PowerPlant X 1.0 API Reference Manual

Thu Aug 14 16:41:03 2003

Contents

Chapter 1

PowerPlant X 1.0 API Reference Namespace Index

1.1 PowerPlant X 1.0 API Reference Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

PowerPlant X 1.0 API Reference Hierarchical Index

2.1 PowerPlant X 1.0 API Reference Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- PPx::AutoValueSaver< T > 232
- PPx::CFObject< TCFRef > 293
  - PPx::CFMutableObject< TCFRef, TMutableRef > 289
- PPx::CFObject< CFArrayRef > 293
  - PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef > 289
    - PPx::CFArray< TValue > 251
- PPx::CFObject< CFBundleRef > 293
  - PPx::CFBundle 263
- PPx::CFObject< CFDataRef > 293
  - PPx::CFMutableObject< CFDataRef, CFMutableDataRef > 289
    - PPx::CFData 273
- PPx::CFObject< CFDictionaryRef > 293
  - PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef > 289
    - PPx::CFDictionary< TKey, TValue > 279
- PPx::CFObject< CFStringRef > 293
  - PPx::CFMutableObject< CFStringRef, CFMutableStringRef > 289
    - PPx::CFString 301
- PPx::CFObject< CFTreeRef > 293
  - PPx::CFTree 318
    - PPx::CFXMLTree 346
- PPx::CFObject< CFURLRef > 293
  - PPx::CFURL 326
- PPx::CFObject< CFXMLNodeRef > 293
  - PPx::CFXMLNode 341
- PPx::CFXMLElement 338
- PPx::CGContextSaver 351
- PPx::CommandIDType< TCommandID > 375
- PPx::ControlPartCodeStruct 406
- PPx::DataReader 423
- PPx::DataScrap 428
- PPx::DataWriter 432
- PPx::XMLEncoder::EncoderInfo 456
- PPx::EventDoer 457
  - PPx::EventDoerAttachment 460
    - PPx::MessageAttachment 578
    - PPx::ResponseAttachment 647
  - PPx::EventDoerCallback< T > 462
  - PPx::SpecificEventDoer< TEventClass, TEventKind > 687
  - PPx::SpecificEventDoer< eventClass_ProcessCommand, TCommandID > 687
    - PPx::SpecificCommandDoer< TCommandID > 685
      - PPx::CommandHandler< TCommandID > 374
      - PPx::SpecificMenuCommandDoer< TCommandID > 688

- PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusNextDocumentWindow > 687
  - PPx::AppFocusNextDocumentWindowDoer 188
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusNextFloatingWindow > 687
  - PPx::AppFocusNextFloatingWindowDoer 189
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusToolbar > 687
  - PPx::AppFocusToolbarDoer 190
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppFrontSwitched > 687
  - PPx::AppFrontSwitchedDoer 191
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppGetDockTileMenu > 687
  - PPx::AppGetDockTileMenuDoer 192
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppHidden > 687
  - PPx::AppHiddenDoer 193
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppLaunched > 687
  - PPx::AppLaunchedDoer 194
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppLaunchNotification > 687
  - PPx::AppLaunchNotificationDoer 195
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppQuit > 687
  - PPx::AppQuitDoer 202
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppShown > 687
  - PPx::AppShownDoer 203
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppSystemUIModeChanged > 687
  - PPx::AppSystemUIModeChangedDoer 204
- PPx::SpecificEventDoer< kEventClassApplication, kEventAppTerminated > 687
  - PPx::AppTerminatedDoer 205
- PPx::SpecificEventDoer< kEventClassCommand, kEventCommandProcess > 687
  - PPx::CommandProcessDoer 376
    - PPx::CommandConverter 372
    - PPx::SheetAlert 674
- PPx::SpecificEventDoer< kEventClassCommand, kEventCommandUpdateStatus > 687
  - PPx::CommandUpdateStatusDoer 380
    - PPx::CommandConverter 372
- PPx::SpecificEventDoer< kEventClassControl, kEventControlActivate > 687
  - PPx::ControlActivateDoer 381

PPx::VolumeUnmountedDoer 803

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowActivated > 687

PPx::WindowActivatedDoer 811

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowBoundsChanged > 687

PPx::WindowBoundsChangedDoer 813

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowBoundsChanging > 687

PPx::WindowBoundsChangingDoer 814

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowClose > 687

PPx::WindowCloseDoer 818

PPx::Window 804

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCloseAll > 687

PPx::WindowCloseAllDoer 816

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowClosed > 687

PPx::WindowClosedDoer 817

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapse > 687

PPx::WindowCollapseDoer 821

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapseAll > 687

PPx::WindowCollapseAllDoer 819

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapsed > 687

PPx::WindowCollapsedDoer 820

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapsing > 687

PPx::WindowCollapsingDoer 822

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowContextualMenuSelect > 687

PPx::WindowContextualMenuSelectDoer 826

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCursorChange > 687

PPx::WindowCursorChangeDoer 827

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDeactivated > 687

PPx::WindowDeactivatedDoer 828

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDispose > 687

PPx::WindowDisposeDoer 830

PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDragCompleted > 687

PPx::WindowDragCompletedDoer 831

- PPx::FSObject . . . 492
- PPx::FSVolumeRefNumStruct . . . 510
- PPx::GrafPortSaver . . . 511
- PPx::HIObjectRefType< TType > . . . 520
- PPx::HIToolbarItemRefStruct . . . 521
- PPx::HIToolbarRefStruct . . . 522
- PPx::Identifiable . . . 534
  - PPx::Attachment . . . 210
    - PPx::CommandTask . . . 377
      - PPx::StatusCommandTask . . . 694
    - PPx::TargetAttachment . . . 754
      - PPx::EventDoerAttachment . . . 460
  - PPx::View . . . 784
- PPx::IdleTimer . . . 537
  - PPx::IdleTimerCallback< T > . . . 540
- PPx::IntegerType< TType, TValueType, defaultValue > . . . 551
- PPx::MenuCommandStruct . . . 563
- PPx::MenuEventOptionsStruct . . . 569
- PPx::MenuItemIndexStruct . . . 570
- PPx::MenuTrackingModeStruct . . . 577
- PPx::NavEventResponder . . . 589
- PPx::ObjectDescriptor . . . 591
- PPx::ObjectIDStruct
- PPx::ObjectStorageIDStruct
- PPx::OSStatusStruct . . . 598
- PPx::OSTypeStruct . . . 599
- PPx::OwnedPointer< T > . . . 600
- PPx::Persistent . . . 603
  - PPx::Attachment . . . 210
  - PPx::EventTarget . . . 464
    - PPx::ApplicationEventTarget . . . 201
      - PPx::Application . . . 199
    - PPx::Correspondent . . . 414
    - PPx::FrontWindowEventTarget . . . 490
    - PPx::UserFocusEventTarget . . . 783
    - PPx::View . . . 784
    - PPx::Window . . . 804
  - PPx::FrameAdapter . . . 489
    - PPx::BindingsFrameAdapter . . . 249
- PPx::Retained . . . 649
  - PPx::DataObject . . . 422
    - PPx::TDataObject< TData > . . . 756
    - PPx::TDataVector< TData > . . . 757
- PPx::ScrapPromiseKeeper . . . 657

Chapter 3

PowerPlant X 1.0 API Reference Compound Index

3.1 PowerPlant X 1.0 API Reference Compound List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

PowerPlant X 1.0 API Reference File Index

4.1 PowerPlant X 1.0 API Reference File List

Here is a list of all documented files with brief descriptions:

Chapter 5

PowerPlant X 1.0 API Reference Namespace Documentation

5.1 PPx Namespace Reference

5.1.1 Detailed Description

PowerPlantX.

Compounds

- class AccessibleGetAllActionNamesDoer

  Returns names of all supported actions.

- class AccessibleGetAllAttributeNamesDoer

  Returns names of all supported attributes.

- class AccessibleGetChildAtPointDoer

  Returns child object hit by a specified global mouse point.

- class AccessibleGetFocusedChildDoer

  Returns child which is part of the focus chain.

- class AccessibleGetNamedActionDescriptionDoer

  Returns a description of an action's significance.

- class AccessibleGetNamedAttributeDoer

Returns the value of an attribute.

- class AccessibleIsNamedAttributeSettableDoer

  Returns whether an attriubute is settable.

- class AccessiblePerformNamedActionDoer

  Performs an action.

- class AccessibleSetNamedAttributeDoer

  Sets the value of an attribute.

- class AEOpenDocumentsDoer

  Handles request to open a list of documents.

- class AEPrintDocumentsDoer

  Handles request to print a list of documents.

- class AEQuitApplicationDoer

  Handles request to quit the application.

- class AEReopenApplicationDoer

  Handles notification that an already running application has been reactivated from the Finder.

- class AERunApplicationDoer

  Handles notification the application was launched directly and not from opening a document.

- class AppActivatedDoer

  Handles notification that an application has resumed.

- class AppDeactivatedDoer

  Handles notification that an application has suspended.

- class AppearanceScrollBarVariantChangedDoer

  Notification that the scroll bar variant has changed.

- class AppFocusMenuBarDoer

  Handles request to set the keyboard focus to the menu bar.

- class AppFocusNextDocumentWindowDoer

  Handles request to set the keyboard focus to the next document window.

- class AppFocusNextFloatingWindowDoer

  Handles request to set the keyboard focus to the next floating window.

- class AppFocusToolbarDoer

  Handles request to set the keyboard focus to the toolbar in the currently focused window.

- class AppFrontSwitchedDoer

  Handles notification that the active application has changed.

- class AppGetDockTileMenuDoer

  Returns the menu to display from an application's dock tile.

- class AppHiddenDoer

  Handles notification that an application has been hidden.

- class AppLaunchedDoer

  Handles notification that another application has launched.

- class AppLaunchNotificationDoer

  Handles notification that an application we launched asynchronously has actually launched.

- class AppleEventDoer

  Abstract class for an Apple Event handler.

- class Application

  An executable program.

- class ApplicationEventTarget

  The top-level Carbon Event target.

- class AppQuitDoer

  Handles a request to quit an application.

- class AppShownDoer

  Handles notification that an application has been shown.

- class AppSystemUIModeChangedDoer

  Handles notification that the system UI mode of the front application has changed.

- class AppTerminatedDoer

  Handles notification that another application has terminated.

- class Attachable

  Class for objects which have an associated list of Attachments.

- class Attachment

  Abstract class for identifiable persistent objects.

- class AutoAEDesc

  Wrapper for a system Apple Event descriptor.

- class AutoHandle

  Manages ownership of Toolbox Handle data block.

- class AutoNavReply

  Manages ownership of a Toolbox NavReplyRecord.

- class AutoRefCount

  Template class for automatically reference counting objects.

- class AutoRetained

  Template class for automatically retaining and releasing Retained objects.

- class AutoValueSaver

  Template class for automatically saving and restoring a variable's value.

- class BaseView

  A basic view.

- class BevelButton

  A system bevel button control.

- class BindingsFrameAdapter

  Adjusts a view frame based on whether its sides are bound to the corresponding sides of its container frame.

- class CFArray

  Template class wrapper for a Core Foundation Array.

- class CFBundle

  Wrapper class for Core Foundation Bundle.

- class CFData

Wrapper class for a Core Foundataion Data object.

- class CFDictionary

  Template wrapper class for Core Foundation Dictionary.

- class CFMutableObject

  Template base class for Core Foundation wrapper classes for mutable objects.

- class CFObject

  Template base class for Core Foundation wrapper classes.

- class CFString

  Wrapper class for Core Foundation String.

- class CFTree

  Wrapper class for Core Foundation Tree.

- class CFURL

  Wrapper class for Core Foundation URL.

- class CFXMLElement

  Helper class for accessing the attributes of an XML Node for an element.

- class CFXMLNode

  Wrapper class for Core Foundation XML Node.

- class CFXMLTree

  Wrapper class for Core Foundation XML Tree.

- class CGContextSaver

  Saves and restores a Core Graphics context.

- class ChasingArrows

  A system chasing arrows activity indicator.

- class CheckBox

  A system check box control.

- class CheckBoxGroupBox

  A system group box with a check box title.

- class ClockControl

A system clock control.

- class ComboBox

  A system combo box control.

- class CommandConverter

  Handles processing and updating command events by converting them into events for specific commands.

- class CommandHandler

  Handles processing and updating the status of a specific command.

- struct CommandIDType

  Template which creates a unique type for a literal command ID value.

- class CommandProcessDoer

  Handles HICommands.

- class CommandTask

  Abstract class for an Attachment which handles a command event.

- class CommandUpdateStatusDoer

  Handles updating the status of items that invoke commands.

- class ControlActivateDoer

  Handles a control becoming active.

- class ControlAddedSubControlDoer

  Handles notification when a subcontrol is added.

- class ControlApplyBackgroundDoer

  Handles applying a control's background to a port.

- class ControlApplyTextColorDoer

  Handles applying a control's text color to a port/context.

- class ControlArbitraryMessageDoer

  Handles old-style CDEF messages.

- class ControlBoundsChangedDoer

  Handles adapting to a change in the bounds of a control.

- class ControlClickDoer

Handles a mouse down event inside a control.

- class ControlDeactivateDoer

  Handles a control becoming inactive.

- class ControlDisposeDoer

  Handles a control being disposed.

- class ControlDragEnterDoer

  Handles a drag entering a control.

- class ControlDragLeaveDoer

  Handles a drag leaving a control.

- class ControlDragReceiveDoer

  Handles a drag being dropped in a control.

- class ControlDragWithinDoer

  Handles a drag remaining inside a control.

- class ControlDrawDoer

  Handles drawing a control.

- class ControlEnabledStateChangedDoer

  Handles notification when a control is enabled or disabled.

- class ControlGetFocusPartDoer

  Returns the currently focused part of a control.

- class ControlGetOptimalBoundsDoer

  Returns the optimal bounds for a control.

- class ControlGetPartBoundsDoer

  Returns the bounding rectangle of a control part.

- class ControlGetPartRegionDoer

  Returns the bounding region of a control part.

- class ControlGetSizeConstraintsDoer

  Returns the minimum and maximum sizes for a control.

- class ControlHiliteChangedDoer

Handles notification when the hilite state of a control changes.

- class ControlHitDoer

  Handles a click in a control.

- class ControlHitTestDoer

  Handles testing whether a point is within a control.

- class ControlOwningWindowChangedDoer

  Handles notification when a control moves into a different window.

- struct ControlPartCodeStruct

  Wrapper for ControlPartCode.

- class ControlRemovingSubControlDoer

  Handles notification when a subcontrol is being removed.

- class ControlSetCursorDoer

  Handles setting the cursor when the mouse is inside a control.

- class ControlSetFocusPartDoer

  Handles setting the focus to a part of a control.

- class ControlSimulateHitDoer

  Handles a simulating a click in a control.

- class ControlTitleChangedDoer

  Handles notification when the title of a control changes.

- class ControlTrackDoer

  Handles mouse down tracking inside a control.

- class ControlValueFieldChangedDoer

  Handles notification when the value, minimum value, maximum value, or view size of a control changes.

- class Correspondent

  A generic Event Target.

- class DataError

  Exception class for bad input data.

- class DataFork

Wrapper class for the data fork of a file.

- class DataObject

  Base class for objects that store a data value.

- class DataReader

  A data dictionary for reading state information.

- class DataScrap

  A named scrap for storing and retrieving data.

- class DataWriter

  A data dictionary for writing state information.

- class DisclosureButton

  A system disclosure button control.

- class DisclosureTriangle

  A system disclosure triangle control.

- class DrawerWindow

  A drawer which slides out from an edge of a parent window.

- class EditTextControl

  A system edit text control.

- class EditUnicodeText

  A system edit unicode text control.

- class EventDoer

  Abstract class for a Carbon Event handler.

- class EventDoerAttachment

  Abstract attachment that has an associated event target and specific event type.

- class EventDoerCallback

  Template class for an EventDoer that calls a member function of an object.

- struct EventMouseWheelAxisStruct

  Wrapper for EventMouseWheelAxis.

- class EventTarget

Abstract class for the target of a Carbon Event.

- class Exception

  Base class for PowerPlant X exceptions.

- class File

  A file on disk.

- class FileFork

  Wrapper class for a fork of a file.

- class Folder

  Encapsulates a Mac OS file system folder.

- struct FourCharCodeStruct

  Wrapper for FourCharCode.

- class FrameAdapter

  Abstract class for adjusting the frame of a view when its container frame changes size.

- class FrontWindowEventTarget

  Carbon Event target for the front window of a window layer.

- class FSObject

  Wrapper for a system file reference (FSRef) and related File Manager and MoreFiles X functions.

- struct FSVolumeRefNumStruct

  Wrapper for FSVolumeRefNum.

- class GrafPortSaver

  Saves, changes, and restores the current Quickdraw GrafPort.

- class GrayBox

  View which draws a gray box.

- class HIObjectConstructDoer

  Handles constructing an HIObject.

- class HIObjectDestructDoer

  Handles destroying an HIObject.

- class HIObjectInitializeDoer

  Handles initializing an HIObject.

- class HIObjectIsEqualDoer

  Determines if an HIObject is eaual to another HIObject.

- class HIObjectPrintDebugInfoDoer

  Handles request to print debugging information.

- class HIObjectRefType

  Template wrapper class for HIObjectRef types.

- struct HIToolbarItemRefStruct

  Wrapper for HIToolbarItemRef.

- struct HIToolbarRefStruct

  Wrapper for HIToolbarRef.

- class HotKeyPressedDoer

  Handles a hot key being pressed.

- class HotKeyReleasedDoer

  Handles a hot key being released.

- class IconControl

  A system icon control.

- class IconPushButton

  A system push button with icon control.

- class Identifiable

  Mix-in class for objects with an Object ID.

- class IdleTimer

  Abstract class for an Event Loop Idle Timer.

- class IdleTimerCallback

  Template class for an IdleTimer that calls an object member function.

- class ImageView

  A system view which displays a core graphics image.

- class ImageWell

    A system image well view.

- struct IntegerType

    Template which defines a class based on a built-in integer type.

- class ListBox

    A system list box control.

- class LittleArrows

    A system little arrows control.

- class LogicError

    Exception class for a programming error.

- class MenuBeginTrackingDoer

    Handles the start of tracking the menubar or a pop-up menu.

- class MenuChangeTrackingModeDoer

    Handles changing between mouse and keyboard menu tracking modes.

- class MenuClosedDoer

    Handles a menu being closed.

- struct MenuCommandStruct

    Wrapper for MenuCommand.

- class MenuDisposeDoer

    Handles a menu being disposed.

- class MenuDrawItemContentDoer

    Handles drawing the content of a menu item.

- class MenuDrawItemDoer

    Handles drawing a menu item.

- class MenuEnableItemsDoer

    Handles enabling or disabling items in a menu.

- class MenuEndTrackingDoer

    Handles the end of tracking the menubar or a pop-up menu.

- struct MenuEventOptionsStruct

  Wrapper for MenuEventOptions.

- struct MenuItemIndexStruct

  Wrapper for MenuItemIndex.

- class MenuMatchKeyDoer

  Returns menu item matching a command key equivalent.

- class MenuMeasureItemHeightDoer

  Returns the height, in pixels, of a menu item.

- class MenuMeasureItemWidthDoer

  Returns the width, in pixels, of a menu item.

- class MenuOpeningDoer

  Handles a menu being opened (about to be displayed).

- class MenuPopulateDoer

  Handles populating a menu with items prior to use.

- class MenuTargetItemDoer

  Handles the mouse moving over a menu item.

- struct MenuTrackingModeStruct

  Wrapper for MenuTrackingMode.

- class MessageAttachment

  Attachment which responds to an event by sending a message event to another target.

- class MLTEView

  Text edit view base on MLTE.

- class MouseDownDoer

  Handles the mouse button being pressed.

- class MouseDraggedDoer

  Handles the mouse button being moved while the button is down.

- class MouseEnteredDoer

  Handles the mouse entering a tracking area.

- class MouseExitedDoer

  Handles the mouse leaving a tracking area.

- class MouseMovedDoer

  Handles the mouse button being moved.

- class MouseUpDoer

  Handles the mouse button being released.

- class MouseWheelMovedDoer

  Handles the mouse wheel being moved.

- class NavEventResponder

  Abstract class for handling NavServices callbacks.

- struct ObjectDescriptor

  Stores data describing a Persistent object.

- struct **ObjectIDStruct**
- struct **ObjectStorageIDStruct**
- class OSError

  Exception class for a Mac OS error code.

- class OSErrorCode

  Template exception class for a specific Mac OS Error code.

- struct OSStatusStruct

  Wrapper for OSStatus.

- struct OSTypeStruct

  Wrapper for OSType.

- class OwnedPointer

  Template class which manages a pointer created via ”new”.

- class Persistent

  Abstract base class for persistent objects.

- class PictureControl

  A system picture control.

- class Placard

A system placard view.

- class PopupArrow

  A system popup arrow view.

- class PopupButton

  A system popup button control.

- class PopupGroupBox

  A system group box with a popup menu title.

- class ProgressBar

  A system progress bar control.

- class PushButton

  A system push button control.

- class RadioButton

  A system radio button control.

- class RadioGroup

  A system radio group control.

- class RawKeyDownDoer

  Handles a key being pressed.

- class RawKeyModifiersChangedDoer

  Handles change in what modifier keys are pressed.

- class RawKeyRepeatDoer

  Handles a key being held down.

- class RawKeyUpDoer

  Handles a key being released.

- class RelevanceBar

  A system relevance bar control.

- class ResourceFork

  Wrapper class for the resource fork of a file.

- class ResponseAttachment

Attachment which responds to an event by sending another event.

- class Retained

  Base class for reference counted objects.

- class RoundButton

  A system round button control.

- class RuntimeError

  Exception class for a runtime failure.

- class ScrapPromiseKeeper

  Abstract class for keeping promises to supply data for a scrap.

- class ScrollableGetInfoDoer

  Returns information about a scrollable view.

- class ScrollableInfoChangedDoer

  Handles notification that a scrollable view has changed.

- class ScrollableScrollToDoer

  Handles scrolling a view to a specific location.

- class ScrollBar

  A system scroll bar control.

- class ScrollView

  A system scroll view.

- class SeparatorLine

  A system separator line view.

- class ServiceCopyDoer

  Handles the service for copying data from current focus.

- class ServiceGetTypesDoer

  Handles the service getting the types of data which can be copied and pasted.

- class ServicePasteDoer

  Handles the service for pasting data into the current focus.

- class ServicePerformDoer

Handles performing a service.

- class SheetAlert

  An alert displayed as a sheet in a parent window.

- class SheetWindow

  A window displayed as a sheet in a parent window.

- class Slider

  A system slider control.

- struct SourceLocation

  Location within a source file.

- class SpecificAppleEventDoer

  Template class for an Apple Event handler that responds to one specific type of event.

- class SpecificCommandDoer

  Handles processing a specific command.

- class SpecificCommandStatusDoer

  Handles updating the status of a specific command.

- class SpecificEventDoer

  Template class for a Carbon Event handler that responds to one specific type of event.

- class SpecificMenuCommandDoer

  Handles processing a specific menu command that is always enabled when the object is in the current focus chain.

- class SpecificMenuCommandEnableDoer

  Always enables a specific menu command.

- class StaticText

  A system static text control.

- class StatusCommandTask

  Abstract class for a Command event handler Attachment that also handles updating the status of items that invoke the command.

- class SysAEHandler

  Wrapper class for an Apple Event Handler.

- class SysAEHandlerUPP

  Wrapper class for an Apple Event callback function UPP.

- class SysAppleEvent

  Wrapper class for an Apple Event.

- class SysCarbonEvent

  Wrapper class for a Carbon Event.

- class SysEventHandler

  Wrapper class for a Carbon Event Handler.

- class SysEventHandlerUPP

  Wrapper class for a Carbon Event callback function UPP.

- class SysEventLoopIdleTimer

  Wrapper class for an Event Loop Idle Timer.

- class SysEventLoopIdleTimerUPP

  Wrapper class for an Event Loop Idle Timer callback function UPP.

- class SysEventLoopTimer

  Wrapper class for an Event Loop Timer.

- class SysEventLoopTimerUPP

  Wrapper class for an Event Loop Timer callback function UPP.

- struct SysEventSpec

  Struct describing the type of a Carbon Event.

- class SysHIObject

  Wrapper class for a Mac HIObject.

- class SysHIView

  Wrapper class for a Mac Toolbox HIView.

- class SysNavEventUPP

  Wrapper class for a Navigation Services Event callback function UPP.

- class SysScrapPromiseKeeperUPP

  Wrapper class for a Scrap Promise Keeper callback function UPP.

- class SysWindow

  Wrapper class for a Mac Toolbox Window.

- class TabView

  A system tab view.

- class TargetAttachment

  Abstract attachment that has an associated event target.

- struct TDataObject

  Template class for objects that store a single data value of type TData.

- struct TDataVector

  Template class for objects that store a vector of data values of type TData.

- class TextGroupBox

  A system group box with a text title.

- class TextInputGetSelectedTextDoer

  Returns the selected text.

- class TextInputOffsetToPosDoer

  Converts from inline session text offset to global QD point.

- class TextInputPosToOffsetDoer

  Converts from global QD point to inline session text offset.

- class TextInputShowHideBottomWindowDoer

  Shows or hides the bottom line input window.

- class TextInputUnicodeForKeyEventDoer

  Handles unicode text input from the keyboard.

- class TextInputUnicodeTextDoer

  Inputs unicode text.

- class TextInputUpdateActiveInputAreaDoer

  Updates contents of a text input area.

- struct ThemeMenuItemTypeStruct

  Wrapper for ThemeMenuItemType.

- struct ThemeMenuStateStruct

  Wrapper for ThemeMenuState.

- class ThemeTextBox

  View for drawing text using a theme font inside a bounding box.

- class Timer

  Abstract class for an Event Loop Timer.

- class TimerCallback

  Template class for a Timer that calls an object member function.

- class ToolbarCreateItemFromDragDoer

  Creates a new toolbar item from a drag and drop operation.

- class ToolbarCreateItemWithIdentifierDoer

  Creates a new toolbar item with a specified identifier.

- class ToolbarGetAllowedIdentifiersDoer

  Returns list of default item identifiers for a toolbar.

- class ToolbarGetDefaultIdentifiersDoer

  Returns list of default item identifiers for a toolbar.

- struct UniCharStruct

  Wrapper for UniChar.

- class UserFocusEventTarget

  Carbon Event target for the current user focus.

- class View

  Abstract base class for a visual element.

- class VolumeMountedDoer

  Notification that a volume has been mounted.

- class VolumeUnmountedDoer

  Notification that a volume has been unmounted.

- class Window

  Window for displaying data on screen.

- class WindowActivatedDoer

  Handles a window being activated.

- struct WindowAttributesStruct

  Wrapper for WindowAttributes.

- class WindowBoundsChangedDoer

  Handles a window having been moved and/or resized.

- class WindowBoundsChangingDoer

  Handles a window being moved and/or resized.

- struct WindowClassStruct

  Wrapper for WindowClass.

- class WindowCloseAllDoer

  Handles a request to close all windows.

- class WindowClosedDoer

  Handles a window about to be disposed.

- class WindowCloseDoer

  Handles a request to close a window.

- class WindowCollapseAllDoer

  Handles a request to collapse all windows.

- class WindowCollapsedDoer

  Handles a window after being collapsed.

- class WindowCollapseDoer

  Handles a request to collapse a window.

- class WindowCollapsingDoer

  Handles a window about to be collapsed.

- class WindowConstrainDoer

  Handles notification that the available window area has changed.

- class WindowContentView

  Top-level view for the contents of a window.

- class WindowContextualMenuSelectDoer

  Handles a click in a window intended to invoke a contextual menu.

- class WindowCursorChangeDoer

  Handles changing the cursor when the mouse is inside a window.

- class WindowDeactivatedDoer

  Handles a window being deactivated.

- struct WindowDefPartCodeStruct

  Wrapper for WindowDefPartCode.

- class WindowDisposeDoer

  Disposes a window definition.

- class WindowDragCompletedDoer

  Handles a window finishing being dragged.

- class WindowDragHiliteDoer

  Handles drag hiliting for a window.

- class WindowDragStartedDoer

  Handles a window starting to be dragged.

- class WindowDrawContentDoer

  Handles drawing the contents of a window.

- class WindowDrawerClosedDoer

  Handles a drawer being fully closed.

- class WindowDrawerClosingDoer

  Handles a drawer starting to close.

- class WindowDrawerOpenedDoer

  Handles a drawer being fully open.

- class WindowDrawerOpeningDoer

  Handles a drawer starting to open.

- class WindowDrawFrameDoer

  Draws a window’s structure.

- class WindowDrawGrowBoxDoer

  Draws a window's grow box.

- class WindowDrawPartDoer

  Draws a specific part of a window's structure.

- class WindowExpandAllDoer

  Handles a request to expand all windows.

- class WindowExpandDoer

  Handles a request to expand a window.

- class WindowExpandedDoer

  Handles a window after being expanded.

- class WindowExpandingDoer

  Handles a window about to be expanded.

- class WindowFocusAcquiredDoer

  Handles a window acquiring the focus.

- class WindowFocusContentDoer

  Handles a setting the focus to the main view of a window.

- class WindowFocusRelinquishDoer

  Handles a window relinquishing the focus.

- class WindowFocusToolbarDoer

  Handles a setting the focus to the toolbar of a window.

- class WindowGetClickActivationDoer

  Handles a window being activated by a mouse click.

- class WindowGetGrowImageRegionDoer

  Returns the outline for a window being resized.

- class WindowGetIdealSizeDoer

  Returns the ideal size of a window's content region.

- class WindowGetMaximumSizeDoer

  Returns the maximum size of a window's content region.

- class WindowGetMinimumSizeDoer

  Returns the minimum size of a window's content region.

- class WindowGetRegionDoer

  Returns a specified region of a window.

- class WindowHandleContentClickDoer

  Handles a click in a window.

- class WindowHeader

  A system window header view.

- class WindowHiddenDoer

  Handles a window after being hidden.

- class WindowHidingDoer

  Handles a window being hidden.

- class WindowHitTestDoer

  Returns the window part hit by a specified mouse location.

- class WindowInitDoer

  Initializes a window definition.

- class WindowMeasureTitleDoer

  Returns the width of a window's title area.

- class WindowModifiedDoer

  Handles change in modified state of a window.

- class WindowPaintDoer

  Paints a window.

- class WindowPathSelectDoer

  Handles a request to select from the window path popup menu.

- struct WindowRegionCodeStruct

  Wrapper for WindowRegionCode.

- class WindowResizeCompletedDoer

  Handles a window finishing being resized.

- class WindowResizeStartedDoer

  Handles a window starting to be resized.

- class WindowSetupProxyDragImageDoer

  Handles creating a drag image for a window's proxy icon.

- class WindowShowingDoer

  Handles a window being shown.

- class WindowShownDoer

  Handles a window after being shown.

- class WindowStateChangedDoer

  Handles change in window state.

- class WindowUpdateDoer

  Handles a low-level window update event.

- class WindowZoomAllDoer

  Handles a request to zoom all windows.

- class WindowZoomDoer

  Handles a request to zoom a window.

- class WindowZoomedDoer

  Handles a window after being zoomed.

Typedefs

- typedef IntegerType< FourCharCodeStruct, FourCharCode > **FourCharCodeType**
- typedef IntegerType< UniCharStruct, UniChar > **UniCharType**
- typedef IntegerType< OSStatusStruct, OSStatus > **OSStatusType**
- typedef IntegerType< ControlPartCodeStruct, ControlPartCode > **ControlPartCodeType**
- typedef IntegerType< EventMouseWheelAxisStruct, EventMouseWheelAxis > **EventMouseWheelAxisType**
- typedef IntegerType< MenuTrackingModeStruct, MenuTrackingMode > **MenuTrackingModeType**
- typedef IntegerType< MenuItemIndexStruct, MenuItemIndex > **MenuItemIndexType**

- typedef IntegerType< MenuCommandStruct, MenuCommand > **MenuCommandType**
- typedef IntegerType< MenuEventOptionsStruct, MenuEventOptions > **MenuEventOptionsType**
- typedef IntegerType< ThemeMenuStateStruct, ThemeMenuState > **ThemeMenuStateType**
- typedef IntegerType< ThemeMenuItemTypeStruct, ThemeMenuItemType > **ThemeMenuItemTypeType**
- typedef IntegerType< WindowClassStruct, WindowClass > **WindowClassType**

- typedef IntegerType< WindowAttributesStruct, WindowAttributes > **WindowAttributesType**
- typedef IntegerType< WindowDefPartCodeStruct, WindowDefPartCode > **WindowDefPartCodeType**
- typedef IntegerType< WindowRegionCodeStruct, WindowRegionCode > **WindowRegionCodeType**
- typedef IntegerType< OSTypeStruct, OSType > **OSTypeType**
- typedef IntegerType< FSVolumeRefNumStruct, FSVolumeRefNum > **FSVolumeRefNumType**
- typedef SInt16 **ResIDT**
- typedef UInt32 **CommandIDT**
- typedef FourCharCode **EventClassT**
- typedef UInt32 **EventKindT**
- typedef IntegerType< ObjectIDStruct, UInt32 > **ObjectIDT**
- typedef FourCharCode ExceptionIDT

  The ”what” of an exception.

- typedef std::map< CFString, Registrar::CreatorFunction > **RegistryType**
- typedef IntegerType< ObjectStorageIDStruct, UInt32 > **ObjectStorageIDT**
- typedef std::map< CFString, AutoRetained< DataObject > > **KeyDataMap**
- typedef std::map< const Persistent ∗, ObjectStorageIDT > **ObjectIDMap**
- typedef std::deque< const Persistent ∗ > **ObjectQueue**
- typedef std::vector< ObjectDescriptor > **ObjectDescriptorList**
- typedef HIObjectRefType< HIToolbarRefStruct > **HIToolbarRefType**
- typedef HIObjectRefType< HIToolbarItemRefStruct > **HIToolbarItemRefType**

Enumerations

- enum EMetaTarget { **metaTarget_Self** = 0, **metaTarget_UserFocus** = 1, **metaTarget_Application** = 2 }

  Meta targets for events.

Functions

- template<class TException> void ThrowException (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Template function for throwing PPx Exception objects.

- void ThrowOSError (OSStatus inErrorCode, const char ∗inWhy, const SourceLocation &inWhere)

  Throws a PPx::OSError exception.

- template<OSStatus status> void ThrowOSErrorCode (const char ∗inWhy, const SourceLocation &inWhere)

  Template function which throws a PPx::OSErrorCode<> exception.

- void ThrowIfOSError (OSStatus inErrorCode, const char ∗inWhy, const SourceLocation &inWhere)

  Throws a PPx::OSError exception if the error code is not noErr.

- template<class T> std::auto_ptr< T > CreateNew ()

  Template function that creates a new object of a class.

- template<class T, typename TParam1> std::auto_ptr< T > CreateNew (TParam1 inParam1)

  Template function that creates a new object of a class with 1 parameter.

- template<class T, typename TParam1, typename TParam2> std::auto_ptr< T > CreateNew (TParam1 inParam1, TParam2 inParam2)

  Template function that creates a new object of a class with 2 parameters.

- template<class T, typename TParam1, typename TParam2, typename TParam3> std::auto_ptr< T > CreateNew (TParam1 inParam1, TParam2 inParam2, TParam3 inParam3)

  Template function that creates a new object of a class with 3 parameters.

- template<class TCastTo, class TBase> TCastTo SafeDynamicCast (TBase inObjectPointer)

  Template function for performing a dynamic_cast which throws if the cast from TBase to TCastTo is unsuccessful.

- pascal OSErr **AEHandlerTBCallback** (const AppleEvent ∗inAppleEvent, AppleEvent ∗outAEReply, long inRefCon)
- AEEventHandlerUPP **GetSysAEHandlerUPP** ()
- pascal OSStatus **EventHandlerTBCallback** (EventHandlerCallRef, EventRef, void ∗)

- pascal OSStatus **EventHandlerCallback** (EventHandlerCallRef inCallRef, EventRef inEventRef, void ∗inUserData)
- EventHandlerUPP **GetSysEventHandlerUPP** ()
- pascal void **TimerTBCallback** (EventLoopTimerRef, void ∗inUserData)
- EventLoopTimerUPP **GetSysEventLoopTimerUPP** ()
- pascal void **IdleTimerTBCallback** (EventLoopTimerRef, EventLoopIdleTimerMessage inMessage, void ∗inUserData)
- EventLoopIdleTimerUPP **GetSysEventLoopIdleTimerUPP** ()
- void RegisterCommonXMLDecoders ()

  Registers XML Decoders for common data types.

- void RegisterCommonXMLEncoders ()

  Registers XML Encoders for common data types.

- RegistryType & **GetRegistry** ()
- bool **HasStorageID** (const ObjectDescriptor &inObjDesc, ObjectStorageIDT inStorageID)
- RegistryType & **GetRegistry** ()
- RegistryType & **GetRegistry** ()
- pascal OSStatus **ScrapPromiseKeeperCallback** (ScrapRef inScrap, ScrapFlavorType inFlavor, void ∗inUserData)
- ScrapPromiseKeeperUPP **GetSysScrapPromiseKeeperUPP** ()
- pascal void **NavEventCallback** (NavEventCallbackMessage inMessage, NavCBRecPtr inParams, void ∗inUserData)
- NavEventUPP **GetNavEventUPP** ()
- template<typename TCFRef> TCFRef RetainCFRef (TCFRef inRef)

  Template function for retaining a CF reference.

- CFString **operator+** (const CFString &inLeft, const CFString &inRight)
- CFString **operator+** (const CFString &inLeft, CFStringRef inRight)
- CFString **operator+** (CFStringRef inLeft, const CFString &inRight)
- CFString **operator+** (const CFString &inLeft, ConstStringPtr inRight)
- CFString **operator+** (ConstStringPtr inLeft, const CFString &inRight)
- CFString **operator+** (const CFString &inLeft, const char ∗inRight)
- CFString **operator+** (const char ∗inLeft, const CFString &inRight)
- bool **operator==** (const CFString &inLeft, const CFString &inRight)
- bool **operator==** (const CFString &inLeft, CFStringRef inRight)
- bool **operator==** (CFStringRef inLeft, const CFString &inRight)
- bool **operator!=** (const CFString &inLeft, const CFString &inRight)
- bool **operator!=** (const CFString &inLeft, CFStringRef inRight)
- bool **operator!=** (CFStringRef inLeft, const CFString &inRight)
- bool **operator<** (const CFString &inLeft, const CFString &inRight)
- bool **operator<** (const CFString &inLeft, CFStringRef inRight)

- bool **operator<** (CFStringRef inLeft, const CFString &inRight)
- bool **operator<=** (const CFString &inLeft, const CFString &inRight)
- bool **operator<=** (const CFString &inLeft, CFStringRef inRight)
- bool **operator<=** (CFStringRef inLeft, const CFString &inRight)
- bool **operator>** (const CFString &inLeft, const CFString &inRight)
- bool **operator>** (const CFString &inLeft, CFStringRef inRight)
- bool **operator>** (CFStringRef inLeft, const CFString &inRight)
- bool **operator>=** (const CFString &inLeft, const CFString &inRight)
- bool **operator>=** (const CFString &inLeft, CFStringRef inRight)
- bool **operator>=** (CFStringRef inLeft, const CFString &inRight)
- bool **operator==** (const SysEventSpec &inLeft, const SysEventSpec &inRight)

- pascal OSStatus **EventHandlerTBCallback** (EventHandlerCallRef, EventRef, void ∗)
- EventHandlerUPP **GetSysEventHandlerUPP** ()
- pascal OSStatus **ObjectLifetimeEventCallback** (EventHandlerCallRef inCallRef, EventRef inEventRef, void ∗inUserData)
- EventHandlerUPP **GetObjectLifetimeEventUPP** ()
- template<class TView> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible)
- template<class TView, typename T1> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1)
- template<class TView, typename T1, typename T2> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2)
- template<class TView, typename T1, typename T2, typename T3> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2, T3 in3)
- template<class TView, typename T1, typename T2, typename T3, typename T4> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2, T3 in3, T4 in4)
- template<class TView, typename T1, typename T2, typename T3, typename T4, typename T5> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2, T3 in3, T4 in4, T5 in5)

- template<class TView, typename T1, typename T2, typename T3, typename T4, typename T5, typename T6> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2, T3 in3, T4 in4, T5 in5, T6 in6)
- template<class TView, typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2, T3 in3, T4 in4, T5 in5, T6 in6, T7 in7)

- template<class TView, typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7, typename T8> TView ∗ **CreateView** (View ∗inSuperView, const HIRect &inFrame, bool inEnabled, bool inVisible, T1 in1, T2 in2, T3 in3, T4 in4, T5 in5, T6 in6, T7 in7, T8 in8)

Variables

- const CFStringEncoding **encoding_System** = ::CFStringGetSystemEncoding()
- const CFStringRef **cfString_Empty** = CFSTR("")
- const unsigned char **pString_Empty** [ ] = { 0 }
- const CFRange **cfRange_All** = { 0, cfIndex_Last }
- const CFRange **cfRange_Empty** = { 0, 0 }
- const UInt32 **attributes_None** = 0
- const OptionBits **options_None** = 0
- const OptionBits **features_None** = 0
- const bool **visible_Yes** = true
- const bool **visible_No** = false
- const bool **enabled_Yes** = true
- const bool **enabled_No** = false
- const SInt32 **value_Off** = 0
- const SInt32 **value_On** = 1
- const SInt32 **value_Mixed** = 2
- const SInt32 **value_Closed** = 0
- const SInt32 **value_Open** = 1
- const bool **clear_Yes** = true
- const bool **clear_No** = false
- const bool **retain_Yes** = true
- const bool **retain_No** = false
- const ObjectIDT **objectID_None** = 0
- const CommandIDT **commandID_None** = 0
- const CFStringEncoding **encoding_Invalid** = kCFStringEncodingInvalidId
- const CFIndex **cfSize_Unlimited** = 0
- const CFIndex **cfIndex_First** = CFIndex(0)
- const CFIndex **cfIndex_Last** = CFIndex(0x7FFFFFFF)
- const CFIndex **cfIndex_BeforeStart** = CFIndex(-2)
- const CFIndex **cfIndex_AfterEnd** = CFIndex(-3)
- const CFIndex **cfIndex_Bad** = CFIndex(-4)
- const SInt16 **refNum_Invalid** = -1
- const SInt32 **dirID_Invalid** = 0
- const EventClassT eventClass_ProcessCommand = 'Pcmd'

  Custom event class for processing a command.

- const EventClassT eventClass_UpdateCmdStatus = ’Pups’

  Custom event class for updating the status of a command.

- const SourceLocation sourceLocation_Nothing (nil, nil, 0)

  Nil source code location.

- const ResIDT **ALRT_Exception** = 253
- const ResIDT **ALRT_Signal** = 254
- const SInt16 **button_Continue** = 1
- const SInt16 **button_Abort** = 4
- const SInt16 **button_Debugger** = 5
- const SInt16 **button_Quiet** = 6
- const MenuID **menuID_DebugStr** = 19876
- const ExceptionIDT **err_MacOS** = ’oser’
- const ExceptionIDT **err_Logic** = ’logc’
- const ExceptionIDT **err_Runtime** = ’runt’
- const ExceptionIDT **err_DataFormat** = ’data’
- const ExceptionIDT **err_CFCreate** = ’cfcr’
- const ExceptionIDT **err_CFStringConvert** = ’cfsc’
- const ExceptionIDT **err_CFNilRef** = ’cfnr’
- const ExceptionIDT **err_XMLFormat** = ’xmlf’
- const ExceptionIDT **err_DynamicCast** = ’cast’
- const ExceptionIDT **err_BadParam** = ’parm’
- const ExceptionIDT **err_MissingData** = ’misd’
- const CFStringRef **key_CommandID** = CFSTR(”command id”)
- const CFStringRef **key_EventTarget** = CFSTR(”event target”)
- const CFStringRef **key_Target** = CFSTR(”target”)
- const CFStringRef **key_TargetEventClass** = CFSTR(”target event class”)
- const CFStringRef **key_TargetEventKind** = CFSTR(”target event kind”)
- const CFStringRef **key_ResponseMetaTarget** = CFSTR(”response meta target”)
- const CFStringRef **key_ResponseEventClass** = CFSTR(”response event class”)

- const CFStringRef **key_ResponseEventKind** = CFSTR(”response event kind”)
- const CFStringRef **key_MessageTarget** = CFSTR(”message target”)
- const CFStringRef **key_MessageEventClass** = CFSTR(”message event class”)
- const CFStringRef **key_MessageEventKind** = CFSTR(”message event kind”)
- const CFStringRef **key_WindowClass** = CFSTR(”window class”)
- const CFStringRef **key_Attachments** = CFSTR(”attachments”)
- const CFStringRef **key_AttachmentID** = CFSTR(”attachment id”)
- const CFStringRef **xml_TargetName** = CFSTR(”xml”)
- const CFStringRef **xml_ProcessingInst** = CFSTR(”version=\”1.0\” encoding=\”UTF-8\””)

- const CFStringRef **elem_ObjectDescriptors** = CFSTR(”ObjectDescriptors”)
- const CFStringRef **attr_Version** = CFSTR(”version”)
- const CFStringRef **value_VersionOne** = CFSTR(”1”)
- const EventTime eventTime_Now = EventTime(0)

  Indicates ”now” as the time stamp for a Carbon Event.

- Rect **rect_DefaultBounds** = { 0, 0, 1, 1 }
- const EventParamName **param_ControlFeatures** = ’Feat’
- const CFStringRef **name_Class** = CFSTR(”PPx::BaseView”)
- const CFStringRef **key_Features** = CFSTR(”features”)
- const CFStringRef **key_InitValue** = CFSTR(”initial value”)
- const CFStringRef **key_MinValue** = CFSTR(”minimum value”)
- const CFStringRef **key_MaxValue** = CFSTR(”maximum value”)
- const CFStringRef **key_ViewTitle** = CFSTR(”view title”)
- const CFStringRef **key_AutoToggle** = CFSTR(”auto toggle”)
- const CFStringRef **key_IsPrimaryGroup** = CFSTR(”is primary group”)
- const CFStringRef **key_TextContent** = CFSTR(”text content”)
- const CFStringRef **key_TextJustification** = CFSTR(”text justification”)
- const CFStringRef **key_ContentType** = CFSTR(”content type”)
- const CFStringRef **key_ContentResID** = CFSTR(”content resource id”)
- const CFStringRef **key_ViewID** = CFSTR(”view id”)
- const CFStringRef **key_SuperView** = CFSTR(”superview”)
- const CFStringRef **key_FrameAdapter** = CFSTR(”frame adapter”)
- const CFStringRef **key_ViewFrame** = CFSTR(”view frame”)
- const CFStringRef **key_IsVisible** = CFSTR(”is visible”)
- const CFStringRef **key_IsEnabled** = CFSTR(”is enabled”)
- const OSType **property_PPxView** = ’view’
- const CFStringRef **key_StrokeGray** = CFSTR(”stroke gray”)
- const CFStringRef **key_StrokeAlpha** = CFSTR(”stroke alpha”)
- const CFStringRef **key_FillGray** = CFSTR(”fill gray”)
- const CFStringRef **key_FillAlpha** = CFSTR(”fill alpha”)
- const CFStringRef **key_Thickness** = CFSTR(”thickness”)
- const CFStringRef **key_ButtonBehavior** = CFSTR(”button behavior”)
- const CFStringRef **key_TextAlignment** = CFSTR(”text alignment”)
- const CFStringRef **key_TextOffset** = CFSTR(”text offset”)
- const CFStringRef **key_TextPlacement** = CFSTR(”text placement”)
- const CFStringRef **key_IconTransform** = CFSTR(”icon transform”)
- const CFStringRef **key_GraphicAlignment** = CFSTR(”graphic alignment”)
- const CFStringRef **key_GraphicOffset** = CFSTR(”graphic offset”)
- const CFStringRef **key_MenuID** = CFSTR(”menu id”)
- const CFStringRef **key_MenuBehavior** = CFSTR(”menu behavior”)
- const CFStringRef **key_MenuPlacement** = CFSTR(”menu placement”)
- const CFStringRef **key_MenuValue** = CFSTR(”menu value”)

- const CFStringRef **key_CenterPopupGlyph** = CFSTR(”center popup glyph”)
- const ResIDT **menuID_Unspecified** = -12345
- const CFStringRef **key_ClockType** = CFSTR(”clock type”)
- const CFStringRef **key_ClockFlags** = CFSTR(”clock flags”)
- const CFStringRef **key_Attributes** = CFSTR(”attributes”)
- const CFStringRef **key_Orientation** = CFSTR(”orientation”)
- const CFStringRef **key_DrawTitle** = CFSTR(”draw title”)
- const CFStringRef **key_IsPassword** = CFSTR(”is password”)
- const CFStringRef **key_UseInlineInput** = CFSTR(”use inline input”)
- const CFStringRef **key_DontTrack** = CFSTR(”do not track”)
- const CFStringRef **key_IconAlignment** = CFSTR(”icon alignment”)
- const CFStringRef **key_IsOpaque** = CFSTR(”is opaque”)
- const CFStringRef **key_Alpha** = CFSTR(”alpha”)
- const CFStringRef **key_ScaleToFit** = CFSTR(”scale to fit”)
- const CFStringRef **key_Increment** = CFSTR(”increment”)
- const CFStringRef **key_ArrowSize** = CFSTR(”arrow size”)
- const CFStringRef **key_HasVariableWidth** = CFSTR(”has variable width”)
- const CFStringRef **key_TitleWidth** = CFSTR(”title width”)
- const CFStringRef **key_TitleJustification** = CFSTR(”title justification”)
- const CFStringRef **key_TitleStyle** = CFSTR(”title style”)
- const CFStringRef **key_IsDeterminate** = CFSTR(”is determinate”)
- const CFStringRef **key_ButtonSize** = CFSTR(”button size”)
- const CFStringRef **key_ViewSize** = CFSTR(”view size”)
- const CFStringRef **key_HasLiveTracking** = CFSTR(”has live tracking”)
- const CFStringRef **key_ScrollViewOptions** = CFSTR(”scroll options”)
- const CFStringRef **key_AutoHideScrollbars** = CFSTR(”autohide scrollbars”)
- const CFStringRef **key_TickMarksCount** = CFSTR(”tick marks count”)
- const CFStringRef **key_TextColor** = CFSTR(”text color”)
- const CFStringRef **key_IsListHeader** = CFSTR(”IsListHeader”)
- const CFStringRef **key_ThemeFontID** = CFSTR(”theme font id”)
- const CFStringRef **key_OneLineOnly** = CFSTR(”one line only”)
- const View ∗ superView_None = nil

  Nil superview.

- const CFStringRef **key_BindLeft** = CFSTR(”bind left”)
- const CFStringRef **key_BindTop** = CFSTR(”bind top”)
- const CFStringRef **key_BindRight** = CFSTR(”bind right”)
- const CFStringRef **key_BindBottom** = CFSTR(”bind bottom”)
- const CFStringRef **key_PreferredEdge** = CFSTR(”preferred edge”)
- const CFStringRef **key_LeadingOffset** = CFSTR(”leading offset”)
- const CFStringRef **key_TrailingOffset** = CFSTR(”trailing offset”)
- const CFStringRef **key_WindowAttrs** = CFSTR(”window attributes”)
- const CFStringRef **key_WindowTitle** = CFSTR(”window title”)

- const CFStringRef **key_ContentBounds** = CFSTR("content bounds")
- const CFStringRef **key_ContentView** = CFSTR("content view")
- const OSType **property_PPxWindow** = 'wind'
- WindowAttributes **sDefaultAttributes**

5.1.2 Function Documentation

5.1.2.1 void PPx::RegisterCommonXMLDecoders ()

Registers XML Decoders for common data types.

XML Decoders convert information from an XML Tree to Data Objects.

You should call this function before attempting to read persistent data from XML information

Definition at line 26 of file PPxRegisterAll.cp.

5.1.2.2 void PPx::RegisterCommonXMLEncoders ()

Registers XML Encoders for common data types.

XML Encoders convert information from Data Objects to XML Trees.

You should call this function before attempting to write persistent data in XML format

Definition at line 65 of file PPxRegisterAll.cp.

5.1.2.3 template<typename TCFRef> TCFRef RetainCFRef (TCFRef *inRef*)

Template function for retaining a CF reference.

Parameters:

inRef CF reference to retain

Returns:

The CF reference that was retained

Note:

The Toolbox CFRetain() function returns a generic CFTypeRef. This template function returns the same CF reference type as the input parameter, making it type-safe to use the return value.

Definition at line 478 of file SysCFObject.h.

5.1.2.4 template<class TCastTo, class TBase> TCastTo SafeDynamicCast (TBase *inObjectPointer*)

Template function for performing a dynamic_cast which throws if the cast from TBase to TCastTo is unsuccessful.

Parameters:

inObjectPointer Pointer of type TBase to dynamically cast

Returns:

Pointer of type TCastTo

Definition at line 90 of file PPxMemoryUtils.h.

References PPx_ThrowIfNil_.

5.1.2.5 template<class TException> void ThrowException (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[inline]`

Template function for throwing PPx Exception objects.

Parameters:

inWhat Exception ID

inWhy A string describing the cause of the exception

inWhere Location in the source code of the exception

Definition at line 248 of file PPxExceptions.h.

5.1.2.6 void ThrowIfOSError (OSStatus *inErrorCode*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[inline]`

Throws a PPx::OSError exception if the error code is not noErr.

Parameters:

inErrorCode A Mac OS error code

inWhy A string describing the cause of the exception

inWhere Location in the source code of the exception

Definition at line 305 of file PPxExceptions.h.

References ThrowOSError().

5.1.2.7 void ThrowOSError (OSStatus *inErrorCode*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[inline]`

Throws a PPx::OSError exception.

Parameters:

inErrorCode A Mac OS error code

inWhy A string describing the cause of the exception

inWhere Location in the source code of the exception

Definition at line 267 of file PPxExceptions.h.

Referenced by ThrowIfOSError().

5.1.2.8 template<OSStatus status> void ThrowOSErrorCode (const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[inline]`

Template function which throws a PPx::OSErrorCode<> exception.

Template parameter is a literal error code value.

Parameters:

inWhy A string describing the cause of the exception

inWhere Location in the source code of the exception

Definition at line 287 of file PPxExceptions.h.

5.1.3 Variable Documentation

5.1.3.1 WindowAttributes PPx::sDefaultAttributes `[static]`

Initial value:

```
kWindowCompositingAttribute +
                                                  kWindowStandardHandlerAttribute
```

Definition at line 39 of file PPxWindow.cp.

5.1.3.2 const SourceLocation PPx::sourceLocation_Nothing

Nil source code location.

Used when debugging options are off.

Definition at line 47 of file PPxDebugging.h.

Referenced by PPx::Exception::Where().

5.2 PPx::BundleUtils Namespace Reference

5.2.1 Detailed Description

Utility functions for working with Bundles.

Functions

- CFData GetResourceData (CFBundleRef inBundle, CFStringRef inResourceName, CFStringRef inResourceType, CFStringRef inSubDirName=nil)

  Returns the data from a named resource file.

- CFTypeRef GetResourceProperty (CFBundleRef inBundle, CFStringRef inPropertyName, CFStringRef inResourceName, CFStringRef inResourceType, CFStringRef inSubDirName=nil)

  Returns a CFTypeRef for a property of a resource.

- CFString GetInfoDictionaryKeyString (CFBundleRef inBundle, CFStringRef inKey)

  Returns a CFString for the value of key in a bundle's info dictionary.

5.2.2 Function Documentation

5.2.2.1 CFString PPx::BundleUtils::GetInfoDictionaryKeyString (CFBundleRef *inBundle*, CFStringRef *inKey*)

Returns a CFString for the value of key in a bundle's info dictionary.

Parameters:

inBundle Bundle whose info dictionary to search

inKey Dictionary key name

Returns:

String containing the value of the key

The is a wrapper for SysCFBundle's GetValueForInfoDictionaryKey function, which looks up a dictianary value (which is usually in the bundle's info.plist file). That function returns a generic CFTypeRef. This wrapper returns a CFString object for values that are CFStringRefs.

Note:

Returns a CFString with a nil refernce if the key is not found. Throws an exception if key is found but the value is not a string.

Definition at line 143 of file PPxBundleUtils.cp.

References PPx::CFBundle::GetValueForInfoDictionaryKey(), and PPx_Throw_.

5.2.2.2 CFData PPx::BundleUtils::GetResourceData (CFBundleRef *inBundle*, CFStringRef *inResourceName*, CFStringRef *inResourceType*, CFStringRef *inSubDirName* = nil)

Returns the data from a named resource file.

Parameters:

inBundle Bundle containing resource files

inResourceName Name of the resource

inResourceType Type of the resource

inSubDirName Subdirectory of bundle in which to start search

Returns:

CFData object containing the resource data

Bundled programs typically store resources as separate files within subdirectories of the bundle. To get the data in a file called MyPicture.jpg, you would pass ”MyPicture” and the resource name and ”jpg” as the resource type. Note that the ”dot” in the file name is not included in the name or type.

You may pass nil for the subdirectory name. If so, the system uses its default search algorithm, which tries to get the correct localized version of the resource. Read the Mac OS documentation on bundles and url access for more information.

Definition at line 37 of file PPxBundleUtils.cp.

References PPx::CFBundle::GetResourceURL(), PPx::CFObject< CFURLRef >::IsValid(), PPx_ThrowIfOSError_, and PPx_ThrowOSErrorCode_.

5.2.2.3 CFTypeRef PPx::BundleUtils::GetResourceProperty (CFBundleRef *inBundle*, CFStringRef *inPropertyName*, CFStringRef *inResourceName*, CFStringRef *inResourceType*, CFStringRef *inSubDirName* = nil)

Returns a CFTypeRef for a property of a resource.

Parameters:

inBundle Bundle containing resource files

inPropertyName Name of the property

inResourceName Name of the resource

inResourceType Type of the resource

inSubDirName Subdirectory of bundle in which to start search

Returns:

CFTypeRef for the property value

See the comments for BundleUtils::GetResourceData for information about resource names, resource types, and subdirectories.

The kind of CFTypeRef returned depends on the property. See <CFURLAccess.h> and Apple's Core Foundation documentation for information about resource properties.

Definition at line 101 of file PPxBundleUtils.cp.

References PPx::CFBundle::GetResourceURL(), and PPx_ThrowIfOSError_.

5.3 PPx::CFUtils Namespace Reference

5.3.1 Detailed Description

Utility functions for working with Core Foundation.

Functions

- CFStringEncoding GetEncodingFromScriptCode (ScriptCode inScript, LangCode inLanguage=kTextLanguageDontCare, RegionCode inRegion=kTextRegionDontCare, ConstStr255Param inFontName=nil)

  Converts a ScriptCode to a CFStringEncoding.

- CFString GetIndString (ResIDT inSTRxID, SInt16 inIndex, CFStringEncoding inEncoding=encoding_System, CFAllocatorRef inAllocator=nil)

  Returns a CFString from the contents of an item in a STR# resource.

- bool MakeValidRange (CFIndex inMaxLength, CFRange &ioRange)

  Returns whether the supplied range is valid for a CF container of the supplied length.

- void VerifyRange (CFIndex inMaxLength, CFRange &ioRange)

  Throws an exception if the supplied range is invalid for a CF container of the supplied length.

- bool MakeValidIndex (CFIndex inCount, CFIndex &ioIndex)

  Returns whether the supplied index is valid for a CF container of the supplied length.

- void VerifyIndex (CFIndex inCount, CFIndex &ioIndex)

  Throws an exception if the supplied index is invalid for a CF container of the supplied length.

- bool MakeInsertIndex (CFIndex inCount, CFIndex &ioIndex)

  Verifies an insertion index and adjusts it if it is a special flag indication the last index.

- void VerifyInsertIndex (CFIndex inCount, CFIndex &ioIndex)

  Throws an exception if an insertion index into a container is not valid.

5.3.2 Function Documentation

5.3.2.1 CFStringEncoding PPx::CFUtils::GetEncodingFromScriptCode (ScriptCode *inScript*, LangCode *inLanguage* = kTextLanguageDontCare, RegionCode *inRegion* = kTextRegionDontCare, ConstStr255Param *inFontName* = nil)

Converts a ScriptCode to a CFStringEncoding.

A CFStringEncoding is equivalent to a TextEncoding, so UpgradeScriptInfoToTextEncoding can be used for the conversion.

Parameters:

inScript The ScriptCode to convert

inLanguage The language of inScript. The default is kTextLanguageDontCare.

inRegion The region of inScript. The default is kTextRegionDontCare.

inFontName The name of the font used for converting inScript. The default is nil.

Returns:

A CFStringEncoding equivalent to inScript

Definition at line 28 of file SysCFUtils.cp.

References PPx_ThrowIfOSError_.

5.3.2.2 CFString PPx::CFUtils::GetIndString (ResIDT *inSTRxID*, SInt16 *inIndex*, CFStringEncoding *inEncoding* = encoding_System, CFAllocatorRef *inAllocator* = nil)

Returns a CFString from the contents of an item in a STR# resource.

Parameters:

inSTRxID Resource ID of a STR# resource.

inIndex Index of a string within the STR# resource specified by inSTRxID.

inEncoding Encoding used to convert the loaded string to unicode. The default is encoding_System.

inAllocator Allocator for the CFString. The default is nil.

Returns:

A CFString containing the contents of the STR# resource item

Definition at line 58 of file SysCFUtils.cp.

5.3.2.3 bool PPx::CFUtils::MakeInsertIndex (CFIndex *inCount*, CFIndex & *ioIndex*)

Verifies an insertion index and adjusts it if it is a special flag indication the last index.

Parameters:

inCount Number of items in container

ioIndex Index into container

Definition at line 211 of file SysCFUtils.cp.

Referenced by VerifyInsertIndex().

5.3.2.4 bool PPx::CFUtils::MakeValidIndex (CFIndex *inCount*, CFIndex & *ioIndex*)

Returns whether the supplied index is valid for a CF container of the supplied length.

Also, if the supplied index is a logical range rather than actual index, it will be converted to the corresponding actual index.

Parameters:

inCount Number of items. 0 <= index <= inCount - 1

ioIndex On input: the actual or logical index to validate. On output: a valid actual index if the input value was valid, otherwise unchanged from the input value.

Returns:

Whether ioIndex is valid

Definition at line 157 of file SysCFUtils.cp.

Referenced by VerifyIndex().

5.3.2.5 bool PPx::CFUtils::MakeValidRange (CFIndex *inMaxLength*, CFRange & *ioRange*)

Returns whether the supplied range is valid for a CF container of the supplied length.

Also, if the supplied range is a logical range rather than actual range, it will be converted to the corresponding actual range.

Parameters:

inMaxLength The maximum possible length of the range.

ioRange On input: the actual or logical range to validate. On output: a valid actual range if the input value was valid, otherwise an invalid range.

Returns:

Whether ioRange is valid.

Definition at line 88 of file SysCFUtils.cp.

Referenced by VerifyRange().

5.3.2.6 void PPx::CFUtils::VerifyIndex (CFIndex *inCount*, CFIndex & *ioIndex*)

Throws an exception if the supplied index is invalid for a CF container of the supplied length.

Also, if the supplied index is a logical index rather than actual index, it will be converted to the corresponding actual index.

Parameters:

inCount Number of items. 0 <= index <= inCount - 1

ioIndex In: Index value to validate Out: Valid index value

Definition at line 191 of file SysCFUtils.cp.

References MakeValidIndex(), and PPx_Throw_.

5.3.2.7 void PPx::CFUtils::VerifyInsertIndex (CFIndex *inCount*, CFIndex & *ioIndex*)

Throws an exception if an insertion index into a container is not valid.

Parameters:

inCount Number of items in container

ioIndex Index into container

Definition at line 238 of file SysCFUtils.cp.

References MakeInsertIndex(), and PPx_Throw_.

5.3.2.8 void PPx::CFUtils::VerifyRange (CFIndex *inMaxLength*, CFRange & *ioRange*)

Throws an exception if the supplied range is invalid for a CF container of the supplied length.

Also, if the supplied range is a logical range rather than actual range, it will be converted to the corresponding actual range.

Parameters:

- ***inMaxLength*** The maximum possible length of the range.
- ***ioRange*** On input: the actual or logical range to validate. On output: a valid actual range.

Definition at line 131 of file SysCFUtils.cp.

References MakeValidRange(), and PPx_Throw_.

5.4 PPx::Clipboard Namespace Reference

5.4.1 Detailed Description

Special instance of a DataScrap for the system Clipboard.

Functions

- DataScrap & Instance ()

  Returns a reference to the Clipboard scrap.

5.4.2 Function Documentation

5.4.2.1 DataScrap & PPx::Clipboard::Instance ()

Returns a reference to the Clipboard scrap.

The Clipboard is the standard Mac OS scrap for copying and pasting data between programs.

Returns:

Reference to the Clipboard scrap

Definition at line 234 of file PPxDataScrap.cp.

5.5 PPx::Debugging Namespace Reference

5.5.1 Detailed Description

Utility functions for debugging exceptions and signals.

Enumerations

- enum EDebugAction { **debugAction_Nothing** = 0, **debugAction_Alert** = 1, **debugAction_Debugger** = 2, **debugAction_Console** = 3 }

  Possible actions for Throw and Signal debugging.

Functions

- void **DebugException** (const Exception &)
- void **ExceptionAlert** (const Exception &)
- void SetDebugThrowAction (EDebugAction inAction)

  Sets debugging action to take when throwing an exception.

- void SetDebugSignalAction (EDebugAction inAction)

  Sets debugging action to take when raising a PPx signal.

- StringPtr CopyPStr (ConstStringPtr inSourceString, StringPtr outDestString, SInt16 inDestSize=sizeof(Str255))

  Copies a Pascal string.

- StringPtr LoadPStrFromCStr (Str255 outPStr, UInt8 inMaxChars, const char ∗inCStr)

  Copies contents of a C string into a Pascal string.

- StringPtr AppendPStr (Str255 ioBaseString, ConstStringPtr inAppendString, SInt16 inDestSize=sizeof(Str255))

  Append two Pascal strings.

5.5.2 Function Documentation

5.5.2.1 StringPtr PPx::Debugging::AppendPStr (Str255 *ioBaseString*, ConstStringPtr *inAppendString*, SInt16 *inDestSize* = sizeof(Str255))

Append two Pascal strings.

The first string becomes the combination of the first and second strings.

Parameters:

ioBaseString Base string. On exit, the appended string

inAppendString String to append

inDestSize Maximum size of appended string, including the length byte

Returns:

Pointer to the appended string

Definition at line 354 of file PPxDebugging.cp.

5.5.2.2 StringPtr PPx::Debugging::CopyPStr (ConstStringPtr *inSourceString*, StringPtr *outDestString*, SInt16 *inDestSize* = sizeof(Str255))

Copies a Pascal string.

Parameters:

inSourceString String to copy

outDestString Destination string in which to put copy

inDestSize Maximum size of destination string, including the length byte

Note:

inSourceString may be nil, in which case outDestString is set to a zero length string.

Definition at line 281 of file PPxDebugging.cp.

5.5.2.3 unsigned char ∗ PPx::Debugging::LoadPStrFromCStr (Str255 *outPStr*, UInt8 *inMaxChars*, const char ∗ *inCStr*)

Copies contents of a C string into a Pascal string.

Parameters:

outPStr Output Pascal string to which to copy

inMaxChars Maximum size of Pascal string

inCStr C String to copy

Returns:

Pointer to the Pascal string

Definition at line 315 of file PPxDebugging.cp.

5.5.2.4 void SetDebugSignalAction (EDebugAction *inAction*) `[inline]`

Sets debugging action to take when raising a PPx signal.

Parameters:

inAction Action to take. May be debugAction_Noting, debugAction_Alert, debugAction_Debugger, or debugAction_Console

Definition at line 226 of file PPxDebugging.h.

5.5.2.5 void SetDebugThrowAction (EDebugAction *inAction*) `[inline]`

Sets debugging action to take when throwing an exception.

Parameters:

inAction Action to take. May be debugAction_Noting, debugAction_Alert, debugAction_Debugger, or debugAction_Console

Definition at line 115 of file PPxDebugging.h.

5.6 PPx::EventUtils Namespace Reference

5.6.1 Detailed Description

Utility functions for working with CarbonEvents.

Functions

- void SetMenuCommandStatus (MenuCommand inCommand, bool inStatus)

  Enables or disables the menu item with a certain command ID.

- OSStatus ProcessCommandID (HICommand inCommand, UInt32 inKeyModifiers=attributes_None, UInt32 inMenuContext=attributes_None)

  Sends a CarbonEvent to process a command where the kind of event is a comamand ID.

- OSStatus UpdateCommandID (HICommand inCommand, UInt32 inMenuContext=attributes_None)

  Sends a CarbonEvent to update the status of a command where the kind of event is command ID.

- void PostCommandID (CommandIDT inCommandID, EventTargetRef inTarget=nil, EventQueueRef inQueueRef=nil, EventPriority inPriority=kEventPriorityStandard)

  Posts a Carbon Event for a specified command ID.

- OSStatus SendCommandID (CommandIDT inCommandID, EventTargetRef inTarget, OptionBits inOptions=options_None)

  Sends a Carbon Event for a specified command ID.

5.6.2 Function Documentation

5.6.2.1 void PPx::EventUtils::PostCommandID (CommandIDT *inCommandID*, EventTargetRef *inTarget* = nil, EventQueueRef *inQueueRef* = nil, EventPriority *inPriority* = kEventPriorityStandard)

Posts a Carbon Event for a specified command ID.

Parameters:

inCommandID Command ID number

inTarget Target for receiving the event, may be nil

inQueueRef Event queue in which to post the event, may be nil

inPriority Event priority level

Posting an event is asynchronous. The event is queued and execution returns immediately to the caller.

Definition at line 134 of file PPxEventUtils.cp.

References PPx::eventClass_ProcessCommand, and PPx::SysCarbonEvent::PostTo().

5.6.2.2 OSStatus PPx::EventUtils::ProcessCommandID (HICommand *inCommand*, UInt32 *inKeyModifiers* = attributes_None, UInt32 *inMenuContext* = attributes_None)

Sends a CarbonEvent to process a command where the kind of event is a comamand ID.

Parameters:

inCommand HICommand struct for the command to process

inKeyModifiers Keyboard modifier keys

inMenuContext Menu context of the command

The CarbonEvent parameters are the same as for the event (kEventClassCommand, kCommandProcess), but the class is eventClass_DoCommand and the kind is the commandID of the HICommand. The event gets sent to the user focus, so the event can propagate through the command chain.

This gives each command a unique event signature, and clients can process the command by installing a custom handler for that event.

Note:

This function should normally be called from the DoCommandProcess function of a CommandProcessDoer subclass to relay a generic command CarbonEvent as a CarbonEvent for a specific command ID.

Definition at line 56 of file PPxEventUtils.cp.

References PPx::eventClass_ProcessCommand, and PPx::SysCarbonEvent::SendTo().

5.6.2.3 OSStatus PPx::EventUtils::SendCommandID (CommandIDT *inCommandID*, EventTargetRef *inTarget*, OptionBits *inOptions* = options_None)

Sends a Carbon Event for a specified command ID.

Parameters:

inCommandID Command ID number

inTarget Target for receiving the event

inOptions Options for sending the event

Returns:

Status of handling the command event

Sending an event is synchronous. The target receives the event immediately and execution does not return to the caller until the event has been handled.

Definition at line 169 of file PPxEventUtils.cp.

References PPx::eventClass_ProcessCommand, and PPx::SysCarbonEvent::SendTo().

5.6.2.4 void PPx::EventUtils::SetMenuCommandStatus (MenuCommand *inCommand*, bool *inStatus*)

Enables or disables the menu item with a certain command ID.

Parameters:

inCommand Command ID number of a menu item

inStatus true = enable, false = disable

Definition at line 20 of file PPxEventUtils.cp.

5.6.2.5 OSStatus PPx::EventUtils::UpdateCommandID (HICommand *inCommand*, UInt32 *inMenuContext* = attributes_None)

Sends a CarbonEvent to update the status of a command where the kind of event is command ID.

Parameters:

inCommand HICommand struct for the command to process

inMenuContext Menu context of the command

The CarbonEvent parameters are the same as for the event (kEventClassCommand, k-EventCommandUpdateStatus), but the class is eventClass_UpdateCmdStatus and the kind is the commandID of the HICommand. The event gets sent to the user focus, so the event can propagate through the command chain.

This gives each command a unique event signature, and clients can update the command status by installing a custom handler for that event.

Note:

> This function should normally be called from th DoCommandUpdateStatus function of a CommandUpdateStatusDoer subclass to relay a generic update command CarbonEvent as a CarbonEvent to update a specific command ID.

Definition at line 102 of file PPxEventUtils.cp.

References PPx::eventClass_UpdateCmdStatus, and PPx::SysCarbonEvent::SendTo().

5.7 PPx::FindScrap Namespace Reference

5.7.1 Detailed Description

Special instance of a DataScrap for the system FindScrap.

Functions

- DataScrap & Instance ()

  Returns a reference to the FindScrap.

5.7.2 Function Documentation

5.7.2.1 DataScrap & PPx::FindScrap::Instance ()

Returns a reference to the FindScrap.

The FindScrap is a standard system scrap for remembering the text to search for in find operations.

Returns:

Reference to the FindScrap

The Find scrap is a standard feature of programs written with Cocoa

Definition at line 255 of file PPxDataScrap.cp.

5.8 PPx::FSUtils Namespace Reference

5.8.1 Detailed Description

Utility functions for working with files and folders.

Functions

- SInt32 CompareFSNames (const HFSUniStr255 &inNameOne, const HFSUniStr255 &inNameTwo)

  Compares file entity HFSUniStr255 names for sorting purposes.

- SInt32 CompareFSNames (const CFString &inNameOne, const CFString &inNameTwo)

  Compares file entity CFString names for sorting purposes.

- bool FSNamesAreEqual (const HFSUniStr255 &inNameOne, const HFSUniStr255 &inNameTwo)

  Compares file entity HFSUniStr255 names for equivalence.

- bool FSNamesAreEqual (const CFString &inNameOne, const CFString &inNameTwo)

  Compares file entity CFString names for equivalence.

- void StringToHFSUniStr (const CFString &inString, HFSUniStr255 &outHFSUniStr)

  Converts a CFString to a HFS unicode string.

5.8.2 Function Documentation

5.8.2.1 SInt32 PPx::FSUtils::CompareFSNames (const CFString & *inNameOne*, const CFString & *inNameTwo*)

Compares file entity CFString names for sorting purposes.

Uses the same sorting logic as the Mac OS X Finder

Parameters:

inNameOne Right hand side operand

inNameTwo Left hand side operand

Returns:

Result of comparing inNameOne to inNameTwo

Return values:

Zero Names are equivalent

$>$ 0 inNameOne is greater than inNameTwo

$<$ 0 inNameOne is less than inNameTwo

Definition at line 115 of file PPxFSUtils.cp.

References CompareFSNames(), and StringToHFSUniStr().

5.8.2.2 SInt32 PPx::FSUtils::CompareFSNames (const HFSUniStr255 & *inNameOne*, const HFSUniStr255 & *inNameTwo*)

Compares file entity HFSUniStr255 names for sorting purposes.

Uses the same sorting logic as the Mac OS X Finder

Parameters:

inNameOne Right hand side operand

inNameTwo Left hand side operand

Returns:

Result of comparing inNameOne to inNameTwo

Return values:

Zero Names are equivalent

$>$ 0 inNameOne is greater than inNameTwo

$<$ 0 inNameOne is less than inNameTwo

Definition at line 82 of file PPxFSUtils.cp.

References PPx_ThrowIfOSError_.

Referenced by CompareFSNames().

5.8.2.3 bool PPx::FSUtils::FSNamesAreEqual (const CFString & *inNameOne*, const CFString & *inNameTwo*)

Compares file entity CFString names for equivalence.

Parameters:

inNameOne Right hand side operand

inNameTwo Left hand side operand

Returns:

Whether the two strings are equivalent

Note:

Comparing specifically for equality may be significantly faster than the general comparison performed by FSUtils::CompareFSNames

Definition at line 175 of file PPxFSUtils.cp.

References FSNamesAreEqual(), and StringToHFSUniStr().

5.8.2.4 bool PPx::FSUtils::FSNamesAreEqual (const HFSUniStr255 & *inNameOne*, const HFSUniStr255 & *inNameTwo*)

Compares file entity HFSUniStr255 names for equivalence.

Parameters:

inNameOne Right hand side operand

inNameTwo Left hand side operand

Returns:

Whether the two strings are equivalent

Note:

Comparing specifically for equality may be significantly faster than the general comparison performed by FSUtils::CompareFSNames

Definition at line 144 of file PPxFSUtils.cp.

References PPx_ThrowIfOSError_.

Referenced by FSNamesAreEqual().

5.8.2.5 void PPx::FSUtils::StringToHFSUniStr (const CFString & *inString*, HFSUniStr255 & *outHFSUniStr*)

Converts a CFString to a HFS unicode string.

Parameters:

inString Input CFString

outHFSUniStr String converted to HFSUniStr255

Definition at line 198 of file PPxFSUtils.cp.

References PPx::CFString::GetLength(), PPx::CFString::GetSubstring(), and PPx_Throw_.

Referenced by CompareFSNames(), and FSNamesAreEqual().

5.9 PPx::MenuDebugStr Namespace Reference

5.9.1 Detailed Description

Debugging utility functions for displaying information in the menu bar.

Stopping at a breakpoint in the source debugger or posting an alert may disturb the state of a program by obscuring windows or generating window deactivate/activate events. This may hamper debugging of code dealing with window drawing and event handling. In such cases, you can use these routines to display debugging information in the menu bar.

Functions

- void Display (ConstStringPtr inString, unsigned long inDelay=120)

  Temporarily displays a Pascal string in the menu bar.

- void Display (const char ∗inCString, unsigned long inDelay=120)

  Temporarily displays a Pascal string in the menu bar.

- void Display (long inNumber, unsigned long inDelay=120)

  Temporarily displays a number in the menu bar.

5.9.2 Function Documentation

5.9.2.1 void PPx::MenuDebugStr::Display (long *inNumber*, unsigned long *inDelay* = 120)

Temporarily displays a number in the menu bar.

Parameters:

inNumber Signed integer number

inDelay Duration in ticks for displaying the string

Definition at line 455 of file PPxDebugging.cp.

References Display().

5.9.2.2 void PPx::MenuDebugStr::Display (const char ∗ *inCString*, unsigned long *inDelay* = 120)

Temporarily displays a Pascal string in the menu bar.

Parameters:

inCString C string

inDelay Duration in ticks for displaying the string

Definition at line 424 of file PPxDebugging.cp.

References Display().

5.9.2.3 void PPx::MenuDebugStr::Display (ConstStringPtr *inString*, unsigned long *inDelay* = 120)

Temporarily displays a Pascal string in the menu bar.

Parameters:

inString Pascal string

inDelay Duration in ticks for displaying the string

Definition at line 393 of file PPxDebugging.cp.

Referenced by Display().

5.10 PPx::NavServices Namespace Reference

5.10.1 Detailed Description

Utility functions for displaying NavServices dialogs.

Functions

- void GetDefaultCreationOptions (NavDialogCreationOptions &outOptions)

  Passes back the default options for creating NavServices dialogs.

- void AskSaveChanges (NavEventResponder &inResponder, NavAskSaveChangesAction inAction, const NavDialogCreationOptions &inOptions)

  Displays dialog asking whether to save changes before closing or quitting.

- void AskSaveChanges (NavEventResponder &inResponder, NavAskSaveChangesAction inAction, WindowRef inParentWindow)

  Displays dialog asking whether to save changes before closing or quitting.

- void AskDiscardChanges (NavEventResponder &inResponder, const NavDialogCreationOptions &inOptions)

  Displays dialog asking user if it is OK to discard changes to a document.

- void AskDiscardChanges (NavEventResponder &inResponder, WindowRef inParentWindow)

  Displays dialog asking user if it is OK to discard changes to a document.

- void AskReviewDocuments (NavEventResponder &inResponder, UInt32 inDocumentCount, const NavDialogCreationOptions &inOptions)

  Displays dialog asking how to handle multiple unssave documents when quitting an application.

- void AskReviewDocuments (NavEventResponder &inResponder, UInt32 inDocumentCount)

  Displays dialog asking how to handle multiple unssave documents when quitting an application.

- void AskDesignateFile (NavEventResponder &inResponder, OSType inFileType, OSType inFileCreator, const NavDialogCreationOptions &inOptions)

  Display NavServices dialog for designating a new file.

- void AskDesignateFile (NavEventResponder &inResponder, OSType inFileType, CFStringRef inDefaultName, WindowRef inParentWindow)

Display NavServices dialog for designating a new file.

- void AskGetFile (NavEventResponder &inResponder, NavTypeListHandle inTypeList, const NavDialogCreationOptions &inOptions)

  Display NavServices dialog for getting a file to open.

- void AskGetFile (NavEventResponder &inResponder, NavTypeListHandle inTypeList)

  Display NavServices dialog for getting a file to open.

- void AskChooseFile (NavEventResponder &inResponder, NavTypeListHandle inTypeList, const NavDialogCreationOptions &inOptions)

  Display NavServices dialog for choosing a file on which to perform an operation.

- void AskChooseFile (NavEventResponder &inResponder, NavTypeListHandle inTypeList)

  Display NavServices dialog for choosing a file on which to perform an operation.

5.10.2 Function Documentation

5.10.2.1 void PPx::NavServices::AskChooseFile (NavEventResponder & *inResponder*, NavTypeListHandle *inTypeList*)

Display NavServices dialog for choosing a file on which to perform an operation.

Parameters:

inResponder Object for handling callbacks

inTypeList List of file types for file to choose

Definition at line 565 of file PPxNavServices.cp.

References AskChooseFile(), and GetDefaultCreationOptions().

5.10.2.2 void PPx::NavServices::AskChooseFile (NavEventResponder & *inResponder*, NavTypeListHandle *inTypeList*, const NavDialogCreationOptions & *inOptions*)

Display NavServices dialog for choosing a file on which to perform an operation.

Parameters:

inResponder Object for handling callbacks

inTypeList List of file types for file to choose

inOptions Options for making the dialog

Note:

Use AskGetFile for selecting a file to open. The system presents different dialogs for ”get” and ”choose”.

Definition at line 533 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_.

Referenced by AskChooseFile().

5.10.2.3 void PPx::NavServices::AskDesignateFile (NavEventResponder & *inResponder*, OSType *inFileType*, CFStringRef *inDefaultName*, WindowRef *inParentWindow*)

Display NavServices dialog for designating a new file.

Generally referred to as ”Put File”

Parameters:

inResponder Object for handling callbacks

inFileType Type of file

inDefaultName Default name for file

inParentWindow Parent window if dialog is a sheet

Definition at line 433 of file PPxNavServices.cp.

References AskDesignateFile(), and GetDefaultCreationOptions().

5.10.2.4 void PPx::NavServices::AskDesignateFile (NavEventResponder & *inResponder*, OSType *inFileType*, OSType *inFileCreator*, const NavDialogCreationOptions & *inOptions*)

Display NavServices dialog for designating a new file.

Generally referred to as ”Put File”

Parameters:

inResponder Object for handling callbacks

inFileType Type of file

inFileCreator Creator code for file

inOptions Options for making the dialog

Definition at line 395 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_, and PPx_ThrowOSError_.

Referenced by AskDesignateFile().

5.10.2.5 void PPx::NavServices::AskDiscardChanges (NavEventResponder & *inResponder*, WindowRef *inParentWindow*)

Displays dialog asking user if it is OK to discard changes to a document.

Usually called before performing a ”Revert” operation.

Parameters:

inResponder Object for handling callbacks

inParentWindow Parent window if dialog is a sheet

Definition at line 294 of file PPxNavServices.cp.

References AskDiscardChanges(), PPx::CFObject< CFStringRef >::AttachRef(), GetDefaultCreationOptions(), and PPx::CFObject< CFStringRef >::UseRef().

5.10.2.6 void PPx::NavServices::AskDiscardChanges (NavEventResponder & *inResponder*, const NavDialogCreationOptions & *inOptions*)

Displays dialog asking user if it is OK to discard changes to a document.

Usually called before performing a ”Revert” operation.

Parameters:

inResponder Object for handling callbacks

inOptions Options for making the dialog

Note:

You must fill in the saveFileName field of inOptions

Definition at line 262 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_, and PPx_ThrowOSError_.

Referenced by AskDiscardChanges().

5.10.2.7 void PPx::NavServices::AskGetFile (NavEventResponder & *inResponder*, NavTypeListHandle *inTypeList*)

Display NavServices dialog for getting a file to open.

Parameters:

- ***inResponder*** Object for handling callbacks
- ***inTypeList*** List of file types for file to choose

Definition at line 504 of file PPxNavServices.cp.

References AskGetFile(), and GetDefaultCreationOptions().

5.10.2.8 void PPx::NavServices::AskGetFile (NavEventResponder & *inResponder*, NavTypeListHandle *inTypeList*, const NavDialogCreationOptions & *inOptions*)

Display NavServices dialog for getting a file to open.

Parameters:

- ***inResponder*** Object for handling callbacks
- ***inTypeList*** List of file types for file to choose
- ***inOptions*** Options for making the dialog

Definition at line 469 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_, and PPx_ThrowOSError_.

Referenced by AskGetFile().

5.10.2.9 void PPx::NavServices::AskReviewDocuments (NavEventResponder & *inResponder*, UInt32 *inDocumentCount*)

Displays dialog asking how to handle multiple unssave documents when quitting an application.

Parameters:

- ***inResponder*** Object for handling callbacks
- ***inDocumentCount*** Number of unsaved documents needing review

Definition at line 367 of file PPxNavServices.cp.

References AskReviewDocuments(), and GetDefaultCreationOptions().

5.10.2.10 void PPx::NavServices::AskReviewDocuments (NavEventResponder & *inResponder*, UInt32 *inDocumentCount*, const NavDialogCreationOptions & *inOptions*)

Displays dialog asking how to handle multiple unssave documents when quitting an application.

Parameters:

- ***inResponder*** Object for handling callbacks
- ***inDocumentCount*** Number of unsaved documents needing review
- ***inOptions*** Options for making the dialog

Definition at line 333 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_, and PPx_ThrowOSError_.

Referenced by AskReviewDocuments().

5.10.2.11 void PPx::NavServices::AskSaveChanges (NavEventResponder & *inResponder*, NavAskSaveChangesAction *inAction*, WindowRef *inParentWindow*)

Displays dialog asking whether to save changes before closing or quitting.

Parameters:

- ***inResponder*** Object for handling callbacks
- ***inAction*** Action code
- ***inParentWindow*** Parent window if dialog is a sheet

Action code is either kNavSaveChangesClosingDocument or kNavSaveChanges-QuittingApplication

Definition at line 228 of file PPxNavServices.cp.

References AskSaveChanges(), and GetDefaultCreationOptions().

5.10.2.12 void PPx::NavServices::AskSaveChanges (NavEventResponder & *inResponder*, NavAskSaveChangesAction *inAction*, const NavDialogCreationOptions & *inOptions*)

Displays dialog asking whether to save changes before closing or quitting.

Parameters:

- ***inResponder*** Object for handling callbacks
- ***inAction*** Action code
- ***inOptions*** Options for making the dialog

Action code is either kNavSaveChangesClosingDocument or kNavSaveChanges-QuittingApplication

Definition at line 191 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_, and PPx_ThrowOSError_.

Referenced by AskSaveChanges().

5.10.2.13 void PPx::NavServices::GetDefaultCreationOptions (NavDialogCreationOptions & *outOptions*)

Passes back the default options for creating NavServices dialogs.

Parameters:

outOptions Default options for creating NavServices dialogs

Definition at line 168 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_.

Referenced by AskChooseFile(), AskDesignateFile(), AskDiscardChanges(), AskGetFile(), AskReviewDocuments(), and AskSaveChanges().

5.11 PPx::PrimaryBundle Namespace Reference

5.11.1 Detailed Description

Utility functions for working with the primary bundle for a program.

Functions

- CFBundle & Instance ()

  Returns a reference to the primary bundle object.

- void Set (CFBundleRef inBundle)

  Specify the primary bundle.

- CFData GetResourceData (CFStringRef inResourceName, CFStringRef inResourceType, CFStringRef inSubDirName=nil)

  Returns the data from a named resource file.

- CFTypeRef GetResourceProperty (CFStringRef inPropertyName, CFStringRef inResourceName, CFStringRef inResourceType, CFStringRef inSubDirName=nil)

  Returns a CFTypeRef for a property of a resource.

- CFString GetLocalizedString (CFStringRef inKey)

  Returns the localized string for a key using the default localized string file.

- CFString GetLocalizedString (CFStringRef inKey, CFStringRef inTable)

  Returns the localized string for a key using a specified strings table file.

- CFString GetLocalizedString (CFStringRef inKey, CFStringRef inDefaultValue, CFStringRef inTable)

  Returns the localized string for a key using a specified strings table file and default value.

5.11.2 Function Documentation

5.11.2.1 CFString PPx::PrimaryBundle::GetLocalizedString (CFStringRef *inKey*, CFStringRef *inDefaultValue*, CFStringRef *inTable*)

Returns the localized string for a key using a specified strings table file and default value.

Parameters:

inKey Key for string

inDefaultValue String to return if key is not in table

inTable Name of .strings file to use as the look-up table

Returns:

Localized string for key

Definition at line 169 of file PPxPrimaryBundle.cp.

References PPx::CFBundle::GetLocalizedString(), and Instance().

5.11.2.2 CFString PPx::PrimaryBundle::GetLocalizedString (CFStringRef *inKey*, CFStringRef *inTable*)

Returns the localized string for a key using a specified strings table file.

Parameters:

inKey Key for string

inTable Name of .strings file to use as the look-up table

Returns:

Localized string for key

Returns the key as the localized string if the key is not in the look-up table

Definition at line 148 of file PPxPrimaryBundle.cp.

References PPx::CFBundle::GetLocalizedString(), and Instance().

5.11.2.3 CFString PPx::PrimaryBundle::GetLocalizedString (CFStringRef *inKey*)

Returns the localized string for a key using the default localized string file.

Parameters:

inKey Key for string

Returns:

Localized string for key

Returns the key as the localized string if the key is not in the Localized.strings file

Definition at line 126 of file PPxPrimaryBundle.cp.

References PPx::CFBundle::GetLocalizedString(), and Instance().

5.11.2.4 CFData PPx::PrimaryBundle::GetResourceData (CFStringRef *inResourceName*, CFStringRef *inResourceType*, CFStringRef *inSubDirName* = nil)

Returns the data from a named resource file.

Parameters:

inResourceName Name of the resource

inResourceType Type of the resource

inSubDirName Subdirectory of bundle in which to start search

Returns:

CFData object containing the resource data

Bundled programs typically store resources as separate files within subdirectories of the bundle. To get the data in a file called MyPicture.jpg, you would pass ”MyPicture” and the resource name and ”jpg” as the resource type. Note that the ”dot” in the file name is not included in the name or type.

You may pass nil for the subdirectory name. If so, the system uses its default search algorithm, which tries to get the correct localized version of the resource. Read the Mac OS documentation on bundles and url access for more information.

Definition at line 70 of file PPxPrimaryBundle.cp.

References Instance().

5.11.2.5 CFTypeRef PPx::PrimaryBundle::GetResourceProperty (CFStringRef *inPropertyName*, CFStringRef *inResourceName*, CFStringRef *inResourceType*, CFStringRef *inSubDirName* = nil)

Returns a CFTypeRef for a property of a resource.

Parameters:

inPropertyName Name of the property

inResourceName Name of the resource

inResourceType Type of the resource

inSubDirName Subdirectory of bundle in which to start search

Returns:

CFTypeRef for the property value

See the comments for BundleUtils::GetResourceData for information about resource names, resource types, and subdirectories.

The kind of CFTypeRef returned depends on the property. See <CFURLAccess.h> and Apple's Core Foundation documentation for information about resource properties.

Definition at line 100 of file PPxPrimaryBundle.cp.

References Instance().

5.11.2.6 CFBundle & PPx::PrimaryBundle::Instance ()

Returns a reference to the primary bundle object.

Note:

Implements the Singleton design pattern by using a static local variable. The default value is the application's main bundle.

Definition at line 19 of file PPxPrimaryBundle.cp.

Referenced by GetLocalizedString(), GetResourceData(), GetResourceProperty(), Set(), and PPx::Signature::SetSignatureFromBundle().

5.11.2.7 void PPx::PrimaryBundle::Set (CFBundleRef *inBundle*)

Specify the primary bundle.

Parameters:

inBundle Bundle reference to designate as the primary bundle

Note:

Plug-ins or other external code modules may wish to set the primary bundle to their bundle in order access their resources rather than those of the host application.

Definition at line 40 of file PPxPrimaryBundle.cp.

References PPx::CFObject< CFBundleRef >::AttachRef(), and Instance().

5.12 PPx::Registrar Namespace Reference

5.12.1 Detailed Description

Implements new-by-name creation of Persistent objects.

Typedefs

- typedef Persistent ∗(∗ **CreatorFunction**)()

Functions

- void RegisterClass (const CFString &inClassName, CreatorFunction inCreatorFunc)

  Registers a class name and creator function pair.

- template<class TSubClass> void RegisterClass (const CFString &inClassName)

  Template function to register a class.

- void UnregisterClass (const CFString &inClassName)

  Removes registration for a class.

- bool IsRegistered (const CFString &inClassName)

  Tests if a class is already registered.

- Persistent ∗ CreateObject (const CFString &inClassName)

  Create an object of the class with the specified name.

- template<class TSubClass> Persistent ∗ CreateNewObject ()

  Creates a new object of the class specified by the template parameter.

5.12.2 Function Documentation

5.12.2.1 template<class TSubClass> Persistent ∗ PPx::Registrar::CreateNewObject ()

Creates a new object of the class specified by the template parameter.

Returns:

Pointer to newly created object

Definition at line 77 of file PPxRegistrar.h.

5.12.2.2 Persistent ∗ PPx::Registrar::CreateObject (const CFString & *inClassName*)

Create an object of the class with the specified name.

Parameters:

inClassName Class name

Returns:

Pointer to newly created object

Definition at line 83 of file PPxRegistrar.cp.

References PPx_Throw_.

5.12.2.3 bool PPx::Registrar::IsRegistered (const CFString & *inClassName*)

Tests if a class is already registered.

Parameters:

inClassName Class name

Returns:

Whether or not the class is registered

Definition at line 65 of file PPxRegistrar.cp.

5.12.2.4 template<class TSubClass> void PPx::Registrar::RegisterClass (const CFString & *inClassName*)

Template function to register a class.

The template parameter is the class, which must be a subclass of Persistent

Parameters:

inClassName Class name

Definition at line 61 of file PPxRegistrar.h.

References RegisterClass().

5.12.2.5 void PPx::Registrar::RegisterClass (const CFString & *inClassName*, CreatorFunction *inCreatorFunc*)

Registers a class name and creator function pair.

Parameters:

inClassName Name of class to register

inCreatorFunc Function which creates an object of the class

Definition at line 32 of file PPxRegistrar.cp.

Referenced by RegisterClass().

5.12.2.6 void PPx::Registrar::UnregisterClass (const CFString & *inClassName*)

Removes registration for a class.

Parameters:

inClassID Name of class to unregister

Definition at line 48 of file PPxRegistrar.cp.

5.13 PPx::Serializer Namespace Reference

5.13.1 Detailed Description

Functions for reading and wwiting state information for Peristent objects to flattened data structures.

Functions

- std::auto_ptr< Persistent > DescriptorsToObjects (ObjectDescriptorList &ioDescriptors)

  Creates Persistent objects from a list of object descriptors.

- void ObjectsToDescriptors (const Persistent ∗inRootObject, ObjectDescriptorList &outDescriptors)

  Stores Persistent objects as a list of object descriptors.

5.13.2 Function Documentation

5.13.2.1 std::auto_ptr< Persistent > PPx::Serializer::DescriptorsToObjects (ObjectDescriptorList & *ioDescriptors*)

Creates Persistent objects from a list of object descriptors.

Parameters:

ioDescriptors List of object descriptors

Returns:

The first Persistent object created from the list

An object descriptor contains the class name and a data dictionary of state information for a persistent object. This function iterates over a list of object descriptors, creating and initializing a Persistent object based on each descriptor.

The process has three steps: (1) contruction via new and the default constructor; (2) call to InitPersistent with the data dictionary of state information; (3) call to FinishInitPersistent.

The first object should be the root of an object hierarchy. The return value is a std::auto_ptr containing a pointer to the first object, meaning the caller receives ownership of it.

This reverses the processes of the ObjectsToDescriptors function.

Definition at line 38 of file PPxSerializer.cp.

5.13.2.2 void PPx::Serializer::ObjectsToDescriptors (const Persistent ∗ *inRootObject*, ObjectDescriptorList & *outDescriptors*)

Stores Persistent objects as a list of object descriptors.

Parameters:

inRootObject Persistent object to store

outDescriptors List of object descriptors

This function puts the Persistent object passes as a parameter into a first in, first out deque. It then begins a loop were it removes the first object from the deque, creates an object descriptor with a data dictionary, adds the descriptor to a list, then tells the object to write its state to the data dictionary.

In the course of writing its state, an object may request that other Persistent objects write their state. Such objects are added to the deque. The loop continues until the deque is empty. In this way, an entire object hierarchy is flattened into a list of object descriptors.

Definition at line 105 of file PPxSerializer.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), PPx::ObjectDescriptor::className, PPx::Persistent::GetClassName(), PPx::ObjectDescriptor::objectPtr, PPx::ObjectDescriptor::objectState, PPx::AutoRefCount< KeyDataMap >::Reset(), PPx::ObjectDescriptor::storageID, and PPx::Persistent::WritePersistent().

5.14 PPx::Signature Namespace Reference

5.14.1 Detailed Description

Sets/Gets the four-character code signature for the program.

In addition to its common use as the creator code for files, the Toolbox also uses a signature to identify user-supplied data stored by the OS. For example, properties associated with Controls.

If you don't call Set, the Get function will try to get the signature from info.plist of the PrimaryBundle

Functions

- void SetSignatureFromBundle ()

  Tries to get signature for the program from its primary bundle.

- void Set (OSType inSignature)

  Sets the signature for the program.

- OSType Get ()

  Returns the signature for the program.

Variables

- const OSType **signature_Default** = 'PPxS'
- const CFString **propListKey_Signature** = CFSTR("CFBundleSignature")
- OSType **sSignature** = 0

5.14.2 Function Documentation

5.14.2.1 OSType PPx::Signature::Get ()

Returns the signature for the program.

Returns:

Signature for the program

If you don't call Set, we try to get the signature from the info.plist of the PrimaryBundle

Definition at line 50 of file PPxSignature.cp.

References SetSignatureFromBundle().

5.14.2.2 void PPx::Signature::Set (OSType *inSignature*)

Sets the signature for the program.

Parameters:

inSignature Signature for the program

Definition at line 32 of file PPxSignature.cp.

5.15 PPx::StreamUtils Namespace Reference

5.15.1 Detailed Description

Utility functions for working with the standard IOStream library.

Functions

- void WriteLinesOfText (std::ostream &inOutputStream, const char ∗inTextPtr, long inTextLength, char inLneEndChar)

  Writes lines of text to an output stream.

5.15.2 Function Documentation

5.15.2.1 void PPx::StreamUtils::StreamUtils::WriteLinesOfText (std::ostream & *inOutputStream*, const char ∗ *inTextPtr*, long *inTextLength*, char *inLineEndChar*)

Writes lines of text to an output stream.

Parameters:

inOutputStream Output stream

inTextPtr Pointer to text

inTextLength Lenght of text

inLineEndChar Character used for line endings in text

Lines of text end with an inLineEndChar. Text for a line is streamed, followed by a std::endl manipulator, which writes a new line and flushes the stream.

This function is useful when writing large blocks of text to a buffered stream. Writing the whole block of text at once could exceed the buffer capacity, resulting in lost characters.

Definition at line 154 of file PPxStreamUtils.cp.

5.16 PPx::SysCreateView Namespace Reference

5.16.1 Detailed Description

Functions for creating system views.

Functions

- HIViewRef BevelButton (CFStringRef inTitle, ControlBevelThickness inThickness, ControlBevelButtonBehavior inButtonBehavior, const ControlButtonContentInfo &inButtonContent, SInt16 inMenuID, ControlBevelButtonMenuBehavior inMenuBehavior, ControlBevelButtonMenuPlacement inMenuPlacement)

  Creates a system bevel button.

- HIViewRef ChasingArrows ()

  Creates a system chasing arrows indicator.

- HIViewRef CheckBox (CFStringRef inTitle, SInt32 inInitialValue, bool inAutoToggle)

  Creates a system check box control.

- HIViewRef CheckBoxGroupBox (CFStringRef inTitle, SInt32 inInitialValue, bool inIsPrimary, bool inAutoToggle)

  Creates a system check box group box.

- HIViewRef ClockControl (ControlClockType inClockType, ControlClockFlags inClockFlags)

  Creates a system clock control.

- HIViewRef ComboBox (const HIRect &inBounds, CFStringRef inDefaultText, const ControlFontStyleRec ∗inStyle, CFArrayRef inValueList, OptionBits inAttributes)

  A system combo box, which combines an edit field and list of values.

- HIViewRef DisclosureButton (SInt32 inInitialValue, bool inAutoToggle)

  Creates a system disclosure button.

- HIViewRef DisclosureTriangle (ControlDisclosureTriangleOrientation inOrientation, CFStringRef inTitle, SInt32 inInitialValue, bool inDrawTitle, bool inAutoToggle)

  Creates a system disclosur triangle.

- HIViewRef EditTextControl (CFStringRef inText, bool inIsPassword, bool inUseInlineInput, const ControlFontStyleRec ∗inStyle)

  Creates a system edit text control.

- HIViewRef EditUnicodeText (CFStringRef inText, bool inIsPassword, const ControlFontStyleRec ∗inStyle)

  Creates a system edit unicode text control.

- HIViewRef IconControl (const ControlButtonContentInfo &inContent, bool inDontTrack)

  Creates a system icon control.

- HIViewRef IconPushButton (CFStringRef inTitle, const ControlButtonContentInfo &inContent, ControlPushButtonIconAlignment inAlignment)

  Creates a system icon push button control.

- HIViewRef ImageView (CGImageRef inImage)

  Creates a system image view.

- HIViewRef ImageWell (const ControlButtonContentInfo &inContent)

  Creates a system image well.

- HIViewRef ListBox (bool inAutoSize, SInt16 inRowCount, SInt16 inColCount, bool inHorizScroll, bool inVertScroll, SInt16 inCellHeight, SInt16 inCellWidth, bool inHasGrowSpace, const ListDefSpec &inListDef)

  Creates a system list box.

- HIViewRef LittleArrows (SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, SInt32 inIncrement)

  Creates a system little arrows control.

- HIViewRef PictureControl (SInt16 inPictResID, PicHandle inPictureHandle, bool inDontTrack)

  Creates a system picture control.

- HIViewRef Placard ()

  Creates a system placard.

- HIViewRef PopupArrow (ControlPopupArrowOrientation inOrientation, ControlPopupArrowSize inArrowSize)

  Creates a system popup arrow control.

- HIViewRef PopupButton (CFStringRef inTitle, SInt32 inMenuID, bool inHasVariableWidth, SInt16 inTitleWidth, SInt16 inTitleJust, Style inTitleStyle)

  Creates a system popup button.

- HIViewRef PopupGroupBox (CFStringRef inTitle, bool inIsPrimary, SInt32 inMenuID, bool inHasVariableWidth, SInt16 inTitleWidth, SInt16 inTitleJust, Style inTitleStyle)

  Creates a system popup group box.

- HIViewRef ProgressBar (SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, bool inIsIndeterminate)

  Creates a system jprogress bar.

- HIViewRef PushButton (CFStringRef inTitle)

  Creates a system push button.

- HIViewRef RadioButton (CFStringRef inTitle, SInt32 inInitialValue, bool inAutoToggle)

  Creates a system radio button.

- HIViewRef RadioGroup ()

  Creates a system radio group.

- HIViewRef RelevanceBar (SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue)

  Creates a system relevance bar.

- HIViewRef RoundButton (ControlRoundButtonSize inButtonSize, const ControlButtonContentInfo &inContent)

  Creates a system round button.

- HIViewRef ScrollBar (SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, SInt32 inViewSize, bool inHasLiveTracking, ControlActionUPP inLiveTrackingUPP)

  Creates a system scroll bar.

- HIViewRef ScrollView (OptionBits inOptions)

  Creates a system scroll view.

- HIViewRef SeparatorLine ()

  Creates a system separator line.

- HIViewRef Slider (SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, ControlSliderOrientation inOrientation, UInt16 inTickMarksCount, bool inHasLiveTracking, ControlActionUPP inLiveTrackingUPP)

  Creates a system slider control.

- HIViewRef StaticText (CFStringRef inText, const ControlFontStyleRec ∗inStyle)

  Creates a system static text control.

- HIViewRef TabView (ControlTabSize inTabSize, ControlTabDirection inTabDirection, UInt16 inTabCount, const ControlTabEntry ∗inTabEntries)

  Creates a system tab view.

- HIViewRef TextGroupBox (CFStringRef inTitle, bool inIsPrimary)

  Creates a system text group box.

- HIViewRef WindowHeader (bool inIsListHeader)

  Creates a system window header.

5.16.2 Function Documentation

5.16.2.1 HIViewRef PPx::SysCreateView::BevelButton (CFStringRef *inTitle*, ControlBevelThickness *inThickness*, ControlBevelButtonBehavior *inButtonBehavior*, const ControlButtonContentInfo & *inButtonContent*, SInt16 *inMenuID*, ControlBevelButtonMenuBehavior *inMenuBehavior*, ControlBevelButtonMenuPlacement *inMenuPlacement*)

Creates a system bevel button.

Parameters:

inTitle Text title for button

inThickness Thickness of the beveled edges

inButtonBehavior How button behaves when clicked

inButtonContent Kind of button content

inMenuID Menu ID for popup menu

inMenuBehavior How menu behaves when item is selected

inMenuPlacement Placement of menu glyph within button

Returns:

HIViewRef for bevel button

Definition at line 60 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.2 HIViewRef PPx::SysCreateView::ChasingArrows ()

Creates a system chasing arrows indicator.

Returns:
HIViewRef for chasing arrows

Definition at line 92 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.3 HIViewRef PPx::SysCreateView::CheckBox (CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inAutoToggle*)

Creates a system check box control.

Parameters:

inTitle Title of check box

inInitialValue State of check box (0 = unchecked, 1 = checked, 2 = mixed)

inAutoToggle Whether box is checked/unchecked automatically when clicked

Returns:
HIViewRef for check box

Definition at line 118 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.4 HIViewRef PPx::SysCreateView::CheckBoxGroupBox (CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inIsPrimary*, bool *inAutoToggle*)

Creates a system check box group box.

Parameters:

inTitle Title of check box

inInitialValue State of check box (0 = unchecked, 1 = checked, 2 = mixed)

inIsPrimary Group box kind (true = primary, false = secondary)

inAutoToggle Whether box is checked/unchecked automatically when clicked

Definition at line 147 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.5 HIViewRef PPx::SysCreateView::ClockControl (ControlClockType *inClockType*, ControlClockFlags *inClockFlags*)

Creates a system clock control.

Parameters:

inClockType Kind of clock (time or date)

inClockFlags Clock options

Returns:

HIView for clock control

Definition at line 176 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.6 HIViewRef PPx::SysCreateView::ComboBox (const HIRect & *inBounds*, CFStringRef *inDefaultText*, const ControlFontStyleRec ∗ *inStyle*, CFArrayRef *inValueList*, OptionBits *inAttributes*)

A system combo box, which combines an edit field and list of values.

Parameters:

inBounds Bounding box

inDefaultText Initial text in the edit field

inStyle Text style

inValueList List of value to diplay as choices

inAttributes Option flags

Returns:

HIViewRef for combo box

Definition at line 205 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.7 HIViewRef PPx::SysCreateView::DisclosureButton (SInt32 *inInitialValue*, bool *inAutoToggle*)

Creates a system disclosure button.

Parameters:

inInitialValue kControlDisclosureButtonClosed or kControlDisclosureButton-Disclosed

inAutoToggle Whether button automatically toggles state when clicked

Returns:

HIViewRef for disclosure button

Definition at line 236 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.8 HIViewRef PPx::SysCreateView::DisclosureTriangle (ControlDisclosureTriangleOrientation *inOrientation*, CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inDrawTitle*, bool *inAutoToggle*)

Creates a system disclosur triangle.

Parameters:

inOrientation Direction triangle points when closed

inTitle Title for disclosure triangle

inInitialValue 0 = closed, 1 = open

inDrawTitle Whether to draw the title

inAutoToggle Whether the triangle automaticallys toggles beteeen open/closed when clicked

Returns:

HIViewRef for disclosure triangle

Definition at line 266 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.9 HIViewRef PPx::SysCreateView::EditTextControl (CFStringRef *inText*, bool *inIsPassword*, bool *inUseInlineInput*, const ControlFontStyleRec ∗ *inStyle*)

Creates a system edit text control.

Parameters:

inText Initial text in edit field

inIsPassword Whether the field is a for password

inUseInlineInput Whether to use inline input

inStyle Text style

Returns:

HIViewRef for edit text control

Definition at line 298 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.10 HIViewRef PPx::SysCreateView::EditUnicodeText (CFStringRef *inText*, bool *inIsPassword*, const ControlFontStyleRec ∗ *inStyle*)

Creates a system edit unicode text control.

Parameters:

inText Initial text in edit field

inIsPassword Whether the field is for a password

inStyle Font style for the text

Returns:

HIViewRef for edit unicode text control

Definition at line 328 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.11 HIViewRef PPx::SysCreateView::IconControl (const ControlButtonContentInfo & *inContent*, bool *inDontTrack*)

Creates a system icon control.

Parameters:

inContent Content of icon control

inDontTrack Whether to not track mouse downs in the control

Returns:

HIViewRef for icon control

Definition at line 355 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.12 HIViewRef PPx::SysCreateView::IconPushButton (CFStringRef *inTitle*, const ControlButtonContentInfo & *inContent*, ControlPushButtonIconAlignment *inAlignment*)

Creates a system icon push button control.

Parameters:

inTitle Title for button

inContent Content of icon push button

inAlignment Alignement of icon in the button

Returns:

HIViewRef for icon push button

Definition at line 387 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.13 HIViewRef PPx::SysCreateView::ImageView (CGImageRef *inImage*)

Creates a system image view.

Parameters:

inImage CGImage to display

Returns:

HIViewRef for image view

Definition at line 422 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.14 HIViewRef PPx::SysCreateView::ImageWell (const ControlButtonContentInfo & *inContent*)

Creates a system image well.

Parameters:

inContent Content of image well

Returns:

HIViewRef for image well

Definition at line 444 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.15 HIViewRef PPx::SysCreateView::ListBox (bool *inAutoSize*, SInt16 *inRowCount*, SInt16 *inColCount*, bool *inHorizScroll*, bool *inVertScroll*, SInt16 *inCellHeight*, SInt16 *inCellWidth*, bool *inHasGrowSpace*, const ListDefSpec & *inListDef*)

Creates a system list box.

Parameters:

inAutoSize Whether to automatically set the cell size

inRowCount Number of rows

inColCount Number of columns

inHorizScroll Whether list box has a horizontal scroll bar

inVertScroll Whether list box has a vertical scroll bar

inCellHeight Pixel height of cells

inCellWidth Pixel widht of cells

inHasGrowSpace Whether to leave space for a grow box

inListDef List defintion

Returns:

HIViewRef for list box

Definition at line 476 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.16 HIViewRef PPx::SysCreateView::LittleArrows (SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, SInt32 *inIncrement*)

Creates a system little arrows control.

Parameters:

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inIncrement Amout to increment/decrement value when clicked

Returns:

HIViewRef for little arrows control

Definition at line 513 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.17 HIViewRef PPx::SysCreateView::PictureControl (SInt16 *inPictResID*, PicHandle *inPictureHandle*, bool *inDontTrack*)

Creates a system picture control.

Parameters:

inPictResID PICT resource ID of picture to display

inPictureHandle Handle to picture to dispaly

inDontTrack Whether not to track mouse downs

Returns:

HIViewRef for picture control

Definition at line 543 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.18 HIViewRef PPx::SysCreateView::Placard ()

Creates a system placard.

Returns:

HIViewRef for placard

Definition at line 583 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.19 HIViewRef PPx::SysCreateView::PopupArrow (ControlPopupArrowOrientation *inOrientation*, ControlPopupArrowSize *inArrowSize*)

Creates a system popup arrow control.

Parameters:

inOrientation Direction arrow points (north, east, south, west)

inArrowSize Size of arrow (normal or small)

Returns:

HIViewRef for popup arrow

Definition at line 611 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.20 HIViewRef PPx::SysCreateView::PopupButton (CFStringRef *inTitle*, SInt32 *inMenuID*, bool *inHasVariableWidth*, SInt16 *inTitleWidth*, SInt16 *inTitleJust*, Style *inTitleStyle*)

Creates a system popup button.

Parameters:

inTitle Title for popup

inMenuID Menu ID for popup

inHasVariableWidth Whether the menu has variable width

inTitleWidth Width of title (use -1 for variable width)

inTitleJust Justification of title text

inTitleStyle Font style for title

Returns:

HIViewRef for popup button

Definition at line 646 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.21 HIViewRef PPx::SysCreateView::PopupGroupBox (CFStringRef *inTitle*, bool *inIsPrimary*, SInt32 *inMenuID*, bool *inHasVariableWidth*, SInt16 *inTitleWidth*, SInt16 *inTitleJust*, Style *inTitleStyle*)

Creates a system popup group box.

Parameters:

inTitle Title for popup

inIsPrimary Group box kind (true = primary, false = secondary)

inMenuID Menu ID for popup

inHasVariableWidth Whether the menu has variable width

inTitleWidth Width of title (use -1 for variable width)

inTitleJust Justification of title text

inTitleStyle Font style for title

Returns:

HIViewRef for popup group box

Definition at line 688 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.22 HIViewRef PPx::SysCreateView::ProgressBar (SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, bool *inIsIndeterminate*)

Creates a system jprogress bar.

Parameters:

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inIsIndeterminate Whether progress bar is indeterminate

Returns:

HIViewRef for progress bar

An indeterminate progress bar just animates and has no value

Definition at line 725 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.23 HIViewRef PPx::SysCreateView::PushButton (CFStringRef *inTitle*)

Creates a system push button.

Parameters:

inTitle Title of push button

Returns:

HIViewRef for push button

Definition at line 754 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.24 HIViewRef PPx::SysCreateView::RadioButton (CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inAutoToggle*)

Creates a system radio button.

Parameters:

itTile Title of radio button

inInitialValue 0 = off, 1 = on

inAutoToggle Whether button automatically toggles when clicked

Returns:
HIViewRef for radio button

Definition at line 779 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.25 HIViewRef PPx::SysCreateView::RadioGroup ()

Creates a system radio group.

Returns:
HIViewRef for radio group

All views inside a radio group must be radio buttons

Definition at line 805 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.26 HIViewRef PPx::SysCreateView::RelevanceBar (SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*)

Creates a system relevance bar.

Parameters:

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

Returns:
HIViewRef for relevance bar

Definition at line 829 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.27 HIViewRef PPx::SysCreateView::RoundButton (ControlRoundButtonSize *inButtonSize*, const ControlButtonContentInfo & *inContent*)

Creates a system round button.

Parameters:

inButtonSize Size of round button (large or small)

inContent Content of round button

Returns:

HIViewRef for round button

Definition at line 857 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.28 HIViewRef PPx::SysCreateView::ScrollBar (SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, SInt32 *inViewSize*, bool *inHasLiveTracking*, ControlActionUPP *inLiveTrackingUPP*)

Creates a system scroll bar.

