Specifies the ID of the item to validate. |
| context     | Private, opaque IDE state.                                                                                                                                                                                                                                                                                  |         |                            |           |                                           |
| whichItem   | Specifies the ID of the item to validate.                                                                                                                                                                                                                                                                   |         |                            |           |                                           |
| Remarks     | <p>Call CWPanlValidItem to inform the IDE and host operating system that a control need not be redrawn. CWPanlValidItem cancels any pending paint (update) messages for the specified item.</p> <p>Most panels do not need to call CWPanlValidItem. CWPanlValidItem is most useful for custom controls.</p> |         |                            |           |                                           |
| See Also    | <p>“Redrawing Panel Items” in the <i>IDE SDK Developer’s Guide</i></p> <p>“CWPanlInvalItem” on page 291</p>                                                                                                                                                                                                 |         |                            |           |                                           |

## CWPanlActivateItem

|             |                                                                                                                            |
|-------------|----------------------------------------------------------------------------------------------------------------------------|
| Description | The Mac OS version of CWPanlActivateItem.                                                                                  |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWPanlActivateItem(     PanelParamBlkPtr ppb,     long whichItem);</pre> |

## CWPanlAppendItems

|             |                                                                                                                         |
|-------------|-------------------------------------------------------------------------------------------------------------------------|
| Description | The Mac OS version of CWPanlAppendItems.                                                                                |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWPanlAppendItems(     PanelParamBlkPtr ppb,     short ditlID);</pre> |

## CWPanlChooseRelativePath

Description The Mac OS version of CWPanlChooseRelativePath.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlChooseRelativePath(
 PanelParamBlkPtr ppb,
 CWRelativePath* path,
 Boolean isFolder,
 short filterCount,
 void* filterList,
 char* prompt);
```

## CWPanlDrawPanelBox

Description Draws a rectangle around a dialog item and optionally draws a title for the rectangle.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlDrawPanelBox(
 PanelParamBlkPtr ppb,
 long whichItem,
 ConstStr255Param title);
```

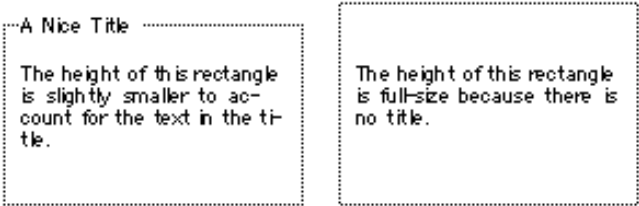
Parameters The parameters for this method include:

|           |                                                                                           |
|-----------|-------------------------------------------------------------------------------------------|
| ppb       | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests. |
| whichItem | The plug-in specifies the dialog item to draw the box around.                             |
| title     | The plug-in specifies the title of the box, formatted as a Pascal string.                 |

Remarks Draws a rectangle around a dialog box item and optionally draws a title for the rectangle.

The IDE draws a rectangle around the dialog item specified by *whichItem*. *title* specifies the title to give the rectangle. If *title* contains an empty string, no title appears (Figure 6.1).

**Figure 6.1    How CWPanelDrawPanelBox draws a rectangle**



See Also    “Drawing Custom Items” in the *IDE SDK Developer’s Guide*

### CWPanelDrawUserItemBox

Description    Obsolete call to the IDE.

Prototype    `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWPanelDrawUserItemBox(`  
`DialogPtr                  dialog,`  
`short                      whichItem,`  
`ConstStr255Param        title);`

### CWPanelEnableItem

Description    The Mac OS version of CWPanelEnableItem.

Prototype    `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWPanelEnableItem(`  
`PanelParamBlkPtr    ppb,`  
`long                          whichItem,`  
`Boolean                      enableIt);`

### CWPanelGetArraySettingElement

Description    The Mac OS version of CWGetArraySettingElement.

Prototype    `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWPanelGetArraySettingElement(`  
`PanelParamBlkPtr    ppb,`  
`CWSettingID                  settingID,`  
`long                          index,`  
`CWSettingID*              elementSettingID);`

## Settings Panel Plug-in API Reference

### CWPanlGetArraySettingSize

---

#### CWPanlGetArraySettingSize

**Description** The Mac OS version of CWGetArraySettingSize.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetArraySettingSize(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 long* size);
```

#### CWPanlGetBooleanValue

**Description** The Mac OS version of CWGetBooleanValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetBooleanValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 Boolean* value);
```

#### CWPanlGetFloatingPointValue

**Description** The Mac OS version of CWGetFloatingPointValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetFloatingPointValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 double* value);
```

#### CWPanlGetIntegerValue

**Description** The Mac OS version of CWGetIntegerValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetIntegerValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 long* value);
```

#### CWPanlGetItemControl

**Description** Retrieves the Mac OS control handle for a dialog item.

Prototype    `#include <CWDropInPanel.h>`  
               `CW_CALLBACK CWPanlGetItemControl (`  
                   `PanelParamBlkPtr    ppb,`  
                   `long                        whichItem,`  
                   `ControlRef*                control);`

Parameters    The parameters for this method include:

|           |                                                                                           |
|-----------|-------------------------------------------------------------------------------------------|
| ppb       | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests. |
| whichItem | Specifies the index of the item for which to return a control.                            |
| control   | Returns the Mac OS control for the specified panel item.                                  |

Remarks    A plug-in calls this routine to get the control handle for the item specified by `whichItem`. For items that contain menus, the IDE returns a menu handle.

In rare cases, a plug-in may require access to the Mac OS control corresponding to a panel item. To obtain this control, call `CWPanlGetItemControl` with the index of the panel item.

**NOTE**    Whenever possible, plug-ins should use API routines to access controls, rather than Mac OS toolbox calls.

The returned control handle remains property of the IDE. If an item has no associated control, `CWPanlGetItemControl` returns an error.

See Also    “CWPanlGetMacPort” on page 309

## CWPanlGetItemData

Description    The Mac OS version of `CWPanelGetItemData`.

Prototype    `#include <CWDropInPanel.h>`  
               `CW_CALLBACK CWPanlGetItemData (`  
                   `PanelParamBlkPtr    ppb,`  
                   `long                        whichItem,`  
                   `void*                        outData,`  
                   `long*                        outDataLength);`

## CWPanelGetItemMaxLength

**Description** The Mac OS version of CWPanelGetItemMaxLength.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetItemMaxLength(
 PanelParamBlkPtr ppb,
 long whichItem,
 short* outLength);
```

## CWPanelGetItemRect

**Description** Returns the bounding rectangle of the dialog item.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetItemRect(
 PanelParamBlkPtr ppb,
 long whichItem,
 Rect* rect);
```

**Parameters** The parameters for this method include:

|           |                                                                                           |
|-----------|-------------------------------------------------------------------------------------------|
| ppb       | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests. |
| whichItem | Specifies the index of the item for which to get the bounding rectangle.                  |
| rect      | The IDE returns the rectangle in this argument.                                           |

**Remarks** A plug-in calls this routine to get the bounding rectangle of a dialog item, in local coordinates. This is most useful when drawing and hit-testing custom controls (user items).

**See Also** “Drawing Custom Items” in the *IDE SDK Developer’s Guide*  
 “CWPanelGetItemControl” on page 306

## CWPanelGetItemText

**Description** The Mac OS version of CWPanelGetItemText.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetItemText(
 PanelParamBlkPtr ppb,
```



```
long whichItem,
StringPtr str,
short maxlen);
```

CWPanlGetItemTextHandle

Description The Mac OS version of CWPanlGetItemTextHandle.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetItemTextHandle(
 PanelParamBlkPtr ppb,
 long whichItem,
 Handle* text);
```

CWPanlGetItemValue

Description The Mac OS version of CWPanlGetItemValue.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetItemValue(
 PanelParamBlkPtr ppb,
 long whichItem,
 long* value);
```

CWPanlGetMacPort

Description Returns the graphics port of the settings dialog window.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetMacPort(
 PanelParamBlkPtr ppb,
 GrafPtr* port);
```

Parameters The parameters for this method include:

|      |                                                                                           |
|------|-------------------------------------------------------------------------------------------|
| ppb  | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests. |
| port | Returns the QuickDraw graphics port of the <b>Target Settings</b> dialog box.             |

Remarks CWPanlGetMacPort returns a pointer to the Mac OS graphics port in which all settings dialog drawing takes place.

## Settings Panel Plug-in API Reference

### CWPanelGetNamedSetting

Plug-ins can use this method to create custom controls. For example, a custom control can change the pen size, foreground or background colors, text font or size, or the clipping region.

**CAUTION** A panel should always restore any graphics port characteristics it has changed before returning to the IDE.

See Also “Drawing Custom Items” in the *IDE SDK Developer’s Guide*  
 “CWPanelDrawUserItemBox” on page 305

### CWPanelGetNamedSetting

Description The Mac OS version of CWGetNamedSetting.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetNamedSetting(
 PanelParamBlkPtr ppb,
 const char* name,
 CWSettingID* settingID);
```

### CWPanelGetPanelPrefs

Description The Mac OS version of CWPanelGetPanelPrefs.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetPanelPrefs(
 PanelParamBlkPtr ppb,
 StringPtr inPanelName,
 Handle* prefs,
 Boolean* requiresByteSwap);
```

### CWPanelGetRelativePathString

Description The Mac OS version of CWPanelGetRelativePathString.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanelGetRelativePathString(
 PanelParamBlkPtr ppb,
 CWRelativePath* path,
 char* pathString,
 long* maxLength);
```

## CWPanlGetRelativePathValue

**Description** The Mac OS version of CWGetRelativePathValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetRelativePathValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 CWRelativePath* value);
```

## CWPanlGetStringValue

**Description** The Mac OS version of CWGetStringValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetStringValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 const char** value);
```

## CWPanlGetStructureSettingField

**Description** The Mac OS version of CWGetStructureSettingField.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlGetStructureSettingField(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 const char* name,
 CWSettingID* fieldSettingID);
```

## CWPanlInstallUserItem

**Description** Obsolete call to the IDE.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlInstallUserItem(
 PanelParamBlkPtr ppb,
 short whichItem,
 UserItemProcPtr proc);
```

## Settings Panel Plug-in API Reference

### CWPanlInvalItem

**Description** The Mac OS version of CWPanlInvalItem.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlInvalItem(
 PanelParamBlkPtr ppb,
 long whichItem);
```

### CWPanlReadBooleanSetting

**Description** The Mac OS version of CWPanlReadBooleanSetting.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlReadBooleanSetting(
 PanelParamBlkPtr ppb,
 const char* name,
 Boolean* value);
```

### CWPanlReadFloatingPointSetting

**Description** The Mac OS version of CWPanlReadFloatingPointSetting.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlReadFloatingPointSetting(
 PanelParamBlkPtr ppb,
 const char* name,
 double* value);
```

### CWPanlReadIntegerSetting

**Description** The Mac OS version of CWPanlReadIntegerSetting.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlReadIntegerSetting(
 PanelParamBlkPtr ppb,
 const char* name,
 long* value);
```

### CWPanlReadRelativePathAEDesc

**Description** Extracts a relative path from a given Apple Event descriptor record.

**Prototype**

```
#include <CWDropInPanel.h>
```

```
CW_CALLBACK CWPanlReadRelativePathAEDesc(
 PanelParamBlkPtr ppb,
 CWRelativePath* path,
 const AEDesc* desc);
```

Parameters    The parameters for this method include:

|      |                                                                                               |
|------|-----------------------------------------------------------------------------------------------|
| ppb  | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests.     |
| path | Returns the relative path specified by the Apple Event descriptor passed in desc.             |
| desc | Provides the Apple Event descriptor data from an Apple Event, which contains a relative path. |

Remarks    CWPanlReadRelativePathAEDesc simplifies the task of extracting a CWRelativePath record from an Apple Event descriptor. Plug-ins call CWPanlReadRelativePathAEDesc in response to a reqAESetPref request, to extract a relative path setting from an Apple Event descriptor.

A plug-in passes the AEDesc from the prefsDesc field of the PanelParameterBlock in the desc field. The IDE extracts the relative path information from the Apple Event descriptor and stores it in the path structure. desc remains the property of the plug-in.

See Also    “CWPanlWriteRelativePathAEDesc” on page 317

### CWPanlReadRelativePathSetting

Description    The Mac OS version of CWReadRelativePathSetting.

```
Prototype #include <CWDropInPanel.h>
CW_CALLBACK CWPanlReadRelativePathSetting(
 PanelParamBlkPtr ppb,
 const char* name,
 CWRelativePath* value);
```

### CWPanlReadStringSetting

Description    The Mac OS version of CWReadStringSetting.

```
Prototype #include <CWDropInPanel.h>
```

## Settings Panel Plug-in API Reference

### *CWPanlRemoveUserItem*

---

```
CW_CALLBACK CWPanlReadStringSetting(
 PanelParamBlkPtr ppb,
 const char* name,
 const char** value);
```

### **CWPanlRemoveUserItem**

Description Obsolete call to the IDE.

```
Prototype #include <CWDropInPanel.h>
CW_CALLBACK CWPanlRemoveUserItem(
 PanelParamBlkPtr ppb,
 short whichItem);
```

### **CWPanlSetBooleanValue**

Description The Mac OS version of CWSetBooleanValue.

```
Prototype #include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetBooleanValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 Boolean value);
```

### **CWPanlSetFloatingPointValue**

Description The Mac OS version of CWSetFloatingPointValue.

```
Prototype #include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetFloatingPointValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 double value);
```

### **CWPanlSetIntegerValue**

Description The Mac OS version of CWSetIntegerValue.

```
Prototype #include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetIntegerValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 long value);
```

## CWPanlSetItemData

**Description** The Mac OS version of CWPanlSetItemData.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetItemData(
 PanelParamBlkPtr ppb,
 long whichItem,
 void* inData,
 long inDataLength);
```

## CWPanlSetItemMaxLength

**Description** The Mac OS version of CWPanlSetItemMaxLength.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetItemMaxLength(
 PanelParamBlkPtr ppb,
 long whichItem,
 short inLength);
```

## CWPanlSetItemText

**Description** The Mac OS version of CWPanlSetItemText.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetItemText(
 PanelParamBlkPtr ppb,
 long whichItem,
 ConstStr255Param str);
```

## CWPanlSetItemTextHandle

**Description** The Mac OS version of CWPanlSetItemTextHandle.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetItemTextHandle(
 PanelParamBlkPtr ppb,
 long whichItem,
 Handle text);
```

## Settings Panel Plug-in API Reference

### CWPanlSetItemValue

**Description** The Mac OS version of CWPanlSetItemValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetItemValue(
 PanelParamBlkPtr ppb,
 long whichItem,
 long value);
```

### CWPanlSetRelativePathValue

**Description** The Mac OS version of CWPanlSetRelativePathValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetRelativePathValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 const CWRelativePath* value);
```

### CWPanlSetStringValue

**Description** The Mac OS version of CWPanlSetStringValue.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlSetStringValue(
 PanelParamBlkPtr ppb,
 CWSettingID settingID,
 const char* value);
```

### CWPanlShowItem

**Description** The Mac OS version of CWPanlShowItem.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWPanlShowItem(
 PanelParamBlkPtr ppb,
 long whichItem,
 Boolean showIt);
```

### CWPanlValidItem

**Description** The Mac OS version of CWPanlValidItem.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanlValidItem(`  
                          `PanelParamBlkPtr   ppb,`  
                          `long                   whichItem);`

### **CWPanlWriteBooleanSetting**

Description    The Mac OS version of CWPanlWriteBooleanSetting.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanlWriteBooleanSetting(`  
                          `PanelParamBlkPtr   ppb,`  
                          `const char*         name,`  
                          `Boolean             value);`

### **CWPanlWriteFloatingPointSetting**

Description    The Mac OS version of CWPanlWriteFloatingPointSetting.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanlWriteFloatingPointSetting(`  
                          `PanelParamBlkPtr   ppb,`  
                          `const char*         name,`  
                          `double              value);`

### **CWPanlWriteIntegerSetting**

Description    The Mac OS version of CWPanlWriteIntegerSetting.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanlWriteIntegerSetting(`  
                          `PanelParamBlkPtr   ppb,`  
                          `const char*         name,`  
                          `long                 value);`

### **CWPanlWriteRelativePathAEDesc**

Description    Constructs an Apple Event descriptor for a given relative path.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWPanlWriteRelativePathAEDesc(`  
                          `PanelParamBlkPtr         ppb,`  
                          `const CWRelativePath*    path,`

## Settings Panel Plug-in API Reference

### CWPanlWriteRelativePathSetting

|            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-------------------------------------------------------------------------------------------|------|-------------------------------------------------------------------------------|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
|            | <code>AEDesc* desc);</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
| Parameters | The parameters for this method include:                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
|            | <table> <tr> <td>ppb</td><td>Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests.</td></tr> <tr> <td>path</td><td>Specifies the relative path for which to construct an Apple Event descriptor.</td></tr> <tr> <td>desc</td><td>Returns an Apple Event descriptor describing the relative path specified in path. The IDE allocates this descriptor, which becomes the property of the plug-in.</td></tr> </table>                                                              | ppb | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests. | path | Specifies the relative path for which to construct an Apple Event descriptor. | desc | Returns an Apple Event descriptor describing the relative path specified in path. The IDE allocates this descriptor, which becomes the property of the plug-in. |
| ppb        | Parameter block passed to Mac OS panels, required by the IDE to service plug-in requests.                                                                                                                                                                                                                                                                                                                                                                                                                            |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
| path       | Specifies the relative path for which to construct an Apple Event descriptor.                                                                                                                                                                                                                                                                                                                                                                                                                                        |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
| desc       | Returns an Apple Event descriptor describing the relative path specified in path. The IDE allocates this descriptor, which becomes the property of the plug-in.                                                                                                                                                                                                                                                                                                                                                      |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
| Remarks    | <p>CWPanlWriteRelativePathAEDesc simplifies the task of creating an Apple Event descriptor for a CWRelativePath record. Plug-ins call CWPanlWriteRelativePathAEDesc in response to a reqAEGetPref request, to return an Apple Event descriptor for a relative path setting.</p> <p>The AEDesc returned in desc becomes the property of the plug-in. Generally, the plug-in returns the AEDesc object in the prefsDesc field of the PanelParameterBlock. The plug-in should release the AEDesc when done with it.</p> |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |
| See Also   | <p>“CWPanlReadRelativePathAEDesc” on page 312</p> <p>“reqAEGetPref” on page 350</p>                                                                                                                                                                                                                                                                                                                                                                                                                                  |     |                                                                                           |      |                                                                               |      |                                                                                                                                                                 |

## CWPanlWriteRelativePathSetting

|             |                                                                                                                                                                                       |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | The Mac OS version of CWPanlWriteRelativePathSetting.                                                                                                                                 |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWPanlWriteRelativePathSetting(     PanelParamBlkPtr    ppb,     const char*          name,     const CWRelativePath* value);</pre> |

## CWPanlWriteStringSetting

|             |                                                                                   |
|-------------|-----------------------------------------------------------------------------------|
| Description | The Mac OS version of CWPanlWriteStringSetting.                                   |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWPanlWriteStringSetting(</pre> |

```
PanelParamBlkPtr ppb,
const char* name,
const char* value);
```

### CWReadBooleanSetting

**Description** Reads a boolean value from the top level of a panel’s XML input stream.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWReadBooleanSetting(
 CWPluginContext context,
 const char* name,
 Boolean* value);
```

**Parameters** The parameters for this method include:

|         |                                                            |
|---------|------------------------------------------------------------|
| context | Private, opaque IDE state.                                 |
| name    | A C string specifying the name of the XML setting to read. |
| value   | Returns the boolean value of the named setting.            |

**Remarks** CWReadBooleanSetting returns the value of a top-level boolean XML setting (a boolean value contained in the panel’s XML stream, but not in a structure or array). If the IDE finds no boolean setting with the specified name, the IDE returns an error.

CWReadBooleanSetting returns an error if the named setting cannot be found or is not boolean. CWReadBooleanSetting should only be called in response to a reqReadSettings request.

To read a boolean component of an XML structure or array, use CWGetBooleanValue.

**See Also** “reqReadSettings” on page 364

“CWGetBooleanValue” on page 262

### CWReadFloatingPointSetting

**Description** Reads a floating point value from the top level of a panel’s XML input stream.

## Settings Panel Plug-in API Reference

### CWReadIntegerSetting

**Prototype**    `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWReadFloatingPointSetting(`  
`CWPluginContext    context,`  
`const char*        name,`  
`double*            value);`

**Parameters**    The parameters for this method include:

|         |                                                            |
|---------|------------------------------------------------------------|
| context | Private, opaque IDE state.                                 |
| name    | A C string specifying the name of the XML setting to read. |
| value   | Returns the floating point value of the named setting.     |

**Remarks**    CWReadFloatingPointSetting returns the value of a top-level floating point XML setting (a floating point value contained in the panel's XML stream, but not in a structure or array). If the IDE finds no floating point value with the specified name, the IDE returns an error.

CWReadFloatingPointSetting returns an error if the given setting does not exist or is not a floating point value. Plug-ins should only call CWReadFloatingPointSetting should in response to a reqReadSettings request.

To read a floating point component of an XML structure or array, use CWGetFloatingPointValue.

**See Also**        "reqReadSettings" on page 364

" CWGetFloatingPointValue" on page 263

## CWReadIntegerSetting

**Description**    Reads a long integer value from the top level of a panel's XML input stream.

**Prototype**    `#include <CWDropInPanel.h>`  
`CW_CALLBACK CWReadIntegerSetting(`  
`CWPluginContext    context,`  
`const char*        name,`  
`long*              value);`

**Parameters**    The parameters for this method include:

|         |                                                            |
|---------|------------------------------------------------------------|
| context | Private, opaque IDE state.                                 |
| name    | A C string specifying the name of the XML setting to read. |
| value   | Returns the long integer value of the named setting.       |

**Remarks** *CWReadIntegerSetting* returns the value of a top-level integer XML setting (an integer value contained in the panel’s XML stream, but not in a structure or array). If the IDE finds no integer setting with the specified name, the IDE returns an error.

*CWReadIntegerSetting* returns an error if the given setting does not exist or is not an integer value. Plug-ins should only call *CWReadIntegerSetting* in response to a *reqReadSettings* request.

To read a *CWReadIntegerSetting* component of an XML structure or array, use *CWGetIntegerValue*.

**See Also** “*reqReadSettings*” on page 364

*CWGetIntegerValue*

### CWReadRelativePathSetting

**Description** Reads a relative path value from the top level of a panel’s XML input stream.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWReadRelativePathSetting(
 CWPluginContext context,
 const char* name,
 CWRelativePath* value);
```

**Parameters** The parameters for this method include:

|         |                                                            |
|---------|------------------------------------------------------------|
| context | Private, opaque IDE state.                                 |
| name    | A C string specifying the name of the XML setting to read. |
| value   | Returns the relative path value of the named setting.      |

## Settings Panel Plug-in API Reference

### CWReadStringSetting

- Remarks

CWReadRelativePathSetting returns the value of a top-level relative path XML setting (a relative path value contained in the panel's XML stream, but not in a structure or array). If the IDE finds no relative path setting with the specified name, the IDE returns an error.

CWReadRelativePathSetting returns an error if the given setting does not exist or is not of relative path type. Plug-ins should only call CWReadRelativePathSetting in response to a reqReadSettings request.

To read a relative path component of an XML structure or array, use CWGetRelativePathValue.
- See Also

"reqReadSettings" on page 364

"CWGetRelativePathValue" on page 266

### CWReadStringSetting

- Description

Reads a string value from the top level of a panel's XML input stream.
- Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWReadStringSetting(
 CWPluginContext context,
 const char* name,
 const char** value);
```
- Parameters

The parameters for this method include:

|         |                                                                                                                    |
|---------|--------------------------------------------------------------------------------------------------------------------|
| context | Private, opaque IDE state.                                                                                         |
| name    | A C string specifying the name of the XML setting to read.                                                         |
| value   | Returns a C string containing the value of the named setting. The string returned remains the property of the IDE. |
- Remarks

CWReadStringSetting returns the value of a top-level string XML setting (a string value contained in the panel's XML stream, not in a structure or array). If the IDE finds no string setting with the specified name, the IDE returns an error. Otherwise, the IDE returns the string setting's value is returned in a newly allocated string.



**CAUTION** The returned string remains the property of the IDE. The IDE can release the string at the end of the pending request. Therefore, to retain the text returned in `value`, copy the string into another string before returning to the IDE.

`CWReadStringSetting` returns an error if the named setting does not exist or is not of string type. Plug-ins should only call `CWReadStringSetting` in response to a `reqReadSettings` request.

To read a string component of an XML structure or array, use `CWGetStringValue`.

See Also “`reqReadSettings`” on page 364  
“`CWGetStringValue`” on page 266

**CWSetBooleanValue**

Description Sets the value of a boolean element of an XML array or structure.

Prototype 

```
#include <CWDropInPanel.h>
CW_CALLBACK CWSetBooleanValue(
 CWPluginContext context,
 CWSettingID settingID,
 Boolean value);
```

Parameters The parameters for this method include:

|           |                                            |
|-----------|--------------------------------------------|
| context   | Private, opaque IDE state.                 |
| settingID | Specifies the ID of the setting to change. |
| value     | Specifies the new setting value.           |

Remarks `CWSetBooleanValue` sets the type of an XML setting to boolean, and specifies its value. Plug-ins call `CWSetBooleanValue` in response to a `reqWriteSettings` request, to write out the value of a boolean array element or structure field.

If a `CWSet...Value` call has been previously issued for the setting specified by `settingID`, the IDE changes its type to boolean and its value to `value`.

## Settings Panel Plug-in API Reference

### CWSetFloatingPointValue

To create a boolean array element setting or structure field setting and obtain its ID, call `CWGetArraySettingElement` or `CWGetStructureSettingField`, respectively. To write out a top-level boolean setting, use `CWWriteBooleanSetting`.

**TIP** `CWSetBooleanValue` can be used to change the value of a top-level setting, whose ID was obtained from `CWGetNamedSetting`. However, `CWWriteBooleanSetting` offers more direct method.

See Also “reqWriteSettings” on page 368  
“CWGetArraySettingElement” on page 259  
“CWGetStructureSettingField” on page 267  
“CWWriteBooleanSetting” on page 328

## CWSetFloatingPointValue

**Description** Sets the value of a floating point element of an XML array or structure.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWSetFloatingPointValue(
 CWPluginContext context,
 CWSettingID settingID,
 double value);
```

**Parameters** The parameters for this method include:

|           |                                            |
|-----------|--------------------------------------------|
| context   | Private, opaque IDE state.                 |
| settingID | Specifies the ID of the setting to change. |
| value     | Specifies the new setting value.           |

**Remarks** `CWSetFloatingPointValue` sets the type of an XML setting to floating point and specifies its value. Plug-ins call `CWSetFloatingPointValue` in response to a `reqWriteSettings` request, to write out the value of a floating point array element or structure field.



If a CWSet...Value call has been previously issued for the setting specified by settingID, the IDE changes its type to floating point and changes its value to value.

To create a floating point array element setting or structure field setting and obtain its ID, call CWGetArraySettingElement or CWGetStructureSettingField, respectively. To write out a top-level floating point setting, use CWWriteFloatingPointSetting.

**TIP** Plugins can use CWSetFloatingPointValue to change the value of a top-level setting, whose ID was obtained from CWGetNamedSetting. However, CWWriteFloatingPointSetting offers a more direct method.

- See Also
- “reqWriteSettings” on page 368
  - “CWGetArraySettingElement” on page 259
  - “CWGetStructureSettingField” on page 267
  - “CWWriteFloatingPointSetting” on page 329

CWSetIntegerValue

Description Sets the value of an integer element of an XML array or structure.

Prototype

```
#include <CWDropInPanel.h>
CW_CALLBACK CWSetIntegerValue(
 CWPluginContext context,
 CWSettingID settingID,
 long value);
```

Parameters The parameters for this method include:

|           |                                            |
|-----------|--------------------------------------------|
| context   | Private, opaque IDE state.                 |
| settingID | Specifies the ID of the setting to change. |
| value     | Specifies the new setting value.           |

Remarks CWSetIntegerValue sets the type of an XML setting to integer, and specifies its value. plug-ins call CWSetIntegerValue in

## Settings Panel Plug-in API Reference

### CWSetRelativePathValue

response to a reqWriteSettings request, to write out the value of an integer array element or structure field.

If a CWSet...Value call has been previously issued for the setting specified by settingID, the IDE changes its type to integer and changes its value to value.

To create an integer array element setting or structure field setting and obtain its ID, call CWGetArraySettingElement or CWGetStructureSettingField, respectively. To write out a top-level integer setting, use CWWriteIntegerSetting.

**TIP** Plug-ins can use CWSetIntegerValue to change the value of a top-level setting, whose ID was obtained from CWGetNamedSetting. However, CWWriteIntegerSetting offers a more direct method.

See Also “reqWriteSettings” on page 368  
“CWGetArraySettingElement” on page 259  
“CWGetStructureSettingField” on page 267  
“CWWriteIntegerSetting” on page 330

### CWSetRelativePathValue

**Description** Sets the value of a relative path element of an XML array or structure.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWSetRelativePathValue(
 CWPluginContext context,
 CWSettingID settingID,
 const CWRelativePath* value);
```

**Parameters** The parameters for this method include:

|           |                                            |
|-----------|--------------------------------------------|
| context   | Private, opaque IDE state.                 |
| settingID | Specifies the ID of the setting to change. |
| value     | Specifies the new setting value.           |

|          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remarks  | <p><code>CWSetRelativePathValue</code> sets the type of an XML setting to relative path, and specifies its value. plug-ins call <code>CWSetRelativePathValue</code> in response to a <code>reqWriteSettings</code> request, to write out the value of a relative path array element or structure field.</p> <p>If a <code>CWSet...Value</code> call has been previously issued for the setting specified by <code>settingID</code>, the IDE changes its type to relative path and changes its value to <code>value</code>.</p> <p>To create a relative path array element setting or structure field setting and obtain its ID, call <code>CWGetArraySettingElement</code> or <code>CWGetStructureSettingField</code>, respectively. To write out a top-level relative path setting, use <code>CWWriteRelativePathSetting</code>.</p> |
| TIP      | <p>Plug-ins can use <code>CWSetRelativePathValue</code> to change the value of a top-level setting, whose ID was obtained from <code>CWGetNamedSetting</code>. However, <code>CWWriteRelativePathSetting</code> offers a more direct method.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| See Also | <p>“<code>reqWriteSettings</code>” on page 368</p> <p>“<code>CWGetArraySettingElement</code>” on page 259</p> <p>“<code>CWGetStructureSettingField</code>” on page 267</p> <p>“<code>CWWriteRelativePathSetting</code>” on page 330</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

### CWSetStringValue

|             |                                                                                                                                                                     |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Sets the value of a string element of an XML array or structure.                                                                                                    |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWSetStringValue(     CWPluginContext context,     CWSettingID      settingID,     const char*      value);</pre> |
| Parameters  | The parameters for this method include:                                                                                                                             |

## Settings Panel Plug-in API Reference

### CWWriteBooleanSetting

|           |                                            |
|-----------|--------------------------------------------|
| context   | Private, opaque IDE state.                 |
| settingID | Specifies the ID of the setting to change. |
| value     | Specifies the new setting value.           |

**Remarks** CWSetStringValue sets the type of an XML setting to string and specifies its value. Plug-ins call CWSetStringValue in response to a reqWriteSettings request to write out the value of a string array element or structure field.

If a CWSet...Value call has been previously issued for the setting specified by settingID, the IDE changes its type to string and changes its value to value.

To create a string array element setting or structure field setting and obtain its ID, call CWGetArraySettingElement or CWGetStructureSettingField, respectively. To write out a top-level string setting, use CWWriteStringSetting.

**TIP** Plug-ins can use CWSetStringValue to change the value of a top-level setting, whose ID was obtained from CWGetNamedSetting. However, CWWriteStringSetting offers a more direct method.

**See Also** “reqWriteSettings” on page 368  
 “CWGetArraySettingElement” on page 259  
 “CWGetStructureSettingField” on page 267  
 “CWWriteStringSetting” on page 331

## CWWriteBooleanSetting

**Description** Writes a boolean value to the top level of a panel’s exported XML data.

**Prototype**

```
#include <CWDropInPanel.h>
CW_CALLBACK CWWriteBooleanSetting(
 CWPluginContext context,
 const char* name,
 Boolean value);
```

|            |                                                                                                                                                                                                                                                                                                          |         |                            |      |                                                                       |       |                                  |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----------------------------|------|-----------------------------------------------------------------------|-------|----------------------------------|
| Parameters | <p>The parameters for this method include:</p> <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>name</td><td>Specifies the name of the top-level setting to create, as a C string.</td></tr> <tr> <td>value</td><td>Specifies the new setting value.</td></tr> </table>    | context | Private, opaque IDE state. | name | Specifies the name of the top-level setting to create, as a C string. | value | Specifies the new setting value. |
| context    | Private, opaque IDE state.                                                                                                                                                                                                                                                                               |         |                            |      |                                                                       |       |                                  |
| name       | Specifies the name of the top-level setting to create, as a C string.                                                                                                                                                                                                                                    |         |                            |      |                                                                       |       |                                  |
| value      | Specifies the new setting value.                                                                                                                                                                                                                                                                         |         |                            |      |                                                                       |       |                                  |
| Remarks    | <p>CWWriteBooleanSetting writes the value of a top-level boolean setting to the plug-in's XML output stream. Plug-ins should only call CWWriteBooleanSetting in response to a reqWriteSettings request.</p> <p>To write a boolean component of an XML structure or array, use CWPanlSetBooleanValue.</p> |         |                            |      |                                                                       |       |                                  |
| See Also   | <p>"reqWriteSettings" on page 368</p> <p>"CWSetBooleanValue" on page 323</p>                                                                                                                                                                                                                             |         |                            |      |                                                                       |       |                                  |

### CWWriteFloatingPointSetting

|             |                                                                                                                                                                                                                                                                                                       |         |                            |      |                                                                       |       |                                  |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----------------------------|------|-----------------------------------------------------------------------|-------|----------------------------------|
| Description | Writes a floating point value to the top level of a panel's exported XML data.                                                                                                                                                                                                                        |         |                            |      |                                                                       |       |                                  |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CW_CALLBACK CWWriteFloatingPointSetting(     CWDropInContext context,     const char*      name,     double           value);</pre>                                                                                                                             |         |                            |      |                                                                       |       |                                  |
| Parameters  | <p>The parameters for this method include:</p> <table> <tr> <td>context</td><td>Private, opaque IDE state.</td></tr> <tr> <td>name</td><td>Specifies the name of the top-level setting to create, as a C string.</td></tr> <tr> <td>value</td><td>Specifies the new setting value.</td></tr> </table> | context | Private, opaque IDE state. | name | Specifies the name of the top-level setting to create, as a C string. | value | Specifies the new setting value. |
| context     | Private, opaque IDE state.                                                                                                                                                                                                                                                                            |         |                            |      |                                                                       |       |                                  |
| name        | Specifies the name of the top-level setting to create, as a C string.                                                                                                                                                                                                                                 |         |                            |      |                                                                       |       |                                  |
| value       | Specifies the new setting value.                                                                                                                                                                                                                                                                      |         |                            |      |                                                                       |       |                                  |
| Remarks     | <p>CWWriteFloatingPointSetting writes the value of a top-level floating point setting to the plug-in's XML output stream. Plug-ins should only call CWWriteFloatingPointSetting in response to a reqWriteSettings request.</p>                                                                        |         |                            |      |                                                                       |       |                                  |



Settings Panel Plug-in API Reference  
*CWWriteIntegerSetting*

---

To write a floating point component of an XML structure or array, use `CWPanelSetFloatingPointValue` instead.

See Also    “`reqWriteSettings`” on page 368  
              “`CWSetFloatingPointValue`” on page 324

**CWWriteIntegerSetting**

Description    Writes an integer value to the top level of a panel’s exported XML data.

Prototype    `#include <CWDropInPanel.h>`  
              `CW_CALLBACK CWWriteIntegerSetting(`  
                  `CWPluginContext context,`  
                  `const char* name,`  
                  `long value);`

Parameters    The parameters for this method include:

|         |                                                                       |
|---------|-----------------------------------------------------------------------|
| context | Private, opaque IDE state.                                            |
| name    | Specifies the name of the top-level setting to create, as a C string. |
| value   | Specifies the new setting value.                                      |

Remarks    `CWWriteIntegerSetting` writes the value of a top-level integer setting to the plug-in’s XML output stream. Plug-ins should only call `CWWriteIntegerSetting` in response to a `reqWriteSettings` request.

To write a integer component of an XML structure or array, use `CWPanelSetIntegerValue`.

See Also    “`reqWriteSettings`” on page 368  
              “`CWSetIntegerValue`” on page 325

**CWWriteRelativePathSetting**

Description    Writes a relative path value to the top level of a panel’s exported XML data.

Prototype    `#include <CWDropInPanel.h>`

```
CW_CALLBACK CWWriteRelativePathSetting(
 CWPluginContext context,
 const char* name,
 const CWRelativePath* value);
```

Parameters The parameters for this method include:

|         |                                                                       |
|---------|-----------------------------------------------------------------------|
| context | Private, opaque IDE state.                                            |
| name    | Specifies the name of the top-level setting to create, as a C string. |
| value   | Specifies the new setting value.                                      |

Remarks CWWriteRelativePathSetting writes the value of a top-level relative path setting to the plug-in’s XML output stream. Plug-ins should only call CWWriteRelativePathSetting in response to a reqWriteSettings request.

To write a relative path component of an XML structure or array, use CWPanelSetRelativePathValue.

See Also “reqWriteSettings” on page 368  
 “CWSetRelativePathValue” on page 326

### CWWriteStringSetting

Description Writes a string value to the top level of a panel’s exported XML data.

```
#include <CWDropInPanel.h>
CW_CALLBACK CWWriteStringSetting(
 CWPluginContext context,
 const char* name,
 const char* value);
```

Parameters The parameters for this method include:

|         |                                                                       |
|---------|-----------------------------------------------------------------------|
| context | Private, opaque IDE state.                                            |
| name    | Specifies the name of the top-level setting to create, as a C string. |
| value   | Specifies the new setting value.                                      |



# Freescale Semiconductor, Inc.

## Settings Panel Plug-in API Reference

User Routines for Settings Panel Plug-ins

|          |                                                                                                                                                                                                                                                                                                     |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remarks  | <p>CWWriteStringSetting writes the value of a top-level string setting to the plug-in's XML output stream. Plug-ins should only call CWWriteStringSetting in response to a reqWriteSettings request.</p> <p>To write a string component of an XML structure or array, use CWPanlSetStringValue.</p> |
| See Also | <p>"reqWriteSettings" on page 368</p> <p>"CWSetStringValue" on page 327</p>                                                                                                                                                                                                                         |

## User Routines for Settings Panel Plug-ins

This section documents informational entry points specific to panel plug-ins.

The routines documented in this section are:

- CWPlugin\_GetHelpInfo Entry Point

### CWPlugin\_GetHelpInfo Entry Point

|             |                                                                                                                                                                                                                                                                                                                |          |                                                    |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|----------------------------------------------------|
| Description | An optional entry point implemented by Windows plug-ins to inform the IDE about help information associated with a panel.                                                                                                                                                                                      |          |                                                    |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; CWPLUGIN_ENTRY (CWPlugin_GetHelpInfo)     (const CWHelpInfo** helpInfo);</pre>                                                                                                                                                                                           |          |                                                    |
| Parameters  | <p>The parameters for this method include:</p> <table><tr><td>helpInfo</td><td>The plug-in returns a pointer to help information.</td></tr></table>                                                                                                                                                            | helpInfo | The plug-in returns a pointer to help information. |
| helpInfo    | The plug-in returns a pointer to help information.                                                                                                                                                                                                                                                             |          |                                                    |
| Remarks     | <p>Windows plug-ins can use this optional entry point to specify the name of a WinHelp help file to be displayed when the user clicks <b>Help</b> in the <b>Target Settings</b> dialog.</p> <p>The IDE searches for plug-in help files in the Help directory within the CodeWarrior IDE install directory.</p> |          |                                                    |
| See Also    | <p>"Help Information Entry Point" in the <i>IDE SDK Developer's Guide</i></p> <p>"CWHelpInfo" on page 333</p>                                                                                                                                                                                                  |          |                                                    |

## Data Structures for Settings Panel Plug-ins

This section discusses the following topics:

- CWDIALOG
- CWHelpInfo
- CWSettingID
- PanelFlags
- PanelParamBlkPtr
- PanelParameterBlock

### CWDIALOG

|             |                                                                                                                                                                                                                                                                                                |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Specifies the type of a Mac OS panel dialog.                                                                                                                                                                                                                                                   |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; #if CW_STRICT_DIALOGS     typedef struct DummyDialog* CWDIALOG; #else     typedef DialogPtr            CWDIALOG; #endif</pre>                                                                                                                            |
| Remarks     | <p>Depending on the definition of CW_STRICT_DIALOGS, this data type is equivalent to an opaque type or to the Mac OS DialogPtr type.</p> <p>All new Mac OS panel plug-ins should be compiled with CW_STRICT_DIALOGS enabled and should not use the dialog member of a PanelParameterBlock.</p> |
| See Also    | <p>“CW_STRICT_DIALOGS” on page 345</p> <p>“PanelParameterBlock” on page 336</p>                                                                                                                                                                                                                |

### CWHelpInfo

|             |                                                                                                                 |
|-------------|-----------------------------------------------------------------------------------------------------------------|
| Description | Describes a Windows panel’s associated help file to the IDE.                                                    |
| Prototype   | <pre>#include &lt;CWPlugins.h&gt; /* Windows */ typedef struct CWHelpInfo {     short            version;</pre> |

## Settings Panel Plug-in API Reference

### CWSettingID

```
const char * helpFileName;
} CWHelpInfo;
```

The fields in this structure are:

|              |                                                                                                           |
|--------------|-----------------------------------------------------------------------------------------------------------|
| version      | Specifies the version of the CWHelpInfo structure. plug-ins should set this to kCurrentCWHelpInfoVersion. |
| helpFileName | Specifies the name of a settings panel's associated WinHelp file as a C string.                           |

- Remarks Windows panels optionally return a pointer to a CWHelpInfo structure to tell the IDE the name of any associated help file. The help file name should end with a '.hlp' extension. This help file is displayed by the IDE when the user clicks the **Help** button in the **Target Settings** dialog, and should be a standard WinHelp file.
- See Also "Help Informaton Entry Point" in the *IDE SDK Developer's Guide*  
 "kCurrentCWHelpInfoVersion" on page 346

### CWSettingID

- Description Identifies a container for a simple XML data type, an XML array type, or an XML structure type.
- Prototype
 

```
#include <CWDropInPanel.h> /* Windows */
#include <CWDropInPanel.h> /* Mac OS */
typedef struct MWSetting* CWSettingID;
```
- Remarks A CWSettingID represents an XML setting container, associating an XML name with a data value of a particular type. CWSettingIDs can store array, structure, and simple types.  
  
 Plug-ins call CWGetNamedSetting to obtain CWSettingIDs for simple types. Plug-ins call CWGetStructureSettingField to obtain CWSettingIDs for structure fields, and CWGetArraySettingElement to obtain CWSettingIDs for array elements.
- See Also "CWGetNamedSetting" on page 264  
 "CWGetStructureSettingField" on page 267

“CWGetArraySettingElement” on page 259

### PanelFlags

|                              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------------------------------------------------|------------|--------------------------------------------------------------------|------------------------------|------------------------------------------------------------------|-------------|------------------------------------------------------------------------------------------------------------------------------|-------------|------------------------------------------------------------------------------------------------------------------------------|------------------|---------------------------------------------------------------------------------------------------------------------|
| Description                  | Describes a panel plug-in’s capabilities to the IDE.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| Prototype                    | <pre>#include &lt;CWDropInPanel.h&gt; /* Windows */ #include &lt;CWDropInPanel.h&gt; /* Mac OS */  typedef struct PanelFlags {     unsigned short    rsrcversion;     CWDDataType       dropintype;     unsigned short    earliestCompatibleAPIVersion;     unsigned long     dropinflags;     CWDDataType       panelfamily;     unsigned short    newestAPIVersion;     unsigned short    dataversion;     unsigned short    panelscope; } PanelFlags;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                               |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| Fields                       | <p>The fields in this structure are:</p> <table> <tr> <td>rsrcversion</td><td>Specifies the version of the PanelFlags structure.</td></tr> <tr> <td>dropintype</td><td>Specifies the plug-in type (should be 'PanL' for settings panels).</td></tr> <tr> <td>earliestCompatibleAPIVersion</td><td>Specifies the earliest API version a plug-in is compatible with.</td></tr> <tr> <td>dropinflags</td><td>Specifies the capabilities of a plug-in. Use a combination of usesStrictAPI, supportsByteSwapping, and supportsTextSettings.</td></tr> <tr> <td>panelfamily</td><td>Specifies the family of a plug-in. See “Standard settings panel family type codes” in the <i>IDE SDK Developer’s Guide</i>.</td></tr> <tr> <td>newestAPIVersion</td><td>Specifies the most recent API version a plug-in is compatible with. Most plug-ins should use DROPINPANELAPIVERSION.</td></tr> </table> | rsrcversion | Specifies the version of the PanelFlags structure. | dropintype | Specifies the plug-in type (should be 'PanL' for settings panels). | earliestCompatibleAPIVersion | Specifies the earliest API version a plug-in is compatible with. | dropinflags | Specifies the capabilities of a plug-in. Use a combination of usesStrictAPI, supportsByteSwapping, and supportsTextSettings. | panelfamily | Specifies the family of a plug-in. See “Standard settings panel family type codes” in the <i>IDE SDK Developer’s Guide</i> . | newestAPIVersion | Specifies the most recent API version a plug-in is compatible with. Most plug-ins should use DROPINPANELAPIVERSION. |
| rsrcversion                  | Specifies the version of the PanelFlags structure.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| dropintype                   | Specifies the plug-in type (should be 'PanL' for settings panels).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| earliestCompatibleAPIVersion | Specifies the earliest API version a plug-in is compatible with.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| dropinflags                  | Specifies the capabilities of a plug-in. Use a combination of usesStrictAPI, supportsByteSwapping, and supportsTextSettings.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| panelfamily                  | Specifies the family of a plug-in. See “Standard settings panel family type codes” in the <i>IDE SDK Developer’s Guide</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |
| newestAPIVersion             | Specifies the most recent API version a plug-in is compatible with. Most plug-ins should use DROPINPANELAPIVERSION.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |             |                                                    |            |                                                                    |                              |                                                                  |             |                                                                                                                              |             |                                                                                                                              |                  |                                                                                                                     |

## Settings Panel Plug-in API Reference

### PanelParamBlkPtr

|             |                                                                                                                                                                              |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dataversion | Specifies the version number of a panel's settings data.                                                                                                                     |
| panelscope  | Specifies the scope of a plug-in, which determines which CodeWarrior IDE dialog the plug-in appears in. Use one of panelScopeGlobal, panelScopeProject, or panelScopeTarget. |

**Remarks** The PanelFlags structure reports information to the IDE about a plug-in's capabilities. The CWPlugin\_GetDropInFlags entry point for a panel plug-in returns a PanelFlags structure.

**See Also** "Specifying Preference Panel Capabilities" in the *IDE SDK Developer's Guide*  
 "Specifying Panel Family" in the *IDE SDK Developer's Guide*  
 "PanelFlags Structure" in the *IDE SDK Developer's Guide*  
 "Settings Panel Flags" in the *IDE SDK Developer's Guide*

### PanelParamBlkPtr

**Description** A pointer to a PanelParameterBlock.

**Prototype**

```
#include <CWDropInPanel.h>
typedef PanelParameterBlock* PanelParamBlkPtr;
```

**Remarks** PanelParamBlkPtr is a pointer to a panel parameter block, PanelParameterBlock. The IDE passes a PanelParamBlkPtr to the panel plug-in to communicate information to the panel.

**See Also** "PanelParameterBlock" on page 336

### PanelParameterBlock

**Description** Contains information passed to the plug-in by the IDE, and required by the IDE when servicing calls made by panel plug-ins.

**Prototype**

```
#include <CWDropInPanel.h>
/* parameter block -- this is passed to the dropin
at each request */
```



---

```
typedef struct PanelParameterBlock {
 /* common to all dropins */
 long request;
 /* [->] requested action (see below) */
 long version;
 /* [->] version # of shell's API */
 void *context;
 /* [->] reserved for use by shell */
 void *storage;
 /* [->] reserved for use by the dropin */
 FSSpec targetfile;
 /* [->] FSSpec of current project */

 /* specific to panels */
 CWDialog dialog;
 /* [->] pointer to PreferencesÉ dialog */
 Handle originalPrefs;
 /* [->] panel's original options data */
 Handle currentPrefs;
 /* [->] panel's current options data */
 Handle factoryPrefs;
 /* [->] panel's "factory" options data */
 EventRecord *event;
 /* [->] dialog event (for reqFilterEvent) */
 short baseItems;
 /* [->] # of items in dialog shell */
 short itemHit;
 /* [->] for reqFilterEvent and reqItemHit */
 Boolean canRevert;
 /* [->] enable Revert button */
 Boolean canFactory;
 /* [->] enable Factory button */
 Boolean reset;
 /* [->] access paths must be reset */
 Boolean recompile;
 /* [->] files must be recompiled */
 Boolean relink;
 /* [->] project must be relinked */
 AEKeyword prefsKeyword;
 /* [->] for reqAEGetPref and reqAESetPref */
 AEDesc prefsDesc;
}
```

---

## Settings Panel Plug-in API Reference

### PanelParameterBlock

---

```

 /* [->] for reqAESetPref */
 Boolean debugOn;
 /* [->] turning on debugging? */
 FSSpec oldtargfile;
 /* [->] previous project file FSSpec */

 /* version 2 API */
 CWPanellCallbacks* callbacks;

 /* version 3 API */
 Boolean reparse; /* [->] project must
 be reparsed */

 /* version 4 API */
 DragReference dragref;
 /* [->] for drag-related requests */
 Rect dragrect;
 /* [->] rect to track mouse in */
 Point dragmouse;
 /* [->] mouse location during drag */

 /* version 5 API */
 unsigned char toEndian;
 /* [->] for reqByteSwapData, the
 endian we are swapping to */

 /* CWPro 3 temporary placeholders for opaque
 references to prefs data. These will be removed in
 Pro 4. */
 CWMemHandle originalPrefsMemHandle;
 CWMemHandle currentPrefsMemHandle;
 CWMemHandle factoryPrefsMemHandle;
 CWMemHandle panelPrefsMemHandle;

 /* version 11 api */
 long listViewCellRow;
 /* [->] the cell row of the listView */
 long listViewCellCol;
 /* [->] the cell row of the listView */
 /*
 These fields are supported only from
 DropinPanelPrefVersion 12 and higher for long

```

---



filename support. Older plug-ins will get will use FSSpec based fields targetFile and oldTargetFile

Since the IDE is LFN enabled, the will attempt to convert the internal LFN data struture to create a valid FSSpec. If the API fails to create a valid FSSpec, the FSSpec name field would be filled with the project name.

The newer plug-ins or those plug-ins that decide to upgrade to the newer library(PluginLib5) will have to be aware that these fields are the new means of accessing the target file and old target file. The older fsspec based fields will be deprecated for the newer plug-ins (version 12 and higher) and will be zeroed out.  
\*/

```
/* version 12 api */
CWPanelFileSpec lfnTargetFile;
CWPanelFileSpec lfnOldTargetFile;

} PanelParameterBlock, *PanelParameterBlockPtr;
```

Fields The fields in PanelParameterBlock include:

|         |                                                                                                                                                                                                                                                       |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| request | Specifies the action requested of a panel by the IDE. Also determines which of the remaining parameter block fields are required to service the request.                                                                                              |
| version | Specifies the current version of the panel API.                                                                                                                                                                                                       |
| context | Reserved for use by the IDE. Do not modify this field.                                                                                                                                                                                                |
| storage | Reserved for use by the panel drop-in. The IDE guarantees that the contents of this field will be preserved between reqInitPanel and reqTermPanel requests to the panel. Most plug-ins should <i>not</i> use this field. See important warning below. |

## Settings Panel Plug-in API Reference

PanelParameterBlock

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| targetfile    | Specifies the location of the current project file.                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| dialog        | Contains a pointer to the IDE's <b>Target Settings</b> dialog box for plug-ins that clear the <code>usesStrictAPI</code> flag (see the <i>IDE SDK Developer's Guide</i> for detail about this flag). All new plug-ins should set this flag, and be compiled with <code>CW_STRICT_DIALOGS</code> set to true, and so this field should not be used.                                                                                                                                       |
| originalPrefs | Contains a copy of the most recently saved settings data. Plug-ins should not modify this field.                                                                                                                                                                                                                                                                                                                                                                                         |
| currentPrefs  | Contains the current panel settings data. Plug-ins are expected to modify this field.                                                                                                                                                                                                                                                                                                                                                                                                    |
| factoryPrefs  | Contains the default settings data for the panel. Plug-ins modify this field when the IDE sends the <code>reqGetFactory</code> request.                                                                                                                                                                                                                                                                                                                                                  |
| event         | Contains a copy of the most recently received (not yet handled) Mac OS event. Plug-ins which implement user items (special controls) may need to examine and possibly modify this field when responding to a <code>reqFilter</code> request.                                                                                                                                                                                                                                             |
| baseItems     | Indicates the number of items in the host dialog, prior to addition of the panel's controls. Plug-ins should subtract this number from <code>itemHit</code> when responding to <code>reqItemHit</code> , <code>reqDrawCustomItem</code> , <code>reqActivateItem</code> , <code>reqDeactivateItem</code> , <code>reqHandleKey</code> , and <code>reqHandleClick</code> requests. The resulting adjusted item number corresponds to item numbers in the panel's original 'PPob' item list. |



|              |                                                                                                                                                                                                                                                                 |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| itemHit      | Specifies the number of the item with which the user has just finished interacting. Also used to specify <b>Edit</b> menu items, for reqFindStatus and reqObeyCommand requests. Plug-ins can modify this value to simulate hits in other items                  |
| canRevert    | Plug-ins set this flag when responding to a reqItemHit request, to indicate whether the current settings differ from the most recently saved (original) settings. When set true, the IDE enables the <b>Revert</b> button in the <b>Target Settings</b> dialog. |
| canFactory   | Plug-ins set this flag when responding to a reqItemHit request, to indicate whether the current settings differ from the factory default settings. When set true, the IDE enables the <b>Factory Settings</b> button in the <b>Target Settings</b> dialog.      |
| reset        | Plug-ins set this flag when responding to a reqValidate request, to indicate whether the current settings require the IDE to re-scan for project files in the project's access paths. When set true, the IDE re-scans for files.                                |
| recompile    | Plug-ins set this flag when responding to a reqValidate request, to indicate whether the current settings require the IDE to recompile all files in the project. When set true, the IDE recompiles the project.                                                 |
| relink       | Plug-ins set this flag when responding to a reqValidate request, to indicate whether the current settings require the IDE to re-link the target executable. When set true, the IDE re-links the project.                                                        |
| prefsKeyword | Specifies the Apple Event keyword for the pending Apple Event as a four-character code.                                                                                                                                                                         |

## Settings Panel Plug-in API Reference

### PanelParameterBlock

|                        |                                                                                                                                                                                                                                                                |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prefsDesc              | Contains the data for the pending Apple Event, in an Apple Event descriptor. The plug-in is responsible for disposing this when responding to a reqAESetPref request, and for allocating it when responding to a reqAEGetPref request.                         |
| debugOn                | Specifies whether the user has enabled debugging in the <b>Project</b> menu.                                                                                                                                                                                   |
| oldtargfile            | Specifies the previous name and location of the current project file, prior to being renamed.                                                                                                                                                                  |
| reparse                | Set by a plug-in if the project's files must be reprocessed by the class browser. This field is currently unused.                                                                                                                                              |
| dragref                | Specifies the Drag Manager drag reference for the current drag operation. The panel uses this value when making calls to the Drag Manager.                                                                                                                     |
| dragrect               | Specifies the rectangle that the IDE treats as the drag destination.                                                                                                                                                                                           |
| dragmouse              | Specifies the current mouse location during a drag operation.                                                                                                                                                                                                  |
| toEndian               | Specifies the byte ordering to which setting data should be converted during a reqByteSwapData request (or the <i>current</i> byte ordering of the settings data, during a reqFirstLoad request). Contains either 1 for "big endian" or 2 for "little endian." |
| originalPrefsMemHandle | Obsolete. Do not use.                                                                                                                                                                                                                                          |
| currentPrefsMemHandle  | Obsolete. Do not use.                                                                                                                                                                                                                                          |
| factoryPrefsMemHandle  |                                                                                                                                                                                                                                                                |



# Freescale Semiconductor, Inc.

## Settings Panel Plug-in API Reference

PanelParameterBlock

Obsolete. Do not use.

panelPrefsMemHandle

Obsolete. Do not use.

listViewCellRow

Specifies the row within a list view panel.

listViewCellCol

Specifies the column within a list view panel.  
(The comment in the header file is incorrect - it's the column, as the name implies, not the row.)

lfnTargetFile

Specifies the new file name for long filename support. "lfn" stands for "long file name."

lfnOldTargetFile

Specifies the old file name for long filename support. "lfn" stands for "long file name."

Remarks    The CodeWarrior IDE issues commands to Mac OS settings panel plug-ins by calling the panel and passing a parameter block data structure, *PanelParameterBlock* to the plug-in's *main()* entry point. The panel parameter block contains information about the kind of action the panel should perform and other information that the panel might need when servicing the IDE requests.

The IDE does not guarantee that all ds of the parameter block are valid for every request. See individual requests for details of field validity.

**CAUTION**    All instances of the same plug-in share the *storage* field. That is, the IDE provides one copy of the *storage* field per plug-in, not one per instance of a panel. Thus, if two projects using a settings panel plug-in are open simultaneously, the *storage* field is shared by both panels. This sharing can lead to problems accessing global storage. In practice, plug-ins should not use the *storage* field.

## Settings Panel Plug-in API Reference

*Constants for Settings Panel Plug-ins*

---

See Also    “PanelParameterBlock” on page 336

## Constants for Settings Panel Plug-ins

The predefined symbols for settings panels are:

- CW\_STRICT\_DIALOGS
- DROPINPANELAPIVERSION
- menu\_Clear
- menu\_Copy
- menu\_Cut
- menu\_Paste
- menu\_SelectAll
- panelScopeGlobal
- panelScopeProject
- panelScopeTarget
- reqActivateItem
- reqAEGetPref
- reqAESetPref
- reqByteSwapData
- reqDeactivateItem
- reqDragDrop
- reqDragEnter
- reqDragExit
- reqDragWithin
- reqDrawCustomItem
- reqFilter
- reqFindStatus
- reqFirstLoad
- reqGetData
- reqGetFactory
- reqHandleClick
- reqHandleKey

- reqInitDialog
- reqInitPanel
- reqItemHit
- reqObeyCommand
- reqPutData
- reqReadSettings
- reqRenameProject
- reqSetupDebug
- reqTermDialog
- reqTermPanel
- reqUpdatePref
- reqValidate
- reqWriteSettings
- supportsByteSwapping
- supportsTextSettings
- usesStrictAPI

## CW\_STRICT\_DIALOGS

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | A compile-time switch that controls whether a Mac OS hosted settings panel can use the Mac OS dialog manager.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt; #define CW_STRICT_DIALOGS 0</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Remarks     | <p>When CW_STRICT_DIALOGS is defined as any nonzero value, settings panels for a Mac OS host may not utilize Mac OS dialog manager routines to manipulate settings panel controls. Instead, a panel plug-in must rely entirely on the settings panel API to draw, get, and set dialog items.</p> <p>CW_STRICT_DIALOGS controls the type of the dialog field in the PanelParameterBlock passed to the plug-in. When true, dialog is an anonymous pointer type. When false, dialog is a Mac OS DialogPtr. This define controls compile-time syntactic checks, not runtime behavior of the IDE.</p> |

## Settings Panel Plug-in API Reference

### DROPINPANELAPIVERSION

CW\_STRICT\_DIALOGS should be set `true` (nonzero) for all current and future Mac OS plug-ins. The IDE no longer support settings panels compiled with CW\_STRICT\_DIALOGS defined as `false`.

**NOTE** All Mac OS settings panels should define CW\_STRICT\_DIALOGS as `true` and should also set the `usesStrictAPI` bit in their `dropinflags`.

See Also “Settings Panel Flags” in the *IDE SDK Developer’s Guide*

### DROPINPANELAPIVERSION

**Description** Specifies the current version of the settings panel plug-in API at compile time.

**Prototype** `#include <CWDropInPanel.h> /* Windows */`  
`#include <CWDropInPanel.h> /* Mac OS */`

**Remarks** This predefined symbol specifies the current version of the settings panel plug-in API. This value is most useful in specifying compatible IDE versions in the `PanelFlags` structure returned by a panel’s `CWPlugin_GetDropInFlags` entry point.

See Also “Specifying Preference Panel Capabilities” in the *IDE SDK Developer’s Guide*

### kCurrentCWHelpInfoVersion

**Description** Indicates the current version of a `CWHelpInfo` structure.

**Prototype** `#include "CWPlugins.h"`

**Remarks** Windows plug-ins should set the `version` field of a `CWHelpInfo` structure to this value.

See Also “CWHelpInfo” on page 333

“CWPlugin\_GetHelpInfo Entry Point” on page 332

### menu\_Clear

**Description** Represents the **Clear** command in the **Edit** menu.

**Prototype** `#include <CWDropInPanel.h>`



## Freescal Semiconductor, Inc.

### Settings Panel Plug-in API Reference *menu\_Copy*

---

|          |                                                                                                                                                                                                                                                          |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remarks  | The IDE sends this value to Mac OS panel plug-ins during <code>reqFindStatus</code> and <code>reqObeyCommand</code> requests in the <code>itemHit</code> field of the <code>PanelParameterBlock</code> , to indicate the relevant <b>Edit</b> menu item. |
| See Also | “Handling Edit Menu Commands” in the <i>IDE SDK Developer’s Guide</i><br>“reqFindStatus” on page 357<br>“reqObeyCommand” on page 363                                                                                                                     |

### menu\_Copy

|             |                                                                                                                                                                                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Represents the <b>Copy</b> command in the <b>Edit</b> menu.                                                                                                                                                                                              |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                            |
| Remarks     | The IDE sends this value to Mac OS panel plug-ins during <code>reqFindStatus</code> and <code>reqObeyCommand</code> requests in the <code>itemHit</code> field of the <code>PanelParameterBlock</code> , to indicate the relevant <b>Edit</b> menu item. |
| See Also    | “Handling Edit Menu Commands” in the <i>IDE SDK Developer’s Guide</i><br>“reqFindStatus” on page 357<br>“reqObeyCommand” on page 363                                                                                                                     |

### menu\_Cut

|             |                                                                                                                                                                                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Represents the <b>Cut</b> command in the <b>Edit</b> menu.                                                                                                                                                                                               |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                            |
| Remarks     | The IDE sends this value to Mac OS panel plug-ins during <code>reqFindStatus</code> and <code>reqObeyCommand</code> requests in the <code>itemHit</code> field of the <code>PanelParameterBlock</code> , to indicate the relevant <b>Edit</b> menu item. |
| See Also    | “Handling Edit Menu Commands” in the <i>IDE SDK Developer’s Guide</i><br>“reqFindStatus” on page 357<br>“reqObeyCommand” on page 363                                                                                                                     |

## Settings Panel Plug-in API Reference

### *menu\_Paste*

---

#### **menu\_Paste**

|             |                                                                                                                                                                                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Represents the <b>Paste</b> command in the <b>Edit</b> menu.                                                                                                                                                                                             |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                            |
| Remarks     | The IDE sends this value to Mac OS panel plug-ins during <code>reqFindStatus</code> and <code>reqObeyCommand</code> requests in the <code>itemHit</code> field of the <code>PanelParameterBlock</code> , to indicate the relevant <b>Edit</b> menu item. |
| See Also    | <p>“Handling Edit Menu Commands” in the <i>IDE SDK Developer’s Guide</i></p> <p>“<code>reqFindStatus</code>” on page 357</p> <p>“<code>reqObeyCommand</code>” on page 363</p>                                                                            |

#### **menu\_SelectAll**

|             |                                                                                                                                                                                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Represents the <b>Select All</b> command in the <b>Edit</b> menu.                                                                                                                                                                                        |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code><br><code>enum {</code>                                                                                                                                                                                     |
| Remarks     | The IDE sends this value to Mac OS panel plug-ins during <code>reqFindStatus</code> and <code>reqObeyCommand</code> requests in the <code>itemHit</code> field of the <code>PanelParameterBlock</code> , to indicate the relevant <b>Edit</b> menu item. |
| See Also    | <p>“Handling Edit Menu Commands” in the <i>IDE SDK Developer’s Guide</i></p> <p>“<code>reqFindStatus</code>” on page 357</p> <p>“<code>reqObeyCommand</code>” on page 363</p>                                                                            |

#### **panelScopeGlobal**

|             |                                                                                                                                                                               |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Indicates that a settings panel should appear in the <b>Edit &gt; Preferences</b> dialog.                                                                                     |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                 |
| Remarks     | Settings panels return this value in the <code>panelscope</code> field of the <code>PanelFlags</code> structure, from their <code>CWPlugin_GetDropInFlags</code> entry point. |

Settings panels which specify a scope of `panelScopeGlobal` appear in the **Preferences** dialog, and are always enabled, regardless of whether a project is open.

See Also “PanelFlags Structure” in the *IDE SDK Developer’s Guide*

## panelScopeProject

Description Indicates that a settings panel should appear in the **Version Control Settings** dialog box.

Prototype `#include <CWDropInPanel.h>`

Remarks Settings panels return this value in the `panelScope` field of the `PanelFlags` structure, from their `CWPlugin_GetDropInFlags` entry point.

Settings panels which specify a scope of `panelScopeProject` appear in the **Version Control Settings** dialog box and are always enabled, regardless of whether a project is open. Currently, only version control plug-ins should use this scope.

See Also “PanelFlags Structure” in the *IDE SDK Developer’s Guide*

## panelScopeTarget

Description Indicates that a settings panel should appear in the **Edit > Target Settings** dialog.

Prototype `#include <CWDropInPanel.h>`

Remarks Settings panels return this value in the `panelScope` field of the `PanelFlags` structure, from their `CWPlugin_GetDropInFlags` entry point.

Settings panels that specify a scope of `panelScopeTarget` appear in the **Target Settings** dialog. The IDE enables these panels if at least one project is open (and that project uses plug-ins associated with the settings panel). P associated with compilers and linkers use this scope.

See Also “PanelFlags Structure” in the *IDE SDK Developer’s Guide*

## Settings Panel Plug-in API Reference

*reqActivateItem*

### reqActivateItem

|             |                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Requests that a Mac OS panel redraw a custom dialog item to show it has input focus.                                                                                                                                                                                                                                                                                     |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                            |
| Remarks     | The IDE sends this request when a custom dialog box item acquires input focus. Plug-ins should respond by highlighting the custom item.                                                                                                                                                                                                                                  |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the current settings data.</li> <li>• <code>itemHit</code> contains the number of the custom dialog item to highlight.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> |
| See Also    | <p>“Managing Input Focus” in the <i>IDE SDK Developer’s Guide</i></p> <p>“Managing Input Focus (Mac OS)” in the <i>IDE SDK Developer’s Guide</i></p>                                                                                                                                                                                                                     |

### reqAEGetPref

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Requests that a Mac OS panel return the value of a setting in response to an Apple Event.                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Remarks     | The IDE asks the panel to access an item of settings information based on an Apple Event.                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the settings data to extract the item from.</li> <li>• <code>prefsKeyword</code> contains the 4 character (32-bit) Apple Event keyword specifying the setting to return.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>prefsDesc</code> contains an Apple Event descriptor containing the data of the requested setting. The plug-in must allocate event using <code>AECreatedDesc</code> from the Mac OS toolbox.</li> </ul> |

See Also “Apple Event Dictionary ('aete') Resource” in the *IDE SDK Developer's Guide*

## reqAESetPref

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Requests that a Mac OS panel change the value of a setting in response to an Apple Event.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Remarks     | The IDE asks the panel to store an item of settings information based on an Apple Event.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> <li>• <code>prefsKeyword</code> contains the 4 character (32-bit) Apple Event keyword specifying the setting to modify.</li> <li>• <code>prefsDesc</code> contains an Apple Event descriptor containing the new data for the specified setting.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the changed settings data.</li> </ul> |
| See Also    | “Apple Event Dictionary ('aete') Resource” in the <i>IDE SDK Developer's Guide</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

## reqByteSwapData

|             |                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks the panel to adjust its settings data for endian-ness.                                                                                                                                                                                                                                                                                                                             |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                           |
| Remarks     | The IDE sends this request to adjust a panel's settings data to little-endian or big-endian format.                                                                                                                                                                                                                                                                                     |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> <li>• <code>toEndian</code> indicates the endian-ness to which the plug-in must convert its data.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the swapped settings data.</li> </ul> |

## Settings Panel Plug-in API Reference

### *reqDeactivateItem*

---

See Also    “Handling a reqByteSwapData Request in the *IDE SDK Developer’s Guide*

### **reqDeactivateItem**

|             |                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Requests that a Mac OS panel redraw a custom control that no longer has input focus.                                                                                                                                                                                                                                                                                                                |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                       |
| Remarks     | The IDE sends this request when a custom dialog item loses input focus. plug-ins should respond by un-highlighting the custom dialog item.                                                                                                                                                                                                                                                          |
| Mac OS      | On entry <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains a handle to a copy of the panel’s current settings data.</li> <li>• <code>itemHit</code> contains the dialog item to draw without the input focus.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> |
| See Also    | “Managing Input Focus” in the <i>IDE SDK Developer’s Guide</i><br>“Managing Input Focus (Mac OS)” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                                                                           |

### **reqDragDrop**

|             |                                                                                                                                                                                                                                                                                                                                                |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Informs a Mac OS panel that a dragged item has been dropped on it.                                                                                                                                                                                                                                                                             |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                  |
| Remarks     | The IDE informs the panel that the user has dropped an object on an item. A panel normally responds by extracting data from the drop using Drag Manager routines and by updating its current settings data.                                                                                                                                    |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>dragref</code> contains the Drag Manager drag reference value of the current drag operation.</li> <li>• <code>dragmouse</code> contains the mouse location, in local coordinates.</li> <li>• <code>dialog</code> contains a pointer the <b>Target Settings</b> dialog box.</li> </ul> |

- `currentPrefs` contains the settings data on which to operate.
- `itemHit` contains the dialog item onto which the user has dragged the object.
- `baseItems` contains the number of control items that the IDE has already placed in the **Target Settings** dialog box.

On exit:

- If the panel sets `itemHit` to a non-zero value, the IDE sends a `reqItemHit` request to the panel with `itemHit` containing the item to act on.

See Also    “Handling Drag and Drop” in the *IDE SDK Developer’s Guide*

              “`reqDragEnter`” on page 353

              “`reqDragExit`” on page 354

              “`reqDragWithin`” on page 355

## reqDragEnter

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Informs a Mac OS panel that a dragged item has been dragged, but not dropped, onto it.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Remarks     | The IDE informs the panel that the user has just dragged, but not dropped, an object onto one of its items. The panel should call the appropriate Drag Manager routines to handle user feedback if the item can receive the object being dragged.                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>dragref</code> contains the Drag Manager drag reference value of the current drag operation.</li> <li>• <code>dragmouse</code> contains the mouse location, in local coordinates.</li> <li>• <code>dialog</code> contains a pointer the <b>Target Settings</b> dialog box.</li> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> <li>• <code>itemHit</code> contains the dialog item onto which the user has dragged the object.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the Target Setings dialog box.</li> </ul> |

## Settings Panel Plug-in API Reference

### reqDragExit

On exit:

- `dragrect` optionally contains new dimensions that delimit the area that the item responds to subsequent drag requests.

Typically a panel plug-in decreases the values in `dragrect` to ignore scroll bars or increases the values in `dragrect` to enable autoscrolling.

The plug-in also returns an error code, which the IDE interprets as follows:

- `noErr` means the object can be dropped onto the item and the IDE can continue to send `reqDragWithin`, `reqDragExit`, and `reqDragDrop` requests for this drag operation
- `dragNotAcceptedErr` means the item supports dragging and dropping but not for objects of the kind that is currently being dragged. The IDE should not send any more drag requests for the current drag operation
- `paramErr` means the item does not support dragging and dropping. The IDE should not send any more drag requests for the current drag operation

See Also “Handling Drag and Drop” in the *IDE SDK Developer’s Guide*

“reqDragDrop” on page 352

“reqDragExit” on page 354

“reqDragWithin” on page 355

### reqDragExit

**Description** Informs a Mac OS panel that a dragged item has been dragged, but not dropped, out of it.

**Prototype** `#include <CWDropInPanel.h>`

**Remarks** The IDE informs the panel that the user has dragged an object out of an item. The panel should call appropriate Drag Manager routines to handle user feedback for the item.

**Mac OS** On entry:

- `dragref` contains the Drag Manager drag reference value of the current drag operation.

- `dragmouse` contains the mouse location, in local coordinates.
  - `dialog` contains a pointer the **Target Settings** dialog.
  - `currentPrefs` contains the settings data on which to operate.
  - `itemHit` contains the dialog item onto which the user has dragged the object.
  - `baseItems` contains the number of control items that the IDE has already placed in the **Target Settings** dialog box.
- See Also

“Handling Drag and Drop” in the *IDE SDK Developer’s Guide*

“reqDragDrop” on page 352

“reqDragEnter” on page 353

“reqDragWithin” on page 355

## reqDragWithin

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Informs a Mac OS panel that a dragged item has been dragged, but not dropped, within it.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Remarks     | The IDE informs the panel that the user has dragged, but not dropped, an object within an item. The IDE sends this request to allow the panel to track the user’s mouse movements during a drag operation and to perform any additional drop point highlighting that may be required.                                                                                                                                                                                                                                                                                                                                                                                                     |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>dragref</code> contains the Drag Manager drag reference value of the current drag operation.</li> <li>• <code>dragmouse</code> contains the mouse location, in local coordinates.</li> <li>• <code>dialog</code> contains a pointer the <b>Target Settings</b> dialog box.</li> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> <li>• <code>itemHit</code> contains the dialog item onto which the user has dragged the object.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> |
| See Also    | “Handling Drag and Drop” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

## Settings Panel Plug-in API Reference

### *reqDrawCustomItem*

---

“reqDragDrop” on page 352

“reqDragEnter” on page 353

“reqDragExit” on page 354

### **reqDrawCustomItem**

|             |                                                                                                                                                                                                                                                                       |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a Mac OS panel to draw a custom dialog item.                                                                                                                                                                                                                     |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                         |
| Remarks     | The IDE asks the panel to draw a custom dialog box item. The panel should save and restore any port characteristics it modifies.                                                                                                                                      |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>itemHit</code> contains the dialog item to draw.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> |
| See Also    | “Drawing Custom Items” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                        |

### **reqFilter**

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a Mac OS panel to respond to a low-level Mac OS event.                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Remarks     | The IDE asks the panel to process a Mac OS event. The panel can use this call to handle custom dialog controls.                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>event</code> contains a Mac OS event.</li> <li>• <code>dialog</code> contains a pointer to the <b>Target Settings</b> dialog box.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> On exit: <ul style="list-style-type: none"> <li>• <code>itemHit</code> contains the number of the item in the <b>Target Settings</b> dialog box that the user selected.</li> </ul> |
| See Also    | “Filtering Low Level Events” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

## reqFindStatus

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a Mac OS panel to determine the enabled state of an <b>Edit</b> menu item when a custom dialog item has input focus.                                                                                                                                                                                                                                                                                                                                                                                                                |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Remarks     | <p>The IDE sends this request to query the panel about <b>Edit</b> menu commands it supports for the dialog item that has the input focus. The panel returns the status of the item specified in the last <code>reqActivateItem</code> request. The IDE always sends a <code>reqActivateItem</code> request before sending a <code>reqFindStatus</code> request.</p> <p>The IDE automatically supports standard text field items. It only sends the <code>reqFindStatus</code> request for custom dialog items.</p>                      |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li><code>itemHit</code> contains the value of the Edit menu command that the dialog item does or doesn't support.</li> <li><code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>the plug-in sets <code>itemHit</code> to <code>true</code> if the item supports the <b>Edit</b> command or <code>false</code> if not.</li> </ul> |
| See Also    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

## reqFirstLoad

|             |                                                                                                                                                                                                                                                                                                                |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Informs a panel that a new target has been loaded.                                                                                                                                                                                                                                                             |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                  |
| Remarks     | The IDE sends all panels this request when a new target is loaded. Most plug-ins do not need to respond to this request.                                                                                                                                                                                       |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li><code>currentPrefs</code> contains the settings data on which to operate.</li> <li><code>toEndian</code> indicates the current endian-ness of the data.</li> <li><code>targetFile</code> specifies the location of the current target file.</li> </ul> |



# Freescale Semiconductor, Inc.

## Settings Panel Plug-in API Reference

### reqGetData

**NOTE** `toEndian` specifies the current endian-ness of the data, rather than the endian-ness to which the plug-in should convert, unlike the `reqByteSwapData` request.

See Also “Handling Edit Menu Commands” in the *IDE SDK Developer’s Guide*

### reqGetData

**Description** Asks a panel to extract data values from its panel.

**Prototype** `#include <CWDropInPanel.h>`

**Remarks** The IDE sends this request to ask the panel to copy the current settings of its controls into its settings handle. The IDE sends this request when the user saves the current settings.

**Windows** A plug-in calls `CWPanelGetCurrentPrefs` to obtain its current preferences data handle. Handles can be resized using `CWResizeMemHandle`. A plug-in extracts values from its panel UI items using calls such as `CWPanelGetItemValue` and `CWPanelGetItemText`, and stores the item values in its current preferences handle directly.

**Mac OS** On entry:

- `currentPrefs` contains the settings data on which to operate.
- `dialog` contains a pointer to the **Target Settings** dialog box.
- `baseItems` contains the number of control items that the IDE has already placed in the **Target Settings** dialog box.

On exit:

- `currentPrefs` contains the settings data retrieved from the panel’s dialog controls.

See Also “Handling a reqGetData Request” in the *IDE SDK Developer’s Guide*

### reqGetFactory

**Description** Asks a panel to return its default settings.

**Prototype** `#include <CWDropInPanel.h>`

**Remarks** The IDE asks the panel to return its default settings data.

|          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Windows  | A plug-in calls <code>CWPanelGetFactoryPrefs</code> to obtain its factory settings handle. The plug-in then modifies the handle contents to contain default values for all settings.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Mac OS   | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>factoryPrefs</code> contains the settings data on which to operate.</li> <li>• <code>dialog</code> contains a pointer to the <b>Target Settings</b> dialog box (if <code>dialog</code> is not <code>NULL</code>, the dialog box is currently being displayed).</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>factoryPrefs</code> contains the panel's default settings data</li> </ul> |
| See Also | "Handling a <code>reqGetFactory Request</code> " in the <i>IDE SDK Developer's Guide</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

## reqHandleClick

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a Mac OS panel to act on a mouse click on a custom dialog item.                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Remarks     | The IDE sends this request when a panel's custom dialog item receives a mouse click. The IDE only sends this request if the item is active.                                                                                                                                                                                                                                                                                                                                                                                  |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>event</code> contains the Mac OS event record specifying the mouse click.</li> <li>• <code>currentPrefs</code> contains a handle to a copy of the panel's current settings data.</li> <li>• <code>itemHit</code> contains the dialog item that is the target of the mouse click.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> <p>On exit:</p> |

## Settings Panel Plug-in API Reference

### *reqHandleKey*

---

- If the panel sets `itemHit` to a non-zero value, the IDE sends a `reqItemHit` request to the panel with `itemHit` containing the item on which to act.

