Image Kit Reference Collection

Graphics & Animation: 2D Drawing

2006-12-06

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

Image Kit Reference Collection

Framework	System/Library/Frameworks/Quartz.framework
Header file directories	Quartz.framework/ImageKit.framework
Declared in	IKFilterBrowserPanel.h IKFilterBrowserView.h IKFilterUI.h IKFilterUIView.h IKImageBrowserView.h IKImageEditPanel.h IKImageView.h IKPictureTaker.h IKSaveOptions.h IKSlideshow.h

Introduction

Important: This is a preliminary document for an API in development. Although this document has been reviewed for technical accuracy, it is not final. Apple Computer is supplying this information to help you plan for the adoption of the technologies and programming interfaces described herein. This information is subject to change, and software implemented according to this document should be tested with final operating system software and final documentation. Newer versions of this document may be provided with future seeds of the API. For information about updates to this and other developer documentation, view the New & Updated sidebars in subsequent seeds of the Reference Library.

The Image Kit framework is an image handling framework that's based on Quartz, Core Image, OpenGL, and Layer Kit. It defines a set of Objective-C classes that provide user interface support for:

- Finding, browsing, viewing, and editing images. The Image Kit framework builds on the `NSView` class to provide a view that is optimized for image operations such as rotating, zooming, scrolling, and dragging. Client applications can support powerful in-place editing with a minimum effort.
- Browsing, previewing, and setting input parameters for Core Image filters. The Image Kit framework extends the Core Image API by defining additions to the `CIFilter` class that make it easy for applications to present a user interface for a Core Image filter.
- Extending the Save panel with options for saving images
- Showing slideshows

Classes

CIFilter Image Kit Additions

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	IIKFilterUI.h
Companion guide	Core Image Programming Guide

Overview

This Image Kit addition to the `CIFilter` class, introduced in Mac OS X v10.5, consists of one method and a set of constants that generate a view with input parameter controls for a Core Image filter. Using this method, it is easier for applications to present a user interface for a filter than it was in Mac OS X v10.4. Then, applications could create a filter user interface only by analyzing the keys and key attributes of a filter and then writing the code to implement the user interface.

You use the `viewForUIConfiguration:excludedKeys:` method to request a view from Core Image. The view is a subclass of the `NSView` class so that you can insert it easily into any other view as a subview or into an `NSWindow` object as a content view. Core Image automatically generates the view for you unless you implement the `IKFilterCustomUIProvider` protocol, in which case calling `viewForUIConfiguration:excludedKeys:` causes Core Image to provide your custom view.

Tasks

Creating a View for a Filter

`- viewForUIConfiguration:excludedKeys:` (page 12)
Returns a filter view for the filter.

Instance Methods

viewForUIConfiguration:excludedKeys:

Returns a filter view for the filter.

```
-(IKFilterUIView*)viewForUIConfiguration:(NSDictionary*)inUIConfiguration
    excludedKeys:(NSArray*)inKeys;
```

Parameters

inUIConfiguration

A dictionary that contains values for the `IKUISizeFlavor` and `kCIUIParameterSet` keys. See "User Interface Options" (page 13) for the constants that you can provide as values for `IKUISizeFlavor`. For `kCIUIParameterSet` you can provide one of the following values: `kCIUISetBasic`, `kCIUISetIntermediate`, `kCIUISetAdvanced`, or `kCIUISetDevelopment`. When you request a user interface for a parameter set, all keys for that set and below are included. For example, the advanced set consists of all parameters in the basic, intermediate and advanced sets. The development set should contain parameters that are either experimental or for debugging purposes. You should use them only during the development of filters and client applications, and not in a shipping product.

inKeys

An array of the input keys for which you do *not* want to provide a user interface. Pass `nil` if you want all input keys to be represented in the user interface.

Return Value

An `IKFilterUIView` object. You should retain the view as long as you need it, but make sure to release it when you no longer need it as the view is retaining the filter.

Discussion

Calling this method to receive a view for a filter causes the `CIFilter` class to invoke the `provideViewForUIConfiguration:excludedKeys:` (page 91) method. If you override `provideViewForUIConfiguration:excludedKeys:` the user interface is created by your filter subclass . Otherwise, Core Image automatically generates the user interface based on the filter keys and attributes.