Parameters:

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inViewSize Size of view being scrolled

inHasLiveTracking Whether content scrolls as thumb is dragged

inLiveTrackingUPP Callback function for handling live tracking

Returns:

HIViewRef for scroll bar

Definition at line 901 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.29 HIViewRef PPx::SysCreateView::ScrollView (OptionBits *inOptions*)

Creates a system scroll view.

Parameters:

inOptions Options for scroll view

Returns:

HIViewRef for scroll view

Definition at line 932 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.30 HIViewRef PPx::SysCreateView::SeparatorLine ()

Creates a system separator line.

Returns:

HIViewRef for separator line

Definition at line 952 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.31 HIViewRef PPx::SysCreateView::Slider (SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, ControlSliderOrientation *inOrientation*, UInt16 *inTickMarksCount*, bool *inHasLiveTracking*, ControlActionUPP *inLiveTrackingUPP*)

Creates a system slider control.

Parameters:

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inOrientation Orientation of thumb

inTickMarkCount Number of tick marks to draw

inHasLiveTracking Whether slider live tracks the thumb

inLiveTrackingUPP Function for handling live tracking

Returns:

HIViewRef for slider

Definition at line 980 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.32 HIViewRef PPx::SysCreateView::StaticText (CFStringRef *inText*, const ControlFontStyleRec ∗ *inStyle*)

Creates a system static text control.

Parameters:

inText Text to display

inStyle Text style

Returns:

HIViewRef for static text

Definition at line 1013 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.33 HIViewRef PPx::SysCreateView::TabView (ControlTabSize *inTabSize*, ControlTabDirection *inTabDirection*, UInt16 *inTabCount*, const ControlTabEntry ∗ *inTabEntries*)

Creates a system tab view.

Parameters:

inTabSize Size of tabs (normal or small)

inTabDirection Dirction of tabs (north, south, east, west)

inTabCount Number of tabs

inTabEntries Data for each tab

Returns:

HIViewRef for tab view

Definition at line 1047 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.34 HIViewRef PPx::SysCreateView::TextGroupBox (CFStringRef *inTitle*, bool *inIsPrimary*)

Creates a system text group box.

Parameters:

inTitle Title for text group box

inIsPrimary Group box kind (true = primary, false = secondary)

Returns:

HIViewRef for text group box

Definition at line 1077 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.16.2.35 HIViewRef PPx::SysCreateView::WindowHeader (bool *inIsListHeader*)

Creates a system window header.

Parameters:

inIsListHeader Whether it's a list header, which as not bottom line

Returns:

HIViewRef for window header

Definition at line 1103 of file SysCreateView.cp.

References PPx_ThrowIfOSError_.

5.17 PPx::SysEventParam Namespace Reference

5.17.1 Detailed Description

Utility functions for getting and setting Carbon Event parameters.

Functions

- **PPx_Declare_SysEventParam_Traits** (Boolean, typeBoolean)
- **PPx_Declare_SysEventParam_Traits** (char, typeChar)
- **PPx_Declare_SysEventParam_Traits** (SInt16, typeSInt16)
- **PPx_Declare_SysEventParam_Traits** (SInt32, typeSInt32)
- **PPx_Declare_SysEventParam_Traits** (UInt32, typeUInt32)
- **PPx_Declare_SysEventParam_Traits** (SInt64, typeSInt64)
- **PPx_Declare_SysEventParam_Traits** (Float32, typeShortFloat)
- **PPx_Declare_SysEventParam_Traits** (Float64, typeFloat)
- **PPx_Declare_SysEventParam_Traits** (HIObjectRef, typeHIObjectRef)
- **PPx_Declare_SysEventParam_Traits** (WindowRef, typeWindowRef)
- **PPx_Declare_SysEventParam_Traits** (ControlRef, typeControlRef)
- **PPx_Declare_SysEventParam_Traits** (MenuRef, typeMenuRef)
- **PPx_Declare_SysEventParam_Traits** (ScrapRef, typeScrapRef)
- **PPx_Declare_SysEventParam_Traits** (DragRef, typeDragRef)
- **PPx_Declare_SysEventParam_Traits** (EventRef, typeEventRef)
- **PPx_Declare_SysEventParam_Traits** (EventTargetRef, typeEventTargetRef)
- **PPx_Declare_SysEventParam_Traits** (HIToolbarRefType, typeHIToolbarRef)
- **PPx_Declare_SysEventParam_Traits** (HIToolbarItemRefType, typeHIToolbarItemRef)
- **PPx_Declare_SysEventParam_Traits** (HICommand, typeHICommand)
- **PPx_Declare_SysEventParam_Traits** (ControlPartCodeType, typeControlPartCode)
- **PPx_Declare_SysEventParam_Traits** (ControlActionUPP, typeControlActionUPP)
- **PPx_Declare_SysEventParam_Traits** (EventHotKeyID, typeEventHotKeyID)
- **PPx_Declare_SysEventParam_Traits** (MouseTrackingRef, typeMouseTrackingRef)
- **PPx_Declare_SysEventParam_Traits** (EventMouseWheelAxisType, typeMouseWheelAxis)
- **PPx_Declare_SysEventParam_Traits** (MenuTrackingModeType, typeMenuTrackingMode)
- **PPx_Declare_SysEventParam_Traits** (MenuItemIndexType, typeMenuItemIndex)
- **PPx_Declare_SysEventParam_Traits** (MenuCommandType, typeMenuCommand)

- **PPx_Declare_SysEventParam_Traits** (MenuEventOptionsType, typeMenuEventOptions)
- **PPx_Declare_SysEventParam_Traits** (ThemeMenuStateType, typeThemeMenuState)
- **PPx_Declare_SysEventParam_Traits** (ThemeMenuItemTypeType, typeThemeMenuItemType)
- **PPx_Declare_SysEventParam_Traits** (WindowDefPartCodeType, typeWindowDefPartCode)
- **PPx_Declare_SysEventParam_Traits** (WindowRegionCodeType, typeWindowRegionCode)
- **PPx_Declare_SysEventParam_Traits** (CFStringRef, typeCFStringRef)
- **PPx_Declare_SysEventParam_Traits** (CFMutableStringRef, typeCFMutableStringRef)
- **PPx_Declare_SysEventParam_Traits** (CFTypeRef, typeCFTypeRef)
- **PPx_Declare_SysEventParam_Traits** (CFMutableArrayRef, typeCFMutableArrayRef)
- **PPx_Declare_SysEventParam_Traits** (AXUIElementRef, typeCFTypeRef)
- **PPx_Declare_SysEventParam_Traits** (CGContextRef, typeCGContextRef)
- **PPx_Declare_SysEventParam_Traits** (HIPoint, typeHIPoint)
- **PPx_Declare_SysEventParam_Traits** (HISize, typeHISize)
- **PPx_Declare_SysEventParam_Traits** (HIRect, typeHIRect)
- **PPx_Declare_SysEventParam_Traits** (GrafPtr, typeGrafPtr)
- **PPx_Declare_SysEventParam_Traits** (Point, typeQDPoint)
- **PPx_Declare_SysEventParam_Traits** (Rect, typeRectangle)
- **PPx_Declare_SysEventParam_Traits** (RgnHandle, typeQDRgnHandle)
- **PPx_Declare_SysEventParam_Traits** (FSRef, typeFSRef)
- **PPx_Declare_SysEventParam_Traits** (FSVolumeRefNumType, typeFSVolumeRefNum)
- **PPx_Declare_SysEventParam_Traits** (OSStatusType, typeOSStatus)
- **PPx_Declare_SysEventParam_Traits** (UniCharType, typeUnicodeText)
- **PPx_Declare_SysEventParam_Traits** (OSTypeType, typeType)
- **PPx_Declare_SysEventParam_Traits** (ProcessSerialNumber, typeProcessSerialNumber)
- template<typename TData> void Set (EventRef inEvent, EventParamName inName, const TData &inValue)

  Set a parameter for a Carbon Event.

- template<typename TData> void Set (EventRef inEvent, EventParamName inName, EventParamType inType, const TData &inValue)

  Set a parameter for a Carbon Event.

- template<typename TData> void Get (EventRef inEvent, EventParamName inName, TData &outValue)

Get a parameter from a Carbon Event.

- template<typename TData> void Get (EventRef inEvent, EventParamName inName, EventParamType inType, TData &outValue)

  Get a parameter from a Carbon Event.

- template<typename TData> OSStatus GetOptional (EventRef inEvent, EventParamName inName, TData &outValue)

  Get an optional parameter from a Carbon Event.

- template<typename TData> OSStatus GetOptional (EventRef inEvent, EventParamName inName, EventParamType inType, TData &outValue)

  Get an optional parameter from a Carbon Event.

5.17.2 Function Documentation

5.17.2.1 template<typename TData> void Get (EventRef *inEvent*, EventParamName *inName*, EventParamType *inType*, TData & *outValue*)

Get a parameter from a Carbon Event.

Throws if there is an error getting the parameter.

Parameters:

inEvent A reference to a Carbon Event

inName Name of the parameter to get

inType Type ID of the parameter

outValue Parameter value

Definition at line 245 of file SysEventParam.h.

References PPx_ThrowIfOSError_.

5.17.2.2 template<typename TData> void Get (EventRef *inEvent*, EventParamName *inName*, TData & *outValue*)

Get a parameter from a Carbon Event.

Throws if there is an error getting the parameter.

Parameters:

inEvent A reference to a Carbon Event

inName Name of the parameter to get

outValue Parameter value

Definition at line 218 of file SysEventParam.h.

References PPx_ThrowIfOSError_.

5.17.2.3 template<typename TData> OSStatus GetOptional (EventRef *inEvent*, EventParamName *inName*, EventParamType *inType*, TData & *outValue*)

Get an optional parameter from a Carbon Event.

Check the return value to determine if the parameter is present.

Parameters:

inEvent A reference to a Carbon Event

inName Name of the parameter to get

inType Type ID of the parameter

outValue Parameter value

Returns:

A Mac OS error code

Definition at line 301 of file SysEventParam.h.

5.17.2.4 template<typename TData> OSStatus GetOptional (EventRef *inEvent*, EventParamName *inName*, TData & *outValue*)

Get an optional parameter from a Carbon Event.

Check the return value to determine if the parameter is present.

Parameters:

inEvent A reference to a Carbon Event

inName Name of the parameter to get

outValue Parameter value

Returns:

A Mac OS error code

Definition at line 272 of file SysEventParam.h.

5.17.2.5 template<typename TData> void Set (EventRef *inEvent*, EventParamName *inName*, EventParamType *inType*, const TData & *inValue*)

Set a parameter for a Carbon Event.

Parameters:

inEvent A reference to a Carbon Event

inName Name of the parameter to set

inType Type ID of the parameter

inValue Parameter value

Definition at line 193 of file SysEventParam.h.

References PPx_ThrowIfOSError_.

5.17.2.6 template<typename TData> void Set (EventRef *inEvent*, EventParamName *inName*, const TData & *inValue*)

Set a parameter for a Carbon Event.

Parameters:

inEvent A reference to a Carbon Event

inName Name of the parameter to set

inValue Parameter value

Definition at line 167 of file SysEventParam.h.

References PPx_ThrowIfOSError_.

5.18 PPx::SysScrap Namespace Reference

5.18.1 Detailed Description

Wrapper functions for the Scrap Manager.

Functions

- ScrapRef GetNamedScrap (CFStringRef inScrapName, bool inClear=clear_No)

  Returns the ScrapRef for the named scrap.

- void GetData (CFStringRef inScrapName, ScrapFlavorType inFlavor, Size &ioByteCount, void ∗outDataPtr)

  Get flavor data from the named scrap.

- Size GetDataSize (CFStringRef inScrapName, ScrapFlavorType inFlavor)

  Returns the size of the specified flavor of data in the scrap.

- bool HasData (CFStringRef inScrapName, ScrapFlavorType inFlavor)

  Returns whether the scrap has data of the specified flavor.

- void ClearData (CFStringRef inScrapName)

  Clears all data from the scrap.

- void SetData (CFStringRef inScrapName, ScrapFlavorType inFlavor, Size inDataSize, const void ∗inDataPtr, ScrapFlavorFlags inFlags=kScrapFlavorMaskNone, bool inClear=clear_Yes)

  Put data into the scrap.

- void PromiseData (CFStringRef inScrapName, ScrapFlavorType inFlavor, Size inDataSize=kScrapFlavorSizeUnknown, ScrapFlavorFlags inFlags=kScrapFlavorMaskNone, bool inClear=clear_Yes)

  Put a promise to supply data into the scrap.

- void SetPromiseKeeper (CFStringRef inScrapName, ScrapPromiseKeeperUPP inPromiseUPP, const void ∗inUserData)

  Specify the promise keeper function that will supply promised data.

5.18.2 Function Documentation

5.18.2.1 void PPx::SysScrap::GetData (CFStringRef *inScrapName*, ScrapFlavorType *inFlavor*, Size & *ioByteCount*, void ∗ *outDataPtr*)

Get flavor data from the named scrap.

Parameters:

inScrapName Name of the scrap

inFlavor Flavor of data

ioByteCount On input, maximum bytes to get; On output, actual bytes returned

outDataPtr Pointer to data buffer

Call GetDataSize if you need to know the size of the data before getting it.

Definition at line 50 of file SysScrap.cp.

References GetNamedScrap(), and PPx_ThrowIfOSError_.

5.18.2.2 Size PPx::SysScrap::GetDataSize (CFStringRef *inScrapName*, ScrapFlavorType *inFlavor*)

Returns the size of the specified flavor of data in the scrap.

Parameters:

inScrapName Name of the scrap

inFlavor Flavor of data

Returns:

Size in bytes of the data

Definition at line 75 of file SysScrap.cp.

References GetNamedScrap(), and PPx_ThrowIfOSError_.

5.18.2.3 ScrapRef PPx::SysScrap::GetNamedScrap (CFStringRef *inScrapName*, bool *inClear* = clear_No)

Returns the ScrapRef for the named scrap.

Parameters:

inScrapName Name of the scrap

inClear Whether to clear the scrap

Returns:

ScrapRef for the named scrap

Definition at line 20 of file SysScrap.cp.

References PPx_ThrowIfOSError_.

Referenced by ClearData(), GetData(), GetDataSize(), HasData(), PromiseData(), SetData(), and SetPromiseKeeper().

5.18.2.4 bool PPx::SysScrap::HasData (CFStringRef *inScrapName*, ScrapFlavorType *inFlavor*)

Returns whether the scrap has data of the specified flavor.

Parameters:

inScrapName Name of the scrap

inFlavor Flavor of data

Returns:

Whether the scrap has data of the specified flavor

Definition at line 100 of file SysScrap.cp.

References GetNamedScrap().

5.18.2.5 void PPx::SysScrap::PromiseData (CFStringRef *inScrapName*, ScrapFlavorType *inFlavor*, Size *inDataSize* = kScrapFlavorSizeUnknown, ScrapFlavorFlags *inFlags* = kScrapFlavorMaskNone, bool *inClear* = clear_Yes)

Put a promise to supply data into the scrap.

Parameters:

inScrapName Name of the scrap

inFlavor Flavor of data

inDataSize Number of bytes of data

inFlags Options for storing data

inClear Whether to clear the scrap before adding data

The system will caller your promise keeper function if a client requests the flavor data from the scrap.

If you do not know how much data there is, pass -1 for inDataSize

Definition at line 172 of file SysScrap.cp.

References GetNamedScrap(), and PPx_ThrowIfOSError_.

5.18.2.6 void PPx::SysScrap::SetData (CFStringRef *inScrapName*, ScrapFlavorType *inFlavor*, Size *inDataSize*, const void ∗ *inDataPtr*, ScrapFlavorFlags *inFlags* = kScrapFlavorMaskNone, bool *inClear* = clear_Yes)

Put data into the scrap.

Parameters:

inScrapName Name of the scrap

inFlavor Flavor of data

inDataSize Number of bytes of data

inDataPtr Pointer to data buffer

inFlags Options for storing data

inClear Whether to clear the scrap before adding data

Definition at line 139 of file SysScrap.cp.

References GetNamedScrap(), and PPx_ThrowIfOSError_.

5.18.2.7 void PPx::SysScrap::SetPromiseKeeper (CFStringRef *inScrapName*, ScrapPromiseKeeperUPP *inPromiseUPP*, const void ∗ *inUserData*)

Specify the promise keeper function that will supply promised data.

Parameters:

inScrapName Name of the scrap

inPromiseUPP Pointer to promise keeper object

inUserData Pointer to user-defined data

Definition at line 197 of file SysScrap.cp.

References GetNamedScrap(), and PPx_ThrowIfOSError_.

5.19 PPx::ViewUtils Namespace Reference

5.19.1 Detailed Description

Utility functions for working with Views.

Functions

- void SetControlThemeFontID (View ∗inViewRef, ThemeFontID inFontID)

  Specify the theme font ID for drawing text in a view.

- bool GetControlThemeFontID (View ∗inViewRef, ThemeFontID &outFontID)

  Get the theme font ID used for drawing text in a view.

- void QDToHIPoint (const Point &inQDPoint, HIPoint &outHIPoint)

  Converts a Quickdraw Point to a HIPoint.

- void HIToQDPoint (const HIPoint &inHIPoint, Point &outQDPoint)

  Converts a HIPoint to a Quickdraw Point.

- void QDToHIRect (const Rect &inQDRect, HIRect &outHIRect)

  Converts a Quickdraw Rect to a HIRect.

- void HIToQDRect (const HIRect &inHIRect, Rect &outQDRect)

  Converts a HIRect to a Quickdraw REct.

5.19.2 Function Documentation

5.19.2.1 bool PPx::ViewUtils::GetControlThemeFontID (View ∗ *inView*, ThemeFontID & *outFontID*)

Get the theme font ID used for drawing text in a view.

Parameters:

inView Pointer to View object

outFontID Theme font ID

Returns:

Whetherthe view uses a theme font ID. If true, outFontID is the theme font ID used. If false, outFontID is unchanged.

Definition at line 49 of file PPxViewUtils.cp.

References PPx::View::GetDataTag().

5.19.2.2 void PPx::ViewUtils::HIToQDPoint (const HIPoint & *inHIPoint*, Point & *outQDPoint*)

Converts a HIPoint to a Quickdraw Point.

Parameters:

inHIPoint HIPoint

outQDPoint Corresponding Quickdraw Point

Truncates the HIPoint coordinates from 32 to 16 bits

Definition at line 97 of file PPxViewUtils.cp.

5.19.2.3 void PPx::ViewUtils::HIToQDRect (const HIRect & *inHIRect*, Rect & *outQDRect*)

Converts a HIRect to a Quickdraw REct.

Parameters:

inHIRect HIRect

outQDRect Corresponding Quickdraw Rect

Truncates the HIPoint coordinates from 32 to 16 bits

Definition at line 137 of file PPxViewUtils.cp.

5.19.2.4 void PPx::ViewUtils::QDToHIPoint (const Point & *inQDPoint*, HIPoint & *outHIPoint*)

Converts a Quickdraw Point to a HIPoint.

Parameters:

inQDPoint Quickdraw Point

outHIPoint Corresponding HIPoint

Definition at line 77 of file PPxViewUtils.cp.

5.19.2.5 void PPx::ViewUtils::QDToHIRect (const Rect & *inQDRect*, HIRect & *outHIRect*)

Converts a Quickdraw Rect to a HIRect.

Parameters:

inQDRect Quickdraw Rect

outHIRect Corresponding HIRect

Definition at line 115 of file PPxViewUtils.cp.

5.19.2.6 void PPx::ViewUtils::SetControlThemeFontID (View ∗ *inView*, ThemeFontID *inFontID*)

Specify the theme font ID for drawing text in a view.

Parameters:

inView Pointer to View object

inFontID Theme font ID

Note:

The view must support the data tag kControlFontStyleTag

Definition at line 22 of file PPxViewUtils.cp.

References PPx::View::SetDataTag().

5.20 PPx::XMLConstants Namespace Reference

5.20.1 Detailed Description

Constants for XML identifiers.

Variables

- const CFStringRef **elem_Persistent** = CFSTR(”Persistent”)
- const CFStringRef **attr_Name** = CFSTR(”name”)
- const CFStringRef **attr_StorageID** = CFSTR(”id”)
- const CFStringRef **attr_Class** = CFSTR(”class”)
- const CFStringRef **bool_True** = CFSTR(”true”)
- const CFStringRef **bool_False** = CFSTR(”false”)
- const int **max_TabLevel** = 5
- const CFStringRef **whitespace_NewLineTabs** [ ]
- const CFStringRef **type_String** = CFSTR(”String”)
- const CFStringRef **type_ObjectID** = CFSTR(”ObjectID”)
- const CFStringRef **type_ObjectRef** = CFSTR(”ObjectRef”)
- const CFStringRef **type_ObjectRefVector** = CFSTR(”ObjectRefVector”)
- const CFStringRef **type_bool** = CFSTR(”bool”)
- const CFStringRef **type_SInt8** = CFSTR(”SInt8”)
- const CFStringRef **type_UInt8** = CFSTR(”UInt8”)
- const CFStringRef **type_SInt16** = CFSTR(”SInt16”)
- const CFStringRef **type_UInt16** = CFSTR(”UInt16”)
- const CFStringRef **type_SInt32** = CFSTR(”SInt32”)
- const CFStringRef **type_UInt32** = CFSTR(”UInt32”)
- const CFStringRef **type_float** = CFSTR(”float”)
- const CFStringRef **type_double** = CFSTR(”double”)
- const CFStringRef **type_Point** = CFSTR(”Point”)
- const CFStringRef **field_Point_h** = CFSTR(”h”)
- const CFStringRef **field_Point_v** = CFSTR(”v”)
- const CFStringRef **type_Rect** = CFSTR(”Rect”)
- const CFStringRef **field_Rect_top** = CFSTR(”top”)
- const CFStringRef **field_Rect_left** = CFSTR(”left”)
- const CFStringRef **field_Rect_bottom** = CFSTR(”bottom”)
- const CFStringRef **field_Rect_right** = CFSTR(”right”)
- const CFStringRef **type_CGPoint** = CFSTR(”CGPoint”)
- const CFStringRef **field_CGPoint_x** = CFSTR(”x”)
- const CFStringRef **field_CGPoint_y** = CFSTR(”y”)
- const CFStringRef **type_CGSize** = CFSTR(”CGSize”)
- const CFStringRef **field_CGSize_width** = CFSTR(”width”)

- const CFStringRef **field_CGSize_height** = CFSTR(”height”)
- const CFStringRef **type_CGRect** = CFSTR(”CGRect”)
- const CFStringRef **field_CGRect_origin** = CFSTR(”origin”)
- const CFStringRef **field_CGRect_size** = CFSTR(”size”)

5.20.2 Variable Documentation

5.20.2.1 const CFStringRef PPx::XMLConstants::whitespace_NewLineTabs[ ]

Initial value:

```
{ CFSTR("\n"),
                                                    CFSTR("\n\t"),
                                                    CFSTR("\n\t\t"),
                                                    CFSTR("\n\t\t\t"),
                                                    CFSTR("\n\t\t\t\t"),
                                                    CFSTR("\n\t\t\t\t\t") }
```

Definition at line 37 of file PPxXMLConstants.h.

5.21 PPx::XMLDecoder Namespace Reference

5.21.1 Detailed Description

Maintains a table of which maps XML decoder functions to data types.

An XML decoder function converts information in an XML tree to a Data Object.

You need to registef a decoder for every data type that you wish to read from XML descriptions

Typedefs

- typedef AutoRetained< DataObject >(∗ DecoderFuncT)(const CFXMLTree &)

  Decoder function signature.

Functions

- void Register (CFStringRef inTypeName, DecoderFuncT inDecoderFunc)

  Registers an XML decoder function for a data type.

- DecoderFuncT Find (CFStringRef inTypeName)

  Returns the XML decoder function for a data type.

5.21.2 Function Documentation

5.21.2.1 XMLDecoder::DecoderFuncT PPx::XMLDecoder::Find (CFStringRef *inTypeName*)

Returns the XML decoder function for a data type.

Parameters:

inTypeName Name of data type

Returns:

XML Decorder function for the data type

Definition at line 58 of file PPxXMLDecoder.cp.

References PPx_Throw_.

5.21.2.2 void PPx::XMLDecoder::Register (CFStringRef *inTypeName*, DecoderFuncT *inDecoderFunc*)

Registers an XML decoder function for a data type.

Parameters:

inTypeName Name of data type

inDecoderFunc XML Decoder function

Definition at line 40 of file PPxXMLDecoder.cp.

5.22 PPx::XMLDecoderFuncs Namespace Reference

5.22.1 Detailed Description

XML Decoder functions for common data types.

Functions

- template<typename TData> AutoRetained< DataObject > DecodeData (const CFXMLTree &inDataTree)

  Template function which returns a DataObject containing a value extracted from an XML Tree.

- template<typename TData> AutoRetained< DataObject > DecodeVector (const CFXMLTree &inDataTree)

  Template function which returns a DataObject containing a vector of values extracted from an XML Tree.

- template<> AutoRetained< DataObject > DecodeData< Point > (const CFXMLTree &inDataTree)

  Function template specialization for an XML Decoder for a Quickdraw Point.

- template<> AutoRetained< DataObject > DecodeData< Rect > (const CFXMLTree &inDataTree)

  Function template specialization for an XML Decoder for a Quickdraw Rect.

- template<> AutoRetained< DataObject > DecodeData< CGPoint > (const CFXMLTree &inDataTree)

  Function template specialization for an XML Decoder for a CGPoint.

- template<> AutoRetained< DataObject > DecodeData< CGSize > (const CFXMLTree &inDataTree)

  Function template specialization for an XML Decoder for a CGSize.

- template<> AutoRetained< DataObject > DecodeData< CGRect > (const CFXMLTree &inDataTree)

  Function template specialization for an XML Decoder for a CGRect.

5.22.2 Function Documentation

5.22.2.1 template<typename TData> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeData (const CFXMLTree & *inDataTree*)

Template function which returns a DataObject containing a value extracted from an XML Tree.

TData is a template parameter for the value type

Parameters:

inDataTree XML Tree containing the data value

This is the general case template which assumes the value is a simple type (number, bool, or string). Create template specializations for complex types such as structs.

Definition at line 184 of file PPxXMLDecoder.h.

5.22.2.2 template<> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeData< CGPoint > (const CFXMLTree & *inDataTree*)

Function template specialization for an XML Decoder for a CGPoint.

Parameters:

inDataTree XML Tree containing data for a CGPoint

Returns:

DataObject containing a CGPoint value

5.22.2.3 template<> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeData< CGRect > (const CFXMLTree & *inDataTree*)

Function template specialization for an XML Decoder for a CGRect.

Parameters:

inDataTree XML Tree containing data for a CGRect

Returns:

DataObject containing a CGRect value

5.22.2.4 template<> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeData< CGSize > (const CFXMLTree & *inDataTree*)

Function template specialization for an XML Decoder for a CGSize.

Parameters:

inDataTree XML Tree containing data for a CGSize

Returns:

DataObject containing a CGSize value

5.22.2.5 template<> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeData< Point > (const CFXMLTree & *inDataTree*)

Function template specialization for an XML Decoder for a Quickdraw Point.

Parameters:

inDataTree XML Tree containing data for a Point

Returns:

DataObject containing a Point value

5.22.2.6 template<> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeData< Rect > (const CFXMLTree & *inDataTree*)

Function template specialization for an XML Decoder for a Quickdraw Rect.

Parameters:

inDataTree XML Tree containing data for a Rect

Returns:

DataObject containing a Rect value

5.22.2.7 template<typename TData> AutoRetained< DataObject > PPx::XMLDecoderFuncs::DecodeVector (const CFXMLTree & *inDataTree*)

Template function which returns a DataObject containing a vector of values extracted from an XML Tree.

TData is a template parameter for the value type

Parameters:

inDataTree XML Tree containing the vector of values

Definition at line 204 of file PPxXMLDecoder.h.

References PPx::CFTree::GetChildAtIndex(), PPx::CFTree::GetChildCount(), PPx::CFTree::GetFirstChild(), PPx::CFXMLTree::GetNode(), PPx::CFXMLNode::GetString(), and PPx::TDataObject< TData >::mValue.

5.23 PPx::XMLEncoder Namespace Reference

5.23.1 Detailed Description

Maintains a table which maps XML encoder functions to data types.

An XML encoder function converts information in a DataObject to an XML Tree.

You must register an encoder for every data type that you wish to write to XML descriptions

Compounds

- struct EncoderInfo

  Data stored for each registered enccoder type.

Typedefs

- typedef void(∗ EncoderFuncT)(const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Encoder function signature.

Functions

- void Register (const std::type_info &inTypeInfo, CFStringRef inTypeName, EncoderFuncT inEncoderFunc)

  Registers an XML encoder function for a data type.

- EncoderInfo Find (const std::type_info &inTypeInfo)

  Returns the XML encoder information for a DataObject class.

5.23.2 Function Documentation

5.23.2.1 XMLEncoder::EncoderInfo PPx::XMLEncoder::Find (const std::type_info & *inTypeInfo*)

Returns the XML encoder information for a DataObject class.

Parameters:

inTypeInfo type_info for a DataObject class

Definition at line 71 of file PPxXMLEncoder.cp.

References PPx_Throw_.

5.23.2.2 void PPx::XMLEncoder::Register (const std::type_info & *inTypeInfo*, CFStringRef *inTypeName*, EncoderFuncT *inEncoderFunc*)

Registers an XML encoder function for a data type.

Parameters:

inTypeInfo type_info for the DataObject class used to store values of the data type

inTypeName Name of the data type. Used as the XML element tag for values of the data type

inEncoderFunc XML Encoder function

Definition at line 52 of file PPxXMLEncoder.cp.

5.24 PPx::XMLEncoderFuncs Namespace Reference

5.24.1 Detailed Description

XML Encoder functions for common data types.

Functions

- template<typename TData> void EncodeData (const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Template function which stores a value from a DataObject as a child XML Tree of a another tree TData is a template parameter for the value type.

- template<typename TData> void EncodeVector (const DataObject &inDataVector, CFXMLTree &ioDataTree)

  Template function which stores a vectore of values a DataObject as a child XML Tree of a another tree TData is a template parameter for the value type.

- template<> void EncodeData< Point > (const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Function template specialization for an XML encoder for a Quickdraw Point.

- template<> void EncodeData< Rect > (const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Function template specialization for an XML encoder for a Quickdraw Rect.

- template<> void EncodeData< CGPoint > (const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Function template specialization for an XML encoder for a CGPoint.

- template<> void EncodeData< CGSize > (const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Function template specialization for an XML encoder for a CGSize.

- template<> void EncodeData< CGRect > (const DataObject &inDataObject, CFXMLTree &ioDataTree)

  Function template specialization for an XML encoder for a CGRect.

5.24.2 Function Documentation

5.24.2.1 template<typename TData> void PPx::XMLEncoderFuncs::EncodeData (const DataObject & *inDataObject*, CFXMLTree & *ioDataTree*)

Template function which stores a value from a DataObject as a child XML Tree of a another tree TData is a template parameter for the value type.

Parameters:

inDataObject DataObject containing the value

ioDataTree Parent XML Tree

This is the general case for the function template which assumes the value is a simple type which can be represented as a single text item. Create template specializations for complex types such as structs.

Definition at line 207 of file PPxXMLEncoder.h.

References PPx::CFTree::AppendChild(), and PPx::TDataObject< TData >::mValue.

5.24.2.2 template<> void PPx::XMLEncoderFuncs::EncodeData< CGPoint > (const DataObject & *inDataObject*, CFXMLTree & *ioDataTree*)

Function template specialization for an XML encoder for a CGPoint.

Parameters:

inDataObject DataObject containing a CGPoint value

ioDataTree Parent XML Tree

5.24.2.3 template<> void PPx::XMLEncoderFuncs::EncodeData< CGRect > (const DataObject & *inDataObject*, CFXMLTree & *ioDataTree*)

Function template specialization for an XML encoder for a CGRect.

Parameters:

inDataObject DataObject containing a CGRect value

ioDataTree Parent XML Tree

5.24.2.4 template< > void PPx::XMLEncoderFuncs::EncodeData< CGSize > (const DataObject & *inDataObject*, CFXMLTree & *ioDataTree*)

Function template specialization for an XML encoder for a CGSize.

Parameters:

inDataObject DataObject containing a CGSize value

ioDataTree Parent XML Tree

5.24.2.5 template< > void PPx::XMLEncoderFuncs::EncodeData< Point > (const DataObject & *inDataObject*, CFXMLTree & *ioDataTree*)

Function template specialization for an XML encoder for a Quickdraw Point.

Parameters:

inDataObject DataObject containing a Point value

ioDataTree Parent XML Tree

5.24.2.6 template< > void PPx::XMLEncoderFuncs::EncodeData< Rect > (const DataObject & *inDataObject*, CFXMLTree & *ioDataTree*)

Function template specialization for an XML encoder for a Quickdraw Rect.

Parameters:

inDataObject DataObject containing a Rect value

ioDataTree Parent XML Tree

5.24.2.7 template<typename TData> void PPx::XMLEncoderFuncs::EncodeVector (const DataObject & *inDataVector*, CFXMLTree & *ioDataTree*)

Template function which stores a vectore of values a DataObject as a child XML Tree of a another tree TData is a template parameter for the value type.

Parameters:

inDataVector DataObject containing the vector of values

ioDataTree Parent XML Tree

Definition at line 229 of file PPxXMLEncoder.h.

References PPx::CFTree::AppendChild(), PPx::XMLEncoder::EncoderInfo::encoderFunc, PPx::TDataVector< TData >::mDataValues, and PPx::XMLEncoder::EncoderInfo::typeName.

5.25 PPx::XMLTreeBrowser Namespace Reference

5.25.1 Detailed Description

Utility functions for extracting values from XML Trees.

Functions

- template<typename TData> TData GetValue (const CFXMLTree &inDataTree)

  Template function for getting a value from an XML Tree.

- CFXMLTree GetStructField (const CFXMLTree &inStructTree, const CFString &inFieldName)

  Returns XML Tree for a named field within the XML Tree for a struct.

- template<typename TData> bool GetFieldValue (const CFXMLTree &inStructTree, const CFString &inFieldName, TData &outValue)

  Template function for getting a value for the field of a struct from an XML Tree.

5.25.2 Function Documentation

5.25.2.1 template<typename TData> bool PPx::XMLTreeBrowser::GetFieldValue (const CFXMLTree & *inStructTree*, const CFString & *inFieldName*, TData & *outValue*)

Template function for getting a value for the field of a struct from an XML Tree.

TData is a template parameter for the type of the value

Parameters:

inStructTree XML Tree containg a struct

inFieldName Name of the field

outValue Value of the field

Returns:

Whether the field data exists in the struct XML Tree

Definition at line 108 of file PPxXMLDecoder.h.

References PPx::CFXMLTree::GetNode(), PPx::CFXMLNode::GetString(), GetStructField(), PPx::CFObject< CFTreeRef >::IsValid(), and PPx::TDataObject< TData >::mValue.

5.25.2.2 CFXMLTree PPx::XMLTreeBrowser::GetStructField (const CFXMLTree & *inStructTree*, const CFString & *inFieldName*)

Returns XML Tree for a named field within the XML Tree for a struct.

Parameters:

inStructTree XML Tree containing a struct

inFieldName Name of the field

Returns:

XML Tree containing the field

Definition at line 279 of file PPxXMLDecoder.cp.

References PPx::CFXMLElement::GetAttributeValue(), PPx::CFTree::GetFirstChild(), PPx::CFTree::GetNextSibling(), PPx::CFXMLTree::GetNode(), and PPx::CFObject< CFTreeRef >::IsValid().

Referenced by GetFieldValue().

5.25.2.3 template<typename TData> TData PPx::XMLTreeBrowser::GetValue (const CFXMLTree & *inDataTree*)

Template function for getting a value from an XML Tree.

Parameters:

inDataTree XML Tree containing data for a string

Returns:

CFString object

This function takes care of mapping the 5 escape sequences (&amp, &apos, &gt, &lt, &quot) to the correspoding charaters.

Definition at line 204 of file PPxXMLDecoder.cp.

References PPx::CFString::Append(), PPx::CFTree::GetChildAtIndex(), PPx::CFTree::GetChildCount(), PPx::CFXMLTree::GetNode(), PPx::CFXMLNode::GetString(), PPx::CFXMLNode::GetTypeCode(), and PPx::CFObject< CFStringRef >::UseRef().

5.26 PPx::XMLTreeBuilder Namespace Reference

5.26.1 Detailed Description

Utility functions for building XML Trees containing data values.

Functions

- void **ReplaceSubstring** (const CFString &inSubstring, const CFString &inReplacement, CFString &ioTargetString)
- void **IndentLevel** (CFXMLTree &inFirstTree, SInt16 inLevel)
- CFXMLTree MakeElement (CFStringRef inElemTag)

  Returns an XML Tree with an element node.

- CFXMLTree MakeElement (CFStringRef inElemTag, const CFStringRef ∗inAttrNames, const CFStringRef ∗inAttrValues, CFIndex inAttrCount)

  Returns an XML Tree with an element node that has a list of attributes.

- CFXMLTree MakeElement (CFStringRef inElemTag, CFStringRef inAttrName, CFStringRef inAttrValue)

  Returns an XML Tree with an element node that has one attributes.

- CFXMLTree MakePersistentElement (ObjectStorageIDT inStorageID, CFStringRef inClassName)

  Returns an XML Tree with an element node for a Persistent object.

- template<typename TData> void AddChildDataValue (const TData &inValue, CFStringRef inName, const XMLEncoder::EncoderInfo &inEncoder, CFXMLTree &ioParent)

  Template function for adding data as a child XML tree of another tree TData is a template parameter for the value type.

- CFXMLTree MakeTextString (const CFString &inString)

  Returns a XML Tree with a text node.

- template<typename TData> CFXMLTree MakeText (const TData &inData)

  Template function for making an XML Tree with a node containing the text representation of a value TData is a template parameter for the value type.

- CFXMLTree MakeText (const CFString &inString)

  Makes an XML Tree with a text node containing a string.

- CFXMLTree MakeText (bool inBool)

Makes an XML Tree with a text node containing a bool value.

- void FormatDescriptorsTree (CFXMLTree &inXMLTree)

  Adds new line and tab character whitespace to the XML Tree.

- CFXMLTree MakeWhitespace (CFStringRef inWhitespace)

  Returns an XML Tree with a whitespace node.

5.26.2 Function Documentation

5.26.2.1 template<typename TData> void PPx::XMLTreeBuilder::AddChildDataValue (const TData & *inValue*, CFStringRef *inName*, const XMLEncoder::EncoderInfo & *inEncoder*, CFXMLTree & *ioParent*)

Template function for adding data as a child XML tree of another tree TData is a template parameter for the value type.

Parameters:

inValue Data value

inName Name of data item

inEncoder XML Encoder information for the data type

ioParent Parent XML Tree

Definition at line 110 of file PPxXMLEncoder.h.

References PPx::CFTree::AppendChild(), PPx::XMLEncoder::EncoderInfo::encoderFunc, and PPx::XMLEncoder::EncoderInfo::typeName.

5.26.2.2 void PPx::XMLTreeBuilder::FormatDescriptorsTree (CFXMLTree & *inDescTree*)

Adds new line and tab character whitespace to the XML Tree.

Parameters:

inDescTree XML Tree containing object descriptors

New lines and tabs make the XML easier to read when output as text

Definition at line 471 of file PPxXMLEncoder.cp.

References PPx::CFTree::AppendChild(), PPx::CFTree::GetFirstChild(), PPx::CFTree::InsertSibling(), MakeWhitespace(), and PPx::CFTree::PrependChild().

5.26.2.3 CFXMLTree PPx::XMLTreeBuilder::MakeElement (CFStringRef *inElemTag*, CFStringRef *inAttrName*, CFStringRef *inAttrValue*)

Returns an XML Tree with an element node that has one attributes.

Parameters:

inElemTag Name of element tag

inAttrName Naem of attribute

inAttrValue Value of attribute

Returns:

XML Tree with a node having the element tag and attribute

Definition at line 283 of file PPxXMLEncoder.cp.

References MakeElement().

5.26.2.4 CFXMLTree PPx::XMLTreeBuilder::MakeElement (CFStringRef *inElemTag*, const CFStringRef ∗ *inAttrNames*, const CFStringRef ∗ *inAttrValues*, CFIndex *inAttrCount*)

Returns an XML Tree with an element node that has a list of attributes.

Parameters:

inElemTag Name of element tag

inAttrNames Array of attribute names

inAttrValues Array of attribute values

inAttrCount Number of attributes

Returns:

XML Tree with a node having the element tag and attributes

Definition at line 252 of file PPxXMLEncoder.cp.

5.26.2.5 CFXMLTree PPx::XMLTreeBuilder::MakeElement (CFStringRef *inElemTag*)

Returns an XML Tree with an element node.

Parameters:

inElemTag Name of element tag

Returns:

XML Tree with a node having the element tag

Definition at line 230 of file PPxXMLEncoder.cp.

Referenced by MakeElement(), and MakePersistentElement().

5.26.2.6 CFXMLTree PPx::XMLTreeBuilder::MakePersistentElement (ObjectStorageIDT *inStorageID*, CFStringRef *inClassName*)

Returns an XML Tree with an element node for a Persistent object.

Parameters:

inStorageID Storage ID number of the Persistent object

inClassName Name of the class of the Persistent object

Returns:

XML Tree with an element node for a Persistent object

The element tag name specifies a Persistent object and the stoarge ID and class name are attributes

Definition at line 306 of file PPxXMLEncoder.cp.

References PPx::CFString::AssignNumericValue(), and MakeElement().

5.26.2.7 CFXMLTree PPx::XMLTreeBuilder::MakeText (bool *inBool*)

Makes an XML Tree with a text node containing a bool value.

Parameters:

inBool bool value

Returns:

XML Tree with a text node

bool value is written as ”true” or ”false”

Definition at line 411 of file PPxXMLEncoder.cp.

References MakeTextString().

5.26.2.8 CFXMLTree PPx::XMLTreeBuilder::MakeText (const CFString & *inString*)

Makes an XML Tree with a text node containing a string.

Parameters:

inString String to put into text node

Returns:

XML Tree with a text node

This function handles escaping the five special characters (ampersand, apostrophe, greater than, less than, and quotes)

Definition at line 384 of file PPxXMLEncoder.cp.

References MakeTextString().

5.26.2.9 template<typename TData> CFXMLTree PPx::XMLTreeBuilder::MakeText (const TData & *inData*)

Template function for making an XML Tree with a node containing the text representation of a value TData is a template parameter for the value type.

Parameters:

inData Data value

Returns:

XML Tree with a node containing the value as text

Definition at line 137 of file PPxXMLEncoder.h.

References PPx::CFString::AssignNumericValue(), and MakeTextString().

5.26.2.10 CFXMLTree PPx::XMLTreeBuilder::MakeTextString (const CFString & *inString*)

Returns a XML Tree with a text node.

Parameters:

inString Text to put in the node

Returns:

XML Tree with a text node

Definition at line 330 of file PPxXMLEncoder.cp.

Referenced by MakeText().

5.26.2.11 CFXMLTree PPx::XMLTreeBuilder::MakeWhitespace (CFStringRef *inWhitespace*)

Returns an XML Tree with a whitespace node.

Parameters:

inWhitespace Whitespace charactes

Returns:

XML Tree wiht a whitespace node

Definition at line 493 of file PPxXMLEncoder.cp.

Referenced by FormatDescriptorsTree().

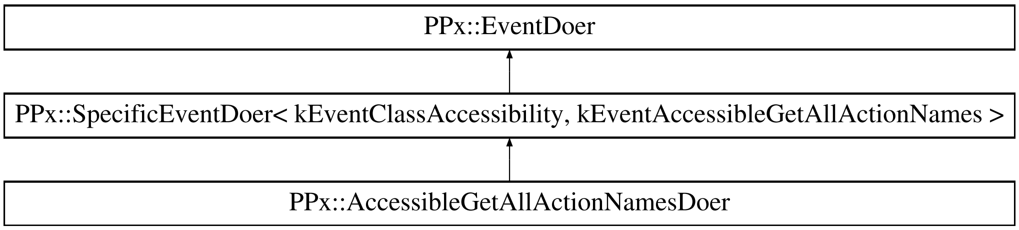
Chapter 6

PowerPlant X 1.0 API Reference Class Documentation

6.1 PPx::AccessibleGetAllActionNamesDoer Class Reference

`#include <PPxAccessibilityEvents.h>`

Inheritance diagram for PPx::AccessibleGetAllActionNamesDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleGetAllActionNames >
PPx::AccessibleGetAllActionNamesDoer

6.1.1 Detailed Description

Returns names of all supported actions.

Definition at line 124 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleGetAllActionNames** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFMutableArrayRef &ioActionNames)=0

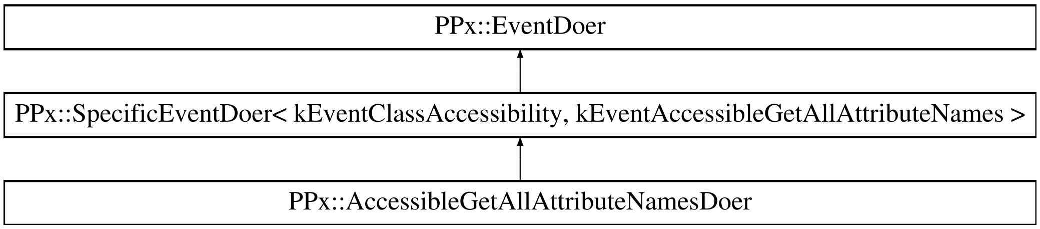
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.2 PPx::AccessibleGetAllAttributeNamesDoer Class Reference

```
#include <PPxAccessibilityEvents.h>
```

Inheritance diagram for PPx::AccessibleGetAllAttributeNamesDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleGetAllAttributeNames >
PPx::AccessibleGetAllAttributeNamesDoer

6.2.1 Detailed Description

Returns names of all supported attributes.

Definition at line 55 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleGetAllAttributeNames** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFMutableArrayRef &ioAttrNames)=0

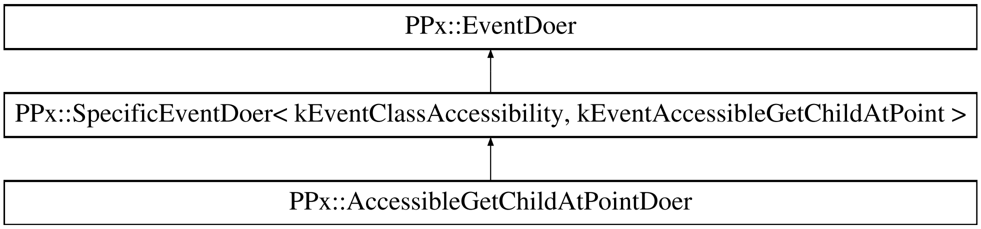
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.3 PPx::AccessibleGetChildAtPointDoer Class Reference

`#include <PPxAccessibilityEvents.h>`

Inheritance diagram for PPx::AccessibleGetChildAtPointDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleGetChildAtPoint >
PPx::AccessibleGetChildAtPointDoer

6.3.1 Detailed Description

Returns child object hit by a specified global mouse point.

Definition at line 20 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleGetChildAtPoint** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, const HIPoint &inPoint, AXUIElementRef &outChild)=0

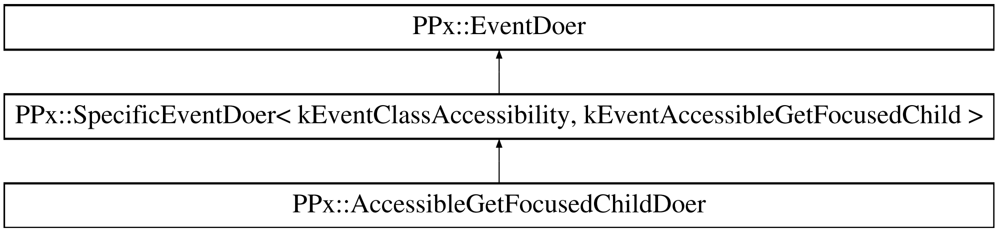
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.4 PPx::AccessibleGetFocusedChildDoer Class Reference

```
#include <PPxAccessibilityEvents.h>
```

Inheritance diagram for PPx::AccessibleGetFocusedChildDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleGetFocusedChild >
PPx::AccessibleGetFocusedChildDoer

6.4.1 Detailed Description

Returns child which is part of the focus chain.

Definition at line 38 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleGetFocusedChild** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, AXUIElementRef &outChild)=0

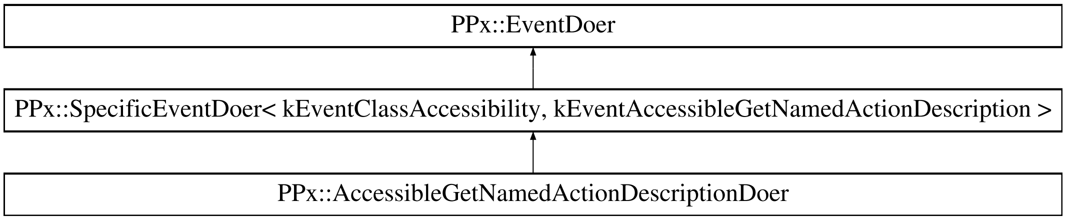
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.5 PPx::AccessibleGetNamedActionDescriptionDoer Class Reference

`#include <PPxAccessibilityEvents.h>`

Inheritance diagram for PPx::AccessibleGetNamedActionDescriptionDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleGetNamedActionDescription >
PPx::AccessibleGetNamedActionDescriptionDoer

6.5.1 Detailed Description

Returns a description of an action's significance.

Definition at line 158 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleGetNamedActionDescription** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFStringRef inActionName, CFMutableStringRef ioDescription)=0

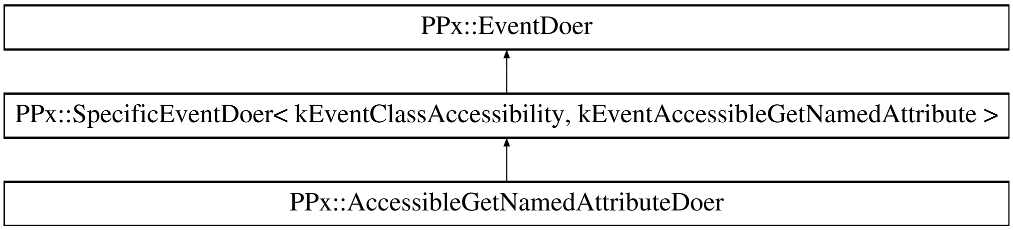
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.6 PPx::AccessibleGetNamedAttributeDoer Class Reference

`#include <PPxAccessibilityEvents.h>`

Inheritance diagram for PPx::AccessibleGetNamedAttributeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleGetNamedAttribute >
PPx::AccessibleGetNamedAttributeDoer

6.6.1 Detailed Description

Returns the value of an attribute.

Definition at line 72 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleGetNamedAttribute** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFStringRef inAttrName)=0

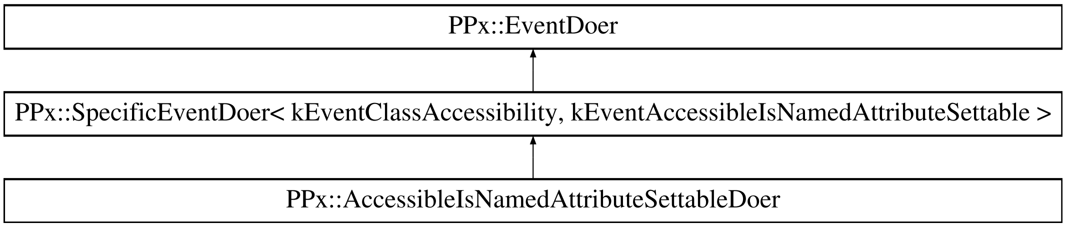
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.7 PPx::AccessibleIsNamedAttributeSettableDoer Class Reference

`#include <PPxAccessibilityEvents.h>`

Inheritance diagram for PPx::AccessibleIsNamedAttributeSettableDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleIsNamedAttributeSettable >
PPx::AccessibleIsNamedAttributeSettableDoer

6.7.1 Detailed Description

Returns whether an attriubute is settable.

Definition at line 106 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleIsNamedAttributeSettable** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFStringRef inAttrName, bool &outIsSettable)=0

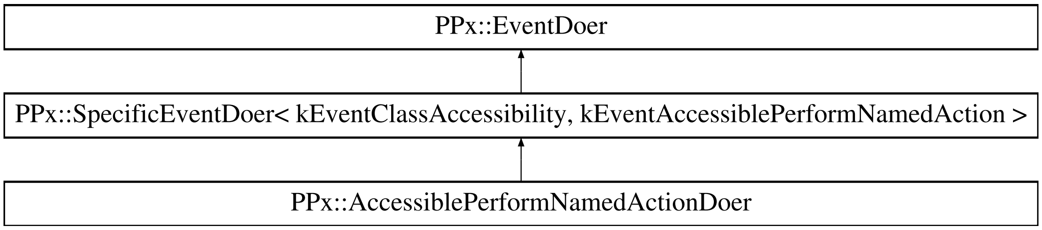
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.8 PPx::AccessiblePerformNamedActionDoer Class Reference

```
#include <PPxAccessibilityEvents.h>
```

Inheritance diagram for PPx::AccessiblePerformNamedActionDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessiblePerformNamedAction >
PPx::AccessiblePerformNamedActionDoer

6.8.1 Detailed Description

Performs an action.

Definition at line 141 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessiblePerformNamedAction** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFStringRef inActionName)=0

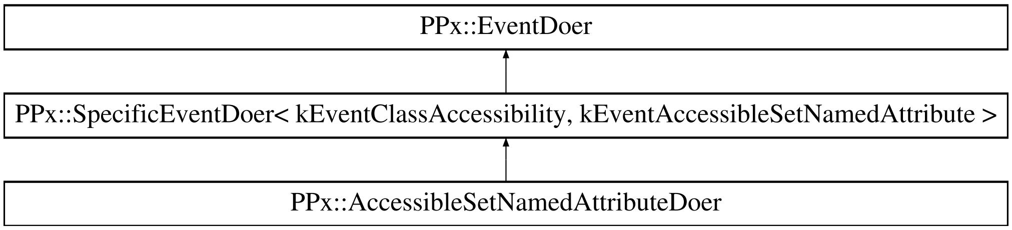
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.9 PPx::AccessibleSetNamedAttributeDoer Class Reference

`#include <PPxAccessibilityEvents.h>`

Inheritance diagram for PPx::AccessibleSetNamedAttributeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAccessibility, kEventAccessibleSetNamedAttribute >
PPx::AccessibleSetNamedAttributeDoer

6.9.1 Detailed Description

Sets the value of an attribute.

Definition at line 89 of file PPxAccessibilityEvents.h.

Protected Member Functions

- virtual OSStatus **DoAccessibleSetNamedAttribute** (SysCarbonEvent &ioEvent, AXUIElementRef inAccessible, CFStringRef inAttrName)=0

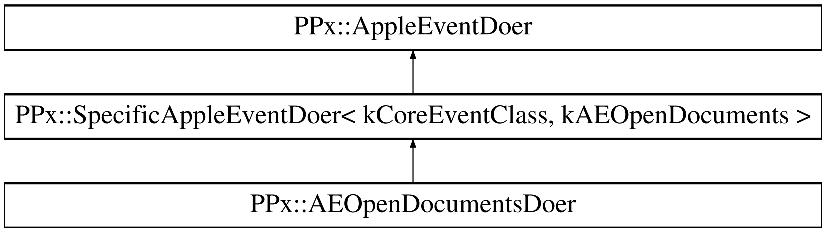
The documentation for this class was generated from the following files:

- PPxAccessibilityEvents.h
- PPxAccessibilityEvents.cp

6.10 PPx::AEOpenDocumentsDoer Class Reference

`#include <PPxAEStandardEvents.h>`

Inheritance diagram for PPx::AEOpenDocumentsDoer::


PPx::AppleEventDoer
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEOpenDocuments >
PPx::AEOpenDocumentsDoer

6.10.1 Detailed Description

Handles request to open a list of documents.

Definition at line 43 of file PPxAEStandardEvents.h.

Protected Member Functions

- virtual OSStatus **DoAEOpenDocuments** (const AutoAEDesc &inAppleEvent, AutoAEDesc &outAEReply)=0

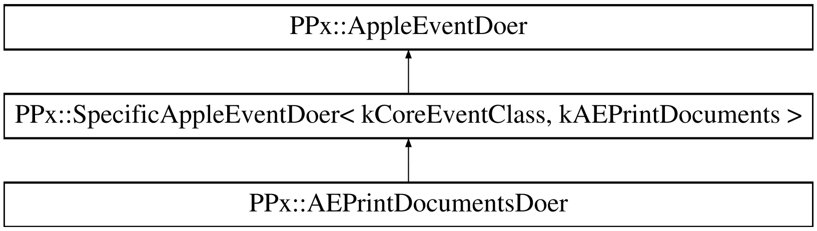
The documentation for this class was generated from the following files:

- PPxAEStandardEvents.h
- PPxAEStandardEvents.cp

6.11 PPx::AEPrintDocumentsDoer Class Reference

```
#include <PPxAEStandardEvents.h>
```

Inheritance diagram for PPx::AEPrintDocumentsDoer::


PPx::AppleEventDoer
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEPrintDocuments >
PPx::AEPrintDocumentsDoer

6.11.1 Detailed Description

Handles request to print a list of documents.

Definition at line 63 of file PPxAEStandardEvents.h.

Protected Member Functions

- virtual OSStatus **DoAEPrintDocuments** (const AutoAEDesc &inAppleEvent, AutoAEDesc &outAEReply)=0

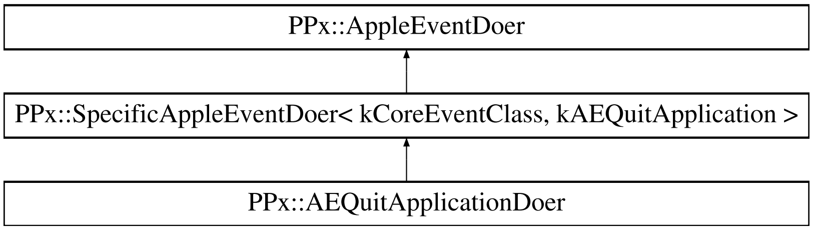
The documentation for this class was generated from the following files:

- PPxAEStandardEvents.h
- PPxAEStandardEvents.cp

6.12 PPx::AEQuitApplicationDoer Class Reference

`#include <PPxAEStandardEvents.h>`

Inheritance diagram for PPx::AEQuitApplicationDoer::


PPx::AppleEventDoer
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEQuitApplication >
PPx::AEQuitApplicationDoer

6.12.1 Detailed Description

Handles request to quit the application.

Definition at line 104 of file PPxAEStandardEvents.h.

Protected Member Functions

- virtual OSStatus **DoAEQuitApplication** (const AutoAEDesc &inAppleEvent, AutoAEDesc &outAEReply)=0

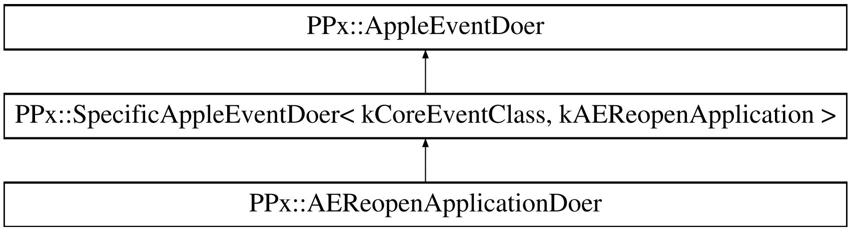
The documentation for this class was generated from the following files:

- PPxAEStandardEvents.h
- PPxAEStandardEvents.cp

6.13 PPx::AEReopenApplicationDoer Class Reference

`#include <PPxAEStandardEvents.h>`

Inheritance diagram for PPx::AEReopenApplicationDoer::


PPx::AppleEventDoer
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEReopenApplication >
PPx::AEReopenApplicationDoer

6.13.1 Detailed Description

Handles notification that an already running application has been reactivated from the Finder.

Definition at line 84 of file PPxAEStandardEvents.h.

Protected Member Functions

- virtual OSStatus **DoAEReopenApplication** (const AutoAEDesc &inAppleEvent, AutoAEDesc &outAEReply)=0

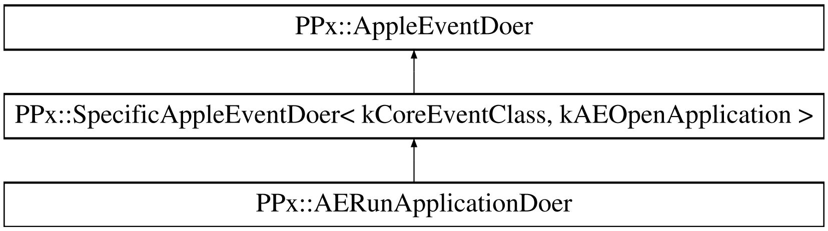
The documentation for this class was generated from the following files:

- PPxAEStandardEvents.h
- PPxAEStandardEvents.cp

6.14 PPx::AERunApplicationDoer Class Reference

`#include <PPxAEStandardEvents.h>`

Inheritance diagram for PPx::AERunApplicationDoer::


PPx::AppleEventDoer
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEOpenApplication >
PPx::AERunApplicationDoer

6.14.1 Detailed Description

Handles notification the application was launched directly and not from opening a document.

Definition at line 23 of file PPxAEStandardEvents.h.

Protected Member Functions

- virtual OSStatus **DoAERunApplication** (const AutoAEDesc &inAppleEvent, AutoAEDesc &outAEReply)=0

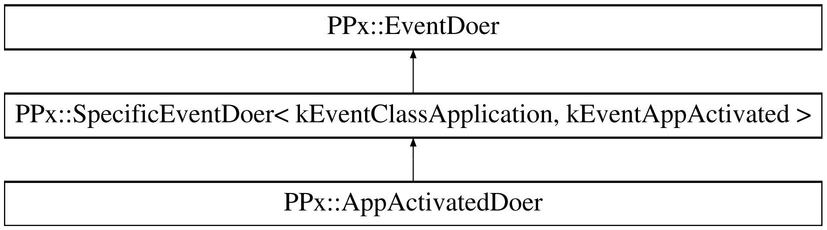
The documentation for this class was generated from the following files:

- PPxAEStandardEvents.h
- PPxAEStandardEvents.cp

6.15 PPx::AppActivatedDoer Class Reference

```
#include <PPxApplicationEvents.h>
```

Inheritance diagram for PPx::AppActivatedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppActivated >
PPx::AppActivatedDoer

6.15.1 Detailed Description

Handles notification that an application has resumed.

Definition at line 20 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppActivated** (SysCarbonEvent &ioEvent, WindowRef inClickedWindow)=0

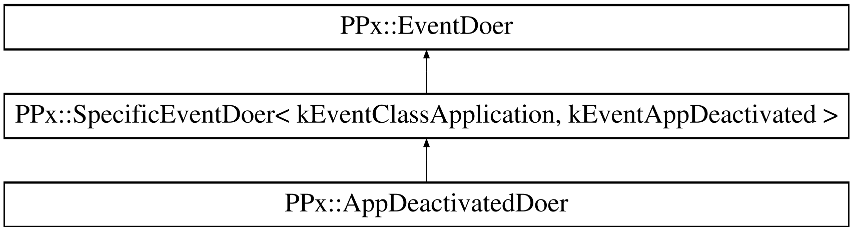
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.16 PPx::AppDeactivatedDoer Class Reference

```
#include <PPxApplicationEvents.h>
```

Inheritance diagram for PPx::AppDeactivatedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppDeactivated >
PPx::AppDeactivatedDoer

6.16.1 Detailed Description

Handles notification that an application has suspended.

Definition at line 36 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppDeactivated** (SysCarbonEvent &ioEvent)=0

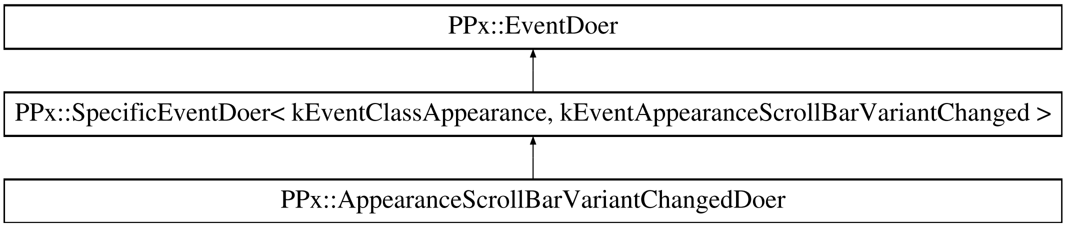
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.17 PPx::AppearanceScrollBarVariantChangedDoer Class Reference

`#include <PPxMiscellaneousEvents.h>`

Inheritance diagram for PPx::AppearanceScrollBarVariantChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassAppearance, kEventAppearanceScrollBarVariantChanged >
PPx::AppearanceScrollBarVariantChangedDoer

6.17.1 Detailed Description

Notification that the scroll bar variant has changed.

Definition at line 54 of file PPxMiscellaneousEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppearanceScrollBarVariantChanged** (SysCarbonEvent &ioEvent, SInt16 inVariant)=0

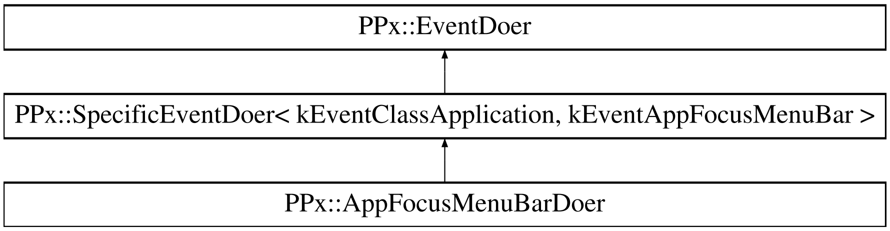
The documentation for this class was generated from the following files:

- PPxMiscellaneousEvents.h
- PPxMiscellaneousEvents.cp

6.18 PPx::AppFocusMenuBarDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppFocusMenuBarDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusMenuBar >
PPx::AppFocusMenuBarDoer

6.18.1 Detailed Description

Handles request to set the keyboard focus to the menu bar.

Definition at line 131 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppFocusMenuBar** (SysCarbonEvent &ioEvent, UInt32 inKeyModifiers)=0

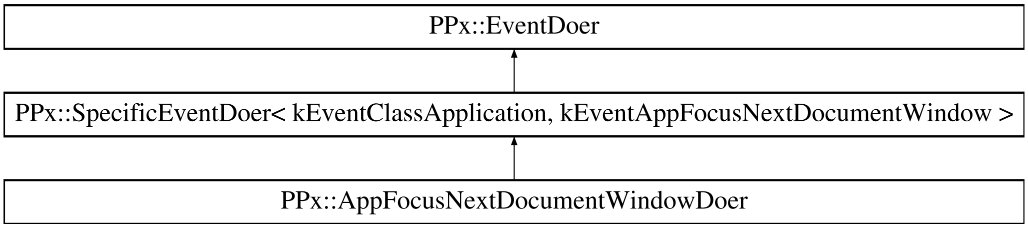
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.19 PPx::AppFocusNextDocumentWindowDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppFocusNextDocumentWindowDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusNextDocumentWindow >
PPx::AppFocusNextDocumentWindowDoer

6.19.1 Detailed Description

Handles request to set the keyboard focus to the next document window.

Definition at line 148 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppFocusNextDocumentWindow** (SysCarbonEvent &ioEvent, UInt32 inKeyModifiers)=0

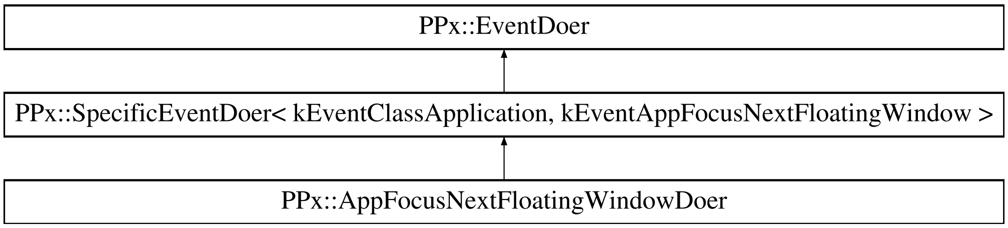
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.20 PPx::AppFocusNextFloatingWindowDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppFocusNextFloatingWindowDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusNextFloatingWindow >
PPx::AppFocusNextFloatingWindowDoer

6.20.1 Detailed Description

Handles request to set the keyboard focus to the next floating window.

Definition at line 164 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppFocusNextFloatingWindow** (SysCarbonEvent &ioEvent, UInt32 inKeyModifiers)=0

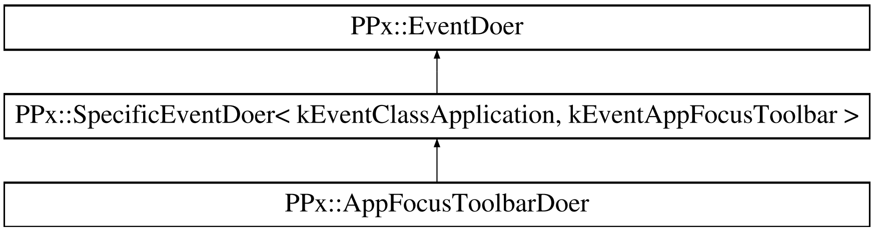
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.21 PPx::AppFocusToolbarDoer Class Reference

```
#include <PPxApplicationEvents.h>
```

Inheritance diagram for PPx::AppFocusToolbarDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppFocusToolbar >
PPx::AppFocusToolbarDoer

6.21.1 Detailed Description

Handles request to set the keyboard focus to the toolbar in the currently focused window.

Definition at line 181 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppFocusToolbar** (SysCarbonEvent &ioEvent, UInt32 inKeyModifiers)=0

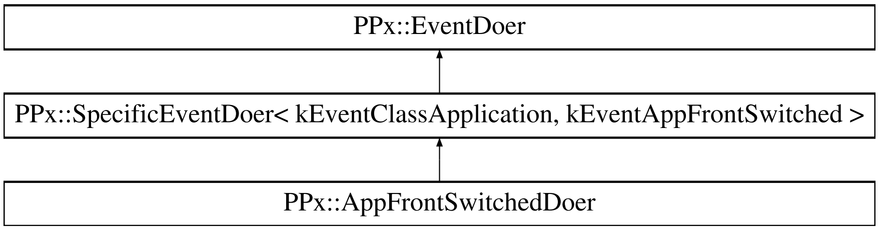
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.22 PPx::AppFrontSwitchedDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppFrontSwitchedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppFrontSwitched >
PPx::AppFrontSwitchedDoer

6.22.1 Detailed Description

Handles notification that the active application has changed.

Definition at line 115 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppFrontSwitched** (SysCarbonEvent &ioEvent, const ProcessSerialNumber &inPSN)=0

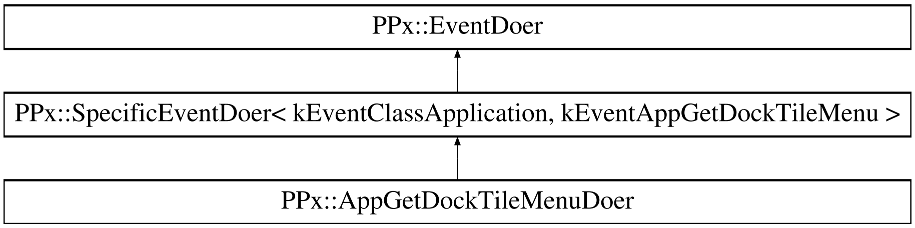
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.23 PPx::AppGetDockTileMenuDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppGetDockTileMenuDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppGetDockTileMenu >
PPx::AppGetDockTileMenuDoer

6.23.1 Detailed Description

Returns the menu to display from an application's dock tile.

Definition at line 197 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppGetDockTileMenu** (SysCarbonEvent &ioEvent, MenuRef &outMenu)=0

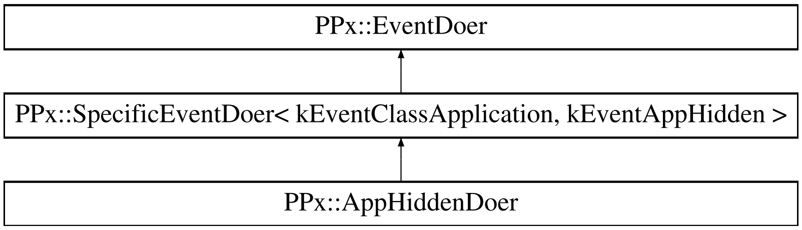
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.24 PPx::AppHiddenDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppHiddenDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppHidden >
PPx::AppHiddenDoer

6.24.1 Detailed Description

Handles notification that an application has been hidden.

Definition at line 213 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppHidden** (SysCarbonEvent &ioEvent)=0

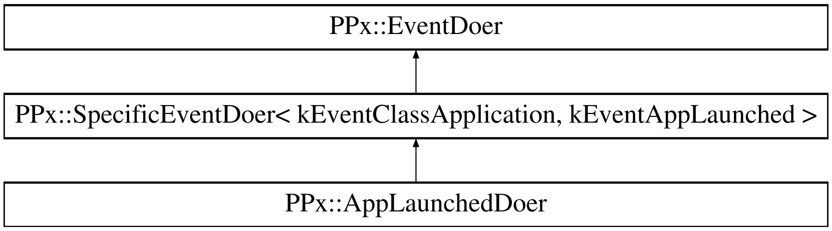
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.25 PPx::AppLaunchedDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppLaunchedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppLaunched >
PPx::AppLaunchedDoer

6.25.1 Detailed Description

Handles notification that another application has launched.

Definition at line 83 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppLaunched** (SysCarbonEvent &ioEvent, const ProcessSerialNumber &inPSN)=0

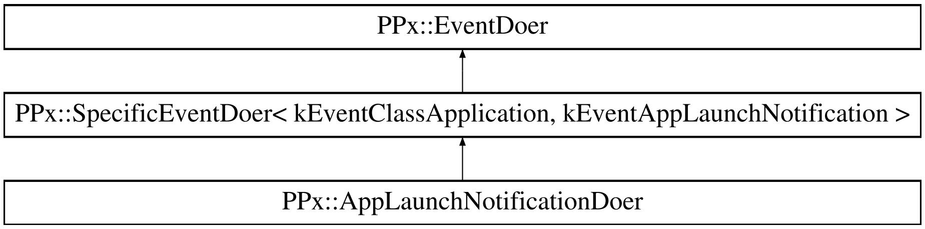
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.26 PPx::AppLaunchNotificationDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppLaunchNotificationDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppLaunchNotification >
PPx::AppLaunchNotificationDoer

6.26.1 Detailed Description

Handles notification that an application we launched asynchronously has actually launched.

Definition at line 65 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppLaunchNotification** (SysCarbonEvent &ioEvent, const ProcessSerialNumber &inPSN, UInt32 inLaunchRefCon, OSStatus inLaunchError)=0

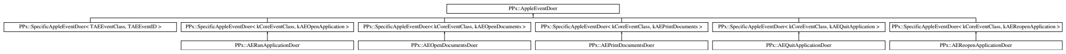
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.27 PPx::AppleEventDoer Class Reference

`#include <PPxAppleEventDoer.h>`

Inheritance diagram for PPx::AppleEventDoer::


PPx::AppleEventDoer
PPx::SpecificAppleEventDoer< TAEEventClass, TAEEventID >
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEOpenApplication >
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEOpenDocuments >
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEPrintDocuments >
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEQuitApplication >
PPx::SpecificAppleEventDoer< kCoreEventClass, kAEReopenApplication >
PPx::AERunApplicationDoer
PPx::AEOpenDocumentsDoer
PPx::AEPrintDocumentsDoer
PPx::AEQuitApplicationDoer
PPx::AEReopenApplicationDoer

6.27.1 Detailed Description

Abstract class for an Apple Event handler.

Definition at line 24 of file PPxAppleEventDoer.h.

Public Member Functions

- AppleEventDoer ()

  Default constructor.

- AppleEventDoer (AEEventClass inEventClass, AEEventID inEventID, bool inIsSystemHandler=false)

  Constructs from an AppleEvent class and ID and installs a handler.

- virtual ∼AppleEventDoer ()

  Destructor.

- void Install (AEEventClass inEventClass, AEEventID inEventID, bool inIsSystemHandler=false)

  Installs handler for an Apple Event.

- void Remove ()

  Removes handler for the Apple Event.

- OSStatus Invoke (const AutoAEDesc &inAppleEvent, AutoAEDesc &outAEReply)

  Calls function to handle an AppleEvent.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 PPx::AppleEventDoer::AppleEventDoer (AEEventClass *inEventClass*, AEEventID *inEventID*, bool *inIsSystemHandler* = false)

Constructs from an AppleEvent class and ID and installs a handler.

Parameters:

inEventClass AppleEvent class

inEventID AppleEvent ID

inIsSystemHandler Whether handler is system-wide (vs. local)

Definition at line 85 of file PPxAppleEventDoer.cp.

References Install().

6.27.3 Member Function Documentation

6.27.3.1 void PPx::AppleEventDoer::Install (AEEventClass *inEventClass*, AEEventID *inEventID*, bool *inIsSystemHandler* = false)

Installs handler for an Apple Event.

Parameters:

inEventClass AppleEvent class

inEventID AppleEvent ID

inIsSystemHandler Whether the handler is system-wide or local

Definition at line 115 of file PPxAppleEventDoer.cp.

References PPx::SysAEHandler::Install().

Referenced by AppleEventDoer().

6.27.3.2 OSStatus PPx::AppleEventDoer::Invoke (const AutoAEDesc & *inAppleEvent*, AutoAEDesc & *outAEReply*)

Calls function to handle an AppleEvent.

Parameters:

inAppleEvent AppleEvent to handle

outAEReply Reply AppleEvent

Returns:

OS error code

Definition at line 148 of file PPxAppleEventDoer.cp.

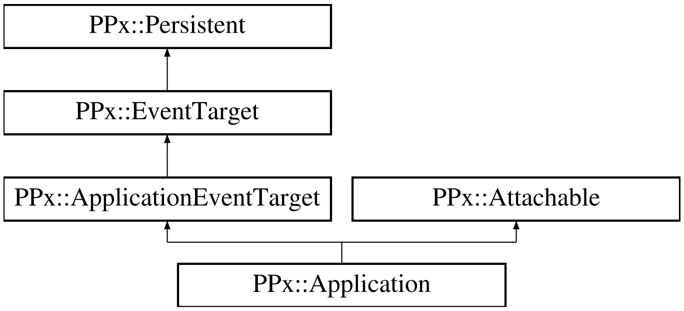
The documentation for this class was generated from the following files:

- PPxAppleEventDoer.h
- PPxAppleEventDoer.cp

6.28 PPx::Application Class Reference

`#include <PPxApplication.h>`

Inheritance diagram for PPx::Application::



6.28.1 Detailed Description

An executable program.

Definition at line 23 of file PPxApplication.h.

Public Member Functions

- Application ()

  Default constructor.

- virtual ∼Application ()

  Destructor.

- void Run ()

  Run the main applicaton event loop.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.28.2 Member Function Documentation

6.28.2.1 void PPx::Application::InitState (const DataReader & *inReader*) [protected, virtual]

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Persistent.

Definition at line 63 of file PPxApplication.cp.

References PPx::Attachable::ReadAttachments().

6.28.2.2 void PPx::Application::WriteState (DataWriter & *ioWriter*) const [protected, virtual]

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Definition at line 78 of file PPxApplication.cp.

References PPx::Attachable::WriteAttachments().

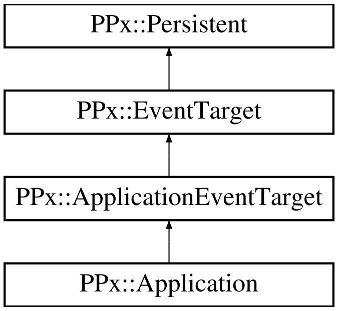
The documentation for this class was generated from the following files:

- PPxApplication.h
- PPxApplication.cp

6.29 PPx::ApplicationEventTarget Class Reference

`#include <PPxEventTarget.h>`

Inheritance diagram for PPx::ApplicationEventTarget::



6.29.1 Detailed Description

The top-level Carbon Event target.

Definition at line 50 of file PPxEventTarget.h.

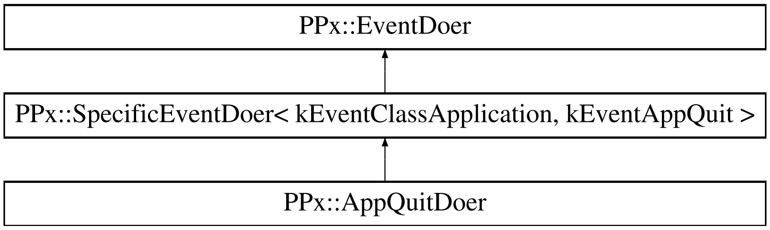
The documentation for this class was generated from the following files:

- PPxEventTarget.h
- PPxEventTarget.cp

6.30 PPx::AppQuitDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppQuitDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppQuit >
PPx::AppQuitDoer

6.30.1 Detailed Description

Handles a request to quit an application.

Definition at line 50 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppQuit** (SysCarbonEvent &ioEvent)=0

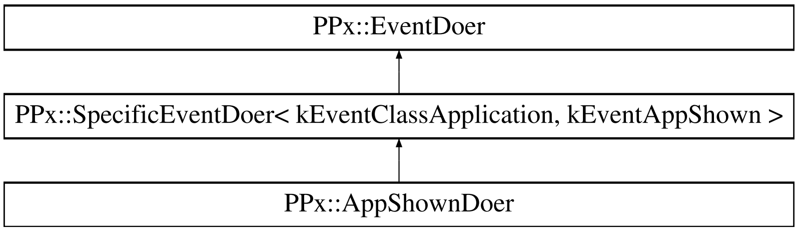
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.31 PPx::AppShownDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppShownDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppShown >
PPx::AppShownDoer

6.31.1 Detailed Description

Handles notification that an application has been shown.

Definition at line 227 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppShown** (SysCarbonEvent &ioEvent)=0

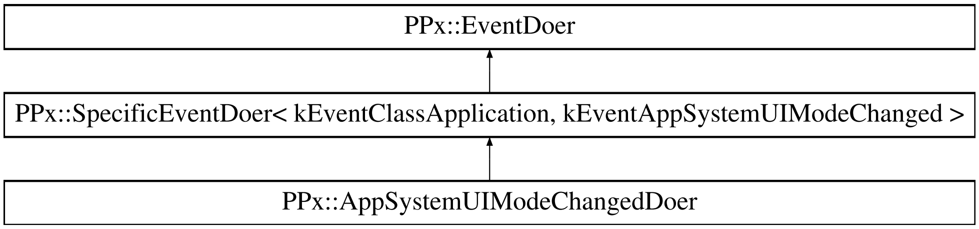
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.32 PPx::AppSystemUIModeChangedDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppSystemUIModeChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppSystemUIModeChanged >
PPx::AppSystemUIModeChangedDoer

6.32.1 Detailed Description

Handles notification that the system UI mode of the front application has changed.

Definition at line 242 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppSystemUIModeChanged** (SysCarbonEvent &ioEvent, UInt32 inUIMode)=0

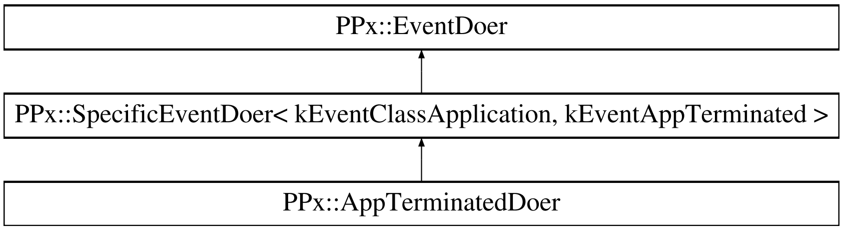
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.33 PPx::AppTerminatedDoer Class Reference

`#include <PPxApplicationEvents.h>`

Inheritance diagram for PPx::AppTerminatedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassApplication, kEventAppTerminated >
PPx::AppTerminatedDoer

6.33.1 Detailed Description

Handles notification that another application has terminated.

Definition at line 99 of file PPxApplicationEvents.h.

Protected Member Functions

- virtual OSStatus **DoAppTerminated** (SysCarbonEvent &ioEvent, const ProcessSerialNumber &inPSN)=0

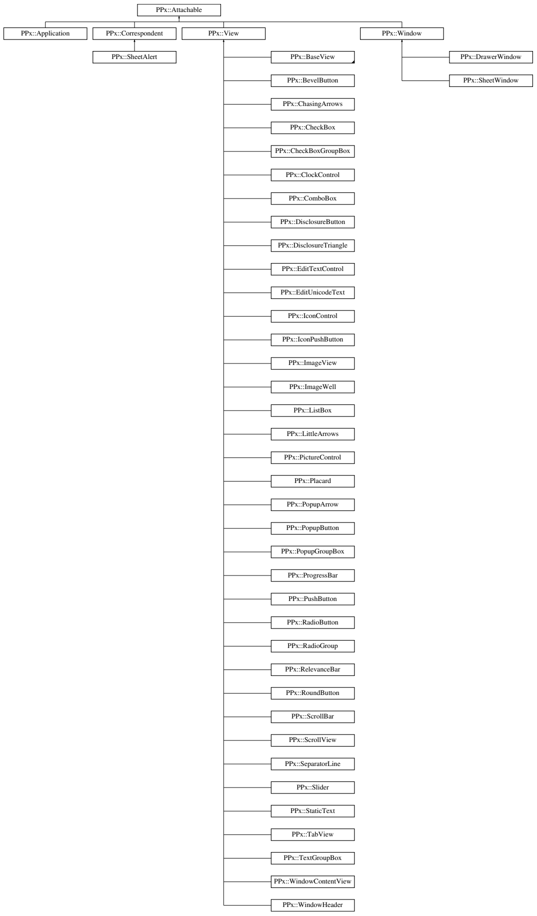
The documentation for this class was generated from the following files:

- PPxApplicationEvents.h
- PPxApplicationEvents.cp

6.34 PPx::Attachable Class Reference

`#include <PPxAttachable.h>`

Inheritance diagram for PPx::Attachable::



6.34.1 Detailed Description

Class for objects which have an associated list of Attachments.

Attachable is designed to be a mix-in base class or data member of a Persistent subclass that wants to support Attachments.

Definition at line 29 of file PPxAttachable.h.

Public Member Functions

- Attachable ()

  Default constructor.

- Attachable (const Attachable &inOriginal)

  Copy constructor.

- virtual ∼Attachable ()

  Destructor.

- Attachable & operator= (const Attachable &inSource)

  Assignment operator.

- void AddAttachment (Attachment ∗inAttachment)

  Adds an Attachment.

- void RemoveAttachment (Attachment ∗inAttachment)

  Removes an Attachment.

- void RemoveAllAttachments ()

  Remove and deletes all Attachments.

- Attachment ∗ FindAttachmentByID (ObjectIDT inID) const

  Returns the Attachment with the specified object ID.

Protected Member Functions

- void ReadAttachments (const DataReader &inReader)

  Reads Attachment objects from a DataReader.

- void WriteAttachments (DataWriter &ioWriter) const

  Writes Attachment objects to a DataWriter.

6.34.2 Member Function Documentation

6.34.2.1 void PPx::Attachable::AddAttachment (Attachment ∗ *inAttachment*)

Adds an Attachment.

Parameters:

inAttachment Attachment object to add

The Attachable takes ownership of the Attachment and is responsible for deleting it.

Definition at line 82 of file PPxAttachable.cp.

6.34.2.2 Attachment ∗ PPx::Attachable::FindAttachmentByID (ObjectIDT *inID*) const

Returns the Attachment with the specified object ID.

Parameters:

inID Object ID of Attachment to find

Returns:

Pointer to Attachment object

Return value is nil if there is no Attachment with the specified ID

Definition at line 153 of file PPxAttachable.cp.

6.34.2.3 void PPx::Attachable::ReadAttachments (const DataReader & *inReader*) `[protected]`

Reads Attachment objects from a DataReader.

Parameters:

inReader DataReader from which to get Attahcments

Note:

Attachable is not a subclass of Persistent. An Attachable subclass that also inherits from Persistent should call ReadAttahcments from its InitState function.

Definition at line 184 of file PPxAttachable.cp.

References PPx::DataReader::ContainsKey(), and PPx::DataReader::ReadObjectContainer().

Referenced by PPx::WindowContentView::InitState(), PPx::Window::InitState(), PPx::Correspondent::InitState(), PPx::Application::InitState(), and PPx::View::InitViewState().

6.34.2.4 void PPx::Attachable::RemoveAttachment (Attachment ∗ *inAttachment*)

Removes an Attachment.

Parameters:

inAttachment Attachment object to remove

Attachable gives up ownership of the Attachment, so the caller is resposible for deleting it.

Definition at line 104 of file PPxAttachable.cp.

6.34.2.5 void PPx::Attachable::WriteAttachments (DataWriter & *ioWriter*) const [protected]

Writes Attachment objects to a DataWriter.

Parameters:

ioWriter DataWriter into which to put Attachments

Note:

Attachable is not a subclass of Persistent. An Attachable subclass that also inherits from Persistent should call WriteAttachmens from its WriteState function.

Definition at line 212 of file PPxAttachable.cp.

References PPx::DataWriter::WriteObjectContainer().

Referenced by PPx::WindowContentView::WriteState(), PPx::Window::WriteState(), PPx::View::WriteState(), PPx::Correspondent::WriteState(), and PPx::Application::WriteState().

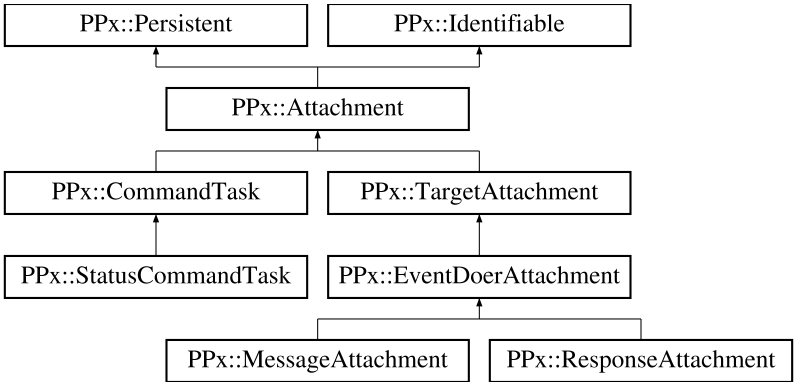
The documentation for this class was generated from the following files:

- PPxAttachable.h
- PPxAttachable.cp

6.35 PPx::Attachment Class Reference

`#include <PPxAttachment.h>`

Inheritance diagram for PPx::Attachment::



6.35.1 Detailed Description

Abstract class for identifiable persistent objects.

Other persistent objects may store pointers to Attachments. The Object ID allows an Attachment created from persistent data to be found at runtime.

Definition at line 27 of file PPxAttachment.h.

Public Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.35.2 Member Function Documentation

6.35.2.1 void PPx::Attachment::InitState (const DataReader & *inReader*) `[virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::CommandTask, PPx::TargetAttachment, PPx::EventDoerAttachment, PPx::ResponseAttachment, and PPx::MessageAttachment.

Definition at line 26 of file PPxAttachment.cp.

References PPx::DataReader::ReadOptional(), and PPx::Identifiable::SetID().

6.35.2.2 void PPx::Attachment::WriteState (DataWriter & *ioWriter*) const `[virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::CommandTask, PPx::TargetAttachment, PPx::EventDoerAttachment, PPx::ResponseAttachment, and PPx::MessageAttachment.

Definition at line 44 of file PPxAttachment.cp.

References PPx::Identifiable::GetID(), and PPx::DataWriter::WriteValue().

The documentation for this class was generated from the following files:

- PPxAttachment.h
- PPxAttachment.cp

6.36 PPx::AutoAEDesc Class Reference

```
#include <SysAEDesc.h>
```

6.36.1 Detailed Description

Wrapper for a system Apple Event descriptor.

Implements single owner, shared use seamantics

Definition at line 25 of file SysAEDesc.h.

Public Member Functions

- AutoAEDesc ()

  Default constructor.

- AutoAEDesc (const AEDesc &inDesc)

  Constructs from an existing AEDsc.

- AutoAEDesc (const AutoAEDesc &inOther)

  Copy constructor.

- ∼AutoAEDesc ()

  Destructor.

- AutoAEDesc & operator= (const AutoAEDesc &inOther)

  Assignment operator.

- AEDesc & **GetRef** ()
- const AEDesc & **GetRef** () const
- AEDesc ∗ **GetPtr** ()
- const AEDesc ∗ **GetPtr** () const
- AEDesc Release () const

  Releases ownership of the AEDesc.

- void Reset ()

  Resets this object by disposing its AEDesc and initializing it to a null descriptor.

- void Reset (const AEDesc &inDesc)

  Resets AEDesc of this object to the input one, disposing of its current AEDesc Caller retains ownership of the AEDesc.

- void Adopt (AEDesc &inDesc)

  Takes ownership of an AEDesc.

- bool IsOwner () const

  Returns whether this object owns its AEDesc.

- DescType **GetDescType** () const
- bool **IsNull** () const
- bool **IsList** () const
- bool **IsRecord** () const
- bool **IsAppleEvent** () const
- SInt32 GetCount () const

  Returns the number of items contained by its AEDesc.

- template<typename TItem> void **GetNthItem** (SInt32 inIndex, TItem &outItem) const
- AutoAEDesc GetNthDesc (SInt32 inIndex, DescType inDesiredType=typeWildCard) const

  Returns a copy of a contained descriptor referred to by index number.

- template<typename TItem> void **GetNthItem** (SInt32 inIndex, AEKeyword &outKeyword, TItem &outItem) const
- AutoAEDesc GetNthDesc (SInt32 inIndex, DescType inDesiredType, AEKeyword &outKeyword) const

  Returns a copy of a contained descriptor referred to by index number and passes back its keywork name.

- template<typename TParam> void **GetRequiredParam** (AEKeyword inKeyword, TParam &outParam) const
- template<typename TParam> void **GetOptionalParam** (AEKeyword inKeyword, TParam &outParam) const
- AutoAEDesc GetRequiredParamDesc (AEKeyword inKeyword, DescType inDesiredType=typeWildCard) const

  Gets a required descriptor parameter referred to by keywork name.

- AutoAEDesc GetOptionalParamDesc (AEKeyword inKeyword, DescType inDesiredType=typeWildCard) const

  Gets an optional descriptor parameter referred to by keywork name.

- template<typename TAttribute> void **GetAttribute** (AEKeyword inKeyword, TAttribute &outAttribute) const
- AutoAEDesc GetAttributeDesc (AEKeyword inKeyword, DescType inDesiredType=typeWildCard) const

  Gets the descriptor for a keyword named attribute.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 PPx::AutoAEDesc::AutoAEDesc (const AEDesc & *inDesc*) `[explicit]`

Constructs from an existing AEDsc.

Caller retains ownership of the AEDesc.

Parameters:

inDesc AEDesc to use

Definition at line 28 of file SysAEDesc.cp.

6.36.2.2 PPx::AutoAEDesc::AutoAEDesc (const AutoAEDesc & *inOther*)

Copy constructor.

Object being copied transfers its ownership rights for the AEDesc to this object. That is, if inOther was the owner, this object becomes the owner. If inOther was not the owner, this object is not the owner either.

Definition at line 44 of file SysAEDesc.cp.

References mIsOwner, and Release().

6.36.3 Member Function Documentation

6.36.3.1 void PPx::AutoAEDesc::Adopt (AEDesc & *inDesc*)

Takes ownership of an AEDesc.

Disposes of its current AEDesc.

Parameters:

inDesc AEDesc to adopt

Definition at line 143 of file SysAEDesc.cp.

6.36.3.2 AutoAEDesc PPx::AutoAEDesc::GetAttributeDesc (AEKeyword *inKeyword*, DescType *inDesiredType* = typeWildCard) const

Gets the descriptor for a keyword named attribute.

Parameters:

inKeyword Keyword name for attribute

inDesiredType Desired type for descriptor

Returns:

AutuAEDesc for the keyword named attribute

Throws an exception it fails to get the attribute

Definition at line 315 of file SysAEDesc.cp.

References AutoAEDesc(), and PPx_ThrowIfOSError_.

6.36.3.3 SInt32 PPx::AutoAEDesc::GetCount () const

Returns the number of items contained by its AEDesc.

Returns:

Number of items contained by its AEDesc

Definition at line 189 of file SysAEDesc.cp.

References PPx_ThrowIfOSError_.

6.36.3.4 AutoAEDesc PPx::AutoAEDesc::GetNthDesc (SInt32 *inIndex*, DescType *inDesiredType*, AEKeyword & *outKeyword*) const

Returns a copy of a contained descriptor referred to by index number and passes back its keywork name.

Parameters:

inIndex Index of descriptor to get

inDesiredType Desired type for descriptor

outKeyword Keyword name for the indexed descriptor

Returns:

AutuAEDesc for the indexed descriptor

Definition at line 233 of file SysAEDesc.cp.

References AutoAEDesc(), and PPx_ThrowIfOSError_.

6.36.3.5 AutoAEDesc PPx::AutoAEDesc::GetNthDesc (SInt32 *inIndex*, DescType *inDesiredType* = typeWildCard) const

Returns a copy of a contained descriptor referred to by index number.

Parameters:

inIndex Index of descriptor to get

inDesiredType Desired type for descriptor

Returns:

AutuAEDesc for the indexed descriptor

Definition at line 210 of file SysAEDesc.cp.

6.36.3.6 AutoAEDesc PPx::AutoAEDesc::GetOptionalParamDesc (AEKeyword *inKeyword*, DescType *inDesiredType* = typeWildCard) const

Gets an optional descriptor parameter referred to by keywork name.

Parameters:

inKeyword Keyword name for parameter

inDesiredType Desired type for descriptor

Returns:

AutuAEDesc for the keyword named descriptor

Returned AEDesc is a null descriptor if the parameter does not exist

Definition at line 288 of file SysAEDesc.cp.

References AutoAEDesc().

6.36.3.7 AutoAEDesc PPx::AutoAEDesc::GetRequiredParamDesc (AEKeyword *inKeyword*, DescType *inDesiredType* = typeWildCard) const

Gets a required descriptor parameter referred to by keywork name.

Parameters:

inKeyword Keyword name for parameter

inDesiredType Desired type for descriptor

Returns:

AutuAEDesc for the keyword named descriptor

Throws an exception it fails to get the parameter

Definition at line 261 of file SysAEDesc.cp.

References AutoAEDesc(), and PPx_ThrowIfOSError_.

6.36.3.8 bool PPx::AutoAEDesc::IsOwner () const

Returns whether this object owns its AEDesc.

Returns:

Whether this object owns its AEDesc

Definition at line 163 of file SysAEDesc.cp.

6.36.3.9 AutoAEDesc & PPx::AutoAEDesc::operator= (const AutoAEDesc & *inOther*)

Assignment operator.

Object being copied transfers its ownership rights for the AEDesc to this object. That is, if inOther was the owner, this object becomes the owner. If inOther was not the owner, this object does not become the owner.

Definition at line 72 of file SysAEDesc.cp.

References mAEDesc, and mIsOwner.

6.36.3.10 AEDesc PPx::AutoAEDesc::Release () const

Releases ownership of the AEDesc.

If this object was the owner of the AEDesc, caller becomes the new owner of the AEDesc.

Returns:

AEDesc used by this object

Definition at line 94 of file SysAEDesc.cp.

Referenced by AutoAEDesc().

6.36.3.11 void PPx::AutoAEDesc::Reset (const AEDesc & *inDesc*)

Resets AEDesc of this object to the input one, disposing of its current AEDesc Caller retains ownership of the AEDesc.

Parameters:

inDesc AEDesc to use

Definition at line 124 of file SysAEDesc.cp.

The documentation for this class was generated from the following files:

- SysAEDesc.h
- SysAEDesc.cp

6.37 PPx::AutoHandle Class Reference

```
#include <PPxMemoryUtils.h>
```

6.37.1 Detailed Description

Manages ownership of Toolbox Handle data block.

Definition at line 257 of file PPxMemoryUtils.h.

Public Member Functions

- AutoHandle ()

  Default constructor.

- AutoHandle (Handle inHandle)

  Constructs from a Handle which becomes the owned Handle.

- ~AutoHandle ()

  Destructor.

- operator Handle () const

  Returns the Handle.

- Handle Get () const

  Returns the Handle.

- void Reset ()

  Disposes existing Handle and sets owned Handle to nil.

- void Reset (Handle inHandle)

  Disposes existing Handle and takes ownership of input Handle.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 PPx::AutoHandle::AutoHandle () `[inline]`

Default constructor.

Owned Handle is nil.

Definition at line 288 of file PPxMemoryUtils.h.

6.37.2.2 PPx::AutoHandle::AutoHandle (Handle *inHandle*) `[inline, explicit]`

Constructs from a Handle which becomes the owned Handle.

Parameters:

inHandle Object takes ownership of this Handle

Definition at line 302 of file PPxMemoryUtils.h.

6.37.2.3 PPx::AutoHandle::∼AutoHandle () `[inline]`

Destructor.

Disposes of the owned Handle

Definition at line 315 of file PPxMemoryUtils.h.

References Reset().

6.37.3 Member Function Documentation

6.37.3.1 Handle PPx::AutoHandle::Get () const `[inline]`

Returns the Handle.

Returns:

Handle owned by the AutoHandle

Definition at line 343 of file PPxMemoryUtils.h.

6.37.3.2 PPx::AutoHandle::operator Handle () const `[inline]`

Returns the Handle.

Returns:

Handle owned by the AutoHandle

Definition at line 329 of file PPxMemoryUtils.h.

6.37.3.3 void PPx::AutoHandle::Reset (Handle *inHandle*) `[inline]`

Disposes existing Handle and takes ownership of input Handle.

Parameters:

inHandle Object takes ownership of this Handle

Definition at line 372 of file PPxMemoryUtils.h.

References Reset().

The documentation for this class was generated from the following file:

- PPxMemoryUtils.h

6.38 PPx::AutoNavReply Class Reference

```
#include <PPxNavServices.h>
```

6.38.1 Detailed Description

Manages ownership of a Toolbox NavReplyRecord.

The constructor fills in the record and the destructor disposes of it.

Definition at line 100 of file PPxNavServices.h.

Public Member Functions

- AutoNavReply (NavDialogRef inNavDialog)

  Constructor.

- ~AutoNavReply ()

  Destructor.

- const NavReplyRecord & Get () const

  Returns a reference to the reply record.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 PPx::AutoNavReply::AutoNavReply (NavDialogRef *inNavDialog*)

Constructor.

Parameters:

inNavDialog Nav dialog for which to get the reply record

Definition at line 126 of file PPxNavServices.cp.

References PPx_ThrowIfOSError_.

6.38.3 Member Function Documentation

6.38.3.1 const NavReplyRecord & PPx::AutoNavReply::Get () const

Returns a reference to the reply record.

Returns:

Reference to the reply record

Definition at line 153 of file PPxNavServices.cp.

The documentation for this class was generated from the following files:

- PPxNavServices.h
- PPxNavServices.cp

6.39 PPx::AutoRefCount< TObject > Class Template Reference

```
#include <PPxRetained.h>
```

6.39.1 Detailed Description

template<class TObject> class PPx::AutoRefCount< TObject >

Template class for automatically reference counting objects.

Definition at line 331 of file PPxRetained.h.

Public Member Functions

- AutoRefCount (TObject ∗inObject)

  Constructs from a pointer to an object.

- AutoRefCount (const AutoRefCount &inOriginal)

  Copy constructor.

- template<class T> AutoRefCount (const AutoRefCount< T > &inOriginal)

  Member template overload of copy constructor.

- ∼AutoRefCount ()

  Destructor.

- AutoRefCount & operator= (const AutoRefCount &inSource)

  Assignment operator.

- template<class T> AutoRefCount & operator= (const AutoRefCount< T > &inSource)

  Member template overload of assignment operator.

- TObject ∗ Get () const

  Returns a pointer to the reference counted object.

- TObject ∗ operator → () const

  Returns a pointer to the reference counted object.

- TObject & operator ∗ () const

  Returns a reference to the reference counted object.

- void Reset ()

  Releases currently retained object and reinitializes the reference counted object pointer to nil.

- void Reset (TObject ∗inObject)

  Releases currently retained object and reinitializes the reference counted object pointer to the input value.

- UInt32 GetRefCount () const

  Returns the reference count for the object.

Friends

- class AutoRefCount

  Default constructor.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 template<class TObject> PPx::AutoRefCount< TObject >::AutoRefCount (TObject ∗ *inObject*) `[explicit]`

Constructs from a pointer to an object.

Parameters:

inObject Pointer to object to reference count

Definition at line 390 of file PPxRetained.h.

References PPx::Retained::Retain().

6.39.2.2 template<class TObject> template<class T> PPx::AutoRefCount< TObject >::AutoRefCount (const AutoRefCount< T > & *inOriginal*)

Member template overload of copy constructor.

Allows copy construction of AutoRefCount<TObject> from AutoRefCount<T>, where T is a subclass of TObject.

Definition at line 430 of file PPxRetained.h.

References PPx::AutoRefCount< TObject >::mCounter, PPx::AutoRefCount< TObject >::mObject, and PPx::Retained::Retain().

6.39.3 Member Function Documentation

6.39.3.1 template<class TObject> TObject ∗ PPx::AutoRefCount< TObject >::Get () const

Returns a pointer to the reference counted object.

Returns:

Pointer to the retained object

Definition at line 514 of file PPxRetained.h.

6.39.3.2 template<class TObject> UInt32 PPx::AutoRefCount< TObject >::GetRefCount () const

Returns the reference count for the object.

Returns:

Reference count for the object

Definition at line 606 of file PPxRetained.h.

References PPx::Retained::GetRetainCount().

6.39.3.3 template<class TObject> TObject & PPx::AutoRefCount< TObject >::operator ∗ () const

Returns a reference to the reference counted object.

Returns:

Reference to the reference counted object

Note:

Behavior is undefined if object pointer is nil

Definition at line 544 of file PPxRetained.h.

6.39.3.4 template<class TObject> template<class T> AutoRefCount< TObject > & PPx::AutoRefCount< TObject >::operator= (const AutoRefCount< T > & *inSource*)

Member template overload of assignment operator.

Allows assignment of AutoRefCount<TObject> from AutoRefCount<T>, where T is a subclass of TObject.

Definition at line 488 of file PPxRetained.h.

References PPx::AutoRefCount< TObject >::mCounter, PPx::AutoRefCount< TObject >::mObject, and PPx::Retained::Retain().

6.39.3.5 template<class TObject> void PPx::AutoRefCount< TObject >::Reset (TObject ∗ *inObject*)

Releases currently retained object and reinitializes the reference counted object pointer to the input value.

Parameters:

inObject Pointer to object to reference count

Definition at line 583 of file PPxRetained.h.

References PPx::AutoRefCount< TObject >::Reset(), and PPx::Retained::Retain().

The documentation for this class was generated from the following file:

- PPxRetained.h

6.40 PPx::AutoRetained< TRetained > Class Template Reference

`#include <PPxRetained.h>`

6.40.1 Detailed Description

template<class TRetained> class PPx::AutoRetained< TRetained >

Template class for automatically retaining and releasing Retained objects.

TRetained must be a subclass of PPx::Retained (or implement its API).

Definition at line 52 of file PPxRetained.h.

Public Member Functions

- AutoRetained ()

  Default constructor.

- AutoRetained (TRetained ∗inRetained)

  Constructs from a pointer to a Retained object.

- AutoRetained (const AutoRetained &inOriginal)

  Copy constructor.

- template<class T> AutoRetained (const AutoRetained< T > &inOriginal)

  Member template overload of copy constructor.

- ~AutoRetained ()

  Destructor.

- AutoRetained & operator= (const AutoRetained &inSource)

  Assignment operator.

- template<class T> AutoRetained & operator= (const AutoRetained< T > &inSource)

  Member template overload of assignment operator.

- TRetained ∗ Get () const

  Returns a pointer to the retained object.

- TRetained ∗ operator → () const

  Returns a pointer to the retained object.

- TRetained & operator ∗ () const

  Returns a reference to the retained object.

- void Reset ()

  Reinitializes the retained object pointer to nil.

- void Reset (TRetained ∗inRetained)

  Reinitializes the retained object pointer to the input value.

- UInt32 GetRetainCount () const

  Returns the retain count (number of shared owners) for the retained object.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 template< class TRetained> PPx::AutoRetained< TRetained >::AutoRetained (TRetained ∗ *inRetained*) `[explicit]`

Constructs from a pointer to a Retained object.

Parameters:

inRetained Pointer to object to retain

Definition at line 111 of file PPxRetained.h.

6.40.2.2 template< class TRetained> template< class T> PPx::AutoRetained< TRetained >::AutoRetained (const AutoRetained< T > & *inOriginal*)

Member template overload of copy constructor.

Allows copy construction of AutoRetained<TRetained> from AutoRetained<T>, where T is a subclass of TRetained.

Definition at line 142 of file PPxRetained.h.

References PPx::AutoRetained< TRetained >::Get().

6.40.3 Member Function Documentation

6.40.3.1 template< class TRetained > TRetained ∗ PPx::AutoRetained< TRetained >::Get () const

Returns a pointer to the retained object.

Returns:

Pointer to the retained object

Definition at line 204 of file PPxRetained.h.

Referenced by PPx::AutoRetained< TRetained >::AutoRetained(), and PPx::AutoRetained< TRetained >::operator=().

6.40.3.2 template< class TRetained > UInt32 PPx::AutoRetained< TRetained >::GetRetainCount () const

Returns the retain count (number of shared owners) for the retained object.

Returns:

Retain count for the retained object

Definition at line 285 of file PPxRetained.h.

6.40.3.3 template< class TRetained > TRetained & PPx::AutoRetained< TRetained >::operator ∗ () const

Returns a reference to the retained object.

Returns:

Reference to the retained object

Note:

Behavior is undefined if retained object pointer is nil

Definition at line 236 of file PPxRetained.h.

6.40.3.4 template< class TRetained > TRetained ∗ PPx::AutoRetained< TRetained >::operator → () const

Returns a pointer to the retained object.

Returns:

Pointer to the retained object

Definition at line 219 of file PPxRetained.h.

6.40.3.5 template<class TRetained> template<class T> AutoRetained< TRetained > & PPx::AutoRetained< TRetained >::operator= (const AutoRetained< T > & *inSource*)

Member template overload of assignment operator.

Allows assignment of AutoRetained<TRetained> from AutoRetained<T>, where T is a subclass of TRetained.

Definition at line 187 of file PPxRetained.h.

References PPx::AutoRetained< TRetained >::Get(), and PPx::AutoRetained< TRetained >::Reset().

6.40.3.6 template<class TRetained> void PPx::AutoRetained< TRetained >::Reset (TRetained ∗ *inRetained*)

Reinitializes the retained object pointer to the input value.

Parameters:

inRetained Pointer to object to retain

Definition at line 265 of file PPxRetained.h.

The documentation for this class was generated from the following file:

- PPxRetained.h

6.41 PPx::AutoValueSaver< T > Class Template Reference

```
#include <PPxMemoryUtils.h>
```

6.41.1 Detailed Description

template<typename T> class PPx::AutoValueSaver< T >

Template class for automatically saving and restoring a variable's value.

Definition at line 113 of file PPxMemoryUtils.h.

Public Member Functions

- AutoValueSaver (T &ioValue)

  Constructs from an input variable.

- AutoValueSaver (T &ioValue, const T &inNewValue)

  Constructs from an input variable and a new value to set it to.

- ~AutoValueSaver ()

  Destructor.

- const T & Get () const

  Returns the saved value.

- void Restore ()

  Sets the variable to the saved value.

- void Reset ()

  Clears identity of the variable to save.

- void Reset (const T &inNewValue)

  Sets a new saved value for the variable.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 template<typename T> PPx::AutoValueSaver< T >::AutoValueSaver (T & *ioValue*) `[explicit]`

Constructs from an input variable.

Parameters:

ioValue Variable whose current value we want to save

Definition at line 149 of file PPxMemoryUtils.h.

6.41.2.2 template<typename T> PPx::AutoValueSaver< T >::AutoValueSaver (T & *ioValue*, const T & *inNewValue*)

Constructs from an input variable and a new value to set it to.

Parameters:

ioValue Variable whose current value we want to save

inNewValue New vale to which to set variable

Definition at line 167 of file PPxMemoryUtils.h.

6.41.2.3 template<typename T> PPx::AutoValueSaver< T >::~AutoValueSaver ()

Destructor.

Restores saved value of variable.

Definition at line 184 of file PPxMemoryUtils.h.

References PPx::AutoValueSaver< T >::Restore().

6.41.3 Member Function Documentation

6.41.3.1 template<typename T> const T & PPx::AutoValueSaver< T >::Get () const

Returns the saved value.

Returns:

Saved value

Definition at line 199 of file PPxMemoryUtils.h.

6.41.3.2 template<typename T> void PPx::AutoValueSaver< T >::Reset (const T & *inNewValue*)

Sets a new saved value for the variable.

Restore() and destructor will set the variable to this new value.

Parameters:

inNewValue New value to which the variable will be restored

Definition at line 244 of file PPxMemoryUtils.h.

6.41.3.3 template<typename T> void PPx::AutoValueSaver< T >::Reset ()

Clears identity of the variable to save.

Restore() and destructor will do nothing, but the saved value is still accessible via Get().

Definition at line 228 of file PPxMemoryUtils.h.

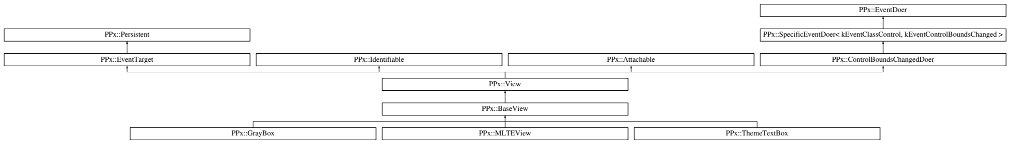
The documentation for this class was generated from the following file:

- PPxMemoryUtils.h

6.42 PPx::BaseView Class Reference

`#include <PPxBaseView.h>`

Inheritance diagram for PPx::BaseView::



6.42.1 Detailed Description

A basic view.

BaseView is a concrete View subclass that does nothing on its own. You add behavior by installing event handlers.

Definition at line 23 of file PPxBaseView.h.

Public Member Functions

- BaseView ()

  Default constructor.

- virtual ∼BaseView ()

  Destructor.

- void Initialize (const HIRect &inFrame, OptionBits inFeatures=features_None)

  Intializes from parameters.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, OptionBits inFeatures=features_None)

  Intializes from parameters.

Protected Member Functions

- virtual OptionBits GetFeatureFlags () const

  Returns the control feature flags for the view.

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.42.2 Member Function Documentation

6.42.2.1 OptionBits PPx::BaseView::GetFeatureFlags () const `[protected, virtual]`

Returns the control feature flags for the view.

Returns:

Feature flags for the view

Subclasses should override to return the features that they support

Definition at line 160 of file PPxBaseView.cp.

Referenced by Initialize(), and InitState().

6.42.2.2 void PPx::BaseView::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, OptionBits *inFeatures* = features_None)

Intializes from parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inFeatures Control features supported by this view

See <Controls.h> for a list of control features. The most common one that you may want to specify is kControlSupportsEmbedding, which allows the view to have subviews.

Definition at line 81 of file PPxBaseView.cp.

References GetFeatureFlags().

6.42.2.3 void PPx::BaseView::Initialize (const HIRect & *inFrame*, OptionBits *inFeatures* = features_None)

Intializes from parameters.