See Also “Handling User Interaction” in the *IDE SDK Developer’s Guide*

### **reqHandleKey**

**Description** Asks a Mac OS panel to act on a key press directed at a custom dialog item.

**Prototype** `#include <CWDropInPanel.h>`

**Remarks** The IDE sends this request when a panel’s custom dialog item receives a keyboard event. The IDE only sends this request if the item is active.

**Mac OS** On entry:

- `event` contains the Mac OS event record specifying the keyboard event.
- `currentPrefs` contains a handle to a copy of the panel’s current settings data.
- `itemHit` contains the dialog item that is the target of the keyboard event.
- `baseItems` contains the number of control items that the IDE has already placed in the **Target Settings** dialog box.

On exit:

- If the panel sets `itemHit` to a non-zero value, the CodeWarrior IDE will send a `reqItemHit` request to the panel with `itemHit` containing the item on which to act.

See Also “Handling User Interaction” in the *IDE SDK Developer’s Guide*

### **reqInitDialog**

**Description** Asks a panel to prepare to be displayed.

**Prototype** `#include <CWDropInPanel.h>`

**Remarks** The IDE sends this request just before a panel is displayed when the user selects the panel in the **Target Settings** dialog. In response to this request, a panel should construct its user interface.



|          |                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Windows  | Panels which include a list box control must construct the contents of the item's list using <code>CWPanelInsertListItem</code> during the <code>reqInitDialog</code> request. The IDE does not support constructing list boxes from resource descriptions. Panels can also set item values with <code>CWPanelSetItemValue</code> and enable and disable items with <code>CWPanelEnableItem</code> . |
| Mac OS   | On entry: <ul style="list-style-type: none"><li>• <code>dialog</code> contains a pointer to the <b>Target Settings</b> dialog box.</li><li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li></ul>                                                                                                         |
| See Also | "Handling a <code>reqInitDialog</code> Request" in the <i>IDE SDK Developer's Guide</i>                                                                                                                                                                                                                                                                                                              |

reqInitPanel

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a panel plug-in to initialize its state.                                                                                                                                                                                                                                                                                                                                                                                          |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                          |
| Remarks     | The IDE sends this request when a plug-in is first loaded into memory. In response, plug-ins should initialize any internal state. This request is a deprecated synonym for the <code>reqInitialize</code> request.                                                                                                                                                                                                                    |
| Mac OS      | On entry: <ul style="list-style-type: none"><li>• <code>version</code> contains the runtime version of the panel plug-in API that the IDE supports.</li><li>• <code>storage</code> contains the value of a pointer that the IDE saves between requests to the panel until the IDE sends a <code>reqTermPanel</code> request.</li><li>• <code>targetfile</code> contains the Mac OS file specification of the active project.</li></ul> |
| See Also    | "Handling a <code>reqInitialize (reqInitPanel)</code> Request" in the <i>IDE SDK Developer's Guide</i>                                                                                                                                                                                                                                                                                                                                 |

reqItemHit

|             |                                                           |
|-------------|-----------------------------------------------------------|
| Description | Informs the panel of user interaction with a dialog item. |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>             |

## Settings Panel Plug-in API Reference

### reqItemHit

|          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remarks  | <p>The IDE sends this request immediately after user interaction with a control. In response, plug-ins normally get, set, enable, disable, and show or hide other items.</p> <p>Before returning to the IDE and after accounting for any changes in its settings data resulting from the user interaction, a plug-in should indicate whether the current panel settings match the most recently saved settings, and the factory default settings. The IDE uses this information to enable and disable the <b>Revert Panel</b> and <b>Factory Settings</b> buttons.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Windows  | <p>Windows panels set the state of the factory flag using <code>CWPanelSetFactoryFlag</code> and the state of the revert flag using <code>CWPanelSetRevertFlag</code>. Plug-ins obtain the current settings handle with <code>CWPanelGetCurrentPrefs</code>, the original (most recently saved) settings with <code>CWPanelGetOriginalPrefs</code>, and the factory default settings with <code>CWPanelGetFactoryPrefs</code>.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Mac OS   | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>dialog</code> contains a pointer to the <b>Target Settings</b> dialog box.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> <li>• <code>itemHit</code> contains the number of the item in the panel that the user selected.</li> <li>• <code>originalPrefs</code> contains the settings at the time the <b>Target Settings</b> dialog first appeared or the settings panel was first displayed.</li> <li>• <code>factoryPrefs</code> contains the settings the panel returned to the IDE for the <code>reqGetFactory</code>.</li> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the changed settings data.</li> <li>• <code>canRevert</code> is true if changes in the settings data requires the IDE to activate the <b>Target Settings</b> dialog box Revert button.</li> <li>• <code>canFactory</code> is true if changes in the settings data requires the IDE to activate the <b>Target Settings</b> dialog box Factory button.</li> </ul> |
| See Also | <p>“Handling a reqItemHit Request” in the <i>IDE SDK Developer’s Guide</i></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

## reqObeyCommand

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a Mac OS panel to act on an <b>Edit</b> menu command.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Remarks     | <p>The IDE sends this request when the user applies an <b>Edit</b> menu command to a custom panel item having input focus. The request applies to the item specified in the last reqActivateItem request. The IDE always sends reqActivateItem and reqFindStatus requests before sending a reqObeyCommand request.</p> <p>The IDE automatically supports <b>Edit</b> menu commands for standard text field items. It only sends the reqFindStatus request for custom dialog items.</p>                                                                                                                                                                                             |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>itemHit</code> contains the value of the command to perform. <code>itemHit</code> only specifies one of the commands that the panel supports, based on previous reqFindStatus requests.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• if the panel sets <code>itemHit</code> to a non-zero value, the CodeWarrior IDE will send a reqItemHit request to the panel with <code>itemHit</code> containing the item on which to act.</li> </ul> |
| See Also    | <p>“Handling Edit Menu Commands” in the <i>IDE SDK Developer’s Guide</i></p> <p>“Managing Input Focus” in the <i>IDE SDK Developer’s Guide</i></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## reqPutData

|             |                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a panel to install its settings in its dialog items.                                                                                                                                                                                                                                                                                                                       |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                   |
| Remarks     | <p>The IDE sends this request to indicate that the panel should install values from its settings handle in its panel controls. The IDE sends this request prior to displaying the panel’s user interface items.</p> <p>Panels install values in controls with <code>CWPanelSetItemValue</code>, <code>CWPanelSetItemText</code>, and <code>CWPanelSetItemTextHandle</code>.</p> |

## Settings Panel Plug-in API Reference

### reqReadSettings

Panels enable and disable items with `CWPanelEnableItem`. Panels hide and show items with `CWPanelShowItem`.

|          |                                                                                                                                                                                                                                                                                                                                                                                         |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Windows  | A panel obtains its current settings handle with <code>CWPanelGetCurrentPrefs</code> .                                                                                                                                                                                                                                                                                                  |
| Mac OS   | On entry: <ul style="list-style-type: none"> <li>• <code>dialog</code> contains a pointer to the <b>Target Settings</b> dialog box.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> <li>• <code>currentPrefs</code> contains the on which to settings data to operate.</li> </ul> |
| See Also | “Handling a <code>reqPutData</code> Request” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                                                                                                                    |

### reqReadSettings

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Asks a panel to import XML settings data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Remarks     | <p>The IDE sends the <code>reqReadSettings</code> request when importing a project from XML. A plug-in should respond by reading all expected name-value pairs from the XML input stream with calls such as <code>CWReadIntegerSetting</code>, <code>CWGetIntegerValue</code>, <code>CWGetArraySettingElement</code>, <code>CWGetStructureSettingField</code>, and <code>CWGetNamedSetting</code>.</p> <p>A panel returns the newly read settings to the IDE by storing them in its current settings handle. A plug-in can resize the handle by calling <code>CWResizeMemHandle</code>.</p> |
| Windows     | Windows plug-ins read XML settings into the current settings handle, obtained by calling <code>CWPanelGetCurrentPrefs</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| See Also    | “Handling a <code>reqReadSettings</code> Request” in the <i>IDE SDK Developer’s Guide</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

## reqRenameProject

|             |                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | <p>Informs the panel that the project name has changed.</p>                                                                                                                                                                                                                                                                                                                          |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt;</pre>                                                                                                                                                                                                                                                                                                                                          |
| Remarks     | <p>The IDE asks the panel to change its settings information based on a new project name. The IDE sends this request when creating a new project based on a project stationery document. The IDE does <i>not</i> send this request when the user renames a project using Windows Explorer or the Mac OS Finder.</p>                                                                  |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>oldtargfile</code> contains the Mac OS file specification for the old target file.</li> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the changed settings data.</li> </ul> |
| See Also    | <p>“Handling a <code>reqRenameProject</code> Request” in the <i>IDE SDK Developer’s Guide</i></p>                                                                                                                                                                                                                                                                                    |

## reqSetupDebug

|             |                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | <p>Informs the panel that debugging has been enabled or disabled.</p>                                                                                                                                                                                                                                                                                                                                                     |
| Prototype   | <pre>#include &lt;CWDropInPanel.h&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                               |
| Remarks     | <p>The IDE asks the panel to change its settings information to support debugging of the current project. If the state of debugging does not affect a panel, this request can be ignored.</p>                                                                                                                                                                                                                             |
| Windows     | <p>Windows plug-ins can determine whether debugging is enabled by calling <code>CWPanelGetDebugFlag</code>.</p>                                                                                                                                                                                                                                                                                                           |
| Mac OS      | <p>On entry:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the settings data on which to operate.</li> <li>• <code>debugOn</code> is <code>true</code> if debugging the project is enabled or <code>false</code> if debugging is disabled.</li> </ul> <p>On exit:</p> <ul style="list-style-type: none"> <li>• <code>currentPrefs</code> contains the changed settings data.</li> </ul> |

## Settings Panel Plug-in API Reference

### *reqTermDialog*

---

See Also “Handling a reqSetupDebug Request” in the *IDE SDK Developer’s Guide*

“CWPanelGetDebugFlag” on page 276

### reqTermDialog

|             |                                                                                                                                                                                                                                                                                                 |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Notifies a panel that its user interface is about to be closed.                                                                                                                                                                                                                                 |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                   |
| Remarks     | The IDE asks the panel to prepare to be removed from the <b>Target Settings</b> dialog box. The IDE sends this request when the user changes panels or when the user closes the <b>Target Settings</b> dialog box.                                                                              |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>dialog</code> contains a pointer to the <b>Target Settings</b> dialog box.</li> <li>• <code>baseItems</code> contains the number of control items that the IDE has already placed in the <b>Target Settings</b> dialog box.</li> </ul> |
| See Also    | “Handling a reqTermDialog Request in the <i>IDE SDK Developer’s Guide</i> ”                                                                                                                                                                                                                     |

### reqTermPanel

|             |                                                                                                                                                                                                                                                                                                                                          |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Notifies the panel that it is about to be unloaded from memory.                                                                                                                                                                                                                                                                          |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                            |
| Remarks     | The IDE issues this request prior to unloading a panel from memory. This constant is a deprecated synonym for the <code>reqTerminate</code> request.<br><br>Plug-ins should clean up internal state and free any resources they acquired when responding to the <code>reqInitPanel</code> request.                                       |
| Mac OS      | On entry: <ul style="list-style-type: none"> <li>• <code>version</code> contains the runtime version of the panel plug-in API that the IDE supports.</li> <li>• <code>storage</code> contains the value of a pointer that the IDE saved between requests to the panel since the IDE sent a <code>reqInitPanel</code> request.</li> </ul> |

- `targetfile` contains the Mac OS file specification of the active project.
- See Also “Handling a `reqTerminate` Request” in the *IDE SDK Developer’s Guide*

### reqUpdatePref

Description Asks a panel to update the version of its settings data.

Prototype `#include <CWDropInPanel.h>`

Remarks The IDE asks the panel to update an older version of its settings data to the latest version. plug-ins modify the contents of the current settings handle to return the updated settings to the IDE.

**NOTE** Plug-ins must always store the data version in the first 16-bit integer of the settings data handle.

Typically, plug-ins construct a new settings handle containing default values for current settings and then copy and convert any values still supported from the old settings data. The current default settings can be constructed using the same code that responds to the `reqGetFactory` request. The current settings handle may need to be expanded, by calling `CWResizeMemHandle`.

Windows Windows plug-ins obtain the current settings data handle by calling `CWPanelGetCurrentPrefs`. Plug-ins return updated settings by modifying the handle’s contents directly.

Mac OS On entry:

- `currentPrefs` contains the settings data on which to operate.

On exit:

- `currentPrefs` contains the changed settings data.

See Also “Handling a `reqUpdatePref` Request” in the *IDE SDK Developer’s Guide*

### reqValidate

Description Asks a plug-in to determine any actions required by the IDE in response to changes in the plug-in’s settings.

## Settings Panel Plug-in API Reference

### *reqWriteSettings*

---

|           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Prototype | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Remarks   | The IDE sends this request to determine whether changes in the panel's settings (the differences between the original state and the current state) require the project to reset its file paths, to be recompiled, or to be relinked.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Windows   | Windows plug-ins obtain the current settings by calling <code>CWPanelGetCurrentPrefs</code> and the most recently saved settings by calling <code>CWPanelGetOriginalPrefs</code> .<br><br>To indicate whether project paths should be reset, a plug-in calls <code>CWPanelSetResetPathsFlag</code> . To indicate whether the project should be recompiled, a plug-in calls <code>CWPanelSetRecompileFlag</code> . To indicate whether the project should be relinked, a plug-in calls <code>CWPanelSetRelinkFlag</code> . To indicate whether the browse information for the current project should be reparsed, a plug-in calls <code>CWPanelSetReparseFlag</code> . |
| Mac OS    | On entry: <ul style="list-style-type: none"> <li>• <code>originalPrefs</code> contains the most recently saved settings data.</li> <li>• <code>currentPrefs</code> contains the current settings data.</li> </ul> On exit: <ul style="list-style-type: none"> <li>• <code>reset</code> is true if the project's file paths should be reset.</li> <li>• <code>recompile</code> is true if the project should be recompiled.</li> <li>• <code>relink</code> is true if the project should be relinked.</li> <li>• <code>reparse</code> is true if the project's files should be reprocessed by the class browser. The IDE does not use this flag.</li> </ul>            |
| See Also  | "Handling a <code>reqValidate</code> Request" in the <i>IDE SDK Developer's Guide</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

### **reqWriteSettings**

|             |                                                                                                                |
|-------------|----------------------------------------------------------------------------------------------------------------|
| Description | Asks a plug-in to export its settings data as XML.                                                             |
| Prototype   | <code>#include &lt;CWDropInPanel.h&gt;</code>                                                                  |
| Remarks     | The IDE sends this request to tell a plug-in to export its settings as XML. In response, a plug-in should call |

The IDE sends the `reqWriteSettings` request when exporting a project to XML. A plug-in should respond by writing its settings data as name-value pairs to the XML output stream, by using calls such as `CWWriteIntegerSetting`, `CWSetIntegerValue`, `CWGetArraySettingElement`, `CWGetStructureSettingField`, and `CWGetNamedSetting`.

- Windows Windows panels obtain their current settings handle to write as XML by calling `CWPanelGetCurrentPrefs`.
- Mac OS On entry:
  - `currentPrefs` contains the current settings data.
- See Also “Handling a `reqWriteSettings` Request” in the *IDE SDK Developer’s Guide*

## supportsByteSwapping

- Description A panel flag indicating that a plug-in knows how to swap the byte ordering of its settings data.
- Prototype `#include <CWDropInPanel.h>`
- Remarks A panel sets this flag in its `PanelFlags` to indicate that it supports the `reqByteSwapData` request. Panels should support this request if at all possible.
- See Also “Handling a `reqByteSwapData` Request” in the *IDE SDK Developer’s Guide*  
 “PanelFlags Structure” in the *IDE SDK Developer’s Guide*  
 “PanelFlags” on page 335  
 “reqByteSwapData” on page 351

## supportsTextSettings

- Description A panel flag indicating that a plug-in supports XML import and export of its settings data.
- Prototype `#include <CWDropInPanel.h>`

## Settings Panel Plug-in API Reference

### *usesStrictAPI*

**Remarks** A panel sets this flag in its `PanelFlags` to indicate that it supports the `reqReadSettings` and `reqWriteSettings` requests, used when importing and exporting XML data, respectively.

In response to these requests, a panel should read and write a series of XML settings corresponding to its settings data. Panels should support these requests if at all possible.

**See Also** “Importing and Exporting Settings Data” in the *IDE SDK Developer’s Guide*

“PanelFlags” on page 335

### **usesStrictAPI**

**Description** A panel flag indicating that a Mac OS plug-in uses exclusively the CodeWarrior Plug-in API to manipulate user interface controls.

**Prototype** `#include <CWDropInPanel.h>`

**Remarks** Panel plug-ins set this bit in the `dropinflags` field of their `DropinFlags` structure to indicate that they utilize only plug-in API routines for manipulating user interface controls, rather than Mac OS dialog manager routines. This setting allows the IDE to construct a panel’s user interface with more flexibility, without necessarily creating or exposing a Mac OS dialog manager window.

All new Mac OS panel plug-ins should set this flag, and should also enable `CW_STRICT_DIALOGS` at compile time, to help enforce dialog access using the API only.

**See Also** “DropinFlags Structure” in the *IDE SDK Developer’s Guide*

“CW\_STRICT\_DIALOGS” on page 345

“PanelFlags” on page 335

## Settings Panel Result Codes

The IDE supports returning additional error codes from panel plug-ins. This section documents the panel-specific error codes.

The API defines the following settings panel error codes:

- kBadPrefVersion
- kMissingPrefErr
- kSettingNotFoundErr
- kSettingTypeMismatchErr
- kInvalidCallbackErr
- kSettingOutOfRangeErr

### kBadPrefVersion

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Indicates that a panel cannot convert its preference data to the requested version.                                                                                                                                                                                                                                                                                                                                                          |
| Definition  | <pre>#include &lt;CWDropInPanel.h&gt;  /* Windows */ #include &lt;CWDropInPanel.h&gt;  /* Mac OS */</pre>                                                                                                                                                                                                                                                                                                                                    |
| Remarks     | A plug-in returns this error code to the IDE when they receive a request to convert a version of the settings data that they do not understand to the current version. This situation arises, for example, when a project containing a newer version of the plug-in data is opened by an IDE running an older version of a plug-in. It also happens when the IDE asks a plug-in to an old version of its data that it may no longer support. |

### kMissingPrefErr

|             |                                                                                                           |
|-------------|-----------------------------------------------------------------------------------------------------------|
| Description | Unused.                                                                                                   |
| Definition  | <pre>#include &lt;CWDropInPanel.h&gt;  /* Windows */ #include &lt;CWDropInPanel.h&gt;  /* Mac OS */</pre> |

### kSettingNotFoundErr

|             |                                                                                                                                                                                        |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Indicates that an XML setting by the specified name could not be found.                                                                                                                |
| Definition  | <pre>#include &lt;CWDropInPanel.h&gt;  /* Windows */ #include &lt;CWDropInPanel.h&gt;  /* Mac OS */</pre>                                                                              |
| Remarks     | The IDE returns this error code when a plug-in calls either CWGetNamedSetting or one of the CWRead calls, to obtain a setting which does not appear in the plug-in's input XML stream. |



## Freescal Semiconductor, Inc.

### Settings Panel Plug-in API Reference

*kSettingTypeMismatchErr*

---

#### **kSettingTypeMismatchErr**

|             |                                                                                                                                                                                                                   |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Indicates that the requested setting was not of the expected type.                                                                                                                                                |
| Definition  | <pre>#include &lt;CWDropInPanel.h&gt;    /* Windows */ #include &lt;CWDropInPanel.h&gt;    /* Mac OS  */</pre>                                                                                                    |
| Remarks     | The IDE returns this error code when a panel plug-in attempts to read an XML settings, and a setting by the requested name is available, but the type of the setting does not match that implied by the API call. |

For example, the IDE returns this result if a plug-in attempts to read a string settings using `CWGetIntegerValue`.

#### **kInvalidCallbackErr**

|             |                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Indicates that the called routine is not available during the current request.                                                                                                                                                                                                                                                                                                                      |
| Definition  | <pre>#include &lt;CWDropInPanel.h&gt;    /* Windows */ #include &lt;CWDropInPanel.h&gt;    /* Mac OS  */</pre>                                                                                                                                                                                                                                                                                      |
| Remarks     | The IDE returns this error code when a panel plug-in calls an API routine that is not supported for the current request. For example, the IDE returns this value when plug-ins call one of the <code>CWRead</code> or <code>CWWrite</code> routines, or one of the other XML import or export routines, during a request other than <code>reqReadSettings</code> or <code>reqWriteSettings</code> . |

#### **kSettingOutOfRangeErr**

|             |                                                                                                                |
|-------------|----------------------------------------------------------------------------------------------------------------|
| Description | The IDE uses this error code only internally.                                                                  |
| Definition  | <pre>#include &lt;CWDropInPanel.h&gt;    /* Windows */ #include &lt;CWDropInPanel.h&gt;    /* Mac OS  */</pre> |
| Remarks     | The does not currently return this error code to panel plug-ins. This may change in the future.                |

# Version Control System Plug-in API

---

The CodeWarrior IDE supports version control system (VCS) plug-ins. This chapter describes the CodeWarrior IDE Plug-in API for implementing VCS plug-ins.

This chapter covers the following topics:

- Plug-In Callbacks
- VCS Requests
- VCS API Version 1 Compatibility

## Plug-In Callbacks

Callbacks let a plug-in get information from the IDE, send communicate information back to the IDE, ask the IDE to perform actions, and inquire into file states. The VCS API exposes both standard callbacks and callbacks specific to VCS plug-ins.

Version 1 of the VCS API does not support all of these callbacks. See the later section on V1 compatibility mode for which callbacks are available to be used. A callback returns an error of type `CWResult` under the following circumstances:

- A plug-in tries to use a callback in an inappropriate mode.
- A plug-in passes in an invalid parameter.
- A plug-in fails.

Check the return error values of the callbacks in your code to ensure proper operation.



## Freescale Semiconductor, Inc.

### Version Control System Plug-in API

#### Standard Callbacks

---

The IDE specifies all files as `CWFileSpec` objects. On the Mac OS, the IDE uses a standard `FSSpec`. On Windows, the IDE uses a structure with one member variable, `path`, which contains the absolute path to the file or directory.

### Standard Callbacks

The following standard callbacks are accessible to VCS plug-ins:

| Callback                             | 1 | 2 | 3 |
|--------------------------------------|---|---|---|
| <code>CWGetPluginRequest</code>      | X |   | X |
| <code>CWDonePluginRequest</code>     | X |   | X |
| <code>CWGetAPIVersion</code>         | X |   | X |
| <code>CWGetIDEInfo</code>            | X |   |   |
| <code>CWGetCallbackOSError</code>    | X |   |   |
| <code>CWSetPluginOSError</code>      |   |   |   |
| <code>CWGetFileText</code>           |   |   |   |
| <code>CWReleaseFileText</code>       |   |   |   |
| <code>CWReportMessage</code>         |   |   | X |
| <code>CWAlert</code>                 |   |   | X |
| <code>CWShowStatus</code>            |   |   | X |
| <code>CWUserBreak</code>             |   |   | X |
| <code>CWGetNamedPreferences</code>   |   |   | X |
| <code>CWStorePluginData</code>       |   |   |   |
| <code>CWGetPluginData</code>         |   |   |   |
| <code>CWCreateNewTextDocument</code> |   |   | X |
| <code>CWAllocateMemory</code>        | X |   |   |
| <code>CWFreeMemory</code>            | X |   |   |
| <code>CWAllocMemHandle</code>        | X |   | X |
| <code>CWFreeMemHandle</code>         | X |   | X |



Freescale Semiconductor, Inc.

Version Control System Plug-in API  
Standard Callbacks

| Callback           | 1 | 2 | 3 |
|--------------------|---|---|---|
| CWGetMemHandleSize | X |   | X |
| CWResizeMemHandle  | X |   | X |
| CWLockMemHandle    | X |   | X |
| CWUnlockMemHandle  | X |   | X |
| CWPreDialog        | X | X | X |
| CWPostDialog       | X | X | X |
| CWPreFileAction    |   | X |   |
| CWPostFileAction   |   | X |   |

1: callback can be used during reqInititalize and reqTerminate under Pro 4 IDE.

2: callback is new with plug-in API Version 9 (Pro 4 IDE)

3: callback works with version 1 plug-ins

The following table describes callbacks that have somewhat different behaviors for VCS plug-ins:

| Callback          | How it differs                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CWGetPluginData   | These two functions can be used by a VCS plug-in to store non-volatile information associated with a project. Each project has its unique data storage area. These functions have behavioral differences for VCS plug-ins in that the data is stored projectwide and not on a pertarget basis. The filenumber passed in as an argument should always be -1. Any tag that is not all lowercase letters can be used. |
| CWStorePluginData | as CWGetPluginData                                                                                                                                                                                                                                                                                                                                                                                                 |
| CWReportMessage   | This function will display a message inside of the VCS messages window instead of the errors & warnings window. Any file and line information that is passed in with the message is ignored for VCS plug-ins.                                                                                                                                                                                                      |

# Freescale Semiconductor, Inc.

## Version Control System Plug-in API

### Standard Callbacks

| Callback         | How it differs                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CWPreDialog      | This function takes the <code>context</code> that is passed in to the main entry point as its only argument. It tells the IDE to prepare for the plug-in to display a dialog. By calling this function, the IDE can disable any windows and make preparations for the Dialog Manager to be called so it can function properly.                                                                                                                                                                                                                                                        |
| CWPostDialog     | This function takes as its argument the <code>context</code> passed into the main entry point. It is used by the plug-in to notify the IDE that the plug-in's dialog is finished displaying and has been destroyed so the IDE can reset the windows for the user.                                                                                                                                                                                                                                                                                                                     |
| CWPreFileAction  | <p>This callback takes as its arguments the <code>context</code> passed to the main entry point along with a <code>CWFileSpec</code> that contains the appropriate file system specification for a file.</p> <p>The plug-in can call this function before it attempts to do any type of processing of a file. By calling <code>CWPreFileAction</code>, the IDE will make sure that it can do everything possible to insure accessibility of the file by, for example, closing down any access paths the IDE has to the file so the plug-in can open it for exclusive read/writes.</p> |
| CWPostFileAction | Plug-ins that process files should use this callback after all processing of a file is done and all of the plug-in's access paths to the file have been closed. This then allows the IDE to reestablish any access it needs to the file in order to function. The callback takes the <code>context</code> that is passed to a plug-in's main entry point along with a <code>CWFileSpec</code> indicating which file has just finished being processed by the plug-in.                                                                                                                 |



VCS Callbacks

The VCS extension of the base plug-in API features the following VCS specific callbacks. Your plug-in can use these callbacks to initiate actions, retrieve information from the IDE, and send information back to the IDE. The following table shows the VCS callbacks:

| Callback                    | 1 | 2 |
|-----------------------------|---|---|
| CWAllowV1Compatibility      |   |   |
| CWGetComment                |   |   |
| CWFileStateChanged          |   |   |
| CWVCSStateChanged           |   |   |
| CWGetProjectFileSpecifier   |   |   |
| CWIsAdvancedRequest         |   |   |
| CWIsRecursiveRequest        |   |   |
| CWIsCommandSupportedRequest |   |   |
| CWGetDatabaseConnectionInfo |   |   |
| CWGetCommandStatus          |   |   |
| CWSetCommandStatus          |   |   |
| CWGetVCSItemCount           |   |   |
| CWGetVCSItem                |   |   |
| CWSetVCSItem                |   |   |
| CWGetVCSPointerStorage      |   |   |
| CWSetVCSPointerStorage      |   |   |
| CWDoVisualDifference        | X |   |
| CWCompletionRatio           |   | X |
| CWSetCommandDescription     |   | X |

1: not supported in all older IDEs (Pro2 and earlier)



# Freescal Semiconductor, Inc.

## Version Control System Plug-in API

### *Standard Callbacks*

---

2: not supported in version 1

As with other callbacks, the first argument to every callback routine contains the `context` object sent to the plug-in's main entry point when it is called by the IDE. If a plug-in calls a callback and while running in a mode that does not allow the callback, the IDE returns `cwErrInvalidCallback`.



CWAllowV1Compatibility

Description CWAllowV1Compatibility lets plug-ins tell the IDE whether they can run as version 1 plug-ins. It also lets plug-ins gracefully exit if an older version of the IDE calls them.

Prototype CW\_CALLBACK CWAllowV1Compatibility(  
CWPluginContext context,  
Boolean canHandleV1Mode,  
Boolean \*isV1);

Parameters The following table shows the parameters for this method:

|                 |                                                                                                                                                                                                                                                                                                                                                                         |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| context         | Private, opaque IDE state.                                                                                                                                                                                                                                                                                                                                              |
| canHandleV1Mode | When set to true, lets the plug-in run as a version 1 plug-in if the IDE does not recognize version 7+ plug-ins. In other words, this setting lets a plug-in run on an older IDE. When canHandleV1Mode is set to false, the callback returns an error if the IDE cannot run the plug-in.                                                                                |
| isV1            | Specifies whether the plug-in runs as a version 1 plug-in. true indicates that the plug-in runs as a version 1 argument. In that case, the IDE does not send certain requests, and the plug-in cannot use certain callbacks. false indicates that the plug-in runs as a version 7+ plug-in, can handle the full set of requests, and can use the full set of callbacks. |

Remarks Version 7+ plug-ins must call this callback before otherwise communicating with the IDE. It ensures functionality (either through compatibility emulation by the IDE or by failing to load) in IDE versions earlier than Pro 4.

NOTE If canHandleV1Mode returns an error, the plug-in should immediately return the error to the IDE, so that the IDE can unload the plug-in. Failure to do so can crash the IDE or the system.

## Version Control System Plug-in API

### Standard Callbacks

#### CWGetComment

**Description** CWGetComment displays a window that asks the user for a comment to go with the file in the VCS database.

**Prototype**

```
CW_CALLBACK CWGetComment (
 CWPluginContext context,
 const char *pPrompt,
 const char *pComment,
 const long lBufferSize);
```

**Parameters** The following table shows the parameters for this method:

|             |                                                                                            |
|-------------|--------------------------------------------------------------------------------------------|
| context     | Private, opaque IDE state.                                                                 |
| pPrompt     | A pointer to a string that appears as the prompt                                           |
| pComment    | On return, this parameter contains a pointer to a string that contains the user's comment. |
| lBufferSize | A long integer that indicates the maximum length of the buffer in bytes                    |

**Remarks** The plug-in can call this method to display a comment dialog box. This method lets the plug-in get a line of comment text from the user.

The function returns `cwNoErr` if the operation succeeded. If the user cancels the operation, the function returns `cwErrUserCanceled`.

#### CWFileStateChanged

**Description** CWFileStateChanged tells the IDE that the state of a file changed. This method has been superceded by `CWVCSStateChanged`.

**Prototype**

```
CW_CALLBACK CWFileStateChanged (
 CWPluginContext context,
 const CWFileSpec *file,
 CWVCSCheckoutState eCheckoutState,
 const CWVCSVersion version);
```

Parameters The following table shows the parameters for this method:

|                |                                          |
|----------------|------------------------------------------|
| context        | Private, opaque IDE state.               |
| file           | A pointer to a file specification.       |
| eCheckoutState | The new state of the file in question.   |
| version        | The new version of the file in question. |

Remarks This is a deprecated version of `CWVCSStateChanged`. The two routines are identical except that `CWVCSStateChanged` passes the version parameter by reference rather than by value.

**NOTE** The version 8+ API headers no longer declare this method. However, the plug-in API still supports it for backward compatibility. In order to use this routine, plug-ins must define `CWFileStateChanged` as extern.

## CWVCSStateChanged

Description `CWVCSStateChanged` tells the IDE when the state of a file in the VCS database has changed.

Prototype `CW_CALLBACK CWVCSStateChanged(
 CWPluginContext context,
 const CWFileSpec* file,
 CWVCSCheckoutState eCheckoutState,
 const CWVCSVersion* version);`

Parameters The following table shows the parameters for this method:

|                |                                          |
|----------------|------------------------------------------|
| context        | Private, opaque IDE state.               |
| file           | A pointer to a file specification.       |
| eCheckoutState | The new state of the file in question.   |
| version        | The new version of the file in question. |

Remarks The plug-in must use this callback whenever it changes a file on the disk. Possible changes includes changing the file's checkout state due to changes in the VCS database and changing local file's information (such as modification date or locked state). Unless the plug-in uses this callback, the IDE cannot know that the plug-in has changed the file in any way.

## Version Control System Plug-in API

### Standard Callbacks

The `eCheckoutState` argument must be one of the constants from the following enumeration:

```
enum // checkout state
{
 cwCheckoutStateUnknown
 // unknown
 cwCheckoutStateNotCheckedOut
 // not checked out
 cwCheckoutStateCheckedOut
 // checked out
 cwCheckoutStateNotInDatabase
 // not in database
 cwCheckoutStateMultiplyCheckedOut
 // multiply checked out
 cwCheckoutStateNotCheckedOutShared
 //not checked out and shared
 cwCheckoutStateCheckedOutShared
 // checked out and shared
 cwCheckoutStateMultiplyCheckedOutShared
 // multiply checked out and shared
 cwCheckoutStateNotCheckedOutBranched
 // not checked out and branched
 cwCheckoutStateCheckedOutBranched
 //checked out and branched
 cwCheckoutStateMultiplyCheckedOutBranched
 // checked out and branched
 cwCheckoutStateNotCheckedOutSharedBranched
 // not checked out, shared and branched
 cwCheckoutStateCheckedOutSharedBranched
 // checked out, shared and branched
 cwCheckoutStateMultipleCheckedSharedOutBranched
 // checked out, shared and branched
 cwCheckoutStateCheckedOutExclusive
 // exclusively checked out
 cwCheckoutStateCheckedOutExclusiveShared
 // exclusively checked out and shared
 cwCheckoutStateCheckedOutExclusiveBranched
 // exclusively checked out and branched
 cwCheckoutStateCheckedOutExclusiveSharedBranch
 // exclusively checked out,
 // shared and branched
 cwCheckoutStateMultiplyCheckedOutMask
}
```

```

 // multiply checked out mask
 cwCheckoutStateSharedMask
 // shared mask
 cwCheckoutStateBranchedMask
 // branched mask
 cwCheckoutStateExclusiveMask
 // exclusive mask
};

```

The version argument must be a value from the CWVCSVersion structure:

```

typedef struct CWVCSVersion // version
{
 CWVCSVersionForm eVersionForm;
 // version form
 CWVCSVersionData sVersionData;
 // version data
}

```

The version form value must be one of the values from the following enumeration:

```

enum // version form
{
 cwVersionFormNone // no record
 cwVersionFormNumeric // integral numeric
 cwVersionFormAlpha // alphabetic
 cwVersionFormDate // date / time
 cwVersionFormLabel // label
};

```

The version form value describes which member of the CWVCSVersionData union is occupied.

```

typedef union CWVCSVersionData // version data
{
 unsigned long numeric; // integral numeric
 char *pAlpha; // alphabetic
 CWVCSDateTime date; // date / time
 char *pLabel; // label
}

```

# Freescale Semiconductor, Inc.

## Version Control System Plug-in API Standard Callbacks

The following table describes the possible version form values:

| Value        | Description                                                                                                                                                                      |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| None         | No version information for this item is provided.                                                                                                                                |
| Numeric      | In the numeric case the version is an integer, specifically an unsigned long.                                                                                                    |
| Alphanumeric | In this case the version is represented by a character string. An example of this would be "1.3.6a2".                                                                            |
| Date         | In the date case the standard library struct tm format is used. Any function from the standard library in <time.h> can be used to work with these times and do time differences. |
| Label        | The label case is essentially the same as the alphanumeric case, but is included for clarity.                                                                                    |

### CWDoVisualDifference

Description CWDoVisualDifference tells the IDE to display a window showing the contents of two files.

Prototype

```

CW_CALLBACK CWDoVisualDifference(
 CWPluginContext context,
 const CWFileSpec *file1,
 const char *pTitle1,
 const char *pText1,
 unsigned long lengthText1,
 const CWFileSpec *file2,
 const char *pTitle2,
 const char *pText2,
 unsigned long lengthText2);

```

Parameters The following table shows the parameters for this method:

|         |                                                                                                            |
|---------|------------------------------------------------------------------------------------------------------------|
| context | Private, opaque IDE state.                                                                                 |
| file1   | If the file is on the local disk, , a pointer to CWFileSpec object that defines the file. Otherwise, NULL. |
| pTitle1 | If the file is on the local disk, NULL. Otherwise, a pointer to the name of the file.                      |

|             |                                                                                                   |
|-------------|---------------------------------------------------------------------------------------------------|
| pText1      | If the file is on the local disk, NULL. Otherwise, a pointer to the text of the file.             |
| lengthText1 | If the file is on the local disk, 0. Otherwise, the length of the text to be displayed, in bytes. |
| file2       | as file1                                                                                          |
| pTitle2     | as pTitle1                                                                                        |
| pText2      | as pText1                                                                                         |
| lengthText2 | as lengthText1                                                                                    |

Remarks If the user doesn't cancel the operation, this method modifies the second file. If one file is local, you should specify it as file2, so that any differences between the local copy and the one stored in the database can be applied.

## CWCompletionRatio

Description CWCompletionRatio tells the IDE .

Prototype 

```
CW_CALLBACK CWCompletionRatio(
 CWPluginContext context,
 int totalItems,
 int completedItems);
```

Parameters The following table shows the parameters for this method:

|                |                                                                       |
|----------------|-----------------------------------------------------------------------|
| context        | Private, opaque IDE state.                                            |
| totalItems     | The total number of items the plug-in must process in this operation. |
| completedItems | The number of items the plug-in has processed so far.                 |

Remarks Plug-ins use this method to report their status to the IDE while performing a long task. For example, a plug-in could periodically use this callback while performing a recursive operation on a directory. When it receives this callback the IDE displays a visual progress bar to the user.

If your plug-in does not call this function, the IDE displays an indeterminate animated progress bar instead.

## CWGetProjectFileSpecifier

CWGetProjectFileSpecifier gets the CWFileSpec for the project file the plug-in is being asked to process.

Prototype `CW_CALLBACK CWGetProjectFileSpecifier( CWPluginContext context, CWFileSpec *projectFileSpec);`

Parameters The following table shows the parameters for this method:

|                |                                                                       |
|----------------|-----------------------------------------------------------------------|
| context        | Private, opaque IDE state.                                            |
| totalItems     | The total number of items the plug-in must process in this operation. |
| completedItems | The number of items the plug-in has processed so far.                 |

## CWIsAdvancedRequest

CWIsAdvancedRequest indicates whether the most recent request from the IDE was an advanced request.

Prototype `CW_CALLBACK CWIsAdvancedRequest( CWPluginContext context, Boolean *isAdvanced);`

Parameters The following table shows the parameters for this method:

|            |                                                                                                 |
|------------|-------------------------------------------------------------------------------------------------|
| context    | Private, opaque IDE state.                                                                      |
| isAdvanced | On return, this parameter contains true if the most recent request is advanced or false if not. |

Remarks If the request was sent in advanced mode, the plug-in should show the user a dialog box that lets the user choose the more advanced options for a command. For example, to process an advanced history command, the plug-in might show the user a dialog box that would let the user choose to filter the display by user name.

## CWIsRecursiveRequest

CWIsAdvancedRequest indicates whether the most recent request from the IDE was a recursive request.



Prototype    `CW_CALLBACK CWIsRecursiveRequest (`  
                  `CWPluginContext context,`  
                  `Boolean *isRecursive);`

Parameters    The following table shows the parameters for this method:

|             |                                                                                                                            |
|-------------|----------------------------------------------------------------------------------------------------------------------------|
| context     | Private, opaque IDE state.                                                                                                 |
| isRecursive | On return, this parameter contains <code>true</code> if the most recent request is recursive or <code>false</code> if not. |

Remarks      This callback can be used to determine if the request was sent in the recursive mode. If the request was sent in recursive mode and a directory is encountered in the `CWVCSItem` list of items to process, the plug-in should apply the command to all files in the directory and recursively on any directories the item contains.

**CWIsCommandSupportedRequest**

`CWIsCommandSupportedRequest` indicates whether the IDE sent the most recent command to determine whether the plug-in supports the command, rather than to have the plug-in perform the command.

Prototype    `CW_CALLBACK CWIsCommandSupportedRequest (`  
                  `CWPluginContext context,`  
                  `Boolean *isCommandSupported);`

Parameters    The following table shows the parameters for this method:

|                    |                                                                                                                            |
|--------------------|----------------------------------------------------------------------------------------------------------------------------|
| context            | Private, opaque IDE state.                                                                                                 |
| isCommandSupported | On return, this parameter contains <code>true</code> if the most recent request is recursive or <code>false</code> if not. |

Remarks      The plug-in uses `isCommandSupported` during request negotiation. If the IDE sent the most recent command in this mode, the plug-in should set the command status to one of the following constants:

```
 cwCommandStatusCommandUnknown
 // plug-in does not recognize command
 cwCommandStatusUnsupported
 // plug-in cannot perform command
```

Version Control System Plug-in API  
Standard Callbacks

```
 cwCommandStatusSupported
 // plug-in can perform command
```

CWIsCommandSupportedRequest has only two valid responses: `cwCommandStatusSupported` and `cwCommandStatusUnsupported`. During execution mode, if the IDE makes a request about which it did not previously query, the plug-in should return `cwCommandStatusCommandUnknown`.

If the IDE sends a command in this mode, the plug-in should not perform the command.

**CWSetCommandDescription**

`CWSetCommandDescription` lets a plug-in set custom strings in the VCS menus and in the VCS progress dialog box.

```
Prototype CW_CALLBACK CWSetCommandDescription(
 CWPluginContext context,
 CWVCSCommandDescription *description);
```

Parameters The following table shows the parameters for this method:

|             |                                                                                                   |
|-------------|---------------------------------------------------------------------------------------------------|
| context     | Private, opaque IDE state.                                                                        |
| description | A pointer to the command description structure that holds custom strings for the current command. |

Remarks Plug-ins can use this callback to return more information about the command to the IDE. These descriptions can be used by the IDE to better reflect the appropriate terminology for your command in the VCS menus and dialogs.

After a plug-in receives a negotiation request, the plug-in can use this callback to specify custom strings to display in the user interface. This callback works only in the negotiation stage.

A plug-in specifies information about a command by populating the fields of a `CWVCSCommandDescription` structure:

```
typedef struct CWVCSCommandDescription
{
 long version;
 char menuItem[40];
```

```
char progressMessage[200];
} CWVCSCommandDescription;
```

When creating a command description structure, a plug-in must set its version field to an appropriate non-zero value. The IDE stores the current version of the command description structure in `cwCommandDescriptionVersion`. Your plug-in can use that constant for the version.

In version 1 of the command description, you can set the `menuItem` string to be a short one or two word description of your command. The IDE uses `menuItem` in the VCS menus. The IDE uses the `progressMessage` string to display more detailed information about the command in the VCS progress dialog box. If the plug-in does not call `CWShowStatus`, the IDE uses the `progressMessage` string as the status message in the VCS progress dialog box.

**CWGetDatabaseConnectionInfo**

`CWGetDatabaseConnectionInfo` lets the plug-in obtain the current database connection information.

Prototype

```
CW_CALLBACK CWGetDatabaseConnectionInfo(
 CWPluginContext context,
 CWVCSDatabaseConnection *dbConnection);
```

Parameters The following table shows the parameters for this method:

|              |                                                                                                                  |
|--------------|------------------------------------------------------------------------------------------------------------------|
| context      | Private, opaque IDE state.                                                                                       |
| dbConnection | On return, this parameter contains a pointer to a structure that contains details about the database connection. |

Remarks This callback lets the plug-in obtain the current database connection information, including:

- The path to the database directory (if applicable)
- The path to the user’s local root directory
- The username and password for the user’s local root directory

The returned structure has the following form:

```
typedef struct CWVCSDatabaseConnection
{
```



Freescale Semiconductor, Inc.

Version Control System Plug-in API  
Standard Callbacks

```
CWFileSpec sDatabasePath;
CWFileSpec sProjectRoot;
char *pUsername;
char *pPassword;
};
```

CWGetCommandStatus

CWGetCommandStatus gets the IDE’s perception of the status of the current operation.

```
Prototype CW_CALLBACK CWGetCommandStatus (
 CWPluginContext context,
 CWCVSCCommandStatus *status);
```

Parameters The following table shows the parameters for this method:

|         |                                                                                                             |
|---------|-------------------------------------------------------------------------------------------------------------|
| context | Private, opaque IDE state.                                                                                  |
| status  | On return, this parameter contains a pointer to one of the values from the CWCVSCCommandStatus enumeration. |

Remarks This callback returns the IDE’s perception of the current status of the operation. A plug-in can update the state of the operation by using CWSetCommandStatus. The following enumeration shows the possible values that the IDE can return:

```
enum // command status
{
 cwCommandStatusCommandUnknown,
 cwCommandStatusUnknown,
 cwCommandStatusUnsupported,
 cwCommandStatusSupported,
 cwCommandStatusSucceeded,
 cwCommandStatusFailed,
 cwCommandStatusPartial,
 cwCommandStatusCancelled,
 cwCommandStatusConnectionLost,
 cwCommandStatusInvalidLogin
};
```

## CWSetCommandStatus

CWSetCommandStatus sets the status of the current operation.

Prototype `CW_CALLBACK CWSetCommandStatus ( CWPluginContext context, CWVCSCommandStatus status );`

Parameters The following table shows the parameters for this method:

|         |                                                                                                                   |
|---------|-------------------------------------------------------------------------------------------------------------------|
| context | Private, opaque IDE state.                                                                                        |
| status  | One of the values from the CWCVSCCommandStatus enumeration. See CWGetCommandStatus for a list of possible values. |

Remarks Plug-ins can use this callback to change the completion status of the current command. Plug-ins should set the completion state of an individual item by using CWSetVCSItem.

See Also “CWGetCommandStatus” on page 390

## CWGetVCSItemCount

CWGetVCSItemCount gets the number of items to be processed for a given request.

Prototype `CW_CALLBACK CWGetVCSItemCount ( CWPluginContext context, unsigned long *count );`

Parameters The following table shows the parameters for this method:

|         |                                                                                                                                       |
|---------|---------------------------------------------------------------------------------------------------------------------------------------|
| context | Private, opaque IDE state.                                                                                                            |
| count   | On return, this parameter contains a pointer to a long integer containing the number of items to be processed in the current request. |

Remarks When the IDE calls a VCS plug-in with certain commands, the IDE can specify a list of items (files or directories) to be processed.

## CWGetVCSItem

CWGetVCSItem gets a specified item from a list of items to be processed.

## Version Control System Plug-in API

### Standard Callbacks

**Prototype**    `CW_CALLBACK CWGetVCSItem(`  
                   `CWPluginContext context,`  
                   `long index,`  
                   `CWVCSItem *item);`

**Parameters**    The following table shows the parameters for this method:

|         |                                                                      |
|---------|----------------------------------------------------------------------|
| context | Private, opaque IDE state.                                           |
| index   | The index value of the item to get. This index starts at 0.          |
| item    | On return, this parameter contains a pointer to the item to process. |

**Remarks**    The returned item matches the following structure:

```

typedef struct CWVCSItem
{
 CWFileSpec fsItem;
 CWVCSItemStatus eItemStatus;
 CWVCSVersion version;
 // following field added for version 8 API:
 CWVCSCheckoutState eCheckoutState;
}

```

The `fsItem` member of the structure contains the `CWFileSpec` that specifies the file or directory to be processed. The `eItemStatus` member contains the completion state of the operation on that member. The following enumeration defines the possible completion states:

```

enum // item status
{
 cwItemStatusUnprocessed,
 cwItemStatusUnknown,
 cwItemStatusSucceeded,
 cwItemStatusFailed,
 cwItemStatusCancelled
};

```

After the plug-in finishes processing an item, it should change `eItemStatus` to the appropriate code and call `CWSetVCSItem` with the updated `CWVCSItem` structure.



version is a CWVCSVersion structure in which the IDE returns the current version of the file. See CWVCSStateChanged for the version structure.

eCheckoutState indicates the current check out state of the file. See CWVCSStateChanged for the constants that a plug-in can use for this value.

See Also “CWVCSStateChanged” on page 381

CWSetVCSItem

CWSetVCSItem updates information about an item that the plug-in has processed.

Prototype CW\_CALLBACK CWSetVCSItem(  
    CWPluginContext context,  
    long index,  
    CWVCSItem \*item);

Parameters The following table shows the parameters for this method:

|         |                                                             |
|---------|-------------------------------------------------------------|
| context | Private, opaque IDE state.                                  |
| index   | The index value of the item to set. This index starts at 0. |
| item    | A pointer to the item about which to update information.    |

Remarks Plug-ins use this callback to update information about an item that the plug-in has processed.

**NOTE** Plug-ins must set the value of the CWVCSCheckoutState field in the CWVCSItem data structure prior to calling CWSetVCSItem.

See Also “CWGetVCSItem” on page 391

CWGetVCSPointerStorage

CWGetVCSPointerStorage lets a plug-in get the contents of a thread-specific storage area.

Prototype CW\_CALLBACK CWGetVCSPointerStorage(  
    CWPluginContext context,

## Version Control System Plug-in API VCS Commands

```
void **storage);
```

Parameters The following table shows the parameters for this method:

|         |                                                                        |
|---------|------------------------------------------------------------------------|
| context | Private, opaque IDE state.                                             |
| storage | The address of a pointer to the contents of the private storage block. |

Remarks Version 7+ of the VCS API lets plug-ins have one non-volatile storage area for each thread on which the IDE calls the plug-in. By examining the pointer that provides access to the storage area, a plug-in can determine the thread from which the plug-in is being called.

### CWSetVCSPointerStorage

CWSetVCSPointerStorage lets a plug-in set the contents of a thread-specific storage area.

Prototype CCW\_CALLBACK CWSetVCSPointerStorage(  
CWPluginContext context,  
void \*storage);

Parameters The following table shows the parameters for this method:

|         |                                                                     |
|---------|---------------------------------------------------------------------|
| context | Private, opaque IDE state.                                          |
| storage | A pointer to the information to store in the private storage block. |

Remarks Version 7+ of the VCS API lets plug-ins have one non-volatile storage area for each thread on which the IDE calls the plug-in. This callback lets a plug-in set the contents of a per-thread storage area.

### VCS Commands

The IDE uses two modes of operation to communicate with VCS plug-ins: negotiation and execution.

#### Determining the Mode

Every time the plug-in receives a command from the IDE, it should use the CWIsCommandSupportedRequest to determine whether the IDE wants the plug-in to run the command or wants to query

the plug-in about the command. If `CWIsCommandSupportedRequest` returns `true`, the IDE wants to know whether the plug-in supports the most recent command. In other words, if `CWIsCommandSupportedRequest` returns `true`, the IDE has enter negotiation mode. If `CWIsCommandSupportedRequest` returns `false`, the IDE has entered command mode.

## Negotiation Mode

After performing an initialization sequence, the IDE queries the VCS plug-in to determine which commands the plug-in supports. The IDE uses this information to construct and enable the VCS menu within the IDE. In this mode, the plug-in should not execute the command. The plug-in should indicate whether it supports the command. The IDE queries about every possible command.

`CWIsCommandSupportedRequest` has only two valid responses: `cwCommandStatusSupported` and `cwCommandStatusUnsupported`. During execution mode, if the IDE makes a request about which it did not previously query, the plug-in should return `cwCommandStatusCommandUnknown`.

## Execution Mode

In execution mode, the IDE asks the VCS plug-in to perform version control services. When a plug-in receives a command (rather than a query) from the IDE, the plug-in should then use two callbacks to determine exactly what the IDE wants the plug-in to do:

- `CWIsAdvancedRequest`, to see if the IDE wants the plug-in to get more detail from the user before executing the command.
- `CWIsRecursiveRequest`, to see if the IDE wants the plug-in to apply the command recursively.

## Command Types

The IDE can pass five types of execution mode commands to a VCS plug-in:

- Initialization and Termination
- Database Connection and Disconnection
- Queries

- Information
- File Processing

## Advanced Options

When the IDE needs to have the user specify advanced options for a VCS operation, it has the plug-in display a dialog box, in which users can specify various advanced options. The plug-in then performs the operation, using the options specified by the user.

## Reporting Status

Plug-ins use a two-tiered approach to report their status while processing a command. Plug-ins use `CWSetCommandStatus` to report status for the command. Plug-ins also report the status of each item to be processed in a request by creating a `CWVCSItem` object for each item and updating the `eItemStatus` field in each object at various points during processing.

The plug-in starts by setting the high-level command status with `CWSetCommandStatus`. If the plug-in must report status before fully processing the command, it should return a value of `cwCommandStatusUnknown`. If the plug-in successfully processed all items, it should return a value of `cwCommandStatusSucceeded`. If the plug-in could process no items, it should return a value of `cwCommandStatusFailed`. If the plug-in successfully processed some but not all items, it should return a value of `cwCommandStatusPartial`. If the user cancels the operation, the plug-in should return a value of `cwCommandStatusCancelled`.

After it has set the high-level status for a command, the plug-in returns additional detail about each item that it processed as part of processing the command. Before processing any item, the plug-in should create a `CWVCSItem` object for each item. The plug-in should set the `eItemStatus` field of each `CWVCSItem` object to `cwItemStatusUnknown`. Then the plug-in should prepare to process the items. When the plug-in is ready to start processing the items, it should set the `eItemStatus` field of each `CWVCSItem` object to `cwItemStatusProcessed`. As the plug-in processes each item, it should set the `eItemStatus` field of each `CWVCSItem` object to `cwItemStatusSucceeded`, `cwItemStatusFailed`, or `cwItemStatusCancelled`, depending on its ability to process the

item and on whether the user canceled the operation. Changing the status of each object at certain times lets the IDE determine how much processing the plug-in was able to do if an unexpected event arises.

## VCS Requests

The IDE can send a number of requests (also called commands) to the VCS plug-in. This

The following enumeration shows all the requests that the IDE can send to a VCS plug-in:

```
enum // IDE requests to VCS plug-in
{
 reqInit,
 reqTerminate,
 reqPrefsChange, // *
 reqIdle, // *
 reqDatabaseConnect,
 reqDatabaseDisconnect,
 reqDatabaseVariables,
 reqFileAdd,
 reqFileCheckin,
 reqFileCheckout,
 reqFileComment,
 reqFileDelete,
 reqFileDestroy,
 reqFileDifference,
 reqFileGet,
 reqFileHistory,
 reqFileLabel,
 reqFileProperties,
 reqFilePurge,
 reqFileRename,
 reqFileRollback,
 reqFileStatus,
 reqFileUndoCheckout,
 reqFileVersion,
 reqFileBranch, // *
 reqFileShare, // *
 reqFileView // *
}
```



# Freescale Semiconductor, Inc.

## Version Control System Plug-in API

Initialization and Termination

```
} ;
```

**NOTE** The IDE does not send the requests with an asterisk in the comment to version 1 plug-ins.

Each request (or command) description includes a small table indicating whether the plug-in must support the request and whether the request can be advanced or recursive.

For example, the following table indicates that the plug-in must support the request but that the IDE cannot send the request in advanced or recursive mode:

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

### Initialization and Termination

The IDE uses two commands to tell the VCS plug-in it is being loaded or unloaded. These commands let the plug-in initialize or clean up its data, respectively.

**CAUTION** Plug-ins can use only the standard memory callbacks (CWGetAPIVersion, CWGetPluginRequest, CWDonePluginRequest, and CWGetIDEInfo) during initialization and termination requests.

#### reqInitalize

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                                                                                                                                                      |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description         | The IDE sends this command immediately after it loads the plug-in. When the plug-in receives this command, it should perform any initialization procedures that let it prepare for further commands. |
| Header              | CWPlugins.h                                                                                                                                                                                          |
| Returns status with | CWSetCommandStatus();<br>CWDonePluginRequest();<br>entry point return value                                                                                                                          |
| See Also            | “reqInitialize” on page 127                                                                                                                                                                          |



“CWSetCommandStatus” on page 391

“CWDonePluginRequest” on page 45

reqTerminate

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                                                                                                                                                     |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description         | The IDE sends this command to tell the VCS plug-in that it is about to be unloaded. The plug-in should finish any file operations and release any memory it uses before responding to this command. |
| Header              | CWPlugins.h                                                                                                                                                                                         |
| Returns status with | CWSetCommandStatus();<br>CWDonePluginRequest();<br>entry point return value                                                                                                                         |
| See Also            | “reqTerminate” on page 128<br><br>“CWSetCommandStatus” on page 391<br><br>“CWDonePluginRequest” on page 45                                                                                          |

Database Connection and Disconnection

The IDE uses two commands for connecting and disconnecting to a VCS database.

reqDatabaseConnect

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | The IDE uses this command to tell the VCS plug-in to connect to its database.<br><br>pDatabaseConnection specifies the connection information. The plug-in should store this information, with any other session specific information, in the context field. The IDE retains the context information for the duration of the session and passes it to the plug-in on all subsequent requests. The IDE does not modify the context information of VCS plug-ins. The plug-in must allocate the memory for the context information. |
| Header      | DropInVCS.h                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |





“CWPostDialog” on page 72

“CWDonePluginRequest” on page 45

reqDatabaseVariables

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                            |
|---------------------|----------------------------------------------------------------------------|
| Description         | This command tells the VCS plug-in to output its initialization variables. |
| Header              | DropInVCS.h                                                                |
| Returns status with | CWSetCommandStatus();                                                      |
| See Also            | “CWSetCommandStatus” on page 391                                           |

Information

The IDE uses two commands to deal with information updates.

reqIdle

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                                                                           |
|---------------------|---------------------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to advance any periodic operations it may have in progress.             |
| Header              | CWPlugins.h                                                                                                               |
| Callbacks available | memory callbacks<br>CWPreDialog();<br>CWPostDialog();<br>CWGetAPIVersion();<br>CWGetIDEInfo();                            |
| Returns status with | CWDonePluginRequest();                                                                                                    |
| See Also            | “CWPreDialog” on page 73<br>“CWPostDialog” on page 72<br>“CWGetAPIVersion” on page 52<br>“CWDonePluginRequest” on page 45 |



# Freescale Semiconductor, Inc.

## Version Control System Plug-in API

### File Processing

---

#### reqPrefsChange

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                                 |
|---------------------|-----------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in that the user changed values in one of its preference panels. |
| Header              | CWPlugins.h                                                                                                     |
| Callbacks available | All universal plug-in callbacks.                                                                                |
| Returns status with | CWDonePluginRequest();                                                                                          |
| See Also            | "CWDonePluginRequest" on page 45                                                                                |

#### File Processing

Commands that change the checkout state of files must update (if possible) the version field of `pItemData` in the `pItemList`. Items have a status of unprocessed when the plug-in starts a request. The plug-in must set the status for each item before finishing the request. See "Reporting Status" on page 396.

The overall command status should only be set to success or failure if all items have the corresponding status. When the user cancels an operation, the IDE checks each item to determine the appropriate action to take for each.

#### reqFileAdd

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                           |
|---------------------|-------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to add a list of files to the database. |
| Header              | DropInVCS.h                                                                               |
| Returns status with | CWSetCommandStatus();<br>CWSetVCSItem();                                                  |
| See Also            | "CWSetCommandStatus" on page 391<br>"CWSetVCSItem" on page 393                            |



**reqFileBranch**

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

- Description The IDE uses this command to tell the VCS plug-in to branch in a list of files in the database.
- Header DropInVCS.h
- Returns status with CWSetCommandStatus();  
CWSetVCSItem();
- See Also “CWSetCommandStatus” on page 391  
“CWSetVCSItem” on page 393

**reqFileCheckin**

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

- Description The IDE uses this command to tell the VCS plug-in to check in a list of files into the database.
- Header DropInVCS.h
- Returns status with CWSetCommandStatus();  
CWSetVCSItem();
- See Also “CWSetCommandStatus” on page 391  
“CWSetVCSItem” on page 393

**reqFileCheckout**

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

- Description The IDE uses this command to tell the VCS plug-in to check out a list of files from the database.
- Header DropInVCS.h
- Returns status with CWSetCommandStatus();  
CWSetVCSItem();
- See Also “CWSetCommandStatus” on page 391  
“CWSetVCSItem” on page 393

## Version Control System Plug-in API

### File Processing

#### reqFileComment

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to change the comment of a list of files in the database. |
| Header              | DropInVCS.h                                                                                                 |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                              |
| See Also            | "CWSetCommandStatus" on page 391<br>"CWSetVCSItem" on page 393                                              |

#### reqFileDelete

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                   |
|---------------------|---------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to delete of a list of files from the database. |
| Header              | DropInVCS.h                                                                                       |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                    |
| See Also            | "CWSetCommandStatus" on page 391<br>"CWSetVCSItem" on page 393                                    |

#### reqFileDestroy

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                                                                                    |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to destroy of a list of files in the database. This command has the same effect as a delete followed by a purge. |
| Header              | DropInVCS.h                                                                                                                                                        |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                                                                                     |
| See Also            | "CWSetCommandStatus" on page 391<br>"CWSetVCSItem" on page 393                                                                                                     |

## reqFileDifference

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

- Description The IDE uses this command to tell the VCS plug-in to output a difference listing between a list of files in the database and those local.
- Header DropInVCS.h
- Returns status with `CWSetCommandStatus()` ;  
`CWSetVCSItem()` ;
- See Also “CWSetCommandStatus” on page 391  
“CWSetVCSItem” on page 393

## reqFileGet

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

- Description The IDE uses this command to tell the VCS plug-in to get a list of files from the database.
- Header DropInVCS.h
- Returns status with `CWSetCommandStatus()` ;  
`CWSetVCSItem()` ;
- See Also “CWSetCommandStatus” on page 391  
“CWSetVCSItem” on page 393

## reqFileHistory

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

- Description The IDE uses this command to tell the VCS plug-in to output a history listing for a list of files in the database.
- Header DropInVCS.h
- Returns status with `CWSetCommandStatus()` ;  
`CWSetVCSItem()` ;
- See Also “CWSetCommandStatus” on page 391  
“CWSetVCSItem” on page 393

## Version Control System Plug-in API

### File Processing

#### reqFileLabel

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

**Description** The IDE uses this command to tell the VCS plug-in to label a list of files in the database. The start version field is used to indicate label to be applied.

**Header** DropInVCS.h

**Returns status with** CWSetCommandStatus();

**See Also** "CWSetCommandStatus" on page 391

#### reqFileProperties

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

**Description** The IDE uses this command to tell the VCS plug-in to output the properties of a list of files in the database.

**Header** DropInVCS.h

**Returns status with** CWSetCommandStatus();  
CWSetVCSItem();

**See Also** "CWSetCommandStatus" on page 391

"CWSetVCSItem" on page 393

#### reqFilePurge

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

**Description** The IDE uses this command to tell the VCS plug-in to permanently remove a list of files from the database.

**Header** DropInVCS.h

**Returns status with** CWSetCommandStatus();  
CWSetVCSItem();

**See Also** "CWSetCommandStatus" on page 391

"CWSetVCSItem" on page 393



**reqFileRename**

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                                              |
|---------------------|----------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to rename a list of files in the database. |
| Header              | DropInVCS.h                                                                                  |
| Returns status with | CWSetCommandStatus() ;<br>CWSetVCSItem() ;                                                   |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                               |

**reqFileRollback**

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                                                |
|---------------------|------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to rollback a list of files in the database. |
| Header              | DropInVCS.h                                                                                    |
| Returns status with | CWSetCommandStatus() ;<br>CWSetVCSItem() ;                                                     |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                                 |

**reqFileShare**

|          |          |           |
|----------|----------|-----------|
| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|

|                     |                                                                                                |
|---------------------|------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to share in a list of files in the database. |
| Header              | DropInVCS.h                                                                                    |
| Returns status with | CWSetCommandStatus() ;<br>CWSetVCSItem() ;                                                     |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                                 |

## Version Control System Plug-in API

### File Processing

#### reqFileStatus

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                            |
|---------------------|------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to output the status of a list of files in the database. |
| Header              | DropInVCS.h                                                                                                |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                             |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                                             |

#### reqFileUndoCheckout

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                              |
|---------------------|--------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to cancel the checkout of a list of files in the database. |
| Header              | DropInVCS.h                                                                                                  |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                               |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                                               |

#### reqFileVersion

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                                                  |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to retrieve the version and checkout state of a list of files in the database. |
| Header              | DropInVCS.h                                                                                                                      |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                                                   |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                                                                   |



reqFileView

| REQUIRED | ADVANCED | RECURSIVE |
|----------|----------|-----------|
|----------|----------|-----------|

|                     |                                                                                                                 |
|---------------------|-----------------------------------------------------------------------------------------------------------------|
| Description         | The IDE uses this command to tell the VCS plug-in to display the contents of a list of files from the database. |
| Header              | DropInVCS.h                                                                                                     |
| Returns status with | CWSetCommandStatus ( ) ;<br>CWSetVCSItem ( ) ;                                                                  |
| See Also            | “CWSetCommandStatus” on page 391<br>“CWSetVCSItem” on page 393                                                  |

VCS API Version 1 Compatibility

The plug-in libraries that you link against when writing version 7+ and later VCS plug-ins allow for backwards compatibility. That is, you can run a version 7+ plug-in under an IDE built prior to the Pro 4 IDE, from CodeWarrior 10 and up. Plug-ins running on those versions of the IDE must only use specific callbacks and do not receive certain requests.

The following sections detail the overall behavioral differences, which callbacks are supported when running in V1 compatibility mode, and which requests are not sent in V1 compatibility mode.

Version 1 VCS API

If you are writing a version 7+ plug-in and want it to run as a version 1 plug-in, your plug-in must:

- Pass true as the argument to all calls to `CWAllowV1Compatibility`.
- Assume the behavior of a V1 plug-in.
- Never use any of the unmapped callbacks.
- Never expect to receive any requests specific to later versions of the API.

## Installing on a Pre-Pro 4 CodeWarrior Installation

To properly run a version 7+ plug-in on a version of the IDE earlier than Pro 4 CodeWarrior IDE, `PluginLib4.dll` on the user's system must be replaced with the one in the SDK to gain access to the new VCS callbacks and compatibility code. Without it, version 7+ VCS plug-ins cannot function.

## Behavioral Differences

When a VCS plug-in runs as a version 1 plug-in, it should expect the following major behavioral differences:

- The IDE never gives idle time to the plug-in (see “`reqIdle`” on page 401). The plug-in must do all of its processing during requests.
- The IDE sends no notification about preferences changes.
- The IDE always prepares files before calling the plug-in's main entry point. The plug-in needs to perform no pre- or post-file actions, but users can experience long delays.
- The IDE processes all events before calling the plug-in. The plug-in never sees events.

You can write a plug-in that is intended to run as a version 1 plug-in in two ways. You can implement work-arounds for the missing features. For example, you could write a Time Manager task to mimic `reqIdle`. If you use work-arounds, you can test them by calling `CWAllowV1Compatibility` and checking the `isV1` value.

The other way involves writing your plug-in such that it never needs the missing features. If you choose this method of creating a plug-in, your plug-in must set certain flags to tell the IDE what to expect. Your plug-in should specify the following as the capability flags in the flag resource or entry point return structure:

```
flags.dropinflags =
 vcsDoesntPrePostSingleFile |
 vcsDoesntPrePostRecursiveFile |
 vcsRequiresEvents;
```

---

**NOTE** The IDE puts all the constants, version structures (struct `tm`), VCS items, and the file structures (FSSpec on Mac, full path on Windows) in the version 7+ format. When porting an older plug-in,

remember the differences in the structures and change how you access their members accordingly.

---

## Supported Callbacks

The functionality of the version 1 VCS API limits the callbacks that a plug-in can use when it runs as a version 1 plug-in. Such a plug-in can only use the following callbacks:

Standard callbacks:

- CWGetPluginRequest
- CWDonePluginRequest
- CWGetAPIVersion
- CWReportMessage
- CWAlert
- CWShowStatus
- CWUserBreak
- CWGetNamedPreferences
- CWCreateNewTextDocument
- CWAllocMemHandle
- CWFreeMemHandle
- CWGetMemHandleSize
- CWResizeMemHandle
- CWLockMemHandle
- CWUnlockMemHandle
- CWPreDialog
- CWPostDialog

VCS-specific callbacks:

- CWAllowV1Compatibility
- CWGetComment
- CWDoVisualDifference (Pro 2 and later only)
- CWGetProjectFileSpecifier
- CWIsAdvancedRequest



## Freescal Semiconductor, Inc.

### Version Control System Plug-in API

*Porting a Version 1 VCS Plug-in to Version 7+*

---

- `CWIsRecursiveRequest`
- `CWIsCommandSupportedRequest`
- `CWGetDatabaseConnectionInfo`
- `CWGetCommandStatus`
- `CWSetCommandStatus`
- `CWGetVCSItemCount`
- `CWGetVCSItem`
- `CWSetVCSItem`

Plug-ins can use only the following callbacks during initialization and termination requests: `CWGetPluginRequest`, `CWDonePluginRequest`, `CWPreDialog`, `CWPostDialog`, and `CWGetAPIVersion`.

If you try to use a callback that is not supported in the version 1 VCS API, the IDE returns an error and ignores the callback.

#### Unsupported Requests

A version 1 plug-in does not receive the following requests:

- `reqIdle`
- `reqPrefsChange`
- `reqFileBranch`
- `reqFileShare`
- `reqFileView`

### Porting a Version 1 VCS Plug-in to Version 7+

In order to ease the process of porting an existing version 1 VCS plug-in to version 7+, a header file called `DropInVCS_V1Compatibility.h` can be included. This header file lets you avoid renaming all the constants and structure types that are supported in the old Version 1 API. You must change your logical structure for the following changed structures:

- `CWVCSDateTime`
- `CWVCSItemData`
- `CWVCSFileSpecifier`



In general, you have two options for porting a version 1 plug-in to the new API. You can create your own version 1 structures from the old header and translate from the new accessor to the old parameter block structure on entry and reverse on exit. In this case, you must also implement your own glue routines for the callbacks that accept version 1 structures and translate them to version 7+ structures when calling back to the IDE. Although this method allows existing code to work, using newer callbacks directly in the code can cause much extra work, thanks to translating between formats.

As your second option, you can eliminate all the version 1 callbacks and parameter block references and rewrite your code to use the version 7+ API directly. The following process describes how to rewrite a plug-in for version 7+:

- 1) Change the callbacks you use to the new VCS callbacks and standard plug-in callbacks.
- 2) Eliminate any access to a parameter block and replace it with the accessor routines.
- 3) Insert calls to the `CWPreFileAction` and `CWPostFileAction` callbacks when working with files.
- 4) Modify the format of your main entry point to make the required calls to `CWAllowV1Compatibility` and `CWDonePluginRequest`.
- 5) Modify your `Flag` resource or entry point to use the new Version 3 `Flag` format.
- 6) Examine each point where you use a `CWVCSDatetime` and `CWVCSItem` structure to be sure that you are using the proper version 7+ structures.

The tables below map old callbacks to new callbacks and parameter block entries to accessors.

## Version Control System Plug-in API

Porting a Version 1 VCS Plug-in to Version 7+

### V1 to V7 Callback Mappings

Version 7 plug-ins use the following callbacks for the function pointer entries in the version 1 parameter block:

| Version 1 Callback             | Version 7 Callback      |
|--------------------------------|-------------------------|
| pVCSPreDialogRoutine           | CWPreDialog             |
| pVCSPostDialogRoutine          | CWPostDialog            |
| pVCSMessageOutputRoutine       | CWReportMessage         |
| pVCSReportProgressRoutine      | CWShowStatus            |
| pVCSGetPreferencesRoutine      | CWGetNamedPreferences   |
| pVCSCreateDocumentRoutine      | CWCreateNewTextDocument |
| pVCSYieldTimeRoutine           | CWUserBreak             |
| pVCSGetCommentRoutine          | CWGetComment            |
| pVCSUpdateCheckoutStateRoutine | CWFileStateChanged      |

### V1 Param Block Members to Accessor Mappings

| V1 PB Member      | Retrieval Accessor          | Assignment Accessor    |
|-------------------|-----------------------------|------------------------|
| request           | CWGetPluginRequest          |                        |
| version           | CWGetAPIVersion             |                        |
| context           | CWGetVCSPointerStorage      | CWSetVCSPointerStorage |
| storage           |                             |                        |
| targetfile        | CWGetProjectFileSpecifier   |                        |
| bAdvanced         | CWIsAdvancedRequest         |                        |
| bRecursive        | CWIsRecursiveRequest        |                        |
| bCheckIfSupported | CWIsCommandSupportedRequest |                        |
| eCommandStatus    | CWGetCommandStatus          | CWSetCommandStatus     |
| sDatabasePath     | CWGetDatabaseConnectionInfo |                        |
| sProjectRoot      | CWGetDatabaseConnectionInfo |                        |
| rcUsername        | CWGetDatabaseConnectionInfo |                        |
| rcPassword        | CWGetDatabaseConnectionInfo |                        |



Freescale Semiconductor, Inc.

Version Control System Plug-in API  
Porting a Version 1 VCS Plug-in to Version 7+

| V1 PB Member | Retrieval Accessor | Assignment Accessor |
|--------------|--------------------|---------------------|
| lItemCount   | CWGetVCSItemCount  |                     |
| pItemList    | CWGetVCSItem       | CWSetVCSItem        |



# Freescal Semiconductor, Inc.

## Version Control System Plug-in API

*Porting a Version 1 VCS Plug-in to Version 7+*

---

## Section 2: COM API

---

This section includes the following chapters:

- Introduction to the COM API
- Access Paths
- Application
- Collections
- Commands
- Components
- Creatable Items
- Designs
- Dialog Services
- Documents
- Error Info
- Files
- Menus
- Messages
- Projects
- Symbols
- Targets
- Text
- Toolbar
- Version Control
- Windows



**Freescale Semiconductor, Inc.**

---

# Introduction to the COM API

---

COM is Microsoft's Component Object Model. You can learn more about COM on the web at:

---

<http://msdn.microsoft.com>

---

For information on writing IDE plug-ins, see the *IDE SDK Developer's Guide*. The information needed to write plug-ins is mainly in that manual.

This chapter contains the following sections:

- Requirements — what you'll need to develop COM plug-ins
- What You Should Already Know — what this manual assumes you know about COM programming
- Starting Points — how to use this manual

## Requirements

To write COM programs that control the CodeWarrior IDE, you'll need a CodeWarrior package that comes with the tools and files needed to develop software for the operating system or computer platform on which your program will run.

Follow the instructions in the QuickStart guide of your CodeWarrior product to install the software.

## Introduction to the COM API

### *What You Should Already Know*

---

## What You Should Already Know

The manual shows you how to use the CodeWarrior COM APIs in your programs to manipulate the CodeWarrior IDE.

It is assumed that you have some experience with an object-oriented language such as C++, Java, or VBScript and are familiar with Microsoft COM.

If you are not yet familiar with Microsoft COM, the following website has everything you need to learn the basics of COM, including tutorials:

<http://microsoft.com/com/default.asp>

## Starting Points

The COM API exposes two kinds of interfaces: Services and Callbacks.

### Services

The IDE exposes services that you can call through an interface pointer. Most of the interfaces exposed by the IDE are services.

Examples of services include:

- ICodeWarriorApp
- ICodeWarriorMenu
- ICodeWarriorWindow

### Callbacks

In some cases, the IDE calls back into your code. For such an interface, you inherit and implement the abstract interface in full, since every method has been declared as pure virtual. You must write the whole thing from scratch. No inherited implementation exists, only an inherited interface. COM and Java share this feature: objects inherit only interfaces.

Examples of callbacks include:



# Freescale Semiconductor, Inc.

## Introduction to the COM API

*Callbacks*

- 
- ICodeWarriorToggleButtonToolBarItem
  - ICodeWarriorPopupMenuToolBarItem
  - ICodeWarriorCustomToolBarItem



# Freescale Semiconductor, Inc.

## Introduction to the COM API

### *Callbacks*

---

# Access Paths

---

This chapter shows how to use the Access Paths API to create and manipulate access paths in the CodeWarrior IDE.

## Access Paths API Reference

This section describes the methods contained in the following interfaces:

- ICodeWarriorAccessPath
- ICodeWarriorAccessPaths
- ICodeWarriorUserTree

The following data types are used with these interfaces:

- EAccessPathLocation
- EAccessPathType
- EUserDefinedTree

ICodeWarriorAccessPath

This interface represents a CodeWarrior access path.

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface exposes the following methods:

|                        |                        |
|------------------------|------------------------|
| get_AccessPathLocation | get_SubDirectories     |
| get_AccessPathType     | get_UserTree           |
| get_Path               | put_AccessPathLocation |
| get_Recursive          | put_Recursive          |

get\_AccessPathLocation

This method gets the origin of this access path.

```
virtual HRESULT AccessPathLocation(
 EAccessPathLocation *pval) = 0;
```

pval

On return, this parameter contains a pointer to a value within the range defined by the EAccessPathLocation enumeration, indicating the origin of this access path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

## get\_AccessPathType

This method gets the type of this access path.

```
virtual HRESULT get_AccessPathType(
 EAccessPathType *pval) = 0;
```

pval

On return, this parameter a pointer to a value in the range defined by the EAccessPathType enumeration, representing whether this access path is a user path or a system path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Path

This method gets a file specification for this access path.

```
virtual HRESULT get_Path(IFileSpec **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the folder for this access path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Access Paths

### *get\_Recursive*

---

#### get\_Recursive

This method gets whether an access path is recursive or not.

```
virtual HRESULT get_Recursive(
 VARIANT_BOOL *pval) = 0;
```

pval

On return, this parameter contains a pointer to a boolean value that is set to `true` if this access path is recursive or `false` if this access path is not recursive.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### get\_SubDirectories

This method gets a list of all subfolders contained by the folder pointed to by this access path.

```
virtual HRESULT get_SubDirectories(
 ICodeWarriorAccessPathCollection **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the list of access paths, one for each subfolder contained by the folder pointed to by this access path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_UserTree

If this access path has a user-defined origin, this method gets the corresponding user tree object. If this access path instead uses one of the origins defined in EAccessPathLocation, this method gets nothing.

```
virtual HRESULT get_UserTree(
 ICodeWarriorUserTree **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the user tree for this access path.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed. This method can also return nothing if you specify one of the access paths defined in the EAccessPathLocation enumeration.

**See Also** "EAccessPathLocation" on page 440

---

## put\_AccessPathLocation

This method sets the origin of this access path.