The algorithm used to lay out the controls for a filter operates in a manner similar to the Core Image Fun House application (`/Developer/Applications/Graphics Tools/`). Applications can retrieve a view whose control sizes complement the size of user interface elements already used in the application. It is also possible to choose which filter input parameters appear in the view. Consumer applications, for example, may want to show a small, basic set of input parameters whereas professional applications may want to provide access to all input parameters.

The controls in the view use bindings to set the values of the filter. See *Cocoa Bindings Programming Topics* if you are unfamiliar with bindings.

Availability

Available in Mac OS X v10.5 and later.

Related Sample Code

CIRAWFilterSample

Declared In

`IKFilterUI.h`

Constants

User Interface Options

Keys or values for the size of the input parameter controls for a filter view.

```
NSString *IKUISizeFlavor;
NSString *IKUISizeMini;
NSString *IKUISizeSmall;
NSString *IKUISizeRegular;
NSString *IKUImaxSize;
NSString *IKUIFlavorAllowFallback;
```

Constants

`IKUISizeFlavor`

A key for the size of the controls in a filter view. The associated value can be `IKUISizeMini`, `IKUISizeSmall`, or `IKUISizeRegular`.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterUI.h`.

`IKUISizeMini`

Controls whose size is mini, as defined by Interface Builder 2.5.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterUI.h`.

`IKUISizeSmall`

Controls whose size is small, as defined by Interface Builder 2.5.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterUI.h`.

`IKUISizeRegular`

Controls whose size is regular or normal, as defined by Interface Builder 2.5.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterUI.h`.

`IKUImaxSize`

Controls whose dimensions are the maximum allowable for the filter view. A width or height of `0` indicates that that dimension of the view is not restricted. If the size requested is too small, the filter is expected to return a view as small as possible. It is up to the client to verify that the returned view fits into the context.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterUI.h`.

`IKUIFlavorAllowFallback`

Substitute controls of another size. The associated value is a Boolean value. If the filter cannot provide a view for the requested size and a fallback is allowed, the filter can use controls of a different size.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterUI.h`.

Declared In

`IKFilterUI.h`

IKFilterBrowserPanel Class Reference

Inherits from	NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKFilterBrowserPanel.h
Related sample code	CIRAWFilterSample

Overview

The `IKFilterBrowserPanel` class provides a user interface that allows users to browse Core Image filters (`CIFilter`), to preview a filter, and to get additional information about the filter, such as its description.

An `IKFilterBrowserPanel` object can be displayed as:

- a separate panel, that is, a utility window that floats on top of document windows
- a modal dialog
- a sheet, that is, a dialog that is attached to its parent window and must be dismissed by the user
- a view that an application can insert into a custom user interface

An `IKFilterBrowserPanel` object can be configured through a style mask to use either the default or brushed metal look for windows. The size and number of visible controls are specified through an options dictionary. An `IKFilterBrowserPanel` object communicates selection changes through notifications.

The `IKFilterBrowserPanel` class allows the user to create filter collections that are stored with the `filterCollections` key in the `com.apple.CoreImageKit.plist` property list located in `~/Library/Preferences/`.

Tasks

Getting a Filter Name

`- filterName` (page 19)
Returns the name of the filter that is currently selected in the filter browser.

Displaying and Running the Panel

`- filterBrowserViewWithOptions:` (page 19)
Returns a view that contains a filter browser.

`- beginWithOptions:modelessDelegate:didEndSelector:contextInfo:` (page 18)
Displays the filter browser in a new utility window, unless the filter browser is already open.

`- beginSheetWithOptions:modalForWindow:modalDelegate:didEndSelector:contextInfo:` (page 17)
Displays the filter browser in a sheet—that is, a dialog that is attached to its parent window and must be dismissed by the user.

`- runModalWithOptions:` (page 20)
Displays the filter browser in a modal dialog that must be dismissed by the user but that is not attached to a window.

`- finish:` (page 19)
Closes a filter browser view.

Creating a Filter Browser Panel

`+ filterBrowserPanelWithStyleMask:` (page 16)
Creates a shared instance of the `IKFilterBrowserPanel` class.

Class Methods

filterBrowserPanelWithStyleMask:

Creates a shared instance of the `IKFilterBrowserPanel` class.

```
+ (id)filterBrowserPanelWithStyleMask:(unsigned int)styleMask;
```

Parameters

styleMask
A mask that specifies whether to use the default or brushed metal look for the window. You can select or deselect the `NSTexturedBackgroundWindowMask` style bit.