Parameters:

inFrame Bounds for view, in local coordinates of parent

inFeatures Control features supported by this view

Creates BaseView with no superview in the default state, which is invisible and enabled.

See <Controls.h> for a list of control features. The most common one that you may want to specify is kControlSupportsEmbedding, which allows the view to have subviews.

Definition at line 55 of file PPxBaseView.cp.

References GetFeatureFlags().

Referenced by PPx::GrayBox::Initialize().

6.42.2.4 void PPx::BaseView::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Note:

A subclass should call this function from its override

Reimplemented from PPx::View.

Reimplemented in PPx::MLTEView, and PPx::ThemeTextBox.

Definition at line 120 of file PPxBaseView.cp.

References GetFeatureFlags(), PPx::View::InitViewState(), and PPx::DataReader::ReadOptional().

Referenced by PPx::ThemeTextBox::InitState(), PPx::MLTEView::InitState(), and PPx::GrayBox::InitState().

6.42.2.5 void PPx::BaseView::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Note:

A subclass should call this function from its override

Reimplemented from PPx::View.

Reimplemented in PPx::GrayBox, PPx::MLTEView, and PPx::ThemeTextBox.

Definition at line 140 of file PPxBaseView.cp.

References PPx::DataWriter::WriteValue().

Referenced by PPx::ThemeTextBox::WriteState(), PPx::MLTEView::WriteState(), and PPx::GrayBox::WriteState().

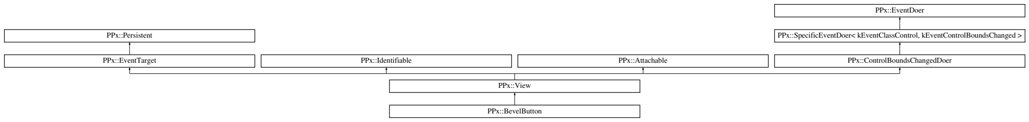
The documentation for this class was generated from the following files:

- PPxBaseView.h
- PPxBaseView.cp

6.43 PPx::BevelButton Class Reference

```
#include <PPxBevelButton.h>
```

Inheritance diagram for PPx::BevelButton::



6.43.1 Detailed Description

A system bevel button control.

Definition at line 22 of file PPxBevelButton.h.

Public Member Functions

- BevelButton ()

  Default constructor.

- virtual ∼BevelButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, ControlBevelThickness inThickness, ControlBevelButtonBehavior inButtonBehavior, const ControlButtonContentInfo &inContent, SInt16 inMenuID, ControlBevelButtonMenuBehavior inMenuBehavior, ControlBevelButtonMenuPlacement inMenuPlacement)

  Initialize from bevel button creation parameters.

- void SetTextAlignment (ControlButtonTextAlignment inAlignment)

  Sets the text alignment for the bevel button's title.

- ControlButtonTextAlignment GetTextAlignment () const

  Returns the text alignment for the bevel button's title.

- void SetTextOffset (SInt16 inOffset)

  Sets the text offset for the bevel button's title.

- SInt16 GetTextOffset () const

  Returns the text offset for the bevel button's title.

- void SetTextPlacement (ControlButtonTextPlacement inPlacement)

  Sets the text placement for the bevel button's title.

- ControlButtonTextPlacement GetTextPlacement () const

  Returns the text placement for the bevel button's title.

- void SetIconTransform (IconTransformType inTransform)

  Sets the icon transform for the bevel button's icon.

- IconTransformType GetIconTransform () const

  Returns the icon transform for the bevel button's icon.

- void SetGraphicAlignment (ControlButtonGraphicAlignment inAlignment)

  Sets the graphic alignment for the bevel button's content.

- ControlButtonGraphicAlignment GetGraphicAlignment () const

  Returns the graphic alignment for the bevel button's content.

- void SetGraphicOffset (const Point &inOffset)

  Sets the graphic offset for the bevel button's content.

- Point GetGraphicOffset () const

  Returns the graphic offset for the bevel button's content.

- void SetMenuValue (SInt16 inValue)

  Sets the value for the bevel button's popup menu.

- SInt16 GetMenuValue () const

  Returns the value for the bevel button's popup menu.

- void SetMenuRef (MenuRef inMenu)

  Sets the MenuRef for the bevel button's popup menu.

- MenuRef GetMenuRef () const

  Returns the MenuRef for the bevel button's popup menu.

- void SetCenterPopupGlyph (bool inCenter)

  Sets the center popup glyph option for the bevel button.

- bool GetCenterPopupGlyph () const

  Returns the center popup glyph option for the bevel button.

- void SetContentInfo (const ControlButtonContentInfo &inContent)

  Sets the content inforomation for the bevel button.

- void GetContentInfo (ControlButtonContentInfo &outContent) const

  Passes back the content inforomation for the bevel button.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.43.2 Member Function Documentation

6.43.2.1 bool PPx::BevelButton::GetCenterPopupGlyph () const

Returns the center popup glyph option for the bevel button.

Returns:

Whether to center the popup glyph

Definition at line 556 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.2 void PPx::BevelButton::GetContentInfo (ControlButtonContentInfo & *outContent*) const

Passes back the content inforomation for the bevel button.

Parameters:

outContent Content inforomation

Definition at line 592 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.3 ControlButtonGraphicAlignment PPx::BevelButton::GetGraphicAlignment () const

Returns the graphic alignment for the bevel button's content.

Returns:
Graphic alignment

Definition at line 411 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.4 Point PPx::BevelButton::GetGraphicOffset () const

Returns the graphic offset for the bevel button's content.

Returns:
Graphic offset

Definition at line 446 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.5 IconTransformType PPx::BevelButton::GetIconTransform () const

Returns the icon transform for the bevel button's icon.

Returns:
Icon transform

Definition at line 376 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.6 MenuRef PPx::BevelButton::GetMenuRef () const

Returns the MenuRef for the bevel button's popup menu.

Returns:
MenuRef

Definition at line 519 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

6.43.2.7 SInt16 PPx::BevelButton::GetMenuValue () const

Returns the value for the bevel button's popup menu.

Returns:
Menu value

Definition at line 482 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.8 ControlButtonTextAlignment PPx::BevelButton::GetTextAlignment () const

Returns the text alignment for the bevel button's title.

Returns:
Text alignment for the bevel button's title

Definition at line 270 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.9 SInt16 PPx::BevelButton::GetTextOffset () const

Returns the text offset for the bevel button's title.

Returns:
Text offset for the bevel button's title

Definition at line 305 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.10 ControlButtonTextPlacement PPx::BevelButton::GetTextPlacement () const

Returns the text placement for the bevel button's title.

Returns:
Text placement for the bevel button's title

Definition at line 340 of file PPxBevelButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.43.2.11 void PPx::BevelButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, ControlBevelThickness *inThickness*, ControlBevelButtonBehavior *inButtonBehavior*, const ControlButtonContentInfo & *inButtonContent*, SInt16 *inMenuID*, ControlBevelButtonMenuBehavior *inMenuBehavior*, ControlBevelButtonMenuPlacement *inMenuPlacement*)

Initialize from bevel button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Text title for button

inThickness Thickness of the beveled edges

inButtonBehavior How button behaves when clicked

inButtonContent Kind of button content

inMenuID Menu ID for popup menu

inMenuBehavior How menu behaves when item is selected

inMenuPlacement Placement of menu glyph within button

Definition at line 77 of file PPxBevelButton.cp.

6.43.2.12 void PPx::BevelButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 127 of file PPxBevelButton.cp.

References PPx::DataReader::ReadOptional(), SetCenterPopupGlyph(), SetGraphicAlignment(), SetGraphicOffset(), SetIconTransform(), SetMenuValue(), SetTextAlignment(), SetTextOffset(), and SetTextPlacement().

6.43.2.13 void PPx::BevelButton::SetCenterPopupGlyph (bool *inCenter*)

Sets the center popup glyph option for the bevel button.

Parameters:

inCenter Whether to center the popup glyph

Definition at line 538 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.14 void PPx::BevelButton::SetContentInfo (const ControlButtonContentInfo & *inContent*)

Sets the content inforomation for the bevel button.

Parameters:

inContent Content inforomation

Definition at line 576 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

6.43.2.15 void PPx::BevelButton::SetGraphicAlignment (ControlButtonGraphicAlignment *inAlignment*)

Sets the graphic alignment for the bevel button's content.

Parameters:

inAlignment Graphic alignemnt

Definition at line 395 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.16 void PPx::BevelButton::SetGraphicOffset (const Point & *inOffset*)

Sets the graphic offset for the bevel button's content.

Parameters:

inOffset Graphic offset

Definition at line 430 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.17 void PPx::BevelButton::SetIconTransform (IconTransformType *inTransform*)

Sets the icon transform for the bevel button's icon.

Parameters:

inTransform Icon transform

Definition at line 360 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.18 void PPx::BevelButton::SetMenuRef (MenuRef *inMenu*)

Sets the MenuRef for the bevel button's popup menu.

Parameters:

inMenu MenuRef

Definition at line 501 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

6.43.2.19 void PPx::BevelButton::SetMenuValue (SInt16 *inValue*)

Sets the value for the bevel button’s popup menu.

Parameters:
inValue Menu value

Definition at line 466 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.20 void PPx::BevelButton::SetTextAlignment (ControlButtonTextAlignment *inAlignment*)

Sets the text alignment for the bevel button’s title.

Parameters:
inAlignment Text alignment for title

Definition at line 254 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.21 void PPx::BevelButton::SetTextOffset (SInt16 *inOffset*)

Sets the text offset for the bevel button’s title.

Parameters:
inOffset Text offset for title

Definition at line 289 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.22 void PPx::BevelButton::SetTextPlacement (ControlButtonTextPlacement *inPlacement*)

Sets the text placement for the bevel button’s title.

Parameters:
inPlacement Text placement for title

Definition at line 324 of file PPxBevelButton.cp.

References PPx::View::SetDataTag().

Referenced by InitState().

6.43.2.23 void PPx::BevelButton::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 205 of file PPxBevelButton.cp.

References GetCenterPopupGlyph(), GetContentInfo(), GetGraphicAlignment(), GetGraphicOffset(), GetIconTransform(), GetMenuValue(), GetTextAlignment(), GetTextOffset(), GetTextPlacement(), PPx::View::GetTitle(), and PPx::DataWriter::WriteValue().

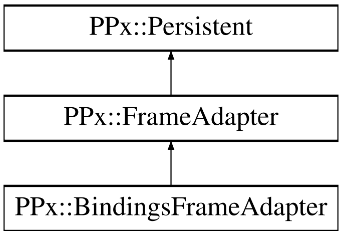
The documentation for this class was generated from the following files:

- PPxBevelButton.h
- PPxBevelButton.cp

6.44 PPx::BindingsFrameAdapter Class Reference

```
#include <PPxFrameAdapter.h>
```

Inheritance diagram for PPx::BindingsFrameAdapter::



6.44.1 Detailed Description

Adjusts a view frame based on whether its sides are bound to the corresponding sides of its container frame.

Definition at line 59 of file PPxFrameAdapter.h.

Public Member Functions

- BindingsFrameAdapter ()

  Default constructor.

- virtual ∼BindingsFrameAdapter ()

  Destructor.

- void SetBindings (bool inBindLeft, bool inBindTop, bool inBindRight, bool inBindBottom)

  Sets the bindings for each side of the frame.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.44.2 Member Function Documentation

6.44.2.1 void PPx::BindingsFrameAdapter::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Persistent.

Definition at line 134 of file PPxFrameAdapter.cp.

References PPx::DataReader::ReadOptional().

6.44.2.2 void PPx::BindingsFrameAdapter::SetBindings (bool *inBindLeft*, bool *inBindTop*, bool *inBindRight*, bool *inBindBottom*)

Sets the bindings for each side of the frame.

Parameters:

inBindLeft Binding for left of frame

inBindTop Binding for top of frame

inBindRight Binding for right of frame

inBindBottom Binding for bottom of frame

Definition at line 57 of file PPxFrameAdapter.cp.

6.44.2.3 void PPx::BindingsFrameAdapter::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Definition at line 152 of file PPxFrameAdapter.cp.

References PPx::DataWriter::WriteValue().

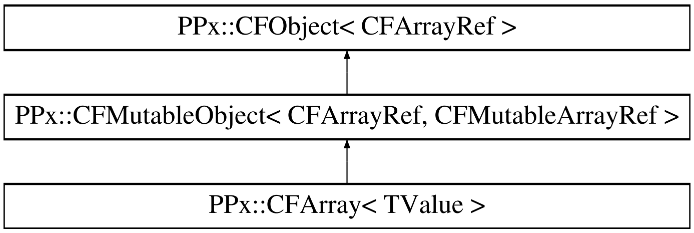
The documentation for this class was generated from the following files:

- PPxFrameAdapter.h
- PPxFrameAdapter.cp

6.45 PPx::CFArray< TValue > Class Template Reference

```
#include <SysCFArray.h>
```

Inheritance diagram for PPx::CFArray< TValue >::


PPx::CFObject< CFArrayRef >
PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >
PPx::CFArray< TValue >

6.45.1 Detailed Description

template<typename TValue> class PPx::CFArray< TValue >

Template class wrapper for a Core Foundation Array.

The template parameter specifies the type of values stored in the CFArray. As with all CF containers, the sizeof(TValue) must be 4 bytes. Consider using std::vector or std::deque if you need to store values that are not 4 bytes.

Definition at line 30 of file SysCFArray.h.

Public Member Functions

- CFArray ()

  Default constructor.

- CFArray (CFArrayRef inCFArray, bool inRetain)

  Constructs from an existing immutable CF array.

- CFArray (CFMutableArrayRef inCFArray, bool inRetain)

  Constructs from an existing mutable CF array.

- CFArray (CFIndex inCapacity, const CFArrayCallBacks ∗inCallBacks=nil, CFAllocatorRef inAllocator=nil)

  Constructs an array with the specified capacity.

- CFArray (const TValue ∗inValues, CFIndex inValueCount, const CFArrayCallBacks ∗inCallBacks=nil, CFAllocatorRef inAllocator=nil)

  Constructs from a buffer of values.

- CFArray (const CFArray &inOriginal)

  Copy constructor.

- CFArray & operator= (const CFArray &inSource)

  Assignment operator.

- CFIndex GetCount () const

  Returns the number of items in the array.

- bool IsEmpty () const

  Returns whether the array has no items.

- CFIndex GetCountOfValue (TValue inValue, CFRange inRange=cfRange_All) const

  Returns number of items within range that have the specified value.

- bool ContainsValue (TValue inValue, CFRange inRange=cfRange_All) const

  Returns whether the specified value is within a range of the array.

- TValue GetValueAt (CFIndex inIndex) const

  Returns value at a specified index in the array.

- TValue operator[ ] (CFIndex inIndex) const

  Returns value at a specified index in the array.

- void GetValues (TValue ∗outValues, CFRange inRange=cfRange_All) const

  Copies values from a range of the array into a buffer.

- CFIndex GetFirstIndexOf (TValue inValue, CFRange inRange=cfRange_All) const

  Returns the first index at which the specified value occurs within a range of the array.

- CFIndex GetLastIndexOf (TValue inValue, CFRange inRange=cfRange_All) const

  Returns the last index at which the specified value occurs within a range of the array.

- void AppendValue (TValue inValue)

  Appends a value to the end of the array.

- void InsertValueAt (CFIndex inIndex, TValue inValue)

  Inserts a value into the array at the specified index.

- void SetValueAt (CFIndex inIndex, TValue inValue)

  Assigns a value to the item at the specified index in the array.

- void RemoveValueAt (CFIndex inIndex)

  Removes the value at the specified index from the array.

- void RemoveAllValues ()

  Makes an array empty by removing all its values.

- void ReplaceValues (CFRange inRange, const TValue ∗inValues, CFIndex inValueCount)

  Replaces a range of values within the array with values from a buffer.

- void ExchangeValuesAt (CFIndex inIndexOne, CFIndex inIndexTwo)

  Swaps the values of the items at the specified indexes in the array.

- CFIndex BinarySearchFor (TValue inValue, CFComparatorFunction inComparator, void ∗inParam, CFRange inRange=cfRange_All)

  Searches the array for a value using a binary search algorithm.

- void Sort (CFComparatorFunction inComparator, void ∗inParam, CFRange inRange=cfRange_All)

  Sorts the values within a range of the array.

- void ApplyFunction (CFArrayApplierFunction inFunction, void ∗inParam, CFRange inRange=cfRange_All) const

  Calls a function once for each item in a range of the array.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 template<class TValue> PPx::CFArray< TValue >::CFArray (CFArrayRef *inArrayRef*, bool *inRetain*)

Constructs from an existing immutable CF array.

Parameters:

inArrayRef CF array to adopt

inRetain Whether to retain the CF array

Note:

Pass false for inRetain if you are transferring ownership of the CF array, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF array.

Pase true for inRetain if the caller wants to maintain shared ownership of the CF array.

Definition at line 161 of file SysCFArray.h.

6.45.2.2 template<class TValue> PPx::CFArray< TValue >::CFArray (CFMutableArrayRef *inArrayRef*, bool *inRetain*)

Constructs from an existing mutable CF array.

Parameters:

inArrayRef CF array to adopt

inRetain Whether to retain the CF array

Note:

Pass false for inRetain if you are transferring ownership of the CF array, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF array.

Pase true for inRetain if the caller wants to maintain ownership of the CF array. In this case, we make a copy, so that multiple clients aren't changing the same data.

Definition at line 187 of file SysCFArray.h.

References PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::AttachMutableRef().

6.45.2.3 template<class TValue> PPx::CFArray< TValue >::CFArray (CFIndex *inCapacity*, const CFArrayCallBacks ∗ *inCallBacks* = nil, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs an array with the specified capacity.

Parameters:

inCapacity Maximum number of items in the array. Use cfSize_Unlimited for an unbounded size.

inCallBacks Callback functions for managing array values. Use &kCFTypeArrayCallBacks if TValue is a CF reference type.

inAllocator Allocator for memory used by the array

Definition at line 208 of file SysCFArray.h.

References PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::AttachMutableRef(), and PPx_ThrowIfCFCreateFailed_.

6.45.2.4 template<class TValue> PPx::CFArray< TValue >::CFArray (const TValue ∗ *inValues*, CFIndex *inValuesCount*, const CFArrayCallBacks ∗ *inCallBacks* = nil, CFAllocatorRef *inAllocator* = nil)

Constructs from a buffer of values.

Parameters:

inValues Pointer to buffer of values to store in array

inValuesCount Number of values

inCallBacks Callback functions for managing array values. Use &kCFTypeArrayCallBacks if TValue is a CF reference type.

inAllocator Allocator for memory used by the array

Definition at line 234 of file SysCFArray.h.

References PPx::CFObject< CFArrayRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.45.3 Member Function Documentation

6.45.3.1 template<class TValue> void PPx::CFArray< TValue >::AppendValue (TValue *inValue*)

Appends a value to the end of the array.

Parameters:

inValue Value to append

Definition at line 467 of file SysCFArray.h.

References PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

6.45.3.2 template<class TValue> void PPx::CFArray< TValue >::ApplyFunction (CFArrayApplierFunction *inFunction*, void ∗ *inParam*, CFRange *inRange* = cfRange_All) const

Calls a function once for each item in a range of the array.

Parameters:

inFunction Function to call

inParam User-defined parameter for function

inRange Range of items over which to apply function

Definition at line 659 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.3 template<class TValue> CFIndex PPx::CFArray< TValue >::BinarySearchFor (TValue *inValue*, CFComparatorFunction *inComparator*, void ∗ *inParam*, CFRange *inRange* = cfRange_All)

Searches the array for a value using a binary search algorithm.

Parameters:

inValue Value to look for

inComparator Comparison function for values

inParam User-defined parameter for comparison function

inRange Range to search

Returns:

Index in range where item would be inserted in sorted order

If value exists in the range, return index is that of a matching value. If value is larger than all items in range, return index is greater than or equal to the end of the range. Otherwise, return index is that of the value in the array that is just larger the target value.

Behavior is undefined if the array is not sorted according to the comparison function.

Definition at line 615 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.4 template<class TValue> bool PPx::CFArray< TValue >::ContainsValue (TValue *inValue*, CFRange *inRange* = cfRange_All) const

Returns whether the specified value is within a range of the array.

Parameters:

inValue Value to look for

inRange Range which to search

Returns:

Whether the specified value is within a range of the array

Definition at line 341 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.5 template<class TValue> void PPx::CFArray< TValue >::ExchangeValuesAt (CFIndex *inIndexOne*, CFIndex *inIndexTwo*)

Swaps the values of the items at the specified indexes in the array.

Parameters:

inIndexOne Index of first item

inIndexTwo Index of second item

Definition at line 581 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

6.45.3.6 template<class TValue> CFIndex PPx::CFArray< TValue >::GetCount () const

Returns the number of items in the array.

Returns:

Number of items in the array

Definition at line 286 of file SysCFArray.h.

References PPx::CFObject< CFArrayRef >::UseRef().

Referenced by PPx::CFArray< TValue >::ApplyFunction(), PPx::CFArray< TValue >::BinarySearchFor(), PPx::CFArray< TValue >::ContainsValue(), PPx::CFArray< TValue >::ExchangeValuesAt(), PPx::CFArray< TValue >::GetCountOfValue(), PPx::CFArray< TValue >::GetFirstIndexOf(), PPx::CFArray< TValue >::GetLastIndexOf(), PPx::CFArray< TValue >::GetValueAt(), PPx::CFArray< TValue >::GetValues(), PPx::CFArray< TValue >::InsertValueAt(), PPx::CFArray< TValue >::IsEmpty(), PPx::CFArray< TValue >::RemoveValueAt(), PPx::CFArray< TValue >::ReplaceValues(), PPx::CFArray< TValue >::SetValueAt(), and PPx::CFArray< TValue >::Sort().

6.45.3.7 template<class TValue> CFIndex PPx::CFArray< TValue >::GetCountOfValue (TValue *inValue*, CFRange *inRange* = cfRange_All) const

Returns number of items within range that have the specified value.

Parameters:

inValue Value to look for

inRange Range which to search

Returns:

Number of itemw within range that have the specified value

Definition at line 319 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.8 template<class TValue> CFIndex PPx::CFArray< TValue >::GetFirstIndexOf (TValue *inValue*, CFRange *inRange* = cfRange_All) const

Returns the first index at which the specified value occurs within a range of the array.

Parameters:

inValue Value to look for

inRange Range to search

Returns:

First index at which the values occurs with the range

Definition at line 425 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.9 template<class TValue> CFIndex PPx::CFArray< TValue >::GetLastIndexOf (TValue *inValue*, CFRange *inRange* = cfRange_All) const

Returns the last index at which the specified value occurs within a range of the array.

Parameters:

inValue Value to look for

inRange Range to search

Returns:

Last index at which the values occurs with the range

Definition at line 448 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.10 template<class TValue> TValue PPx::CFArray< TValue >::GetValueAt (CFIndex *inIndex*) const

Returns value at a specified index in the array.

Parameters:

inIndex Index into array

Returns:

Value at a specified index in the array

Definition at line 362 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFObject< CFArrayRef >::UseRef().

Referenced by PPx::CFArray< TValue >::operator[ ]().

6.45.3.11 template<class TValue> void PPx::CFArray< TValue >::GetValues (TValue ∗ *outValues*, CFRange *inRange* = cfRange_All) const

Copies values from a range of the array into a buffer.

Parameters:

outValues Pointer to buffer

inRange Range to copy

Buffer must be large enough to hold all values, at least sizeof(TValue) ∗ length of range

Definition at line 401 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), PPx_BadParamIfNil_, and PPx::CFObject< CFArrayRef >::UseRef().

6.45.3.12 template<class TValue> void PPx::CFArray< TValue >::InsertValueAt (CFIndex *inIndex*, TValue *inValue*)

Inserts a value into the array at the specified index.

Parameters:

inIndex Index at which to insert value

inValue Value to insert

Definition at line 485 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

6.45.3.13 template<class TValue> bool PPx::CFArray< TValue >::IsEmpty () const

Returns whether the array has no items.

Returns:

Whether the array has no items

Definition at line 301 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount().

6.45.3.14]

template<class TValue> TValue PPx::CFArray< TValue >::operator[ ] (CFIndex *inIndex*) const

Returns value at a specified index in the array.

Parameters:

inIndex Index into array

Returns:

Value at a specified index in the array

Definition at line 381 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetValueAt().

6.45.3.15 template<class TValue> void PPx::CFArray< TValue >::RemoveValueAt (CFIndex *inIndex*)

Removes the value at the specified index from the array.

Parameters:

inIndex Index of item to remove

Definition at line 524 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

6.45.3.16 template<class TValue> void PPx::CFArray< TValue >::ReplaceValues (CFRange *inRange*, const TValue ∗ *inValues*, CFIndex *inValueCount*)

Replaces a range of values within the array with values from a buffer.

Parameters:

inRange Range to replace

inValues Butter of values

inValueCount Number of values, may be zero

The array grows or shrinks if the number of values is different from the length of the range

Definition at line 559 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

6.45.3.17 template<class TValue> void PPx::CFArray< TValue >::SetValueAt (CFIndex *inIndex*, TValue *inValue*)

Assigns a value to the item at the specified index in the array.

Parameters:

inIndex Index of item to set

inValue New value for item at the index

Definition at line 505 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

6.45.3.18 template<class TValue> void PPx::CFArray< TValue >::Sort (CFComparatorFunction *inComparator*, void ∗ *inParam*, CFRange *inRange* = cfRange_All)

Sorts the values within a range of the array.

Parameters:

inComparator Comparison function for values

inParam User-defineed parameter for comparison function

inRange Range to sort

Definition at line 638 of file SysCFArray.h.

References PPx::CFArray< TValue >::GetCount(), and PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::UseMutableRef().

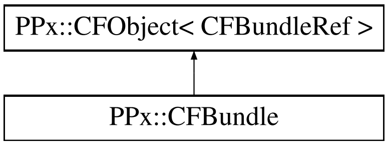
The documentation for this class was generated from the following file:

- SysCFArray.h

6.46 PPx::CFBundle Class Reference

```
#include <SysCFBundle.h>
```

Inheritance diagram for PPx::CFBundle::



6.46.1 Detailed Description

Wrapper class for Core Foundation Bundle.

Definition at line 27 of file SysCFBundle.h.

Public Member Functions

- CFBundle ()

  Default constructor.

- CFBundle (CFBundleRef inBundleRef, bool inRetain)

  Constructs from a CFBundleRef.

- CFBundle (CFStringRef inBundleID)

  Constructs from a Bundle identifier.

- CFBundle (CFURLRef inBundleURL, CFAllocatorRef inAllocator=nil)

  Constructs from a URL.

- CFBundle (const CFBundle &inOriginal)

  Copy constructor.

- CFBundle & operator= (const CFBundle &inSource)

  Assignment operator.

- CFURL GetBundleURL () const

  Returns a CFURL object for the Bundle's location.

- CFTypeRef GetValueForInfoDictionaryKey (CFStringRef inKey) const

Returns a CFTypeRef for a value in the Bundle's information dictionary.

- CFDictionary< CFStringRef, CFStringRef > GetGlobalInfoDictionary () const

  Returns a CFDictionary containing the Bundle's global information dictionary.

- CFDictionary< CFStringRef, CFStringRef > GetLocalInfoDictionary () const

  Returns a CFDictionary object containing the Bundle's local information dictionary.

- void GetPackageInfo (UInt32 &outPackageType, UInt32 &outPackageCreator) const

  Passes back the type and creator codes for the Bundle's package.

- CFString GetIdentifier () const

  Returns a CFString containing the Bundle's identifier.

- UInt32 GetVersionNumber () const

  Returns the version number of the Bundle.

- CFString GetDevelopmentRegion () const

  Returns a CFString containing the Bundle's development region.

- CFURL GetSupportFilesDirectoryURL () const

  Returns CRURL for the location of the Bundle's support files directory.

- CFURL GetResourcesDirectoryURL () const

  Returns CFURL for the location of the Bundle's resources directory.

- CFURL GetPrivateFrameworksURL () const

  Returns CFURL for the location of the Bundle's private frameworks.

- CFURL GetSharedFrameworksURL () const

  Returns CFURL for the location of the Bundle's shared frameworks.

- CFURL GetSharedSupportURL () const

  Returns CFURL for the location of the Bundle's shared support files directory.

- CFURL GetBuiltInPlugInsURL () const

  Returns CFURL for the location of the Bundle's built-in plug-ins.

- CFURL GetResourceURL (CFStringRef inResourceName, CFStringRef inResourceType=nil, CFStringRef inSubDirName=nil) const

  Returns CFURL for the location of a resource file within the Bundle.

- CFArray< CFURLRef > GetResourceURLsOfType (CFStringRef inResourceType, CFStringRef inSubDirName=nil) const

  Returns a CFArray of CFURLs for the locations of all resources of a specified type.

- CFString GetLocalizedString (CFStringRef inKey, CFStringRef inDefaultValue, CFStringRef inTableName) const

  Returns localized string by performing a table look-up.

- CFArray< CFStringRef > GetBundleLocalizations () const

  Returns a CFArray of CFStringRefs of all the Bundle's localizations.

- CFArray< CFStringRef > GetPreferredLocalizations () const

  Returns a CFArray of CFStringRefs of all the Bundle's preferred localizations.

- SInt16 OpenResourceMap () const

  Opens the Bundle's resource map and returns its reference number.

- void CloseResourceMap (SInt16 inRefNum) const

  Closes Bundle's resource map.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 PPx::CFBundle::CFBundle (CFBundleRef *inBundleRef*, bool *inRetain*)

Constructs from a CFBundleRef.

Parameters:

inBundleRef CFBundleRef to use

inRetain Whether to retain the CFBundleRef

Definition at line 27 of file SysCFBundle.cp.

6.46.2.2 PPx::CFBundle::CFBundle (CFStringRef *inBundleID*) `[explicit]`

Constructs from a Bundle identifier.

Parameters:

inBundleID Bundle identifier

Definition at line 43 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.46.2.3 PPx::CFBundle::CFBundle (CFURLRef *inBundleURL*, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a URL.

Parameters:

inBundleURL URL for a Bundle

inAllocator CF Allocator

Definition at line 61 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.46.3 Member Function Documentation

6.46.3.1 void PPx::CFBundle::CloseResourceMap (SInt16 *inRefNum*) const

Closes Bundle's resource map.

Parameters:

inRefNum Reference number of Bundle's resource map, previously obtained from OpenResourceMap()

Definition at line 458 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.2 CFURL PPx::CFBundle::GetBuiltInPlugInsURL () const

Returns CFURL for the location of the Bundle's built-in plug-ins.

Returns:

CFURL for the location of the Bundle's built-in plug-ins

Definition at line 310 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.3 CFArray< CFStringRef > PPx::CFBundle::GetBundleLocalizations () const

Returns a CFArray of CFStringRefs of all the Bundle's localizations.

Returns:

CFArray of CFStringRefs of all the Bundle's localizations

Definition at line 403 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

Referenced by GetPreferredLocalizations().

6.46.3.4 CFURL PPx::CFBundle::GetBundleURL () const

Returns a CFURL object for the Bundle's location.

Returns:

CFURL object for the Bundle's location

Definition at line 108 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.5 CFString PPx::CFBundle::GetDevelopmentRegion () const

Returns a CFString containing the Bundle's development region.

Returns:

CFString containing the Bundle's development region

Definition at line 218 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.6 CFDictionary< CFStringRef, CFStringRef > PPx::CFBundle::GetGlobalInfoDictionary () const

Returns a CFDictionary containing the Bundle's global information dictionary.

Returns:

CFDictionary containing the Bundle's global information dictionary

Definition at line 141 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.7 CFString PPx::CFBundle::GetIdentifier () const

Returns a CFString containing the Bundle's identifier.

Returns:

CFString containing the Bundle's identifier

Definition at line 190 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.8 CFDictionary< CFStringRef, CFStringRef > PPx::CFBundle::GetLocalInfoDictionary () const

Returns a CFDictionary object containing the Bundle's local information dictionary.

Returns:

CFDictionary object containing the Bundle's local information dictionary

Definition at line 158 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.9 CFString PPx::CFBundle::GetLocalizedString (CFStringRef *inKey*, CFStringRef *inDefaultValue*, CFStringRef *inTableName*) const

Returns localized string by performing a table look-up.

The table is a .strings file which contains (key, value) pairs of strings. If the key string is in the table, function returns the corresponding value string. If the key is not in the table, function returns the default value string.

Parameters:

inKey Key string for performing table look-up

inDefaultValue String returned if key is not found

inTableName Name of .strings file containing look-up table. If nil, uses the default Localized.strings file.

Returns:

CFString with localized text

Definition at line 384 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

Referenced by PPx::PrimaryBundle::GetLocalizedString().

6.46.3.10 void PPx::CFBundle::GetPackageInfo (UInt32 & *outPackageType*, UInt32 & *outPackageCreator*) const

Passes back the type and creator codes for the Bundle's package.

Parameters:

outPackageType Type code of the Bundle's package

outPackageCreator Creator code of the Bundle's package

Definition at line 174 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

Referenced by PPx::Signature::SetSignatureFromBundle().

6.46.3.11 CFArray< CFStringRef > PPx::CFBundle::GetPreferredLocalizations () const

Returns a CFArray of CFStringRefs of all the Bundle's preferred localizations.

Returns:

CFArray of CFStringRefs of all the Bundle's preferred localizations

Definition at line 420 of file SysCFBundle.cp.

References GetBundleLocalizations(), and PPx::CFObject< CFArrayRef >::IsValid().

6.46.3.12 CFURL PPx::CFBundle::GetPrivateFrameworksURL () const

Returns CFURL for the location of the Bundle's private frameworks.

Returns:

CFURL for the location of the Bundle's private frameworks

Definition at line 263 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.13 CFURL PPx::CFBundle::GetResourcesDirectoryURL () const

Returns CFURL for the location of the Bundle's resources directory.

Returns:

CFURL for the location of the Bundle's resources directory

Definition at line 248 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.14 CFURL PPx::CFBundle::GetResourceURL (CFStringRef *inResourceName*, CFStringRef *inResourceType* = nil, CFStringRef *inSubDirName* = nil) const

Returns CFURL for the location of a resource file within the Bundle.

Parameters:

- ***inResourceName*** Name of resource file
- ***inResourceType*** Type of resource file
- ***inSubDirName*** Subdirectory which to search. Pass nil to use the standard search locations

Returns:

CFURL for the location of the resource file

Definition at line 330 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

Referenced by PPx::BundleUtils::GetResourceData(), and PPx::BundleUtils::GetResourceProperty().

6.46.3.15 CFArray< CFURLRef > PPx::CFBundle::GetResourceURLsOfType (CFStringRef *inResourceType*, CFStringRef *inSubDirName* = nil) const

Returns a CFArray of CFURLs for the locations of all resources of a specified type.

Parameters:

- ***inResourceType*** Type of resource file
- ***inSubDirName*** Subdirectory which to search. Pass nil to use the standard search locations

Returns:

CFArray of CFURLRefs for the locations of all resources of a specified type

Definition at line 355 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.16 CFURL PPx::CFBundle::GetSharedFrameworksURL () const

Returns CFURL for the location of the Bundle’s shared frameworks.

Returns:
CFURL for the location of the Bundle’s shared frameworks

Definition at line 278 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.17 CFURL PPx::CFBundle::GetSharedSupportURL () const

Returns CFURL for the location of the Bundle’s shared support files directory.

Returns:
CFURL for the location of the Bundle’s shared support files directory

Definition at line 295 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.18 CFURL PPx::CFBundle::GetSupportFilesDirectoryURL () const

Returns CRURL for the location of the Bundle’s support files directory.

Returns:
CFURL for the location of the Bundle’s support files directory

Definition at line 233 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.19 CFTypeRef PPx::CFBundle::GetValueForInfoDictionaryKey (CFStringRef *inKey*) const

Returns a CFTypeRef for a value in the Bundle’s information dictionary.

Parameters:
inKey Key for the value

Returns:
CTTypeRef for the value with the specified key

Definition at line 124 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

Referenced by PPx::BundleUtils::GetInfoDictionaryKeyString().

6.46.3.20 UInt32 PPx::CFBundle::GetVersionNumber () const

Returns the version number of the Bundle.

Returns:

Version number of the Bundle

Definition at line 204 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

6.46.3.21 SInt16 PPx::CFBundle::OpenResourceMap () const

Opens the Bundle's resource map and returns its reference number.

Returns:

Reference number for Bundle's open resource map

Definition at line 443 of file SysCFBundle.cp.

References PPx::CFObject< CFBundleRef >::UseRef().

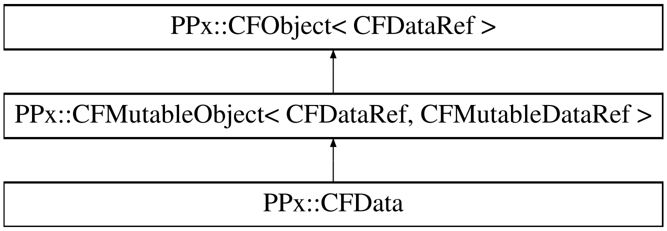
The documentation for this class was generated from the following files:

- SysCFBundle.h
- SysCFBundle.cp

6.47 PPx::CFData Class Reference

`#include <SysCFData.h>`

Inheritance diagram for PPx::CFData::


PPx::CFObject< CFDataRef >
PPx::CFMutableObject< CFDataRef, CFMutableDataRef >
PPx::CFData

6.47.1 Detailed Description

Wrapper class for a Core Foundataion Data object.

Definition at line 23 of file SysCFData.h.

Public Member Functions

- CFData ()

  Default constructor.

- CFData (CFDataRef inDataRef, bool inRetain)

  Construct from an immutable CFDataRef.

- CFData (CFMutableDataRef inDataRef, bool inRetain)

  Constructs from an existing CFMutableDataRef.

- CFData (CFIndex inCapacity, CFAllocatorRef inAllocator=nil)

  Constructs a CFData with the specified capacity.

- CFData (const UInt8 *inBytes, CFIndex inByteCount, CFAllocatorRef inAllocator=nil)

  Constructs a CFData with the specified data.

- CFData (const CFData &inOriginal)

  Copy constructor.

- CFData & operator= (const CFData &inSource)

Assignment operator.

- CFIndex GetLength () const

  Returns the Byte length of the data.

- const UInt8 ∗ GetBytePtr () const

  Returns a const pointer to the bytes in the CFData object.

- void GetDataBytes (CFRange inRange, UInt8 ∗outBuffer) const

  Copies bytes from the CFData to a supplied buffer.

- UInt8 ∗ GetMutableBytePtr ()

  Returns a non-const pointer to the bytes in the CFData.

- void SetLength (CFIndex inNewLength)

  Sets the length of the CFData's internal buffer.

- void IncreaseLength (CFIndex inExtraLength)

  Increases the size of the CFData's internal buffer.

- void AppendBytes (const UInt8 ∗inBytes, CFIndex inByteCount)

  Appends bytes from a buffer to a CFData object.

- void ReplaceBytes (CFRange inRange, const UInt8 ∗inNewBytes, CFIndex inByteCount)

  Replaces a range within the CFData with new bytes from a buffer.

- void DeleteBytes (CFRange inRange)

  Deletes a range of bytes from the CFData.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 PPx::CFData::CFData (CFMutableDataRef *inDataRef*, bool *inRetain*)

Constructs from an existing CFMutableDataRef.

Parameters:

inDataRef CF data to adopt

inRetain Whether to retain the CF data

Note:

Pass false for inRetain if you are transferring ownership of the CF data, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF data.

Pase true for inRetain if the caller wants to maintain ownership of the CF data. In this case, we make a copy, so that multiple clients aren't changing the same data.

Definition at line 49 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::AttachMutableRef().

6.47.2.2 PPx::CFData::CFData (CFIndex *inCapacity*, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs a CFData with the specified capacity.

Parameters:

inCapacity Size of the CFData

inAllocator CF Allocator

Specify 0 for inCapacity to create a CFData that can grow to arbitrary size. Otherwise, if you pass a postive value, it creates a fixed-size CFData.

Definition at line 69 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::AttachMutableRef(), and PPx_ThrowIfCFCreateFailed_.

6.47.2.3 PPx::CFData::CFData (const UInt8 ∗ *inBytes*, CFIndex *inByteCount*, CFAllocatorRef *inAllocator* = nil)

Constructs a CFData with the specified data.

Parameters:

inBytes Pointer to buffer of data to store

inByteCount Size of buffer

inAllocator CF Allocator

Definition at line 89 of file SysCFData.cp.

References PPx::CFObject< CFDataRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.47.3 Member Function Documentation

6.47.3.1 void PPx::CFData::AppendBytes (const UInt8 $*$ *inBytes*, CFIndex *inByteCount*)

Appends bytes from a buffer to a CFData object.

Parameters:

inBytes Pointer to a buffer

inByteCount Number of bytes from buffer to append

Definition at line 235 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::UseMutableRef().

6.47.3.2 void PPx::CFData::DeleteBytes (CFRange *inRange*)

Deletes a range of bytes from the CFData.

Parameters:

inRange Range of bytes to delete

Definition at line 270 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::UseMutableRef().

6.47.3.3 const UInt8 $*$ PPx::CFData::GetBytePtr () const

Returns a const pointer to the bytes in the CFData object.

Returns:

Const pointer to the bytes in the CFData object

Definition at line 150 of file SysCFData.cp.

References PPx::CFObject< CFDataRef >::UseRef().

Referenced by PPx::EditTextControl::GetText(), PPx::EditTextControl::SetText(), and PPx::DataFork::WriteContents().

6.47.3.4 void PPx::CFData::GetDataBytes (CFRange *inRange*, UInt8 $*$ *outBuffer*) const

Copies bytes from the CFData to a supplied buffer.

Parameters:

inRange Range of bytes to copy

outBuffer Pointer to a buffer which must be large enough to store the requested bytes

Definition at line 166 of file SysCFData.cp.

References PPx::CFObject< CFDataRef >::UseRef().

6.47.3.5 CFIndex PPx::CFData::GetLength () const

Returns the Byte length of the data.

Returns:

Byte lenght of the data

Definition at line 136 of file SysCFData.cp.

References PPx::CFObject< CFDataRef >::UseRef().

Referenced by PPx::DataFork::WriteContents().

6.47.3.6 UInt8 ∗ PPx::CFData::GetMutableBytePtr ()

Returns a non-const pointer to the bytes in the CFData.

Returns:

Non-const pointer to the bytes in the CFData

Definition at line 182 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::UseMutableRef().

Referenced by PPx::EditTextControl::GetText(), PPx::DataFork::ReadContents(), and PPx::EditTextControl::SetText().

6.47.3.7 void PPx::CFData::IncreaseLength (CFIndex *inExtraLength*)

Increases the size of the CFData's internal buffer.

Parameters:

inExtraLength Amount to increase size of internal buffer

Zero fills extra bytes. Growing the length greater than the capacity of the fixed-size CGData results in undefined behavior

Definition at line 219 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::UseMutableRef().

6.47.3.8 void PPx::CFData::ReplaceBytes (CFRange *inRange*, const UInt8 ∗ *inNewBytes*, CFIndex *inByteCount*)

Replaces a range within the CFData with new bytes from a buffer.

Parameters:

inRange Range within CFData to replace

inNewBytes Pointer to a buffer

inByteCount Number of bytes from buffer to copy

Definition at line 253 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::UseMutableRef().

6.47.3.9 void PPx::CFData::SetLength (CFIndex *inNewLength*)

Sets the length of the CFData's internal buffer.

Parameters:

inNewLength New length for CFData's buffer

Discards excess bytes if new length is less than current length. Zero fills extra bytes if new length is greater than current length. Trying to set the length greater than the capacity of a fixed-size CFData results in undefined behavior.

Definition at line 201 of file SysCFData.cp.

References PPx::CFMutableObject< CFDataRef, CFMutableDataRef >::UseMutableRef().

Referenced by PPx::DataFork::ReadContents().

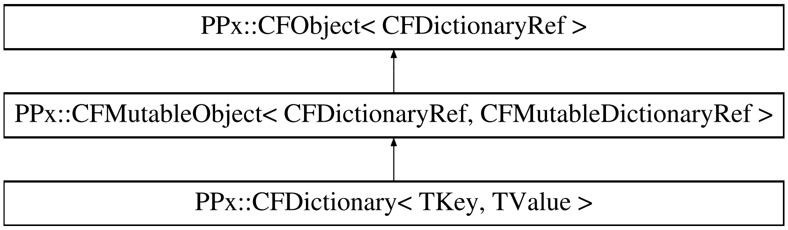
The documentation for this class was generated from the following files:

- SysCFData.h
- SysCFData.cp

6.48 PPx::CFDictionary< TKey, TValue > Class Template Reference

```
#include <SysCFDictionary.h>
```

Inheritance diagram for PPx::CFDictionary< TKey, TValue >::


PPx::CFObject< CFDictionaryRef >
PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >
PPx::CFDictionary< TKey, TValue >

6.48.1 Detailed Description

template<class TKey, class TValue> class PPx::CFDictionary< TKey, TValue >

Template wrapper class for Core Foundation Dictionary.

The template parameters specify the types of the key and the value. As with all CF containers, sizeof(TKey) and sizeof(TValue) must be 4 bytes.

Definition at line 25 of file SysCFDictionary.h.

Public Member Functions

- CFDictionary ()

  Default constructor.

- CFDictionary (CFDictionaryRef inCFDictionary, bool inRetain)

  Constructs from an existing immutable CFDictionaryRef.

- CFDictionary (CFMutableDictionaryRef inCFDictionary, bool inRetain)

  Constructs from an existing CFMutableDictionaryRef.

- CFDictionary (CFIndex inCapacity, const CFDictionaryKeyCallBacks ∗inKeyCallBacks=nil, const CFDictionaryValueCallBacks ∗inValueCallBacks=nil, CFAllocatorRef inAllocator=nil)

  Constructs a dictionary with the specified capacity.

- CFDictionary (const TKey ∗inKeys, const TValue ∗inValues, CFIndex inItemCount, const CFDictionaryKeyCallBacks ∗inKeyCallBacks=nil, const CFDictionaryValueCallBacks ∗inValueCallBacks=nil, CFAllocatorRef inAllocator=nil)

  Constructs from buffers of keys and values.

- CFDictionary (const CFDictionary &inOriginal)

  Copy constructor.

- CFDictionary & operator= (const CFDictionary &inSource)

  Assignment operator.

- CFIndex GetCount () const

  Returns the number of items in the dictionary.

- bool IsEmpty () const

  Returns whether the dictionary has no items.

- CFIndex GetCountOfKey (TKey inKey) const

  Returns number of items with the given key that are in the dictionary.

- bool ContainsKey (TKey inKey) const

  Returns whether an item with the given key is in the dictionary.

- CFIndex GetCountOfValue (TValue inValue) const

  Returns number of items with the given value that are in the dictionary.

- bool ContainsValue (TValue inValue) const

  Returns whether an item with the given value is in the dictionary.

- TValue GetValue (TKey inKey) const

  Returns value of item with the given key.

- TValue operator[ ] (TKey inKey) const

  Returns value of item with the given key.

- bool GetValueIfPresent (TKey inKey, TValue &outValue) const

  Passes back value of item with the given key and returns whether an item was found.

- void GetKeysAndValues (TKey ∗outKeys, TValue ∗outValues) const

  Passes back all keys and values to supplied buffers.

- void AddValue (TKey inKey, TValue inValue)

Adds a key/value pair to the dictionary.

- void SetValue (TKey inKey, TValue inValue)

  Sets the value for the item with the specified key.

- void ReplaceValue (TKey inKey, TValue inValue)

  Replaces the value for the item with the specified key.

- void RemoveValue (TKey inKey)

  Removes the item with the given key from the dictionary.

- void RemoveAllValues ()

  Removes all items from the dictionary, making it empty.

- void ApplyFunction (CFDictionaryApplierFunction inFunction, void ∗inParam) const

  Call a function for each item in the dictionary.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 template<class TKey, class TValue> PPx::CFDictionary< TKey, TValue >::CFDictionary (CFMutableDictionaryRef *inDictRef*, bool *inRetain*)

Constructs from an existing CFMutableDictionaryRef.

Parameters:

inDictRef CF dictionary to adopt

inRetain Whether to retain the CF dictionary

Note:

Pass false for inRetain if you are transferring ownership of the CF dictionary, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF dictionary.

Pase true for inRetain if the caller wants to maintain ownership of the CF dictionary. In this case, we make a copy, so that multiple clients aren't changing the same data.

Definition at line 153 of file SysCFDictionary.h.

References PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >::AttachMutableRef().

6.48.2.2 template<class TKey, class TValue> PPx::CFDictionary< TKey, TValue >::CFDictionary (CFIndex *inCapacity*, const CFDictionaryKeyCallBacks ∗ *inKeyCallBacks* = nil, const CFDictionaryValueCallBacks ∗ *inValueCallBacks* = nil, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs a dictionary with the specified capacity.

Parameters:

inCapacity Size of the dictionary

inKeyCallBacks Callbacks for the dictionary keys

inValueCallBacks Callbacks for the dictionary values

inAllocator CF Allocator

If the keys are CFTypeRefs, use kCFTypeDictionaryKeyCallBacks for inKeyCallBacks. If the values are CFTypeRefs, use kCFTypeDictionaryValueCallBacks for inValueCallBacks.

Definition at line 176 of file SysCFDictionary.h.

References PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >::AttachMutableRef(), and PPx_ThrowIfCFCreateFailed_.

6.48.2.3 template<class TKey, class TValue> PPx::CFDictionary< TKey, TValue >::CFDictionary (const TKey ∗ *inKeys*, const TValue ∗ *inValues*, CFIndex *inItemCount*, const CFDictionaryKeyCallBacks ∗ *inKeyCallBacks* = nil, const CFDictionaryValueCallBacks ∗ *inValueCallBacks* = nil, CFAllocatorRef *inAllocator* = nil)

Constructs from buffers of keys and values.

Parameters:

inKeys Array of keys

inValues Array of values

inItemCount Number of key/value pairs

inKeyCallBacks Callbacks for the keys

inValueCallBacks Callbacks for the values

inAllocator CF Allocator

If the keys are CFTypeRefs, use kCFTypeDictionaryKeyCallBacks for inKeyCallBacks. If the values are CFTypeRefs, use kCFTypeDictionaryValueCallBacks for inValueCallBacks.

Definition at line 210 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.48.3 Member Function Documentation

6.48.3.1 template<class TKey, class TValue> void PPx::CFDictionary< TKey, TValue >::AddValue (TKey *inKey*, TValue *inValue*)

Adds a key/value pair to the dictionary.

Parameters:

inKey Item key

inValue Item value

Definition at line 464 of file SysCFDictionary.h.

References PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >::UseMutableRef().

6.48.3.2 template<class TKey, class TValue> void PPx::CFDictionary< TKey, TValue >::ApplyFunction (CFDictionaryApplierFunction *inFunction*, void * *inParam*) const

Call a function for each item in the dictionary.

Parameters:

inFunction Function to call

inParam User-defined parameter to the function

Definition at line 565 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.3 template<class TKey, class TValue> bool PPx::CFDictionary< TKey, TValue >::ContainsKey (TKey *inKey*) const

Returns whether an item with the given key is in the dictionary.

Parameters:

inKey Item key

Returns:

Whether an item with the given key is in the dictionary

Definition at line 320 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.4 template<class TKey, class TValue> bool PPx::CFDictionary< TKey, TValue >::ContainsValue (TValue *inValue*) const

Returns whether an item with the given value is in the dictionary.

Parameters:

inValue Item value

Returns:

Whether an item with the given value is in the dictionary

Definition at line 358 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.5 template<class TKey, class TValue> CFIndex PPx::CFDictionary< TKey, TValue >::GetCount () const

Returns the number of items in the dictionary.

Returns:

Number of items in the dictionary

Definition at line 269 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

Referenced by PPx::CFDictionary< TKey, TValue >::IsEmpty().

6.48.3.6 template<class TKey, class TValue> CFIndex PPx::CFDictionary< TKey, TValue >::GetCountOfKey (TKey *inKey*) const

Returns number of items with the given key that are in the dictionary.

Parameters:

inKey Item key

Returns:

Number of items with the given key that are in the dictionary

Definition at line 301 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.7 template<class TKey, class TValue> CFIndex PPx::CFDictionary< TKey, TValue >::GetCountOfValue (TValue *inValue*) const

Returns number of items with the given value that are in the dictionary.

Parameters:

inValue Item value

Returns:

Number of items with the given value that are in the dictionary

Definition at line 339 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.8 template<class TKey, class TValue> void PPx::CFDictionary< TKey, TValue >::GetKeysAndValues (TKey ∗ *outKeys*, TValue ∗ *outValues*) const

Passes back all keys and values to supplied buffers.

Parameters:

outKeys Buffer to which to copy the keys

outValues Buffer to which to copy the values

Buffers must be large enough to hold all the items

Definition at line 444 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.9 template<class TKey, class TValue> TValue PPx::CFDictionary< TKey, TValue >::GetValue (TKey *inKey*) const

Returns value of item with the given key.

Parameters:

inKey Item key

Returns:

Value of item with the given key

Returns zero if the key is not in the dictionary. Call GetValueIfPresent() if zero may be a valid item value.

Definition at line 380 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

Referenced by PPx::CFDictionary< TKey, TValue >::operator[ ]().

6.48.3.10 template<class TKey, class TValue> bool PPx::CFDictionary< TKey, TValue >::GetValueIfPresent (TKey *inKey*, TValue & *outValue*) const

Passes back value of item with the given key and returns whether an item was found.

Parameters:

inKey Item key

outValue Item value

Returns:

Whether an item with the given key was found

Definition at line 422 of file SysCFDictionary.h.

References PPx::CFObject< CFDictionaryRef >::UseRef().

6.48.3.11 template<class TKey, class TValue> bool PPx::CFDictionary< TKey, TValue >::IsEmpty () const

Returns whether the dictionary has no items.

Returns:

Whether the dictionar has no items

Definition at line 284 of file SysCFDictionary.h.

References PPx::CFDictionary< TKey, TValue >::GetCount().

6.48.3.12]

template<class TKey, class TValue> TValue PPx::CFDictionary< TKey, TValue >::operator[ ] (TKey *inKey*) const

Returns value of item with the given key.

Parameters:

inKey Item key

Returns:

Value of item with the given key

Returns zero if the key is not in the dictionary. Call GetValueIfPresent() if zero may be a valid item value.

Definition at line 402 of file SysCFDictionary.h.

References PPx::CFDictionary< TKey, TValue >::GetValue().

6.48.3.13 template<class TKey, class TValue> void PPx::CFDictionary< TKey, TValue >::RemoveValue (TKey *inKey*)

Removes the item with the given key from the dictionary.

Parameters:

inKey Item key

Does nothng if there is no item with the given key in the dictionary

Definition at line 534 of file SysCFDictionary.h.

References PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >::UseMutableRef().

6.48.3.14 template<class TKey, class TValue> void PPx::CFDictionary< TKey, TValue >::ReplaceValue (TKey *inKey*, TValue *inValue*)

Replaces the value for the item with the specified key.

Parameters:

inKey Item key

inValue Item value

If an item with the key is present, changes the value of that item to the input value. If an item with the key is not present, does nothing.

Definition at line 513 of file SysCFDictionary.h.

References PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >::UseMutableRef().

6.48.3.15 template<class TKey, class TValue> void PPx::CFDictionary< TKey, TValue >::SetValue (TKey *inKey*, TValue *inValue*)

Sets the value for the item with the specified key.

Parameters:

inKey Item key

inValue Item value

If an item with the key is present, changes the value of that item to the input value. If an item with the key is not present, adds the key/value pair as a new item.

Definition at line 488 of file SysCFDictionary.h.

References PPx::CFMutableObject< CFDictionaryRef, CFMutableDictionaryRef >::UseMutableRef().

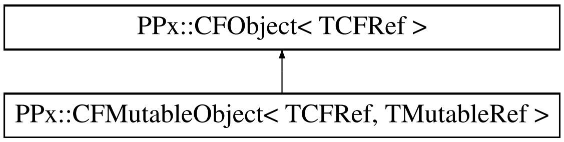
The documentation for this class was generated from the following file:

- SysCFDictionary.h

6.49 PPx::CFMutableObject< TCFRef, TMutableRef > Class Template Reference

```
#include <SysCFMutableObject.h>
```

Inheritance diagram for PPx::CFMutableObject< TCFRef, TMutableRef >::



6.49.1 Detailed Description

template<typename TCFRef, typename TMutableRef> class PPx::CFMutableObject< TCFRef, TMutableRef >

Template base class for Core Foundation wrapper classes for mutable objects.

The template parameters are the CF reference type and the CF mutable reference type. CFMutableObject implements copy-on-modify behavior. Using a mutable reference results in a copy in two cases:

(1) the object is currently immutable (2) the object is mutable and it is being shared (retain count greater than one)

Definition at line 33 of file SysCFMutableObject.h.

Public Member Functions

- TMutableRef UseMutableRef ()

  Returns mutable CF reference.

- operator TMutableRef ()

  Converts to a TMutableRef.

- void AttachMutableRef (TMutableRef inMutableRef, bool inRetain)

  Takes ownership of another CF reference.

- TMutableRef DetachMutableRef ()

  Relinquishes ownership of the object's CF reference.

Protected Member Functions

- CFMutableObject ()

  Default constructor.

- CFMutableObject (TCFRef inRef, bool inRetain)

  Constructs from an existing CF Reference.

- CFMutableObject (const CFMutableObject &inOriginal)

  Copy constructor.

- virtual ~CFMutableObject ()

  Destructor.

- void AssignObject (const CFMutableObject &inOriginal)

  Shares ownership of the CF reference owned by another object.

6.49.2 Constructor & Destructor Documentation

6.49.2.1 template<typename TCFRef, typename TMutableRef> PPx::CFMutableObject< TCFRef, TMutableRef >::CFMutableObject (TCFRef *inRef*, bool *inRetain*) `[protected]`

Constructs from an existing CF Reference.

Parameters:

inRef CF reference to adopt

inRetain Whether to retain the CF reference

Pass false for inRetain if you are transferring ownership of the CF reference, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF reference.

Pase true for inRetain if the caller wants to maintain shared ownership of the CF reference.

Definition at line 179 of file SysCFMutableObject.h.

6.49.3 Member Function Documentation

6.49.3.1 template<typename TCFRef, typename TMutableRef> void PPx::CFMutableObject< TCFRef, TMutableRef >::AssignObject (const CFMutableObject< TCFRef, TMutableRef > & *inOriginal*) `[protected]`

Shares ownership of the CF reference owned by another object.

Parameters:

inOriginal Share the CF refernce of this CFObject

Definition at line 224 of file SysCFMutableObject.h.

References PPx::CFMutableObject< TCFRef, TMutableRef >::mIsMutable.

6.49.3.2 template<typename TCFRef, typename TMutableRef> void PPx::CFMutableObject< TCFRef, TMutableRef >::AttachMutableRef (TMutableRef *inMutableRef*, bool *inRetain*)

Takes ownership of another CF reference.

Releases the currently owned CF reference, and optionally retains the input one.

Parameters:

inMutableRef CF reference to adopt

inRetain Whether to retain the CF reference

Pass false for inRetain if you are transferring ownership of the CF reference, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF reference.

Pase true for inRetain if the caller wants to maintain ownership of the CF reference. In this case, we make a copy, so that multiple clients aren't changing the same data.

Definition at line 112 of file SysCFMutableObject.h.

6.49.3.3 template<typename TCFRef, typename TMutableRef> TMutableRef PPx::CFMutableObject< TCFRef, TMutableRef >::DetachMutableRef ()

Relinquishes ownership of the object's CF reference.

Caller is responsible for releasing the CF reference.

Returns:

CF reference that object used to own

After detaching, the object does not own any CF reference. Any attempt to use the CF reference is an error.

Definition at line 141 of file SysCFMutableObject.h.

6.49.3.4 template<typename TCFRef, typename TMutableRef> TMutableRef PPx::CFMutableObject< TCFRef, TMutableRef >::UseMutableRef ()

Returns mutable CF reference.

Returns:
A mutable CF refenrece

Definition at line 85 of file SysCFMutableObject.h.

Referenced by PPx::CFMutableObject< CFArrayRef, CFMutableArrayRef >::operator TMutableRef().

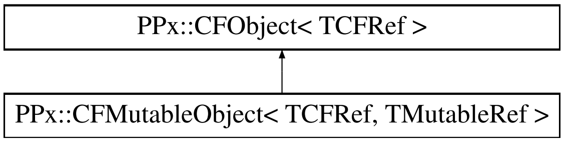
The documentation for this class was generated from the following file:

- SysCFMutableObject.h

6.50 PPx::CFObject< TCFRef > Class Template Reference

```
#include <SysCFObject.h>
```

Inheritance diagram for PPx::CFObject< TCFRef >::



6.50.1 Detailed Description

template<typename TCFRef> class PPx::CFObject< TCFRef >

Template base class for Core Foundation wrapper classes.

Template parameter is the underlying system CF reference type. This class controls access to the CF reference, automatically retaining and releasing it. It also implements the CF Base functions.

This class is the base class for immutable Core Foundation entities.

Definition at line 30 of file SysCFObject.h.

Public Member Functions

- TCFRef UseRef () const

  Returns the CF reference for the object.

- TCFRef GetRefValue () const

  Returns the CF reference for the object without checking if it is nil.

- operator TCFRef () const

  Converts to a type ref.

- bool IsValid () const

  Returns whether the CF reference for the object is valid.

- bool HasSameRef (TCFRef inRef) const

  Returns whether the CFObject has the same CF reference as the input one.

- CFTypeID GetTypeID () const

  Returns the CFTypeID for the object's CF reference.

- CFStringRef CopyTypeIDDescription () const

  Returns the type ID description of the object's CF reference.

- CFAllocatorRef GetAllocator () const

  Returns the CF Allocator used by the object's CF reference.

- CFStringRef CopyDescription () const

  Returns the description of the object's CF reference.

- CFHashCode GetHashCode () const

  Returns the hash code for the object's CF reference.

- CFIndex GetRetainCount () const

  Returns the retain count for the object's CF reference.

- bool IsEqualTo (TCFRef inRef) const

  Returns whether the object's CF reference is equalent to another CF reference.

- void AttachRef (TCFRef inRef, bool inRetain)

  Takes ownership of another CF reference.

- TCFRef DetachRef ()

  Relinquishes ownership of the object's CF reference.

- void FreeRef ()

  Releases ownership of the object's CF reference.

Protected Member Functions

- CFObject ()

  Default constructor.

- CFObject (TCFRef inRef, bool inRetain)

  Constructs from an existing CF Reference.

- CFObject (const CFObject &inOriginal)

  Copy constructor.

- virtual ~CFObject ()

  Destructor.

- void AssignObject (const CFObject &inOriginal)

  Share ownership of the CF reference owned by another object.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 template<typename TCFRef> PPx::CFObject< TCFRef >::CFObject (TCFRef *inRef*, bool *inRetain*) `[protected]`

Constructs from an existing CF Reference.

Parameters:

inRef CF reference to adopt

inRetain Whether to retain the CF reference

Pass false for inRetain if you are transferring ownership of the CF reference, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF reference.

Pase true for inRetain if the caller wants to maintain shared ownership of the CF reference.

Definition at line 385 of file SysCFObject.h.

6.50.3 Member Function Documentation

6.50.3.1 template<typename TCFRef> void PPx::CFObject< TCFRef >::AssignObject (const CFObject< TCFRef > & *inOriginal*) `[protected]`

Share ownership of the CF reference owned by another object.

Parameters:

inOriginal Share CF reference of this CFObject

Definition at line 432 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::AttachRef(), and PPx::CFObject< TCFRef >::mCFRef.

Referenced by PPx::CFObject< TCFRef >::CFObject().

6.50.3.2 template<typename TCFRef> void PPx::CFObject< TCFRef >::AttachRef (TCFRef *inRef*, bool *inRetain*)

Takes ownership of another CF reference.

Releases the currently owned CF reference, and optionally retains the input one.

Parameters:

inRef CF reference to adopt

inRetain Whether to retain the CF reference

Pass false for inRetain if you are transferring ownership of the CF reference, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF reference.

Pase true for inRetain if the caller wants to maintain shared ownership of the CF reference.

Definition at line 303 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::HasSameRef().

Referenced by PPx::CFObject< TCFRef >::AssignObject().

6.50.3.3 template<typename TCFRef> CFStringRef PPx::CFObject< TCFRef >::CopyDescription () const

Returns the description of the object's CF reference.

Returns:

Description of the object's CF reference

Caller is responsible for releasing the returned CFStringRef

Definition at line 228 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::UseRef().

6.50.3.4 template<typename TCFRef> CFStringRef PPx::CFObject< TCFRef >::CopyTypeIDDescription () const

Returns the type ID description of the object's CF reference.

Returns:

Type ID description of the object's CF reference

Caller is responsible for releasing the returned CFStringRef

Definition at line 196 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::UseRef().

6.50.3.5 template<typename TCFRef> TCFRef PPx::CFObject< TCFRef >::DetachRef ()

Relinquishes ownership of the object's CF reference.

Caller is responsible for releasing the CF reference.

Returns:

CF reference that object used to own

After detaching, the object does not own any CF reference. Any attempt to use the CF reference is an error.

Definition at line 331 of file SysCFObject.h.

6.50.3.6 template<typename TCFRef> void PPx::CFObject< TCFRef >::FreeRef ()

Releases ownership of the object's CF reference.

After freeing, the object does not own any CF reference. Any attempt to use the CF reference is an error.

Definition at line 350 of file SysCFObject.h.

6.50.3.7 template<typename TCFRef> CFAllocatorRef PPx::CFObject< TCFRef >::GetAllocator () const

Returns the CF Allocator used by the object's CF reference.

Returns:

CF Allocator used by the object's CF reference

Definition at line 211 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::UseRef().

6.50.3.8 template<typename TCFRef> CFHashCode PPx::CFObject< TCFRef >::GetHashCode () const

Returns the hash code for the object's CF reference.

Returns:

Hash code for the object's CF reference

Definition at line 243 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::UseRef().

6.50.3.9 template<typename TCFRef> TCFRef PPx::CFObject< TCFRef >::GetRefValue () const

Returns the CF reference for the object without checking if it is nil.

Returns:

CF reference for the object

Use this routine if you are prepared for the Ref to be nil.

Note:

This function is called GetRefValue instead of just GetRef so that you are less likely to use it when UseRef is what you really should call to guarantee a valid ref.

Definition at line 130 of file SysCFObject.h.

6.50.3.10 template<typename TCFRef> CFIndex PPx::CFObject< TCFRef >::GetRetainCount () const

Returns the retain count for the object's CF reference.

@ return Retain count for the object's CF reference

Definition at line 258 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::UseRef().

6.50.3.11 template<typename TCFRef> CFTypeID PPx::CFObject< TCFRef >::GetTypeID () const

Returns the CFTypeID for the object's CF reference.

Returns:

CFTypeID for the object's CF reference

Definition at line 179 of file SysCFObject.h.

References PPx::CFObject< TCFRef >::UseRef().

6.50.3.12 template<typename TCFRef> bool PPx::CFObject< TCFRef >::HasSameRef (TCFRef *inRef*) const

Returns whether the CFObject has the same CF reference as the input one.

Parameters:

inRef CF reference to check for equality

Returns:

Whether the CFObject has the same CF reference as the input one

Definition at line 163 of file SysCFObject.h.

Referenced by PPx::CFObject< TCFRef >::AttachRef().

6.50.3.13 template<typename TCFRef> bool PPx::CFObject< TCFRef >::IsEqualTo (TCFRef *inRef*) const

Returns whether the object's CF reference is equalent to another CF reference.

Parameters:

inRef CF reference to test for equivalence'

Unlike HasSameRef(), this function tests the equality of the items' contents rather than the CF reference values. The meaning of equality depends on the actual CF type of the item.

Definition at line 278 of file SysCFObject.h.

6.50.3.14 template<typename TCFRef> bool PPx::CFObject< TCFRef >::IsValid () const

Returns whether the CF reference for the object is valid.

Returns:

Whether the CF reference for the object is valid

Definition at line 146 of file SysCFObject.h.

6.50.3.15 template<typename TCFRef> TCFRef PPx::CFObject< TCFRef >::UseRef () const

Returns the CF reference for the object.

Throws an exception if the reference is not valid.

Returns:

CF reference for the object

Call this routine if you are using the Ref in a context where it must be valid (not nil)

Definition at line 108 of file SysCFObject.h.

References PPx_ThrowIfNil_.

Referenced by PPx::CFObject< TCFRef >::CopyDescription(), PPx::CFObject< TCFRef >::CopyTypeIDDescription(), PPx::CFObject< TCFRef >::GetAllocator(), PPx::CFObject< TCFRef >::GetHashCode(), PPx::CFObject< TCFRef >::GetRetainCount(), PPx::CFObject< TCFRef >::GetTypeID(), and PPx::CFObject< CFURLRef >::operator TCFRef().

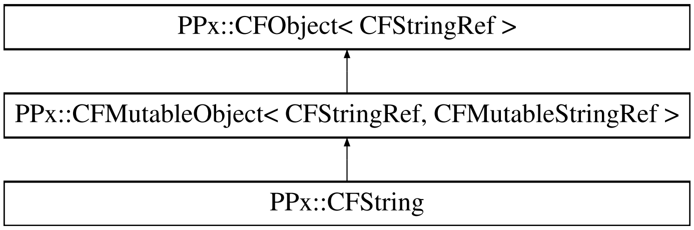
The documentation for this class was generated from the following file:

- SysCFObject.h

6.51 PPx::CFString Class Reference

`#include <SysCFString.h>`

Inheritance diagram for PPx::CFString::


PPx::CFObject< CFStringRef >
PPx::CFMutableObject< CFStringRef, CFMutableStringRef >
PPx::CFString

6.51.1 Detailed Description

Wrapper class for Core Foundation String.

Definition at line 27 of file SysCFString.h.

Public Member Functions

- CFString ()

  Default constructor.

- CFString (CFStringRef inStringRef, bool inRetain=true)

  Construct from an immutable CFStringRef.

- CFString (CFMutableStringRef inStringRef, bool inRetain)

  Constructs from an existing mutable CF string.

- CFString (ConstStringPtr inPascalString, CFStringEncoding inEncoding=encoding_System, CFAllocatorRef inAllocator=nil)

  Constructs from a Pascal string.

- CFString (const char ∗inCString, CFStringEncoding inEncoding=encoding_System, CFAllocatorRef inAllocator=nil)

  Constructs from a C string.

- CFString (const UniChar ∗inUniChars, CFIndex inCharCount, CFAllocatorRef inAllocator=nil)

  Constructs from a buffer of unicode characters.

- CFString (const HFSUniStr255 &inHFSUniStr, CFAllocatorRef inAllocator=nil)

  Constructs from a file system unicode string.

- CFString (const void ∗inBuffer, CFIndex inByteCount, CFStringEncoding inEncoding=encoding_System, bool inIsExternalRep=false, CFAllocatorRef inAllocator=nil)

  Constructs from a buffer of bytes.

- CFString (const std::string &inString, CFStringEncoding inEncoding=encoding_System, CFAllocatorRef inAllocator=nil)

  Constructs from a std::string.

- CFString (const CFString &inOriginal)

  Copy constructor.

- CFString & operator= (const CFString &inSource)

  Assignment Operator.

- CFIndex GetLength () const

  Returns number of unicode characters in the string.

- CFIndex GetByteLength (CFRange inRange=cfRange_All, CFStringEncoding inEncoding=encoding_System, UInt8 inLossByte=0, bool inIsExternalRep=false) const

  Returns the byte length of a range of characters if they were converted to the specified encoding.

- CFIndex GetByteRange (CFIndex inBufferSize, UInt8 ∗outBuffer, CFRange inRange=cfRange_All, CFStringEncoding inEncoding=encoding_System, UInt8 inLossByte=0, bool inIsExternalRep=false) const

  Fills in buffer with bytes converted from a range in the string to the specified encoding.

- UniChar GetCharacterAt (CFIndex inIndex) const

  Returns the unicode character at the given index in the string.

- UniChar operator[ ] (CFIndex inIndex) const

  Returns the unicode character at the given index in the string.

- void GetSubstring (CFRange inRange, UniChar ∗outBuffer) const

  Passes back a range of the string in a unicode characater buffer.

- const UniChar ∗ GetUniStringPtr () const

  Returns a pointer to a UniChar string.

- bool GetPascalString (StringPtr outBuffer, CFIndex inBufferSize, CFStringEncoding inEncoding=encoding_System) const

  Passes back the CFString as a Pascal string.

- ConstStringPtr GetPascalStringPtr (CFStringEncoding inEncoding=encoding_System) const

  Returns a pointer to a Pascal string.

- bool GetCString (char ∗outBuffer, CFIndex inBufferSize, CFStringEncoding inEncoding=encoding_System) const

  Passes back the CFString as a C string.

- const char ∗ GetCStringPtr (CFStringEncoding inEncoding=encoding_System) const

  Returns pointer to a C string.

- void GetString (std::string &outString, CFStringEncoding inEncoding=encoding_System) const

  Pass back a copy of the string as a std::string.

- FourCharCode Get4CharCodeValue () const

  Returns FourCharCode respresented by the string.

- template<typename TNumber> TNumber GetNumericValue () const

  Template function for returning a numeric value from a string.

- void Append (CFStringRef inString)

  Appends a CFStringRef to the string.

- void Append (const UniChar ∗inChars, CFIndex inCharCount)

  Appends an array of unicode characters to the string.

- void Append (ConstStringPtr inPascalString, CFStringEncoding inEncoding=encoding_System)

  Appends a Pascal string to the string.

- void Append (const char ∗inCString, CFStringEncoding inEncoding=encoding_System)

  Appends a C string to the string.

- void Insert (CFIndex inIndex, CFStringRef inString)

  Insert a CFStringRef into the string.

- void Pad (CFStringRef inPadString, CFIndex inLength, CFIndex inPadPosition)

  Pads or reduces a string to the specified length.

- void Delete (CFRange inRange)

  Deletes a range of characters from the string.

- void Replace (CFRange inRange, CFStringRef inReplacement)

  Replaces a range of characters in the string with another string.

- void ReplaceAll (CFStringRef inReplacement)

  Replaces entires contents of string with another string.

- template<typename TNumber> void AssignNumericValue (TNumber inNumber, CFAllocatorRef inAllocator=nil)

  Template function for setting a string from a numeric value.

- void **AssignNumericValue** (SInt8 inNumber, CFAllocatorRef inAllocator=nil)
- void **AssignNumericValue** (UInt8 inNumber, CFAllocatorRef inAllocator=nil)
- void Assign4CharCode (FourCharCode inCode, CFAllocatorRef inAllocator=nil)

  Sets contents of string to a four-character code.

- CFComparisonResult CompareTo (CFStringRef inStringRef, CFOptionFlags inOptions=0) const

  Compares string to a CFStringRef.

- CFRange FindInRange (CFRange inRange, CFStringRef inSearchStr, CFOptionFlags inOptions=0) const

  Finds a search string within a range of the string.

6.51.2 Constructor & Destructor Documentation

6.51.2.1 PPx::CFString::CFString (CFStringRef *inStringRef*, bool *inRetain* = true)

Construct from an immutable CFStringRef.

Parameters:

inStringRef CF string reference

inRetain Whether to retain the CF string

Note:

Toolbox functions which return a CFStringRef and have the word ”copy” or ”create” in their name return a string with a retain count of one. Pass false for inRetain when constructing from such a string. The CFString won’t retain the string, but will release the string in its destructor.

Toolbox functions which return a CFStringRef and have the word ”get” in their name return a string without incrementing its retain count. Pass true for inRetain when constructing from such a string. The CFString will retain the string, and later release it.

Definition at line 39 of file SysCFString.cp.

6.51.2.2 PPx::CFString::CFString (CFMutableStringRef *inStringRef*, bool *inRetain*)

Constructs from an existing mutable CF string.

Parameters:

inStringRef CF string reference

inRetain Whether to retain the CF string

Note:

Pass false for inRetain if you are transferring ownership of the CF string, which is normally the case if you obtained it from a call to a Toolbox function which copies a CF string.

Pase true for inRetain if the caller wants to maintain ownership of the CF string. In this case, we make a copy, so that multiple clients aren’t changing the same data.

Definition at line 64 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::AttachMutableRef().

6.51.2.3 PPx::CFString::CFString (ConstStringPtr *inPascalString*, CFStringEncoding *inEncoding* = encoding_System, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a Pascal string.

Parameters:

inPascalString Pascal string to copy

inEncoding Encoding of Pascal string

inAllocator CF Allocator

Definition at line 81 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.51.2.4 PPx::CFString::CFString (const char ∗ *inCString*, CFStringEncoding *inEncoding* = encoding_System, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a C string.

Parameters:

inCString C string to copy

inEncoding Encoding of C string

inAllocator CF Allocator

Definition at line 103 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.51.2.5 PPx::CFString::CFString (const UniChar ∗ *inUniChars*, CFIndex *inCharCount*, CFAllocatorRef *inAllocator* = nil)

Constructs from a buffer of unicode characters.

Parameters:

inUniChars Array of unicode characters

inCharCount Number of unicode characters to copy

inAllocator CF Allocator

Definition at line 125 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.51.2.6 PPx::CFString::CFString (const HFSUniStr255 & *inHFSUniStr*, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a file system unicode string.

Parameters:

inHFSUniStr A HFSUniStr255 string

inAllocator CF Allocator

Definition at line 146 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.51.2.7 PPx::CFString::CFString (const void ∗ *inBuffer*, CFIndex *inByteCount*, CFStringEncoding *inEncoding* = encoding_System, bool *inIsExternalRep* = false, CFAllocatorRef *inAllocator* = nil)

Constructs from a buffer of bytes.

Parameters:

inBuffer Pointer to a buffer

inByteCount Number of bytes to copy from buffer

inEncoding Encoding of bytes in buffer

inIsExternalRep Whether the bytes come from an external representation (where there may be a BOM (btye order mark) character

inAllocator CF Allocator

Definition at line 171 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.51.2.8 PPx::CFString::CFString (const std::string & *inString*, CFStringEncoding *inEncoding* = encoding_System, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a std::string.

Parameters:

inString std::string from which to copy

inEncoding Encoding of text in std::string

inAllocator CF Allocator

Definition at line 196 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.51.3 Member Function Documentation

6.51.3.1 void PPx::CFString::Append (const char $*$ *inCString*, CFStringEncoding *inEncoding* = encoding_System)

Appends a C string to the string.

Parameters:

inCString C string pointer

inEncoding Encoding of C string

Definition at line 571 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.2 void PPx::CFString::Append (ConstStringPtr *inPascalString*, CFStringEncoding *inEncoding* = encoding_System)

Appends a Pascal string to the string.

Parameters:

inPascalString Pascal string pointer

inEncoding Encoding of Pascal string

Definition at line 554 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.3 void PPx::CFString::Append (const UniChar $*$ *inChars*, CFIndex *inCharCount*)

Appends an array of unicode characters to the string.

Parameters:

inChars Pointer to unicode character array

inCharCount Number of unicdoe characters to append

Definition at line 537 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.4 void PPx::CFString::Append (CFStringRef *inString*)

Appends a CFStringRef to the string.

Parameters:

inString String to append

Definition at line 521 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

Referenced by PPx::XMLTreeBrowser::GetValue().

6.51.3.5 void PPx::CFString::Assign4CharCode (FourCharCode *inCode*, CFAllocatorRef *inAllocator* = nil)

Sets contents of string to a four-character code.

Parameters:

inCode Four-character code

inAllocator CF Allocator

Definition at line 686 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::AssignObject(), and CFString().

6.51.3.6 template<typename TNumber> void PPx::CFString::AssignNumericValue (TNumber *inNumber*, CFAllocatorRef *inAllocator* = nil)

Template function for setting a string from a numeric value.

Template parameter is the numeric type.

Parameters:

inNumber Number value

inAllocator CF Allocator

Definition at line 278 of file SysCFString.h.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::AssignObject(), and CFString().

Referenced by PPx::XMLTreeBuilder::MakePersistentElement(), and PPx::XMLTreeBuilder::MakeText().

6.51.3.7 CFComparisonResult PPx::CFString::CompareTo (CFStringRef *inStringRef*, CFOptionFlags *inOptions* = 0) const

Compares string to a CFStringRef.

Parameters:

inStringRef String to which to compare

inOptions Comparison options

Returns:

Comparison result

Return values:

kCFCompareLessThan String is less than input string

kCFCompareEqualTo String is equal to input string

kCFCompareGreaterThan String is greater than input string

Definition at line 711 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

6.51.3.8 void PPx::CFString::Delete (CFRange *inRange*)

Deletes a range of characters from the string.

Parameters:

inRange Range of characters to delete

Definition at line 633 of file SysCFString.cp.

References GetLength(), and PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.9 CFRange PPx::CFString::FindInRange (CFRange *inRange*, CFStringRef *inSearchStr*, CFOptionFlags *inOptions* = 0) const

Finds a search string within a range of the string.

Parameters:

inRange Range to search

inSearchStr String to search for

inOptions String comparison flags

Returns:

Range of where search string is in the String

Definition at line 731 of file SysCFString.cp.

References GetLength(), and PPx::CFObject< CFStringRef >::UseRef().

6.51.3.10 FourCharCode PPx::CFString::Get4CharCodeValue () const

Returns FourCharCode respresented by the string.

Returns:

FourCharCode respresented by the string

Definition at line 499 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

6.51.3.11 CFIndex PPx::CFString::GetByteLength (CFRange *inRange* = cfRange_All, CFStringEncoding *inEncoding* = encoding_System, UInt8 *inLossByte* = 0, bool *inIsExternalRep* = false) const

Returns the byte length of a range of characters if they were converted to the specified encoding.

Parameters:

inRange Range in string

inEncoding Encoding for which to determine byte length

inLossByte Byte for characters that can't be converted to encoding

inIsExternalRep Whether bytes are intended for external storage

Returns:

Bytes length of a range of characters if they were converted to the specified encoding

Definition at line 267 of file SysCFString.cp.

References GetLength(), and PPx::CFObject< CFStringRef >::UseRef().

Referenced by GetString(), and PPx::EditTextControl::SetText().

6.51.3.12 CFIndex PPx::CFString::GetByteRange (CFIndex *inBufferSize*, UInt8 ∗ *outBuffer*, CFRange *inRange* = cfRange_All, CFStringEncoding *inEncoding* = encoding_System, UInt8 *inLossByte* = 0, bool *inIsExternalRep* = false) const

Fills in buffer with bytes converted from a range in the string to the specified encoding.

Parameters:

- ***inBufferSize*** Size of buffer
- ***outBuffer*** Pointer to buffer
- ***inRange*** Range of characters in the string
- ***inEncoding*** Encoding to use for bytes
- ***inLossByte*** Byte to use for characters that can't be converted to the encoding
- ***inIsExternalRep*** Whether bytes are intended for external storage

Definition at line 295 of file SysCFString.cp.

References GetLength(), and PPx::CFObject< CFStringRef >::UseRef().

Referenced by PPx::EditTextControl::SetText().

6.51.3.13 UniChar PPx::CFString::GetCharacterAt (CFIndex *inIndex*) const

Returns the unicode character at the given index in the string.

Parameters:

- ***inIndex*** Index in string

Returns:

Unicode character at the given index

Definition at line 320 of file SysCFString.cp.

References GetLength(), and PPx::CFObject< CFStringRef >::UseRef().

Referenced by operator[ ]().

6.51.3.14 bool PPx::CFString::GetCString (char * *outBuffer*, CFIndex *inBufferSize*, CFStringEncoding *inEncoding* = encoding_System) const

Passes back the CFString as a C string.

Parameters:

- ***outBuffer*** Pointer to a C string buffer
- ***inBufferSize*** Size of the C striing buffer
- ***inEncoding*** Encoding to use to convert characters

Returns:

Whether the conversion to a C string was successful

Definition at line 432 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

6.51.3.15 const char ∗ PPx::CFString::GetCStringPtr (CFStringEncoding *inEncoding* = encoding_System) const

Returns pointer to a C string.

Parameters:

inEncoding Encoding to use to convert string

Returns:

C string pointer to internal buffer of the CFString

Returns nil if the internal storage format is not compatible with a C string

Definition at line 455 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

6.51.3.16 CFIndex PPx::CFString::GetLength () const

Returns number of unicode characters in the string.

Returns:

Number of unicode characters in the string

Definition at line 245 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

Referenced by Delete(), FindInRange(), GetByteLength(), GetByteRange(), GetCharacterAt(), GetString(), GetSubstring(), Insert(), Replace(), and PPx::FSUtils::StringToHFSUniStr().

6.51.3.17 SInt32 PPx::CFString::GetNumericValue () const [inline]

Template function for returning a numeric value from a string.

Templater parameter is the numeric type.

Returns:

Numeric value represented by the string

Definition at line 199 of file SysCFString.h.

References GetString().

6.51.3.18 bool PPx::CFString::GetPascalString (StringPtr *outBuffer*, CFIndex *inBufferSize*, CFStringEncoding *inEncoding* = encoding_System) const

Passes back the CFString as a Pascal string.

Parameters:

- ***outBuffer*** Pointer to a Pascal string buffer
- ***inBufferSize*** Size of Pascal string buffer
- ***inEncoding*** Encoding to use to convert string

Returns:

Whether the conversion to a Pascal string was successful

Definition at line 390 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

Referenced by PPx::FSObject::GetFSSpec().

6.51.3.19 ConstStringPtr PPx::CFString::GetPascalStringPtr (CFStringEncoding *inEncoding* = encoding_System) const

Returns a pointer to a Pascal string.

Parameters:

- ***inEncoding*** Encoding to use to convert string

Returns:

Pascal string pointer to internal buffer of the CFString

Returns nil if the internal storage format is not compatible with a Pascal string

Definition at line 413 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

6.51.3.20 void PPx::CFString::GetString (std::string & *outString*, CFStringEncoding *inEncoding* = encoding_System) const

Pass back a copy of the string as a std::string.

Parameters:

- ***outString*** std::string in which to copy string
- ***inEncoding*** Encoding to use to convert string

Definition at line 471 of file SysCFString.cp.

References GetByteLength(), GetLength(), and PPx::CFObject< CFStringRef >::UseRef().

Referenced by GetNumericValue().

6.51.3.21 void PPx::CFString::GetSubstring (CFRange *inRange*, UniChar ∗ *outBuffer*) const

Passes back a range of the string in a unicode characater buffer.

Parameters:

inRange Range of string to copy

outBuffer Pointer to unicode character buffer

Definition at line 354 of file SysCFString.cp.

References GetLength(), and PPx::CFObject< CFStringRef >::UseRef().

Referenced by PPx::FSUtils::StringToHFSUniStr().

6.51.3.22 const UniChar ∗ PPx::CFString::GetUniStringPtr () const

Returns a pointer to a UniChar string.

Returns nil if the internal storage format is not compatible with an array of UniChar characters

Definition at line 372 of file SysCFString.cp.

References PPx::CFObject< CFStringRef >::UseRef().

6.51.3.23 void PPx::CFString::Insert (CFIndex *inIndex*, CFStringRef *inString*)

Insert a CFStringRef into the string.

Parameters:

inIndex Index at which to insert

inString CFStringRef for string to insert

Definition at line 589 of file SysCFString.cp.

References GetLength(), and PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.24]

UniChar PPx::CFString::operator[ ] (CFIndex *inIndex*) const

Returns the unicode character at the given index in the string.

Parameters:

inIndex Index in string

Returns:

Unicode character at the given index

Definition at line 338 of file SysCFString.cp.

References GetCharacterAt().

6.51.3.25 void PPx::CFString::Pad (CFStringRef *inPadString*, CFIndex *inLength*, CFIndex *inPadPosition*)

Pads or reduces a string to the specified length.

Parameters:

inPadString String of padding characters

inLength Set string to this character length

inPadPosition Start position in pad string

If inLength is larger than the current string size, string grows to inLength and the extra characters are filled with characters from the pad string, starting at inPodPosition. If there are more extra characters than in the pad string, it wraps around to the start of the pad string.

If inLength is smaller tha the current string size, string is truncated to inLength characters.

Definition at line 616 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.26 void PPx::CFString::Replace (CFRange *inRange*, CFStringRef *inReplacement*)

Replaces a range of characters in the string with another string.

Parameters:

inRange Range of characters to replace

inReplacement String to put in place of character range

String grows or shrinks if range and replacment string have different lengths.

Definition at line 653 of file SysCFString.cp.

References GetLength(), and PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

6.51.3.27 void PPx::CFString::ReplaceAll (CFStringRef *inReplacement*)

Replaces entires contents of string with another string.

Parameters:

inReplacement Replacement string

Definition at line 670 of file SysCFString.cp.

References PPx::CFMutableObject< CFStringRef, CFMutableStringRef >::UseMutableRef().

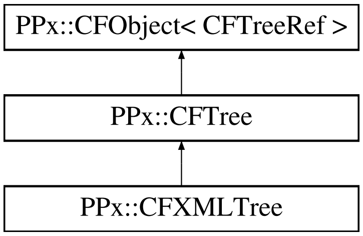
The documentation for this class was generated from the following files:

- SysCFString.h
- SysCFString.cp

6.52 PPx::CFTree Class Reference

`#include <SysCFTree.h>`

Inheritance diagram for PPx::CFTree::



6.52.1 Detailed Description

Wrapper class for Core Foundation Tree.

Definition at line 23 of file SysCFTree.h.

Public Member Functions

- CFTree ()

  Default constructor.

- CFTree (CFTreeRef inTreeRef, bool inRetain)

  Constructs from a CFTreeRef.

- CFTree (const CFTreeContext &inContext, CFAllocatorRef inAllocator=nil)

  Constructs an empty tree from a context.

- CFTree (const CFTree &inOriginal)

  Copy constructor.

- CFTree & operator= (const CFTree &inSource)

  Assignment operator.

- void GetContext (CFTreeContext &ioContext) const

  Passes back the context for the Tree.

- CFTree FindRoot () const

  Returns the root tree containing this Tree.

- CFTree GetParent () const

  Returns the parent tree of this Tree.

- CFTree GetNextSibling () const

  Returns the next sibling tree of this Tree.

- CFIndex GetChildCount () const

  Returns the number of children of this Tree.

- CFTree GetFirstChild () const

  Returns the first child tree of this Tree.

- CFTree GetChildAtIndex (CFIndex inIndex) const

  Returns the child tree at the given index.

- void GetChildren (CFTreeRef ∗outChildren) const

  Passes back an array of CFTreeRefs for all the children of this Tree.

- void SetContext (const CFTreeContext &inContext)

  Sets the context for the Tree.

- void RemoveFromParent ()

  Removes Tree from its parent tree.

- void InsertSibling (CFTreeRef inNewSibling)

  Inserts a sibling tree after this tree.

- void PrependChild (CFTreeRef inNewChild)

  Adds a child tree as the first child of this Tree.

- void AppendChild (CFTreeRef inNewChild)

  Adds a child tree as the last child of this Tree.

- void RemoveAllChildren ()

  Removes all child trees from this Tree.

- void ApplyFunctionToChildren (CFTreeApplierFunction inFunction, void ∗inParam)

  Calls function for each child of the Tree.

- void SortChildren (CFComparatorFunction inComparator, void ∗inParam)

  Sorts child trees.

6.52.2 Constructor & Destructor Documentation

6.52.2.1 PPx::CFTree::CFTree ()

Default constructor.

Note:

Default construction does not create an underlying tree data structure. You must call AttachRef() to associate this object with a valid CFTreeRef before you can use it.

Definition at line 19 of file SysCFTree.cp.

Referenced by FindRoot(), GetChildAtIndex(), GetFirstChild(), GetNextSibling(), and GetParent().

6.52.2.2 PPx::CFTree::CFTree (CFTreeRef *inTreeRef*, bool *inRetain*)

Constructs from a CFTreeRef.

Note:

Although CFTrees are always mutable, the Toolbox does not have a function to copy a CFTree. So both the caller and this object share the CFTreeRef. Any change made to the tree will be reflected in all trees that share the same CFTreeRef.

Definition at line 34 of file SysCFTree.cp.

6.52.2.3 PPx::CFTree::CFTree (const CFTreeContext & *inContext*, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs an empty tree from a context.

Parameters:

inContext CFTreeContext from which to make tree

inAllocator CF Allocator

Definition at line 51 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.52.2.4 PPx::CFTree::CFTree (const CFTree & *inOriginal*)

Copy constructor.

Note:
See comments for CFTree(CFTreeRef) about the CFTreeRef being shared rather than copied

Definition at line 70 of file SysCFTree.cp.

6.52.3 Member Function Documentation

6.52.3.1 void PPx::CFTree::AppendChild (CFTreeRef *inNewChild*)

Adds a child tree as the last child of this Tree.

Parameters:
inNewChild Child tree to add

Definition at line 284 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLTreeBuilder::AddChildDataValue(), PPx::XMLEncoderFuncs::EncodeData(), PPx::XMLEncoderFuncs::EncodeVector(), and PPx::XMLTreeBuilder::FormatDescriptorsTree().

6.52.3.2 void PPx::CFTree::ApplyFunctionToChildren (CFTreeApplierFunction *inFunction*, void ∗ *inParam*)

Calls function for each child of the Tree.

Parameters:
inFunction Function to call

inParam User-defined parameter for the function

Definition at line 312 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

6.52.3.3 CFTree PPx::CFTree::FindRoot () const

Returns the root tree containing this Tree.

Returns:

Root tree containing this Tree

Definition at line 123 of file SysCFTree.cp.

References CFTree(), and PPx::CFObject< CFTreeRef >::UseRef().

6.52.3.4 CFTree PPx::CFTree::GetChildAtIndex (CFIndex *inIndex*) const

Returns the child tree at the given index.

Parameters:

inIndex Index of child tree

Returns:

Child tree at the given index

Definition at line 195 of file SysCFTree.cp.

References CFTree(), GetChildCount(), and PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLDecoderFuncs::DecodeVector(), and PPx::XMLTreeBrowser::GetValue().

6.52.3.5 CFIndex PPx::CFTree::GetChildCount () const

Returns the number of children of this Tree.

Returns:

Nnumber of children of this Tree

Definition at line 165 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLDecoderFuncs::DecodeVector(), GetChildAtIndex(), and PPx::XMLTreeBrowser::GetValue().

6.52.3.6 void PPx::CFTree::GetChildren (CFTreeRef ∗ *outChildren*) const

Passes back an array of CFTreeRefs for all the children of this Tree.

Parameters:

outChildren Pointer to array of CFTreeRef

Definition at line 211 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

6.52.3.7 void PPx::CFTree::GetContext (CFTreeContext & *ioContext*) const

Passes back the context for the Tree.

Parameters:

ioContext CFTreeContext for the Tree

Note:

Caller must fill in the version field of ioContext with a valid version number. See <CoreFouncation/CFTree.h> for CFTreeContext version numbers.

Definition at line 108 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

6.52.3.8 CFTree PPx::CFTree::GetFirstChild () const

Returns the first child tree of this Tree.

Returns:

First child tree of this Tree

Definition at line 179 of file SysCFTree.cp.

References CFTree(), and PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLDecoderFuncs::DecodeVector(), PPx::XMLTreeBuilder::FormatDescriptorsTree(), and PPx::XMLTreeBrowser::GetStructField().

6.52.3.9 CFTree PPx::CFTree::GetNextSibling () const

Returns the next sibling tree of this Tree.

Returns:

Next sibling tree of this Tree

Definition at line 151 of file SysCFTree.cp.

References CFTree(), and PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLTreeBrowser::GetStructField().

6.52.3.10 CFTree PPx::CFTree::GetParent () const

Returns the parent tree of this Tree.

Returns:

Parent tree of this Tree

Definition at line 137 of file SysCFTree.cp.

References CFTree(), and PPx::CFObject< CFTreeRef >::UseRef().

6.52.3.11 void PPx::CFTree::InsertSibling (CFTreeRef *inNewSibling*)

Inserts a sibling tree after this tree.

Parameters:

inNewSibling Tree to insert as a sibling after this tree

Definition at line 254 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLTreeBuilder::FormatDescriptorsTree().

6.52.3.12 CFTree & PPx::CFTree::operator= (const CFTree & *inSource*)

Assignment operator.

Note:

See comments for CFTree(CFTreeRef) about the CFTreeRef being shared rather than copied

Definition at line 87 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::AssignObject().

6.52.3.13 void PPx::CFTree::PrependChild (CFTreeRef *inNewChild*)

Adds a child tree as the first child of this Tree.

Parameters:

inNewChild Child tree to add

Definition at line 269 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLTreeBuilder::FormatDescriptorsTree().

6.52.3.14 void PPx::CFTree::SetContext (const CFTreeContext & *inContext*)

Sets the context for the Tree.

Parameters:

inContext CFTreeContext to use for this tree

Definition at line 227 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

6.52.3.15 void PPx::CFTree::SortChildren (CFComparatorFunction *inComparator*, void ∗ *inParam*)

Sorts child trees.

Parameters:

inComparator Comparison function for trees

inParam User-defined parameter for comparison function

Definition at line 329 of file SysCFTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

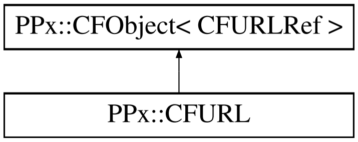
The documentation for this class was generated from the following files:

- SysCFTree.h
- SysCFTree.cp

6.53 PPx::CFURL Class Reference

`#include <SysCFURL.h>`

Inheritance diagram for PPx::CFURL::



6.53.1 Detailed Description

Wrapper class for Core Foundation URL.

Definition at line 25 of file SysCFURL.h.

Public Member Functions

- CFURL ()

  Default constructor.

- CFURL (CFURLRef inURLRef, bool inRetain)

  Constructs from a CFURLRef.

- CFURL (const FSRef &inFSRef, CFAllocatorRef inAllocator=nil)

  Constructs from a FSRef.

- CFURL (CFStringRef inString, CFURLRef inBaseURL=nil, CFAllocatorRef inAllocator=nil)

  Constructs from a string and base URL.

- CFURL (const void ∗inBuffer, CFIndex inBufferLength, CFStringEncoding inEncoding=encoding_System, CFURLRef inBaseURL=nil, CFAllocatorRef inAllocator=nil)

  Constructs from text in a buffer and a base URL.

- CFURL (CFStringRef inFilePath, bool inIsDirectory, CFURLRef inBaseURL=nil, CFURLPathStyle inPathStyle=kCFURLPOSIXPathStyle, CFAllocatorRef inAllocator=nil)

  Constructs from a file system path and a base URL.

- CFURL (const void ∗inPathBuffer, CFIndex inBufferLength, bool inIsDirectory, CFURLRef inBaseURL=nil, CFAllocatorRef inAllocator=nil)

  Constructs from a file system representation and a base URL.

- CFURL (const CFURL &inOriginal)

  Copy constructor.

- CFURL & operator= (const CFURL &inSource)

  Assignment operator.

- CFData GetAsData (bool inEscapeWhitespace, CFStringEncoding inEncoding=encoding_System, CFAllocatorRef inAllocator=nil) const

  Extracts content of the URL into a CFDataRef.

- CFString GetString () const

  Returns the string component of the URL.

- CFURL GetBaseURL () const

  Returns the Base URL of this URL.

- bool CanBeDecomposed () const

  Returns whether the URL can be decomposed into separate pieces.

- bool HasDirectoryPath () const

  Returns whether the URL represents a directory.

- SInt32 GetPortNumber () const

  Returns URL's port number.

- bool GetFSRef (FSRef &outFSRef) const

  Passes back the FSRef corresponding to the URL.

- CFString GetScheme () const

  Returns scheme portion of the URL.

- CFString GetNetLocation () const

  Returns the net location protion of the URL.

- CFString GetPath () const

  Returns the path of the URL.

- CFString GetStrictPath (bool inIsAbsolute) const

  Returns the strict path of the URL.

- CFString GetFileSystemPath (CFURLPathStyle inPathStyle=kCFURLPOSIXPathStyle) const

  Returns the file system path of the URL.

- CFString GetResourceSpecifier () const

  Returns the resource specifier of the URL.

- CFString GetHostName () const

  Returns the host name of the URL.

- CFString GetUserName () const

  Returns the user name of the URL.

- CFString GetPassword () const

  Returns the password of the URL.

- CFString GetParameterString (CFStringRef inEscapedChars=nil) const

  Returns the parameter string of the URL.

- CFString GetQueryString (CFStringRef inEscapedChars=nil) const

  Returns the query string of the URL.

- CFString GetFragment (CFStringRef inEscapedChars=nil) const

  Returns the fragment of the URL.

- CFString GetLastPathComponent () const

  Returns the last path component of the URL.

- CFString GetPathExtension () const

  Returns the path extension of the URL.

- void AppendPathComponent (CFStringRef inPathComponent, bool inIsDirectory)

  Appends a path component to the URL.

- void DeleteLastPathComponent ()

  Deletes the last path component from the URL.

- void AppendPathExtension (CFStringRef inExtension)

  Appends a path extension to the URL.

- void DeletePathExtension ()

Deletes the path extension from the URL.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 PPx::CFURL::CFURL (CFURLRef *inURLRef*, bool *inRetain*)

Constructs from a CFURLRef.

Parameters:

inURLRef CFURLRef to use for this object

inRetain Whether to retain the input CFURLRef

Definition at line 27 of file SysCFURL.cp.

6.53.2.2 PPx::CFURL::CFURL (const FSRef & *inFSRef*, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a FSRef.

Parameters:

inFSRef FSRef which specifies an item in the file system

inAllocator CF Allocator

Definition at line 44 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.53.2.3 PPx::CFURL::CFURL (CFStringRef *inString*, CFURLRef *inBaseURL* = nil, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a string and base URL.

Parameters:

inString String part of URL

inBaseURL Base URL for this URL

inAllocator CF Allocator

Definition at line 64 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.53.2.4 PPx::CFURL::CFURL (const void ∗ *inBuffer*, CFIndex *inBufferLength*, CFStringEncoding *inEncoding* = encoding_System, CFURLRef *inBaseURL* = nil, CFAllocatorRef *inAllocator* = nil)

Constructs from text in a buffer and a base URL.

Parameters:

inBuffer Pointer to text buffer

inBufferLength Size of buffer

inEncoding Encoding of characters in buffer

inBaseURL Base URL for this URL

inAllocator CF Allocator

Definition at line 87 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.53.2.5 PPx::CFURL::CFURL (CFStringRef *inFilePath*, bool *inIsDirectory*, CFURLRef *inBaseURL* = nil, CFURLPathStyle *inPathStyle* = kCFURLPOSIXPathStyle, CFAllocatorRef *inAllocator* = nil)

Constructs from a file system path and a base URL.

Parameters:

inFilePath File path

inIsDirectory Whether the item is a directory

inBaseURL Base URL for this URL

inPathStyle OS Path Style for URL (POSIX, HFS, Windows)

inAllocator CF Allocator

Definition at line 114 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.53.2.6 PPx::CFURL::CFURL (const void ∗ *inPathBuffer*, CFIndex *inBufferLength*, bool *inIsDirectory*, CFURLRef *inBaseURL* = nil, CFAllocatorRef *inAllocator* = nil)

Constructs from a file system representation and a base URL.

Parameters:

inPathBuffer Pointer to buffer containing path

inBufferLength Length of path buffer

inIsDirectory Whether the item is a directory

inBaseURL Base URL for this URL

inAllocator CF Allocator

Definition at line 149 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.53.3 Member Function Documentation

6.53.3.1 void PPx::CFURL::AppendPathComponent (CFStringRef *inPathComponent*, bool *inIsDirectory*)

Appends a path component to the URL.

Parameters:

inPathComponent Path component to append

inIsDirectory Whether the path compontent is a directory

Definition at line 556 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), PPx::CFObject< CFURLRef >::GetAllocator(), PPx_ThrowIfCFCreateFailed_, and PPx::CFObject< CFURLRef >::UseRef().

6.53.3.2 void PPx::CFURL::AppendPathExtension (CFStringRef *inExtension*)

Appends a path extension to the URL.

Parameters:

inExtension Extension to append

Definition at line 595 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::AttachRef(), PPx::CFObject< CFURLRef >::GetAllocator(), PPx_ThrowIfCFCreateFailed_, and PPx::CFObject< CFURLRef >::UseRef().

6.53.3.3 bool PPx::CFURL::CanBeDecomposed () const

Returns whether the URL can be decomposed into separate pieces.

Returns:

Whether the URL can be decomposed into separate pieces

If it can be decomposed, you can get the scheme, net location, path and resource specifier as separate strings.

If it cannot be deomposed, you can get the scheme and resource specifier, but the net location and path will be nil.

Definition at line 268 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.4 CFData PPx::CFURL::GetAsData (bool *inEscapeWhitespace*, CFStringEncoding *inEncoding* = encoding_System, CFAllocatorRef *inAllocator* = nil) const

Extracts content of the URL into a CFDataRef.

Parameters:

inEscapeWhitespace Whether to escape whitespace characters

inEncoding Encoding for characters

inAllocator CF Allocator

Returns:

CFData object containing the content of the URL

Definition at line 215 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.5 CFURL PPx::CFURL::GetBaseURL () const

Returns the Base URL of this URL.

Returns:

Base URL of this URL

Definition at line 248 of file SysCFURL.cp.

References CFURL(), and PPx::CFObject< CFURLRef >::UseRef().

6.53.3.6 CFString PPx::CFURL::GetFileSystemPath (CFURLPathStyle *inPathStyle* = kCFURLPOSIXPathStyle) const

Returns the file system path of the URL.

Parameters:

inPathStyle OS Path Style for URL (POSIX, HFS, Windows)

Returns:

File system path of the URL

Definition at line 403 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

Referenced by PPx::FSObject::GetPath().

6.53.3.7 CFString PPx::CFURL::GetFragment (CFStringRef *inEscapedChars* = nil) const

Returns the fragment of the URL.

Returns:

Fragment of the URL

The fragment is the text following a # character, generally used to indicate locations in a single file

Definition at line 510 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.8 bool PPx::CFURL::GetFSRef (FSRef & *outFSRef*) const

Passes back the FSRef corresponding to the URL.

Parameters:

outFSRef FSRef corresponding to the URL

Returns:

Whether there is a FSRef corresponding to the URL

Definition at line 314 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.9 CFString PPx::CFURL::GetHostName () const

Returns the host name of the URL.

Returns:
Host name of the URL

Definition at line 433 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.10 CFString PPx::CFURL::GetLastPathComponent () const

Returns the last path component of the URL.

Returns:
Last path component of the URL

Definition at line 527 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.11 CFString PPx::CFURL::GetNetLocation () const

Returns the net location protion of the URL.

Returns:
Net location portion of the URL

The net location contains the host name or IP address, and username and password

Definition at line 349 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.12 CFString PPx::CFURL::GetParameterString (CFStringRef *inEscapedChars* = nil) const

Returns the parameter string of the URL.

Returns:
Parameter string of the URL

Definition at line 475 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.13 CFString PPx::CFURL::GetPassword () const

Returns the password of the URL.

Returns:

Password of the URL

Definition at line 461 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.14 CFString PPx::CFURL::GetPath () const

Returns the path of the URL.

Returns:

Path of the URL

Definition at line 363 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.15 CFString PPx::CFURL::GetPathExtension () const

Returns the path extension of the URL.

Returns:

Path extension of the URL

Definition at line 541 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.16 SInt32 PPx::CFURL::GetPortNumber () const

Returns URL's port number.

Returns:

Port number of URL

Return values:

-1 No port number specified

Definition at line 298 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.17 CFString PPx::CFURL::GetQueryString (CFStringRef *inEscapedChars* = nil) const

Returns the query string of the URL.

Returns:
Query string of the URL

Definition at line 491 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.18 CFString PPx::CFURL::GetResourceSpecifier () const

Returns the resource specifier of the URL.

Returns:
Resource specifier of the URL

Definition at line 419 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.19 CFString PPx::CFURL::GetScheme () const

Returns scheme portion of the URL.

Returns:
Scheme portion of the URL

The scheme is the transport type, such as http or ftp

Definition at line 332 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.20 CFString PPx::CFURL::GetStrictPath (bool *outIsAbsolute*) const

Returns the strict path of the URL.

Parameters:
outIsAbsolute Whether the path is a asboute

Returns:
Strict path of the URL

Definition at line 379 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.21 CFString PPx::CFURL::GetString () const

Returns the string component of the URL.

Returns:
String component of the URL

Definition at line 234 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.22 CFString PPx::CFURL::GetUserName () const

Returns the user name of the URL.

Returns:
User name of the URL

Definition at line 447 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.53.3.23 bool PPx::CFURL::HasDirectoryPath () const

Returns whether the URL represents a directory.

Returns:
Whether the URL represents a directory

Definition at line 282 of file SysCFURL.cp.

References PPx::CFObject< CFURLRef >::UseRef().

The documentation for this class was generated from the following files:

- SysCFURL.h
- SysCFURL.cp

6.54 PPx::CFXMLElement Class Reference

```
#include <SysCFXMLNode.h>
```

6.54.1 Detailed Description

Helper class for accessing the attributes of an XML Node for an element.

Definition at line 95 of file SysCFXMLNode.h.

Public Member Functions

- CFXMLElement (const CFXMLNode &inNode)

  Constructs from an XML element node.

- CFIndex GetAttributeCount () const

  Returns the number of attributes in the element.

- CFStringRef GetAttributeValue (CFIndex inAttrIndex) const

  Returns the value for the attribute specified by index.

- CFStringRef GetAttributeValue (CFStringRef inAttrName) const

  Returns the value of the attribute specify by name.

6.54.2 Constructor & Destructor Documentation

6.54.2.1 PPx::CFXMLElement::CFXMLElement (const CFXMLNode & *inNode*)

Constructs from an XML element node.

Parameters:

inNode Access the attributes of this element node

Definition at line 259 of file SysCFXMLNode.cp.

References PPx::CFObject< CFDictionaryRef >::AttachRef(), PPx::CFObject< CFArrayRef >::AttachRef(), PPx::CFXMLNode::GetInfoPtr(), PPx::CFXMLNode::GetTypeCode(), and PPx_Throw_.

6.54.3 Member Function Documentation

6.54.3.1 CFIndex PPx::CFXMLElement::GetAttributeCount () const

Returns the number of attributes in the element.

Returns:
Number of attributes in the element

Definition at line 286 of file SysCFXMLNode.cp.

References PPx::CFArray< CFStringRef >::GetCount(), and PPx::CFObject< CFArrayRef >::IsValid().

6.54.3.2 CFStringRef PPx::CFXMLElement::GetAttributeValue (CFStringRef *inAttrName*) const

Returns the value of the attribute specify by name.

Parameters:
inAttrName Name of the attribute

Returns:
Value of the attribute

Definition at line 321 of file SysCFXMLNode.cp.

References PPx::CFDictionary< CFStringRef, CFStringRef >::GetValue().

6.54.3.3 CFStringRef PPx::CFXMLElement::GetAttributeValue (CFIndex *inAttrIndex*) const

Returns the value for the attribute specified by index.

Parameters:
inAttrIndex Attribute index number

Returns:
Value of the attribute

Attributes are indexed in the order that they appear in the XML data

Definition at line 304 of file SysCFXMLNode.cp.

References PPx::CFDictionary< CFStringRef, CFStringRef >::GetValue().

Referenced by PPx::XMLTreeBrowser::GetStructField().

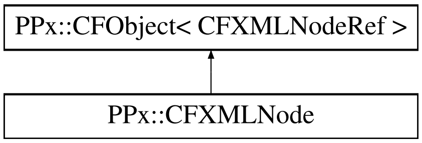
The documentation for this class was generated from the following files:

- SysCFXMLNode.h
- SysCFXMLNode.cp

6.55 PPx::CFXMLNode Class Reference

`#include <SysCFXMLNode.h>`

Inheritance diagram for PPx::CFXMLNode::



6.55.1 Detailed Description

Wrapper class for Core Foundation XML Node.

Definition at line 26 of file SysCFXMLNode.h.

Public Member Functions

- CFXMLNode ()

  Default constructor.

- CFXMLNode (CFXMLNodeRef inNodeRef, bool inRetain)

  Constructs from a CFXMLNodeRef.

- CFXMLNode (CFXMLNodeTypeCode inType, CFStringRef inDataString, const void ∗inInfoPtr, CFAllocatorRef inAllocator=nil, CFIndex inVersion=kCFXMLNodeCurrentVersion)

  Constructs from a specified node type code and associated data.

- CFXMLNode (const CFXMLElementInfo &inElementInfo, CFStringRef inTagName, CFAllocatorRef inAllocator=nil, CFIndex inVersion=kCFXMLNodeCurrentVersion)

  Constructs an XML Element node.

- CFXMLNode (const CFXMLDocumentInfo &inDocInfo, CFAllocatorRef inAllocator=nil, CFIndex inVersion=kCFXMLNodeCurrentVersion)

  Constructs an XML Document node.

- CFXMLNode (const CFXMLProcessingInstructionInfo &inInstructionInfo, CFStringRef inTarget, CFAllocatorRef inAllocator=nil, CFIndex inVersion=kCFXMLNodeCurrentVersion)

Constructs an XML processing instruction node.

- CFXMLNode (CFStringRef inText, CFXMLNodeTypeCode inType=kCFXMLNodeTypeText, CFAllocatorRef inAllocator=nil, CFIndex inVersion=kCFXMLNodeCurrentVersion)

  Constructs a node requiring a single text string as its information.

- CFXMLNode (const CFXMLNode &inOriginal)

  Copy constructor.

- CFXMLNode & operator= (const CFXMLNode &inSource)

  Assignment operator.

- CFIndex GetVersion () const

  Returns the node version.

- CFXMLNodeTypeCode GetTypeCode () const

  Returns the node type.

- CFString GetString () const

  Returns the data string for the nade.

- const void ∗ GetInfoPtr () const

  Returns a information pointer for the node.

6.55.2 Constructor & Destructor Documentation

6.55.2.1 PPx::CFXMLNode::CFXMLNode (CFXMLNodeRef *inNodeRef*, bool *inRetain*)

Constructs from a CFXMLNodeRef.

Parameters:

inNodeRef Node ref to use for this XMLNode

inRetain Whether to retain the input node ref

Definition at line 27 of file SysCFXMLNode.cp.

6.55.2.2 PPx::CFXMLNode::CFXMLNode (CFXMLNodeTypeCode *inType*, CFStringRef *inDataString*, const void $*$ *inInfoPtr*, CFAllocatorRef *inAllocator* = nil, CFIndex *inVersion* = kCFXMLNodeCurrentVersion)

Constructs from a specified node type code and associated data.

Parameters:

inType Type of node

inDataString String information for node

inInfoPtr Pointer to extra data for the ndoe

inAllocator CF Allocator,

inVersion Node version

The data pointed to by inInfoPtr depends on the type of the node. This is the generic constructor. Other constructors create specific node types and pass a reference to the struct containing the data for that type of node.

Definition at line 52 of file SysCFXMLNode.cp.

6.55.2.3 PPx::CFXMLNode::CFXMLNode (const CFXMLElementInfo & *inElementInfo*, CFStringRef *inTagName*, CFAllocatorRef *inAllocator* = nil, CFIndex *inVersion* = kCFXMLNodeCurrentVersion)

Constructs an XML Element node.

Parameters:

inElementInfo XML Element information

inTagName Name for XML element tag

inAllocator CF Allocator

inVersion Node version

Definition at line 73 of file SysCFXMLNode.cp.

6.55.2.4 PPx::CFXMLNode::CFXMLNode (const CFXMLDocumentInfo & *inDocInfo*, CFAllocatorRef *inAllocator* = nil, CFIndex *inVersion* = kCFXMLNodeCurrentVersion) `[explicit]`

Constructs an XML Document node.

Parameters:

inDocInfo Document information

inAllocator CF Allocator

inVersion Node version

Definition at line 93 of file SysCFXMLNode.cp.