```
virtual HRESULT AccessPathLocation(
 EAccessPathLocation val) = 0;
```

val

Set this parameter to a value within the range defined by the EAccessPathLocation enumeration, indicating the origin of this access path.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



**Access Paths**

*put\_Recursive*

---

---

**put\_Recursive**

This method sets whether this access path is recursive.

```
virtual HRESULT put_Recursive(
 VARIANT_BOOL pval) = 0;
```

*pval*

Set this parameter to `true` if this access path is recursive or `false` if this access path is not recursive.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.



## ICodeWarriorAccessPaths

### Inherited Interfaces

- IDispatch
- IUnknown

### Methods

This interface provides the following methods:

|                            |                           |
|----------------------------|---------------------------|
| ApplyChanges               | get_SystemAccessPaths     |
| CreateAccessPath           | get_UserAccessPaths       |
| CreateAccessPathByFileSpec | put_AlwaysSearchUserPaths |
| CreateAccessPathByPosition | RemoveAccessPath          |
| get_AlwaysSearchUserPaths  |                           |

### ApplyChanges

This method applies any changes you have made to this access path. You must call this method in order for changes you make to take effect.

```
virtual HRESULT ApplyChanges(void) = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### CreateAccessPath

This method creates a new access path by using a string to specify the new access path. To remove an access path, use RemoveAccessPath.

```
virtual HRESULT CreateAccessPath(
 BSTR path,
 VARIANT_BOOL Recursion,
```

## Access Paths

### CreateAccessPathByFileSpec

---

```
EAccessPathLocation inLocation,
EAccessPathType inType,
ICodeWarriorAccessPath **pval) = 0;
```

path

The full path to the folder you want to add.

Recursion

Set this parameter to true if you would like the CodeWarrior IDE to search subfolders of this access path. Otherwise, set it to false.

inLocation

A value within the range defined by the EAccessPathLocation enumeration, indicating the origin of the new access path.

inType

A value in the range defined by the EAccessPathType enumeration, indicating whether this is a user path or a system path.

pval

On return, this parameter contains the address of a pointer to the newly created access path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "RemoveAccessPath" on page 435

---

### CreateAccessPathByFileSpec

This method creates a new access path by using the IFileSpec interface. To remove an access path, use RemoveAccessPath.

```
virtual HRESULT CreateAccessPathByFileSpec(
 IFileSpec *path,
 VARIANT_BOOL Recursion,
 EAccessPathLocation inLocation,
```

```

EAccessPathType inType,
ICodeWarriorAccessPath **pval) = 0;

```

path

A pointer to the IFileSpec interface representing the folder you want to add.

Recursion

Set this parameter to true if you would like the CodeWarrior IDE to search subfolders of this access path. Otherwise, set it to false.

inLocation

A value within the range defined by the EAccessPathLocation enumeration, indicating the origin of this access path.

inType

A value in the range defined by the EAccessPathType enumeration, indicating whether this is a user path or a system path.

pval

On return, this parameter contains the address of a pointer to the newly created access path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "RemoveAccessPath" on page 435

## CreateAccessPathByPosition

This method creates a new access path by using the EAccessPathLocation interface. To remove an access path, use RemoveAccessPath.

```

virtual HRESULT CreateAccessPathByPosition(
 BSTR path,
 VARIANT_BOOL Recursion,

```

## Access Paths

### CreateAccessPathByPosition

---

```
EAccessPathLocation inLocation,
EAccessPathType inType,
BSTR userTreeName,
long position,
ICodeWarriorAccessPath **pval) = 0;
```

path

The path you want to add.

Recursion

Set this parameter to true if you would like the CodeWarrior IDE to search subfolders of this access path. Otherwise, set it to false.

inLocation

A value within the range defined by the EAccessPathLocation enumeration, indicating the origin of this access path.

inType

A value in the range defined by the EAccessPathType enumeration, indicating whether this is a user path or a system path.

userTreeName

The name of the user tree to which the path is being added.

position

An integer indicating the position for the new path.

pval

On return, this parameter contains the address of a pointer to the newaccess path.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "RemoveAccessPath" on page 435

---

## get\_AlwaysSearchUserPaths

This method obtains the state of the **Always Search User Paths** option. When enabled, this option tells CodeWarrior to always search user paths before searching system paths.

```
virtual HRESULT get_AlwaysSearchUserPaths(
 VARIANT_BOOL *pval) = 0;
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if this option is enabled or `false` if this option is disabled.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_SystemAccessPaths

This method gets a list of all system access paths in this collection. The system paths collection contains all compiler-relative paths.

```
virtual HRESULT get_SystemAccessPaths(
 ICodeWarriorAccessPathCollection **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all system access paths.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487



## Access Paths

### *get\_UserAccessPaths*

---

### get\_UserAccessPaths

This method gets a list of all user access paths in this collection. The user paths collection contains all project-relative paths.

```
virtual HRESULT get_UserAccessPaths(
 ICodeWarriorAccessPathCollection **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all user access paths.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

---

### put\_AlwaysSearchUserPaths

This method sets the state of the **Always Search User Paths** option. When enabled, this option tells the CodeWarrior IDE to always search user paths before searching system paths.

```
virtual HRESULT put_AlwaysSearchUserPaths(
 VARIANT_BOOL pval) = 0;
```

pval

Set this parameter to `true` if this option is enabled or `false` if this option is disabled.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

## RemoveAccessPath

This method removes an access path from the list of access paths. To create an access path, use `CreateAccessPath`, `CreateAccessPathByFileSpec`, or `CreateAccessPathByPosition`.

```
virtual HRESULT RemoveAccessPath(
 ICodeWarriorAccessPath* pval) = 0;
```

`pval`

A pointer to the access path to remove.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “CreateAccessPath” on page 429  
“CreateAccessPathByFileSpec” on page 430  
“CreateAccessPathByPosition” on page 431



ICodeWarriorUserTree

The ICodeWarriorUserTree interface lets you work with user-defined access paths.

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface provides the following methods:

|             |             |
|-------------|-------------|
| get_KeyName | put_KeyName |
| get_Name    | put_Name    |
| get_Type    | put_Type    |
| get_Value   | put_Value   |

get\_KeyName

This method gets the key name of the current user tree.

```
virtual HRESULT get_KeyName(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains a pointer to the key name of the current user tree.



---

## get\_Name

This method gets the name of this user tree.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the name of this user tree.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Type

This method gets the type of this user tree.

```
virtual HRESULT get_Type(
 EUserDefinedTree *val) = 0;
```

val

On return, this parameter contains a pointer to a value in the range defined by the EUserDefinedTree enumeration.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Value

This method gets the value of this user tree.

```
virtual HRESULT get_Value(
 BSTR *pval) = 0;
```



# Freescal Semiconductor, Inc.

## Access Paths

put\_KeyName

pval

On return, this parameter contains the value of this user tree.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## put\_KeyName

This method sets the key name of the current user tree.

```
virtual HRESULT put_KeyName(
 BSTR val) = 0;
```

val

The string to which to set the key name of the current user tree.

## put\_Name

This method sets the name of this user tree.

```
virtual HRESULT put_Name(
 BSTR val) = 0;
```

val

The name of this user tree.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## put\_Type

This method sets the type of this user tree.

```
virtual HRESULT put_Type(

```



**Access Paths**  
*put\_Value*

---

```
EUserDefinedTree val) = 0;
```

val

A value in the range defined by the EUserDefinedTree enumeration.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

put\_Value

This method sets the value of this user tree.

```
virtual HRESULT put_Value(
 BSTR val) = 0;
```

val

The value of this user tree.



Access Paths Data Types

The following data types are used with the Access Paths API:

- EAccessPathLocation
- EAccessPathType
- EUserDefinedTree

EAccessPathLocation

This enumeration describes the origin of an access path.

Table 9.1 EAccessPathLocation Enumeration

| Constant          | Description                                                             |
|-------------------|-------------------------------------------------------------------------|
| kAbsolute         | An absolute (that is, fully qualified) file path.                       |
| kProjectRelative  | A file path relative to the location of the project file.               |
| kCompilerRelative | A file path relative to the location of the compiler’s executable file. |
| kSystemRelative   | A file path relative to the location of the operating system files.     |
| kUserDefined      | A file path relative to a user-defined location.                        |

EAccessPathType

This enumeration describes the type of an access path.

Table 9.2 EAccessPathType Enumeration

| Constant    | Description            |
|-------------|------------------------|
| kUserPath   | A user-specified path. |
| kSystemPath | A system path.         |

EUserDefinedTree

This enumeration describes the type of a user tree.

Table 9.3 EUserDefinedTree Enumeration

| Constant      | Description                                       |
|---------------|---------------------------------------------------|
| kAbsolutePath | An absolute (that is, fully qualified) file path. |
| kEnvironment  | A path stored as an environment variable.         |
| kRegistry     | A path stored in the registry.                    |



# Freescale Semiconductor, Inc.

**Access Paths**  
*Access Paths Data Types*

---

# Application

---

This chapter describes the Application API to work with and receive events from the CodeWarrior IDE about the application object..

This chapter contains the following sections:

- Application API Overview
- Application API Reference

## Application API Overview

The Application API is a set of interfaces that allows a plug-in to manipulate and receive events from the CodeWarrior IDE.

You can use the application object API to manipulate application properties, for project and target management, document and file management, command management, and other tasks.



**Application**

*Application API Reference*

---

## **Application API Reference**

This section describes the functions contained in the following interfaces:

- ICodeWarriorApp
- ICodeWarriorAppEvents
- ICodeWarriorCompare

These interfaces use the following data types:

- ECodeWarriorConvertOption
- ECodeWarriorRevertPanelOption
- ECodeWarriorProjectOption
- ECodeWarriorSaveOption
- ECodeWarriorVCSInteractionOption
- Standard Folder Names

## ICodeWarriorApp

This is the CodeWarrior application object. Use it to manipulate application properties, for project and target management, document and file management, command management, and other miscellaneous tasks.

### Inherited Interfaces

- IDispatch
- IUnknown

### Methods

This interface provides the following methods:

|                            |                                  |
|----------------------------|----------------------------------|
| AddCreatableItem           | get_VersionControl               |
| AddUserTree                | get_Visible                      |
| AttemptModify              | ImportProject                    |
| CreateProject              | ImportProjectByFileSpec          |
| CreateProjectByFileSpec    | IsBuildInProgress                |
| CreateUserTree             | OpenDocument                     |
| DoCommand                  | OpenProject                      |
| FindDesignForDataModel     | OpenProjectWithOptions           |
| FindLogicalFolder          | OpenProjectByFileSpec            |
| get_ActiveDocument         | OpenProjectByFileSpecWithOptions |
| get_Application            | OpenTextDocument                 |
| get_CompareInterface       | OpenTextDocumentByFileSpec       |
| get_CreatableItems         | OpenUntitledTextDocument         |
| get_Debugger               | put_AllowUserInteraction         |
| get_DefaultProject         | put_Visible                      |
| get_DefaultProjectDocument | QueueDeferredAction              |
| get_Documents              | Quit                             |
| get_FullName               | RemoveCreatableItem              |
| get_Name                   | RemoveNamedPluginData            |
| GetNamedPluginData         | RemoveUserTree                   |



Application

AddCreatableItem

|               |                    |
|---------------|--------------------|
| get_Projects  | SetNamedPluginData |
| GetSetting    | SetSetting         |
| get_UserTrees |                    |

AddCreatableItem

Use this method to add a creatable item to the CodeWarrior IDE. Creatable items encapsulate the items visible in the **New** window. See “Creatable Items” on page 523 for more information on creatable items.

```

virtual HRESULT AddCreatableItem(
 IUnknown *item) = 0;

item

```

The creatable item to add to the CodeWarrior IDE.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

AddUserTree

This method adds an existing user tree to the application.

```

virtual HRESULT AddUserTree(
 ICodeWarriorUserTree *pval) = 0;

pval

```

A pointer to the user tree to add to the application.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorUserTree” on page 436

---

## AttemptModify

Use this method to request that a CodeWarrior file be made writable.

**Prototype**    `virtual HRESULT AttemptModify(  
                  IFileSpec *fileSpec,  
                  ECodeWarriorVCSInteractionOption uiParameter,  
                  ICodeWarriorProject *project) = 0;`

**Parameters**    `fileSpec`

A pointer to the IFileSpec interface containing the file in question.

`uiParameter`

A ECodeWarriorVCSInteractionOption set to a value in the range defined by the ECodeWarriorVCSInteractionOption enumeration, representing how the IDE should handle user interaction.

`project`

If the file is in a project that has version control enabled, use a pointer to the ICodeWarriorProject interface. Otherwise, use NULL.

**Returns**        `S_OK` if the file was found and made writable, `S_FALSE` if the file cannot be modified, or `E_ABORT` if the user cancelled the operation.

---

## CreateProject

Use this method to create a new project in the CodeWarrior application by specifying a file path.

`virtual HRESULT CreateProject(  
                  BSTR filePath,  
                  BSTR linkerName,  
                  BSTR designName,  
                  BSTR targetName,`

## Application

### CreateProject

---

```
VARIANT_BOOL fMakeVisible,
ICodeWarriorProject **pval) = 0;
```

filepath

The full path to the new project file. CodeWarrior creates this file. It should not exist prior to this call.

linkerName

The name of the linker used in this project. CodeWarrior configures the new project to use the linker you specify here. You can set this value to NULL or an empty string to use the default values from the project.

designName

The name of the initial design in this project. CodeWarrior creates a new design with the name specified. You can set this value to NULL or an empty string to use the default values from the project.

targetName

The name of the initial target in this project. CodeWarrior creates a new target with the name specified. You can set this value to NULL or an empty string to use the default values from the project.

fMakeVisible

Set this parameter to true if this project should be visible to users or false if not.

pval

On return, this parameter contains the address of a pointer to the new project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

## CreateProjectByFileSpec

Use this method to create a new project in the CodeWarrior application, by file specification.

```
virtual HRESULT CreateProjectByFileSpec(
 IFileSpec *projectFileSpec,
 BSTR linkerName,
 BSTR designName,
 BSTR targetName,
 IFileSpec *stationeryFileSpec,
 VARIANT_BOOL fMakeVisible,
 ICodeWarriorProject **pval) = 0;
```

**projectFileSpec**

A pointer to the IFileSpec interface. CodeWarrior creates this file. It should not exist prior to this call.

**linkerName**

The name of the linker used in this project. CodeWarrior configures the new project to use the linker you specify here. You can set this value to NULL or an empty string to use the default values from the project.

**designName**

The name of the initial design in this project. CodeWarrior creates a new design with the name specified. You can set this value to NULL or an empty string to use the default values from the project.

**targetName**

The name of the initial target in this project. CodeWarrior creates a new target with the name specified. You can set this value to NULL or an empty string to use the default values from the project.

**stationeryFileSpec**

If this project is to be based on existing stationery, use a pointer to the IFileSpec interface set to a stationery project file. If not,

## Application

### CreateUserTree

---

use NULL.

fMakeVisible

Set this parameter to true if this project should be visible to users or false if not.

pval

On return, this parameter contains the address of a pointer to the new project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599  
"ICodeWarriorProject" on page 630

---

## CreateUserTree

This method creates a new user tree.

```
virtual HRESULT CreateUserTree(
 BSTR displayName,
 BSTR value,
 EUserDefinedTree type,
 BSTR keyName,
 ICodeWarriorUserTree **pVal) = 0;
```

displayName

The name of the user tree that will appear in the IDE.

value

The value string of the user tree.

type

The type of the tree, which must be one of the values specified by the EUserDefinedTree Tree enumeration.



keyName

The key name of the user tree.

pval

On return, this parameter contains the address of a pointer to the new user tree.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "EUserDefinedTree" on page 441  
"ICodeWarriorUserTree" on page 436

DoCommand

Use this method to invoke a command in the CodeWarrior IDE.

```
virtual HRESULT DoCommand(
 long commandID) = 0;
```

commandID

Set this long value to the command ID of the command to invoke. The command ID must be previously registered with the IDE.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Commands API Overview" on page 493

FindDesignForDataModel

Use this method

```
virtual HRESULT FindDesignForDataModel(
 IUnknown *dataModel,
 ICodeWarriorDesign **project) = 0;
```

## Application

### FindLogicalFolder

---

dataModel

Supply a pointer to the IUnknown interface containing the data model corresponding to the design you are looking for.

project

On return, this parameter contains the address of a pointer to the design corresponding to the specified data model.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorDesign" on page 542

---

## FindLogicalFolder

Use this method to obtain a file specification for one of the standard folders used by the CodeWarrior IDE. A list of these folder names is provided under "Standard Folder Names" on page 484.

```
virtual HRESULT FindLogicalFolder(
 BSTR folderName,
 IFileSpec **folder) = 0;
```

folderName

The name of the folder you want. For a list of accepted folder names, see "Standard Folder Names" on page 484.

folder

On return, this parameter contains the address of a pointer to a file specification for the folder in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599

## get\_ActiveDocument

Call this method to obtain the currently active document in the CodeWarrior application.

```
virtual HRESULT get_ActiveDocument(
 ICodeWarriorDocument **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the active document in the CodeWarrior application.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorDocument" on page 572

---

## get\_Application

Call this method to obtain the CodeWarrior application object.

```
virtual HRESULT get_Application(
 IDispatch **pval) = 0;
```

pval

Upon return this parameter contains the address of a pointer to the CodeWarrior application object.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Application

*get\_CompareInterface*

---

### get\_CompareInterface

This method gives access the comparison dialog, so that the ICodeWarriorCompare interface can be used to compare files and folders.

```
virtual HRESULT get_CompareInterface(
 ICodeWarriorCompare **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the comparison information.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorCompare" on page 479

---

### get\_CreatableItems

Call this method to obtain a collection of all creatable items in the CodeWarrior application.

```
virtual HRESULT get_CreatableItems(
 ICodeWarriorCreatableItemCollection **pval
) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all creatable items in the CodeWarrior application.

---

### get\_Debugger

This method gets an object that defines the current debugger.

```
virtual HRESULT get_Debugger(

```

---

```
ICodeWarriorDebugger **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the object that defines the current debugger.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_DefaultProject

Use this method to obtain the default project object in the CodeWarrior application.

```
virtual HRESULT get_DefaultProject(
 ICodeWarriorProject **project) = 0;
```

project

On return, this parameter contains the address of a pointer to the default project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---

## get\_DefaultProjectDocument

Call this method to obtain the default project document in the CodeWarrior application.

```
virtual HRESULT get_DefaultProjectDocument(
 ICodeWarriorProjectDocument **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the default project document in the CodeWarrior application.

## Application

### *get\_Documents*

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectDocument" on page 580

---

## get\_Documents

Call this method to obtain a collection of all open documents in the CodeWarrior application.

```
virtual HRESULT get_Documents(
 ICodeWarriorDocumentCollection **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all open documents in the CodeWarrior application.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_FullName

Call this method to obtain the full path of the CodeWarrior application file.

```
virtual HRESULT get_FullName(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the full path to the application file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



get\_Name

Call this method to obtain the name of the CodeWarrior application file.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the name of the application file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

GetNamedPluginData

Use this method to obtain plug-in data from a given plug-in.

```
virtual HRESULT GetNamedPluginData(
 BSTR resourceName,
 IStream **pluginData) = 0;
```

resourceName

The name of the plug-in resource you want to obtain data from.

pluginData

On return, this parameter contains the address of a pointer to the data for plug-in resource specified.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Application

*get\_Projects*

---

### get\_Projects

Call this method to obtain a collection of all currently open projects in the CodeWarrior application.

```
virtual HRESULT get_Projects(
 ICodeWarriorProjectCollection **pval) = 0;
```

*pval*

On return, this parameter contains the address of a pointer to a collection of all currently open projects in the CodeWarrior application.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### GetSetting

Use this method to obtain the value of a given IDE preference setting.

```
virtual HRESULT GetSetting(
 BSTR settingsName,
 VARIANT *pval) = 0;
```

*settingsName*

The name of the setting value to get.

*pval*

On return, this parameter contains the value of the specified setting.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

---

## get\_UserTrees

This method gets the user-defined trees, as a collection.

```
virtual HRESULT get_UserTrees(
 ICodeWarriorUserTreeCollection **pval) = 0;
```

*pval*

On return, this parameter contains the address of a pointer to a collection that holds the user-defined trees.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

---

## get\_VersionControl

Use this method to obtain the version control interface to the CodeWarrior application.

```
virtual HRESULT get_VersionControl(
 ICodeWarriorVersionControl **vcs) = 0;
```

*vcs*

On return, this parameter contains the address of a pointer to the version control interface to the CodeWarrior application.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorVersionControl" on page 804

## Application

*get\_Visible*

---

### get\_Visible

Call this method to obtain the visible state of the CodeWarrior application.

```
virtual HRESULT get_Visible(
 VARIANT_BOOL *pval) = 0;
```

pval

On return, this parameter is set to `true` if the application is visible or `false` if the application is not visible.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### ImportProject

Use this method to import an XML project file into the CodeWarrior IDE, specifying the full path to the import file.

```
virtual HRESULT ImportProject(
 BSTR textFilePath,
 BSTR projectFilePath,
 VARIANT_BOOL fMakeVisible,
 ICodeWarriorProject **pval) = 0;
```

textFilePath

The full path to the XML file you are importing.

projectFilePath

The full path to the new project file. This file must not exist. It is created by CodeWarrior.

fMakeVisible

Set this parameter to `true` if this operation is to be visible to the user or `false` if the operation should be hidden from the user.

pval

On return, this parameter contains the address of a pointer to the resulting project.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProject" on page 630

---

## ImportProjectByFileSpec

Use this method to import an XML project file into the CodeWarrior IDE, specifying a file specification to the import file.

```
virtual HRESULT ImportProjectByFileSpec(
 IFileSpec *textFileSpec,
 IFileSpec *projectFileSpec,
 VARIANT_BOOL fMakeVisible,
 ICodeWarriorProject **pval) = 0;
```

textFileSpec

A pointer to the IFileSpec interface containing the file specification for the XML file you are importing.

projectFileSpec

A pointer to the IFileSpec interface containing the file specification for the new project file. This file must not exist. It is created by CodeWarrior.

fMakeVisible

Set this parameter to true if this operation is to be visible to the user or false if the operation should be hidden from the user.

pval

On return, this parameter contains the the address of a pointer to the new project.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

## Application

### *IsBuildInProgress*

---

See Also    "IFileSpec" on page 599  
               "ICodeWarriorProject" on page 630

---

## IsBuildInProgress

Use this method to determine if a build is currently in progress in the CodeWarrior IDE.

```
virtual HRESULT IsBuildInProgress(
 VARIANT_BOOL *pval) = 0;
```

pval

On return, this parameter is set to `true` if a build is in progress or `false` if not.

Returns    S\_OK if this method call succeeded or an appropriate error if it failed.

---

## OpenDocument

This method opens a document specified by a full file path.

```
virtual HRESULT OpenDocument(
 BSTR filePath) = 0;
```

filePath

The full path to the document to open.

## OpenProject

Use this method to open a project in the CodeWarrior IDE, by specifying the project file by full path.

```
virtual HRESULT OpenProject(
 BSTR filePath,
 VARIANT_BOOL fMakeVisible,
 ECodeWarriorConvertOption convertOption,
 ECodeWarriorRevertPanelOption revertOption,
 ICodeWarriorProject **pval) = 0;
```

**filePath**

The full path to the project file.

**fMakeVisible**

Set this parameter to `true` if this operation is to be visible to the user or `false` if the operation should be hidden from the user.

**convertOption**

A value in the range defined by the `ECodeWarriorConvertOption` enumeration, representing how the CodeWarrior IDE should handle this project if the project is found to be a project created by an older version of the IDE.

**revertOption**

A value in the range defined by the enumeration `ECodeWarriorRevertPanelOption`, representing whether revert is allowed in settings panels of the project being opened.

**pval**

On return, this parameter contains the address of a pointer to the new project.

**Returns** `S_OK` if this method call succeeded or an appropriate error if it failed.

**See Also** "ECodeWarriorConvertOption" on page 482

## Application

### *OpenProjectWithOptions*

---

“ECodeWarriorRevertPanelOption” on page 482

“ICodeWarriorProject” on page 630

---

## OpenProjectWithOptions

Use this method to open a project in the CodeWarrior IDE, by specifying the project file by full path and applying certain options.

```
virtual HRESULT OpenProjectWithOptions(
 BSTR filePath,
 VARIANT_BOOL fMakeVisible,
 ECodeWarriorConvertOption convertOption,
 ECodeWarriorRevertPanelOption revertOption,
 ECodeWarriorProjectOption projectOption,
 ICodeWarriorProject **pval) = 0;
```

**filePath**

The full path to the project file.

**fMakeVisible**

Set this parameter to true if this operation is to be visible to the user or false if the operation should be hidden from the user.

**convertOption**

A value in the range defined by the ECodeWarriorConvertOption enumeration, indicating how the CodeWarrior IDE should handle this project if the project is found to be a project created by an older version of the IDE.

**revertOption**

A value in the range defined by the enumeration ECodeWarriorRevertPanelOption, indicating whether revert is allowed in settings panels of the project being opened.

**projectOption**

A value in the range defined by the enumeration ECodeWarriorProjectOption, indicating whether to cache

subprojects when the project is opened.

pval

On return, this parameter contains the address of a pointer to the specified project.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ECodeWarriorConvertOption" on page 482

"ECodeWarriorRevertPanelOption" on page 482

"ECodeWarriorProjectOption" on page 483

"ICodeWarriorProject" on page 630

---

## OpenProjectByFileSpec

This method opens a project in the CodeWarrior IDE, by specifying the project file with a file specification.

```
virtual HRESULT OpenProjectByFileSpec(
 IFileSpec *fileSpec,
 VARIANT_BOOL fMakeVisible,
 ECodeWarriorConvertOption convertOption,
 ECodeWarriorRevertPanelOption revertOption,
 CodeWarriorProject **pval) = 0;
```

fileSpec

A pointer to the IFileSpec interface containing the file specification of the project file to open.

fMakeVisible

Set this parameter to true if this operation is to be visible or false if it is to be hidden from the user.

convertOption

A value in the range defined by the ECodeWarriorConvertOption enumeration, representing



## Freescal Semiconductor, Inc.

### Application

*OpenProjectByFileSpec*

---

how the CodeWarrior IDE should handle this project if the project is found to be a project created by an older version of the IDE.

revertOption

A value in the range defined by the enumeration  
ECodeWarriorRevertPanelOption, representing whether  
revert is allowed in settings panels of the project being opened.

pval

On return, this parameter contains the address of a pointer to  
the new project.

Returns S\_OK if this method call succeeded or an appropriate error if it  
failed.

See Also "IFileSpec" on page 599

"ECodeWarriorConvertOption" on page 482

"ECodeWarriorRevertPanelOption" on page 482

"ICodeWarriorProject" on page 630

## OpenProjectByFileSpecWithOptions

This method opens a project in the CodeWarrior IDE, by specifying  
the project file with a file specification and applying certain options.

```
virtual HRESULT OpenProjectByFileSpecWithOptions(
 IFileSpec *fileSpec,
 VARIANT_BOOL fMakeVisible,
 ECodeWarriorConvertOption convertOption,
 ECodeWarriorRevertPanelOption revertOption,
 ECodeWarriorProjectOption projectOption,
 ICodeWarriorProject **pval) = 0;
```

fileSpec

A pointer to the IFileSpec interface containing the file  
specification of the project file to open.

fMakeVisible

Set this parameter to true if this operation is to be visible or

## Application

*OpenProjectByFileSpecWithOptions*

---

false if it is to be hidden from the user.

`convertOption`

A value in the range defined by the `ECodeWarriorConvertOption` enumeration, representing how the CodeWarrior IDE should handle this project if the project is found to be a project created by an older version of the IDE.

`revertOption`

A value in the range defined by the enumeration `ECodeWarriorRevertPanelOption`, representing whether revert is allowed in settings panels of the project being opened.

`projectOption`

A value in the range defined by the enumeration `ECodeWarriorProjectOption`, indicating whether to cache subprojects when the project is opened.

`pval`

On return, this parameter contains the address of a pointer to the new project.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599

"ECodeWarriorConvertOption" on page 482

"ECodeWarriorRevertPanelOption" on page 482

"ECodeWarriorProjectOption" on page 483

"ICodeWarriorProject" on page 630

## OpenTextDocument

Use this method to open and optionally create a text document in the CodeWarrior IDE by specifying the full path.

```
virtual HRESULT OpenTextDocument (
 BSTR inPath,
 VARIANT_BOOL create,
 ICodeWarriorTextDocument **document) = 0;
```

**inPath**

The full path for the file to open.

**create**

Set this parameter to `true` if the CodeWarrior IDE should create the file if it does not exist or `false` otherwise.

**document**

On return, this parameter contains the address of a pointer to the specified document.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorTextDocument" on page 583

---

## OpenTextDocumentByFileSpec

Use this method to open and optionally create a text document in the CodeWarrior IDE specifying the file specification.

```
virtual HRESULT OpenTextDocumentByFileSpec (
 IFileSpec *fileSpec,
 VARIANT_BOOL create,
 ICodeWarriorTextDocument **document) = 0;
```

**fileSpec**

A pointer to the IFileSpec interface containing the file to

## Application

### *OpenUntitledTextDocument*

---

open.

create

Set this parameter to `true` if the CodeWarrior IDE should create the file if it does not exist or `false` if CodeWarrior should not create the file.

document

On return, this parameter contains the address of a pointer to the document specified.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599

"ICodeWarriorTextDocument" on page 583

---

## OpenUntitledTextDocument

Use this method to open a new text document window with no associated file in the CodeWarrior IDE.

```
virtual HRESULT OpenUntitledTextDocument(
 ICodeWarriorTextDocument **document) = 0;
```

document

On return, this parameter contains the address of a pointer to the new text document.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTextDocument" on page 583



---

## put\_AllowUserInteraction

Use this method to set whether or not the CodeWarrior IDE should allow user interaction.

```
virtual HRESULT put_AllowUserInteraction(
 VARIANT_BOOL pval) = 0;
```

pval

Set this parameter to `true` if the IDE should allow user interaction or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_Visible

Call this method to set the visible state of the CodeWarrior application.

```
virtual HRESULT put_Visible(
 VARIANT_BOOL val) = 0;
```

val

Set this parameter to `true` if the application is visible or `false` if the application is not visible.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## QueueDeferredAction

Use this method to queue a deferred action in the CodeWarrior IDE.

```
virtual HRESULT QueueDeferredAction(
 IUnknown *action) = 0;
```

## Application

### Quit

---

action

A pointer to the IUnknown interface containing the deferred action. The deferred action must be previously registered with the IDE.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Commands API Overview" on page 493

---

## Quit

This method closes the CodeWarrior IDE, applying a save option in the process.

```
virtual HRESULT Quit(
 ECodeWarriorSaveOption val) = 0;
```

val

A value in the range defined by the enumeration ECodeWarriorSaveOption, indicating whether to save all the files open in the IDE, ask the user whether to save, or save none of the files.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorSaveOption" on page 483

---

## RemoveCreatableItem

Use this method to remove a creatable item to the CodeWarrior IDE. Creatable items encapsulate the items visible in the **New** window.

```
virtual HRESULT RemoveCreatableItem(
 IUnknown *item) = 0;
```

---



item

A pointer to the IUnknown interface, containing the creatable item to be removed from the CodeWarrior IDE.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Creatable Items" on page 523

---

## RemoveNamedPluginData

Use this method to remove the plug-in data for a given plug-in.

```
virtual HRESULT RemoveNamedPluginData(
 BSTR resourceName)
```

resourceName

The name of the plug-in resource you want to modify.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## RemoveUserTree

This method removes a specified user tree.

```
virtual HRESULT RemoveUserTree(
 ICodeWarriorUserTree *pval) = 0;
```

pval

A pointer to the user tree to remove.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorUserTree" on page 436



## Freescal Semiconductor, Inc.

### Application

*SetNamedPluginData*

---

### SetNamedPluginData

---

Use this method to set the plug-in data for a given plug-in.

```
virtual HRESULT SetNamedPluginData(
 BSTR resourceName,
 IStream *pluginData) = 0;
```

resourceName

The name of the plug-in resource you want to modify.

pluginData

A pointer to the IStream interface containing the data to load into the plug-in.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SetSetting

Use this method to set the value of a given IDE preference setting.

```
virtual HRESULT SetSetting(
 BSTR settingsName,
 VARIANT pval) = 0;
```

settingsName

The name of the setting value to modify.

pval

The new value of the specified setting.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorAppEvents

This interface lets plug-ins respond to certain events in the CodeWarrior IDE. The methods outlined below are called by the IDE when a corresponding event occurs.

Inherited Interfaces

- IUnknown

Methods

This interface provides the following methods:

|                        |                |
|------------------------|----------------|
| DataModelCreated       | ProjectVisible |
| DataModelLoaded        | QueryQuit      |
| Application Data Types | Quit           |
| ProjectOpened          | Startup        |

DataModelCreated

The CodeWarrior IDE calls this method when it creates a data model.

```
virtual HRESULT DataModelCreated(
 IUnknown *dataModel,
 VARIANT_BOOL fFromStorage) = 0;
```

dataModel

When the IDE calls this method, this parameter contains the created data model.

fFromStorage

When the IDE calls this method, this parameter contains `true` if the data model is from storage or `false` if the data model is not from storage.

Returns    S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Application

#### *DataModelLoaded*

---

### DataModelLoaded

The CodeWarrior IDE calls this method when it loads a data model.

```
virtual HRESULT DataModelLoaded(
 IUnknown *dataModel) = 0;
```

*dataModel*

When the IDE calls this method, this parameter contains the loaded data model.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### ProjectOpened

The CodeWarrior IDE calls this method when it opens a project.

```
virtual HRESULT ProjectOpened(
 ICodeWarriorProject *project,
 VARIANT_BOOL fVisible) = 0;
```

*project*

When the IDE calls this method, this parameter contains the opened project.

*fVisible*

When the IDE calls this method, this parameter contains `true` if the project is visible, or `false` if the project is not visible.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630



---

## ProjectVisible

The CodeWarrior IDE calls this method when it makes an invisible project visible.

```
virtual HRESULT ProjectVisible(
 ICodeWarriorProject *project) = 0;
```

project

When the IDE calls this method, this parameter contains the visible project.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProject" on page 630

---

## QueryQuit

The CodeWarrior IDE calls this method when it wants to quit. Your plug-in should determine whether it is safe to quit, and return a success or failure code accordingly.

```
virtual HRESULT QueryQuit(void) = 0;
```

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## Quit

The CodeWarrior IDE calls this method when it quits.

```
virtual HRESULT Quit(void) = 0;
```

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



**Application**

*Startup*

---

**Startup**

The CodeWarrior IDE calls this method when it starts.

```
virtual HRESULT Startup(void) = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## ICodeWarriorCompare

This interface provides methods for comparing files and folders.

### Inherited Interfaces

- IUnknown

### Methods

This interface provides the following methods:

|                        |                |
|------------------------|----------------|
| CompareFiles           | CompareFolders |
| CompareFilesByFileSpec |                |

### CompareFiles

This method compares the contents of two files, optionally comparing case and white space. The results of the comparison appear in the File Compare Results window in the IDE.

```
virtual HRESULT CompareFiles(
 BSTR srcFile,
 BSTR destFile,
 VARIANT_BOOL ignoreCase,
 VARIANT_BOOL ignoreSpace) = 0;

srcFile
 The first file to compare.

destFile
 The second file to compare.

ignoreCase
 true if you want to ignore case in this comparison or false if
 you want to compare case.

ignoreSpace
 true if you want to ignore white space or false if you want to
```

## Application

### *CompareFilesByFileSpec*

---

compare white space.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## CompareFilesByFileSpec

This method compares the contents of two files, as indicated by file specifications, optionally comparing case and white space. The results of the comparison appear in the File Compare Results window in the IDE.

```
virtual HRESULT CompareFilesByFileSpec(
 IFileSpec *srcFile,
 IFileSpec *destFile,
 VARIANT_BOOL ignoreCase,
 VARIANT_BOOL ignoreSpace) = 0;
```

srcFile

A pointer to the first file to compare.

destFile

A pointer to the second file to compare

ignoreCase

true if you want to ignore case in this comparison or false if you want to compare case.

ignoreSpace

true if you want to ignore white space or false if you want to compare white space.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## CompareFolders

This method compares the contents of two folders, optionally comparing case, white space, files that exist in only one folder or the other, and the contents of text files. The results of the comparison appear in the Folder Compare Results window in the IDE.

```
virtual HRESULT CompareFolders(
 BSTR srcFolder,
 BSTR destFolder,
 VARIANT_BOOL inIgnoreCase,
 VARIANT_BOOL inIgnoreSpace,
 VARIANT_BOOL showDifferentFiles,
 VARIANT_BOOL compareTextFileContents) = 0;
```

srcFile

The first folder to compare.

destFile

The second folder to compare

ignoreCase

true if you want to ignore case in this comparison or false if you want to compare case.

ignoreSpace

true if you want to ignore white space or false if you want to compare white space.

showDifferentFiles

true to show files that exist in one folder but not the other or false to ignore such files.

compareTextFileContents

true to compare the contents of text files within the two folders or false to ignore the contents of text files.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

### Application Data Types

The following data types are used with the Application API:

- `ECodeWarriorConvertOption`
- `ECodeWarriorRevertPanelOption`
- `ECodeWarriorProjectOption`
- `ECodeWarriorSaveOption`
- `ECodeWarriorVCSInteractionOption`
- Standard Folder Names

#### ECodeWarriorConvertOption

This enumeration is used to describe how to treat a project being opened via the `OpenProject` and `OpenProjectByFileSpec` methods of `ICodeWarriorApp`.

Table 10.1 ECodeWarriorConvertOption enumeration

| Constant                   | Description                                         |
|----------------------------|-----------------------------------------------------|
| <code>kCWConvertYes</code> | Convert the project without user interaction.       |
| <code>kCWConvertNo</code>  | Do not convert the project.                         |
| <code>kCWConvertAsk</code> | Ask the user whether to convert the project or not. |

#### ECodeWarriorRevertPanelOption

This enumeration is used to describe whether revert is allowed in the settings panels of a project being opened via the `OpenProject` and `OpenProjectByFileSpec` methods of `ICodeWarriorApp`.

**Table 10.2 ECodeWarriorRevertPanelOption enumeration**

| Constant            | Description                               |
|---------------------|-------------------------------------------|
| kCWDonotRevertPanel | Do not allow the user to revert settings. |
| kCWAllowPanelRevert | Allow the user to revert settings.        |

## ECodeWarriorProjectOption

This enumeration is used to describe whether revert is allowed in the settings panels of a project being opened via the `OpenProjectWithOptions` and `OpenProjectByFileSpecWithOptions` methods of `ICodeWarriorApp`.

**Table 10.3 ECodeWarriorProjectOption enumeration**

| Constant                    | Description                                                 |
|-----------------------------|-------------------------------------------------------------|
| kCWNone                     | Apply the default settings when opening a project.          |
| kCWDisableSubProjectCaching | Disable the caching of sub projects when opening a project. |

## ECodeWarriorSaveOption

This enumeration describes settings for saving files when making the CodeWarrior IDE Quit. It is used by the `Quit` method of `ICodeWarriorApp`.

**Table 10.4 ECodeWarriorSaveOption enumeration**

| Constant    | Description                                                         |
|-------------|---------------------------------------------------------------------|
| kCWAskSave  | Asks the user whether to save all the files before closing the IDE. |
| kCWSaveAll  | Saves all the files before closing the IDE.                         |
| kCWSaveNone | Closes the IDE without saving any files.                            |

## ECodeWarriorVCSInteractionOption

This enumeration is used to describe how user interaction should be handled when a version control operation is performed by the CodeWarrior IDE. It is used in the `AttemptModify` method of `ICodeWarriorApp`.

**Table 10.5 ECodeWarriorVCSInteraction enumeration**

| Constant      | Description                                                                                                |
|---------------|------------------------------------------------------------------------------------------------------------|
| kCWAsk        | Ask the user whether to make version control changes                                                       |
| kCWDoNothing  | Do not make version control changes                                                                        |
| kCWUseDefault | Use the default version control behavior when making changes - useful when working with invisible projects |

## Standard Folder Names

These folder names represent standard folders within the CodeWarrior folder and are for use with the `FindLogicalFolder` method of `ICodeWarriorApp`.



Table 10.6 Standard folder name enumeration

| Constant                  | Description                                               |
|---------------------------|-----------------------------------------------------------|
| kMWStationeryFolder       | The folder where CodeWarrior project stationery is stored |
| kMWRadStationeryFolder    | The folder where CodeWarrior RAD stationery is stored     |
| kMWCompilerFolder         | The folder where the CodeWarrior IDE resides              |
| kMWPluginsFolder          | The folder where CodeWarrior plug-ins reside              |
| kLocalizedResourcesFolder | The folder where localized resources reside               |



# Freescal Semiconductor, Inc.

## Application

*Application Data Types*

---

# Collections

---

This chapter describes how to use the Collections API to create and manage Collections in the CodeWarrior IDE.

This chapter contains the following sections:

- Collections API Overview
- Using the Collections API
- Collections API Reference

## Collections API Overview

The Collections API is a set of interfaces that allows a plug-in to create and manipulate collections of IDE-related objects. A collection is a class that holds a list of similar items. The CodeWarrior IDE uses collections to hold lists of IDE-related objects.

## Using the Collections API

Most of the collections returned by the IDE are read-only. Calling **Add** or **Remove** methods on them returns `E_FAIL`. The **Add** and **Remove** methods are provided for collections that users create to pass into the IDE.

## Collections API Reference

The Collections API contains numerous interfaces for working with numerous types of data. However, all collections in this API implement the same methods and behaviors. This section describes a single generic collection that applies to all CodeWarrior collections. CodeWarrior collections are provided for each of the data types in Table 11.1.

**Table 11.1 Data Types with Associated Collections**

|                               |
|-------------------------------|
| BSTR                          |
| ICodeWarriorAccessPath        |
| ICodeWarriorBaseClass         |
| ICodeWarriorClass             |
| ICodeWarriorComponent         |
| ICodeWarriorComponentEvent    |
| ICodeWarriorComponentEventSet |
| ICodeWarriorComponentProperty |
| ICodeWarriorCreatableItem     |
| ICodeWarriorDataMember        |
| ICodeWarriorDesign            |
| ICodeWarriorDocument          |
| ICodeWarriorMessage           |
| ICodeWarriorMethod            |
| ICodeWarriorProject           |
| ICodeWarriorProjectFile       |
| ICodeWarriorSubTarget         |
| ICodeWarriorSymbol            |
| ICodeWarriorTarget            |
| ICodeWarriorTargetFile        |
| ICodeWarriorUserTree          |
| IFileSpec                     |

CodeWarrior Collections

This is a generic description of all collection interfaces provided by the CodeWarrior COM API. This description applies to all CodeWarrior collection interfaces.

As a rule, the name of each CodeWarrior collection interface is constructed by taking the name of the data interface for the collection (for example, ICodeWarriorAccessPath), and appending the word “Collection” to the end of it (as in ICodeWarriorAccessPathCollection). All of the collection data types are listed in Table 11.1 on page 488.

**NOTE** The exception to this rule is IBSTRCollection, where “I” is prepended to the data type name.

Inherited Interfaces

- IDispatch
- IUnknown

Methods

Every CodeWarrior collection interface provides these methods:

|              |              |
|--------------|--------------|
| Add          | get_ReadOnly |
| get_Count    | Item         |
| get__NewEnum | Remove       |

Add

This method appends an item to a collection. The get\_Count method reflects whether or not this method can be used.

```
virtual HRESULT Add(
 VarType *var) = 0;

var
```

One of the collection data types listed in Table 11.1 on page 488.



Collections

get\_Count

|         |                                                                               |
|---------|-------------------------------------------------------------------------------|
| NOTE    | For the BSTR collection data type, use a BSTR instead of a pointer to a BSTR. |
| Returns | S_OK if this method call succeeded or an appropriate error if it failed.      |

get\_Count

This method gets the number of items in a collection.

```
virtual HRESULT get_Count(
 long *pval) = 0;
```

pval

On return, this parameter contains a pointer to the number of items contained in this collection.

|         |                                                                          |
|---------|--------------------------------------------------------------------------|
| Returns | S_OK if this method call succeeded or an appropriate error if it failed. |
|---------|--------------------------------------------------------------------------|

get\_\_NewEnum

This method gets an enumerator for a collection.

```
virtual HRESULT get__NewEnum(
 IDispatch **pval) = 0;
```

pval

Upon return, this parameter the address of a pointer to an enumerator for this collection.

|         |                                                                          |
|---------|--------------------------------------------------------------------------|
| Returns | S_OK if this method call succeeded or an appropriate error if it failed. |
|---------|--------------------------------------------------------------------------|

get\_ReadOnly

This method gets the read-only status of a collection. The result of this method determines whether the **Add** and **Remove** methods of a collections may be used.

**NOTE** All CodeWarrior-generated collections are read only. Only user-generated collections can be modified.

```
virtual HRESULT get_ReadOnly(
 VARIANT_BOOL *bool) = 0;
```

bool

On return, this parameter is set to true if the collection is read-only or false if the collection is modifiable.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

Item

This method gets an item from a collection.

```
virtual HRESULT Item(
 long index,
 VarType *var) = 0;
```

index

The index of the item you want.

var

One of the collection data types listed in Table 11.1 on page 488. On return, this parameter contains a pointer to the requested item.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



Collections

Remove

---

Remove

This method removes an item from a collection. The `get_Count` method reflects whether or not this method can be used.

```
virtual HRESULT Remove(
 VarType *var) = 0;
```

var

One of the collection data types listed in Table 11.1 on page 488.

---

**NOTE** For the `BSTR` collection data type, use a `BSTR` instead of a pointer to a `BSTR`.

---

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

# Commands

---

This chapter shows how to use the Command API to allow plug-ins to intercept commands from the CodeWarrior IDE.

This chapter contains the following sections:

- Commands API Overview
- Using the Commands API
- Commands API Reference

## Commands API Overview

The Command API allows plug-ins to respond to commands from the CodeWarrior IDE.

The commands interface allows users to define command groups and assign certain command features at initialization and during run time.

All interfaces defined in the commands API are pure abstract base classes. All interfaces defined in this section inherit from `IUnknown`.

Command groups are created only when the IDE launches. Run time manipulation of commands is achieved via `ICodeWarriorCommandHandler::GetCommandStatus()`.

## Commands

Using the Commands API

## Using the Commands API

Creating a command group to intercept user events within the IDE typically involves several steps and can involve several interfaces. However, some of the most common uses of command groups are for creating menus and menu items at IDE launch and enabling menu items to perform desired actions. Typical steps for creating a command group, such as menus, involve:

- Creating a Command Group
- Assigning a Command Handler
- Registering a Command
- Displaying a Command Group

The Commands API also uses data types, as described in this section:

- Commands Data Types

### Creating a Command Group

All commands are created with the `ICodeWarriorCommandRegistry` interface. However, you must first provide a service provider interface (`IServiceProvider`) before you can create a command group. You also must know the GUID of the CodeWarrior Service ID as well as the Interface ID. Listing 12.1 demonstrates how to use the `QueryService()` method when creating a new command group.

#### Listing 12.1 Example Code - Creating a Command Group

```
ICodeWarriorCommandRegistry *cmdRegistry;

// Here we query the service provider to get the command registry
// interface.

servProv->QueryService(
 SID_SCodeWarriorCommandRegistry,
 IID_ICodeWarriorCommandRegistry, &cmdRegistry);

// If the command is supported by the IDE then we can create
// our command info and register our commands
```

---

```

if(cmdRegistry){

 // Assign command handlers if we want to here

 // Create the command group here, which will be used for our
 // menu with the title "My Menu"
 BSTR bstr = SysAllocString(OLESTR("My Menu"));
 cmdRegistry->CreateNewCommandGroup(kToolbarTestPluginID,
 cmdGroup_TestPlugin, bstr, cmdGroup_Nothing);

 // Get toolbar icon info if any

 // Register Commands

 // Release
}

```

---

### Assigning a Command Handler

Assigning a command handler to a command group item allows that item to respond to events handled from a particular command. We can create a reference to an interface by creating a command handler and then registering it in the `commandHandler` field of the `SRegisterCommandInfo` structure. Listing 12.2 shows how to assign a command handler that will be used for a menu item, which is demonstrated in Listing 12.3.

#### Listing 12.2 Example Code - Assigning a Command Handler

---

```

ICodeWarriorWindowManager *windowMgr = NULL;

servProv->QueryService(SID_SCodeWarriorWindowManager,
 IID_ICodeWarriorWindowManager, &windowMgr);

/* Handle commands for creating a new window */
ICodeWarriorCommandHandler *cmdHandler = new
 ExampleCommandHandler(windowMgr);

```

---

### Registering a Command

Once you have a command group, you can register individual commands. All commands should be registered with the

## Commands

Using the Commands API

`RegisterCommand()` method from the `ICodeWarriorCommandHandler` interface. The `commandHandler` field of the `SRegisterCommandInfo` structure contains a pointer to the interface whose routines you want to access for this object. Listing 12.2 shows how to assign a command handler for a command group item.

In order to respond to commands sent by the IDE, your plug-in must register a command handler for the particular command (`commandID`) you want the plug-in group (`commandGroupID`) to intercept within the main plug-in (`pluginID`). An example of using the command registry is shown in Listing 12.3.

### Listing 12.3 Example Code - Registering a Command

```
// A new menu item identified by cmd_NewPluginWindow will be
// added to the command group specified by cmdGroup_TestPlugin
// You will need to make a similar call for each item in you
// menu.

SysReAllocString(&bstr, OLESTR("My New Window"));
SRegisterCommandInfo cmdInfo;
cmdInfo.pluginID = kToolbarTestPluginID;
cmdInfo.commandID = cmd_NewPluginWindow;
cmdInfo.commandGroupID = cmdGroup_TestPlugin;
cmdInfo.commandName = bstr;
cmdInfo.toolbarIcon = tbIconInfo;
cmdInfo.visibleInMenu = true;
cmdInfo.itemType = CWCommandItemType_Command;
cmdInfo.extraInfo.commandHandler = cmdHandler;
cmdRegistry->RegisterCommand(cmdInfo, cmd_Nothing);

// Always call Release when you are through with an interface!
cmdRegistry->Release();
}
```

### Displaying a Command Group

The last step in establishing your command group is displaying it within the IDE so the user can access it. Listing 12.4 demonstrates how to display a command group as a menu on the menu bar at IDE launch.



## Freescale Semiconductor, Inc.

### Commands

*Using the Commands API*

---

#### **Listing 12.4    Example Code - Displaying a Command Group**

---

```
// Show the command group in a menu
// We will do this by providing a menu manager interface to
// manage our commands

ICodeWarriorMenuManager *menuMgr;
servProv->QueryService(SID_SCodeWarriorMenuManager,
 IID_ICodeWarriorMenuManager, &menuMgr);
if (menuMgr){
 menuMgr->ShowCommandGroupMenu(kToolbarTestPluginID,
 cmdGroup_TestPlugin, true);
 menuMgr->Release();
}
```

---



**Commands**

*Commands API Reference*

---

## **Commands API Reference**

This section describes the functions contained in the following interfaces:

- ICodeWarriorCommandHandler
- ICodeWarriorCommandRegistry
- ICodeWarriorDeferredAction

The following data types are used with these interfaces:

- Command status
- SRegisterCommandGroup

## ICodeWarriorCommandHandler

This interface allows you to intercept and handle built-in commands as well as custom commands registered by your plug-in.

### Inherited Interfaces

- IDispatch
- IUnknown

### Methods

This interface provides the following methods:

|                |                  |
|----------------|------------------|
| ExecuteCommand | GetCommandStatus |
|----------------|------------------|

### ExecuteCommand

The IDE calls this method when a command needs to be executed by your plug-in. Usually this is the result of an associated menu or toolbar button being selected by the user.

```
virtual HRESULT ExecuteCommand(
 CWCommandID inCommandNumber,
 ICodeWarriorCommandHandler *inDefaultHandler
) = 0;
```

**inCommandNumber**

The ID of a command handler to be executed. Use a built-in command handler or an external command handler registered by the plug-in via the RegisterCommand method. To handle a built-in command, test this parameter to see if it is equal to the appropriate predefined constant in CodeWarriorCommandNumbers.h.

**inDefaultHandler**

The default command handler for this command. If the plug-in does not handle this command, it should call this handler to pass control on to that handler.

## Commands

### *GetCommandStatus*

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorCommandHandler” on page 499  
 “RegisterCommand” on page 503

---

## GetCommandStatus

This method is called to update menu items at run-time.

```
virtual HRESULT GetCommandStatus(
 CWCommandID inCommandNumber,
 BOOL &outEnabled, SH
 ORT &outCheckedState,
 BSTR &outNewName,
 ICodeWarriorCommandHandler *inDefaultHandler
) = 0;
```

inCommandNumber

The ID of the command handler to be executed. This may be a built-in command handler or a command handler that was registered by the plug-in.

outEnabled

The current enabled state of this command. If the command is enabled, this parameter returns true. Otherwise, it returns false. The visual element for this command in the CodeWarrior IDE is drawn accordingly.

outCheckedState

The current checked state of the menu item for this command. If the item is checked, this parameter returns true. Otherwise, it returns false.

outNewName

The current name of this command, as it is displayed in the IDE menus or the toolbar tool tips. Set this item to NULL for no change of the current menu item title.



`inDefaultHandler`

The default command handler for this command. If the plug-in does not handle this command, it should call this handler to pass control on to that handler.

Returns    `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also    “Enumeration for Command Status” on page 505



**ICodeWarriorCommandRegistry**

This interface allows you to create and dispose of your own custom commands in the CodeWarrior IDE.

**Inherited Interfaces**

- IUnknown

**Methods**

This interface provides the following methods:

|                       |                 |
|-----------------------|-----------------|
| CreateNewCommandGroup | RegisterCommand |
|-----------------------|-----------------|

CreateNewCommandGroup

This method creates a new command group. A command group is a menu or sub menu. The constant cmdGroup\_Nothing is used to define a new menu. You will need to call ICodeWarriorMenuManager::ShowCommandGroupMenu() to display the menu.

```
virtual HRESULT CreateNewCommandGroup(
 const CWPluginID inPluginID,
 CWCommandGroupID inGroupID,
 BSTR inGroupName,
 CWCommandGroupID inParentGroupID) = 0;
```

inPluginID

The ID for the plug-in. Usually this is the class ID of the main class of your plug-in.

inGroupID

The ID for the group.

inGroupName

The name of the group you want to create (that is, the title of the menu).

`inParentGroupID`

The ID of the parent group.

Returns `HRESULT`

See Also “`ICodeWarriorMenuManager`” on page 612

---

## RegisterCommand

This method registers an external command with the CodeWarrior IDE.

```
virtual HRESULT RegisterCommand(
 const SRegisterCommandInfo &inCmdInfo,
 LONG inInsertBeforeCommandID) = 0;
```

`SRegisterCommandGroup`

This data structure must be filled in with the appropriate IDs, names, and types to be registered as a command.

`inInsertBeforeCommandID`

Specifies the insertion location where your command appears. This is typically `cmd_Nothing`, which is defined in `CodeWarriorCommandNumbers.h`.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also “`SRegisterCommandGroup`” on page 506



ICodeWarriorDeferredAction

CodeWarriorDeferredAction.h defines this interface. The method in this interface can be posted on an application-level queue for execution after the handling of the current event.

Inherited Interfaces

- IUnknown

Methods

This interface provides the following methods:

|         |  |
|---------|--|
| Execute |  |
|---------|--|

Execute

This method executes a command.

```
virtual HRESULT Execute(void) = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Commands Data Types

### Command status

The following enumerations in Table 12.1 are used as constants to return the `outCheckedState` parameter of the `ICodeWarriorCommandHandler::GetCommandStatus()` method:

Table 12.1 Enumeration for Command Status

| Constant                                  | Description                          |
|-------------------------------------------|--------------------------------------|
| <code>CWCommand_CheckMark_NoChange</code> | The specified menu has not changed.  |
| <code>CWCommand_CheckMark_Clear</code>    | The specified menu is to be cleared. |
| <code>CWCommand_CheckMark_Set</code>      | The specified menu is set.           |

### Menu Commands

The following enumeration in Table 12.2 are used to fill out the `itemType` field of the `SRegisterCommandGroup` structure when using the `RegisterCommand()` method:

Table 12.2 Enumeration for menu commands

| Constant                                 | Description                       |
|------------------------------------------|-----------------------------------|
| <code>CWCommandItemType_Command</code>   | The item is a command             |
| <code>CWCommandItemType_Separator</code> | The item is a separator on a menu |
| <code>CWCommandItemType_SubMenu</code>   | The item is a sub menu            |

## Commands

### Commands Data Types

---

#### SRegisterCommandGroup

The SRegisterCommandGroup structure is used to register a command and set its properties as a parameter of ICodeWarriorCommandRegistry::RegisterCommand(). See Listing 12.3 for an example of registering a command with SRegisterCommandGroup. This structure is defined as follows:

```
struct SRegisterCommandInfo {
 CWPluginID pluginID;
 CWCommandID commandID;
 CWCommandGroupID commandGroupID;
 BSTR commandName;
 CWToolBarIconInfo toolbarIcon;
 BOOL visibleInMenu;
 long itemType;
 union {
 IUnknown *commandHandler;
 CWCommandGroupID subGroupID;
 } extraInfo;
};
```

pluginID

The plug-in ID for the plug-in to which the command belongs.

commandID

The ID of the command group provides the location of the command within a command group specified by commandGroupID. The commandID values must be in the range of 10,000 to 10,999.

commandGroupID

The command group (menu) to which the item belongs.

commandName

The name of the command to be displayed, as specified by commandID.



## Freescale Semiconductor, Inc.

### Commands

#### Commands Data Types

---

##### `toolbarIcon`

A toolbar icon reference for the command you are registering. Pass `toolbarIcon_None` for no icon information. This is a platform specific data type.

On the Mac OS, the `CWToolbarIconInfo` is a `Handle` to the item. Mac users need to provide Mac Toolbox calls to get this resource, which are small icon sweet resources (`ics#`). See `UseResFile()` and `GetIconSuite()` in the Mac Toolbox for more information.

Windows icons must be registered first before they are used. On Windows, icons are identified by an index into bitmaps that are registered via `ICodeWarriorToolBarRegistry::RegisterToolbarIcons()`.

##### `visibleInMenu`

Set this item to `true` to display the menu item or `false` to hide the menu item from the user.

##### `itemType`

Specify the type of item in the command group. Use the enumerations specified in Table 12.2.

##### `commandHandler`

A pointer to the `IUnknown` interface whose methods you want to use. Set this item to `NULL` if you do not need any additional interface routines. Otherwise, you will need to pass a pointer to the interface whose methods you want to access from the command specified in `commandID`.

##### `subGroupID`

A sub menu group ID, if one exists.



# Freescale Semiconductor, Inc.

## Commands

*Commands Data Types*

---

# Components

---

This chapter shows how to use the Components API to add your own components in the CodeWarrior IDE.

This chapter contains the following sections:

- Components API Overview
- Components API Reference

## Components API Overview

The Components API is a set of interfaces that allows a plug-in to create and manipulate components in the CodeWarrior IDE.

## Components API Reference

This section describes the functions contained in the following interfaces:

- ICodeWarriorComponent
- ICodeWarriorComponentEvent
- ICodeWarriorComponentEventSet
- ICodeWarriorComponentProperty

ICodeWarriorComponent

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface provides the following methods:

|                              |                |
|------------------------------|----------------|
| get_CanHaveMultipleEventSets | get_EventSets  |
| get_Class                    | get_Methods    |
| get_DefaultEvent             | get_Properties |
| get_EventConnectionsEnabled  |                |

get\_CanHaveMultipleEventSets

This method gets whether this component can have multiple event sets.

```
virtual HRESULT get_CanHaveMultipleEventSets(
 BOOL *pval) = 0;
```

pval

On return, this parameter is set to `true` if the component can have multiple event sets or `false` if the component cannot have multiple event sets.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

## get\_Class

The IDE calls this method to obtain the class of this component.

```
virtual HRESULT get_Class(
 ICodeWarriorClass **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the class of this component.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorClass" on page 668

---

## get\_DefaultEvent

This method gets the default event for this component.

```
virtual HRESULT get_DefaultEvent(
 ICodeWarriorComponentEvent **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the default event for this component.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorComponentEvent" on page 514

## Components

*get\_EventConnectionsEnabled*

---

### get\_EventConnectionsEnabled

This method gets whether event connections are enabled for this component.

```
virtual HRESULT get_EventConnectionsEnabled(
 BOOL *pval) = 0;
```

pval

On return, this parameter is set to `true` if event connections are enabled for this component or `false` if event connections are not enabled.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_EventSets

This method gets the events sets for this component.

```
virtual HRESULT get_EventSets(
 ICodeWarriorComponentEventSetCollection
 **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all the methods for this component.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---



get\_Methods

This method gets all of the methods for this component.

```
virtual HRESULT get_Methods(
 ICodeWarriorMethodCollection **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all the methods for this component.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

get\_Properties

This method gets all of the properties for this component.

```
virtual HRESULT get_Properties(
 ICodeWarriorComponentPropertyCollection
 **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a collection of all the properties for this component.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

ICodeWarriorComponentEvent

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface provides the following methods:

|                      |            |
|----------------------|------------|
| GetDefaultMethodName | get_Method |
| get_EventSet         | get_Name   |

GetDefaultMethodName

This method gets the default method name for this component.

```
virtual HRESULT GetDefaultMethodName(
 IUnknown *modelobject,
 BSTR *pdefname) = 0;
```

modelobject

A pointer to the current object.

pdefname

On return, this parameter contains the default method name for this component.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_EventSet

This method gets the component event set that this component event belongs to.

```
virtual HRESULT get_EventSet(
 ICodeWarriorComponentEventSet **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the event set that this component event belongs to.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorComponentEventSet" on page 517

---

## get\_Method

This method gets the method for this component event.

```
virtual HRESULT get_Method(
 ICodeWarriorMethod **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the method for this component event.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorMethod" on page 677



# Freescale Semiconductor, Inc.

## Components

*get\_Name*

---

### get\_Name

This method gets the name of this component event.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

The name of this component event.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorComponentEventSet

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface provides the following methods:

|            |                  |
|------------|------------------|
| get_Class  | get_EventSetName |
| get_Events |                  |

get\_Class

This method gets the class for this component event set.

```

virtual HRESULT get_Class(
 ICodeWarriorClass **pval) = 0;

pval

```

On return, this parameter contains the address of a pointer to the class for this component event set.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorClass" on page 668

get\_Events

This method gets a collection of the events in this component event set.

```

virtual HRESULT get_Events(
 ICodeWarriorComponentEventCollection **pval
) = 0;

```



**Components**

*get\_EventSetName*

---

pval

On return, this parameter contains the address of a pointer to the collection of events for this component event set.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

---

**get\_EventSetName**

This method gets the name of this component event set.

```
virtual HRESULT get_EventSetName(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the name of this component event set.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

ICodeWarriorComponentProperty

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface provides the following methods:

|            |            |
|------------|------------|
| get_Getter | get_Setter |
| get_Name   | get_Type   |

get\_Getter

This method gets the component method that is responsible for getting this component property.

```
virtual HRESULT get_Getter(
 ICodeWarriorMethod **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the component method that is responsible for getting this component property.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorMethod" on page 677



# Freescal Semiconductor, Inc.

## Components

get\_Name

---

### get\_Name

This method gets the name of this component property.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the name of this component property.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_Setter

This method gets the component method that is responsible for setting this component property.

```
virtual HRESULT get_Setter(
 ICodeWarriorMethod **pval) = 0;
```

pval

Supply the address of a pointer to the interface. Upon return it contains the component method that is responsible for setting this component property.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorMethod" on page 677



get\_Type

This method gets the type of this component property.

```
virtual HRESULT get_Type(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the type of this component property.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescale Semiconductor, Inc.

## Components

*get\_Type*

---

# Creatable Items

---

This chapter shows how to use the Creatable Items API to implement your own creatable items for use in the CodeWarrior IDE.

## Creatable Items API Reference

This section describes the functions contained in the following interfaces:

- `ICodeWarriorCreatableItem`
- `ICodeWarriorCreateFileItem`
- `ICodeWarriorCreateObjectItem`
- `ICodeWarriorCreateProjectItem`

The following data types are used with these interfaces:

- Built-in Icon Index Values
- Creatable Item Category Constants
- `ECreateProjectType`

## ICodeWarriorCreatableItem

This interface defines an item that fits into one of the panes of creatable items visible in the **New** window in the CodeWarrior IDE. Creatable items represent stationery or wizards that the user may use to start a project, file or some other item in the CodeWarrior IDE.

### Inherited Interfaces

- IUnknown

### Methods

This interface provides the following methods:

|                |               |
|----------------|---------------|
| GetCategory    | GetIcon       |
| GetDisplayName | InvokesWizard |

## GetCategory

The CodeWarrior IDE calls this method to get the category of this creatable item, which determines where the creatable item is displayed in the New window.

```

virtual HRESULT GetCategory(
 BSTR *category) = 0;

```

category

Set this parameter to one of the creatable item category constants already defined for you, representing the category of this creatable item.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Creatable Item Category Constants” on page 538



---

## GetDisplayName

The CodeWarrior IDE calls this method to get the display name for this creatable item. The string you supply is used to display the name of this creatable item in the **New** window.

```
virtual HRESULT GetDisplayName(
 BSTR *displayName) = 0;
```

displayName

Set this string to the name of this creatable item as it should appear in CodeWarrior windows and dialog boxes.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## GetIcon

The CodeWarrior IDE calls this method to get the icon for this creatable item. The icon you supply is displayed next to this creatable item in the New window.

```
virtual HRESULT GetIcon(
 IUnknown *iconList,
 int *index) = 0;
```

iconList

A pointer to the IUnknown interface containing the icon list that holds the icon for this creatable item.

index

Set this integer to one of the predefined icon index values or, if your creatable item uses a custom icon, supply the index in the icon list of the custom icon.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Creatable Items**  
*InvokesWizard*

---

See Also    “Built-in Icon Index Values” on page 538

---

**InvokesWizard**

The CodeWarrior IDE calls this method to discover whether this creatable item invokes a wizard or not. If it does invoke a wizard, the IDE appends the localized string for “Wizard” to the display name of the creatable item (for example, “Java Applet Wizard”).

```
virtual HRESULT InvokesWizard(void) = 0;
```

Returns    S\_OK if creating the item invokes a wizard, or S\_FALSE if creating the item does not invoke a wizard.

### ICodeWarriorCreateFileItem

This interface is used for a creatable file item that is displayed in the **File** pane of creatable items visible in the **New** window in the CodeWarrior IDE.

#### Inherited Interfaces

- ICodeWarriorCreatableItem
- IUnknown

#### Methods

This interface provides the following methods:

|                       |                    |
|-----------------------|--------------------|
| CanAddFileToProject   | CreateAndAddFile   |
| CanCreateUntitledFile | CreateUntitledFile |

#### CanAddFileToProject

The CodeWarrior IDE calls this method to determine if this creatable item is able to add a file to a project.

```
virtual HRESULT CanAddFileToProject(void) = 0;
```

Returns S\_OK if this creatable item is able to add files to a project or an appropriate error if this creatable item is not able to add files to a project.

#### CanCreateUntitledFile

The CodeWarrior IDE calls this method to determine if this creatable item is able to create an untitled file.

```
virtual HRESULT CanCreateUntitledFile(void) = 0;
```

## Creable Items

### CreateAndAddFile

---

**Returns** S\_OK if this creable item is able to create an untitled file or an appropriate error if this creable item is not able to create an untitled file.

---

### CreateAndAddFile

This method is called by the CodeWarrior IDE when the user instructs it to create a new file with this creable item selected, and the **Add to project** option is checked.

```
virtual HRESULT CreateAndAddFile(
 IFileSpec *newFileSpec,
 ICodeWarriorProject *project,
 ICodeWarriorTargetCollection *targets,
 VARIANT_BOOL *fileAdded) = 0;
```

**newFileSpec**

A pointer to the IFileSpec interface that contains the file specification for the newly created file.

**project**

A pointer to the ICodeWarriorProject interface containing the project to which the new file is to be added.

**targets**

A pointer to the ICodeWarriorTargetCollection interface containing a list of the targets to which the new file is to be added.

**fileAdded**

Set this parameter to true if the file is successfully added to the specified project or false if the file could not be added.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "IFileSpec" on page 599

"ICodeWarriorProject" on page 630

---



“Using the Collections API” on page 487

CreateUntitledFile

The CodeWarrior IDE calls this method when the user instructs the IDE to create a new file with this creatable item selected, and the **Add to project** option is unchecked.

```
virtual HRESULT CreateUntitledFile(void) = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**ICodeWarriorCreateObjectItem**

This interface is used for a creatable file item that is displayed in the **Object** pane of creatable items visible in the **New** window in the CodeWarrior IDE.

**Inherited Interfaces**

- ICodeWarriorCreatableItem
- IUnknown

**Methods**

This interface provides the following methods:

|                           |                       |
|---------------------------|-----------------------|
| AreObjectsCreatedInDesign | CreateObjectInTargets |
| CreateObjectInDesign      | NeedsObjectName       |

---

**AreObjectsCreatedInDesign**

The CodeWarrior IDE calls this method to determine if this creatable item creates objects in designs.

```
virtual HRESULT AreObjectsCreatedInDesign(
 void) = 0;
```

Returns     Return S\_OK if this creatable item is able to create objects in designs or an appropriate error if this creatable item is not able to create objects in designs.

---

## CreateObjectInDesign

The CodeWarrior IDE calls this method to instruct this creatable item to create an object in a specific design of a project.

```
virtual HRESULT CreateObjectInDesign(
 BSTR newItemName,
 ICodeWarriorProject *project,
 ICodeWarriorDesign *design) = 0;
```

**newItemName**

The requested name of the new object being created.

**project**

A pointer to the ICodeWarriorProject interface. This parameter specifies the project containing the design to which the object is being added.

**design**

A pointer to the ICodeWarriorDesign interface. This parameter specifies the design to which the object is being added.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProject" on page 630

"ICodeWarriorDesign" on page 542

## Creatable Items

### CreateObjectInTargets

---

## CreateObjectInTargets

The CodeWarrior IDE calls this method to instruct this creatable item to create an object in a specific target of a project.

```
virtual HRESULT CreateObjectInTargets (
 BSTR newItemName,
 ICodeWarriorProject *project,
 ICodeWarriorTargetCollection *targets) = 0;
```

**newItemName**

The requested name for the new object being created.

**project**

A pointer to the `ICodeWarriorProject` interface. This parameter specifies the project containing the design to which the object is being added.

**targets**

A pointer to the `ICodeWarriorTargetCollection` interface. This parameter specifies a list of the targets to which the object is being added.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProject" on page 630  
 "Using the Collections API" on page 487



NeedsObjectName

The CodeWarrior IDE calls this method to determine if this creatable item requires a user-specified name. If so, the IDE enables the appropriate option in the **New** dialog..

```
virtual HRESULT NeedsObjectName(void) = 0;
```

Returns S\_OK if this creatable item requires a user-specified name or an appropriate error if this creatable item does not require a user-specified name.

## ICodeWarriorCreateProjectItem

This interface is used for a creatable file item that is displayed in the **Project** pane of creatable items visible in the **New** window in the CodeWarrior IDE.

### Inherited Interfaces

- ICodeWarriorCreatableItem
- IUnknown

### Methods

This interface provides the following methods:

|                         |                       |
|-------------------------|-----------------------|
| CreateInExistingProject | GetCreatedProjectType |
| CreateNewProject        | RequiresFileExtension |

## CreateInExistingProject

The CodeWarrior IDE calls this method when the user instructs CodeWarrior to create a new project using this creatable item with the **Add to project** option selected in the **New** window. Your plug-in is expected to create a new project and add it to an existing project as a subproject.

```

virtual HRESULT CreateInExistingProject(
 BSTR newItemName,
 ICodeWarriorProject *project) = 0;

```

newItemName

A BSTR containing the name of the new project being created.

project

A pointer to the ICodeWarriorProject interface containing the project to which the new project is being added.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---

## CreateNewProject

The CodeWarrior IDE calls this method when the user instructs IDE to create a new project using this creatable item.

```
virtual HRESULT CreateNewProject(
 IFileSpec *newFileSpec) = 0;
```

*newFileSpec*

A pointer to the IFileSpec interface containing a file specification with the user-specified name of the new project and pointing to the location where the new project should be created.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599

---

## GetCreatedProjectType

The CodeWarrior IDE calls this method to determine what type of project this creatable item generates.

```
virtual HRESULT GetCreatedProjectType(
 ECreateProjectType *pval) = 0;
```

*pval*

A pointer to a value in the range defined by the ECreateProjectType enumeration, reflecting the type of project this creatable item generates.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECreateProjectType" on page 538



**Creatable Items**  
*RequiresFileExtension*

---

**RequiresFileExtension**

The CodeWarrior calls this method to determine if this creatable item requires a file extension.

```
virtual HRESULT RequiresFileExtension(
 VARIANT_BOOL *pval) = 0;
```

*pval*

Set this parameter to `true` if this creatable item requires a file extension (such as `.mcp` or `.txt`) for the file to be valid. Set it to `false` if this creatable item does not require a file extension. For example, a Mac OS plug-in might return `false`, because a file extension is not required for the file to be valid on Mac OS.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.



## Creatable Items Data Types

The following data types are used with the Creatable Items API:

- Built-in Icon Index Values
- Creatable Item Category Constants
- ECreateProjectType

## Creable Items

### Creable Items Data Types

#### Built-in Icon Index Values

CodeWarrior supplies built-in icons for use with creable items. Table 14.1 shows the indexes of the built-in icons for creable items.

**Table 14.1 Built-in Icons**

| Constant        | Icon                                                                              | Value | Description               |
|-----------------|-----------------------------------------------------------------------------------|-------|---------------------------|
| newIconProject  |  | -1    | CodeWarrior project files |
| newIconTextFile |  | -2    | CodeWarrior text files    |
| newIconCatalog  |  | -3    | CodeWarrior catalog files |

#### Creable Item Category Constants

These constants are used to describe the possible categories of creable items displayed by the CodeWarrior IDE in the New window. Implementations of the `ICodeWarriorCreableItem` interface should return one of these constants for their category.

**Table 14.2 Creable Item Categories**

| Constant                  | Description                                                           |
|---------------------------|-----------------------------------------------------------------------|
| kMWNewProjectCategoryName | Creable items with this category are displayed in the "Project" pane. |
| kMWNewFileCategoryName    | Creable items with this category are displayed in the "File" pane.    |
| kMWNewObjectCategoryName  | Creable items with this category are displayed in the "Object" pane.  |

#### ECreateProjectType

CodeWarrior categorizes `ICodeWarriorCreateProjectItem` interfaces with the following types based on the capabilities of the creable item in question.



Table 14.3 ECreateProjectType Enumerations

| Constant           | Description                                            |
|--------------------|--------------------------------------------------------|
| createsProjectOnly | Creatable item only creates projects.                  |
| createsDesign      | Creatable item creates and designs.                    |
| createsTargets     | Creatable item creates projects, designs, and targets. |



# Freescal Semiconductor, Inc.

## Creatable Items

*Creatable Items Data Types*

---

# Designs

---

This chapter shows how to use the Designs API to manage designs in the CodeWarrior IDE.

This chapter contains the following sections:

- Designs API Overview
- Designs API Reference

## Designs API Overview

This API lets you manipulate designs within a project. You can use it to add and remove files and targets, initialize and close designs, and otherwise change the details of designs.

## Designs API Reference

This section describes the functions contained in the following interfaces:

- ICodeWarriorDesign
- ICodeWarriorDesignAttachment
- ICodeWarriorDesignEvents

The following data types are used with these interfaces:

- ECodeWarriorLinkFlags



ICodeWarriorDesign

This interface provides methods for working with designs within the CodeWarrior IDE.

Inherited Interfaces

- IDispatch
- IUnknown

Methods

This interface provides the following methods:

|                               |                        |
|-------------------------------|------------------------|
| AddAttachment                 | FindAndAddFile2        |
| AddFile                       | get_BrowserDB          |
| AddFile2                      | get_DataModel          |
| AddFileByFileSpec             | get_Name               |
| AddFile2ByFileSpec            | get_Project            |
| CompileFiles                  | get_Targets            |
| CompileFilesAndWaitToComplete | put_Name               |
| ContainsTarget                | RemoveAttachment       |
| FindAndAddFile                | RemoveTargetFromDesign |

AddAttachment

This method adds an attachment to the design.

```
virtual HRESULT AddAttachment(
 const CLSID *attachmentCLSID) = 0;

attachmentCLSID
```

A pointer to the attachment.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "RemoveAttachment" on page 551

---

## AddFile

This method adds a file to the design.

```
virtual HRESULT AddFile(
 BSTR path,
 BSTR groupPath,
 ICodeWarriorProjectFile **projectFile) = 0;
```

path

The absolute path to the file you want to add to the design.

groupPath

The absolute path to the group within which the new file should be added.

projectFile

The address of a pointer to the file you want to add to the project.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## AddFile2

This method adds a file to the design and set link flags on the file.

```
virtual HRESULT AddFile2(
 BSTR path,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 ICodeWarriorProjectFile **projectFile) = 0;
```

path

The absolute path to the file you want to add to the design.

## Designs

### AddFile2ByFileSpec

---

groupPath

The path to the group within which the new file should be added.

linkFlags

A value in the range defined by the ECodeWarriorLinkFlags enumeration, representing how the linker should link this file.

projectFile

The address of a pointer to the file you want to add to the project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorLinkFlags" on page 557

---

### AddFile2ByFileSpec

This method adds a file to the design, by using a file specification object, and set link flags on the file.

```
virtual HRESULT AddFile2ByFileSpec(
 IFileSpec *fileSpec,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 ICodeWarriorProjectFile **projectFile) = 0;
```

fileSpec

The file specification (as a pointer to an IFileSpec object) that defines the file to add to the project.

groupPath

The absolute path to the group within which the new file should be added.

linkFlags

A value in the range defined by the ECodeWarriorLinkFlags

enumeration, representing how the linker should link this file.

projectFile

The address of a pointer to the file you want to add to the project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## AddFileByFileSpec

This method adds a file to the design, by using a file specification object.

```
virtual HRESULT AddFileByFileSpec(
 IFileSpec *fileSpec,
 BSTR groupPath,
 ICodeWarriorProjectFile **projectFile) = 0;
```

fileSpec

The file specification (as a pointer to an IFileSpec object) that defines the file to add to the project.

groupPath

The absolute path to the group within which the new file should be added.

projectFile

The address of a pointer to the file you want to add to the project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Designs

### CompileFiles

---

## CompileFiles

This method causes a compilation of the current design.

```
virtual HRESULT CompileFiles(
 ICodeWarriorProjectFileCollection *collection,
 long *cookie) = 0;
```

collection

A pointer of type ICodeWarriorProjectFileCollection indicating the collection of files to compile.

cookie

On return, this parameter contains a unique identifier for the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

---

## CompileFilesAndWaitToComplete

This method causes a compilation of the current design and returns all the build messages created by the compiler.

```
virtual HRESULT CompileFilesAndWaitToComplete(
 ICodeWarriorProjectFileCollection *collection,
 ICodeWarriorBuildMessages **buildMessages) = 0;
```

collection

A pointer of type ICodeWarriorProjectFileCollection indicating the collection of files to compile.

buildMessages

On return, this parameter contains the address of a pointer to the build messages generated by the compiler.