Return Value
The shared instance.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CIRAWFilterSample

Declared In
`IKFilterBrowserPanel.h`

Instance Methods

beginSheetWithOptions:modalForWindow:modalDelegate:didEndSelector:contextInfo:

Displays the filter browser in a sheet—that is, a dialog that is attached to its parent window and must be dismissed by the user.

```
- (void)beginSheetWithOptions:(NSDictionary*)inOptions modalForWindow:(NSWindow
    *)docWindow modalDelegate:(id)modalDelegate didEndSelector:(SEL)didEndSelector
     contextInfo:(void *)contextInfo;
```

Parameters

inOptions
: A dictionary of options that describe the configuration to use for the filter browser user interface. For the possible keys you can supply see "Filter Browser Option Keys" (page 20) and the constant `IKUISizeFlavor`.

modalForWindow
: The parent window for the dialog.

modalDelegate
: The object that will invoke the selector `didEndSelector` when the filter browser session terminates.

didEndSelector
: The selector to invoke when the filter browser session terminates.

contextInfo
: Any data that must be passed as an argument to the delegate through `didEndSelector` after the filter browser session terminates.

Discussion
When the filter browser session ends, `didEndSelector` is invoked on the modeless delegate, passing `contextInfo` as an argument. The selector `didEndSelector` must have the following signature:

```
- (void)openPanelDidEnd:(NSOpenPanel *)panel returnCode:(int)returnCode
contextInfo:(void *)contextInfo
```

The `returnCode` value passed to the selector is set to `NSOKButton` if the user validates, or to `NSCancelButton` if the user cancels.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- beginWithOptions:modelessDelegate:didEndSelector:contextInfo:` (page 18)

- runModalWithOptions (page 20)

Declared In
IKFilterBrowserPanel.h

beginWithOptions:modelessDelegate:didEndSelector:contextInfo:

Displays the filter browser in a new utility window, unless the filter browser is already open.

```
- (void)beginWithOptions:(NSDictionary*)inOptions
    modelessDelegate:(id)modelessDelegate didEndSelector:(SEL)didEndSelector
    contextInfo:(void *)contextInfo;
```

Parameters

inOptions

A dictionary of options that describe the configuration to use for the filter browser user interface. For the possible keys you can supply see "Filter Browser Option Keys" (page 20) and the constant IKUISizeFlavor.

modelessDelegate

The object that will invoke the selector didEndSelector when the filter browser session terminates.

didEndSelector

The selector to invoke when the filter browser session terminates.

contextInfo

Any data that must be passed as an argument to the delegate through didEndSelector after the filter browser session terminates.

Discussion
When the filter browser session ends, didEndSelector is invoked on the modeless delegate, passing contextInfo as an argument. The selector didEndSelector must have the following signature:

```
- (void)openPanelDidEnd:(NSOpenPanel *)panel returnCode:(int)returnCode
contextInfo:(void *)contextInfo
```

The returnCode value passed to the selector is set to NSOKButton if the user validates, or to NSCancelButton if the user cancels.

Availability
Available in Mac OS X v10.5 and later.

See Also
- beginSheetWithOptions:modalForWindow:modalDelegate:didEndSelector:contextInfo: (page 17)
- runModalWithOptions (page 20)

Related Sample Code
CIRAWFilterSample

Declared In
IKFilterBrowserPanel.h

filterBrowserViewWithOptions:

Returns a view that contains a filter browser.

```
- (IKFilterBrowserView*)filterBrowserViewWithOptions:(NSDictionary*)inOptions;
```

Parameters

inOptions

A dictionary of options that describe the configuration to use for the filter browser user interface. For the possible keys you can supply see "Filter Browser Option Keys" (page 20) and the constant IKUISizeFlavor.

Return Value
A filter browser view that is configured as specified.

Discussion
Use this method to add a view that contains the filter browser to your custom user interface. To dismiss the filter browser view, invoke the finish (page 19) method.

Availability
Available in Mac OS X v10.5 and later.

Declared In
IKFilterBrowserPanel.h

filterName

Returns the name of the filter that is currently selected in the filter browser.

```
- (NSString*)filterName;
```

Return Value
The name of the currently selected filter.

Discussion
Use this method in response to the notifications IKFilterBrowserFilterSelectedNotification (page 22) or IKFilterBrowserFilterDoubleClickNotification (page 22), or after the user makes a choice in a dialog.

Availability
Available in Mac OS X v10.5 and later.

Declared In
IKFilterBrowserPanel.h

finish:

Closes a filter browser view.

```
- (void)finish:(id)sender;
```

Parameters

sender

The object that invokes the action, such as an OK or Cancel button.