6.55.2.5 PPx::CFXMLNode::CFXMLNode (const CFXMLProcessingInstructionInfo & *inInstructionInfo*, CFStringRef *inTarget*, CFAllocatorRef *inAllocator* = nil, CFIndex *inVersion* = kCFXMLNodeCurrentVersion)

Constructs an XML processing instruction node.

Parameters:

inInstructionInfo Processing instruction information

inTarget XML Target

inAllocator CF Allocator

inVersion Node version

Definition at line 113 of file SysCFXMLNode.cp.

6.55.2.6 PPx::CFXMLNode::CFXMLNode (CFStringRef *inText*, CFXMLNodeTypeCode *inType* = kCFXMLNodeTypeText, CFAllocatorRef *inAllocator* = nil, CFIndex *inVersion* = kCFXMLNodeCurrentVersion) `[explicit]`

Constructs a node requiring a single text string as its information.

Parameters:

inText Text string

inType Type of node

inAllocator CF Allocator

inVersion Node version

Definition at line 134 of file SysCFXMLNode.cp.

6.55.3 Member Function Documentation

6.55.3.1 const void ∗ PPx::CFXMLNode::GetInfoPtr () const

Returns a information pointer for the node.

Returns:

Information pointer for the node

The struct pointed to by the information pointer depends on the type of the node

Definition at line 224 of file SysCFXMLNode.cp.

References PPx::CFObject< CFXMLNodeRef >::UseRef().

Referenced by PPx::CFXMLElement::CFXMLElement().

6.55.3.2 CFString PPx::CFXMLNode::GetString () const

Returns the data string for the nade.

Returns:
Data string for the node

Definition at line 207 of file SysCFXMLNode.cp.

References PPx::CFObject< CFXMLNodeRef >::UseRef().

Referenced by PPx::XMLDecoderFuncs::DecodeVector(), PPx::XMLTreeBrowser::GetFieldValue(), and PPx::XMLTreeBrowser::GetValue().

6.55.3.3 CFXMLNodeTypeCode PPx::CFXMLNode::GetTypeCode () const

Returns the node type.

Returns:
Node type

Definition at line 193 of file SysCFXMLNode.cp.

References PPx::CFObject< CFXMLNodeRef >::UseRef().

Referenced by PPx::CFXMLElement::CFXMLElement(), and PPx::XMLTreeBrowser::GetValue().

6.55.3.4 CFIndex PPx::CFXMLNode::GetVersion () const

Returns the node version.

Returns:
Node version

Definition at line 179 of file SysCFXMLNode.cp.

References PPx::CFObject< CFXMLNodeRef >::UseRef().

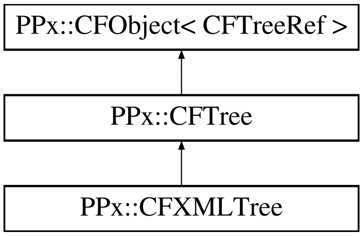
The documentation for this class was generated from the following files:

- SysCFXMLNode.h
- SysCFXMLNode.cp

6.56 PPx::CFXMLTree Class Reference

`#include <SysCFXMLTree.h>`

Inheritance diagram for PPx::CFXMLTree::



6.56.1 Detailed Description

Wrapper class for Core Foundation XML Tree.

Definition at line 27 of file SysCFXMLTree.h.

Public Member Functions

- CFXMLTree ()

  Default constructor.

- CFXMLTree (CFXMLTreeRef inTreeRef, bool inRetain)

  Constructs from a CFXMLTreeRef.

- CFXMLTree (CFDataRef inXMLData, CFURLRef inSourceURL=nil, CFOptionFlags inOptions=kCFXMLParserSkipWhitespace, CFIndex inNodeVersion=kCFXMLNodeCurrentVersion, CFAllocatorRef inAllocator=nil)

  Constructs from a CFDataRef containing XML text.

- CFXMLTree (CFURLRef inSourceURL, CFOptionFlags inOptions=kCFXMLParserSkipWhitespace, CFIndex inNodeVersion=kCFXMLNodeCurrentVersion, CFAllocatorRef inAllocator=nil)

  Constructs from a CFURLRef containing XML text.

- CFXMLTree (CFXMLNodeRef inNodeRef, CFAllocatorRef inAllocator=nil)

  Constructs from an XML Node.

- CFXMLTree (const CFTree &inTree)

Constructs from a CF Tree.

- CFXMLTree (const CFXMLTree &inOriginal)

  Copy constructor.

- CFXMLTree & operator= (const CFXMLTree &inSource)

  Assignment operator.

- CFData GetXMLData (CFAllocatorRef inAllocator=nil) const

  Generates XML text suitable for external output from the XML tree.

- CFXMLNode GetNode () const

  Returns the XML node associated with the XML tree.

6.56.2 Constructor & Destructor Documentation

6.56.2.1 PPx::CFXMLTree::CFXMLTree ()

Default constructor.

Default construction does not create an underlying tree data structure. You must call AttachRef() to associate this object with a valid CFXMLTreeRef before you can use it.

Definition at line 19 of file SysCFXMLTree.cp.

6.56.2.2 PPx::CFXMLTree::CFXMLTree (CFXMLTreeRef *inTreeRef*, bool *inRetain*)

Constructs from a CFXMLTreeRef.

Parameters:

inTreeRef CFXMLTreeRef to use for this Tree

inRetain Whether to retain the input CFXMLTreeRef

Note:

Although CFXMLTreeRefs are always mutable, the Toolbox does not have a function to copy a CFXMLTreeRef. So both the caller and this object share the CFXMLTreeRef. Any change made to the tree will be reflected in all trees that share the same CFXMLTreeRef.

Definition at line 37 of file SysCFXMLTree.cp.

6.56.2.3 PPx::CFXMLTree::CFXMLTree (CFDataRef *inXMLData*, CFURLRef *inSourceURL* = nil, CFOptionFlags *inOptions* = kCFXMLParserSkipWhitespace, CFIndex *inNodeVersion* = kCFXMLNodeCurrentVersion, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a CFDataRef containing XML text.

Parameters:

inXMLData XML Data

inSourceURL Source URL for the XML data, may be nil

inOptions XML parsing options

inNodeVersion Node Version

inAllocator CF Allocator

Parses the xML data and builds a hierarchy of XML Trees/Nodes that corresponds the the structure of the XML data

Definition at line 60 of file SysCFXMLTree.cp.

References PPx::CFObject< CFTreeRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.56.2.4 PPx::CFXMLTree::CFXMLTree (CFURLRef *inSourceURL*, CFOptionFlags *inOptions* = kCFXMLParserSkipWhitespace, CFIndex *inNodeVersion* = kCFXMLNodeCurrentVersion, CFAllocatorRef *inAllocator* = nil) `[explicit]`

Constructs from a CFURLRef containing XML text.

Parameters:

inSourceURL URL for XML document

inOptions XML parsing options

inNodeVersion Node Version

inAllocator CF Allocator

Parses the xML data and builds a hierarchy of XML Trees/Nodes that corresponds the the structure of the XML data

Definition at line 88 of file SysCFXMLTree.cp.

References PPx::CFObject< CFTreeRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.56.2.5 PPx::CFXMLTree::CFXMLTree (CFXMLNodeRef *inNodeRef*, CFAllocatorRef *inAllocator* = nil) [explicit]

Constructs from an XML Node.

Parameters:

inNodeRef XML Node for the Tree

inAllocator CF Allocator

Every XML Tree has an associated XML Node. The Tree stores information about the tree structure relationships (parent, sibling, and child trees) and the Node stores the XML entity information.

Definition at line 114 of file SysCFXMLTree.cp.

References PPx::CFObject< CFTreeRef >::AttachRef(), and PPx_ThrowIfCFCreateFailed_.

6.56.2.6 PPx::CFXMLTree::CFXMLTree (const CFTree & *inTree*) [explicit]

Constructs from a CF Tree.

Parameters:

inTree CFTree object

Note:

Although CFXMLTree is a subclass of CFTree, the underlying Core Foundation type is the same, as CFXMLTreeRef is typedef'd to CFTreeRef. So we allow construction of a CFXMLTree from a CFTree, but the caller is responsible for ensuring this is appropriate.

Definition at line 137 of file SysCFXMLTree.cp.

6.56.2.7 PPx::CFXMLTree::CFXMLTree (const CFXMLTree & *inOriginal*)

Copy constructor.

Note:

See comments for CFXMLTree(CFXMLTreeRef) about the CFXMLTreeRef being shared rather than copied

Definition at line 153 of file SysCFXMLTree.cp.

6.56.3 Member Function Documentation

6.56.3.1 CFXMLNode PPx::CFXMLTree::GetNode () const

Returns the XML node associated with the XML tree.

Returns:

XML Node associated with the XML Tree

Definition at line 202 of file SysCFXMLTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

Referenced by PPx::XMLDecoderFuncs::DecodeVector(), PPx::XMLTreeBrowser::GetFieldValue(), PPx::XMLTreeBrowser::GetStructField(), and PPx::XMLTreeBrowser::GetValue().

6.56.3.2 CFData PPx::CFXMLTree::GetXMLData (CFAllocatorRef *inAllocator* = nil) const

Generates XML text suitable for external output from the XML tree.

Parameters:

inAllocator CF Allocator

Definition at line 186 of file SysCFXMLTree.cp.

References PPx::CFObject< CFTreeRef >::UseRef().

6.56.3.3 CFXMLTree & PPx::CFXMLTree::operator= (const CFXMLTree & *inSource*)

Assignment operator.

Note:

See comments for CFXMLTree(CFXMLTreeRef) about the CFXMLTreeRef being shared rather than copied

Definition at line 170 of file SysCFXMLTree.cp.

References PPx::CFObject< CFTreeRef >::AssignObject().

The documentation for this class was generated from the following files:

- SysCFXMLTree.h
- SysCFXMLTree.cp

6.57 PPx::CGContextSaver Class Reference

`#include <PPxViewUtils.h>`

6.57.1 Detailed Description

Saves and restores a Core Graphics context.

Definition at line 61 of file PPxViewUtils.h.

Public Member Functions

- CGContextSaver (CGContextRef inContext)

  Constructor from a CGContextRef.

- ∼CGContextSaver ()

  Destructor.

- CGContextRef Get () const

  Returns the CGContext.

- void Save (CGContextRef inContext)

  Saves the input CGContext and restores the one formerly being saved.

- void Restore ()

  Restores the CGContext that was being saved.

6.57.2 Constructor & Destructor Documentation

6.57.2.1 PPx::CGContextSaver::CGContextSaver (CGContextRef *inContext*)

Constructor from a CGContextRef.

Parameters:

inContext CGContext to save

Definition at line 156 of file PPxViewUtils.cp.

References Save().

6.57.2.2 PPx::CGContextSaver::∼CGContextSaver ()

Destructor.

Restores saved CGContext

Definition at line 169 of file PPxViewUtils.cp.

References Restore().

6.57.3 Member Function Documentation

6.57.3.1 CGContextRef PPx::CGContextSaver::Get () const

Returns the CGContext.

Returns:
Returns the CGContext

Definition at line 183 of file PPxViewUtils.cp.

6.57.3.2 void PPx::CGContextSaver::Save (CGContextRef *inContext*)

Saves the input CGContext and restores the one formerly being saved.

Parameters:
inContext CGContext to save

Definition at line 197 of file PPxViewUtils.cp.

References Restore().

Referenced by CGContextSaver().

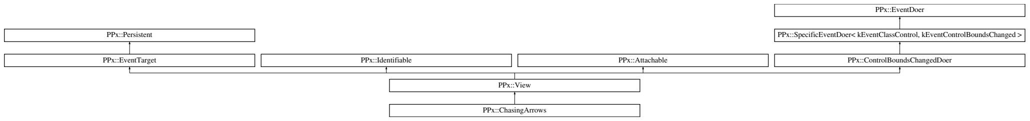
The documentation for this class was generated from the following files:

- PPxViewUtils.h
- PPxViewUtils.cp

6.58 PPx::ChasingArrows Class Reference

`#include <PPxChasingArrows.h>`

Inheritance diagram for PPx::ChasingArrows::



6.58.1 Detailed Description

A system chasing arrows activity indicator.

Definition at line 22 of file PPxChasingArrows.h.

Public Member Functions

- ChasingArrows ()

  Default constructor.

- virtual ∼ChasingArrows ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled)

  Initialize from chasing arrows creation parameters.

- void SetAnimating (bool inIsAnimating)

  Sets the option for animating the chasing arrows.

- bool IsAnimating () const

  Returns whether the chasing arrows are animating.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

6.58.2 Member Function Documentation

6.58.2.1 void PPx::ChasingArrows::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*)

Initialize from chasing arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

Definition at line 42 of file PPxChasingArrows.cp.

6.58.2.2 void PPx::ChasingArrows::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 77 of file PPxChasingArrows.cp.

6.58.2.3 bool PPx::ChasingArrows::IsAnimating () const

Returns whether the chasing arrows are animating.

Returns:

Whether the chasing arrows are animating

Definition at line 113 of file PPxChasingArrows.cp.

References PPx::View::GetDataTag().

6.58.2.4 void PPx::ChasingArrows::SetAnimating (bool *inIsAnimating*)

Sets the option for animating the chasing arrows.

Parameters:

inIsAnimating Whether the arrows should be animating

Definition at line 95 of file PPxChasingArrows.cp.

References PPx::View::SetDataTag().

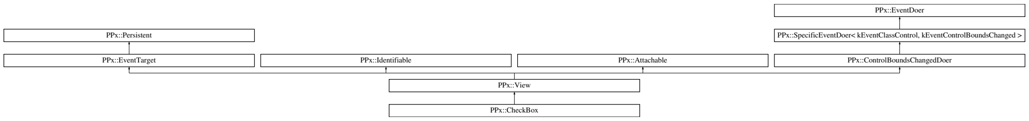
The documentation for this class was generated from the following files:

- PPxChasingArrows.h
- PPxChasingArrows.cp

6.59 PPx::CheckBox Class Reference

`#include <PPxCheckBox.h>`

Inheritance diagram for PPx::CheckBox::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::CheckBox

6.59.1 Detailed Description

A system check box control.

Definition at line 22 of file PPxCheckBox.h.

Public Member Functions

- CheckBox ()

  Default constructor.

- virtual ~CheckBox ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, SInt32 inInitialValue, bool inAutoToggle)

  Initializes from check box creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.59.2 Member Function Documentation

6.59.2.1 void PPx::CheckBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inAutoToggle*)

Initializes from check box creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title of check box

inInitialValue Initial value for check box

inAutoToggle Whether check box toggles automatically when clicked

Definition at line 47 of file PPxCheckBox.cp.

6.59.2.2 void PPx::CheckBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 88 of file PPxCheckBox.cp.

References PPx::DataReader::ReadOptional().

6.59.2.3 void PPx::CheckBox::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 113 of file PPxCheckBox.cp.

References PPx::View::GetTitle(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

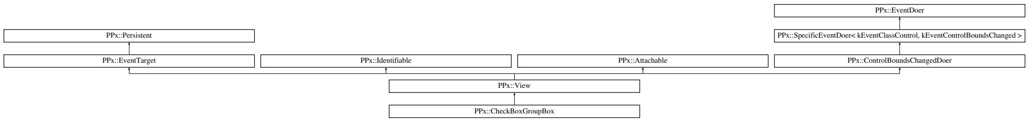
The documentation for this class was generated from the following files:

- PPxCheckBox.h
- PPxCheckBox.cp

6.60 PPx::CheckBoxGroupBox Class Reference

```
#include <PPxCheckBoxGroupBox.h>
```

Inheritance diagram for PPx::CheckBoxGroupBox::



6.60.1 Detailed Description

A system group box with a check box title.

Definition at line 22 of file PPxCheckBoxGroupBox.h.

Public Member Functions

- CheckBoxGroupBox ()

  Default constructor.

- virtual ∼CheckBoxGroupBox ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, SInt32 inInitialValue, bool inIsPrimary, bool inAutoToggle)

  Initialize from check box group box creation parameters.

- void GetTitleRect (Rect &outTitleRect) const

  Passes back the title rectangle for the check box group box.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.60.2 Member Function Documentation

6.60.2.1 void PPx::CheckBoxGroupBox::GetTitleRect (Rect & *outTitleRect*) const

Passes back the title rectangle for the check box group box.

Parameters:

outTitleRect Title rectangle

Definition at line 144 of file PPxCheckBoxGroupBox.cp.

References PPx::View::GetDataTag().

6.60.2.2 void PPx::CheckBoxGroupBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inIsPrimary*, bool *inAutoToggle*)

Initialize from check box group box creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title of check box

inInitialValue State of check box (0 = unchecked, 1 = checked, 2 = mixed)

inIsPrimary Group box kind (true = primary, false = secondary)

inAutoToggle Whether box is checked/unchecked automatically when clicked

Definition at line 50 of file PPxCheckBoxGroupBox.cp.

6.60.2.3 void PPx::CheckBoxGroupBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 94 of file PPxCheckBoxGroupBox.cp.

References PPx::DataReader::ReadOptional().

6.60.2.4 void PPx::CheckBoxGroupBox::WriteState (DataWriter & *ioWriter*) const [protected, virtual]

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 121 of file PPxCheckBoxGroupBox.cp.

References PPx::View::GetTitle(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

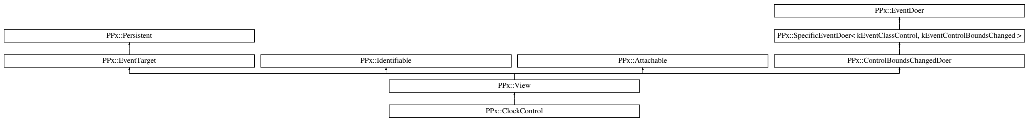
The documentation for this class was generated from the following files:

- PPxCheckBoxGroupBox.h
- PPxCheckBoxGroupBox.cp

6.61 PPx::ClockControl Class Reference

```
#include <PPxClockControl.h>
```

Inheritance diagram for PPx::ClockControl::



6.61.1 Detailed Description

A system clock control.

Definition at line 24 of file PPxClockControl.h.

Public Member Functions

- ClockControl ()

  Default constructor.

- virtual ∼ClockControl ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, ControlClockType inClockType, ControlClockFlags inClockFlags)

  Initialize from clock creation parameters.

- void SetLongDate (const LongDateRec &inLongDate)

  Sets the long date for the clock.

- void GetLongDate (LongDateRec &outLongDate)

  Passes back long date stored in the clock.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID for the clock.

- void SetAnimating (bool inIsAnimating)

  Sets the option for animating the clock.

- bool IsAnimating () const

  Returns whether the clock is animating.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.61.2 Member Function Documentation

6.61.2.1 void PPx::ClockControl::GetLongDate (LongDateRec & *outLongDate*)

Passes back long date stored in the clock.

Parameters:

outLongDate Long date from clock

Definition at line 156 of file PPxClockControl.cp.

References PPx::View::GetDataTag().

6.61.2.2 void PPx::ClockControl::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, ControlClockType *inClockType*, ControlClockFlags *inClockFlags*)

Initialize from clock creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inClockType Kind of clock (time or date)

inClockFlags Clock options

Definition at line 57 of file PPxClockControl.cp.

6.61.2.3 void PPx::ClockControl::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 98 of file PPxClockControl.cp.

References PPx::DataReader::ReadOptional().

6.61.2.4 bool PPx::ClockControl::IsAnimating () const

Returns whether the clock is animating.

Returns:

Whether the clock is animating

Definition at line 205 of file PPxClockControl.cp.

References PPx::View::GetDataTag().

6.61.2.5 void PPx::ClockControl::SetAnimating (bool *inIsAnimating*)

Sets the option for animating the clock.

Parameters:

inIsAnimating Whether the clock should be animating

Definition at line 187 of file PPxClockControl.cp.

References PPx::View::SetDataTag().

6.61.2.6 void PPx::ClockControl::SetLongDate (const LongDateRec & *inLongDate*)

Sets the long date for the clock.

Parameters:

inLongDate Long data to display in clock

Definition at line 140 of file PPxClockControl.cp.

References PPx::View::SetDataTag().

6.61.2.7 void PPx::ClockControl::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID for the clock.

Parameters:

inFont Theme font ID to use for text

Definition at line 172 of file PPxClockControl.cp.

6.61.2.8 void PPx::ClockControl::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 120 of file PPxClockControl.cp.

References PPx::DataWriter::WriteValue().

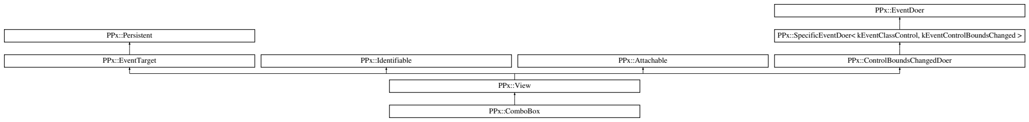
The documentation for this class was generated from the following files:

- PPxClockControl.h
- PPxClockControl.cp

6.62 PPx::ComboBox Class Reference

`#include <PPxComboBox.h>`

Inheritance diagram for PPx::ComboBox::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::ComboBox

6.62.1 Detailed Description

A system combo box control.

Definition at line 22 of file PPxComboBox.h.

Public Member Functions

- ComboBox ()

  Default constructor.

- virtual ∼ComboBox ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inDefaultText, const ControlFontStyleRec ∗inStyle, CFArrayRef inValueList, OptionBits inAttributes)

  Initialize from chasing arrows creation parameters.

- void SetText (CFStringRef inText)

  Sets the text in the edit field of the combo box.

- CFString GetText () const

  Returns the text from the edit field of the combo box.

- SInt32 GetListItemsCount () const

  Returns the number of items in the list of the combo box.

- void InsertListItemAt (CFIndex inIndex, CFStringRef inItemText)

  Inserts an item into the list of the combo box.

- void AppendListItem (CFStringRef inItemText, CFIndex &outIndex)

  Appends an item to the list of the combo box.

- void RemoveListItem (CFIndex inIndex)

  Removes an item from the list of the combo box.

- CFString GetListItemText (CFIndex inIndex) const

  Returns the text of an item in the list of the combo box.

- void ChangeAttributes (OptionBits inAttributesToSet, OptionBits inAttributesToClear)

  Changes the attributes of the combo box.

- OptionBits GetAttributes () const

  Returns the attributes of the comno box.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.62.2 Member Function Documentation

6.62.2.1 void PPx::ComboBox::AppendListItem (CFStringRef *inItemText*, CFIndex & *outIndex*)

Appends an item to the list of the combo box.

Parameters:

inItemText Text of item to append

outIndex Index at which item was appended

Definition at line 225 of file PPxComboBox.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

6.62.2.2 void PPx::ComboBox::ChangeAttributes (OptionBits *inAttributesToSet*, OptionBits *inAttributesToClear*)

Changes the attributes of the combo box.

Parameters:

inAttributesToSet Bit mask of attributes to set

inAttributesToClear Bit mask of attributes to clear

Definition at line 283 of file PPxComboBox.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

6.62.2.3 OptionBits PPx::ComboBox::GetAttributes () const

Returns the attributes of the comno box.

Returns:

Attributes of the comno box

Definition at line 303 of file PPxComboBox.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

Referenced by WriteState().

6.62.2.4 SInt32 PPx::ComboBox::GetListItemsCount () const

Returns the number of items in the list of the combo box.

Returns:

Number of items in the list of the combo box

Definition at line 190 of file PPxComboBox.cp.

References PPx::View::GetSysView().

6.62.2.5 CFString PPx::ComboBox::GetListItemText (CFIndex *inIndex*) const

Returns the text of an item in the list of the combo box.

Parameters:

inIndex Index of item whose text to get

Definition at line 261 of file PPxComboBox.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

6.62.2.6 CFString PPx::ComboBox::GetText () const

Returns the text from the edit field of the combo box.

Returns:

Text from the edit field of the combo box

Definition at line 168 of file PPxComboBox.cp.

References PPx::View::GetDataTag(), and PPx_ThrowIfOSError_.

Referenced by WriteState().

6.62.2.7 void PPx::ComboBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inDefaultText*, const ControlFontStyleRec ∗ *inStyle*, CFArrayRef *inValueList*, OptionBits *inAttributes*)

Initialize from chasing arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inDefaultText Initial text in the edit field

inStyle Text style

inValueList List of value to diplay as choices

inAttributes Option flags

Definition at line 55 of file PPxComboBox.cp.

6.62.2.8 void PPx::ComboBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 96 of file PPxComboBox.cp.

References PPx::DataReader::ReadOptional().

6.62.2.9 void PPx::ComboBox::InsertListItemAt (CFIndex *inIndex*, CFStringRef *inItemText*)

Inserts an item into the list of the combo box.

Parameters:

inIndex Index in list at which to insert item

inItemText Text of item to insert

Definition at line 205 of file PPxComboBox.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

6.62.2.10 void PPx::ComboBox::RemoveListItem (CFIndex *inIndex*)

Removes an item from the list of the combo box.

Parameters:

inIndex Index of item to remove

Definition at line 244 of file PPxComboBox.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

6.62.2.11 void PPx::ComboBox::SetText (CFStringRef *inText*)

Sets the text in the edit field of the combo box.

Parameters:

inText Text to put in edit field

Definition at line 152 of file PPxComboBox.cp.

References PPx::View::SetDataTag().

6.62.2.12 void PPx::ComboBox::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 128 of file PPxComboBox.cp.

References GetAttributes(), GetText(), and PPx::DataWriter::WriteValue().

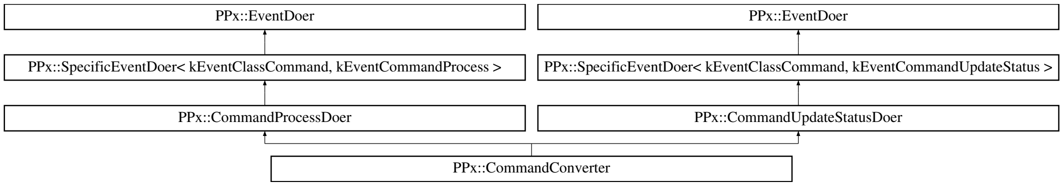
The documentation for this class was generated from the following files:

- PPxComboBox.h
- PPxComboBox.cp

6.63 PPx::CommandConverter Class Reference

`#include <PPxCommandEvents.h>`

Inheritance diagram for PPx::CommandConverter::


PPx::EventDoer
PPx::EventDoer
PPx::SpecificEventDoer< kEventClassCommand, kEventCommandProcess >
PPx::SpecificEventDoer< kEventClassCommand, kEventCommandUpdateStatus >
PPx::CommandProcessDoer
PPx::CommandUpdateStatusDoer
PPx::CommandConverter

6.63.1 Detailed Description

Handles processing and updating command events by converting them into events for specific commands.

Definition at line 63 of file PPxCommandEvents.h.

Public Member Functions

- void Install (EventTargetRef inTarget)

  Installs handlers for command and update command status events.

Protected Member Functions

- virtual OSStatus DoCommandProcess (SysCarbonEvent &ioEvent, HICommand inCommand, UInt32 inKeyModifiers, UInt32 inMenuContext)

  Handles a command process CarbonEvent by relaying it as an event for a specific command.

- virtual OSStatus DoCommandUpdateStatus (SysCarbonEvent &ioEvent, HICommand inCommand, UInt32 inMenuContext)

  Handles a command update status CarbonEvent by relaying it as an event for a specific command.

6.63.2 Member Function Documentation

6.63.2.1 void PPx::CommandConverter::Install (EventTargetRef *inTarget*)

Installs handlers for command and update command status events.

Parameters:

inTarget Event target for which to install handlers

Reimplemented from PPx::SpecificEventDoer< kEventClassCommand, kEventCommandUpdateStatus >.

Definition at line 58 of file PPxCommandEvents.cp.

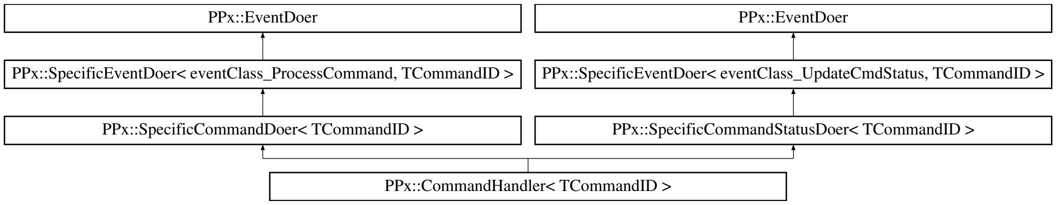
The documentation for this class was generated from the following files:

- PPxCommandEvents.h
- PPxCommandEvents.cp

6.64 PPx::CommandHandler< TCommandID > Class Template Reference

```
#include <PPxCommandEvents.h>
```

Inheritance diagram for PPx::CommandHandler< TCommandID >::


PPx::EventDoer
PPx::EventDoer
PPx::SpecificEventDoer< eventClass_ProcessCommand, TCommandID >
PPx::SpecificEventDoer< eventClass_UpdateCmdStatus, TCommandID >
PPx::SpecificCommandDoer< TCommandID >
PPx::SpecificCommandStatusDoer< TCommandID >
PPx::CommandHandler< TCommandID >

6.64.1 Detailed Description

template<UInt32 TCommandID> class PPx::CommandHandler< TCommandID >

Handles processing and updating the status of a specific command.

Definition at line 157 of file PPxCommandEvents.h.

Public Member Functions

- void **Install** (EventTargetRef inTarget)

The documentation for this class was generated from the following file:

- PPxCommandEvents.h

6.65 PPx::CommandIDType< TCommandID > Struct Template Reference

```
#include <PPxCommandEvents.h>
```

6.65.1 Detailed Description

template<UInt32 TCommandID> struct PPx::CommandIDType< TCommandID >

Template which creates a unique type for a literal command ID value.

Definition at line 86 of file PPxCommandEvents.h.

Public Types

- enum { **value** = TCommandID }

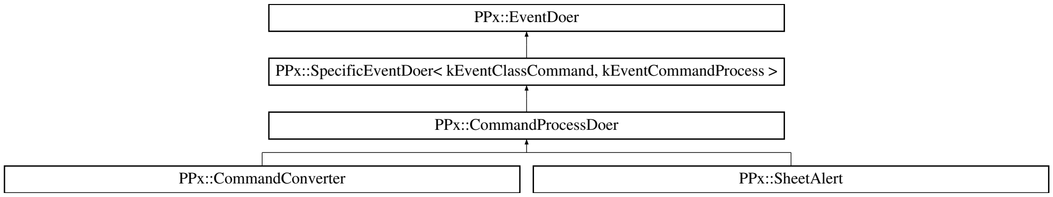
The documentation for this struct was generated from the following file:

- PPxCommandEvents.h

6.66 PPx::CommandProcessDoer Class Reference

```
#include <PPxCommandEvents.h>
```

Inheritance diagram for PPx::CommandProcessDoer::



6.66.1 Detailed Description

Handles HICommands.

Definition at line 23 of file PPxCommandEvents.h.

Protected Member Functions

- virtual OSStatus **DoCommandProcess** (SysCarbonEvent &ioEvent, HICommand inCommand, UInt32 inKeyModifiers, UInt32 inMenuContext)=0

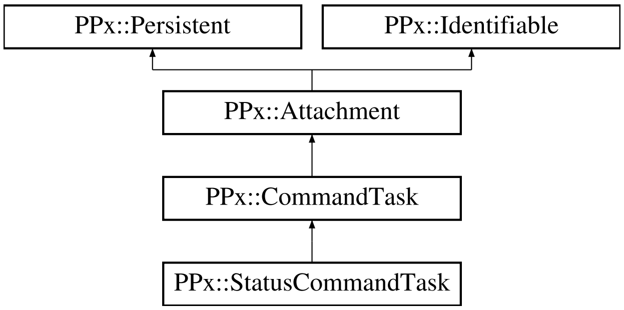
The documentation for this class was generated from the following files:

- PPxCommandEvents.h
- PPxCommandEvents.cp

6.67 PPx::CommandTask Class Reference

`#include <PPxCommandTask.h>`

Inheritance diagram for PPx::CommandTask::



6.67.1 Detailed Description

Abstract class for an Attachment which handles a command event.

Definition at line 24 of file PPxCommandTask.h.

Public Member Functions

- CommandTask ()

  Default constructor.

- void Initialize (EventTarget ∗inTarget, CommandIDT inCommandID)

  Specifies target and command ID.

- OSStatus **DoCommandProcessEvent** (SysCarbonEvent &ioEvent)

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus **DoCommandProcess** (HICommand inCommand, UInt32 inKeyModifiers, UInt32 inMenuContext)=0

6.67.2 Member Function Documentation

6.67.2.1 void PPx::CommandTask::Initialize (EventTarget ∗ *inTarget*, CommandIDT *inCommandID*)

Specifies target and command ID.

Parameters:

inTarget Target which receives command events

inCommandID Command to handle

Reimplemented in PPx::StatusCommandTask.

Definition at line 41 of file PPxCommandTask.cp.

References PPx::eventClass_ProcessCommand, and PPx::EventDoerCallback< CommandTask >::Install().

Referenced by InitState().

6.67.2.2 void PPx::CommandTask::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Attachment.

Definition at line 67 of file PPxCommandTask.cp.

References Initialize(), PPx::DataReader::ReadObjectValue(), and PPx::DataReader::ReadOptional().

6.67.2.3 void PPx::CommandTask::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Attachment.

Definition at line 90 of file PPxCommandTask.cp.

References PPx::DataWriter::WriteObjectValue(), and PPx::DataWriter::WriteValue().

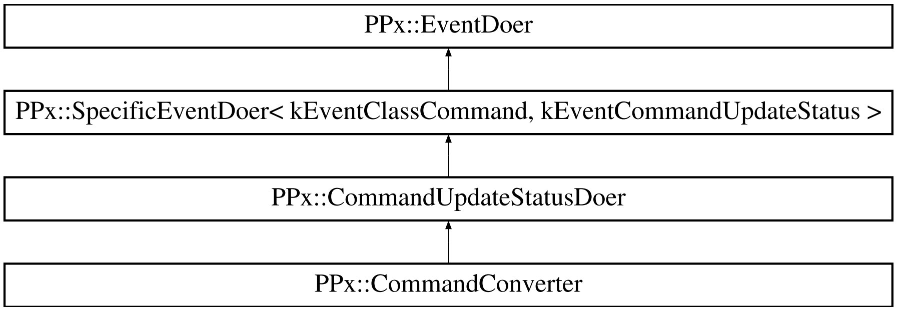
The documentation for this class was generated from the following files:

- PPxCommandTask.h
- PPxCommandTask.cp

6.68 PPx::CommandUpdateStatusDoer Class Reference

```
#include <PPxCommandEvents.h>
```

Inheritance diagram for PPx::CommandUpdateStatusDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassCommand, kEventCommandUpdateStatus >
PPx::CommandUpdateStatusDoer
PPx::CommandConverter

6.68.1 Detailed Description

Handles updating the status of items that invoke commands.

Definition at line 43 of file PPxCommandEvents.h.

Protected Member Functions

- virtual OSStatus **DoCommandUpdateStatus** (SysCarbonEvent &ioEvent, HICommand inCommand, UInt32 inMenuContext)=0

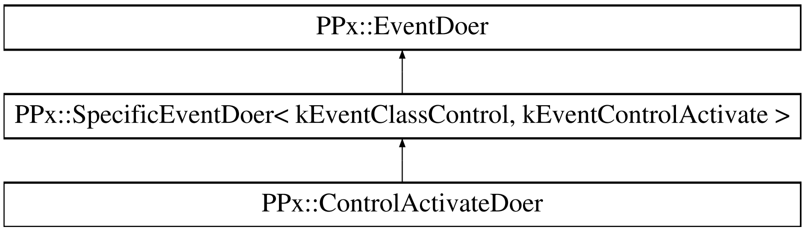
The documentation for this class was generated from the following files:

- PPxCommandEvents.h
- PPxCommandEvents.cp

6.69 PPx::ControlActivateDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlActivateDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlActivate >
PPx::ControlActivateDoer

6.69.1 Detailed Description

Handles a control becoming active.

Definition at line 184 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlActivate** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

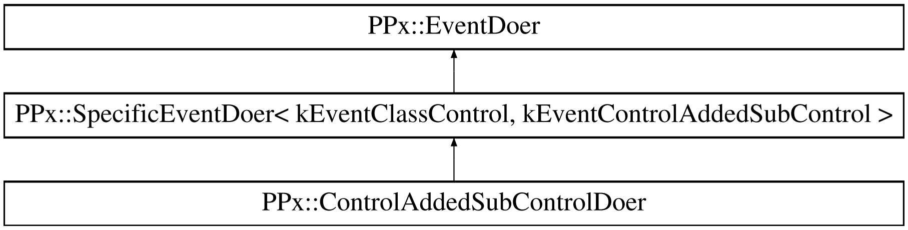
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.70 PPx::ControlAddedSubControlDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlAddedSubControlDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlAddedSubControl >
PPx::ControlAddedSubControlDoer

6.70.1 Detailed Description

Handles notification when a subcontrol is added.

Definition at line 514 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlAddedSubControl** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlRef inSubControl)=0

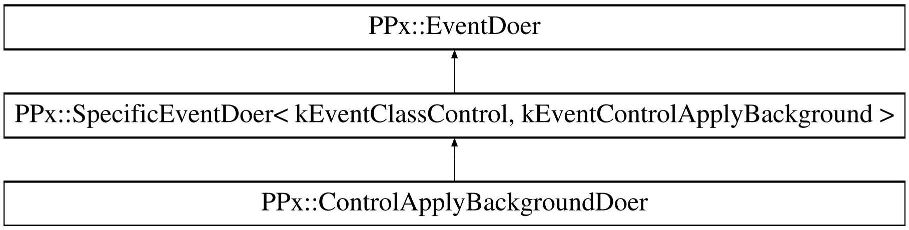
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.71 PPx::ControlApplyBackgroundDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlApplyBackgroundDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlApplyBackground >
PPx::ControlApplyBackgroundDoer

6.71.1 Detailed Description

Handles applying a control's background to a port.

Definition at line 110 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlApplyBackground** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlRef inSubControl, SInt16 inDrawDepth, bool inDrawInColor)=0

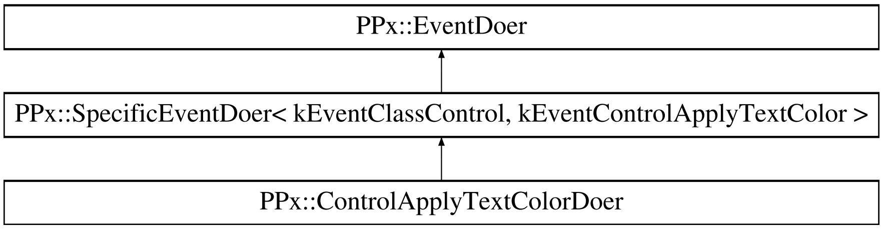
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.72 PPx::ControlApplyTextColorDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlApplyTextColorDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlApplyTextColor >
PPx::ControlApplyTextColorDoer

6.72.1 Detailed Description

Handles applying a control's text color to a port/context.

Definition at line 129 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlApplyTextColor** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlRef inSubControl, SInt16 inDrawDepth, bool inDrawInColor, CGContextRef inContext)=0

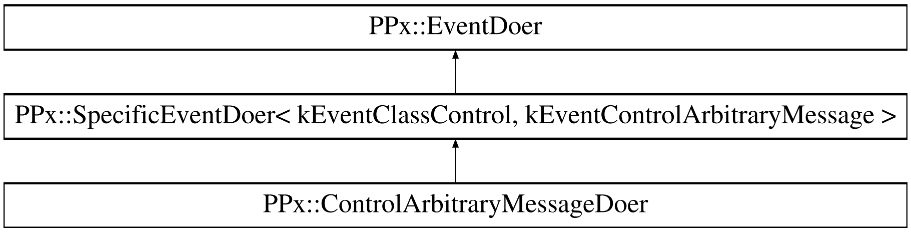
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.73 PPx::ControlArbitraryMessageDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlArbitraryMessageDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlArbitraryMessage >
PPx::ControlArbitraryMessageDoer

6.73.1 Detailed Description

Handles old-style CDEF messages.

Definition at line 548 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlArbitraryMessage** (SysCarbonEvent &ioEvent, ControlRef inControl, SInt16 inMessage, SInt32 inParameter, SInt32 &outResult)=0

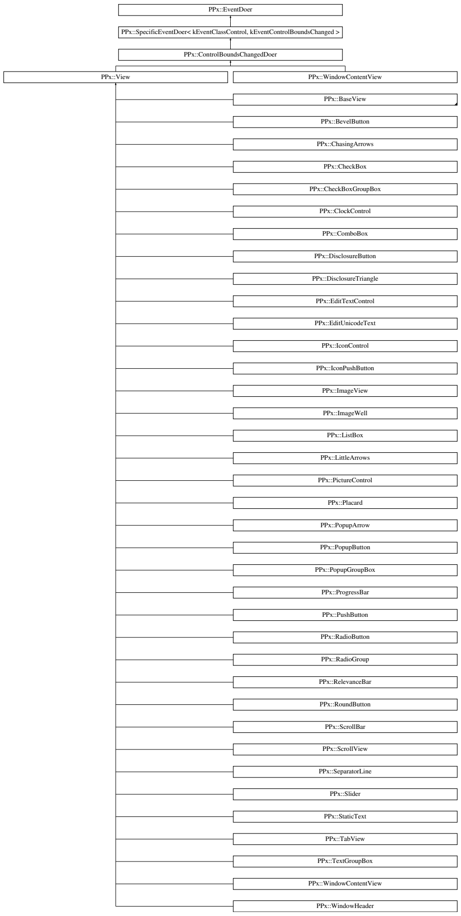
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.74 PPx::ControlBoundsChangedDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlBoundsChangedDoer::



6.74.1 Detailed Description

Handles adapting to a change in the bounds of a control.

Definition at line 444 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlBoundsChanged** (SysCarbonEvent &ioEvent, ControlRef inControl, UInt32 inChangeAttributes, const HIRect &inOriginalBounds, const HIRect &inCurrentBounds)=0

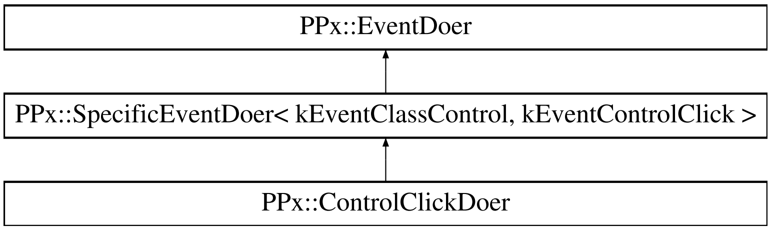
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.75 PPx::ControlClickDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlClickDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlClick >
PPx::ControlClickDoer

6.75.1 Detailed Description

Handles a mouse down event inside a control.

Definition at line 234 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlClick** (SysCarbonEvent &ioEvent, ControlRef inControl, const HIPoint &inMouseLocation)=0

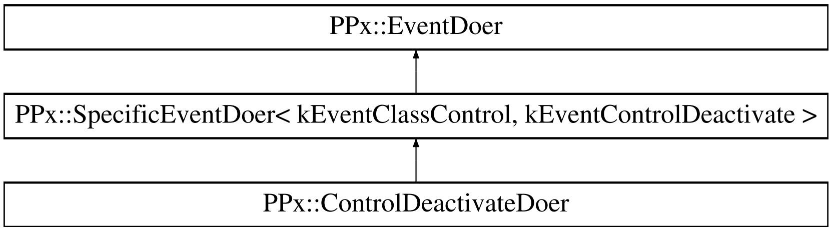
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.76 PPx::ControlDeactivateDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlDeactivateDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlDeactivate >
PPx::ControlDeactivateDoer

6.76.1 Detailed Description

Handles a control becoming inactive.

Definition at line 200 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDeactivate** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

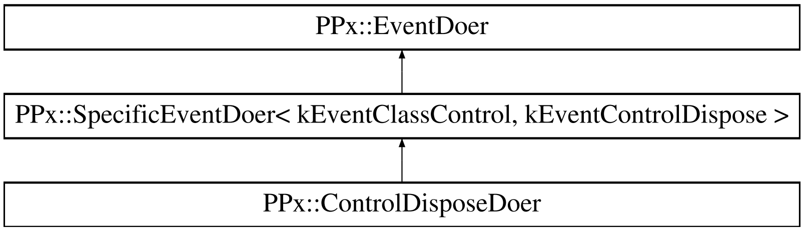
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.77 PPx::ControlDisposeDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlDisposeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlDispose >
PPx::ControlDisposeDoer

6.77.1 Detailed Description

Handles a control being disposed.

Definition at line 20 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDispose** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

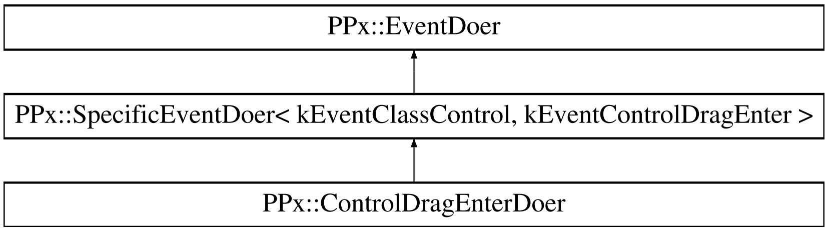
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.78 PPx::ControlDragEnterDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlDragEnterDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlDragEnter >
PPx::ControlDragEnterDoer

6.78.1 Detailed Description

Handles a drag entering a control.

Definition at line 271 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDragEnter** (SysCarbonEvent &ioEvent, ControlRef inControl, DragRef inDragRef)=0

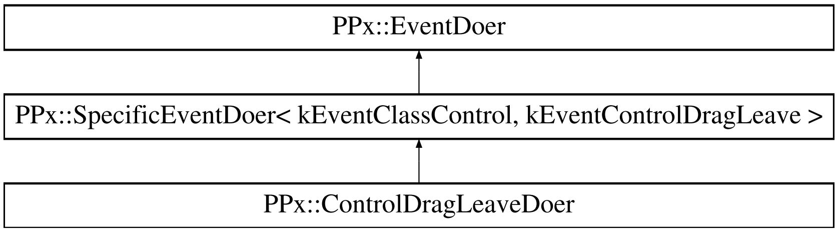
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.79 PPx::ControlDragLeaveDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlDragLeaveDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlDragLeave >
PPx::ControlDragLeaveDoer

6.79.1 Detailed Description

Handles a drag leaving a control.

Definition at line 305 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDragLeave** (SysCarbonEvent &ioEvent, ControlRef inControl, DragRef inDragRef)=0

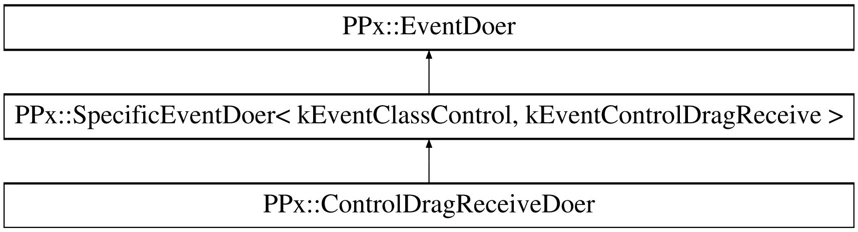
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.80 PPx::ControlDragReceiveDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlDragReceiveDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlDragReceive >
PPx::ControlDragReceiveDoer

6.80.1 Detailed Description

Handles a drag being dropped in a control.

Definition at line 322 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDragReceive** (SysCarbonEvent &ioEvent, ControlRef inControl, DragRef inDragRef)=0

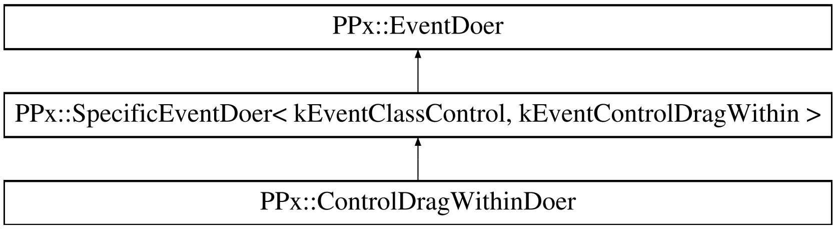
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.81 PPx::ControlDragWithinDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlDragWithinDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlDragWithin >
PPx::ControlDragWithinDoer

6.81.1 Detailed Description

Handles a drag remaining inside a control.

Definition at line 288 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDragWithin** (SysCarbonEvent &ioEvent, ControlRef inControl, DragRef inDragRef)=0

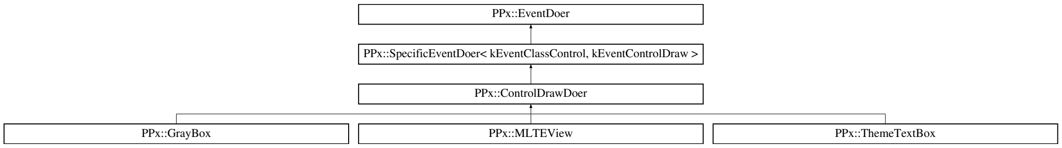
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.82 PPx::ControlDrawDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlDrawDoer::



6.82.1 Detailed Description

Handles drawing a control.

Definition at line 91 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlDraw** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, RgnHandle inClipRgn, CGContextRef inContext)=0

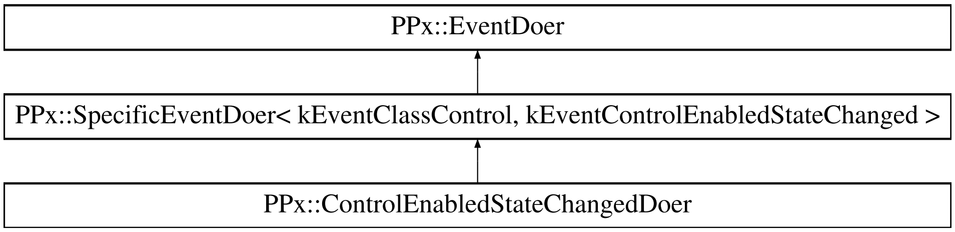
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.83 PPx::ControlEnabledStateChangedDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlEnabledStateChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlEnabledStateChanged >
PPx::ControlEnabledStateChangedDoer

6.83.1 Detailed Description

Handles notification when a control is enabled or disabled.

Definition at line 479 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlEnbaledStateChanged** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

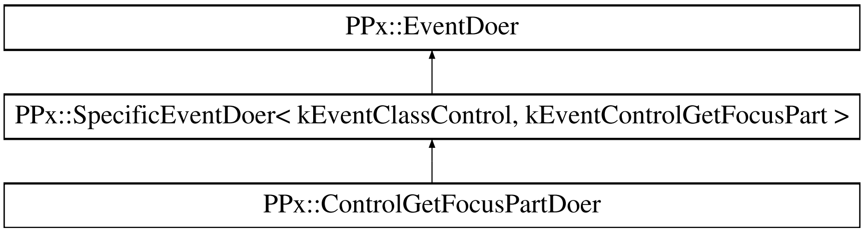
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.84 PPx::ControlGetFocusPartDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlGetFocusPartDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlGetFocusPart >
PPx::ControlGetFocusPartDoer

6.84.1 Detailed Description

Returns the currently focused part of a control.

Definition at line 167 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlGetFocusPart** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode &outFocusPart)=0

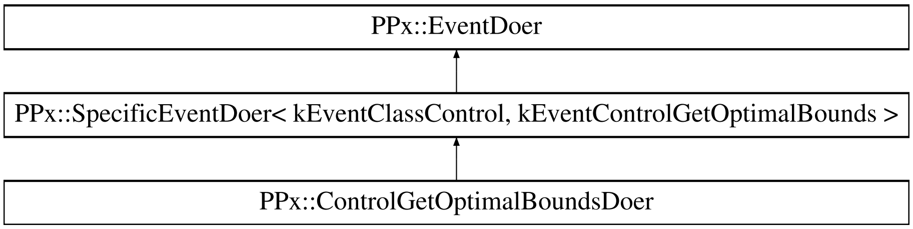
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.85 PPx::ControlGetOptimalBoundsDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlGetOptimalBoundsDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlGetOptimalBounds >
PPx::ControlGetOptimalBoundsDoer

6.85.1 Detailed Description

Returns the optimal bounds for a control.

Definition at line 375 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlGetOptimalBounds** (SysCarbonEvent &ioEvent, ControlRef inControl, Rect &outBounds, SInt16 &outBaseline)=0

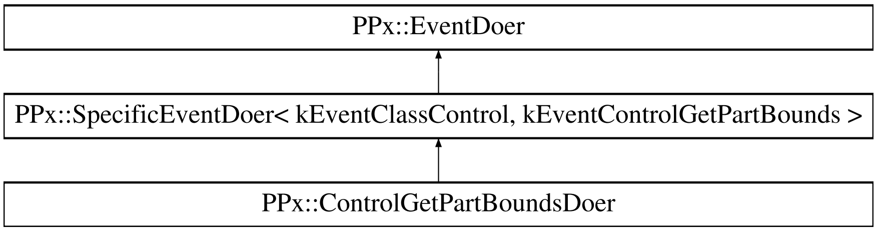
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.86 PPx::ControlGetPartBoundsDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlGetPartBoundsDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlGetPartBounds >
PPx::ControlGetPartBoundsDoer

6.86.1 Detailed Description

Returns the bounding rectangle of a control part.

Definition at line 357 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlGetPartBounds** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, Rect &outBounds)=0

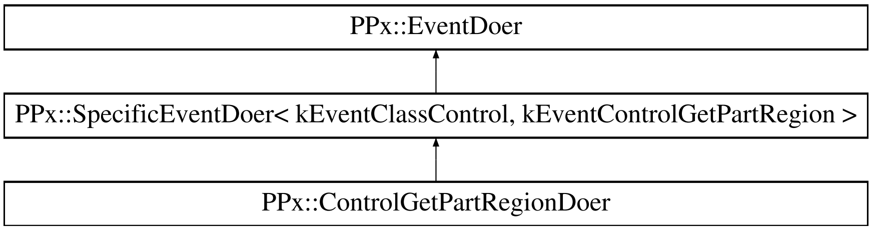
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.87 PPx::ControlGetPartRegionDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlGetPartRegionDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlGetPartRegion >
PPx::ControlGetPartRegionDoer

6.87.1 Detailed Description

Returns the bounding region of a control part.

Definition at line 339 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlGetPartRegion** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, RgnHandle inRegion)=0

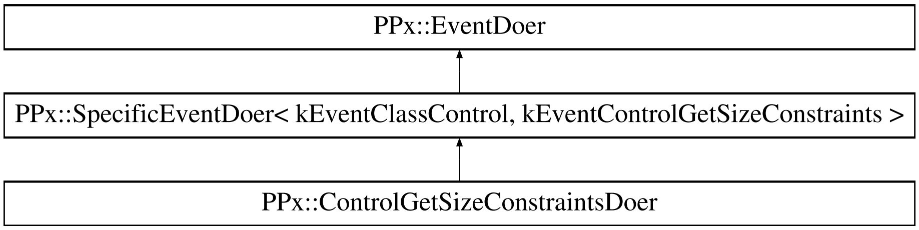
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.88 PPx::ControlGetSizeConstraintsDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlGetSizeConstraintsDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlGetSizeConstraints >
PPx::ControlGetSizeConstraintsDoer

6.88.1 Detailed Description

Returns the minimum and maximum sizes for a control.

Definition at line 393 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlGetSizeConstraints** (SysCarbonEvent &ioEvent, ControlRef inControl, HISize &outMinSize, HISize &outMaxSize)=0

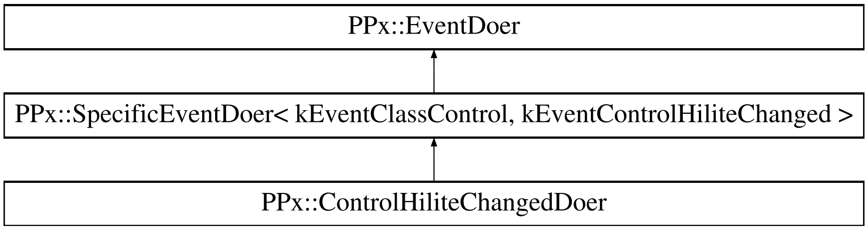
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.89 PPx::ControlHiliteChangedDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlHiliteChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlHiliteChanged >
PPx::ControlHiliteChangedDoer

6.89.1 Detailed Description

Handles notification when the hilite state of a control changes.

Definition at line 428 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlHiliteChanged** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

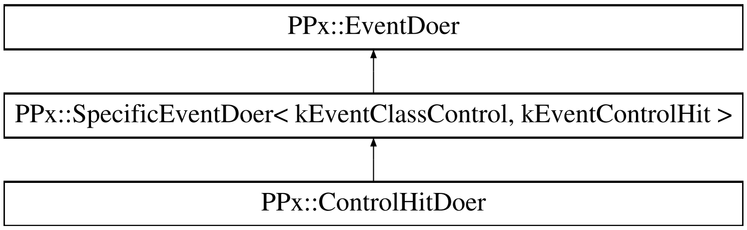
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.90 PPx::ControlHitDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlHitDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlHit >
PPx::ControlHitDoer

6.90.1 Detailed Description

Handles a click in a control.

A click occurs when a mouse down and subsequent mouse up are within the same part of a control.

Definition at line 37 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlHit** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, UInt32 inKeyModifiers)=0

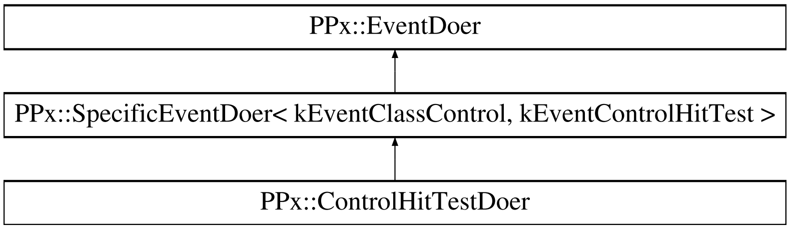
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.91 PPx::ControlHitTestDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlHitTestDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlHitTest >
PPx::ControlHitTestDoer

6.91.1 Detailed Description

Handles testing whether a point is within a control.

Definition at line 73 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlHitTest** (SysCarbonEvent &ioEvent, ControlRef inControl, const HIPoint &inHitPoint, ControlPartCode &outPartCode)=0

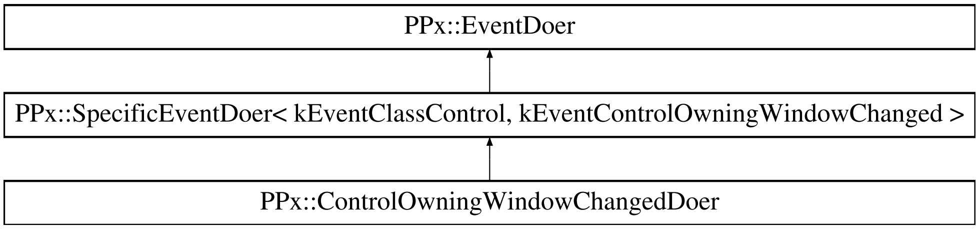
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.92 PPx::ControlOwningWindowChangedDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlOwningWindowChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlOwningWindowChanged >
PPx::ControlOwningWindowChangedDoer

6.92.1 Detailed Description

Handles notification when a control moves into a different window.

Definition at line 495 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlOwningWindowChanged** (SysCarbonEvent &ioEvent, ControlRef inControl, UInt32 inAttributes, WindowRef inFormerWindow, WindowRef inCurrentWindow)=0

The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.93 PPx::ControlPartCodeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.93.1 Detailed Description

Wrapper for ControlPartCode.

Definition at line 63 of file PPxSysTypes.h.

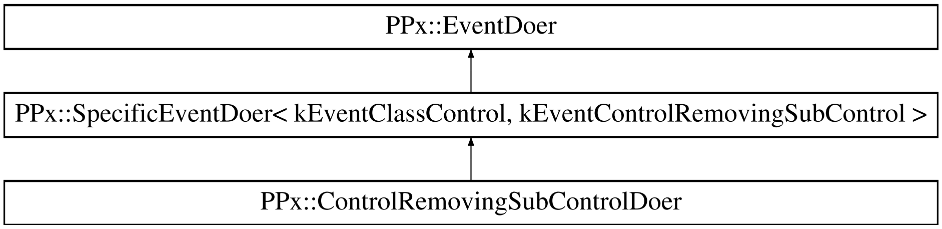
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.94 PPx::ControlRemovingSubControlDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlRemovingSubControlDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlRemovingSubControl >
PPx::ControlRemovingSubControlDoer

6.94.1 Detailed Description

Handles notification when a subcontrol is being removed.

Definition at line 531 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlRemovingSubControl** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlRef inSubControl)=0

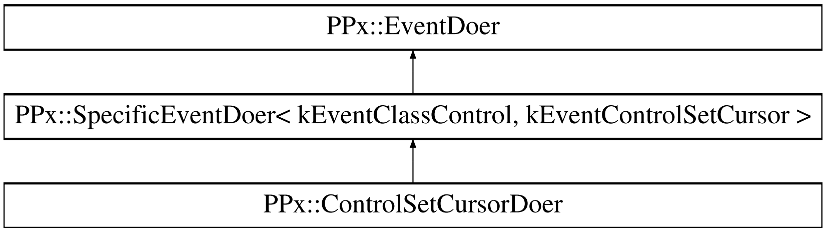
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.95 PPx::ControlSetCursorDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlSetCursorDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlSetCursor >
PPx::ControlSetCursorDoer

6.95.1 Detailed Description

Handles setting the cursor when the mouse is inside a control.

Definition at line 216 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlSetCursor** (SysCarbonEvent &ioEvent, ControlRef inControl, const HIPoint &inMouseLocation, UInt32 inKeyModifiers)=0

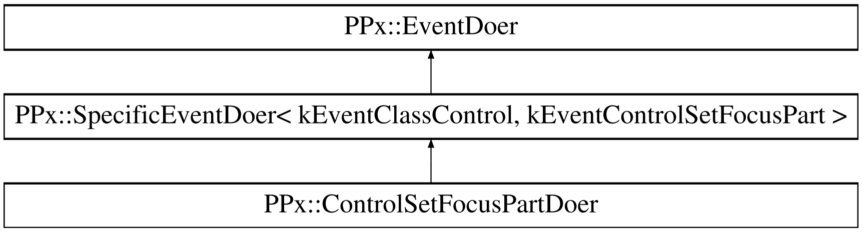
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.96 PPx::ControlSetFocusPartDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlSetFocusPartDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlSetFocusPart >
PPx::ControlSetFocusPartDoer

6.96.1 Detailed Description

Handles setting the focus to a part of a control.

Definition at line 149 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlSetFocusPart** (SysCarbonEvent &ioEvent, ControlRef inControl, bool inFocusEverything, ControlPartCode &ioFocusPart)=0

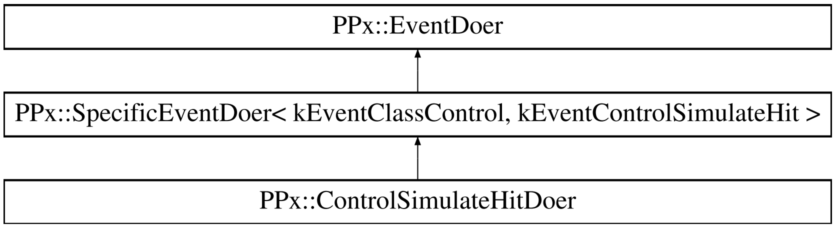
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.97 PPx::ControlSimulateHitDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlSimulateHitDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlSimulateHit >
PPx::ControlSimulateHitDoer

6.97.1 Detailed Description

Handles a simulating a click in a control.

Definition at line 55 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlSimulateHit** (SysCarbonEvent &ioEvent, ControlRef inControl, UInt32 inKeyModifiers, ControlPartCode &ioPartCode)=0

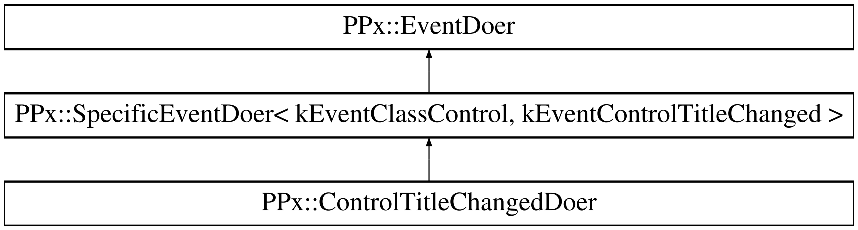
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.98 PPx::ControlTitleChangedDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlTitleChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlTitleChanged >
PPx::ControlTitleChangedDoer

6.98.1 Detailed Description

Handles notification when the title of a control changes.

Definition at line 463 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlTitleChanged** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

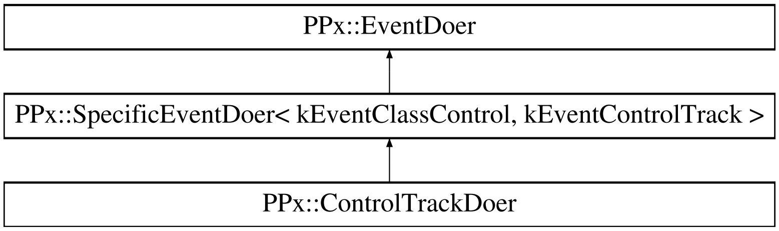
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.99 PPx::ControlTrackDoer Class Reference

```
#include <PPxViewEvents.h>
```

Inheritance diagram for PPx::ControlTrackDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlTrack >
PPx::ControlTrackDoer

6.99.1 Detailed Description

Handles mouse down tracking inside a control.

Definition at line 251 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlTrack** (SysCarbonEvent &ioEvent, ControlRef inControl, const HIPoint &inMouseLocation, UInt32 inKeyModifiers, ControlActionUPP inActionUPP, ControlPartCode &outPartCode)=0

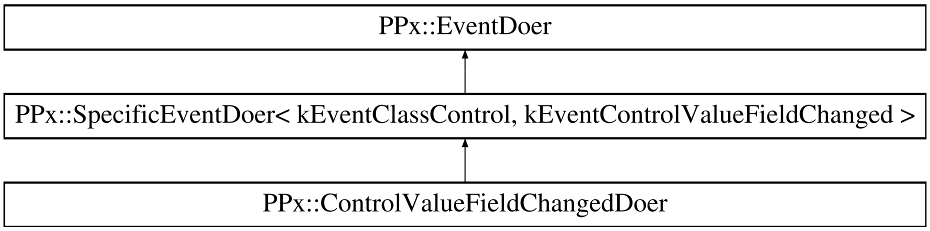
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.100 PPx::ControlValueFieldChangedDoer Class Reference

`#include <PPxViewEvents.h>`

Inheritance diagram for PPx::ControlValueFieldChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlValueFieldChanged >
PPx::ControlValueFieldChangedDoer

6.100.1 Detailed Description

Handles notification when the value, minimum value, maximum value, or view size of a control changes.

Definition at line 412 of file PPxViewEvents.h.

Protected Member Functions

- virtual OSStatus **DoControlValueFieldChanged** (SysCarbonEvent &ioEvent, ControlRef inControl)=0

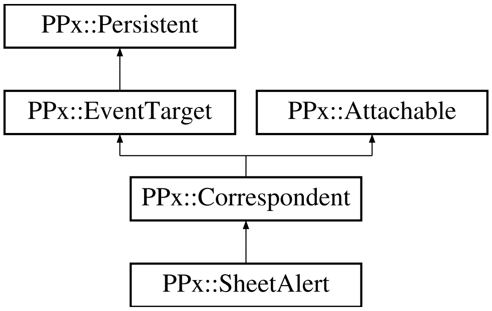
The documentation for this class was generated from the following files:

- PPxViewEvents.h
- PPxViewEvents.cp

6.101 PPx::Correspondent Class Reference

`#include <PPxCorrespondent.h>`

Inheritance diagram for PPx::Correspondent::



6.101.1 Detailed Description

A generic Event Target.

Install handlers for Carbon Events by adding Attachments or by creating subclasses that multiply inherit from Correspondent and EventDoer subclasses.

Definition at line 24 of file PPxCorrespondent.h.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.101.2 Member Function Documentation

6.101.2.1 void PPx::Correspondent::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::SheetAlert.

Definition at line 82 of file PPxCorrespondent.cp.

References PPx::Attachable::ReadAttachments().

6.101.2.2 void PPx::Correspondent::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::SheetAlert.

Definition at line 97 of file PPxCorrespondent.cp.

References PPx::Attachable::WriteAttachments().

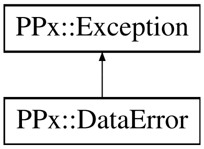
The documentation for this class was generated from the following files:

- PPxCorrespondent.h
- PPxCorrespondent.cp

6.102 PPx::DataError Class Reference

`#include <PPxExceptions.h>`

Inheritance diagram for PPx::DataError::



6.102.1 Detailed Description

Exception class for bad input data.

PPx programs are data-driven, using information stored in text files and Mac OS resources to describe user interface elements and program behavior. A DataError indicates a problem in interpreting that information. Such errors should be found during testing.

Definition at line 221 of file PPxExceptions.h.

Public Member Functions

- DataError (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Constructor.

Static Public Member Functions

- void Throw (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Throws a DataError exception.

6.102.2 Constructor & Destructor Documentation

6.102.2.1 PPx::DataError::DataError (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*)

Constructor.

Parameters:

inWhat Kind of data error

inWhy C string describing why the exception occurred

inWhere Source code location where exception was thrown

Note:

If PPx_Debug_Exceptions is false, the why and where are not stored.

Definition at line 405 of file PPxExceptions.cp.

6.102.3 Member Function Documentation

6.102.3.1 void PPx::DataError::Throw (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) [static]

Throws a DataError exception.

Parameters:

inWhat Kind of data error

inWhy C string description of why the exception was thrown

inWhere Source location where exception was throw

Definition at line 425 of file PPxExceptions.cp.

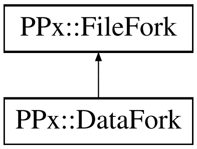
The documentation for this class was generated from the following files:

- PPxExceptions.h
- PPxExceptions.cp

6.103 PPx::DataFork Class Reference

`#include <PPxDataFork.h>`

Inheritance diagram for PPx::DataFork::



6.103.1 Detailed Description

Wrapper class for the data fork of a file.

Definition at line 23 of file PPxDataFork.h.

Public Member Functions

- DataFork (SInt16 inRefNum, bool inOwnsRefNum)

  Constructs a DataFork object for an already open data fork.

- DataFork (const FSRef &inFile, SInt8 inPermissions=fsRdWrPerm)

  Constructs a DataFork object for a file and opens the fork.

- CFData ReadContents ()

  Reads entire contents of a data fork into a CFData object.

- void WriteContents (const CFData &inData)

  Fills contents of a data fork with bytes from a CFData object.

- void WriteContents (const void ∗inBuffer, ByteCount inBufferSize)

  Fills contents of a data fork with bytes from a buffer.

- void ReadData (void ∗inBuffer, ByteCount inBufferSize, UInt16 inPositionMode=fsFromMark, SInt64 inOffset=0)

  Reads bytes from a data fork into a supplied buffer.

- void WriteData (const void ∗inBuffer, ByteCount inBufferSize, UInt16 inPositionMode=fsFromMark, SInt64 inOffset=0)

  Writes bytes from a buffer into a data fork.

Static Public Member Functions

- const HFSUniStr255 & GetForkName ()

  Returns the constant system name for the data fork.

6.103.2 Constructor & Destructor Documentation

6.103.2.1 PPx::DataFork::DataFork (SInt16 *inRefNum*, bool *inOwnsRefNum*)

Constructs a DataFork object for an already open data fork.

Parameters:

inRefNum Reference number for the data fork

inOwnsRefNum Whether this object should close the fork when finished

Definition at line 19 of file PPxDataFork.cp.

6.103.2.2 PPx::DataFork::DataFork (const FSRef & *inFile*, SInt8 *inPermissions* = fsRdWrPerm)

Constructs a DataFork object for a file and opens the fork.

Parameters:

inFile FSRef for the file

inPermissions Access permissions

Definition at line 36 of file PPxDataFork.cp.

6.103.3 Member Function Documentation

6.103.3.1 const HFSUniStr255 & PPx::DataFork::GetForkName () `[static]`

Returns the constant system name for the data fork.

Returns:

Name of the data fork

Definition at line 185 of file PPxDataFork.cp.

References PPx_ThrowIfOSError_.

6.103.3.2 CFData PPx::DataFork::ReadContents ()

Reads entire contents of a data fork into a CFData object.

Returns:

CFData object containing the fork contents

Note:

Since CFData uses signed 32-bit indexes, data length is limited to 2 Gigabytes
The returned CFData object has variable capacity, so you can add more data afterwards

Definition at line 59 of file PPxDataFork.cp.

References PPx::CFData::GetMutableBytePtr(), PPx::FileFork::GetSize(), PPx_-Throw_, ReadData(), and PPx::CFData::SetLength().

6.103.3.3 void PPx::DataFork::ReadData (void ∗ *inBuffer*, ByteCount *inBufferSize*, UInt16 *inPositionMode* = fsFromMark, SInt64 *inOffset* = 0)

Reads bytes from a data fork into a supplied buffer.

Parameters:

inBuffer Pointer to the buffer

inBufferSize Capacity of the buffer

inPositionMode Reference location in fork

inOffset Offset from reference location at which to start reading

Definition at line 131 of file PPxDataFork.cp.

References PPx_ThrowIfOSError_, and PPx::FileFork::UseRefNum().

Referenced by ReadContents().

6.103.3.4 void PPx::DataFork::WriteContents (const void ∗ *inBuffer*, ByteCount *inBufferSize*)

Fills contents of a data fork with bytes from a buffer.

Parameters:

inBuffer Pointer to the buffer

inBufferSize Number of bytes in the buffer

Any previous contents are overwritten and the fork size is set to the size of the buffer

Definition at line 110 of file PPxDataFork.cp.

References PPx::FileFork::SetSize(), and WriteData().

6.103.3.5 void PPx::DataFork::WriteContents (const CFData & *inData*)

Fills contents of a data fork with bytes from a CFData object.

Parameters:

inData The CFData object

Any previous contents are overwritten and the fork size is set to the size of the CFData contents

Definition at line 88 of file PPxDataFork.cp.

References PPx::CFData::GetBytePtr(), PPx::CFData::GetLength(), PPx::FileFork::SetSize(), and WriteData().

6.103.3.6 void PPx::DataFork::WriteData (const void ∗ *inBuffer*, ByteCount *inBufferSize*, UInt16 *inPositionMode* = fsFromMark, SInt64 *inOffset* = 0)

Writes bytes from a buffer into a data fork.

Parameters:

inBuffer Pointer to bytes to write

inBufferSize Number of bytes to write

inPositionMode Reference location in fork

inOffset Offset from reference location at which to start writing

Definition at line 160 of file PPxDataFork.cp.

References PPx_ThrowIfOSError_, and PPx::FileFork::UseRefNum().

Referenced by WriteContents().

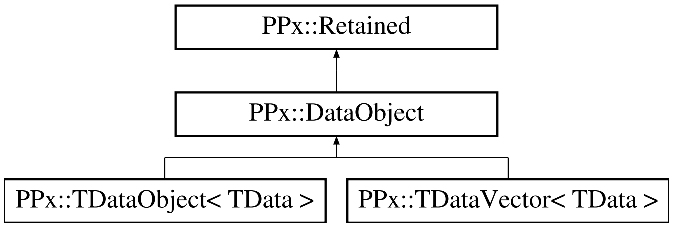
The documentation for this class was generated from the following files:

- PPxDataFork.h
- PPxDataFork.cp

6.104 PPx::DataObject Class Reference

`#include <PPxDataObject.h>`

Inheritance diagram for PPx::DataObject::



6.104.1 Detailed Description

Base class for objects that store a data value.

The template classes TDataObject and TDataVector inherit from this class, so all their template instantiations have a common base class. This lets us store pointers to DataObjects in containers.

Definition at line 26 of file PPxDataObject.h.

The documentation for this class was generated from the following file:

- PPxDataObject.h

6.105 PPx::DataReader Class Reference

```
#include <PPxSerializer.h>
```

6.105.1 Detailed Description

A data dictionary for reading state information.

The items in the dictionary are (key name, data object) pairs.

Definition at line 80 of file PPxSerializer.h.

Public Member Functions

- DataReader (KeyDataMap &inKeyDataMap, ObjectDescriptorList &inDescriptors)

  Constructs from a data dictionary and list of object descriptors.

- bool ContainsKey (CFStringRef inKey) const

  Returns whether a data object with a certain key is in the DataReader.

- template<typename TData> bool ReadOptional (CFStringRef inKey, TData &outValue) const

  Template function for reading the value for an optional item from the dictionary.

- template<typename TData> void ReadRequired (CFStringRef inKey, TData &outValue) const

  Template function for reading the value for a required item from the dictionary.

- template<class TData, class TOutputIterator> bool ReadContainer (CFStringRef inKey, TOutputIterator outDestination) const

  Template function for reading from the dictionary an item that is a list of values.

- template<class TPersistent> bool ReadObjectValue (CFStringRef inKey, TPersistent &outObject) const

  Template function for reading from the dictionary an item that is a pointer to a Persistent object.

- template<class TPersistent, class TOutputIterator> bool ReadObjectContainer (CFStringRef inKey, TOutputIterator outDestination) const

  Template function for reading from the dictionary an item that is a list of pointers to Persistent objects.

6.105.2 Constructor & Destructor Documentation

6.105.2.1 PPx::DataReader::DataReader (KeyDataMap & *inKeyDataMap*, ObjectDescriptorList & *inDescriptors*)

Constructs from a data dictionary and list of object descriptors.

Parameters:

inKeyDataMap Dictionary of (key, data object) pairs

inDescriptors List of object descriptors

Definition at line 176 of file PPxSerializer.cp.

6.105.3 Member Function Documentation

6.105.3.1 bool PPx::DataReader::ContainsKey (CFStringRef *inKey*) const

Returns whether a data object with a certain key is in the DataReader.

Parameters:

inKey Key name of item

Returns:

Whether an item wih the key name exists

Definition at line 196 of file PPxSerializer.cp.

Referenced by PPx::Attachable::ReadAttachments().

6.105.3.2 template<class TData, class TOutputIterator> bool PPx::DataReader::ReadContainer (CFStringRef *inKey*, TOutputIterator *outDestination*) const

Template function for reading from the dictionary an item that is a list of values.

TData is a template parameter for the data type of the item. TOutputIterator is a template parameter for the iterator type.

Parameters:

inKey Key name of item

outDestination Output iterator for storing data values

Returns:

Whether an item with the key name exists

Copies a list of TData values using the output iterator if the key item exists.

Throws an exception if the key item exists but is not a list of TData items.

Definition at line 219 of file PPxSerializer.h.

References PPx_ThrowIfNil_.

6.105.3.3 template<class TPersistent, class TOutputIterator> bool PPx::DataReader::ReadObjectContainer (CFStringRef *inKey*, TOutputIterator *outDestination*) const

Template function for reading from the dictionary an item that is a list of pointers to Persistent objects.

TPersistent is a template parameter for the Persitent object type, which must be a pointer to a subclass of Persistent.

TOutputIterator is a template parameter for the iteration type

Parameters:

inKey Key name of item

outDestination Output iterator for storing object pointers

Returns:

Whether an item with the key name exists

Copies a list of TPersistent pointers using the output iterator if the key item exists.

Throws an exception if the key item exists but is not a list of TPersistent objects.

Definition at line 300 of file PPxSerializer.h.

References PPx_ThrowIfNil_.

Referenced by PPx::Attachable::ReadAttachments().

6.105.3.4 template<class TPersistent> bool PPx::DataReader::ReadObjectValue (CFStringRef *inKey*, TPersistent & *outObject*) const

Template function for reading from the dictionary an item that is a pointer to a Persistent object.

TPersistent is a template parameter for the Persitent object type, which must be a pointer to a subclass of Persistent.

Parameters:

inKey Key name of item

outObject Pointer to persistent object

Whether an item with the key name exists

Sets outObject to nil if the key item does not exist.

Throws an exception if the key item exists but is not a TPersistent object

Definition at line 266 of file PPxSerializer.h.

Referenced by PPx::Window::InitState(), PPx::MessageAttachment::InitState(), PPx::TargetAttachment::InitState(), PPx::CommandTask::InitState(), and PPx::View::InitViewState().

6.105.3.5 template<typename TData> bool PPx::DataReader::ReadOptional (CFStringRef *inKey*, TData & *outValue*) const

Template function for reading the value for an optional item from the dictionary.

Parameters:

TData Template parameter for data type of the item

inKey Key name of item

outValue Vale of the item

Returns:

Whether an item with the key name exists

outValue is unchanged if the key item does not exist.

Throws an exception if key item does exist, but has a type different from TData

Definition at line 148 of file PPxSerializer.h.

References PPx_ThrowIfNil_.

Referenced by PPx::WindowHeader::InitState(), PPx::Window::InitState(), PPx::ThemeTextBox::InitState(), PPx::TextGroupBox::InitState(), PPx::StaticText::InitState(), PPx::Slider::InitState(), PPx::ScrollView::InitState(), PPx::ScrollBar::InitState(), PPx::RoundButton::InitState(), PPx::RelevanceBar::InitState(), PPx::RadioButton::InitState(), PPx::PushButton::InitState(), PPx::ProgressBar::InitState(), PPx::PopupGroupBox::InitState(), PPx::PopupButton::InitState(), PPx::PopupArrow::InitState(), PPx::PictureControl::InitState(), PPx::MLTEView::InitState(), PPx::LittleArrows::InitState(), PPx::ImageWell::InitState(), PPx::ImageView::InitState(), PPx::IconPushButton::InitState(), PPx::IconControl::InitState(), PPx::GrayBox::InitState(), PPx::BindingsFrameAdapter::InitState(), PPx::FrontWindowEventTarget::InitState(), PPx::ResponseAttachment::InitState(), PPx::EditUnicodeText::InitState(), PPx::EditTextControl::InitState(), PPx::DrawerWindow::InitState(), PPx::DisclosureTriangle::InitState(), PPx::DisclosureButton::InitState(), PPx::CommandTask::InitState(), PPx::ComboBox::InitState(), PPx::ClockControl::InitState(), PPx::CheckBoxGroupBox::InitState(), PPx::CheckBox::InitState(), PPx::BevelButton::InitState(), PPx::BaseView::InitState(), PPx::Attachment::InitState(), PPx::View::InitViewState(), and ReadRequired().

6.105.3.6 template<typename TData> void PPx::DataReader::ReadRequired (CFStringRef *inKey*, TData & *outValue*) const

Template function for reading the value for a required item from the dictionary.

Template parameter is the data type of the item.

Parameters:

inKey Key name of item

outValue Value of the item

The same as ReadOptional except that it throws an exception if an item with the key name does not exist.

Definition at line 187 of file PPxSerializer.h.

References PPx_Throw_, and ReadOptional().

Referenced by PPx::Window::InitState(), PPx::MessageAttachment::InitState(), PPx::ResponseAttachment::InitState(), and PPx::EventDoerAttachment::InitState().

The documentation for this class was generated from the following files:

- PPxSerializer.h
- PPxSerializer.cp

6.106 PPx::DataScrap Class Reference

```
#include <PPxDataScrap.h>
```

6.106.1 Detailed Description

A named scrap for storing and retrieving data.

Definition at line 62 of file PPxDataScrap.h.

Public Member Functions

- DataScrap (CFStringRef inScrapName)

  Constructs from a scrap name ∗.

- void GetData (ScrapFlavorType inFlavor, Size &ioByteCount, void ∗outDataPtr) const

  Get data of the specified flavor from the scrap.

- Size GetDataSize (ScrapFlavorType inDataType) const

  Returns the size of the specified flavor of data in the scrap.

- bool HasData (ScrapFlavorType inDataType) const

  Returns whether the scrap has data of the specified flavor.

- void ClearData ()

  Clears all data from the scrap.

- void SetData (ScrapFlavorType inFlavor, Size inDataSize, const void ∗inDataPtr, ScrapFlavorFlags inFlags=kScrapFlavorMaskNone, bool inClear=true)

  Put data into the scrap.

- void PromiseData (ScrapFlavorType inFlavor, Size inDataSize=kScrapFlavorSizeUnknown, ScrapFlavorFlags inFlags=kScrapFlavorMaskNone, bool inClear=true)

  Put a promise to supply data into the scrap.

- void SetPromiseKeeper (ScrapPromiseKeeper ∗inPromiseKeeper)

  Specify the promise keeper object that will supply promised data.

6.106.2 Constructor & Destructor Documentation

6.106.2.1 PPx::DataScrap::DataScrap (CFStringRef *inScrapName*)

Constructs from a scrap name ∗.

Parameters:

inScrapName Name to identify the scrap

Scraps are system-wide entities. Other programs can get data from scrap if they know its name.

Definition at line 64 of file PPxDataScrap.cp.

6.106.3 Member Function Documentation

6.106.3.1 void PPx::DataScrap::GetData (ScrapFlavorType *inFlavor*, Size & *ioByteCount*, void ∗ *outDataPtr*) const

Get data of the specified flavor from the scrap.

Parameters:

inFlavor Flavor of data to get

ioByteCount On input, maximum bytes to get; On output, actual bytes returned

outDataPtr Pointer to data buffer

Call GetDataSize if you need to know the size of the data before getting it.

Definition at line 87 of file PPxDataScrap.cp.

6.106.3.2 SInt32 PPx::DataScrap::GetDataSize (ScrapFlavorType *inFlavor*) const

Returns the size of the specified flavor of data in the scrap.

Parameters:

inFlavor Flavor of data

Returns:

Size in bytes of the data

Definition at line 106 of file PPxDataScrap.cp.

6.106.3.3 bool PPx::DataScrap::HasData (ScrapFlavorType *inFlavor*) const

Returns whether the scrap has data of the specified flavor.

Parameters:

inFlavor Flavor of data

Returns:

Whether the scrap has data of the specified flavor

Definition at line 123 of file PPxDataScrap.cp.

6.106.3.4 void PPx::DataScrap::PromiseData (ScrapFlavorType *inFlavor*, Size *inDataSize* = kScrapFlavorSizeUnknown, ScrapFlavorFlags *inFlags* = kScrapFlavorMaskNone, bool *inClear* = true)

Put a promise to supply data into the scrap.

Parameters:

inFlavor Flavor of data

inDataSize Number of bytes of data

inFlags Options for storing data

inClear Whether to clear the scrap before adding data

The system will caller your promise keeper function if a client requests the flavor data from the scrap.

If you do not know how much data there is, pass -1 for inDataSize

Definition at line 182 of file PPxDataScrap.cp.

6.106.3.5 void PPx::DataScrap::SetData (ScrapFlavorType *inFlavor*, Size *inDataSize*, const void ∗ *inDataPtr*, ScrapFlavorFlags *inFlags* = kScrapFlavorMaskNone, bool *inClear* = true)

Put data into the scrap.

Parameters:

inFlavor Flavor of data

inDataSize Number of bytes of data

inDataPtr Pointer to data buffer

inFlags Options for storing data

inClear Whether to clear the scrap before adding data

Definition at line 154 of file PPxDataScrap.cp.

6.106.3.6 void PPx::DataScrap::SetPromiseKeeper (ScrapPromiseKeeper ∗ *inPromiseKeeper*)

Specify the promise keeper object that will supply promised data.

Parameters:

inPromiseKeeper Pointer to promise keeper object

ScrapPromiseKeeper is an abstract base class. You must create a subclass and override the KeepScrapPromise function. Then pass a pointer to an object of your subclass to this function.

Definition at line 204 of file PPxDataScrap.cp.

The documentation for this class was generated from the following files:

- PPxDataScrap.h
- PPxDataScrap.cp

6.107 PPx::DataWriter Class Reference

```
#include <PPxSerializer.h>
```

6.107.1 Detailed Description

A data dictionary for writing state information.

The items in the dictionary are (key name, data object) pairs.

Definition at line 339 of file PPxSerializer.h.

Public Member Functions

- DataWriter (KeyDataMap &inKeyDataMap, ObjectIDMap &inObjectsProcessed, ObjectQueue &inObjectsToWrite)

  Constructor.

- template<typename TData> void WriteValue (CFStringRef inKey, const TData &inValue)

  Template function for writing a value in the dictionary.

- template<typename TInputIterator> void WriteContainer (CFStringRef inKey, TInputIterator inFirst, TInputIterator inLast)

  Template function for writing a list of values in the dictionary TInputIterator is a template parameter for the iterator type.

- void WriteObjectValue (CFStringRef inKey, const Persistent ∗inObject)

  Writes a Persistent object to the data dictionary.

- ObjectStorageIDT WriteObject (const Persistent ∗inObject)

  Queues a Persistent object for writing.

- template<typename TInputIterator> void WriteObjectContainer (CFStringRef inKey, TInputIterator inFirst, TInputIterator inLast)

  Template function for writing a list of pointers to Persistent objects.

6.107.2 Constructor & Destructor Documentation

6.107.2.1 PPx::DataWriter::DataWriter (KeyDataMap & *inKeyDataMap*, ObjectIDMap & *inObjectsProcessed*, ObjectQueue & *inObjectsToWrite*)

Constructor.

Parameters:

inKeyDataMap Data dictionary of (key, data object) pairs

inObjectsProcessed Map of objects already processed

inObjectsToWrite Queue of objects still to write

Definition at line 324 of file PPxSerializer.cp.

6.107.3 Member Function Documentation

6.107.3.1 template<typename TInputIterator> void PPx::DataWriter::WriteContainer (CFStringRef *inKey*, TInputIterator *inFirst*, TInputIterator *inLast*)

Template function for writing a list of values in the dictionary TInputIterator is a template parameter for the iterator type.

Parameters:

inKey Key name of item

inFirst Input iterator for first value to write

inLast Input iterator for last value to write

Definition at line 416 of file PPxSerializer.h.

6.107.3.2 ObjectStorageIDT PPx::DataWriter::WriteObject (const Persistent ∗ *inObject*)

Queues a Persistent object for writing.

Parameters:

inObject Pointer to a Persistent object

Returns:

Storage ID for the object

Unlike WriteObjectValue, this function does not write a key item for the object in the data dictionary. Call this function when you want to write an object, but do not later need to get back a pointer to the object when restoring state.

For exmaple, if two objects P and Q have pointers to each other, you could have P call WriteObject for Q, and P call WriteObjectValue for Q. When reading, P would do nothing about Q, and Q would call ReadObjectValue for P and then make some other calls that would associate Q with P.

Definition at line 378 of file PPxSerializer.cp.

Referenced by WriteObjectContainer(), WriteObjectValue(), and PPx::View::WriteViewHierarchy().

6.107.3.3 template<typename TInputIterator> void PPx::DataWriter::WriteObjectContainer (CFStringRef *inKey*, TInputIterator *inFirst*, TInputIterator *inLast*)

Template function for writing a list of pointers to Persistent objects.

Parameters:

TInputIterator Template parameter for the iterator type

inKey Key name of item

inFirst Input iterator for first value to write

inLast Input iterator for last value to write

Definition at line 441 of file PPxSerializer.h.

References WriteObject().

Referenced by PPx::Attachable::WriteAttachments().

6.107.3.4 void PPx::DataWriter::WriteObjectValue (CFStringRef *inKey*, const Persistent ∗ *inObject*)

Writes a Persistent object to the data dictionary.

Parameters:

inKey Key name for item

inObject Pointer to a Persistent object

Definition at line 345 of file PPxSerializer.cp.

References WriteObject().

Referenced by PPx::Window::WriteState(), PPx::View::WriteState(), PPx::MessageAttachment::WriteState(), PPx::TargetAttachment::WriteState(), PPx::CommandTask::WriteState(), and PPx::View::WriteViewHierarchy().

6.107.3.5 template<typename TData> void PPx::DataWriter::WriteValue (CFStringRef *inKey*, const TData & *inValue*)

Template function for writing a value in the dictionary.

TData is a template parameter for the value type.

Parameters:

inKey Key name of item

inValue Value of item

Definition at line 396 of file PPxSerializer.h.

Referenced by PPx::WindowHeader::WriteState(), PPx::Window::WriteState(), PPx::View::WriteState(), PPx::ThemeTextBox::WriteState(), PPx::TextGroupBox::WriteState(), PPx::StaticText::WriteState(), PPx::Slider::WriteState(), PPx::ScrollView::WriteState(), PPx::ScrollBar::WriteState(), PPx::RoundButton::WriteState(), PPx::RelevanceBar::WriteState(), PPx::RadioButton::WriteState(), PPx::PushButton::WriteState(), PPx::ProgressBar::WriteState(), PPx::PopupGroupBox::WriteState(), PPx::PopupButton::WriteState(), PPx::PopupArrow::WriteState(), PPx::PictureControl::WriteState(), PPx::MLTEView::WriteState(), PPx::LittleArrows::WriteState(), PPx::ImageWell::WriteState(), PPx::ImageView::WriteState(), PPx::IconPushButton::WriteState(), PPx::IconControl::WriteState(), PPx::GrayBox::WriteState(), PPx::BindingsFrameAdapter::WriteState(), PPx::FrontWindowEventTarget::WriteState(), PPx::MessageAttachment::WriteState(), PPx::ResponseAttachment::WriteState(), PPx::EventDoerAttachment::WriteState(), PPx::EditUnicodeText::WriteState(), PPx::EditTextControl::WriteState(), PPx::DrawerWindow::WriteState(), PPx::DisclosureTriangle::WriteState(), PPx::DisclosureButton::WriteState(), PPx::CommandTask::WriteState(), PPx::ComboBox::WriteState(), PPx::ClockControl::WriteState(), PPx::CheckBoxGroupBox::WriteState(), PPx::CheckBox::WriteState(), PPx::BevelButton::WriteState(), PPx::BaseView::WriteState(), and PPx::Attachment::WriteState().

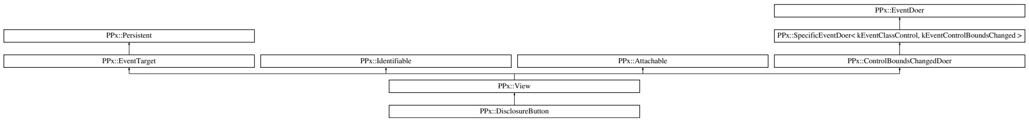
The documentation for this class was generated from the following files:

- PPxSerializer.h
- PPxSerializer.cp

6.108 PPx::DisclosureButton Class Reference

`#include <PPxDisclosureButton.h>`

Inheritance diagram for PPx::DisclosureButton::



6.108.1 Detailed Description

A system disclosure button control.

Definition at line 22 of file PPxDisclosureButton.h.

Public Member Functions

- DisclosureButton ()

  Default constructor.

- virtual ∼DisclosureButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt32 inInitialValue, bool inAutoToggle)

  Initialize from disclosure button creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.108.2 Member Function Documentation

6.108.2.1 void PPx::DisclosureButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt32 *inInitialValue*, bool *inAutoToggle*)

Initialize from disclosure button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inInitialValue kControlDisclosureButtonClosed or kControlDisclosureButtonDisclosed

inAutoToggle Whether button automatically toggles state when clicked

Definition at line 47 of file PPxDisclosureButton.cp.

6.108.2.2 void PPx::DisclosureButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 87 of file PPxDisclosureButton.cp.

References PPx::DataReader::ReadOptional().

6.108.2.3 void PPx::DisclosureButton::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 111 of file PPxDisclosureButton.cp.

References PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

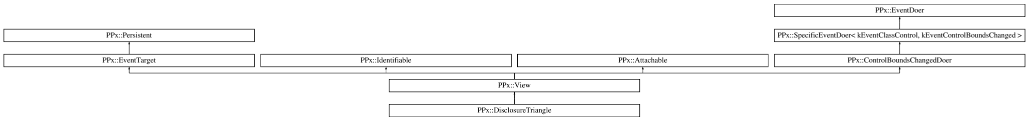
The documentation for this class was generated from the following files:

- PPxDisclosureButton.h
- PPxDisclosureButton.cp

6.109 PPx::DisclosureTriangle Class Reference

`#include <PPxDisclosureTriangle.h>`

Inheritance diagram for PPx::DisclosureTriangle::



6.109.1 Detailed Description

A system disclosure triangle control.

Definition at line 22 of file PPxDisclosureTriangle.h.

Public Member Functions

- DisclosureTriangle ()

  Default constructor.

- virtual ∼DisclosureTriangle ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, ControlDisclosureTriangleOrientation inOrientation, CFStringRef inTitle, SInt32 inInitialValue, bool inDrawTitle, bool inAutoToggle)

  Initialize from disclosure triangle creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.109.2 Member Function Documentation

6.109.2.1 void PPx::DisclosureTriangle::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, ControlDisclosureTriangleOrientation *inOrientation*, CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inDrawTitle*, bool *inAutoToggle*)

Initialize from disclosure triangle creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inOrientation Direction triangle points when closed

inTitle Title for disclosure triangle

inInitialValue 0 = closed, 1 = open

inDrawTitle Whether to draw the title

inAutoToggle Whether the triangle automatically toggles between open/closed when clicked

Definition at line 62 of file PPxDisclosureTriangle.cp.

6.109.2.2 void PPx::DisclosureTriangle::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 107 of file PPxDisclosureTriangle.cp.

References PPx::DataReader::ReadOptional().

6.109.2.3 void PPx::DisclosureTriangle::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 136 of file PPxDisclosureTriangle.cp.

References PPx::View::GetTitle(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

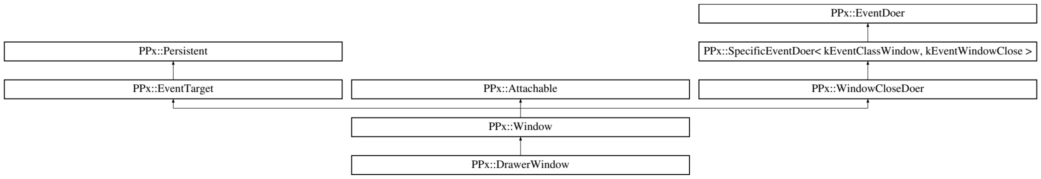
The documentation for this class was generated from the following files:

- PPxDisclosureTriangle.h
- PPxDisclosureTriangle.cp

6.110 PPx::DrawerWindow Class Reference

`#include <PPxDrawerWindow.h>`

Inheritance diagram for PPx::DrawerWindow::



6.110.1 Detailed Description

A drawer which slides out from an edge of a parent window.

Definition at line 22 of file PPxDrawerWindow.h.

Public Member Functions

- DrawerWindow ()

  Default Constructor.

- void Initialize (WindowAttributes inWindAttrs, Window ∗inParentWindow)

  Initializes from parameters.

- void SetParentWindow (Window ∗inParent)

  Sets the parent window for the drawer.

- Window ∗ GetParentWindow () const

  Returns the parent window for the drawer.

- void SetPreferredEdge (OptionBits inPreferredEdge)

  Set the preferred edge from which the drawer slides.

- OptionBits GetPreferredEdge () const

  Returns the preferred edge from which the drawer slides.

- OptionBits GetCurrentEdge () const

  Returns the edge on which a drawer is or would be displayed.

- WindowDrawerState GetDrawerState () const

Returns the current state of the drawer.

- void SetDrawerOffsets (float inLeadingOffset, float inTrailingOffset)

  Sets the offsets of the drawer's size from its parent's size.

- void GetDrawerOffsets (float &outLeadingOffset, float &outTrailingOffset) const

  Passes back the offsets of the drawer's size from its parent's size.

- void Toggle ()

  Toggles the drawer from open to closes, or from closed to open.

- void OpenDrawer (OptionBits inPreferredEdge, bool inAsync)

  Opens the drawer.

- void CloseDrawer (bool inAsync)

  Close the drawer.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.110.2 Member Function Documentation

6.110.2.1 void PPx::DrawerWindow::CloseDrawer (bool *inAsync*)

Close the drawer.

Parameters:

inAsync Whether to close the drawer async or sync

With async opening, the system installs a timer to handle drawer closing and returns immediately. With sysn opening, the draw is fully closed before this function returns.

Programs will have better performance using async closing and handling the kEventWindowDrawerClosed CarbonEvent that informs when the drawer is fully closed.

Definition at line 319 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx_ThrowIfOSError_.

6.110.2.2 OptionBits PPx::DrawerWindow::GetCurrentEdge () const

Returns the edge on which a drawer is or would be displayed.

Returns:

Current edge of the drawer

Definition at line 196 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow().

6.110.2.3 void PPx::DrawerWindow::GetDrawerOffsets (float & *outLeadingEdge*, float & *outTrailingEdge*) const

Passes back the offsets of the drawer's size from its parent's size.

Parameters:

outLeadingEdge Offset of leading edge (top or left)

outTrailingEdge Offset of trailing edge (bottom or right)

A drawer starts with the same width or height of its parent window, depending on which edge it opens. The offsets specify how much to indent the drawer from the edges of the parent window.

Definition at line 255 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx_ThrowIfOSError_.

Referenced by WriteState().

6.110.2.4 WindowDrawerState PPx::DrawerWindow::GetDrawerState () const

Returns the current state of the drawer.

Returns:

Current state of the drawer

The drawer state may be kWindowDrawerOpening, kWindowDrawerOpen, kWindowDrawerClosing, or kWindowDrawerClosed

Definition at line 213 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow().

6.110.2.5 Window ∗ PPx::DrawerWindow::GetParentWindow () const

Returns the parent window for the drawer.

Returns:

Parent window for the drawer

Definition at line 148 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx::Window::GetWindowObject().

6.110.2.6 OptionBits PPx::DrawerWindow::GetPreferredEdge () const

Returns the preferred edge from which the drawer slides.

Returns:

Preferred edge for the drawer

Definition at line 182 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow().

Referenced by WriteState().

6.110.2.7 void PPx::DrawerWindow::Initialize (WindowAttributes *inWindAttrs*, Window ∗ *inParentWindow*)

Initializes from parameters.

Parameters:

inWindAttrs Toolbox window attributes

inParentWindow Window to which drawer is attached

See <MacWindows.h> for information about window attributes

Definition at line 42 of file PPxDrawerWindow.cp.

References SetParentWindow().

6.110.2.8 void PPx::DrawerWindow::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Window.

Definition at line 76 of file PPxDrawerWindow.cp.

References PPx::DataReader::ReadOptional(), SetDrawerOffsets(), and SetPreferredEdge().

6.110.2.9 void PPx::DrawerWindow::OpenDrawer (OptionBits *inPreferredEdge*, bool *inAsync*)

Opens the drawer.

Parameters:

inPreferredEdge Preferred edge on which to open

inAsync Whether to open the drawer async or sync

With async opening, the system installs a timer to handle drawer opening and returns immediately. With sysn opening, the draw is fully opened before this function returns.

Programs will have better performance using async opening and handling the kEventWindowDrawerOpened CarbonEvent that informs when the drawer is fully opened.

Definition at line 293 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx_ThrowIfOSError_.

6.110.2.10 void PPx::DrawerWindow::SetDrawerOffsets (float *inLeadingEdge*, float *inTrailingEdge*)

Sets the offsets of the drawer's size from its parent's size.

Parameters:

inLeadingEdge Offset of leading edge (top or left)

inTrailingEdge Offset of trailing edge (bottom or right)

A drawer starts with the same width or height of its parent window, depending on which edge it opens. The offsets specify how much to indent the drawer from the edges of the parent window.

Definition at line 232 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx_ThrowIfOSError_.

Referenced by InitState().

6.110.2.11 void PPx::DrawerWindow::SetParentWindow (Window ∗ *inParent*)

Sets the parent window for the drawer.

Parameters:

inParent Parent window for the drawer

Definition at line 131 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx_ThrowIfOSError_.

Referenced by Initialize().

6.110.2.12 void PPx::DrawerWindow::SetPreferredEdge (OptionBits *inPreferredEdge*)

Set the preferred edge from which the drawer slides.

Parameters:

inPreferredEdge Preferred edge for the drawer

Values for the preferred edge are KWindowEdgeDefault, kWindowEdgeTop, kWindowEdgeLeft, kWindowEdgeBottom, and kWindowEdgeRight

Definition at line 165 of file PPxDrawerWindow.cp.

References PPx::Window::GetSysWindow(), and PPx_ThrowIfOSError_.

Referenced by InitState().

6.110.2.13 void PPx::DrawerWindow::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Window.

Definition at line 102 of file PPxDrawerWindow.cp.

References GetDrawerOffsets(), GetPreferredEdge(), and PPx::DataWriter::WriteValue().

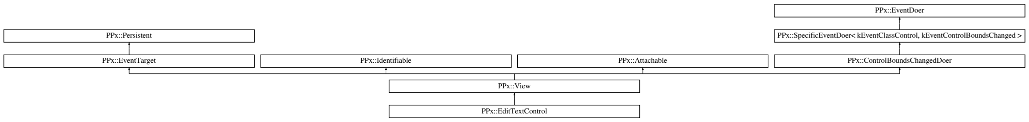
The documentation for this class was generated from the following files:

- PPxDrawerWindow.h
- PPxDrawerWindow.cp

6.111 PPx::EditTextControl Class Reference

`#include <PPxEditTextControl.h>`

Inheritance diagram for PPx::EditTextControl::



6.111.1 Detailed Description

A system edit text control.

Definition at line 22 of file PPxEditTextControl.h.

Public Member Functions

- EditTextControl ()

  Default constructor.

- virtual ∼EditTextControl ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inText, bool inIsPassword, bool inUseInlineInput, const ControlFontStyleRec ∗inStyle)

  Initialize from edit text creation parameters.

- void SetText (CFStringRef inText)

  Sets the text in the edit field.

- CFString GetText () const

  Returns the text from the edit field.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.111.2 Member Function Documentation

6.111.2.1 CFString PPx::EditTextControl::GetText () const

Returns the text from the edit field.

Returns:

Text from the edit field

Definition at line 179 of file PPxEditTextControl.cp.

References PPx::CFData::GetBytePtr(), PPx::View::GetDataTag(), PPx::CFData::GetMutableBytePtr(), and PPx_ThrowIfOSError_.

Referenced by WriteState().

6.111.2.2 void PPx::EditTextControl::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inText*, bool *inIsPassword*, bool *inUseInlineInput*, const ControlFontStyleRec ∗ *inStyle*)

Initialize from edit text creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inText Initial text in edit field

inIsPassword Whether the field is a for password

inUseInlineInput Whether to use inline input

inStyle Text style

Definition at line 60 of file PPxEditTextControl.cp.

6.111.2.3 void PPx::EditTextControl::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 104 of file PPxEditTextControl.cp.

References PPx::DataReader::ReadOptional().

6.111.2.4 void PPx::EditTextControl::SetText (CFStringRef *inText*)

Sets the text in the edit field.

Parameters:

inText Text to put in edit field

Definition at line 155 of file PPxEditTextControl.cp.

References PPx::CFString::GetByteLength(), PPx::CFData::GetBytePtr(), PPx::CFString::GetByteRange(), PPx::CFData::GetMutableBytePtr(), and PPx::View::SetDataTag().

6.111.2.5 void PPx::EditTextControl::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID.

Parameters:

inFont Theme font ID to use for text

Definition at line 207 of file PPxEditTextControl.cp.

6.111.2.6 void PPx::EditTextControl::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 131 of file PPxEditTextControl.cp.

References GetText(), and PPx::DataWriter::WriteValue().

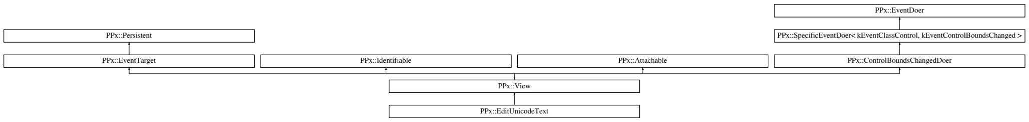
The documentation for this class was generated from the following files:

- PPxEditTextControl.h
- PPxEditTextControl.cp

6.112 PPx::EditUnicodeText Class Reference

`#include <PPxEditUnicodeText.h>`

Inheritance diagram for PPx::EditUnicodeText::



6.112.1 Detailed Description

A system edit unicode text control.

Definition at line 22 of file PPxEditUnicodeText.h.

Public Member Functions

- EditUnicodeText ()

  Default constructor.

- virtual ∼EditUnicodeText ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inText, bool inIsPassword, const ControlFontStyleRec ∗inStyle)

  Initialize from edit text creation parameters.

- virtual void SetText (CFStringRef inText)

  Sets the text in the edit field.

- virtual CFString GetText () const

  Returns the text from the edit field.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.112.2 Member Function Documentation

6.112.2.1 CFString PPx::EditUnicodeText::GetText () const `[virtual]`

Returns the text from the edit field.

Returns:

Text from the edit field

Definition at line 178 of file PPxEditUnicodeText.cp.

References PPx::View::GetDataTag(), and PPx_ThrowIfOSError_.

Referenced by WriteState().

6.112.2.2 void PPx::EditUnicodeText::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inText*, bool *inIsPassword*, const ControlFontStyleRec ∗ *inStyle*)

Initialize from edit text creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inText Initial text in edit field

inIsPassword Whether the field is a for password

inStyle Text style

Definition at line 55 of file PPxEditUnicodeText.cp.

References SetText().

6.112.2.3 void PPx::EditUnicodeText::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 110 of file PPxEditUnicodeText.cp.

References PPx::DataReader::ReadOptional(), and SetText().

6.112.2.4 void PPx::EditUnicodeText::SetText (CFStringRef *inText*) `[virtual]`

Sets the text in the edit field.

Parameters:

inText Text to put in edit field

Definition at line 162 of file PPxEditUnicodeText.cp.

References PPx::View::SetDataTag().

Referenced by Initialize(), and InitState().

6.112.2.5 void PPx::EditUnicodeText::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID.

Parameters:

inFont Theme font ID to use for text

Definition at line 200 of file PPxEditUnicodeText.cp.

6.112.2.6 void PPx::EditUnicodeText::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 140 of file PPxEditUnicodeText.cp.

References GetText(), and PPx::DataWriter::WriteValue().

The documentation for this class was generated from the following files:

- PPxEditUnicodeText.h
- PPxEditUnicodeText.cp

6.113 PPx::XMLEncoder::EncoderInfo Struct Reference

```
#include <PPxXMLEncoder.h>
```

6.113.1 Detailed Description

Data stored for each registered enccoder type.

Definition at line 37 of file PPxXMLEncoder.h.

Public Attributes

- CFStringRef typeName

  XML element tag name.

- EncoderFuncT encoderFunc

  Encoder function pointer.

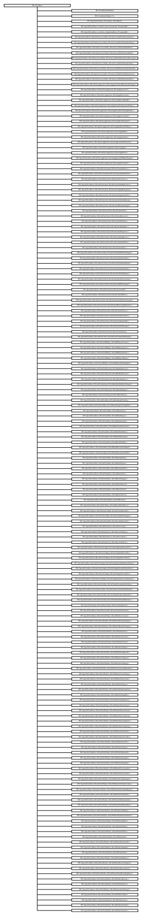
The documentation for this struct was generated from the following file:

- PPxXMLEncoder.h

6.114 PPx::EventDoer Class Reference

```
#include <PPxEventDoer.h>
```

Inheritance diagram for PPx::EventDoer::



6.114.1 Detailed Description

Abstract class for a Carbon Event handler.

Definition at line 20 of file PPxEventDoer.h.

Public Member Functions

- virtual ~EventDoer ()

  Destructor.

- EventHandlerRef Install (EventTargetRef inTarget, UInt32 inNumTypes, const EventTypeSpec ∗inTypeList)

  Installs handler for a list of events.

- EventHandlerRef Install (EventTargetRef inTarget, UInt32 inEventClass, UInt32 inEventKind)

  Installs handler for a single event.

- OSStatus Invoke (SysCarbonEvent &ioEvent)

  Calls function to handle an event.

6.114.2 Member Function Documentation

6.114.2.1 EventHandlerRef PPx::EventDoer::Install (EventTargetRef *inTarget*, UInt32 *inEventClass*, UInt32 *inEventKind*)

Installs handler for a single event.

Parameters:

inTarget Target on which to install handler

inEventClass Carbon Event class

inEventKind Carbon Event kind

Returns:

EventHandlerRef for the installed event handler

Definition at line 107 of file PPxEventDoer.cp.

References Install().

6.114.2.2 EventHandlerRef PPx::EventDoer::Install (EventTargetRef *inTarget*, UInt32 *inNumTypes*, const EventTypeSpec ∗ *inTypeList*)

Installs handler for a list of events.

Parameters:

inTarget Target on which to install handler

inNumTypes Number of event types

inTypeList Array of event types

Returns:

EventHandlerRef for the installed event handler

Definition at line 80 of file PPxEventDoer.cp.

References PPx::SysEventHandler::Detach(), and PPx::SysEventHandler::Install().

Referenced by Install().

6.114.2.3 OSStatus PPx::EventDoer::Invoke (SysCarbonEvent & *ioEvent*)

Calls function to handle an event.

Parameters:

ioEvent CarbonEvent being handled

Returns:

OS error code

Definition at line 128 of file PPxEventDoer.cp.

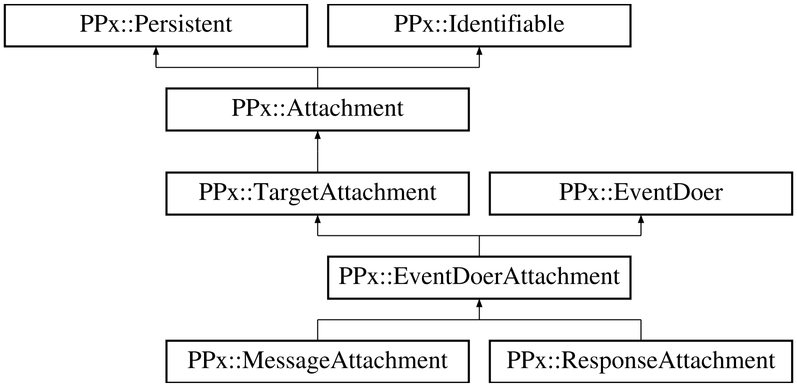
The documentation for this class was generated from the following files:

- PPxEventDoer.h
- PPxEventDoer.cp

6.115 PPx::EventDoerAttachment Class Reference

`#include <PPxEventAttachments.h>`

Inheritance diagram for PPx::EventDoerAttachment::



6.115.1 Detailed Description

Abstract attachment that has an associated event target and specific event type.

Definition at line 51 of file PPxEventAttachments.h.

Public Member Functions

- void **Initialize** (EventTarget ∗inTarget, EventClassT inEventClass, EventKindT inEventKind)
- void **InstallEventHandler** ()
- void **RemoveEventHandler** ()

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.115.2 Member Function Documentation

6.115.2.1 void PPx::EventDoerAttachment::InitState (const DataReader & *inReader*) [protected, virtual]

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::TargetAttachment.

Reimplemented in PPx::ResponseAttachment, and PPx::MessageAttachment.

Definition at line 143 of file PPxEventAttachments.cp.

References PPx::SysEventSpec::eventClass, PPx::SysEventSpec::eventKind, and PPx::DataReader::ReadRequired().

6.115.2.2 void PPx::EventDoerAttachment::WriteState (DataWriter & *ioWriter*) const [protected, virtual]

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::TargetAttachment.

Reimplemented in PPx::ResponseAttachment, and PPx::MessageAttachment.

Definition at line 163 of file PPxEventAttachments.cp.

References PPx::SysEventSpec::eventClass, PPx::SysEventSpec::eventKind, and PPx::DataWriter::WriteValue().