---

|          |                                                                          |
|----------|--------------------------------------------------------------------------|
| Returns  | S_OK if this method call succeeded or an appropriate error if it failed. |
| See Also | “Using the Collections API” on page 487                                  |

---

## ContainsTarget

This method discovers if the current design contains a particular target.

```
virtual HRESULT ContainsTarget(
 ICodeWarriorTarget *target) = 0;
```

target

A pointer to the target you are looking for within the design.

|         |                                                                                                   |
|---------|---------------------------------------------------------------------------------------------------|
| Returns | S_OK if the specified target is present within the current design or an appropriate error if not. |
|---------|---------------------------------------------------------------------------------------------------|

---

## FindAndAddFile

This method finds a file and adds it to the design.

```
virtual HRESULT FindAndAddFile(
 BSTR path,
 BSTR groupPath,
 ICodeWarriorProjectFile **projectFile) = 0;
```

path

Either the absolute (fully qualified) path to the file you want to add to the design or just the name of the file. If you provide just the file name, the IDE searches the access paths and adds the first file of that name that it finds. If you want to add two one files with identical names, use the fully qualified path to each one.

## Designs

### FindAndAddFile2

---

groupPath

The absolute path to the group within which the new file should be added.

projectFile

On return, this parameter contains the address of a pointer to the project file to which the file was added.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectFile" on page 659

---

## FindAndAddFile2

This method finds a file and adds it to the design. This method also lets you set link flags on the file.

```
virtual HRESULT FindAndAddFile2(
 BSTR path,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 ICodeWarriorProjectFile **projectFile) = 0;
```

path

Either the absolute (fully qualified) path to the file you want to add to the design or just the name of the file. If you provide just the file name, the IDE searches the access paths and adds the first file of that name that it finds. If you want to add two one files with identical names, use the fully qualified path to each one.

groupPath

The absolute path to the group within which the new file should be added.

linkFlags

A value in the range defined by the ECodeWarriorLinkFlags enumeration, representing how the linker should link this file.

projectFile

On return, this parameter contains the address of a pointer to the project file to which the file was added.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorLinkFlags" on page 557

---

## get\_BrowserDB

This method gets a pointer to a listing of the symbols created by a compilation of the files in the design.

```
virtual HRESULT get_BrowserDB(
 ICodeWarriorSymbolContainer **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to a container holding the symbols created by the latest compilation of the files in the design.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorSymbolContainer" on page 694

---

## get\_DataModel

This method gets the data model for this design.

```
virtual HRESULT get_DataModel(
 IUnknown **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the data model for the design.

## Designs

### *get\_Name*

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_Name

This method gets the name of a design.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the name of the design.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_Project

This method gets the project object for the project to which the current design belongs.

```
virtual HRESULT get_Project(
 ICodeWarriorProject **pval) = 0;
```

pval

Upon return, this parameter contains the address of a pointer to the project that the design belongs to.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---

---

**get\_Targets**

This method gets a list of build targets in this design.

```
virtual HRESULT get_Targets(
 ICodeWarriorTargetCollection **pval) = 0;
```

**pval**  
Upon return, this parameter contains a collection of the build targets in this design.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

---

**put\_Name**

This method to set the name of a design.

```
virtual HRESULT put_Name(
 BSTR pval) = 0;
```

**pval**  
The name you are assigning to the design.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

**RemoveAttachment**

This method removes an attachment from the design.

```
virtual HRESULT RemoveAttachment(
 const CLSID *attachmentCLSID) = 0;
```



## Freescal Semiconductor, Inc.

### Designs

#### *RemoveTargetFromDesign*

---

`*attachmentCLSID`

A pointer to the attachment you want to remove.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "AddAttachment" on page 542

---

#### RemoveTargetFromDesign

This method removes a target from a design

```
virtual HRESULT RemoveTargetFromDesign(
 ICodeWarriorTarget *target) = 0;
```

`*target`

A pointer to the target you want to remove.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## ICodeWarriorDesignAttachment

This interface provides methods to detect whether certain events performed on a design have completed.

### Inherited Interfaces

- IUnknown

### Methods

This interface provides the following methods:

|                   |                    |
|-------------------|--------------------|
| DesignClosing     | RemovingAttachment |
| DesignInitialized |                    |

### DesignClosing

This method closes the design.

```
virtual HRESULT DesignClosing(
 ICodeWarriorDesign *__MIDL_0015) = 0;

__MIDL_0015
```

A pointer to the design to be closed.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### DesignInitialized

This method prepares a design for use.

```
virtual HRESULT DesignInitialized(
 ICodeWarriorDesign *__MIDL_0014) = 0;

__MIDL_0014
```

A pointer to the design you want to initialize.



# Freescal Semiconductor, Inc.

## Designs

### RemovingAttachment

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### RemovingAttachment

Use this method to determine if the CodeWarrior IDE has finished removing an attachment from the design.

```
virtual HRESULT RemovingAttachment(
 ICodeWarriorDesign *__MIDL_0016) = 0;

__MIDL_0016
```

A pointer to the attachment being removed.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## ICodeWarriorDesignEvents

Use this interface to determine if certain events have taken place while working with a design.

### Inherited Interfaces

- IUnknown

### Methods

This interface provides the following methods:

|                |             |
|----------------|-------------|
| RemovingTarget | TargetAdded |
|----------------|-------------|

### RemovingTarget

This method detects whether the CodeWarrior IDE has finished removing a target. (How does one know the target has been removed?)

```
virtual HRESULT RemovingTarget(
 ICodeWarriorTarget *target) = 0;
```

target

A pointer to the target being removed.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Designs**  
*TargetAdded*

---

---

**TargetAdded**

This method detects whether the CodeWarrior IDE has finished adding a target to the design. (How does one know the target has been added?)

```
virtual HRESULT TargetAdded(
 ICodeWarriorTarget *target) = 0;
```

target

A pointer to the target being added.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



Data Types

ECodeWarriorLinkFlags

This enumeration is used to define linker flags. It is used in the AddFile2, AddFile2ByFileSpec, and FindAndAddFile2 methods of the ICodeWarriorDesign interface.

Table 15.1 ECodeWarriorLinkFlags Enumeration

| Constant          | Description                                                                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| cwNoLinkFlags     | Set no link flags on this file                                                                                                                               |
| cwGenerateSymbols | Instruct the linker to only generate symbols                                                                                                                 |
| cwMergeLibrary    | Instruct the linker to link to libraries when building.                                                                                                      |
| cwWeakImport      | Instruct the linker to use the Weak Import option. This option only works on the Mac OS                                                                      |
| cwInitBefore      | Instruct the linker to use shared libraries. This flag is only valid if the currently mapped compiler support import. This option works only on the Power PC |



# Freescal Semiconductor, Inc.

## Designs

*Data Types*

---

## Dialog Services

---

This chapter shows how to use the Dialog Services API to manage dialog operations in the CodeWarrior IDE.

This chapter contains the following sections:

- Dialog Services API Overview
- Using the Dialog Services API
- Dialog Services API Reference

### Dialog Services API Overview

The Dialog Services API is a set of interfaces that allows a plug-in to create and manipulate dialog boxes in the CodeWarrior IDE.

### Using the Dialog Services API

This section covers these topics:

- Registering the Command
- Implementing the Command

#### Registering the Command

The following code snippet, from `PluginMain.cpp`, shows how to register a dialog command:

---

```
/*
 * RegisterCommands
 *
 * Creates a new command group, adds two commands to it, and
 * tells the IDE to display the group in the menu bar.
 */
```

---



## Freescale Semiconductor, Inc.

### Dialog Services

#### Registering the Command

---

```
static void RegisterCommands(IServiceProvider *servProv)
{
 ICodeWarriorCommandRegistry *cmdRegistry;
 servProv->QueryService(SID_SCodeWarriorCommandRegistry,
 IID_ICodeWarriorCommandRegistry, &cmdRegistry);
 if(cmdRegistry)
 {
 // We don't need the window manager to register commands,
 // but the command handler we're installing will need a
 // reference to it in order to create windows...
 ICodeWarriorWindowManager *windowMgr = NULL;
 servProv->QueryService(SID_SCodeWarriorWindowManager,
 IID_ICodeWarriorWindowManager, &windowMgr);

 ICodeWarriorCommandHandler *cmdHandler = new
 ExampleCommandHandler(windowMgr, servProv);
 BSTR bstr = SysAllocString(OLESTR("Example plugin"));
 cmdRegistry->CreateNewCommandGroup(kToolbarTestPluginID,
 cmdGroup_TestPlugin, bstr, cmdGroup_Nothing);

 CWToolbarIconInfo tbIconInfo =
 GetToolbarIcon(iconIndex_NewPluginWindow);

 // register test dialog services commands
 SysReAllocString(&bstr, OLESTR("Test Info Dialog"));
 cmdInfo.pluginID = kToolbarTestPluginID;
 cmdInfo.commandID = cmd_TestInfoDialog;
 cmdInfo.commandGroupID = cmdGroup_TestPlugin;
 cmdInfo.commandName = bstr;
 cmdInfo.toolbarIcon = tbIconInfo;
 cmdInfo.visibleInMenu = true;
 cmdInfo.itemType = CWCommandItemType_Command;
 cmdInfo.extraInfo.commandHandler = cmdHandler;
 cmdRegistry->RegisterCommand(cmdInfo, cmd_Nothing);

 SysFreeString(bstr);
 FreeToolbarIcon(tbIconInfo);

 cmdRegistry->Release();
 }
}
```

---

```

IcodeWarriorMenuManager *menuMgr;
servProv->QueryService(SID_ScodeWarriorMenuManager,
 IID_IcodeWarriorMenuManager, &menuMgr);
if (menuMgr)
{
 menuMgr->ShowCommandGroupMenu(kToolbarTestPluginID,
 cmdGroup_TestPlugin, true);
 menuMgr->Release();
}
}

```

---

## Implementing the Command

The following code snippet, from ExampleCommandHandler.cpp, shows how to implement a dialog command:

---

```

/*
 *ExecuteCommand
 */

HRESULT STDMETHODCALLTYPE ExampleCommandHandler::ExecuteCommand(
 CWCommandID inCommandNumber,
 IcodeWarriorCommandHandler *inDefaultHandler)
{
 HRESULT result = S_OK;

 switch (inCommandNumber)
 {
 case cmd_TestInfoDialog:
 case cmd_TestWarningDialog:
 case cmd_TestErrorDialog:
 {
 if (mServProv)
 {
 IcodeWarriorDialogServices* dlgSrvc = NULL;
 result =
 mServProv->QueryService(IID_IcodeWarriorDialogServices,
 IID_IcodeWarriorDialogServices, (void**) &dlgSrvc);
 if (SUCCEEDED(result))
 {
 short dialogType = 0;

```

---

## Dialog Services

### Implementing the Command

---

```

 BSTR bstr;
 switch(inCommandNumber)
 {
 case cmd_TestInfoDialog:
 dialogType = cwInfoDialog;
 bstr = SysAllocString(OLESTR("Info OKCancelDialog
 - cwInfoDialog"));
 break;
 case cmd_TestWarningDialog:
 dialogType = cwWarningDialog;
 bstr = SysAllocString(OLESTR("Warning OKCancelDialog
 - cwWarningDialog"));
 break;
 case cmd_TestErrorDialog:
 dialogType = cwErrorDialog;
 bstr = SysAllocString(OLESTR("Error OKCancelDialog -
 cwErrorDialog"));
 break;
 };
 result = dlgSrv->OKCancelDialog(dialogType, bstr);
 SysFreeString(bstr);

 dlgSrv->Release();
}
}
break;
}

default:
 if(inDefaultHandler)
 result = inDefaultHandler->ExecuteCommand(inCommandNumber,
 inDefaultHandler);
 break;
}

return result;
}

```

---



## Dialog Services API Reference

This section describes the functions contained in the following interface:

- ICodeWarriorDialogServices

The ICodeWarriorDialogServices interface uses the following data types:

- Dialog Box Types
- SaveDontSaveDialog Result Types

### ICodeWarriorDialogServices

This interface allows plug-ins to present dialogs to the user and request feedback from the user.

#### Inherited Interfaces

- IUnknown

#### Methods

This interface provides the following methods:

|                 |                               |
|-----------------|-------------------------------|
| NewItemDialog   | ReportErrorFromErrorInfo      |
| OKCancelDialog  | SaveDontSaveDialog            |
| OKDialog        | SetPluginDialogCommandHandler |
| PostModalDialog | UpdatePluginDialogMenus       |
| PreModalDialog  |                               |

### NewItemDialog

This method shows the user a new item dialog.

```
virtual HRESULT NewItemDialog(
 BSTR pageToSelect,
 BSTR itemToSelect) = 0;
```

## Dialog Services

### *OKCancelDialog*

---

pageToSelect

If the dialog has multiple pages, the page to show to the user.  
This parameter is optional

itemToSelect

The item on the selected page to highlight when the dialog appears to the user. If the dialog has multiple pages, the items is relative to the page. This parameter is optional

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### OKCancelDialog

This method shows the user a dialog box with an **OK** button and a **Cancel** button.

```
virtual HRESULT OKCancelDialog(
 short dialogType,
 BSTR message) = 0;
```

dialogType

A value within the range defined by the Dialog Box Types enumeration, indicating the type of dialog box to present to the end user.

message

The message you want to display in this dialog.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Dialog Box Types" on page 569

## OKDialog

This method shows the user a dialog with an **OK** button.

```
virtual HRESULT OKDialog(
 short dialogType,
 BSTR message) = 0;
```

dialogType

A value within the range defined by the Dialog Box Types enumeration, indicating the type of dialog box to present to the end user.

message

The message you want to display in this dialog.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Dialog Box Types" on page 569

---

## PostModalDialog

This method informs the CodeWarrior IDE that you are done with a modal dialog.

```
PostModalDialog(void) = 0;
```

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## PreModalDialog

This method informs the CodeWarrior IDE that you are about to display a modal dialog. You can then use

## Dialog Services

### *ReportErrorFromErrorInfo*

---

SetPluginDialogCommandHandler and UpdatePluginDialogMenus to update the menu while the modal dialog has focus.

```
virtual HRESULT PreModalDialog(
 BOOL fActivateIDE) = 0;
```

fActivateIDE

true to bring the IDE to the front or false to leave it behind other applications (if any).

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "SetPluginDialogCommandHandler" on page 567

"UpdatePluginDialogMenus" on page 568

---

### ReportErrorFromErrorInfo

This method shows the user an error message in a dialog.

```
virtual HRESULT ReportErrorFromErrorInfo(
 ICodeWarriorErrorInfo *info) = 0;
```

info

A pointer to the ICodeWarriorErrorInfo interface containing the error message to show.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorErrorInfo" on page 590

## SaveDontSaveDialog

This method shows a message to the user in a dialog with a **Save** button and a **Don't Save** button..

```
virtual HRESULT SaveDontSaveDialog(
 BSTR objectType,
 BSTR objectName,
 long *result) = 0;
```

**objectType**

The type of object being saved. If the object is a file, you may pass NULL for this parameter.

**objectName**

The default name of the object being saved. This name appears in the edit field of the dialog box, and may be edited by the user before saving.

**result**

On return, this parameter contains an integer indicating which option (Save, Don't Save, or Cancel) the user chose.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "SaveDontSaveDialog Result Types" on page 569

---

## SetPluginDialogCommandHandler

This method sets up a command handler to process menu events while a plugin is showing a modal dialog. Call this method after PreModalDialog(). After using this method, you can use UpdatePluginDialogMenus to update the menu bar while the modal dialog has focus.

```
virtual HRESULT SetPluginDialogCommandHandler(
 ICodeWarriorCommandHandler* inCommandHandler
```



## Freescal Semiconductor, Inc.

### Dialog Services

#### *UpdatePluginDialogMenus*

---

) = 0;

inCommandHandler

A pointer to the command handler to use while a modal dialog has the focus.

See Also "PreModalDialog" on page 565

"UpdatePluginDialogMenus" on page 568

---

### UpdatePluginDialogMenus

This method updates the menu bar while a modal dialog has the focus. Once a plugin dialog command handler has been set, call this method in your event loop to update the menus. Use this method after calling SetPluginDialogCommandHandler.

virtual HRESULT UpdatePluginDialogMenus() = 0;

See Also "PreModalDialog" on page 565

"SetPluginDialogCommandHandler" on page 567

### Dialog Services Data Types

The following data types are used with the Dialog Services API:

- Dialog Box Types

#### Dialog Box Types

These constants are used to describe the types of dialogs available for use with the OKDialog and OKCancelDialog methods of the ICodeWarriorDialogServices interface.

Table 16.1 Dialog Box Types

| Constant        | Icon                                                                               | Description                                          |
|-----------------|------------------------------------------------------------------------------------|------------------------------------------------------|
| cwInfoDialog    |   | Used to present information to the end user          |
| cwWarningDialog |   | Used to present a caution or warning to the end user |
| cwErrorDialog   |  | Used to present an error condition to the end user   |

#### SaveDontSaveDialog Result Types

The SaveDontSaveDialog method of the ICodeWarriorDialogServices interface returns one of three possible values, depending on the user's choice:

- cwSaveResponse\_Save
- cwSaveResponse\_DontSave
- cwSaveResponse\_Cancel



# Freescal Semiconductor, Inc.

## Dialog Services

*Dialog Services Data Types*

---

# Documents

---

This chapter shows how to use the Documents API to create and manage documents in the CodeWarrior IDE.

This chapter contains the following sections:

- Documents API Overview
- Documents API Reference

## Documents API Overview

The Documents API is a set of interfaces that allows a plug-in to create and manipulate components in the CodeWarrior IDE.

## Documents API Reference

This section describes the functions contained in the following interfaces:

- ICodeWarriorDocument
- ICodeWarriorProjectDocument
- ICodeWarriorTextDocument



## ICodeWarriorDocument

This interface is used to get information about and control CodeWarrior documents and their windows.

### Inherited Interfaces

- IUnknown

### Methods

This interface has the following properties:

|                    |             |
|--------------------|-------------|
| Activate           | get_Width   |
| Close              | get_XPos    |
| get_ActiveDocument | get_YPos    |
| get_Dirty          | put_Height  |
| get_FileSpec       | put_Visible |
| get_Height         | put_Width   |
| get_Name           | put_XPos    |
| get_ReadOnly       | put_YPos    |
| get_Visible        | Save        |

### Activate

This method activates a document. Activating a document makes the document the frontmost document.

```
virtual HRESULT Activate(void) = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

Close

This method closes the document window and optionally saves the document.

```
virtual HRESULT Close(
 VARIANT_BOOL bSaveChanges) = 0;
```

**bSaveChanges**  
A VARIANT\_BOOL containing true to save the document before closing or false to close the document without saving.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

get\_ActiveDocument

This method gets whether a document is the active document.

```
virtual HRESULT get_ActiveDocument(
 VARIANT_BOOL *pval) = 0;
```

**pval**  
On return it contains true if the document is active or false if the document is inactive.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

get\_Dirty

This method gets the dirty status (whether or not it has been modified since the last save) of a document.

```
virtual HRESULT get_Dirty(
 VARIANT_BOOL *pval) = 0;
```

## Documents

### *get\_FileSpec*

---

pval

Supply a pointer to the VARIANT\_BOOL interface. Upon return it contains `true` if the document is dirty (needs to be written to disk), and `false` if the document is not dirty.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### *get\_FileSpec*

This method gets the file specification for a document.

```
virtual HRESULT get_FileSpec(
 IFileSpec **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the file specification for the document.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### *get\_Height*

This method gets the height of a document window.

```
virtual HRESULT get_Height(
 int *pval) = 0;
```

pval

On return, this parameter contains a pointer to the current height (in pixels) of the document window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Name

This method gets the name of a document.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains a pointer to the name of the document.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_ReadOnly

This method gets the read-only status of a document.

```
virtual HRESULT get_ReadOnly(
 VARIANT_BOOL *pval) = 0;
```

pval

On return it contains true if the document is read only or false if the document is modifiable.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Visible

This method gets the visible status of a document window.

```
virtual HRESULT get_Visible(
 VARIANT_BOOL *pval) = 0;
```



## Freescal Semiconductor, Inc.

### Documents

#### *get\_Width*

---

pval

On return, this parameter contains a pointer to a boolean indicating `true` if the document window is visible or `false` if the document window is not visible.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### *get\_Width*

This method gets the width of a document window.

```
virtual HRESULT get_Width(
 int *pval) = 0;
```

pval

On return, this parameter contains a pointer to an integer indicating the current width (in pixels) of the document window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### *get\_XPos*

This method gets the horizontal position of a document window.

```
virtual HRESULT get_XPos(
 int *pval) = 0;
```

pval

On return, this parameter contains a pointer to an integer indicating the current horizontal coordinate of the top-left corner of the document window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

---

## get\_YPos

This method gets the vertical position of a document window.

```
virtual HRESULT get_YPos(
 int *pval) = 0;
```

pval

On return, this parameter contains a pointer to an integer indicating the current verticle coordinate of the top-left corner of the document window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_Height

This method sets the height of a document window.

```
virtual HRESULT put_Height(
 int val) = 0;
```

val

An integer set to the new height (in pixels) of the document window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_Visible

This method sets the visible state of a document window.

```
virtual HRESULT put_Visible(
 VARIANT_BOOL val) = 0;
```

## Documents

### *put\_Width*

---

val

A `VARIANT_BOOL` set to `true` if you want the document window made visible or `false` if you want the document window hidden.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

---

### *put\_Width*

This method sets the width of a document window.

```
virtual HRESULT put_Width(
 int val) = 0;
```

val

An integer set to the new width (in pixels) of the document window.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

---

### *put\_XPos*

This method sets the horizontal position of a document window.

```
virtual HRESULT put_XPos(
 int val) = 0;
```

val

An integer variable set to the new horizontal coordinate of the top-left corner of the document window.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

---



---

put\_YPos

This method sets the verticle position of a document window.

```
virtual HRESULT put_YPos(
 int val) = 0;
```

val

An integer variable set to the new verticle coordinate of the top-left corner of the document window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

Save

This method saves a document.

```
virtual HRESULT Save(void) = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

**ICodeWarriorProjectDocument**

This interface is used to get information about and control CodeWarrior project documents.

**Inherited Interfaces**

- IUnknown

**Methods**

This interface provides the following methods:

|               |               |
|---------------|---------------|
| CollapseGroup | SelectFiles   |
| ExpandGroup   | SelectedFiles |
| get_Project   |               |

---

**CollapseGroup**

This method collapses a group in a project window.

```
virtual HRESULT CollapseGroup(
 BSTR groupName) = 0;

groupName
```

The name of the group to collapse.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## ExpandGroup

This method expands a group in a project window.

```
virtual HRESULT ExpandGroup(
 BSTR groupName) = 0;
```

groupName

The name of the group to expand.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Project

This method gets the project object related to this document.

```
virtual HRESULT get_Project(
 ICodeWarriorProject **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the project object related to this document.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProject" on page 630

## Documents

SelectFiles

### SelectFiles

This method selects or deselects one or more files in a project.

```
virtual HRESULT SelectFiles(
 ICodeWarriorProjectFileCollection
 *projectFiles,
 VARIANT_BOOL select) = 0;
```

projectFiles

A pointer to the ICodeWarriorProjectFileCollection interface containing the list of project files to select.

select

Set this parameter to true if you want to select the files or false if you want to deselect the files.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

### SelectedFiles

This method gets a list of currently selected files in a project.

```
virtual HRESULT SelectedFiles(
 ICodeWarriorProjectFileCollection
 **projectFiles) = 0;
```

projectFiles

Supply the address of a pointer to the ICodeWarriorProjectFileCollection interface. Upon return it contains a list of the currently selected project files.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487



ICodeWarriorTextDocument

This interface is used to get information about and control CodeWarrior text documents.

Inherited Interfaces

- IUnknown

Methods

This interface provides the following methods:

|                |                       |
|----------------|-----------------------|
|                | SaveACopyAs           |
|                | SaveACopyAsByFileSpec |
|                | SaveAs                |
|                | SaveAsByFileSpec      |
| get_TextEngine | ScrollToSelection     |

get\_Project

This method gets the project object associated with this text file.

```

virtual HRESULT get_Project(
 ICodeWarriorProject** project) = 0;

project

```

On return, this parameter contains a pointer to the address of the project object associated with this text file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Documents

*get\_Target*

---

### get\_Target

This method gets the target object associated with this text file.

```
virtual HRESULT get_Target(
 ICodeWarriorTarget** target) = 0;
```

target

On return, this parameter contains a pointer to the address of the target object associated with this text file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_ProjectFile

This method gets the project file object associated with this text file.

```
virtual HRESULT get_ProjectFile(
 ICodeWarriorProjectFile** projectFile) = 0;
```

projectFile

On return, this parameter contains a pointer to the address of the project file object associated with this text file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_TargetFile

This method gets the target file object associated with this text file.

```
virtual HRESULT get_TargetFile(
 ICodeWarriorTargetFile** targetFile) = 0;
```

---

targetFile

On return, this parameter contains a pointer to the address of the target file object associated with this text file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_TextEngine

This method gets the text engine object of a CodeWarrior text document.

```
virtual HRESULT get_TextEngine(
 ICodeWarriorTextEngine **pval) = 0;
```

pval

On return, this parameter contains the address of a pointer to the text engine object for this text document.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTextEngine" on page 768

---

## SaveACopyAs

This method saves a text file using the **Save A Copy As** dialog box, by specifying the full path of the file.

```
virtual HRESULT SaveACopyAs(
 BSTR val) = 0;
```

val

The full path of the file being saved.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Documents

*SaveACopyAsByFileSpec*

---

### SaveACopyAsByFileSpec

This method saves a text file using the **Save A Copy As** dialog box, by specifying the file specification record for the file.

```
virtual HRESULT SaveACopyAsByFileSpec(
 IFileSpec *fileSpec) = 0;
```

fileSpec

A pointer to the IFileSpec interface containing the file specification for the file being saved.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SaveAs

This method saves a text file using the **Save As** dialog box, by specifying the full path of the file.

```
virtual HRESULT SaveAs(
 BSTR val) = 0;
```

val

The full path of the file being saved.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SaveAsByFileSpec

This method saves a text file using the **Save As** dialog box, by specifying the file specification record for the file.

```
virtual HRESULT SaveAsByFileSpec(
 IFileSpec *fileSpec) = 0;
```



fileSpec

A pointer to the IFileSpec interface containing the file specification for the file being saved.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

ScrollToSelection

This method makes the currently selected text appear in the text document's editor window.

virtual HRESULT ScrollToSelection(void) = 0;

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescal Semiconductor, Inc.

## Documents

*ScrollToSelection*

---

# Error Info

---

This chapter shows how to use the Error Info API to manage and work with error information.

This chapter contains the following sections:

- Error Info API Overview
- Error Info API Reference

## Error Info API Overview

The Error Info API is a set of interfaces that allows a plug-in to work with error information in the CodeWarrior IDE.

## Error Info API Reference

This section describes the functions contained in the following interfaces:

- `ICodeWarriorErrorInfo`



ICodeWarriorErrorInfo

Inherited Interfaces

- IUnknown

Methods

This interface has the following methods:

|              |                 |
|--------------|-----------------|
| get_Action   | put_HelpContext |
| get_DWORDErr | put_HelpFile    |
| get_HRESULT  | put_HRESULT     |
| get_MacOSErr | put_MacOSErr    |
| get_MWErr    | put_MWErr       |
| get_Reason   | put_Reason      |
| put_Action   | put_Source      |
| put_DWORDErr |                 |

get\_Action

This method gets the action string associated with the most recent error.

```
virtual HRESULT get_Action(BSTR *actionStr) = 0;

actionStr
```

On return, this parameter contains the action string.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

## get\_DWORDErr

This method gets the error number of the most recent error.

```
virtual HRESULT get_DWORDErr(DWORD *err) = 0;
```

err

On return, this parameter contains a pointer to a DWORD that contains the error number.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_HRESULT

This method gets the HRESULT value for the most recent operation.

```
virtual HRESULT get_HRESULT(HRESULT *err) = 0;
```

err

On return, this parameter contains a pointer to an HRESULT that contains the HRESULT value for the most recent operation (0 for no error and other values for errors).

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_MWErr

This method gets the MWErr number for the most recent error.

```
virtual HRESULT get_MWErr(long *err) = 0;
```

err

On return, this parameter contains a pointer to a long integer

## Error Info

### *get\_MacOSErr*

---

containing the error number

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### *get\_MacOSErr*

This method gets the error number of the most recent error on a Mac OS.

```
virtual HRESULT get_MacOSErr(short *err) = 0;
```

*err*

On return, this parameter contains a pointer to a short that holds the error number.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### *get\_Reason*

This method gets a string that contains an explanation of what caused the most recent error.

```
virtual HRESULT get_Reason(BSTR *actionStr) = 0;
```

*actionStr*

On return, this parameter contains a string containing the cause of the error.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---



---

put\_Action

This method lets you set the action string for an error message.

```
virtual HRESULT put_Action(BSTR actionStr) = 0;

actionStr
```

A string telling the user what action to take to clear the error.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Error Info

*put\_DWORDErr*

---

#### put\_DWORDErr

This method lets you set the error number for an error.

```
virtual HRESULT put_DWORDErr(DWORD err) = 0;
```

err

The number you want to assign to the error.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### put\_HRESULT

This method lets you set the HRESULT value for an error.

```
virtual HRESULT put_HRESULT(HRESULT err) = 0;
```

err

The HRESULT value you want to assign to an error.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### put\_HelpContext

This method lets you put the help system into a certain state before displaying a help file. Refer to the operating system documentation for available states for the help system.

```
virtual HRESULT put_HelpContext(
 DWORD helpContext) = 0;
```

helpContext

A number indicating the state the help system should be in

---

when you display a help file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_HelpFile

This method lets you show the user a help file. You can use put\_HelpContext to put the help system into a particular state before you call the help file.

```
virtual HRESULT put_HelpFile(BSTR fileName) = 0;
```

fileName

The file name (and path, if necessary) of the help system to show the user.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_MWErr

This method lets you set the MWErr value for an error message.

```
virtual HRESULT put_MWErr(long err) = 0;
```

err

A number representing an MWErr state.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Error Info**  
*put\_MacOSErr*

---

---

**put\_MacOSErr**

This method lets you set the error number to a value recognized by the Mac OS.

```
virtual HRESULT put_MacOSErr(short err) = 0;

err
```

The Mac OS error number to use.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



put\_Reason

This method lets you set the reason string for an error message.

```
virtual HRESULT put_Reason(BSTR actionStr) = 0;

actionStr
```

A string indicating the reason the error occurred.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

put\_Source

This method lets you set the source of the error message.

```
virtual HRESULT put_Source(BSTR sourceStr) = 0;

sourceStr
```

A string stating the source of the error.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescale Semiconductor, Inc.

Error Info  
*put\_Source*

---

# Files

---

This chapter describes the functions contained in the following interface: IFileSpec

## IFileSpec

This interface allows you to work with file specifications.

### Inherited Interfaces

- IUnknown

### Methods

This interface provides the following methods:

|              |              |
|--------------|--------------|
| Clone        | get_Name     |
| Copy         | put_FullPath |
| get_FullPath | put_Name     |

---

## Clone

This method creates a duplicate copy of a file specification pointing to the same file as the original.

```
virtual HRESULT Clone(IFileSpec **pval);
```

pval

On return, this parameter contains the address of a pointer to an IFileSpec object pointing to the same file as the original file specification.



# Freescal Semiconductor, Inc.

**Files**  
*Copy*

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**Copy**

This method creates a duplicate copy of a file, by specifying the name and location of the new file.

```
virtual HRESULT Copy(IFileSpec *inSpec);
```

inSpec

A pointer to the IFileSpec interface containing a file specification with the name and the location of the new file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**get\_FullPath**

This method gets the full path of a file specification.

```
virtual HRESULT get_FullPath(BSTR *path);
```

pval

On return, this parameter contains the full path of the file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**get\_Name**

This method gets the name of a file specification.

```
virtual HRESULT get_Name(BSTR *pval);
```

---

pval

On return, this parameter contains the name of the file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_FullPath

This method sets the path of a file specification.

```
virtual HRESULT put_FullPath(BSTR path);
```

pval

The new path for the file specification. The path you supply does not need to point to an existing file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_Name

This method sets the name of a file specification.

```
virtual HRESULT put_Name(BSTR pval);
```

pval

The new name of the file. The name you supply does not have to point to an existing file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescale Semiconductor, Inc.

**Files**  
*put\_Name*

---

# Menus

---

This chapter shows how to use the Menus API to create and manage menus in the CodeWarrior IDE.

This chapter contains the following sections:

- Menus API Overview
- Using the Menus API
- Menus API Reference

## Menus API Overview

The Menus API is a set of interfaces that allows a plug-in to create and manipulate menus in the CodeWarrior IDE. The API uses the standard COM interface.

Menu can either be created at IDE initialization or during run time. Menus should be created at IDE launch via the `ICodeWarriorMenuManager` interface while dynamic menus should be created with the `ICodeWarriorMenu` interface.

Run time manipulation of menus (`ICodeWarriorMenu` and `ICodeWarriorMenuHandler`) is not yet implemented.

## Using the Menus API

The header file, `CodeWarriorMenuManager.h` defines all the interfaces for the menu API. To create a menu at launch you will need to create a command handler. See `ICodeWarriorCommandRegistry` for more information on creating a command group at IDE launch.

An example of creating a menu at launch is shown in Listing 20.1.



## Menus

### Menus API Reference

---

#### Listing 20.1 Creating a menu at launch

---

```
ICodeWarriorMenuManager *menuMgr;

servProv->QueryService(SID_SCodeWarriorMenuManager,
 IID_ICodeWarriorMenuManager, &menuMgr);
if (menuMgr)
{
 menuMgr->ShowCommandGroupMenu(kToolbarTestPluginID,
 cmdGroup_TestPlugin, true);
 menuMgr->Release();
}
```

---

## Menus API Reference

This section describes the functions contained in the following interfaces:

- ICodeWarriorMenu
- ICodeWarriorMenuHandler
- ICodeWarriorMenuManager

### ICodeWarriorMenu

This interface works with the entire menu bar, a single menu in the menu bar, or a single menu item. The methods in this interface allow you to manage menus and menu items at run time. The interface grants no access to key bindings, toolbars, or submenus.

#### Inherited Interfaces

- IUnknown

**NOTE** ICodeWarriorMenu methods work only with ICodeWarriorMenu interfaces. There is no mechanism for adding or removing an item from a built-in menu at runtime. You can add items to existing menus when CodeWarrior launches using the command registry mechanism. See “Commands API Overview” on page 493 for more information on using the command registry mechanism for creating menus.

#### Methods

This interface provides the following methods:

|                |                |
|----------------|----------------|
| InsertItem     | SetItemChecked |
| RemoveAllItems | SetItemEnabled |
| RemoveItem     | SetItemName    |



## Freescal Semiconductor, Inc.

### Menus

#### *InsertItem*

---

#### InsertItem

This method inserts a menu item on a menu.

```
virtual HRESULT InsertItem(
 BSTR inItemName,
 LONG inBeforeIndex);
```

*inItemName*

The name of the item you wish to appear in the menu.

*inBeforeIndex*

The location of the menu item you wish to insert. The new menu is placed before the value specified in *inBeforeIndex*.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### RemoveAllItems

This method removes all items from a menu.

```
virtual HRESULT RemoveAllItems(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## RemoveItem

This method removes a single item from a menu.

```
virtual HRESULT RemoveItem(LONG inItemIndex);
```

*inItemIndex*

Specifies the item index number for the menu item you want to remove.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## SetItemChecked

This method shows or hides a checkmark next to a menu item.

```
virtual HRESULT SetItemChecked(
 LONG inItemIndex,
 BOOL inNewState);
```

*inItemIndex*

The menu item that is selected.

*inNewState*

Set this paramter to true for a check mark next to the menu item or false for no checkmark.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

## Menus

### *SetItemEnabled*

---

### SetItemEnabled

Enables or disables a specified menu item.

```
virtual HRESULT SetItemEnabled(
 LONG inItemIndex,
 BOOL inNewState);
```

*inItemIndex*

This is a user-defined command ID which is assigned to this menu item when it is created. The command ID must already be registered with the IDE via `RegisterCommand`.

*inNewState*

Specifies whether the menu item is enabled or not. Set *inNewState* to `true` to enable the specified menu item or `false` to disable the menu item.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also `RegisterCommand`

---

### SetItemName

This method changes the name for a menu item.

```
virtual HRESULT SetItemName(
 LONG inItemIndex,
 BSTR inNewName);
```

*inItemIndex*

Specifies the index for the menu item whose name you want to change.

*inNewName*

The name of the menu item.



Returns
S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorMenuHandler

This interface handles events and status for menu item plug-ins. To use it, you must create a COM object that inherits from this interface.

Inherited Interfaces

- IUnknown

Methods

This interface provides the following methods:

|                     |                  |
|---------------------|------------------|
| HandleMenuSelection | UpdateMenuStatus |
|---------------------|------------------|

HandleMenuSelection

The IDE calls this method to let the plug-in know to perform the action associated with the selected menu item.

```
virtual HRESULT HandleMenuSelection(
 long inItemIndex,
 BSTR inItemName);
```

inItemIndex

The item number of the menu item whose event you want to dispatch.

inItemName

The name of the menu item.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

UpdateMenuStatus

This method updates a menu item.

```
virtual HRESULT UpdateMenuStatus(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorMenuManager

This interface allows you to obtain the interface for the CodeWarrior menu bar.

Inherited Interfaces

- IUnknown

Methods

This interface provides the following methods:

|                      |                      |
|----------------------|----------------------|
| CreateTemporaryMenu  | SetMenusEnabledState |
| GetMenusEnabledState | ShowCommandGroupMenu |

CreateTemporaryMenu

This method creates a temporary menu.

```
virtual HRESULT CreateTemporaryMenu(
 BSTR inMenuTitle,
 ICodeWarriorMenuHandler *inHandler,
 ICodeWarriorMenu *&outMenuInterface) = 0;
```

inMenuTitle

The title of the menu you want to create.

inHandler

A pointer to the ICodeWarriorMenuHandler object you are referring to.

outMenuInterface

A reference to a pointer of the ICodeWarriorMenu object.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorMenuHandler" on page 610  
"ICodeWarriorMenu" on page 605



---

## GetMenuEnabledState

This method gets whether the menu bar is enabled or disabled.

```
virtual BOOL GetMenuEnabledState() = 0;
```

Returns true if the menu bar is enabled or false if not.

---

## SetMenuEnabledState

This method enables or disables the menu bar.

```
virtual HRESULT SetMenuEnabledState(
 BOOL inEnableMenus) = 0;
```

inEnabledMenus

Set this parameter to true to enable the menu bar or false to disable it.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Menus

#### *ShowCommandGroupMenu*

---

---

### ShowCommandGroupMenu

This method shows the menu you have created through the methods in the ICodeWarriorMenu interface. To make the menu appear, you first have to register it, by calling the `IServiceProvider::QueryInterface()` method with the ICodeWarriorMenuManager object.

```
virtual HRESULT ShowCommandGroupMenu(
 CWPluginID inPluginID,
 CWCommandGroupID inCommandGroup,
 BOOL inShow) = 0;
```

`inPluginID`

The ID number of the plug-in you are creating.

`inCommandGroup`

The menu group in which your menu items are to receive commands. You must create a command group first.

`inShow`

Set this parameter to `true` to show the menu or `false` to hide the menu item referred to in the `inCommandGroup` parameter.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also "CreateNewCommandGroup" on page 502

# Messages

---

This chapter shows how to use the Messages API to create and manage messages in the CodeWarrior IDE.

This chapter contains the following sections:

- Messages API Overview
- Messages API Reference

## Messages API Overview

The Messages API is a set of interfaces that allows a plug-in to create and manipulate access paths in the CodeWarrior IDE.

## Messages API Reference

This section describes the methods contained in the following interfaces:

- `ICodeWarriorBuildMessages`
- `ICodeWarriorMessage`

The following data type is used with these interfaces:

- `EMsgType`



ICodeWarriorBuildMessages

This interface allows you to examine errors generated by the CodeWarrior IDE while it builds a project.

Inherited Interfaces

- IUnknown

Properties

This interface has the following properties:

|                     |                      |
|---------------------|----------------------|
| get_DefinitionCount | get_InformationCount |
| get_Definitions     | get_Informations     |
| get_ErrorCount      | get_WarningCount     |
| get_Errors          | get_Warnings         |

get\_DefinitionCount

This method gets the number of definitions generated during the last build.

```
virtual HRESULT get_DefinitionCount(
 long *count) = 0;
```

count

On return, this parameter contains a pointer to the number of definitions generated during the last build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

get\_Definitions

This method gets a collection of definitions generated during the last build.

```
virtual HRESULT get_Definitions(
 ICodeWarriorMessageCollection **errors) = 0;

errors
 On return, this parameter contains a collection of all definitions
 generated during the last build.
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

---

get\_ErrorCount

Use this method to obtain the number of errors generated during the last build.

```
virtual HRESULT get_ErrorCount(
 long *count) = 0;

count
 Supply a pointer to a long. Upon return it contains the number
 of errors generated during the last build.
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Messages**  
*get\_Errors*

---

**get\_Errors**

This method gets a collection of errors generated during the last build.

```
virtual HRESULT get_Errors(
 ICodeWarriorMessageCollection **errors) = 0;
```

errors

On return, this parameter contains the address of a pointer to a collection of all errors generated during the last build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

**get\_InformationCount**

This method gets the number of information items generated during the last build.

```
virtual HRESULT get_InformationCount(
 long *count) = 0;
```

count

Supply a pointer to a long. Upon return it contains the number of errors generated during the last build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



get\_Information

This method gets a collection of information items generated during the last build.

```
virtual HRESULT get_Information(
 ICodeWarriorMessageCollection **info) = 0;
```

info

On return, this parameter contains the address of a pointer to a collection of all information items generated during the last build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

get\_WarningCount

This method gets the number of warnings generated during the last build.

```
virtual HRESULT get_WarningCount(long *count) = 0;
```

count

On return, this parameter contains a pointer to the number of warnings generated during the last build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Messages**  
*get\_Warnings*

---

---

**get\_Warnings**

Use this method to obtain a collection of warnings generated during the last build.

```
virtual HRESULT get_Warnings(
 ICodeWarriorMessageCollection **warnings) = 0;
```

warnings

On return, this parameter contains the address of a pointer to a collection of all warnings generated during the last build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

ICodeWarriorMessage

This interface defines a message used by the CodeWarrior IDE.

Inherited Interfaces

- IUnknown

Properties

This interface contains the following properties:

|                      |                      |
|----------------------|----------------------|
| get_ErrorNumber      | get_SourceLineNumber |
| get_FileSpec         | get_SourceOffset     |
| get_MessageLength    | get_Target           |
| get_MessageLineCount | get_TokenLength      |
| get_MessageText      | get_TokenOffset      |
| get_ProjectFile      | get_Type             |
| get_SourceLength     |                      |

get\_ErrorNumber

This method gets the error number of the most recent error.

```
virtual HRESULT get_ErrorNumber(
 long *errorNumber) = 0;
```

errorNumber

On return, this parameter contains a pointer to a long integer that is the error number.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Messages

### *get\_FileSpec*

---

### get\_FileSpec

This method gets the file specification associated with the current message.

```
virtual HRESULT get_FileSpec(
 IFileSpec **fileSpec) = 0;
```

fileSpec

On return, this parameter contains the address of a pointer to the file specification of the message.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599

---

### get\_MessageLength

This method gets the length of the current message, in bytes.

```
virtual HRESULT get_MessageLength(
 long *messageLength) = 0;
```

messageLength

On return, this parameter contains a pointer to a long integer that holds the length of the message, in bytes.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

## get\_MessageLineCount

This method gets the number of lines in the current message.

```
virtual HRESULT get_MessageLineCount(
 long *lineCount) = 0;
```

lineCount

On return, this parameter contains a pointer to a long integer that holds the number of lines in the message.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_MessageText

This message returns a string containing the text of the current message.

```
virtual HRESULT get_MessageText(
 BSTR *message) = 0;
```

message

A string containing the text of the current message.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Messages

#### *get\_ProjectFile*

---

#### get\_ProjectFile

This method gets the project file of the project associated with the current message.

```
virtual HRESULT get_ProjectFile(
 ICodeWarriorProjectFile **projectFile) = 0;
```

projectFile

On return, this parameter contains the address of a pointer to the project file object.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectFile" on page 659

---

#### get\_SourceLength

This method gets the length of the source file that the project was manipulating when it generated the current message.

```
virtual HRESULT get_SourceLength(
 long *length) = 0;
```

length

On return, this parameter contains a pointer to a long integer containing the length of the source file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Messages

*get\_SourceLineNumber*

---

### get\_SourceLineNumber

This method gets the line number within the source file where a problem (or other event) caused a message to be created.

```
virtual HRESULT get_SourceLineNumber(
 long *lineNumber) = 0;
```

lineNumber

On return, this parameter contains a pointer to a long integer containing the line number within the source file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_SourceOffset

This method gets the number of characters, from the beginning of the source file, to the start of the keyword or phrase that caused the message to be created.

```
virtual HRESULT get_SourceOffset(
 long *offset) = 0;
```

offset

On return, this parameter contains a pointer to a long integer containing the number of characters, from the beginning of the line, to the start of the keyword in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Messages**

*get\_Target*

---

**get\_Target**

This method gets the keyword or phrase that caused the message to be created.

```
virtual HRESULT get_Target(
 ICodeWarriorTarget **target) = 0;
```

target

On return, this parameter contains the address of a pointer to the keyword or phrase that caused the message to be created.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorTarget” on page 706

---

**get\_TokenLength**

This method gets the length (the number of characters) of the keyword or phrase that caused the message to be created.

```
virtual HRESULT get_TokenLength(
 long *tokenLength) = 0;
```

tokenLength

On return, this parameter contains a pointer to a long integer containing the number of characters in the keyword or phrase that caused the message.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## get\_TokenOffset

This method gets the number of characters, from the beginning of the line, to the start of the keyword or phrase that caused the message to be created.

```

virtual HRESULT get_TokenOffset(
 long *tokenOffset) = 0;

```

tokenOffset

On return, this parameter contains a pointer to a long integer with the number of characters from the beginning of the line to the keyword or phrase that caused a message to be created.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_Type

This method gets the type of the message.

```

virtual HRESULT get_Type(
 EMsgType *type) = 0;

```

type

On return, this parameter contains a pointer to the message type.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "EMsgType" on page 628



Message Data Types

The following enumeration defines the message types used in the CodeWarrior Message API.

EMsgType

This enumeration is used to define messages created by the build process. It is used by the get\_Type method in the ICodeWarriorMessage interface.

| Constant        | Description                          |
|-----------------|--------------------------------------|
| typeNotDefined  | This message type is undefined.      |
| typeInformation | This message is an information item. |
| typeWarning     | This message is a warning.           |
| typeError       | This message describes an error.     |
| typeDefinition  | This message is a definition.        |

# Projects

---

This chapter shows how to use the Projects API to create and manage projects using plug-in interfaces.

This chapter contains the following sections:

- Projects API Overview
- Projects API Reference

## Projects API Overview

The Projects API is a set of interfaces that allows a plug-in to work with projects in the CodeWarrior IDE.

## Projects API Reference

This section describes the functions contained in the following interfaces:

- `ICodeWarriorProject`
- `ICodeWarriorProjectAssociation`
- `ICodeWarriorProjectEvents`
- `ICodeWarriorProjectFile`

These interfaces use the following data types:

- `EPluginDataStorageLoc`
- `ECodeWarriorCompileChoice`
- `ECodeWarriorRunMode`
- `ECodeWarriorBuildOptions`



ICodeWarriorProject

This interface defines a CodeWarrior project.

Inherited Interfaces

- IUnknown

Methods

This interface provides the following methods:

|                                   |                             |
|-----------------------------------|-----------------------------|
| Build                             | get_FileSpec                |
| BuildWithOptions                  | get_IsVisible               |
| BuildAndWaitToComplete            | get_Name                    |
| BuildAndWaitToCompleteWithOptions | GetNamedPluginData          |
| Close                             | get_Targets                 |
| CloneTarget                       | get_VersionControl          |
| CompileFilesWithChoice            | RemoveDesign                |
| CreateDesign                      | RemoveDesignByName          |
| CreateTarget                      | RemoveFile                  |
| Export                            | RemoveNamedPluginData       |
| ExportByFileSpec                  | RemoveObjectCode            |
| FindDesign                        | RemoveObjectCodeWithOptions |
| FindFileByName                    | RemoveTarget                |
| FindTarget                        | ReportMessage               |
| get_Application                   | SetCurrentTarget            |
| GetCurrentTarget                  | SetNamedPluginData          |
| get_Designs                       | SynchronizeStatus           |

Build

This method starts a build of the current project.

virtual HRESULT Build(

```
long *cookie);
```

cookie

On return, this parameter contains a unique identifier for the build process.

Returns. S\_OK if this method call succeeded or an appropriate error if it failed.

---

## BuildWithOptions

This method builds the current project with one of the options specified in the ECodeWarriorBuildOptions enumeration.

```
virtual HRESULT BuildWithOptions(
 ECodeWarriorBuildOptions options,
 ECodeWarriorRunMode runMode,
 long *cookie) = 0;
```

options

The build options to use with this build.

runmode

Whether to run the resulting program after building it and, if so, whether to run it in debug mode. The ECodeWarriorRunMode enumeration contains the constants that define this parameter.

cookie

On return, this parameter contains a unique identifier for the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorBuildOptions" on page 662

"ECodeWarriorRunMode" on page 662



## Freescal Semiconductor, Inc.

### Projects

#### *BuildAndWaitToComplete*

---

#### BuildAndWaitToComplete

This method starts a build of the current project and has the IDE wait to gather all messages from the build process.

```
virtual HRESULT BuildAndWaitToComplete(
 ICodeWarriorBuildMessages **buildMessages);
```

buildMessages

On return, this parameter contains the address of a pointer to the messages created by the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorBuildMessages" on page 616

---

#### BuildAndWaitToCompleteWithOptions

This method builds the current project with one of the options specified in the ECodeWarriorBuildOptions enumeration. This method accumulates all the messages from the build process before returning.

```
virtual HRESULT BuildAndWaitToCompleteWithOptions(
 ECodeWarriorBuildOptions options,
 ICodeWarriorBuildMessages **buildMessages
) = 0;
```

options

The build options to use with this build.

buildMessages

On return, this parameter contains the address of a pointer to the build messages created by the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also    “ECodeWarriorBuildOptions” on page 662  
              “ICodeWarriorBuildMessages” on page 616

---

## CloneTarget

This method puts a copy of a specified target into a new directory.

```
virtual HRESULT CloneTarget(
 ICodeWarriorTarget *srcTarget,
 ICodeWarriorProject *srcProject,
 BSTR inDestTargetName,
 VARIANT_BOOL fCopyFileList,
 VARIANT_BOOL fCopyTargetSettings,
 ICodeWarriorDesign *design,
 ICodeWarriorTarget **outTarget);
```

srcTarget

A pointer to the target to be cloned.

srcProject

A pointer to the project that contains the target to be cloned.

inDestTargetName

The new name for the cloned target.

fCopyFileList

Set this parameter to true to copy the source target’s files or false to create a target with no files.

fCopyTargetSettings

Set this parameter to true to copy the source target’s settings or false to create a target with default settings.

design

A pointer to the design associated with the source target.

## Projects

*Close*

outTarget

On return, this parameter contains the address of a pointer to the new target object.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTarget" on page 706  
"ICodeWarriorDesign" on page 542

## Close

This method closes the current project.

```
virtual HRESULT Close(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## CompileFilesWithChoice

This method compiles a collection of project files with one of the options specified in the ECodeWarriorCompileChoice enumeration.

```
virtual HRESULT CompileFilesWithChoice(
 ICodeWarriorProjectFileCollection *collection,
 ECodeWarriorCompileChoice compileChoice,
 long *cookie) = 0;
```

collection

A pointer to a collection of file projects to compile.

compileChoice

The kind of compilation the compiler should perform.

cookie

On return, this parameter contains a unique identifier for the build process.

ReturnsS\_OK if this method call succeeded or an appropriate error if it failed.

See Also“ECodeWarriorCompileChoice” on page 662  
“Using the Collections API” on page 487

CreateDesign

This method creates a new design within the current project

virtual HRESULT CreateDesign(  
BSTR designName,  
ICodeWarriorDesign \*\*design);

designName

The name for the new design.

design

On return, this parameter contains the address of a pointer to the new design object.

ReturnsS\_OK if this method call succeeded or an appropriate error if it failed.

See Also“ICodeWarriorDesign” on page 542

CreateTarget

This method creates a new target within the current project.

virtual HRESULT CreateTarget(  
BSTR targetName,  
BSTR linkerName,



# Freescale Semiconductor, Inc.

## Projects

Export

---

|         |                                                                                                            |
|---------|------------------------------------------------------------------------------------------------------------|
|         | <pre>ICodeWarriorDesign *design, ICodeWarriorTarget **target);</pre>                                       |
|         | <p>targetName</p> <p>The name of the new target.</p>                                                       |
|         | <p>linkerName</p> <p>The name of the linker to use to build the new target.</p>                            |
|         | <p>design</p> <p>The name of the design to associate with the new target.</p>                              |
|         | <p>target</p> <p>On return, this parameter contains the address of a pointer to the new target object.</p> |
| Returns | S_OK if this method call succeeded or an appropriate error if it failed.                                   |

---

## Export

|         |                                                                                      |
|---------|--------------------------------------------------------------------------------------|
|         | <p>This method exports the current project to a new location in the file system.</p> |
|         | <pre>virtual HRESULT Export(BSTR filePath);</pre>                                    |
|         | <p>filePath</p> <p>The full path of the new location.</p>                            |
| Returns | S_OK if this method call succeeded or an appropriate error if it failed.             |

---

## ExportByFileSpec

This method exports the current project to a file specification.

```
virtual HRESULT ExportByFileSpec(
 IFileSpec *fileSpec);
```

fileSpec

The file specification to which to export the current project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “IFileSpec” on page 599

---

## FindDesign

This method finds a particular design within the project, given the name of the design.

```
virtual HRESULT FindDesign(
 BSTR name,
 ICodeWarriorDesign **design);
```

name

The name of the design to find.

design

On return, this parameter contains the address of a pointer to the design specified by the name parameter.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorDesign” on page 542



## Freescal Semiconductor, Inc.

### Projects

#### *FindFileByName*

---

#### FindFileByName

This method finds a file within the current project by finding the file's name.

```
virtual HRESULT FindFileByName(
 BSTR fileName,
 ICodeWarriorProjectFileCollection
 **projectFiles) = 0;
```

fileName

The name of the file to find.

projectFiles

On return, this parameter contains the address of a pointer to the file specified in the name parameter.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

#### FindTarget

This method finds a particular target within the current project.

```
virtual HRESULT FindTarget(
 BSTR name,
 ICodeWarriorTarget **target);
```

name

The name of the target to find.

target

On return, this parameter contains the address of a pointer to the target specified by the name parameter.

---

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorTarget" on page 706

---

## get\_Application

This method gets a pointer to the current pointer's application object.

```
virtual HRESULT get_Application(
 ICodeWarriorApp **val);
```

val

On return, this parameter contains the address of a pointer to the application object associated with the current project.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorApp" on page 445

---

## GetCurrentTarget

This method gets a pointer to the current target within the current project.

```
virtual HRESULT GetCurrentTarget(
 ICodeWarriorTarget **target);
```

target

On return, this parameter contains the address of a pointer to the current target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorTarget" on page 706



## Freescal Semiconductor, Inc.

### Projects

#### *get\_Designs*

---

#### get\_Designs

This method gets a collection containing the designs within the current project.

```
virtual HRESULT get_Designs(
 ICodeWarriorDesignCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to a collection containing the designs within the current project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

#### get\_FileSpec

This method gets the file specification of the current project.

```
virtual HRESULT get_FileSpec(
 IFileSpec **pval);
```

pval

On return, this parameter contains the address of a pointer to the current project's file specification.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599



get\_IsVisible

This method gets whether the current project is visible in the IDE.

```
virtual HRESULT get_IsVisible(
 VARIANT_BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the project is visible or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Projects

### *get\_Name*

---

### get\_Name

This method gets the name of the current project.

```
virtual HRESULT get_Name(BSTR *pval);
```

pval

On return, this parameter contains the name of the current project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### GetNamedPluginData

This method gets plug-in data for the project, by name.

```
virtual HRESULT GetNamedPluginData(
 BSTR resourceName,
 EPluginDataStorageLoc storeIn,
 IStream **pluginData);
```

resourceName

The name of the plug-in from which to get the data.

storeIn

The location of the plug-in's data storage.

pluginData

On return, this parameter contains the address of a pointer to the plug-in data.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "EPluginDataStorageLoc" on page 663



---

## get\_Tar ets

This method gets a collection containing the tar ets within the current project.

```
virtual HRESULT get_Tar ets(
 ICodeWarriorTar etCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to a collection of tar ets.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

## get\_Ver sionControl

This method gets the version control object for the current project.

```
virtual HRESULT get_Ver sionControl(
 ICodeWarriorVer sionControl **versionControl);
```

versionControl

On return, this parameter contains the address of a pointer to the current project's version control object.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorVer sionControl" on page 804



## Freescal Semiconductor, Inc.

### Projects

#### *RemoveDesign*

---

#### RemoveDesign

This method removes a design (and possibly any designs nested within the specified design) from the project.

```
virtual HRESULT RemoveDesign(
 ICodeWarriorDesign *design,
 VARIANT_BOOL fDeleteContainedDesigns);
```

*design*

A pointer to the design to remove.

*fDeleteContainedDesigns*

---

**NOTE** *fDeleteContainedDesigns* indicates whether to delete targets, not designs.

---

Set this parameter to `true` to delete any targets contained within the design specified in the *design* parameter or `false` to leave the targets.

**Returns** `S_OK` if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorDesign" on page 542

---

#### RemoveDesignByName

This method removes a design (and possibly any designs nested within the specified design) from the project, given a design name.

```
virtual HRESULT RemoveDesignByName(
 BSTR designName,
 VARIANT_BOOL fDeleteContainedTargets);
```

*designName*

The name of the design to remove.

fDeleteContainedDesigns

**NOTE** fDeleteContainedDesigns indicates whether to delete targets, not designs.

Set this parameter to true to delete any designs contained within the design specified in the design parameter or false to leave the targets.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## RemoveFile

This method removes a specified project file.

```
virtual HRESULT RemoveFile(
 ICodeWarriorProjectFile *projectFile)

projectFile
```

A pointer to the project file to remove.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectFile" on page 659

## RemoveNamedPluginData

This method removes plug-in data, given a name for the plug-in.

```
virtual HRESULT RemoveNamedPluginData(
 BSTR resourceName,
 EPluginDataStorageLoc storeIn);
```

resourceName

The name of the plug-in from which to remove data.

## Projects

### *RemoveObjectCode*

---

`storeIn`

The location of the data.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “EPluginDataStorageLoc” on page 663

---

## RemoveObjectCode

This method removes the object code from a specified target.

```
virtual HRESULT RemoveObjectCode(
 ECodeWarriorWhichTargetOptions whichTarget,
 VARIANT_BOOL compact);
```

`whichTarget`

The target from which to remove object code.

`compact`

`true` to have the IDE destroy the associated data files (from the data folder) and re-create them or `false` to leave the data files as they are.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorWhichTargetOptions” on page 765

---

## RemoveObjectCodeWithOptions

This method removes the object code from a single target or all targets. It also lets you choose whether to remove the object code from all the subprojects within the current project and whether to delete any associated data files.

```
virtual HRESULT RemoveObjectCodeWithOptions(
```



## Freescale Semiconductor, Inc.

### Projects

*RemoveObjectCodeWithOptions*

---

```
ECodeWarriorWhichTargetOptions whichTarget,
VARIANT_BOOL recurseSubProject,
VARIANT_BOOL deleteDataFiles) = 0;
```

whichTarget

A value with the range defined by the ECodeWarriorWhichTargetOptions enumeration that specifies whether to remove the object code from all targets or only the current target.

## Projects

### *RemoveTarget*

---

`recurseSubProject`

Set this parameter to `true` to remove the object code within all the subprojects that match the `whichTarget` parameter or `false` to leave the object code in the subprojects.

`deleteDataFiles`

Set this parameter to `true` to delete data files associated with the current project or `false` to retain the data files.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also “`ECodeWarriorWhichTargetOptions`” on page 765

---

## RemoveTarget

This method removes the specified target from the current project.

```
virtual HRESULT RemoveTarget(
 ICodeWarriorTarget *target);
```

`target`

A pointer to the target to remove.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also “`ICodeWarriorTarget`” on page 706

## ReportMessage

This method makes the specified message appear in the IDE's message window.

```
virtual HRESULT ReportMessage(
 EReportMsgType msgType,
 BSTR message);
```

msgType

The type of message (information, warning, etc.) to report.

message

The message to report.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Message Data Types" on page 628

---

## SetCurrentTarget

This method sets the current build target within the project.

```
virtual HRESULT SetCurrentTarget(
 BSTR targetName);
```

targetName

The name of the target to set as the current target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

## Projects

*SetNamedPluginData*

---

### SetNamedPluginData

---

This method sets the data for a plug-in.

```
virtual HRESULT SetNamedPluginData(
 BSTR resourceName,
 EPluginDataStorageLoc storeIn,
 IStream *pluginData);
```

**resourceName**

The name of the plug-in.

**storeIn**

The location in which to store the plug-in's data.

**pluginData**

The data to store in the plug-in.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "EPluginDataStorageLoc" on page 663

---

### SynchronizeStatus

This method synchronizes the dates of the files stored in a version control system with the dates of the working files. For this method to work, you must have defined the VCS settings either globally or for the project.

```
virtual HRESULT SynchronizeStatus(void);
```

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorProjectAssociation

This interface provides access to the project associated with the current project file.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|             |             |
|-------------|-------------|
| get_Project | put_Project |
|-------------|-------------|

get\_Project

The IDE calls this method to get the project associated with the current project file.

```
virtual HRESULT get_Project(
 ICodeWarriorProject **pval);
```

pval

On return, this parameter contains the address of a pointer to the project associated with the current project file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630



**Projects**

*put\_Project*

---

---

**put\_Project**

The IDE calls this method to associate a new project object with the current project file.

```
virtual HRESULT put_Project(
 ICodeWarriorProject *pval);
```

*pval*

A pointer to the project to associate with the current project file.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProject" on page 630



ICodeWarriorProjectEvents

This interface provides a way to create events while a user works with a project.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                |                   |
|----------------|-------------------|
| BuildEnded     | QueryAboutToBuild |
| BuildStarted   | QueryDeleteDesign |
| DeletingDesign | QueryUIClose      |
| DesignCreated  | RevertCompleted   |
| ProjectClosing | VisibleChanged    |

BuildEnded

This method indicates that a build has ended.

```
virtual HRESULT BuildEnded(
 ECodeWarriorCompileChoice choice,
 long buildID,
 VARIANT_BOOL fBuildSucceeded,
 ICodeWarriorBuildMessages *buildMessages);

choice
 The kind of operation the compile performed.

buildID
 The ID of the build.

fBuildSucceeded
 Set this parameter to true if the build succeeded or false
 otherwise.
```

## Projects

### *BuildStarted*

---

`buildMessages`

A pointer to the messages generated by the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorCompileChoice” on page 662  
 “ICodeWarriorBuildMessages” on page 616

---

### BuildStarted

This method indicates that a build has started.

```
virtual HRESULT BuildStarted(
 ECodeWarriorCompileChoice choice,
 long buildID,
 ICodeWarriorTargetCollection *targetList);
```

`choice`

The kind of operation the compile performed.

`buildID`

The ID of the build.

`targetList`

A pointer to the collection of targets to build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorCompileChoice” on page 662  
 “Using the Collections API” on page 487

---

---

### DeletingDesign

This method indicates that a design is being deleted.

```
virtual HRESULT DeletingDesign(
 ICodeWarriorDesign *design);
```

design

A pointer to the design to delete.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “Using the Collections API” on page 487

---

### DesignCreated

This method indicates that a design is being created.

```
virtual HRESULT DesignCreated(
 ICodeWarriorDesign *design);
```

design

A pointer to the design to create.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “Using the Collections API” on page 487



## Freescal Semiconductor, Inc.

### Projects

#### *ProjectClosing*

---

#### ProjectClosing

This method indicates that a project is being closed.

```
virtual HRESULT ProjectClosing(
 ICodeWarriorProject *project);
```

project

A pointer to the project to close.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---

#### QueryAboutToBuild

The CodeWarrior IDE calls this method to inform the plug-in that a build is about to start.

```
virtual HRESULT QueryAboutToBuild(
 ECodeWarriorCompileChoice choice,
 long buildID,
 ICodeWarriorTargetCollection *targetList);
```

choice

The kind of operation the compile performed.

buildID

The ID of the build.

targetList

A pointer to the collection of targets to build.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorCompileChoice" on page 662

---

---

“Using the Collections API” on page 487

---

## QueryDeleteDesign

The CodeWarrior IDE calls this method to inform the plug-in that a design is about to be deleted.

```
virtual HRESULT QueryDeleteDesign(
 ICodeWarriorDesign *design);
```

design

A pointer to the design to delete.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “Using the Collections API” on page 487

---

## QueryUIClose

The CodeWarrior IDE calls this method to inform the plug-in that the IDE is about to close.

```
virtual HRESULT QueryUIClose(
 ICodeWarriorProject *project);
```

project

A pointer to the project window to close.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “ICodeWarriorProject” on page 630



**Projects**

*RevertCompleted*

---

**RevertCompleted**

---

This method indicates that a reversion (backing up to an earlier state) operation has finished.

```
virtual HRESULT RevertCompleted(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**VisibleChanged**

---

This method indicates that the project window's visibility has changed (it has been hidden or revealed).

```
virtual HRESULT VisibleChanged(
 ICodeWarriorProject *project,
 VARIANT_BOOL fVisible);
```

project

A pointer to the project in question.

fVisible

Set this parameter to true if the project is visible or false if the project is invisible.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---



ICodeWarriorProjectFile

This interface provides the means to manipulate project files..

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|              |              |
|--------------|--------------|
| CheckIn      | get_Project  |
| CheckOut     | get_Targets  |
| get_FileSpec | get_VCSState |
| get_Name     |              |

CheckIn

This method checks in the project file for the current project.

```
virtual HRESULT CheckIn(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

CheckOut

This method checks out the current project file.

```
virtual HRESULT CheckOut(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Projects

#### *get\_FileSpec*

---

#### get\_FileSpec

This method gets the file specification for the current project file.

```
virtual HRESULT get_FileSpec(IFileSpec **pval);
```

pval

On return, this parameter contains the address of a pointer to the file specification for the current project file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599

---

#### get\_Name

This method gets the name of the current project file.

```
virtual HRESULT get_Name(BSTR *pval);
```

pval

On return, this parameter contains the name of the current project file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### get\_Project

This method gets the project object for the current project.

```
virtual HRESULT get_Project(
 ICodeWarriorProject **pval);
```

---

pval

On return, this parameter contains the address of a pointer to the project object for the current project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---

## get\_Targets

This method gets the collection of targets within the current project.

```
virtual HRESULT get_Targets(
 ICodeWarriorTargetCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to the collection of target files within the current project.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

## get\_VCSState

This method gets the current version control system (VCS) status for the current project file.

```
virtual HRESULT get_VCSState(
 ICodeWarriorVCSState **pval);
```

pval

On return, this parameter contains the address of a pointer to the VCSState object for the current project.

## Projects

### Project Data Types

- Returns S\_OK if this method call succeeded or an appropriate error if it failed.
- See Also “ICodeWarriorVCSSState” on page 809

## Project Data Types

### ECodeWarriorBuildOptions

The ECodeWarriorBuildOptions enumeration provides constants for whether to build normal or to skip dependencies.

**Table 22.1 ECodeWarriorBuildOptions Enumeration**

| Constant            | Description                           |
|---------------------|---------------------------------------|
| kCWNormalBuild      | Build the application normally        |
| kCWSkipDependencies | Skip all dependencies when buuilding. |

### ECodeWarriorCompileChoice

The ECodeWarriorCompileChoice enumeration provides constants for what kind of operation the compiler performs.

**Table 22.2 ECodeWarriorCompileChoice Enumeration**

| Constant             | Description            |
|----------------------|------------------------|
| kCWChoiceCheckSyntax | Check the syntax only. |
| kCWChoicePreprocess  | Preprocess only.       |
| kCWChoicePrecompile  | Precompile only.       |
| kCWChoiceCompile     | Compile normally.      |
| kCWChoiceDisassemble | Disassemble.           |

### ECodeWarriorRunMode

The ECodeWarriorRunMode enumeration provides constants for whether to run the resulting output of a build process and whether to run it in debug mode.

Table 22.3 ECodeWarriorRunMode Enumeration

| Constant   | Description                                       |
|------------|---------------------------------------------------|
| kCWDontRun | Don't run the application after building.         |
| kCWRun     | Run the application after building.               |
| kCWDebug   | Run the application in debug mode after building. |

EPluginDataStorageLoc

The EPluginDataStorageLoc enumeration provides constants for where a plug-in's data is stored.

Table 22.4 EPluginDataStorageLoc Enumeration

| Constant                    | Description                                               |
|-----------------------------|-----------------------------------------------------------|
| kStoreInProjectFile         | Store the data in the project file.                       |
| kStoreInTargetDataFile      | Store the data in a data file associated with the target. |
| kStoreInProjectSettingsFile | Store the data in the project settings file.              |



# Freescal Semiconductor, Inc.

## Projects

*Project Data Types*

---

# Symbols

---

This chapter describes the Symbols API, which you can use to manipulate the various symbols and messages associated with the build process.

This chapter contains the following sections:

- Symbols API Reference

## Symbols API Reference

This section describes the functions contained in the following interfaces:

- `ICodeWarriorBaseClassInfo`
- `ICodeWarriorClass`
- `ICodeWarriorDataMember`
- `ICodeWarriorMethod`
- `ICodeWarriorScope`
- `ICodeWarriorSourceContext`
- `ICodeWarriorSymbol`
- `ICodeWarriorSymbolContainer`

The following data types are used with these interfaces:

- `ECodeWarriorAccess`
- `ECodeWarriorShowSymbolLocation`

## Symbols

*ICodeWarriorBaseClassInfo*

### ICodeWarriorBaseClassInfo

This interface provides methods to get information about a base class.

#### Inherited Interfaces

- IUnknown

#### Methods

The following methods are available for your use:

|               |               |
|---------------|---------------|
| get_Access    | get_IsVirtual |
| get_BaseClass |               |

#### get\_Access

This method gets the access level for the current class.

```

virtual HRESULT get_Access(
 ECodeWarriorAccess *pval);

```

pval

On return, this parameter contains a pointer to the access level.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorAccess" on page 703

#### get\_BaseClass

This method gets the base class for the current class.

```

virtual HRESULT get_BaseClass(
 ICodeWarriorClass **pval);

```



pval  
On return, this parameter contains the address of a pointer to the class.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorClass" on page 668

get\_IsVirtual

This method gets whether the base class is virtual.

```
virtual HRESULT get_IsVirtual(
 BOOL *pval);
```

pval  
On return, this parameter contains a pointer to a boolean that is set to true if the base class is virtual or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## ICodeWarriorClass

### Inherited Interfaces

- IUnknown

### Methods

The following methods are available for your use:

|                          |                      |
|--------------------------|----------------------|
| FindDataMemberByName     | get_IsFinal          |
| FindMethodByName         | get_IsPublic         |
| get_BaseClasses          | GetMethods           |
| GetDataMembers           | GetMethodsWithAccess |
| GetDataMembersWithAccess | get_SubClasses       |
| get_IsAbstract           |                      |

## FindDataMemberByName

This method finds a data member within the class, by name.

```
virtual HRESULT FindDataMemberByName (
 BSTR inName,
 ICodeWarriorDataMember **pval);
```

inName

The name of the member to find

pval

On return, this parameter contains the address of a pointer to the data member specified by the inName parameter.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorDataMember" on page 675

## FindMethodByName

This method finds a method within the class, by name.

```
virtual HRESULT FindMethodByName(
 BSTR inName,
 ICodeWarriorMethod **pval);
```

*inName*

The name of the method to find.

*pval*

On return, this parameter contains the address of a pointer to the data member specified by the *inName* parameter.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorMethod" on page 677

---

## get\_BaseClasses

This method gets a collection of base classes for the current class.

```
virtual HRESULT get_BaseClasses(
 ICodeWarriorBaseClassCollection **pval);
```

*pval*

On return, this parameter contains the address of a pointer to a collection of base classes.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

## Symbols

*GetDataMembers*

### GetDataMembers

This method gets the data members of the class.

```

virtual HRESULT GetDataMembers(
 BOOL inIncludeInherited,
 ICodeWarriorDataMemberCollection **pval);

```

inIncludeInherited

Set this parameter to true to include inherited data members or to false to exclude them.

pval

On return, this parameter contains the address of a pointer to a collection of the data members.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

### GetDataMembersWithAccess

This method gets the data members within a class that have a particular kind of access, as determined by the inAccessMask parameter.

```

virtual HRESULT GetDataMembersWithAccess(
 BOOL inIncludeInherited,
 ECodeWarriorAccess inAccessMask,
 ICodeWarriorDataMemberCollection **pval);

```

inIncludeInherited

Set this parameter to true to include inherited data members or to false to exclude them.

inAccessMask

Set this parameter to one of the constants defined in the

|         |                                                                                                                                                                                                                |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         | <div> <div>ECodeWarriorAccess enumeration.</div> <div>pval</div> <div>On return, this parameter contains the address of a pointer to a collection of the data members that match the access mask.</div> </div> |
| Returns | <div>S_OK if this method call succeeded or an appropriate error if it failed.</div>                                                                                                                            |

## get\_IsAbstract

|         |                                                                                                                                                                                                                                                                           |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         | <div> <div>This method gets whether the class is abstract.</div> <div> virtual HRESULT get_IsAbstract(<br/> BOOL *pval); </div> <div>pval</div> <div>On return, this parameter contains a pointer to a boolean that indicates whether the class is abstract.</div> </div> |
| Returns | <div>S_OK if this method call succeeded or an appropriate error if it failed.</div>                                                                                                                                                                                       |

## get\_IsFinal

|         |                                                                                                                                                                                                                                                                  |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         | <div> <div>This method gets whether the class is final.</div> <div> virtual HRESULT get_IsFinal(<br/> BOOL *pval); </div> <div>pval</div> <div>On return, this parameter contains a pointer to a boolean that indicates whether the class is final.</div> </div> |
| Returns | <div>S_OK if this method call succeeded or an appropriate error if it failed.</div>                                                                                                                                                                              |

## Symbols

*get\_IsPublic*

---

### get\_IsPublic

This method gets whether the class is public.

```
virtual HRESULT get_IsPublic(
 BOOL *pval);
```

*pval*

On return, this parameter contains a pointer to a boolean that indicates whether the class is public.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### GetMethods

This method gets the methods within a class.

```
virtual HRESULT GetMethods(
 BOOL inIncludeInherited,
 ICodeWarriorMethodCollection **pval);
```

*inIncludeInherited*

Set this parameter to true to include inherited methods or to false to exclude them.

*pval*

On return, this parameter contains the address of a pointer to a collection of the methods.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

---

**GetMethodsWithAccess**

This method gets the data members within a class that have a particular kind of access, as determined by the `inAccessMask` parameter.

```
virtual HRESULT GetMethodsWithAccess(
 BOOL inIncludeInherited,
 ECodeWarriorAccess inAccessMask,
 ICodeWarriorMethodCollection **pval);
```

`inIncludeInherited`

Set this parameter to `true` to include inherited members or to `false` to exclude them.

`inAccessMask`

Set this parameter to one of the constants defined in the `ECodeWarriorAccess` enumeration.

`pval`

On return, this parameter contains the address of a pointer to a collection of the methods that match the access mask.

**Returns**     `S_OK` if this method call succeeded or an appropriate error if it failed.

**See Also**     “Using the Collections API” on page 487



**Symbols**

*get\_SubClasses*

---

---

**get\_SubClasses**

This method gets a collection containing the subclasses of the class.

```
virtual HRESULT get_SubClasses(
 ICodeWarriorClassCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to a collection of the current class's subclasses.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487



ICodeWarriorDataMember

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|             |                 |
|-------------|-----------------|
| get_IsFinal | get_IsTransient |
|-------------|-----------------|

get\_IsFinal

This method gets whether the current data member is final.

```
virtual HRESULT get_IsFinal(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the data member is final or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Symbols**

*get\_IsTransient*

---

---

**get\_IsTransient**

This method gets whether the current data member is transient.

```
virtual HRESULT get_IsTransient(
 BOOL *pval);
```

*pval*

On return, this parameter contains a pointer to a boolean that is set to `true` if the data member is transient or `false` if not.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

ICodeWarriorMethod

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                   |                    |
|-------------------|--------------------|
| get_IsAbstract    | get_IsNative       |
| get_IsConst       | get_IsStatic       |
| get_IsConstructor | get_IsSynchronized |
| get_IsDestructor  | get_IsVirtual      |

get\_Access

This method gets the access restriction for the current method.

```
virtual HRESULT get_Access(
 ECodeWarriorAccess *pval);
```

pval

On return, this pointer contains a pointer to the access restriction for the current method.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorAccess” on page 703

## Symbols

*get\_IsAbstract*

---

### get\_IsAbstract

This method gets whether the current method is abstract.

```
virtual HRESULT get_IsAbstract(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to true if the method is abstract or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_IsConst

This method gets whether the current method has been declared as constant.

```
virtual HRESULT get_IsConst(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to true if the method is abstract or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_IsConstructor

This method gets whether the current method is one of a class's constructors.

```
virtual HRESULT get_IsConstructor(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the method is a constructor or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_IsDestructor

This method gets whether the current method is one of a class's destructors.

```
virtual HRESULT get_IsDestructor(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the method is a destructor or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescal Semiconductor, Inc.

## Symbols

*get\_IsInline*

---

### get\_IsInline

This method gets whether the current method is inline.

```
virtual HRESULT get_IsInline(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to true if the method is inline or false if not.

Returns    Nothing

---

### get\_IsNative

This method gets whether the current method is native.

```
virtual HRESULT get_IsNative(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to true if the method is native or false if not.

Returns    S\_OK if this method call succeeded or an appropriate error if it failed.

---

---

## get\_IsStatic

This method gets whether the current method is static.

```
virtual HRESULT get_IsStatic(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the method is static or `false` if not.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_IsSynchronized

This method gets whether the current method is synchronized.

```
virtual HRESULT get_IsSynchronized(
 BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the method is synchronized or `false` if not.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



**Symbols**

*get\_IsVirtual*

---

**get\_IsVirtual**

This method gets whether the current method is virtual.

```
virtual HRESULT get_IsVirtual(
 BOOL *pval);
```

*pval*

On return, this parameter contains a pointer to a boolean that is set to `true` if the method is virtual or `false` if not.

Returns    `S_OK` if this method call succeeded or an appropriate error if it failed.

## ICodeWarriorScope

### Inherited Interfaces

- ICodeWarriorSymbol

### Methods

The following methods are available for your use:

|             |             |
|-------------|-------------|
| get_Symbols | FindSymbols |
|-------------|-------------|

### get\_Symbols

This method gets the symbols within the current scope that match a particular symbol type and a particular access mask. Optionally, this method can get the symbols from parent scopes inherited by this scope.

```
virtual HRESULT get_Symbols(
 ECodeWarriorSymbolType inSymbolTypeMask,
 ECodeWarriorAccess inAccessMask,
 BOOL inIncludeInherited,
 ICodeWarriorSymbolCollection** pval) = 0;
```

**inSymbolTypeMask**  
One of the constants from the ECodeWarriorSymbolType enumeration.

**inAccessMask**  
One of the constants defined in the ECodeWarriorAccess enumeration.

**inIncludeInherited**  
true to include matching symbols from inherited scopes or to false to exclude them.

## Symbols

### FindSymbols

pval

On return, this parameter contains the address of a pointer to a collection object containing the specified symbols.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487  
 “ECodeWarriorSymbolType” on page 704  
 “ECodeWarriorAccess” on page 703

## FindSymbols

This method finds symbols, given a search string, a mode, a mask, and a switch for whether to include the symbols from inherited scopes. The resulting symbols return in a symbol collection object that you specify when you call this method.

```
virtual HRESULT FindSymbols(
 BSTR inFindString,
 ECodeWarriorFindSymbolsMode inFindSymbolsMode,
 ECodeWarriorSymbolType inSymbolTypeMask,
 ECodeWarriorAccess inAccessMask,
 BOOL inIncludeInherited,
 ICodeWarriorSymbolCollection** outSymbols) = 0;
```

inFindString

A string that specifies which symbols to find.

inFindSymbolMode

One of the constants from the  
 ECodeWarriorFindSymbolsMode enumeration.

inSymbolTypeMask

One of the constants from the ECodeWarriorSymbolType  
 enumeration.



inIncludeInherited

true to include matching symbols from inherited scopes or to  
false to exclude them.

outSymbols

On return, this parameter contains the address of a pointer to a  
symbol collection object that contains the symbol objects  
returned by this method.

Returns S\_OK if this method call succeeded or an appropriate error if it  
failed.

See Also “Using the Collections API” on page 487  
“ECodeWarriorFindSymbolsMode” on page 703  
“ECodeWarriorSymbolType” on page 704

**Symbols**

*ICodeWarriorSourceContext*

**ICodeWarriorSourceContext**

**Inherited Interfaces**

- IUnknown

**Methods**

The following methods are available for your use:

|                 |                 |
|-----------------|-----------------|
| get_EndOffset   | put_EndOffset   |
| get_FileSpec    | put_FileSpec    |
| get_IsDefined   | put_StartOffset |
| get_StartOffset |                 |

**get\_EndOffset**

This method gets the number of characters from the top of a source file to the end of a specified symbol (usually one identified by the compiler as problematic).

```
virtual HRESULT get_EndOffset(
 long *pval)
```

pval

On return, this parameter contains a pointer to a long integer that indicates how many characters from the top of the file to the end of the symbol in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_FileSpec

This method gets the file specification for a specified symbol (usually one identified by the compiler as problematic).

```
virtual HRESULT get_FileSpec(
 IFileSpec **pval)
```

pval

On return, this parameter contains the address of a pointer to the file specification for the symbol in question.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "IFileSpec" on page 599

---

## get\_IsDefined

This method indicates whether the symbol in question has been defined.

```
virtual HRESULT get_IsDefined(
 BOOL *pval)
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if the symbol in question has been defined or `false` if not.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

## Symbols

*get\_StartOffset*

---

### get\_StartOffset

This method gets the number of characters from the top of a source file to the start of a specified symbol (usually one identified by the compiler as problematic).

```
virtual HRESULT get_StartOffset(
 long *pval)
```

pval

On return, this parameter contains a pointer to a long integer that indicates how many characters from the top of the file to the start of the symbol in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### put\_EndOffset

This method sets the number of characters to the end of a symbol in a source file.

```
virtual HRESULT put_EndOffset(
 long pval)
```

pval

The number of characters from the top of the file to the end of the symbol in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

---

## put\_FileSpec

This method sets the file specification for a symbol.

```
virtual HRESULT put_FileSpec(
 IFileSpec *pval)
```

pval

A pointer to a file specification.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "IFileSpec" on page 599

---

## put\_StartOffset

This method sets the number of characters to the start of a symbol in a source file.

```
virtual HRESULT put_StartOffset(
 long pval)
```

pval

The number of characters from the top of the file to the start of the symbol in question.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



**Symbols**  
*ICodeWarriorSymbol*

**ICodeWarriorSymbol**

**Inherited Interfaces**

- IUnknown

**Methods**

The following methods are available for your use:

|                         |                        |
|-------------------------|------------------------|
| get_Class               | get_DefinitionLocation |
| get_Container           | get_Name               |
| get_DeclarationLocation | get_SimpleName         |

**get\_Class**

This method gets the class in which a symbol appears

```
virtual HRESULT get_Class(
 ICodeWarriorClass **pval);
```

pval

On return, this parameter contains the address of a pointer to the class containing the symbol in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorClass” on page 668

---

## get\_Container

This method gets the container for a symbol.

```
virtual HRESULT get_Container(
 ICodeWarriorSymbolContainer **pval);
```

pval

On return, this parameter contains the address of a pointer to the container that holds the symbol in question.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorSymbolContainer" on page 694

---

## get\_DeclarationLocation

This method gets the location where the symbol was declared.

```
virtual HRESULT get_DeclarationLocation(
 ICodeWarriorSourceContext **pval);
```

pval

On return, this parameter contains the address of a pointer to the source context where the symbol in question was declared.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorSourceContext" on page 686

## Symbols

*get\_DefinitionLocation*

---

### get\_DefinitionLocation

This method gets the location where the symbol was defined.

```
virtual HRESULT get_DefinitionLocation(
 ICodeWarriorSourceContext **pval);
```

pval

On return, this parameter contains the address of a pointer to the source context where the symbol in question was defined.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorSourceContext" on page 686

---

### get\_Name

This method gets the fully qualified name of the symbol.

```
virtual HRESULT get_Name(
 BSTR *pval);
```

pval

On return, this parameter contains the name of the symbol in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_SimpleName

This method gets the simple name of the symbol.

```
virtual HRESULT get_SimpleName(
 BSTR *pval);
```

---



pval  
On return, this parameter contains the simple name of the symbol in question.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



Symbols

ICodeWarriorSymbolContainer

ICodeWarriorSymbolContainer

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                           |                           |
|---------------------------|---------------------------|
| AddComponentAttachment    | get_GlobalScope           |
| FindClass                 | get_Symbols               |
| FindClassInFile           | get_Target                |
| FindSymbolBySourceContext | RemoveComponentAttachment |
| FindSymbols               | ShowSymbolDeclaration     |
| get_ClassList             | ShowSymbolDefinition      |
| get_ExtraDataDescriptors  |                           |

AddComponentAttachment

This method lets you add a component attachment to a symbol container.

```
virtual HRESULT AddComponentAttachment (
 CLSID *attachmentCLSID);
```

attachmentCLSID

A pointer to the component attachment to add.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## FindClass

This method finds a class, given its name.

```
virtual HRESULT FindClass(
 BSTR inClassName,
 ICodeWarriorClass **outClass);
```

*inClassName*

The name of the class to find.

*outClass*

On return, this parameter contains the address of a pointer to the class.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorClass" on page 668

---

## FindClassInFile

This method finds a file within a class, given the class's name and a file specification.

```
virtual HRESULT FindClassInFile(
 BSTR inClassName,
 IFileSpec *inSpec,
 ICodeWarriorClass **outClass);
```

*inClassName*

The name of the class.

*inSpec*

A pointer to a file specification.

## Symbols

### *FindSymbolBySourceContext*

---

outClass

On return, this parameter contains the address of a pointer to the class.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599  
"ICodeWarriorClass" on page 668

---

## FindSymbolBySourceContext

This method finds a debugging symbol, given an object that specifies the source context of the symbol.

```
virtual HRESULT FindSymbolBySourceContext(
 ICodeWarriorSourceContext* inSourceContext,
 ICodeWarriorSymbol** outSymbol) = 0;
```

inSourceContext

The source context object for which to find a debugging symbol.

outSymbol

On return, this parameter contains a pointer to the address of the specified debugging symbol.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorSourceContext" on page 686

---

## FindSymbols

This method finds symbols, given a search string, a mode, a mask, and a switch for whether to include child containers. The resulting

symbols return in a symbol collection object that you specify when you call this method.

```
virtual HRESULT FindSymbols(
 BSTR inFindString,
 ECodeWarriorFindSymbolsMode inFindSymbolsMode,
 ECodeWarriorSymbolType inSymbolTypeMask,
 BOOL inIncludeChildContainers,
 ICodeWarriorSymbolCollection** outSymbols) = 0;
```

*inFindString*

A string that specifies which symbols to find.

*inFindSymbolMode*

One of the constants from the  
ECodeWarriorFindSymbolsMode enumeration.

*inSymbolTypeMask*

One of the constants from the ECodeWarriorSymbolType  
enumeration.

*inIncludeChildContainers*

true to search child symbol containers for the specified  
symbols or false to search only the current symbol container.

*outSymbols*

On return, this parameter contains the address of a pointer to a  
symbol collection object that contains the symbol objects  
returned by this method.

**Returns** S\_OK if this method call succeeded or an appropriate error if it  
failed.

**See Also** "Using the Collections API" on page 487

"ECodeWarriorFindSymbolsMode" on page 703

"ECodeWarriorSymbolType" on page 704

## Symbols

*get\_ClassList*

---

### get\_ClassList

This method gets a list of the classes referenced by the symbol container.

```
virtual HRESULT get_ClassList(
 ICodeWarriorClassCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to the class list.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

### get\_ExtraDataDescriptors

This method gets a collection of strings that further define the symbols within this container.

```
virtual HRESULT get_ExtraDataDescriptors(
 IBSTRCollection** outExtraDataDescriptors) = 0;
```

outExtraDataDescriptors

On return, this parameter contains the address of a pointer to a collection of strings that further define the symbols within this container.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## get\_GlobalScope

This method gets a pointer to the global scope object.

```
virtual HRESULT get_GlobalScope(
 ICodeWarriorScope** pval) = 0;
```

*pval*

On return, this parameter contains a pointer to the address of a scope object that defines the global scope.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorScope" on page 683

---

## get\_Symbols

This method gets a symbol container object containing all the symbol objects that match a given mode and mask. Optionally, this method can include matching symbols from child symbol containers, as well as the current symbol container.

```
virtual HRESULT get_Symbols(
 ECodeWarriorFindSymbolsMode inFindSymbolsMode,
 ECodeWarriorSymbolType inSymbolTypeMask,
 BOOL inIncludeChildContainers,
 ICodeWarriorSymbolCollection** outSymbols) = 0;
```

*inFindSymbolMode*

One of the constants from the ECodeWarriorFindSymbolsMode enumeration.

*inSymbolTypeMask*

One of the constants from the ECodeWarriorSymbolType enumeration.

## Symbols

### *get\_Target*

---

`inIncludeChildContainers`

`true` to search child symbol containers for the specified symbols or `false` to search only the current symbol container.

`outSymbols`

On return, this parameter contains the address of a pointer to a symbol collection object that contains the symbol objects returned by this method.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487  
 "ECodeWarriorFindSymbolsMode" on page 703  
 "ECodeWarriorSymbolType" on page 704

---

### *get\_Target*

This method gets the target associated with the symbol container (that is, the target for which the build process generated the symbols in the symbol container.

```
virtual HRESULT get_Target(
 ICodeWarriorTarget **pval);
```

`pval`

On return, this parameter contains the address of a pointer to the target associated with the current symbol container.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTarget" on page 706

---

## RemoveComponentAttachment

This method removes a component attachment from the symbol container.

```
virtual HRESULT RemoveComponentAttachment(
 CLSID *attachmentCLSID);
```

attachmentCLSID

A pointer to the component attachment to remove.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## ShowSymbolDeclaration

This method shows the user where a specified symbol is located. It may also let the user edit the symbol in that location.

```
virtual HRESULT ShowSymbolDeclaration(
 ICodeWarriorSymbol *inSymbol,
 BOOL inForEditing,
 ECodeWarriorShowSymbolLocation inLocation);
```

inSymbol

The symbol to show the user.

inForEditing

Set this parameter to true to let the user edit the line on which the symbol appears or false to prevent editing.

inLocation

Where to show the symbol.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorSymbol" on page 690

## Symbols

### ShowSymbolDefinition

---

“ECodeWarriorShowSymbolLocation” on page 704

---

### ShowSymbolDefinition

This method shows where the symbol in question is defined.

```
virtual HRESULT ShowSymbolDefinition(
 ICodeWarriorSymbol *inSymbol,
 BOOL inForEditing,
 ECodeWarriorShowSymbolLocation inLocation);
```

*inSymbol*

The symbol to show the user.

*inForEditing*

Set this parameter to `true` to let the user edit the line on which the symbol appears or `false` to prevent editing.

*inLocation*

Where to show the symbol.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “ICodeWarriorSymbol” on page 690

“ECodeWarriorShowSymbolLocation” on page 704

## Symbols Data Types

### ECodeWarriorAccess

The ECodeWarriorAccess enumeration provides constants for member access levels.

| Constant         | Definition                                            |
|------------------|-------------------------------------------------------|
| kAccessNone      | No Access.                                            |
| kPublicAccess    | Access to public members.                             |
| kProtectedAccess | Access to protected (and public) members.             |
| kPrivateAccess   | Access to private (and protected and public members). |
| kAccessAll       | Access to all members.                                |

### ECodeWarriorFindSymbolsMode

The ECodeWarriorFindSymbolsMode enumeration provides constants for controlling the output of methods that find debugging symbols.

| Constant                | Definition                    |
|-------------------------|-------------------------------|
| kFindSymbolsByName      | Specify symbols by name.      |
| kFindSymbolsByPrefix    | Specify symbols by prefix.    |
| kFindSymbolsBySubstring | Specify symbols by substring. |

## Symbols

### Symbols Data Types

#### ECodeWarriorShowSymbolLocation

The ECodeWarriorAccess enumeration provides constants for where a symbol can be shown.

| Constant             | Definition                                                      |
|----------------------|-----------------------------------------------------------------|
| kShowInEditor        | The symbol can be shown in the editor.                          |
| kShowInBrowser       | The symbol can be shown in the symbol browser.                  |
| kUsePreferenceToShow | The symbol conforms to user settings for where to show symbols. |

#### ECodeWarriorSymbolType

The ECodeWarriorSymbolType enumeration provides constants for the types of symbols that the various find functions should return.

| Constant                   | Definition                        |
|----------------------------|-----------------------------------|
| kSymbolTypeNone            | Specifies no symbol.              |
| kSymbolTypeMacro           | Specifies macros.                 |
| kSymbolTypeEnum            | Specifies enumerations.           |
| kSymbolTypeTypedef         | Specifies typedefs                |
| kSymbolTypeClass           | Specifies classes.                |
| kSymbolTypeFunction        | Specifies functions.              |
| kSymbolTypeVariable        | Specifies variables.              |
| kSymbolTypeDataMember      | Specifies data members.           |
| kSymbolTypeMethod          | Specifies methods.                |
| kSymbolTypeConstant        | Specifies constants.              |
| kSymbolTypeTemplate        | Specifies templates.              |
| kSymbolTypeCompilerDefined | Specifies compiler-defined types. |
| kSymbolTypeScope           | Specifies scopes                  |
| kSymbolTypeAll             | Specifies all symbols.            |

# Targets

---

This chapter shows how to use the Targets API to manage operations with targets in the CodeWarrior IDE.

This chapter contains the following sections:

- Targets API Overview
- Targets API Reference

## Targets API Overview

The Targets API is a set of interfaces that allows a plug-in to create and manipulate targets in the CodeWarrior IDE.

## Targets API Reference

This section describes the functions contained in the following interfaces:

- `ICodeWarriorTarget`
- `ICodeWarriorTargetFile`
- `ICodeWarriorTargetOutput`
- `ICodeWarriorSubTarget`
- `ICodeWarriorSubProjectTarget`
- `ICodeWarriorSegment`

These interfaces use the following data types:

- `ECodeWarriorBrowserGenerator`
- `ECodeWarriorTargetLinkOrderType`
- `ECodeWarriorTargetOutputKind`
- `ECodeWarriorWhichTargetOptions`

## Targets

*ICodeWarriorTarget*

### ICodeWarriorTarget

This interface defines a target that a plug-in can manipulate.

#### Inherited Interfaces

- IUnknown

#### Methods

The following methods are available for your use:

|                                   |                             |
|-----------------------------------|-----------------------------|
| AddFile                           | get_Name                    |
| AddFile2                          | GetNamedPluginData          |
| AddFile2ByFileSpec                | get_Project                 |
| AddFile2ByFileSpecCollection      | get_ProjectFileCollection   |
| AddFileByFileSpec                 | GetProjectFileFromFileSpec  |
| AddFileByFileSpecCollection       | GetSegmentByName            |
| AddSegment                        | GetSegmentList              |
| AddSubTarget                      | GetSegmentListAsBSTR        |
| AddUserTree                       | GetSubProjects              |
| Build                             | get_SubTargets              |
| BuildAgainstSubProjectTarget      | get_TargetFileCollection    |
| BuildAndWaitToComplete            | GetTargetFileForProjectFile |
| BuildAndWaitToCompleteWithOptions | GetTargetOutput             |
| BuildWithOptions                  | get_UserTrees               |
| CompileFiles                      | LinkAgainstSubProjectTarget |
| CompileFilesAndWaitToComplete     | LinkAgainstSubTarget        |
| CompileFilesWithChoice            | MoveFileIntoSegment         |
| CreateUserTree                    | put_BrowserEnabled          |
| Debug                             | put_BrowserGenerator        |
| FindAndAddFile                    | put_Name                    |
| FindAndAddFile2                   | RemoveFile                  |
| FindAndAddFile2ByCollection       | RemoveNamedPluginData       |
| FindAndAddFileByCollection        | RemoveObjectCode            |



|                      |                             |
|----------------------|-----------------------------|
| get_AccessPaths      | RemoveObjectCodeWithOptions |
| get_BrowserDB        | RemoveSegment               |
| get_BrowserEnabled   | RemoveUserTree              |
| get_BrowserGenerator | SetNamedPluginData          |
| get_Design           | SetupDebugging              |
| GetLinkerName        | SynchronizeStatus           |
| get_LinkOrderType    |                             |

AddFile

This method adds a file to the current target.

```
virtual HRESULT AddFile(
 BSTR path,
 BSTR groupPath,
 ICodeWarriorProjectFile **projectFile);
```

path

The path to the file to add.

groupPath

The path to the group to which to add the file.

projectFile

On return, this parameter contains the address of a pointer to the project file associated with the added file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectFile" on page 659

## Targets

### AddFile2

---

### AddFile2

This method adds a file to the current target and set link flags on the file.

```
virtual HRESULT AddFile2
 BSTR path,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 ICodeWarriorProjectFile **projectFile) = 0;
```

path

The path to the file to add.

groupPath

The path to the group to which to add the file.

linkFlags

A value in the range defined by the ECodeWarriorLinkFlags enumeration, representing how the linker should link this file.

projectFile

On return, this parameter contains the address of a pointer to the project file associated with the added file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectFile" on page 659

"ECodeWarriorLinkFlags" on page 557

---

### AddFile2ByFileSpec

This method adds a file to the current target, by using a file specification object, and set link flags on the file.

```
virtual HRESULT AddFile2ByFileSpec (
```

## Targets

### AddFile2ByFileSpecCollection

```
IFileSpec __RPC_FAR *fileSpec,
BSTR groupPath,
ECodeWarriorLinkFlags linkFlags,
ICodeWarriorProjectFile **projectFile) = 0;
```

path

The path to the file to add.

groupPath

The path to the group to which to add the file.

linkFlags

A value in the range defined by the ECodeWarriorLinkFlags enumeration, representing how the linker should link this file.

projectFile

On return, this parameter contains the address of a pointer to the project file associated with the added file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599  
 "ICodeWarriorProjectFile" on page 659  
 "ECodeWarriorLinkFlags" on page 557

## AddFile2ByFileSpecCollection

This method adds a collection of files to the current target and set link flags on the files.

```
virtual HRESULT AddFile2ByFileSpecCollection(
 IFileSpecCollection __RPC_FAR *inCollection,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 int *pFilesAdded) = 0;
```



# Freescale Semiconductor, Inc.

## Targets

*AddFile2ByFileSpecCollection*

---

path

The path to the file to add.

groupPath

The path to the group to which to add the file.

linkFlags

A value in the range defined by the ECodeWarriorLinkFlags enumeration, representing how the linker should link this file.

projectFile

On return, this parameter contains the address of a pointer to the project file associated with the added file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487  
"ICodeWarriorProjectFile" on page 659  
"ECodeWarriorLinkFlags" on page 557

---

## AddFileByFileSpec

This method adds a file to the current target, by file specification.

```
virtual HRESULT AddFileByFileSpec(
 IFileSpec *fileSpec,
 BSTR groupPath,
 ICodeWarriorProjectFile **projectFile);
```

path

The path to the file to add.

groupPath

The path to the group to which to add the file.

projectFile

On return, this parameter contains the address of a pointer to the project file associated with the added file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "IFileSpec" on page 599  
"ICodeWarriorProjectFile" on page 659



## Freescal Semiconductor, Inc.

### Targets

#### AddFileByFileSpecCollection

---

#### AddFileByFileSpecCollection

This method adds all the files in a collection of file specifications to the current target.

```
virtual HRESULT AddFileByFileSpecCollection(
 IFileSpecCollection *inCollection,
 BSTR groupPath,
 int *pFilesAdded);
```

**inCollection**

A pointer to the collection of file specifications that defines the files to add.

**groupPath**

The path to the group to which to add the files.

**pFilesAdded**

On return, this parameter contains a pointer to an integer holding the number of files added to the current target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

---

#### AddSegment

This method adds a segment to the target. Once you have added the segment with this method, you can use the methods in `ICodeWarriorSegment` to specify further details about the new segment.

```
HRESULT AddSegment(
 BSTR segmentName,
 ICodeWarriorSegment **outSegment);
```

---

|          |                                                                                 |
|----------|---------------------------------------------------------------------------------|
|          | segmentName                                                                     |
|          | The name to assign to the new segment.                                          |
|          | outSegment                                                                      |
|          | On return, this parameter contains the address of a pointer to the new segment. |
| Returns  | S_OK if this method call succeeded or an appropriate error if it failed.        |
| See Also | "ICodeWarriorSegment" on page 757                                               |

---

### AddSubTarget

|          |                                                                                                                 |
|----------|-----------------------------------------------------------------------------------------------------------------|
|          | CodeWarrior targets can contain other CodeWarrior targets. This method adds a target within the current target. |
|          | virtual HRESULT AddSubTarget(<br>ICodeWarriorTarget *target,<br>VARIANT_BOOL linkAgainstOutput);                |
|          | target                                                                                                          |
|          | A pointer to the target to be added.                                                                            |
|          | linkAgainstOutput                                                                                               |
|          | true to link against the output of the added subtarget or false if not.                                         |
| Returns  | S_OK if this method call succeeded or an appropriate error if it failed.                                        |
| See Also | "ICodeWarriorTarget" on page 706                                                                                |



## Freescal Semiconductor, Inc.

### Targets

#### AddUserTree

---

#### AddUserTree

This method adds an existing user tree to the current target.

```
virtual HRESULT AddUserTree(
 ICodeWarriorUserTree *pval) = 0;
```

pval

A pointer to the user tree to add to the application.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorUserTree" on page 436

---

#### Build

This method tells the CodeWarrior IDE to build the current target.

```
virtual HRESULT Build(
 long *cookie);
```

cookie

On return, this parameter contains a unique identifier for the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### BuildAgainstSubProjectTarget

This method controls whether to build a target within a subproject.

```
virtual HRESULT BuildAgainstSubProjectTarget(
 ICodeWarriorSubProjectTarget *target,
 VARIANT_BOOL val) = 0;
```

---

## Targets

### *BuildAndWaitToComplete*

target

A pointer to the target to build.

val

true to build the target or false to not build it.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorSubProjectTarget” on page 755

## BuildAndWaitToComplete

This method starts a build of the current project and has the IDE wait to gather all messages from the build process.

```
virtual HRESULT BuildAndWaitToComplete(
 ICodeWarriorBuildMessages **buildMessages);
```

buildMessages

On return, this parameter contains the address of a pointer to the messages created by the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorBuildMessages” on page 616

## BuildAndWaitToCompleteWithOptions

This method builds the current project with one of the options specified in the ECodeWarriorBuildOptions enumeration. This method accumulates all the messages from the build process before returning.

```
virtual HRESULT BuildAndWaitToCompleteWithOptions(
 ECodeWarriorBuildOptions options,
```

## Targets

### *BuildWithOptions*

---

```
ICodeWarriorBuildMessages **buildMessages) = 0;
```

options

The build options to use with this build.

buildMessages

On return, this parameter contains the address of a pointer to the build messages created by the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorBuildOptions” on page 662

---

## BuildWithOptions

This method builds the current target with one of the options specified in the ECodeWarriorBuildOptions enumeration.

```
virtual HRESULT BuildWithOptions(
 ECodeWarriorBuildOptions options,
 ECodeWarriorRunMode runMode,
 long *cookie) = 0;
```

options

The build options to use with this build.

runmode

Whether to run the resulting program after building it and, if so, whether to run it in debug mode. The ECodeWarriorRunMode enumeration contains the costants that define this parameter.

cookie

On return, this parameter contains a unique identifier for the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorBuildOptions” on page 662  
“ECodeWarriorRunMode” on page 662

## CompileFiles

Use this method to compile the current target.

```
virtual HRESULT CompileFiles(
 ICodeWarriorProjectFileCollection *collection,
 long *cookie);
```

collection

A pointer of type ICodeWarriorProjectFileCollection indicating the collection of files to compile.

cookie

On return, this parameter contains a unique identifier for the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

## Targets

*CompileFilesAndWaitToComplete*

---

### CompileFilesAndWaitToComplete

Use this method to compile the current target and return all the build messages created by the compiler.

```
virtual HRESULT CompileFilesAndWaitToComplete(
 ICodeWarriorProjectFileCollection *collection,
 ICodeWarriorBuildMessages **buildMessages);
```

*collection*

A pointer of type ICodeWarriorProjectFileCollection indicating the collection of files to compile.

*buildMessages*

On return, this parameter contains the address of a pointer to the build messages generated by the compiler.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

---

### CompileFilesWithChoice

This method compiles a collection of project files with one of the options specified in the ECodeWarriorCompileChoice enumeration.

```
virtual HRESULT CompileFilesWithChoice(
 ICodeWarriorProjectFileCollection *collection,
 ECodeWarriorCompileChoice compileChoice,
 long *cookie) = 0;
```

*collection*

A pointer to a collection of file projects to compile.

*compileChoice*

The kind of compilation the compiler should perform.



cookie  
On return, this parameter contains a unique identifier for the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorCompileChoice" on page 662

CreateUserTree

This method creates a new user tree.

```
virtual HRESULT CreateUserTree(
 BSTR displayName,
 BSTR value,
 EUserDefinedTree type,
 BSTR keyName,
 ICodeWarriorUserTree **pVal) = 0;
```

displayName  
The name of the user tree that will appear in the IDE.

value  
The value string of the user tree.

type  
The type of the tree, which must be one of the values specified by the EUserDefinedTree Tree enumeration.

keyName  
The key name of the user tree.

pval  
On return, this parameter contains the address of a pointer to the new user tree.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Targets

### Debug

See Also "EUserDefinedTree" on page 441  
"ICodeWarriorUserTree" on page 436

## Debug

This method runs the debugger on this target and returns the build messages. You can set which debugger is associated with this target in the IDE.

```
virtual HRESULT Debug(
 ICodeWarriorBuildMessages** buildMessages) = 0;

buildMessages
```

On return, this parameter contains the address of a pointer to the messages created by the build process.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorBuildMessages" on page 616

## FindAndAddFile

This method finds a file and adds it to the current target.

```
virtual HRESULT FindAndAddFile(
 BSTR path,
 BSTR groupPath,
 ICodeWarriorProjectFile **projectFile);

path
```

Either the absolute (fully qualified) path to the file you want to add to the design or just the name of the file. If you provide just the file name, the IDE searches the access paths and adds the first file of that name that it finds. If you want to add two one files with identical names, use the fully qualified path to each



one.

groupPath

The absolute path to the group within which the new file should be added.

projectFile

On return, this parameter contains the address of a pointer to the project file that contains the current target..

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProjectFile" on page 659

## Targets

### *FindAndAddFile2*

---

## FindAndAddFile2

This method finds a file and adds it to the design. This method also lets you set link flags on the file.

```
virtual HRESULT FindAndAddFile2(
 BSTR path,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 ICodeWarriorProjectFile **projectFile) = 0;
```

#### path

Either the absolute (fully qualified) path to the file you want to add to the design or just the name of the file. If you provide just the file name, the IDE searches the access paths and adds the first file of that name that it finds. If you want to add two one files with identical names, use the fully qualified path to each one.

#### groupPath

The absolutepath to the group within which the new file should be added.

#### linkFlags

A value in the range defined by the ECodeWarriorLinkFlags enumeration, representing how the linker should link this file.

#### projectFile

On return, this parameter contains the address of a pointer to the project file to which the file was added.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ECodeWarriorLinkFlags" on page 557

---

## FindAndAddFile2ByCollection

This method adds a named collection of files to the current target, setting link flags on all the added files in the process.

```
virtual HRESULT FindAndAddFile2ByCollection(
 IBSTRCollection *inCollection,
 BSTR groupPath,
 ECodeWarriorLinkFlags linkFlags,
 int *pFilesAdded) = 0;
```

*inCollection*

The name of the collection of files to add to the current target.

*groupPath*

The absolute path to the group within which the new file should be added.

*linkFlags*

A value in the range defined by the *ECodeWarriorLinkFlags* enumeration, representing how the linker should link this file.

*pFilesAdded*

On return, this parameter contains a pointer to the number of files added.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “ECodeWarriorLinkFlags” on page 557

## Targets

### *FindAndAddFileByCollection*

---

## FindAndAddFileByCollection

---

This method adds a collection of files to the current target.

```
virtual HRESULT FindAndAddFileByCollection(
 IBSTRCollection *inCollection,
 BSTR groupPath,
 int *pFilesAdded);
```

*inCollection*

The file collection to add.

*groupPath*

The path to the group within which the new files should be added.

*pFilesAdded*

On return, this parameter contains a pointer to an integer holding the number of files added to the target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

---

## get\_AccessPaths

This method gets the access paths for the current target.

```
virtual HRESULT get_AccessPaths(
 ICodeWarriorAccessPaths **pval);
```

*pval*

On return, this parameter contains the address of a pointer to the access paths of the current target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorAccessPaths" on page 429

---

## get\_BrowserDB

This method gets the symbols created during the most recent build of the current target.

```
virtual HRESULT get_BrowserDB(
 ICodeWarriorSymbolContainer **catalog);
```

catalog

On return, this parameter contains the address of a parameter to the list of symbols generated by the most recent build of the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorSymbolContainer" on page 694

---

## get\_BrowserEnabled

This method gets whether the symbol browser is enabled.

```
virtual HRESULT get_BrowserEnabled(
 VARIANT_BOOL *fEnabled);
```

fEnabled

On return, this parameter contains a pointer to a boolean that is set to `true` if the symbol browse is enabled or `false` otherwise.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Targets

*get\_BrowserGenerator*

---

### get\_BrowserGenerator

This method gets the source of the browser symbols being generated for this target.

```
virtual HRESULT get_BrowserGenerator(
 ECodeWarriorBrowserGenerator* generator) = 0;
```

generator

On return, this parameter contains one of the constants defined in the ECodeWarriorBrowserGenerator enumeration.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorBrowserGenerator" on page 764

---

### get\_Design

This method gets the design associated with the current target.

```
virtual HRESULT get_Design(
 ICodeWarriorDesign **design);
```

design

On return, this parameter contains the address of a pointer to the design associated with the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorDesign" on page 542

---



---

GetLinkerName

This method gets the name of the current linker.

```
virtual HRESULT GetLinkerName(
 BSTR *pval) = 0;
```

pval

On return, this pararameter contains the name of the current linker.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

get\_LinkOrderType

This method gets the type of link order used by this target.

```
HRESULT get_LinkOrderType(
 ECodeWarriorTargetLinkOrderType *type);
```

type

On return, this parameter contains a pointer to the enumerated value that indicates the link order type used by this target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorTargetLinkOrderType” on page 764

---

get\_Name

This method gets the name of the current target.

```
virtual HRESULT get_Name(
 BSTR *pval);
```

## Targets

### *GetNamedPluginData*

---

pval

On return, this parameter contains the name of the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### GetNamedPluginData

This method gets the data from a plug-in, specified by name.

```
virtual HRESULT GetNamedPluginData(
 BSTR resourceName,
 EPluginDataStorageLoc storeIn,
 IStream **pluginData);
```

resourceName

The name of the plug-in from which to get data.

storeIn

The location in which the data is stored.

pluginData

On return, this parameter contains the address of a pointer to the data from the specified plug-in.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "EPluginDataStorageLoc" on page 663

---

### get\_Project

This method gets the project associated with the current target.

```
virtual HRESULT get_Project(
 ICodeWarriorProject **project);
```

## Targets

*get\_ProjectFileCollection*

`project`

On return, this parameter contains the address of a pointer to the project associated with the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorProject" on page 630

---

## get\_ProjectFileCollection

This method gets the project file collection to which the project file associated with the current target belongs.

```
virtual HRESULT get_ProjectFileCollection(
 ICodeWarriorProjectFileCollection
 **projectFileCollection);
```

`projectFileCollection`

On return, this parameter contains the address of a pointer to the project file collection associated with the project file to which the current target belongs.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

## GetProjectFileFromFileSpec

This method gets a project file, by file specification.

```
virtual HRESULT GetProjectFileFromFileSpec(
 IFileSpec *fileSpec,
 ICodeWarriorProjectFile **projectFile);
```

## Targets

### *GetSegmentByName*

---

`fileSpec`

A pointer to the file specification.

`projectFile`

On return, this parameter contains the address of a pointer to the project file specified in the `fileSpec` parameter.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also “IFileSpec” on page 599

---

## GetSegmentByName

This method gets a specified segment, given the segment’s name.

```
HRESULT GetSegmentByName (
 BSTR segmentName,
 ICodeWarriorSegment **outSegment);
```

`segmentName`

The name of the segment to get.

`outSegment`

On return, this parameter contains the address of a pointer to the specified segment.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also “ICodeWarriorSegment” on page 757

---

## GetSegmentList

This method gets the collection of segments within the target. You can then use the Collection API to access the segments.

```
HRESULT GetSegmentList(
 ICodeWarriorSegmentCollection
 **segCollection);
```

segCollection

On return, this parameter contains the address of a pointer to the collection of segments within the target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

---

## GetSegmentListAsBSTR

This method gets a collection of strings that correspond to the segments within the target. You can then use the Collection API to access the strings.

```
HRESULT GetSegmentListAsBSTR(
 IBSTRCollection **segCollection);
```

segCollection

On return, this parameter contains the address of a pointer to the collection of strings that correspond to the segments within the target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "Using the Collections API" on page 487

## Targets

### GetSubProjects

---

### GetSubProjects

This method gets a collection of the projects within the current target.

```
virtual HRESULT GetSubProjects(
 ICodeWarriorSubProjectCollection
 **subProjectList) = 0;
```

subProjectList

On return, this parameter contains the address of a pointer to the a collection of subprojects.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

### get\_SubTargets

A CodeWarrior target can contain other CodeWarrior targets. This method gets the targets contained within the current target.

```
virtual HRESULT get_SubTargets(
 ICodeWarriorSubTargetCollection
 **subTargetList);
```

subTargetList

On return, this parameter contains the address of a pointer to the collection of targets within the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

## get\_TargetFileCollection

This method gets the collection of target files that the current target contains.

```
virtual HRESULT get_TargetFileCollection(
 ICodeWarriorTargetFileCollection
 **targetFileCollection);
```

targetFileCollection

On return, this parameter contains the address of a pointer to the collection of target files within the current target.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “Using the Collections API” on page 487

---

## GetTargetFileForProjectFile

This method gets the target file for a specified project file.

```
virtual HRESULT GetTargetFileForProjectFile(
 ICodeWarriorProjectFile *projectFile,
 ICodeWarriorTargetFile **targetFile);
```

projectFile

A pointer to the project file for which to get the target file.

targetFile

On return, this parameter contains the address of a pointer to the target file associated with the specified project file.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** “ICodeWarriorProjectFile” on page 659

“ICodeWarriorTargetFile” on page 743

## Targets

### GetTargetOutput

---

## GetTargetOutput

This method gets the output of the current target.

```
virtual HRESULT GetTargetOutput(
 ICodeWarriorTargetOutput **targetOutput);
```

targetOutput

On return, this parameter contains the address of a pointer to the output of the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTargetOutput" on page 751

---

## get\_UserTrees

This method gets the user trees, as a collection of trees.

```
virtual HRESULT get_UserTrees(
 ICodeWarriorUserTreeCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to a collection of trees that constitute the user trees.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---



LinkAgainstSubProjectTarget

This method specifies whether to link against the target produced by a particular subproject within the current target.

```
virtual HRESULT LinkAgainstSubProjectTarget(
 ICodeWarriorSubProjectTarget *target,
 VARIANT_BOOL val) = 0;
```

target  
The target (produced by a subproject) to link against (or not).

val  
true to link against the target specified in target or false if not.

See Also "ICodeWarriorSubProjectTarget" on page 755

LinkAgainstSubTarget

This method specifies whether to link against a particular subtarget within the current target.

```
virtual HRESULT LinkAgainstSubTarget(
 ICodeWarriorSubTarget *target,
 VARIANT_BOOL val) = 0;
```

target  
The subtarget to link against (or not).

val  
true to link against the target specified in target or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorSubTarget" on page 753

## Targets

### *MoveFileIntoSegment*

---

## MoveFileIntoSegment

This method moves an existing file into a segment.

```
HRESULT MoveFileIntoSegment (
 ICodeWarriorTargetFile *targetFile,
 ICodeWarriorSegment *segment,
 long segmentPosition);
```

*targetFile*

The file to move into the segment.

*segment*

The segment into which to move the file.

*segmentPosition*

The position of the file within the segment's file list. Use 0 to put the file at the start of the file list. Use a negative number to put the file at the end of the file list. Use a positive number to put the file at the position that corresponds to the number.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorSegment" on page 757  
 "ICodeWarriorTargetFile" on page 743

---

## put\_BrowserEnabled

This method sets whether the symbol browser is enabled.

```
virtual HRESULT put_BrowserEnabled (
 VARIANT_BOOL fEnabled);
```

*fEnabled*

Set this parameter to true to enable the symbol browser or false to disable it.

## Targets

*put\_BrowserGenerator*

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## put\_BrowserGenerator

This method sets the source of debugging messages for this target.

```
virtual HRESULT put_BrowserGenerator(
 ECodeWarriorBrowserGenerator generator) = 0;
```

generator

One of the constants defined in the  
ECodeWarriorBrowserGenerator enumeration.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ECodeWarriorBrowserGenerator" on page 764

## put\_Name

This target assigns the name of the current target.

```
virtual HRESULT put_Name(
 BSTR pval);
```

pval

The new name for the current target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Targets

### *RemoveFile*

---

## RemoveFile

This method removes the project file for this target.

```
virtual HRESULT RemoveFile(
 ICodeWarriorProjectFile* projectFile) = 0;
```

*projectFile*

A pointer to the project file to remove.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorProjectFile" on page 659

---

## RemoveNamedPluginData

This method removes the data from a plug-in specified by name.

```
virtual HRESULT RemoveNamedPluginData(
 BSTR resourceName,
 EPluginDataStorageLoc storeIn);
```

*resourceName*

The name of the plug-in from which to remove data.

*storeIn*

The location where the data is stored.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "EPluginDataStorageLoc" on page 663

---

---

## RemoveObjectCode

This method removes the object code created by building the current target.

```
virtual HRESULT RemoveObjectCode(
 VARIANT_BOOL deleteDataFiles);
```

deleteDataFiles

Set this parameter to true to delete the data files associated with the object data or false to retain them.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## RemoveObjectCodeWithOptions

This method removes the object code from the current target. It also lets you choose whether to remove the object code from all the targets created by subprojects within the current target and whether to delete any associated data files.

```
virtual HRESULT RemoveObjectCodeWithOptions(
 ECodeWarriorWhichTargetOptions
 whichTargetOfSubprojects,
 VARIANT_BOOL recurseSubProject,
 VARIANT_BOOL deleteDataFiles) = 0;
```



## Freescal Semiconductor, Inc.

### Targets

#### *RemoveSegment*

---

`whichTargetOfSubprojects`

A value with the range defined by the `ECodeWarriorWhichTargetOptions` enumeration that specifies whether to remove the object code from all targets or only the current target.

`recurseSubProject`

Set this parameter to `true` to remove the object code within all the subprojects that match the `whichTargetOfSubprojects` parameter or `false` to leave the object code in the subprojects.

`deleteDataFiles`

Set this parameter to `true` to delete data files associated with the current target or `false` to retain the data files.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

See Also “`ECodeWarriorWhichTargetOptions`” on page 765

---

### RemoveSegment

This method removes a segment, given the name of the segment.

```
HRESULT RemoveSegment (BSTR segmentName);
```

`segmentName`

The name of the segment to remove.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.

---

### RemoveUserTree

This method removes a specified user tree.

```
virtual HRESULT RemoveUserTree (
```

## Targets

*SetNamedPluginData*

---

```
ICodeWarriorUserTree *pval) = 0;
```

*pval*

A pointer to the user tree to remove.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorUserTree" on page 436

---

## SetNamedPluginData

This method assigns data to a plug-in specified by name.

```
virtual HRESULT SetNamedPluginData(
 BSTR resourceName,
 EPluginDataStorageLoc storeIn,
 IStream *pluginData);
```

*resourceName*

The name of the plug-in to which to assign data.

*storeIn*

The location in which to store the data.

*pluginData*

A pointer to the data to store.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "EPluginDataStorageLoc" on page 663



## Freescal Semiconductor, Inc.

### Targets

*SetupDebugging*

---

### SetupDebugging

This method sets whether debugging is enabled.

```
virtual HRESULT SetupDebugging(
 VARIANT_BOOL inTurnOn);
```

*inTurnOn*

Set this parameter to true to enable debugging or false to disable debugging.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SynchronizeStatus

This method synchronizes the dates of the files stored in a version control system with the dates of the working files. For this method to work, you must have defined the VCS settings either globally or for the project that contains the target.

```
virtual HRESULT SynchronizeStatus(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## ICodeWarriorTargetFile

This interface lets a plug-in work target files.

### Inherited Interfaces

- IUnknown

### Methods

The following methods are available for your use:

|                  |                  |
|------------------|------------------|
| get_DebugInfo    | get_Target       |
| get_Dependencies | get_WeakImport   |
| get_Dependents   | put_DebugInfo    |
| get_FileSpec     | put_InitBefore   |
| get_InitBefore   | put_MergeLibrary |
| get_MergeLibrary | put_WeakImport   |
| get_Name         |                  |

### get\_DebugInfo

This method gets whether a debugger can get debugging information from the current target file.

```
virtual HRESULT get_DebugInfo(
 VARIANT_BOOL *value);
```

value

On return, this parameter contains a boolean that is set to `true` if a debugging application can get debugging information from the current target file and `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Targets

### *get\_Dependencies*

---

### get\_Dependencies

This method returns a collection containing the dependencies (the files on which this target depends) for the current target file.

```
virtual HRESULT get_Dependencies(
 ICodeWarriorTargetFileCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to a collection of files that form the dependencies for the current target file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

### get\_Dependents

This method returns a collection containing the dependents (the files that depend on this target) for the current target file.

```
virtual HRESULT get_Dependents(
 ICodeWarriorTargetFileCollection **pval);
```

pval

On return, this parameter contains the address of a pointer to a collection of files that form the dependents for the current target file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

## get\_FileSpec

This method gets the file specification for the current target file.

```
virtual HRESULT get_FileSpec(
 IFileSpec **pval);
```

*pval*

On return, this parameter contains the address of a pointer to the file specification for the current target file.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "IFileSpec" on page 599

---

## get\_InitBefore

This method gets the state of the CWInitBefore flag.

```
virtual HRESULT get_InitBefore(
 VARIANT_BOOL *value);
```

*value*

On return, this parameter contains a pointer to a boolean set to true if the CWInitBefore flag is set to true or false if the CWInitBefore flag is set to false.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ECodeWarriorLinkFlags" on page 557



## Freescal Semiconductor, Inc.

### Targets

#### *get\_MergeLibrary*

---

#### get\_MergeLibrary

This method gets whether the build process should merge with libraries when building this target file.

```
virtual HRESULT get_MergeLibrary(
 VARIANT_BOOL *value);
```

value

On return, this parameter contains a boolean that is set to `true` if the current target file should be merged with libraries during the next build or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### get\_Name

This method gets the name of the current target file.

```
virtual HRESULT get_Name(
 BSTR *pval);
```

pval

On return, this parameter contains the name of the current target file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### get\_Target

This method gets the target object associated with the current target file.

```
virtual HRESULT get_Target(

```

---

---

```
ICodeWarriorTarget **pval);
```

pval

On return, this parameter contains the address of a pointer to the target object associated with the current target file.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorTarget" on page 706

---

## get\_WeakImport

This method gets whether this target file should be built with the Weak Import option (available only on the Mac OS).

```
virtual HRESULT get_WeakImport(
 VARIANT_BOOL *value);
```

value

On return, this parameter contains a boolean that is set to true if the current target file should be build with the Weak Import option and false if not.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_DebugInfo

This method sets whether a debugger can get debugging information from the current target file.

```
virtual HRESULT put_DebugInfo(
 VARIANT_BOOL value);
```

value

Set this parameter to true if a debugging application can get



# Freescale Semiconductor, Inc.

## Targets

*put\_DebugInfo*

---

|         |                                                                                       |
|---------|---------------------------------------------------------------------------------------|
|         | debugging information from the current target file and <code>false</code> if not.     |
| Returns | <code>S_OK</code> if this method call succeeded or an appropriate error if it failed. |



---

**put\_InitBefore**

This method sets the CWInitBefore flag.

```
virtual HRESULT put_InitBefore(
 VARIANT_BOOL value);
```

value

true to set the CWInitBefore flag to true or false to set CWInitBefore the flag to false.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorLinkFlags” on page 557

---

**put\_MergeLibrary**

This method sets whether the build process should merge with libraries when building this target file.

```
virtual HRESULT put_MergeLibrary(
 VARIANT_BOOL value);
```

value

Set this parameter to true if the current target file should be merged with libraries during the next build or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescale Semiconductor, Inc.

## Targets

*put\_WeakImport*

---

### put\_WeakImport

---

This method sets whether this target file should be built with the Weak Import option (available only on the Mac OS).

```
virtual HRESULT put_WeakImport(
 VARIANT_BOOL value);
```

value

Set this parameter to `true` if the current target file should be build with the Weak Import option and `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**ICodeWarriorTargetOutput**

.This interface gets information about the output resulting from building the current target.

The following methods are available for your use:

|                           |                             |
|---------------------------|-----------------------------|
| <code>get_FileSpec</code> | <code>get_OutputKind</code> |
|---------------------------|-----------------------------|

**get\_FileSpec**

This method gets the file specification for the output file that is created when the current target is built.

```
virtual HRESULT get_FileSpec(
 IFileSpec **pval);
```

pval

On return, this parameter contains the address of a pointer to a file specification that defines the target output file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “IFileSpec” on page 599



**Targets**

*get\_OutputKind*

---

---

**get\_OutputKind**

This method gets the kind of output generated by building the current target.

```
virtual HRESULT get_OutputKind(
 ECodeWarriorTargetOutputKind *kind);
```

kind

On return, this parameter contains a pointer to the kind of output.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorTargetOutputKind” on page 764



ICodeWarriorSubTarget

This interface provides methods for getting information about a subtarget within the current target.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                       |            |
|-----------------------|------------|
| get_LinkAgainstOutput | get_Target |
|-----------------------|------------|

get\_LinkAgainstOutput

This method gets whether to link against the output of the current subtarget.

```
virtual HRESULT get_LinkAgainstOutput (
 VARIANT_BOOL *pVal);
```

pVal

On return, this parameter contains a pointer to a boolean that is set to true if the output of the current subtarget should be linked against or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Targets**  
*get\_Target*

---

---

**get\_Target**

This method gets the target object for the current subtarget.

```
virtual HRESULT get_Target(
 ICodeWarriorTarget **pval);
```

pval

On return, this parameter contains the address of a pointer to the current subproject object.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



### ICodeWarriorSubProjectTarget

This interface provides methods for getting information about a target within a subproject contained by the current target.

#### Inherited Interfaces

- IDispatch

#### Methods

The following methods are available for your use:

|                  |          |
|------------------|----------|
| get_BuildAgainst | get_Name |
| get_LinkAgainst  |          |

#### get\_BuildAgainst

This method gets whether to build against the current target within a subproject within a containing target.

```
virtual HRESULT get_BuildAgainst(
 VARIANT_BOOL *pval) = 0;
```

pval

On return, this parameter contains a pointer to a boolean that is set to `true` if this target should be built against or `false` if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Targets

*get\_LinkAgainst*

---

### get\_LinkAgainst

This method gets whether to link against the current target within a subproject within a containing target.

```
virtual HRESULT get_LinkAgainst(
 VARIANT_BOOL __RPC_FAR *pval) = 0;
```

pval

On return, this parameter contains a pointer to a boolean that is set to true if this target should be linked against or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_Name

This method gets the name of the current target within a subproject within a containing target.

```
virtual HRESULT get_Name(
 BSTR *pval) = 0;
```

pval

On return, this parameter contains the name of the target.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---



ICodeWarriorSegment

This interface defines a segment within a target, for processors that support segments.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                |                |
|----------------|----------------|
| GetTargetFiles | Protected      |
| Name           | Protected      |
| Name           | Purgable       |
| Locked         | Purgable       |
| Locked         | PutTargetFiles |
| PreLoaded      | SystemHeap     |
| PreLoaded      | SystemHeap     |
|                | Target         |

GetTargetFiles

This method gets the collection of target files in the segment. Once you have the collection, you can manipulate it with the Collection API.

```
HRESULT GetTargetFiles(
 ICodeWarriorTargetFileCollection** val);

val
```

On return, this parameter contains the address of a pointer to a collection of the files in the segment.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescale Semiconductor, Inc.

## Targets

### Name

---

See Also    “ICodeWarriorTargetFile” on page 743  
              “Using the Collections API” on page 487

---

### Name

This method gets the name of the segment.

```
HRESULT Name (BSTR* pval);
```

pval

On return, this parameter contains

Returns    S\_OK if this method call succeeded or an appropriate error if it failed.

---

### Name

This method sets the name of the segment

```
HRESULT Name (BSTR pval);
```

pval

The new name for the segment.

Returns    S\_OK if this method call succeeded or an appropriate error if it failed.

---

### Locked

This method gets whether the segment is locked.

```
HRESULT Locked (VARIANT_BOOL* pval);
```



Targets

Locked

pval

On return, this parameter contains a pointer to a boolean value that is set to `true` if the segment is locked or `false` if not.

Returns

S\_OK if this method call succeeded or an appropriate error if it failed.

Locked

This method sets whether the segment is locked.

HRESULT Locked(VARIANT\_BOOL pval);

pval

`true` to lock the segment or `false` to unlock it.

Returns

S\_OK if this method call succeeded or an appropriate error if it failed.

PreLoaded

This method gets whether the segment is preloaded.

HRESULT PreLoaded(VARIANT\_BOOL\* pval);

pval

On return, this parameter contains a pointer to a boolean value that is set to `true` if the segment is preloaded or `false` if not.

Returns

S\_OK if this method call succeeded or an appropriate error if it failed.

IDE 5.1 SDK API Reference

SDK-759



# Freescal Semiconductor, Inc.

**Targets**  
*PreLoaded*

---

**PreLoaded**

This method sets whether the segment is preloaded.

```
HRESULT PreLoaded(VARIANT_BOOL pval);
```

pval

true to set the segment to preloaded or false to not set it to preloaded.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**Protected**

This method gets whether the segment is protected.

```
HRESULT Protected(VARIANT_BOOL* pval);
```

pval

On return, this parameter contains a pointer to a boolean value that is set to true if the segment is protected or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**Protected**

This method sets whether the segment is protected.

```
HRESULT Protected(VARIANT_BOOL pval);
```

pval

true to set the segment to protected or false to not set it to protected.



|                                                                                                                                                                                                                                                             |                                                                          | Targets  |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|----------|
|                                                                                                                                                                                                                                                             |                                                                          | Purgable |
| Returns                                                                                                                                                                                                                                                     | S_OK if this method call succeeded or an appropriate error if it failed. |          |
| Purgable                                                                                                                                                                                                                                                    |                                                                          |          |
| <p>This method gets whether the segment is purgable.</p> <pre>HRESULT Purgable(VARIANT_BOOL pval);</pre> <p>pval</p> <p>On return, this parameter contains a pointer to a boolean value that is set to true if the segment is purgable or false if not.</p> |                                                                          |          |
| Returns                                                                                                                                                                                                                                                     | S_OK if this method call succeeded or an appropriate error if it failed. |          |
| Purgable                                                                                                                                                                                                                                                    |                                                                          |          |
| <p>This method sets whether the segment is purgable.</p> <pre>HRESULT Purgable([out, retval] VARIANT_BOOL* pval);</pre> <p>pval</p> <p>true to set the segment to purgable or false to not set it to purgable.</p>                                          |                                                                          |          |
| Returns                                                                                                                                                                                                                                                     | S_OK if this method call succeeded or an appropriate error if it failed. |          |
| PutTargetFiles                                                                                                                                                                                                                                              |                                                                          |          |
| <p>This method adds existing target files to the segment.</p> <pre>HRESULT PutTargetFiles(     ICodeWarriorTargetFileCollection* val,</pre>                                                                                                                 |                                                                          |          |

## Targets

### SystemHeap

---

```
long inSegmentPos);
```

val

A collection of files to add to the segment. Use 0 to put the files at the start of the file list. Use a negative number to put the files at the end of the file list. Use a positive number to put the files at the position that corresponds to the number.

inSegmentPos

The position at which to add the files. Use 0 to put the files at the start of the file list. Use a negative number to put the files at the end of the file list. Use a positive number to put the files at the position that corresponds to the number.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTargetFile" on page 743

---

## SystemHeap

This method gets the value of the segment's system heap flag.

```
HRESULT SystemHeap(VARIANT_BOOL* pval);
```

pval

On return, this parameter contains a pointer to a boolean value that is set to `true` if the system heap flag has been set to `true` for this segment or `false` otherwise.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



SystemHeap

This method sets the system heap flag for this segment.

```
HRESULT SystemHeap(VARIANT_BOOL pval);
```

pval

true to set the system heap flag for this segment to true or  
false to set the flag to false.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

Target

This method gets the target that contains the segment.

```
HRESULT Target(ICodeWarriorTarget** pval);
```

pval

On return, this parameter contains the address of a pointer to  
the target containing the segment.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorTarget" on page 706

## Targets Data Types

### ECodeWarriorBrowserGenerator

The ECodeWarriorBrowserGenerator enumeration provides constants to define the object that generates browser data for debugging.

**Table 24.1 ECodeWarriorBrowserGenerator Enumeration**

| Constant        | Description                          |
|-----------------|--------------------------------------|
| kNoBrowserData  | Browser data is unavailable.         |
| kCompiler       | The compiler generates browser data. |
| kLanguageParser | The parser generates browser data.   |

### ECodeWarriorTargetLinkOrderType

The ECodeWarriorTargetLinkOrderType enumeration provides constants that define the link order for a target.

**Table 24.2 ECodeWarriorTargetLinkOrderType Enumeration**

| Constant     | Description                                                         |
|--------------|---------------------------------------------------------------------|
| kCWSegment   | Link the target by segment order.                                   |
| kCWLinkOrder | Follow the link order specified in the IDE's <b>Link Order</b> tab. |
| kCWOOverlay  | Link the target by overlay order.                                   |

### ECodeWarriorTargetOutputKind

The ECodeWarriorTargetOutputKind enumeration provides constants for what kind of output to produce when building a target.

**Table 24.3    ECodeWarriorTargetOutputKind Enumeration**

| Constant           | Description                      |
|--------------------|----------------------------------|
| kCWOutputNone      | Produce no output.               |
| kCWOutputFile      | Write the output to a file.      |
| kCWOutputDirectory | Write the output to a directory. |

**ECodeWarriorWhichTargetOptions**

The ECodeWarriorWhichTargetOptions enumeration provides constants for which target to use in various operations.

**Table 24.4    ECodeWarriorWhichTargetOptions Enumeration**

| Constant       | Description                                     |
|----------------|-------------------------------------------------|
| kAllTargets    | Apply the operation to only the current target. |
| kCurrentTarget | Apply the operation to all targets.             |



# Freescale Semiconductor, Inc.

## Targets

*Targets Data Types*

---

# Text

---

This chapter shows how to use the Text API to create and manage text operations in the CodeWarrior IDE.

This chapter contains the following sections:

- Text API Overview
- Text API Reference

## Text API Overview

The Text API lets plug-ins work with blocks of text in the CodeWarrior IDE.

## Text API Reference

This section describes the methods contained in the following interface:

- `ICodeWarriorTextEngine`

ICodeWarriorTextEngine

This interface provides methods for working with text within the CodeWarrior IDE.

Inherited Interfaces

- IUnknown

Methods

This interface exposes the following methods:

|                        |                        |
|------------------------|------------------------|
| get_HasSelection       | get_TextLength         |
| get_LineCount          | GetTextForLineRange    |
| GetLineForOffset       | GetTextForOffsetRange  |
| GetOffsetForLine       | InsertText             |
| get_SelectionEnd       | put_SelectionEnd       |
| get_SelectionLineEnd   | put_SelectionLineEnd   |
| get_SelectionLineStart | put_SelectionLineStart |
| get_SelectionStart     | put_SelectionStart     |
| get_SelectionText      | put_SelectionText      |

get\_HasSelection

This method gets whether the user has selected a block of text.

```
get_HasSelection(
 VARIANT_BOOL *pval);
```

pval

On return, this parameter contains a pointer to a boolean that is set to true if the user has highlighted a block of text or false if not.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## get\_LineCount

---

This method gets the number of lines of text the user has selected.

```
get_LineCount (
 int *pval);
```

pval

On return, this parameter contains a pointer to an integer that indicates how many lines the user has selected.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## GetLineForOffset

---

This method gets the line number of a given character in the source file.

```
GetLineForOffset (
 int offset,
 int *line);
```

offset

The position from the top of the file for which to get a line number.

line

On return, this parameter contains a pointer to an integer indicating the line number of the specified character.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Text

#### *GetOffsetForLine*

---

#### GetOffsetForLine

This method gets the number of characters from the top of the source file to the specified line.

```
GetOffsetForLine(
 int line,
 int *offset);
```

line

The line number in question.

offset

On return, this parameter contains a pointer to an integer indicating the number of characters to the specified line.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### get\_SelectionEnd

This method gets the number of characters from the top of the file to the end of the user's selection.

```
get_SelectionEnd(
 int *pval);
```

pval

On return, this parameter contains a pointer to an integer that indicates how many characters from the top of the file to the end of the user's selection.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Text**

*get\_SelectionLineEnd*

---

---

## get\_SelectionLineEnd

This method gets how many lines from the top of the source file to the last line of the user's selection.

```
get_SelectionLineEnd(
 int *pval);
```

pval

On return, this parameter contains a pointer to an integer that indicates how many lines from the top of the file to the last line of the user's selection.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_SelectionLineStart

This method gets how many lines from the top of the source file to the first line of the user's selection.

```
get_SelectionLineStart(
 int *pval);
```

pval

On return, this parameter contains a pointer to an integer that indicates how many lines from the top of the file to the first line of the user's selection.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Text

*get\_SelectionStart*

---

### get\_SelectionStart

This method gets the number of characters from the top of the file to the start of the user's selection.

```
get_SelectionStart(
 int *pval);
```

pval

On return, this parameter contains a pointer to an integer that indicates how many characters from the top of the file to the start of the user's selection.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### get\_SelectionText

This method gets the text the user has selected.

```
get_SelectionText(
 BSTR *pval);
```

pval

On return, this parameter contains the text the user has selected.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## get\_TextLength

This method gets the number of characters the user has selected.

```
get_TextLength(
 int *pval);
```

pval

On return, this parameter contains a pointer to an integer that indicates how many characters the user has selected.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## GetTextForLineRange

This method gets the text within a specified line range.

```
GetTextForLineRange(
 int lineStart,
 int lineEnd,
 BSTR *pval);
```

lineStart

The first line of the block of text to get.

lineEnd

The last line of the block of text to get.

pval

On return, this parameter contains the text specified by the lineStart and lineEnd parameters.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Text

### GetTextForOffsetRange

---

### GetTextForOffsetRange

This method gets the text within a specified offset from the top of the source file.

```
GetTextForOffsetRange (
 int selStart,
 int selEnd,
 BSTR *pval);
```

**selStart**

The first character of the block of text to get.

**selEnd**

The last character of the block of text to get.

**pval**

On return, this parameter contains the text specified by the **selStart** and **selEnd** parameters.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

### InsertText

This method inserts text at the current position within the source file. You can use **put\_SelectionStart** to position the text insertion point.

```
InsertText (
 BSTR val);
```

**val**

The text to insert.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**Text**  
*put\_SelectionEnd*

---

## put\_SelectionEnd

This method moves the end of the selection block to the specified position, counted from the top of the source file.

```
put_SelectionEnd(
 int val);
```

val

The number of characters from the top of the file to the new end of the selection block.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_SelectionLineEnd

This method moves the end of the selection block to a specified line.

```
put_SelectionLineEnd(
 int val);
```

val

The line number of the new end of the selection block.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## put\_SelectionLineStart

This method moves the start of the selection block to a specified line.

```
put_SelectionLineStart(
 int val);
```



## Freescal Semiconductor, Inc.

### Text

#### *put\_SelectionStart*

---

val

The line number of the new start of the selection block.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### *put\_SelectionStart*

This method moves the start of the selection block to the specified position, counted from the top of the source file.

```
put_SelectionStart(
 int val);
```

val

The number of characters from the top of the file to the new start of the selection block.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

#### *put\_SelectionText*

This method replaces the current selection with a new block of text.

```
put_SelectionText(
 BSTR val);
```

val

The new block of text.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

# Toolbar

---

This chapter shows how to use the Toolbar API to create and manage buttons in the CodeWarrior IDE toolbar.

This chapter contains the following sections:

- Toolbar API Overview
- Toolbar API Reference

## Toolbar API Overview

The Toolbar API is a set of interfaces that lets a plug-in create and manipulate buttons in the CodeWarrior IDE toolbar. The API uses the standard COM .

## Toolbar API Reference

This section describes the methods contained in the following interfaces:

- ICodeWarriorCustomToolBarItem
- ICodeWarriorPopupMenuToolBarItem
- ICodeWarriorToggleButtonToolBarItem
- ICodeWarriorToolBar
- ICodeWarriorToolBarInstanceCreationNotification
- ICodeWarriorToolBarItemHelp
- ICodeWarriorToolBarItemRegistry

These interfaces use the following data types:

- SPopupMenuToolBarItem
- CWToolBarItemID



# Freescale Semiconductor, Inc.

**Toolbar**  
*Toolbar API Reference*

---

- CWToolbarIconRegistryInfo
- Item Flags



ICodeWarriorCustomToolbarItem

This interface exposes methods for creating, drawing, and getting information about a toolbar item.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                        |                            |
|------------------------|----------------------------|
| CreateItemControl      | GetItemRepresentationWidth |
| DrawItemRepresentation | GetItemSizeInfo            |

CreateItemControl

This method creates a new item to put in a toolbar.

```
virtual void* CreateItemControl(
 ICodeWarriorToolbar *inToolbar,
 void *hwndParent);
```

inToolbar

A pointer to the toolbar in which to create the new item.

hwndParent

A pointer to the window handle of the parent window.

Returns A pointer to the new item.



## Toolbar

### *DrawItemRepresentation*

---

---

## DrawItemRepresentation

This method draws the current item in a specified specified graphics context.

```
virtual HRESULT DrawItemRepresentation(
 void *inGraphicsContext,
 LONG xPos,
 LONG yPos,
 LONG width,
 LONG height);
```

*inGraphicsContext*

A pointer to the graphics context in which to draw the item.

*xPos*

The X position at which to draw the item.

*yPos*

The Y position at which to draw the item.

*width*

The width of the item.

*height*

The height of the item.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## GetItemRepresentationWidth

This method gets the width the current item occupies in a given graphics context.

```
virtual LONG GetItemRepresentationWidth(
 void *inGraphicsContext);
```

*inGraphicsContext*

A pointer to the graphics context.

**Returns** A long integer indicating the width the item occupies in the specified graphics context.

---

## GetItemSizeInfo

This method gets the size of the current item and whether the item can be resized.

```
virtual HRESULT GetItemSizeInfo(
 LONG &outMinWidth,
 BOOL &outResizable);
```

*outMinWidth*

On return, this parameter contains the address of a long integer indicating the size of the item.

*outResizable*

On return, this parameter contains the address of a boolean that is set to `true` if the item can be resized or `false` if not.

**Returns** `S_OK` if this method call succeeded or an appropriate error if it failed.



ICodeWarriorPopupMenuToolbarItem

This interface provides methods for creating and working with popup menus on a toolbar.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                      |                          |
|----------------------|--------------------------|
| BuildPopupMenuList   | GetItemWidth             |
| CleanupPopupMenuList | GetSampleTextString      |
| GetInitialState      | HandlePopupMenuSelection |

BuildPopupMenuList

This method creates a list of items to put in a popup menu.

```
virtual HRESULT BuildPopupMenuList(
 ICodeWarriorToolbar *inToolbar,
 void *inItemData,
 LONG inKeyboardModifiers,
 SPopupMenuToolbarItem *&outItems,
 LONG &outItemCount,
 LONG &outSelItem)

inToolbar
 A pointer to the toolbar on which to place the popup menu.

inItemData
 A pointer to the items to put on the menu.

inKeyboardModifiers
 A pointer to the hotkeys what select items on the menu.
```

outItems

On return, this parameter contains a pointer to the address of the items in the menu.

outItemCount

On return, this parameter contains the address of a long indicating how many items are in the menu.

outSelItem

On return, this parameter contains the address of the selected item.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorToolbar" on page 789  
"SPopupMenuToolbarItem" on page 801

---

## CleanupPopuItemList

This method resets a popup menu's item list.

```
virtual HRESULT CleanupPopuItemList(
 SPopupMenuToolbarItem *inItems,
 LONG inItemCount);
```

inItems

A pointer to the items to place in the menu.

inItemCount

The number of items the menu should have.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.



## Freescal Semiconductor, Inc.

### Toolbar

#### *GetInitialState*

---

#### GetInitialState

---

This method gets the initial state of the popup menu.

```
virtual HRESULT GetInitialState(
 ICodeWarriorToolbar *inToolbar,
 void *inItemData,
 STR &outCurrentStr,
 CWToolbarIconInfo& outIcon,
 BOOL &outIsEnabled)
```

*inToolbar*

A pointer to the toolbar on which the menu appears.

*inItemData*

A pointer to the items to put on the menu.

*outCurrentStr*

On return, this parameter contains the string value of the current item in the popup menu.

*outIcon*

On return, this parameter contains the address of information about the icon associated with the current item in the menu.

*outIsEnabled*

On return, this parameter contains the address of a boolean that is set to `true` if the menu is enabled or `false` if not.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "ICodeWarriorToolbar" on page 789

## GetItemWidth

This method gets the width of the current item.

```
virtual HRESULT GetItemWidth(LONG &outWidth);
```

outWidth

On return, this parameter contains the address of a long indicating the width of the current item.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## GetSampleTextString

This method gets a sample text string from the current popup menu.

```
virtual HRESULT GetSampleTextString(
 BSTR &outSampleStr);
```

outSampleStr

On return, this parameter contains the sample string.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Toolbar

### *HandlePopupSelection*

---

## HandlePopupSelection

---

The IDE calls this method to let the plug-in know to perform the action associated with a popup on a toolbar.

```
virtual HRESULT HandlePopupSelection(
 void *inItemData
 ICodeWarriorToolbar *inToolbar,
 LONG itemIndex,
 SPopupMenuToolbarItem *inItems,
 LONG inItemCount);
```

*inToolbar*

A pointer to the toolbar on which the popup resides.

*inItemData*

A pointer to the items on the toolbar

*itemIndex*

The item number of the item whose event you want to dispatch.

*inItems*

A pointer to the list of choices in the popup.

*inItemCount*

The number of the item (in the list specified by the *inItems* parameter) selected by the user.

Returns

S\_OK if this method call succeeded or an appropriate error if it failed.

## ICodeWarriorToggleButtonToolbarItem

This interface exposes a method that lets you determine if a toggle button on a toolbar has been pressed.

### Inherited Interfaces

- IUnknown

### Methods

The following methods are available for your use:

|                 |              |
|-----------------|--------------|
| GetInitialState | StateChanged |
|-----------------|--------------|

### GetInitialState

This method gets the initial state of a toggle button on a toolbar.

```
virtual HRESULT GetInitialState(
 ICodeWarriorToolbar *inToolbar,
 void *inItemData,
 CWToolbarItemID inItemID,
 BOOL& outInitialState,
 BOOL& outEnabled)
```

inToolbar

A pointer to the toolbar on which the toggle button appears.

inItemData

A pointer to the item data for the toggle button.

outInitialState

On return, this parameter contains the address of a boolean set to true if the toggle button is in its selected state (toggle is on) or false if it is in its deselected state (toggle is off).

outEnabled

On return, this parameter contains the address of a boolean set to true if the toggle button is enable or false if it is disabled.



## Freescal Semiconductor, Inc.

### Toolbar

#### *StateChanged*

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### StateChanged

This method sets the state of a toggle button to on or off.

```
virtual HRESULT StateChanged(
 ICodeWarriorToolbar *inToolbar,
 CWToolbarItemID inItemID,
 BOOL inNewState);
```

*inToolbar*

A pointer to the toolbar on which the toggle button appears.

*inItemID*

The ID of

*inNewState*

Set this parameter to true to put the toggle button in its selected (on) state or false to set the toggle button in its deselected (off) state.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorToolbar

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|                       |                       |
|-----------------------|-----------------------|
| GetContainingDocument | SetToolbarItemEnabled |
| GetToolbarHeight      | SetToolbarItemIcon    |
| GetToolbarItemText    | SetToolbarItemText    |
| GetToolbarItemValue   | SetToolbarItemValue   |
| IsToolbarVisible      | ShowToolbar           |
| ResetToolbarItem      |                       |

GetContainingDocument

This method gets the document that contains the toolbar.

```
virtual ICodeWarriorDocumentPrivate*
 GetContainingDocument(void);
```

Returns A pointer to the document that contains the toolbar.

See Also "ICodeWarriorDocument" on page 572

GetToolbarHeight

This method gets the height of the toolbar.

```
virtual LONG GetToolbarHeight(void);
```

Returns The height of the toolbar.

## Toolbar

### *GetToolbarItemText*

---

## GetToolbarItemText

This method gets the text label for a specified toolbar item.

```
virtual HRESULT GetToolbarItemText(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
 BSTR &outItemText);
```

**inPluginID**

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

**inItemID**

The toolbar item for which to get the text label.

**outItemText**

On return, this parameter contains the text of the toolbar item

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "CWToolbarItemID" on page 801

---

## GetToolbarItemValue

This method gets the value of a specified item in the toolbar

```
virtual HRESULT GetToolbarItemValue(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
 LONG &outValue);
```

**inPluginID**

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

`inItemID`

The toolbar item for which to get the value.

`outValue`

On return, this parameter contains the address of a long indicating the value of the toolbar item specified by the `inItemID` parameter.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "CWToolbarItemID" on page 801

---

## IsToolbarVisible

This method gets whether the current toolbar is visible.

```
virtual BOOL IsToolbarVisible() = 0;
```

**Returns** true if the current toolbar is visible or false if not.

---

## ResetToolbarItem

This method resets a toolbar item to its original state.

```
virtual HRESULT ResetToolbarItem(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID);
```

`inPluginID`

TheGUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

`inItemID`

The toolbar item to reset.

## Toolbar

### *SetToolbarItemEnabled*

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “CWToolbarItemID” on page 801

---

## SetToolbarItemEnabled

This method sets whether an item in the toolbar is enabled.

```
virtual HRESULT SetToolbarItemEnabled(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
 BOOL inIsEnabled);
```

*inPluginID*

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

*inItemID*

The toolbar item to enable or disable.

*inIsEnabled*

Set this parameter to true to enable the toolbar item or false to disable it.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “CWToolbarItemID” on page 801

---

## SetToolbarItemIcon

This method sets the icon for a specified item in the toolbar.

```
virtual HRESULT SetToolbarItemIcon(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
```

---

```
const CWToolbarIconInfo inIconData);
```

```
inPluginID
```

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

```
inItemID
```

The toolbar item for which to set icon information.

```
inIconData
```

The icon data for the icon associated with the specified toolbar item.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "CWToolbarItemID" on page 801

---

## SetToolbarItemText

This method sets the text label for a specified item in the toolbar.

```
virtual HRESULT SetToolbarItemText(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
 BSTR inNewText);
```

```
inPluginID
```

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

```
inItemID
```

The toolbar item for which to set the text label.

```
inNewText
```

The text to which to set the text label of the item specified in the inItemID parameter.

## Toolbar

### SetToolbarItemValue

---

- Returns S\_OK if this method call succeeded or an appropriate error if it failed.
- See Also "CWToolbarItemID" on page 801

---

### SetToolbarItemValue

This method sets the value of a specified item in the toolbar.

```
virtual HRESULT SetToolbarItemValue(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
 LONG inValue);
```

inPluginID

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

inItemID

The toolbar item for which to set the value.

inValue

The value to set for the toolbar item specified in inItemID.

- Returns S\_OK if this method call succeeded or an appropriate error if it failed.
- See Also "CWToolbarItemID" on page 801

---

### ShowToolbar

This method sets whether to show the current toolbar.

```
virtual HRESULT ShowToolbar(
 BOOL inShow) = 0;
```



`inShow`  
`true` to show the current toolbar is visible or `false` to not show it.

Returns `S_OK` if this method call succeeded or an appropriate error if it failed.



ICodeWarriorToolbarInstanceCreationNotification

This interface provides methods for determining whether an item has been created or destroyed.

Inherited Interfaces

- IUnknown

Methods

The following methods are available for your use:

|             |               |
|-------------|---------------|
| ItemCreated | ItemDestroyed |
|-------------|---------------|

ItemCreated

This method lets you find out whether a new item has been created on a toolbar.

```
virtual HRESULT ItemCreated(
 ICodeWarriorToolbar *inToolbar,
 void *&outItemData);
```

inToolbar

A pointer to the toolbar on which to create a new item.

outItemData

On return, this parameter contains a pointer to a reference for the new item's information.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

ItemDestroyed

This method notifies you when an item has been removed from a toolbar.

```
virtual HRESULT ItemDestroyed(
 ICodeWarriorToolbar *inToolbar,
 void *inItemData);
```

inToolbar

A pointer to the toolbar you want to be

inItemData

A pointer to the destroyed item.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

## Toolbar

### ICodeWarriorToolbarItemHelp

---

## ICodeWarriorToolbarItemHelp

This interface provides a way to get the help text for a toolbar item.

### Inherited Interfaces

- IUnknown

The following method is available for your use:

- GetHelpString

---

## GetHelpString

This method gets the help string for a specified toolbar item.

```
virtual HRESULT GetHelpString(
 CWToolbarItemID itemID,
 BSTR &outHelpString);
```

itemID

The ID of the item for which to get the help string.

outHelpString

On return, this parameter contains the help string.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorToolbarItemRegistry

This interface provides methods for creating registry items for toolbar items and icons.

Inherited Interfaces

- IUnknown

The following methods are available for your use:

|                      |                     |
|----------------------|---------------------|
| RegisterToolbarIcons | RegisterToolbarItem |
|----------------------|---------------------|

RegisterToolbarIcons

This method creates a registry entry a toolbar icon.

```
virtual HRESULT RegisterToolbarIcons(
 const CWPluginID inPluginID,
 const CWToolbarIconRegistryInfo &inIconData);
```

inPluginID

The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

inIconData

The address of the icon data to register.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also CWToolbarIconRegistryInfo



**Toolbar**  
*RegisterToolbarItem*

---

**RegisterToolbarItem**

This method registers an item in a toolbar.

```
virtual HRESULT RegisterToolbarItem(
 const CWPluginID inPluginID,
 const CWToolbarItemID inItemID,
 const long inItemType,
 const CWToolbarIconInfo inIconData,
 const BSTR inItemName,
 IUnknown *itemHandler);
```

**inPluginID**  
The GUID for the plug-in. Usually this is the class ID of the main class of your plug-in.

**inItemID**  
The ID of the item to register.

**inItemType**  
The Type of the item to register.

**inIconData**  
The icon data associated with the item.

**inItemName**  
The name of the item

**itemHandler**  
A pointer to the handler for the item.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "CWToolbarItemID" on page 801

## Toolbar Data Types

### SPopupMenuToolbarItem

The following enumeration (from CodeWarriorToolbar.h) defines the toolbar items used in the CodeWarrior Toolbar API:

---

```
struct SPopupMenuToolbarItem
{
 unsigned long itemFlags;
 BSTR itemText;
 void *userData;
 CWToolbarIconInfo itemIcon;
};
```

---

### CWToolbarItemID

The following enumeration (from CodeWarriorToolbar.h) defines the toolbar item IDs used in the CodeWarrior Toolbar API:

---

```
typedef long CWToolbarItemID;
enum
{
 tbItemType_Separator,
 tbItemType_CommandButton,
 tbItemType_ToggleButton,
 tbItemType_CheckBox,
 tbItemType_PopupButton,
 tbItemType_PopupList, // Bevel button w/text on MacOS
 // Combo box on Windows
 tbItemType_Custom
};
```

---



# Freescale Semiconductor, Inc.

## Toolbar

Toolbar Data Types

### CWToolbarIconRegistryInfo

The following structure (from CodeWarriorToolbar.h) defines the toolbar registry information used in the CodeWarrior Toolbar API:

```
#if defined(macintosh) || defined(_LATITUDE_)
typedef void* CWToolbarIconRegistryInfo;
#elif defined(WIN32)
typedef struct
{
 HBITMAP hotImages;
 HBITMAP normalImages;
 COLORREF maskColor;
} CWToolbarIconRegistryInfo;
#endif
```

### Item Flags

The following item flags are defined in CodeWarriorToolbar.h for use with the CodeWarrior toolbar API:

|                   |   |
|-------------------|---|
| CWPopup_Checked   | 1 |
| CWPopup_Disabled  | 2 |
| CWPopup_Underline | 4 |

# Version Control

---

This chapter shows how to use the Version Control API to work with the Version Control system in the CodeWarrior IDE.

This chapter contains the following sections:

- Version Control API Reference

## Version Control API Reference

This section describes the functions contained in the following interfaces:

- ICodeWarriorVersionControl
- ICodeWarriorVCSState
- ICodeWarriorVCSFileStateListener

The VCS interfaces use the following data types:

- ECodeWarriorVCSCKIDState
- ECodeWarriorVCSDBState
- ECodeWarriorVCSFileLockState



ICodeWarriorVersionControl

This interface provides methods for the basic Version Control operations (checking in, checking out, and so on).

Inherited Interfaces

- IUnknown

This interfaces exposes the following methods:

|            |              |
|------------|--------------|
| CheckIn    | get_Name     |
| CheckOut   | GetVCSState  |
| Connect    | IsConnected  |
| Disconnect | UndoCheckOut |
| Get        | UnLock       |

CheckIn

This method checks in the files in a specified collection of files.