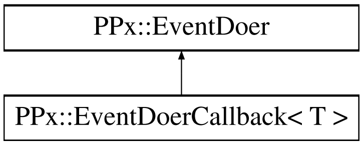
The documentation for this class was generated from the following files:

- PPxEventAttachments.h
- PPxEventAttachments.cp

6.116 PPx::EventDoerCallback< T > Class Template Reference

```
#include <PPxEventDoer.h>
```

Inheritance diagram for PPx::EventDoerCallback< T >::



6.116.1 Detailed Description

template<class T> class PPx::EventDoerCallback< T >

Template class for an EventDoer that calls a member function of an object.

Definition at line 65 of file PPxEventDoer.h.

Public Types

- typedef OSStatus(T::∗ **CallbackFunction**)(SysCarbonEvent &)

Public Member Functions

- EventHandlerRef **Install** (T ∗inObject, CallbackFunction inFunction, EventTargetRef inTarget, UInt32 inNumTypes, const EventTypeSpec ∗inTypeList)
- EventHandlerRef **Install** (T ∗inObject, CallbackFunction inFunction, EventTargetRef inTarget, UInt32 inEventClass, UInt32 inEventKind)
- void **SetCallback** (T ∗inObject, CallbackFunction inFunction)
- virtual OSStatus **DoEvent** (SysCarbonEvent &ioEvent)

The documentation for this class was generated from the following file:

- PPxEventDoer.h

6.117 PPx::EventMouseWheelAxisStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.117.1 Detailed Description

Wrapper for EventMouseWheelAxis.

Definition at line 71 of file PPxSysTypes.h.

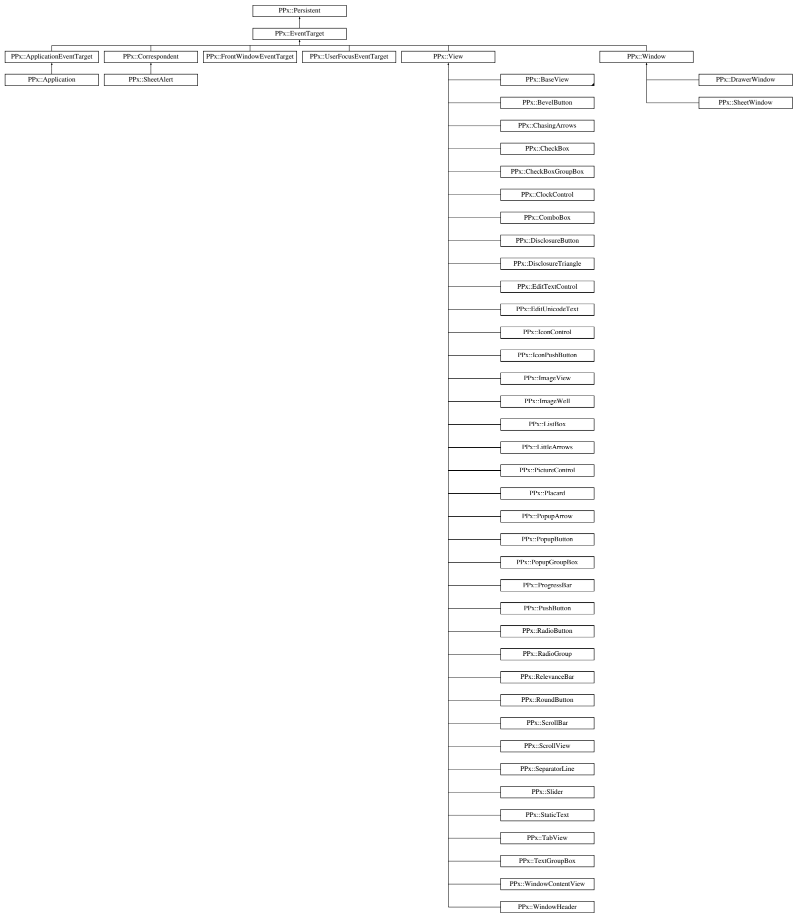
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.118 PPx::EventTarget Class Reference

`#include <PPxEventTarget.h>`

Inheritance diagram for PPx::EventTarget::



6.118.1 Detailed Description

Abstract class for the target of a Carbon Event.

Definition at line 23 of file PPxEventTarget.h.

Public Member Functions

- EventTargetRef GetSysEventTarget () const

  Returns the associated system EventTargetRef.

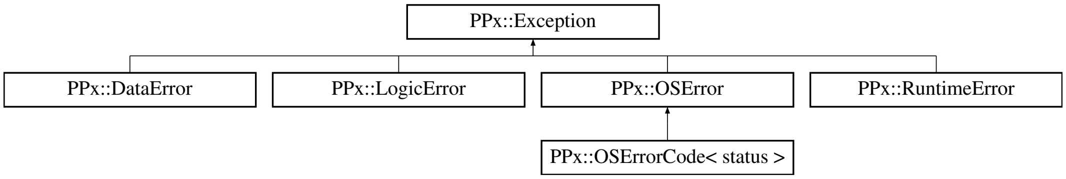
The documentation for this class was generated from the following file:

- PPxEventTarget.h

6.119 PPx::Exception Class Reference

```
#include <PPxExceptions.h>
```

Inheritance diagram for PPx::Exception::



6.119.1 Detailed Description

Base class for PowerPlant X exceptions.

Exception objects store an identifier indicating the kind of error, and, when the PPx_-Debug_Exceptions preprocessor flag is set, an explanation string and source code location of the throw.

The constructors and destructor are protected to enforce that Exception is a base class and never directly instantiated.

Definition at line 47 of file PPxExceptions.h.

Public Member Functions

- ExceptionIDT What () const

  Returns the kind of an exception.

- virtual StringPtr Why (Str255 outWhy) const

  Returns a Pascal string describing why an exception was thrown.

- const SourceLocation & Where () const

  Returns the source code location where the exception was thrown.

Protected Member Functions

- Exception (ExceptionIDT inWnat, const char ∗inWhy, const SourceLocation &inWhere)

  Constructor.

- virtual ~Exception ()

  Destructor.

6.119.2 Constructor & Destructor Documentation

6.119.2.1 PPx::Exception::Exception (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[protected]`

Constructor.

Parameters:

inWhat Kind of exception

inWhy C string describing why the exception occurred

inWhere Source code location where exception was thrown

Note:

When the option PPx_Debug_Exceptiond is false, the why and where are not stored.

Definition at line 24 of file PPxExceptions.cp.

6.119.3 Member Function Documentation

6.119.3.1 const SourceLocation & PPx::Exception::Where () const

Returns the source code location where the exception was thrown.

Returns:

Source code location where exception was thrown

Location specifies the source file name, function name, and line number.

Note:

Location fields are nil if PPx_Debug_Exceptions is false.

Definition at line 105 of file PPxExceptions.cp.

References PPx::sourceLocation_Nothing.

6.119.3.2 StringPtr PPx::Exception::Why (Str255 *outWhy*) const `[virtual]`

Returns a Pascal string describing why an exception was thrown.

Parameters:

outWhy Pascal string in which to store the description

Note:

If PPx_Debug_Exceptions is false, description is an empty string

Reimplemented in PPx::OSError.

Definition at line 75 of file PPxExceptions.cp.

The documentation for this class was generated from the following files:

- PPxExceptions.h
- PPxExceptions.cp

6.120 PPx::File Class Reference

```
#include <PPxFile.h>
```

6.120.1 Detailed Description

A file on disk.

Provides functions for creating a file an accessing its data and resource forks

File does not provide I/O operations. It creates DataFork and ResourceFork classes which implement reading and writing.

Definition at line 28 of file PPxFile.h.

Public Member Functions

- File ()

  Default constructor.

- File (const FSObject &inLocation)

  Constructs from a FSObject.

- File (const File &inOriginal)

  Copy constructor.

- File & operator= (const File &inSource)

  Assignment operator.

- bool IsEqualTo (const File &inOther) const

  Returns whether the File represents the same disk file as another File object.

- const FSObject & GetLocation () const

  Returns a reference to a FSObject specifying the file's location.

- void UpdateLocation ()

  Updates File's location to reflect its on-disk state.

- void Invalidate ()

  Invalidates the internal state of the File object.

- void CreateOnDisk (FSCatalogInfoBitmap inCatInfoFlags, const FSCatalogInfo &inCatInfo, bool inReplace)

Creates file on disk with the specifed catalog information.

- void CreateOnDisk (OSType inFileType, OSType inCreator, bool inReplace)

  Creates file on disk with the specifed type and creator.

- void DeleteOnDisk ()

  Deletes file from disk.

- void GetTotalForkSizes (UInt64 ∗outLogicalSize, UInt64 ∗outPhysicalSize=nil, ItemCount ∗outForkCount=nil) const

  Passes back the total logical size, physical size, and number of forks of a file.

- DataFork ∗ GetDataFork ()

  Returns DataFork object for the File.

- bool IsDataForkOpen () const

  Returns whether the File's data fork is open.

- DataFork ∗ OpenDataFork (SInt8 inPermissions=fsRdWrPerm)

  Opens a File's data fork.

- void CloseDataFork ()

  Closes a File's data fork.

- ResourceFork ∗ GetResourceFork ()

  Returns ResourceFork object for the File.

- bool IsResourceForkOpen () const

  Returns whether the File's resource fork is open.

- ResourceFork ∗ OpenResourceFork (SInt8 inPermissions=fsRdWrPerm)

  Opens a File's resource fork.

- void CloseResourceFork ()

  Closes a File's resource fork.

6.120.2 Constructor & Destructor Documentation

6.120.2.1 PPx::File::File (const FSObject & *inLocation*)

Constructs from a FSObject.

Parameters:

inLocation FSObject specifying the file location

Definition at line 27 of file PPxFile.cp.

6.120.2.2 PPx::File::File (const File & *inOriginal*)

Copy constructor.

The data and resource fork objects are not copied. The File has both forks closed after construction.

Definition at line 43 of file PPxFile.cp.

6.120.3 Member Function Documentation

6.120.3.1 void PPx::File::CreateOnDisk (OSType *inFileType*, OSType *inCreator*, bool *inReplace*)

Creates file on disk with the specifed type and creator.

Parameters:

inFileType File type

inCreator Creator code

inReplace Whether it's OK to replace an existing file

Definition at line 195 of file PPxFile.cp.

References CreateOnDisk().

6.120.3.2 void PPx::File::CreateOnDisk (FSCatalogInfoBitmap *inCatInfoFlags*, const FSCatalogInfo & *inCatInfo*, bool *inReplace*)

Creates file on disk with the specifed catalog information.

Parameters:

inCatInfoFlags Bit flags specifying which information to set

inCatInfo Catalog informatin to set

inReplace Whether it's OK to replace an existing file

Definition at line 156 of file PPxFile.cp.

References PPx::FSObject::Delete(), PPx::FSObject::Exists(), PPx::FSObject::GetName(), PPx::FSObject::GetParent(), PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by CreateOnDisk().

6.120.3.3 void PPx::File::DeleteOnDisk ()

Deletes file from disk.

Note:

The File object is still valid afterwards, so you can recreate the file by calling CreateOnDisk at a later time.

Definition at line 221 of file PPxFile.cp.

References CloseDataFork(), CloseResourceFork(), and PPx::FSObject::Delete().

6.120.3.4 DataFork ∗ PPx::File::GetDataFork () `[inline]`

Returns DataFork object for the File.

Returns:

DataFork object for the File

Definition at line 96 of file PPxFile.h.

6.120.3.5 const FSObject & PPx::File::GetLocation () const

Returns a reference to a FSObject specifying the file’s location.

Returns:

Reference to FSObject specifying the file’s location

Definition at line 98 of file PPxFile.cp.

6.120.3.6 ResourceFork ∗ PPx::File::GetResourceFork () `[inline]`

Returns ResourceFork object for the File.

Returns:

ResourceFork object for the File

Definition at line 124 of file PPxFile.h.

6.120.3.7 void PPx::File::GetTotalForkSizes (UInt64 ∗ *outLogicalSize*, UInt64 ∗ *outPhysicalSize* = nil, ItemCount ∗ *outForkCount* = nil) const

Passes back the total logical size, physical size, and number of forks of a file.

Parameters:

outLogicalSize Sum of all fork logical sizes

outPhysicalSize Sum of all fork physical sizes

outForkCount Number of file forks

Note:

Pass nil for any of the parameters if you do not wish to obtain that piece of information

Definition at line 244 of file PPxFile.cp.

References PPx_ThrowIfOSError_, and PPx::FSObject::UseRef().

6.120.3.8 void PPx::File::Invalidate ()

Invalidates the internal state of the File object.

Until you respecify the file location via the assignment operator, any future attempt to perform a non-const operation on the File will cause an exception.

Call if you know that the File no longer exists through some means external to this object, such as user actions in the Finder.

Definition at line 138 of file PPxFile.cp.

References CloseDataFork(), CloseResourceFork(), and PPx::FSObject::Invalidate().

6.120.3.9 bool PPx::File::IsDataForkOpen () const `[inline]`

Returns whether the File's data fork is open.

Returns:

Whether the File's data fork is open

Definition at line 110 of file PPxFile.h.

6.120.3.10 bool PPx::File::IsEqualTo (const File & *inOther*) const

Returns whether the File represents the same disk file as another File object.

Parameters:

inOther File object with which to test equality

Returns:

Whether the File objects refer to the same file

Definition at line 83 of file PPxFile.cp.

References PPx::FSObject::IsEqualTo(), and mLocation.

6.120.3.11 bool PPx::File::IsResourceForkOpen () const `[inline]`

Returns whether the File's resource fork is open.

Returns:
Whether the File's resource fork is open

Definition at line 138 of file PPxFile.h.

6.120.3.12 DataFork ∗ PPx::File::OpenDataFork (SInt8 *inPermissions* = fsRdWrPerm)

Opens a File's data fork.

Parameters:
inPermissions Access permissions for the data fork

Returns:
DataFork object for the File

Note:
If the data fork is already open, the access permissions are not changed

Definition at line 268 of file PPxFile.cp.

References PPx::FSObject::UseRef().

6.120.3.13 ResourceFork ∗ PPx::File::OpenResourceFork (SInt8 *inPermissions* = fsRdWrPerm)

Opens a File's resource fork.

Parameters:
inPermissions Access permissions for the resource fork

Returns:
ResourceFork object for the File

Note:
If the resource fork is already open, the access permissions are not changed

Definition at line 304 of file PPxFile.cp.

References PPx::FSObject::UseRef().

6.120.3.14 File & PPx::File::operator= (const File & *inSource*)

Assignment operator.

The data resource forks are not assigned. The File has both forks after assignment.

Definition at line 59 of file PPxFile.cp.

References mLocation.

6.120.3.15 void PPx::File::UpdateLocation ()

Updates File's location to reflect its on-disk state.

Call if you think the File may have benn moved or deleted via external means, such as user actions in the Finder

Definition at line 113 of file PPxFile.cp.

References PPx::FSObject::IsValid(), and PPx::FSObject::Update().

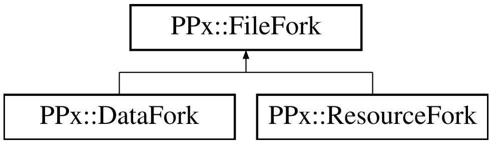
The documentation for this class was generated from the following files:

- PPxFile.h
- PPxFile.cp

6.121 PPx::FileFork Class Reference

```
#include <PPxFileFork.h>
```

Inheritance diagram for PPx::FileFork::



6.121.1 Detailed Description

Wrapper class for a fork of a file.

Definition at line 25 of file PPxFileFork.h.

Public Member Functions

- FileFork (const FSRef &inFile, SInt8 inPermissions, const HFSUniStr255 &inForkName)

  Constructs a FileFork object for a file and opens the fork.

- FileFork (SInt16 inRefNum, bool inOwnsRefNum)

  Constructs from an already open file fork.

- virtual ~FileFork ()

  Destructor.

- SInt16 UseRefNum () const

  Returns the fork's reference number.

- bool IsOpen () const

  Returns whether the fork is open.

- SInt64 GetSize () const

  Returns the size of a file.

- void SetSize (SInt64 inSize, UInt16 inPositionMode=fsFromStart)

  Sets the size of a file.

- SInt64 GetPosition () const

  Returns the current position in the fork.

- void SetPosition (SInt64 inOffset, UInt16 inPositionMode=fsFromStart)

  Sets the current position in the fork.

- void GetFSRef (FSRef &outRef) const

  Passes back the FSRef for the file containing the fork.

- void GetFSObject (FSObject &outFSObject) const

  Passes back the FSObject for the file containing the fork.

- void GetForkName (HFSUniStr255 &outForkName) const

  Passes back the name of the fork.

- void GetForkInfo (FSForkInfo &outForkInfo) const

  Passes back the FSForkInfo struct for the fork.

- SInt16 Open (const FSRef &inFile, SInt8 inPermissions, const HFSUniStr255 &inForkName)

  Opens a file fork with the specified permissions.

- void Close ()

  Close a file fork.

- void Flush ()

  Flushes the fork cotents to disk.

Protected Attributes

- SInt16 mRefNum

  Open file fork referece number.

- bool mOwnsRefNum

  Whether to close fork upon destruction.

6.121.2 Constructor & Destructor Documentation

6.121.2.1 PPx::FileFork::FileFork (const FSRef & *inFile*, SInt8 *inPermissions*, const HFSUniStr255 & *inForkName*)

Constructs a FileFork object for a file and opens the fork.

Parameters:

inFile FSRef for the file

inPermissions Access permissions

inForkName Name of the fork

Definition at line 19 of file PPxFileFork.cp.

References Open().

6.121.2.2 PPx::FileFork::FileFork (SInt16 *inRefNum*, bool *inOwnsRefNum*)

Constructs from an already open file fork.

Parameters:

inRefNum Reference number for an open fork

inOwnsRefNum Whether to close the fork upon destruction

Definition at line 39 of file PPxFileFork.cp.

6.121.2.3 PPx::FileFork::∼FileFork () `[virtual]`

Destructor.

Closes the fork if we own it.

Definition at line 54 of file PPxFileFork.cp.

References Close().

6.121.3 Member Function Documentation

6.121.3.1 void PPx::FileFork::GetForkInfo (FSForkInfo & *outForkInfo*) const

Passes back the FSForkInfo struct for the fork.

Parameters:

outForkInfo FSForkInfo struct to fill in

Definition at line 226 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

6.121.3.2 void PPx::FileFork::GetForkName (HFSUniStr255 & *outForkName*) const

Passes back the name of the fork.

Parameters:

outForkName Name of the fork

Definition at line 209 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

6.121.3.3 void PPx::FileFork::GetFSObject (FSObject & *outFSObject*) const

Passes back the FSObject for the file containing the fork.

Parameters:

outFSObject FSObject for the fork's file

Definition at line 191 of file PPxFileFork.cp.

References GetFSRef().

6.121.3.4 void PPx::FileFork::GetFSRef (FSRef & *outRef*) const

Passes back the FSRef for the file containing the fork.

Parameters:

outRef FSRef for the fork's file

Definition at line 174 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

Referenced by GetFSObject().

6.121.3.5 SInt64 PPx::FileFork::GetPosition () const

Returns the current position in the fork.

Returns:

Current position in the fork

Note:

The reference location fsFromMark refers to the current position

Definition at line 136 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

6.121.3.6 SInt64 PPx::FileFork::GetSize () const

Returns the size of a file.

Returns:

Size of the file, in bytes

Definition at line 98 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

Referenced by PPx::DataFork::ReadContents().

6.121.3.7 bool PPx::FileFork::IsOpen () const

Returns whether the fork is open.

Returns:

Whether the fork is open

Definition at line 84 of file PPxFileFork.cp.

References mRefNum.

6.121.3.8 SInt16 PPx::FileFork::Open (const FSRef & *inFile*, SInt8 *inPermissions*, const HFSUniStr255 & *inForkName*)

Opens a file fork with the specified permissions.

Parameters:

inFile FSRef for the file

inPermissions Access permissions

inForkName Name of the fork to open

Returns:

File reference number for accessing the fork

Definition at line 247 of file PPxFileFork.cp.

References mRefNum, PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by FileFork().

6.121.3.9 void PPx::FileFork::SetPosition (SInt64 *inOffset*, UInt16 *inPositionMode* = fsFromStart)

Sets the current position in the fork.

Parameters:

inOffset Bytes offset from the reference location

inPositionMode Reference location within the file

Note:

The reference location fsFromMark refers to the current position

Definition at line 157 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

6.121.3.10 void PPx::FileFork::SetSize (SInt64 *inSize*, UInt16 *inPositionMode* = fsFromStart)

Sets the size of a file.

Parameters:

inSize Size in bytes from the reference location

inPositionMode Reference location in fork

Definition at line 117 of file PPxFileFork.cp.

References PPx_ThrowIfOSError_, and UseRefNum().

Referenced by PPx::DataFork::WriteContents().

6.121.3.11 SInt16 PPx::FileFork::UseRefNum () const

Returns the fork's reference number.

Returns:

Fork's reference number

Definition at line 68 of file PPxFileFork.cp.

References mRefNum.

Referenced by Flush(), GetForkInfo(), GetForkName(), GetFSRef(), GetPosition(), GetSize(), PPx::DataFork::ReadData(), SetPosition(), SetSize(), and PPx::DataFork::WriteData().

The documentation for this class was generated from the following files:

- PPxFileFork.h
- PPxFileFork.cp

6.122 PPx::Folder Class Reference

```
#include <PPxFolder.h>
```

6.122.1 Detailed Description

Encapsulates a Mac OS file system folder.

Provides functions for creating a folder on disk and obtaining information about an existing folder.

Definition at line 24 of file PPxFolder.h.

Public Member Functions

- Folder ()

  Default constructor.

- Folder (const FSObject &inLocation)

  Constructs from an FSObject specifying the folder location.

- Folder (FSVolumeRefNum inVolume, SInt32 inDirID)

  Constructs from a volume and directory ID.

- Folder (const Folder &inOriginal)

  Copy constructor.

- Folder & operator= (const Folder &inOriginal)

  Assignment operator.

- bool IsEqualTo (const Folder &inOther) const

  Returns whether the Folder refers to the same on-disk folder as another Folder object.

- const FSObject & GetLocation () const

  Returns FSObject specifying the Folders' location.

- FSVolumeRefNum GetVolume () const

  Returns the Folder's volume reference number.

- SInt32 GetDirID () const

  Returns the Folder's directory ID.

- void UpdateLocation ()

Updates Folder's location to reflect its on-disk state.

- void Invalidate ()

  Invalidates the internal state of the Folder object.

- void CreateOnDisk (FSCatalogInfoBitmap inCatInfoFlags, const FSCatalogInfo &inCatInfo, bool inReplace)

  Creates folder on disk with the specified catalog information.

- void DeleteOnDisk ()

  Deletes folder from disk.

- void DeleteContents ()

  Deletes on disk the files and directories within the Folder.

6.122.2 Constructor & Destructor Documentation

6.122.2.1 PPx::Folder::Folder (const FSObject & *inLocation*)

Constructs from an FSObject specifying the folder location.

Parameters:

inLocation FSObject specifying the folder location

Definition at line 28 of file PPxFolder.cp.

6.122.2.2 PPx::Folder::Folder (FSVolumeRefNum *inVolume*, SInt32 *inDirID*)

Constructs from a volume and directory ID.

Parameters:

inVolume Volume reference number

inDirID Directory ID of folder

Definition at line 45 of file PPxFolder.cp.

References UpdateLocation().

6.122.3 Member Function Documentation

6.122.3.1 void PPx::Folder::CreateOnDisk (FSCatalogInfoBitmap *inCatInfoFlags*, const FSCatalogInfo & *inCatInfo*, bool *inReplace*)

Creates folder on disk with the specified catalog information.

Parameters:

inCatInfoFlags Bit flags specifying which information to set

inCatInfo Catalog informatin to set

inReplace Whether it's OK to replace an existing folder

Definition at line 173 of file PPxFolder.cp.

References PPx::FSObject::Exists(), PPx::FSObject::GetName(), PPx::FSObject::GetParent(), PPx::FSObject::GetVolume(), PPx_Throw_, PPx_ThrowIfOSError_, and PPx::FSObject::UseRef().

6.122.3.2 void PPx::Folder::DeleteOnDisk ()

Deletes folder from disk.

Note:

The Folder object is still valid afterwards, so you can recreate the folder by calling CreateOnDisk at a later time.

Definition at line 221 of file PPxFolder.cp.

References PPx::FSObject::DeleteContainer().

6.122.3.3 SInt32 PPx::Folder::GetDirID () const `[inline]`

Returns the Folder's directory ID.

Returns:

Folder's directory ID

Definition at line 106 of file PPxFolder.h.

6.122.3.4 const FSObject & PPx::Folder::GetLocation () const `[inline]`

Returns FSObject specifying the Folders' location.

Returns:

FSObject specifying the Folders' location

Definition at line 78 of file PPxFolder.h.

6.122.3.5 FSVolumeRefNum PPx::Folder::GetVolume () const [inline]

Returns the Folder's volume reference number.

Returns:

Folder's volume reference number

Definition at line 92 of file PPxFolder.h.

6.122.3.6 void PPx::Folder::Invalidate ()

Invalidates the internal state of the Folder object.

Until you respecify the folder location via the assigment operator, any future attempt to perform a non-const operation on the Folder will cause an exception.

Call if you know that the Folder no longer exists through some means external to this object, such as user actions in the Finder.

Definition at line 154 of file PPxFolder.cp.

References PPx::FSObject::Invalidate().

6.122.3.7 bool PPx::Folder::IsEqualTo (const Folder & *inOther*) const

Returns whether the Folder refers to the same on-disk folder as another Folder object.

Parameters:

inOther Folder with which to test equality

Returns:

Whether the Folder objects refer to the same folder

Definition at line 100 of file PPxFolder.cp.

References PPx::FSObject::IsEqualTo(), and mLocation.

6.122.3.8 void PPx::Folder::UpdateLocation ()

Updates Folder's location to reflect its on-disk state.

Call if you think the Folder may have been moved or deleted via external means, such as user actions in the Finder

Definition at line 116 of file PPxFolder.cp.

References PPx_ThrowIfOSError_, and PPx::FSObject::Update().

Referenced by Folder().

The documentation for this class was generated from the following files:

- PPxFolder.h
- PPxFolder.cp

6.123 PPx::FourCharCodeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.123.1 Detailed Description

Wrapper for FourCharCode.

Definition at line 42 of file PPxSysTypes.h.

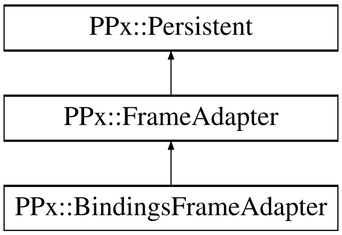
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.124 PPx::FrameAdapter Class Reference

`#include <PPxFrameAdapter.h>`

Inheritance diagram for PPx::FrameAdapter::



6.124.1 Detailed Description

Abstract class for adjusting the frame of a view when its container frame changes size.

Definition at line 23 of file PPxFrameAdapter.h.

Public Member Functions

- void AdaptFrame (const HIRect &inOldSuperFrame, const HIRect &inNewSuperFrame, HIRect &ioFrame) const

  Non-virtual public wrapper for DoAdaptFrame.

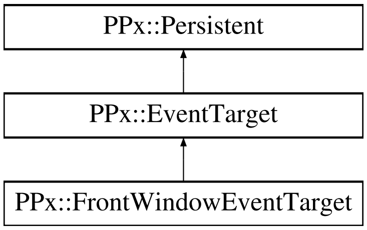
The documentation for this class was generated from the following file:

- PPxFrameAdapter.h

6.125 PPx::FrontWindowEventTarget Class Reference

`#include <PPxEventTarget.h>`

Inheritance diagram for PPx::FrontWindowEventTarget::



6.125.1 Detailed Description

Carbon Event target for the front window of a window layer.

Definition at line 76 of file PPxEventTarget.h.

Public Member Functions

- FrontWindowEventTarget ()

  Default constructor.

- FrontWindowEventTarget (WindowClass inWindowClass)

  Constructs from a window class.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.125.2 Constructor & Destructor Documentation

6.125.2.1 PPx::FrontWindowEventTarget::FrontWindowEventTarget (WindowClass *inWindowClass*) `[explicit]`

Constructs from a window class.

Parameters:

inWindowClass Class of Toolbox window

Definition at line 89 of file PPxEventTarget.cp.

6.125.3 Member Function Documentation

6.125.3.1 void PPx::FrontWindowEventTarget::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Persistent.

Definition at line 104 of file PPxEventTarget.cp.

References PPx::DataReader::ReadOptional().

6.125.3.2 void PPx::FrontWindowEventTarget::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Definition at line 119 of file PPxEventTarget.cp.

References PPx::DataWriter::WriteValue().

The documentation for this class was generated from the following files:

- PPxEventTarget.h
- PPxEventTarget.cp

6.126 PPx::FSObject Class Reference

```
#include <PPxFSObject.h>
```

6.126.1 Detailed Description

Wrapper for a system file reference (FSRef) and related File Manager and MoreFiles X functions.

FSObject supports describing an entity that does not yet exist. FSRef does not have this feature, but FSSpec, the former standard OS file description, does have this feature.

Entities that do not exist are described by their name and parent folder. The File and Folder subclasses of FSObject have CreateOnDisk() functions to create an entity. FSObject throws an exception if you attempt to perform a file system operation on an entity that does not exist.

Definition at line 40 of file PPxFSObject.h.

Public Member Functions

- FSObject ()

  Default constructor.

- FSObject (const FSRef &inFSRef)

  Constructs from a file system FSRef.

- FSObject (const FSRef &inParentRef, const CFString &inName)

  Constructs from a parent FSRef and CFString item name.

- FSObject (const FSRef &inParentRef, const HFSUniStr255 &inName)

  Constructs from a parent FSRef and HFSUniStr255 item name.

- FSObject (FSVolumeRefNum inVRefNum, SInt32 inParentDirID, const CFString &inName)

  Contructs from a volume refnum, parent directory ID, and CFString item name.

- FSObject (FSVolumeRefNum inVRefNum, SInt32 inParentDirID, const HFSUniStr255 &inName)

  Contructs from a volume refnum, parent directory ID, and HFSUniStr255 item name.

- FSObject (const FSSpec &inFSSpec, CFStringEncoding inNameEncoding=encoding_System)

  Constructs from a FSSpec.

- FSObject (CFURLRef inURL)

  Constructs from a Core Foundation URL reference.

- FSObject (const CFString &inAbsolutePath, CFURLPathStyle inPathStyle=kCFURLPOSIXPathStyle)

  Constructs from an absolute path name.

- FSObject (const CFString &inRelativePath, const FSRef &inBaseDir, CFURLPathStyle inPathStyle=kCFURLPOSIXPathStyle)

  Constructs from a relative path name.

- FSObject (const FSObject &inOriginal)

  Copy constructor.

- virtual ∼FSObject ()

  Destructor.

- FSObject & operator= (const FSObject &inSource)

  Assignment operator.

- FSObject & operator= (const FSRef &inFSRef)

  Assigns FSObject from a FSRef.

- const FSRef & UseRef () const

  Returns a const reference to a FSRef.

- bool IsValid () const

  Returns whether the FSObject refers to a valid file system item.

- bool Exists () const

  Returns whether the FSObject refers to an existing file system item.

- bool IsEqualTo (const FSObject &inOther) const

  Returns whether the FSObject is equal to another FSObject.

- bool IsEqualTo (const FSRef &inFSRef) const

  Returns whether the FSObject is equal to a FSRef.

- OSStatus CompareTo (const FSObject &inOther) const

  Compares the FSObject with another FSObject.

- OSStatus CompareTo (const FSRef &inFSRef) const

Compares the FSObject with an FSRef.

- void GetName (HFSUniStr255 &outName) const

  Passes back the name of the file system item in a HFSUniStr255 variable.

- CFString GetName () const

  Returns the name of the item of the file system item.

- CFString GetPath (CFURLPathStyle inPathStyle=kCFURLPOSIXPathStyle) const

  Returns the path name for the file system item.

- FSVolumeRefNum GetVolume () const

  Returns the volume reference number for where the file system item resides.

- void GetParent (FSRef &outParentRef) const

  Passes back the FSRef for the parent directory of the file system item.

- void GetParent (FSObject &outParent) const

  Passes back an FSObject for the parent directory of the file system item.

- SInt32 GetParentDirID () const

  Returns the directory ID of the parent of the file system item.

- void GetFSSpec (FSSpec &outSpec, CFStringEncoding inNameEncoding=encoding_System) const

  Passes back an FSSpec for the file system item.

- CFURL GetURL () const

  Returns the URL for the FSObject.

- void GetCatalogInfo (FSCatalogInfoBitmap inWhichInfo, FSCatalogInfo &outCatInfo) const

  Pass back file system catalog information for the item.

- void SetCatalogInfo (FSCatalogInfoBitmap inWhichInfo, const FSCatalogInfo &inCatInfo)

  Sets file system catalog information for the item.

- bool IsFile () const

  Returns whether the item is a file.

- bool IsFolder () const

Returns whether the item is a folder.

- OSStatus CheckLock () const

  Returns the locked state of the file system item.

- void SetIsLocked (bool inLock)

  Set the locked state of the file system item.

- void GetFinderInfo (FinderInfo ∗outFinderInfo, ExtendedFinderInfo ∗outExtFinderInfo=nil, bool ∗outIsFolder=nil) const

  Psses back the Finder information for the item.

- void SetFinderInfo (const FinderInfo ∗inFinderInfo, const ExtendedFinderInfo ∗inExtFinderInfo=nil)

  Sets the Finder information for the item.

- UInt16 GetFinderFlags () const

  Returns the Finder flags for the item.

- void ChangeFinderFlags (bool inSetFlags, UInt32 inFlagsToChange)

  Changes the Finder flags for the item.

- void Rename (const HFSUniStr255 &inName, TextEncoding inEncodingHint=kTextEncodingUnknown)

  Changes the name of the file system item.

- void Rename (const CFString &inName, TextEncoding inEncodingHint=kTextEncodingUnknown)

  Changes the name of the file system item.

- void Delete ()

  Deletes the file system item.

- void DeleteContainer ()

  Deletes the file system item and all its contained files and folders.

- void DeleteContainerContents ()

  Deletes the files and folders contained within the item.

- void Update ()

  Queries the file system to update the internal state of the FSObject.

- void Invalidate ()

  Invalidates the state of the FSObject.

6.126.2 Constructor & Destructor Documentation

6.126.2.1 PPx::FSObject::FSObject (const FSRef & *inFSRef*)

Constructs from a file system FSRef.

Parameters:

inFSRef File system file reference

Definition at line 31 of file PPxFSObject.cp.

References Update().

6.126.2.2 PPx::FSObject::FSObject (const FSRef & *inParentRef*, const CFString & *inName*)

Constructs from a parent FSRef and CFString item name.

Parameters:

inParentRef FSRef for parent directory

inName Name of item as a CFString

Note:

The entity referred to by the parent and name does not have to currently exist

Definition at line 52 of file PPxFSObject.cp.

References Update().

6.126.2.3 PPx::FSObject::FSObject (const FSRef & *inParentRef*, const HFSUniStr255 & *inName*)

Constructs from a parent FSRef and HFSUniStr255 item name.

Parameters:

inParentRef FSRef for parent directory

inName Name of item as a HFSUniStr255

Note:

The entity referred to by the parent and name does not have to currently exist

Definition at line 75 of file PPxFSObject.cp.

References Update().

6.126.2.4 PPx::FSObject::FSObject (FSVolumeRefNum *inVRefNum*, SInt32 *inParentDirID*, const CFString & *inName*)

Contructs from a volume refnum, parent directory ID, and CFString item name.

Parameters:

inVRefNum Volume reference number

inParentDirID Parent directory ID

inName Name of the item as a CFString

Note:

The entity referred to by the parent and name does not have to currently exist

Definition at line 100 of file PPxFSObject.cp.

6.126.2.5 PPx::FSObject::FSObject (FSVolumeRefNum *inVRefNum*, SInt32 *inParentDirID*, const HFSUniStr255 & *inName*)

Contructs from a volume refnum, parent directory ID, and HFSUniStr255 item name.

Parameters:

inVRefNum Volume reference number

inParentDirID Parent directory ID

inName Name of the item as a HFSUniStr255

Note:

The entity referred to by the parent and name does not have to currently exist

Definition at line 125 of file PPxFSObject.cp.

6.126.2.6 PPx::FSObject::FSObject (const FSSpec & *inFSSpec*, CFStringEncoding *inNameEncoding* = encoding_System)

Constructs from a FSSpec.

Parameters:

inFSSpec FSSpec for the file system object

inNameEncoding Encoding system for file name in the FSSpec

Definition at line 142 of file PPxFSObject.cp.

6.126.2.7 PPx::FSObject::FSObject (CFURLRef *inURL*) `[explicit]`

Constructs from a Core Foundation URL reference.

Parameters:

inURL CF URL descrbing a file system entity

Definition at line 182 of file PPxFSObject.cp.

6.126.2.8 PPx::FSObject::FSObject (const CFString & *inAbsolutePath*, CFURLPathStyle *inPathStyle* = kCFURLPOSIXPathStyle)

Constructs from an absolute path name.

Parameters:

inAbsolutePath Absolute path name to a file system item

inPathStyle OS Path Style for URL (POSIX, HFS, Windows)

Definition at line 197 of file PPxFSObject.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.126.2.9 PPx::FSObject::FSObject (const CFString & *inRelativePath*, const FSRef & *inBaseDirectory*, CFURLPathStyle *inPathStyle* = kCFURLPOSIXPathStyle)

Constructs from a relative path name.

Parameters:

inRelativePath Relative path name to a file system item

inBaseDirectory Base directory of path

inPathStyle OS Path Style for URL (POSIX, HFS, Windows)

Definition at line 218 of file PPxFSObject.cp.

References PPx::CFObject< CFURLRef >::UseRef().

6.126.3 Member Function Documentation

6.126.3.1 void PPx::FSObject::ChangeFinderFlags (bool *inSetFlags*, UInt32 *inFlagsToChange*)

Changes the Finder flags for the item.

Parameters:

inSetFlags If true, set the designated flags. If false, clear the designate flags

inFlagsToChange Bit mask designating which flags to change

Definition at line 826 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

6.126.3.2 OSStatus PPx::FSObject::CheckLock () const

Returns the locked state of the file system item.

Returns:

Locked state of the file system item

Return values:

noErr Unlocked

fLckdErr Item is locked

wPrErr Item is on a read-only value

vLckdErr Item is on a locked volume

Definition at line 685 of file PPxFSObject.cp.

References PPx_Throw_, and UseRef().

6.126.3.3 OSStatus PPx::FSObject::CompareTo (const FSRef & *inFSRef*) const

Compares the FSObject with an FSRef.

Parameters:

inFSRef FSRef to which to compare

Returns:

Whether the FSObject refers to the same item as the FSRef

Return values:

noErr The items are equivalent. Any other value means not equivalent.

Definition at line 365 of file PPxFSObject.cp.

6.126.3.4 OSStatus PPx::FSObject::CompareTo (const FSObject & *inOther*) const

Compares the FSObject with another FSObject.

Parameters:

inOther FSObject to which to compare

Returns:

Whether the FSObjects are equivalent

Return values:

noErr The FSObject are equivalent. Any other value means not equivalent.

Definition at line 327 of file PPxFSObject.cp.

References mFSRef, mName, and mState.

Referenced by IsEqualTo().

6.126.3.5 void PPx::FSObject::Delete ()

Deletes the file system item.

Note:

The actual file system item is deleted, but the FSObject remains valid so you can use it recreate the item on disk at a later time

Definition at line 896 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::File::CreateOnDisk(), and PPx::File::DeleteOnDisk().

6.126.3.6 void PPx::FSObject::DeleteContainer ()

Deletes the file system item and all its contained files and folders.

Note:

Throws an exception if the item is not a directory

The actual file system item is deleted, but the FSObject remains valid so you can use it recreate the item on disk at a later time

Definition at line 928 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::Folder::DeleteOnDisk().

6.126.3.7 void PPx::FSObject::DeleteContainerContents ()

Deletes the files and folders contained within the item.

Note:

Throws an exception if the item is not a directory

Definition at line 958 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::Folder::DeleteContents().

6.126.3.8 bool PPx::FSObject::Exists () const `[inline]`

Returns whether the FSObject refers to an existing file system item.

Returns:

Whether the FSObject refers to an existing file system item

Definition at line 221 of file PPxFSObject.h.

Referenced by PPx::Folder::CreateOnDisk(), PPx::File::CreateOnDisk(), and UseRef().

6.126.3.9 void PPx::FSObject::GetCatalogInfo (FSCatalogInfoBitmap *inWhichInfo*, FSCatalogInfo & *outCatInfo*) const

Pass back file system catalog information for the item.

Parameters:

inWhichInfo Bit flags specifying which information to get

outCatInfo FSCatalogInfo struct filled in with requested info

Definition at line 606 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_, and UseRef().

Referenced by GetFSSpec(), IsFile(), IsFolder(), and SetIsLocked().

6.126.3.10 UInt16 PPx::FSObject::GetFinderFlags () const

Returns the Finder flags for the item.

Returns:

Finder flags for the item

Definition at line 807 of file PPxFSObject.cp.

References GetFinderInfo().

6.126.3.11 void PPx::FSObject::GetFinderInfo (FinderInfo ∗ *outFinderInfo*, ExtendedFinderInfo ∗ *outExtFinderInfo* = nil, bool ∗ *outIsFolder* = nil) const

Psses back the Finder information for the item.

Parameters:

outFinderInfo Pointer to FinderInfo struct

outExtFinderInfo Pointer to ExtendedFinderInfo struct

outIsFolder Whether the item is a folder

For any of the parameters, pass nil if you do not want that piece of infomration.

Definition at line 759 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

Referenced by GetFinderFlags().

6.126.3.12 void PPx::FSObject::GetFSSpec (FSSpec & *outFSSpec*, CFStringEncoding *inNameEncoding* = encoding_System) const

Passes back an FSSpec for the file system item.

Parameters:

outFSSpec FSSpec for the item

inNameEncoding Encoding to use for the file name

Definition at line 544 of file PPxFSObject.cp.

References GetCatalogInfo(), PPx::CFString::GetPascalString(), PPx_Throw_, and PPx_ThrowIfOSError_.

6.126.3.13 CFString PPx::FSObject::GetName () const

Returns the name of the item of the file system item.

Returns:

Name of the file system item

Definition at line 382 of file PPxFSObject.cp.

References PPx_Throw_, and PPx_ThrowIfOSError_.

6.126.3.14 void PPx::FSObject::GetName (HFSUniStr255 & *outName*) const

Passes back the name of the file system item in a HFSUniStr255 variable.

Parameters:

outName Name of the file system item

Definition at line 413 of file PPxFSObject.cp.

References PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by PPx::Folder::CreateOnDisk(), and PPx::File::CreateOnDisk().

6.126.3.15 void PPx::FSObject::GetParent (FSObject & *outParent*) const

Passes back an FSObject for the parent directory of the file system item.

Parameters:

outParent FSObject for the parent of the file system item

Definition at line 500 of file PPxFSObject.cp.

References GetParent(), mFSRef, and mState.

6.126.3.16 void PPx::FSObject::GetParent (FSRef & *outParentRef*) const

Passes back the FSRef for the parent directory of the file system item.

Parameters:

outParentRef FSRef for the parent directory of the item

Definition at line 475 of file PPxFSObject.cp.

References PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by PPx::Folder::CreateOnDisk(), PPx::File::CreateOnDisk(), GetParent(), and GetParentDirID().

6.126.3.17 SInt32 PPx::FSObject::GetParentDirID () const

Returns the directory ID of the parent of the file system item.

Returns:

Direcctor ID of the parent of the item

Definition at line 520 of file PPxFSObject.cp.

References GetParent(), and PPx_ThrowIfOSError_.

6.126.3.18 CFString PPx::FSObject::GetPath (CFURLPathStyle *inPathStyle* = kCFURLPOSIXPathStyle) const

Returns the path name for the file system item.

Parameters:

inPathStyle OS Path Style for URL (POSIX, HFS, Windows)

Returns:

Path name for the file system item

Definition at line 438 of file PPxFSObject.cp.

References PPx::CFURL::GetFileSystemPath(), and GetURL().

6.126.3.19 CFURL PPx::FSObject::GetURL () const

Returns the URL for the FSObject.

Returns:

The URL for the FSObject

Note:

The URL will have a nil reference (be invalid) if the FSObject is invalid

Definition at line 580 of file PPxFSObject.cp.

Referenced by GetPath().

6.126.3.20 FSVolumeRefNum PPx::FSObject::GetVolume () const

Returns the volume reference number for where the file system item resides.

Returns:

Volume reference number for where the item resides

Definition at line 456 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_, and UseRef().

Referenced by PPx::Folder::CreateOnDisk().

6.126.3.21 void PPx::FSObject::Invalidate ()

Invalidates the state of the FSObject.

Call if you know that the file system item no longer exists through some means external to this object. For example, if you delete the directory containing the item.

Definition at line 1031 of file PPxFSObject.cp.

References PPx::CFObject< CFStringRef >::FreeRef().

Referenced by PPx::Folder::Invalidate(), PPx::File::Invalidate(), and Update().

6.126.3.22 bool PPx::FSObject::IsEqualTo (const FSRef & *inFSRef*) const `[inline]`

Returns whether the FSObject is equal to a FSRef.

Parameters:

inFSRef FSRef to which to compare

Returns:

Whether the FSObject refers to the same item as the FSRef

Definition at line 253 of file PPxFSObject.h.

References CompareTo().

6.126.3.23 bool PPx::FSObject::IsEqualTo (const FSObject & *inOther*) const `[inline]`

Returns whether the FSObject is equal to another FSObject.

Parameters:

inOther FSObject to which to compare

Returns:

Whether the FSObject is equal to the other FSObject

Definition at line 237 of file PPxFSObject.h.

References CompareTo().

Referenced by PPx::Folder::IsEqualTo(), and PPx::File::IsEqualTo().

6.126.3.24 bool PPx::FSObject::IsFile () const

Returns whether the item is a file.

Returns:
Whether the item is a file

Definition at line 644 of file PPxFSObject.cp.

References GetCatalogInfo().

6.126.3.25 bool PPx::FSObject::IsFolder () const

Returns whether the item is a folder.

Returns:
Whether the item is a folder

Definition at line 662 of file PPxFSObject.cp.

References GetCatalogInfo().

6.126.3.26 bool PPx::FSObject::IsValid () const [inline]

Returns whether the FSObject refers to a valid file system item.

Returns:
Whether the FSObject refers to a valid file system item

Definition at line 207 of file PPxFSObject.h.

Referenced by PPx::File::UpdateLocation().

6.126.3.27 FSObject & PPx::FSObject::operator= (const FSRef & *inFSRef*)

Assigns FSObject from a FSRef.

Parameters:
inFSRef FSRef from which to assign

Returns:
Reference to this FSObject

Definition at line 288 of file PPxFSObject.cp.

References PPx::CFObject< CFStringRef >::FreeRef().

6.126.3.28 void PPx::FSObject::Rename (const CFString & *inName*, TextEncoding *inEncodingHint* = kTextEncodingUnknown)

Changes the name of the file system item.

Parameters:

inName New name for the item as a CFString

inEncodingHint Suggested text encoding to use when converting the name from unicode to another encoding

Definition at line 877 of file PPxFSObject.cp.

References Rename().

6.126.3.29 void PPx::FSObject::Rename (const HFSUniStr255 & *inName*, TextEncoding *inEncodingHint* = kTextEncodingUnknown)

Changes the name of the file system item.

Parameters:

inName New name for the item as a HFSUniStr255

inEncodingHint Suggested text encoding to use when converting the name from unicode to another encoding

Definition at line 846 of file PPxFSObject.cp.

References PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by Rename().

6.126.3.30 void PPx::FSObject::SetCatalogInfo (FSCatalogInfoBitmap *inWhichInfo*, const FSCatalogInfo & *inCatInfo*)

Sets file system catalog information for the item.

Parameters:

inWhichInfo Bit flags specifying which information to set

inCatInfo FSCatalogInfo struct containing data to set

Definition at line 625 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

Referenced by SetIsLocked().

6.126.3.31 void PPx::FSObject::SetFinderInfo (const FinderInfo ∗ *inFinderInfo*, const ExtendedFinderInfo ∗ *inExtFinderInfo* = nil)

Sets the Finder information for the item.

Parameters:

inFinderInfo Pointer to FinderInfo struct

inExtFinderInfo Pointer to ExtendedFinderInfo struct

For any of the parameters, pass nil if you do not want to set that piece of infomration.

Definition at line 790 of file PPxFSObject.cp.

References PPx_ThrowIfOSError_.

6.126.3.32 void PPx::FSObject::SetIsLocked (bool *inLock*)

Set the locked state of the file system item.

Parameters:

inLock Whether to lock or unlock the item

Definition at line 711 of file PPxFSObject.cp.

References GetCatalogInfo(), and SetCatalogInfo().

6.126.3.33 void PPx::FSObject::Update ()

Queries the file system to update the internal state of the FSObject.

Call if you think that the file system item may have been deleted or created by means external to this object. For example, via the actions of the user in the Finder.

Definition at line 976 of file PPxFSObject.cp.

References PPx::CFObject< CFStringRef >::FreeRef(), and Invalidate().

Referenced by FSObject(), PPx::Folder::UpdateLocation(), and PPx::File::UpdateLocation().

6.126.3.34 const FSRef & PPx::FSObject::UseRef () const

Returns a const reference to a FSRef.

Throws if the FSRef is not valid.

Definition at line 306 of file PPxFSObject.cp.

References Exists(), and PPx_ThrowIf_.

Referenced by CheckLock(), PPx::Folder::CreateOnDisk(), GetCatalogInfo(), PPx::File::GetTotalForkSizes(), GetVolume(), PPx::File::OpenDataFork(), and PPx::File::OpenResourceFork().

The documentation for this class was generated from the following files:

- PPxFSObject.h
- PPxFSObject.cp

6.127 PPx::FSVolumeRefNumStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.127.1 Detailed Description

Wrapper for FSVolumeRefNum.

Definition at line 163 of file PPxSysTypes.h.

The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.128 PPx::GrafPortSaver Class Reference

```
#include <PPxQuickdrawUtils.h>
```

6.128.1 Detailed Description

Saves, changes, and restores the current Quickdraw GrafPort.

Definition at line 23 of file PPxQuickdrawUtils.h.

Public Member Functions

- GrafPortSaver (GrafPtr inPort)

  Constructor.

- ~GrafPortSaver ()

  Desstructor.

6.128.2 Constructor & Destructor Documentation

6.128.2.1 PPx::GrafPortSaver::GrafPortSaver (GrafPtr *inPort*)

Constructor.

Parameters:

inPort GrafPtr to make the current port

Saves the current port upon entry, and sets the current port to the input GrafPtr if it is not nil

Definition at line 19 of file PPxQuickdrawUtils.cp.

6.128.2.2 PPx::GrafPortSaver::~GrafPortSaver ()

Desstructor.

Restores the port that was current when the contructor was called

Definition at line 37 of file PPxQuickdrawUtils.cp.

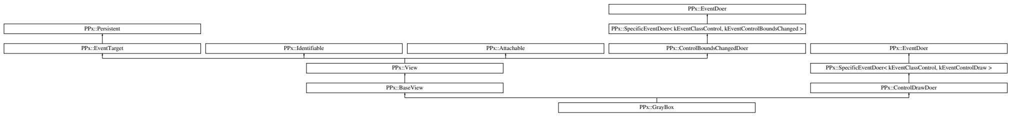
The documentation for this class was generated from the following files:

- PPxQuickdrawUtils.h
- PPxQuickdrawUtils.cp

6.129 PPx::GrayBox Class Reference

`#include <PPxGrayBox.h>`

Inheritance diagram for PPx::GrayBox::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::EventDoer
PPx::View
PPx::SpecificEventDoer< kEventClassControl, kEventControlDraw >
PPx::BaseView
PPx::ControlDrawDoer
PPx::GrayBox

6.129.1 Detailed Description

View which draws a gray box.

Definition at line 23 of file PPxGrayBox.h.

Public Member Functions

- GrayBox ()

  Default constructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, float inStrokeGray, float inStrokeAlpha, float inFillGray, float inFillAlpha, OptionBits inFeatures=PPx::features_None)

  Intializes from parameters.

Protected Member Functions

- virtual void InitState (const PPx::DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (PPx::DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus DoControlDraw (PPx::SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, RgnHandle inClipRgn, CGContextRef inContext)

  Draws the view.

6.129.2 Member Function Documentation

6.129.2.1 OSStatus PPx::GrayBox::DoControlDraw (PPx::SysCarbonEvent & *ioEvent*, ControlRef *inControl*, ControlPartCode *inPartCode*, RgnHandle *inClipRgn*, CGContextRef *inContext*) `[protected, virtual]`

Draws the view.

Parameters:

ioEvent CarbonEvent for control draw

inControl ControlRef for the view

inPartCode Part of the view to draw

inClipRgn Clipping region

inContext CGContext for drawing

Returns:

Status of drawing event. Always returns noErr.

Implements PPx::ControlDrawDoer.

Definition at line 150 of file PPxGrayBox.cp.

References PPx::View::GetLocalFrame().

6.129.2.2 void PPx::GrayBox::Initialize (PPx::View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, float *inStrokeGray*, float *inStrokeAlpha*, float *inFillGray*, float *inFillAlpha*, OptionBits *inFeatures* = PPx::features_None)

Intializes from parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inStrokeGray Gray level for drawing frame

inStrokeAlpha Alpha level for drawing frame

inFillGray Gray level for filling box

inFillAlpha Alpha level for filling box

inFeatures Control features supported by this view

Definition at line 49 of file PPxGrayBox.cp.

References PPx::BaseView::Initialize().

6.129.2.3 void PPx::GrayBox::InitState (const PPx::DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:
inReader Data dictionary from which to read persistent data

Definition at line 104 of file PPxGrayBox.cp.

References PPx::BaseView::InitState(), and PPx::DataReader::ReadOptional().

6.129.2.4 void PPx::GrayBox::WriteState (PPx::DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:
ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::BaseView.

Definition at line 124 of file PPxGrayBox.cp.

References PPx::BaseView::WriteState(), and PPx::DataWriter::WriteValue().

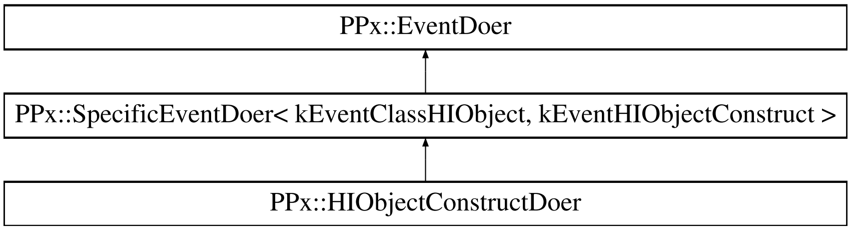
The documentation for this class was generated from the following files:

- PPxGrayBox.h
- PPxGrayBox.cp

6.130 PPx::HIObjectConstructDoer Class Reference

`#include <PPxHIObjectEvents.h>`

Inheritance diagram for PPx::HIObjectConstructDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassHIObject, kEventHIObjectConstruct >
PPx::HIObjectConstructDoer

6.130.1 Detailed Description

Handles constructing an HIObject.

Definition at line 20 of file PPxHIObjectEvents.h.

Protected Member Functions

- virtual OSStatus **DoHIObjectConstruct** (SysCarbonEvent &ioEvent, HIObjectRef inObjectRef)=0

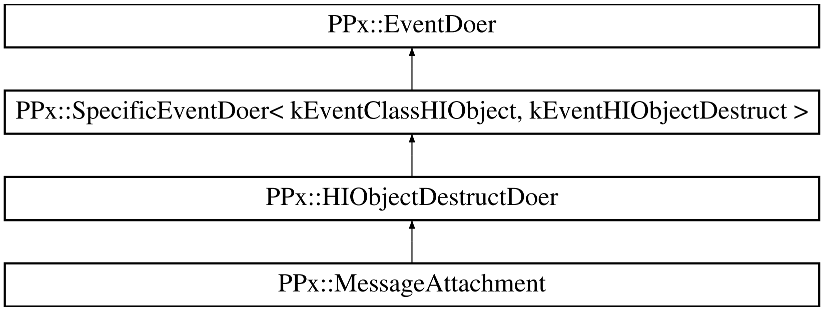
The documentation for this class was generated from the following files:

- PPxHIObjectEvents.h
- PPxHIObjectEvents.cp

6.131 PPx::HIObjectDestructDoer Class Reference

`#include <PPxHIObjectEvents.h>`

Inheritance diagram for PPx::HIObjectDestructDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassHIObject, kEventHIObjectDestruct >
PPx::HIObjectDestructDoer
PPx::MessageAttachment

6.131.1 Detailed Description

Handles destroying an HIObject.

Definition at line 50 of file PPxHIObjectEvents.h.

Protected Member Functions

- virtual OSStatus **DoHIObjectDestruct** (SysCarbonEvent &ioEvent)=0

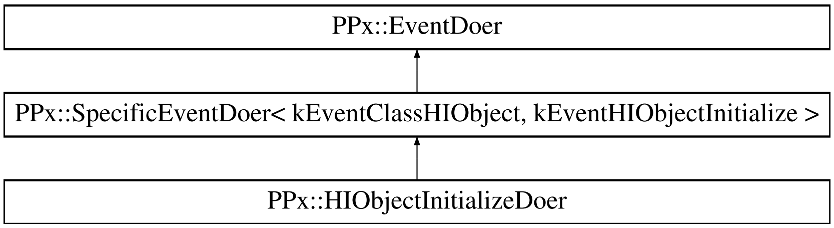
The documentation for this class was generated from the following files:

- PPxHIObjectEvents.h
- PPxHIObjectEvents.cp

6.132 PPx::HIObjectInitializeDoer Class Reference

`#include <PPxHIObjectEvents.h>`

Inheritance diagram for PPx::HIObjectInitializeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassHIObject, kEventHIObjectInitialize >
PPx::HIObjectInitializeDoer

6.132.1 Detailed Description

Handles initializing an HIObject.

Definition at line 36 of file PPxHIObjectEvents.h.

Protected Member Functions

- virtual OSStatus **DoHIObjectInitialize** (SysCarbonEvent &ioEvent)=0

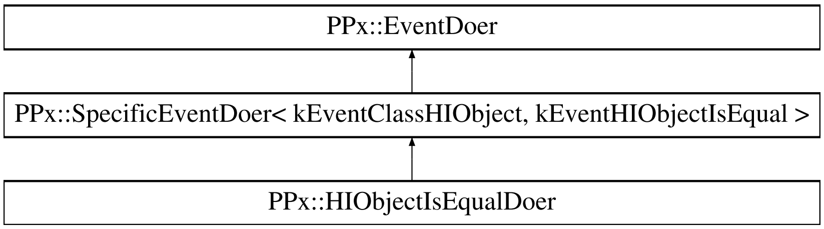
The documentation for this class was generated from the following files:

- PPxHIObjectEvents.h
- PPxHIObjectEvents.cp

6.133 PPx::HIObjectIsEqualDoer Class Reference

```
#include <PPxHIObjectEvents.h>
```

Inheritance diagram for PPx::HIObjectIsEqualDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassHIObject, kEventHIObjectIsEqual >
PPx::HIObjectIsEqualDoer

6.133.1 Detailed Description

Determines if an HIObject is eaual to another HIObject.

Definition at line 64 of file PPxHIObjectEvents.h.

Protected Member Functions

- virtual OSStatus **DoHIObjectIsEqual** (SysCarbonEvent &ioEvent, HIObjectRef inObjectRef)=0

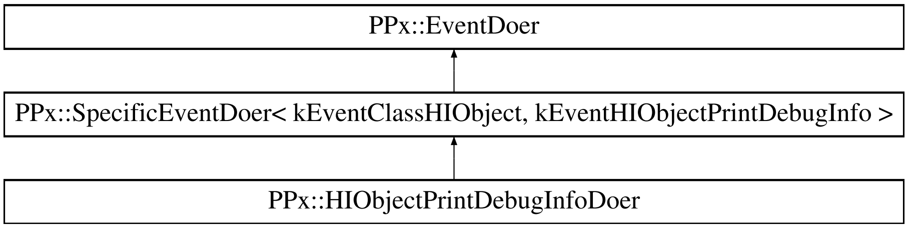
The documentation for this class was generated from the following files:

- PPxHIObjectEvents.h
- PPxHIObjectEvents.cp

6.134 PPx::HIObjectPrintDebugInfoDoer Class Reference

```
#include <PPxHIObjectEvents.h>
```

Inheritance diagram for PPx::HIObjectPrintDebugInfoDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassHIObject, kEventHIObjectPrintDebugInfo >
PPx::HIObjectPrintDebugInfoDoer

6.134.1 Detailed Description

Handles request to print debugging information.

Definition at line 80 of file PPxHIObjectEvents.h.

Protected Member Functions

- virtual OSStatus **DoHIObjectPrintDebugInfo** (SysCarbonEvent &ioEvent)=0

The documentation for this class was generated from the following files:

- PPxHIObjectEvents.h
- PPxHIObjectEvents.cp

6.135 PPx::HIObjectRefType< TType > Class Template Reference

```
#include <SysEventTypes.h>
```

6.135.1 Detailed Description

template<class TType> class PPx::HIObjectRefType< TType >

Template wrapper class for HIObjectRef types.

Definition at line 28 of file SysEventTypes.h.

Public Member Functions

- **HIObjectRefType** (HIObjectRef inRef)
- **operator HIObjectRef** () const
- **operator HIObjectRef &** ()
- HIObjectRef **Get** () const

The documentation for this class was generated from the following file:

- SysEventTypes.h

6.136 PPx::HIToolbarItemRefStruct Struct Reference

```
#include <SysEventTypes.h>
```

6.136.1 Detailed Description

Wrapper for HIToolbarItemRef.

Definition at line 55 of file SysEventTypes.h.

The documentation for this struct was generated from the following file:

- SysEventTypes.h

6.137 PPx::HIToolbarRefStruct Struct Reference

```
#include <SysEventTypes.h>
```

6.137.1 Detailed Description

Wrapper for HIToolbarRef.

Definition at line 48 of file SysEventTypes.h.

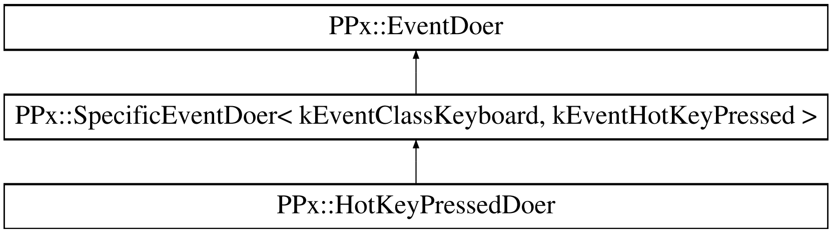
The documentation for this struct was generated from the following file:

- SysEventTypes.h

6.138 PPx::HotKeyPressedDoer Class Reference

`#include <PPxKeyboardEvents.h>`

Inheritance diagram for PPx::HotKeyPressedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassKeyboard, kEventHotKeyPressed >
PPx::HotKeyPressedDoer

6.138.1 Detailed Description

Handles a hot key being pressed.

Definition at line 93 of file PPxKeyboardEvents.h.

Protected Member Functions

- virtual OSStatus **DoHotKeyPressed** (SysCarbonEvent &ioEvent, const EventHotKeyID &inHotKey)=0

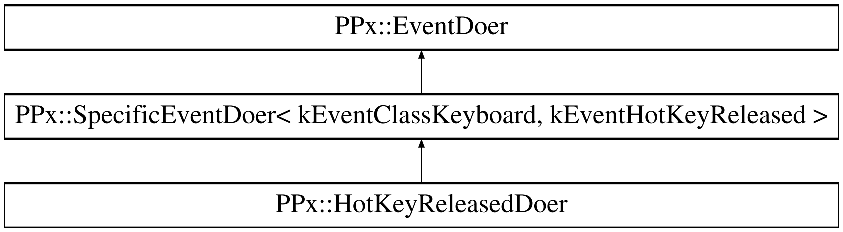
The documentation for this class was generated from the following files:

- PPxKeyboardEvents.h
- PPxKeyboardEvents.cp

6.139 PPx::HotKeyReleasedDoer Class Reference

```
#include <PPxKeyboardEvents.h>
```

Inheritance diagram for PPx::HotKeyReleasedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassKeyboard, kEventHotKeyReleased >
PPx::HotKeyReleasedDoer

6.139.1 Detailed Description

Handles a hot key being released.

Definition at line 109 of file PPxKeyboardEvents.h.

Protected Member Functions

- virtual OSStatus **DoHotKeyReleased** (SysCarbonEvent &ioEvent, const EventHotKeyID &inHotKey)=0

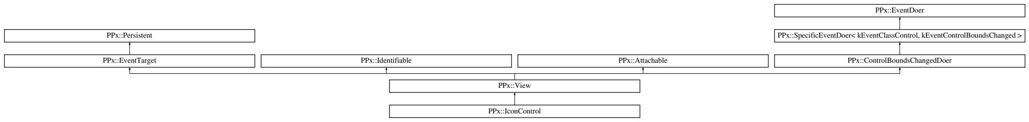
The documentation for this class was generated from the following files:

- PPxKeyboardEvents.h
- PPxKeyboardEvents.cp

6.140 PPx::IconControl Class Reference

`#include <PPxIconControl.h>`

Inheritance diagram for PPx::IconControl::



6.140.1 Detailed Description

A system icon control.

Definition at line 22 of file PPxIconControl.h.

Public Member Functions

- IconControl ()

  Default constructor.

- virtual ∼IconControl ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, const ControlButtonContentInfo &inContent, bool inDontTrack)

  Initialize from icon control creation parameters.

- void SetIconTransform (IconTransformType inTransform)

  Sets the icon transform.

- IconTransformType GetIconTransform () const

  Returns the icon transform.

- void SetIconAlignment (IconAlignmentType inAlignment)

  Sets the icon alignment.

- IconAlignmentType GetIconAlignment () const

  Returns the icon alignment.

- void SetIconResourceID (SInt16 inResID)

  Sets the resource ID for the icon.

- SInt16 GetIconResourceID () const

  Returns the resource ID for the icon.

- void SetContentInfo (const ControlButtonContentInfo &inContent)

  Sets the content inforomation.

- void GetContentInfo (ControlButtonContentInfo &outContent) const

  Passes back the content inforomation.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.140.2 Member Function Documentation

6.140.2.1 void PPx::IconControl::GetContentInfo (ControlButtonContentInfo & *outContent*) const

Passes back the content inforomation.

Parameters:

outContent Content inforomation

Definition at line 273 of file PPxIconControl.cp.

References PPx::View::GetDataTag().

6.140.2.2 IconAlignmentType PPx::IconControl::GetIconAlignment () const

Returns the icon alignment.

Returns:

Icon alignment

Definition at line 203 of file PPxIconControl.cp.

References PPx::View::GetDataTag().

6.140.2.3 SInt16 PPx::IconControl::GetIconResourceID () const

Returns the resource ID for the icon.

Returns:
Resource ID for the icon

Definition at line 238 of file PPxIconControl.cp.

References PPx::View::GetDataTag().

6.140.2.4 IconTransformType PPx::IconControl::GetIconTransform () const

Returns the icon transform.

Returns:
Icon transform

Definition at line 168 of file PPxIconControl.cp.

References PPx::View::GetDataTag().

6.140.2.5 void PPx::IconControl::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, const ControlButtonContentInfo & *inContent*, bool *inDontTrack*)

Initialize from icon control creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inContent Content of icon control

inDontTrack Whether to not track mouse downs in the control

Definition at line 54 of file PPxIconControl.cp.

6.140.2.6 void PPx::IconControl::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 94 of file PPxIconControl.cp.

References PPx::DataReader::ReadOptional().

6.140.2.7 void PPx::IconControl::SetContentInfo (const ControlButtonContentInfo & *inContent*)

Sets the content inforomation.

Parameters:

inContent Content inforomation

Definition at line 257 of file PPxIconControl.cp.

References PPx::View::SetDataTag().

6.140.2.8 void PPx::IconControl::SetIconAlignment (IconAlignmentType *inAlignment*)

Sets the icon alignment.

Parameters:

inAlignment Icon alignemnt

Definition at line 187 of file PPxIconControl.cp.

References PPx::View::SetDataTag().

6.140.2.9 void PPx::IconControl::SetIconResourceID (SInt16 *inResID*)

Sets the resource ID for the icon.

Parameters:

inResID Resource ID for the icon

Definition at line 222 of file PPxIconControl.cp.

References PPx::View::SetDataTag().

6.140.2.10 void PPx::IconControl::SetIconTransform (IconTransformType *inTransform*)

Sets the icon transform.

Parameters:

inTransform Icon transform

Definition at line 152 of file PPxIconControl.cp.

References PPx::View::SetDataTag().

6.140.2.11 void PPx::IconControl::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 121 of file PPxIconControl.cp.

References PPx::View::GetDataTag(), and PPx::DataWriter::WriteValue().

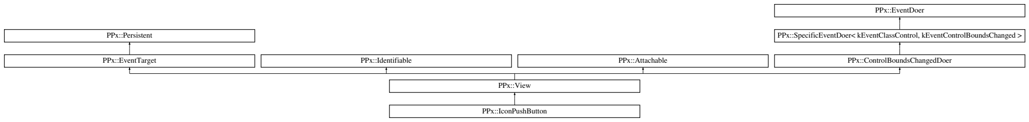
The documentation for this class was generated from the following files:

- PPxIconControl.h
- PPxIconControl.cp

6.141 PPx::IconPushButton Class Reference

```
#include <PPxIconPushButton.h>
```

Inheritance diagram for PPx::IconPushButton::



6.141.1 Detailed Description

A system push button with icon control.

Definition at line 22 of file PPxIconPushButton.h.

Public Member Functions

- IconPushButton ()

  Default constructor.

- virtual ∼IconPushButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, const ControlButtonContentInfo &inContent, ControlPushButtonIconAlignment inAlignment)

  Initialize from icon push button creation parameters.

- void SetDefaultFlag (bool inIsDefault)

  Sets whether this is the default button.

- bool GetDefaultFlag () const

  Returns whether this is the default button.

- void SetCancelFlag (bool inIsCancel)

  Sets whether this is the cancel button.

- bool GetCancelFlag () const

  Returns whether this is the cancel button.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

    Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

    Writes state to a data dictionary.

6.141.2 Member Function Documentation

6.141.2.1 bool PPx::IconPushButton::GetCancelFlag () const

Returns whether this is the cancel button.

Returns:
Whether this is the cancel button

Definition at line 207 of file PPxIconPushButton.cp.

References PPx::View::GetDataTag().

6.141.2.2 bool PPx::IconPushButton::GetDefaultFlag () const

Returns whether this is the default button.

Returns:
Whether this is the default button

Definition at line 170 of file PPxIconPushButton.cp.

References PPx::View::GetDataTag().

6.141.2.3 void PPx::IconPushButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, const ControlButtonContentInfo & *inContent*, ControlPushButtonIconAlignment *inAlignment*)

Initialize from icon push button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title for button

inContent Content of icon push button

inAlignment Alignement of icon in the button

Definition at line 57 of file PPxIconPushButton.cp.

6.141.2.4 void PPx::IconPushButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 99 of file PPxIconPushButton.cp.

References PPx::DataReader::ReadOptional().

6.141.2.5 void PPx::IconPushButton::SetCancelFlag (bool *inIsCancel*)

Sets whether this is the cancel button.

Parameters:

inIsCancel Whether this is the cancel button

Definition at line 189 of file PPxIconPushButton.cp.

References PPx::View::SetDataTag().

6.141.2.6 void PPx::IconPushButton::SetDefaultFlag (bool *inIsDefault*)

Sets whether this is the default button.

Parameters:

inIsDefault Whether this is the default button

Definition at line 152 of file PPxIconPushButton.cp.

References PPx::View::SetDataTag().

6.141.2.7 void PPx::IconPushButton::WriteState (DataWriter & *ioWriter*) const [protected, virtual]

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 124 of file PPxIconPushButton.cp.

References PPx::View::GetTitle(), and PPx::DataWriter::WriteValue().

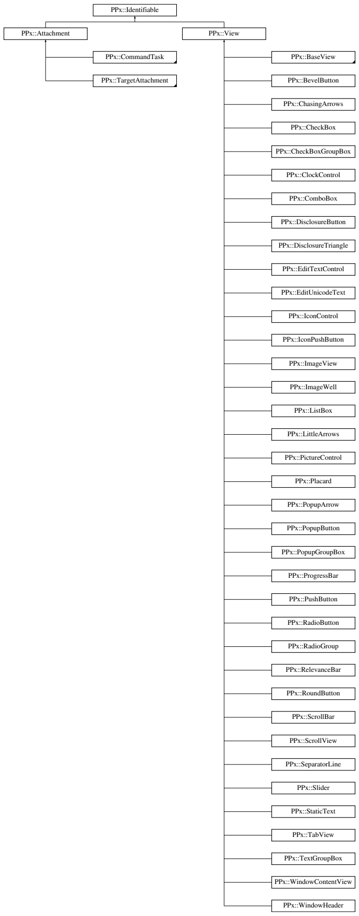
The documentation for this class was generated from the following files:

- PPxIconPushButton.h
- PPxIconPushButton.cp

6.142 PPx::Identifiable Class Reference

```
#include <PPxIdentifiable.h>
```

Inheritance diagram for PPx::Identifiable::



6.142.1 Detailed Description

Mix-in class for objects with an Object ID.

Definition at line 20 of file PPxIdentifiable.h.

Public Member Functions

- Identifiable ()

    Default constructor.

- virtual ∼Identifiable ()

    Destructor.

- void SetID (ObjectIDT inID)

    Sets the ID for an Identiable object.

- ObjectIDT GetID () const

    Returns the ID for an Identiable object.

- bool HasID (ObjectIDT inID) const

    Returns whether the Identiable object has the specified ID number.

6.142.2 Constructor & Destructor Documentation

6.142.2.1 PPx::Identifiable::Identifiable () `[inline]`

Default constructor.

Sets object ID to a default value.

Definition at line 43 of file PPxIdentifiable.h.

6.142.3 Member Function Documentation

6.142.3.1 ObjectIDT PPx::Identifiable::GetID () const `[inline]`

Returns the ID for an Identiable object.

Returns:

the ID for an Identiable object

Definition at line 85 of file PPxIdentifiable.h.

Referenced by PPx::View::FindConstViewByID(), PPx::View::WriteState(), and PPx::Attachment::WriteState().

6.142.3.2 bool PPx::Identifiable::HasID (ObjectIDT *inID*) const `[inline]`

Returns whether the Identiable object has the specified ID number.

Returns:
whether the Identiable object has the specified ID number

Definition at line 100 of file PPxIdentifiable.h.

6.142.3.3 void PPx::Identifiable::SetID (ObjectIDT *inID*) `[inline]`

Sets the ID for an Identiable object.

Parameters:
inID ID number to store for the Identifiable object

Definition at line 69 of file PPxIdentifiable.h.

Referenced by PPx::Attachment::InitState(), and PPx::View::InitViewState().

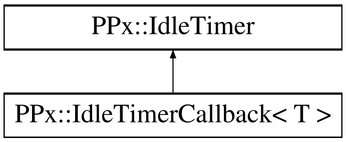
The documentation for this class was generated from the following file:

- PPxIdentifiable.h

6.143 PPx::IdleTimer Class Reference

`#include <PPxTimer.h>`

Inheritance diagram for PPx::IdleTimer::



6.143.1 Detailed Description

Abstract class for an Event Loop Idle Timer.

Idle timers fire only when there is no user activity, such as clicking, typing, and mouse down tracking, directed at the program.

Definition at line 76 of file PPxTimer.h.

Public Member Functions

- IdleTimer ()

  Default constructor.

- IdleTimer (EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval)

  Constructs and installs an Idle Timer.

- virtual ~IdleTimer ()

  Destructor.

- void Install (EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval)

  Installs an Idle Timer.

- void Remove ()

  Uninstalls an Idle Timer.

- bool IsTimerInstalled () const

  Returns whether an Idle Timer is currently installed on an event loop.

- void SetNextFireTime (EventTimerInterval inNextFire)

  Sets time delay until the Idle Timer next fires.

- void **Invoke** (EventLoopIdleTimerMessage inMessage)

6.143.2 Constructor & Destructor Documentation

6.143.2.1 PPx::IdleTimer::IdleTimer (EventLoopRef *inEventLoop*, EventTimerInterval *inFireDelay*, EventTimerInterval *inInterval*)

Constructs and installs an Idle Timer.

Parameters:

inEventLoop Event loop on which to install the timer. Call ::GetMainEventLoop() for the main application event loop; call ::GetCurrentEventLoop() for the event loop of the current thread.

inFireDelay Time, in seconds, to delay before first call

inInterval Time, in seconds, between timer calls

Definition at line 219 of file PPxTimer.cp.

References Install().

6.143.3 Member Function Documentation

6.143.3.1 void PPx::IdleTimer::Install (EventLoopRef *inEventLoop*, EventTimerInterval *inFireDelay*, EventTimerInterval *inInterval*)

Installs an Idle Timer.

Call this function to re-install a timer that you have previously removed.

Parameters:

inEventLoop Event loop on which to install the timer

inFireDelay Time, in seconds, to delay before first call

inInterval Time, in seconds, between timer calls

Definition at line 249 of file PPxTimer.cp.

References PPx::SysEventLoopIdleTimer::Install().

Referenced by IdleTimer().

6.143.3.2 bool PPx::IdleTimer::IsTimerInstalled () const

Returns whether an Idle Timer is currently installed on an event loop.

Returns:
Whether the Idle Timer is currently installed

Definition at line 279 of file PPxTimer.cp.

References PPx::SysEventLoopIdleTimer::IsInstalled().

6.143.3.3 void PPx::IdleTimer::Remove ()

Uninstalls an Idle Timer.

You can later call Install() to re-install it

Definition at line 265 of file PPxTimer.cp.

References PPx::SysEventLoopIdleTimer::Remove().

6.143.3.4 void PPx::IdleTimer::SetNextFireTime (EventTimerInterval *inNextFire*)

Sets time delay until the Idle Timer next fires.

This temporarily overrides the Timer's interval.

Parameters:
inNextFire Time, in seconds, until the Timer next fires

Definition at line 294 of file PPxTimer.cp.

References PPx_ThrowIfOSError_, and PPx::SysEventLoopIdleTimer::SetNextFireTime().

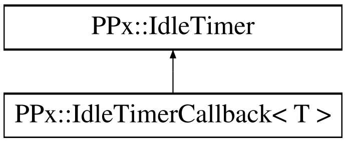
The documentation for this class was generated from the following files:

- PPxTimer.h
- PPxTimer.cp

6.144 PPx::IdleTimerCallback< T > Class Template Reference

```
#include <PPxTimer.h>
```

Inheritance diagram for PPx::IdleTimerCallback< T >::



6.144.1 Detailed Description

template<class T> class PPx::IdleTimerCallback< T >

Template class for an IdleTimer that calls an object member function.

Definition at line 179 of file PPxTimer.h.

Public Types

- typedef void(T::∗ **CallbackFunction**)()

Public Member Functions

- void **Install** (T ∗inObject, CallbackFunction inFunction, EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval)
- virtual void **DoIdleTimer** (EventLoopIdleTimerMessage inMessage)

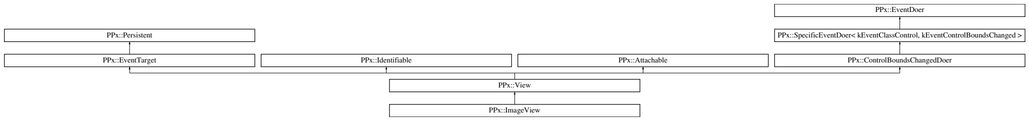
The documentation for this class was generated from the following file:

- PPxTimer.h

6.145 PPx::ImageView Class Reference

`#include <PPxImageView.h>`

Inheritance diagram for PPx::ImageView::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::ImageView

6.145.1 Detailed Description

A system view which displays a core graphics image.

Definition at line 22 of file PPxImageView.h.

Public Member Functions

- ImageView ()

  Default constructor.

- virtual ∼ImageView ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CGImageRef inImage)

  Initialize from image view creation parameters.

- void SetOpaque (bool inOpaque)

  Sets whether the image is opaque.

- bool IsOpaque () const

  Returns whether the image is opaque.

- void SetAlpha (Float32 inAlpha)

  Sets the alpha value for the image view.

- Float32 GetAlpha () const

  Returns the alpha value for the image view.

- void SetScaleToFit (bool inScaleToFit)

Sets whether to scale the image to fit in the frame.

- bool GetScaleToFit () const

  Returns whether the image scales to fit.

- void SetImage (CGImageRef inImage)

  Sets the image for the image view.

- CGImageRef CopyImage () const

  Returns a copy of the image that is in the.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.145.2 Member Function Documentation

6.145.2.1 CGImageRef PPx::ImageView::CopyImage () const

Returns a copy of the image that is in the.

Returns:

Copy of the image in the view

Definition at line 255 of file PPxImageView.cp.

References PPx::View::GetSysView().

6.145.2.2 Float32 PPx::ImageView::GetAlpha () const

Returns the alpha value for the image view.

Returns:

Alpha value for the iimage view

Definition at line 192 of file PPxImageView.cp.

References PPx::View::GetSysView().

Referenced by WriteState().

6.145.2.3 bool PPx::ImageView::GetScaleToFit () const

Returns whether the image scales to fit.

Returns:
Whether the image scales to fit

Definition at line 224 of file PPxImageView.cp.

References PPx::View::GetSysView().

Referenced by WriteState().

6.145.2.4 void PPx::ImageView::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CGImageRef *inImage*)

Initialize from image view creation parameters.

Parameters:
- ***inSuperView*** Parent view
- ***inFrame*** Bounds for view, in local coordinates of parent
- ***inVisible*** Whether the view is visible
- ***inEnabled*** Whether the view is enabled
- ***inImage*** CGImage to display

Definition at line 54 of file PPxImageView.cp.

6.145.2.5 void PPx::ImageView::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:
- ***inReader*** Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 90 of file PPxImageView.cp.

References PPx::DataReader::ReadOptional(), SetAlpha(), SetOpaque(), and SetScaleToFit().

6.145.2.6 bool PPx::ImageView::IsOpaque () const

Returns whether the image is opaque.

Returns:
Whether the image is opaque

Definition at line 161 of file PPxImageView.cp.

References PPx::View::GetSysView().

Referenced by WriteState().

6.145.2.7 void PPx::ImageView::SetAlpha (Float32 *inAlpha*)

Sets the alpha value for the image view.

Parameters:
inAlpha Alpha value for the iimage view

Definition at line 175 of file PPxImageView.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

Referenced by InitState().

6.145.2.8 void PPx::ImageView::SetImage (CGImageRef *inImage*)

Sets the image for the image view.

Parameters:
inImage CG image for the view

Definition at line 238 of file PPxImageView.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

6.145.2.9 void PPx::ImageView::SetOpaque (bool *inOpaque*)

Sets whether the image is opaque.

Parameters:
inOpaque Whether the image is opaque

Definition at line 144 of file PPxImageView.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

Referenced by InitState().

6.145.2.10 void PPx::ImageView::SetScaleToFit (bool *inScaleToFit*)

Sets whether to scale the image to fit in the frame.

Parameters:

inScaleToFit Whether to scale the image to fit

Definition at line 206 of file PPxImageView.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

Referenced by InitState().

6.145.2.11 void PPx::ImageView::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 121 of file PPxImageView.cp.

References GetAlpha(), GetScaleToFit(), IsOpaque(), and PPx::DataWriter::WriteValue().

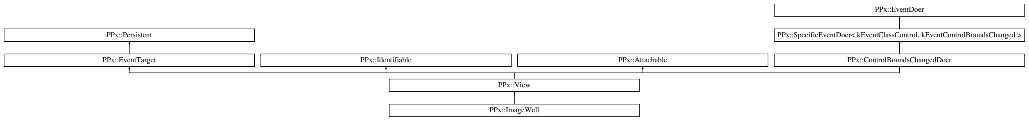
The documentation for this class was generated from the following files:

- PPxImageView.h
- PPxImageView.cp

6.146 PPx::ImageWell Class Reference

```
#include <PPxImageWell.h>
```

Inheritance diagram for PPx::ImageWell::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::ImageWell

6.146.1 Detailed Description

A system image well view.

Definition at line 22 of file PPxImageWell.h.

Public Member Functions

- ImageWell ()

  Default constructor.

- virtual ∼ImageWell ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, const ControlButtonContentInfo &inContent)

  Initialize from image well creation parameters.

- void SetContentInfo (const ControlButtonContentInfo &inContent)

  Sets the content inforomation.

- void GetContentInfo (ControlButtonContentInfo &outContent) const

  Passes back the content inforomation.

- void SetImageTransform (IconTransformType inTransform)

  Sets the image transform.

- IconTransformType GetImageTransform () const

  Returns the image transform.

- void SetDragDestinationFlag (bool inIsDragDestination)

Sets whether the image well is a drag destination.

- bool GetDragDestinationFlag () const

  Returns whether the image well is a drag destination.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.146.2 Member Function Documentation

6.146.2.1 void PPx::ImageWell::GetContentInfo (ControlButtonContentInfo & *outContent*) const

Passes back the content inforomation.

Parameters:

outContent Content inforomation

Definition at line 146 of file PPxImageWell.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.146.2.2 bool PPx::ImageWell::GetDragDestinationFlag () const

Returns whether the image well is a drag destination.

Returns:

Whether the image well is a drag destination

Definition at line 215 of file PPxImageWell.cp.

References PPx::View::GetDataTag().

6.146.2.3 IconTransformType PPx::ImageWell::GetImageTransform () const

Returns the image transform.

Returns:

Image transform

Definition at line 178 of file PPxImageWell.cp.

References PPx::View::GetDataTag().

6.146.2.4 void PPx::ImageWell::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, const ControlButtonContentInfo & *inContent*)

Initialize from image well creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inContent Content of image well

Definition at line 43 of file PPxImageWell.cp.

6.146.2.5 void PPx::ImageWell::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 79 of file PPxImageWell.cp.

References PPx::DataReader::ReadOptional().

6.146.2.6 void PPx::ImageWell::SetContentInfo (const ControlButtonContentInfo & *inContent*)

Sets the content inforomation.

Parameters:

inContent Content inforomation

Definition at line 130 of file PPxImageWell.cp.

References PPx::View::SetDataTag().

6.146.2.7 void PPx::ImageWell::SetDragDestinationFlag (bool *inIsDragDestination*)

Sets whether the image well is a drag destination.

Parameters:

inIsDragDestination Whether it is a drag destination

Definition at line 197 of file PPxImageWell.cp.

References PPx::View::SetDataTag().

6.146.2.8 void PPx::ImageWell::SetImageTransform (IconTransformType *inTransform*)

Sets the image transform.

Parameters:

inTransform Image transform

Definition at line 162 of file PPxImageWell.cp.

References PPx::View::SetDataTag().

6.146.2.9 void PPx::ImageWell::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 103 of file PPxImageWell.cp.

References GetContentInfo(), and PPx::DataWriter::WriteValue().

The documentation for this class was generated from the following files:

- PPxImageWell.h
- PPxImageWell.cp

6.147 PPx::IntegerType< TType, TValueType, defaultValue > Struct Template Reference

```
#include <PPxTypes.h>
```

6.147.1 Detailed Description

template<class TType, typename TValueType, TValueType defaultValue = 0> struct PPx::IntegerType< TType, TValueType, defaultValue >

Template which defines a class based on a built-in integer type.

Definition at line 42 of file PPxTypes.h.

Public Member Functions

- **IntegerType** (TValueType inValue)
- **operator TValueType** () const
- **operator TValueType &** ()
- TValueType **Get** () const

Public Attributes

- TValueType **mValue**

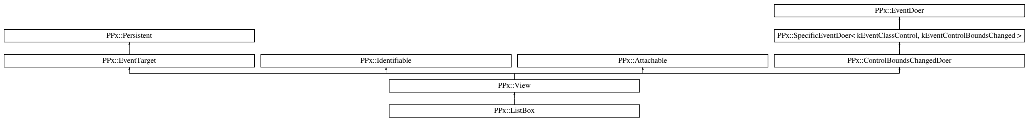
The documentation for this struct was generated from the following file:

- PPxTypes.h

6.148 PPx::ListBox Class Reference

```
#include <PPxListBox.h>
```

Inheritance diagram for PPx::ListBox::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::ListBox

6.148.1 Detailed Description

A system list box control.

Definition at line 22 of file PPxListBox.h.

Public Member Functions

- ListBox ()

  Default constructor.

- virtual ∼ListBox ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, bool inAutoSize, SInt16 inRowCount, SInt16 inColCount, bool inHorizScroll, bool inVertScroll, SInt16 inCellHeight, SInt16 inCellWidth, bool inHasGrowSpace, const ListDefSpec &inListDef)

  Initialize from list box creation parameters.

- ListHandle GetListHandle () const

  Returns ListHandle for the list box control.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID for the clock.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.148.2 Member Function Documentation

6.148.2.1 ListHandle PPx::ListBox::GetListHandle () const

Returns ListHandle for the list box control.

Returns:

ListHandle for the list box control

Definition at line 153 of file PPxListBox.cp.

References PPx::View::GetDataTag().

6.148.2.2 void PPx::ListBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, bool *inAutoSize*, SInt16 *inRowCount*, SInt16 *inColCount*, bool *inHorizScroll*, bool *inVertScroll*, SInt16 *inCellHeight*, SInt16 *inCellWidth*, bool *inHasGrowSpace*, const ListDefSpec & *inListDef*)

Initialize from list box creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inAutoSize Whether to automatically set the cell size

inRowCount Number of rows

inColCount Number of columns

inHorizScroll Whether list box has a horizontal scroll bar

inVertScroll Whether list box has a vertical scroll bar

inCellHeight Pixel height of cells

inCellWidth Pixel widht of cells

inHasGrowSpace Whether to leave space for a grow box

inListDef List defintion

Definition at line 52 of file PPxListBox.cp.

6.148.2.3 void PPx::ListBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 99 of file PPxListBox.cp.

6.148.2.4 void PPx::ListBox::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID for the clock.

Parameters:

inFont Theme font ID to use for text

Definition at line 172 of file PPxListBox.cp.

6.148.2.5 void PPx::ListBox::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 135 of file PPxListBox.cp.

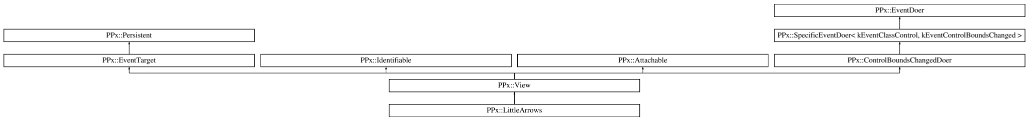
The documentation for this class was generated from the following files:

- PPxListBox.h
- PPxListBox.cp

6.149 PPx::LittleArrows Class Reference

`#include <PPxLittleArrows.h>`

Inheritance diagram for PPx::LittleArrows::



6.149.1 Detailed Description

A system little arrows control.

Definition at line 22 of file PPxLittleArrows.h.

Public Member Functions

- LittleArrows ()

  Default constructor.

- virtual ∼LittleArrows ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, SInt32 inIncrement)

  Initialize from little arrows creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.149.2 Member Function Documentation

6.149.2.1 void PPx::LittleArrows::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, SInt32 *inIncrement*)

Initialize from little arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inIncrement Amout to increment/decrement value when clicked

Definition at line 56 of file PPxLittleArrows.cp.

6.149.2.2 void PPx::LittleArrows::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 99 of file PPxLittleArrows.cp.

References PPx::DataReader::ReadOptional().

6.149.2.3 void PPx::LittleArrows::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 128 of file PPxLittleArrows.cp.

References PPx::View::GetMaxValue(), PPx::View::GetMinValue(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

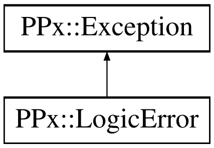
The documentation for this class was generated from the following files:

- PPxLittleArrows.h
- PPxLittleArrows.cp

6.150 PPx::LogicError Class Reference

`#include <PPxExceptions.h>`

Inheritance diagram for PPx::LogicError::



6.150.1 Detailed Description

Exception class for a programming error.

Logic errors are usually due to bugs in code, such as invalid function parameters or violations of assert conditions. Such errors should be found during testing.

Definition at line 179 of file PPxExceptions.h.

Public Member Functions

- LogicError (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Constructor.

Static Public Member Functions

- void Throw (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Throws an LogicError exception.

6.150.2 Constructor & Destructor Documentation

6.150.2.1 PPx::LogicError::LogicError (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*)

Constructor.

Parameters:

inWhat Kind of logic error

inWhy C string describing why the exception occurred

inWhere Source code location where exception was thrown

Note:

If PPx_Debug_Exceptions is false, the why and where are not stored.

Definition at line 313 of file PPxExceptions.cp.

6.150.3 Member Function Documentation

6.150.3.1 void PPx::LogicError::Throw (ExceptionIDT *inWhat*, const char * *inWhy*, const SourceLocation & *inWhere*) `[static]`

Throws an LogicError exception.

Parameters:

inWhat Kind of logic error

inWhy C string description of why the exception was thrown

inWhere Source location where exception was throw

Definition at line 333 of file PPxExceptions.cp.

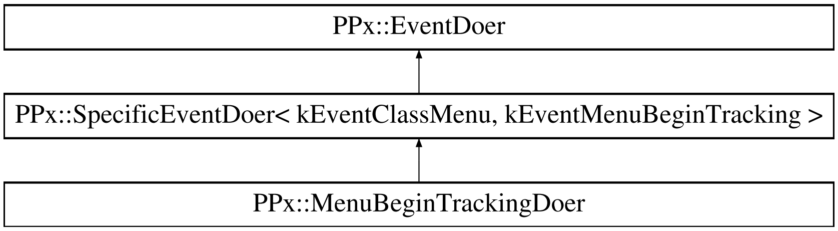
The documentation for this class was generated from the following files:

- PPxExceptions.h
- PPxExceptions.cp

6.151 PPx::MenuBeginTrackingDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuBeginTrackingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuBeginTracking >
PPx::MenuBeginTrackingDoer

6.151.1 Detailed Description

Handles the start of tracking the menubar or a pop-up menu.

Definition at line 20 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuBeginTracking** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, MenuTrackingMode inTrackingMode, UInt32 inMenuContext)=0

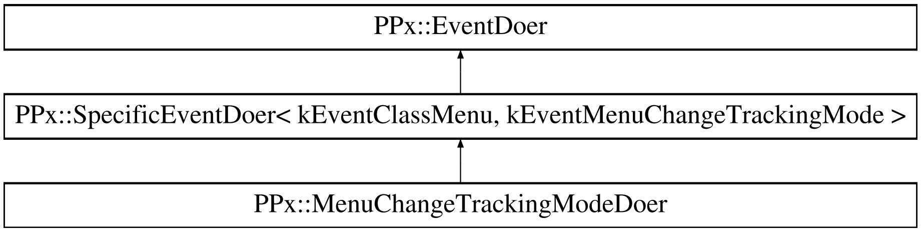
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.152 PPx::MenuChangeTrackingModeDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuChangeTrackingModeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuChangeTrackingMode >
PPx::MenuChangeTrackingModeDoer

6.152.1 Detailed Description

Handles changing between mouse and keyboard menu tracking modes.

Definition at line 55 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuChangeTrackingMode** (SysCarbonEvent &ioEvent, MenuTrackingMode inCurrentMode, MenuTrackingMode inNewMode, UInt32 inMenuContext)=0

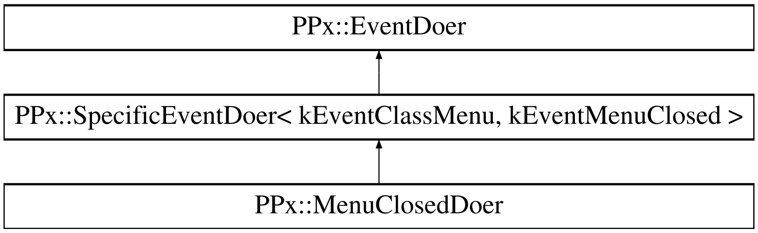
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.153 PPx::MenuClosedDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuClosedDoer::



6.153.1 Detailed Description

Handles a menu being closed.

Definition at line 91 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuClosed** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, UInt32 inMenuContext)=0

The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.154 PPx::MenuCommandStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.154.1 Detailed Description

Wrapper for MenuCommand.

Definition at line 94 of file PPxSysTypes.h.

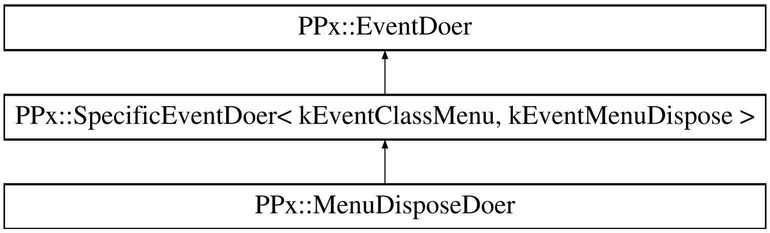
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.155 PPx::MenuDisposeDoer Class Reference

```
#include <PPxMenuEvents.h>
```

Inheritance diagram for PPx::MenuDisposeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuDispose >
PPx::MenuDisposeDoer

6.155.1 Detailed Description

Handles a menu being disposed.

Definition at line 264 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuDispose** (SysCarbonEvent &ioEvent, MenuRef inMenuRef)=0

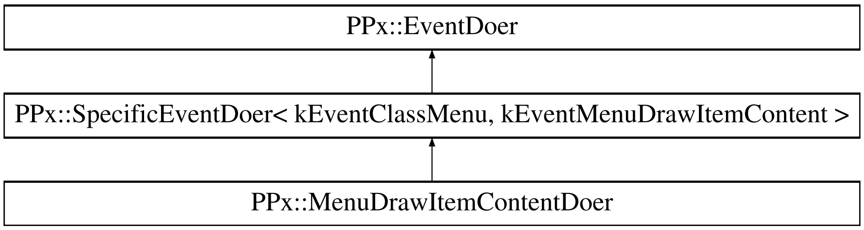
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.156 PPx::MenuDrawItemContentDoer Class Reference

```
#include <PPxMenuEvents.h>
```

Inheritance diagram for PPx::MenuDrawItemContentDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuDrawItemContent >
PPx::MenuDrawItemContentDoer

6.156.1 Detailed Description

Handles drawing the content of a menu item.

Definition at line 243 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuDrawItemContent** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, MenuItemIndex inItem, const Rect &inItemBounds, SInt16 inDeviceDepth, bool inDeviceHasColor, CGContextRef inCGContext)=0

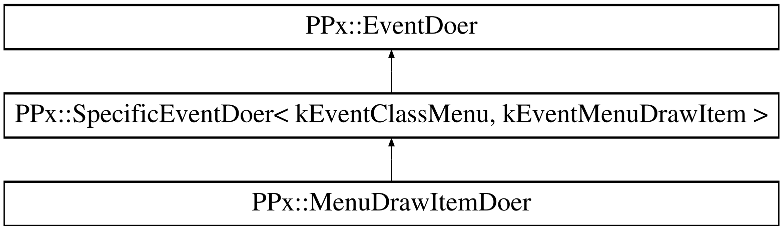
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.157 PPx::MenuDrawItemDoer Class Reference

```
#include <PPxMenuEvents.h>
```

Inheritance diagram for PPx::MenuDrawItemDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuDrawItem >
PPx::MenuDrawItemDoer

6.157.1 Detailed Description

Handles drawing a menu item.

Definition at line 219 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuDrawItem** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, const Rect &inCurrentBounds, MenuItemIndex inItem, const Rect &inItemBounds, SInt32 inVirtualTop, SInt32 inVirtualBottom, ThemeMenuState inDrawState, ThemeMenuItemType inItemType, CGContextRef inCGContext)=0

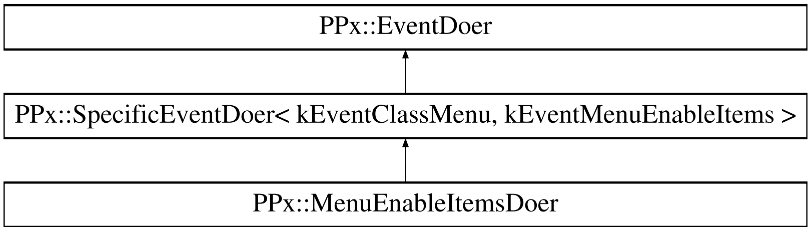
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.158 PPx::MenuEnableItemsDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuEnableItemsDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuEnableItems >
PPx::MenuEnableItemsDoer

6.158.1 Detailed Description

Handles enabling or disabling items in a menu.

Definition at line 147 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuEnableItems** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, bool inIsKeyEvent, UInt32 inMenuContext)=0

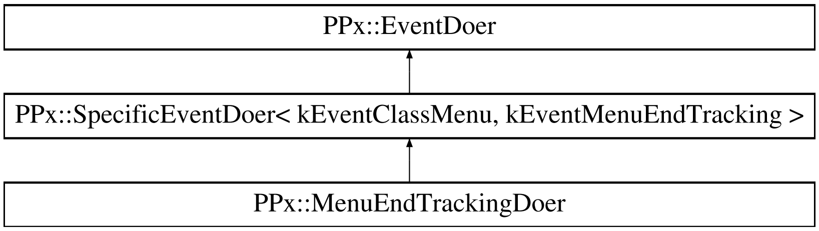
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.159 PPx::MenuEndTrackingDoer Class Reference

```
#include <PPxMenuEvents.h>
```

Inheritance diagram for PPx::MenuEndTrackingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuEndTracking >
PPx::MenuEndTrackingDoer

6.159.1 Detailed Description

Handles the end of tracking the menubar or a pop-up menu.

Definition at line 38 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuEndTracking** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, UInt32 inMenuContext)=0

The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.160 PPx::MenuEventOptionsStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.160.1 Detailed Description

Wrapper for MenuEventOptions.

Definition at line 101 of file PPxSysTypes.h.

The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.161 PPx::MenuItemIndexStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.161.1 Detailed Description

Wrapper for MenuItemIndex.

Definition at line 87 of file PPxSysTypes.h.

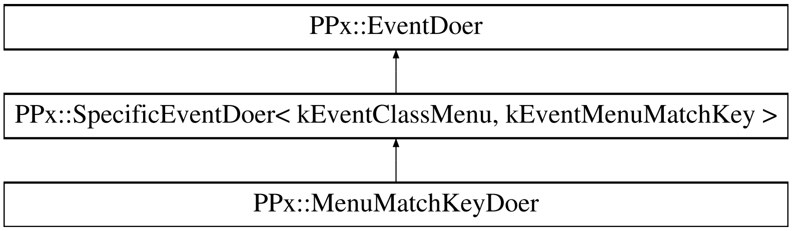
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.162 PPx::MenuMatchKeyDoer Class Reference

```
#include <PPxMenuEvents.h>
```

Inheritance diagram for PPx::MenuMatchKeyDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuMatchKey >
PPx::MenuMatchKeyDoer

6.162.1 Detailed Description

Returns menu item matching a command key equivalent.

Definition at line 127 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuMatchKey** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, EventRef inMenuEventRef, MenuEventOptions inOptions, UInt32 inMenuContext, MenuItemIndex &outItem)=0

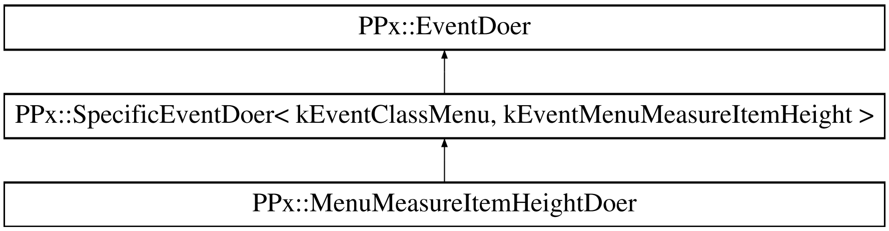
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.163 PPx::MenuMeasureItemHeightDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuMeasureItemHeightDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuMeasureItemHeight >
PPx::MenuMeasureItemHeightDoer

6.163.1 Detailed Description

Returns the height, in pixels, of a menu item.

Definition at line 201 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuMeasureItemHeight** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, MenuItemIndex inItem, SInt16 &outItemHeight)=0

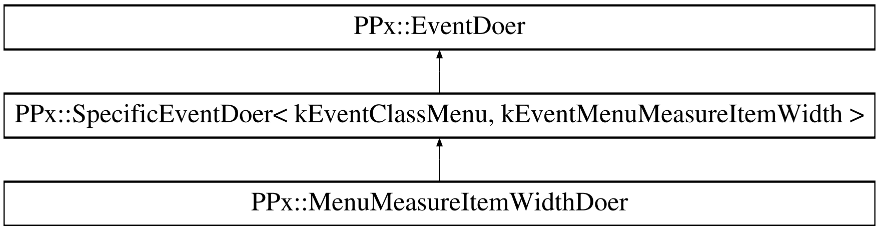
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.164 PPx::MenuMeasureItemWidthDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuMeasureItemWidthDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuMeasureItemWidth >
PPx::MenuMeasureItemWidthDoer

6.164.1 Detailed Description

Returns the width, in pixels, of a menu item.

Definition at line 183 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuMeasureItemWidth** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, MenuItemIndex inItem, SInt16 &outItemWidth)=0

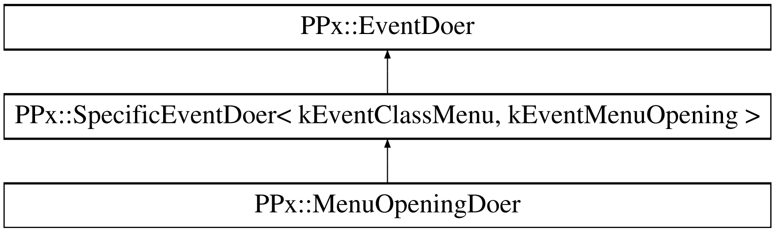
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.165 PPx::MenuOpeningDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuOpeningDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuOpening >
PPx::MenuOpeningDoer

6.165.1 Detailed Description

Handles a menu being opened (about to be displayed).

Definition at line 73 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuOpening** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, bool inFirstOpen, UInt32 inMenuContext)=0

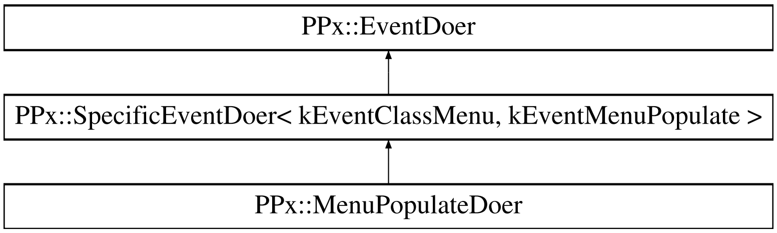
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.166 PPx::MenuPopulateDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuPopulateDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuPopulate >
PPx::MenuPopulateDoer

6.166.1 Detailed Description

Handles populating a menu with items prior to use.

Definition at line 165 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuPopulate** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, UInt32 inMenuContext, MenuCommand inCommand)=0

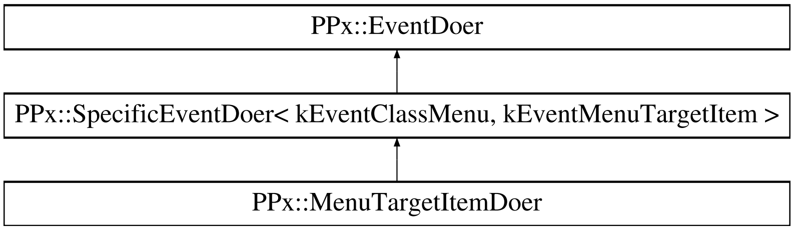
The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.167 PPx::MenuTargetItemDoer Class Reference

`#include <PPxMenuEvents.h>`

Inheritance diagram for PPx::MenuTargetItemDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMenu, kEventMenuTargetItem >
PPx::MenuTargetItemDoer

6.167.1 Detailed Description

Handles the mouse moving over a menu item.

Definition at line 108 of file PPxMenuEvents.h.

Protected Member Functions

- virtual OSStatus **DoMenuTargetItem** (SysCarbonEvent &ioEvent, MenuRef inMenuRef, MenuItemIndex inItem, MenuCommand inCommand, UInt32 inMenuContext)=0

The documentation for this class was generated from the following files:

- PPxMenuEvents.h
- PPxMenuEvents.cp

6.168 PPx::MenuTrackingModeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.168.1 Detailed Description

Wrapper for MenuTrackingMode.

Definition at line 79 of file PPxSysTypes.h.

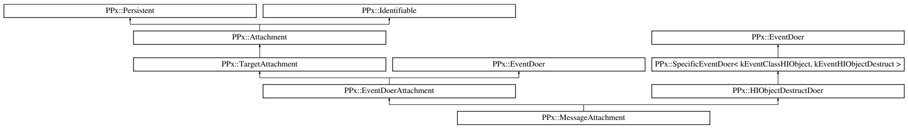
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.169 PPx::MessageAttachment Class Reference

```
#include <PPxEventAttachments.h>
```

Inheritance diagram for PPx::MessageAttachment::



6.169.1 Detailed Description

Attachment which responds to an event by sending a message event to another target.

The message recipient may be any event target.

Definition at line 122 of file PPxEventAttachments.h.

Public Member Functions

- void **Initialize** (EventTarget ∗inTarget, EventClassT inEventClass, EventKindT inEventKind, EventTarget ∗inMessageTarget, const SysCarbonEvent &inMessageEvent)
- void **SetMessageTarget** (EventTarget ∗inMessageTarget)

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus **DoHIObjectDestruct** (SysCarbonEvent &ioEvent)

6.169.2 Member Function Documentation

6.169.2.1 void PPx::MessageAttachment::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::EventDoerAttachment.

Definition at line 395 of file PPxEventAttachments.cp.

References PPx::SysCarbonEvent::MakeEvent(), PPx::DataReader::ReadObjectValue(), and PPx::DataReader::ReadRequired().

6.169.2.2 void PPx::MessageAttachment::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::EventDoerAttachment.

Definition at line 425 of file PPxEventAttachments.cp.

References PPx::SysCarbonEvent::GetEventClass(), PPx::SysCarbonEvent::GetEventKind(), PPx::DataWriter::WriteObjectValue(), and PPx::DataWriter::WriteValue().

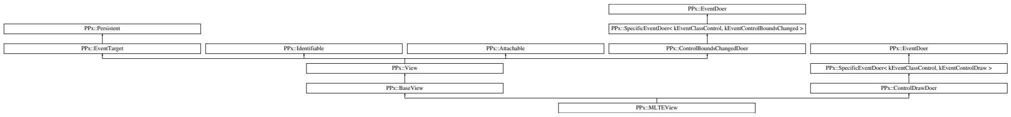
The documentation for this class was generated from the following files:

- PPxEventAttachments.h
- PPxEventAttachments.cp

6.170 PPx::MLTEView Class Reference

`#include <PPxMLTEView.h>`

Inheritance diagram for PPx::MLTEView::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::EventDoer
PPx::View
PPx::SpecificEventDoer< kEventClassControl, kEventControlDraw >
PPx::BaseView
PPx::ControlDrawDoer
PPx::MLTEView

6.170.1 Detailed Description

Text edit view base on MLTE.

Note:

Still under construction. Only displays text.

Definition at line 28 of file PPxMLTEView.h.

Public Member Functions

- void **Initialize** (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, TXNFrameOptions inFrameOptions, TXNFrameType inFrameType, TXNFileType inFileType, TXNPermanentTextEncodingType inEncodingType)
- TXNObject **GetTextObject** () const
- void **SetText** (CFStringRef inText)
- CFString **GetText** () const
- void **SetOneControlTag** (TXNControlTag inTag, TXNControlData inData)
- void **SetCGContext** (CGContextRef inCGContext)
- void **SetOneTypeAttribute** (TXNTypeAttributes ∗inAttribute, TXNOffset inStartOffset, TXNOffset inEndOffset)
- void **SetOneTypeAttributeForSelection** (TXNTypeAttributes ∗inAttribute)
- void **SetFontName** (ConstStringPtr inName)
- void **SetFontSize** (Fixed inSize)
- void **SetFontStyle** (Style inStyle)
- void **SetFontColor** (const RGBColor &inColor)

Static Public Member Functions

- void **InitializeSystem** (TXNInitOptions inOptions=0, const TXNMacOSPreferredFontDescription ∗inFonts=nil, ItemCount inFontCount=0)

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus **DoControlDraw** (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, RgnHandle inClipRgn, CGContextRef inContext)

6.170.2 Member Function Documentation

6.170.2.1 void PPx::MLTEView::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::BaseView.

Definition at line 364 of file PPxMLTEView.cp.

References PPx::BaseView::InitState(), and PPx::DataReader::ReadOptional().

6.170.2.2 void PPx::MLTEView::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::BaseView.

Definition at line 385 of file PPxMLTEView.cp.

References PPx::BaseView::WriteState(), and PPx::DataWriter::WriteValue().

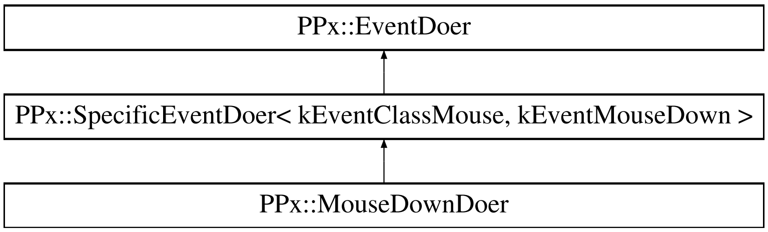
The documentation for this class was generated from the following files:

- PPxMLTEView.h
- PPxMLTEView.cp

6.171 PPx::MouseDownDoer Class Reference

```
#include <PPxMouseEvents.h>
```

Inheritance diagram for PPx::MouseDownDoer::



6.171.1 Detailed Description

Handles the mouse button being pressed.

Definition at line 20 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseDown** (SysCarbonEvent &ioEvent, const HIPoint &inMouseLoc)=0

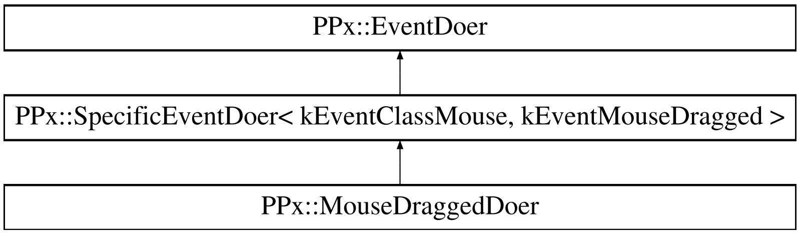
The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.172 PPx::MouseDraggedDoer Class Reference

```
#include <PPxMouseEvents.h>
```

Inheritance diagram for PPx::MouseDraggedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMouse, kEventMouseDragged >
PPx::MouseDraggedDoer

6.172.1 Detailed Description

Handles the mouse button being moved while the button is down.

Definition at line 68 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseDragged** (SysCarbonEvent &ioEvent, const HIPoint &inMouseLoc)=0

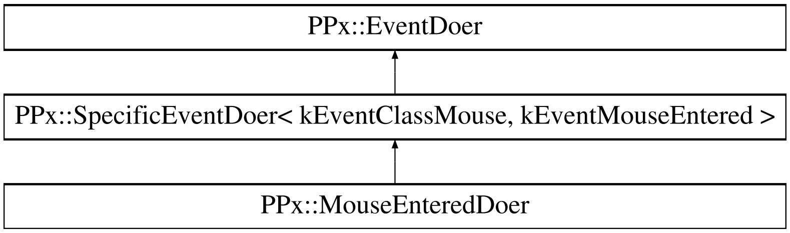
The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.173 PPx::MouseEnteredDoer Class Reference

```
#include <PPxMouseEvents.h>
```

Inheritance diagram for PPx::MouseEnteredDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMouse, kEventMouseEntered >
PPx::MouseEnteredDoer

6.173.1 Detailed Description

Handles the mouse entering a tracking area.