```
virtual HRESULT CheckIn(
 IFileSpecCollection *fileSpecCollection);
```

fileSpecCollection

A pointer to a collection object containing the files to check in.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

CheckOut

This method checks out the files in a specified collection of files.

```
virtual HRESULT CheckOut(

```



---

```
IFileSpecCollection *fileSpecCollection);
```

```
fileSpecCollection
```

A pointer to a collection object containing the files to check out.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487

---

### Connect

This method connects to the Version Control database.

```
virtual HRESULT Connect(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### Disconnect

This method disconnects from the Version Control database.

```
virtual HRESULT Disconnect(void);
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### Get

This method gets a collection of files from the Version Control database.

```
virtual HRESULT Get(
 IFileSpecCollection *fileSpecCollection);
```

## Version Control

### *get\_Name*

---

`fileSpecCollection`

A pointer to a collection object containing the files toget.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "Using the Collections API" on page 487

---

### *get\_Name*

This method gets the name of the Version Control system.

```
virtual HRESULT get_Name(
 BSTR *vcsName);
```

`vcsName`

On return, this parameter contains the name of the Version Control system.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### *GetVCSState*

This method gets the Version Control state for a specified file.

```
virtual HRESULT GetVCSState(
 IFileSpec *fileSpec,
 ICodeWarriorVCSState **vcsState);
```

`fileSpec`

A pointer to the file specification for which to get

`vcsState`

On return, this parameter contains the address of a pointer to the state of the file specified by the `fileSpec` parameter.

---

|          |                                                                          |
|----------|--------------------------------------------------------------------------|
| Returns  | S_OK if this method call succeeded or an appropriate error if it failed. |
| See Also | “IFileSpec” on page 599                                                  |

---

### UnLock

This method unlocks the files in a collection of files.

```
virtual HRESULT UnLock(
 IFileSpecCollection *fileSpecCollection);

fileSpecCollection
```

A pointer to the collection of files to unlock.

|          |                                                                          |
|----------|--------------------------------------------------------------------------|
| Returns  | S_OK if this method call succeeded or an appropriate error if it failed. |
| See Also | “Using the Collections API” on page 487                                  |

---

### IsConnected

This method gets whether a connection to the Version Control database. You might want to check the result of this method before calling Connect or Disconnect.

```
virtual HRESULT IsConnected(
 VARIANT_BOOL *pval);

pval
```

On return, this parameter contains a pointer to a boolean that is set to true if a connection to the Version Control database exists or false is no connection exists.

|         |                                                                          |
|---------|--------------------------------------------------------------------------|
| Returns | S_OK if this method call succeeded or an appropriate error if it failed. |
|---------|--------------------------------------------------------------------------|

---



Version Control  
UndoCheckOut

---

UndoCheckOut

This method performs an UndoCheckOut operation (essentially, it restores files to their previous version) on a collection of files. Using it may cause changes to be lost.

```
virtual HRESULT UndoCheckOut(
 IFileSpecCollection *fileSpecCollection);

fileSpecCollection
```

A pointer to the collection of files on which to perform an UndoCheckOut operation.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “Using the Collections API” on page 487



ICodeWarriorVCSState

This interface provides methods that let you can use to check on the state of the Version Control system.

Inherited Interfaces

- IUnknown

Methods

The interface exposes the following methods:

|                   |             |
|-------------------|-------------|
| get_CKIDState     | get_DBState |
| get_FileLockState |             |

get\_CKIDState

This method gets the Version Control state (checked in, checked out, not in the Version Control system, and so on) for the current file.

```
virtual HRESULT get_CKIDState(
 ECodeWarriorVCSCKIDState *type);
```

type

On return, this parameter contains a pointer to the state of the file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorVCSCKIDState” on page 812



## Freescale Semiconductor, Inc.

### Version Control

#### *get\_DBState*

---

#### get\_DBState

---

This method gets the state of the Version Control database.

```
virtual HRESULT get_DBState(
 ECodeWarriorVCSDBState *type);
```

type

On return, this parameter contains a pointer to the state of the database.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorVCSDBState” on page 813

---

#### get\_FileLockState

---

This method gets the lock state for the current file.

```
virtual HRESULT get_FileLockState(
 ECodeWarriorVCSFileLockState *type);
```

type

On return, this parameter contains a pointer to the lock state of the current file.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “ECodeWarriorVCSFileLockState” on page 813

---

## ICodeWarriorVCSFileStateListener

This interface provides a method that lets you monitor the state of a file in the Version Control system.

### Inherited Interfaces

- IUnknown

### Methods

This interface exposes the following method:

- StateChanged

---

### StateChanged

This method sends a message all listeners of this interface when a file's VCS state has changed.

```
virtual HRESULT StateChanged(
 IFileSpec *fileSpec,
 ICodeWarriorVCSState *vcsState);
```

*fileSpec*

The file specification for the file whose state has changed.

*vcsState*

The new state of the file.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** "IFileSpec" on page 599  
"ICodeWarriorVCSState" on page 809

VCS Data Types

The following data types are used with the VCS API:

- ECodeWarriorVCSCKIDState
- ECodeWarriorVCSDBState
- ECodeWarriorVCSFileLockState

ECodeWarriorVCSCKIDState

This enumeration describes whether a file is in the version control system and its state within the version control system.

**NOTE** CKID only has meaning on the Mac OS, so this enumeration has no meaning on any other operating system.

Table 27.1 ECodeWarriorVCSCKIDState Enumeration

| Constant          | Description                                                              |
|-------------------|--------------------------------------------------------------------------|
| vcsCKIDNotChecked | The file is not in the version control system.                           |
| vcsNoCKID         | No CKID is available.                                                    |
| vcsCKIDCheckedIn  | The file is in the version control system and is currently checked in.   |
| vcsCKIDCheckedOut | The file is in the version control system and is currently checked out.  |
| vcsCKIDMRO        | The file is in the version control system and is marked read-only (MRO). |

ECodeWarriorVCSDBState

This enumeration describes the state of a file in the version control database.

Table 27.2 ECodeWarriorVCSDBState Enumeration

| Constant             | Description                                                                                                                                                                          |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| vcsDBNotChecked      | The file has not been checked for its VCS state. This state can appear in in a dialog box or message window if the version control system was unable to check the state of the file. |
| vcsDBNotInWorkingDir | The file is not in the working directory defined in the VCS Preference Panel.                                                                                                        |
| vcsDBNotInDatabase   | The file is not in the VCS database.                                                                                                                                                 |
| vcsDBCheckedIn       | The file is in the VCS database and is checked in                                                                                                                                    |
| vcsDBCheckedOut      | The file is in the VCS database and is checked out.                                                                                                                                  |

ECodeWarriorVCSFileLockState

This enumeration describes the lock state of a file, from the perspective of the version control system.

Table 27.3 ECodeWarriorVCSFileLockState Enumeration

| Constant          | Description                                                                     |
|-------------------|---------------------------------------------------------------------------------|
| vcsLockNotChecked | This file is not in the version control system.                                 |
| vcsVolLocked      | The volume in which this file resides is locked (as is the file, by extension). |
| vcsFileLocked     | The file is locked.                                                             |
| vcsFileReadOnly   | The file is read-only.                                                          |



# Freescal Semiconductor, Inc.

## Version Control

*VCS Data Types*

---

# Windows

---

This chapter shows how to use the Windows API to create and manage windows in the CodeWarrior IDE.

This chapter contains the following sections:

- Windows API Overview
- Using the Windows API
- Windows API Reference

## Windows API Overview

The Windows API is a set of interfaces that allows a plug-in to create and manipulate windows in the CodeWarrior IDE. The API uses the standard COM interfaces.

Windows are registered through standard commands (menus) and allow setting of standard window attributes.

Windows events are platform-specific and should be handled accordingly.

## Using the Windows API

You will need to create an `ICodeWarriorMenuManager` interface and a call to `QueryInterface()` to get this interface. Once you have created the window you will attach an event/command handler to the window. The latter is done when registering commands with `RegisterCommand()`.



**Windows**

*Windows API Reference*

---

## **Windows API Reference**

This section describes the methods contained in the following interfaces:

- `ICodeWarriorWindowManager`
- `ICodeWarriorWindow`
- `ICodeWarriorWindowEvents`

These interfaces use the following data type:

- `CWNativeXWindowPart`



ICodeWarriorWindowManager

This interface is provided to allow plug-ins to create windows in the CodeWarrior IDE.

Inherited Interfaces

- IUnknown

Methods

This interface implements the following methods:

|                         |                  |
|-------------------------|------------------|
| CenterWindow            | GetIDEMainWindow |
| CreateCodeWarriorWindow | IsIDEInMDIMode   |

CenterWindow

This method centers the selected window over the main window.

This method centers a specified window on the client screen or centers the main window if the IDE is in MDI mode.

```
virtual HRESULT CenterWindow(CWNativeWindowType
 window, BOOL fIsDialog, int reserved) = 0;
```

window

The CodeWarrior window you want to center on the screen.

fIsDialog

Set this flag to true if the window is a dialog or false otherwise.

reserved

Set this parameter to 0.



Windows

CreateCodeWarriorWindow

CreateCodeWarriorWindow

Call this method to create a new window in the CodeWarrior IDE. You can use this interface to access window methods. You will also need an event handler for your window.

```
virtual ICodeWarriorWindow*
CreateCodeWarriorWindow(
 const CWPluginID inPluginID,
 ULONG inAttributes) = 0;
```

inPluginID  
The ID for the plug-in.

inAttributes  
Attributes that describe the type of window you want. Multiple attributes can be set for any window. The following attributes are allowed:  
Attributes for all platforms:

|                               |                                                                     |
|-------------------------------|---------------------------------------------------------------------|
| CWWindow_CanClose             | The window has a close box                                          |
| CWWindow_CanResize            | The window is resizable                                             |
| CWWindow_Floating             | The window floats above all other IDE windows                       |
| CWWindow_PutInOpenWindowsMenu | The window name is put in the available windows in the Window menu. |
| CWWindow_Modal                | The window is modal                                                 |



Mac OS specific attributes:

|                        |                                                                                                                                                               |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CWindow_GetIdleTime    | The window event handler's Idle method will be called                                                                                                         |
| CWindow_GetSelectClick | true if a click both brings a window forward and affects its content or false if a click only brings a window forward. This setting works only on the Mac OS. |
| CWindow_DelaySelect    | true to bring a window forward on a mouse-up event or false to bring a window forward on a mouse-down event. This setting works only on the Mac OS.           |

Returns A pointer to an ICodeWarriorWindow object.

See Also "ICodeWarriorWindow" on page 821

IsIDEInMDIMode

This method is used to determine if the IDE is in Multiple Document Interface (MDI) mode.

virtual BOOL IsIDEInMDIMode()

Returns true if window is in IDE MDI Mode or false otherwise. If this method returns false, the IDE is in Floating Document Interface (FDI) mode.



**Windows**  
*GetIDEMainWindow*

---

GetIDEMainWindow

This method gets a window handle for the IDE’s main window.

**NOTE** This method works only on Win32.

---

```
virtual HRESULT GetIDEMainWindow(
 HWND *mainWnd)
```

mainWnd

On return, this parameter contains a pointer to the IDE’s main window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### ICodeWarriorWindow

This interface is provided to allow plug-ins to manipulate windows in the CodeWarrior IDE.

#### Inherited Interfaces

- IUnknown

#### Methods

The following methods are available for your use:

|                                  |                                   |
|----------------------------------|-----------------------------------|
| AssociateWindowWithProject       | SetBackBrushes                    |
| CreateToolbar                    | SetCodeWarriorWindowInitialBounds |
| DestroyCodeWarriorWindow         | SetCodeWarriorWindowMinMaxSize    |
| GetCodeWarriorWindowSizeLocation | SetCodeWarriorWindowTitle         |
| GetNativeWindowReference         | SetDialogColors                   |
| GetNativeXWindowReference        | SetEventHandler                   |
| GetWindowToolbar                 | SetMaximumSleepTime               |
| HasAttribute                     | SelectCodeWarriorWindow           |
| MoveCodeWarriorWindow            | ShowCodeWarriorWindow             |
| PutBehind                        | UpdatePort                        |
| ReorderCodeWarriorWindow         |                                   |

**NOTE** The PutBehind, ReorderCodeWarriorWindow, SetBackBrushes, SetDialogColors, and SetMaximumSleepTime methods work only on the Mac OS. The GetNativeXWindowReference method works only on unix-based systems.



## Windows

### *AssociateWindowWithProject*

---

## AssociateWindowWithProject

This method associates a window with a project. If the window is in front and the user performs actions (for example, Compile) related to a project, the project associated with the window becomes associated with that window.

```
virtual HRESULT AssociateWindowWithProject(
 IUnknown *inProjectObject)
```

*inProjectObject*

The project your window is to be associated with.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## CreateToolbar

Creates a new toolbar in the window. The IDE will call the event handler's `GetDefulatToobbarItems()`.

```
virtual HRESULT CreateToolbar(
 BSTR inToolbarTitle,
 ICodeWarriorToolbar *&outToolbar) = 0;
```

*inToolbarTitle*

The title of the new toolbar.

*outToolbar*

The IDE will provide a pointer to the toolbar interface.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also "ICodeWarriorToolbar" on page 789

---

## DestroyCodeWarriorWindow

This method closes the current window.

```
virtual HRESULT DestroyCodeWarriorWindow()
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## GetCodeWarriorWindowSizeLocation

This method gets the size and location of the window.

```
virtual HRESULT GetCodeWarriorWindowSizeLocation(
 SHORT &xPos,
 SHORT &yPos,
 SHORT &width,
 SHORT &height) = 0;
```

xPos

The horizontal position of the upper left-hand corner of the window.

yPos

The vertical position of the upper left-hand corner of the window.

width

The width of the window.

height

The height of the window.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



# Freescal Semiconductor, Inc.

## Windows

### GetNativeWindowReference

---

### GetNativeWindowReference

This method obtains the window reference for the window, according to the platform the plug-in is running on. Once you have this reference, you can use native OS methods to draw whatever you need.

```
virtual CWNativeWindowType
 GetNativeWindowReference()
```

Returns A value of type `CWNativeWindowType`, which is either a Windows `HWND` structure, or a Mac OS `WindowPtr` structure, depending on the target platform.

---

### GetNativeXWindowReference

This method gets a native reference to a window on Unix-based operating systems.

---

**NOTE** This method works only on Unix-based operating systems.

---

```
virtual void * GetNativeXWindowReference(
 CWNativeXWindowPart part) = 0;
```

part

A value within the range specified by the `CWNativeXWindowPart` enumeration.

Returns A pointer to the window reference.

See Also “`CWNativeXWindowPart`” on page 840

## GetWindowToolbar

This method gets the toolbar associated with the current window, if any.

```
virtual ICodeWarriorToolbar*
 GetWindowToolbar()
```

**Returns** A pointer to an ICodeWarriorToolbar interface.

**See Also** "ICodeWarriorToolbar" on page 789

---

## HasAttribute

This method discovers whether the current window has a specified attribute.

```
virtual BOOL HasAttribute(
 ULONG inAttribute)
```

*inAttribute*

An unsigned long integer indicating the attribute for which to check..

**Returns** true if the current window has the specified attribute or false if not.

---

## MoveCodeWarriorWindow

This method changes the size and location of the window.

```
virtual HRESULT MoveCodeWarriorWindow(
 SHORT xPos,
 SHORT yPos,
 SHORT width,
 SHORT height,
```

## Windows PutBehind

---

|         |                                                                                                                                                                               |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         | <code>BOOL refresh) = 0;</code>                                                                                                                                               |
|         | <code>xPos</code>                                                                                                                                                             |
|         | The horizontal position of the upper left-hand corner of the window.                                                                                                          |
|         | <code>yPos</code>                                                                                                                                                             |
|         | The vertical position of the upper left-hand corner of the window.                                                                                                            |
|         | <code>width</code>                                                                                                                                                            |
|         | The width of the window.                                                                                                                                                      |
|         | <code>height</code>                                                                                                                                                           |
|         | The height of the window.                                                                                                                                                     |
|         | <code>refresh</code>                                                                                                                                                          |
|         | Set this parameter to <code>true</code> to generate an update event for the content of the window. Set it to <code>false</code> to leave the content of the window unchanged. |
| Returns | <code>S_OK</code> if this method call succeeded or an appropriate error if it failed.                                                                                         |

---

## PutBehind

This method places the current window behind another window.

---

**NOTE** This method works only on the Mac OS and Windows.

---

```
virtual HRESULT PutBehind(
 CWNativeWindowType inBehindWindow);
```

`inBehindWindow`

The window to place the current window behind. This parameter must be either a Windows `HWND` structure or a Mac OS `WindowPtr` structure, depending on the target

platform.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### ReorderCodeWarriorWindow

This method reorders the current window.

**NOTE** This method works only on the Mac OS.

```
virtual HRESULT ReorderCodeWarriorWindow(
 ULONG reorderType) = 0;
```

reorderType

An unsigned long integer indicating the type of reordering operation to perform. See the Mac programmer's documentation for more detail.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### SetBackBrushes

This method sets the back brush colors for windows for appearance-savvy versions of the Mac OS.

**NOTE** This method works only on the Mac OS.

```
virtual HRESULT SetBackBrushes(
 ThemeBrush inActiveBackBrush,
 ThemeBrush inInactiveBackBrush)
```

inActiveBackBrush

Describes the appearance of a window while it is in the

## Windows

### *SetCodeWarriorWindowInitialBounds*

---

background. See the Mac Toolbox ThemeBrush constants for more information.

inInactiveBackBrush

Describes the state of the window when it is in front. See the Mac Toolbox ThemeBrush constants for more information.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SetCodeWarriorWindowInitialBounds

This method sets the initial size and position of the window when it is displayed.

```
virtual HRESULT SetCodeWarriorWindowInitialBounds(
 SHORT xPos,
 SHORT yPos,
 SHORT width,
 SHORT height)
```

xPos

The horizontal position of the window, in pixels.

yPos

The vertical position of the window, in pixels.

width

The width of the window, in pixels.

height

The height of the window, in pixels.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

## SetCodeWarriorWindowMinMaxSize

This method sets the minimum and maximum allowable sizes for a window. Users will not be able to resize the window smaller than the minimum size or larger than the maximum size.

```
virtual HRESULT SetCodeWarriorWindowMinMaxSize(
 SHORT minWidth,
 SHORT maxWidth,
 SHORT minHeight,
 SHORT maxHeight) = 0;
```

`minWidth`

The minimum width of the window, in pixels.

`maxWidth`

The maximum width of the window, in pixels.

`minHeight`

The minimum height of the window, in pixels.

`maxHeight`

The maximum height of the window, in pixels.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## SetCodeWarriorWindowTitle

This method sets the title of the window.

```
virtual HRESULT SetCodeWarriorWindowTitle(
 BSTR newTitle)
```

`newTitle`

The new window title.



# Freescale Semiconductor, Inc.

## Windows

### SetDialogColors

---

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SetDialogColors

This method sets the dialog colors for windows for non-appearance-savvy versions of the Mac OS.

---

**NOTE** This method works only on the Mac OS.

---

virtual HRESULT SetDialogColors()

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SetEventHandler

This method sets the event handler for the window. The event handler is called for each event the window receives.

virtual HRESULT SetEventHandler(  
    IUnknown \*inEventHandler)

\*inEventHandler

The event handler for the window. This should be an existing ICodeWarriorWindowEvents object or existing ICodeWarriorCommandHandler object.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

### SetMaximumSleepTime

This method sets the maximum sleep time for the current window.

---

**NOTE** This method works only on the Mac OS.

```
virtual HRESULT SetMaximumSleepTime (
 ULONG inSleepTime) = 0;

inSleepTime
 An unsigned long specifying the maximum sleep time for the
 current window. The unit of time is a Mac OS "tick" (1/60 of a
 second).
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### SelectCodeWarriorWindow

This method selects the window and brings it to the front, if necessary.

```
virtual HRESULT SelectCodeWarriorWindow()
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

### ShowCodeWarriorWindow

This method shows or hides the window.

```
virtual HRESULT ShowCodeWarriorWindow(BOOL
 visible) = 0;
```

```
visible
 Set this field to true if you want the IDE to make the window
 visible or false if not.
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



Windows

UpdatePort

---

UpdatePort

This method immediately updates the contents of the window that would normally be updated during the window’s next update event.

---

**NOTE** This method works only on the Mac OS.

---

```
virtual HRESULT UpdatePort() = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



ICodeWarriorWindowEvents

This interface is provided to allow plug-ins to respond to window events.

Inherited Interfaces

- IUnknown

Methods

The following methods apply to windows on all platforms:

|                        |                   |
|------------------------|-------------------|
| ActivateEvent          | OkToClose         |
| AdjustCursor           | PreBeginUpdate    |
| DeactivateEvent        | ToolbarSizeChange |
| GetDefaultToolbarItems | UpdateEvent       |
| Idle                   | WindowDestroyed   |
| KeyDownEvent           | WindowResizedBy   |
| MouseDownEvent         |                   |

**NOTE** The AdjustCursor, Idle, KeyDownEvent, MouseDownEvent, PreBeginUpdate, UpdateEvent, and WindowResizedBy methods work only on the Mac OS.

ActivateEvent

This method activates window events.

virtual HRESULT ActivateEvent()

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

See Also “DeactivateEvent” on page 834

## Windows

### *AdjustCursor*

---

## AdjustCursor

The method adjusts a cursor over a window on Mac OS.

```
virtual HRESULT AdjustCursor(
 Point inPortPt,
 const EventRecord&inMacEvent)
```

*inPortPt*

Specifies the location of the mouse in a window.

*EventRecord*

Specifies the event that occurred in the window. See the Mac OS Toolbox *EventRecord* structure for more information.

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

---

## DeactivateEvent

This method deactivates window events.

```
virtual HRESULT DeactivateEvent()
```

**Returns** S\_OK if this method call succeeded or an appropriate error if it failed.

**See Also** *ActivateEvent*

---

## GetDefaultToolbarItems

This method gets the default toolbar items for a window.

```
virtual HRESULT GetDefaultToolbarItems(
 const SDefaultToolbarItemInfo *&items,
 long &itemCount) = 0;
```

|         |                                                                          |
|---------|--------------------------------------------------------------------------|
|         | items                                                                    |
|         | The registered toolbar items.                                            |
|         | itemCount                                                                |
|         | The number of toolbar items.                                             |
| Returns | S_OK if this method call succeeded or an appropriate error if it failed. |

## Idle

This method handles idle events for a window on the Mac OS.

**NOTE** This method works only on the Mac OS.

|         |                                                                                           |
|---------|-------------------------------------------------------------------------------------------|
|         | virtual HRESULT Idle(<br>const EventRecord &idleEvent);                                   |
|         | idleEvent                                                                                 |
|         | A Mac OS event record. See the Mac OS Toolbox EventRecord structure for more information. |
| Returns | S_OK if this method call succeeded or an appropriate error if it failed.                  |

## KeyDownEvent

This method handles key down events for window on Mac OS.

**NOTE** This method works only on the Mac OS.

|  |                                                            |
|--|------------------------------------------------------------|
|  | virtual BOOL KeyDownEvent(<br>const EventRecord &keyEvent) |
|--|------------------------------------------------------------|



# Freescale Semiconductor, Inc.

## Windows

### MouseDownEvent

---

keyEvent

This field indicates the key event that was recieved by the window. See the Mac OS Toolbox `EventRecord` structure for more information.

Returns    `true` if the event was handled successfully, otherwise `false`.

---

### MouseDownEvent

This method handles a mouse down event within a window on the Mac OS.

---

**NOTE**    This method works only on the Mac OS.

---

```
virtual HRESULT STDMETHODCALLTYPE
MouseDownEvent(const EventRecord &mouseEvent,
 BOOL &delaySelect)
```

mouseEvent

This field indicates the type of mouse event received within a window. See the Mac OS Toolbox `EventRecord` structure for more information.

delaySelect

`true` to have the window be selected after a mouse-down event or `false` if not.

Returns    `S_OK` if this method call succeeded or an appropriate error if it failed.



---

## OkToClose

The CodeWarrior IDE calls this method to determine whether it is OK to close the window. This method can be used to determine if changes should be saved or not when closing a window.

```
virtual BOOL OkToClose()
```

Returns true if it is OK to close the window or false if not.

---

## PreBeginUpdate

The IDE calls this method to inform a plug-in that the IDE is about to update the window.

---

**NOTE** This method works only on the Mac OS.

---

```
virtual HRESULT PreBeginUpdate() = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

## ToolbarSizeChange

This method changes the height of the toolbar for the window.

```
virtual HRESULT ToolbarSizeChange(
 SHORT inNewHeight) = 0;
```

inNewHeight

The new height for the toolbar.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



**Windows**  
*UpdateEvent*

---

**UpdateEvent**

This method update an event in a window on the Mac OS.

**NOTE** This method works only on the Mac OS.

---

```
virtual HRESULT UpdateEvent(
 const RgnHandle updateRgn) = 0;

updateRgn
```

A window area specified by the updateRgn field. See the Mac OS Toolbox RgnHandle structure for more information.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.

---

**WindowDestroyed**

The CodeWarrior IDE calls this method when the window handle structure has been destroyed. General cleanup and memory release is done within this mehtod.

```
virtual HRESULT WindowDestroyed() = 0;
```

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



---

WindowResizedBy

This method specifies how much a window’s dimension has changed after a resize.

```
virtual HRESULT WindowResizedBy(
 SHORT inDeltaH,
 SHORT inDeltaV)
```

inDeltaH

The change in horizontal direction from ititial state to final state.

inDeltaV

The change in vertical direction from initial state to final state.

Returns S\_OK if this method call succeeded or an appropriate error if it failed.



Windows Data Types

The following data types are used with the Windows API:

- CWNativeXWindowPart

CWNativeXWindowPart

This enumeration describes the type of a user tree.

Table 28.1 CWNativeXWindowPart Enumeration

| Constant      | Description                      |
|---------------|----------------------------------|
| CW_X_WINDOW   | The window.                      |
| CW_X_DRAWABLE | The drawable part of the window. |

# A

## CodeWarrior IDE Interface Definition Language (IDL)

---

This appendix contains the IDL for the CodeWarrior COM API.

```
// Generated .IDL file (by the OLE/COM Object Viewer)
//
// typelib filename: IDE.EXE
// Forward declare all types defined in this typelib
interface ICodeWarriorProject;
interface IFileSpec;
interface ICodeWarriorDesignCollection;
interface ICodeWarriorDesign;
interface ICodeWarriorTargetCollection;
interface ICodeWarriorTarget;
interface ICodeWarriorSymbolContainer;
interface ICodeWarriorClassCollection;
interface ICodeWarriorClass;
interface ICodeWarriorSymbol;
interface ICodeWarriorSourceContext;
interface ICodeWarriorBaseClassCollection;
interface ICodeWarriorBaseClassInfo;
interface ICodeWarriorDataMemberCollection;
interface ICodeWarriorDataMember;
interface ICodeWarriorMethodCollection;
interface ICodeWarriorMethod;
interface ICodeWarriorProjectFileCollection;
interface ICodeWarriorProjectFile;
interface ICodeWarriorVCSState;
interface ICodeWarriorTargetFileCollection;
interface ICodeWarriorTargetFile;
interface ICodeWarriorAccessPaths;
interface ICodeWarriorAccessPathCollection;
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
interface ICodeWarriorAccessPath;
interface ICodeWarriorUserTree;
interface ICodeWarriorUserTreeCollection;
interface ICodeWarriorSubTargetCollection;
interface ICodeWarriorSubTarget;
interface IFileSpecCollection;
interface IBSTRCollection;
interface IStream;
interface ISequentialStream;
interface ICodeWarriorBuildMessages;
interface ICodeWarriorMessageCollection;
interface ICodeWarriorMessage;
interface ICodeWarriorTargetOutput;
interface ICodeWarriorApp;
interface ICodeWarriorProjectCollection;
interface ICodeWarriorCreatableItemCollection;
interface ICodeWarriorCreatableItem;
interface ICodeWarriorDocumentCollection;
interface ICodeWarriorDocument;
interface ICodeWarriorProjectDocument;
interface ICodeWarriorVersionControl;
interface ICodeWarriorTextDocument;
interface ICodeWarriorTextEngine;
interface ICodeWarriorComponent;
interface ICodeWarriorComponentPropertyCollection;
interface ICodeWarriorComponentProperty;
interface ICodeWarriorComponentEventSetCollection;
interface ICodeWarriorComponentEventSet;
interface ICodeWarriorComponentEventCollection;
interface ICodeWarriorComponentEvent;
interface ICodeWarriorSymbolCollection;
interface ICodeWarriorComponentCollection;
interface ICodeWarriorAppEvents;
interface ICodeWarriorProjectEvents;
interface ICodeWarriorDesignEvents;
interface ICodeWarriorDesignAttachment;
interface ICodeWarriorCreateProjectItem;
interface ICodeWarriorCreateFileItem;
interface ICodeWarriorCreateObjectItem;
interface ICodeWarriorVCSFileStateListener;
interface ICodeWarriorProjectAssociation;
interface ICodeWarriorErrorInfo;
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

[
 uuid(5EC306A0-283D-11D0-989C-0080C74ADF8C),
 version(1.1),
 helpstring("Metrowerks CodeWarrior IDE")
]
library CodeWarrior
{
 // TLib : OLE Automation : {00020430-0000-0000-C000-
000000000046}
 importlib("stdole2.tlb");

 [
 odl,
 uuid(110C62F0-CD3C-11D0-846D-00805F3E911D),
 dual,
 oleautomation
]
 interface ICodeWarriorProject : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000009), propget]
 HRESULT _stdcall FileSpec([out, retval] IFileSpec** pval);
 [id(0x00000067), propget]
 HRESULT _stdcall Designs([out, retval]
ICodeWarriorDesignCollection** pval);
 [id(0x00000068), propget]
 HRESULT _stdcall Targets([out, retval]
ICodeWarriorTargetCollection** pval);
 [id(0x00000005), propget]
 HRESULT _stdcall Application([out, retval]
ICodeWarriorApp** val);
 [id(0x00000070), propget]
 HRESULT _stdcall IsVisible([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000075), propget]
 HRESULT _stdcall VersionControl([out, retval]
ICodeWarriorVersionControl** VersionControl);
 [id(0x00000064)]
 HRESULT _stdcall Close();
 [id(0x00000077)]
 HRESULT _stdcall Export([in] BSTR filePath);
 }
}

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[id(0x00000078)]
HRESULT _stdcall ExportByFileSpec([in] IFileSpec*
FileSpec);
[id(0x00000065)]
HRESULT _stdcall RemoveBinaries();
[id(0x00000066)]
HRESULT _stdcall SetCurrentTarget([in] BSTR targetName);
[id(0x00000069)]
HRESULT _stdcall CreateDesign(
 [in] BSTR designName,
 [out, retval] ICodeWarriorDesign** Design);
[id(0x0000006a)]
HRESULT _stdcall CreateTarget(
 [in] BSTR targetName,
 [in] BSTR linkerName,
 [in] ICodeWarriorDesign* Design,
 [out, retval] ICodeWarriorTarget** Target);
[id(0x0000006b)]
HRESULT _stdcall RemoveTarget([in] ICodeWarriorTarget*
Target);
[id(0x0000006c)]
HRESULT _stdcall FindDesign(
 [in] BSTR Name,
 [out, retval] ICodeWarriorDesign** Design);
[id(0x0000006d)]
HRESULT _stdcall FindTarget(
 [in] BSTR Name,
 [out, retval] ICodeWarriorTarget** Target);
[id(0x0000006e)]
HRESULT _stdcall CloneTarget(
 [in] ICodeWarriorTarget* srcTarget,
 [in] ICodeWarriorProject* srcProject,
 [in] BSTR inDestTargetName,
 [in] VARIANT_BOOL fCopyFileList,
 [in] VARIANT_BOOL fCopyTargetSettings,
 [in] ICodeWarriorDesign* Design,
 [out] ICodeWarriorTarget** outTarget);
[id(0x0000006f)]
HRESULT _stdcall GetCurrentTarget([out, retval]
ICodeWarriorTarget** Target);
[id(0x00000071)]
HRESULT _stdcall Build([out, retval] long* cookie);
```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[id(0x00000072)]
HRESULT _stdcall RemoveDesignByName(
 [in] BSTR designName,
 [in] VARIANT_BOOL fDeleteContainedDesigns);
[id(0x00000073)]
HRESULT _stdcall RemoveDesign(
 [in] ICodeWarriorDesign* Design,
 [in] VARIANT_BOOL fDeleteContainedDesigns);
[id(0x00000074)]
HRESULT _stdcall ReportMessage(
 [in] EReportMsgType msgType,
 [in] BSTR message);
[id(0x00000076)]
HRESULT _stdcall SynchronizeStatus();
[id(0x00000079)]
HRESULT _stdcall BuildAndWaitToComplete([out, retval]
ICodeWarriorBuildMessages** buildMessages);
[id(0x0000007a)]
HRESULT _stdcall GetNamedPluginData(
 [in] BSTR resourceName,
 [in] EPluginDataStorageLoc storeIn,
 [out] IStream** pluginData);
[id(0x0000007b)]
HRESULT _stdcall SetNamedPluginData(
 [in] BSTR resourceName,
 [in] EPluginDataStorageLoc storeIn,
 [in] IStream* pluginData);
[id(0x0000007c)]
HRESULT _stdcall RemoveNamedPluginData(
 [in] BSTR resourceName,
 [in] EPluginDataStorageLoc storeIn);
};

[
 odl,
 uuid(229924D2-FA29-11D1-B330-0060081C5489),
 dual,
 oleautomation
]
interface IFileSpec : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
};
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[id(0x00000001), propput]
HRESULT _stdcall Name([in] BSTR pval);
[id(0x00000064), propget]
HRESULT _stdcall FullPath([out, retval] BSTR* path);
[id(0x00000064), propput]
HRESULT _stdcall FullPath([in] BSTR path);
[id(0x00000065)]
HRESULT _stdcall Copy([in] IFileSpec* inSpec);
[id(0x00000066)]
HRESULT _stdcall Clone([out] IFileSpec** pval);
};

[
 odl,
 uuid(C694D140-95C4-11D1-B31B-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorDesignCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorDesign** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorDesign* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorDesign* pval);
};

[
 odl,
 uuid(4B0EF0A0-95C5-11D1-B31B-0060081C5489),
 dual,
 oleautomation
]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
interface ICodeWarriorDesign : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000001), propput]
 HRESULT _stdcall Name([in] BSTR pval);
 [id(0x00000064), propget]
 HRESULT _stdcall DataModel([out, retval] IUnknown** pval);
 [id(0x00000065), propget]
 HRESULT _stdcall Project([out, retval]
ICodeWarriorProject** pval);
 [id(0x00000066), propget]
 HRESULT _stdcall Targets([out, retval]
ICodeWarriorTargetCollection** pval);
 [id(0x0000006b), propget]
 HRESULT _stdcall BrowserDB([out, retval]
ICodeWarriorSymbolContainer** pval);
 [id(0x00000067)]
 HRESULT _stdcall AddFile(
 [in] BSTR path,
 [in] BSTR groupPath,
 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x00000068)]
 HRESULT _stdcall AddFileByFileSpec(
 [in] IFileSpec* FileSpec,
 [in] BSTR groupPath,
 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x0000006e)]
 HRESULT _stdcall FindAndAddFile(
 [in] BSTR path,
 [in] BSTR groupPath,
 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x00000069)]
 HRESULT _stdcall ContainsTarget(ICodeWarriorTarget*
Target);
 [id(0x0000006a)]
 HRESULT _stdcall
RemoveTargetFromDesign(ICodeWarriorTarget* Target);
 [id(0x0000006f)]
 HRESULT _stdcall CompileFiles(
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [in] ICodeWarriorProjectFileCollection*
collection,
 [out, retval] long* cookie);
 [id(0x0000006c)]
 HRESULT _stdcall AddAttachment(GUID* attachmentCLSID);
 [id(0x0000006d)]
 HRESULT _stdcall RemoveAttachment(GUID* attachmentCLSID);
};

[
 odl,
 uuid(5976F990-F99B-11D1-B330-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorTargetCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorTarget** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorTarget* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorTarget* pval);
};

[
 odl,
 uuid(F094A000-F996-11D1-B330-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorTarget : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
};
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [id(0x00000001), propget]
 HRESULT _stdcall Name([in] BSTR pval);
 [id(0x00000064), propget]
 HRESULT _stdcall Project([out, retval]
ICodeWarriorProject** Project);
 [id(0x00000065), propget]
 HRESULT _stdcall Design([out, retval] ICodeWarriorDesign**
Design);
 [id(0x0000007a), propget]
 HRESULT _stdcall BrowserEnabled([out, retval]
VARIANT_BOOL* fEnabled);
 [id(0x0000007a), propget]
 HRESULT _stdcall BrowserEnabled([in] VARIANT_BOOL
fEnabled);
 [id(0x00000066), propget]
 HRESULT _stdcall BrowserDB([out, retval]
ICodeWarriorSymbolContainer** catalog);
 [id(0x00000070), propget]
 HRESULT _stdcall ProjectFileCollection([out, retval]
ICodeWarriorProjectFileCollection** ProjectFileCollection);
 [id(0x00000069), propget]
 HRESULT _stdcall TargetFileCollection([out, retval]
ICodeWarriorTargetFileCollection** TargetFileCollection);
 [id(0x0000006a), propget]
 HRESULT _stdcall AccessPaths([out, retval]
ICodeWarriorAccessPaths** pval);
 [id(0x0000006b), propget]
 HRESULT _stdcall UserTrees([out, retval]
ICodeWarriorUserTreeCollection** pval);
 [id(0x00000072), propget]
 HRESULT _stdcall SubTargets([out, retval]
ICodeWarriorSubTargetCollection** subTargetList);
 [id(0x00000067)]
 HRESULT _stdcall AddFile(
 [in] BSTR path,
 [in] BSTR groupPath,
 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x00000068)]
 HRESULT _stdcall AddFileByFileSpec(
 [in] IFileSpec* FileSpec,
 [in] BSTR groupPath,

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x0000007d)]
 HRESULT _stdcall AddFileByFileSpecCollection(
 [in] IFileSpecCollection* inCollection,
 [in] BSTR groupPath,
 [out] int* pFilesAdded);
 [id(0x0000006d)]
 HRESULT _stdcall FindAndAddFile(
 [in] BSTR path,
 [in] BSTR groupPath,
 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x0000007c)]
 HRESULT _stdcall FindAndAddFileByCollection(
 [in] IBSTRCollection* inCollection,
 [in] BSTR groupPath,
 [out] int* pFilesAdded);
 [id(0x0000006c)]
 HRESULT _stdcall AddSubTarget(
 [in] ICodeWarriorTarget* Target,
 [in] VARIANT_BOOL LinkAgainstOutput);
 [id(0x0000006e)]
 HRESULT _stdcall SetupDebugging([in] VARIANT_BOOL
inTurnOn);
 [id(0x0000006f)]
 HRESULT _stdcall CompileFiles(
 [in] ICodeWarriorProjectFileCollection*
collection,
 [out, retval] long* cookie);
 [id(0x00000071)]
 HRESULT _stdcall GetTargetFileForProjectFile(
 [in] ICodeWarriorProjectFile* projectFile,
 [out, retval] ICodeWarriorTargetFile**
targetFile);
 [id(0x00000073)]
 HRESULT _stdcall GetProjectFileFromFileSpec(
 [in] IFileSpec* FileSpec,
 [out, retval] ICodeWarriorProjectFile**
projectFile);
 [id(0x00000074)]
 HRESULT _stdcall GetNamedPluginData(
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[in] BSTR resourceName,
[in] EPluginDataStorageLoc storeIn,
[out] IStream** pluginData);
[id(0x00000075)]
HRESULT _stdcall SetNamedPluginData(
[in] BSTR resourceName,
[in] EPluginDataStorageLoc storeIn,
[in] IStream* pluginData);
[id(0x00000077)]
HRESULT _stdcall RemoveNamedPluginData(
[in] BSTR resourceName,
[in] EPluginDataStorageLoc storeIn);
[id(0x00000076)]
HRESULT _stdcall Build([out, retval] long* cookie);
[id(0x00000078)]
HRESULT _stdcall SynchronizeStatus();
[id(0x00000079)]
HRESULT _stdcall RemoveObjectCode([in] VARIANT_BOOL
deleteDataFiles);
[id(0x0000007b)]
HRESULT _stdcall BuildAndWaitToComplete([out, retval]
ICodeWarriorBuildMessages** buildMessages);
[id(0x0000007e)]
HRESULT _stdcall GetTargetOutput([out, retval]
ICodeWarriorTargetOutput** targetOutput);
[id(0x0000007f)]
HRESULT _stdcall CompileFilesAndWaitToComplete(
[in] ICodeWarriorProjectFileCollection*
collection,
[out, retval] ICodeWarriorBuildMessages**
buildMessages);
};

[
 odl,
 uuid(F385EEA1-048E-11D2-80C4-006008C3EEF1),
 dual,
 oleautomation
]
interface ICodeWarriorSymbolContainer : IDispatch {
 [id(0x00000064), propget]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall Target([out, retval] ICodeWarriorTarget**
pval);
 [id(0x00000065), propget]
 HRESULT _stdcall ClassList([out, retval]
ICodeWarriorClassCollection** pval);
 [id(0x00000066)]
 HRESULT _stdcall FindClass(
 [in] BSTR inClassName,
 [out, retval] ICodeWarriorClass**
outClass);
 [id(0x00000067)]
 HRESULT _stdcall FindClassInFile(
 [in] BSTR inClassName,
 [in] IFileSpec* inSpec,
 [out, retval] ICodeWarriorClass**
outClass);
 [id(0x00000068)]
 HRESULT _stdcall AddComponentAttachment([in] GUID*
attachmentCLSID);
 [id(0x00000069)]
 HRESULT _stdcall RemoveComponentAttachment([in] GUID*
attachmentCLSID);
 [id(0x0000006a)]
 HRESULT _stdcall ShowSymbolDeclaration(
 [in] ICodeWarriorSymbol* inSymbol,
 [in] long inForEditing,
 [in] ECodeWarriorShowSymbolLocation
inLocation);
 [id(0x0000006b)]
 HRESULT _stdcall ShowSymbolDefinition(
 [in] ICodeWarriorSymbol* inSymbol,
 [in] long inForEditing,
 [in] ECodeWarriorShowSymbolLocation
inLocation);
};

[
 odl,
 uuid(E98527B0-258E-11D2-80E6-006008C3EEF1),
 dual,
 oleautomation
]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
interface ICodeWarriorClassCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorClass** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorClass* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorClass* pval);
};

[
 odl,
 uuid(2F48B6D2-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]

interface ICodeWarriorClass : ICodeWarriorSymbol {
 [id(0x000000c8), propget]
 HRESULT _stdcall BaseClasses([out, retval]
ICodeWarriorBaseClassCollection** pval);
 [id(0x000000c9), propget]
 HRESULT _stdcall SubClasses([out, retval]
ICodeWarriorClassCollection** pval);
 [id(0x000000ca), propget]
 HRESULT _stdcall IsPublic([out, retval] long* pval);
 [id(0x000000cb), propget]
 HRESULT _stdcall IsAbstract([out, retval] long* pval);
 [id(0x000000cc), propget]
 HRESULT _stdcall IsFinal([out, retval] long* pval);
 [id(0x000000cd)]
 HRESULT _stdcall GetDataMembers(
 [in] long inIncludeInherited,
 [out, retval]
ICodeWarriorDataMemberCollection** pval);
 [id(0x000000ce)]
 HRESULT _stdcall GetDataMembersWithAccess(
 [in] long inIncludeInherited,
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [in] ECodeWarriorAccess inAccessMask,
 [out, retval]
ICodeWarriorDataMemberCollection** pval);
 [id(0x000000cf)]
 HRESULT _stdcall FindDataMemberByName(
 [in] BSTR inName,
 [out, retval] ICodeWarriorDataMember**
pval);
 [id(0x000000d0)]
 HRESULT _stdcall GetMethods(
 [in] long inIncludeInherited,
 [out, retval]
ICodeWarriorMethodCollection** pval);
 [id(0x000000d1)]
 HRESULT _stdcall GetMethodsWithAccess(
 [in] long inIncludeInherited,
 [in] ECodeWarriorAccess inAccessMask,
 [out, retval]
ICodeWarriorMethodCollection** pval);
 [id(0x000000d2)]
 HRESULT _stdcall FindMethodByName(
 [in] BSTR inName,
 [out, retval] ICodeWarriorMethod** pval);
};

[
 odl,
 uuid(2F48B6D0-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]
interface ICodeWarriorSymbol : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall Container([out, retval]
ICodeWarriorSymbolContainer** pval);
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000065), propget]
 HRESULT _stdcall SimpleName([out, retval] BSTR* pval);
 [id(0x00000066), propget]
 HRESULT _stdcall Class([out, retval] ICodeWarriorClass**
pval);
};
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [id(0x00000068), propget]
 HRESULT _stdcall DeclarationLocation([out, retval]
ICodeWarriorSourceContext** pval);
 [id(0x00000067), propget]
 HRESULT _stdcall DefinitionLocation([out, retval]
ICodeWarriorSourceContext** pval);
 };

 [
 odl,
 uuid(F385EEA0-048E-11D2-80C4-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorSourceContext : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall FileSpec([out, retval] IFileSpec** pval);
 [id(0x00000064), propput]
 HRESULT _stdcall FileSpec([in] IFileSpec* pval);
 [id(0x00000065), propget]
 HRESULT _stdcall StartOffset([out, retval] long* pval);
 [id(0x00000065), propput]
 HRESULT _stdcall StartOffset([in] long pval);
 [id(0x00000066), propget]
 HRESULT _stdcall EndOffset([out, retval] long* pval);
 [id(0x00000066), propput]
 HRESULT _stdcall EndOffset([in] long pval);
 [id(0x00000067), propget]
 HRESULT _stdcall IsDefined([out, retval] long* pval);
 };

 [
 odl,
 uuid(2F48B6D6-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorBaseClassCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 };

```

---



## Freescall Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[id(00000000)]
HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorBaseClassInfo**
pval);

[id(0x00000007)]
HRESULT _stdcall Add([in] ICodeWarriorBaseClassInfo*
pval);

[id(0x00000008)]
HRESULT _stdcall Remove([in] ICodeWarriorBaseClassInfo*
pval);
};

[
 odl,
 uuid(2F48B6D7-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]
interface ICodeWarriorBaseClassInfo : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall BaseClass([out, retval]
ICodeWarriorClass** pval);
 [id(0x00000065), propget]
 HRESULT _stdcall Access([out, retval] ECodeWarriorAccess*
pval);
 [id(0x00000066), propget]
 HRESULT _stdcall IsVirtual([out, retval] long* pval);
};

typedef [uuid(2F48B6D1-052A-11D2-80C5-006008C3EEF1)]
enum {
 kAccessNone = 0,
 kPublicAccess = 1,
 kProtectedAccess = 2,
 kPrivateAccess = 4,
 kAccessAll = 65535
} ECodeWarriorAccess;

[
 odl,
 uuid(E98527B1-258E-11D2-80E6-006008C3EEF1),
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 dual,
 oleautomation
]
 interface ICodeWarriorDataMemberCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorDataMember**
pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorDataMember* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorDataMember*
pval);
 };

 [
 odl,
 uuid(2F48B6D3-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorDataMember : ICodeWarriorSymbol {
 [id(0x000000c8), propget]
 HRESULT _stdcall Access([out, retval] ECodeWarriorAccess*
pval);
 [id(0x000000c9), propget]
 HRESULT _stdcall IsStatic([out, retval] long* pval);
 [id(0x000000ca), propget]
 HRESULT _stdcall IsFinal([out, retval] long* pval);
 [id(0x000000cb), propget]
 HRESULT _stdcall IsTransient([out, retval] long* pval);
 [id(0x000000cc), propget]
 HRESULT _stdcall IsVolatile([out, retval] long* pval);
 };

 [
 odl,

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 uuid(E98527B2-258E-11D2-80E6-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorMethodCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorMethod** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorMethod* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorMethod* pval);
 };

 [
 odl,
 uuid(2F48B6D4-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorMethod : ICodeWarriorSymbol {
 [id(0x000000c8), propget]
 HRESULT _stdcall IsConstructor([out, retval] long* pval);
 [id(0x000000c9), propget]
 HRESULT _stdcall IsDestructor([out, retval] long* pval);
 [id(0x000000ca), propget]
 HRESULT _stdcall Access([out, retval] ECodeWarriorAccess*
pval);
 [id(0x000000cb), propget]
 HRESULT _stdcall IsVirtual([out, retval] long* pval);
 [id(0x000000cc), propget]
 HRESULT _stdcall IsAbstract([out, retval] long* pval);
 [id(0x000000cd), propget]
 HRESULT _stdcall IsStatic([out, retval] long* pval);
 [id(0x000000ce), propget]
 HRESULT _stdcall IsInline([out, retval] long* pval);
 [id(0x000000cf), propget]

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 HRESULT _stdcall IsConst([out, retval] long* pval);
 [id(0x000000d0), propget]
 HRESULT _stdcall IsNative([out, retval] long* pval);
 [id(0x000000d1), propget]
 HRESULT _stdcall IsSynchronized([out, retval] long* pval);
 };

typedef [uuid(DC953130-943B-11D2-8183-006008C3EEF1)]
enum {
 kShowInEditor = 0,
 kShowInBrowser = 1,
 kUsePreferenceToShow = 2
} ECodeWarriorShowSymbolLocation;

[
 odl,
 uuid(76624FA0-7987-11D2-B361-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorProjectFileCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorProjectFile**
pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorProjectFile* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorProjectFile*
pval);
};

[
 odl,

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 uuid(59846760-7986-11D2-B361-0060081C5489),
 dual,
 oleautomation
]
 interface ICodeWarriorProjectFile : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000064), propget]
 HRESULT _stdcall Project([out, retval]
ICodeWarriorProject** pval);
 [id(0x00000009), propget]
 HRESULT _stdcall FileSpec([out, retval] IFileSpec** pval);
 [id(0x00000065), propget]
 HRESULT _stdcall Targets([out, retval]
ICodeWarriorTargetCollection** pval);
 [id(0x00000066), propget]
 HRESULT _stdcall VCSState([out, retval]
ICodeWarriorVCSState** pval);
 [id(0x00000067)]
 HRESULT _stdcall CheckOut();
 [id(0x00000068)]
 HRESULT _stdcall CheckIn();
 };

 [
 odl,
 uuid(9DD0D0B6-ABDA-11D2-9AC2-00C04F79DE48)
]
 interface ICodeWarriorVCSState : IDispatch {
 [propget]
 HRESULT _stdcall FileLockState([out, retval]
ECodeWarriorVCSFileLockState* type);
 [propget]
 HRESULT _stdcall CKIDState([out, retval]
ECodeWarriorVCSCKIDState* type);
 [propget]
 HRESULT _stdcall DBState([out, retval]
ECodeWarriorVCSDBState* type);
 };

 typedef enum {
 vcsLockNotChecked = 0,
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 vcsVolLocked = 1,
 vcsFileLocked = 2,
 vcsFileReadOnly = 3,
 vcsFileReadWrite = 4
 } ECodeWarriorVCSFileLockState;

typedef enum {
 vcsCKIDNotChecked = 0,
 vcsNoCKID = 1,
 vcsCKIDCheckedIn = 2,
 vcsCKIDCheckedOut = 3,
 vcsCKIDMRO = 4
} ECodeWarriorVCCKIDState;

typedef enum {
 vcsDBNotChecked = 0,
 vcsDBNotInWorkingDir = 1,
 vcsDBNotInDatabase = 2,
 vcsDBCheckedIn = 3,
 vcsDBCheckedOut = 4
} ECodeWarriorVCSDBState;

[
 odl,
 uuid(E14C280E-6799-11D2-9A80-00C04F79DE48),
 dual,
 oleautomation
]
interface ICodeWarriorTargetFileCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorTargetFile**
pval);
 [id(0x00000007)]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall Add([in] ICodeWarriorTargetFile* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorTargetFile*
pval);
 };

 [
 odl,
 uuid(3D452250-14EA-11D2-B33B-0060081C5489),
 dual,
 oleautomation
]
 interface ICodeWarriorTargetFile : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000064), propget]
 HRESULT _stdcall Target([out, retval] ICodeWarriorTarget**
pval);
 [id(0x00000009), propget]
 HRESULT _stdcall FileSpec([out, retval] IFileSpec** pval);
 [id(0x00000065), propget]
 HRESULT _stdcall Dependents([out, retval]
ICodeWarriorTargetFileCollection** pval);
 [id(0x00000066), propget]
 HRESULT _stdcall Dependencies([out, retval]
ICodeWarriorTargetFileCollection** pval);
 [id(0x00000067), propput]
 HRESULT _stdcall DebugInfo([in] VARIANT_BOOL value);
 [id(0x00000067), propget]
 HRESULT _stdcall DebugInfo([out, retval] VARIANT_BOOL*
value);
 [id(0x00000068), propput]
 HRESULT _stdcall InitBefore([in] VARIANT_BOOL value);
 [id(0x00000068), propget]
 HRESULT _stdcall InitBefore([out, retval] VARIANT_BOOL*
value);
 [id(0x00000069), propput]
 HRESULT _stdcall MergeLibrary([in] VARIANT_BOOL value);
 [id(0x00000069), propget]
 HRESULT _stdcall MergeLibrary([out, retval] VARIANT_BOOL*
value);
 [id(0x0000006a), propput]
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 HRESULT _stdcall WeakImport([in] VARIANT_BOOL value);
 [id(0x0000006a), propget]
 HRESULT _stdcall WeakImport([out, retval] VARIANT_BOOL*
value);
 };

 [
 odl,
 uuid(BACE41C0-6DE6-11D2-AD83-006008A5C0A5),
 dual,
 oleautomation
]
 interface ICodeWarriorAccessPaths : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall UserAccessPaths([out, retval]
ICodeWarriorAccessPathCollection** pval);
 [id(0x00000065), propget]
 HRESULT _stdcall SystemAccessPaths([out, retval]
ICodeWarriorAccessPathCollection** pval);
 [id(0x00000066), propget]
 HRESULT _stdcall AlwaysSearchUserPaths([out, retval]
VARIANT_BOOL* pval);
 [id(0x00000066), propput]
 HRESULT _stdcall AlwaysSearchUserPaths([in] VARIANT_BOOL
pval);
 [id(0x00000068)]
 HRESULT _stdcall CreateAccessPath(
 [in] BSTR path,
 [in] VARIANT_BOOL Recursion,
 [in] EAccessPathLocation inLocation,
 [in] EAccessPathType inType,
 [out, retval] ICodeWarriorAccessPath**
pval);
 [id(0x0000006a)]
 HRESULT _stdcall CreateAccessPathByFileSpec(
 [in] IFileSpec* path,
 [in] VARIANT_BOOL Recursion,
 [in] EAccessPathLocation inLocation,
 [in] EAccessPathType inType,
 [out, retval] ICodeWarriorAccessPath**
pval);
 [id(0x00000069)]

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall ApplyChanges();
 };

 [
 odl,
 uuid(916DA200-6DE6-11D2-AD83-006008A5C0A5),
 dual,
 oleautomation
]
 interface ICodeWarriorAccessPathCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorAccessPath**
pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorAccessPath* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorAccessPath*
pval);
 };

 [
 odl,
 uuid(833D6550-6DE6-11D2-AD83-006008A5C0A5),
 dual,
 oleautomation
]
 interface ICodeWarriorAccessPath : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall Recursive([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000064), propput]
 HRESULT _stdcall Recursive([in] VARIANT_BOOL pval);
 [id(0x00000065), propget]
 HRESULT _stdcall path([out, retval] IFileSpec** pval);
 [id(0x00000066), propget]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall AccessPathLocation([out, retval]
EAccessPathLocation* pval);
 [id(0x00000066), propget]
 HRESULT _stdcall AccessPathLocation([in]
EAccessPathLocation pval);
 [id(0x00000067), propget]
 HRESULT _stdcall AccessPathType([out, retval]
EAccessPathType* pval);
 [id(0x00000068), propget]
 HRESULT _stdcall UserTree([out, retval]
ICodeWarriorUserTree** pval);
 [id(0x00000069), propget]
 HRESULT _stdcall SubDirectories([out, retval]
ICodeWarriorAccessPathCollection** pval);
 };

 typedef enum {
 kAbsolute = 0,
 kProjectRelative = 1,
 kCompilerRelative = 2,
 kSystemRelative = 3,
 kUserDefined = 4
 } EAccessPathLocation;

 typedef enum {
 kUserPath = 0,
 kSystemPath = 1
 } EAccessPathType;

 [
 odl,
 uuid(50993290-6DE6-11D2-AD83-006008A5C0A5),
 dual,
 oleautomation
]
 interface ICodeWarriorUserTree : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000001), propget]
 HRESULT _stdcall Name([in] BSTR pval);
 [id(0x00000064), propget]
 HRESULT _stdcall value([out, retval] BSTR* pval);
 }
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [id(0x00000064), propget]
 HRESULT _stdcall value([in] BSTR pval);
 [id(0x00000065), propget]
 HRESULT _stdcall type([out, retval] EUserDefinedTree*
val);
 [id(0x00000065), propget]
 HRESULT _stdcall type([in] EUserDefinedTree val);
 };

 typedef enum {
 kAbsolutePath = 0,
 kEnvironment = 1,
 kRegistry = 2
 } EUserDefinedTree;

 [
 odl,
 uuid(A7B77820-6DE6-11D2-AD83-006008A5C0A5),
 dual,
 oleautomation
]
 interface ICodeWarriorUserTreeCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorUserTree** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorUserTree* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorUserTree* pval);
 };

 [
 odl,
 uuid(8463C22C-7E72-11D2-9A8E-00C04F79DE48),

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 dual,
 oleautomation
]
 interface ICodeWarriorSubTargetCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorSubTarget**
pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorSubTarget* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorSubTarget* pval);
 };

 [
 odl,
 uuid(CF40CE56-95FC-11D2-9A8D-00C04F79DE48),
 dual,
 oleautomation
]
 interface ICodeWarriorSubTarget : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall Target([out, retval] ICodeWarriorTarget**
pval);
 [id(0x00000065), propget]
 HRESULT _stdcall LinkAgainstOutput([out, retval]
VARIANT_BOOL* pval);
 };

 [
 odl,
 uuid(F49299BE-C072-11D2-9ADC-00C04F79DE48),
 dual,
 oleautomation
]

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

]
 interface IFileSpecCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] IFileSpec** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] IFileSpec* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] IFileSpec* pval);
 };

 [
 odl,
 uuid(E1179B70-EB6F-11D2-ADDA-00C04F804195),
 dual,
 oleautomation
]
 interface IBSTRCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] BSTR* pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] BSTR val);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] BSTR val);
 };

```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
typedef enum {
 kStoreInProjectFile = 0,
 kStoreInTargetDataFile = 1,
 kStoreInProjectSettingsFile = 2
} EPluginDataStorageLoc;

[
 odl,
 uuid(0000000C-0000-0000-C000-000000000046)
]
interface IStream : ISequentialStream {
 HRESULT _stdcall RemoteSeek(
 [in] _LARGE_INTEGER dlibMove,
 [in] unsigned long dwOrigin,
 [out] _ULARGE_INTEGER* plibNewPosition);
 HRESULT _stdcall SetSize([in] _ULARGE_INTEGER libNewSize);
 HRESULT _stdcall RemoteCopyTo(
 [in] IStream* pstm,
 [in] _ULARGE_INTEGER cb,
 [out] _ULARGE_INTEGER* pcbRead,
 [out] _ULARGE_INTEGER* pcbWritten);
 HRESULT _stdcall Commit([in] unsigned long
grfCommitFlags);
 HRESULT _stdcall Revert();
 HRESULT _stdcall LockRegion(
 [in] _ULARGE_INTEGER libOffset,
 [in] _ULARGE_INTEGER cb,
 [in] unsigned long dwLockType);
 HRESULT _stdcall UnlockRegion(
 [in] _ULARGE_INTEGER libOffset,
 [in] _ULARGE_INTEGER cb,
 [in] unsigned long dwLockType);
 HRESULT _stdcall Stat(
 [out] tagSTATSTG* pstatstg,
 [in] unsigned long grfStatFlag);
 HRESULT _stdcall Clone([out] IStream** ppstm);
};

[
 odl,
 uuid(0C733A30-2A1C-11CE-ADE5-00AA0044773D)
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
]
interface ISequentialStream : IUnknown {
 HRESULT _stdcall RemoteRead(
 [out] char* pv,
 [in] unsigned long cb,
 [out] unsigned long* pcbRead);
 HRESULT _stdcall RemoteWrite(
 [in] char* pv,
 [in] unsigned long cb,
 [out] unsigned long* pcbWritten);
};

typedef struct tag_LARGE_INTEGER {
int64 QuadPart;
} _LARGE_INTEGER;

typedef struct tag_ULARGE_INTEGER {
uint64 QuadPart;
} _ULARGE_INTEGER;

typedef struct tagtagSTATSTG {
LPWSTR pwcsName;
unsigned long type;
_ULARGE_INTEGER cbSize;
_FILETIME mtime;
_FILETIME ctime;
_FILETIME atime;
unsigned long grfMode;
unsigned long grfLocksSupported;
GUID clsid;
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

unsigned long grfStateBits;

unsigned long reserved;
 } tagSTATSTG;

 typedef struct tag_FILETIME {

unsigned long dwLowDateTime;

unsigned long dwHighDateTime;
 } _FILETIME;

[
 odl,
 uuid(6980FC87-A00A-11D2-9AB2-00C04F79DE48)
]
interface ICodeWarriorBuildMessages : IDispatch {
 [propget]
 HRESULT _stdcall Errors([out]
ICodeWarriorMessageCollection** Errors);
 [propget]
 HRESULT _stdcall Warnings([out]
ICodeWarriorMessageCollection** Warnings);
 [propget]
 HRESULT _stdcall Informations([out]
ICodeWarriorMessageCollection** info);
 [propget]
 HRESULT _stdcall Definitions([out]
ICodeWarriorMessageCollection** Errors);
 [propget]
 HRESULT _stdcall ErrorCount([out] long* Count);
 [propget]
 HRESULT _stdcall WarningCount([out] long* Count);
 [propget]
 HRESULT _stdcall InformationCount([out] long* Count);
 [propget]
 HRESULT _stdcall DefinitionCount([out] long* Count);
};

[
 odl,
 uuid(A341D251-A00B-11D2-9AB2-00C04F79DE48),

```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 dual,
 oleautomation
]
interface ICodeWarriorMessageCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorMessage** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorMessage* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorMessage* pval);
};

[
 odl,
 uuid(DF1E763E-96A6-11D2-9AA9-00C04F79DE48)
]
interface ICodeWarriorMessage : IDispatch {
 [propget]
 HRESULT _stdcall type([out] EMsgType* type);
 [propget]
 HRESULT _stdcall FileSpec([out] IFileSpec** FileSpec);
 [propget]
 HRESULT _stdcall projectFile([out]
ICodeWarriorProjectFile** projectFile);
 [propget]
 HRESULT _stdcall MessageText([out] BSTR* message);
 [propget]
 HRESULT _stdcall ErrorNumber([out] long* ErrorNumber);
 [propget]
 HRESULT _stdcall Target([out] ICodeWarriorTarget**
Target);
 [propget]
 HRESULT _stdcall SourceOffset([out] long* offset);

```

---



## Freescall Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[propget]
HRESULT _stdcall SourceLength([out] long* length);
[propget]
HRESULT _stdcall SourceLineNumber([out] long* lineNumber);
[propget]
HRESULT _stdcall TokenOffset([out] long* TokenOffset);
[propget]
HRESULT _stdcall TokenLength([out] long* TokenLength);
[propget]
HRESULT _stdcall MessageLineCount([out] long* lineCount);
[propget]
HRESULT _stdcall MessageLength([out] long* MessageLength);
};

typedef enum {
 typeNotDefined = 0,
 typeInformation = 1,
 typeWarning = 2,
 typeError = 3,
 typeDefinition = 4
} EMsgType;

[
 odl,
 uuid(6971AB76-EC83-11D2-9B0C-00C04F79DE48),
 dual,
 oleautomation
]
interface ICodeWarriorTargetOutput : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall OutputKind([out, retval]
ECodeWarriorTargetOutputKind* kind);
 [id(0x00000009), propget]
 HRESULT _stdcall FileSpec([out, retval] IFileSpec** pval);
};

typedef enum {
 kCWOutputNone = 0,
 kCWOutputFile = 1,
 kCWOutputDirectory = 2
} ECodeWarriorTargetOutputKind;
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [
 odl,
 uuid(5EC306A1-283D-11D0-989C-0080C74ADF8C),
 dual,
 oleautomation
]
 interface ICodeWarriorApp : IDispatch {
 [id(0x00000066), propget]
 HRESULT _stdcall Application([out, retval] IDispatch**
pval);
 [id(0x00000067), propget]
 HRESULT _stdcall FullName([out, retval] BSTR* pval);
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000004), propget]
 HRESULT _stdcall Visible([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000004), propput]
 HRESULT _stdcall Visible([in] VARIANT_BOOL pval);
 [id(0x00000069), propget]
 HRESULT _stdcall Projects([out, retval]
ICodeWarriorProjectCollection** pval);
 [id(0x0000006c), propget]
 HRESULT _stdcall CreatableItems([out, retval]
ICodeWarriorCreatableItemCollection** pval);
 [id(0x00000077), propget]
 HRESULT _stdcall Documents([out, retval]
ICodeWarriorDocumentCollection** pval);
 [id(0x00000078), propget]
 HRESULT _stdcall ActiveDocument([out, retval]
ICodeWarriorDocument** pval);
 [id(0x00000083), propget]
 HRESULT _stdcall DefaultProjectDocument([out, retval]
ICodeWarriorProjectDocument** pval);
 [id(0x0000007b), propget]
 HRESULT _stdcall DefaultProject([out, retval]
ICodeWarriorProject** Project);
 [id(0x0000007c), propput]
 HRESULT _stdcall AllowUserInteraction([in] VARIANT_BOOL
rhs);
 [id(0x0000007d), propget]

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 HRESULT _stdcall VersionControl([out, retval]
ICodeWarriorVersionControl** vcs);
 [id(0x0000006d)]
 HRESULT _stdcall CreateProject(
 [in] BSTR filePath,
 [in] BSTR linkerName,
 [in] BSTR designName,
 [in] BSTR targetName,
 [in] VARIANT_BOOL fMakeVisible,
 [out, retval] ICodeWarriorProject** pval);
 [id(0x0000006e)]
 HRESULT _stdcall CreateProjectByFileSpec(
 [in] IFileSpec* projectFileSpec,
 [in] BSTR linkerName,
 [in] BSTR designName,
 [in] BSTR targetName,
 [in] IFileSpec* stationeryFileSpec,
 [in] VARIANT_BOOL fMakeVisible,
 [out, retval] ICodeWarriorProject** pval);
 [id(0x00000080)]
 HRESULT _stdcall ImportProject(
 [in] BSTR textFilePath,
 [in] BSTR projectFilePath,
 [in] VARIANT_BOOL fMakeVisible,
 [out, retval] ICodeWarriorProject** pval);
 [id(0x00000081)]
 HRESULT _stdcall ImportProjectByFileSpec(
 [in] IFileSpec* textFileSpec,
 [in] IFileSpec* projectFileSpec,
 [in] VARIANT_BOOL fMakeVisible,
 [out, retval] ICodeWarriorProject** pval);
 [id(0x00000065)]
 HRESULT _stdcall OpenProject(
 [in] BSTR filePath,
 [in] VARIANT_BOOL fMakeVisible,
 [in] ECodeWarriorConvertOption
convertOption,
 [in] ECodeWarriorRevertPanelOption
revertOption,
 [out, retval] ICodeWarriorProject** pval);
 [id(0x0000006f)]
 HRESULT _stdcall OpenProjectByFileSpec(

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

[in] IFileSpec* FileSpec,
[in] VARIANT_BOOL fMakeVisible,
[in] ECodeWarriorConvertOption
convertOption,
[in] ECodeWarriorRevertPanelOption
revertOption,
[out, retval] ICodeWarriorProject** pval);
[id(0x0000006a)]
HRESULT _stdcall AddCreatableItem([in] IUnknown* Item);
[id(0x0000006b)]
HRESULT _stdcall RemoveCreatableItem([in] IUnknown* Item);
[id(0x00000070)]
HRESULT _stdcall FindLogicalFolder(
[in] BSTR folderName,
[out, retval] IFileSpec** folder);
[id(0x00000071)]
HRESULT _stdcall FindDesignForDataModel(
[in] IUnknown* DataModel,
[out, retval] ICodeWarriorDesign**
Project);
[id(0x00000072)]
HRESULT _stdcall GetNamedPluginData(
[in] BSTR resourceName,
[out] IStream** pluginData);
[id(0x00000073)]
HRESULT _stdcall SetNamedPluginData(
[in] BSTR resourceName,
[in] IStream* pluginData);
[id(0x00000074)]
HRESULT _stdcall RemoveNamedPluginData([in] BSTR
resourceName);
[id(0x00000075)]
HRESULT _stdcall GetSetting(
[in] BSTR settingsName,
[out, retval] VARIANT* pval);
[id(0x00000076)]
HRESULT _stdcall SetSetting(
[in] BSTR settingsName,
[in] VARIANT pval);
[id(0x00000079)]
HRESULT _stdcall OpenTextDocumentByFileSpec(
[in] IFileSpec* FileSpec,

```

---

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [in] VARIANT_BOOL create,
 [out, retval] ICodeWarriorTextDocument**
document);
 [id(0x0000007a)]
 HRESULT _stdcall OpenTextDocument(
 [in] BSTR inPath,
 [in] VARIANT_BOOL create,
 [out, retval] ICodeWarriorTextDocument**
document);
 [id(0x00000084)]
 HRESULT _stdcall OpenUntitledTextDocument([out, retval]
ICodeWarriorTextDocument** document);
 [id(0x0000007e)]
 HRESULT _stdcall AttemptModify(
 [in] IFileSpec* FileSpec,
 [in] ECodeWarriorVCSInteractionOption
uiParameter,
 [in] ICodeWarriorProject* Project);
 [id(0x0000007f)]
 HRESULT _stdcall DoCommand([in] long commandID);
 [id(0x00000082)]
 HRESULT _stdcall QueueDeferredAction(IUnknown* action);
 [id(0x00000085)]
 HRESULT _stdcall IsBuildInProgress([out, retval]
VARIANT_BOOL* pval);
};

[
 odl,
 uuid(1A657F50-95B5-11D1-B31B-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorProjectCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(00000000)]

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorProject** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorProject* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorProject* pval);
};

[
 odl,
 uuid(6161C790-FB3B-11D1-B331-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorCreatableItemCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorCreatableItem**
pval);
};

[
 odl,
 uuid(145691E0-FA29-11D1-B330-0060081C5489)
]
interface ICodeWarriorCreatableItem : IUnknown {
 HRESULT _stdcall GetDisplayName([out] BSTR* __MIDL_0017);
 HRESULT _stdcall GetIcon(
 IUnknown* iconList,
 int* index);
 HRESULT _stdcall GetCategory([out] BSTR* __MIDL_0018);
 HRESULT _stdcall InvokesWizard();
};

[
 odl,
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 uuid(BA875690-B46A-11D2-ADB6-00C04F804195),
 dual,
 oleautomation
]
 interface ICodeWarriorDocumentCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorDocument** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorDocument* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorDocument* pval);
 };

 [
 odl,
 uuid(08A1D280-B468-11D2-ADB6-00C04F804195),
 dual,
 oleautomation
]
 interface ICodeWarriorDocument : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000009), propget]
 HRESULT _stdcall FileSpec([out, retval] IFileSpec** pval);
 [id(0x00000064), propget]
 HRESULT _stdcall ActiveDocument([out, retval]
VARIANT_BOOL* pval);
 [id(0x00000006), propget]
 HRESULT _stdcall ReadOnly([out, retval] VARIANT_BOOL*
pval);
 [id(0x00000065), propget]
 HRESULT _stdcall Dirty([out, retval] VARIANT_BOOL* pval);
 [id(0x00000004), propget]

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
HRESULT _stdcall Visible([out, retval] VARIANT_BOOL*
pval);
[id(0x00000004), propput]
HRESULT _stdcall Visible([in] VARIANT_BOOL pval);
[id(0x00000069), propget]
HRESULT _stdcall XPos([out, retval] int* pval);
[id(0x00000069), propput]
HRESULT _stdcall XPos([in] int pval);
[id(0x0000006a), propget]
HRESULT _stdcall YPos([out, retval] int* pval);
[id(0x0000006a), propput]
HRESULT _stdcall YPos([in] int pval);
[id(0x0000006b), propget]
HRESULT _stdcall Width([out, retval] int* pval);
[id(0x0000006b), propput]
HRESULT _stdcall Width([in] int pval);
[id(0x0000006c), propget]
HRESULT _stdcall Height([out, retval] int* pval);
[id(0x0000006c), propput]
HRESULT _stdcall Height([in] int pval);
[id(0x00000066)]
HRESULT _stdcall Save();
[id(0x00000067)]
HRESULT _stdcall Close([in] VARIANT_BOOL bSaveChanges);
[id(0x00000068)]
HRESULT _stdcall Activate();
};

[
 odl,
 uuid(41A67F00-B584-11D2-ADB6-00C04F804195),
 dual,
 oleautomation
]
interface ICodeWarriorProjectDocument : ICodeWarriorDocument
{
 [id(0x000000c8), propget]
 HRESULT _stdcall Project([out, retval]
ICodeWarriorProject** pval);
[id(0x000000c9)]
 HRESULT _stdcall SelectFiles(
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [in] ICodeWarriorProjectFileCollection*
projectFiles,
 [in] VARIANT_BOOL select);
 [id(0x000000ca)]
 HRESULT _stdcall SelectedFiles([out, retval]
ICodeWarriorProjectFileCollection** projectFiles);
 [id(0x000000cb)]
 HRESULT _stdcall ExpandGroup([in] BSTR groupName);
 [id(0x000000cc)]
 HRESULT _stdcall CollapseGroup([in] BSTR groupName);
};

[
 odl,
 uuid(5C5A784E-C070-11D2-9ADC-00C04F79DE48)
]
interface ICodeWarriorVersionControl : IDispatch {
 [propget]
 HRESULT _stdcall Name([out, retval] BSTR* vcsName);
 HRESULT _stdcall GetVCSSState(
 [in] IFileSpec* FileSpec,
 [out, retval] ICodeWarriorVCSSState**
VCSSState);
 HRESULT _stdcall CheckIn([in] IFileSpecCollection*
fileSpecCollection);
 HRESULT _stdcall CheckOut([in] IFileSpecCollection*
fileSpecCollection);
 HRESULT _stdcall UnLock([in] IFileSpecCollection*
fileSpecCollection);
 HRESULT _stdcall Get([in] IFileSpecCollection*
fileSpecCollection);
 HRESULT _stdcall UndoCheckOut([in] IFileSpecCollection*
fileSpecCollection);
 HRESULT _stdcall Connect();
 HRESULT _stdcall Disconnect();
 HRESULT _stdcall IsConnected([out, retval] VARIANT_BOOL*
pval);
};

typedef enum {
 kCWConvertYes = 0,
 kCWConvertNo = 1,

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 kCWConvertAsk = 2
 } ECodeWarriorConvertOption;

typedef enum {
 kCWDonotRevertPanel = 0,
 kCWAllowPanelRevert = 1
} ECodeWarriorRevertPanelOption;

[
 odl,
 uuid(1AD264D0-B46C-11D2-ADB6-00C04F804195),
 dual,
 oleautomation
]
interface ICodeWarriorTextDocument : ICodeWarriorDocument {
 [id(0x000000c8), propget]
 HRESULT _stdcall TextEngine([out, retval]
ICodeWarriorTextEngine** pval);
 [id(0x000000cb)]
 HRESULT _stdcall SaveAsByFileSpec([in] IFileSpec*
FileSpec);
 [id(0x000000c9)]
 HRESULT _stdcall SaveAs([in] BSTR val);
 [id(0x000000cc)]
 HRESULT _stdcall SaveACopyAsByFileSpec([in] IFileSpec*
FileSpec);
 [id(0x000000ca)]
 HRESULT _stdcall SaveACopyAs([in] BSTR val);
};

[
 odl,
 uuid(B31823F0-B470-11D2-ADB6-00C04F804195),
 dual,
 oleautomation
]
interface ICodeWarriorTextEngine : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall SelectionStart([out, retval] int* pval);
 [id(0x00000064), propput]
 HRESULT _stdcall SelectionStart([in] int pval);
 [id(0x00000065), propget]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
HRESULT _stdcall SelectionEnd([out, retval] int* pval);
[id(0x00000065), propput]
HRESULT _stdcall SelectionEnd([in] int pval);
[id(0x00000066), propget]
HRESULT _stdcall SelectionLineStart([out, retval] int*
pval);
[id(0x00000066), propput]
HRESULT _stdcall SelectionLineStart([in] int pval);
[id(0x00000067), propget]
HRESULT _stdcall SelectionLineEnd([out, retval] int*
pval);
[id(0x00000067), propput]
HRESULT _stdcall SelectionLineEnd([in] int pval);
[id(0x00000068), propget]
HRESULT _stdcall HasSelection([out, retval] VARIANT_BOOL*
pval);
[id(0x00000069), propget]
HRESULT _stdcall SelectionText([out, retval] BSTR* pval);
[id(0x00000069), propput]
HRESULT _stdcall SelectionText([in] BSTR pval);
[id(0x0000006a), propget]
HRESULT _stdcall lineCount([out, retval] int* pval);
[id(0x0000006b), propget]
HRESULT _stdcall TextLength([out, retval] int* pval);
[id(0x0000006c)]
HRESULT _stdcall GetTextForOffsetRange(
[in] int selStart,
[in] int selEnd,
[out, retval] BSTR* pval);
[id(0x0000006d)]
HRESULT _stdcall GetTextForLineRange(
[in] int lineStart,
[in] int lineEnd,
[out, retval] BSTR* pval);
[id(0x0000006e)]
HRESULT _stdcall GetLineForOffset(
[in] int offset,
[out, retval] int* line);
[id(0x0000006f)]
HRESULT _stdcall GetOffsetForLine(
[in] int line,
[out, retval] int* offset);
```

---



## Freescall Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[id(0x00000070)]
HRESULT _stdcall InsertText([in] BSTR val);
};

typedef enum {
 kCWAsk = 0,
 kCWDoNothing = 1,
 kCWUseDefault = 2
} ECodeWarriorVCSInteractionOption;

typedef enum {
 kReportMsgAlert = 0,
 kReportMsgInformation = 1,
 kReportMsgWarning = 2
} EReportMsgType;

[
 odl,
 uuid(AE200FB0-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
interface ICodeWarriorComponent : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall Class([out, retval] ICodeWarriorClass**
pval);
 [id(0x00000065), propget]
 HRESULT _stdcall Properties([out, retval]
ICodeWarriorComponentPropertyCollection** pval);
 [id(0x00000066), propget]
 HRESULT _stdcall Methods([out, retval]
ICodeWarriorMethodCollection** pval);
 [id(0x00000067), propget]
 HRESULT _stdcall EventSets([out, retval]
ICodeWarriorComponentEventSetCollection** pval);
 [id(0x00000068), propget]
 HRESULT _stdcall CanHaveMultipleEventSets([out, retval]
long* pval);
 [id(0x00000069), propget]
 HRESULT _stdcall DefaultEvent([out, retval]
ICodeWarriorComponentEvent** pval);
 [id(0x0000006a), propget]
```