Definition at line 84 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseEntered** (SysCarbonEvent &ioEvent, MouseTrackingRef inTrackingRef, WindowRef inWindowRef, const HIPoint &inMouseLoc, const HIPoint &inWindowMouseLoc, UInt32 inKeyModifiers)=0

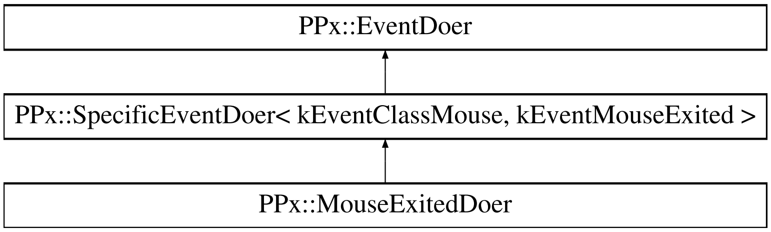
The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.174 PPx::MouseExitedDoer Class Reference

`#include <PPxMouseEvents.h>`

Inheritance diagram for PPx::MouseExitedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMouse, kEventMouseExited >
PPx::MouseExitedDoer

6.174.1 Detailed Description

Handles the mouse leaving a tracking area.

Definition at line 104 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseExited** (SysCarbonEvent &ioEvent, MouseTrackingRef inTrackingRef, WindowRef inWindowRef, const HIPoint &inMouseLoc, const HIPoint &inWindowMouseLoc, UInt32 inKeyModifiers)=0

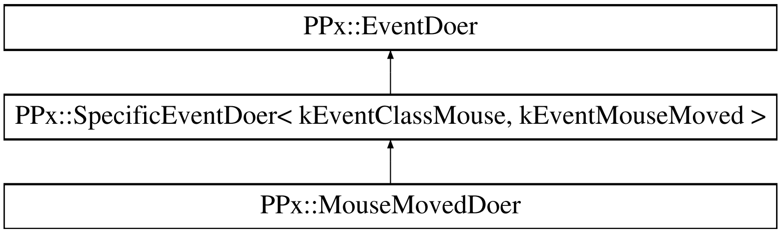
The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.175 PPx::MouseMovedDoer Class Reference

```
#include <PPxMouseEvents.h>
```

Inheritance diagram for PPx::MouseMovedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMouse, kEventMouseMoved >
PPx::MouseMovedDoer

6.175.1 Detailed Description

Handles the mouse button being moved.

Definition at line 52 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseMoved** (SysCarbonEvent &ioEvent, const HIPoint &inMouseLoc)=0

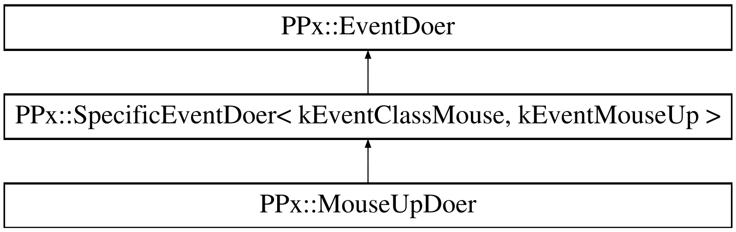
The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.176 PPx::MouseUpDoer Class Reference

```
#include <PPxMouseEvents.h>
```

Inheritance diagram for PPx::MouseUpDoer::



6.176.1 Detailed Description

Handles the mouse button being released.

Definition at line 36 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseUp** (SysCarbonEvent &ioEvent, const HIPoint &inMouseLoc)=0

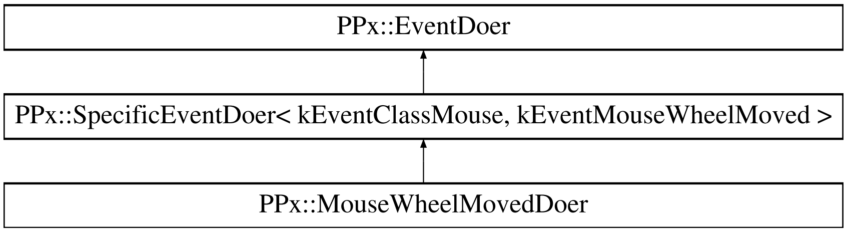
The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.177 PPx::MouseWheelMovedDoer Class Reference

```
#include <PPxMouseEvents.h>
```

Inheritance diagram for PPx::MouseWheelMovedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassMouse, kEventMouseWheelMoved >
PPx::MouseWheelMovedDoer

6.177.1 Detailed Description

Handles the mouse wheel being moved.

Definition at line 124 of file PPxMouseEvents.h.

Protected Member Functions

- virtual OSStatus **DoMouseWheelMoved** (SysCarbonEvent &ioEvent, WindowRef inWindowRef, const HIPoint &inMouseLoc, const HIPoint &inWindowMouseLoc, UInt32 inKeyModifiers, EventMouseWheelAxis inWheelAxis, SInt32 inWheelDelta)=0

The documentation for this class was generated from the following files:

- PPxMouseEvents.h
- PPxMouseEvents.cp

6.178 PPx::NavEventResponder Class Reference

```
#include <PPxNavServices.h>
```

6.178.1 Detailed Description

Abstract class for handling NavServices callbacks.

Definition at line 24 of file PPxNavServices.h.

Public Member Functions

- virtual ~NavEventResponder ()

  Destructor.

- void InvokeNavUserAction (NavCBRecPtr inParams)

  Non-virtual wrapper for calling DoNavUserAction function.

- void InvokeNavTerminate (NavCBRecPtr inParams)

  Non-virtual wrapper for calling DoNavTerminate function.

- void InvokeNavEventCallback (NavEventCallbackMessage inMessage, NavCBRecPtr inParams)

  Non-virtual wrapper for calling DoNavEventCallback function.

6.178.2 Member Function Documentation

6.178.2.1 void PPx::NavEventResponder::InvokeNavEventCallback (NavEventCallbackMessage *inMessage*, NavCBRecPtr *inParams*) `[inline]`

Non-virtual wrapper for calling DoNavEventCallback function.

Parameters:

inMessage NavServices callback message

inParams NavServices callback record

Definition at line 86 of file PPxNavServices.h.

6.178.2.2 void PPx::NavEventResponder::InvokeNavTerminate (NavCBRecPtr *inParams*) `[inline]`

Non-virtual wrapper for calling DoNavTerminate function.

Parameters:

inParams NavServices callback record

Definition at line 70 of file PPxNavServices.h.

6.178.2.3 void PPx::NavEventResponder::InvokeNavUserAction (NavCBRecPtr *inParams*) `[inline]`

Non-virtual wrapper for calling DoNavUserAction function.

Parameters:

inParams NavServices callback record

Definition at line 55 of file PPxNavServices.h.

The documentation for this class was generated from the following files:

- PPxNavServices.h
- PPxNavServices.cp

6.179 PPx::ObjectDescriptor Struct Reference

```
#include <PPxSerializer.h>
```

6.179.1 Detailed Description

Stores data describing a Persistent object.

Definition at line 45 of file PPxSerializer.h.

Public Attributes

- Persistent ∗ objectPtr

  Pointer to object.

- ObjectStorageIDT storageID

  ID number in descriptor list.

- CFString className

  Name of class as a string.

- AutoRefCount< KeyDataMap > objectState

  Dictionary of state data.

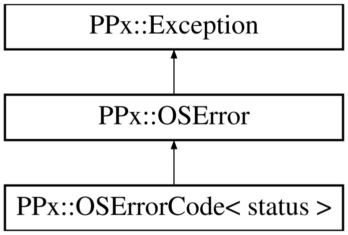
The documentation for this struct was generated from the following file:

- PPxSerializer.h

6.180 PPx::OSError Class Reference

`#include <PPxExceptions.h>`

Inheritance diagram for PPx::OSError::



6.180.1 Detailed Description

Exception class for a Mac OS error code.

An OSError is thrown when a Mac Toolbox routine returns an error code.

Definition at line 85 of file PPxExceptions.h.

Public Types

- typedef void(∗ ThrowFunc)(OSStatus, const char ∗, const SourceLocation &)

  Custom throw function signature.

Public Member Functions

- OSError (OSStatus inOSError, const char ∗inWhy, const SourceLocation &inWhere)

  Constructor.

- virtual StringPtr Why (Str255 outWhy) const

  Returns a Pascal string describing why an exception was thrown.

- OSStatus GetOSErrorCode () const

  Returns the OS error code which caused the exception.

Static Public Member Functions

- void Throw (OSStatus inOSError, const char ∗inWhy, const SourceLocation &inWhere)

  Throws an OSError exception.

- void SetThrowFunc (ThrowFunc inThrowFunc)

  Specifies the hook function to call when throwing an OSError exception.

6.180.2 Constructor & Destructor Documentation

6.180.2.1 PPx::OSError::OSError (OSStatus *inOSError*, const char ∗ *inWhy*, const SourceLocation & *inWhere*)

Constructor.

Parameters:

inOSError OS error code

inWhy C string describing why the exception occurred

inWhere Source code location where exception was thrown

Note:

If PPx_Debug_Exceptions is false, the why and where are not stored.

Definition at line 134 of file PPxExceptions.cp.

6.180.3 Member Function Documentation

6.180.3.1 OSStatus PPx::OSError::GetOSErrorCode () const

Returns the OS error code which caused the exception.

Returns:

OS error code

Definition at line 191 of file PPxExceptions.cp.

6.180.3.2 void PPx::OSError::SetThrowFunc (ThrowFunc *inThrowFunc*) `[static]`

Specifies the hook function to call when throwing an OSError exception.

Parameters:

inThrowFunc Pointer to custom throw function

Before throwing an exception when a Toolbox function returns an unexpected error code, PPx calls a user-supplied function which should throw an explicit instantiation of the template class PPx::OSErrorCode<OSStatus> for each OS error code which you wish to catch. See <MacErrors.h> for the list of OS error codes. For example,

```
void
MyThrowOSErrorCode(
    OSStatus                    inErrorCode,
    const char*                 inWhy,
    const PPx::SourceLocation&  inWhere)
{
    switch (inErrorCode) {

        case fnfErr:    // File not found
            PPx::ThrowOSErrorCode<fnfErr>(inWhy, inWhere);
            break;

        case opWrErr:   // File already open with write access
            PPx::ThrowOSErrorCode<opWrErr>(inWhy, inWhere);
            break;
    }
}
```

Then, somewhere in your code, register your custom throw function, and enter a try/catch block.

```
PPx::OSError::SetThrowFunc( MyThrowOSErrorCode );

try {
    // Do something that may throw exceptions

}

catch ( const PPx::OSErrorCode<fnfErr>& inErr ) {
    // File not found
    // Take specific recovery action for this kind of error
}

catch ( const PPx::OSErrorCode<opWrErr>& inErr ) {
    // File already open with write permission
    // Take specific recovery action for this kind of error
}

catch (...) {
    // Some other exceptions
    // Take generic recovery action
}
```

This allows you to take specific recovery actions for certain OS errors.

Definition at line 293 of file PPxExceptions.cp.

6.180.3.3 void PPx::OSError::Throw (OSStatus *inOSError*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[static]`

Throws an OSError exception.

Parameters:

inOSError OS error code

inWhy C string description of why the exception was thrown

inWhere Source location where exception was throw

Definition at line 207 of file PPxExceptions.cp.

6.180.3.4 StringPtr PPx::OSError::Why (Str255 *outWhy*) const `[virtual]`

Returns a Pascal string describing why an exception was thrown.

Parameters:

outWhy Pascal string in which to store the description

Note:

If PPx_Debug_Exceptions is false, description is an empty string

Reimplemented from PPx::Exception.

Definition at line 155 of file PPxExceptions.cp.

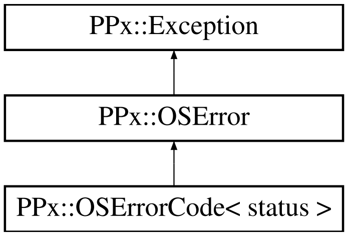
The documentation for this class was generated from the following files:

- PPxExceptions.h
- PPxExceptions.cp

6.181 PPx::OSErrorCode< status > Class Template Reference

`#include <PPxExceptions.h>`

Inheritance diagram for PPx::OSErrorCode< status >::



6.181.1 Detailed Description

template<OSStatus status> class PPx::OSErrorCode< status >

Template exception class for a specific Mac OS Error code.

The template parameter is the actual error code integer value. This creates a unique type which you can use for catching exceptions.

Definition at line 121 of file PPxExceptions.h.

Public Member Functions

- OSErrorCode (const char ∗inWhy, const SourceLocation &inWhere)

  Constuctor.

Static Public Member Functions

- void Throw (const char ∗inWhy, const SourceLocation &inWhere)

  Throws an OSErrorCode exception.

6.181.2 Constructor & Destructor Documentation

6.181.2.1 template<OSStatus status> PPx::OSErrorCode< status >::OSErrorCode (const char * *inWhy*, const SourceLocation & *inWhere*)

Constuctor.

Parameters:

inWhy A string describing the cause of the error

inWhere Location within the source of the exception

Definition at line 141 of file PPxExceptions.h.

6.181.3 Member Function Documentation

6.181.3.1 template<OSStatus status> void PPx::OSErrorCode< status >::Throw (const char * *inWhy*, const SourceLocation & *inWhere*) `[static]`

Throws an OSErrorCode exception.

Parameters:

inWhy A string describing the cause of the error

inWhere Location within the source of the caller

Definition at line 160 of file PPxExceptions.h.

The documentation for this class was generated from the following file:

- PPxExceptions.h

6.182 PPx::OSStatusStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.182.1 Detailed Description

Wrapper for OSStatus.

Definition at line 56 of file PPxSysTypes.h.

The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.183 PPx::OSTypeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.183.1 Detailed Description

Wrapper for OSType.

Definition at line 156 of file PPxSysTypes.h.

The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.184 PPx::OwnedPointer< T > Class Template Reference

`#include <PPxOwnedPointer.h>`

6.184.1 Detailed Description

template<class T> class PPx::OwnedPointer< T >

Template class which manages a pointer created via ”new”.

OwnedPointer objects own their underlying raw pointer, and delete it upon destruction or when adopting a different pointer. The owned object may be nil.

OwnedPointers maintain exclusive ownership of their underlying pointers. You cannot transfer ownership via copy construction or assignment.

OwnedPointer implements operators ∗ and ->, so you can use it with the same syntax as a raw pointer. As with raw pointers, attempts to use operator ∗ or -> on a nil owned object results in undefined behavior.

Definition at line 32 of file PPxOwnedPointer.h.

Public Member Functions

- OwnedPointer ()

  Default constructor.

- OwnedPointer (T ∗inPointer)

  Constructs from a raw pointer, which must have been created via a call to ”new” or is nil.

- ∼OwnedPointer ()

  Destructor.

- T ∗ Get () const

  Returns a pointer to the owned object.

- T ∗ operator → () const

  Returns a pointer to the owned object.

- T & operator ∗ () const

  Returns a reference to the owned object.

- void Reset ()

  Deletes existing owned object and sets owned object pointer to nil.

- void Reset (T ∗inPointer)

  Deletes existing owned object and takes ownership of input object pointer.

6.184.2 Constructor & Destructor Documentation

6.184.2.1 template<class T> PPx::OwnedPointer< T >::OwnedPointer (T ∗ *inPointer*) [explicit]

Constructs from a raw pointer, which must have been created via a call to ”new” or is nil.

Parameters:

inPointer Object takes ownership of this pointer

Definition at line 81 of file PPxOwnedPointer.h.

6.184.3 Member Function Documentation

6.184.3.1 template<class T> T ∗ PPx::OwnedPointer< T >::Get () const

Returns a pointer to the owned object.

Returns:

Pointer to the owned object

Definition at line 110 of file PPxOwnedPointer.h.

6.184.3.2 template<class T> T & PPx::OwnedPointer< T >::operator ∗ () const

Returns a reference to the owned object.

Returns:

Reference to the owned object

Definition at line 140 of file PPxOwnedPointer.h.

6.184.3.3 template<class T> T ∗ PPx::OwnedPointer< T >::operator → () const

Returns a pointer to the owned object.

Returns:
Pointer to the owned object

Definition at line 125 of file PPxOwnedPointer.h.

6.184.3.4 template<class T> void PPx::OwnedPointer< T >::Reset (T ∗ *inPointer*)

Deletes existing owned object and takes ownership of input object pointer.

Parameters:
inPointer Object takes ownership of this pointer

Definition at line 169 of file PPxOwnedPointer.h.

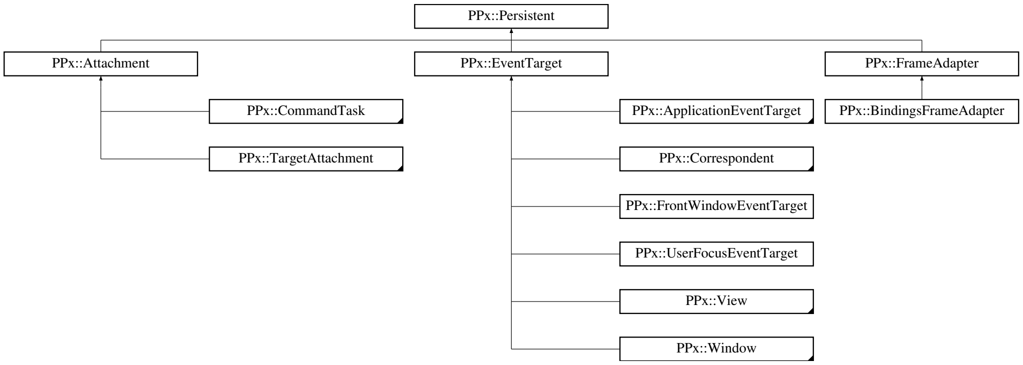
The documentation for this class was generated from the following file:

- PPxOwnedPointer.h

6.185 PPx::Persistent Class Reference

```
#include <PPxPersistent.h>
```

Inheritance diagram for PPx::Persistent::



6.185.1 Detailed Description

Abstract base class for persistent objects.

A persistent object can write its state to external data, and then later on be recreated from that data.

Definition at line 28 of file PPxPersistent.h.

Public Member Functions

- virtual ∼Persistent ()

  Destructor.

- CFStringRef GetClassName () const

  Returns name of class as a string.

- void InitPersistent (const DataReader &inReader)

  Initializes persistent object from a DataReader.

- void FinishInitPersistent ()

  Completes the initialization of a Peristent object.

- void WritePersistent (DataWriter &ioWriter) const

  Writes data of a persistent object to a DataWriter.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state of persistent object from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state of persistent object to a data dictionary.

6.185.2 Member Function Documentation

6.185.2.1 void PPx::Persistent::FinishInitPersistent () `[inline]`

Completes the initialization of a Peristent object.

This is a non-virtual wrapper function that calls the virtual FinishInit() function.

Definition at line 88 of file PPxPersistent.h.

6.185.2.2 void PPx::Persistent::InitPersistent (const DataReader & *inReader*) `[inline]`

Initializes persistent object from a DataReader.

This is a non-virtual wrapper function that calls the virtual InitState() function.

Definition at line 73 of file PPxPersistent.h.

References InitState().

6.185.2.3 void PPx::Persistent::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state of persistent object from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Note:

Subclasses with state data should override this function to read their own data, and call the inherited function

Reimplemented in PPx::CommandTask, PPx::Correspondent, PPx::TargetAttachment, PPx::EventDoerAttachment, PPx::ResponseAttachment, PPx::MessageAttachment, PPx::FrontWindowEventTarget, PPx::Attachment, PPx::Application, PPx::BaseView, PPx::WindowContentView, PPx::BevelButton, PPx::ChasingArrows, PPx::CheckBox, PPx::CheckBoxGroupBox, PPx::ClockControl, PPx::ComboBox, PPx::DisclosureButton, PPx::DisclosureTriangle, PPx::EditTextControl, PPx::EditUnicodeText, PPx::IconControl, PPx::IconPushButton, PPx::ImageView, PPx::ImageWell, PPx::ListBox, PPx::LittleArrows, PPx::PictureControl, PPx::Placard, PPx::PopupArrow, PPx::PopupButton, PPx::PopupGroupBox, PPx::ProgressBar, PPx::PushButton, PPx::RadioButton, PPx::RadioGroup, PPx::RelevanceBar, PPx::RoundButton, PPx::ScrollBar, PPx::ScrollView, PPx::SeparatorLine, PPx::Slider, PPx::StaticText, PPx::TabView, PPx::TextGroupBox, PPx::WindowHeader, PPx::MLTEView, PPx::ThemeTextBox, PPx::BindingsFrameAdapter, PPx::DrawerWindow, PPx::SheetAlert, and PPx::Window.

Definition at line 30 of file PPxPersistent.cp.

Referenced by InitPersistent().

6.185.2.4 void PPx::Persistent::WritePersistent (DataWriter & *ioWriter*) const `[inline]`

Writes data of a persistent object to a DataWriter.

This is a non-virtual wrapper function that calls the virtual WriteState() function.

Definition at line 102 of file PPxPersistent.h.

References WriteState().

Referenced by PPx::Serializer::ObjectsToDescriptors().

6.185.2.5 void PPx::Persistent::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state of persistent object to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Note:

Subclasses with state data should override this function to write their own data, and call the inherited function

Reimplemented in PPx::CommandTask, PPx::Correspondent, PPx::TargetAttachment, PPx::EventDoerAttachment, PPx::ResponseAttachment, PPx::MessageAttachment, PPx::FrontWindowEventTarget, PPx::Attachment, PPx::Application, PPx::BaseView, PPx::View, PPx::WindowContentView, PPx::GrayBox, PPx::BevelButton, PPx::CheckBox, PPx::CheckBoxGroupBox, PPx::ClockControl, PPx::ComboBox, PPx::DisclosureButton, PPx::DisclosureTriangle, PPx::EditTextControl, PPx::EditUnicodeText, PPx::IconControl, PPx::IconPushButton, PPx::ImageView, PPx::ImageWell, PPx::ListBox, PPx::LittleArrows, PPx::PictureControl, PPx::PopupArrow, PPx::PopupButton, PPx::PopupGroupBox, PPx::ProgressBar, PPx::PushButton, PPx::RadioButton, PPx::RelevanceBar, PPx::RoundButton, PPx::ScrollBar, PPx::ScrollView, PPx::Slider, PPx::StaticText, PPx::TabView, PPx::TextGroupBox, PPx::WindowHeader, PPx::MLTEView, PPx::ThemeTextBox, PPx::BindingsFrameAdapter, PPx::DrawerWindow, PPx::SheetAlert, and PPx::Window.

Definition at line 47 of file PPxPersistent.cp.

Referenced by WritePersistent().

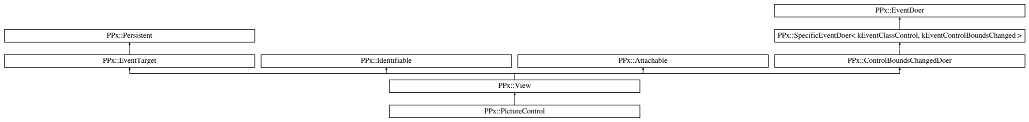
The documentation for this class was generated from the following files:

- PPxPersistent.h
- PPxPersistent.cp

6.186 PPx::PictureControl Class Reference

```
#include <PPxPictureControl.h>
```

Inheritance diagram for PPx::PictureControl::



6.186.1 Detailed Description

A system picture control.

Definition at line 22 of file PPxPictureControl.h.

Public Member Functions

- PictureControl ()

  Default constructor.

- virtual ∼PictureControl ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt16 inPictResID, PicHandle inPictureHandle, bool inDontTrack)

  Initialize from icon control creation parameters.

- void SetPicture (PicHandle inPicture)

  Sets the picture.

- PicHandle GetPicture () const

  Returns the PicHandle for the picture.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.186.2 Member Function Documentation

6.186.2.1 PicHandle PPx::PictureControl::GetPicture () const

Returns the PicHandle for the picture.

Returns:

PicHandle for the picture

Definition at line 157 of file PPxPictureControl.cp.

References PPx::View::GetDataTag().

6.186.2.2 void PPx::PictureControl::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt16 *inPictResID*, PicHandle *inPictureHandle*, bool *inDontTrack*)

Initialize from icon control creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inPictResID PICT resource ID of picture to display

inPictureHandle Handle to picture to dispaly

inDontTrack Whether not to track mouse downs

Definition at line 56 of file PPxPictureControl.cp.

6.186.2.3 void PPx::PictureControl::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 98 of file PPxPictureControl.cp.

References PPx::DataReader::ReadOptional().

6.186.2.4 void PPx::PictureControl::SetPicture (PicHandle *inPicture*)

Sets the picture.

Parameters:

inPicture PicHandle for the picture

Definition at line 139 of file PPxPictureControl.cp.

References PPx::View::SetDataTag().

6.186.2.5 void PPx::PictureControl::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 119 of file PPxPictureControl.cp.

References PPx::DataWriter::WriteValue().

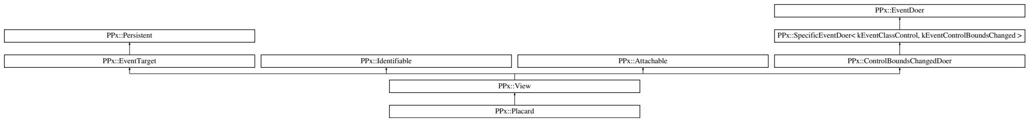
The documentation for this class was generated from the following files:

- PPxPictureControl.h
- PPxPictureControl.cp

6.187 PPx::Placard Class Reference

`#include <PPxPlacard.h>`

Inheritance diagram for PPx::Placard::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::Placard

6.187.1 Detailed Description

A system placard view.

Definition at line 22 of file PPxPlacard.h.

Public Member Functions

- Placard ()

  Default constructor.

- virtual ∼Placard ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled)

  Initialize from chasing arrows creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

6.187.2 Member Function Documentation

6.187.2.1 void PPx::Placard::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*)

Initialize from chasing arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

Definition at line 42 of file PPxPlacard.cp.

6.187.2.2 void PPx::Placard::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 77 of file PPxPlacard.cp.

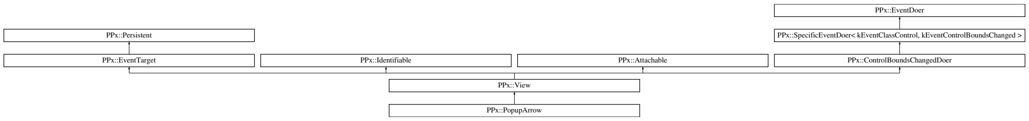
The documentation for this class was generated from the following files:

- PPxPlacard.h
- PPxPlacard.cp

6.188 PPx::PopupArrow Class Reference

`#include <PPxPopupArrow.h>`

Inheritance diagram for PPx::PopupArrow::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::PopupArrow

6.188.1 Detailed Description

A system popup arrow view.

Definition at line 22 of file PPxPopupArrow.h.

Public Member Functions

- PopupArrow ()

  Default constructor.

- virtual ∼PopupArrow ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, ControlPopupArrowOrientation inOrientation, ControlPopupArrowSize inArrowSize)

  Initialize from popup arrow creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.188.2 Member Function Documentation

6.188.2.1 void PPx::PopupArrow::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, ControlPopupArrowOrientation *inOrientation*, ControlPopupArrowSize *inArrowSize*)

Initialize from popup arrow creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inOrientation Direction arrow points (north, east, south, west)

inArrowSize Size of arrow (normal or small)

Definition at line 56 of file PPxPopupArrow.cp.

6.188.2.2 void PPx::PopupArrow::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 97 of file PPxPopupArrow.cp.

References PPx::DataReader::ReadOptional().

6.188.2.3 void PPx::PopupArrow::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 119 of file PPxPopupArrow.cp.

References PPx::DataWriter::WriteValue().

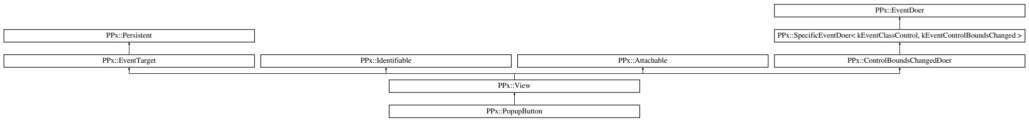
The documentation for this class was generated from the following files:

- PPxPopupArrow.h
- PPxPopupArrow.cp

6.189 PPx::PopupButton Class Reference

```
#include <PPxPopupButton.h>
```

Inheritance diagram for PPx::PopupButton::



6.189.1 Detailed Description

A system popup button control.

Definition at line 22 of file PPxPopupButton.h.

Public Member Functions

- PopupButton ()

  Default constructor.

- virtual ∼PopupButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, SInt32 inMenuID, bool inHasVariableWidth, SInt16 inTitleWidth, SInt16 inTitleJustification, Style inTitleStyle)

  Initialize from popup button creation parameters.

- void SetMenuRef (MenuRef inMenu)

  Sets the MenuRef for the popup menu.

- MenuRef GetMenuRef () const

  Returns the MenuRef for the popup menu.

- void SetOwnedMenuRef (MenuRef inMenu)

  Sets the MenuRef that the popup button owns.

- MenuRef GetOwnedMenuRef () const

  Returns the MenuRef that the popup button owns.

- void SetMenuID (SInt16 inMenuID)

  Sets the Menu ID for the popup menu.

- SInt16 GetMenuID () const

  Returns the Menu ID for the popup menu.

- void SetExtraHeight (SInt16 inExtraHeight)

  Sets the extra height for te popup button.

- SInt16 GetExtraHeight () const

  Returns the extra height for te popup button.

- void SetCheckCurrentItemFlag (bool inCheckIt)

  Sets whether to check the current item in the menu.

- bool GetCheckCurrentItemFlag () const

  Returns whether to check the current item in the menu.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.189.2 Member Function Documentation

6.189.2.1 bool PPx::PopupButton::GetCheckCurrentItemFlag () const

Returns whether to check the current item in the menu.

Returns:

Whether check the current menu item

Definition at line 334 of file PPxPopupButton.cp.

References PPx::View::GetDataTag().

6.189.2.2 SInt16 PPx::PopupButton::GetExtraHeight () const

Returns the extra height for te popup button.

Returns:
Extra height for te popup button

Definition at line 297 of file PPxPopupButton.cp.

References PPx::View::GetDataTag().

6.189.2.3 SInt16 PPx::PopupButton::GetMenuID () const

Returns the Menu ID for the popup menu.

Returns:
Menu ID

Definition at line 267 of file PPxPopupButton.cp.

6.189.2.4 MenuRef PPx::PopupButton::GetMenuRef () const

Returns the MenuRef for the popup menu.

Returns:
MenuRef

Definition at line 193 of file PPxPopupButton.cp.

References PPx::View::GetDataTag().

6.189.2.5 MenuRef PPx::PopupButton::GetOwnedMenuRef () const

Returns the MenuRef that the popup button owns.

Returns:
MenuRef

Definition at line 230 of file PPxPopupButton.cp.

References PPx::View::GetDataTag().

6.189.2.6 void PPx::PopupButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, SInt32 *inMenuID*, bool *inHasVariableWidth*, SInt16 *inTitleWidth*, SInt16 *inTitleJust*, Style *inTitleStyle*)

Initialize from popup button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title for popup

inMenuID Menu ID for popup

inHasVariableWidth Whether the menu has variable width

inTitleWidth Width of title (use -1 for variable width)

inTitleJust Justification of title text

inTitleStyle Font style for title

Definition at line 66 of file PPxPopupButton.cp.

6.189.2.7 void PPx::PopupButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 116 of file PPxPopupButton.cp.

References PPx::DataReader::ReadOptional(), and PPx::View::SetValue().

6.189.2.8 void PPx::PopupButton::SetCheckCurrentItemFlag (bool *inCheckIt*)

Sets whether to check the current item in the menu.

Parameters:

inCheckIt Whether check the current menu item

Definition at line 316 of file PPxPopupButton.cp.

References PPx::View::SetDataTag().

6.189.2.9 void PPx::PopupButton::SetExtraHeight (SInt16 *inExtraHeight*)

Sets the extra height for te popup button.

Parameters:

inExtraHeight Extra height for te popup button

Definition at line 281 of file PPxPopupButton.cp.

References PPx::View::SetDataTag().

6.189.2.10 void PPx::PopupButton::SetMenuID (SInt16 *inMenuID*)

Sets the Menu ID for the popup menu.

Parameters:

inMenuID Menu Id

Definition at line 249 of file PPxPopupButton.cp.

References PPx::View::SetDataTag().

6.189.2.11 void PPx::PopupButton::SetMenuRef (MenuRef *inMenu*)

Sets the MenuRef for the popup menu.

Parameters:

inMenu MenuRef

Definition at line 175 of file PPxPopupButton.cp.

References PPx::View::SetDataTag().

6.189.2.12 void PPx::PopupButton::SetOwnedMenuRef (MenuRef *inMenu*)

Sets the MenuRef that the popup button owns.

Parameters:

inMenu MenuRef

Definition at line 212 of file PPxPopupButton.cp.

References PPx::View::SetDataTag().

6.189.2.13 void PPx::PopupButton::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 149 of file PPxPopupButton.cp.

References PPx::View::GetTitle(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

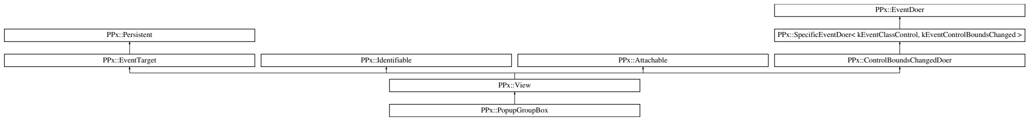
The documentation for this class was generated from the following files:

- PPxPopupButton.h
- PPxPopupButton.cp

6.190 PPx::PopupGroupBox Class Reference

`#include <PPxPopupGroupBox.h>`

Inheritance diagram for PPx::PopupGroupBox::



6.190.1 Detailed Description

A system group box with a popup menu title.

Definition at line 22 of file PPxPopupGroupBox.h.

Public Member Functions

- PopupGroupBox ()

  Default constructor.

- virtual ∼PopupGroupBox ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, bool inIsPrimary, SInt32 inMenuID, bool inHasVariableWidth, SInt16 inTitleWidth, SInt16 inTitleJust, Style inTitleStyle)

  Initialize from popup group box creation parameters.

- void SetMenuRef (MenuRef inMenu)

  Sets the MenuRef for the popup menu.

- MenuRef GetMenuRef () const

  Returns the MenuRef for the popup menu.

- void GetTitleRect (Rect &outTitleRect) const

  Passes back the title rectangle for the check box group box.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.190.2 Member Function Documentation

6.190.2.1 MenuRef PPx::PopupGroupBox::GetMenuRef () const

Returns the MenuRef for the popup menu.

Returns:
MenuRef

Definition at line 198 of file PPxPopupGroupBox.cp.

References PPx::View::GetDataTag().

6.190.2.2 void PPx::PopupGroupBox::GetTitleRect (Rect & *outTitleRect*) const

Passes back the title rectangle for the check box group box.

Parameters:
outTitleRect Title rectangle

Definition at line 217 of file PPxPopupGroupBox.cp.

References PPx::View::GetDataTag().

6.190.2.3 void PPx::PopupGroupBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, bool *inIsPrimary*, SInt32 *inMenuID*, bool *inHasVariableWidth*, SInt16 *inTitleWidth*, SInt16 *inTitleJust*, Style *inTitleStyle*)

Initialize from popup group box creation parameters.

Parameters:
inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title for popup

inIsPrimary Group box kind (true = primary, false = secondary)

inMenuID Menu ID for popup

inHasVariableWidth Whether the menu has variable width

inTitleWidth Width of title (use -1 for variable width)

inTitleJust Justification of title text

inTitleStyle Font style for title

Definition at line 68 of file PPxPopupGroupBox.cp.

6.190.2.4 void PPx::PopupGroupBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 119 of file PPxPopupGroupBox.cp.

References PPx::DataReader::ReadOptional(), and PPx::View::SetValue().

6.190.2.5 void PPx::PopupGroupBox::SetMenuRef (MenuRef *inMenu*)

Sets the MenuRef for the popup menu.

Parameters:

inMenu MenuRef

Definition at line 180 of file PPxPopupGroupBox.cp.

References PPx::View::SetDataTag().

6.190.2.6 void PPx::PopupGroupBox::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 153 of file PPxPopupGroupBox.cp.

References PPx::View::GetTitle(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

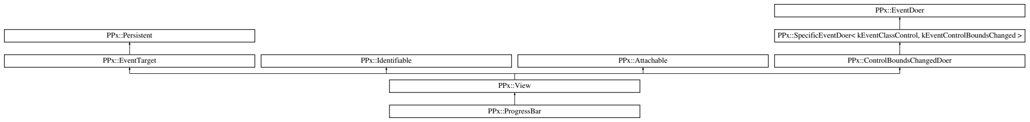
The documentation for this class was generated from the following files:

- PPxPopupGroupBox.h
- PPxPopupGroupBox.cp

6.191 PPx::ProgressBar Class Reference

`#include <PPxProgressBar.h>`

Inheritance diagram for PPx::ProgressBar::



6.191.1 Detailed Description

A system progress bar control.

Definition at line 22 of file PPxProgressBar.h.

Public Member Functions

- ProgressBar ()

  Default constructor.

- virtual ~ProgressBar ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, bool inIsIndeterminate)

  Initialize from popup button creation parameters.

- void SetIndeterminate (bool inIsIndeterminate)

  Sets whether the progress bar is indeterminate.

- bool IsIndeterminate () const

  Returns whether the progress bar is indeterminate.

- void SetAnimating (bool inIsAnimating)

  Sets the option for animating the progress bar.

- bool IsAnimating () const

  Returns whether the progress bar is animating.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.191.2 Member Function Documentation

6.191.2.1 void PPx::ProgressBar::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, bool *inIsIndeterminate*)

Initialize from popup button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inIsIndeterminate Whether progress bar is indeterminate

Definition at line 55 of file PPxProgressBar.cp.

6.191.2.2 void PPx::ProgressBar::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 96 of file PPxProgressBar.cp.

References PPx::DataReader::ReadOptional().

6.191.2.3 bool PPx::ProgressBar::IsAnimating () const

Returns whether the progress bar is animating.

Returns:
Whether the progress bar is animating

Definition at line 202 of file PPxProgressBar.cp.

References PPx::View::GetDataTag().

6.191.2.4 bool PPx::ProgressBar::IsIndeterminate () const

Returns whether the progress bar is indeterminate.

Returns:
Whether the progress bar is indeterminate

Definition at line 165 of file PPxProgressBar.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.191.2.5 void PPx::ProgressBar::SetAnimating (bool *inIsAnimating*)

Sets the option for animating the progress bar.

Parameters:
inIsAnimating Whether the progress bar should be animating

Definition at line 184 of file PPxProgressBar.cp.

References PPx::View::SetDataTag().

6.191.2.6 void PPx::ProgressBar::SetIndeterminate (bool *inIsIndeterminate*)

Sets whether the progress bar is indeterminate.

Parameters:
inIsIndeterminate Wwhether the progress bar is indeterminate

Definition at line 147 of file PPxProgressBar.cp.

References PPx::View::SetDataTag().

6.191.2.7 void PPx::ProgressBar::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 125 of file PPxProgressBar.cp.

References PPx::View::GetMaxValue(), PPx::View::GetMinValue(), PPx::View::GetValue(), IsIndeterminate(), and PPx::DataWriter::WriteValue().

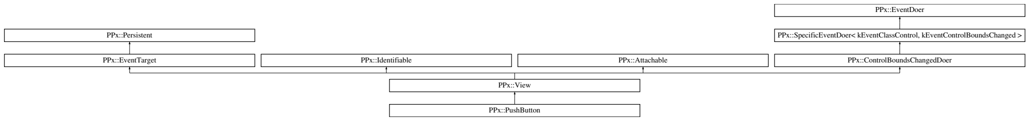
The documentation for this class was generated from the following files:

- PPxProgressBar.h
- PPxProgressBar.cp

6.192 PPx::PushButton Class Reference

`#include <PPxPushButton.h>`

Inheritance diagram for PPx::PushButton::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::PushButton

6.192.1 Detailed Description

A system push button control.

Definition at line 20 of file PPxPushButton.h.

Public Member Functions

- PushButton ()

  Default constructor.

- virtual ∼PushButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle)

  Initialize from push button creation parameters.

- void SetDefaultFlag (bool inIsDefault)

  Sets whether this is the default button.

- bool GetDefaultFlag () const

  Returns whether this is the default button.

- void SetCancelFlag (bool inIsCancel)

  Sets whether this is the cancel button.

- bool GetCancelFlag () const

  Returns whether this is the cancel button.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.192.2 Member Function Documentation

6.192.2.1 bool PPx::PushButton::GetCancelFlag () const

Returns whether this is the cancel button.

Returns:

Whether this is the cancel button

Definition at line 171 of file PPxPushButton.cp.

References PPx::View::GetDataTag().

6.192.2.2 bool PPx::PushButton::GetDefaultFlag () const

Returns whether this is the default button.

Returns:

Whether this is the default button

Definition at line 134 of file PPxPushButton.cp.

References PPx::View::GetDataTag().

6.192.2.3 void PPx::PushButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*)

Initialize from push button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Text title for button

Definition at line 43 of file PPxPushButton.cp.

6.192.2.4 void PPx::PushButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 78 of file PPxPushButton.cp.

References PPx::DataReader::ReadOptional().

6.192.2.5 void PPx::PushButton::SetCancelFlag (bool *inIsCancel*)

Sets whether this is the cancel button.

Parameters:

inIsCancel Whether this is the cancel button

Definition at line 153 of file PPxPushButton.cp.

References PPx::View::SetDataTag().

6.192.2.6 void PPx::PushButton::SetDefaultFlag (bool *inIsDefault*)

Sets whether this is the default button.

Parameters:

inIsDefault Whether this is the default button

Definition at line 116 of file PPxPushButton.cp.

References PPx::View::SetDataTag().

6.192.2.7 void PPx::PushButton::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 98 of file PPxPushButton.cp.

References PPx::View::GetTitle(), and PPx::DataWriter::WriteValue().

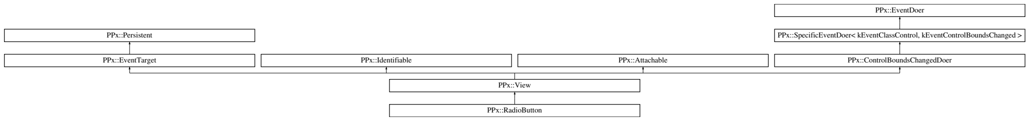
The documentation for this class was generated from the following files:

- PPxPushButton.h
- PPxPushButton.cp

6.193 PPx::RadioButton Class Reference

```
#include <PPxRadioButton.h>
```

Inheritance diagram for PPx::RadioButton::



6.193.1 Detailed Description

A system radio button control.

Definition at line 22 of file PPxRadioButton.h.

Public Member Functions

- RadioButton ()

  Default constructor.

- virtual ∼RadioButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, SInt32 inInitialValue, bool inAutoToggle)

  Intializes from redio button creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.193.2 Member Function Documentation

6.193.2.1 void PPx::RadioButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, SInt32 *inInitialValue*, bool *inAutoToggle*)

Intializes from redio button creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title of radio button

inInitialValue Initial value for radio button

inAutoToggle Whether button toggles automatically when clicked

Definition at line 47 of file PPxRadioButton.cp.

6.193.2.2 void PPx::RadioButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 88 of file PPxRadioButton.cp.

References PPx::DataReader::ReadOptional().

6.193.2.3 void PPx::RadioButton::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 114 of file PPxRadioButton.cp.

References PPx::View::GetTitle(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

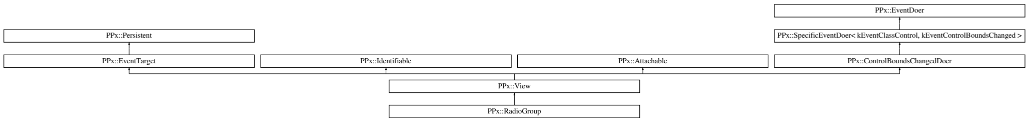
The documentation for this class was generated from the following files:

- PPxRadioButton.h
- PPxRadioButton.cp

6.194 PPx::RadioGroup Class Reference

```
#include <PPxRadioGroup.h>
```

Inheritance diagram for PPx::RadioGroup::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::RadioGroup

6.194.1 Detailed Description

A system radio group control.

Definition at line 22 of file PPxRadioGroup.h.

Public Member Functions

- RadioGroup ()

  Default constructor.

- virtual ∼RadioGroup ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled)

  Initialize from chasing arrows creation parameters.

- View ∗ GetCurrentButton () const

  Returns the current radio button.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

6.194.2 Member Function Documentation

6.194.2.1 View ∗ PPx::RadioGroup::GetCurrentButton () const

Returns the current radio button.

Returns:

Current radio button

Definition at line 95 of file PPxRadioGroup.cp.

References PPx::View::GetSysView(), PPx::View::GetValue(), and PPx::View::GetViewObject().

6.194.2.2 void PPx::RadioGroup::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*)

Initialize from chasing arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

Definition at line 42 of file PPxRadioGroup.cp.

6.194.2.3 void PPx::RadioGroup::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 77 of file PPxRadioGroup.cp.

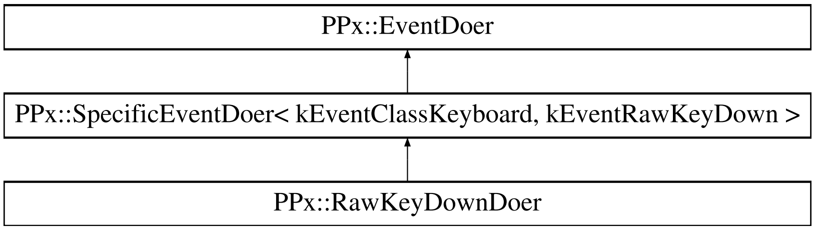
The documentation for this class was generated from the following files:

- PPxRadioGroup.h
- PPxRadioGroup.cp

6.195 PPx::RawKeyDownDoer Class Reference

```
#include <PPxKeyboardEvents.h>
```

Inheritance diagram for PPx::RawKeyDownDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassKeyboard, kEventRawKeyDown >
PPx::RawKeyDownDoer

6.195.1 Detailed Description

Handles a key being pressed.

Definition at line 20 of file PPxKeyboardEvents.h.

Protected Member Functions

- virtual OSStatus **DoRawKeyDown** (SysCarbonEvent &ioEvent, char inCharacter, UInt32 inKeyCode, UInt32 inKeyModifiers, UInt32 inKeyboardType)=0

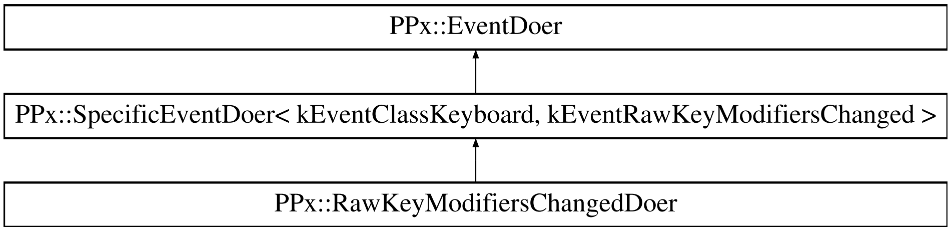
The documentation for this class was generated from the following files:

- PPxKeyboardEvents.h
- PPxKeyboardEvents.cp

6.196 PPx::RawKeyModifiersChangedDoer Class Reference

`#include <PPxKeyboardEvents.h>`

Inheritance diagram for PPx::RawKeyModifiersChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassKeyboard, kEventRawKeyModifiersChanged >
PPx::RawKeyModifiersChangedDoer

6.196.1 Detailed Description

Handles change in what modifier keys are pressed.

Definition at line 77 of file PPxKeyboardEvents.h.

Protected Member Functions

- virtual OSStatus **DoRawKeyModifiersChanged** (SysCarbonEvent &ioEvent, UInt32 inKeyModifiers)=0

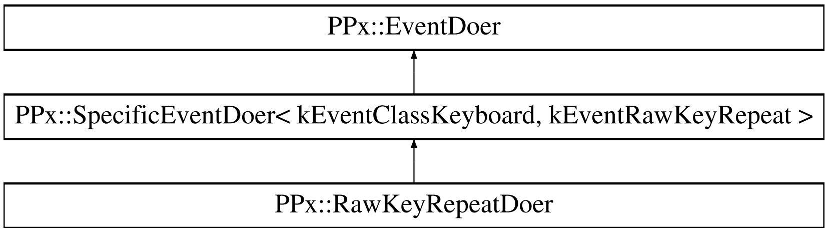
The documentation for this class was generated from the following files:

- PPxKeyboardEvents.h
- PPxKeyboardEvents.cp

6.197 PPx::RawKeyRepeatDoer Class Reference

```
#include <PPxKeyboardEvents.h>
```

Inheritance diagram for PPx::RawKeyRepeatDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassKeyboard, kEventRawKeyRepeat >
PPx::RawKeyRepeatDoer

6.197.1 Detailed Description

Handles a key being held down.

Definition at line 39 of file PPxKeyboardEvents.h.

Protected Member Functions

- virtual OSStatus **DoRawKeyRepeat** (SysCarbonEvent &ioEvent, char inCharacter, UInt32 inKeyCode, UInt32 inKeyModifiers, UInt32 inKeyboardType)=0

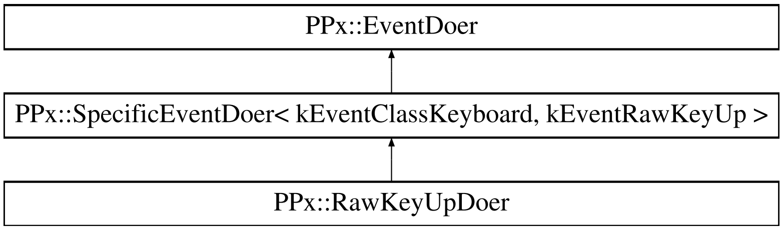
The documentation for this class was generated from the following files:

- PPxKeyboardEvents.h
- PPxKeyboardEvents.cp

6.198 PPx::RawKeyUpDoer Class Reference

```
#include <PPxKeyboardEvents.h>
```

Inheritance diagram for PPx::RawKeyUpDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassKeyboard, kEventRawKeyUp >
PPx::RawKeyUpDoer

6.198.1 Detailed Description

Handles a key being released.

Definition at line 58 of file PPxKeyboardEvents.h.

Protected Member Functions

- virtual OSStatus **DoRawKeyUp** (SysCarbonEvent &ioEvent, char inCharacter, UInt32 inKeyCode, UInt32 inKeyModifiers, UInt32 inKeyboardType)=0

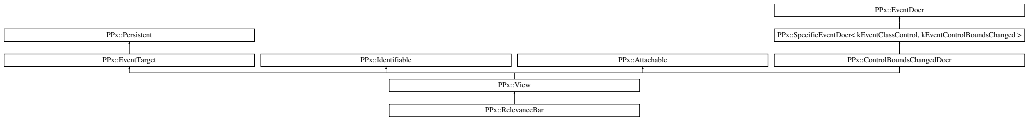
The documentation for this class was generated from the following files:

- PPxKeyboardEvents.h
- PPxKeyboardEvents.cp

6.199 PPx::RelevanceBar Class Reference

`#include <PPxRelevanceBar.h>`

Inheritance diagram for PPx::RelevanceBar::



6.199.1 Detailed Description

A system relevance bar control.

Definition at line 22 of file PPxRelevanceBar.h.

Public Member Functions

- RelevanceBar ()

  Default constructor.

- virtual ∼RelevanceBar ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue)

  Initialize from relevance bar creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.199.2 Member Function Documentation

6.199.2.1 void PPx::RelevanceBar::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*)

Initialize from relevance bar creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

Definition at line 45 of file PPxRelevanceBar.cp.

6.199.2.2 void PPx::RelevanceBar::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 84 of file PPxRelevanceBar.cp.

References PPx::DataReader::ReadOptional().

6.199.2.3 void PPx::RelevanceBar::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 110 of file PPxRelevanceBar.cp.

References PPx::View::GetMaxValue(), PPx::View::GetMinValue(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

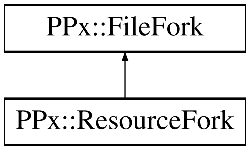
The documentation for this class was generated from the following files:

- PPxRelevanceBar.h
- PPxRelevanceBar.cp

6.200 PPx::ResourceFork Class Reference

```
#include <PPxResourceFork.h>
```

Inheritance diagram for PPx::ResourceFork::



6.200.1 Detailed Description

Wrapper class for the resource fork of a file.

Definition at line 21 of file PPxResourceFork.h.

Public Member Functions

- ResourceFork (SInt16 inRefNum, bool inOwnsRefNum)

  Constructs a ResourceFork object for an already open resource fork.

- ResourceFork (const FSRef &inFile, SInt8 inPermissions=fsRdWrPerm)

  Constructs a ResourceFork object for a file and opens the fork.

Static Public Member Functions

- const HFSUniStr255 & GetForkName ()

  Returns the constant system name for the resource fork.

6.200.2 Constructor & Destructor Documentation

6.200.2.1 PPx::ResourceFork::ResourceFork (SInt16 *inRefNum*, bool *inOwnsRefNum*)

Constructs a ResourceFork object for an already open resource fork.

Parameters:

inRefNum Reference number for the resource fork

inOwnsRefNum Whether this object should close the fork when finished

Definition at line 20 of file PPxResourceFork.cp.

6.200.2.2 PPx::ResourceFork::ResourceFork (const FSRef & *inFile*, SInt8 *inPermissions* = fsRdWrPerm)

Constructs a ResourceFork object for a file and opens the fork.

Parameters:

inFile FSRef for a file

inPermissions Access permissions

Definition at line 37 of file PPxResourceFork.cp.

6.200.3 Member Function Documentation

6.200.3.1 const HFSUniStr255 & PPx::ResourceFork::GetForkName () `[static]`

Returns the constant system name for the resource fork.

Returns:

Name of the resource fork

Definition at line 54 of file PPxResourceFork.cp.

References PPx_ThrowIfOSError_.

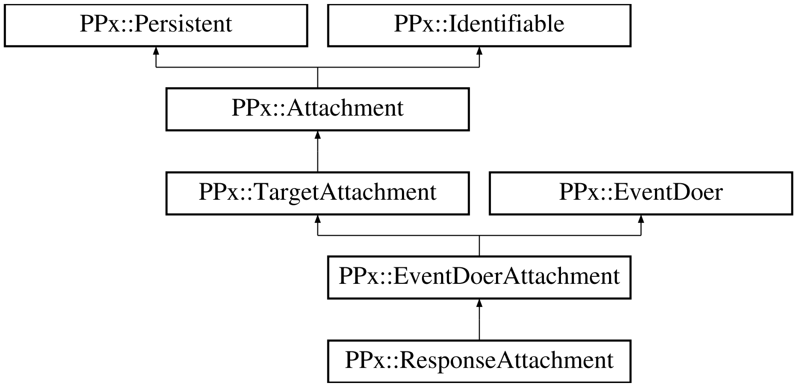
The documentation for this class was generated from the following files:

- PPxResourceFork.h
- PPxResourceFork.cp

6.201 PPx::ResponseAttachment Class Reference

`#include <PPxEventAttachments.h>`

Inheritance diagram for PPx::ResponseAttachment::



6.201.1 Detailed Description

Attachment which responds to an event by sending another event.

The target of the response event is either the same target, the current user focus, or the application.

Definition at line 85 of file PPxEventAttachments.h.

Public Member Functions

- void **Initialize** (EventTarget ∗inTarget, EventClassT inEventClass, EventKindT inEventKind, EMetaTarget inResponseTarget, const SysCarbonEvent &inResponseEvent)

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- EventTargetRef **GetResponseSysEventTarget** () const

6.201.2 Member Function Documentation

6.201.2.1 void PPx::ResponseAttachment::InitState (const DataReader & *inReader*) [protected, virtual]

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::EventDoerAttachment.

Definition at line 257 of file PPxEventAttachments.cp.

References PPx::EMetaTarget, PPx::SysCarbonEvent::MakeEvent(), PPx::DataReader::ReadOptional(), and PPx::DataReader::ReadRequired().

6.201.2.2 void PPx::ResponseAttachment::WriteState (DataWriter & *ioWriter*) const [protected, virtual]

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::EventDoerAttachment.

Definition at line 286 of file PPxEventAttachments.cp.

References PPx::SysCarbonEvent::GetEventClass(), PPx::SysCarbonEvent::GetEventKind(), and PPx::DataWriter::WriteValue().

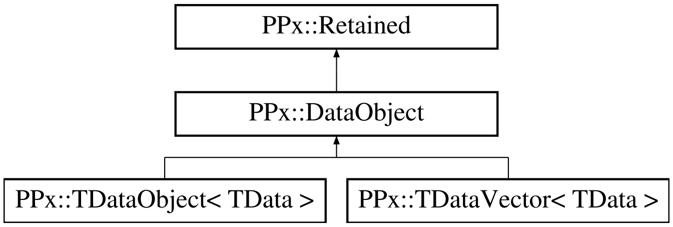
The documentation for this class was generated from the following files:

- PPxEventAttachments.h
- PPxEventAttachments.cp

6.202 PPx::Retained Class Reference

`#include <PPxRetained.h>`

Inheritance diagram for PPx::Retained::



6.202.1 Detailed Description

Base class for reference counted objects.

Definition at line 22 of file PPxRetained.h.

Public Member Functions

- Retained ()

  Default constructor.

- void Retain () const

  Increment object's retain count.

- void Release ()

  Decrements object's retain count.

- UInt32 GetRetainCount () const

  Returns object's retain count.

Protected Member Functions

- Retained (const Retained &inOriginal)

  Copy constructor.

- virtual ~Retained ()

  Destructor.

- Retained & operator= (const Retained &inSource)

  Assignment operator.

6.202.2 Member Function Documentation

6.202.2.1 UInt32 PPx::Retained::GetRetainCount () const

Returns object's retain count.

Returns:

Retain count of object

Definition at line 89 of file PPxRetained.cp.

Referenced by PPx::AutoRefCount< TObject >::GetRefCount(), and PPx::AutoRefCount< TObject >::Reset().

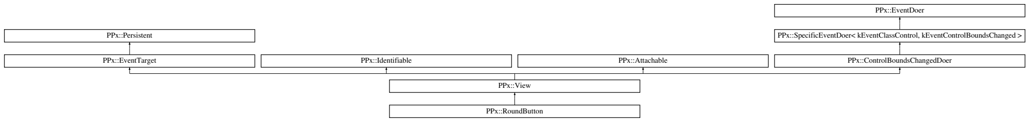
The documentation for this class was generated from the following files:

- PPxRetained.h
- PPxRetained.cp

6.203 PPx::RoundButton Class Reference

```
#include <PPxRoundButton.h>
```

Inheritance diagram for PPx::RoundButton::



6.203.1 Detailed Description

A system round button control.

Definition at line 22 of file PPxRoundButton.h.

Public Member Functions

- RoundButton ()

  Default constructor.

- virtual ∼RoundButton ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, ControlRoundButtonSize inButtonSize, const ControlButtonContentInfo &inContent)

  Initialize from chasing arrows creation parameters.

- void SetContentInfo (const ControlButtonContentInfo &inContent)

  Sets the content inforomation.

- void GetContentInfo (ControlButtonContentInfo &outContent) const

  Passes back the content inforomation.

- void SetButtonSize (ControlRoundButtonSize inButtonSize)

  Sets the button size.

- ControlRoundButtonSize GetButtonSize () const

  Returns the button size.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.
- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.203.2 Member Function Documentation

6.203.2.1 ControlRoundButtonSize PPx::RoundButton::GetButtonSize () const

Returns the button size.

Returns:

Button size

Definition at line 197 of file PPxRoundButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.203.2.2 void PPx::RoundButton::GetContentInfo (ControlButtonContentInfo & *outContent*) const

Passes back the content inforomation.

Parameters:

outContent Content inforomation

Definition at line 165 of file PPxRoundButton.cp.

References PPx::View::GetDataTag().

Referenced by WriteState().

6.203.2.3 void PPx::RoundButton::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, ControlRoundButtonSize *inButtonSize*, const ControlButtonContentInfo & *inContent*)

Initialize from chasing arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inButtonSize Size of round button (large or small)

inContent Content of round button

Definition at line 53 of file PPxRoundButton.cp.

6.203.2.4 void PPx::RoundButton::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 91 of file PPxRoundButton.cp.

References PPx::DataReader::ReadOptional().

6.203.2.5 void PPx::RoundButton::SetButtonSize (ControlRoundButtonSize *inButtonSize*)

Sets the button size.

Parameters:

inButtonSize New size for button

Definition at line 181 of file PPxRoundButton.cp.

References PPx::View::SetDataTag().

6.203.2.6 void PPx::RoundButton::SetContentInfo (const ControlButtonContentInfo & *inContent*)

Sets the content inforomation.

Parameters:

inContent Content inforomation

Definition at line 149 of file PPxRoundButton.cp.

References PPx::View::SetDataTag().

6.203.2.7 void PPx::RoundButton::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 120 of file PPxRoundButton.cp.

References GetButtonSize(), GetContentInfo(), and PPx::DataWriter::WriteValue().

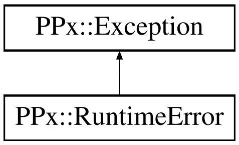
The documentation for this class was generated from the following files:

- PPxRoundButton.h
- PPxRoundButton.cp

6.204 PPx::RuntimeError Class Reference

`#include <PPxExceptions.h>`

Inheritance diagram for PPx::RuntimeError::



6.204.1 Detailed Description

Exception class for a runtime failure.

Runtime errors are due to conditions external to the program, such as running out of memory or some other system resource.

Definition at line 199 of file PPxExceptions.h.

Public Member Functions

- RuntimeError (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Constructor.

Static Public Member Functions

- void Throw (ExceptionIDT inWhat, const char ∗inWhy, const SourceLocation &inWhere)

  Throws a RuntimeError exception.

6.204.2 Constructor & Destructor Documentation

6.204.2.1 PPx::RuntimeError::RuntimeError (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*)

Constructor.

Parameters:

inWhat Kind of runtime error

inWhy C string describing why the exception occurred

inWhere Source code location where exception was thrown

Note:

If PPx_Debug_Exceptions is false, the why and where are not stored.

Definition at line 359 of file PPxExceptions.cp.

6.204.3 Member Function Documentation

6.204.3.1 void PPx::RuntimeError::Throw (ExceptionIDT *inWhat*, const char ∗ *inWhy*, const SourceLocation & *inWhere*) `[static]`

Throws a RuntimeError exception.

Parameters:

inWhat Kind of logic error

inWhy C string description of why the exception was thrown

inWhere Source location where exception was throw

Definition at line 379 of file PPxExceptions.cp.

The documentation for this class was generated from the following files:

- PPxExceptions.h
- PPxExceptions.cp

6.205 PPx::ScrapPromiseKeeper Class Reference

```
#include <PPxDataScrap.h>
```

6.205.1 Detailed Description

Abstract class for keeping promises to supply data for a scrap.

Definition at line 24 of file PPxDataScrap.h.

Public Member Functions

- OSStatus Invoke (ScrapRef inScrap, ScrapFlavorType inFlavor)

  Non-virtual wrapper function for invoking the KeepScrapPromise function.

6.205.2 Member Function Documentation

6.205.2.1 OSStatus PPx::ScrapPromiseKeeper::Invoke (ScrapRef *inScrap*, ScrapFlavorType *inFlavor*) `[inline]`

Non-virtual wrapper function for invoking the KeepScrapPromise function.

Parameters:

inScrap Scrap reference

inFlavor Flavor of data

Definition at line 49 of file PPxDataScrap.h.

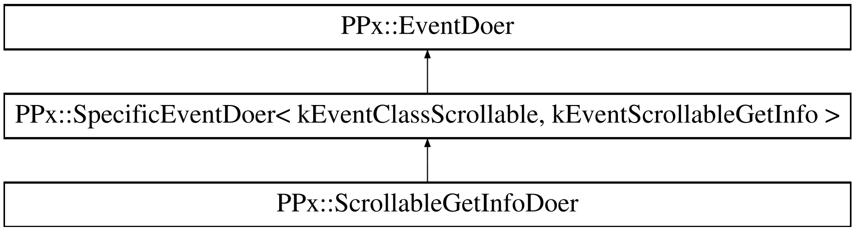
The documentation for this class was generated from the following file:

- PPxDataScrap.h

6.206 PPx::ScrollableGetInfoDoer Class Reference

```
#include <PPxScrollableEvents.h>
```

Inheritance diagram for PPx::ScrollableGetInfoDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassScrollable, kEventScrollableGetInfo >
PPx::ScrollableGetInfoDoer

6.206.1 Detailed Description

Returns information about a scrollable view.

Definition at line 20 of file PPxScrollableEvents.h.

Protected Member Functions

- virtual OSStatus **DoScrollableGetInfo** (SysCarbonEvent &ioEvent, HIViewRef inViewRef, HISize &outImageSize, HISize &outViewSize, HISize &outLineSize, HIPoint &outOrigin)=0

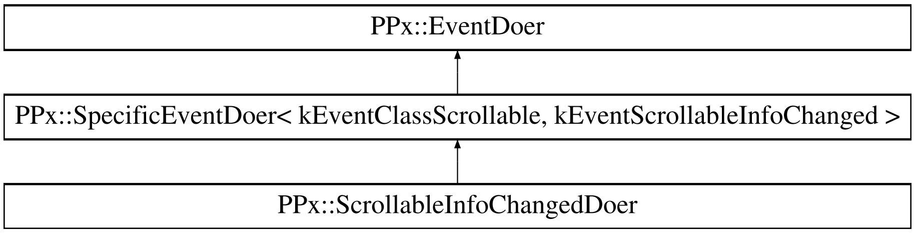
The documentation for this class was generated from the following files:

- PPxScrollableEvents.h
- PPxScrollableEvents.cp

6.207 PPx::ScrollableInfoChangedDoer Class Reference

`#include <PPxScrollableEvents.h>`

Inheritance diagram for PPx::ScrollableInfoChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassScrollable, kEventScrollableInfoChanged >
PPx::ScrollableInfoChangedDoer

6.207.1 Detailed Description

Handles notification that a scrollable view has changed.

Definition at line 40 of file PPxScrollableEvents.h.

Protected Member Functions

- virtual OSStatus **DoScrollableInfoChanged** (SysCarbonEvent &ioEvent, HIViewRef inViewRef)=0

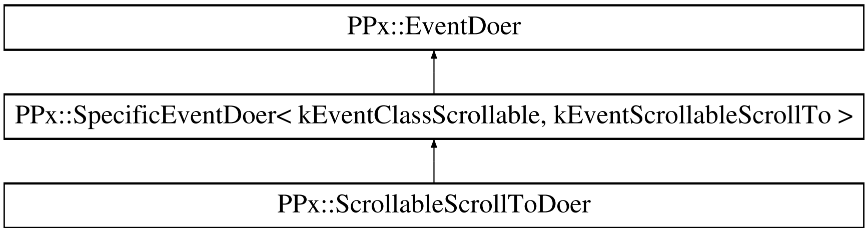
The documentation for this class was generated from the following files:

- PPxScrollableEvents.h
- PPxScrollableEvents.cp

6.208 PPx::ScrollableScrollToDoer Class Reference

`#include <PPxScrollableEvents.h>`

Inheritance diagram for PPx::ScrollableScrollToDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassScrollable, kEventScrollableScrollTo >
PPx::ScrollableScrollToDoer

6.208.1 Detailed Description

Handles scrolling a view to a specific location.

Definition at line 56 of file PPxScrollableEvents.h.

Protected Member Functions

- virtual OSStatus **DoScrollableScrollTo** (SysCarbonEvent &ioEvent, HIViewRef inViewRef, const HIPoint &inLocation)=0

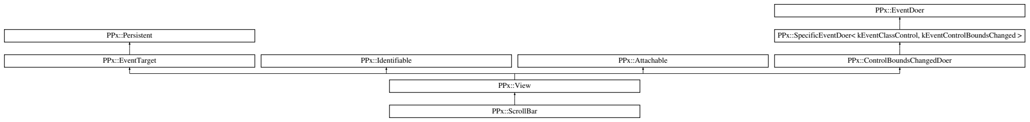
The documentation for this class was generated from the following files:

- PPxScrollableEvents.h
- PPxScrollableEvents.cp

6.209 PPx::ScrollBar Class Reference

```
#include <PPxScrollBar.h>
```

Inheritance diagram for PPx::ScrollBar::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::ScrollBar

6.209.1 Detailed Description

A system scroll bar control.

Definition at line 22 of file PPxScrollBar.h.

Public Member Functions

- ScrollBar ()

  Default constructor.

- virtual ∼ScrollBar ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, SInt32 inViewSize, bool inHasLiveTracking)

  Initialize from icon control creation parameters.

- void SetViewSize (SInt32 inViewSize)

  Sets the view size for the view being scrolled.

- SInt32 GetViewSize () const

  Returns the view size for the view being scrolled.

- void SetShowsArrowsFlag (bool inShowsArrows)

  Sets whether to show the scroll bar arrows.

- bool GetShowsArrowsFlag () const

  Returns whether to show the scroll bar arrows.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.209.2 Member Function Documentation

6.209.2.1 bool PPx::ScrollBar::GetShowsArrowsFlag () const

Returns whether to show the scroll bar arrows.

Returns:

Whether to show the scroll bar arrows

Definition at line 214 of file PPxScrollBar.cp.

References PPx::View::GetDataTag().

6.209.2.2 SInt32 PPx::ScrollBar::GetViewSize () const

Returns the view size for the view being scrolled.

Returns:

View size for the view being scrolled

Definition at line 181 of file PPxScrollBar.cp.

References PPx::View::GetSysView(), and PPx::SysHIView::GetViewSize().

Referenced by WriteState().

6.209.2.3 void PPx::ScrollBar::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, SInt32 *inViewSize*, bool *inHasLiveTracking*)

Initialize from icon control creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent
inVisible Whether the view is visible
inEnabled Whether the view is enabled
inInitialValue Initial value of control
inMinValue Minimum value of control
inMaxValue Maximum value fo control
inViewSize Size of view being scrolled
inHasLiveTracking Whether content scrolls as thumb is dragged

Definition at line 57 of file PPxScrollBar.cp.

6.209.2.4 void PPx::ScrollBar::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 105 of file PPxScrollBar.cp.

References PPx::DataReader::ReadOptional().

6.209.2.5 void PPx::ScrollBar::SetShowsArrowsFlag (bool *inShowsArrows*)

Sets whether to show the scroll bar arrows.

Parameters:

inShowsArrows Whether to show the scroll bar arrows

Definition at line 196 of file PPxScrollBar.cp.

References PPx::View::SetDataTag().

6.209.2.6 void PPx::ScrollBar::SetViewSize (SInt32 *inViewSize*)

Sets the view size for the view being scrolled.

Parameters:

inViewSize View size for the view being scrolled

Definition at line 165 of file PPxScrollBar.cp.

References PPx::View::GetSysView(), and PPx::SysHIView::SetViewSize().

6.209.2.7 void PPx::ScrollBar::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 143 of file PPxScrollBar.cp.

References PPx::View::GetMaxValue(), PPx::View::GetMinValue(), PPx::View::GetValue(), GetViewSize(), and PPx::DataWriter::WriteValue().

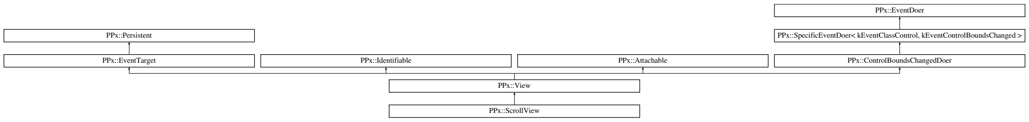
The documentation for this class was generated from the following files:

- PPxScrollBar.h
- PPxScrollBar.cp

6.210 PPx::ScrollView Class Reference

`#include <PPxScrollView.h>`

Inheritance diagram for PPx::ScrollView::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::ScrollView

6.210.1 Detailed Description

A system scroll view.

Definition at line 22 of file PPxScrollView.h.

Public Member Functions

- ScrollView ()

  Default constructor.

- virtual ~ScrollView ()

  Destructor.

- void Initialize (View *inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, OptionBits inOptions)

  Initialize from screoll view creation parameters.

- void SetAutoHideScrollBars (bool inAutoHide)

  Sets whether the automatically hide the scroll bars.

- bool GetAutoHideScrollBars () const

  Returns whether the automatically hide the scroll bars.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

Writes state to a data dictionary.

6.210.2 Member Function Documentation

6.210.2.1 bool PPx::ScrollView::GetAutoHideScrollBars () const

Returns whether the automatically hide the scroll bars.

Returns:

Whether the automatically hide the scroll bars

Definition at line 154 of file PPxScrollView.cp.

References PPx::View::GetSysView().

Referenced by WriteState().

6.210.2.2 void PPx::ScrollView::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, OptionBits *inOptions*)

Initialize from screoll view creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inOptions Options for scroll view

Definition at line 54 of file PPxScrollView.cp.

6.210.2.3 void PPx::ScrollView::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 92 of file PPxScrollView.cp.

References PPx::DataReader::ReadOptional(), and SetAutoHideScrollBars().

6.210.2.4 void PPx::ScrollView::SetAutoHideScrollBars (bool *inAutoHide*)

Sets whether the automatically hide the scroll bars.

Parameters:

inAutoHide Whether the automatically hide the scroll bars

Definition at line 136 of file PPxScrollView.cp.

References PPx::View::GetSysView(), and PPx_ThrowIfOSError_.

Referenced by InitState().

6.210.2.5 void PPx::ScrollView::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 116 of file PPxScrollView.cp.

References GetAutoHideScrollBars(), and PPx::DataWriter::WriteValue().

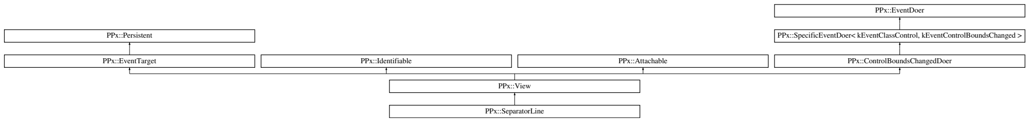
The documentation for this class was generated from the following files:

- PPxScrollView.h
- PPxScrollView.cp

6.211 PPx::SeparatorLine Class Reference

`#include <PPxSeparatorLine.h>`

Inheritance diagram for PPx::SeparatorLine::



6.211.1 Detailed Description

A system separator line view.

Definition at line 22 of file PPxSeparatorLine.h.

Public Member Functions

- SeparatorLine ()

  Default constructor.

- virtual ∼SeparatorLine ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled)

  Initialize from chasing arrows creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

6.211.2 Member Function Documentation

6.211.2.1 void PPx::SeparatorLine::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*)

Initialize from chasing arrows creation parameters.

Parameters:

- ***inSuperView*** Parent view
- ***inFrame*** Bounds for view, in local coordinates of parent
- ***inVisible*** Whether the view is visible
- ***inEnabled*** Whether the view is enabled

Definition at line 42 of file PPxSeparatorLine.cp.

6.211.2.2 void PPx::SeparatorLine::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

- ***inReader*** Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 77 of file PPxSeparatorLine.cp.

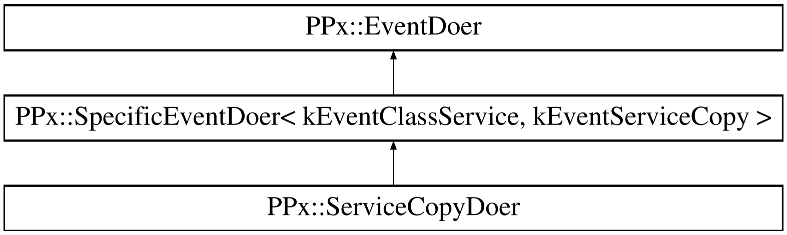
The documentation for this class was generated from the following files:

- PPxSeparatorLine.h
- PPxSeparatorLine.cp

6.212 PPx::ServiceCopyDoer Class Reference

`#include <PPxServiceEvents.h>`

Inheritance diagram for PPx::ServiceCopyDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassService, kEventServiceCopy >
PPx::ServiceCopyDoer

6.212.1 Detailed Description

Handles the service for copying data from current focus.

Definition at line 20 of file PPxServiceEvents.h.

Protected Member Functions

- virtual OSStatus **DoServiceCopy** (SysCarbonEvent &ioEvent, ScrapRef inScrap)=0

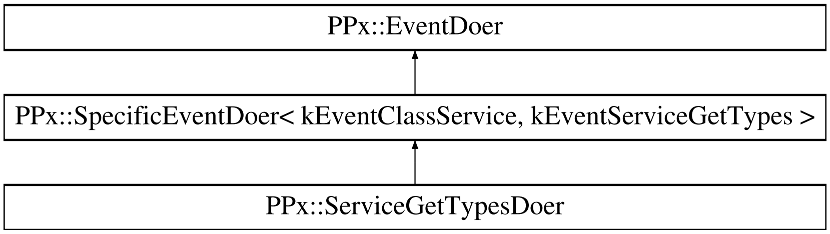
The documentation for this class was generated from the following files:

- PPxServiceEvents.h
- PPxServiceEvents.cp

6.213 PPx::ServiceGetTypesDoer Class Reference

`#include <PPxServiceEvents.h>`

Inheritance diagram for PPx::ServiceGetTypesDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassService, kEventServiceGetTypes >
PPx::ServiceGetTypesDoer

6.213.1 Detailed Description

Handles the service getting the types of data which can be copied and pasted.

Definition at line 53 of file PPxServiceEvents.h.

Protected Member Functions

- virtual OSStatus **DoServiceGetTypes** (SysCarbonEvent &ioEvent, CFMutableArrayRef inCopyTypes, CFMutableArrayRef inPasteTypes)=0

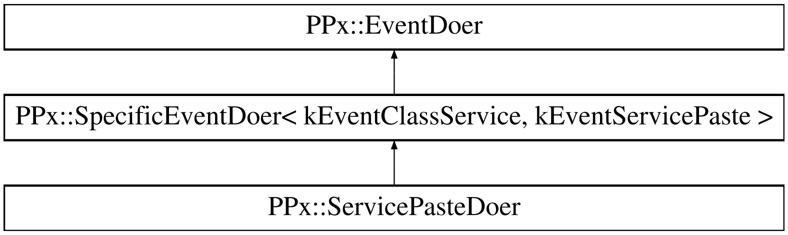
The documentation for this class was generated from the following files:

- PPxServiceEvents.h
- PPxServiceEvents.cp

6.214 PPx::ServicePasteDoer Class Reference

`#include <PPxServiceEvents.h>`

Inheritance diagram for PPx::ServicePasteDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassService, kEventServicePaste >
PPx::ServicePasteDoer

6.214.1 Detailed Description

Handles the service for pasting data into the current focus.

Definition at line 36 of file PPxServiceEvents.h.

Protected Member Functions

- virtual OSStatus **DoServicePaste** (SysCarbonEvent &ioEvent, ScrapRef inScrap)=0

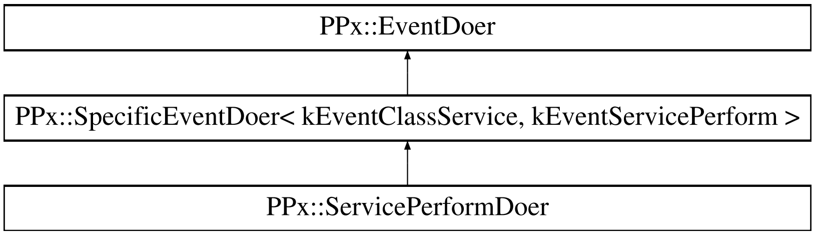
The documentation for this class was generated from the following files:

- PPxServiceEvents.h
- PPxServiceEvents.cp

6.215 PPx::ServicePerformDoer Class Reference

`#include <PPxServiceEvents.h>`

Inheritance diagram for PPx::ServicePerformDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassService, kEventServicePerform >
PPx::ServicePerformDoer

6.215.1 Detailed Description

Handles performing a service.

Definition at line 70 of file PPxServiceEvents.h.

Protected Member Functions

- virtual OSStatus **DoServicePerform** (SysCarbonEvent &ioEvent, ScrapRef inScrap, CFStringRef inMessageName, CFStringRef inUserData)=0

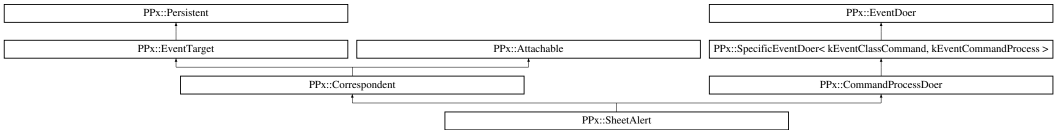
The documentation for this class was generated from the following files:

- PPxServiceEvents.h
- PPxServiceEvents.cp

6.216 PPx::SheetAlert Class Reference

```
#include <PPxSheetWindow.h>
```

Inheritance diagram for PPx::SheetAlert::



6.216.1 Detailed Description

An alert displayed as a sheet in a parent window.

Definition at line 53 of file PPxSheetWindow.h.

Public Member Functions

- SheetAlert ()

  Default constructor.

- virtual ∼SheetAlert ()

  Destructor.

- void Initialize (CFStringRef inErrorMessage, CFStringRef inExplanation=nil, AlertType inAlertType=kAlertNoteAlert, const AlertStdCFStringAlertParamRec ∗inParams=nil)

  Initialize from parameters.

- void Initialize (CFStringRef inErrorMessage, CFStringRef inExplanation, AlertType inAlertType, CFStringRef inOKLabel, CFStringRef inCancelLabel, CFStringRef inOtherLabel)

  Initialize from parameters.

- void Show (WindowRef inParent, EventTargetRef inNotifyTarget=nil, CommandIDT inOKCommand=0, CommandIDT inCancelCommand=0, CommandIDT inOtherCommand=0)

  Displays a sheet alert in a parent window.

- void Close ()

  Closes the sheet alert.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus DoCommandProcess (SysCarbonEvent &ioEvent, HICommand inCommand, UInt32 inKeyModifiers, UInt32 inMenuContext)

  Handles CarbonEvent for command process.

6.216.2 Member Function Documentation

6.216.2.1 OSStatus PPx::SheetAlert::DoCommandProcess (SysCarbonEvent & *ioEvent*, HICommand *inCommand*, UInt32 *inKeyModifiers*, UInt32 *inMenuContext*) `[protected, virtual]`

Handles CarbonEvent for command process.

Parameters:

ioEvent CarbonEvent for command process

inCommand Command information

inKeyModifiers State of modifiers keys

inMenuContext Menu context for command

Relays the event as an event for a specific command ID

Implements PPx::CommandProcessDoer.

Definition at line 314 of file PPxSheetWindow.cp.

6.216.2.2 void PPx::SheetAlert::Initialize (CFStringRef *inErrorMessage*, CFStringRef *inExplanation*, AlertType *inAlertType*, CFStringRef *inOKLabel*, CFStringRef *inCancelLabel*, CFStringRef *inOtherLabel*)

Initialize from parameters.

Parameters:

inErrorMessage Error message for alert

inExplanation Explanation message for alert

inAlertType Kind of alert

inOKLabel Title for the OK button

inCancelLabel Title for the cancel button

inOtherLabel Title for the other button

The button labels are the most common options to specify for an alert. Call the version of Initialize that takes a AlertStdCFStringAlertParamRec paraemter if you need to set other options.

Pass nil for the title of a button if you don't want the button to appear in the alert.

Definition at line 183 of file PPxSheetWindow.cp.

References Initialize(), and PPx_ThrowIfOSError_.

6.216.2.3 void PPx::SheetAlert::Initialize (CFStringRef *inErrorMessage*, CFStringRef *inExplanation* = nil, AlertType *inAlertType* = kAlertNoteAlert, const AlertStdCFStringAlertParamRec ∗ *inParams* = nil)

Initialize from parameters.

Parameters:

inErrorMessage Error message for alert

inExplanation Explanation message for alert

inAlertType Kind of alert

inParams Toolbox Alert settings

The AlertStdCFStringAlertParamRec contains several fields where you can set options for the alert. See <Dialogs.h> for details.

Definition at line 146 of file PPxSheetWindow.cp.

References PPx::EventTarget::GetSysEventTarget(), and PPx_ThrowIfOSError_.

Referenced by Initialize().

6.216.2.4 void PPx::SheetAlert::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Correspondent.

Definition at line 228 of file PPxSheetWindow.cp.

6.216.2.5 void PPx::SheetAlert::Show (WindowRef *inParent*, EventTargetRef *inNotifyTarget* = nil, CommandIDT *inOKCommand* = 0, CommandIDT *inCancelCommand* = 0, CommandIDT *inOtherCommand* = 0)

Displays a sheet alert in a parent window.

Parameters:

inParent Parent window in which to display sheet alert

inNotifyTarget Event target for button commands

inOKCommand Command sent after clicking the OK button

inCancelCommand Command sent after clicking the cancel button

inOtherCommand Cammand sent after clicking the other button

Pass in an event target and command ID number for the buttons if you wish to be notified that a particular button was clicked to dismiss the alert.

The system will qutomatically close the alert no matter what button is clicked.

Definition at line 269 of file PPxSheetWindow.cp.

References PPx_ThrowIfOSError_.

6.216.2.6 void PPx::SheetAlert::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Correspondent.

Definition at line 243 of file PPxSheetWindow.cp.

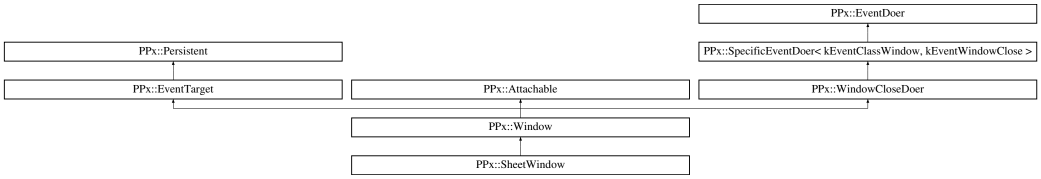
The documentation for this class was generated from the following files:

- PPxSheetWindow.h
- PPxSheetWindow.cp

6.217 PPx::SheetWindow Class Reference

`#include <PPxSheetWindow.h>`

Inheritance diagram for PPx::SheetWindow::



6.217.1 Detailed Description

A window displayed as a sheet in a parent window.

Definition at line 24 of file PPxSheetWindow.h.

Public Member Functions

- SheetWindow ()

  Default constructor.

- void Initialize (WindowAttributes inWindAttrs, const Rect &inBounds, Window ∗inParent)

  Initializes from parameters and shows the sheet.

- void Show (Window ∗inParent)

  Display the sheet window in a parent window.

- void Hide ()

  Hide the sheet window.

- WindowRef GetParentWindow () const

  Returns the parent window of the sheet.

6.217.2 Member Function Documentation

6.217.2.1 void PPx::SheetWindow::Initialize (WindowAttributes *inWindAttrs*, const Rect & *inBounds*, Window ∗ *inParent*)

Initializes from parameters and shows the sheet.

Parameters:

inWindAttrs Toolbox window attributes

inBounds Bounds of sheet window

inParent Window to which drawer is attached

See <MacWindows.h> for information about window attributes

Definition at line 32 of file PPxSheetWindow.cp.

References PPx::Window::Show().

6.217.2.2 void PPx::SheetWindow::Show (Window ∗ *inParent*)

Display the sheet window in a parent window.

Parameters:

inParent Parent window in which to display sheet

Definition at line 51 of file PPxSheetWindow.cp.

References PPx::Window::GetSysWindow().

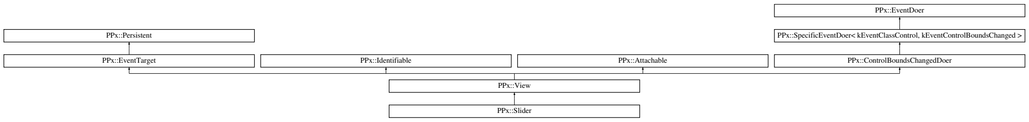
The documentation for this class was generated from the following files:

- PPxSheetWindow.h
- PPxSheetWindow.cp

6.218 PPx::Slider Class Reference

`#include <PPxSlider.h>`

Inheritance diagram for PPx::Slider::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::Slider

6.218.1 Detailed Description

A system slider control.

Definition at line 22 of file PPxSlider.h.

Public Member Functions

- Slider ()

  Default constructor.

- virtual ∼Slider ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, SInt32 inInitialValue, SInt32 inMinValue, SInt32 inMaxValue, ControlSliderOrientation inOrientation, UInt16 inTickMarksCount, bool inHasLiveTracking)

  Initialize from slider creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.218.2 Member Function Documentation

6.218.2.1 void PPx::Slider::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, SInt32 *inInitialValue*, SInt32 *inMinValue*, SInt32 *inMaxValue*, ControlSliderOrientation *inOrientation*, UInt16 *inTickMarksCount*, bool *inHasLiveTracking*)

Initialize from slider creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inInitialValue Initial value of control

inMinValue Minimum value of control

inMaxValue Maximum value fo control

inOrientation Orientation of thumb

inTickMarksCount Number of tick marks to draw

inHasLiveTracking Whether slider live tracks the thumb

Definition at line 61 of file PPxSlider.cp.

6.218.2.2 void PPx::Slider::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 112 of file PPxSlider.cp.

References PPx::DataReader::ReadOptional().

6.218.2.3 void PPx::Slider::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 146 of file PPxSlider.cp.

References PPx::View::GetMaxValue(), PPx::View::GetMinValue(), PPx::View::GetValue(), and PPx::DataWriter::WriteValue().

The documentation for this class was generated from the following files:

- PPxSlider.h
- PPxSlider.cp

6.219 PPx::SourceLocation Struct Reference

```
#include <PPxDebugging.h>
```

6.219.1 Detailed Description

Location within a source file.

Definition at line 25 of file PPxDebugging.h.

Public Member Functions

- **SourceLocation** (const char ∗inFunction, const char ∗inFile, int inLine)

Public Attributes

- const char ∗ **function**
- const char ∗ **file**
- int **line**

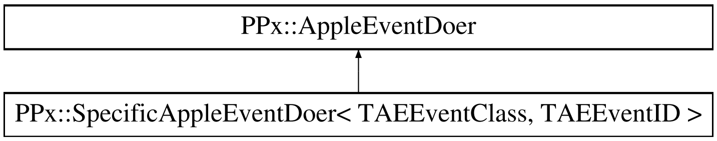
The documentation for this struct was generated from the following files:

- PPxDebugging.h
- PPxDebugging.cp

6.220 PPx::SpecificAppleEventDoer< TAEEventClass, TAEEventID > Class Template Reference

`#include <PPxAppleEventDoer.h>`

Inheritance diagram for PPx::SpecificAppleEventDoer< TAEEventClass, TAEEventID >::



6.220.1 Detailed Description

template<AEEventClass TAEEventClass, AEEventID TAEEventID> class PPx::SpecificAppleEventDoer< TAEEventClass, TAEEventID >

Template class for an Apple Event handler that responds to one specific type of event.

The template parameters specify the class and kind of the event.

Definition at line 64 of file PPxAppleEventDoer.h.

Public Member Functions

- void **Install** (bool inIsSystemHandler=false)

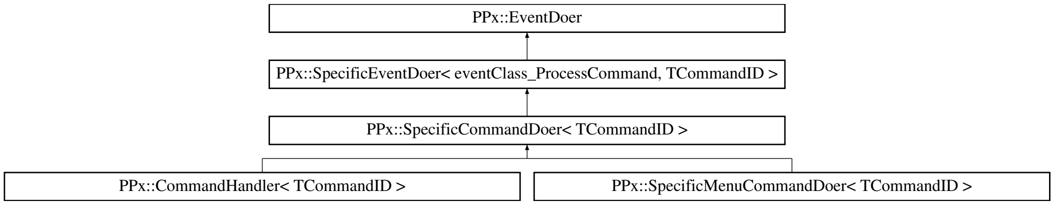
The documentation for this class was generated from the following file:

- PPxAppleEventDoer.h

6.221 PPx::SpecificCommandDoer< TCommandID > Class Template Reference

```
#include <PPxCommandEvents.h>
```

Inheritance diagram for PPx::SpecificCommandDoer< TCommandID >::


PPx::EventDoer
PPx::SpecificEventDoer< eventClass_ProcessCommand, TCommandID >
PPx::SpecificCommandDoer< TCommandID >
PPx::CommandHandler< TCommandID >
PPx::SpecificMenuCommandDoer< TCommandID >

6.221.1 Detailed Description

template<UInt32 TCommandID> class PPx::SpecificCommandDoer< TCommandID >

Handles processing a specific command.

Definition at line 97 of file PPxCommandEvents.h.

Protected Member Functions

- virtual OSStatus **DoSpecificCommand** (CommandIDType< TCommandID >, SysCarbonEvent &ioEvent)=0

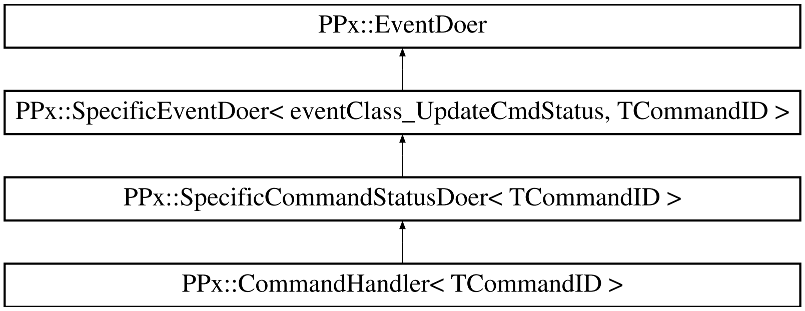
The documentation for this class was generated from the following file:

- PPxCommandEvents.h

6.222 PPx::SpecificCommandStatusDoer< TCommandID > Class Template Reference

```
#include <PPxCommandEvents.h>
```

Inheritance diagram for PPx::SpecificCommandStatusDoer< TCommandID >::


PPx::EventDoer
PPx::SpecificEventDoer< eventClass_UpdateCmdStatus, TCommandID >
PPx::SpecificCommandStatusDoer< TCommandID >
PPx::CommandHandler< TCommandID >

6.222.1 Detailed Description

template<UInt32 TCommandID> class PPx::SpecificCommandStatusDoer< TCommandID >

Handles updating the status of a specific command.

Definition at line 127 of file PPxCommandEvents.h.

Protected Member Functions

- virtual OSStatus **DoSpecificCommandStatus** (CommandIDType< TCommandID >, SysCarbonEvent &ioEvent)=0

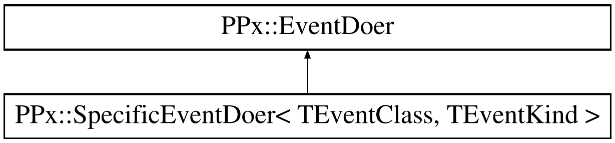
The documentation for this class was generated from the following file:

- PPxCommandEvents.h

6.223 PPx::SpecificEventDoer< TEventClass, TEventKind > Class Template Reference

```
#include <PPxEventDoer.h>
```

Inheritance diagram for PPx::SpecificEventDoer< TEventClass, TEventKind >::



6.223.1 Detailed Description

template<EventClassT TEventClass, EventKindT TEventKind> class PPx::SpecificEventDoer< TEventClass, TEventKind >

Template class for a Carbon Event handler that responds to one specific type of event.

The template parameters specify the class and kind of the event.

Definition at line 48 of file PPxEventDoer.h.

Public Member Functions

- EventHandlerRef **Install** (EventTargetRef inTarget)

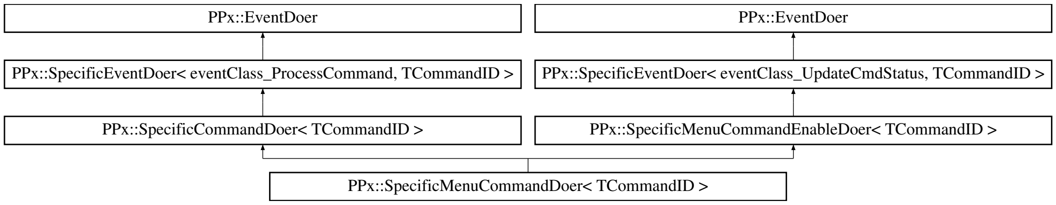
The documentation for this class was generated from the following file:

- PPxEventDoer.h

6.224 PPx::SpecificMenuCommandDoer< TCommandID > Class Template Reference

`#include <PPxCommandEvents.h>`

Inheritance diagram for PPx::SpecificMenuCommandDoer< TCommandID >::


PPx::EventDoer
PPx::EventDoer
PPx::SpecificEventDoer< eventClass_ProcessCommand, TCommandID >
PPx::SpecificEventDoer< eventClass_UpdateCmdStatus, TCommandID >
PPx::SpecificCommandDoer< TCommandID >
PPx::SpecificMenuCommandEnableDoer< TCommandID >
PPx::SpecificMenuCommandDoer< TCommandID >

6.224.1 Detailed Description

template<UInt32 TCommandID> class PPx::SpecificMenuCommandDoer< TCommandID >

Handles processing a specific menu command that is always enabled when the object is in the current focus chain.

Definition at line 201 of file PPxCommandEvents.h.

Public Member Functions

- void **Install** (EventTargetRef inTarget)

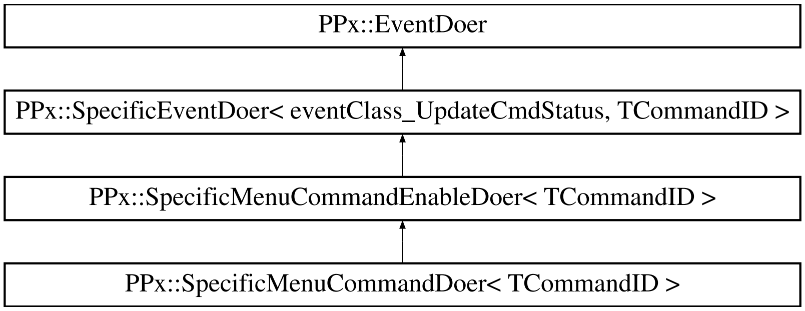
The documentation for this class was generated from the following file:

- PPxCommandEvents.h

6.225 PPx::SpecificMenuCommandEnableDoer< TCommandID > Class Template Reference

```
#include <PPxCommandEvents.h>
```

Inheritance diagram for PPx::SpecificMenuCommandEnableDoer< TCommandID >::


PPx::EventDoer
PPx::SpecificEventDoer< eventClass_UpdateCmdStatus, TCommandID >
PPx::SpecificMenuCommandEnableDoer< TCommandID >
PPx::SpecificMenuCommandDoer< TCommandID >

6.225.1 Detailed Description

template<UInt32 TCommandID> class PPx::SpecificMenuCommandEnableDoer< TCommandID >

Always enables a specific menu command.

Definition at line 174 of file PPxCommandEvents.h.

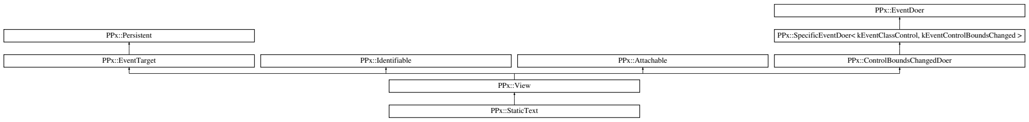
The documentation for this class was generated from the following file:

- PPxCommandEvents.h

6.226 PPx::StaticText Class Reference

```
#include <PPxStaticText.h>
```

Inheritance diagram for PPx::StaticText::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::StaticText

6.226.1 Detailed Description

A system static text control.

Definition at line 22 of file PPxStaticText.h.

Public Member Functions

- StaticText ()

  Default constructor.

- virtual ∼StaticText ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inText, const ControlFontStyleRec ∗inStyle)

  Initialize from static text creation parameters.

- void SetText (CFStringRef inText)

  Sets the text tp dosplay.

- CFString GetText () const

  Returns the text from the static text view.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID.

- void SetFontStyle (const ControlFontStyleRec &inStyleRec)

  Sets the font and style for the static text.

- void GetFontStyle (ControlFontStyleRec &outStyleRec)

  Passes back the font and style options.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.226.2 Member Function Documentation

6.226.2.1 void PPx::StaticText::GetFontStyle (ControlFontStyleRec & *outStyleRec*)

Passes back the font and style options.

Parameters:

outStyleRec Font and system information

Definition at line 208 of file PPxStaticText.cp.

References PPx::View::SetDataTag().

6.226.2.2 CFString PPx::StaticText::GetText () const

Returns the text from the static text view.

Returns:

Text from the static text view

Definition at line 155 of file PPxStaticText.cp.

References PPx::View::GetDataTag(), and PPx_ThrowIfOSError_.

Referenced by WriteState().

6.226.2.3 void PPx::StaticText::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inText*, const ControlFontStyleRec ∗ *inStyle*)

Initialize from static text creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inText Text to display

inStyle Text style

Definition at line 55 of file PPxStaticText.cp.

6.226.2.4 void PPx::StaticText::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 91 of file PPxStaticText.cp.

References PPx::DataReader::ReadOptional().

6.226.2.5 void PPx::StaticText::SetFontStyle (const ControlFontStyleRec & *inStyleRec*)

Sets the font and style for the static text.

Parameters:

inStyleRec Font and style to use for text

Definition at line 192 of file PPxStaticText.cp.

References PPx::View::SetDataTag().

6.226.2.6 void PPx::StaticText::SetText (CFStringRef *inText*)

Sets the text tp dosplay.

Parameters:

inText Text to display

Definition at line 139 of file PPxStaticText.cp.

References PPx::View::SetDataTag().

6.226.2.7 void PPx::StaticText::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID.

Parameters:

inFont Theme font ID to use for text

Definition at line 177 of file PPxStaticText.cp.

6.226.2.8 void PPx::StaticText::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 115 of file PPxStaticText.cp.

References GetText(), and PPx::DataWriter::WriteValue().

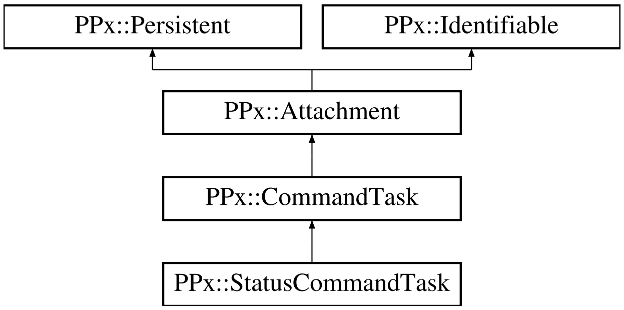
The documentation for this class was generated from the following files:

- PPxStaticText.h
- PPxStaticText.cp

6.227 PPx::StatusCommandTask Class Reference

`#include <PPxCommandTask.h>`

Inheritance diagram for PPx::StatusCommandTask::



6.227.1 Detailed Description

Abstract class for a Command event handler Attachment that also handles updating the status of items that invoke the command.

Typically, updating the status means enabling or disabling a menu item.

Definition at line 59 of file PPxCommandTask.h.

Public Member Functions

- void Initialize (EventTarget ∗inTarget, CommandIDT inCommandID)

  Specifies target and command ID.

- OSStatus **DoCommandUpdateStatusEvent** (SysCarbonEvent &ioEvent)

Protected Member Functions

- virtual OSStatus **DoCommandUpdateStatus** (HICommand inCommand, UInt32 inMenuContext)=0

6.227.2 Member Function Documentation

6.227.2.1 void PPx::StatusCommandTask::Initialize (EventTarget ∗ *inTarget*, CommandIDT *inCommandID*)

Specifies target and command ID.

Parameters:

inTarget Target which receives command events

inCommandID Command to handle

Reimplemented from PPx::CommandTask.

Definition at line 133 of file PPxCommandTask.cp.

References PPx::eventClass_UpdateCmdStatus, PPx::EventTarget::GetSysEventTarget(), and PPx::EventDoerCallback< StatusCommandTask >::Install().

The documentation for this class was generated from the following files:

- PPxCommandTask.h
- PPxCommandTask.cp

6.228 PPx::SysAEHandler Class Reference

```
#include <SysAEHandler.h>
```

6.228.1 Detailed Description

Wrapper class for an Apple Event Handler.

Definition at line 23 of file SysAEHandler.h.

Public Member Functions

- SysAEHandler ()

  Default constructor.

- ∼SysAEHandler ()

  Destructor.

- void Install (AEEventClass inClassID, AEEventID inEventID, AEEventHandlerUPP inCallback, void ∗inRefCon, bool inIsSystemHandler=false)

  Installs Handler for the specified kind of AppleEvent.

- void Remove ()

  Removes AppleEvent handler.

- void GetInfo (AEEventClass &outClassID, AEEventID &outEventID, bool &outIsSystemHandler) const

  Get information about the AE handler.

6.228.2 Member Function Documentation

6.228.2.1 void PPx::SysAEHandler::GetInfo (AEEventClass & *outClassID*, AEEventID & *outEventID*, bool & *outIsSystemHandler*) const

Get information about the AE handler.

Parameters:

outClassID AppleEvent class ID that is handled

outEventID AppleEvent event ID that is handled

outIsSystemHandler Whether handler is system-wide or local

Definition at line 107 of file SysAEHandler.cp.

6.228.2.2 void PPx::SysAEHandler::Install (AEEventClass *inClassID*, AEEventID *inEventID*, AEEventHandlerUPP *inCallback*, void ∗ *inRefCon*, bool *inIsSystemHandler* = false)

Installs Handler for the specified kind of AppleEvent.

Parameters:

inClassID AppleEvent class ID

inEventID AppleEvent event ID

inCallback Callback function to handle event

inRefCon User-defined data stored in AE handler

inIsSystemHandler Whether handler is system-wide or local

Definition at line 47 of file SysAEHandler.cp.

References PPx_ThrowIfOSError_, and Remove().

Referenced by PPx::AppleEventDoer::Install().

6.228.2.3 void PPx::SysAEHandler::Remove ()

Removes AppleEvent handler.

You can call Install() at a later time to re-install the handler

Definition at line 84 of file SysAEHandler.cp.

References PPx_ThrowIfOSError_.

Referenced by Install(), PPx::AppleEventDoer::Remove(), and ∼SysAEHandler().

The documentation for this class was generated from the following files:

- SysAEHandler.h
- SysAEHandler.cp

6.229 PPx::SysAEHandlerUPP Class Reference

```
#include <SysAEHandler.h>
```

6.229.1 Detailed Description

Wrapper class for an Apple Event callback function UPP.

Definition at line 64 of file SysAEHandler.h.

Public Member Functions

- **SysAEHandlerUPP** (AEEventHandlerProcPtr inCallbackFunc)
- AEEventHandlerUPP **Get** () const

The documentation for this class was generated from the following files:

- SysAEHandler.h
- SysAEHandler.cp

6.230 PPx::SysAppleEvent Class Reference

```
#include <SysAppleEvent.h>
```

6.230.1 Detailed Description

Wrapper class for an Apple Event.

Definition at line 23 of file SysAppleEvent.h.

Public Member Functions

- SysAppleEvent ()

  Constructor.

- SysAppleEvent (const AppleEvent &inAppleEvent)

  Constructs from an existing AppleEvent record.

- SysAppleEvent (AEEventClass inEventClass, AEEventID inEventID, const AEAddressDesc &inAETarget, AEReturnID inReturnID=kAutoGenerateReturnID, AETransactionID inTransID=kAnyTransactionID)

  Constructs from AppleEvent creation parameters.

- SysAppleEvent (const SysAppleEvent &inOriginal)

  Copy constructor.

- SysAppleEvent & operator= (const SysAppleEvent &inOriginal)

  Assignment operator.

- ~SysAppleEvent ()

  Destructor.

- const AppleEvent & GetAppleEvent () const

  Returns const reference to AppleEvent record.

- AppleEvent & GetAppleEvent ()

  Returns non-const reference to AppleEvent record.

- AEEventClass GetEventClass () const

  Returns the class of the AppleEvent.

- AEEventID GetEventKind () const

Returns the kind of the AppleEvent.

- OSStatus GetParameter (AEKeyword inName, DescType inDesiredType, DescType ∗outActualType, Size inBufferSize, Size ∗outActualSize, void ∗outData) const

  Gets a named parameter from the AppleEvent.

- void SetParameter (AEKeyword inName, DescType inType, Size inSize, const void ∗inDataPtr)

  Sets a named parameter in a AppleEvent.

- OSStatus GetParamDesc (AEKeyword inName, DescType inType, AEDesc &outDesc) const

  Gets a keyword descriptor from an AppleEvent.

- void SetParamDesc (AEKeyword inName, const AEDesc &inDesc)

  Sets a keyword descriptor in an AppleEvent.

- void Send (AppleEvent ∗inReply=nil, AESendMode inSendMode=kAENoReply, AESendPriority inPriority=kAENormalPriority, SInt32 inTimeOut=kAEDefaultTimeout, AEIdleUPP inIdleProc=nil, AEFilterUPP inFilterProc=nil)

  Sends an AppleEvent.

6.230.2 Constructor & Destructor Documentation

6.230.2.1 PPx::SysAppleEvent::SysAppleEvent (const AppleEvent & *inAppleEvent*)

Constructs from an existing AppleEvent record.

Parameters:

inAppleEvent AppleEvent record

Creates a copy of the input AppleEvent

Definition at line 29 of file SysAppleEvent.cp.

References PPx_ThrowIfOSError_.

6.230.2.2 PPx::SysAppleEvent::SysAppleEvent (AEEventClass *inEventClass*, AEEventID *inEventID*, const AEAddressDesc & *inAETarget*, AEReturnID *inReturnID* = kAutoGenerateReturnID, AETransactionID *inTransID* = kAnyTransactionID)

Constructs from AppleEvent creation parameters.

Parameters:

inEventClass Apple Event class

inEventID Apple Event ID

inAETarget Target program for sending the event

inReturnID Return ID

inTransID Transaction ID

Definition at line 48 of file SysAppleEvent.cp.

6.230.3 Member Function Documentation

6.230.3.1 AppleEvent & PPx::SysAppleEvent::GetAppleEvent ()

Returns non-const reference to AppleEvent record.

Returns:

Non-const reference to AppleEvent record

Definition at line 154 of file SysAppleEvent.cp.

6.230.3.2 const AppleEvent & PPx::SysAppleEvent::GetAppleEvent () const

Returns const reference to AppleEvent record.

Returns:

Const reference to AppleEvent record

Definition at line 140 of file SysAppleEvent.cp.

6.230.3.3 AEEventClass PPx::SysAppleEvent::GetEventClass () const

Returns the class of the AppleEvent.

Returns:

Class of the AppleEvent

Definition at line 168 of file SysAppleEvent.cp.

References PPx_ThrowIfOSError_.

6.230.3.4 AEEventID PPx::SysAppleEvent::GetEventKind () const

Returns the kind of the AppleEvent.

Returns:

Kind of the AppleEvent

Definition at line 193 of file SysAppleEvent.cp.

References PPx_ThrowIfOSError_.

6.230.3.5 OSStatus PPx::SysAppleEvent::GetParamDesc (AEKeyword *inName*, DescType *inType*, AEDesc & *outDesc*) const

Gets a keyword descriptor from an AppleEvent.

Parameters:

inName Keyword name

inType Descriptor type

outDesc Descriptor for the keyword

Returns:

OS error code

Definition at line 278 of file SysAppleEvent.cp.

6.230.3.6 OSStatus PPx::SysAppleEvent::GetParameter (AEKeyword *inName*, DescType *inDesiredType*, DescType ∗ *outActualType*, Size *inBufferSize*, Size ∗ *outActualSize*, void ∗ *outData*) const

Gets a named parameter from the AppleEvent.

Parameters:

inName Parameter name

inDesiredType Desired type for parameter value

outActualType Actual type of value retrieved

inBufferSize Size of buffer for storing parameter

outActualSize Actual number of bytes retrieved

outData Pointer to buffer for storing parameter

Returns:

OS error code

Definition at line 225 of file SysAppleEvent.cp.

6.230.3.7 void PPx::SysAppleEvent::Send (AppleEvent ∗ *inReply* = nil, AESendMode *inSendMode* = kAENoReply, AESendPriority *inPriority* = kAENormalPriority, SInt32 *inTimeOut* = kAEDefaultTimeout, AEIdleUPP *inIdleProc* = nil, AEFilterUPP *inFilterProc* = nil)

Sends an AppleEvent.

Parameters:

inReply Reply AppleEvent

inSendMode Mode for sending the event

inPriority Event priority

inTimeOut Time out interval

inIdleProc Idle proc

inFilterProc Filter proc

Definition at line 321 of file SysAppleEvent.cp.

References PPx_ThrowIfOSError_.

6.230.3.8 void PPx::SysAppleEvent::SetParamDesc (AEKeyword *inName*, const AEDesc & *inDesc*)

Sets a keyword descriptor in an AppleEvent.

Parameters:

inName Keyword name

inDesc Descriptor

Definition at line 298 of file SysAppleEvent.cp.

References PPx_ThrowIfOSError_.

6.230.3.9 void PPx::SysAppleEvent::SetParameter (AEKeyword *inName*, DescType *inType*, Size *inSize*, const void ∗ *inDataPtr*)

Sets a named parameter in a AppleEvent.

Parameters:

inName Parameter name

inType Parameter type

inSize Size of parameter value

inDataPtr Pointer to parameter data

Definition at line 253 of file SysAppleEvent.cp.

References PPx_ThrowIfOSError_.

The documentation for this class was generated from the following files:

- SysAppleEvent.h
- SysAppleEvent.cp

6.231 PPx::SysCarbonEvent Class Reference

```
#include <SysCarbonEvent.h>
```

6.231.1 Detailed Description

Wrapper class for a Carbon Event.

Definition at line 83 of file SysCarbonEvent.h.

Public Member Functions

- SysCarbonEvent ()

  Default constructor.

- SysCarbonEvent (EventRef inEventRef, EventHandlerCallRef inCallRef=nil)

  Constructs from an EventRef and EventHandlerCallRef.

- SysCarbonEvent (EventClassT inEventClass, EventKindT inEventKind, EventAttributes inAttrs=kEventAttributeNone, EventTime inWhen=eventTime_Now, CFAllocatorRef inAllocator=nil)

  Constructs from event creation parameters.

- SysCarbonEvent (const SysCarbonEvent &inOriginal)

  Copy constructor.

- SysCarbonEvent & operator= (const SysCarbonEvent &inOriginal)

  Assignment operator.

- ∼SysCarbonEvent ()

  Destructor.

- void Adopt (EventRef inEventRef, EventHandlerCallRef inCallRef=nil)

  Adopts existing EventRef and EventHnadlerCallRef.

- void MakeEvent (EventClassT inEventClass, EventKindT inEventKind, EventAttributes inAttrs=kEventAttributeNone, EventTime inWhen=eventTime_Now, CFAllocatorRef inAllocator=nil)

  Makes a CarbonEvent from the input creation parametrs.

- **operator EventRef** () const
- EventRef **GetEventRef** () const

- EventClassT GetEventClass () const

  Returns the class of the CarbonEvent.

- EventKindT GetEventKind () const

  Returns the kind of the CarbonEvent.

- EventTime GetTime () const

  Returns the time of the CarbonEvent.

- void SetTime (EventTime inTime)

  Sets the time of the CarbonEvent.

- OSStatus GetParameter (EventParamName inName, EventParamType inDesiredType, EventParamType ∗outActualType, UInt32 inBufferSize, UInt32 ∗outActualSize, void ∗outData) const

  Gets a named parameter from the CarbonEvent.

- void SetParameter (EventParamName inName, EventParamType inType, UInt32 inSize, const void ∗inData)

  Sets a named parameter in the CarbonEvent.

- void PostTo (EventTargetRef inTargetRef, EventQueueRef inQueueRef=nil, EventPriority inPriority=kEventPriorityStandard)

  Post CarbonEvent for a target on an event queue.

- OSStatus SendTo (EventTargetRef inTargetRef, OptionBits inOptions=options_None)

  Send CarbonEvent to a target.

- OSStatus CallNextHandler ()

  Sends the CarbonEvent to the next event handler.

6.231.2 Constructor & Destructor Documentation

6.231.2.1 PPx::SysCarbonEvent::SysCarbonEvent (EventRef *inEventRef*, EventHandlerCallRef *inCallRef* = nil)

Constructs from an EventRef and EventHandlerCallRef.

Parameters:

inEventRef System EventRef

inCallRef System EventHandlerCallRef

This constructor is designed for wrapping EventRefs that you receive from the system.

Definition at line 32 of file SysCarbonEvent.cp.

6.231.2.2 PPx::SysCarbonEvent::SysCarbonEvent (EventClassT *inEventClass*, EventKindT *inEventKind*, EventAttributes *inAttrs* = kEventAttributeNone, EventTime *inWhen* = eventTime_Now, CFAllocatorRef *inAllocator* = nil)

Constructs from event creation parameters.

Parameters:

inEventClass CarbonEvent class

inEventKind CarbonEvent kind

inAttrs Event attributes

inWhen Time of event

inAllocator CF Allocator

Definition at line 54 of file SysCarbonEvent.cp.

References MakeEvent().

6.231.3 Member Function Documentation

6.231.3.1 void PPx::SysCarbonEvent::Adopt (EventRef *inEventRef*, EventHandlerCallRef *inCallRef* = nil)

Adopts existing EventRef and EventHnadlerCallRef.

Parameters:

inEventRef System EventRef

inCallRef System EventHandlerCallRef

Releases its existing event and retains the input one

Definition at line 127 of file SysCarbonEvent.cp.

6.231.3.2 OSStatus PPx::SysCarbonEvent::CallNextHandler ()

Sends the CarbonEvent to the next event handler.

Returns:

OS error code

Definition at line 384 of file SysCarbonEvent.cp.

6.231.3.3 EventClassT PPx::SysCarbonEvent::GetEventClass () const

Returns the class of the CarbonEvent.

Returns:
Class of the CarbonEvent

Definition at line 192 of file SysCarbonEvent.cp.

Referenced by PPx::MessageAttachment::WriteState(), and PPx::ResponseAttachment::WriteState().

6.231.3.4 EventKindT PPx::SysCarbonEvent::GetEventKind () const

Returns the kind of the CarbonEvent.

Returns:
Kind of the CarbonEvent

Definition at line 206 of file SysCarbonEvent.cp.

Referenced by PPx::MessageAttachment::WriteState(), and PPx::ResponseAttachment::WriteState().

6.231.3.5 OSStatus PPx::SysCarbonEvent::GetParameter (EventParamName *inName*, EventParamType *inDesiredType*, EventParamType ∗ *outActualType*, UInt32 *inBufferSize*, UInt32 ∗ *outActualSize*, void ∗ *outData*) const

Gets a named parameter from the CarbonEvent.

Parameters:
inName Parameter name ID

inDesiredType Desired type for the parameter

outActualType Actual type of parameter retrieved

inBufferSize Size of buffer

outActualSize Actual size of data retrieved

outData Pointer to buffer for storing parameter value

Returns:
OS error code

Definition at line 256 of file SysCarbonEvent.cp.

6.231.3.6 EventTime PPx::SysCarbonEvent::GetTime () const

Returns the time of the CarbonEvent.

Returns:

Time of the CarbonEvent

Definition at line 220 of file SysCarbonEvent.cp.

6.231.3.7 void PPx::SysCarbonEvent::MakeEvent (EventClassT *inEventClass*, EventKindT *inEventKind*, EventAttributes *inAttrs* = kEventAttributeNone, EventTime *inWhen* = eventTime_Now, CFAllocatorRef *inAllocator* = nil)

Makes a CarbonEvent from the input creation parametrs.

Parameters:

inEventClass CarbonEvent class

inEventKind CarbonEvent kind

inAttrs Event attributes

inWhen Time of event

inAllocator CF Allocator

Releases its existing event and uses the new one created from the input parameters

Definition at line 159 of file SysCarbonEvent.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::MessageAttachment::InitState(), PPx::ResponseAttachment::InitState(), and SysCarbonEvent().

6.231.3.8 void PPx::SysCarbonEvent::PostTo (EventTargetRef *inTargetRef*, EventQueueRef *inQueueRef* = nil, EventPriority *inPriority* = kEventPriorityStandard)

Post CarbonEvent for a target on an event queue.

Parameters:

inTargetRef Target for event. May be nil.

inQueueRef Event queue to which to post the event. May be nil.

inPriority Event priority

If the target is nil, system dispatches the event in the standard manner (sending to the user focus target).

If the event queue is nil, we post the event to the current queue.

Note:

Posting an event is asynchronous. Function returns immediately after posting the event, which is handled at a later time.

Definition at line 314 of file SysCarbonEvent.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::Window::Close(), and PPx::EventUtils::PostCommandID().

6.231.3.9 OSStatus PPx::SysCarbonEvent::SendTo (EventTargetRef *inTargetRef*, OptionBits *inOptions* = options_None)

Send CarbonEvent to a target.

Parameters:

inTargetRef Target for event

inOptions Options for sending the event

See <CarbonEventsCore.h> for the list of options. At present, there are two options: kEventTargetDontPropagate means to send the event to the target only and do not propagate if the event isn't handled. CallNextEventHandler does nothing.

kEventTargetSendToAllHandlers means to send the event to all installed handlers, not stopping when a handler reports that it has handled the event. This is a broadcast or notification style of event.

Normal event dispatching sends the event to the target. If the target returns eventNotHandledErr, propagate the event to the next handler, and so on. Stop when a handler returns a result other than eventNotHandlerErr.

Definition at line 365 of file SysCarbonEvent.cp.

Referenced by PPx::EventUtils::ProcessCommandID(), PPx::EventUtils::SendCommandID(), and PPx::EventUtils::UpdateCommandID().

6.231.3.10 void PPx::SysCarbonEvent::SetParameter (EventParamName *inName*, EventParamType *inType*, UInt32 *inSize*, const void ∗ *inData*)

Sets a named parameter in the CarbonEvent.

Parameters:

- ***inName*** Parameter name ID
- ***inType*** Type of the parameter
- ***inSize*** Size of the parameter data
- ***inData*** Buffer containing parameter data

Throws an exception if there's an error setting the parameter

Definition at line 283 of file SysCarbonEvent.cp.

References PPx_ThrowIfOSError_.

6.231.3.11 void PPx::SysCarbonEvent::SetTime (EventTime *inTime*)

Sets the time of the CarbonEvent.

Parameters:

- ***inTime*** Time of the CarbonEvent

Definition at line 234 of file SysCarbonEvent.cp.

The documentation for this class was generated from the following files:

- SysCarbonEvent.h
- SysCarbonEvent.cp

6.232 PPx::SysEventHandler Class Reference

```
#include <SysEventHandler.h>
```

6.232.1 Detailed Description

Wrapper class for a Carbon Event Handler.

Definition at line 23 of file SysEventHandler.h.

Public Member Functions

- SysEventHandler ()

  Default constructor.

- ∼SysEventHandler ()

  Destructor.

- void Install (EventTargetRef inTarget, EventHandlerUPP inCallback, void ∗inUserData, UInt32 inNumTypes, const EventTypeSpec ∗inTypeList)

  Install event handler for a list of event types.

- void Install (EventTargetRef inTarget, EventHandlerUPP inCallback, void ∗inUserData, UInt32 inEventClass, UInt32 inEventKind)

  Install event handler for one event type.

- void Remove ()

  Removes the event handler from the system registry.

- bool IsInstalled () const

  Returns whether the handler is installed.

- void Adopt (EventHandlerRef inHandlerRef)

  Takes ownership of an existing EventHandlerRef.

- EventHandlerRef Detach ()

  Relinquishes ownership of its EventHanderRef.

6.232.2 Member Function Documentation

6.232.2.1 void PPx::SysEventHandler::Adopt (EventHandlerRef *inHandlerRef*)

Takes ownership of an existing EventHandlerRef.

Parameters:

inHandlerRef The EventHandlerRef to adopt

Definition at line 126 of file SysEventHandler.cp.

References Remove().

6.232.2.2 EventHandlerRef PPx::SysEventHandler::Detach ()

Relinquishes ownership of its EventHanderRef.

Returns:

Formerly owned EventHandlerRef. Caller now owns it.

Definition at line 142 of file SysEventHandler.cp.

Referenced by PPx::EventDoer::Install().

6.232.2.3 void PPx::SysEventHandler::Install (EventTargetRef *inTarget*, EventHandlerUPP *inCallback*, void ∗ *inUserData*, UInt32 *inEventClass*, UInt32 *inEventKind*)

Install event handler for one event type.

Parameters:

inTarget Target on which to install handler

inCallback Callback function UPP for handling the event

inUserData User data passed to callback function

inEventClass Class of CarbonEvent to handle

inEventKind Kind of CarbonEvent to handle

Removes existing handler if already istalled

Definition at line 76 of file SysEventHandler.cp.

References Install().

6.232.2.4 void PPx::SysEventHandler::Install (EventTargetRef *inTarget*, EventHandlerUPP *inCallback*, void ∗ *inUserData*, UInt32 *inNumTypes*, const EventTypeSpec ∗ *inTypeList*)

Install event handler for a list of event types.

Parameters:

inTarget Target on which to install handler

inCallback Callback function UPP for handling the event

inUserData User data passed to callback function

inNumTypes Number of event types

inTypeList Array of event type specifiers

Removes existing handler if already istalled

Definition at line 45 of file SysEventHandler.cp.

References PPx_ThrowIfOSError_, and Remove().

Referenced by Install(), and PPx::EventDoer::Install().

6.232.2.5 bool PPx::SysEventHandler::IsInstalled () const

Returns whether the handler is installed.

Returns:

Whether the handler is installed

Definition at line 112 of file SysEventHandler.cp.

The documentation for this class was generated from the following files:

- SysEventHandler.h
- SysEventHandler.cp

6.233 PPx::SysEventHandlerUPP Class Reference

```
#include <SysEventHandler.h>
```

6.233.1 Detailed Description

Wrapper class for a Carbon Event callback function UPP.

Definition at line 66 of file SysEventHandler.h.

Public Member Functions

- **SysEventHandlerUPP** (EventHandlerProcPtr inCallbackFunc)
- EventHandlerUPP **Get** () const

The documentation for this class was generated from the following files:

- SysEventHandler.h
- SysEventHandler.cp

6.234 PPx::SysEventLoopIdleTimer Class Reference

```
#include <SysEventLoopTimer.h>
```

6.234.1 Detailed Description

Wrapper class for an Event Loop Idle Timer.

Definition at line 57 of file SysEventLoopTimer.h.

Public Member Functions

- SysEventLoopIdleTimer ()

  Default constructor.

- ∼SysEventLoopIdleTimer ()

  Destructor.

- OSStatus Install (EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval, EventLoopIdleTimerUPP inIdleTimerUPP, void ∗inUserData)

  Install idle timer onto a System event loop.

- void Remove ()

  Remove idle timer from its event loop.

- bool IsInstalled () const

  Returns whether the idle imer is installed onto an event loop.

- OSStatus SetNextFireTime (EventTimerInterval inNextFire)

  Specify the next firing time.

6.234.2 Member Function Documentation

6.234.2.1 OSStatus PPx::SysEventLoopIdleTimer::Install (EventLoopRef *inEventLoop*, EventTimerInterval *inFireDelay*, EventTimerInterval *inInterval*, EventLoopIdleTimerUPP *inIdleTimerUPP*, void ∗ *inUserData*)

Install idle timer onto a System event loop.

Parameters:

inEventLoop System event loop. Usually you will pass in the return value from ::GetMainEventLoop() or ::GetCurrentEventLoop().

inFireDelay Seconds before first firing of the timer

inInterval Seconds between successive firings of the timer. If zero, the timer fires once, but is still installed. To fire it again, you can call SetNextFireTime().

inIdleTimerUPP Timer callback function

inUserData User data passed to callback function

Returns:

System error code

Note:

This function removes the timer if it is already installed, then re-installs it. Thus, you may call Install() more than once. You will need to do this in order to change the Event Loop or the Interval.

Definition at line 167 of file SysEventLoopTimer.cp.

References Remove().

Referenced by PPx::IdleTimer::Install().

6.234.2.2 bool PPx::SysEventLoopIdleTimer::IsInstalled () const

Returns whether the idle imer is installed onto an event loop.

Returns:

Whether the idle timer is installed

Definition at line 208 of file SysEventLoopTimer.cp.

Referenced by PPx::IdleTimer::IsTimerInstalled().

6.234.2.3 void PPx::SysEventLoopIdleTimer::Remove ()

Remove idle timer from its event loop.

You can re-install the idle timer later by calling Install().

Definition at line 191 of file SysEventLoopTimer.cp.

Referenced by Install(), PPx::IdleTimer::Remove(), and ∼SysEventLoopIdleTimer().

6.234.2.4 OSStatus PPx::SysEventLoopIdleTimer::SetNextFireTime (EventTimerInterval *inNextFire*)

Specify the next firing time.

Parameters:

inNextFire Next firing time, in seconds

Note:

Firing time temporarily overrides the interval until the next firing. For example, if you installed the timer with a 1 second interval, and then call SetNextFireTime(5), the timer will cease firing for 5 seconds, fire, and then fire again at 1 second intervals.

Definition at line 227 of file SysEventLoopTimer.cp.

Referenced by PPx::IdleTimer::SetNextFireTime().

The documentation for this class was generated from the following files:

- SysEventLoopTimer.h
- SysEventLoopTimer.cp

6.235 PPx::SysEventLoopIdleTimerUPP Class Reference

```
#include <SysEventLoopTimer.h>
```

6.235.1 Detailed Description

Wrapper class for an Event Loop Idle Timer callback function UPP.

Definition at line 125 of file SysEventLoopTimer.h.

Public Member Functions

- SysEventLoopIdleTimerUPP (EventLoopIdleTimerProcPtr inCallbackFunc)

  Constructs from a pointer to an event loop idle timer callback function.

- ∼SysEventLoopIdleTimerUPP ()

  Destructor.

- EventLoopIdleTimerUPP **Get** () const
- void **Dispose** ()

The documentation for this class was generated from the following files:

- SysEventLoopTimer.h
- SysEventLoopTimer.cp

6.236 PPx::SysEventLoopTimer Class Reference

`#include <SysEventLoopTimer.h>`

6.236.1 Detailed Description

Wrapper class for an Event Loop Timer.

Definition at line 23 of file SysEventLoopTimer.h.

Public Member Functions

- SysEventLoopTimer ()

  Default constructor.

- ∼SysEventLoopTimer ()

  Destructor.

- OSStatus Install (EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval, EventLoopTimerUPP inTimerUPP, void ∗inUserData)

  Install timer onto a System event loop.

- void Remove ()

  Remove timer from its event loop.

- bool IsInstalled () const

  Returns whether the timer is installed onto an event loop.

- OSStatus SetNextFireTime (EventTimerInterval inNextFire)

  Specify the next firing time.

6.236.2 Member Function Documentation

6.236.2.1 OSStatus PPx::SysEventLoopTimer::Install (EventLoopRef *inEventLoop*, EventTimerInterval *inFireDelay*, EventTimerInterval *inInterval*, EventLoopTimerUPP *inTimerUPP*, void ∗ *inUserData*)

Install timer onto a System event loop.

Parameters:

inEventLoop System event loop. Usually you will pass in the return value from ::GetMainEventLoop() or ::GetCurrentEventLoop().

inFireDelay Seconds before first firing of the timer

inInterval Seconds between successive firings of the timer. If zero, the timer fires once, but is still installed. To fire it again, you can call SetNextFireTime().

inTimerUPP Timer callback function

inUserData User data passed to callback function

Returns:

System error code

Note:

This function removes the timer if it is already installed, then re-installs it. Thus, you may call Install() more than once. You will need to do this in order to change the Event Loop or the Interval.

Definition at line 54 of file SysEventLoopTimer.cp.

References Remove().

Referenced by PPx::Timer::Install().

6.236.2.2 bool PPx::SysEventLoopTimer::IsInstalled () const

Returns whether the timer is installed onto an event loop.

Returns:

Whether the timer is installed

Definition at line 95 of file SysEventLoopTimer.cp.

Referenced by PPx::Timer::IsTimerInstalled().

6.236.2.3 void PPx::SysEventLoopTimer::Remove ()

Remove timer from its event loop.

You can re-install the timer later by calling Install().

Definition at line 78 of file SysEventLoopTimer.cp.

Referenced by Install(), PPx::Timer::Remove(), and ∼SysEventLoopTimer().

6.236.2.4 OSStatus PPx::SysEventLoopTimer::SetNextFireTime (EventTimerInterval *inNextFire*)

Specify the next firing time.

Parameters:

inNextFire Next firing time, in seconds

Note:

Firing time temporarily overrides the interval until the next firing. For example, if you installed the timer with a 1 second interval, and then call SetNextFireTime(5), the timer will cease firing for 5 seconds, fire, and then fire again at 1 second intervals.

Definition at line 114 of file SysEventLoopTimer.cp.

Referenced by PPx::Timer::SetNextFireTime().

The documentation for this class was generated from the following files:

- SysEventLoopTimer.h
- SysEventLoopTimer.cp

6.237 PPx::SysEventLoopTimerUPP Class Reference

`#include <SysEventLoopTimer.h>`

6.237.1 Detailed Description

Wrapper class for an Event Loop Timer callback function UPP.

Definition at line 91 of file SysEventLoopTimer.h.

Public Member Functions

- SysEventLoopTimerUPP (EventLoopTimerProcPtr inCallbackFunc)

  Constructs from a pointer to an event loop timer callback function.

- ∼SysEventLoopTimerUPP ()

  Destructor.

- EventLoopTimerUPP **Get** () const
- void **Dispose** ()

The documentation for this class was generated from the following files:

- SysEventLoopTimer.h
- SysEventLoopTimer.cp

6.238 PPx::SysEventSpec Struct Reference

```
#include <SysCarbonEvent.h>
```

6.238.1 Detailed Description

Struct describing the type of a Carbon Event.

The system identifies a Carbon Event by its class and kind, both 4-byte values. This struct groups the two identifiers so that we can initialize and compare them as a single unit.

Definition at line 33 of file SysCarbonEvent.h.

Public Member Functions

- **SysEventSpec** (EventClassT inClass, EventKindT inKind)

Public Attributes

- EventClassT **eventClass**
- EventKindT **eventKind**

The documentation for this struct was generated from the following file:

- SysCarbonEvent.h

6.239 PPx::SysHIObject Class Reference

```
#include <SysHIObject.h>
```

6.239.1 Detailed Description

Wrapper class for a Mac HIObject.

Definition at line 24 of file SysHIObject.h.

Public Member Functions

- SysHIObject ()

  Default constructor.

- SysHIObject (CFStringRef inClassID, EventRef inConstructData=nil)

  Constructs from HIObject creation parameters.

- ∼SysHIObject ()

  Destructor.

- void CreateSysObject (CFStringRef inClassID, EventRef inConstructData=nil)

  Create system HIObject from creation parameters.

- EventTargetRef GetSysEventTarget () const

  Returns the event target for the HIObject.

Static Public Member Functions

- void RegisterSysClass (CFStringRef inClassID, CFStringRef inBaseClassID, OptionBits inOptions, EventHandlerUPP inContructProc, UInt32 inNumEvents, const EventTypeSpec ∗inEventList, void ∗inConstructData, HIObjectClassRef ∗outClassRef)

  Register with the system a class for creating HIObjects.

- void RegisterSysClass (CFStringRef inClassID, CFStringRef inBaseClassID=nil)

  Simple registration for an HIObject subclass that handles its own construction and destruction process.

6.239.2 Constructor & Destructor Documentation

6.239.2.1 PPx::SysHIObject::SysHIObject (CFStringRef *inClassID*, EventRef *inConstructData* = nil)

Constructs from HIObject creation parameters.

Parameters:

inClassID Class ID of HIObject

inConstructData Event to send during construction

Definition at line 65 of file SysHIObject.cp.

References CreateSysObject().

6.239.3 Member Function Documentation

6.239.3.1 void PPx::SysHIObject::CreateSysObject (CFStringRef *inClassID*, EventRef *inConstructData* = nil)

Create system HIObject from creation parameters.

Parameters:

inClassID Class ID of HIObject

inConstructData Event to send during construction

Definition at line 97 of file SysHIObject.cp.

References PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by SysHIObject().

6.239.3.2 EventTargetRef PPx::SysHIObject::GetSysEventTarget () const

Returns the event target for the HIObject.

Returns:

Event target for the HIObject

Definition at line 120 of file SysHIObject.cp.

6.239.3.3 void PPx::SysHIObject::RegisterSysClass (CFStringRef *inClassID*, CFStringRef *inBaseClassID* = nil) `[static]`

Simple registration for an HIObject subclass that handles its own construction and destruction process.

Parameters:

inClassID Class ID of HIObject

inBaseClassID Class ID of base class of HIObject

Definition at line 170 of file SysHIObject.cp.

References PPx_ThrowIfOSError_.

6.239.3.4 void PPx::SysHIObject::RegisterSysClass (CFStringRef *inClassID*, CFStringRef *inBaseClassID*, OptionBits *inOptions*, EventHandlerUPP *inContructProc*, UInt32 *inNumEvents*, const EventTypeSpec ∗ *inEventList*, void ∗ *inConstructData*, HIObjectClassRef ∗ *outClassRef*) `[static]`

Register with the system a class for creating HIObjects.

Parameters:

inClassID Class ID of HIObject

inBaseClassID Class ID of base class of HIObject

inOptions Special optons (always 0 for now)

inContructProc Event handler for constructing HIObject

inNumEvents Number of events to handle

inEventList List of events to handle

inConstructData User-defined intialization data

outClassRef Reference registered HIObject class

Definition at line 141 of file SysHIObject.cp.

References PPx_ThrowIfOSError_.

The documentation for this class was generated from the following files:

- SysHIObject.h
- SysHIObject.cp

6.240 PPx::SysHIView Class Reference

```
#include <SysHIView.h>
```

6.240.1 Detailed Description

Wrapper class for a Mac Toolbox HIView.

Definition at line 24 of file SysHIView.h.

Public Member Functions

- SysHIView ()

  Default constructor.

- SysHIView (HIViewRef inViewRef)

  Constructs from an existing HIViewRef.

- ∼SysHIView ()

  Destructor.

- void Adopt (HIViewRef inViewRef)

  Adopts an existing HIViewRef.

- EventTargetRef GetSysEventTarget () const

  Returns the system event target for the HIView.

- HIViewRef GetSysView () const

  Returns the HIViewRef for the view.

- void AddSubView (HIViewRef inSubview)

  Adds a subview to this view.

- void RemoveFromSuperView ()

  Remove this view from its superview.

- HIViewRef GetSuperView () const

  Returns the HIViewRef for this view's superview.

- bool IsVisible () const

  Returns whether the view is visible.

- void SetVisible (bool inMakeVisible)

  Sets whether view is visible or invisible.

- bool IsActive () const

  Returns whether the view is active.

- void SetActive (bool inActivate)

  Sets whether view is active or inactive.

- bool IsEnabled () const

  Returns whether the view is enabled.

- void SetEnabled (bool inEnable)

  Sets whether view is enabled or disabled.

- void SetValue (SInt32 inValue)

  Sets the value for the view.

- SInt32 GetValue () const

  Returns the value for the view.

- void SetMinValue (SInt32 inMinValue)

  Sets the minimum value for the view.

- SInt32 GetMinValue () const

  Returns the minimum value for the view.

- void SetMaxValue (SInt32 inMaxValue)

  Sets the maximum value for the view.

- SInt32 GetMaxValue () const

  Returns the maximum value for the view.

- void SetViewSize (SInt32 inViewSize)

  Set the view size used to determine scrolling.

- SInt32 GetViewSize () const

  Returns the view size of the view.

- void SetTitle (CFStringRef inTitle)

  Sets the title of the view.

- CFString GetTitle () const

  Returns the title of the view.

- void SetCommandID (UInt32 inCommandID)

  Sets the Command ID sent when the view is clicked.

- UInt32 GetCommandID () const

  Returns the command ID for the view.

- void SetDataTag (SInt16 inPartCode, FourCharCode inTag, Size inDataSize, const void ∗inDataPtr)

  Sets a tagged data value for the view.

- OSStatus GetDataTag (SInt16 inPartCode, FourCharCode inTag, Size inBufferSize, void ∗inBuffer, Size ∗outDataSize=nil) const

  Gets a tggaed data value for the view.

- void SetProperty (OSType inCreator, OSType inTag, UInt32 inSize, const void ∗inPropertyPtr)

  Sets a property for the view.

- OSStatus GetProperty (OSType inCreator, OSType inTag, UInt32 inBufferSize, void ∗inBuffer, UInt32 ∗outSize=nil) const

  Gets a property for the view.

- void GetFrame (HIRect &outFrame) const

  Gets the frame of the view.

- void SetFrame (const HIRect &inFrame)

  Sets the frame of a view.

- void MoveFrameBy (float inDeltaX, float inDeltaY)

  Move the view's frame the specified distance.

- void PlaceFrameAt (float inLeftX, float inTopY)

  Places the view's frame at a particular location in its superview.

- void CreateOffscreenImage (HIRect &outFrame, CGImageRef &outImage)

  Creates an offscreen image for the view.

Static Public Member Functions

- void RegisterSysViewClass (CFStringRef inClassID, CFStringRef inBaseClassID)

  Registers a class with the system.

- HIViewRef CreateSysView (CFStringRef inClassID, OptionBits inFeatures)

  Create a new HIView.

6.240.2 Constructor & Destructor Documentation

6.240.2.1 PPx::SysHIView::SysHIView (HIViewRef *inViewRef*)

Constructs from an existing HIViewRef.

Parameters:

inViewRef HIViewRef for this object to use

Definition at line 30 of file SysHIView.cp.

6.240.3 Member Function Documentation

6.240.3.1 void PPx::SysHIView::AddSubView (HIViewRef *inSubview*)

Adds a subview to this view.

Parameters:

inSubview Subview to add

Definition at line 114 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::AddSubView().

6.240.3.2 void PPx::SysHIView::Adopt (HIViewRef *inViewRef*)

Adopts an existing HIViewRef.

Parameters:

inViewRef HIViewRef to adopt

Releases current HIViewRef and takes ownership of the input HIViewRef

Definition at line 65 of file SysHIView.cp.

Referenced by PPx::View::AdoptSysView(), and PPx::View::Initialize().

6.240.3.3 void PPx::SysHIView::CreateOffscreenImage (HIRect & *outFrame*, CGImageRef & *outImage*)

Creates an offscreen image for the view.

Parameters:

outFrame Frame of the offsceen image

outImage CFImageRef of the offscreen image

Definition at line 622 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.4 HIViewRef PPx::SysHIView::CreateSysView (CFStringRef *inClassID*, OptionBits *inFeatures*) `[static]`

Create a new HIView.

Parameters:

inClassID Class ID for the HIView

inFeatures Control features supported by the HIView

You must call RegisterSysViewClass() for the class ID before calling this function. inFeatures are the control featurees supported by the view. See <Controls.h> for a list of features.

Definition at line 811 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.5 UInt32 PPx::SysHIView::GetCommandID () const

Returns the command ID for the view.

Returns:

Command ID for the view

Definition at line 429 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.6 OSStatus PPx::SysHIView::GetDataTag (SInt16 *inPartCode*, FourCharCode *inTag*, Size *inBufferSize*, void ∗ *inBuffer*, Size ∗ *outDataSize* = nil) const

Gets a tggaed data value for the view.

Parameters:

inPartCode Part of the view to which the data applies

inTag Tag name of daa value

inBufferSize Length of data buffer

inBuffer Poitner to data buffer

outDataSize Actual size of data value

Definition at line 477 of file SysHIView.cp.

Referenced by PPx::View::GetDataTag().

6.240.3.7 void PPx::SysHIView::GetFrame (HIRect & *outFrame*) const

Gets the frame of the view.

Parameters:

outFrame View's frame

Definition at line 548 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::GetFrame(), PPx::View::GetLocalFrame(), and PPx::View::WriteState().

6.240.3.8 SInt32 PPx::SysHIView::GetMaxValue () const

Returns the maximum value for the view.

Returns:

Maximum value for the view

Definition at line 331 of file SysHIView.cp.

Referenced by PPx::View::GetMaxValue().

6.240.3.9 SInt32 PPx::SysHIView::GetMinValue () const

Returns the minimum value for the view.

Returns:

Minimum value for the view

Definition at line 302 of file SysHIView.cp.

Referenced by PPx::View::GetMinValue().

6.240.3.10 OSStatus PPx::SysHIView::GetProperty (OSType *inCreator*, OSType *inTag*, UInt32 *inBufferSize*, void ∗ *inBuffer*, UInt32 ∗ *outSize* = nil) const

Gets a property for the view.

Parameters:

inCreator Creator code for identifying the property

inTag Tag for identifying the property

inBufferSize Size of the property data

inBuffer Pointer to buffer for property data

outSize Actual number of bytes retrieved

Definition at line 527 of file SysHIView.cp.

6.240.3.11 HIViewRef PPx::SysHIView::GetSuperView () const

Returns the HIViewRef for this view's superview.

Returns:

HIViewRef for this view's superview

Definition at line 145 of file SysHIView.cp.

6.240.3.12 HIViewRef PPx::SysHIView::GetSysView () const

Returns the HIViewRef for the view.

Returns:

HIViewRef for the view

Definition at line 100 of file SysHIView.cp.

Referenced by PPx::View::GetSysView().

6.240.3.13 CFString PPx::SysHIView::GetTitle () const

Returns the title of the view.

Returns:
Title of the view

Definition at line 392 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::GetTitle().

6.240.3.14 SInt32 PPx::SysHIView::GetValue () const

Returns the value for the view.

Returns:
Value for the view

Definition at line 273 of file SysHIView.cp.

Referenced by PPx::View::GetValue().

6.240.3.15 SInt32 PPx::SysHIView::GetViewSize () const

Returns the view size of the view.

Returns:
View size of the view

Definition at line 360 of file SysHIView.cp.

Referenced by PPx::ScrollBar::GetViewSize().

6.240.3.16 bool PPx::SysHIView::IsActive () const

Returns whether the view is active.

Returns:
Whether the view is active

Definition at line 191 of file SysHIView.cp.

Referenced by PPx::View::IsActive().

6.240.3.17 bool PPx::SysHIView::IsEnabled () const

Returns whether the view is enabled.

Returns:

Whether the view is enabled

Definition at line 224 of file SysHIView.cp.

Referenced by PPx::View::IsEnabled(), and PPx::View::WriteState().

6.240.3.18 bool PPx::SysHIView::IsVisible () const

Returns whether the view is visible.

Returns:

Whether the view is visible

Definition at line 160 of file SysHIView.cp.

Referenced by PPx::View::IsVisible(), and PPx::View::WriteState().

6.240.3.19 void PPx::SysHIView::MoveFrameBy (float *inDeltaX*, float *inDeltaY*)

Move the view's frame the specified distance.

Parameters:

inDeltaX Horizontal offset

inDeltaY Vertical offset

Definition at line 583 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.20 void PPx::SysHIView::PlaceFrameAt (float *inLeftX*, float *inTopY*)

Places the view's frame at a particular location in its superview.

Parameters:

inLeftX Left location

inTopY Top location

Definition at line 602 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.21 void PPx::SysHIView::RegisterSysViewClass (CFStringRef *inClassID*, CFStringRef *inBaseClassID*) `[static]`

Registers a class with the system.

Before you can create a HIView, you must register its class name

Definition at line 772 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.22 void PPx::SysHIView::SetActive (bool *inActivate*)

Sets whether view is active or inactive.

Parameters:

inActivate Whether to make the view active or inactive

Definition at line 205 of file SysHIView.cp.

Referenced by PPx::View::SetActive().

6.240.3.23 void PPx::SysHIView::SetCommandID (UInt32 *inCommandID*)

Sets the Command ID sent when the view is clicked.

Parameters:

inCommandID New command ID for view

Definition at line 412 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

6.240.3.24 void PPx::SysHIView::SetDataTag (SInt16 *inPartCode*, FourCharCode *inTag*, Size *inDataSize*, const void ∗ *inDataPtr*)

Sets a tagged data value for the view.

Parameters:

inPartCode Part of the view to which the data applies

inTag Tag name of daa value

inDataSize Byte length of data

inDataPtr Pointer to data buffer

Definition at line 452 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::SetDataTag().

6.240.3.25 void PPx::SysHIView::SetEnabled (bool *inEnable*)

Sets whether view is enabled or disabled.

Parameters:

inEnable Whether to make the view enabled or disabled

Definition at line 238 of file SysHIView.cp.

Referenced by PPx::View::Initialize(), PPx::View::InitViewState(), and PPx::View::SetEnabled().

6.240.3.26 void PPx::SysHIView::SetFrame (const HIRect & *inFrame*)

Sets the frame of a view.

Parameters:

inFrame New frame for view

Definition at line 565 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::SetFrame().

6.240.3.27 void PPx::SysHIView::SetMaxValue (SInt32 *inMaxValue*)

Sets the maximum value for the view.

Parameters:

inMaxValue New maximum value for the view

Definition at line 316 of file SysHIView.cp.

Referenced by PPx::View::SetMaxValue().

6.240.3.28 void PPx::SysHIView::SetMinValue (SInt32 *inMinValue*)

Sets the minimum value for the view.

Parameters:

inMinValue New minimum value for the view

Definition at line 287 of file SysHIView.cp.

Referenced by PPx::View::SetMinValue().

6.240.3.29 void PPx::SysHIView::SetProperty (OSType *inCreator*, OSType *inTag*, UInt32 *inSize*, const void ∗ *inPropertyPtr*)

Sets a property for the view.

Parameters:

inCreator Creator code for identifying the property

inTag Tag for identifying the property

inSize Size of the property data

inPropertyPtr Pointer to buffer of property data

A creator code and tag identify a property

Definition at line 502 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::Initialize().

6.240.3.30 void PPx::SysHIView::SetTitle (CFStringRef *inTitle*)

Sets the title of the view.

Parameters:

inTitle New title for the view

Definition at line 375 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::SetTitle().

6.240.3.31 void PPx::SysHIView::SetValue (SInt32 *inValue*)

Sets the value for the view.

Parameters:

inValue New value for the view

Definition at line 258 of file SysHIView.cp.

Referenced by PPx::View::SetValue().

6.240.3.32 void PPx::SysHIView::SetViewSize (SInt32 *inViewSize*)

Set the view size used to determine scrolling.

Parameters:

inViewSize New view size for view

Definition at line 345 of file SysHIView.cp.

Referenced by PPx::ScrollBar::SetViewSize().

6.240.3.33 void PPx::SysHIView::SetVisible (bool *inMakeVisible*)

Sets whether view is visible or invisible.

Parameters:

inMakeVisible Whether to make the view visible or invisible

Definition at line 174 of file SysHIView.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::View::Initialize(), PPx::View::InitViewState(), and PPx::View::SetVisible().

The documentation for this class was generated from the following files:

- SysHIView.h
- SysHIView.cp

6.241 PPx::SysNavEventUPP Class Reference

```
#include <PPxNavServices.h>
```

6.241.1 Detailed Description

Wrapper class for a Navigation Services Event callback function UPP.

Definition at line 190 of file PPxNavServices.h.

Public Member Functions

- **SysNavEventUPP** (NavEventProcPtr inCallbackFunc)
- NavEventUPP **Get** () const

The documentation for this class was generated from the following files:

- PPxNavServices.h
- PPxNavServices.cp

6.242 PPx::SysScrapPromiseKeeperUPP Class Reference

`#include <SysScrap.h>`

6.242.1 Detailed Description

Wrapper class for a Scrap Promise Keeper callback function UPP.

Definition at line 73 of file SysScrap.h.

Public Member Functions

- **SysScrapPromiseKeeperUPP** (ScrapPromiseKeeperProcPtr inCallbackFunc)
- ScrapPromiseKeeperUPP Get () const

  Returns the UPP for a scrap promise keeper callback function.

6.242.2 Member Function Documentation

6.242.2.1 ScrapPromiseKeeperUPP PPx::SysScrapPromiseKeeperUPP::Get () const `[inline]`

Returns the UPP for a scrap promise keeper callback function.

Returns:

UPP for a scrap promise keeper callback function

Definition at line 101 of file SysScrap.h.

The documentation for this class was generated from the following files:

- SysScrap.h
- SysScrap.cp

6.243 PPx::SysWindow Class Reference

`#include <SysWindow.h>`

6.243.1 Detailed Description

Wrapper class for a Mac Toolbox Window.

Definition at line 23 of file SysWindow.h.

Public Member Functions

- SysWindow ()

  Constructor.

- SysWindow (WindowRef inWindowRef)

  Constructs from an existing WindowRef.

- SysWindow (WindowClass inWindClass, WindowAttributes inWindAttrs, const Rect &inContentBounds)

  Contructs from window creating parameters.

- ∼SysWindow ()

  Destructor.

- void Adopt (WindowRef inWindowRef)

  Adopts an existing WindowRef.

- void MakeWindow (WindowClass inWindClass, WindowAttributes inWindAttrs, const Rect &inContentBounds)

  Makes a new Toolbox window from creation parameters.

- WindowRef GetWindowRef () const

  Returns the WindowRef for this Window.

- void BecomeCurrentPort () const

  Make the Window's GrafPort the current port.

- void Select ()

  Brings a window to the front of its layer and activates it.

- bool IsVisible () const

Returns whether the window is visible.

- void Show ()

  Makes the window visible.

- void Hide ()

  Makes the window invisible.

- void SetTitle (CFStringRef inTitle)

  Sets the title of the window.

- CFString GetTitle () const

  Returns the title of the window.

- void SetBounds (Rect inBounds, WindowRegionCode regionCode=kWindowContentRgn)

  Sets the bounds for the specified region of the window.

- Rect GetBounds (WindowRegionCode regionCode=kWindowContentRgn) const

  Returns the bounds of a specified region of the window.

- WindowClass GetWindowClass () const

  Returns the class of the Toolbox window.

- WindowAttributes GetWindowAttributes () const

  Returns the window attributes.

- void SetProperty (OSType inCreator, OSType inTag, UInt32 inSize, const void ∗inPropertyPtr)

  Sets a property for the window.

- OSStatus GetProperty (OSType inCreator, OSType inTag, UInt32 inBufferSize, void ∗outBuffer, UInt32 ∗outSize=nil) const

  Gets a property for the window.

- void MoveStructureTo (SInt16 inHoriz, SInt16 inVert)

  Move window's structure to the specified screen location.

- void SetStructureBounds (const Rect &inBounds)

  Sets the structure bounds of the Window.

- void MoveContentTo (SInt16 inHoriz, SInt16 inVert)

Moves window's content to th specified screen location.

- void SetContentBounds (const Rect &inBounds)

  Sets the content bounds of the window.

Static Public Member Functions

- WindowRef GetScratchWindow ()

  Returns WindowRef for the scratch window.

6.243.2 Constructor & Destructor Documentation

6.243.2.1 PPx::SysWindow::SysWindow (WindowClass *inWindClass*, WindowAttributes *inWindAttrs*, const Rect & *inContentBounds*)

Contructs from window creating parameters.

Parameters:

inWindClass Window class

inWindAttrs Window attributes

inContentBounds Bounds of content region is global coordinates

Definition at line 43 of file SysWindow.cp.

References MakeWindow().

6.243.3 Member Function Documentation

6.243.3.1 void PPx::SysWindow::Adopt (WindowRef *inWindowRef*)

Adopts an existing WindowRef.

Parameters:

inWindowRef WindowRef to adopt

Releases current WindowRef and takes ownership of input WindowRef

Definition at line 77 of file SysWindow.cp.

6.243.3.2 Rect PPx::SysWindow::GetBounds (WindowRegionCode *inRegionCode* = kWindowContentRgn) const

Returns the bounds of a specified region of the window.

Parameters:

inRegionCode Window region whose bounds to return

Returns:

Bounds of specified window region

Definition at line 256 of file SysWindow.cp.

Referenced by PPx::Window::WriteState().

6.243.3.3 OSStatus PPx::SysWindow::GetProperty (OSType *inCreator*, OSType *inTag*, UInt32 *inBufferSize*, void ∗ *outBuffer*, UInt32 ∗ *outSize* = nil) const

Gets a property for the window.

Parameters:

inCreator Creator code for identifying the property

inTag Tag for identifying the property

inBufferSize Size of the property data

outBuffer Pointer to buffer for property data

outSize Actual number of bytes retrieved

Definition at line 341 of file SysWindow.cp.

6.243.3.4 WindowRef PPx::SysWindow::GetScratchWindow () `[static]`

Returns WindowRef for the scratch window.

Returns:

WindowRef for the scrath window

Some Toolbox calls require a valid WindowRef in order to work properly. In most cases, this is a bug/limitation in the OS. For such situations, we use a scratch window that has an empty bounds and is invisible.

Definition at line 430 of file SysWindow.cp.

References GetWindowRef().

6.243.3.5 CFString PPx::SysWindow::GetTitle () const

Returns the title of the window.

Returns:
Title of the window

Definition at line 222 of file SysWindow.cp.

Referenced by PPx::Window::GetTitle(), and PPx::Window::WriteState().

6.243.3.6 WindowAttributes PPx::SysWindow::GetWindowAttributes () const

Returns the window attributes.

Returns:
Window attributes

Definition at line 292 of file SysWindow.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::Window::WriteState().

6.243.3.7 WindowClass PPx::SysWindow::GetWindowClass () const

Returns the class of the Toolbox window.

Returns:
Class of the Toolbox window

Definition at line 273 of file SysWindow.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::Window::WriteState().

6.243.3.8 WindowRef PPx::SysWindow::GetWindowRef () const

Returns the WindowRef for this Window.

Returns:
WindowRef for this Window

Definition at line 129 of file SysWindow.cp.

Referenced by GetScratchWindow(), PPx::Window::GetSysWindow(), and PPx::Window::Initialize().

6.243.3.9 bool PPx::SysWindow::IsVisible () const

Returns whether the window is visible.

Returns:
Whether the window is visible

Definition at line 168 of file SysWindow.cp.

Referenced by PPx::Window::IsVisible().

6.243.3.10 void PPx::SysWindow::MakeWindow (WindowClass *inWindClass*, WindowAttributes *inWindAttrs*, const Rect & *inContentBounds*)

Makes a new Toolbox window from creation parameters.

Parameters:

inWindClass Window class

inWindAttrs Window attributes

inContentBounds Bounds of content region is global coordinates

Definition at line 105 of file SysWindow.cp.

References PPx_Throw_, and PPx_ThrowIfOSError_.

Referenced by PPx::Window::Initialize(), and SysWindow().

6.243.3.11 void PPx::SysWindow::MoveContentTo (SInt16 *inHoriz*, SInt16 *inVert*)

Moves window's content to th specified screen location.

Parameters:

inHoriz Horizontal location

inVert Vertical location

Definition at line 395 of file SysWindow.cp.

6.243.3.12 void PPx::SysWindow::MoveStructureTo (SInt16 *inHoriz*, SInt16 *inVert*)

Move window's structure to the specified screen location.

Parameters:

inHoriz Horizontal location

inVert Vertical location

Definition at line 363 of file SysWindow.cp.

6.243.3.13 void PPx::SysWindow::SetBounds (Rect *inBounds*, WindowRegionCode *inRegionCode* = kWindowContentRgn)

Sets the bounds for the specified region of the window.

Parameters:

inBounds Bounds in global coordinates

inRegionCode Region of window to set the bounds of

Definition at line 239 of file SysWindow.cp.

6.243.3.14 void PPx::SysWindow::SetContentBounds (const Rect & *inBounds*)

Sets the content bounds of the window.

Parameters:

inBounds New bounds the window's content

Definition at line 411 of file SysWindow.cp.

6.243.3.15 void PPx::SysWindow::SetProperty (OSType *inCreator*, OSType *inTag*, UInt32 *inSize*, const void ∗ *inPropertyPtr*)

Sets a property for the window.

Parameters:

inCreator Creator code for identifying the property

inTag Tag for identifying the property

inSize Size of the property data

inPropertyPtr Pointer to buffer of property data

A creator code and tag identify a property

Definition at line 316 of file SysWindow.cp.

References PPx_ThrowIfOSError_.

Referenced by PPx::Window::Initialize().

6.243.3.16 void PPx::SysWindow::SetStructureBounds (const Rect & *inBounds*)

Sets the structure bounds of the Window.

Parameters:

inBounds New bounds for the window’s structure

Definition at line 379 of file SysWindow.cp.

6.243.3.17 void PPx::SysWindow::SetTitle (CFStringRef *inTitle*)

Sets the title of the window.

Parameters:

inTitle Title for the window

Definition at line 207 of file SysWindow.cp.

Referenced by PPx::Window::Initialize(), and PPx::Window::SetTitle().

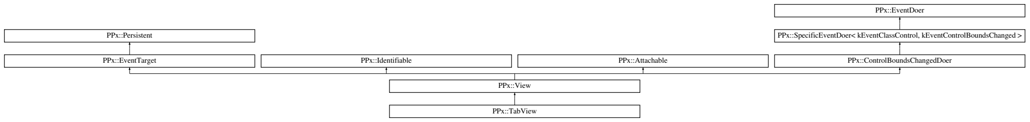
The documentation for this class was generated from the following files:

- SysWindow.h
- SysWindow.cp

6.244 PPx::TabView Class Reference

```
#include <PPxTabView.h>
```

Inheritance diagram for PPx::TabView::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::View
PPx::TabView

6.244.1 Detailed Description

A system tab view.

Definition at line 22 of file PPxTabView.h.

Public Member Functions

- TabView ()

  Default constructor.

- virtual ∼TabView ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, ControlTabSize inTabSize, ControlTabDirection inTabDirection, UInt16 inTabCount, const ControlTabEntry ∗inTabEntries)

  Initialize from chasing arrows creation parameters.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.244.2 Member Function Documentation

6.244.2.1 void PPx::TabView::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, ControlTabSize *inTabSize*, ControlTabDirection *inTabDirection*, UInt16 *inTabCount*, const ControlTabEntry ∗ *inTabEntries*)

Initialize from chasing arrows creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTabSize Size of tabs (normal or small)

inTabDirection Dirction of tabs (north, south, east, west)

inTabCount Number of tabs

inTabEntries Data for each tab

Definition at line 47 of file PPxTabView.cp.

6.244.2.2 void PPx::TabView::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 88 of file PPxTabView.cp.

6.244.2.3 void PPx::TabView::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID.

Parameters:

inFont Theme font ID to use for text

Definition at line 134 of file PPxTabView.cp.

6.244.2.4 void PPx::TabView::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 116 of file PPxTabView.cp.

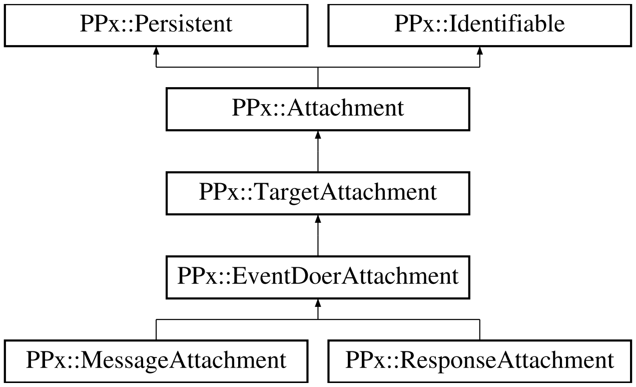
The documentation for this class was generated from the following files:

- PPxTabView.h
- PPxTabView.cp

6.245 PPx::TargetAttachment Class Reference

`#include <PPxEventAttachments.h>`

Inheritance diagram for PPx::TargetAttachment::



6.245.1 Detailed Description

Abstract attachment that has an associated event target.

Definition at line 25 of file PPxEventAttachments.h.

Public Member Functions

- void **SetEventTarget** (EventTarget ∗inTarget)
- EventTarget ∗ **GetEventTarget** () const

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.245.2 Member Function Documentation

6.245.2.1 void PPx::TargetAttachment::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Attachment.

Reimplemented in PPx::EventDoerAttachment, PPx::ResponseAttachment, and PPx::MessageAttachment.

Definition at line 75 of file PPxEventAttachments.cp.

References PPx::DataReader::ReadObjectValue().

6.245.2.2 void PPx::TargetAttachment::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Attachment.

Reimplemented in PPx::EventDoerAttachment, PPx::ResponseAttachment, and PPx::MessageAttachment.

Definition at line 92 of file PPxEventAttachments.cp.

References PPx::DataWriter::WriteObjectValue().

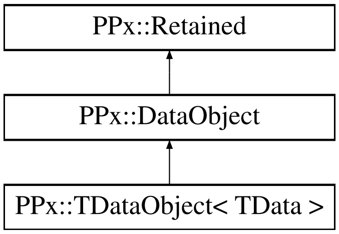
The documentation for this class was generated from the following files:

- PPxEventAttachments.h
- PPxEventAttachments.cp

6.246 PPx::TDataObject< TData > Struct Template Reference

`#include <PPxDataObject.h>`

Inheritance diagram for PPx::TDataObject< TData >::



6.246.1 Detailed Description

template<typename TData> struct PPx::TDataObject< TData >

Template class for objects that store a single data value of type TData.

Definition at line 40 of file PPxDataObject.h.

Public Member Functions

- **TDataObject** (const TData &inValue)

Public Attributes

- TData **mValue**

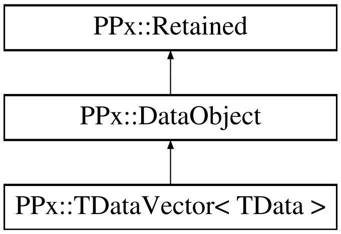
The documentation for this struct was generated from the following file:

- PPxDataObject.h

6.247 PPx::TDataVector< TData > Struct Template Reference

```
#include <PPxDataObject.h>
```

Inheritance diagram for PPx::TDataVector< TData >::



6.247.1 Detailed Description

template<typename TData> struct PPx::TDataVector< TData >

Template class for objects that store a vector of data values of type TData.

Definition at line 59 of file PPxDataObject.h.

Public Member Functions

- template<typename TInputIterator> **TDataVector** (TInputIterator inFirst, TInputIterator inLast)

Public Attributes

- std::vector< TData > **mDataValues**

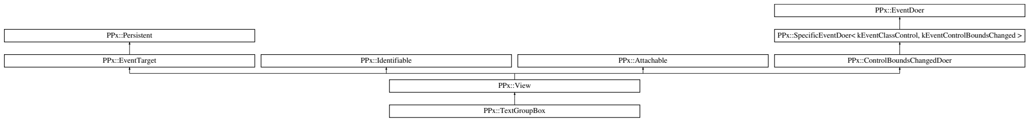
The documentation for this struct was generated from the following file:

- PPxDataObject.h

6.248 PPx::TextGroupBox Class Reference

`#include <PPxTextGroupBox.h>`

Inheritance diagram for PPx::TextGroupBox::



6.248.1 Detailed Description

A system group box with a text title.

Definition at line 22 of file PPxTextGroupBox.h.

Public Member Functions

- TextGroupBox ()

  Default constructor.

- virtual ∼TextGroupBox ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inTitle, bool inIsPrimary)

  Initialize from tab view creation parameters.

- void SetThemeFontID (ThemeFontID inFontID)

  Sets the theme font ID.

- void GetTitleRect (Rect &outTitleRect) const

  Passes back the title rectangle for the text group box.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

Writes state to a data dictionary.

6.248.2 Member Function Documentation

6.248.2.1 void PPx::TextGroupBox::GetTitleRect (Rect & *outTitleRect*) const

Passes back the title rectangle for the text group box.

Parameters:

outTitleRect Title rectangle

Definition at line 143 of file PPxTextGroupBox.cp.

References PPx::View::GetDataTag().

6.248.2.2 void PPx::TextGroupBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inTitle*, bool *inIsPrimary*)

Initialize from tab view creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inTitle Title for text group box

inIsPrimary Group box kind (true = primary, false = secondary)

Definition at line 47 of file PPxTextGroupBox.cp.

6.248.2.3 void PPx::TextGroupBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 86 of file PPxTextGroupBox.cp.

References PPx::DataReader::ReadOptional().

6.248.2.4 void PPx::TextGroupBox::SetThemeFontID (ThemeFontID *inFont*)

Sets the theme font ID.

Parameters:

inFont Theme font ID to use for text

Definition at line 128 of file PPxTextGroupBox.cp.

6.248.2.5 void PPx::TextGroupBox::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 108 of file PPxTextGroupBox.cp.

References PPx::View::GetTitle(), and PPx::DataWriter::WriteValue().

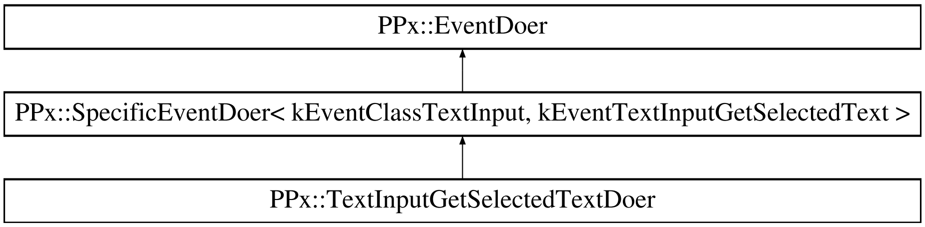
The documentation for this class was generated from the following files:

- PPxTextGroupBox.h
- PPxTextGroupBox.cp

6.249 PPx::TextInputGetSelectedTextDoer Class Reference

`#include <PPxTextInputEvents.h>`

Inheritance diagram for PPx::TextInputGetSelectedTextDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputGetSelectedText >
PPx::TextInputGetSelectedTextDoer

6.249.1 Detailed Description

Returns the selected text.

Definition at line 100 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputGetSelectedText** (SysCarbonEvent &ioEvent)=0

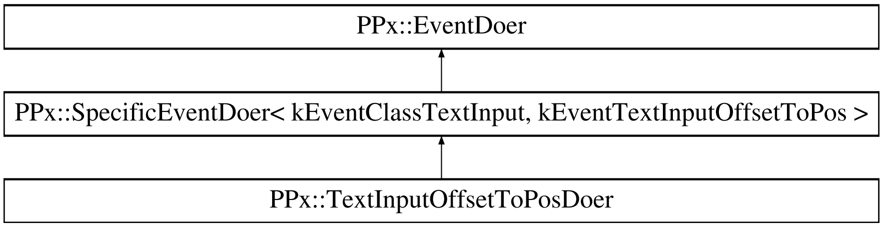
The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.250 PPx::TextInputOffsetToPosDoer Class Reference

`#include <PPxTextInputEvents.h>`

Inheritance diagram for PPx::TextInputOffsetToPosDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputOffsetToPos >
PPx::TextInputOffsetToPosDoer

6.250.1 Detailed Description

Converts from inline session text offset to global QD point.

Definition at line 50 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputOffsetToPos** (SysCarbonEvent &ioEvent, SInt32 inTextOffset, Point &outGlobalPoint)=0

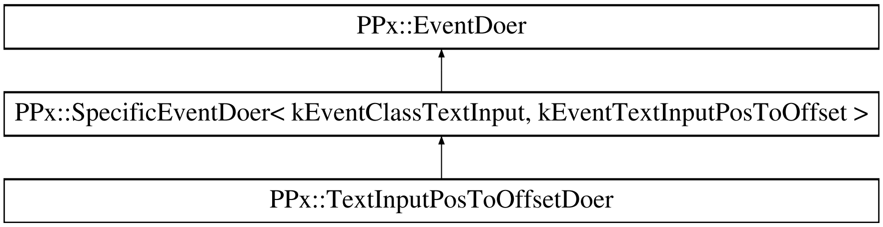
The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.251 PPx::TextInputPosToOffsetDoer Class Reference

`#include <PPxTextInputEvents.h>`

Inheritance diagram for PPx::TextInputPosToOffsetDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputPosToOffset >
PPx::TextInputPosToOffsetDoer

6.251.1 Detailed Description

Converts from global QD point to inline session text offset.

Definition at line 67 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputPosToOffset** (SysCarbonEvent &ioEvent, const Point &inGlobalPoint, SInt32 &outRegionClass, SInt32 &outTextOffset)=0

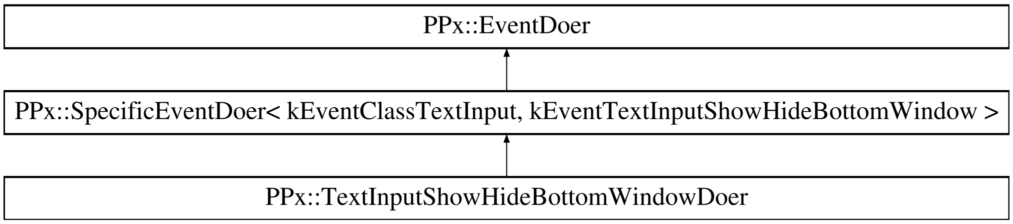
The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.252 PPx::TextInputShowHideBottomWindowDoer Class Reference

```
#include <PPxTextInputEvents.h>
```

Inheritance diagram for PPx::TextInputShowHideBottomWindowDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputShowHideBottomWindow >
PPx::TextInputShowHideBottomWindowDoer

6.252.1 Detailed Description

Shows or hides the bottom line input window.

Definition at line 85 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputShowHideBottomWindow** (SysCarbonEvent &ioEvent)=0

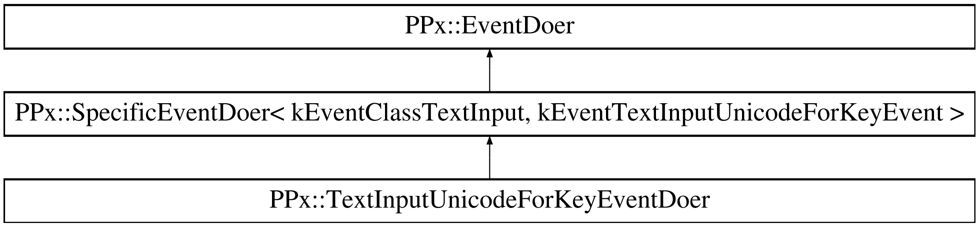
The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.253 PPx::TextInputUnicodeForKeyEventDoer Class Reference

`#include <PPxTextInputEvents.h>`

Inheritance diagram for PPx::TextInputUnicodeForKeyEventDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputUnicodeForKeyEvent >
PPx::TextInputUnicodeForKeyEventDoer

6.253.1 Detailed Description

Handles unicode text input from the keyboard.

Definition at line 34 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputUnicodeForKeyEvent** (SysCarbonEvent &ioEvent, UniChar inUniChar)=0

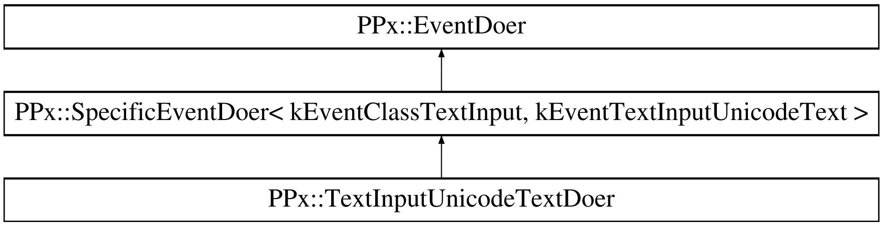
The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.254 PPx::TextInputUnicodeTextDoer Class Reference

```
#include <PPxTextInputEvents.h>
```

Inheritance diagram for PPx::TextInputUnicodeTextDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputUnicodeText >
PPx::TextInputUnicodeTextDoer

6.254.1 Detailed Description

Inputs unicode text.

Definition at line 115 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputUnicodeText** (SysCarbonEvent &ioEvent, UniChar inUniChar)=0

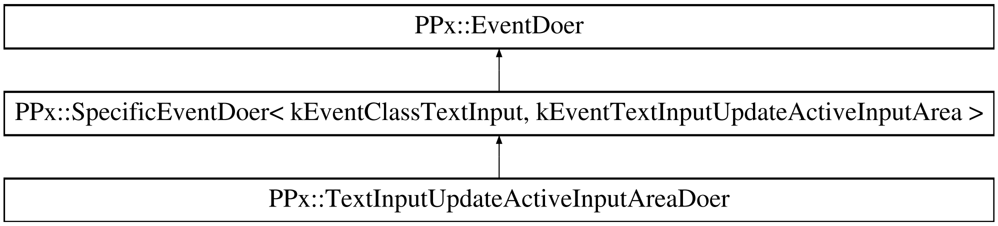
The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.255 PPx::TextInputUpdateActiveInputAreaDoer Class Reference

`#include <PPxTextInputEvents.h>`

Inheritance diagram for PPx::TextInputUpdateActiveInputAreaDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassTextInput, kEventTextInputUpdateActiveInputArea >
PPx::TextInputUpdateActiveInputAreaDoer

6.255.1 Detailed Description

Updates contents of a text input area.

Definition at line 20 of file PPxTextInputEvents.h.

Protected Member Functions

- virtual OSStatus **DoTextInputUpdateActiveInputArea** (SysCarbonEvent &ioEvent)=0

The documentation for this class was generated from the following files:

- PPxTextInputEvents.h
- PPxTextInputEvents.cp

6.256 PPx::ThemeMenuItemTypeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.256.1 Detailed Description

Wrapper for ThemeMenuItemType.

Definition at line 117 of file PPxSysTypes.h.

The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.257 PPx::ThemeMenuStateStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.257.1 Detailed Description

Wrapper for ThemeMenuState.

Definition at line 109 of file PPxSysTypes.h.

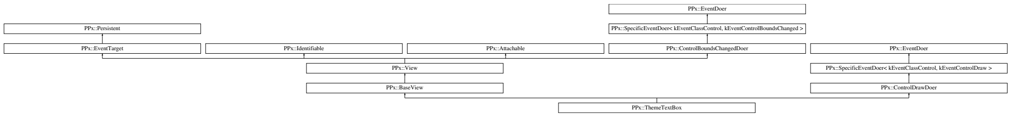
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.258 PPx::ThemeTextBox Class Reference

```
#include <PPxThemeTextBox.h>
```

Inheritance diagram for PPx::ThemeTextBox::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::EventDoer
PPx::View
PPx::SpecificEventDoer< kEventClassControl, kEventControlDraw >
PPx::BaseView
PPx::ControlDrawDoer
PPx::ThemeTextBox

6.258.1 Detailed Description

View for drawing text using a theme font inside a bounding box.

Definition at line 22 of file PPxThemeTextBox.h.

Public Member Functions

- ThemeTextBox ()

  Default constructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, CFStringRef inText, ThemeFontID inThemeFontID=kThemeSystemFont, SInt16 inJustification=teFlushDefault, bool inOneLineOnly=false)

  Intializes from parameters.

- void SetText (CFStringRef inText)

  Sets the text to display.

- CFString GetText () const

  Returns the text string.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus DoControlDraw (SysCarbonEvent &ioEvent, ControlRef inControl, ControlPartCode inPartCode, RgnHandle inClipRgn, CGContextRef inContext)

  Draws the view.

6.258.2 Member Function Documentation

6.258.2.1 OSStatus PPx::ThemeTextBox::DoControlDraw (SysCarbonEvent & *ioEvent*, ControlRef *inControl*, ControlPartCode *inPartCode*, RgnHandle *inClipRgn*, CGContextRef *inContext*) `[protected, virtual]`

Draws the view.

Parameters:

ioEvent CarbonEvent for control draw

inControl ControlRef for the view

inPartCode Part of the view to draw

inClipRgn Clipping region

inContext CGContext for drawing

Returns:

Status of drawing event. Always returns noErr.

Implements PPx::ControlDrawDoer.

Definition at line 150 of file PPxThemeTextBox.cp.

References PPx::View::GetLocalFrame(), and PPx::View::IsActive().

6.258.2.2 CFString PPx::ThemeTextBox::GetText () const

Returns the text string.

Returns:

Text string

Definition at line 197 of file PPxThemeTextBox.cp.

6.258.2.3 void PPx::ThemeTextBox::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, CFStringRef *inText*, ThemeFontID *inThemeFontID* = kThemeSystemFont, SInt16 *inJustification* = teFlushDefault, bool *inOneLineOnly* = false)

Intializes from parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inText Text to draw

inThemeFontID Theme font ID for drawing text

inJustification Text justification

inOneLineOnly Whether text is all on one line or word wraps

Definition at line 48 of file PPxThemeTextBox.cp.

6.258.2.4 void PPx::ThemeTextBox::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::BaseView.

Definition at line 101 of file PPxThemeTextBox.cp.

References PPx::BaseView::InitState(), and PPx::DataReader::ReadOptional().

6.258.2.5 void PPx::ThemeTextBox::SetText (CFStringRef *inText*)

Sets the text to display.

Parameters:

inText Text string

Definition at line 182 of file PPxThemeTextBox.cp.

References PPx::CFObject< CFStringRef >::AttachRef().

6.258.2.6 void PPx::ThemeTextBox::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::BaseView.

Definition at line 122 of file PPxThemeTextBox.cp.

References PPx::BaseView::WriteState(), and PPx::DataWriter::WriteValue().

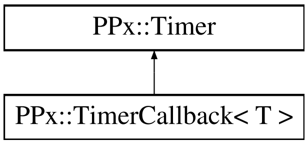
The documentation for this class was generated from the following files:

- PPxThemeTextBox.h
- PPxThemeTextBox.cp

6.259 PPx::Timer Class Reference

`#include <PPxTimer.h>`

Inheritance diagram for PPx::Timer::



6.259.1 Detailed Description

Abstract class for an Event Loop Timer.

Timers fires at regular intervals while the program is running. This includes during system calls that may block, such as mouse down tracking.

Definition at line 24 of file PPxTimer.h.

Public Member Functions

- Timer ()

  Default Constructor.

- Timer (EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval)

  Constructs and installs a Timer object.

- virtual ~Timer ()

  Destructor.

- void Install (EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval)

  Installs a Timer.

- void Remove ()

  Uninstalls a Timer.

- bool IsTimerInstalled () const

  Returns whether a Timer is currently installed on an event loop.

- void SetNextFireTime (EventTimerInterval inNextFire)

  Sets time delay until the Timer next fires.

- void **Invoke** ()

6.259.2 Constructor & Destructor Documentation

6.259.2.1 PPx::Timer::Timer (EventLoopRef *inEventLoop*, EventTimerInterval *inFireDelay*, EventTimerInterval *inInterval*)

Constructs and installs a Timer object.

Parameters:

inEventLoop Event loop on which to install the timer. Call ::GetMainEventLoop() for the main application event loop; call ::GetCurrentEventLoop() for the event loop of the current thread.

inFireDelay Time, in seconds, to delay before first call

inInterval Time, in seconds, between timer calls

Definition at line 71 of file PPxTimer.cp.

References Install().

6.259.3 Member Function Documentation

6.259.3.1 void PPx::Timer::Install (EventLoopRef *inEventLoop*, EventTimerInterval *inFireDelay*, EventTimerInterval *inInterval*)

Installs a Timer.

Call this function to re-install a timer that you have previously removed.

Parameters:

inEventLoop Event loop on which to install the timer

inFireDelay Time, in seconds, to delay before first call

inInterval Time, in seconds, between timer calls

Definition at line 101 of file PPxTimer.cp.

References PPx::SysEventLoopTimer::Install().

Referenced by Timer().

6.259.3.2 bool PPx::Timer::IsTimerInstalled () const

Returns whether a Timer is currently installed on an event loop.

Returns:

Whether the Timer is currently installed

Definition at line 131 of file PPxTimer.cp.

References PPx::SysEventLoopTimer::IsInstalled().

6.259.3.3 void PPx::Timer::Remove ()

Uninstalls a Timer.

You can call Install() later to re-intall it.

Definition at line 117 of file PPxTimer.cp.

References PPx::SysEventLoopTimer::Remove().

6.259.3.4 void PPx::Timer::SetNextFireTime (EventTimerInterval *inNextFire*)

Sets time delay until the Timer next fires.

This temporarily overrides the Timer's interval.

Parameters:

inNextFire Time, in seconds, until the Timer next fires

Definition at line 146 of file PPxTimer.cp.

References PPx_ThrowIfOSError_, and PPx::SysEventLoopTimer::SetNextFireTime().

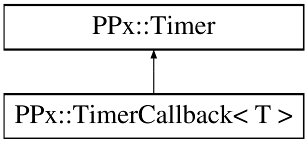
The documentation for this class was generated from the following files:

- PPxTimer.h
- PPxTimer.cp

6.260 PPx::TimerCallback< T > Class Template Reference

```
#include <PPxTimer.h>
```

Inheritance diagram for PPx::TimerCallback< T >::



6.260.1 Detailed Description

template<class T> class PPx::TimerCallback< T >

Template class for a Timer that calls an object member function.

Definition at line 126 of file PPxTimer.h.

Public Types

- typedef void(T::∗ **CallbackFunction**)()

Public Member Functions

- void **Install** (T ∗inObject, CallbackFunction inFunction, EventLoopRef inEventLoop, EventTimerInterval inFireDelay, EventTimerInterval inInterval)
- virtual void **DoTimer** ()

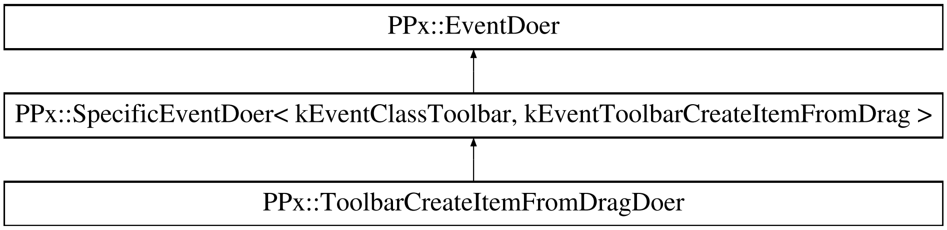
The documentation for this class was generated from the following file:

- PPxTimer.h

6.261 PPx::ToolbarCreateItemFromDragDoer Class Reference

`#include <PPxToolbarEvents.h>`

Inheritance diagram for PPx::ToolbarCreateItemFromDragDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassToolbar, kEventToolbarCreateItemFromDrag >
PPx::ToolbarCreateItemFromDragDoer

6.261.1 Detailed Description

Creates a new toolbar item from a drag and drop operation.

Definition at line 74 of file PPxToolbarEvents.h.

Protected Member Functions

- virtual OSStatus **DoTToolbarCreateItemFromDrag** (SysCarbonEvent &ioEvent, HIToolbarRef inToolbarRef, DragRef inDragRef, HIToolbarItemRef &outToolbarItem)=0

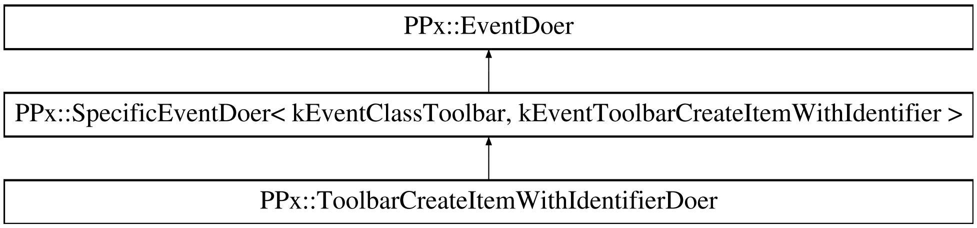
The documentation for this class was generated from the following files:

- PPxToolbarEvents.h
- PPxToolbarEvents.cp

6.262 PPx::ToolbarCreateItemWithIdentifierDoer Class Reference

`#include <PPxToolbarEvents.h>`

Inheritance diagram for PPx::ToolbarCreateItemWithIdentifierDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassToolbar, kEventToolbarCreateItemWithIdentifier >
PPx::ToolbarCreateItemWithIdentifierDoer

6.262.1 Detailed Description

Creates a new toolbar item with a specified identifier.

Definition at line 55 of file PPxToolbarEvents.h.

Protected Member Functions

- virtual OSStatus **DoToolbarCreateItemWithIdentifier** (SysCarbonEvent &ioEvent, HIToolbarRef inToolbarRef, CFStringRef inItemIdentifier, CFTypeRef inItemConfigData, HIToolbarItemRef &outToolbarItem)=0

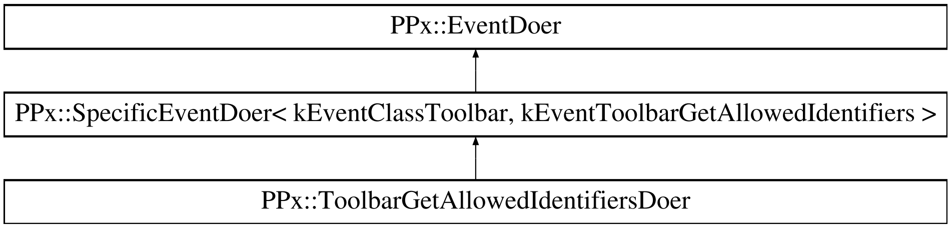
The documentation for this class was generated from the following files:

- PPxToolbarEvents.h
- PPxToolbarEvents.cp

6.263 PPx::ToolbarGetAllowedIdentifiersDoer Class Reference

```
#include <PPxToolbarEvents.h>
```

Inheritance diagram for PPx::ToolbarGetAllowedIdentifiersDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassToolbar, kEventToolbarGetAllowedIdentifiers >
PPx::ToolbarGetAllowedIdentifiersDoer

6.263.1 Detailed Description

Returns list of default item identifiers for a toolbar.

Definition at line 38 of file PPxToolbarEvents.h.

Protected Member Functions

- virtual OSStatus **DoToolbarGetAllowedIdentifiers** (SysCarbonEvent &ioEvent, HIToolbarRef inToolbarRef, CFMutableArrayRef ioIdentifiers)=0

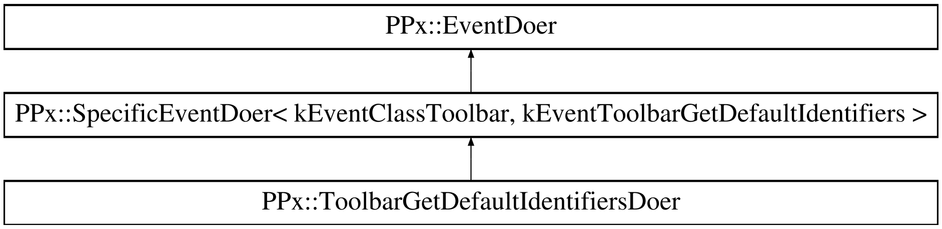
The documentation for this class was generated from the following files:

- PPxToolbarEvents.h
- PPxToolbarEvents.cp

6.264 PPx::ToolbarGetDefaultIdentifiersDoer Class Reference

```
#include <PPxToolbarEvents.h>
```

Inheritance diagram for PPx::ToolbarGetDefaultIdentifiersDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassToolbar, kEventToolbarGetDefaultIdentifiers >
PPx::ToolbarGetDefaultIdentifiersDoer

6.264.1 Detailed Description

Returns list of default item identifiers for a toolbar.

Definition at line 21 of file PPxToolbarEvents.h.

Protected Member Functions

- virtual OSStatus **DoToolbarGetDefaultIdentifiers** (SysCarbonEvent &ioEvent, HIToolbarRef inToolbarRef, CFMutableArrayRef ioIdentifiers)=0

The documentation for this class was generated from the following files:

- PPxToolbarEvents.h
- PPxToolbarEvents.cp

6.265 PPx::UniCharStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.265.1 Detailed Description

Wrapper for UniChar.

Definition at line 49 of file PPxSysTypes.h.

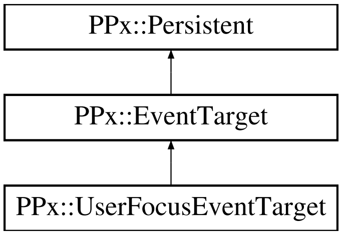
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.266 PPx::UserFocusEventTarget Class Reference

```
#include <PPxEventTarget.h>
```

Inheritance diagram for PPx::UserFocusEventTarget::



6.266.1 Detailed Description

Carbon Event target for the current user focus.

Definition at line 63 of file PPxEventTarget.h.

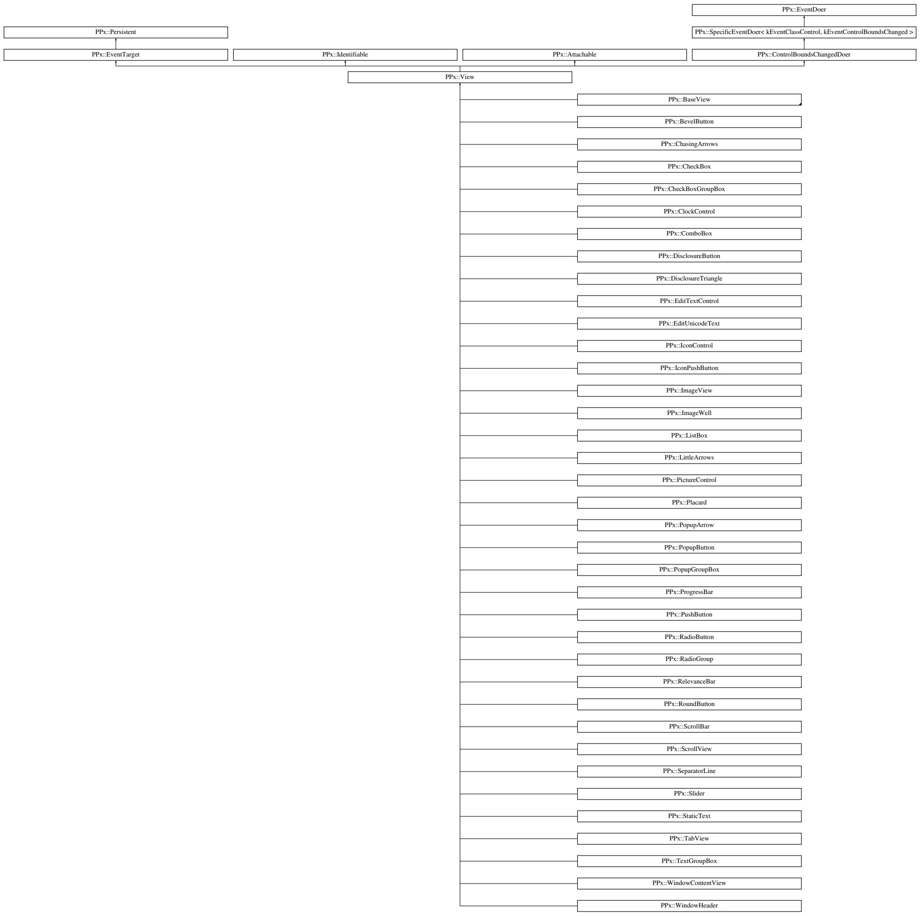
The documentation for this class was generated from the following files:

- PPxEventTarget.h
- PPxEventTarget.cp

6.267 PPx::View Class Reference

`#include <PPxView.h>`

Inheritance diagram for PPx::View::



6.267.1 Detailed Description

Abstract base class for a visual element.

Definition at line 30 of file PPxView.h.

Public Member Functions

- virtual ∼View ()

  Destructor.

- HIViewRef GetSysView () const

  Returns the system HIViewRef for the View.

- void AddSubView (View ∗inSubView)

  Adds a view as a subview of this view.

- void RemoveFromSuperView ()

  Removes view from its superview.

- View ∗ GetSuperView () const

  Returns the SuperView of the View.

- WindowRef GetSysWindow () const

  Returns the system WindowRef for the Window containing the View.

- SInt32 SubViewCount () const

  Returns the number of subviews of the View.

- View ∗ GetSubViewByIndex (SInt32 inIndex) const

  Returns subview specified by a zero-based index.

- View ∗ FindViewByID (ObjectIDT inID)

  Returns the View with teh specified Object ID.

- const View ∗ FindConstViewByID (ObjectIDT inID) const

  Returns the View with teh specified Object ID.

- void SetFrameAdapter (FrameAdapter ∗inAdapter)

  Sets the FrameAdapter object for the View.

- void SetFrame (const HIRect &inFrame)

  Sets the frame of the View.

- void GetFrame (HIRect &outFrame) const

  Passes back the View's frame.

- void GetLocalFrame (HIRect &outFrame) const

  Passes back the View's frame in local coordinates.

- bool IsVisible () const

  Returns whether the View is visible.

- void SetVisible (bool inMakeVisible)

  Makes the View visible or invisible.

- bool IsActive () const

  Returns whether the View is active.

- void SetActive (bool inActivate)

  Makes the View active or inactive.

- bool IsEnabled () const

  Returns whether the View is enabled.

- void SetEnabled (bool inEnable)

  Makes the View enabled or disabled.

- void SetValue (SInt32 inValue)

  Sets the View's value.

- SInt32 GetValue () const

  Returns the View's value.

- void SetMinValue (SInt32 inMinValue)

  Sets the minimum value for the View.

- SInt32 GetMinValue () const

  Gets the minimum value for the View.

- void SetMaxValue (SInt32 inMaxValue)

  Sets the maximum value for the View.

- SInt32 GetMaxValue () const

  Gets the maximum value for the View.

- void SetTitle (CFStringRef inTitle)

  Sets the title of the View.

- CFString GetTitle () const

  Gets the title of the View.

- void SetDataTag (SInt16 inPartCode, FourCharCode inTag, Size inDataSize, const void ∗inDataPtr)

  Sets a View property specified by a data tag.

- OSStatus GetDataTag (SInt16 inPartCode, FourCharCode inTag, Size inBufferSize, void ∗inBuffer, Size ∗outDataSize=nil) const

  Gets a View property specified by a data tag.

Static Public Member Functions

- View ∗ GetViewObject (HIViewRef inViewRef)

  Returns the View object associated with a HIViewRef.

Protected Member Functions

- View ()

  Default constructor.

- void Initialize (HIViewRef inViewRef, const HIRect &inFrame)

  Intializes from parameters.

- void Initialize (HIViewRef inViewRef, View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled)

  Intializes from parameters.

- void RemoveSubView (View ∗inSubView)

  Removes a subview from a View.

- void InitViewState (HIViewRef inViewRef, const DataReader &inReader)

  Initialize view from persistent data.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- void WriteViewHierarchy (DataWriter &ioWriter) const

  Write subviews to a data dictionary.

- void AdoptSysView (HIViewRef inViewRef)

  Uses the input HIViewRef.

- virtual OSStatus DoControlBoundsChanged (SysCarbonEvent &ioEvent, ControlRef inControl, UInt32 inChangeAttributes, const HIRect &inOriginalBounds, const HIRect &inCurrentBounds)

  Handles event for the View's bounds being changed.

- virtual void AdaptToSuperFrameSize (const HIRect &inOldSuperFrame, const HIRect &inNewSuperFrame)

  Adjusts the size and/or location of the View's frame in response to a change in the size of the superview's frame.

6.267.2 Member Function Documentation

6.267.2.1 void PPx::View::AdaptToSuperFrameSize (const HIRect & *inOldSuperFrame*, const HIRect & *inNewSuperFrame*) `[protected, virtual]`

Adjusts the size and/or location of the View's frame in response to a change in the size of the superview's frame.

Parameters:

inOldSuperFrame SuperView frame before change

inNewSuperFrame SuperView frame after change

Uses its FrameAdapter object to determine how to adjust its frame

Definition at line 646 of file PPxView.cp.

References GetFrame(), and SetFrame().

6.267.2.2 void PPx::View::AdoptSysView (HIViewRef *inViewRef*) `[protected]`

Uses the input HIViewRef.

Parameters:

inViewRef System HIViewRef to adopt

Definition at line 180 of file PPxView.cp.

References PPx::SysHIView::Adopt(), and PPx_BadParamIfNil_.

Referenced by PPx::WindowContentView::Initialize().

6.267.2.3 OSStatus PPx::View::DoControlBoundsChanged (SysCarbonEvent & *ioEvent*, ControlRef *inControl*, UInt32 *inChangeAttributes*, const HIRect & *inOriginalBounds*, const HIRect & *inCurrentBounds*) `[protected, virtual]`

Handles event for the View's bounds being changed.

Parameters:

ioEvent CarbonEvent for control bounds changed

inControl ControlRef for View

inChangeAttributes Indicates if size and/or location changed

inOriginalBounds Bounds before change

inCurrentBounds Bounds after change

Informs subviews of the bounds change so that they can adjust their size and/or location if necessary

Implements PPx::ControlBoundsChangedDoer.

Definition at line 607 of file PPxView.cp.

6.267.2.4 const View ∗ PPx::View::FindConstViewByID (ObjectIDT *inID*) const

Returns the View with teh specified Object ID.

View returned is the View itself or one of its subviews.

Parameters:

inID Object ID of View to find

Returns:

Const View with the specified object ID

Searches the entire hierarchy rooted at the View. View returned is the View itself, a descendent View, or nil if not found.

Definition at line 499 of file PPxView.cp.

References PPx::Identifiable::GetID().

Referenced by FindViewByID().

6.267.2.5 View ∗ PPx::View::FindViewByID (ObjectIDT *inID*)

Returns the View with teh specified Object ID.

Parameters:

inID Object ID of View to find

Returns:

View with the specified object ID

Searches the entire hierarchy rooted at the View. View returned is the View itself, a descendent View, or nil if not found.

Definition at line 473 of file PPxView.cp.

References FindConstViewByID().

6.267.2.6 OSStatus PPx::View::GetDataTag (SInt16 *inPartCode*, FourCharCode *inTag*, Size *inBufferSize*, void ∗ *inBuffer*, Size ∗ *outDataSize* = nil) const

Gets a View property specified by a data tag.

Parameters:

inPartCode Part of View to which the property applies

inTag Data tag ID

inBufferSize Byte size of buffer

inBuffer Pointer to data buffer

outDataSize Actual size of data returned. Pass nil if you don't want this information

Definition at line 902 of file PPxView.cp.

References PPx::SysHIView::GetDataTag().

Referenced by PPx::RoundButton::GetButtonSize(), PPx::PushButton::GetCancelFlag(), PPx::IconPushButton::GetCancelFlag(), PPx::BevelButton::GetCenterPopupGlyph(), PPx::PopupButton::GetCheckCurrentItemFlag(), PPx::RoundButton::GetContentInfo(), PPx::ImageWell::GetContentInfo(), PPx::IconControl::GetContentInfo(), PPx::BevelButton::GetContentInfo(), PPx::ViewUtils::GetControlThemeFontID(), PPx::PushButton::GetDefaultFlag(), PPx::IconPushButton::GetDefaultFlag(), PPx::ImageWell::GetDragDestinationFlag(), PPx::PopupButton::GetExtraHeight(), PPx::BevelButton::GetGraphicAlignment(), PPx::BevelButton::GetGraphicOffset(), PPx::IconControl::GetIconAlignment(), PPx::IconControl::GetIconResourceID(), PPx::IconControl::GetIconTransform(), PPx::BevelButton::GetIconTransform(), PPx::ImageWell::GetImageTransform(), PPx::ListBox::GetListHandle(), PPx::ClockControl::GetLongDate(), PPx::PopupGroupBox::GetMenuRef(), PPx::PopupButton::GetMenuRef(), PPx::BevelButton::GetMenuRef(), PPx::BevelButton::GetMenuValue(), PPx::PopupButton::GetOwnedMenuRef(), PPx::PictureControl::GetPicture(), PPx::ScrollBar::GetShowsArrowsFlag(), PPx::StaticText::GetText(), PPx::EditUnicodeText::GetText(), PPx::EditTextControl::GetText(), PPx::ComboBox::GetText(), PPx::BevelButton::GetTextAlignment(), PPx::BevelButton::GetTextOffset(), PPx::BevelButton::GetTextPlacement(), PPx::TextGroupBox::GetTitleRect(), PPx::PopupGroupBox::GetTitleRect(), PPx::CheckBoxGroupBox::GetTitleRect(), PPx::ProgressBar::IsAnimating(), PPx::ClockControl::IsAnimating(), PPx::ChasingArrows::IsAnimating(), PPx::ProgressBar::IsIndeterminate(), and PPx::IconControl::WriteState().

6.267.2.7 void PPx::View::GetFrame (HIRect & *outFrame*) const

Passes back the View's frame.

Parameters:

outFrame View's frame in coordinates of parent View

Definition at line 546 of file PPxView.cp.

References PPx::SysHIView::GetFrame().

Referenced by AdaptToSuperFrameSize().

6.267.2.8 void PPx::View::GetLocalFrame (HIRect & *outFrame*) const

Passes back the View's frame in local coordinates.

Parameters:

outFrame View's frame in local coordinates

Local coordinates for a View has (0, 0) at its top left corner

Definition at line 563 of file PPxView.cp.

References PPx::SysHIView::GetFrame().

Referenced by PPx::ThemeTextBox::DoControlDraw(), and PPx::GrayBox::DoControlDraw().

6.267.2.9 SInt32 PPx::View::GetMaxValue () const

Gets the maximum value for the View.

Returns:

Maximum value for the View

Definition at line 832 of file PPxView.cp.

References PPx::SysHIView::GetMaxValue().

Referenced by PPx::Slider::WriteState(), PPx::ScrollBar::WriteState(), PPx::RelevanceBar::WriteState(), PPx::ProgressBar::WriteState(), and PPx::LittleArrows::WriteState().

6.267.2.10 SInt32 PPx::View::GetMinValue () const

Gets the minimum value for the View.

Returns:
Minimum value for the View

Definition at line 803 of file PPxView.cp.

References PPx::SysHIView::GetMinValue().

Referenced by PPx::Slider::WriteState(), PPx::ScrollBar::WriteState(), PPx::RelevanceBar::WriteState(), PPx::ProgressBar::WriteState(), and PPx::LittleArrows::WriteState().

6.267.2.11 View ∗ PPx::View::GetSubViewByIndex (SInt32 *inIndex*) const

Returns subview specified by a zero-based index.

Parameters:
inIndex Zero-based index

Returns:
SubView at the specified index in subview list of View

Returns nil if index is out of range

Definition at line 444 of file PPxView.cp.

6.267.2.12 View ∗ PPx::View::GetSuperView () const

Returns the SuperView of the View.

Returns:
SuperView of the View

Definition at line 385 of file PPxView.cp.

Referenced by AddSubView().

6.267.2.13 HIViewRef PPx::View::GetSysView () const

Returns the system HIViewRef for the View.

Returns:
System HIViewRef for the View

Definition at line 166 of file PPxView.cp.

References PPx::SysHIView::GetSysView().

Referenced by AddSubView(), PPx::ComboBox::AppendListItem(), PPx::ComboBox::ChangeAttributes(), PPx::ImageView::CopyImage(), PPx::ImageView::GetAlpha(), PPx::ComboBox::GetAttributes(), PPx::ScrollView::GetAutoHideScrollBars(), PPx::RadioGroup::GetCurrentButton(), PPx::ComboBox::GetListItemsCount(), PPx::ComboBox::GetListItemText(), PPx::ImageView::GetScaleToFit(), GetSysWindow(), PPx::ScrollBar::GetViewSize(), PPx::ComboBox::InsertListItemAt(), PPx::ImageView::IsOpaque(), PPx::ComboBox::RemoveListItem(), PPx::ImageView::SetAlpha(), PPx::ScrollView::SetAutoHideScrollBars(), PPx::ImageView::SetImage(), PPx::ImageView::SetOpaque(), PPx::ImageView::SetScaleToFit(), and PPx::ScrollBar::SetViewSize().

6.267.2.14 WindowRef PPx::View::GetSysWindow () const

Returns the system WindowRef for the Window containing the View.

Returns:
System WindowRef containing the View

Definition at line 414 of file PPxView.cp.

References GetSysView().

6.267.2.15 CFString PPx::View::GetTitle () const

Gets the title of the View.

Returns:
Title of the View

Definition at line 862 of file PPxView.cp.

References PPx::SysHIView::GetTitle().

Referenced by PPx::TextGroupBox::WriteState(), PPx::RadioButton::WriteState(), PPx::PushButton::WriteState(), PPx::PopupGroupBox::WriteState(), PPx::PopupButton::WriteState(), PPx::IconPushButton::WriteState(), PPx::DisclosureTriangle::WriteState(), PPx::CheckBoxGroupBox::WriteState(), PPx::CheckBox::WriteState(), and PPx::BevelButton::WriteState().

6.267.2.16 SInt32 PPx::View::GetValue () const

Returns the View's value.

Returns:
View's value

Definition at line 774 of file PPxView.cp.

References PPx::SysHIView::GetValue().

Referenced by PPx::RadioGroup::GetCurrentButton(), PPx::Slider::WriteState(), PPx::ScrollBar::WriteState(), PPx::RelevanceBar::WriteState(), PPx::RadioButton::WriteState(), PPx::ProgressBar::WriteState(), PPx::PopupGroupBox::WriteState(), PPx::PopupButton::WriteState(), PPx::LittleArrows::WriteState(), PPx::DisclosureTriangle::WriteState(), PPx::DisclosureButton::WriteState(), PPx::CheckBoxGroupBox::WriteState(), and PPx::CheckBox::WriteState().

6.267.2.17 View ∗ PPx::View::GetViewObject (HIViewRef *inViewRef*) `[static]`

Returns the View object associated with a HIViewRef.

Parameters:
inViewRef System HIViewRef

Returns:
View object associated with the HIViewRef

Returns nil if the HIViewRef does not belong to a View created by the PPx::View class of the current program.

PPx adds a property to each HIViewRef with a pointer to associated View object, and tags that property with the signature (four-character creator code) of the program.

Therefore, a host program and plug-in modules or other external code call all use PPx and their Views won't get confused as long as they have different signatures.

Definition at line 937 of file PPxView.cp.

Referenced by PPx::RadioGroup::GetCurrentButton().

6.267.2.18 void PPx::View::Initialize (HIViewRef *inViewRef*, View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*) `[protected]`

Intializes from parameters.

Parameters:

inViewRef System HIViewRef for this view

inSuperView Parent view

inFrame Bounds for view, in local coordinates of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

Definition at line 130 of file PPxView.cp.

References AddSubView(), PPx::SysHIView::Adopt(), PPx::SysHIView::SetEnabled(), SetFrame(), PPx::SysHIView::SetProperty(), and PPx::SysHIView::SetVisible().

6.267.2.19 void PPx::View::Initialize (HIViewRef *inViewRef*, const HIRect & *inFrame*) `[protected]`

Intializes from parameters.

Parameters:

inViewRef System HIViewRef for this view

inFrame Bounds for view, in local coordinates of parent

Creates View with no superview

Definition at line 101 of file PPxView.cp.

References PPx::SysHIView::Adopt(), SetFrame(), and PPx::SysHIView::SetProperty().

Referenced by InitViewState().

6.267.2.20 void PPx::View::InitViewState (HIViewRef *inViewRef*, const DataReader & *inReader*) `[protected]`

Initialize view from persistent data.

Parameters:

inViewRef System HIViewRef for this View

inReader Data dictionary from which to read persistent data

View does not use an override of Peristent::InitState() because it requires a valid HIViewRef before it can initialize itself.

A View subclass should call this function from its override of InitState() after creating its HIViewRef.

Definition at line 220 of file PPxView.cp.

References AddSubView(), Initialize(), PPx::Attachable::ReadAttachments(), PPx::DataReader::ReadObjectValue(), PPx::DataReader::ReadOptional(), PPx::SysHIView::SetEnabled(), PPx::Identifiable::SetID(), and PPx::SysHIView::SetVisible().

Referenced by PPx::BaseView::InitState().

6.267.2.21 bool PPx::View::IsActive () const

Returns whether the View is active.

Returns:
Whether the View is active

Definition at line 700 of file PPxView.cp.

References PPx::SysHIView::IsActive().

Referenced by PPx::ThemeTextBox::DoControlDraw().

6.267.2.22 bool PPx::View::IsEnabled () const

Returns whether the View is enabled.

Returns:
Whether the View is enabled

Definition at line 729 of file PPxView.cp.

References PPx::SysHIView::IsEnabled().

6.267.2.23 bool PPx::View::IsVisible () const

Returns whether the View is visible.

Returns:
Whether the View is visible

Definition at line 671 of file PPxView.cp.

References PPx::SysHIView::IsVisible().

6.267.2.24 void PPx::View::RemoveSubView (View ∗ *inSubView*) [protected]

Removes a subview from a View.

Parameters:

inSubView SubView to remove

Definition at line 368 of file PPxView.cp.

Referenced by RemoveFromSuperView().

6.267.2.25 void PPx::View::SetActive (bool *inActivate*)

Makes the View active or inactive.

Parameters:

inActivate Whether to activate or deactivate the View

Definition at line 714 of file PPxView.cp.

References PPx::SysHIView::SetActive().

6.267.2.26 void PPx::View::SetDataTag (SInt16 *inPartCode*, FourCharCode *inTag*, Size *inDataSize*, const void ∗ *inDataPtr*)

Sets a View property specified by a data tag.

Parameters:

inPartCode Part of View to which the property applies

inTag Data tag ID

inDataSize Byte size of data

inDataPtr Pointer to data buffer

Definition at line 879 of file PPxView.cp.

References PPx::SysHIView::SetDataTag().

Referenced by PPx::StaticText::GetFontStyle(), PPx::ProgressBar::SetAnimating(), PPx::ClockControl::SetAnimating(), PPx::ChasingArrows::SetAnimating(), PPx::RoundButton::SetButtonSize(), PPx::PushButton::SetCancelFlag(), PPx::IconPushButton::SetCancelFlag(), PPx::BevelButton::SetCenterPopupGlyph(), PPx::PopupButton::SetCheckCurrentItemFlag(), PPx::RoundButton::SetContentInfo(), PPx::ImageWell::SetContentInfo(), PPx::IconControl::SetContentInfo(), PPx::BevelButton::SetContentInfo(), PPx::ViewUtils::SetControlThemeFontID(), PPx::PushButton::SetDefaultFlag(), PPx::IconPushButton::SetDefaultFlag(), PPx::ImageWell::SetDragDestinationFlag(), PPx::PopupButton::SetExtraHeight(), PPx::StaticText::SetFontStyle(), PPx::BevelButton::SetGraphicAlignment(), PPx::BevelButton::SetGraphicOffset(), PPx::IconControl::SetIconAlignment(), PPx::IconControl::SetIconResourceID(), PPx::IconControl::SetIconTransform(), PPx::BevelButton::SetIconTransform(), PPx::ImageWell::SetImageTransform(), PPx::ProgressBar::SetIndeterminate(), PPx::ClockControl::SetLongDate(), PPx::PopupButton::SetMenuID(), PPx::PopupGroupBox::SetMenuRef(), PPx::PopupButton::SetMenuRef(), PPx::BevelButton::SetMenuRef(), PPx::BevelButton::SetMenuValue(), PPx::PopupButton::SetOwnedMenuRef(), PPx::PictureControl::SetPicture(), PPx::ScrollBar::SetShowsArrowsFlag(), PPx::StaticText::SetText(), PPx::EditUnicodeText::SetText(), PPx::EditTextControl::SetText(), PPx::ComboBox::SetText(), PPx::BevelButton::SetTextAlignment(), PPx::BevelButton::SetTextOffset(), and PPx::BevelButton::SetTextPlacement().

6.267.2.27 void PPx::View::SetEnabled (bool *inEnable*)

Makes the View enabled or disabled.

Parameters:

inEnable Whether to enable or diable the View

Definition at line 743 of file PPxView.cp.

References PPx::SysHIView::SetEnabled().

6.267.2.28 void PPx::View::SetFrame (const HIRect & *inFrame*)

Sets the frame of the View.

Parameters:

inFrame New frame for View, in coordinates of parent View

Definition at line 531 of file PPxView.cp.

References PPx::SysHIView::SetFrame().

Referenced by AdaptToSuperFrameSize(), and Initialize().

6.267.2.29 void PPx::View::SetFrameAdapter (FrameAdapter ∗ *inAdapter*)

Sets the FrameAdapter object for the View.

Parameters:

inAdapter FrameAdapter object

A FrameAdapter controls how a view moves and/or resizes when its superview changes size. View takes ownership of the FrameAdapter and is responsible for deleting it.

Definition at line 585 of file PPxView.cp.

6.267.2.30 void PPx::View::SetMaxValue (SInt32 *inMaxValue*)

Sets the maximum value for the View.

Parameters:

inMaxValue New maximum value for the View

Definition at line 817 of file PPxView.cp.

References PPx::SysHIView::SetMaxValue().

6.267.2.31 void PPx::View::SetMinValue (SInt32 *inMinValue*)

Sets the minimum value for the View.

Parameters:

inMinValue New minimum value for the View

Definition at line 788 of file PPxView.cp.

References PPx::SysHIView::SetMinValue().

6.267.2.32 void PPx::View::SetTitle (CFStringRef *inTitle*)

Sets the title of the View.

Parameters:

inTitle New title for the View

Definition at line 847 of file PPxView.cp.

References PPx::SysHIView::SetTitle().

6.267.2.33 void PPx::View::SetValue (SInt32 *inValue*)

Sets the View’s value.

Parameters:

inValue New value for the View

Definition at line 759 of file PPxView.cp.

References PPx::SysHIView::SetValue().

Referenced by PPx::PopupGroupBox::InitState(), and PPx::PopupButton::InitState().

6.267.2.34 void PPx::View::SetVisible (bool *inMakeVisible*)

Makes the View visible or invisible.

Parameters:

inMakeVisible Whether to show or hide the View

Definition at line 685 of file PPxView.cp.

References PPx::SysHIView::SetVisible().

6.267.2.35 void PPx::View::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::BaseView, PPx::WindowContentView, PPx::GrayBox, PPx::BevelButton, PPx::CheckBox, PPx::CheckBoxGroupBox, PPx::ClockControl, PPx::ComboBox, PPx::DisclosureButton, PPx::DisclosureTriangle, PPx::EditTextControl, PPx::EditUnicodeText, PPx::IconControl, PPx::IconPushButton, PPx::ImageView, PPx::ImageWell, PPx::ListBox, PPx::LittleArrows, PPx::PictureControl, PPx::PopupArrow, PPx::PopupButton, PPx::PopupGroupBox, PPx::ProgressBar, PPx::PushButton, PPx::RadioButton, PPx::RelevanceBar, PPx::RoundButton, PPx::ScrollBar, PPx::ScrollView, PPx::Slider, PPx::StaticText, PPx::TabView, PPx::TextGroupBox, PPx::WindowHeader, PPx::MLTEView, and PPx::ThemeTextBox.

Definition at line 264 of file PPxView.cp.

References PPx::SysHIView::GetFrame(), PPx::Identifiable::GetID(), PPx::SysHIView::IsEnabled(), PPx::SysHIView::IsVisible(), PPx::Attachable::WriteAttachments(), PPx::DataWriter::WriteObjectValue(), PPx::DataWriter::WriteValue(), and WriteViewHierarchy().

6.267.2.36 void PPx::View::WriteViewHierarchy (DataWriter & *ioWriter*) const [protected]

Write subviews to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Note:

You should not call this function. Call WriteState instead.

Definition at line 297 of file PPxView.cp.

References PPx::DataWriter::WriteObject(), and PPx::DataWriter::WriteObjectValue().

Referenced by PPx::WindowContentView::WriteState(), and WriteState().

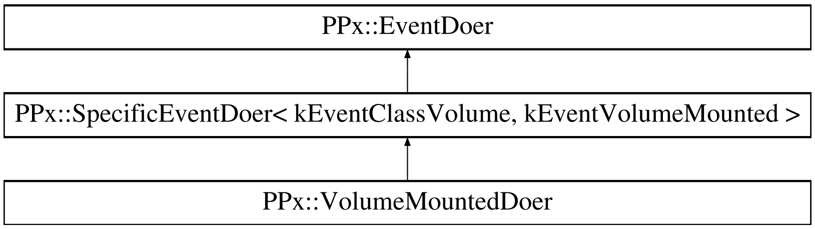
The documentation for this class was generated from the following files:

- PPxView.h
- PPxView.cp

6.268 PPx::VolumeMountedDoer Class Reference

`#include <PPxMiscellaneousEvents.h>`

Inheritance diagram for PPx::VolumeMountedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassVolume, kEventVolumeMounted >
PPx::VolumeMountedDoer

6.268.1 Detailed Description

Notification that a volume has been mounted.

Definition at line 22 of file PPxMiscellaneousEvents.h.

Protected Member Functions

- virtual OSStatus **DoVolumeMounted** (SysCarbonEvent &ioEvent, FSVolumeRefNum inVolumeRefNum)=0

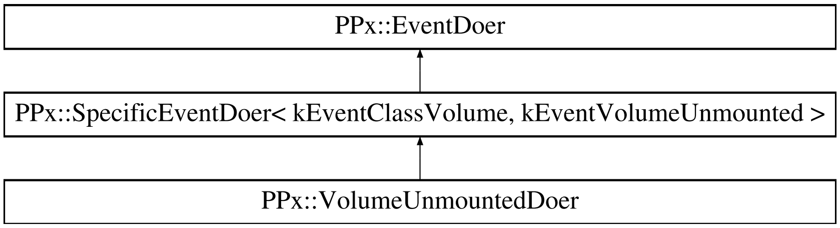
The documentation for this class was generated from the following files:

- PPxMiscellaneousEvents.h
- PPxMiscellaneousEvents.cp

6.269 PPx::VolumeUnmountedDoer Class Reference

```
#include <PPxMiscellaneousEvents.h>
```

Inheritance diagram for PPx::VolumeUnmountedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassVolume, kEventVolumeUnmounted >
PPx::VolumeUnmountedDoer

6.269.1 Detailed Description

Notification that a volume has been unmounted.

Definition at line 38 of file PPxMiscellaneousEvents.h.

Protected Member Functions

- virtual OSStatus **DoVolumeUnmounted** (SysCarbonEvent &ioEvent, FSVolumeRefNum inVolumeRefNum)=0

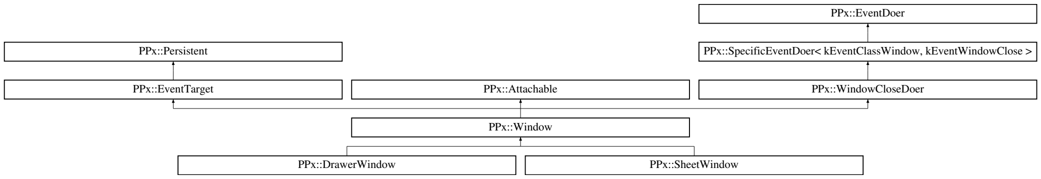
The documentation for this class was generated from the following files:

- PPxMiscellaneousEvents.h
- PPxMiscellaneousEvents.cp

6.270 PPx::Window Class Reference

```
#include <PPxWindow.h>
```

Inheritance diagram for PPx::Window::



6.270.1 Detailed Description

Window for displaying data on screen.

Definition at line 30 of file PPxWindow.h.

Public Member Functions

- Window ()

  Default constructor.

- virtual ∼Window ()

  Destructor.

- void Initialize (WindowClass inWindowClass, WindowAttributes inWindAttrs, const Rect &inContentBounds, CFStringRef inTitle)

  Initializes from parameters.

- WindowRef GetSysWindow () const

  Returns the System WindowRef associated with the Window.

- View ∗ GetContentView () const

  Returns the content view of the Window.

- void AddSubView (View ∗inSubView)

  Adds a subview to the content view of a Window.

- void Select ()

  Brings the Window to the front of its layer and activates it.

- bool IsVisible () const

  Returns whether the Window is visible.

- void Show ()

  Makes the Window visible.

- void Hide ()

  Makes the Window invisible.

- void SetTitle (CFStringRef inTitle)

  Sets the title of the Window.

- CFString GetTitle () const

  Returns the title of the Window.

- void Close ()

  Closes the window.

Static Public Member Functions

- Window ∗ GetWindowObject (WindowRef inWindowRef)

  Returns the Window object associated with a WindowRef.

- void SetDefaultAttributes (WindowAttributes inWindAttrs)

  Sets the window attributes used for all Toolbox windows.

- WindowAttributes GetDefaultAttributes ()

  Returns the window attributes used for all Toolbox window.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus DoWindowClose (SysCarbonEvent &ioEvent, WindowRef inWindow)

  Handles window close event.

6.270.2 Member Function Documentation

6.270.2.1 void PPx::Window::AddSubView (View ∗ *inSubView*)

Adds a subview to the content view of a Window.

Parameters:

inSubView SubView to add

Definition at line 245 of file PPxWindow.cp.

6.270.2.2 void PPx::Window::Close ()

Closes the window.

Posts a ”close” CarbonEvent for the Window

Definition at line 341 of file PPxWindow.cp.

References PPx::EventTarget::GetSysEventTarget(), and PPx::SysCarbonEvent::PostTo().

6.270.2.3 OSStatus PPx::Window::DoWindowClose (SysCarbonEvent & *ioEvent*, WindowRef *inWindow*) `[protected, virtual]`

Handles window close event.

Parameters:

ioEvent CarbonEvent for window close

inWindow Window to close

Returns:

Status of event handling. Alwyas returns noErr.

Implements PPx::WindowCloseDoer.

Definition at line 360 of file PPxWindow.cp.

6.270.2.4 View ∗ PPx::Window::GetContentView () const

Returns the content view of the Window.

Returns:

Content view of the Window

A Window is not a View. The content view is the top-level View within a Window. Add subviews to the content view if you want to have them displayed in a Window.

Definition at line 231 of file PPxWindow.cp.

6.270.2.5 WindowAttributes PPx::Window::GetDefaultAttributes () [static]

Returns the window attributes used for all Toolbox window.

Returns:

Toolbox window attributes bit flags

Definition at line 429 of file PPxWindow.cp.

6.270.2.6 WindowRef PPx::Window::GetSysWindow () const

Returns the System WindowRef associated with the Window.

Returns:

System WindowRef associated with the Window

Definition at line 213 of file PPxWindow.cp.

References PPx::SysWindow::GetWindowRef().

Referenced by PPx::DrawerWindow::CloseDrawer(), PPx::DrawerWindow::GetCurrentEdge(), PPx::DrawerWindow::GetDrawerOffsets(), PPx::DrawerWindow::GetDrawerState(), PPx::SheetWindow::GetParentWindow(), PPx::DrawerWindow::GetParentWindow(), PPx::DrawerWindow::GetPreferredEdge(), PPx::SheetWindow::Hide(), PPx::DrawerWindow::OpenDrawer(), PPx::DrawerWindow::SetDrawerOffsets(), PPx::DrawerWindow::SetParentWindow(), PPx::DrawerWindow::SetPreferredEdge(), PPx::SheetWindow::Show(), and PPx::DrawerWindow::Toggle().

6.270.2.7 CFString PPx::Window::GetTitle () const

Returns the title of the Window.

Returns:

Title fo the Window

Definition at line 326 of file PPxWindow.cp.

References PPx::SysWindow::GetTitle().

6.270.2.8 Window ∗ PPx::Window::GetWindowObject (WindowRef *inWindowRef*) [static]

Returns the Window object associated with a WindowRef.

Parameters:

inWindowRef System window reference

Returns:

PPx Window object

Returns nil if the WindowRef does not belong to a Window created by PPx::Window class of the current program.

PPx adds a property to each WindowRef with a pointer to associated Window object, and tags that property with the signature (four-character creator code) of the program.

Therefore, a host program and plug-in modules or other external code call all use PPx and their Windows won't get confused as long as they have different signatures.

Definition at line 392 of file PPxWindow.cp.

Referenced by PPx::DrawerWindow::GetParentWindow().

6.270.2.9 void PPx::Window::Initialize (WindowClass *inWindClass*, WindowAttributes *inWindAttrs*, const Rect & *inContentBounds*, CFStringRef *inTitle*)

Initializes from parameters.

Parameters:

inWindClass Toolbox window class

inWindAttrs Toolbox window attributes

inContentBounds Bounding box of content area of window in global coordinates

inTitle Title for the window

See <MacWindow.h> for information about Toolbox window classes and attributes.

The default attributes for PPx::Window are automatically added. Call PPx::Window::SetDefaultAttributes to change the default attributes which apply to all PPx Windows.

Definition at line 83 of file PPxWindow.cp.

References PPx::EventTarget::GetSysEventTarget(), PPx::SysWindow::GetWindowRef(), PPx::SysWindow::MakeWindow(), PPx::SysWindow::SetProperty(), and PPx::SysWindow::SetTitle().

Referenced by InitState().

6.270.2.10 void PPx::Window::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::DrawerWindow.

Definition at line 142 of file PPxWindow.cp.

References Initialize(), PPx::Attachable::ReadAttachments(), PPx::DataReader::ReadObjectValue(), PPx::DataReader::ReadOptional(), and PPx::DataReader::ReadRequired().

6.270.2.11 bool PPx::Window::IsVisible () const

Returns whether the Window is visible.

Returns:

Whether the Window is visible

Definition at line 273 of file PPxWindow.cp.

References PPx::SysWindow::IsVisible().

6.270.2.12 void PPx::Window::SetDefaultAttributes (WindowAttributes *inWindAttrs*) `[static]`

Sets the window attributes used for all Toolbox windows.

Parameters:

inWindAttrs Toolbox window attributes bit flags

Definition at line 414 of file PPxWindow.cp.

6.270.2.13 void PPx::Window::SetTitle (CFStringRef *inTitle*)

Sets the title of the Window.

Parameters:

inTitle New title for the Window

Definition at line 311 of file PPxWindow.cp.

References PPx::SysWindow::SetTitle().

6.270.2.14 void PPx::Window::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::Persistent.

Reimplemented in PPx::DrawerWindow.

Definition at line 174 of file PPxWindow.cp.

References PPx::SysWindow::GetBounds(), PPx::SysWindow::GetTitle(), PPx::SysWindow::GetWindowAttributes(), PPx::SysWindow::GetWindowClass(), PPx::Attachable::WriteAttachments(), PPx::DataWriter::WriteObjectValue(), and PPx::DataWriter::WriteValue().

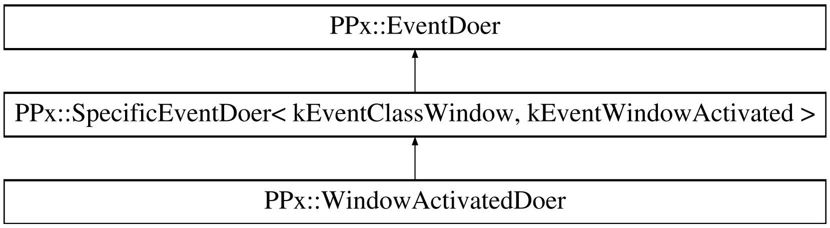
The documentation for this class was generated from the following files:

- PPxWindow.h
- PPxWindow.cp

6.271 PPx::WindowActivatedDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowActivatedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowActivated >
PPx::WindowActivatedDoer

6.271.1 Detailed Description

Handles a window being activated.

Definition at line 52 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowActivated** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.272 PPx::WindowAttributesStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.272.1 Detailed Description

Wrapper for WindowAttributes.

Definition at line 132 of file PPxSysTypes.h.