## Freescale Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall EventConnectionsEnabled([out, retval]
long* pval);
 };

 [
 odl,
 uuid(AE200FB5-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorComponentPropertyCollection : IDispatch
 {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval]
 ICodeWarriorComponentProperty** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorComponentProperty*
pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in]
 ICodeWarriorComponentProperty* pval);
 };

 [
 odl,
 uuid(AE200FB1-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorComponentProperty : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000064), propget]
 HRESULT _stdcall type([out, retval] BSTR* pval);
 [id(0x00000065), propget]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall Getter([out, retval] ICodeWarriorMethod**
pval);
 [id(0x00000066), propget]
 HRESULT _stdcall Setter([out, retval] ICodeWarriorMethod**
pval);
 };

 [
 odl,
 uuid(AE200FB6-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorComponentEventSetCollection : IDispatch
 {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval]
 ICodeWarriorComponentEventSet** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorComponentEventSet*
pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in]
 ICodeWarriorComponentEventSet* pval);
 };

 [
 odl,
 uuid(AE200FB2-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorComponentEventSet : IDispatch {
 [id(0x00000064), propget]
 HRESULT _stdcall Class([out, retval] ICodeWarriorClass**
pval);
```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [id(0x00000065), propget]
 HRESULT _stdcall EventSetName([out, retval] BSTR* pval);
 [id(0x00000066), propget]
 HRESULT _stdcall Events([out, retval]
ICodeWarriorComponentEventCollection** pval);
 };

 [
 odl,
 uuid(AE200FB7-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorComponentEventCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorComponentEvent**
pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorComponentEvent*
pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorComponentEvent*
pval);
 };

 [
 odl,
 uuid(AE200FB3-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
 interface ICodeWarriorComponentEvent : IDispatch {
 [id(0x00000001), propget]
 HRESULT _stdcall Name([out, retval] BSTR* pval);
 [id(0x00000064), propget]

```

---



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall Method([out, retval] ICodeWarriorMethod**
pval);
 [id(0x00000065), propget]
 HRESULT _stdcall EventSet([out, retval]
ICodeWarriorComponentEventSet** pval);
 [id(0x00000066)]
 HRESULT _stdcall GetDefaultMethodName(
 [in] IUnknown* modelobject,
 [out] BSTR* pdefname);
};

[
 odl,
 uuid(2F48B6D5-052A-11D2-80C5-006008C3EEF1),
 dual,
 oleautomation
]
interface ICodeWarriorSymbolCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
 HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
 [id(00000000)]
 HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorSymbol** pval);
 [id(0x00000007)]
 HRESULT _stdcall Add([in] ICodeWarriorSymbol* pval);
 [id(0x00000008)]
 HRESULT _stdcall Remove([in] ICodeWarriorSymbol* pval);
};

[
 odl,
 uuid(AE200FB4-5C69-11D2-8120-006008C3EEF1),
 dual,
 oleautomation
]
interface ICodeWarriorComponentCollection : IDispatch {
 [id(0x00000002), propget]
 HRESULT _stdcall Count([out, retval] long* pval);
 [id(0x00000003), propget]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
HRESULT _stdcall _NewEnum([out, retval] IDispatch** pval);
[id(00000000)]
HRESULT _stdcall Item(
 [in] long index,
 [out, retval] ICodeWarriorComponent**
pval);
[id(0x00000007)]
HRESULT _stdcall Add([in] ICodeWarriorComponent* pval);
[id(0x00000008)]
HRESULT _stdcall Remove([in] ICodeWarriorComponent* pval);
};

typedef enum {
 kMaximumTargetNameLength = 31
} ECodeWarriorProjectConstants;

[
 uuid(D6D02BB0-ACCC-11D2-ADB3-00C04F804195),
 appobject
]
coclass CodeWarriorApp {
 [default] interface ICodeWarriorApp;
 [default, source] interface ICodeWarriorAppEvents;
};

[
 odl,
 uuid(5EC306A3-283D-11D0-989C-0080C74ADF8C)
]
interface ICodeWarriorAppEvents : IUnknown {
 HRESULT _stdcall Startup();
 HRESULT _stdcall QueryQuit();
 HRESULT _stdcall Quit();
 HRESULT _stdcall ProjectOpened(
 [in] ICodeWarriorProject* Project,
 VARIANT_BOOL fVisible);
 HRESULT _stdcall ProjectVisible([in] ICodeWarriorProject*
Project);
 HRESULT _stdcall DataModelCreated(
 [in] IUnknown* DataModel,
 VARIANT_BOOL fFromStorage);
};
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall DataModelLoaded([in] IUnknown*
DataModel);
 };

 typedef enum {
 kCWChoiceCheckSyntax = 0,
 kCWChoicePreprocess = 1,
 kCWChoicePrecompile = 2,
 kCWChoiceCompile = 3,
 kCWChoiceDisassemble = 4
 } ECodeWarriorCompileChoice;

 [
 uuid(7153E430-AE65-11D2-ADB4-00C04F804195)
]
 coclass CWAutomationProject {
 [default] interface ICodeWarriorProject;
 [default, source] interface ICodeWarriorProjectEvents;
 };

 [
 odl,
 uuid(3D3B7F80-9694-11D1-B31B-0060081C5489)
]
 interface ICodeWarriorProjectEvents : IUnknown {
 HRESULT _stdcall QueryUIClose([in] ICodeWarriorProject*
Project);
 HRESULT _stdcall VisibleChanged(
 [in] ICodeWarriorProject* Project,
 [in] VARIANT_BOOL fVisible);
 HRESULT _stdcall ProjectClosing([in] ICodeWarriorProject*
Project);
 HRESULT _stdcall DesignCreated([in] ICodeWarriorDesign*
Design);
 HRESULT _stdcall QueryDeleteDesign([in]
ICodeWarriorDesign* Design);
 HRESULT _stdcall DeletingDesign([in] ICodeWarriorDesign*
Design);
 HRESULT _stdcall BuildStarted(
 [in] ECodeWarriorCompileChoice choice,
 [in] long buildID,
```



## Freescall Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
targetList);
 HRESULT _stdcall BuildEnded(
 [in] ICodeWarriorTargetCollection*
 [in] ECodeWarriorCompileChoice choice,
 [in] long buildID,
 [in] VARIANT_BOOL fBuildSucceeded,
 [out] ICodeWarriorBuildMessages*
buildMessages);
 HRESULT _stdcall QueryAboutToBuild(
 [in] ECodeWarriorCompileChoice choice,
 [in] long buildID,
 [in] ICodeWarriorTargetCollection*
targetList);
 HRESULT _stdcall RevertCompleted();
};

[
 uuid(92794C60-AE65-11D2-ADB4-00C04F804195)
]
coclass CWAutomationTarget {
 [default] interface ICodeWarriorTarget;
};

[
 uuid(A4C352E0-AE65-11D2-ADB4-00C04F804195)
]
coclass CWAutomationDesign {
 [default] interface ICodeWarriorDesign;
 [default, source] interface ICodeWarriorDesignEvents;
};

[
 odl,
 uuid(51FB0BD0-E515-11D1-B32A-0060081C5489)
]
interface ICodeWarriorDesignEvents : IUnknown {
 HRESULT _stdcall TargetAdded([in] ICodeWarriorTarget*
Target);
 HRESULT _stdcall RemovingTarget([in] ICodeWarriorTarget*
Target);
};
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
[
 odl,
 uuid(8967DC00-57CD-11D2-B358-0060081C5489)
]
interface ICodeWarriorDesignAttachment : IUnknown {
 HRESULT _stdcall DesignInitialized(ICodeWarriorDesign*
__MIDL_0014);
 HRESULT _stdcall DesignClosing(ICodeWarriorDesign*
__MIDL_0015);
 HRESULT _stdcall RemovingAttachment(ICodeWarriorDesign*
__MIDL_0016);
};

[
 uuid(BB058510-AE65-11D2-ADB4-00C04F804195)
]
coclass CWAutomationAccessPath {
 [default] interface ICodeWarriorAccessPath;
};

[
 uuid(D69AC280-AE65-11D2-ADB4-00C04F804195)
]
coclass CWAutomationAccessPaths {
 [default] interface ICodeWarriorAccessPaths;
};

[
 uuid(B4B07CB0-B467-11D2-ADB6-00C04F804195)
]
coclass CWAutomationDocument {
 [default] interface ICodeWarriorDocument;
};

[
 uuid(0ADC5170-B46C-11D2-ADB6-00C04F804195)
]
coclass CWAutomationTextDocument {
 [default] interface ICodeWarriorTextDocument;
};

[
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 uuid(52247090-B583-11D2-ADB6-00C04F804195)
]
coclass CWAutomationProjectDocument {
 [default] interface ICodeWarriorProjectDocument;
};

[
 uuid(8F920920-B46C-11D2-ADB6-00C04F804195)
]
coclass CWAutomationTextEngine {
 [default] interface ICodeWarriorTextEngine;
};

typedef enum {
 newIconProject = -1,
 newIconTextFile = -2,
 newIconCatalog = -3
} __MIDL___MIDL_itf_CodeWarriorComIntf_0114_0001;

typedef [public]
__MIDL___MIDL_itf_CodeWarriorComIntf_0115_0001
ECreateProjectType;

typedef enum {
 createsProjectOnly = 0,
 createsDesign = 1,
 createsTargets = 2
} __MIDL___MIDL_itf_CodeWarriorComIntf_0115_0001;

[
 odl,
 uuid(229924D0-FA29-11D1-B330-0060081C5489),
 dual,
 oleautomation
]
interface ICodeWarriorCreateProjectItem :
ICodeWarriorCreatableItem {
 [id(0x60020000)]
 HRESULT _stdcall GetCreatedProjectType([out, retval]
ECreateProjectType* pval);
 [id(0x60020001)]
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```
 HRESULT _stdcall RequiresFileExtension([out, retval]
VARIANT_BOOL* pval);
 [id(0x60020002)]
 HRESULT _stdcall CreateNewProject([in] IFileSpec*
newFileSpec);
 [id(0x60020003)]
 HRESULT _stdcall CreateInExistingProject(
 [in] BSTR newItemName,
 [in] ICodeWarriorProject* Project);
};

[
 odl,
 uuid(229924D1-FA29-11D1-B330-0060081C5489)
]
interface ICodeWarriorCreateFileItem :
ICodeWarriorCreatableItem {
 HRESULT _stdcall CanCreateUntitledFile();
 HRESULT _stdcall CanAddFileToProject();
 HRESULT _stdcall CreateUntitledFile();
 HRESULT _stdcall CreateAndAddFile(
 [in] IFileSpec* newFileSpec,
 [in] ICodeWarriorProject* Project,
 [in] ICodeWarriorTargetCollection* Targets,
 [out] VARIANT_BOOL* fFilesAdded);
};

[
 odl,
 uuid(229924D3-FA29-11D1-B330-0060081C5489)
]
interface ICodeWarriorCreateObjectItem :
ICodeWarriorCreatableItem {
 HRESULT _stdcall AreObjectsCreatedInDesign();
 HRESULT _stdcall CreateObjectInDesign(
 [in] BSTR newItemName,
 [in] ICodeWarriorProject* Project,
 [in] ICodeWarriorDesign* Design);
 HRESULT _stdcall CreateObjectInTargets(
 [in] BSTR newItemName,
 [in] ICodeWarriorProject* Project,
```



## Freescal Semiconductor, Inc.

### CodeWarrior IDE Interface Definition Language (IDL)

---

```

 [in] ICodeWarriorTargetCollection*
Targets);
 HRESULT _stdcall NeedsObjectName();
};

[
 uuid(B980537C-C37E-11D2-9ADF-00C04F79DE48),
 appobject
]
coclass CWCodeWarriorVCS {
 [default] interface ICodeWarriorVersionControl;
 [default, source] interface
ICodeWarriorVCSFileStateListener;
};

[
 odl,
 uuid(B980537A-C37E-11D2-9ADF-00C04F79DE48)
]
interface ICodeWarriorVCSFileStateListener : IUnknown {
 HRESULT _stdcall StateChanged(
 [in] IFileSpec* FileSpec,
 [in] ICodeWarriorVCSState* VCSState);
};

[
 odl,
 uuid(BE65AD59-C4BC-11D2-8065-006008C3EEB0),
 dual,
 oleautomation
]
interface ICodeWarriorProjectAssociation : IUnknown {
 [id(0x00000064), propget]
 HRESULT _stdcall Project([out, retval]
ICodeWarriorProject** pval);
 [id(0x00000064), propput]
 HRESULT _stdcall Project([in] ICodeWarriorProject* pval);
};

[
 odl,
 uuid(9DDD415E-AD7C-11D2-B26C-00C04F72E4D1),

```

## CodeWarrior IDE Interface Definition Language (IDL)

---

```

 dual,
 oleautomation
]
interface ICodeWarriorErrorInfo : IUnknown {
 [id(0x00000064), propput]
 HRESULT _stdcall action([in] BSTR actionStr);
 [id(0x00000064), propget]
 HRESULT _stdcall action([out, retval] BSTR* actionStr);
 [id(0x00000065), propput]
 HRESULT _stdcall Reason([in] BSTR actionStr);
 [id(0x00000065), propget]
 HRESULT _stdcall Reason([out, retval] BSTR* actionStr);
 [id(0x00000066), propput]
 HRESULT _stdcall MWErr([in] long err);
 [id(0x00000066), propget]
 HRESULT _stdcall MWErr([out, retval] long* err);
 [id(0x00000067), propput]
 HRESULT _stdcall HRESULT([in] HRESULT err);
 [id(0x00000067), propget]
 HRESULT _stdcall HRESULT([out, retval] HRESULT* err);
 [id(0x00000068), propput]
 HRESULT _stdcall DWORDErr([in] unsigned long err);
 [id(0x00000068), propget]
 HRESULT _stdcall DWORDErr([out, retval] unsigned long*
err);
 [id(0x00000069), propput]
 HRESULT _stdcall MacOSErr([in] short err);
 [id(0x00000069), propget]
 HRESULT _stdcall MacOSErr([out, retval] short* err);
 [id(0x0000006a), propput]
 HRESULT _stdcall Source([in] BSTR rhs);
 [id(0x0000006b), propput]
 HRESULT _stdcall HelpContext([in] unsigned long rhs);
 [id(0x0000006c), propput]
 HRESULT _stdcall HelpFile([in] BSTR rhs);
};
};

```

## PowerPC Object Code (Mac OS)

---

This chapter describes the object code format used by the CodeWarrior plug-in compilers and linkers for the PowerPC family of processors.

This chapter contains the following sections:

- PowerPC Object Code Structure
- PowerPC Object Header
- PowerPC Object Data Section
- PowerPC Symbolic Data Header
- PowerPC Symbolic Function Data Section
- PowerPC Symbolic Type Data
- PowerPC Name Table

You should bear in mind the following points when working with the CodeWarrior tools for developing PowerPC applications:

- The MPW-hosted compiler MWCPPC generates a library file.
- All values are in big-endian order. A `SInt32` is a 32-bit signed integral type (commonly a C/C++ `long`), while a `SInt16` is a 16-bit signed integral type (commonly a C/C++ `short`).
- The Metrowerks PowerPC linker handles object code in the Metrowerks PowerPC object code format and in XCOFF.



## PowerPC Object Code Structure

The structure of CodeWarrior’s PowerPC object code can be broken down into the following sections:

- PowerPC Object Header
- PowerPC Object Data Section
- PowerPC Symbolic Data Header (optional)
  - PowerPC Symbolic Function Data Section
  - PowerPC Symbolic Type Data
- PowerPC Name Table

## PowerPC Object Header

The following describes the object header and its fields.

**Listing 28.1** PowerPC object header structure

```
typedef struct ObjHeader { // object file header
 SInt32 magic_word;
 SInt16 version;
 SInt16 flags;
 SInt32 obj_size;
 SInt32 nametable_offset;
 SInt32 nametable_names;
 SInt32 symtable_offset;
 SInt32 symtable_size;
 SInt32 code_size;
 SInt32 udata_size;
 SInt32 idata_size;
 SInt32 toc_size;
 SInt32 old_def_version;
 SInt32 old_imp_version;
 SInt32 current_version;
 SInt32 reserved[13];
} ObjHeader;
```

Table 28.2 describes the values in Listing 28.1.

Table 28.2 PowerPC object header fields

| This field       | Contains this information                                                                                                                                                                                                                        |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| magic_word       | Always OBJ_MAGIC_WORD, defined to be 0x504F5752, ('POWR' in ASCII).                                                                                                                                                                              |
| version          | The version number of the object code format. Its value is OBJ_VERSION.                                                                                                                                                                          |
| flags            | A bitfield describing the type of object data, shown in Listing 28.2.                                                                                                                                                                            |
| obj_size         | The size, in bytes, of the object data section that follows the object header.                                                                                                                                                                   |
| nametable_offset | The offset to the first byte of the name table, relative to the start of the object header. For more information on the name table see "PowerPC Name Table" on page 929.                                                                         |
| nametable_names  | The number of strings defined in the name table.                                                                                                                                                                                                 |
| symtable_offset  | The offset to the first byte of the symbolic data header, relative to the start of the object header. If there is no symbolic data for this object code, then this field contains 0L.                                                            |
| symtable_size    | The total size in bytes of all of the symbolic data, including the symbolic data header, the symbolic function data section, and the symbolic type data section. If there is no symbolic data for this object code, then this field contains 0L. |
| code_size        | The size, in bytes, of the code defined in the object data section.                                                                                                                                                                              |
| udata_size       | The size, in bytes, of the uninitialized data defined in the object data section.                                                                                                                                                                |
| idata_size       | The size, in bytes, of the initialized data defined in the object data section.                                                                                                                                                                  |
| toc_size         | The size, in bytes, of the TOC data defined in the object data section.                                                                                                                                                                          |

**PowerPC Object Code (Mac OS)**  
*PowerPC Object Data Section*

| This field      | Contains this information                                                                                                                 |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| old_def_version | The old definition version number of the code fragment. For object code that doesn't define a shared library, this field contains 0L.     |
| old_imp_version | The old implementation version number of the code fragment. For object code that doesn't define a shared library, this field contains 0L. |
| current_version | The current version number of the code fragment. For object code that doesn't define a shared library, this field contains 0L.            |
| reserved        | Reserved by Metrowerks. All elements must contain 0L.                                                                                     |

Listing 28.2 shows the values of the `flags` bitfield.

**Listing 28.2    Defined values for `flags`**

```
enum {
 fObjIsSharedLib = 0x0001, /* A shared library */
 fObjIsLibrary = 0x0002, /* A library */
 fObjIsPascal = 0x0004, /* A pascal source file */
 fObjIsWeak = 0x0008, /* A CFM lib, "Weak Import" */
 fObjIsInitBefore = 0x0010 /* A CFM lib, "Initialize Before" */
};
```

## PowerPC Object Data Section

The object data section is composed of a series of containers called "hunks". There are several different types of hunks, each one playing a different role in the object data definition. The object data section begins with a special starting hunk and ends with a special ending hunk. Each hunk structure begins with a tag that uniquely identifies its type.

Listing 28.3 shows the defined hunk types.



# Freescale Semiconductor, Inc.

PowerPC Object Code (Mac OS)  
PowerPC Object Data Section

---

## Listing 28.3 PowerPC hunk types

---

```
enum {
 HUNK_START=0x4567,
 HUNK_END,
 HUNK_SEGMENT,
 HUNK_LOCAL_CODE,
 HUNK_GLOBAL_CODE,
 HUNK_LOCAL_UDATA,
 HUNK_GLOBAL_UDATA,
 HUNK_LOCAL_IDATA,
 HUNK_GLOBAL_IDATA,
 HUNK_GLOBAL_ENTRY,
 HUNK_LOCAL_ENTRY,
 HUNK_IMPORT,
 HUNK_XREF_16BIT,
 HUNK_XREF_16BIT_IL,
 HUNK_XREF_24BIT,
 HUNK_XREF_32BIT,
 HUNK_XREF_32BIT_REL,
 HUNK_DEINIT_CODE, /* Reserved */
 HUNK_LIBRARY_BREAK, /* Obsolete */
 HUNK_IMPORT_CONTAINER,
 HUNK_SOURCE_BREAK,
 HUNK_XREF_16BIT_REL,
 HUNK_METHOD_REF,
 HUNK_CLASS_DEF,
 HUNK_FORCE_ACTIVE
};
```

---

The rest of this section discusses the following topics:

- Preventing Dead-Stripping
- PowerPC Simple Hunks
- PowerPC Regular Code Hunks
- PowerPC Data Hunks
- PowerPC Alternate Entry Point Hunks
- PowerPC Cross-Reference Hunks
- PowerPC PEF Import Hunks
- PowerPC Source File Specification Hunks

- PowerPC Reserved Hunks

## Preventing Dead-Stripping

A `HUNK_FORCE_ACTIVE` hunk causes the linker to never strip out the object defined by the following hunk.

## PowerPC Simple Hunks

The first hunk of the object code is of type `HUNK_START`. The last hunk is of type `HUNK_END`. The object code contains no other hunks of these types. The `ObjMiscHunk` structure defines these simple hunks.

Listing 28.4 shows the `ObjMiscHunk` structure.

### Listing 28.4 PowerPC simple hunk structure

---

```
typedef struct ObjMiscHunk {
 SInt16 hunk_type;
 SInt16 unused; /* Padding */
} ObjMiscHunk;
```

---

## PowerPC Regular Code Hunks

A `HUNK_LOCAL_CODE` hunk defines the code for a function with static (that is, internal) linkage. The `ObjCodeHunk` structure defines a `HUNK_LOCAL_CODE` hunk. The structure must have the relevant machine code immediately after it.

A `HUNK_GLOBAL_CODE` hunk defines the code for a function with external linkage. The `ObjCodeHunk` structure defines a `HUNK_GLOBAL_CODE` hunk. The structure must have the relevant machine code immediately after it.

Listing 28.5 shows the `ObjCodeHunk` structure.

### Listing 28.5 PowerPC regular object code structure

---

```
typedef struct ObjCodeHunk {
 SInt16 hunk_type;
```

---



Freescale Semiconductor, Inc.

PowerPC Object Code (Mac OS)  
PowerPC Regular Code Hunks

```

char sm_class;
unsigned char
// multi_def : 1,
// over_load : 1,
// exported : 1,
// reserved : 1, /* Reserved */
// alignment : 4;
SInt32 name_id;
SInt32 size;
SInt32 sym_offset;
SInt32 sym_decl_offset;
// char code[size];
} ObjCodeHunk;

```

Table 28.3 describes the values in Listing 28.5.

**Table 28.3**    PowerPC regular object code fields

| This field | Contains this information                                                                                                                                 |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | Either HUNK_LOCAL_CODE or HUNK_GLOBAL_CODE.                                                                                                               |
| sm_class   | What type of data the hunk defines. Listing 28.6 shows the possible values for this field. The Metrowerks PowerPC linker supports only XMC_PR and XMC_GL. |
| multi_def  | If true, then this hunk may have multiple identical definitions. If false, the module does not have multiple definitions.                                 |
| over_load  | If true, another definition may overload this hunk. If false, no other definition overloads this hunk.                                                    |
| exported   | If true, this hunk will be exported. If false, the module will not be exported.                                                                           |
| reserved   | Reserved for future use.                                                                                                                                  |

**PowerPC Object Code (Mac OS)**  
*PowerPC Regular Code Hunks*

| This field      | Contains this information                                                                                                                                                                                                                                                |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| alignment       | This field has several possible values, each of which has a different meaning:<br>1: alignment to the next byte<br>2: alignment to the next half-word<br>4: alignment to the next word<br>8: alignment to the next double-word<br>odd number other than 1: <<(number)>>1 |
| name_id         | The item in the name table that specifies the name of the module or routine this hunk contains. For more information on the name table see “PowerPC Name Table” on page 929.                                                                                             |
| size            | The size, in bytes, of this object code hunk.                                                                                                                                                                                                                            |
| sym_offset      | The byte offset of the function’s symbolic data in the function symbolic data section, relative to the start of the symbolic data header. If there is no symbolic data for this function, this field contains 0x80000000.                                                |
| sym_decl_offset | The character offset where the method appears in the source file. If no symbolic data for this function exists, this field contains 0L.                                                                                                                                  |

Listing 28.6 shows the definitions for the storage mapping classes.

**Listing 28.6 PowerPC types for storage mapping classes**

```

/* Read-only classes */
#define XMC_PR 0 /* Program Code */
#define XMC_RO 1 /* Read Only Constant */
#define XMC_GL 6 /* Global Linkage */

/* Read/write classes */
#define XMC_RW 5 /* Read Write Data */
#define XMC_TC0 15 /* TOC Anchor */
#define XMC_TC 3 /* General TOC Entry */
#define XMC_TD 16 /* Scalar TOC Data */
#define XMC_DS 10 /* Routine Descriptor */

```

## PowerPC Data Hunks

A `HUNK_LOCAL_UDATA` hunk defines a block of memory used for the storage of uninitialized data with `static` linkage. A `HUNK_GLOBAL_UDATA` hunk defines a block of memory used for the storage of uninitialized data with `extern` linkage. `ObjDataHunk` defines the structure of both kinds of hunks.

A `HUNK_LOCAL_IDATA` hunk defines a block of memory used for the storage of a piece of initialized data with `static` linkage. A `HUNK_GLOBAL_IDATA` hunk defines a block of memory used for the storage of a piece of initialized data with `extern` linkage. `ObjDataHunk` defines the structure of both kinds of hunks. For both kinds of hunk, the data to be stored must be appended immediately after the hunk's declaration.

Listing 28.7 shows the structure for `ObjDataHunk`.

**Listing 28.7 Structure for PowerPC initialized and uninitialized data**

```
typedef struct ObjDataHunk {
 SInt16 hunk_type;
 char sm_class;
 unsigned char
 // multi_def : 1,
 // over_load : 1,
 // exported : 1,
 // reserved : 1, /* Reserved */
 // alignment : 4;
 SInt32 name_id;
 SInt32 size;
 SInt32 sym_type_id;
 SInt32 sym_decl_offset;
 // char data[size];
} ObjDataHunk;
```

**Table 28.4** PowerPC initialized and uninitialized data fields

| This field | Contains this information                                                                                                                                                                                                                                                    |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | HUNK_LOCAL_IDATA,<br>HUNK_GLOBAL_IDATA,<br>HUNK_LOCAL_UDATA, or<br>HUNK_GLOBAL_UDATA                                                                                                                                                                                         |
| sm_class   | For a list of possible values, see Listing 28.6.                                                                                                                                                                                                                             |
| multi_def  | If true, this hunk can have multiple identical definitions. If false, the hunk cannot have multiple definitions.                                                                                                                                                             |
| over_load  | If true, another definition can overload this hunk. If false, no other definition can overload this hunk.                                                                                                                                                                    |
| exported   | If true, this hunk can be exported. If false, this hunk cannot be exported.                                                                                                                                                                                                  |
| reserved   | Reserved for future use.                                                                                                                                                                                                                                                     |
| alignment  | This field has several possible values, each of which has a different meaning:<br>1: alignment to the next byte<br>2: alignment to the next half-word<br>4: alignment to the next word<br>8: alignment to the next double-word<br>odd number other than 1:<br><<(number)>>1) |
| name_id    | The value in the name table that specifies the name of the data this hunk contains. For more information on the name table, see "PowerPC Name Table" on page 929.                                                                                                            |
| size       | The number of bytes that the data occupies in memory. For HUNK_LOCAL_IDATA and HUNK_GLOBAL_IDATA hunks, this value holds the number of bytes of data that follows the ObjDataHunk.                                                                                           |

| This field      | Contains this information                                                                                                                                                              |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| sym_type_id     | Type identifier for the data object. If the object has no data type, this field contains 0x80000000. See “PowerPC Symbolic Type Data” on page 919 for information on type identifiers. |
| sym_decl_offset | The character offset of the data in the source file. If no symbolic data exists for this function, this field contains 0L.                                                             |

PowerPC Alternate Entry Point Hunks

A HUNK\_GLOBAL\_ENTRY or HUNK\_LOCAL\_ENTRY hunk defines an alternate entry point for the function that was last defined with a HUNK\_GLOBAL\_CODE or HUNK\_LOCAL\_CODE hunk. The linkage of HUNK\_GLOBAL\_ENTRY uses an external entry point. The linkage of HUNK\_LOCAL\_ENTRY uses a static entry point. ObjDataHunk defines the structure of both kinds of hunks.

Listing 28.7 shows alternate entry point hunks.

Listing 28.8    PowerPC alternate entry point for PowerPC

```
typedef struct ObjEntryHunk {
 SInt16 hunk_type;
 SInt16 unused;
 SInt32 name_id;
 SInt32 offset;
 SInt32 sym_type_id;
 SInt32 sym_decl_offset;
} ObjEntryHunk;
```

Table 28.5 describes the fields in an alternate entry point structure.

**Table 28.5 PowerPC alternate entry point fields**

| This field      | Contains this information                                                                                                                                                              |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type       | Either HUNK_GLOBAL_ENTRY or HUNK_LOCAL_ENTRY.                                                                                                                                          |
| unused          | Padding.                                                                                                                                                                               |
| name_id         | The item in the name table that specifies the name of the alternate entry point. For more information on the name table see “PowerPC Name Table” on page 929.                          |
| offset          | The offset of the alternate entry point from the beginning of the module or routine contained in the previous hunk, in bytes.                                                          |
| sym_type_id     | Type identifier for the data object. If the object has no data type, this field contains 0x80000000. See “PowerPC Symbolic Type Data” on page 919 for information on type identifiers. |
| sym_decl_offset | The character offset of the routine in the source file. If no symbolic data for this function exists, this field contains 0L.                                                          |

## PowerPC Cross-Reference Hunks

Hunks of types HUNK\_XREF\_16BIT, HUNK\_XREF\_16BIT\_IL, HUNK\_XREF\_24BIT, HUNK\_XREF\_32BIT, HUNK\_XREF\_32BIT\_REL, and HUNK\_XREF\_16BIT\_REL mark positions in the previous code or data hunk whose value depends on the load address of another symbol. The different types mark the different relocation addressing modes supported by the linker. These hunks must follow a code hunk and any entry hunks the code hunk might have. ObjXRefHunk defines these hunks.

Listing 28.9 shows the ObjXRefHunk structure

**Listing 28.9 PowerPC cross-reference structure**

```
typedef struct ObjXRefHunk {
 SInt16 hunk_type;
```

```
char sm_class;
char unused;
SInt32 name_id;
SInt32 offset;
} ObjXRefHunk;
```

Table 28.6 describes the fields in the ObjXRefHunk structure.

Table 28.6 PowerPC cross-reference fields

| This field | Contains this information                                                                                                                                                    |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | HUNK_XREF_16BIT, HUNK_XREF_16BIT_IL, HUNK_XREF_24BIT, HUNK_XREF_32BIT, HUNK_XREF_32BIT_REL, or HUNK_XREF_16BIT_REL                                                           |
| sm_class   | The storage-mapping class of the referenced symbol. See Listing 28.6 for information on storage mapping classes.                                                             |
| unused     | Padding.                                                                                                                                                                     |
| name_id    | The value in the name table that specifies the name of the hunk to which this structure refers. For more information on the name table see “PowerPC Name Table” on page 929. |
| offset     | The offset of the cross-reference within the hunk to which this hunk applies.                                                                                                |

PowerPC PEF Import Hunks

A HUNK\_IMPORT\_CONTAINER hunk specifies the name and PEF version numbers for a code fragment from which symbol definitions are imported. Hunks of this type only appear in shared library object code. ObjContainerHunk defines the structure of import container hunks.

Listing 28.10 shows the definition of the ObjContainerHunk structure.



**Listing 28.10**    PowerPC PEF import container structure

```

typedef struct ObjContainerHunk {
 SInt16 hunk_type;
 UInt16 flags;
 // unused: 15, /* Reserved */
 // auto_weak: 1;
 SInt32 name_id;
 SInt32 old_def_version;
 SInt32 old_imp_version;
 SInt32 current_version;
} ObjContainerHunk;

```

**Table 28.7**    PowerPC PEF import container fields

| This field       | Contains this information                                                                                                                                                                                                                                  |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type        | HUNK_IMPORT_CONTAINER                                                                                                                                                                                                                                      |
| flags: auto_weak | 1 if the linker should automatically mark the imported fragment as weak, regardless of the user’s "import Weak" setting, or 0 if not. The PEF Importer sets this bit if the shared library's cfrg resource's usage field equals kWeakStubLibraryCFrag (4). |
| name_id          | The item in the name table that specifies the run-time name of the import container. For more information on the name table, see "PowerPC Name Table" on page 929.                                                                                         |
| old_def_version  | The old definition version of this PEF container.                                                                                                                                                                                                          |
| old_imp_version  | The old implementation version of this PEF container.                                                                                                                                                                                                      |
| current_version  | The current version of this PEF container.                                                                                                                                                                                                                 |

A HUNK\_IMPORT hunk specifies a symbol that is imported from a different code fragment (the one that was last named with a HUNK\_IMPORT\_CONTAINER hunk). A hunk of this type appears

only in the object data for a shared library. Its structure is an ObjImportHunk.

Listing 28.11 shows the ObjImportHunk structure.

Listing 28.11 PowerPC PEF import symbol structure

```
typedef struct ObjImportHunk {
 SInt16 hunk_type;
 char sm_class;
 char unused;
 SInt32 name_id;
} ObjImportHunk;
```

Table 28.8 PowerPC PEF import fields

| This field | Contains this information                                                                                                                                |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | HUNK_IMPORT                                                                                                                                              |
| sm_class   | The storage class of the imported symbol, which must be XMC_RW or XMC_DS. See Listing 28.6 for other information on storage classes.                     |
| unused     | Padding.                                                                                                                                                 |
| name_id    | The value in the name table that specifies the name of the imported symbol. For more information on the name table see “PowerPC Name Table” on page 929. |

PowerPC Source File Specification Hunks

A HUNK\_SOURCE\_BREAK hunk specifies the name of the source file that defines subsequent code and data. The hunk contains the index of the entry in the name table that contains the file’s full path. Among other uses, you can use this hunk to tell a source-level debugger that a header file defines the code and data that follows the hunk. If the name index of the source file in a subsequent HUNK\_SOURCE\_BREAK hunk is 0, then the source file reverts to the original file. ObjSourceHunk defines the HUNK\_SOURCE\_BREAK structure.

Listing 28.12 shows the structure of ObjSourceHunk.



Freescale Semiconductor, Inc.

**PowerPC Object Code (Mac OS)**  
*PowerPC Object Pascal Hunks*

**Listing 28.12    Source file specification structure for PowerPC**

```

typedef struct ObjSourceHunk {
 SInt16 hunk_type;
 SInt16 unused;
 SInt32 name_id;
 SInt32 moddate;
} ObjSourceHunk;

```

Table 28.9 describes the fields in the ObjSourceHunk structure.

**Table 28.9    PowerPC source file specification fields**

| This field | Contains this information                                                                                                                                                                                                             |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | HUNK_SRC_BREAK                                                                                                                                                                                                                        |
| unused     | Padding.                                                                                                                                                                                                                              |
| name_id    | The item within the name table that specifies the full path name of the file that contains the source code. Use 0L to specify the original source file. For more information on the name table, see “PowerPC Name Table” on page 929. |
| moddate    | The modification date of this file, in the format used by the Mac OS GetDate() routine.                                                                                                                                               |

**PowerPC Object Pascal Hunks**

Metrowerks Object Pascal uses the HUNK\_METHOD\_REF hunk type to specify a method table. The ObjMethHunk structure defines HUNK\_METHOD\_REF.

Listing 28.13 shows the structure of ObjMethHunk.

**Listing 28.13    PowerPC Object Pascal method table structure**

```

typedef struct ObjMethHunk {
 SInt16 hunk_type;
 SInt32 name_id;
 SInt32 size;
} ObjMethHunk;

```

Table 28.10 describes the members of ObjMethHunk.

**Table 28.10    PowerPC Object Pascal method table fields**

| This field | Contains this information                                                                                                                                 |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | HUNK_METHOD_REF                                                                                                                                           |
| name_id    | The item within the name table that specifies the name of the method table. For more information on the name table, see “PowerPC Name Table” on page 929. |
| size       | The number of method table references.                                                                                                                    |

Object Pascal uses the HUNK\_CLASS\_DEF hunk type to specify a class. ObjClassHunk defines the structure of HUNK\_CLASS\_DEF.

Listing 28.14 shows the structure of ObjClassHunk.

**Listing 28.14    PowerPC Object Pascal class structure**

```
typedef struct ObjClassHunk {
 SInt16 hunk_type;
 SInt32 name_id;
 SInt16 methods;
 SInt16 pairs;
} ObjClassHunk;
```

Table 28.11 describes the members of ObjClassHunk.

**Table 28.11    PowerPC Object Pascal class fields**

| This field | Contains this information                                                                                                                         |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| hunk_type  | HUNK_CLASS_REF                                                                                                                                    |
| name_id    | The item within the name table that specifies the name of the class. For more information on the name table see “PowerPC Name Table” on page 929. |



# Freescale Semiconductor, Inc.

**PowerPC Object Code (Mac OS)**  
*PowerPC Reserved Hunks*

| This field | Contains this information                                                          |
|------------|------------------------------------------------------------------------------------|
| methods    | The number of methods in the class.                                                |
| pairs      | The number of base classes. See Listing 28.15 for information on base class pairs. |

The ObjClassPair structure defines base pairs.

Listing 28.15 shows the structure of ObjClassPair.

**Listing 28.15    PowerPC Object Pascal class base pair**

```
typedef struct ObjClassPair {
 SInt32 base_id;
 SInt32 bias;
} ObjClassPair;
```

Table 28.12 describes the fields in ObjClassPair.

**Table 28.12    PowerPC Object Pascal class base pair fields**

| This field | Contains this information                                                                                             |
|------------|-----------------------------------------------------------------------------------------------------------------------|
| base_id    | The base class ID within the name table. For more information on the name table see “PowerPC Name Table” on page 929. |
| bias       | The base bias in the class.                                                                                           |

## PowerPC Reserved Hunks

PowerPC object code does not use hunks of type HUNK\_SEGMENT.

The CodeWarrior IDE reserves the following hunks:

- HUNK\_INIT\_CODE
- HUNK\_DEINIT\_CODE
- HUNK\_LIBRARY\_BREAK



PowerPC Symbolic Data Header

The symbolic data header, the symbolic function data section, and the symbolic type data section are optional. The object header records the location of the symbolic data header.

Listing 28.16 shows the SymHeader structure.

Listing 28.16    PowerPC symbolic data header structure

```

typedef struct {
 SInt32 header_id;
 SInt32 typeoffset;
 SInt32 types;
 SInt32 unnamed;
 SInt32 res[4];
} SymHeader;
```

Table 28.13 describes the fields in SymHeader.

Table 28.13    PowerPC symbolic data header fields

| This field | Contains this information                                                                                                                                                                 |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| header_id  | Always SYM_HEADER_ID, defined to be 0x53594D48 ('SYMH' in ASCII).                                                                                                                         |
| typeoffset | The byte offset to the type data, relative to the start of the symbolic data header.                                                                                                      |
| types      | The number of types defined in the type data section.                                                                                                                                     |
| unnamed    | The number of unnamed types defined in the type data section. Structure and enumerated types are unnamed in C/C++ when they don't have a tag. Anonymous types are also unnamed in Pascal. |
| res        | Reserved for future use. This is an array of 4 SInt32 values, each containing 0L.                                                                                                         |

## PowerPC Symbolic Function Data Section

The symbolic function data section contains a series of variable-length records, each holding information that describes a function or procedure. Each record begins with a `SInt16` that contains 0 for a procedure (that is, a routine without a return value) or 1 for a function (a routine that returns a value).

A list of statement locations follows the `SInt16`. Each statement location defines a correspondence between an offset in the function code and a location in the source code where the user can place a breakpoint in a debugger. Each entry in the statement list contains two `SInt32` values. The first `SInt32` value holds the code offset, relative to the start of the function. The second `SInt32` value holds the character offset of the statement location in the source file. A special list entry whose first `SInt32` is `0xFFFF` and whose second `SInt32` is the same as the last statement's terminates the list.

A `SInt16` containing the number of local variables and parameters defined in this function follows the statement. The remainder of the data contains a corresponding number of local variable definitions.

Listing 28.17 shows the `LocalVar` structure.

**Listing 28.17    PowerPC local variable and parameter structure**

```

typedef struct {
 SInt32 name;
 SInt32 type;
 char kind;
 char sclass;
 SInt32 where;
} LocalVar;

```

Table 28.14 describes the fields in `LocalVar`.

**Table 28.14    PowerPC local variable and parameter fields**

| This field | Contains this information                                                                                                                      |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The index of the local variable’s name in the name table section. See “PowerPC Name Table” on page 929 for more information on the name table. |
| type       | The type identifier of the local variable. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919.                  |
| kind       | How the variable is stored. This field contains one of the values listed in Listing 28.18.                                                     |
| sclass     | The variable’s storage class. This field contains one of the values listed in Listing 28.19.                                                   |
| where      | Where the variable is stored. This value depends on the value of the sclass field. See Listing 28.19 for more information.                     |

An anonymous enumeration defines the possible values for the value of the kind field.

Listing 28.18 shows the values in the anonymous enumeration.

**Listing 28.18    Kinds of local variable and parameter storage for PowerPC**

```
enum {
 STORAGE_KIND_LOCAL=0,
 STORAGE_KIND_VALUE,
 STORAGE_KIND_REFERENCE
};
```

Table 28.15 describes the meaning of each value in the anonymous enumeration.

**Table 28.15    PowerPC local variable and parameter storage values**

| This value         | Specifies this information |
|--------------------|----------------------------|
| STORAGE_KIND_LOCAL | Local variables.           |



# Freescal Semiconductor, Inc.

**PowerPC Object Code (Mac OS)**  
*PowerPC Symbolic Function Data Section*

| This value             | Specifies this information |
|------------------------|----------------------------|
| STORAGE_KIND_VALUE     | Parameters.                |
| STORAGE_KIND_REFERENCE | Pascal VAR parameters.     |

Listing 28.19 shows the structure of the anonymous enumeration that defines storage classes for the PowerPC.

**Listing 28.19 Register storage for PowerPC**

```
enum {
 STORAGE_CLASS_REGISTER=0,
 STORAGE_CLASS_A5,
 STORAGE_CLASS_A6,
 STORAGE_CLASS_A7
};
```

Table 28.16 describes the fields in the anonymous enumeration that defines storage classes for the PowerPC.

**Table 28.16 PowerPC register storage values**

| This value             | Specifies this information                                                                                                                                                                          |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| STORAGE_CLASS_REGISTER | A variable stored in a register.<br><br>The <code>where</code> field identifies the register: 0-31 identify GPRs, 32-63 identify FPRs.                                                              |
| STORAGE_CLASS_A6       | A variable stored on the stack in a function that supports <code>alloca()</code> . GPR31 is reserved as a stack frame pointer.<br><br>The <code>where</code> field contains the stack frame offset. |
| STORAGE_CLASS_A7       | A variable stored on the stack.<br><br>The <code>where</code> field contains the stack offset.                                                                                                      |
| STORAGE_CLASS_A5       | Not used.                                                                                                                                                                                           |

## PowerPC Symbolic Type Data

Certain predefined types do not require type definitions.

Listing 28.20 shows the predefined types and their values.

**Listing 28.20 PowerPC symbolic type data section structure**

---

```
enum {
 BASICTYPE_VOID, /* No type */
 BASICTYPE_PSTRING, /* Pascal string */
 BASICTYPE_ULONG, /* unsigned long word */
 BASICTYPE_LONG, /* signed long word */
 BASICTYPE_FLOAT10, /* extended (10 bytes) */
 BASICTYPE_BOOLEAN, /* Pascal boolean (1 byte) */
 BASICTYPE_UBYTE, /* unsigned byte */
 BASICTYPE_BYTE, /* signed byte */
 BASICTYPE_CHAR, /* character (1 byte) */
 BASICTYPE_WCHAR, /* character (2 bytes) */
 BASICTYPE_UWORD, /* unsigned word */
 BASICTYPE_WORD, /* signed word */
 BASICTYPE_FLOAT4, /* single */
 BASICTYPE_FLOAT8, /* double */
 BASICTYPE_FLOAT12, /* extended (12 bytes) */
 BASICTYPE_COMP, /* computational (8 bytes) */
 BASICTYPE_CSTRING, /* C string */
 BASICTYPE_AISTRING, /* as-is string */
 MYBASICTYPE_VOIDPTR=100, /* void * */
 MYBASICTYPE_VOIDHDL, /* void ** */
 MYBASICTYPE_CHARPTR, /* char * */
 MYBASICTYPE_CHARHDL, /* char ** */
 MYBASICTYPE_UCHARPTR, /* unsigned char * */
 MYBASICTYPE_UCHARHDL, /* unsigned char ** */
 MYBASICTYPE_FUNC, /* void func(void) */
 MYBASICTYPE_STRINGPTR, /* C string pointer */
 MYBASICTYPE_PSTRINGPTR, /* Pascal str. pointer */
};
```

---

The MYBASICTYPE\_CHARPTR type differs from the MYBASICTYPE\_STRINGPTR type only in semantics. A debugger might display a MYBASICTYPE\_CHARPTR type as a pointer to

## PowerPC Object Code (Mac OS)

*Other Types for PowerPC*

character and display a MYBASICTYPE\_STRINGPTR type directly as a C string without any dereferencing or type coercion.

The rest of this section discusses the following types:

- Other Types for PowerPC
- PowerPC Pointer Type
- PowerPC Array Type
- PowerPC Structured Type
- PowerPC Enumerated Type
- PowerPC Pascal Array Type
- PowerPC Pascal Subrange Type
- PowerPC Pascal String Type

## Other Types for PowerPC

Other types require a type definition. The symbolic type data section stores these type definitions in a series of variable-length records. The symbolic data header stores the number of user-defined types.

Each type definition begins with a tag and a type ID. The SInt16 tag identifies the structure of the remainder of the definition.

Type IDs for user-defined types are sequentially defined in decreasing order, starting with -1. Positive type IDs are reserved for pre-defined types.

Listing 28.21 shows the pre-defined types.

### Listing 28.21 Other data types for PowerPC

```
enum {
 LOCTYPE_POINTER,
 LOCTYPE_ARRAY,
 LOCTYPE_STRUCT,
 LOCTYPE_ENUM,
 LOCTYPE_PARRAY,
 LOCTYPE_RANGE,
 LOCTYPE_SET,
 LOCTYPE_PENUM,
```



```
 LOCTYPE_PSTRING
};
```

Table 28.17 describes each pre-defined type.

Table 28.17 PowerPC other data type tags

| This tag        | Specifies this information |
|-----------------|----------------------------|
| LOCTYPE_POINTER | A pointer type.            |
| LOCTYPE_ARRAY   | An array type.             |
| LOCTYPE_STRUCT  | A structured type.         |
| LOCTYPE_ENUM    | An enumeration type.       |
| LOCTYPE_PARRAY  | A Pascal array type.       |
| LOCTYPE_RANGE   | A Pascal subrange type.    |
| LOCTYPE_SET     | A Pascal set type.         |
| LOCTYPE_PENUM   | A Pascal enumeration type. |
| LOCTYPE_PSTRING | A Pascal string type.      |

For more information about the pre-defined types, see “PowerPC Pascal String Type” on page 929.

PowerPC Pointer Type

A pointer type definition consists of a LOCTYPE\_POINTER tag, followed by the SInt32 type ID being defined, followed by a LocPointerStruct.

Listing 28.22 shows the LocPointerStruct structure.

Listing 28.22 PowerPC pointer data type structure

```
typedef struct LocPointerStruct {
 SInt16 number;
 SInt32 type;
} LocPointerStruct;
```

Table 28.18 describes the fields in LocPointerStruct.

**PowerPC Object Code (Mac OS)**  
*PowerPC Array Type*

**Table 28.18 PowerPC pointer type fields**

| This field | Contains this information                                                                                                               |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| number     | The number of pointer indirections. For example, in C/C++, 1 is equivalent to *, 2 is **, 3 is *** and so on.                           |
| type       | The type ID of the item to which the pointer refers. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919. |

## PowerPC Array Type

An array type definition consists of a `LOCTYPE_ARRAY` tag, followed by the `SInt32` type ID being defined, followed by a `LocArrayStruct`.

Listing 28.23 shows the `LocArrayStruct` structure.

**Listing 28.23 PowerPC array data type structure**

```
typedef struct LocArrayStruct {
 SInt32 size;
 SInt32 esize;
 SInt32 type;
} LocArrayStruct;
```

Table 28.19 describes the fields in `LocArrayStruct`.

**Table 28.19 PowerPC array type fields**

| This field | Contains this information                                                                                       |
|------------|-----------------------------------------------------------------------------------------------------------------|
| size       | Size of the array, in bytes.                                                                                    |
| esize      | Size of a single element.                                                                                       |
| type       | The array elements’ type ID. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919. |

PowerPC Structured Type

A struct type definition consists of a LOCTYPE\_STRUCT tag, followed by the SInt32 type ID being defined, followed by a LocStructStruct, followed by a LocStructStructMember for each member of the struct.

Listing 28.24 shows the LocStructStruct structure.

Listing 28.24 PowerPC structured data type structure

```
typedef struct LocStructStruct {
 SInt32 name;
 SInt32 size;
 SInt16 members;
 // LocStructStructMember member[members];
} LocStructStruct;
```

Table 28.20 describes the methods in LocStructStruct.

Table 28.20 PowerPC structured type fields

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |
| size       | The size of the structure.                                                                                                                       |
| members    | The number of members within the structure.                                                                                                      |

Listing 28.25 shows the LocStructStructMember structure.

Listing 28.25 PowerPC structured type member structure

```
typedef struct LocStructStructMember {
 SInt32 name;
 SInt32 type;
 SInt32 offset;
} LocStructStructMember;
```

Table 28.21 describes the fields in LocStructStructMember.

**PowerPC Object Code (Mac OS)**  
*PowerPC Enumerated Type*

**Table 28.21 PowerPC member structure fields**

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |
| type       | The type ID of the member. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919.                                    |
| offset     | The offset of the member from the beginning of the structure, in bytes.                                                                          |

## PowerPC Enumerated Type

An enumerated type definition consists of a `LOCTYPE_ENUM` tag, followed by the `SInt32` type ID being defined, followed by a `LocEnumStruct`, followed by a `LocEnumMember` for each member of the struct.

Listing 28.26 shows the `LocEnumStruct` structure.

**Listing 28.26 PowerPC enumerated type structure**

```
typedef struct LocEnumStruct {
 SInt32 name;
 SInt16 baseid;
 SInt16 members;
 // LocEnumMember member[members];
} LocEnumStruct;
```

Table 28.22 describes the fields in `LocStructStructMember`.

**Table 28.22 PowerPC enumeration type fields**

| This field | Contains this information                                                                                                                                        |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929.                 |
| baseid     | The type ID of the enumeration’s base type. This must always be a basic type. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919. |
| members    | The number of members within the enumeration.                                                                                                                    |

Listing 28.27 shows the `LocEnumMember` structure.

**Listing 28.27 PowerPC enumerated type member structure**

```
typedef struct LocEnumMember {
 SInt32 name;
 SInt32 value;
} LocEnumMember;
```

Table 28.23 describes the fields in `LocEnumMember`.

**Table 28.23 PowerPC member enumeration fields**

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |
| value      | The value of the enumeration member.                                                                                                             |

## PowerPC Pascal Array Type

A Pascal array type definition consists of a `LOCTYPE_PARRAY` tag, followed by the `SInt32` type ID being defined, followed by a `LocArrayStruct`.

Listing 28.28 shows the `LocArrayStruct` structure.

**PowerPC Object Code (Mac OS)**  
*PowerPC Pascal Subrange Type*

**Listing 28.28 PowerPC Pascal array type structure**

```
typedef struct LocPArrayStruct {
 SInt32 pckd;
 SInt32 size;
 SInt32 iid;
 SInt32 eid;
 SInt32 name;
} LocPArrayStruct;
```

Table 28.24 describes the fields in LocPArrayStruct.

**Table 28.24 PowerPC Pascal array type fields**

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| pckd       | true if the array is packed or false if not.                                                                                                     |
| size       | The size, in bytes, of the array.                                                                                                                |
| iid        | The type ID of the array's index.                                                                                                                |
| eid        | The type ID of the array's elements. For information on type identifiers, see "PowerPC Symbolic Type Data" on page 919.                          |
| name       | The item within the name table that specifies the name of the type. For more information on the name table see "PowerPC Name Table" on page 929. |

## PowerPC Pascal Subrange Type

A Pascal subrange type definition consists of a LOCTYPE\_RANGE tag, followed by the SInt32 type ID being defined, followed by a LocRangeStruct.

Listing 28.29 shows the LocRangeStruct structure.

**Listing 28.29 PowerPC Pascal subrange structure**

```
typedef struct LocRangeStruct {
 SInt32 name;
 SInt32 base;
 SInt32 size;
 SInt32 lbound;
```

```
SInt32 hbound;
} LocRangeStruct;
```

Table 28.25 describes the fields in LocRangeStruct.

**Table 28.25 PowerPC Pascal subrange type fields**

| This field | Contains this information                                                                                                                         |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The value within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |
| base       | The type ID of the subrange’s base type. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919.                       |
| size       | The size, in bytes, of the subrange.                                                                                                              |
| lbound     | The lower bound value of the subrange.                                                                                                            |
| hbound     | The upper bound value of the subrange.                                                                                                            |

### PowerPC Pascal Set Type

A Pascal set type definition consists of a LOCTYPE\_SET tag, followed by the SInt32 type ID being defined, followed by a LocSetStruct.

Listing 28.30 shows the LocSetStruct structure.

**Listing 28.30 PowerPC Pascal set type structure**

```
typedef struct LocSetStruct {
 SInt32 name;
 SInt32 base;
 SInt32 size;
} LocSetStruct;
```

Table 28.26 describes the fields in LocSetStruct.

**Table 28.26 PowerPC Pascal set type fields**

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |
| base       | The type ID of the set’s base type. For information on type identifiers, see “PowerPC Symbolic Type Data” on page 919.                           |
| size       | The size, in bytes, of the set.                                                                                                                  |

## PowerPC Pascal Enumerated Type

A Pascal enumerated type definition consists of a LOCTYPE\_PENUM tag, followed by the SInt32 type ID being defined, followed by a LocPEnumStruct, followed by a SInt32 for each enumerated value of the type. The values of the SInt32s are the indexes into the name table for the names of the enumerated values.

Listing 28.31 shows the LocPEnumStruct structure.

**Listing 28.31 PowerPC Pascal enumerated type structure**

```
typedef struct LocPEnumStruct {
 SInt32 name;
 SInt32 count;
 // SInt32 cname[count];
} LocPEnumStruct;
```

Table 28.27 describes the fields in LocPEnumStruct.

**Table 28.27 PowerPC Pascal enumeration type fields**

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |
| count      | The number of constants in the enumeration.                                                                                                      |

### PowerPC Pascal String Type

Listing 28.32 shows the LocPStringStruct structure.

**Listing 28.32    PowerPC Pascal string type structure**

```
typedef struct LocPStringStruct {
 SInt32 size;
 SInt32 name;
} LocPStringStruct;
```

Table 28.28 describes the fields in LocPStringStruct.

**Table 28.28    PowerPC member enumeration fields**

| This field | Contains this information                                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| size       | The number of elements in the character string array.                                                                                            |
| name       | The item within the name table that specifies the name of the type. For more information on the name table see “PowerPC Name Table” on page 929. |

### PowerPC Name Table

The object header specifies the number of string table entries. Each entry of the name table consists of a 16 bit hash value, followed by a null-terminated character string no longer than 255 characters.

Use the routine shown in Listing 28.33 to compute a hash value for the PowerPC name table section.

Other structures in the object code refer to strings in the name table section by a number from 1 to the total number of strings, not by a byte offset. The index value -1 indicates the empty name for unnamed types.

**Listing 28.33    Getting a hash value for the PowerPC name table section**

```
#define NAMEHASH 2048

SInt16 CHash(
 char *string)
```



## Freescal Semiconductor, Inc.

### PowerPC Object Code (Mac OS)

*PowerPC Name Table*

---

```
{
 SInt16 i, hashval;
 unsigned char u;

 if ((hashval = strlen(string) & 0xFF) != 0)
 {
 for (i = hashval, u = 0; i > 0; i--)
 {
 u = (u >> 3) | (u << 5);
 u += *string++;
 }
 hashval = (hashval << 8) | u;
 }
 return (hashval & (NAMEHASH - 1));
}
```

## Symbols

'Dhlp' resource 170, 172  
 'EMap' Resource 167  
 'PPob' 271  
 'Targ' Resource 168

## A

Access Paths 423  
     API Reference 423  
     Data Types 440  
     Interface  
         IcodeWarriorAccessPath 424  
         IcodeWarriorAccessPaths 429  
         IcodeWarriorUserTree 436  
 access paths 46, 95  
     counting 51  
     getting information 50  
     subdirectories, enumerating 52  
 accessing files  
     files open in IDE 72, 74  
 access paths  
     iterating 50  
 Activate  
     IcodeWarriorDocument 572  
 ActivateEvent  
     IcodeWarriorWindowEvents 833  
 Add  
     Collection 489  
 AddAttachment  
     IcodeWarriorDesign 542  
 AddComponentAttachment  
     IcodeWarriorSymbolContainer 694  
 AddCreatableItem  
     IcodeWarriorApp 446  
 AddFile  
     IcodeWarriorDesign 543  
     IcodeWarriorTarget 707  
 AddFile2  
     IcodeWarriorDesign 543  
     IcodeWarriorTarget 708  
 AddFile2ByFileSpec  
     IcodeWarriorDesign 544  
     IcodeWarriorTarget 708  
 AddFile2ByFileSpecCollection  
     IcodeWarriorTarget 709  
 AddFileByFileSpec  
     IcodeWarriorDesign 545

    IcodeWarriorTarget 711  
 AddFileByFileSpecCollection  
     IcodeWarriorTarget 712  
 AddSegment  
     IcodeWarriorTarget 712  
 AddSubTarget  
     IcodeWarriorTarget 713  
 AddUserTree  
     IcodeWarriorApp 446  
     IcodeWarriorTarget 714  
 alert  
     displaying 41  
 allocating memory  
     handles 43  
     permanent 43  
     pointer 42  
 API Reference  
     Access Paths 423  
     Application 444  
 AppendMenuCommand  
     IcodeWarriorMenu 606  
 Apple Events  
     changing settings 351  
     constructing from relative path 318  
     extracting relative path 313  
     getting settings 350  
 Application  
     API Reference 444  
     Data Types 482  
 Application Object, Working with 443  
 ApplyChanges  
     IcodeWarriorAccessPaths 429  
 AreObjectsCreatedInDesign  
     IcodeWarriorCreateObjectItem 530  
 AssociateWindowWithProject 822  
 AttemptModify  
     IcodeWarriorApp 447  
 automatic precompilation 157

## B

balancing  
     calls to load and free text 74  
     calls to lock and unlock a handle 83  
     handle allocation and disposal 48  
 base items  
     determining 285  
 baseItems 340

---

|                                    |     |                                  |     |
|------------------------------------|-----|----------------------------------|-----|
| Browse Header                      | 230 | CanAddFileToProject              |     |
| BROWSE_EARLIEST_COMPATIBLE_VERSION | 244 | IcodeWarriorCreateFileItem       | 527 |
| BROWSE_HEADER                      | 245 | CanCreateUntitledFile            |     |
| BROWSE_VERSION                     | 245 | IcodeWarriorCreateFileItem       | 527 |
| browseCompSymbolStart              | 247 | canFactory                       | 341 |
| BrowseHeader                       | 243 | canRevert                        | 341 |
| browser data                       |     | cantDisassemble                  | 194 |
| class member list                  | 239 | checkbox                         | 283 |
| fields for all records             | 232 | setting value                    | 296 |
| fields for classes                 | 237 | checkFileLocation                | 56  |
| fields for functions               | 236 | CheckIn                          |     |
| fields for templates               | 242 | IcodeWarriorProjectFile          | 659 |
| function IDs                       | 240 | IcodeWarriorVersionControl       | 804 |
| matching functions and methods     | 240 | CheckOut                         |     |
| storing                            | 181 | IcodeWarriorProjectFile          | 659 |
| Browser Data Stream                | 231 | IcodeWarriorVersionControl       | 804 |
| browser information                |     | checkout states                  | 382 |
| file IDs                           | 145 | choosing a relative path         | 272 |
| browser symbol type names          | 176 | CleanupPopupItemList             |     |
| browser symbols, generating        |     | IcodeWarriorPopupMenuToolbarItem | 783 |
| determining enabled symbol types   | 141 | Clone                            |     |
| Build                              |     | IFileSpec                        | 599 |
| IcodeWarriorProject                | 630 | CloneTarget                      |     |
| IcodeWarriorTarget                 | 714 | IcodeWarriorProject              | 633 |
| build operations                   |     | Close                            |     |
| identifying and comparing          | 141 | IcodeWarriorDocument             | 573 |
| BuildAgainstSubProjectTarget       |     | IcodeWarriorProject              | 634 |
| IcodeWarriorTarget                 | 714 | CollapseGroup                    |     |
| BuildAndWaitToComplete             |     | IcodeWarriorProjectDocument      | 580 |
| IcodeWarriorProject                | 632 | Collection                       |     |
| IcodeWarriorTarget                 | 715 | Add                              | 489 |
| BuildAndWaitToCompleteWithOptions  | 632 | get__NewEnum                     | 490 |
| IcodeWarriorTarget                 | 715 | get_Count                        | 490 |
| BuildEnded                         |     | get_ReadOnly                     | 491 |
| IcodeWarriorProjectEvents          | 653 | Item                             | 491 |
| BuildPopupItemList                 |     | Remove                           | 492 |
| IcodeWarriorPopupMenuToolbarItem   | 782 | Collections API                  |     |
| BuildStarted                       |     | Overview                         | 487 |
| IcodeWarriorProjectEvents          | 654 | Reference                        | 488 |
| BuildWithOptions                   |     | COM routines                     | 40  |
| IcodeWarriorProject                | 631 | combo box                        |     |
| IcodeWarriorTarget                 | 716 | getting items                    | 284 |
|                                    |     | inserting list items             | 290 |
|                                    |     | removing items                   | 274 |
|                                    |     | setting items                    | 298 |
|                                    |     | command line arguments           | 187 |
|                                    |     | command line options             | 142 |

---

---

|                                                 |                                      |
|-------------------------------------------------|--------------------------------------|
| command line tools                              | CreateAccessPathByFileSpec           |
| dependencies 166                                | IcodeWarriorAccessPaths 430          |
| Command Status Data Type 505                    | CreateAccessPathByPosition           |
| Commands API                                    | IcodeWarriorAccessPaths 431          |
| Overview 493                                    | CreateAndAddFile                     |
| Reference 498                                   | IcodeWarriorCreateFileItem 528       |
| Using 494                                       | CreateDesign                         |
| CompareFiles                                    | IcodeWarriorProject 635              |
| IcodeWarriorCompare 479                         | CreateInExistingProject              |
| CompareFilesByFileSpec                          | IcodeWarriorCreateProjectItem 534    |
| IcodeWarriorCompare 480                         | CreateItemControl                    |
| CompareFolders                                  | IcodeWarriorCustomToolBarItem 779    |
| IcodeWarriorCompare 481                         | CreateNewCommandGroup                |
| CompileFiles                                    | IcodeWarriorCommandRegistry 502      |
| IcodeWarriorDesign 546                          | CreateNewProject                     |
| IcodeWarriorTarget 717                          | IcodeWarriorCreateProjectItem 535    |
| CompileFilesAndWaitToComplete                   | CreateObjectInDesign                 |
| IcodeWarriorDesign 546                          | IcodeWarriorCreateObjectItem 531     |
| IcodeWarriorTarget 718                          | CreateObjectInTargets                |
| CompileFilesWithChoice                          | IcodeWarriorCreateObjectItem 532     |
| IcodeWarriorProject 634                         | CreateProject                        |
| IcodeWarriorTarget 718                          | IcodeWarriorApp 447                  |
| compiling                                       | CreateProjectByFileSpec              |
| getting current file index 146                  | IcodeWarriorApp 449                  |
| getting current file location 147               | CreateSubMenu                        |
| getting file text 148                           | IcodeWarriorMenu 606                 |
| Components API                                  | CreateTarget                         |
| Overview 509                                    | IcodeWarriorProject 635              |
| Reference 509                                   | CreateTemporaryMenu                  |
| Connect                                         | IcodeWarriorMenuManager 612          |
| IcodeWarriorVersionControl 805                  | CreateToolBar                        |
| ContainsTarget                                  | IcodeWarriorWindow 822               |
| IcodeWarriorDesign 547                          | CreateUntitledFile                   |
| control handle                                  | IcodeWarriorCreateFileItem 529       |
| getting 307                                     | CreateUserTree                       |
| control titles                                  | IcodeWarriorApp 450                  |
| getting 281                                     | IcodeWarriorTarget 719               |
| control values                                  | currentPrefs 340                     |
| getting 283                                     | custom browser symbols 169           |
| Copy                                            | custom symbol type names 175         |
| IFileSpec 600                                   | CW_STRICT_DIALOGS 333, 340, 345, 370 |
| CPU, target                                     | cwAccessAbsolute 195                 |
| reporting 168                                   | cwAccessFileName 195                 |
| Creatable Item Category Constants Data Type 538 | cwAccessFileRelative 196             |
| Creatable Items Data Types 537                  | CWAccessPathInfo 89                  |
| CreateAccessPath                                | CWAccessPathListInfo 90              |
| IcodeWarriorAccessPaths 429                     | cwAccessPathRelative 195             |

---

---

|                           |          |                               |               |
|---------------------------|----------|-------------------------------|---------------|
| CWAccessPathType          | 92       | CWFileStateChanged            | 380           |
| CWAddProjectEntry         | 40       | CWFileTime                    | 100           |
| CWAddr64                  | 92       | cwFileTypePrecompiledHeader   | 122           |
| CWAlert                   | 41       | cwFileTypeText                | 123           |
| CWAllocateMemory          | 42       | cwFileTypeUnknown             | 123           |
| CWAllocMemHandle          | 43       | CWFindAndLoadFile             | 45, 74, 94    |
| CWAllowV1Compatibility    | 379      | CWFrameworkInfo               | 100           |
| CWBrowseOptions           | 173      | CWFreeMemHandle               | 47, 61, 66    |
| CWCachePrecompiledHeader  | 138, 158 | CWFreeMemory                  | 48            |
| CWCompilerBrSymbol        | 170, 174 | CWFreeObjectData              | 112, 140      |
| CWCompilerBrSymbolInfo    | 169, 175 | CWGetAccessPathInfo           | 49            |
| CWCompilerBrSymbolList    | 175      | CWGetAccessPathListInfo       | 50            |
| CWCompletionRatio         | 385      | CWGetAccessPathSubdirectory   | 51            |
| CWCreateNewTextDocument   | 44       | CWGetAPIVersion               | 52            |
| CWDataType                | 93       | CWGetArraySettingElement      | 259, 265, 268 |
| CWDependencyInfo          | 176      | CWGetArraySettingSize         | 260, 262, 268 |
| CWDependencyType          | 93       | CWGetBooleanValue             | 262           |
| CWDialog                  | 333      | CWGetBrowseOptions            | 140, 174      |
| CWDisplayLines            | 139      | CWGetBuildSequenceNumber      | 141           |
| CWDonePluginRequest       | 45, 84   | CWGetCallbackOSError          | 53            |
| CWDoVisualDifference      | 384      | CWGetCOMApplicationInterface  | 53            |
| CWDROPINCOMPILETYPE       | 120      | CWGetCOMDesignInterface       | 54            |
| CWDROPINLINKERTYPE        | 120      | CWGetCommandLineArgs          | 142, 188      |
| CWDROPINPREFSTYPE         | 121      | CWGetCommandStatus            | 390           |
| CWDROPINPREFSTYPE_1       | 121      | CWGetComment                  | 380           |
| CWDROPINVCSTYPE           | 122      | CWGetCOMProjectInterface      | 54            |
| cwErrCantSetAttribute     | 129      | CWGetCOMTargetInterface       | 55            |
| cwErrFileNotFound         | 130      | CWGetDatabaseConnectionInfo   | 389           |
| cwErrInvalidCallback      | 130      | CWGetEnvironmentVariable      | 143           |
| cwErrInvalidMPCallback    | 130      | CWGetEnvironmentVariableCount | 144           |
| cwErrInvalidParameter     | 131      | CWGetFileInfo                 | 55, 149       |
| cwErrObjectFileNotStored  | 152, 227 | CWGetFileText                 | 56, 74        |
| cwErrOSError              | 53, 131  | CWGetFloatingPointValue       | 263           |
| cwErrOutOfMemory          | 131      | CWGetIDEInfo                  | 59            |
| cwErrRequestFailed        | 131      | CWGetIntegerValue             | 264           |
| cwErrSilent               | 132      | CWGetMainFileID               | 145           |
| cwErrStringBufferOverflow | 132      | CWGetMainFileNumber           | 146           |
| cwErrUnknownFile          | 132      | CWGetMainFileSpec             | 147           |
| cwErrUnknownSegment       | 228      | CWGetMainFileText             | 147           |
| cwErrUserCanceled         | 133      | CWGetMemHandleSize            | 60            |
| CWExtensionMapping        | 177      | CWGetModifiedFiles            | 148           |
| CWExtMapList              | 178      | CWGetNamedPreferences         | 61            |
| CWFileInfo                | 95       | CWGetNamedSetting             | 264, 268      |
| CWFileName                | 97       | CWGetOutputFileDirectory      | 61            |
| CWFileSpec                | 98       | CWGetOverlay1FileInfo         | 62            |

---

---

|                               |                    |                              |          |
|-------------------------------|--------------------|------------------------------|----------|
| CWGetOverlay1GroupInfo        | 63                 | cwNormalDependency           | 47, 94   |
| CWGetOverlay1GroupsCount      | 64                 | CWObjectData                 | 180      |
| CWGetOverlay1Info             | 64                 | CWOSResult                   | 106      |
| CWGetPluginData               | 65, 94, 375        | CWOverlay1FileInfo           | 106      |
| CWGetPluginRequest            | 66, 84             | CWOverlay1GroupInfo          | 107      |
| CWGetPrecompiledHeaderSpec    | 149, 157           | CWOverlay1Info               | 107      |
| CWGetProjectFile              | 67                 | CWPanelActivateItem          | 270      |
| CWGetProjectFileCount         | 67                 | CWPanelAppendItems           | 270      |
| CWGetProjectFileSpecifier     | 386                | CWPanelChooseRelativePath    | 272      |
| CWGetRelativePathValue        | 266                | CWPanelDeleteListItem        | 271, 274 |
| CWGetResourceFile             | 150                | CWPanelEnableItem            | 274      |
| CWGetSegmentInfo              | 68                 | CWPanelGetCurrentPrefs       | 275      |
| CWGetStoredObjectFileSpec     | 151, 153           | CWPanelGetDebugFlag          | 276      |
| CWGetStringValue              | 266                | CWPanelGetDialogItemHit      | 277      |
| CWGetStructureSettingField    | 260, 265, 267      | CWPanelGetFactoryPrefs       | 278      |
| CWGetSuggestedObjectFileSpec  | 152                | CWPanelGetItemData           | 279      |
| CWGetTargetDataDirectory      | 68                 | CWPanelGetItemMaxLength      | 280      |
| CWGetTargetInfo               | 153                | CWPanelGetItemText           | 280      |
| CWGetTargetName               | 69                 | CWPanelGetItemTextHandle     | 282      |
| CWGetTargetStorage            | 154, 164           | CWPanelGetItemValue          | 283      |
| CWGetVCSItem                  | 391                | CWPanelGetListItemText       | 271, 284 |
| CWGetVCSItemCount             | 391                | CWPanelGetNumBaseDialogItems | 277, 285 |
| CWGetVCSPointerStorage        | 393                | CWPanelGetOriginalPrefs      | 285      |
| CWGetWorkingDirectory         | 156                | CWPanelGetPanelPrefs         | 287      |
| CWHelpInfo                    | 333                | CWPanelGetRelativePathString | 288      |
| CWIDEInfo                     | 101                | CWPanelInsertListItem        | 271, 290 |
| cwInterfaceDependency         | 47, 94, 182        | CWPanelInvalItem             | 291      |
| CWIsAdvancedRequest           | 386                | CWPanelSetFactoryFlag        | 279, 291 |
| CWIsAutoPrecompiling          | 157                | CWPanelSetItemData           | 292      |
| CWIsCachingPrecompiledHeaders | 139, 158           | CWPanelSetItemMaxLength      | 293      |
| CWIsCommandSupportedRequest   | 387                | CWPanelSetItemText           | 294      |
| CWIsGeneratingDebugInfo       | 158                | CWPanelSetItemTextHandle     | 295      |
| CWIsPrecompiling              | 159                | CWPanelSetItemValue          | 296      |
| CWIsPreprocessing             | 160                | CWPanelSetListItemText       | 271, 297 |
| CWIsRecursiveRequest          | 386                | CWPanelSetRecompileFlag      | 298      |
| CWLoadObjectData              | 112, 140, 153, 161 | CWPanelSetRelinkFlag         | 299      |
| CWLockMemHandle               | 61, 66, 70         | CWPanelSetReparsingFlag      | 299      |
| CWMacOSErrToCWResult          | 71                 | CWPanelSetResetPathsFlag     | 300      |
| CWMemHandle                   | 102                | CWPanelSetRevertFlag         | 301      |
| CWMessageRef                  | 102                | CWPanelShowItem              | 302      |
| CWNativeXWindowPart Data Type | 840                | CWPanelValidItem             | 303      |
| CWNewProjectEntryInfo         | 103                | CWPanelActivateItem          | 303      |
| CWNewTextDocumentInfo         | 105                | CWPanelAppendItems           | 303      |
| cwNoDependency                | 93                 | CWPanelChooseRelativePath    | 304      |
| cwNoErr                       | 133                | CWPanelDrawPanelBox          | 304      |

---

---

|                                     |                                                |
|-------------------------------------|------------------------------------------------|
| CWPanlDrawUserItemBox 305           | CWPanlWriteRelativePathAEDesc 317              |
| CWPanlEnableItem 305                | CWPanlWriteRelativePathSetting 318             |
| CWPanlGetArraySettingElement 305    | CWPanlWriteStringSetting 318                   |
| CWPanlGetArraySettingSize 306       | CWPlugin_GetDefaultMappingList Entry Point 167 |
| CWPanlGetBooleanValue 306           | CWPlugin_GetDisplayName Entry Point 86         |
| CWPanlGetFloatingPointValue 306     | CWPlugin_GetDropInFlags 336                    |
| CWPanlGetIntegerValue 306           | CWPlugin_GetDropInFlags Entry Point 85         |
| CWPanlGetItemControl 306            | CWPlugin_GetDropInName Entry Point 86          |
| CWPanlGetItemData 307               | CWPlugin_GetHelpInfo Entry Point 332           |
| CWPanlGetItemMaxLength 308          | CWPlugin_GetTargetList Entry Point 168         |
| CWPanlGetItemRect 308               | CWPluginContext 35, 108                        |
| CWPanlGetItemText 308               | CWPostDialog 72, 376                           |
| CWPanlGetItemTextHandle 309         | CWPostFileAction 72, 376                       |
| CWPanlGetItemValue 309              | CWPreDialog 73, 376                            |
| CWPanlGetMacPort 309                | CWPreFileAction 73, 376                        |
| CWPanlGetNamedSetting 310           | CWProjectFileInfo 109                          |
| CWPanlGetPanelPrefs 310             | CWProjectSegmentInfo 113                       |
| CWPanlGetRelativePathString 310     | CWPutResourceFile 162                          |
| CWPanlGetRelativePathValue 311      | CWReadBooleanSetting 319                       |
| CWPanlGetStringValue 311            | CWReadFloatingPointSetting 319                 |
| CWPanlGetStructureSettingField 311  | CWReadIntegerSetting 320                       |
| CWPanlInstallUserItem 311           | CWReadRelativePathSetting 321                  |
| CWPanlInvalItem 312                 | CWReadStringSetting 322                        |
| CWPanlReadBooleanSetting 312        | CWRelativePath 113                             |
| CWPanlReadFloatingPointSetting 312  | CWRelativePathFormat 115                       |
| CWPanlReadIntegerSetting 312        | CWRelativePathTypes 116                        |
| CWPanlReadRelativePathAEDesc 312    | CWReleaseFileText 47, 57, 74                   |
| CWPanlReadRelativePathSetting 313   | CWRemoveProperty 75                            |
| CWPanlReadStringSetting 313         | CWReportMessage 75, 375                        |
| CWPanlRemoveUserItem 314            | CWResizeMemHandle 77                           |
| CWPanlSetBooleanValue 314           | CWResolveRelativePath 77                       |
| CWPanlSetFloatingPointValue 314     | CWResult 117                                   |
| CWPanlSetIntegerValue 314           | CWSetBooleanValue 323                          |
| CWPanlSetItemData 315               | CWSetCommandDescription 388                    |
| CWPanlSetItemMaxLength 315          | CWSetCommandStatus 391                         |
| CWPanlSetItemText 315               | CWSetFloatingPointValue 324                    |
| CWPanlSetItemTextHandle 315         | CWSetIntegerValue 325                          |
| CWPanlSetItemValue 316              | CWSetModDate 78                                |
| CWPanlSetRelativePathValue 316      | CWSetPluginOSError 79                          |
| CWPanlSetStringValue 316            | CWSetRelativePathValue 326                     |
| CWPanlShowItem 316                  | CWSetStringValue 327                           |
| CWPanlValidItem 316                 | CWSetTargetInfo 154, 163                       |
| CWPanlWriteBooleanSetting 317       | CWSetTargetStorage 155, 164                    |
| CWPanlWriteFloatingPointSetting 317 | CWSettingID 334                                |
| CWPanlWriteIntegerSetting 317       | CWSetVCSItem 393                               |

---

CWSetVCSPointerStorage 394  
 CWShowStatus 80  
 CWStoreObjectData 94, 152, 153, 161, 165  
 CWStorePluginData 81, 94, 375  
 cwSystemPath 51, 92, 123  
 CWTargeInfo 183  
 CWTargeList 188  
 CWToolBarItemID Data Type 801  
 CWToolBarRegistryInfo Data Type 802  
 CWUnlockMemHandle 82  
 CWUnmangleInfo 190  
 CWUserBreak 83  
 cwUserPath 51, 92, 124  
 CWVCSStateChangd 381  
 CWWriteBooleanSetting 328  
 CWWriteFloatingPointSetting 329  
 CWWriteIntegerSetting 330  
 CWWriteRelativePathSetting 330  
 CWWriteStringSetting 331

## D

Data Type  
   Command Status 505  
   Creatable Item Category Constants 538  
   Creatable Items 537  
   CWNativeXWindowPart 840  
   CWToolBarIconRegistryInfo 802  
   Dialog Services 569  
   ECodeWarriorAccess 703  
   ECodeWarriorBuildOptions 662  
   ECodeWarriorCompileChoice 662  
   ECodeWarriorConvertOption 482  
   ECodeWarriorLinkFlags 557  
   ECodeWarriorProjectOption 483  
   ECodeWarriorRevertPanelOption 482  
   ECodeWarriorRunMode 662  
   ECodeWarriorSaveOption 483  
   ECodeWarriorShowSymbolLocation 704  
   ECodeWarriorTargetLinkOrderType 764  
   ECodeWarriorTargetOutputKind 764  
   ECodeWarriorVCSCKIDState 812  
   ECodeWarriorVCSDBState 813  
   ECodeWarriorVCSInteractionOption 484  
   ECodeWarriorWhichTargetOptions 765  
   ECreateProjectType 538  
   EMsgType 628  
   IcodeWarriorToolbar

CWToolBarItemID 801  
 Menu Commands 505  
 SPopupMenuToolBarItem 801  
 SRegisterCommandGroup 506  
 Data Types  
   Access Paths 440  
   Application 482  
   Designs 557  
   IcodeWarriorToolbar 801  
   Message 628  
   Project 662  
   Symbols 703  
   Targets 764  
   Windows 840  
 DataModelCreated  
   IcodeWarriorAppEvents 475  
 DataModelLoaded  
   IcodeWarriorAppEvents 476  
 DataType  
   EcodeWarriorVCSFileLockState 813  
 dataversion 336  
 DeactivateEvent  
   IcodeWarriorWindowEvents 834  
 debugging  
   determining if enabled 276  
 debugging information 181  
   determining if enabled 159  
 debugHelperIsRegKey 187  
 debugHelperName 187  
 debugOn 342  
 DeleteMenuItem  
   IcodeWarriorMenu 608  
 DeletingDesign  
   IcodeWarriorProjectEvents 655  
 dependencies  
   establishing 166  
   specifying 47  
 dependency information 177  
 dependency types 93  
 dependencyType 47  
 DesignClosing  
   IcodeWarriorDesignAttachment 553  
 DesignCreated  
   IcodeWarriorProjectEvents 655  
 DesignInitialized  
   IcodeWarriorDesignAttachment 553  
 Designs API

- Data Types 557
- Overview 541
- Reference 541
- DestroyCodeWarriorWindow
  - ICodeWarriorWindow 823
- dialog 340
- dialog controls
  - setting title as handle 295
- dialog items
  - cancelling redraw 303
  - framing 304
  - getting control from 307
  - getting item rectangle 308
  - getting port 309
  - number of base items 285
  - showing 302
- dialog manager, Mac OS 345
- Dialog Services API
  - Overview 559
  - Reference 563
  - Using 559
- Dialog Services Data Types 569
- DialogPtr 345
- dialogs
  - operating system, displaying 72, 73
- disabling panel items 275
- Disconnect
  - ICodeWarriorVersionControl 805
- display name
  - entry point 87
- displaying text 45
- disposing
  - handles 48
- disposing text 74
- DoCommand
  - ICodeWarriorApp 451
- Documents API
  - Overview 571
  - Reference 571
- DOS wildcard 273
- dragmouse 342
- dragrect 342
- dragref 342
- DrawItemRepresentation
  - ICodeWarriorCustomToolbarItem 780
- dropin name
  - entry point 86

- DROPINCOMPILERLINKERAPIVERSION 196
- DropInFlags 117
- dropinflags 335
- DROPINPANELAPIVERSION 346
- dropintype 335

## E

- EAccess 245
- earliestCompatibleAPIVersion 335
- EBrowserItem 246
- ECodeWarriorAccess Data Type 703
- ECodeWarriorBuildOptions Data Type 662
- ECodeWarriorCompileChoice Data Type 662
- ECodeWarriorConvertOption Data Type 482
- ECodeWarriorLinkFlags Data Type 557
- ECodeWarriorProjectOption
  - Data Type 483
- ECodeWarriorRevertPanelOption Data Type 482
- ECodeWarriorRunMode Data Type 662
- ECodeWarriorSaveOption
  - Data Type 483
- ECodeWarriorShowSymbolLocation Data
  - Type 704
- ECodeWarriorTargetLinkOrderType Data
  - Type 764
- ECodeWarriorTargetOutputKind Data Type 764
- ECodeWarriorVCCKIDState Data Type 812
- ECodeWarriorVCSDBState Data Type 813
- ECodeWarriorVCSFileLockState Data Type 813
- ECodeWarriorVCSInteractionOption
  - Data Type 484
- ECodeWarriorWhichTargetOptions Data
  - Type 765
- ECreateProjectType Data Type 538
- edit fields
  - determining maximum length 280
  - getting as handle 282
  - getting as numbers 283
  - getting text 281
  - setting 294
  - setting to a handle 295
  - settings maximum length 293
- edit text 283
  - setting value 297
- editor window
  - creating 45

- 
- ELanguage 247
  - EMember 248
  - EMsgType
    - Data Type 628
  - enabling panel items 275
  - entry point
    - browser symbol 169
    - display name 87
    - dropin name 86
    - file mappings 167
    - help information 332
    - plug-in
      - flags 85
    - symbol unmangling 171
    - target list 168
  - Entry points, plug-in 84
  - enumerating subdirectories 52
  - environment variables
    - counting 144
  - EPluginDataStorageLoc Data Type
    - Data Type
      - EPluginDataStorageLoc 663
  - error codes
    - converting Mac OS to IDE 71
    - returning to IDE 84
  - Error Info API
    - Overview 589
    - Reference 589
  - error result
    - returning 45
  - errors
    - displaying 76
  - errors, operating system
    - returning to IDE 80
  - ETemplateType 249
  - event 340
  - executable
    - linkage model 187
  - Execute
    - ICodeWarriorDeferredAction 504
  - ExecuteCommand
    - ICodeWarriorCommandHandler 499
  - Execution Mode
    - VCS 395
  - exelinkageFlat 196
  - exelinkageOverlay1 197
  - exelinkageSegmented 197
  - ExpandGroup
    - ICodeWarriorProjectDocument 581
  - Export
    - ICodeWarriorProject 636
  - ExportByFileSpec
    - ICodeWarriorProject 637
- ## F
- factoryPrefs 340
  - file IDs
    - in browser information 145
  - file information
    - obtaining 56
  - file mapping entry point 167
  - file mappings 178, 179
    - when added to project 167
  - File Mappings panel 167
  - file specification
    - for project file 67
    - getting 56
  - file types
    - associating with plug-ins 167
  - filterList 273
  - FindAndAddFile
    - ICodeWarriorDesign 547
    - ICodeWarriorTarget 720
  - FindAndAddFile2
    - ICodeWarriorDesign 548
    - ICodeWarriorTarget 722
  - FindAndAddFile2ByCollection
    - ICodeWarriorTarget 723
  - FindAndAddFileByCollection
    - ICodeWarriorTarget 724
  - FindClass
    - ICodeWarriorSymbolContainer 695
  - FindClassInFile
    - ICodeWarriorSymbolContainer 695
  - FindDataMemberByName
    - ICodeWarriorClass 668
  - FindDesign
    - ICodeWarriorProject 637
  - FindDesignForDataModel
    - ICodeWarriorApp 451
  - FindFileByName
    - ICodeWarriorProject 638
  - finding files 46
  - FindLogicalFolder
-

- 
- ICodeWarriorApp 452
  - FindMethodByName
    - ICodeWarriorClass 669
  - FindTarget
    - ICodeWarriorProject 638
  - Folder Names, Standard 484
  - forcing
    - panel item redraw 291
  - four-character code 93
  - framing panel items 304
  - freeing
    - handles 48
  - freeing text 74
  - G**
  - Get
    - ICodeWarriorVersionControl 805
  - get\_\_NewEnum
    - Collection 490
  - get\_Access
    - ICodeWarriorBaseClassInfo 666
    - ICodeWarriorMethod 677
  - get\_AccessPathLocation
    - ICodeWarriorAccessPath 424
  - get\_AccessPaths
    - ICodeWarriorTarget 724
  - get\_AccessPathType
    - ICodeWarriorAccessPath 425
  - get\_Action
    - ICodeWarriorErrorInfo 590
  - get\_ActiveDocument
    - ICodeWarriorApp 453
    - ICodeWarriorDocument 573
  - get\_AlwaysSearchUserPaths
    - ICodeWarriorAccessPaths 433
  - get\_Application
    - ICodeWarriorApp 453
    - ICodeWarriorProject 639
  - get\_BaseClass
    - ICodeWarriorBaseClassInfo 666
  - get\_BaseClasses
    - ICodeWarriorClass 669
  - get\_BrowserDB
    - ICodeWarriorDesign 549
    - ICodeWarriorTarget 725
  - get\_BrowserEnabled
    - ICodeWarriorTarget 725
  - get\_BuildAgainst
    - ICodeWarriorSubProjectTarget 755
  - get\_CanHaveMultipleEventSets
    - ICodeWarriorComponent 510
  - get\_CKIDState
    - ICodeWarriorVCSSState 809
  - get\_Class
    - ICodeWarriorComponent 511
    - ICodeWarriorComponentEventSet 517
    - ICodeWarriorSymbol 690
  - get\_ClassList
    - ICodeWarriorSymbolContainer 698
  - get\_CompareInterface
    - ICodeWarriorApp 454
  - get\_Container
    - ICodeWarriorSourceContext 690
    - ICodeWarriorSymbol 691
  - get\_Count
    - Collection 490
  - get\_CreatableItems
    - ICodeWarriorApp 454
  - get\_DataModel
    - ICodeWarriorDesign 549
  - get\_DBState
    - ICodeWarriorVCSSState 810
  - get\_Debugger
    - ICodeWarriorApp 454
  - get\_DebugInfo
    - ICodeWarriorTargetFile 743
  - get\_DeclarationLocation
    - ICodeWarriorSymbol 691
  - get\_DefaultEvent
    - ICodeWarriorComponent 511
  - get\_DefaultProject
    - ICodeWarriorApp 455
  - get\_DefaultProjectDocument
    - ICodeWarriorApp 455
  - get\_DefinitionCount
    - ICodeWarriorBuildMessages 616
  - get\_DefinitionLocation
    - ICodeWarriorSymbol 692
  - get\_Definitions
    - ICodeWarriorBuildMessages 617
  - get\_Dependencies
    - ICodeWarriorTargetFile 744
  - get\_Dependents
    - ICodeWarriorTargetFile 744
-

---

|                                   |                                  |
|-----------------------------------|----------------------------------|
| get_Design                        | get_HRESULT                      |
| IcodeWarriorTarget 726            | IcodeWarriorErrorInfo 591        |
| get_Designs                       | get_InformationCount             |
| IcodeWarriorProject 640           | IcodeWarriorBuildMessages 618    |
| get_Dirty                         | get_Informations                 |
| IcodeWarriorDocument 573          | IcodeWarriorBuildMessages 619    |
| get_Documents                     | get_InitBefore                   |
| IcodeWarriorApp 456               | IcodeWarriorTargetFile 745       |
| get_DWORDErr                      | get_IsAbstract                   |
| IcodeWarriorErrorInfo 591         | IcodeWarriorClass 671            |
| get_EndOffset                     | IcodeWarriorMethod 678           |
| IcodeWarriorSourceContext 686     | get_IsConst                      |
| get_ErrorCount                    | IcodeWarriorMethod 678           |
| IcodeWarriorBuildMessages 617     | get_IsConstructor                |
| get_ErrorNumber                   | IcodeWarriorMethod 679           |
| IcodeWarriorMessage 621           | get_IsDefined                    |
| get_Errors                        | IcodeWarriorSourceContext 687    |
| IcodeWarriorBuildMessages 618     | get_IsDestructor                 |
| get_EventConnectionsEnabled       | IcodeWarriorMethod 679           |
| IcodeWarriorComponent 512         | get_IsFinal                      |
| get_Events                        | IcodeWarriorClass 671            |
| IcodeWarriorComponentProperty 517 | IcodeWarriorDataMember 675       |
| get_EventSet                      | get_IsInline                     |
| GetDefaultMethodName 515          | IcodeWarriorMethod 680           |
| get_EventSetName                  | get_IsNative                     |
| IcodeWarriorComponentProperty 518 | IcodeWarriorMethod 680           |
| get_EventSets                     | get_IsPublic                     |
| IcodeWarriorComponent 512         | IcodeWarriorClass 672            |
| get_FileLockState                 | get_IsStatic                     |
| IcodeWarriorVCSSState 810         | IcodeWarriorMethod 681           |
| get_FileSpec                      | get_IsSynchronized               |
| IcodeWarriorDocument 574          | IcodeWarriorMethod 681           |
| IcodeWarriorMessage 622           | get_IsTransient                  |
| IcodeWarriorProject 640           | IcodeWarriorDataMember 676       |
| IcodeWarriorProjectFile 660       | get_IsVirtual                    |
| IcodeWarriorSourceContext 687     | IcodeWarriorBaseClassInfo 667    |
| IcodeWarriorTargetFile 745        | IcodeWarriorMethod 682           |
| IcodeWarriorTargetOutput 751      | get_IsVisible                    |
| get_FullName                      | IcodeWarriorProject 641          |
| IcodeWarriorApp 456               | get_KeyName                      |
| get_FullPath                      | IcodeWarriorUserTree 436         |
| IFileSpec 600                     | get_LineCount                    |
| get_Getter                        | IcodeWarriorTextEngine 769       |
| IcodeWarriorComponentProperty 519 | get_LinkAgainst                  |
| get_HasSelection                  | IcodeWarriorSubProjectTarget 756 |
| IcodeWarriorTextEngine 768        | get_LinkAgainstOutput            |
| get_Height                        | IcodeWarriorSubTarget 753        |
| IcodeWarriorDocument 574          |                                  |

---

---

|                                    |                                   |
|------------------------------------|-----------------------------------|
| Get_LinkOrderType                  | ICodeWarriorApp 458               |
| ICodeWarriorTarget 727             | get_Properties                    |
| get_MacOSErr                       | ICodeWarriorComponent 513         |
| ICodeWarriorErrorInfo 592          | get_ReadOnly                      |
| get_MergeLibrary                   | Collection 491                    |
| ICodeWarriorTargetFile 746         | ICodeWarriorDocument 575          |
| get_MessageLength                  | get_Reason                        |
| ICodeWarriorMessage 622            | ICodeWarriorErrorInfo 592         |
| get_MessageLineCount               | get_Recursive                     |
| ICodeWarriorMessage 623            | ICodeWarriorAccessPath 426        |
| get_MessageText                    | get_SelectionEnd                  |
| ICodeWarriorMessage 623            | ICodeWarriorTextEngine 770        |
| get_Method                         | get_SelectionLineEnd              |
| GetDefaultMethodName 515           | ICodeWarriorTextEngine 771        |
| get_Methods                        | get_SelectionLineStart            |
| ICodeWarriorComponent 513          | ICodeWarriorTextEngine 771        |
| get_MWErr                          | get_SelectionStart                |
| ICodeWarriorErrorInfo 591          | ICodeWarriorTextEngine 772        |
| get_Name                           | get_SelectionText                 |
| GetDefaultMethodName 516           | ICodeWarriorTextEngine 772        |
| ICodeWarriorApp 457                | get_Setter                        |
| ICodeWarriorComponentProperty 520  | ICodeWarriorComponentProperty 520 |
| ICodeWarriorDesign 550             | get_SimpleName                    |
| ICodeWarriorDocument 575           | ICodeWarriorSymbol 692            |
| ICodeWarriorProject 642            | get_SourceLength                  |
| ICodeWarriorProjectFile 660        | ICodeWarriorMessage 624           |
| ICodeWarriorSubProjectTarget 756   | get_SourceLineNumber              |
| ICodeWarriorSymbol 692             | ICodeWarriorMessage 625           |
| ICodeWarriorTarget 727             | get_SourceOffset                  |
| ICodeWarriorTargetFile 746         | ICodeWarriorMessage 625           |
| ICodeWarriorUserTree 437           | get_StartOffset                   |
| ICodeWarriorVersionControl 806     | ICodeWarriorSourceContext 688     |
| IFileSpec 600                      | get_SubClasses                    |
| get_OutputKind                     | ICodeWarriorClass 674             |
| ICodeWarriorTargetOutput 752       | get_SubDirectories                |
| get_Path                           | ICodeWarriorAccessPath 426        |
| ICodeWarriorAccessPath 425         | get_SubTargets                    |
| get_Project                        | ICodeWarriorTarget 732            |
| ICodeWarriorDesign 550             | get_SystemAccessPaths             |
| ICodeWarriorProjectAssociation 651 | ICodeWarriorAccessPaths 433       |
| ICodeWarriorProjectDocument 581    | get_Target                        |
| ICodeWarriorProjectFile 660        | ICodeWarriorMessage 626           |
| ICodeWarriorTarget 728             | ICodeWarriorSubTarget 754         |
| get_ProjectFile                    | ICodeWarriorSymbolContainer 700   |
| ICodeWarriorMessage 624            | ICodeWarriorTargetFile 746        |
| get_ProjectFileCollection          | get_TargetFileCollection          |
| ICodeWarriorTarget 729             | ICodeWarriorTarget 733            |
| get_Projects                       |                                   |

---

---

|                                   |                                         |
|-----------------------------------|-----------------------------------------|
| get_Targets                       | ICodeWarriorCommandHandler 500          |
| ICodeWarriorDesign 551            | GetContainingDocument                   |
| ICodeWarriorProject 643           | ICodeWarriorToolBar 789                 |
| ICodeWarriorProjectFile 661       | GetCreatedProjectType                   |
| get_TextEngine                    | ICodeWarriorCreateProjectItem 535       |
| ICodeWarriorTextDocument 585      | GetCurrentTarget                        |
| get_TextLength                    | ICodeWarriorProject 639                 |
| ICodeWarriorTextEngine 773        | GetDataMembers                          |
| get_TokenLength                   | ICodeWarriorClass 670                   |
| ICodeWarriorMessage 626           | GetDataMembersWithAccess                |
| get_TokenOffset                   | ICodeWarriorClass 670                   |
| ICodeWarriorMessage 627           | GetDefaultMethodName                    |
| get_Type                          | get_EventSet 515                        |
| ICodeWarriorComponentProperty 521 | get_Method 515                          |
| ICodeWarriorMessage 627           | get_Name 516                            |
| ICodeWarriorUserTree 437          | GetDefaultMethodName 514                |
| get_UserAccessPaths               | getDefaultMethodName 514                |
| ICodeWarriorAccessPaths 434       | GetDefaultToolBarItems                  |
| get_UserTree                      | ICodeWarriorWindowEvents 834            |
| ICodeWarriorAccessPath 427        | GetDisplayName                          |
| get_UserTrees                     | ICodeWarriorCreatableItem 525           |
| ICodeWarriorApp 459               | GetHelpString                           |
| ICodeWarriorTarget 734            | ICodeWarriorToolBarItemHelp 798         |
| get_Value                         | GetIcon                                 |
| ICodeWarriorUserTree 437          | ICodeWarriorCreatableItem 525           |
| get_VCSState                      | GetIDEMainWindow                        |
| ICodeWarriorProjectFile 661       | ICodeWarriorWindowManager 820           |
| get_VersionControl                | GetInitialState                         |
| ICodeWarriorApp 459               | ICodeWarriorPopupMenuToolBarItem 784    |
| ICodeWarriorProject 643           | ICodeWarriorToggleButtonToolBarItem 787 |
| get_Visible                       | GetItemRepresentationWidth              |
| ICodeWarriorApp 460               | ICodeWarriorCustomToolBarItem 781       |
| ICodeWarriorDocument 575          | GetItemSizeInfo                         |
| get_WarningCount                  | ICodeWarriorCustomToolBarItem 781       |
| ICodeWarriorBuildMessages 619     | GetItemWidth                            |
| get_WeakImport                    | ICodeWarriorPopupMenuToolBarItem 785    |
| ICodeWarriorTargetFile 747        | GetLineForOffset                        |
| get_Width                         | ICodeWarriorTextEngine 769              |
| ICodeWarriorDocument 576          | GetLinkerName                           |
| get_XPos                          | ICodeWarriorTarget 727                  |
| ICodeWarriorDocument 576          | GetMenusEnabledState                    |
| get_YPos                          | ICodeWarriorMenuManager 613             |
| ICodeWarriorDocument 577          | GetMethods                              |
| GetCategory                       | ICodeWarriorClass 672                   |
| ICodeWarriorCreatableItem 524     | GetMethodsWithAccess                    |
| GetCodeWarriorWindowSizeLocation  | ICodeWarriorClass 673                   |
| ICodeWarriorWindow 823            | GetNamedPluginData                      |
| GetCommandStatus                  |                                         |

---

---

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>ICodeWarriorApp 457</li> <li>ICodeWarriorProject 642</li> <li>ICodeWarriorTarget 728</li> <li>GetNativeWindowReference <ul style="list-style-type: none"> <li>ICodeWarriorWindow 824</li> </ul> </li> <li>GetNativeXWindowReference <ul style="list-style-type: none"> <li>ICodeWarriorWindow 824</li> </ul> </li> <li>GetOffsetForLine <ul style="list-style-type: none"> <li>ICodeWarriorTextEngine 770</li> </ul> </li> <li>GetProjectFileFromFileSpec <ul style="list-style-type: none"> <li>ICodeWarriorTarget 729</li> </ul> </li> <li>GetSampleTextString <ul style="list-style-type: none"> <li>ICodeWarriorPopupMenuToolbarItem 785</li> </ul> </li> <li>GetSegmentByName <ul style="list-style-type: none"> <li>ICodeWarriorTarget 730</li> </ul> </li> <li>GetSegmentList <ul style="list-style-type: none"> <li>ICodeWarriorTarget 731</li> </ul> </li> <li>GetSegmentListAsBSTR <ul style="list-style-type: none"> <li>ICodeWarriorTarget 731</li> </ul> </li> <li>GetSetting <ul style="list-style-type: none"> <li>ICodeWarriorApp 458</li> </ul> </li> <li>GetSubProjects <ul style="list-style-type: none"> <li>ICodeWarriorTarget 732</li> </ul> </li> <li>GetTargetFileForProjectFile <ul style="list-style-type: none"> <li>ICodeWarriorTarget 733</li> </ul> </li> <li>GetTargetFiles <ul style="list-style-type: none"> <li>ICodeWarriorSegment 757</li> </ul> </li> <li>GetTargetOutput <ul style="list-style-type: none"> <li>ICodeWarriorTarget 734</li> </ul> </li> <li>GetTextForLineRange <ul style="list-style-type: none"> <li>ICodeWarriorTextEngine 773</li> </ul> </li> <li>GetTextForOffsetRange <ul style="list-style-type: none"> <li>ICodeWarriorTextEngine 774</li> </ul> </li> <li>getting started 419</li> <li>GetToolBarHeight <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 789</li> </ul> </li> <li>GetToolBarItemText <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 790</li> </ul> </li> <li>GetToolBarItemValue <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 790</li> </ul> </li> <li>GetVCSSState <ul style="list-style-type: none"> <li>ICodeWarriorVersionControl 806</li> </ul> </li> <li>GetWindowToolBar <ul style="list-style-type: none"> <li>ICodeWarriorWindow 825</li> </ul> </li> <li>global data <ul style="list-style-type: none"> <li>storing 164</li> </ul> </li> <li>GrafPort <ul style="list-style-type: none"> <li>getting 309</li> </ul> </li> </ul> | <h2>H</h2> <ul style="list-style-type: none"> <li>handle allocation 43</li> <li>handle size <ul style="list-style-type: none"> <li>determining 60</li> </ul> </li> <li>HandleMenuSelection <ul style="list-style-type: none"> <li>ICodeWarriorMenuHandler 610</li> </ul> </li> <li>HandlePopupSelection <ul style="list-style-type: none"> <li>ICodeWarriorPopupMenuToolbarItem 786</li> </ul> </li> <li>handles <ul style="list-style-type: none"> <li>dereferencing 70</li> <li>disposing 48</li> <li>lock count 70</li> <li>locking 70</li> <li>Mac OS 252</li> <li>unlocking 82</li> </ul> </li> <li>HasAttribute <ul style="list-style-type: none"> <li>ICodeWarriorWindow 825</li> </ul> </li> <li>hasobjectcode 112</li> <li>hasresources 112</li> <li>help information 334 <ul style="list-style-type: none"> <li>entry point 332</li> </ul> </li> <li>Helper_GetCompilerBrSymbols Entry Point 168</li> <li>Helper_Unmangle Entry Point 171</li> </ul> |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | <h2>I</h2> <ul style="list-style-type: none"> <li>ICodeWarriorAccessPath <ul style="list-style-type: none"> <li>get_AccessPathLocation 424</li> <li>get_AccessPathType 425</li> <li>get_Path 425</li> <li>get_Recursive 426</li> <li>get_SubDirectories 426</li> <li>get_UserTree 427</li> <li>put_AccessPathLocation 427</li> <li>put_Recursive 428</li> </ul> </li> <li>ICodeWarriorAccessPaths <ul style="list-style-type: none"> <li>ApplyChanges 429</li> <li>CreateAccessPath 429</li> <li>CreateAccessPathByFileSpec 430</li> <li>CreateAccessPathByPosition 431</li> <li>get_AlwaysSearchUserPaths 433</li> <li>get_SystemAccessPaths 433</li> <li>get_UserAccessPaths 434</li> <li>put_AlwaysSearchUserPaths 434</li> </ul> </li> <li>ICodeWarriorApp <ul style="list-style-type: none"> <li>AddCreatableItem 446</li> </ul> </li> </ul>                                                                                                              |

---

---

|                                     |     |                                       |          |
|-------------------------------------|-----|---------------------------------------|----------|
| AddUserTree                         | 446 | get_DefinitionCount                   | 616      |
| AttemptModify                       | 447 | get_Definitions                       | 617      |
| CreateProject                       | 447 | get_ErrorCount                        | 617      |
| CreateProjectByFileSpec             | 449 | get_Errors                            | 618      |
| CreateUserTree                      | 450 | get_InformationCount                  | 618      |
| DoCommand                           | 451 | get_Information                       | 619      |
| FindDesignForDataModel              | 451 | get_WarningCount                      | 619      |
| FindLogicalFolder                   | 452 | ICodeWarriorBuildMessages Interface   | 616      |
| get_ActiveDocument                  | 453 | ICodeWarriorClass                     |          |
| get_Application                     | 453 | FindDataMemberByName                  | 668      |
| get_CompareInterface                | 454 | FindMethodByName                      | 669      |
| get_CreatableItems                  | 454 | get_BaseClasses                       | 669      |
| get_Debugger                        | 454 | get_IsAbstract                        | 671      |
| get_DefaultProject                  | 455 | get_IsFinal                           | 671      |
| get_DefaultProjectDocument          | 455 | get_IsPublic                          | 672      |
| get_Documents                       | 456 | get_SubClasses                        | 674      |
| get_FullName                        | 456 | GetDataMembers                        | 670      |
| get_Name                            | 457 | GetDataMembersWithAccess              | 670      |
| get_Projects                        | 458 | GetMethods                            | 672      |
| get_UserTrees                       | 459 | GetMethodsWithAccess                  | 673      |
| get_VersionControl                  | 459 | ICodeWarriorClass Interface           | 668      |
| get_Visible                         | 460 | ICodeWarriorCommandHandler            |          |
| GetNamedPluginData                  | 457 | GetCommandStatus                      | 500      |
| GetSetting                          | 458 | ICodeWarriorCommandHandler            | 499      |
| ImportProject                       | 460 | ICodeWarriorCommandHandler Interface  |          |
| ImportProjectByFileSpec             | 461 | Interface                             |          |
| IsBuildInProgress                   | 462 | ICodeWarriorCommandHandler            | 499      |
| OpenDocument                        | 462 | ICodeWarriorCommandRegistry           |          |
| OpenProject                         | 463 | CreateNewCommandGroup                 | 502      |
| OpenProjectByFileSpec               | 465 | RegisterExternalCommand               | 503, 505 |
| OpenProjectByFileSpecWithOptions    | 467 | ICodeWarriorCommandRegistry Interface |          |
| OpenProjectWithOptions              | 464 | Interface                             |          |
| OpenTextDocument                    | 469 | ICodeWarriorCommandRegistry           | 502      |
| OpenTextDocumentByFileSpec          | 469 | ICodeWarriorCompare                   |          |
| OpenUntitledTextDocument            | 470 | CompareFiles                          | 479      |
| put_AllowUserInteraction            | 471 | CompareFilesByFileSpec                | 480      |
| put_Visible                         | 471 | CompareFolders                        | 481      |
| QueueDeferredAction                 | 471 | ICodeWarriorCompare Interface         | 479      |
| RemoveCreatableItem                 | 472 | ICodeWarriorComponent                 |          |
| RemoveNamedPluginData               | 473 | get_CanHaveMultipleEventSets          | 510      |
| RemoveUserTree                      | 473 | get_Class                             | 511      |
| SetNamedPluginData                  | 474 | get_DefaultEvent                      | 511      |
| SetSetting                          | 474 | get_EventConnectionsEnabled           | 512      |
| ICodeWarriorBaseClassInfo           |     | get_EventSets                         | 512      |
| get_Access                          | 666 | get_Methods                           | 513      |
| get_BaseClass                       | 666 | get_Properties                        | 513      |
| get_IsVirtual                       | 667 | ICodeWarriorComponentEvent Interface  | 514      |
| ICodeWarriorBaseClassInfo Interface | 666 | ICodeWarriorComponentEventSet         |          |
| ICodeWarriorBuildMessages           |     |                                       |          |

---

---

|                                             |                                        |
|---------------------------------------------|----------------------------------------|
| get_Class 517                               | AddFile 543                            |
| ICodeWarriorComponentEventSet Interface 517 | AddFile2 543                           |
| ICodeWarriorComponentProperty               | AddFile2ByFileSpec 544                 |
| get_Events 517                              | AddFileByFileSpec 545                  |
| get_EventSetName 518                        | CompileFiles 546                       |
| get_Getter 519                              | CompileFilesAndWaitToComplete 546      |
| get_Name 520                                | ContainsTarget 547                     |
| get_Setter 520                              | FindAndAddFile 547                     |
| get_Type 521                                | FindAndAddFile2 548                    |
| ICodeWarriorComponentProperty Interface 519 | get_BrowserDB 549                      |
| ICodeWarriorCreatableItem                   | get_DataModel 549                      |
| GetCategory 524                             | get_Name 550                           |
| GetDisplayName 525                          | get_Project 550                        |
| GetIcon 525                                 | get_Targets 551                        |
| InvokesWizard 526                           | put_Name 551                           |
| ICodeWarriorCreateFileItem                  | RemoveAttachment 551                   |
| CanAddFileToProject 527                     | RemoveTargetFromDesign 552             |
| CanCreateUntitledFile 527                   | ICodeWarriorDesignAttachment           |
| CreateAndAddFile 528                        | DesignClosing 553                      |
| CreateUntitledFile 529                      | DesignInitialized 553                  |
| ICodeWarriorCreateFileItem Interface 527    | RemovingAttachment 554                 |
| ICodeWarriorCreateObjectItem                | ICodeWarriorDesignAttachment Interface |
| AreObjectsCreatedInDesign 530               | ICodeWarriorDesignAttachment 553       |
| CreateObjectInDesign 531                    | ICodeWarriorDesignEvents               |
| CreateObjectInTargets 532                   | RemovingTarget 555                     |
| NeedsObjectName 533                         | TargetAdded 556                        |
| ICodeWarriorCreateObjectItem Interface 530  | ICodeWarriorDesignEvents Interface 555 |
| ICodeWarriorCreateProjectItem               | ICodeWarriorDialogServices             |
| CreateInExistingProject 534                 | NewItemDialog 563                      |
| CreateNewProject 535                        | OKCancelDialog 564                     |
| GetCreatedProjectType 535                   | OKDialog 565                           |
| RequiresFileExtension 536                   | PostModalDialog 565                    |
| ICodeWarriorCreateProjectItem Interface 534 | PreModalDialog 565                     |
| ICodeWarriorCustomToolbarItem               | ReportErrorFromErrorInfo 566           |
| CreateItemControl 779                       | SaveDontSaveDialog 567                 |
| DrawItemRepresentation 780                  | SetPluginDialogCommandHandler 567      |
| GetItemRepresentationWidth 781              | UpdatePluginDialogMenus 568            |
| GetItemSizeInfo 781                         | ICodeWarriorDocument                   |
| ICodeWarriorCustomToolbarItem Interface 779 | Activate 572                           |
| ICodeWarriorDataMember                      | Close 573                              |
| get_IsFinal 675                             | get_ActiveDocument 573                 |
| get_IsTransient 676                         | get_Dirty 573                          |
| ICodeWarriorDataMember Interface 675        | get_FileSpec 574                       |
| ICodeWarriorDeferredAction                  | get_Height 574                         |
| Execute 504                                 | get_Name 575                           |
| ICodeWarriorDeferredAction Interface 504    | get_ReadOnly 575                       |
| ICodeWarriorDesign                          | get_Visible 575                        |
| AddAttachment 542                           | get_Width 576                          |

---

---

|                                       |                                       |
|---------------------------------------|---------------------------------------|
| get_XPos 576                          | ICodeWarriorMenuManager Interface 612 |
| get_YPos 577                          | ICodeWarriorMessage                   |
| put_Height 577                        | get_ErrorNumber 621                   |
| put_Visible 577                       | get_FileSpec 622                      |
| put_Width 578                         | get_MessageLength 622                 |
| put_XPos 578                          | get_MessageLineCount 623              |
| put_YPos 579                          | get_MessageText 623                   |
| Save 579                              | get_ProjectFile 624                   |
| ICodeWarriorDocument Interface 572    | get_SourceLength 624                  |
| ICodeWarriorErrorInfo                 | get_SourceLineNumber 625              |
| get_Action 590                        | get_SourceOffset 625                  |
| get_DWORDErr 591                      | get_Target 626                        |
| get_HRESULT 591                       | get_TokenLength 626                   |
| get_MacOSErr 592                      | get_TokenOffset 627                   |
| get_MWErr 591                         | get_Type 627                          |
| get_Reason 592                        | ICodeWarriorMessage Interface 621     |
| put_Action 593                        | ICodeWarriorMethod                    |
| put_DWORDErr 594                      | get_Access 677                        |
| put_HelpContext 594                   | get_IsAbstract 678                    |
| put_HelpFile 595                      | get_IsConst 678                       |
| put_HRESULT 594                       | get_IsConstructor 679                 |
| put_MacOSErr 596                      | get_IsDestructor 679                  |
| put_MWErr 595                         | get_IsInline 680                      |
| put_Reason 597                        | get_IsNative 680                      |
| put_Source 597                        | get_IsStatic 681                      |
| ICodeWarriorErrorInfo Interface 590   | get_IsSynchronized 681                |
| ICodeWarriorEvents                    | get_IsVirtual 682                     |
| DataModelCreated 475                  | ICodeWarriorMethod Interface 677      |
| DataModelLoaded 476                   | ICodeWarriorPopupMenuToolbarItem      |
| ProjectVisible 477                    | BuildPopupMenuList 782                |
| QueryQuit 477                         | CleanupPopupMenuList 783              |
| Quit 477                              | GetInitialState 784                   |
| Startup 478                           | GetItemWidth 785                      |
| ICodeWarriorMenu                      | GetSampleTextString 785               |
| AppendMenuCommand 606                 | HandlePopupMenuSelection 786          |
| CreateSubMenu 606                     | ICodeWarriorPopupMenuToolbarItem      |
| DeleteMenuItem 608                    | Interface 782                         |
| RemoveItem 607                        | ICodeWarriorProject                   |
| RenameMenuCommand 607                 | Build 630                             |
| SetItemName 608                       | BuildAndWaitToComplete 632            |
| ICodeWarriorMenuHandler               | BuildAndWaitToCompleteWithOptions 632 |
| HandleMenuSelection 610               | BuildWithOptions 631                  |
| UpdateMenuStatus 611                  | CloneTarget 633                       |
| ICodeWarriorMenuHandler Interface 610 | Close 634                             |
| ICodeWarriorMenuManager               | CompileFilesWithChoice 634            |
| CreateTemporaryMenu 612               | CreateDesign 635                      |
| GetMenusEnabledState 613              | CreateTarget 635                      |
| SetMenusEnabledState 613              | Export 636                            |
| ShowCommandGroupMenu 614              | ExportByFileSpec 637                  |

---

---

|                                              |                                            |
|----------------------------------------------|--------------------------------------------|
| FindDesign 637                               | ICodeWarriorProjectFile                    |
| FindFileByName 638                           | CheckIn 659                                |
| FindTarget 638                               | CheckOut 659                               |
| get_Application 639                          | get_FileSpec 660                           |
| get_Designs 640                              | get_Name 660                               |
| get_FileSpec 640                             | get_Project 660                            |
| get_IsVisible 641                            | get_Targets 661                            |
| get_Name 642                                 | get_VCSState 661                           |
| get_Targets 643                              | ICodeWarriorProjectFile Interface 659      |
| get_VersionControl 643                       | ICodeWarriorSegment                        |
| GetCurrentTarget 639                         | GetTargetFiles 757                         |
| GetNamedPluginData 642                       | Locked 758, 759                            |
| RemoveDesign 644                             | Name 758                                   |
| RemoveDesignByName 644                       | PreLoaded 759, 760                         |
| RemoveFile 645                               | Protected 760                              |
| RemoveNamedPluginData 645                    | Purgable 761                               |
| RemoveObjectCode 646                         | PutTargetFiles 761                         |
| RemoveObjectCodeWithOptions 646              | SystemHeap 762, 763                        |
| RemoveTarget 648                             | Target 763                                 |
| ReportMessage 649                            | ICodeWarriorSegment Interface 757          |
| SetCurrentTarget 649                         | ICodeWarriorSourceContext                  |
| SetNamedPluginData 650                       | get_Container 690                          |
| SynchronizeStatus 650                        | get_EndOffset 686                          |
| ICodeWarriorProject Interface 630            | get_FileSpec 687                           |
| ICodeWarriorProjectAssociation               | get_IsDefined 687                          |
| get_Project 651                              | get_StartOffset 688                        |
| put_Project 652                              | put_EndOffset 688                          |
| ICodeWarriorProjectAssociation Interface 651 | put_FileSpec 689                           |
| ICodeWarriorProjectDocument                  | put_StartOffset 689                        |
| CollapseGroup 580                            | ICodeWarriorSourceContext Interface 686    |
| ExpandGroup 581                              | ICodeWarriorSubProjectTarget               |
| get_Project 581                              | get_BuildAgainst 755                       |
| SelectedFiles 582                            | get_LinkAgainst 756                        |
| SelectFiles 582                              | get_Name 756                               |
| ICodeWarriorProjectDocument Interface 580    | ICodeWarriorSubProjectTarget Interface 755 |
| ICodeWarriorProjectEvents                    | ICodeWarriorSubTarget                      |
| BuildEnded 653                               | get_LinkAgainstOutput 753                  |
| BuildStarted 654                             | get_Target 754                             |
| DeletingDesign 655                           | ICodeWarriorSubTarget Interface 753        |
| DesignCreated 655                            | ICodeWarriorSymbol                         |
| ProjectClosing 656                           | get_Class 690                              |
| QueryAboutToBuild 656                        | get_Container 691                          |
| QueryDeleteDesign 657                        | get_DeclarationLocation 691                |
| QueryUIClose 657                             | get_DefinitionLocation 692                 |
| RevertCompleted 658                          | get_Name 692                               |
| VisibleChanged 658                           | get_SimpleName 692                         |
| ICodeWarriorProjectEvents Interface          | ICodeWarriorSymbol Interface 690           |
| Interface                                    | ICodeWarriorSymbolContainer                |
| ICodeWarriorProjectEvents 653                |                                            |

---

---

|                                           |                                              |
|-------------------------------------------|----------------------------------------------|
| AddComponentAttachment 694                | GetTargetFileForProjectFile 733              |
| FindClass 695                             | GetTargetOutput 734                          |
| FindClassInFile 695                       | LinkAgainstSubProjectTarget 735              |
| get_ClassList 698                         | LinkAgainstSubTarget 735                     |
| get_Target 700                            | put_BrowserEnabled 736                       |
| RemoveComponentAttachment 701             | put_Name 737                                 |
| ShowSymbolDeclaration 701                 | RemoveNamedPluginData 738                    |
| ShowSymbolDefinition 702                  | RemoveObjectCode 739                         |
| ICodeWarriorSymbolContainer Interface 694 | RemoveObjectCodeWithOptions 739              |
| ICodeWarriorTarget                        | RemoveSegment 740                            |
| AddFile 707                               | RemoveUserTree 740                           |
| AddFile2 708                              | SetNamedPluginData 741                       |
| AddFile2ByFileSpec 708                    | SetupDebugging 742                           |
| AddFile2ByFileSpecification 709           | SynchronizeStatus 742                        |
| AddFileByFileSpec 711                     | ICodeWarriorTarget Interface 706             |
| AddFileByFileSpecCollection 712           | ICodeWarriorTargetFile                       |
| AddSegment 712                            | get_DebugInfo 743                            |
| AddSubTarget 713                          | get_Dependencies 744                         |
| AddUserTree 714                           | get_Dependents 744                           |
| Build 714                                 | get_FileSpec 745                             |
| BuildAgainstSubProjectTarget 714          | get_InitBefore 745                           |
| BuildAndWaitToComplete 715                | get_MergeLibrary 746                         |
| BuildAndWaitToCompleteWithOptions 715     | get_Name 746                                 |
| BuildWithOptions 716                      | get_Target 746                               |
| CompileFiles 717                          | get_WeakImport 747                           |
| CompileFilesAndWaitToComplete 718         | put_DebugInfo 747                            |
| CompileFilesWithChoice 718                | put_InitBefore 749                           |
| CreateUserTree 719                        | put_MergeLibrary 749                         |
| FindAndAddFile 720                        | put_WeakImport 750                           |
| FindAndAddFile2 722                       | ICodeWarriorTargetFile Interface 743         |
| FindAndAddFileByCollection 724            | ICodeWarriorTargetFindandAddFile2ByCollectio |
| get_AccessPaths 724                       | n 723                                        |
| get_BrowserDB 725                         | ICodeWarriorTargetOutput                     |
| get_BrowserEnabled 725                    | get_FileSpec 751                             |
| get_Design 726                            | get_OutputKind 752                           |
| Get_LinkOrderType 727                     | ICodeWarriorTargetOutput Interface 751       |
| get_Name 727                              | ICodeWarriorTextDocument                     |
| get_Project 728                           | get_TextEngine 585                           |
| get_ProjectFileCollection 729             | SaveACopyAs 585                              |
| get_SubTargets 732                        | SaveACopyAsByFileSpec 586                    |
| get_TargetFileCollection 733              | SaveAs 586                                   |
| get_UserTrees 734                         | SaveAsByFileSpec 586                         |
| GetLinkerName 727                         | ScrollToSelection 587                        |
| GetNamedPluginData 728                    | ICodeWarriorTextDocument Interface 583       |
| GetProjectFileFromFileSpec 729            | ICodeWarriorTextEngine                       |
| GetSegmentByName 730                      | get_HasSelection 768                         |
| GetSegmentList 731                        | get_LineCount 769                            |
| GetSegmentListAsBSTR 731                  | get_SelectionEnd 770                         |
| GetSubProjects 732                        |                                              |

---

---

|                                                 |                                                |
|-------------------------------------------------|------------------------------------------------|
| get_SelectionLineEnd 771                        | get_Type 437                                   |
| get_SelectionLineStart 771                      | get_Value 437                                  |
| get_SelectionStart 772                          | put_KeyName 438                                |
| get_SelectionText 772                           | put_Name 438                                   |
| get_TextLength 773                              | put_Type 438                                   |
| GetLineForOffset 769                            | put_Value 439                                  |
| GetOffsetForLine 770                            | ICodeWarriorVCSFileStateListener Interface 811 |
| GetTextForLineRange 773                         | ICodeWarriorVCSSState                          |
| GetTextForOffsetRange 774                       | get_CKIDState 809                              |
| InsertText 774                                  | get_DBState 810                                |
| put_SelectionEnd 775                            | get_FileLockState 810                          |
| put_SelectionLineEnd 775                        | ICodeWarriorVCSSState Interface 809            |
| put_SelectionLineStart 775                      | ICodeWarriorVCSSStateListener                  |
| put_SelectionStart 776                          | StateChanged 811                               |
| put_SelectionText 776                           | ICodeWarriorVersionControl                     |
| ICodeWarriorTextEngine Interface 768            | CheckIn 804                                    |
| ICodeWarriorToggleButtonToolBarItem             | CheckOut 804                                   |
| GetInitialState 787                             | Connect 805                                    |
| StateChanged 788                                | Disconnect 805                                 |
| ICodeWarriorToggleButtonToolBarItem             | Get 805                                        |
| Interface 787                                   | get_Name 806                                   |
| ICodeWarriorToolBar                             | GetVCSSState 806                               |
| Data Types 801                                  | IsConnected 807                                |
| GetContainingDocument 789                       | UndoCheckOut 808                               |
| GetToolBarHeight 789                            | Unlock 807                                     |
| GetToolBarItemText 790                          | ICodeWarriorVersionControl Interface 804       |
| GetToolBarItemValue 790                         | ICodeWarriorWindow                             |
| IsToolBarVisible 791                            | AssociateWindowWithProject 822                 |
| ItemCreated 796                                 | CreateToolBar 822                              |
| ItemDestroyed 797                               | DestroyCodeWarriorWindow 823                   |
| ResetToolBarItem 791                            | GetCodeWarriorWindowSizeLocation 823           |
| SetToolBarItemEnabled 792                       | GetNativeWindowReference 824                   |
| SetToolBarItemIcon 792                          | GetNativeXWindowReference 824                  |
| SetToolBarItemText 793                          | GetWindowToolBar 825                           |
| SetToolBarItemValue 794                         | HasAttribute 825                               |
| ShowToolBar 794                                 | MoveCodeWarriorWindow 825                      |
| ICodeWarriorToolBar Interface 789               | PutBehind 826                                  |
| ICodeWarriorToolBarInstanceCreationNotification | ReorderCodeWarriorWindow 827                   |
| Interface 796                                   | SelectCodeWarriorWindow 831                    |
| ICodeWarriorToolBarItemHelp                     | SetBackBrushes 827                             |
| GetHelpString 798                               | SetCodeWarriorWindowInitialBounds 828          |
| ICodeWarriorToolBarItemHelp Interface 798       | SetCodeWarriorWindowMinMaxSize 829             |
| ICodeWarriorToolBarItemRegistry                 | SetCodeWarriorWindowTitle 829                  |
| RegisterToolBarIcons 799                        | SetDialogColors 830                            |
| RegisterToolBarItem 800                         | SetEventHandler 830                            |
| ICodeWarriorUserTree                            | SetMaximumSleepTime 830                        |
| get_KeyName 436                                 | ShowCodeWarriorWindow 831                      |
| get_Name 437                                    | ICodeWarriorWindow Interface 821               |

---

---

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>ICodeWarriorWindowEvents <ul style="list-style-type: none"> <li>ActivateEvent 833</li> <li>DeactivateEvent 834</li> <li>GetDefaultToolBarItems 834</li> <li>Idle 834, 835</li> <li>KeyDownEvent 835</li> <li>MouseDownEvent 836</li> <li>OkToClose 837</li> <li>PreBeginUpdate 837</li> <li>ToolBarSizeChange 837</li> <li>UpdateEvent 838</li> <li>WindowDestroyed 838</li> <li>WindowResizedBy 839</li> </ul> </li> <li>ICodeWarriorWindowManager <ul style="list-style-type: none"> <li>CenterWindow 817</li> <li>CreateCodeWarriorWindow 818</li> <li>GetIDEMainWindow 820</li> <li>IsIDEInMDIMode 819</li> </ul> </li> <li>ICodeWarriorWindowManager Interface 817</li> <li>ICodeWarriorApp Interface 445</li> <li>IDE <ul style="list-style-type: none"> <li>settings panel API version variable 339</li> </ul> </li> <li>Idle <ul style="list-style-type: none"> <li>ICodeWarriorWindowEvents 834, 835</li> </ul> </li> <li>IFileSpec <ul style="list-style-type: none"> <li>Clone 599</li> <li>Copy 600</li> <li>get_FullPath 600</li> <li>get_Name 600</li> <li>put_FullPath 601</li> <li>put_Name 601</li> </ul> </li> <li>IFileSpec Interface 599</li> <li>ImportProject <ul style="list-style-type: none"> <li>ICodeWarriorApp 460</li> </ul> </li> <li>ImportProjectByFileSpec <ul style="list-style-type: none"> <li>ICodeWarriorApp 461</li> </ul> </li> <li>InsertText <ul style="list-style-type: none"> <li>ICodeWarriorTextEngine 774</li> </ul> </li> <li>installing <ul style="list-style-type: none"> <li>CodeWarrior 419</li> </ul> </li> <li>Interface <ul style="list-style-type: none"> <li>ICodeWarriorAccessPath 424</li> <li>ICodeWarriorAccessPaths 429</li> <li>ICodeWarriorApp 445</li> <li>ICodeWarriorBaseClassInfo 666</li> <li>ICodeWarriorBuildMessages 616</li> <li>ICodeWarriorClass 668</li> <li>ICodeWarriorCompare 479</li> </ul> </li> </ul> | <ul style="list-style-type: none"> <li>ICodeWarriorComponentEvent 514</li> <li>ICodeWarriorComponentEventSet 517</li> <li>ICodeWarriorComponentProperty 519</li> <li>ICodeWarriorCreateFileItem 527</li> <li>ICodeWarriorCreateObjectItem 530</li> <li>ICodeWarriorCreateProjectItem 534</li> <li>ICodeWarriorCustomToolBarItem 779</li> <li>ICodeWarriorDataMember 675</li> <li>ICodeWarriorDeferredAction 504</li> <li>ICodeWarriorDesignEvents 555</li> <li>ICodeWarriorDocument 572</li> <li>ICodeWarriorErrorInfo 590</li> <li>ICodeWarriorMenuHandler 610</li> <li>ICodeWarriorMenuManager 612</li> <li>ICodeWarriorMessage 621</li> <li>ICodeWarriorMethod 677</li> <li>ICodeWarriorPopupMenuToolBarItem 782</li> <li>ICodeWarriorProject 630</li> <li>ICodeWarriorProjectAssociation 651</li> <li>ICodeWarriorProjectDocument 580</li> <li>ICodeWarriorProjectFile 659</li> <li>ICodeWarriorSegment 757</li> <li>ICodeWarriorSourceContext 686</li> <li>ICodeWarriorSubProjectTarget 755</li> <li>ICodeWarriorSubTarget 753</li> <li>ICodeWarriorSymbol 690</li> <li>ICodeWarriorSymbolContainer 694</li> <li>ICodeWarriorTarget 706</li> <li>ICodeWarriorTargetFile 743</li> <li>ICodeWarriorTargetOutput 751</li> <li>ICodeWarriorTextDocument 583</li> <li>ICodeWarriorTextEngine 768</li> <li>ICodeWarriorToggleButtonToolBarItem 787</li> <li>ICodeWarriorToolBar 789</li> <li>ICodeWarriorToolBarInstanceCreationNotification 796</li> <li>ICodeWarriorToolBarItemHelp 798</li> <li>ICodeWarriorUserTree 436</li> <li>ICodeWarriorVCSFileStateListener 811</li> <li>ICodeWarriorVCSState 809</li> <li>ICodeWarriorVersionControl 804</li> <li>ICodeWarriorWindow 821</li> <li>ICodeWarriorWindowManager 817</li> <li>IFileSpec 599</li> <li>interrupting long operations 83</li> <li>invalidating <ul style="list-style-type: none"> <li>panel item 291</li> </ul> </li> <li>InvokesWizard <ul style="list-style-type: none"> <li>ICodeWarriorCreatableItem 526</li> </ul> </li> </ul> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

---

- IsBuildInProgress
  - ICodeWarriorApp 462
- IsConnected
  - ICodeWarriorVersionControl 807
- isGenerated 40, 79
- IsIDEInMDIMode
  - ICodeWarriorWindowManager 819
- isPermanent 42
- isPostLinker 198
- isPreLinker 198
- isresourcefile 112
- IsToolbarVisible
  - ICodeWarriorToolbar 791
- Item
  - Collection 491
- ItemCreated
  - ICodeWarriorToolbar 796
- ItemDestroyed
  - ICodeWarriorToolbar 797
- itemHit 341
- iterating access paths 50

## K

- kBadPrefVersion 371
- kCandisassemble 198
- kCanimport 198
- kCanprecompile 199
- kCanpreprocess 199
- kCompAllowDupFileNames 199
- kCompAlwaysReload 200
- kCompEmitsOwnBrSymbols 169, 200
- kCompMultiTargAware 201
- kCompRequiresFileListBuildStartedMsg 201
- kCompRequiresProjectBuildStartedMsg 201
- kCompRequiresSubProjectBuildStartedMsg 202
- kCompRequiresTargetBuildStartedMsg 202
- kCompUsesTargetStorage 155, 165, 203
- kCurrentCompiledFile 47, 95, 124
- kCurrentCWHelpInfoVersion 346
- kDefaultLinkPosition 124
- KeyDownEvent
  - ICodeWarriorWindowEvents 835
- kGeneratescode 203
- kGeneratessrcs 203
- kIgnored 203
- kInvalidCallbackErr 372

- kIsMPAware 204
- kIspascal 204
- kLaunchable 204
- kMissingPrefErr 371
- kPersistent 204
- kPrecompile 205
- kRsrcfile 205
- kSettingNotFoundErr 371
- kSettingOutOfRangeErr 372
- kSettingTypeMismatchErr 372
- kTargetGlobalPluginData 125

## L

- latest API version
  - determining 60
- launching
  - target executable 143
- linkAgainstFile 188
- LinkAgainstSubProjectTarget
  - ICodeWarriorTarget 735
- LinkAgainstSubTarget
  - ICodeWarriorTarget 735
- linkage models 187
- linkAllowDupFileNames 205
- linkAlwaysReload 205
- linkerGetTargetInfoThreadSafe 209
- linkerUnmangles 171, 210
- linkerUsesTargetStorage 155, 165, 211
- linkerWantsPreRunRequest 211
- linkMultiTargAware 206
- linkOutputDirectory 207
- linkOutputFile 206
- linkOutputNone 206
- linkRequiresFileListBuildStartedMsg 207
- linkRequiresProjectBuildStartedMsg 207
- linkRequiresSubProjectBuildStartedMsg 208
- linkRequiresTargetBuildStartedMsg 208
- linkRequiresTargetLinkStartedMsg 208
- loading
  - file text 57
- loading files
  - headers and interfaces 46
- loading settings data 61
- localized type names 169
- location
  - of output directory 62

lock count  
     handles 70  
 Locked  
     IcodeWarriorSegment 758, 759

## M

Mac OS  
     handles 252  
 Mac OS dialog manager 345  
 Mac OS errors  
     converting to IDE errors 71  
 magicCapLinker 211  
 main entry point  
     result 84  
 Main Plugin Entry Point 84  
 main() 45  
 Menu Commands Data Type 505  
 menu\_Clear 346  
 menu\_Copy 347  
 menu\_Cut 347  
 menu\_Paste 348  
 menu\_SelectAll 348  
 Menus API  
     Overview 603  
     Reference 604  
     Using 603  
 message  
     displaying 41  
 Message Data Types 628  
 messages  
     displaying 76  
 Messages API  
     Overview 615  
     Reference 615  
 messageTypeError 126  
 messageTypeInfo 125  
 messageTypeWarning 125  
 modification date 56  
     notifying IDE of change 79  
 modified files  
     determining 149  
 MouseDownEvent  
     IcodeWarriorWindowEvents 836  
 MoveCodeWarriorWindow  
     IcodeWarriorWindow 825  
 moveHi 70

## N

Name  
     IcodeWarriorSegment 758  
 naming conventions  
     panel API 252  
 NeedsObjectName  
     IcodeWarriorCreateObjectItem 533  
 Negotiation Mode  
     VCS 395  
 newestAPIVersion 335  
 NewItemDialog  
     IcodeWarriorDialogServices 563  
 notifying  
     IDE of file changes 79  
 numeric values  
     getting 283

## O

object code  
     storing 181  
 object code, PPC  
     alternate entry point hunks 907  
     array type 922  
     cross-reference hunks 908  
     data header 915  
     data hunks 905  
     data section 900  
     enumerated type 924  
     function data section 916  
     header 898  
     name table section 929  
     Object Pascal hunks 912  
     Pascal array type 925  
     Pascal enumerated type 928  
     Pascal set type 927  
     Pascal string type 929  
     Pascal subrange type 926  
     PEF import hunks 909  
     pointer type 921  
     regular hunks 902  
     reserved hunks 914  
     simple hunks 902  
     source file specification hunks 911  
     structure 898  
     structured type 923  
     type data section 919  
 object data

- format of 161
- freeing 140
- loading 161
- storing 152, 166
- object files
  - recommended location 152
- object files, external
  - determining location 151
- obtaining version 52
- OKCancelDialog
  - ICodeWarriorDialogServices 564
- OKDialog
  - ICodeWarriorDialogServices 565
- OkToClose
  - ICodeWarriorWindowEvents 837
- oldtargetfile 342
- OpenDocument
  - ICodeWarriorApp 462
- OPENFILENAME 273
- OpenProject
  - ICodeWarriorApp 463
- OpenProjectByFileSpec
  - ICodeWarriorApp 465
- OpenProjectByFileSpecWithOptions
  - ICodeWarriorApp 467
- OpenProjectWithOptions
  - ICodeWarriorApp 464
- OpenTextDocument
  - ICodeWarriorApp 469
- OpenTextDocumentByFileSpec
  - ICodeWarriorApp 469
- OpenUntitledTextDocument
  - ICodeWarriorApp 470
- operating system error
  - obtaining 53
- operating systems
  - target 168
- originalPrefs 340
- output directory
  - determining location 62
- overlay file
  - getting position 63
- overlay groups
  - counting 64
  - getting information 63
- overlays
  - getting information 65

- Overview
  - Collections API 487
  - Commands API 493
  - Components API 509
  - Designs API 541
  - Dialog Services API 559
  - Documents API 571
  - Error Info API 589
  - Menus API 603
  - Messages API 615
  - Projects API 629
  - Toolbar API 777
  - Windows API 815

## P

- panel information 336
- panel items
  - activating 270
  - appending 271
  - cancelling redraw 303
  - creating 271
  - determining item hit 277
  - enabling and disabling 275
  - framing 304
  - getting control from 307
  - getting item rectangle 308
  - getting port 309
  - redrawing 291
  - showing 302
- panel settings
  - current 275
- panelfamily 335
- PanelFlags 335
- PanelParamBlkPtr 336
- PanelParameterBlock 336
- panelscope 336
- panelScopeGlobal 348
- panelScopeProject 349
- panelScopeTarget 349
- path reset flag
  - setting 301
- Paths, Access 423
  - API Reference 423
  - ICodeWarriorAccessPath Interface 424
  - ICodeWarriorAccessPaths Interface 429
  - ICodeWarriorUserTree Interface 436
- permanent memory allocation 43
- plug-in

---

|                                |                               |
|--------------------------------|-------------------------------|
| capabilities, reporting 85     | Overview 629                  |
| enabling 168                   | Reference 629                 |
| plug-in API 52                 | ProjectVisible                |
| plug-in context 35, 252        | ICodeWarriorAppEvents 477     |
| plug-in data                   | ProjectOpened 476             |
| loading 65                     | Protected                     |
| Plug-in entry points 84        | ICodeWarriorSegment 760       |
| pointer memory                 | Purgable                      |
| allocating 42                  | ICodeWarriorSegment 761       |
| popup menu 283                 | put_AccessPathLocation        |
| setting value 296              | ICodeWarriorAccessPath 427    |
| PostModalDialog                | put_Action                    |
| ICodeWarriorDialogServices 565 | ICodeWarriorErrorInfo 593     |
| PreBeginUpdate                 | put_AllowUserInteraction      |
| ICodeWarriorWindowEvents 837   | ICodeWarriorApp 471           |
| Precompile command 160         | put_AlwaysSearchUserPaths     |
| precompiled header 138         | ICodeWarriorAccessPaths 434   |
| caching 138                    | put_BrowserEnabled            |
| save location 150              | ICodeWarriorTarget 736        |
| precompiled header caching     | put_DebugInfo                 |
| determining if enabled 158     | ICodeWarriorTargetFile 747    |
| verifying enabled 139          | put_DWORDErr                  |
| precompiling                   | ICodeWarriorErrorInfo 594     |
| automatic 157                  | put_EndOffst                  |
| determining if enabled 160     | ICodeWarriorSourceContext 688 |
| prefsDesc 342                  | put_FileSpec                  |
| prefsKeyword 341               | ICodeWarriorSourceContext 689 |
| PreLoaded                      | put_FullPath                  |
| ICodeWarriorSegment 759, 760   | IFileSpec 601                 |
| PreModalDialog                 | put_Height                    |
| ICodeWarriorDialogServices 565 | ICodeWarriorDocument 577      |
| preprocessing                  | put_HelpContext               |
| determining if enabled 160     | ICodeWarriorErrorInfo 594     |
| progress window 139            | put_HelpFile                  |
| project                        | ICodeWarriorErrorInfo 595     |
| counting files in 68           | put_HRESULT                   |
| file specification to 340      | ICodeWarriorErrorInfo 594     |
| Project Data Types 662         | put_InitBefore                |
| project file                   | ICodeWarriorTargetFile 749    |
| getting file specification 67  | put_KeyName                   |
| project files                  | ICodeWarriorUserTree 438      |
| counting 68                    | put_MacOSErr                  |
| ProjectClosing                 | ICodeWarriorErrorInfo 596     |
| ICodeWarriorProjectEvents 656  | put_MergeLibrary              |
| ProjectOpened                  | ICodeWarriorTargetFile 749    |
| ProjectVisible 476             | put_MWErr                     |
| Projects API                   | ICodeWarriorErrorInfo 595     |

---

---

- put\_Name
  - ICodeWarriorDesign 551
  - ICodeWarriorTarget 737
  - ICodeWarriorUserTree 438
  - IFileSpec 601
- put\_Project
  - ICodeWarriorProjectAssociation 652
- put\_Reason
  - ICodeWarriorErrorInfo 597
- put\_Recursive
  - ICodeWarriorAccessPath 428
- put\_SelectionEnd
  - ICodeWarriorTextEngine 775
- put\_SelectionLineEnd
  - ICodeWarriorTextEngine 775
- put\_SelectionLineStart
  - ICodeWarriorTextEngine 775
- put\_SelectionStart
  - ICodeWarriorTextEngine 776
- put\_SelectionText
  - ICodeWarriorTextEngine 776
- put\_Source
  - ICodeWarriorErrorInfo 597
- put\_StartOffset
  - ICodeWarriorSourceContext 689
- put\_Type
  - ICodeWarriorUserTree 438
- put\_Value
  - ICodeWarriorUserTree 439
- put\_Visible
  - ICodeWarriorApp 471
  - ICodeWarriorDocument 577
- put\_WeakImport
  - ICodeWarriorTargetFile 750
- put\_Width
  - ICodeWarriorDocument 578
- put\_XPos
  - ICodeWarriorDocument 578
- put\_YPos
  - ICodeWarriorDocument 579
- PutBehind
  - ICodeWarriorWindow 826
- PutTargetFiles
  - ICodeWarriorSegment 761

## Q

QueryAboutToBuild

- ICodeWarriorProjectEvents 656
- QueryDeleteDesign
  - ICodeWarriorProjectEvents 657
- QueryQuit
  - ICodeWarriorAppEvents 477
- QueryUIClose
  - ICodeWarriorProjectEvents 657
- QueueDeferredAction
  - ICodeWarriorApp 471
- QuickStart 419
- Quit
  - ICodeWarriorAppEvents 477

## R

- radio button 283
  - setting value 296
- recompile 341
- recompile flag
  - setting 298
- redrawing
  - panel item 291
- RegisterExternalCommand
  - ICodeWarriorCommandRegistry 503, 505
- RegisterToolBarIcons
  - ICodeWarriorToolBarItemRegistry 799
- RegisterToolBarItem
  - ICodeWarriorToolBarItemRegistry 800
- registry 187
- relative path
  - choosing 272
  - converting to human-readable string 289
  - creating Apple Event from 318
  - extracting from Apple Event 313
  - resolving to full path 78
- relink 341
- relink flag
  - setting 299
- Remove
  - Collection 492
- RemoveAttachment
  - ICodeWarriorDesign 551
- RemoveComponentAttachment
  - ICodeWarriorSymbolContainer 701
- RemoveCreatableItem
  - ICodeWarriorApp 472
- RemoveDesign
  - ICodeWarriorProject 644

---

|                                  |                             |
|----------------------------------|-----------------------------|
| RemoveDesignByName               | reqCheckSyntax 211          |
| IcodeWarriorProject 644          | reqCompDisassemble 212      |
| RemoveFile                       | reqCompile 157, 211         |
| IcodeWarriorProject 645          | reqDatabaseConnect 399      |
| RemoveItem                       | reqDatabaseDisconnect 400   |
| IcodeWarriorMenu 607             | reqDatabaseVariables 401    |
| RemoveNamedPluginData            | reqDeactivateItem 352       |
| IcodeWarriorApp 473              | reqDisassemble 213          |
| IcodeWarriorProject 645          | reqDragDrop 352             |
| IcodeWarriorTarget 738           | reqDragEnter 353            |
| RemoveObjectCode                 | reqDragExit 354             |
| IcodeWarriorProject 646          | reqDragWithin 355           |
| IcodeWarriorTarget 739           | reqDrawCustomItem 356       |
| RemoveObjectCodeWithOptions      | reqFileAdd 402              |
| IcodeWarriorProject 646          | reqFileBranch 403           |
| IcodeWarriorTarget 739           | reqFileCheckin 403          |
| RemoveSegment                    | reqFileCheckout 403         |
| IcodeWarriorTarget 740           | reqFileComment 404          |
| RemoveTarget                     | reqFileDelete 404           |
| IcodeWarriorProject 648          | reqFileDestroy 404          |
| RemoveTargetFromDesign           | reqFileDifference 405       |
| IcodeWarriorDesign 552           | reqFileGet 405              |
| RemoveUserTree                   | reqFileHistory 405          |
| IcodeWarriorApp 473              | reqFileLabel 406            |
| IcodeWarriorTarget 740           | reqFileListBuildEnded 213   |
| removing                         | reqFileListBuildStarted 214 |
| combo box items 274              | reqFileProperties 406       |
| RemovingAttachment               | reqFilePurge 406            |
| IcodeWarriorDesignAttachment 554 | reqFileRename 407           |
| RemovingTarget                   | reqFileRollback 407         |
| IcodeWarriorDesignEvents 555     | reqFileShare 407            |
| RenameMenuCommand                | reqFileStatus 408           |
| IcodeWarriorMenu 607             | reqFileUndoCheckout 408     |
| ReorderCodeWarriorWindow         | reqFileVersion 408          |
| IcodeWarriorWindow 827           | reqFileView 409             |
| reparse 342                      | reqFilter 356               |
| reparse flag                     | reqFindStatus 357           |
| setting 300                      | reqFirstLoad 357            |
| ReportErrorFromErrorInfo         | reqGetData 358              |
| IcodeWarriorDialogServices 566   | reqGetFactory 358           |
| ReportMessage                    | reqHandleClick 359          |
| IcodeWarriorProject 649          | reqHandleKey 360            |
| reqAbout 126, 400                | reqIdle 127, 401            |
| reqActivateItem 350              | reqInitialize 398           |
| reqAEGetPref 350                 | reqInitDialog 271, 298, 360 |
| reqAESetPref 351                 |                             |
| reqByteSwapData 351, 369         |                             |

---

---

- reqInitialize 127
- reqInitPanel 361
- reqItemHit 277, 279, 294, 361
- reqLink 214
- reqObeyCommand 363
- reqPrefsChange 128, 402
- reqPreRun 215
- reqProjectBuildEnded 215
- reqProjectBuildStarted 215
- reqPutData 294, 363
- reqReadSettings 260, 268, 364
- reqRenameProject 365
- reqSetupDebug 365
- reqSubProjectBuildEnded 216
- reqSubProjectBuildStarted 216
- reqTargetBuildEnded 217
- reqTargetBuildStarted 217
- reqTargetInfo 154, 163, 218
- reqTargetLinkEnded 219
- reqTargetLinkStarted 219
- reqTargetLoaded 43, 165, 219
- reqTargetPrefsChanged 220
- reqTargetUnloaded 43, 165, 220
- reqTermDialog 366
- reqTerminate 128, 399
- reqTermPanel 366
- request 339
  - determining 66, 84
- RequiresFileExtension
  - ICodeWarriorCreateProjectItem 536
- reqUpdatePref 367
- reqValidate 367
- reqWriteSettings 260, 268, 368
- rescanning for project files 301
- reset 341
- reset paths flag
  - setting 301
- ResetToolbarItem
  - ICodeWarriorToolbar 791
- resolving a relative path 78
- resource attributes 68
- resource data
  - storing 181
- resource file
  - saving 163
- resource files

- selecting 151
- returning errors 45
- revert flag
  - setting 301
- RevertCompleted
  - ICodeWarriorProjectEvents 658
- run helper 187
  - executing 187
- runHelperIsRegKey 187
- runHelperName 187

## S

- Save
  - ICodeWarriorDocument 579
- SaveACopyAs
  - ICodeWarriorTextDocument 585
- SaveACopyAsByFileSpec
  - ICodeWarriorTextDocument 586
- SaveAs
  - ICodeWarriorTextDocument 586
- SaveAsByFileSpec
  - ICodeWarriorTextDocument 586
- SaveDontSaveDialog
  - ICodeWarriorDialogServices 567
- ScrollToSelection 587
- segment information
  - getting 68
- SelectCodeWarriorWindow
  - ICodeWarriorWindow 831
- SelectedFiles
  - ICodeWarriorProjectDocument 582
- SelectFiles
  - ICodeWarriorProjectDocument 582
- selecting files 151
- SetBackBrushes
  - ICodeWarriorWindow 827
- SetCodeWarriorWindowInitialBounds
  - ICodeWarriorWindow 828
- SetCodeWarriorWindowMinMaxSize
  - ICodeWarriorWindow 829
- SetCodeWarriorWindowTitle
  - ICodeWarriorWindow 829
- SetCurrentTarget
  - ICodeWarriorProject 649
- SetDialogColors
  - ICodeWarriorWindow 830
- SetEventHandler

---

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |  |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| <ul style="list-style-type: none"> <li>ICodeWarriorWindow 830</li> <li>SetItemName <ul style="list-style-type: none"> <li>ICodeWarriorMenu 608</li> </ul> </li> <li>SetMaximumSleepTime <ul style="list-style-type: none"> <li>ICodeWarriorWindow 830</li> </ul> </li> <li>SetMenusEnabledState <ul style="list-style-type: none"> <li>ICodeWarriorMenuManager 613</li> </ul> </li> <li>SetNamedPluginData <ul style="list-style-type: none"> <li>ICodeWarriorApp 474</li> <li>ICodeWarriorProject 650</li> <li>ICodeWarriorTarget 741</li> </ul> </li> <li>SetPluginDialogCommandHandler <ul style="list-style-type: none"> <li>ICodeWarriorDialogServices 567</li> </ul> </li> <li>SetSetting <ul style="list-style-type: none"> <li>ICodeWarriorApp 474</li> </ul> </li> <li>settings <ul style="list-style-type: none"> <li>XML 334</li> </ul> </li> <li>settings data <ul style="list-style-type: none"> <li>current 275</li> <li>factory defaults 278</li> <li>loading 61</li> <li>loading for a different panel 287</li> <li>original 286</li> </ul> </li> <li>settings panel <ul style="list-style-type: none"> <li>data structures 333</li> <li>request variable 339</li> </ul> </li> <li>settings panel API version 339</li> <li>settings panels <ul style="list-style-type: none"> <li>activating items 270</li> <li>appending items 271</li> <li>base items 285</li> <li>cancelling item redraw 303</li> <li>determining item hit 277</li> <li>factory default settings 278</li> <li>framing items 304</li> <li>getting control from item 307</li> <li>getting item rectangle 308</li> <li>getting port 309</li> <li>redrawing items 291</li> <li>setting text items 294</li> <li>showing items 302</li> </ul> </li> <li>SetToolBarItemEnabled <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 792</li> </ul> </li> <li>SetToolBarItemIcon <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 792</li> </ul> </li> <li>SetToolBarItemText <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 793</li> </ul> </li> <li>SetToolBarItemValue <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 794</li> </ul> </li> <li>SetupDebugging <ul style="list-style-type: none"> <li>ICodeWarriorTarget 742</li> </ul> </li> <li>SFTypeList 273</li> <li>ShowCodeWarriorWindow <ul style="list-style-type: none"> <li>ICodeWarriorWindow 831</li> </ul> </li> <li>ShowCommandGroupMenu <ul style="list-style-type: none"> <li>ICodeWarriorMenuManager 614</li> </ul> </li> <li>ShowSymbolDeclaration <ul style="list-style-type: none"> <li>ICodeWarriorSymbolContainer 701</li> </ul> </li> <li>ShowSymbolDefinition <ul style="list-style-type: none"> <li>ICodeWarriorSymbolContainer 702</li> </ul> </li> <li>ShowToolBar <ul style="list-style-type: none"> <li>ICodeWarriorToolBar 794</li> </ul> </li> <li>sizes <ul style="list-style-type: none"> <li>of object code and data 182</li> </ul> </li> <li>specifying dependencies 47</li> <li>SPopupMenuToolBarItem Data Type 801</li> <li>SRegisterCommandGroup Data Type 506</li> <li>Standard Folder Names 484</li> <li>Startup <ul style="list-style-type: none"> <li>ICodeWarriorAppEvents 478</li> </ul> </li> <li>StateChanged <ul style="list-style-type: none"> <li>ICodeWarriorToggleButtonToolBarItem 788</li> <li>ICodeWarriorVCSStateListener 811</li> </ul> </li> <li>static text 283 <ul style="list-style-type: none"> <li>getting 281</li> <li>setting 294</li> <li>setting to a handle 295</li> <li>setting value 297</li> </ul> </li> <li>storage 339</li> <li>storing object data 166</li> <li>supportsByteSwapping 369</li> <li>supportsTextSettings 369</li> <li>supressload 46</li> <li>symbol unmangling 155, 191</li> <li>symbolics file <ul style="list-style-type: none"> <li>location 187</li> </ul> </li> <li>Symbols Data Types 703</li> <li>symName 169</li> <li>symUIName 170</li> <li>SynchronizeStatus <ul style="list-style-type: none"> <li>ICodeWarriorProject 650</li> <li>ICodeWarriorTarget 742</li> </ul> </li> <li>SystemHeap <ul style="list-style-type: none"> <li>ICodeWarriorSegment 762, 763</li> </ul> </li> </ul> |  |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|

---

## T

- Target
  - ICodeWarriorSegment 763
- target executable
  - launching 143
- target information
  - getting 154
  - setting 163
- target name
  - getting 70
- target storage 155
  - and unmangling 171
  - lifetime of 164
  - managing 155, 164
- TargetAdded
  - ICodeWarriorDesignEvents 556
- targetCPU68K 221
- targetCPUAny 222
- targetCPUEmbeddedPowerPC 222
- targetCPUi80x86 222
- targetCPUMips 223
- targetCPUNECv800 223
- targetCPUPowerPC 224
- targetfile 340
- targetOSAny 224
- targetOSEmbeddedABI 225
- targetOSMacintosh 225
- targetOSMagicCap 225
- targetOSOS9 226
- targetOSWindows 226
- Targets Data Types 764
- temporary files
  - storing 69
- text
  - disposing 74
  - freeing 74
  - getting 282
  - validation 281
- Text API
  - Overview
    - Overview
    - Text API 767
  - Reference 767
- text files
  - loading 57
- toEndian 342, 351, 357

## Toolbar API

- Overview 777
- Reference 777

## ToolbarSizeChange

- ICodeWarriorWindowEvents 837

## touched files

- determining 149

## type code 93

## U

### UndoCheckOut

- ICodeWarriorVersionControl 808

### Unlock

- ICodeWarriorVersionControl 807

### unlocking a handle 82

### unmangling 165, 171, 191

- storing associated data 155

### UpdateEvent

- ICodeWarriorWindowEvents 838

### UpdateMenuStatus 611

### UpdatePluginDialogMenus

- ICodeWarriorDialogServices 568

### UpdatePort

- ICodeWarriorWindow/ICodeWarriorWindow

- UpdatePort 832

### URL syntax 187

### user break 83

### usesStrictAPI 340, 346, 370

### useTempMemory 43

### Using

- Collections API

- Collections API

- Using 487

- Commands API 494

- Dialog Services API 559

- Menus API 603

- Windows API 815

## V

### VCS 373

### version 339

### Version Control API Reference 803

### version control system 373

### version of plug-in API

- determining 52

- determining latest 60

VisibleChanged  
 ICodeWarriorProjectEvents 658

## W

warnings  
   displaying 76  
 wildcard, DOS 273  
 window  
   creating 45  
 WindowDestroyed  
   ICodeWarriorWindowEvents 838  
 WindowResizedBy  
   ICodeWarriorWindowEvents 839  
 windows  
   operating system, displaying 72, 73  
 Windows API  
   Overview 815  
   Reference 816  
   Using 815  
 Windows Data Types 840  
 WinHelp 332, 334  
 working directory  
   getting 156

writing structured relative path values 327  
 writing structured string values 328  
 writing top-level boolean values 329  
 writing top-level floating point values 329  
 writing top-level integer values 330  
 writing top-level relative path values 331  
 writing top-level string values 332

## X

XCOFF 897  
 XML  
   arrays, number of elements 262  
   reading an array or structure setting 265  
   reading array elements 260  
   reading nested structures and arrays 260  
   reading structure fields 268  
   reading structured boolean values 263  
   reading structured floating point values 263  
   reading structured integer values 264  
   reading structured relative path values 266  
   reading structured string values 267  
   reading top-level boolean values 319  
   reading top-level floating point values 320  
   reading top-level integer values 321  
   reading top-level relative path values 322  
   reading top-level string values 322  
   setting ID 334  
   writing nested structures and arrays 261, 269  
   writing structured boolean values 323  
   writing structured floating point values 324  
   writing structured integer values 325



**Freescale Semiconductor, Inc.**