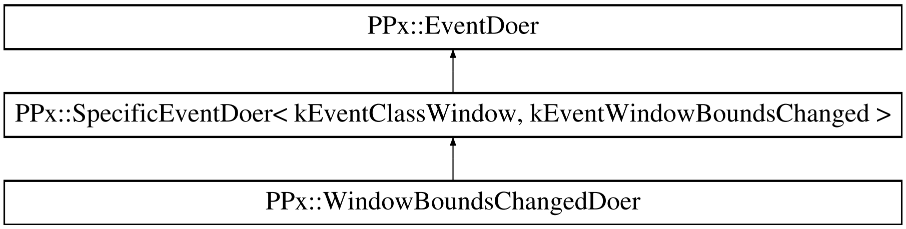
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.273 PPx::WindowBoundsChangedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowBoundsChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowBoundsChanged >
PPx::WindowBoundsChangedDoer

6.273.1 Detailed Description

Handles a window having been moved and/or resized.

Definition at line 365 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowBoundsChanged** (SysCarbonEvent &ioEvent, WindowRef inWindow, UInt32 inAttributes, const Rect &inOriginalBounds, const Rect &inPreviousBounds, const Rect &inCurrentBounds)=0

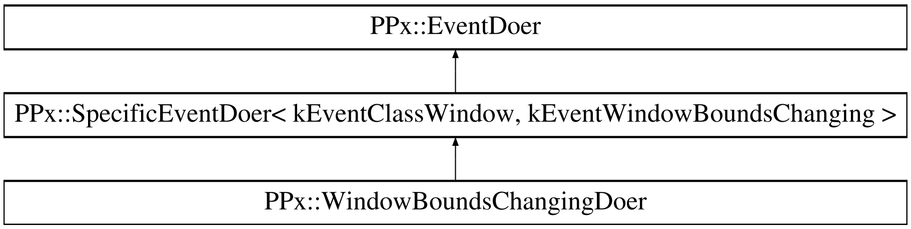
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.274 PPx::WindowBoundsChangingDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowBoundsChangingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowBoundsChanging >
PPx::WindowBoundsChangingDoer

6.274.1 Detailed Description

Handles a window being moved and/or resized.

Definition at line 345 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowBoundsChanging** (SysCarbonEvent &ioEvent, WindowRef inWindow, UInt32 inAttributes, const Rect &inOriginalBounds, const Rect &inPreviousBounds, Rect &ioCurrentBounds)=0

The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.275 PPx::WindowClassStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.275.1 Detailed Description

Wrapper for WindowClass.

Definition at line 125 of file PPxSysTypes.h.

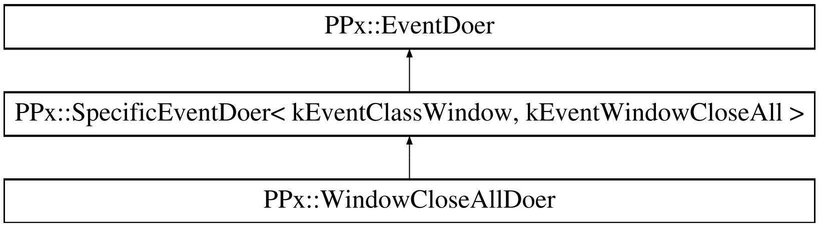
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.276 PPx::WindowCloseAllDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowCloseAllDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCloseAll >
PPx::WindowCloseAllDoer

6.276.1 Detailed Description

Handles a request to close all windows.

Definition at line 465 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowCloseAll** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

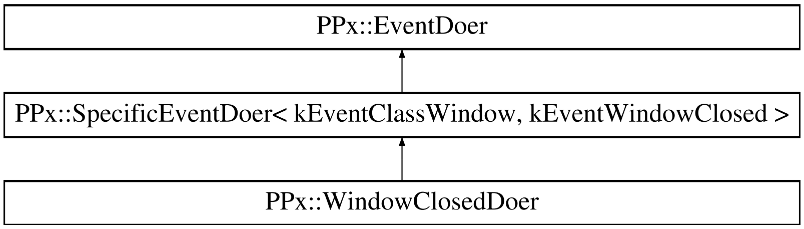
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.277 PPx::WindowClosedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowClosedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowClosed >
PPx::WindowClosedDoer

6.277.1 Detailed Description

Handles a window about to be disposed.

Definition at line 481 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowClosed** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

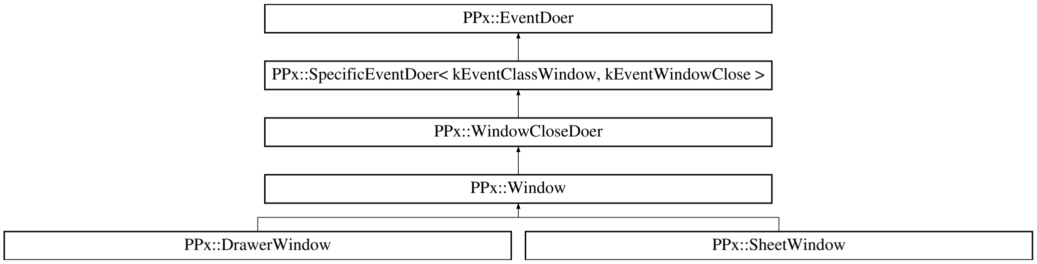
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.278 PPx::WindowCloseDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowCloseDoer::



6.278.1 Detailed Description

Handles a request to close a window.

Definition at line 449 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowClose** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

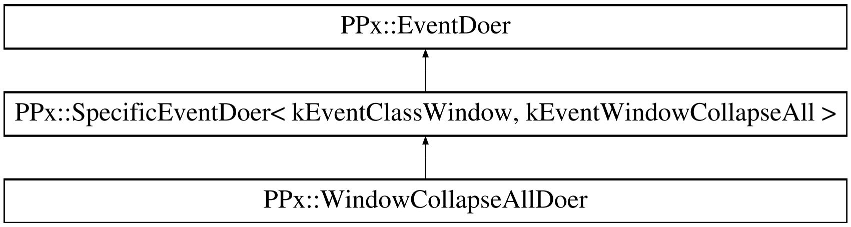
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.279 PPx::WindowCollapseAllDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowCollapseAllDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapseAll >
PPx::WindowCollapseAllDoer

6.279.1 Detailed Description

Handles a request to collapse all windows.

Definition at line 185 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowCollapseAll** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

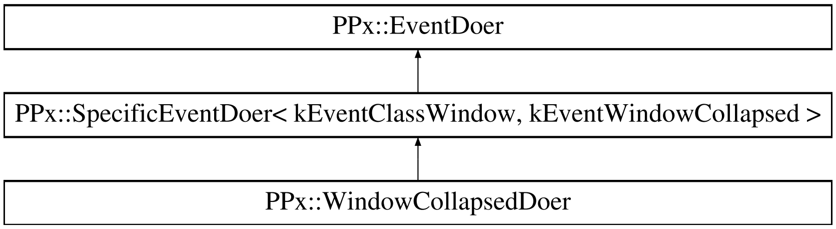
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.280 PPx::WindowCollapsedDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowCollapsedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapsed >
PPx::WindowCollapsedDoer

6.280.1 Detailed Description

Handles a window after being collapsed.

Definition at line 217 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowCollapsed** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

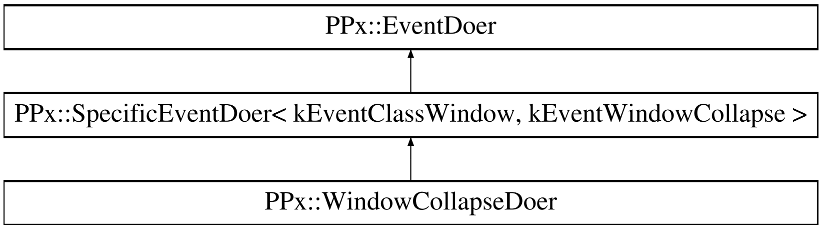
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.281 PPx::WindowCollapseDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowCollapseDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapse >
PPx::WindowCollapseDoer

6.281.1 Detailed Description

Handles a request to collapse a window.

Definition at line 169 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowCollapse** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

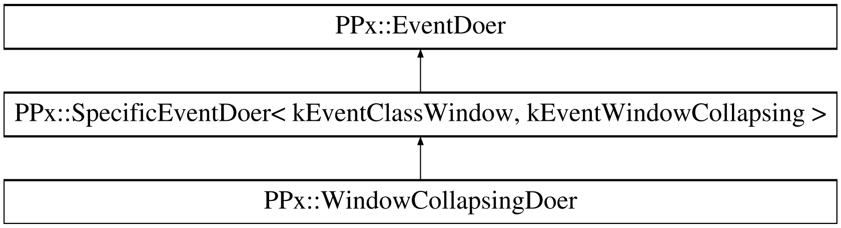
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.282 PPx::WindowCollapsingDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowCollapsingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCollapsing >
PPx::WindowCollapsingDoer

6.282.1 Detailed Description

Handles a window about to be collapsed.

Definition at line 201 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowCollapsing** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

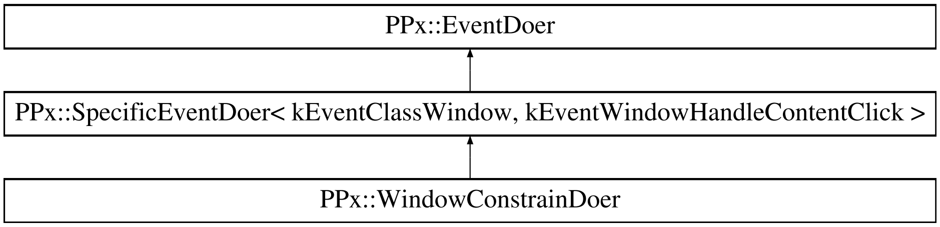
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.283 PPx::WindowConstrainDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowConstrainDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowHandleContentClick >
PPx::WindowConstrainDoer

6.283.1 Detailed Description

Handles notification that the available window area has changed.

Definition at line 598 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowConstrain** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

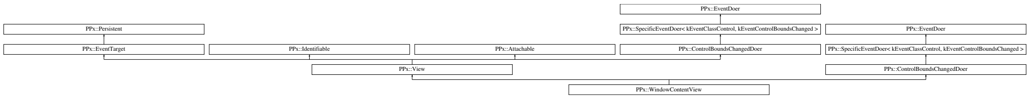
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.284 PPx::WindowContentView Class Reference

`#include <PPxWindowContentView.h>`

Inheritance diagram for PPx::WindowContentView::


PPx::EventDoer
PPx::Persistent
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::EventDoer
PPx::EventTarget
PPx::Identifiable
PPx::Attachable
PPx::ControlBoundsChangedDoer
PPx::SpecificEventDoer< kEventClassControl, kEventControlBoundsChanged >
PPx::View
PPx::ControlBoundsChangedDoer
PPx::WindowContentView

6.284.1 Detailed Description

Top-level view for the contents of a window.

Definition at line 22 of file PPxWindowContentView.h.

Public Member Functions

- WindowContentView ()

  Default constructor.

- virtual ~WindowContentView ()

  Destructor.

- void Initialize (WindowRef inSysWindow)

  Initialize from a WindowRef.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

- virtual OSStatus **DoControlBoundsChanged** (SysCarbonEvent &ioEvent, ControlRef inControl, UInt32 inChangeAttributes, const HIRect &inOriginalBounds, const HIRect &inCurrentBounds)

6.284.2 Member Function Documentation

6.284.2.1 void PPx::WindowContentView::Initialize (WindowRef *inSysWindow*)

Initialize from a WindowRef.

Parameters:

inSysWindow Window reference

Definition at line 38 of file PPxWindowContentView.cp.

References PPx::View::AdoptSysView(), and PPx_ThrowIfOSError_.

6.284.2.2 void PPx::WindowContentView::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 64 of file PPxWindowContentView.cp.

References PPx::Attachable::ReadAttachments().

6.284.2.3 void PPx::WindowContentView::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 82 of file PPxWindowContentView.cp.

References PPx::Attachable::WriteAttachments(), and PPx::View::WriteViewHierarchy().

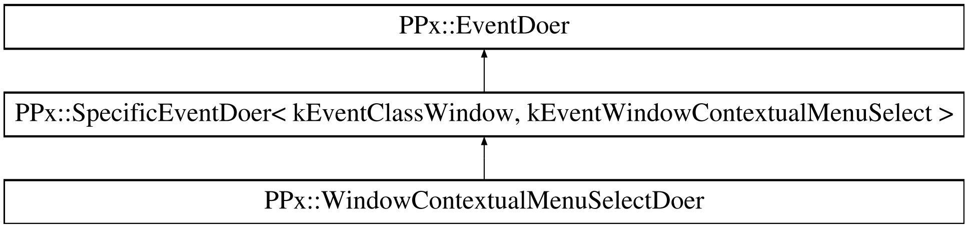
The documentation for this class was generated from the following files:

- PPxWindowContentView.h
- PPxWindowContentView.cp

6.285 PPx::WindowContextualMenuSelectDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowContextualMenuSelectDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowContextualMenuSelect >
PPx::WindowContextualMenuSelectDoer

6.285.1 Detailed Description

Handles a click in a window intended to invoke a contextual menu.

Definition at line 515 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowContextualMenuSelect** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

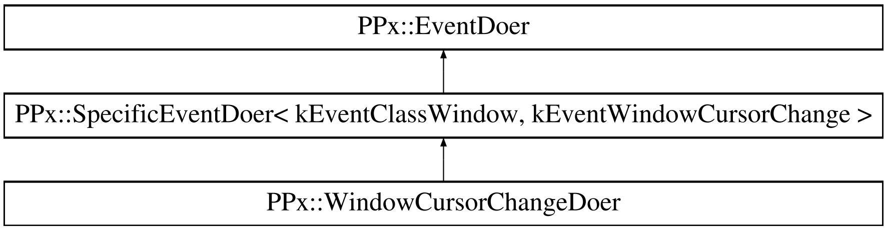
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.286 PPx::WindowCursorChangeDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowCursorChangeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowCursorChange >
PPx::WindowCursorChangeDoer

6.286.1 Detailed Description

Handles changing the cursor when the mouse is inside a window.

Definition at line 497 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowCursorChange** (SysCarbonEvent &ioEvent, WindowRef inWindow, Point inMouseLocation, UInt32 inKeyModifiers)=0

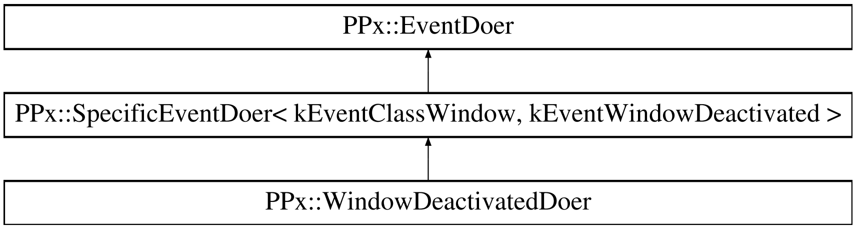
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.287 PPx::WindowDeactivatedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowDeactivatedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDeactivated >
PPx::WindowDeactivatedDoer

6.287.1 Detailed Description

Handles a window being deactivated.

Definition at line 68 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDeactivated** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.288 PPx::WindowDefPartCodeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.288.1 Detailed Description

Wrapper for WindowDefPartCode.

Definition at line 140 of file PPxSysTypes.h.

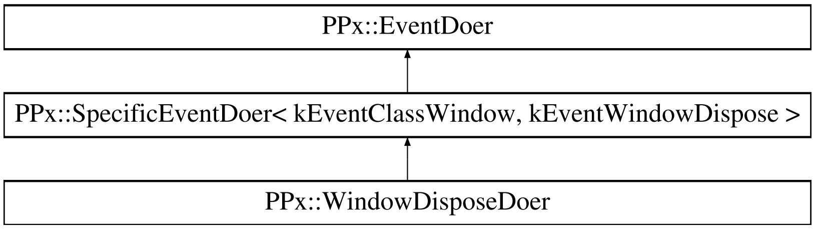
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.289 PPx::WindowDisposeDoer Class Reference

`#include <PPxWindowDefEvents.h>`

Inheritance diagram for PPx::WindowDisposeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDispose >
PPx::WindowDisposeDoer

6.289.1 Detailed Description

Disposes a window definition.

Definition at line 105 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDispose** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

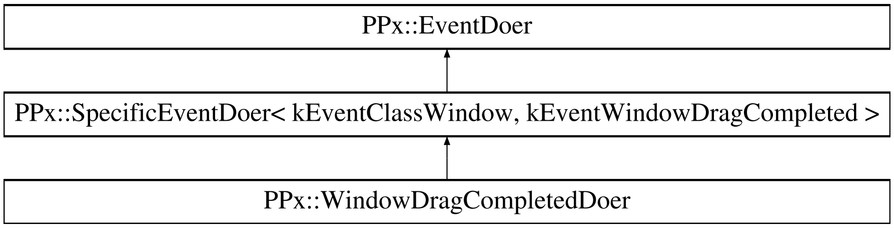
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.290 PPx::WindowDragCompletedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowDragCompletedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDragCompleted >
PPx::WindowDragCompletedDoer

6.290.1 Detailed Description

Handles a window finishing being dragged.

Definition at line 433 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDragCompleted** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

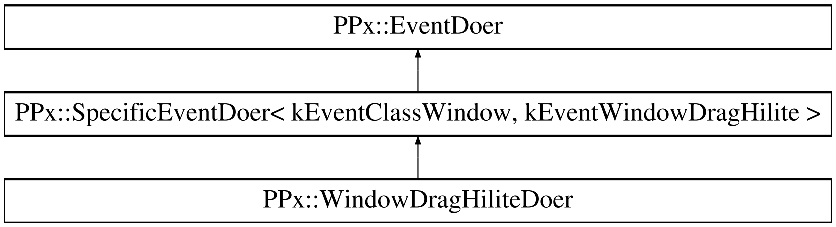
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.291 PPx::WindowDragHiliteDoer Class Reference

`#include <PPxWindowDefEvents.h>`

Inheritance diagram for PPx::WindowDragHiliteDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDragHilite >
PPx::WindowDragHiliteDoer

6.291.1 Detailed Description

Handles drag hiliting for a window.

Definition at line 121 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDragHilite** (SysCarbonEvent &ioEvent, WindowRef inWindow, bool inDrawHilite)=0

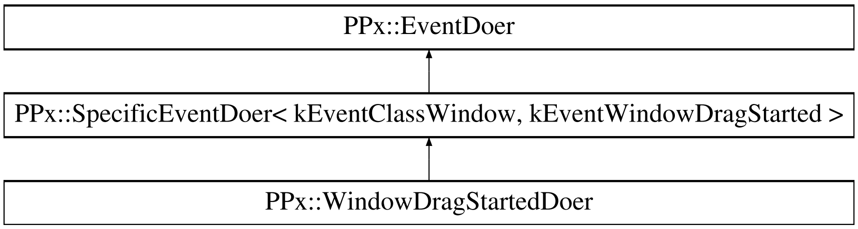
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.292 PPx::WindowDragStartedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowDragStartedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDragStarted >
PPx::WindowDragStartedDoer

6.292.1 Detailed Description

Handles a window starting to be dragged.

Definition at line 417 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDragStarted** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

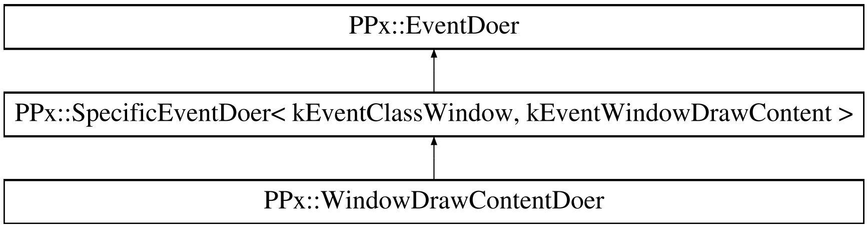
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.293 PPx::WindowDrawContentDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowDrawContentDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawContent >
PPx::WindowDrawContentDoer

6.293.1 Detailed Description

Handles drawing the contents of a window.

Definition at line 36 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawContent** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

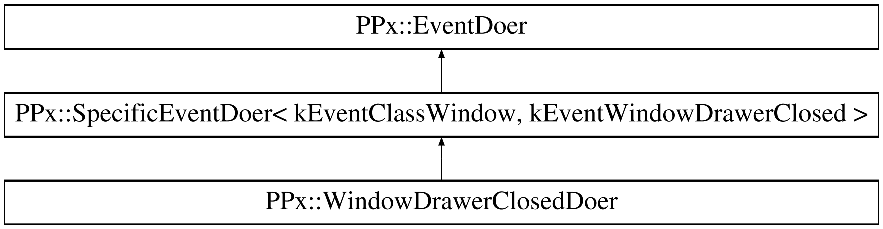
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.294 PPx::WindowDrawerClosedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowDrawerClosedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawerClosed >
PPx::WindowDrawerClosedDoer

6.294.1 Detailed Description

Handles a drawer being fully closed.

Definition at line 743 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawerClosed** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

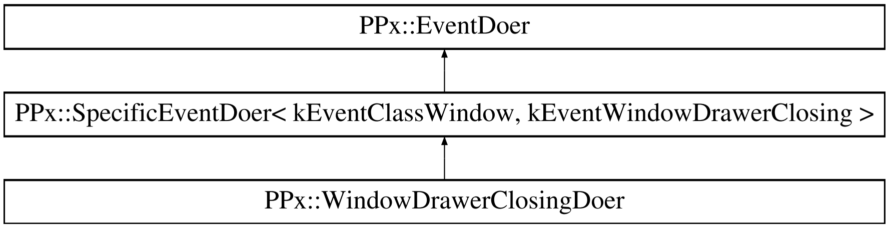
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.295 PPx::WindowDrawerClosingDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowDrawerClosingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawerClosing >
PPx::WindowDrawerClosingDoer

6.295.1 Detailed Description

Handles a drawer starting to close.

Definition at line 727 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawerClosing** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

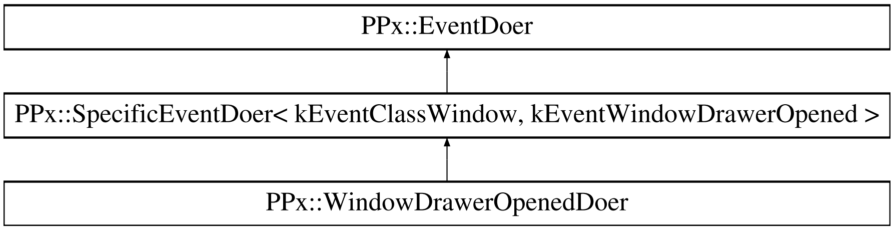
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.296 PPx::WindowDrawerOpenedDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowDrawerOpenedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawerOpened >
PPx::WindowDrawerOpenedDoer

6.296.1 Detailed Description

Handles a drawer being fully open.

Definition at line 711 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawerOpened** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

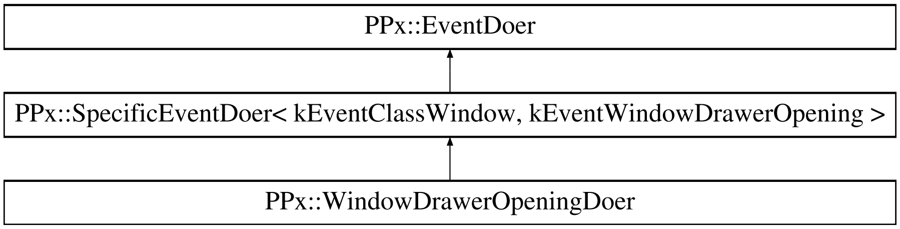
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.297 PPx::WindowDrawerOpeningDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowDrawerOpeningDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawerOpening >
PPx::WindowDrawerOpeningDoer

6.297.1 Detailed Description

Handles a drawer starting to open.

Definition at line 695 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawerOpening** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

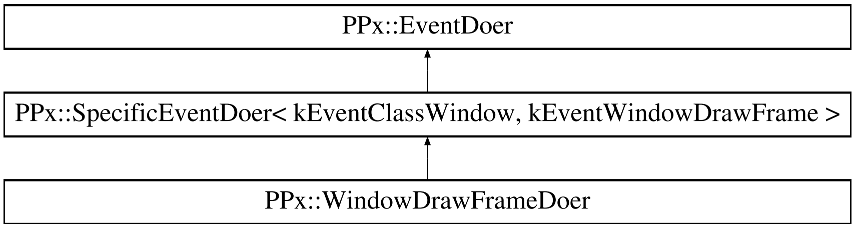
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.298 PPx::WindowDrawFrameDoer Class Reference

`#include <PPxWindowDefEvents.h>`

Inheritance diagram for PPx::WindowDrawFrameDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawFrame >
PPx::WindowDrawFrameDoer

6.298.1 Detailed Description

Draws a window's structure.

Definition at line 19 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawFrame** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

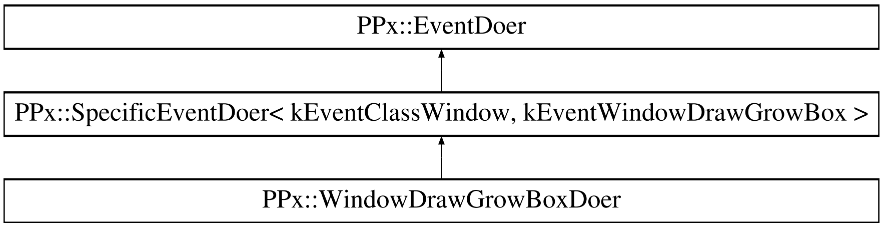
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.299 PPx::WindowDrawGrowBoxDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowDrawGrowBoxDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawGrowBox >
PPx::WindowDrawGrowBoxDoer

6.299.1 Detailed Description

Draws a window's grow box.

Definition at line 209 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawGrowBox** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

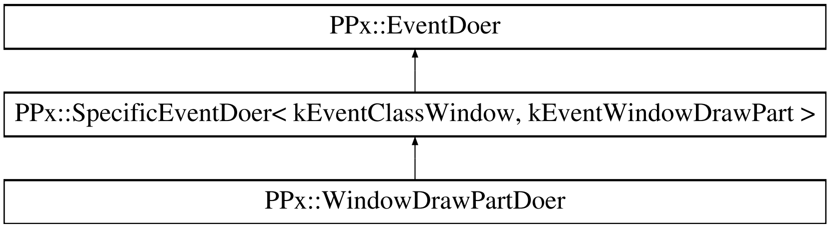
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.300 PPx::WindowDrawPartDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowDrawPartDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowDrawPart >
PPx::WindowDrawPartDoer

6.300.1 Detailed Description

Draws a specific part of a window's structure.

Definition at line 35 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowDrawPart** (SysCarbonEvent &ioEvent, WindowRef inWindow, WindowDefPartCode inPart)=0

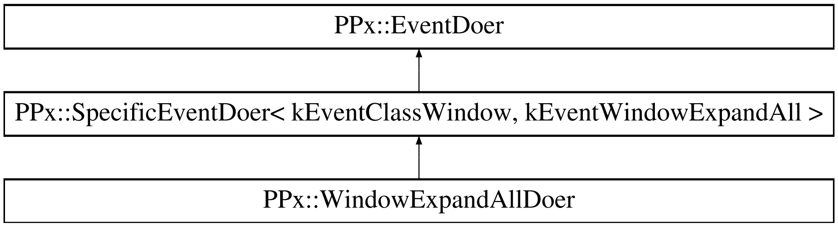
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.301 PPx::WindowExpandAllDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowExpandAllDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowExpandAll >
PPx::WindowExpandAllDoer

6.301.1 Detailed Description

Handles a request to expand all windows.

Definition at line 249 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowExpandAll** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

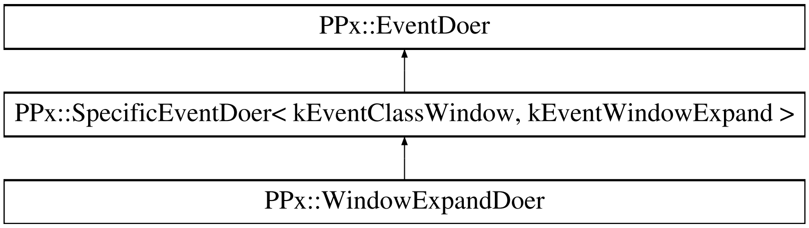
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.302 PPx::WindowExpandDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowExpandDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowExpand >
PPx::WindowExpandDoer

6.302.1 Detailed Description

Handles a request to expand a window.

Definition at line 233 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowExpand** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

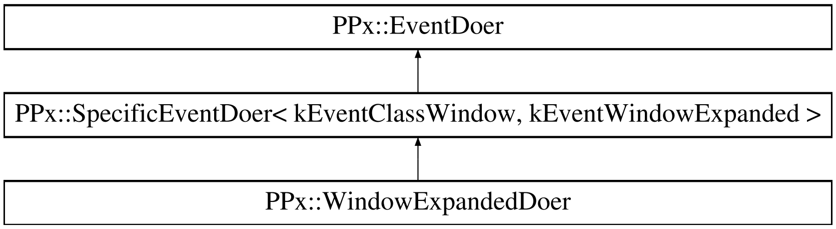
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.303 PPx::WindowExpandedDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowExpandedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowExpanded >
PPx::WindowExpandedDoer

6.303.1 Detailed Description

Handles a window after being expanded.

Definition at line 281 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowExpanded** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

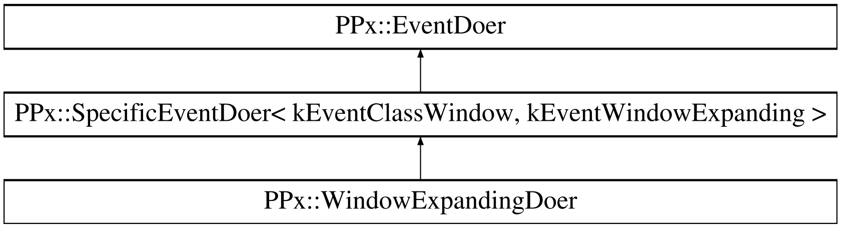
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.304 PPx::WindowExpandingDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowExpandingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowExpanding >
PPx::WindowExpandingDoer

6.304.1 Detailed Description

Handles a window about to be expanded.

Definition at line 265 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowExpanding** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

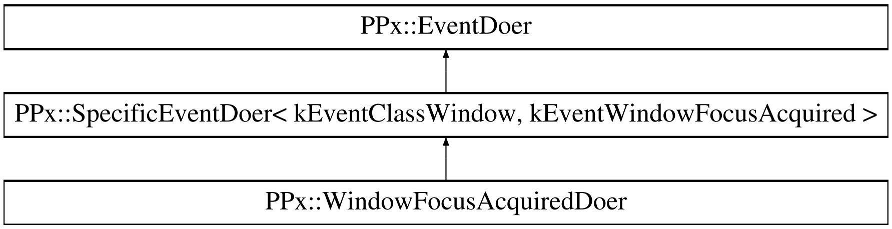
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.305 PPx::WindowFocusAcquiredDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowFocusAcquiredDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowFocusAcquired >
PPx::WindowFocusAcquiredDoer

6.305.1 Detailed Description

Handles a window acquiring the focus.

Definition at line 631 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowFocusAcquired** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

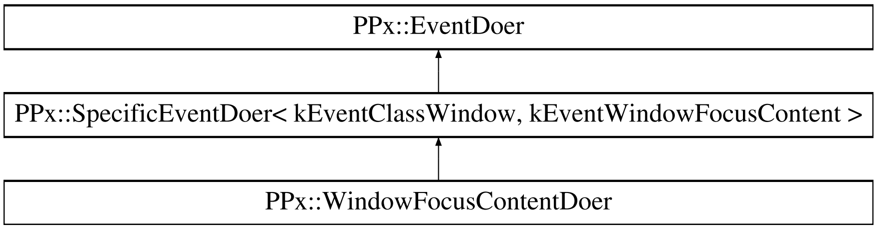
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.306 PPx::WindowFocusContentDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowFocusContentDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowFocusContent >
PPx::WindowFocusContentDoer

6.306.1 Detailed Description

Handles a setting the focus to the main view of a window.

Definition at line 663 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowFocusContent** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

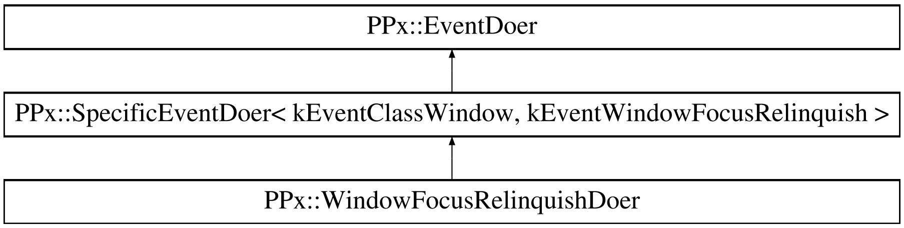
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.307 PPx::WindowFocusRelinquishDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowFocusRelinquishDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowFocusRelinquish >
PPx::WindowFocusRelinquishDoer

6.307.1 Detailed Description

Handles a window relinquishing the focus.

Definition at line 647 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowFocusRelinquish** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

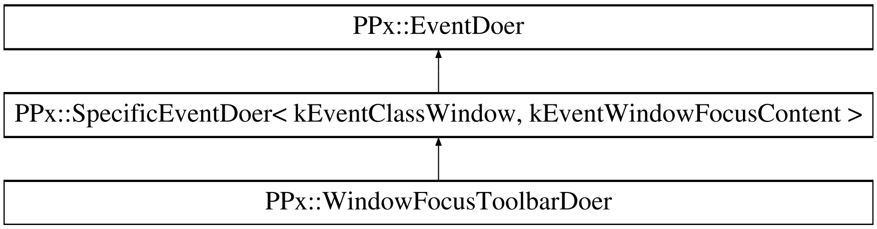
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.308 PPx::WindowFocusToolbarDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowFocusToolbarDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowFocusContent >
PPx::WindowFocusToolbarDoer

6.308.1 Detailed Description

Handles a setting the focus to the toolbar of a window.

Definition at line 679 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowFocusToolbar** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

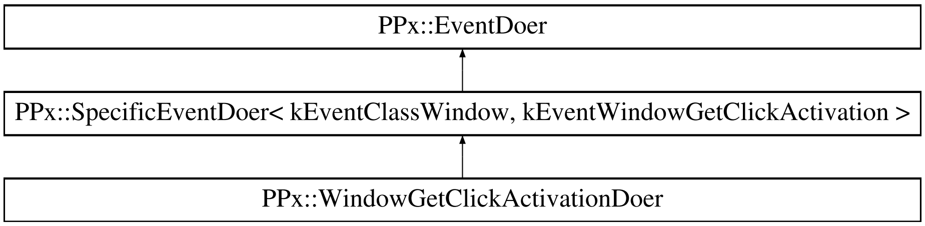
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.309 PPx::WindowGetClickActivationDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowGetClickActivationDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowGetClickActivation >
PPx::WindowGetClickActivationDoer

6.309.1 Detailed Description

Handles a window being activated by a mouse click.

Definition at line 84 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowGetClickActivation** (SysCarbonEvent &ioEvent, WindowRef inWindow, const HIPoint &inMouseLocation, UInt32 inKeyModifiers, WindowDefPartCode inWindowPart, ControlRef inControlHit, ClickActivationResult &outResult)=0

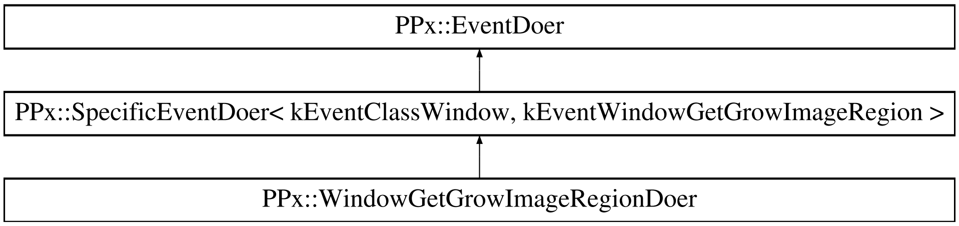
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.310 PPx::WindowGetGrowImageRegionDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowGetGrowImageRegionDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowGetGrowImageRegion >
PPx::WindowGetGrowImageRegionDoer

6.310.1 Detailed Description

Returns the outline for a window being resized.

Definition at line 225 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowGetGrowImageRegion** (SysCarbonEvent &ioEvent, WindowRef inWindow, const Rect &inGlobalBounds, RgnHandle ioImageRgn)=0

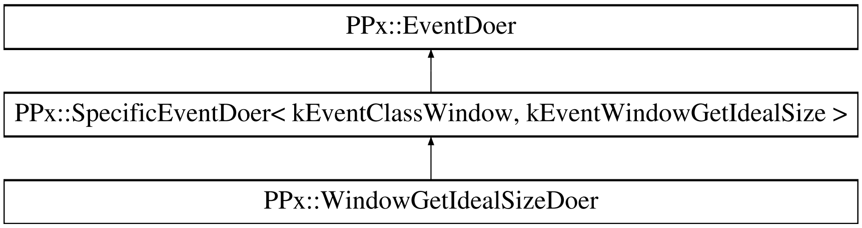
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.311 PPx::WindowGetIdealSizeDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowGetIdealSizeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowGetIdealSize >
PPx::WindowGetIdealSizeDoer

6.311.1 Detailed Description

Returns the ideal size of a window's content region.

Definition at line 547 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowGetIdealSize** (SysCarbonEvent &ioEvent, WindowRef inWindow, Point &outIdealSize)=0

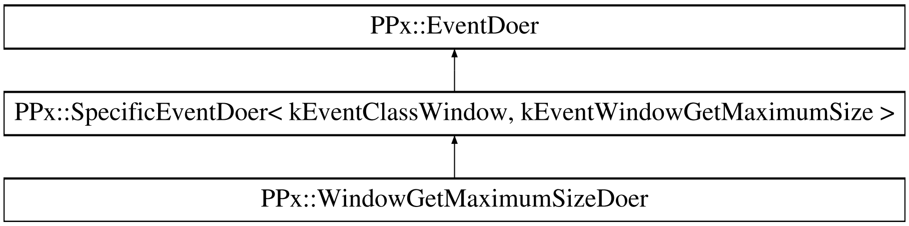
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.312 PPx::WindowGetMaximumSizeDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowGetMaximumSizeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowGetMaximumSize >
PPx::WindowGetMaximumSizeDoer

6.312.1 Detailed Description

Returns the maximum size of a window's content region.

Definition at line 581 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowGetMaximumSize** (SysCarbonEvent &ioEvent, WindowRef inWindow, Point &outMaximumSize)=0

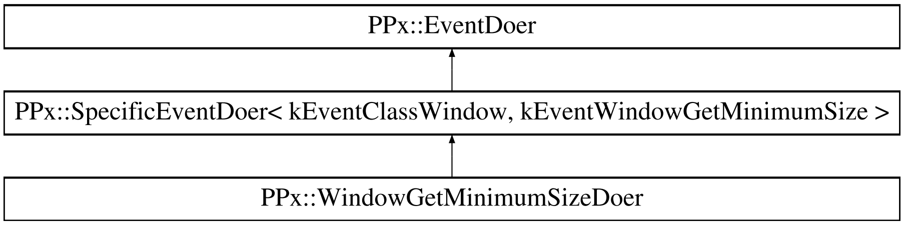
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.313 PPx::WindowGetMinimumSizeDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowGetMinimumSizeDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowGetMinimumSize >
PPx::WindowGetMinimumSizeDoer

6.313.1 Detailed Description

Returns the minimum size of a window's content region.

Definition at line 564 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowGetMinimumSize** (SysCarbonEvent &ioEvent, WindowRef inWindow, Point &outMinimumSize)=0

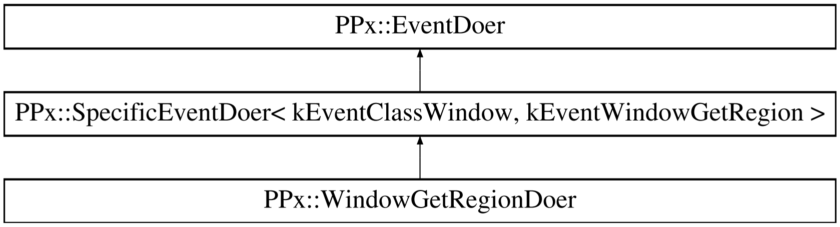
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.314 PPx::WindowGetRegionDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowGetRegionDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowGetRegion >
PPx::WindowGetRegionDoer

6.314.1 Detailed Description

Returns a specified region of a window.

Definition at line 52 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowGetRegion** (SysCarbonEvent &ioEvent, WindowRef inWindow, WindowRegionCode inRegionCode, RgnHandle ioRegion)=0

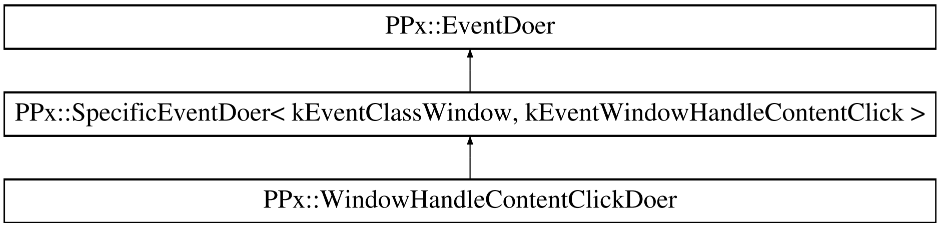
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.315 PPx::WindowHandleContentClickDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowHandleContentClickDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowHandleContentClick >
PPx::WindowHandleContentClickDoer

6.315.1 Detailed Description

Handles a click in a window.

The click is not a contextual menu click and is not within any subview.

Definition at line 615 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowHandleContentClick** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

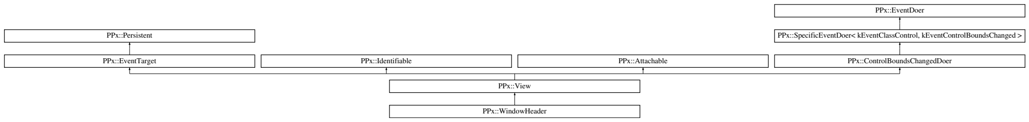
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.316 PPx::WindowHeader Class Reference

`#include <PPxWindowHeader.h>`

Inheritance diagram for PPx::WindowHeader::



6.316.1 Detailed Description

A system window header view.

Definition at line 22 of file PPxWindowHeader.h.

Public Member Functions

- WindowHeader ()

  Default constructor.

- virtual ∼WindowHeader ()

  Destructor.

- void Initialize (View ∗inSuperView, const HIRect &inFrame, bool inVisible, bool inEnabled, bool inIsListHeader)

  Initialize from window header creation parameters.

Protected Member Functions

- virtual void InitState (const DataReader &inReader)

  Initializes state from a data dictionary.

- virtual void WriteState (DataWriter &ioWriter) const

  Writes state to a data dictionary.

6.316.2 Member Function Documentation

6.316.2.1 void PPx::WindowHeader::Initialize (View ∗ *inSuperView*, const HIRect & *inFrame*, bool *inVisible*, bool *inEnabled*, bool *inIsListHeader*)

Initialize from window header creation parameters.

Parameters:

inSuperView Parent view

inFrame Bounds for view, in local coords of parent

inVisible Whether the view is visible

inEnabled Whether the view is enabled

inIsListHeader Whether header is for a list view

Definition at line 53 of file PPxWindowHeader.cp.

6.316.2.2 void PPx::WindowHeader::InitState (const DataReader & *inReader*) `[protected, virtual]`

Initializes state from a data dictionary.

Parameters:

inReader Data dictionary from which to read persistent data

Reimplemented from PPx::View.

Definition at line 90 of file PPxWindowHeader.cp.

References PPx::DataReader::ReadOptional().

6.316.2.3 void PPx::WindowHeader::WriteState (DataWriter & *ioWriter*) const `[protected, virtual]`

Writes state to a data dictionary.

Parameters:

ioWriter Data dictionary to which to write persistent data

Reimplemented from PPx::View.

Definition at line 110 of file PPxWindowHeader.cp.

References PPx::DataWriter::WriteValue().

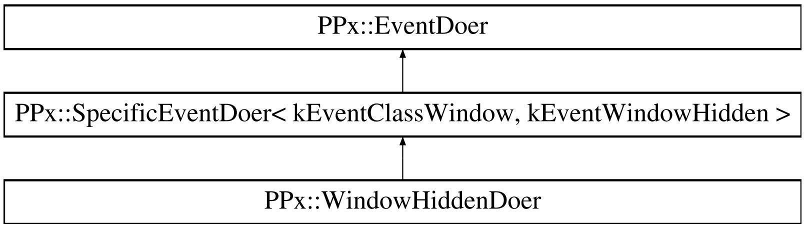
The documentation for this class was generated from the following files:

- PPxWindowHeader.h
- PPxWindowHeader.cp

6.317 PPx::WindowHiddenDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowHiddenDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowHidden >
PPx::WindowHiddenDoer

6.317.1 Detailed Description

Handles a window after being hidden.

Definition at line 153 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowHidden** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

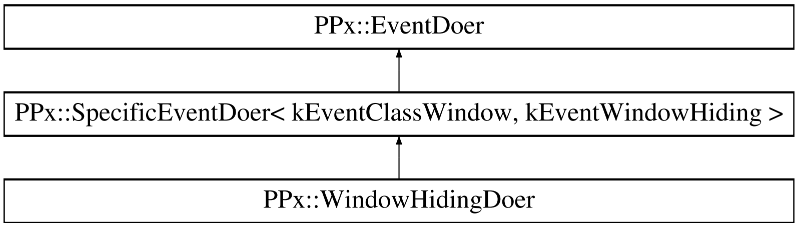
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.318 PPx::WindowHidingDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowHidingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowHiding >
PPx::WindowHidingDoer

6.318.1 Detailed Description

Handles a window being hidden.

Definition at line 121 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowHiding** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

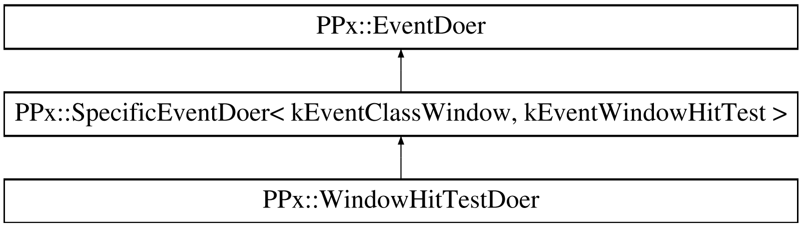
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.319 PPx::WindowHitTestDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowHitTestDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowHitTest >
PPx::WindowHitTestDoer

6.319.1 Detailed Description

Returns the window part hit by a specified mouse location.

Definition at line 70 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowHitTest** (SysCarbonEvent &ioEvent, WindowRef inWindow, const Point &inGlobalPoint, WindowDefPartCode &outPartHit)=0

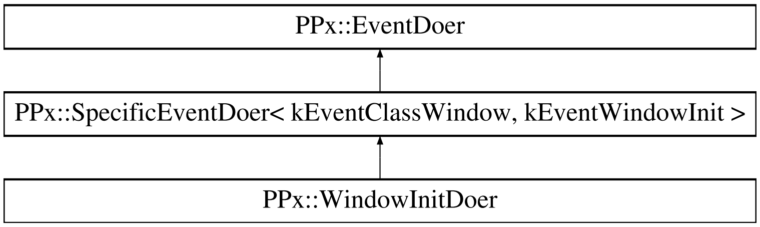
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.320 PPx::WindowInitDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowInitDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowInit >
PPx::WindowInitDoer

6.320.1 Detailed Description

Initializes a window definition.

Definition at line 88 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowInit** (SysCarbonEvent &ioEvent, WindowRef inWindow, UInt32 &outFeatures)=0

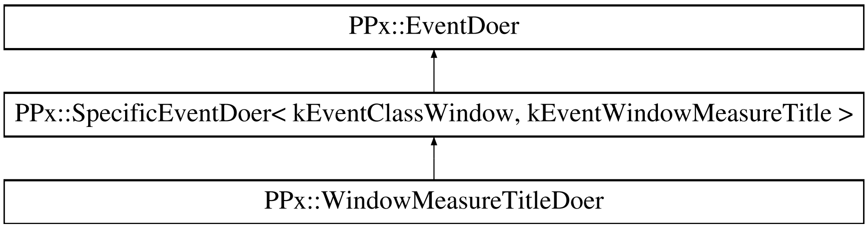
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.321 PPx::WindowMeasureTitleDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowMeasureTitleDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowMeasureTitle >
PPx::WindowMeasureTitleDoer

6.321.1 Detailed Description

Returns the width of a window’s title area.

Definition at line 191 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowMeasureTitle** (SysCarbonEvent &ioEvent, WindowRef inWindow, SInt16 &outFullWidth, SInt16 &outTextWidth)=0

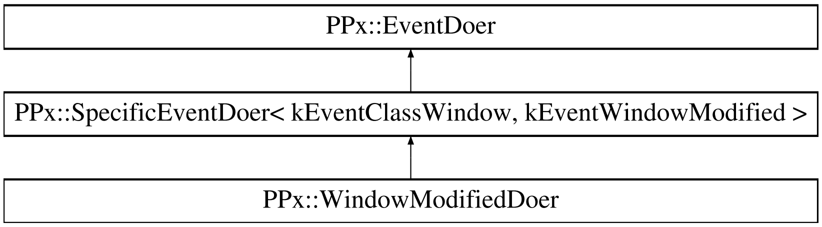
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.322 PPx::WindowModifiedDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowModifiedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowModified >
PPx::WindowModifiedDoer

6.322.1 Detailed Description

Handles change in modified state of a window.

Definition at line 138 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowModified** (SysCarbonEvent &ioEvent, WindowRef inWindow, bool inIsModified)=0

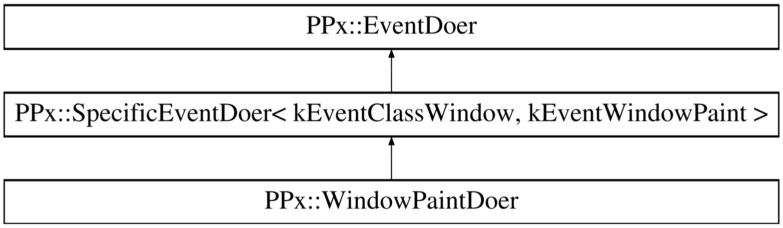
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.323 PPx::WindowPaintDoer Class Reference

```
#include <PPxWindowDefEvents.h>
```

Inheritance diagram for PPx::WindowPaintDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowPaint >
PPx::WindowPaintDoer

6.323.1 Detailed Description

Paints a window.

Definition at line 243 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowPaint** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

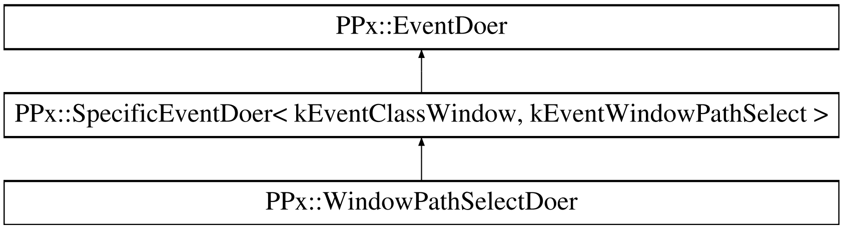
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.324 PPx::WindowPathSelectDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowPathSelectDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowPathSelect >
PPx::WindowPathSelectDoer

6.324.1 Detailed Description

Handles a request to select from the window path popup menu.

Definition at line 531 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowPathSelect** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.325 PPx::WindowRegionCodeStruct Struct Reference

```
#include <PPxSysTypes.h>
```

6.325.1 Detailed Description

Wrapper for WindowRegionCode.

Definition at line 148 of file PPxSysTypes.h.

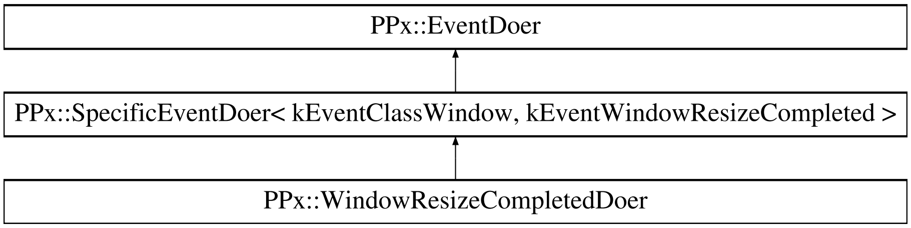
The documentation for this struct was generated from the following file:

- PPxSysTypes.h

6.326 PPx::WindowResizeCompletedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowResizeCompletedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowResizeCompleted >
PPx::WindowResizeCompletedDoer

6.326.1 Detailed Description

Handles a window finishing being resized.

Definition at line 401 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowResizeCompleted** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

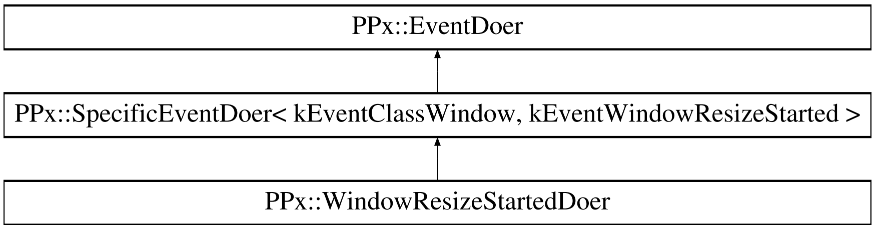
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.327 PPx::WindowResizeStartedDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowResizeStartedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowResizeStarted >
PPx::WindowResizeStartedDoer

6.327.1 Detailed Description

Handles a window starting to be resized.

Definition at line 385 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowResizeStarted** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

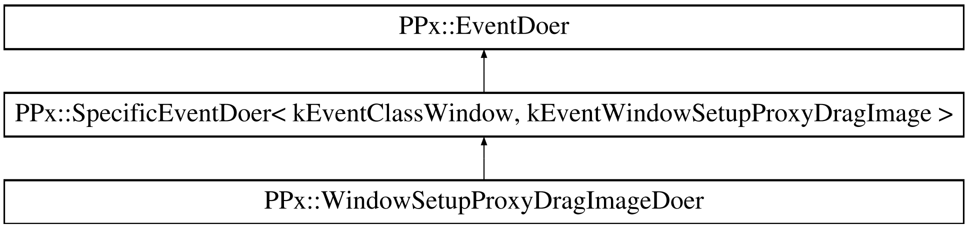
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.328 PPx::WindowSetupProxyDragImageDoer Class Reference

`#include <PPxWindowDefEvents.h>`

Inheritance diagram for PPx::WindowSetupProxyDragImageDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowSetupProxyDragImage >
PPx::WindowSetupProxyDragImageDoer

6.328.1 Detailed Description

Handles creating a drag image for a window's proxy icon.

Definition at line 155 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowSetupProxyDragImage** (SysCarbonEvent &ioEvent, WindowRef inWindow, RgnHandle ioImageClipRgn, RgnHandle ioDragOutline, GWorldPtr &outImage)=0

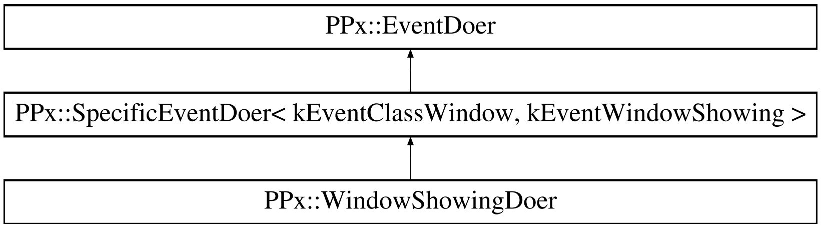
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.329 PPx::WindowShowingDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowShowingDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowShowing >
PPx::WindowShowingDoer

6.329.1 Detailed Description

Handles a window being shown.

Definition at line 105 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowShowing** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

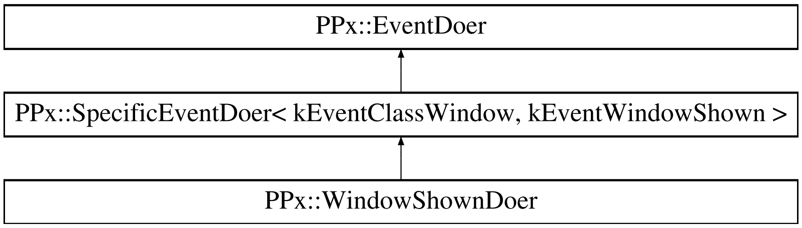
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.330 PPx::WindowShownDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowShownDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowShown >
PPx::WindowShownDoer

6.330.1 Detailed Description

Handles a window after being shown.

Definition at line 137 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowShown** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

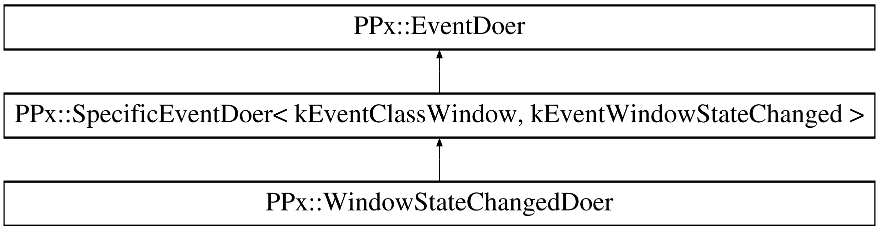
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.331 PPx::WindowStateChangedDoer Class Reference

`#include <PPxWindowDefEvents.h>`

Inheritance diagram for PPx::WindowStateChangedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowStateChanged >
PPx::WindowStateChangedDoer

6.331.1 Detailed Description

Handles change in window state.

Definition at line 174 of file PPxWindowDefEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowStateChanged** (SysCarbonEvent &ioEvent, WindowRef inWindow, UInt32 inStateFlags)=0

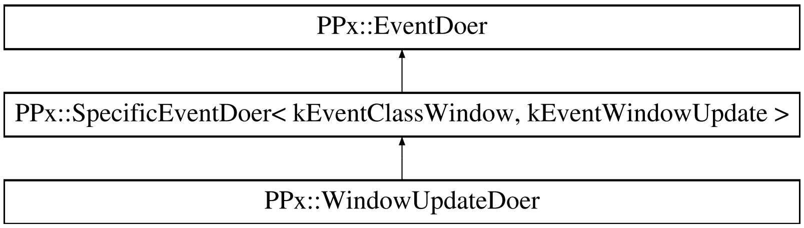
The documentation for this class was generated from the following files:

- PPxWindowDefEvents.h
- PPxWindowDefEvents.cp

6.332 PPx::WindowUpdateDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowUpdateDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowUpdate >
PPx::WindowUpdateDoer

6.332.1 Detailed Description

Handles a low-level window update event.

Definition at line 20 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowUpdate** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

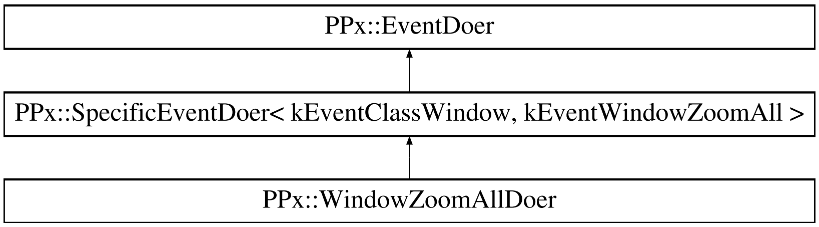
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.333 PPx::WindowZoomAllDoer Class Reference

`#include <PPxWindowEvents.h>`

Inheritance diagram for PPx::WindowZoomAllDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowZoomAll >
PPx::WindowZoomAllDoer

6.333.1 Detailed Description

Handles a request to zoom all windows.

Definition at line 313 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowZoomAll** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

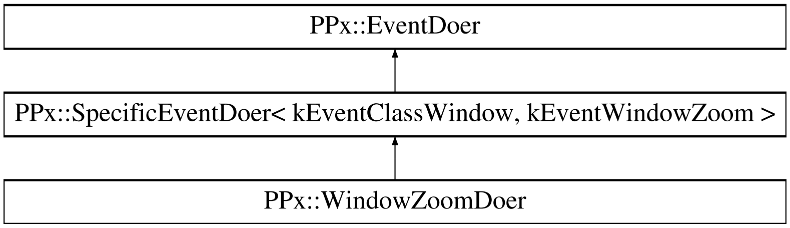
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.334 PPx::WindowZoomDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowZoomDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowZoom >
PPx::WindowZoomDoer

6.334.1 Detailed Description

Handles a request to zoom a window.

Definition at line 297 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowZoom** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

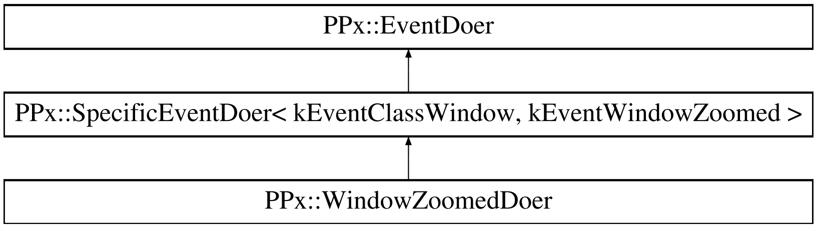
The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

6.335 PPx::WindowZoomedDoer Class Reference

```
#include <PPxWindowEvents.h>
```

Inheritance diagram for PPx::WindowZoomedDoer::


PPx::EventDoer
PPx::SpecificEventDoer< kEventClassWindow, kEventWindowZoomed >
PPx::WindowZoomedDoer

6.335.1 Detailed Description

Handles a window after being zoomed.

Definition at line 329 of file PPxWindowEvents.h.

Protected Member Functions

- virtual OSStatus **DoWindowZoomed** (SysCarbonEvent &ioEvent, WindowRef inWindow)=0

The documentation for this class was generated from the following files:

- PPxWindowEvents.h
- PPxWindowEvents.cp

Chapter 7

PowerPlant X 1.0 API Reference File Documentation

7.1 PPxAccessibilityEvents.h File Reference

7.1.1 Detailed Description

Event handlers for accessibility Carbon Events.

Definition in file PPxAccessibilityEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.2 PPxAEStandardEvents.h File Reference

7.2.1 Detailed Description

Handlers for events in the Apple Event Standard Suite.

Definition in file PPxAEStandardEvents.h.

```
#include <PPxAppleEventDoer.h>
```

Namespaces

- namespace PPx

7.3 PPxAppleEventDoer.h File Reference

7.3.1 Detailed Description

Classes for handling Apple Events.

Definition in file PPxAppleEventDoer.h.

```
#include <SysAEHandler.h>
```

```
#include <SysAppleEvent.h>
```

```
#include <SysAEDesc.h>
```

Namespaces

- namespace PPx

7.4 PPxApplication.h File Reference

7.4.1 Detailed Description

Class for an executable program.

Definition in file PPxApplication.h.

```
#include <PPxEventTarget.h>
#include <PPxAttachable.h>
```

Namespaces

- namespace PPx

7.5 PPxApplicationEvents.h File Reference

7.5.1 Detailed Description

Event handlers for application Carbon Events.

Definition in file PPxApplicationEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.6 PPxAttachable.h File Reference

7.6.1 Detailed Description

Class for objects which have an associated list of attachments.

Definition in file PPxAttachable.h.

```
#include <PPxPrefix.h>
#include <vector>
```

Namespaces

- namespace PPx

7.7 PPxAttachment.h File Reference

7.7.1 Detailed Description

Abstract class for identifiable persistent objects.

Definition in file PPxAttachment.h.

```
#include <PPxPersistent.h>
#include <PPxIdentifiable.h>
```

Namespaces

- namespace PPx

7.8 PPxBaseView.h File Reference

7.8.1 Detailed Description

Basic View subclass.

Definition in file PPxBaseView.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.9 PPxBevelButton.h File Reference

7.9.1 Detailed Description

A system bevel button control.

Definition in file PPxBevelButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.10 PPxBundleUtils.h File Reference

7.10.1 Detailed Description

Utility functions for working with Bundles.

Definition in file PPxBundleUtils.h.

```
#include <PPxPrefix.h>
#include <SysCFData.h>
#include <SysCFString.h>
#include <CFBundle.h>
```

Namespaces

- namespace PPx
- namespace PPx::BundleUtils

7.11 PPxChasingArrows.h File Reference

7.11.1 Detailed Description

A system chasing arrows activity indicator.

Definition in file PPxChasingArrows.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.12 PPxCheckBox.h File Reference

7.12.1 Detailed Description

A system check box control.

Definition in file PPxCheckBox.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.13 PPxCheckBoxGroupBox.h File Reference

7.13.1 Detailed Description

A system group box with a check box title.

Definition in file PPxCheckBoxGroupBox.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.14 PPxClockControl.h File Reference

7.14.1 Detailed Description

A system clock control.

Definition in file PPxClockControl.h.

```
#include <PPxView.h>
#include <DateTimeUtils.h>
```

Namespaces

- namespace PPx

7.15 PPxComboBox.h File Reference

7.15.1 Detailed Description

A system combo box control.

Definition in file PPxComboBox.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.16 PPxCommandEvents.h File Reference

7.16.1 Detailed Description

Event handlers for command Carbon Events.

Definition in file PPxCommandEvents.h.

```
#include <PPxEventDoer.h>
```

```
#include <SysEventParam.h>
```

Namespaces

- namespace PPx

7.17 PPxCommandTask.h File Reference

7.17.1 Detailed Description

Attachment classes for handling commands.

Definition in file PPxCommandTask.h.

```
#include <PPxAttachment.h>
#include <PPxEventTarget.h>
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.18 PPxConstants.h File Reference

7.18.1 Detailed Description

Declarations of commonly used constants.

Definition in file PPxConstants.h.

```
#include <PPxTypes.h>
#include <CFString.h>
```

Namespaces

- namespace PPx

7.19 PPxCorrespondent.h File Reference

7.19.1 Detailed Description

Definition in file PPxCorrespondent.h.

```
#include <PPxEventTarget.h>
#include <PPxAttachable.h>
#include <SysHIObject.h>
```

Namespaces

- namespace PPx

7.20 PPxCreateView.h File Reference

7.20.1 Detailed Description

Template functions for creating PPx views.

The template functions in this file create and initialize an object of a View subclass. There are nine versions of the function, taking from zero to eight parameters beyond the four standard parameters.

Via the magic of template parameter type deduction, these functions suffice to create any view. For example,

```
    PPx:ChasingArrows*  chasers = PPx::CreateView<PPx::ChasingArrows>(
                                        PPx::superView_None,
                                        frame,
                                        PPx::visible_Yes,
                                        PPx::enabled_Yes);

    PPx::CheckBox*      checker = PPx::CreateView<PPx::CheckBox>(
                                        PPx::superView_None,
                                        frame,
                                        PPx::visible_Yes,
                                        PPx::enabled_No,
                                        CFSTR("Check Box Title");
                                        PPx::value_Off,
                                        true);      // auto toggle
```

Definition in file PPxCreateView.h.

`#include <PPxMemoryUtils.h>`

`#include <PPxView.h>`

`#include <memory>`

Namespaces

- namespace PPx

7.21 PPxDataFork.h File Reference

7.21.1 Detailed Description

Class for accessing the contents of a file's data fork.

Definition in file PPxDataFork.h.

```
#include <PPxFileFork.h>
#include <SysCFData.h>
```

Namespaces

- namespace PPx

7.22 PPxDataObject.h File Reference

7.22.1 Detailed Description

Classes for storing data values of a particular type.

Definition in file PPxDataObject.h.

```
#include <PPxRetained.h>
#include <SysCFString.h>
```

Namespaces

- namespace PPx

7.23 PPxDataScrap.h File Reference

7.23.1 Detailed Description

Classes for managing scraps which store and retrieve data.

Definition in file PPxDataScrap.h.

```
#include <SysCFString.h>
#include <Scrap.h>
```

Namespaces

- namespace PPx
- namespace PPx::Clipboard
- namespace PPx::FindScrap

7.24 PPxDebugging.h File Reference

7.24.1 Detailed Description

Debugging Utilities.

Definition in file PPxDebugging.h.

```
#include <PPxOptions.h>
#include <PPxTypes.h>
```

Namespaces

- namespace PPx
- namespace PPx::Debugging
- namespace PPx::MenuDebugStr

Defines

- #define PPx_ExceptLoc_Here PPx::sourceLocation_Nothing

  Location within source code.

- #define PPx_SignalLoc_Here PPx::sourceLocation_Nothing

  Location within source code.

- #define PPx_SetDebugThrow_Nothing_()

  Sets option to do nothing extra when throwing an exception.

- #define PPx_SetDebugThrow_Alert_()

  Sets option to display a modal alert when throwing an exception.

- #define PPx_SetDebugThrow_Debugger_()

  Sets option to break into the debugger when throwing an exception.

- #define PPx_SetDebugThrow_Console_()

  Sets option to write to the console when throwing an exception.

- #define **PPx_SignalString_**(str)
- #define **PPx_SignalIf_**(test)
- #define **PPx_SignalIfNot_**(test)
- #define PPx_SetDebugSignal_Nothing_()

  Sets option to do nothing when raising a PPx signal.

- #define PPx_SetDebugSignal_Alert_()

    Sets option to display a modal alert when raising a PPx signal.

- #define PPx_SetDebugSignal_Debugger_()

    Sets option to break into the debugger when raising a PPx signal.

- #define PPx_SetDebugSignal_Console_()

    Sets option to write to the console when raising a PPx signal.

7.24.2 Define Documentation

7.24.2.1 #define PPx_ExceptLoc_Here PPx::sourceLocation_Nothing

Location within source code.

Set to a nil location when debugging exceptions is off.

Definition at line 55 of file PPxDebugging.h.

7.24.2.2 #define PPx_SetDebugSignal_Alert_()

Value:

```
PPx::Debugging::SetDebugSignalAction(                              \
                                PPx::Debugging::debugAction_Alert)
```

Sets option to display a modal alert when raising a PPx signal.

Definition at line 252 of file PPxDebugging.h.

7.24.2.3 #define PPx_SetDebugSignal_Console_()

Value:

```
PPx::Debugging::SetDebugSignalAction(                              \
                                PPx::Debugging::debugAction_Console)
```

Sets option to write to the console when raising a PPx signal.

Definition at line 272 of file PPxDebugging.h.

7.24.2.4 #define PPx_SetDebugSignal_Debugger_()

Value:

```
PPx::Debugging::SetDebugSignalAction(                            \
                                PPx::Debugging::debugAction_Debugger)
```

Sets option to break into the debugger when raising a PPx signal.

Definition at line 262 of file PPxDebugging.h.

7.24.2.5 #define PPx_SetDebugSignal_Nothing_()

Value:

```
PPx::Debugging::SetDebugSignalAction(                            \
                                PPx::Debugging::debugAction_Nothing)
```

Sets option to do nothing when raising a PPx signal.

Definition at line 242 of file PPxDebugging.h.

7.24.2.6 #define PPx_SetDebugThrow_Alert_()

Value:

```
PPx::Debugging::SetDebugThrowAction(                             \
                                PPx::Debugging::debugAction_Alert)
```

Sets option to display a modal alert when throwing an exception.

Definition at line 141 of file PPxDebugging.h.

7.24.2.7 #define PPx_SetDebugThrow_Console_()

Value:

```
PPx::Debugging::SetDebugThrowAction(                             \
                                PPx::Debugging::debugAction_Console)
```

Sets option to write to the console when throwing an exception.

Definition at line 161 of file PPxDebugging.h.

7.24.2.8 #define PPx_SetDebugThrow_Debugger_()

Value:

```
PPx::Debugging::SetDebugThrowAction(                              \
                                   PPx::Debugging::debugAction_Debugger)
```

Sets option to break into the debugger when throwing an exception.

Definition at line 151 of file PPxDebugging.h.

7.24.2.9 #define PPx_SetDebugThrow_Nothing_()

Value:

```
PPx::Debugging::SetDebugThrowAction(                              \
                                   PPx::Debugging::debugAction_Nothing)
```

Sets option to do nothing extra when throwing an exception.

Definition at line 131 of file PPxDebugging.h.

7.24.2.10 #define PPx_SignalLoc_Here PPx::sourceLocation_Nothing

Location within source code.

Set to a nil location when debugging signals is off.

Definition at line 63 of file PPxDebugging.h.

7.25 PPxDisclosureButton.h File Reference

7.25.1 Detailed Description

A system disclosure button control.

Definition in file PPxDisclosureButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.26 PPxDisclosureTriangle.h File Reference

7.26.1 Detailed Description

A system disclosure triangle control.

Definition in file PPxDisclosureTriangle.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.27 PPxDrawerWindow.h File Reference

7.27.1 Detailed Description

A drawer which slides out from an edge of a parent window.

Definition in file PPxDrawerWindow.h.

```
#include <PPxWindow.h>
```

Namespaces

- namespace PPx

7.28 PPxEditTextControl.h File Reference

7.28.1 Detailed Description

A system edit text control.

Definition in file PPxEditTextControl.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.29 PPxEditUnicodeText.h File Reference

7.29.1 Detailed Description

A system edit unicode text control.

Definition in file PPxEditUnicodeText.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.30 PPxEventAttachments.h File Reference

7.30.1 Detailed Description

Attachment classes for handling Carbon Events.

Definition in file PPxEventAttachments.h.

```
#include <PPxAttachment.h>
#include <PPxEventDoer.h>
#include <PPxEventTarget.h>
#include <PPxHIObjectEvents.h>
```

Namespaces

- namespace PPx

7.31 PPxEventDoer.h File Reference

7.31.1 Detailed Description

Definition in file PPxEventDoer.h.

```
#include <SysCarbonEvent.h>
#include <SysEventHandler.h>
```

Namespaces

- namespace PPx

7.32 PPxEventTarget.h File Reference

7.32.1 Detailed Description

Definition in file PPxEventTarget.h.

```
#include <PPxPersistent.h>
#include <CarbonEvents.h>
```

Namespaces

- namespace PPx

7.33 PPxEventUtils.h File Reference

7.33.1 Detailed Description

Utility functions for working with CarbonEvents.

Definition in file PPxEventUtils.h.

```
#include <PPxPrefix.h>
```

```
#include <CarbonEvents.h>
```

Namespaces

- namespace PPx
- namespace PPx::EventUtils

7.34 PPxExceptions.h File Reference

7.34.1 Detailed Description

Exception classes.

Definition in file PPxExceptions.h.

`#include <PPxDebugging.h>`

Namespaces

- namespace PPx

Defines

- #define PPx_Throw_(ExceptionClass, inWhat, inWhy) PPx::ThrowException< ExceptionClass >(inWhat, inWhy, PPx_ExceptLoc_Here)

  Throws an exception.

- #define PPx_ThrowIf_(test, ExceptionClass, inWhat, inWhy)

  Throws an exception if a boolean test condition is true.

- #define PPx_ThrowIfNil_(inValue, ExceptionClass, inWhat, inWhy)

  Throws an exception if a value is nil.

- #define PPx_ThrowIfOSError_(inErrorCode, inWhy) PPx::ThrowIfOSError(inErrorCode, inWhy, PPx_ExceptLoc_Here)

  Throws a PPx::OSError exception if the error code is not noErr.

- #define PPx_ThrowOSError_(inErrorCode, inWhy) PPx::ThrowOSError(inErrorCode, inWhy, PPx_ExceptLoc_Here)

  Throws a PPx::OSError exception with the specified error code.

- #define PPx_ThrowOSErrorCode_(inErrorCode, inWhy) PPx::ThrowOSErrorCode< inErrorCode >(inWhy, PPx_ExceptLoc_Here)

  Throws a PPx::OSErrorCode exception.

- #define **PPx_BadParamIf_**(test)
- #define PPx_BadParamIfNil_(inPtr) PPx_BadParamIf_(inPtr == nil)

  Throws a err_BadParam exception if the parameter is nil.

7.34.2 Define Documentation

7.34.2.1 #define PPx_Throw_(ExceptionClass, inWhat, inWhy) PPx::ThrowException< ExceptionClass >(inWhat, inWhy, PPx_ExceptLoc_Here)

Throws an exception.

Parameters:

ExceptionClass Name of the exception class

inWhat Exception ID

inWhy A string describing the cause of the exception

Definition at line 328 of file PPxExceptions.h.

Referenced by PPx::CFXMLElement::CFXMLElement(), PPx::FSObject::CheckLock(), PPx::Registrar::CreateObject(), PPx::Folder::CreateOnDisk(), PPx::File::CreateOnDisk(), PPx::SysHIObject::CreateSysObject(), PPx::XMLEncoder::Find(), PPx::XMLDecoder::Find(), PPx::FSObject::GetFSSpec(), PPx::BundleUtils::GetInfoDictionaryKeyString(), PPx::FSObject::GetName(), PPx::FSObject::GetParent(), PPx::SysWindow::MakeWindow(), PPx::FileFork::Open(), PPx::DataFork::ReadContents(), PPx::DataReader::ReadRequired(), PPx::FSObject::Rename(), PPx::FSUtils::StringToHFSUniStr(), PPx::CFUtils::VerifyIndex(), PPx::CFUtils::VerifyInsertIndex(), and PPx::CFUtils::VerifyRange().

7.34.2.2 #define PPx_ThrowIf_(test, ExceptionClass, inWhat, inWhy)

Value:

```
if (test)                                              \
                PPx_Throw_(ExceptionClass, inWhat, inWhy)
```

Throws an exception if a boolean test condition is true.

Parameters:

test C++ code of boolean test condition

ExceptionClass Name of the exception class

inWhat Exception ID

inWhy A string describing the cause of the exception

Definition at line 342 of file PPxExceptions.h.

Referenced by PPx::FSObject::UseRef().

7.34.2.3 #define PPx_ThrowIfNil_(inValue, ExceptionClass, inWhat, inWhy)

Value:

```
if ((inValue) == nil)                                   \
                PPx_Throw_(ExceptionClass, inWhat, inWhy)
```

Throws an exception if a value is nil.

Parameters:

inValue A pointer type value

ExceptionClass Name of the exception class

inWhat Exception ID

inWhy A string describing the cause of the exception

Definition at line 357 of file PPxExceptions.h.

Referenced by PPx::DataReader::ReadContainer(), PPx::DataReader::ReadObjectContainer(), PPx::DataReader::ReadOptional(), PPx::SafeDynamicCast(), and PPx::CFObject< TCFRef >::UseRef().

7.34.2.4 #define PPx_ThrowIfOSError_(inErrorCode, inWhy) PPx::ThrowIfOSError(inErrorCode, inWhy, PPx_ExceptLoc_Here)

Throws a PPx::OSError exception if the error code is not noErr.

Parameters:

inErrorCode A Mac OS error code

inWhy A string describing the cause of the exception

Definition at line 370 of file PPxExceptions.h.

Referenced by PPx::SysHIView::AddSubView(), PPx::ComboBox::AppendListItem(), PPx::NavServices::AskChooseFile(), PPx::NavServices::AskDesignateFile(), PPx::NavServices::AskDiscardChanges(), PPx::NavServices::AskGetFile(), PPx::NavServices::AskReviewDocuments(), PPx::NavServices::AskSaveChanges(), PPx::AutoNavReply::AutoNavReply(), PPx::SysCreateView::BevelButton(), PPx::ComboBox::ChangeAttributes(), PPx::FSObject::ChangeFinderFlags(), PPx::SysCreateView::ChasingArrows(), PPx::SysCreateView::CheckBox(), PPx::SysCreateView::CheckBoxGroupBox(), PPx::SysCreateView::ClockControl(), PPx::DrawerWindow::CloseDrawer(), PPx::SysCreateView::ComboBox(), PPx::FSUtils::CompareFSNames(), PPx::SysHIView::CreateOffscreenImage(), PPx::Folder::CreateOnDisk(), PPx::File::CreateOnDisk(), PPx::SysHIObject::CreateSysObject(), PPx::SysHIView::CreateSysView(), PPx::FSObject::Delete(), PPx::FSObject::DeleteContainer(), PPx::FSObject::DeleteContainerContents(), PPx::SysCreateView::DisclosureButton(), PPx::SysCreateView::DisclosureTriangle(), PPx::SysCreateView::EditTextControl(), PPx::SysCreateView::EditUnicodeText(), PPx::FSUtils::FSNamesAreEqual(), PPx::SysEventParam::Get(), PPx::AutoAEDesc::GetAttributeDesc(), PPx::ComboBox::GetAttributes(), PPx::FSObject::GetCatalogInfo(), PPx::SysHIView::GetCommandID(), PPx::AutoAEDesc::GetCount(), PPx::SysScrap::GetData(), PPx::SysScrap::GetDataSize(), PPx::NavServices::GetDefaultCreationOptions(), PPx::DrawerWindow::GetDrawerOffsets(), PPx::CFUtils::GetEncodingFromScriptCode(), PPx::SysAppleEvent::GetEventClass(), PPx::SysAppleEvent::GetEventKind(), PPx::FSObject::GetFinderInfo(), PPx::FileFork::GetForkInfo(), PPx::ResourceFork::GetForkName(), PPx::FileFork::GetForkName(), PPx::DataFork::GetForkName(), PPx::SysHIView::GetFrame(), PPx::FileFork::GetFSRef(), PPx::FSObject::GetFSSpec(), PPx::ComboBox::GetListItemText(), PPx::FSObject::GetName(), PPx::SysScrap::GetNamedScrap(), PPx::AutoAEDesc::GetNthDesc(), PPx::FSObject::GetParent(), PPx::FSObject::GetParentDirID(), PPx::SheetWindow::GetParentWindow(), PPx::FileFork::GetPosition(), PPx::AutoAEDesc::GetRequiredParamDesc(), PPx::BundleUtils::GetResourceData(), PPx::BundleUtils::GetResourceProperty(), PPx::FileFork::GetSize(), PPx::StaticText::GetText(), PPx::EditUnicodeText::GetText(), PPx::EditTextControl::GetText(), PPx::ComboBox::GetText(), PPx::SysHIView::GetTitle(), PPx::File::GetTotalForkSizes(), PPx::FSObject::GetVolume(), PPx::SysWindow::GetWindowAttributes(), PPx::SysWindow::GetWindowClass(), PPx::SysCreateView::IconControl(), PPx::SysCreateView::IconPushButton(), PPx::SysCreateView::ImageView(), PPx::SysCreateView::ImageWell(), PPx::WindowContentView::Initialize(), PPx::SheetAlert::Initialize(), PPx::ComboBox::InsertListItemAt(), PPx::SysEventHandler::Install(), PPx::SysAEHandler::Install(), PPx::SysCreateView::ListBox(), PPx::SysCreateView::LittleArrows(), PPx::SysCarbonEvent::MakeEvent(), PPx::SysWindow::MakeWindow(), PPx::SysHIView::MoveFrameBy(), PPx::FileFork::Open(), PPx::DrawerWindow::OpenDrawer(), PPx::SysAppleEvent::operator=(), PPx::SysCreateView::PictureControl(), PPx::SysCreateView::Placard(), PPx::SysHIView::PlaceFrameAt(), PPx::SysCreateView::PopupArrow(), PPx::SysCreateView::PopupButton(), PPx::SysCreateView::PopupGroupBox(), PPx::SysCarbonEvent::PostTo(), PPx::SysCreateView::ProgressBar(), PPx::SysScrap::PromiseData(), PPx::SysCreateView::PushButton(), PPx::SysCreateView::RadioButton(), PPx::SysCreateView::RadioGroup(), PPx::DataFork::ReadData(), PPx::SysHIObject::RegisterSysClass(), PPx::SysHIView::RegisterSysViewClass(), PPx::SysCreateView::RelevanceBar(), PPx::SysAEHandler::Remove(), PPx::SysHIView::RemoveFromSuperView(), PPx::ComboBox::RemoveListItem(), PPx::FSObject::Rename(), PPx::SysCreateView::RoundButton(), PPx::SysCreateView::ScrollBar(), PPx::SysCreateView::ScrollView(), PPx::SysAppleEvent::Send(), PPx::SysCreateView::SeparatorLine(), PPx::SysEventParam::Set(), PPx::ImageView::SetAlpha(), PPx::ScrollView::SetAutoHideScrollBars(), PPx::FSObject::SetCatalogInfo(), PPx::SysHIView::SetCommandID(), PPx::SysScrap::SetData(), PPx::SysHIView::SetDataTag(), PPx::DrawerWindow::SetDrawerOffsets(), PPx::FSObject::SetFinderInfo(), PPx::SysHIView::SetFrame(), PPx::ImageView::SetImage(), PPx::Idle-

Timer::SetNextFireTime(), PPx::Timer::SetNextFireTime(), PPx::ImageView::SetOpaque(), PPx::SysAppleEvent::SetParamDesc(), PPx::SysCarbonEvent::SetParameter(), PPx::SysAppleEvent::SetParameter(), PPx::DrawerWindow::SetParentWindow(), PPx::FileFork::SetPosition(), PPx::DrawerWindow::SetPreferredEdge(), PPx::SysScrap::SetPromiseKeeper(), PPx::SysWindow::SetProperty(), PPx::SysHIView::SetProperty(), PPx::ImageView::SetScaleToFit(), PPx::FileFork::SetSize(), PPx::SysHIView::SetTitle(), PPx::SysHIView::SetVisible(), PPx::SheetAlert::Show(), PPx::SysCreateView::Slider(), PPx::SysCreateView::StaticText(), PPx::SysAppleEvent::SysAppleEvent(), PPx::SysCreateView::TabView(), PPx::SysCreateView::TextGroupBox(), PPx::Folder::UpdateLocation(), PPx::SysCreateView::WindowHeader(), and PPx::DataFork::WriteData().

7.34.2.5 #define PPx_ThrowOSError_(inErrorCode, inWhy) PPx::ThrowOSError(inErrorCode, inWhy, PPx_ExceptLoc_Here)

Throws a PPx::OSError exception with the specified error code.

Parameters:

inErrorCode A Mac OS error code

inWhy A string describing the cause of the exception

Definition at line 382 of file PPxExceptions.h.

Referenced by PPx::NavServices::AskDesignateFile(), PPx::NavServices::AskDiscardChanges(), PPx::NavServices::AskGetFile(), PPx::NavServices::AskReviewDocuments(), and PPx::NavServices::AskSaveChanges().

7.34.2.6 #define PPx_ThrowOSErrorCode_(inErrorCode, inWhy) PPx::ThrowOSErrorCode< inErrorCode >(inWhy, PPx_ExceptLoc_Here)

Throws a PPx::OSErrorCode exception.

Parameters:

inErrorCode A literal Mac OS error code. Must be a constant value and not a variable.

inWhy A string describing the cause of the exception

Definition at line 395 of file PPxExceptions.h.

Referenced by PPx::BundleUtils::GetResourceData().

7.35 PPxFile.h File Reference

7.35.1 Detailed Description

Class for a file on disk.

Definition in file PPxFile.h.

```
#include <PPxDataFork.h>
#include <PPxFSObject.h>
#include <PPxResourceFork.h>
```

Namespaces

- namespace PPx

7.36 PPxFileFork.h File Reference

7.36.1 Detailed Description

Class for accessing a fork of a file.

Definition in file PPxFileFork.h.

```
#include <PPxPrefix.h>
```

```
#include <Files.h>
```

Namespaces

- namespace PPx

7.37 PPxFolder.h File Reference

7.37.1 Detailed Description

Definition in file PPxFolder.h.

```
#include <PPxFSObject.h>
```

Namespaces

- namespace PPx

7.38 PPxFrameAdapter.h File Reference

7.38.1 Detailed Description

Classes for adjucting the frame of a view.

Definition in file PPxFrameAdapter.h.

```
#include <PPxPersistent.h>
```

Namespaces

- namespace PPx

7.39 PPxFSObject.h File Reference

7.39.1 Detailed Description

Wrapper for FSRef and related File Manager and MoreFiles X functions.

Definition in file PPxFSObject.h.

```
#include <PPxPrefix.h>
#include <SysCFString.h>
#include "MoreFilesX.h"
```

Namespaces

- namespace PPx

7.40 PPxFSUtils.h File Reference

7.40.1 Detailed Description

Definition in file PPxFSUtils.h.

```
#include <PPxPrefix.h>
```

```
#include <Files.h>
```

Namespaces

- namespace PPx
- namespace PPx::FSUtils

Functions

- bool operator== (const FSRef &inRight, const FSRef &inLeft)

  Compares two FSRefs for equality.

- bool operator!= (const FSRef &inRight, const FSRef &inLeft)

  Compares two FSRefs for inequality.

7.40.2 Function Documentation

7.40.2.1 bool operator!= (const FSRef & *inRight*, const FSRef & *inLeft*)

Compares two FSRefs for inequality.

Parameters:

inRight Right hand side FSRef

inLeft Left hand side FSRef

Returns:

Whether the FSRefs are unequal

Definition at line 57 of file PPxFSUtils.cp.

7.40.2.2 bool operator== (const FSRef & *inRight*, const FSRef & *inLeft*)

Compares two FSRefs for equality.

Parameters:

inRight Right hand side FSRef

inLeft Left hand side FSRef

Returns:

Whether the FSRefs are equal

Definition at line 38 of file PPxFSUtils.cp.

7.41 PPxGrayBox.h File Reference

7.41.1 Detailed Description

View which draws a gray box.

Definition in file PPxGrayBox.h.

```
#include <PPxBaseView.h>
#include <PPxViewEvents.h>
```

Namespaces

- namespace PPx

7.42 PPxHIObjectEvents.h File Reference

7.42.1 Detailed Description

Event handlers for HIObject Carbon Events.

Definition in file PPxHIObjectEvents.h.

`#include <PPxEventDoer.h>`

Namespaces

- namespace PPx

7.43 PPxIconControl.h File Reference

7.43.1 Detailed Description

A system icon control.

Definition in file PPxIconControl.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.44 PPxIconPushButton.h File Reference

7.44.1 Detailed Description

A system push button with icon control.

Definition in file PPxIconPushButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.45 PPxIdentifiable.h File Reference

7.45.1 Detailed Description

Mix-in class for objects with an Object ID.

Definition in file PPxIdentifiable.h.

```
#include <PPxPrefix.h>
```

Namespaces

- namespace PPx

7.46 PPxImageView.h File Reference

7.46.1 Detailed Description

A system view which displays a core graphics image.

Definition in file PPxImageView.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.47 PPxImageWell.h File Reference

7.47.1 Detailed Description

A system image well view.

Definition in file PPxImageWell.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.48 PPxKeyboardEvents.h File Reference

7.48.1 Detailed Description

Event handlers for keyboard Carbon Events.

Definition in file PPxKeyboardEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.49 PPxListBox.h File Reference

7.49.1 Detailed Description

A system list box control.

Definition in file PPxListBox.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.50 PPxLittleArrows.h File Reference

7.50.1 Detailed Description

A system little arrows control.

Definition in file PPxLittleArrows.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.51 PPxMemoryUtils.h File Reference

7.51.1 Detailed Description

Function and classes for managing objects and data stored in memory.

Definition in file PPxMemoryUtils.h.

```
#include <PPxPrefix.h>
#include <memory>
```

Namespaces

- namespace PPx

7.52 PPxMenuEvents.h File Reference

7.52.1 Detailed Description

Event handlers for menu Carbon Events.

Definition in file PPxMenuEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.53 PPxMiscellaneousEvents.h File Reference

7.53.1 Detailed Description

Event handlers for Apple event, tablet, volume, and appearance Carbon Events.

Definition in file PPxMiscellaneousEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.54 PPxMLTEView.h File Reference

7.54.1 Detailed Description

Text editing view based on MLTE.

Definition in file PPxMLTEView.h.

`#include <PPxBaseView.h>`

`#include <MacTextEditor.h>`

Namespaces

- namespace PPx

7.55 PPxMouseEvents.h File Reference

7.55.1 Detailed Description

Event handlers for mouse Carbon Events.

Definition in file PPxMouseEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.56 PPxNavServices.h File Reference

7.56.1 Detailed Description

Classs and functions for using Navigation Servicers.

Definition in file PPxNavServices.h.

```
#include <PPxPrefix.h>
```

```
#include <Navigation.h>
```

Namespaces

- namespace PPx
- namespace PPx::NavServices

7.57 PPxOptions.h File Reference

7.57.1 Detailed Description

Conditional compilation options.

Definition in file PPxOptions.h.

Defines

- #define PPx_Debug 1

  Master Debugging Switch.

- #define PPx_Debug_Exceptions PPx_Debug

  Debugging Exceptions.

- #define PPx_Debug_Signals PPx_Debug

  Debugging Signals.

- #define PPx_Verify_Parameters PPx_Debug

  Verifying Parameters.

- #define DEBUG PPx_Debug

  Apple Debugging Utilities (see Debugging.h).

7.58 PPxOwnedPointer.h File Reference

7.58.1 Detailed Description

Template class for managing exclusive ownership of a pointer.

Definition in file PPxOwnedPointer.h.

Namespaces

- namespace PPx

7.59 PPxPersistent.h File Reference

7.59.1 Detailed Description

Abstract base class for persistent objects.

Definition in file PPxPersistent.h.

```
#include <PPxPrefix.h>
```

Namespaces

- namespace PPx

7.60 PPxPictureControl.h File Reference

7.60.1 Detailed Description

A system picture control.

Definition in file PPxPictureControl.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.61 PPxPlacard.h File Reference

7.61.1 Detailed Description

A system placard view.

Definition in file PPxPlacard.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.62 PPxPopupArrow.h File Reference

7.62.1 Detailed Description

A system popup arrow view.

Definition in file PPxPopupArrow.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.63 PPxPopupButton.h File Reference

7.63.1 Detailed Description

A system popup button control.

Definition in file PPxPopupButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.64 PPxPopupGroupBox.h File Reference

7.64.1 Detailed Description

A system group box with a popup menu title.

Definition in file PPxPopupGroupBox.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.65 PPxPrefix.h File Reference

7.65.1 Detailed Description

Top-level header file for PowerPlant X.

To ensure that PowerPlant X and compiler options are set properly, the first #include for every file should be PPxPrefix.h or some header file that #include's this file.

Definition in file PPxPrefix.h.

```
#include <PPxOptions.h>
```

```
#include <PPxConstants.h>
```

```
#include <PPxExceptions.h>
```

Defines

- #define PPx_Version 0x01008000

  PowerPlant X version number.

7.66 PPxPrimaryBundle.h File Reference

7.66.1 Detailed Description

Utility functions for working with the primary bundle for a program.

Definition in file PPxPrimaryBundle.h.

```
#include <PPxPrefix.h>
```

```
#include <SysCFBundle.h>
```

```
#include <SysCFString.h>
```

Namespaces

- namespace PPx
- namespace PPx::PrimaryBundle

7.67 PPxProgressBar.h File Reference

7.67.1 Detailed Description

A system progress bar control.

Definition in file PPxProgressBar.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.68 PPxPushButton.h File Reference

7.68.1 Detailed Description

A system push button control.

Definition in file PPxPushButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.69 PPxQuickdrawUtils.h File Reference

7.69.1 Detailed Description

Utility classes and functions for working with Quickdraw.

Definition in file PPxQuickdrawUtils.h.

```
#include <PPxPrefix.h>
#include <Quickdraw.h>
```

Namespaces

- namespace PPx

7.70 PPxRadioButton.h File Reference

7.70.1 Detailed Description

A system radio button control.

Definition in file PPxRadioButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.71 PPxRadioGroup.h File Reference

7.71.1 Detailed Description

A system radio group control.

Definition in file PPxRadioGroup.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.72 PPxRegisterAll.h File Reference

7.72.1 Detailed Description

Helper functions for registering items related to the PPx persistence mechanism.

Definition in file PPxRegisterAll.h.

```
#include <PPxPrefix.h>
```

Namespaces

- namespace PPx

7.73 PPxRegistrar.h File Reference

7.73.1 Detailed Description

Functions for managing a table of class names and creator functions used for implementig new-by-name for Persistent objects.

Definition in file PPxRegistrar.h.

`#include <SysCFString.h>`

`#include <map>`

Namespaces

- namespace PPx
- namespace PPx::Registrar

Defines

- #define PPx_RegisterPersistent_(Class) PPx::Registrar::RegisterClass< Class >(CFSTR(#Class))

  Registers a subclass of PPx::Persistent so that objects can be created via new-by-name using the Registrar.

7.73.2 Define Documentation

7.73.2.1 #define PPx_RegisterPersistent_(Class) PPx::Registrar::RegisterClass< Class >(CFSTR(#Class))

Registers a subclass of PPx::Persistent so that objects can be created via new-by-name using the Registrar.

Parameters:

Class Class name

Definition at line 91 of file PPxRegistrar.h.

7.74 PPxRelevanceBar.h File Reference

7.74.1 Detailed Description

A system relevance bar control.

Definition in file PPxRelevanceBar.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.75 PPxResourceFork.h File Reference

7.75.1 Detailed Description

Class for accessing a file's resource fork.

Definition in file PPxResourceFork.h.

```
#include <PPxFileFork.h>
```

Namespaces

- namespace PPx

7.76 PPxRetained.h File Reference

7.76.1 Detailed Description

Classes for reference counted objects.

Definition in file PPxRetained.h.

```
#include <PPxPrefix.h>
```

Namespaces

- namespace PPx

7.77 PPxRoundButton.h File Reference

7.77.1 Detailed Description

A system round button control.

Definition in file PPxRoundButton.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.78 PPxScrollableEvents.h File Reference

7.78.1 Detailed Description

Carbon event handlers for scrollable events.

Definition in file PPxScrollableEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.79 PPxScrollBar.h File Reference

7.79.1 Detailed Description

A system scroll bar control.

Definition in file PPxScrollBar.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.80 PPxScrollView.h File Reference

7.80.1 Detailed Description

A system scroll view.

Definition in file PPxScrollView.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.81 PPxSeparatorLine.h File Reference

7.81.1 Detailed Description

A system separator line view.

Definition in file PPxSeparatorLine.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.82 PPxSerializer.h File Reference

7.82.1 Detailed Description

Routines for reading and writing state information for Persistent objects to flattened data structures.

Definition in file PPxSerializer.h.

```
#include <PPxDataObject.h>
#include <PPxMemoryUtils.h>
#include <deque>
#include <map>
#include <vector>
```

Namespaces

- namespace PPx
- namespace PPx::Serializer

7.83 PPxServiceEvents.h File Reference

7.83.1 Detailed Description

Event handlers for service Carbon Events.

Definition in file PPxServiceEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.84 PPxSheetWindow.h File Reference

7.84.1 Detailed Description

Classes for a sheet window and a sheet alert.

Definition in file PPxSheetWindow.h.

```
#include <PPxCorrespondent.h>
#include <PPxCommandEvents.h>
#include <PPxWindow.h>
```

Namespaces

- namespace PPx

7.85 PPxSignature.h File Reference

7.85.1 Detailed Description

Functions getting and setting the signature of a program.

Definition in file PPxSignature.h.

```
#include <PPxPrefix.h>
```

Namespaces

- namespace PPx
- namespace PPx::Signature

7.86 PPxSlider.h File Reference

7.86.1 Detailed Description

A system slider control.

Definition in file PPxSlider.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.87 PPxStaticText.h File Reference

7.87.1 Detailed Description

A system static text control.

Definition in file PPxStaticText.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.88 PPxStreamUtils.h File Reference

7.88.1 Detailed Description

Utility functions for working with standard streams.

Definition in file PPxStreamUtils.h.

```
#include <PPxPrefix.h>
#include <CFString.h>
#include <ostream>
```

Namespaces

- namespace PPx
- namespace PPx::StreamUtils

Functions

- std::ostream & operator<< (std::ostream &inStream, ConstStringPtr inPascalStr)

  Writes a Pascal string to an output stream.

- std::ostream & operator<< (std::ostream &inStream, CFStringRef inCFString)

  Writes a CFStringRef to an output stream.

- std::ostream & operator<< (std::ostream &inStream, Point inPoint)

  Writes a Point struct to an output stream.

- std::ostream & operator<< (std::ostream &inStream, const Rect &inRect)

  Writes a Rect struct to an output stream.

7.88.2 Function Documentation

7.88.2.1 std::ostream& operator<< (ostream & *inStream*, const Rect & *inRect*)

Writes a Rect struct to an output stream.

Parameters:

inStream Output stream

inRect Rect to write

Returns:

Reference to output stream object

Definition at line 118 of file PPxStreamUtils.cp.

7.88.2.2 std::ostream& operator<< (ostream & *inStream*, Point *inPoint*)

Writes a Point struct to an output stream.

Parameters:

inStream Output stream

inPoint Point to write

Returns:

Reference to output stream object

Definition at line 96 of file PPxStreamUtils.cp.

7.88.2.3 std::ostream& operator<< (ostream & *inStream*, CFStringRef *inCFString*)

Writes a CFStringRef to an output stream.

Parameters:

inStream Output stream

inCFString String to write

Returns:

Reference to output stream object

Definition at line 42 of file PPxStreamUtils.cp.

7.88.2.4 std::ostream& operator<< (ostream & *inStream*, ConstStringPtr *inPascalStr*)

Writes a Pascal string to an output stream.

Parameters:

inStream Output stream

inPascalStr String to write

Returns:

Reference to output stream object

Definition at line 20 of file PPxStreamUtils.cp.

7.89 PPxSysTypes.h File Reference

7.89.1 Detailed Description

Wrapper classes for Toolbox integer types.

PPx uses function overloading and template argument deduction based on the type of parameters. This requires that different kinds of parameters have unique types.

The Toolbox uses typedefs to define descriptive type names for what are actually integer types. For example, OSStatus is a signed long. However, a typedef declares an alias and not a unique type. So, to the C++ compiler, OSStatus and signed long are the same type, and can't be used to distinguish between different overloaded functions.

The structs in this file are wrappers for integer types so that they have a unique type. Each struct has a single data member, a constructor with a default value, and coercion operators to the underlying type. The name of each struct is the same as the name of the Toolbox typedef with the word "Type" appended.

Definition in file PPxSysTypes.h.

```
#include <PPxPrefix.h>
```

```
#include <Controls.h>
```

```
#include <CarbonEvents.h>
```

```
#include <Files.h>
```

```
#include <MacWindows.h>
```

```
#include <Menus.h>
```

Namespaces

- namespace PPx

7.90 PPxTabView.h File Reference

7.90.1 Detailed Description

A system tab view.

Definition in file PPxTabView.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.91 PPxTextGroupBox.h File Reference

7.91.1 Detailed Description

A system group box with a text title.

Definition in file PPxTextGroupBox.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.92 PPxTextInputEvents.h File Reference

7.92.1 Detailed Description

Event handlers for text input Carbon Events.

Definition in file PPxTextInputEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.93 PPxThemeTextBox.h File Reference

7.93.1 Detailed Description

View for drawing text using a theme font inside a bounding box.

Definition in file PPxThemeTextBox.h.

```
#include <PPxBaseView.h>
```

Namespaces

- namespace PPx

7.94 PPxTimer.h File Reference

7.94.1 Detailed Description

Base classes for event loop timers and idle timers.

Definition in file PPxTimer.h.

```
#include <SysEventLoopTimer.h>
```

Namespaces

- namespace PPx

7.95 PPxToolbarEvents.h File Reference

7.95.1 Detailed Description

Event handlers for toolbar and toolbar item Carbon Events.

Definition in file PPxToolbarEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.96 PPxTypes.h File Reference

7.96.1 Detailed Description

Common type definitions.

Definition in file PPxTypes.h.

```
#include <MacTypes.h>
```

Namespaces

- namespace PPx

7.97 PPxView.h File Reference

7.97.1 Detailed Description

Abstract base class for a visual element.

Definition in file PPxView.h.

```
#include <PPxEventTarget.h>
#include <PPxIdentifiable.h>
#include <PPxAttachable.h>
#include <PPxViewEvents.h>
#include <SysHIView.h>
#include <vector>
```

Namespaces

- namespace PPx

7.98 PPxViewEvents.h File Reference

7.98.1 Detailed Description

Event handlers for view Carbon Events (kEventClassControl).

Definition in file PPxViewEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.99 PPxViewUtils.h File Reference

7.99.1 Detailed Description

Class and functions for working with Views.

Definition in file PPxViewUtils.h.

```
#include <PPxPrefix.h>
#include <Appearance.h>
```

Namespaces

- namespace PPx
- namespace PPx::ViewUtils

7.100 PPxWindow.h File Reference

7.100.1 Detailed Description

Window for displaying data on screen.

Definition in file PPxWindow.h.

```
#include <PPxEventTarget.h>
#include <PPxAttachable.h>
#include <PPxWindowEvents.h>
#include <SysWindow.h>
#include <memory>
```

Namespaces

- namespace PPx

7.101 PPxWindowContentView.h File Reference

7.101.1 Detailed Description

Top-level view for the contents of a window.

Definition in file PPxWindowContentView.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.102 PPxWindowDefEvents.h File Reference

7.102.1 Detailed Description

Event handlers for window definition Carbon Events.

Definition in file PPxWindowDefEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.103 PPxWindowEvents.h File Reference

7.103.1 Detailed Description

Event handlers for window Carbon Events.

Definition in file PPxWindowEvents.h.

```
#include <PPxEventDoer.h>
```

Namespaces

- namespace PPx

7.104 PPxWindowHeader.h File Reference

7.104.1 Detailed Description

A system window header view.

Definition in file PPxWindowHeader.h.

```
#include <PPxView.h>
```

Namespaces

- namespace PPx

7.105 PPxXMLConstants.h File Reference

7.105.1 Detailed Description

Constants for XML identifiers.

Definition in file PPxXMLConstants.h.

```
#include <CFString.h>
```

Namespaces

- namespace PPx
- namespace PPx::XMLConstants

7.106 PPxXMLDecoder.h File Reference

7.106.1 Detailed Description

Functiions for converting information in XML Trees to Data Objects.

Definition in file PPxXMLDecoder.h.

```
#include <PPxDataObject.h>
#include <SysCFXMLNode.h>
#include <SysCFXMLTree.h>
#include <typeinfo>
```

Namespaces

- namespace PPx
- namespace PPx::XMLDecoder
- namespace PPx::XMLDecoderFuncs
- namespace PPx::XMLTreeBrowser

7.107 PPxXMLSerializer.h File Reference

7.107.1 Detailed Description

Definition in file PPxXMLSerializer.h.

```
#include <PPxSerializer.h>
```

```
#include <PPxPrimaryBundle.h>
```

Namespaces

- namespace PPx
- namespace **PPx::XMLSerializer**

7.108 SysAEDesc.h File Reference

7.108.1 Detailed Description

Classes and functions for working with Apple Events.

Definition in file SysAEDesc.h.

```
#include <SysEventParam.h>
```

```
#include <AppleEvents.h>
```

Namespaces

- namespace PPx

7.109 SysAEHandler.h File Reference

7.109.1 Detailed Description

Utility classes for managing Apple Event Handlers.

Definition in file SysAEHandler.h.

```
#include <PPxPrefix.h>
```

```
#include <AppleEvents.h>
```

Namespaces

- namespace PPx

7.110 SysAppleEvent.h File Reference

7.110.1 Detailed Description

Wrapper class for an Apple Event.

Definition in file SysAppleEvent.h.

```
#include <PPxPrefix.h>
#include <AppleEvents.h>
```

Namespaces

- namespace PPx

7.111 SysCarbonEvent.h File Reference

7.111.1 Detailed Description

Classes for managing Carbon Events.

Definition in file SysCarbonEvent.h.

```
#include <PPxPrefix.h>
#include <PPxConstants.h>
#include <CarbonEvents.h>
```

Namespaces

- namespace PPx

7.112 SysCFArray.h File Reference

7.112.1 Detailed Description

Template class wrapper for a Core Foundation Array.

Definition in file SysCFArray.h.

```
#include <SysCFMutableObject.h>
#include <SysCFUtils.h>
#include <CFArray.h>
```

Namespaces

- namespace PPx

7.113 SysCFBundle.h File Reference

7.113.1 Detailed Description

Wrapper class for Core Foundation Bundle.

Definition in file SysCFBundle.h.

```
#include <SysCFObject.h>
#include <SysCFArray.h>
#include <SysCFDictionary.h>
#include <SysCFString.h>
#include <SysCFURL.h>
#include <CFBundle.h>
```

Namespaces

- namespace PPx

7.114 SysCFData.h File Reference

7.114.1 Detailed Description

Wrapper class for a Core Foundataion Data object.

Definition in file SysCFData.h.

```
#include <SysCFMutableObject.h>
#include <CFData.h>
```

Namespaces

- namespace PPx

7.115 SysCFDictionary.h File Reference

7.115.1 Detailed Description

Definition in file SysCFDictionary.h.

```
#include <SysCFMutableObject.h>
#include <CFDictionary.h>
```

Namespaces

- namespace PPx

7.116 SysCFMutableObject.h File Reference

7.116.1 Detailed Description

Template base class for Core Foundation wrapper classes for mutable object.

Definition in file SysCFMutableObject.h.

```
#include <SysCFObject.h>
```

Namespaces

- namespace PPx

7.117 SysCFObject.h File Reference

7.117.1 Detailed Description

Template base class for Core Foundation wrapper classes.

Definition in file SysCFObject.h.

`#include <PPxPrefix.h>`

`#include <CFBase.h>`

Namespaces

- namespace PPx

Defines

- #define PPx_ThrowIfCFCreateFailed_(inCFRef, inFuncName)

  Macro for throwing an exception if a Toolbox function which creates a Core Foundation object fails.

7.117.2 Define Documentation

7.117.2.1 #define PPx_ThrowIfCFCreateFailed_(inCFRef, inFuncName)

Value:

```
PPx_ThrowIfNil_(inCFRef, RuntimeError, PPx::err_CFCreate,   \
                            inFuncName " failed")
```

Macro for throwing an exception if a Toolbox function which creates a Core Foundation object fails.

The CF reference returned is nil if creation failed.

Parameters:

inCFRef A CF reference

inFuncName Literal string name of CF creator function

Definition at line 496 of file SysCFObject.h.

Referenced by PPx::CFURL::AppendPathComponent(), PPx::CFURL::AppendPathExtension(), PPx::CFArray< TValue >::CFArray(), PPx::CFBundle::CFBundle(), PPx::CFData::CFData(), PPx::CFDictionary< TKey, TValue >::CFDictionary(), PPx::CFString::CFString(), PPx::CFTree::CFTree(), PPx::CFURL::CFURL(), PPx::CFXMLTree::CFXMLTree(), PPx::CFURL::DeleteLastPathComponent(), and PPx::CFURL::DeletePathExtension().

7.118 SysCFString.h File Reference

7.118.1 Detailed Description

Wrapper class for Core Foundation String.

Definition in file SysCFString.h.

```
#include <SysCFMutableObject.h>
#include <CFString.h>
#include <string>
#include <sstream>
```

Namespaces

- namespace PPx

7.119 SysCFTree.h File Reference

7.119.1 Detailed Description

Wrapper class for Core Foundation Tree.

Definition in file SysCFTree.h.

```
#include <SysCFObject.h>
#include <CFTree.h>
```

Namespaces

- namespace PPx

7.120 SysCFURL.h File Reference

7.120.1 Detailed Description

Wrapper class for Core Foundation URL.

Definition in file SysCFURL.h.

```
#include <SysCFObject.h>
#include <SysCFData.h>
#include <SysCFString.h>
#include <CFURL.h>
```

Namespaces

- namespace PPx

7.121 SysCFUtils.h File Reference

7.121.1 Detailed Description

Utility functions for working with CoreFoundation.

Definition in file SysCFUtils.h.

```
#include <PPxPrefix.h>
#include <TextCommon.h>
#include <sstream>
```

Namespaces

- namespace PPx
- namespace PPx::CFUtils

Functions

- bool operator== (const CFRange &inLeft, const CFRange &inRight)

  Equality operator for CFRange.

- bool operator!= (const CFRange &inLeft, const CFRange &inRight)

  Inequality operator for CFRange.

7.121.2 Function Documentation

7.121.2.1 bool operator!= (const CFRange & *inLeft*, const CFRange & *inRight*) `[inline]`

Inequality operator for CFRange.

Parameters:

inLeft Left side of != operator

inRight Right side of != operator

Returns:

Whether the two CFRanges are not equal

Definition at line 48 of file SysCFUtils.h.

7.121.2.2 bool operator== (const CFRange & *inLeft*, const CFRange & *inRight*) `[inline]`

Equality operator for CFRange.

Parameters:

inLeft Left side of == operator

inRight Right side of == operator

Returns:

Whether the two CFRanges are equal

Definition at line 30 of file SysCFUtils.h.

7.122 SysCFXMLNode.h File Reference

7.122.1 Detailed Description

Wrapper class for Core Foundation XML Node.

Definition in file SysCFXMLNode.h.

```
#include <SysCFArray.h>
#include <SysCFDictionary.h>
#include <SysCFObject.h>
#include <SysCFString.h>
#include <CFXMLNode.h>
```

Namespaces

- namespace PPx

7.123 SysCFXMLTree.h File Reference

7.123.1 Detailed Description

Wrapper class for Core Foundation XML Tree.

Definition in file SysCFXMLTree.h.

```
#include <SysCFTree.h>
#include <SysCFData.h>
#include <SysCFXMLNode.h>
#include <CFData.h>
#include <CFXMLParser.h>
```

Namespaces

- namespace PPx

7.124 SysCreateView.h File Reference

7.124.1 Detailed Description

Wrapper functions for creating system view objects.

Definition in file SysCreateView.h.

```
#include <PPxPrefix.h>
```

Namespaces

- namespace PPx
- namespace PPx::SysCreateView

7.125 SysEventHandler.h File Reference

7.125.1 Detailed Description

Utility classes for managing Carbon Event Handlers.

Definition in file SysEventHandler.h.

```
#include <PPxPrefix.h>
#include <CarbonEvents.h>
```

Namespaces

- namespace PPx

7.126 SysEventLoopTimer.h File Reference

7.126.1 Detailed Description

Wrapper classes for event loop timers and idle timers.

Definition in file SysEventLoopTimer.h.

```
#include <PPxPrefix.h>
#include <CarbonEvents.h>
```

Namespaces

- namespace PPx

7.127 SysEventParam.h File Reference

7.127.1 Detailed Description

Utility functions for getting and setting Carbon Event parameters.

Definition in file SysEventParam.h.

`#include <PPxSysTypes.h>`

`#include <SysEventTypes.h>`

`#include <CarbonEvents.h>`

Namespaces

- namespace PPx
- namespace PPx::SysEventParam

Defines

- #define PPx_Declare_SysEventParam_Traits(DataT, ParamT)

  Macro for declaring Traits template specializations.

7.127.2 Define Documentation

7.127.2.1 #define PPx_Declare_SysEventParam_Traits(DataT, ParamT)

Value:

```
template <> struct Traits< DataT > {                        \
            static const EventParamType type = ParamT;          \
        }
```

Macro for declaring Traits template specializations.

Definition at line 41 of file SysEventParam.h.

7.128 SysEventTypes.h File Reference

7.128.1 Detailed Description

Wrapper classes for types used as Carbon Event parameters.

Definition in file SysEventTypes.h.

```
#include <PPxPrefix.h>
```

```
#include <HIToolbar.h>
```

Namespaces

- namespace PPx

7.129 SysHIObject.h File Reference

7.129.1 Detailed Description

Wrapper class for a Mac Toolbox HIObject.

Definition in file SysHIObject.h.

```
#include <PPxPrefix.h>
```

```
#include <HIObject.h>
```

Namespaces

- namespace PPx

7.130 SysHIView.h File Reference

7.130.1 Detailed Description

Wrapper class for a Mac Toolbox HIView.

Definition in file SysHIView.h.

```
#include <SysCFString.h>
#include <HIView.h>
```

Namespaces

- namespace PPx

7.131 SysScrap.h File Reference

7.131.1 Detailed Description

Wrapper functions for the Scrap Manager.

Definition in file SysScrap.h.

```
#include <PPxPrefix.h>
```

```
#include <Scrap.h>
```

Namespaces

- namespace PPx
- namespace PPx::SysScrap

7.132 SysWindow.h File Reference

7.132.1 Detailed Description

Wrapper class for a Mac Toolbox Window.

Definition in file SysWindow.h.

```
#include <SysCFString.h>
#include <MacWindows.h>
```

Namespaces

- namespace PPx

Index