Discussion
Invoke this action when you want to dismiss the filter browser.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- filterBrowserViewWithOptions` (page 19)

Declared In
`IKFilterBrowserPanel.h`

runModalWithOptions:

Displays the filter browser in a modal dialog that must be dismissed by the user but that is not attached to a window.

```
- (int)runModalWithOptions:(NSDictionary*)inOptions;
```

Parameters

inOptions

A dictionary of options that describe the configuration to use for the filter browser user interface. For the possible keys you can supply see "`Filter Browser Option Keys`" (page 20) and the constant `IKUISizeFlavor`.

Return Value
Either `NSOKButton` if the user validates, or `NSCancelButton` if the user cancels.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- beginSheetWithOptions:modalForWindow:modalDelegate:didEndSelector:contextInfo:` (page 17)
`- beginWithOptions:modelessDelegate:didEndSelector:contextInfo:` (page 18)

Declared In
`IKFilterBrowserPanel.h`

Constants

Filter Browser Option Keys

Keys for filter browser options.

```
NSString *const IKFilterBrowserDefaultInputImage;
NSString *const IKFilterBrowserExcludeCategories;
NSString *const IKFilterBrowserExcludeFilters;
NSString *const IKFilterBrowserShowCategories;
NSString *const IKFilterBrowserShowPreview;
```

Constants

`IKFilterBrowserDefaultInputImage`

The key for the default input image. The associated value is the `CIImage` object to use as the default input image for the filter preview. Setting the image to `nil` causes Image Kit to use the image supplied by the framework. You can also set the input image and other parameters during the notification IKFilterBrowserWillPreviewFilterNotification (page 21).

Available in Mac OS X v10.5 and later.

Declared in `IKFilterBrowserPanel.h`.

`IKFilterBrowserExcludeCategories`

The key for excluding filter categories. The associated value is an `NSArray` object that lists the categories that you do *not* want to display in the filter browser.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterBrowserPanel.h`.

`IKFilterBrowserExcludeFilters`

The key for excluding filters. The associated value is an `NSArray` object that lists the filters that you do *not* want to display in the filter browser.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterBrowserPanel.h`.

`IKFilterBrowserShowCategories`

The key for showing categories. The associated value is a `BOOL` value that determines if the filter browser should show the category list.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterBrowserPanel.h`.

`IKFilterBrowserShowPreview`

The associated value is a `BOOL` value that determines if the filter browser should provide a preview.

Available in Mac OS X v10.5 and later.

Declared in `IKFilterBrowserPanel.h`.

Declared In

`IKFilterBrowserPanel.h`

Notifications

IKFilterBrowserWillPreviewFilterNotification

Posted before showing a filter preview, allowing an application to set the parameters of a filter.

The selected filter is sent as the object in the notification.

Availability

Available in Mac OS X v10.5 and later.

Declared In
`IKFilterBrowserPanel.h`

IKFilterBrowserFilterSelectedNotification

Posted when the user clicks a filter name in the filter browser.

The name of the selected filter is sent as the object in the notification.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKFilterBrowserPanel.h`

IKFilterBrowserFilterDoubleClickNotification

Posted when the user double-clicks a filter in the filter browser.

The name of the selected filter is send as the object in the notification.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKFilterBrowserPanel.h`

IKFilterBrowserView Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKFilterBrowserView.h

Overview

The `IKFilterBrowserView` class is used as a container for the elements of an `IKFilterBrowserPanel` object.

Tasks

Setting the Preview State

`- setPreviewState:` (page 24)
Sets the preview state.

Getting the Filter Name

`- filterName` (page 23)
Returns the name of the filter that is currently selected in the filter browser.

Instance Methods

filterName

Returns the name of the filter that is currently selected in the filter browser.

```
- (NSString*)filterName;
```

Return Value
The name of the currently selected filter.

Discussion
Use this method in response to the notifications `IKFilterBrowserFilterSelectedNotification` (page 22) or `IKFilterBrowserFilterDoubleClickNotification` (page 22), or after the user makes a choice in a dialog.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKFilterBrowserView.h`

setPreviewState:

Sets the preview state.

```
- (void)setPreviewState:(BOOL)inState;
```

Parameters
inState
A state (`YES` or `NO`) that represents whether a preview is visible.

Discussion
Use this method to show and hide the preview programmatically.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKFilterBrowserView.h`

IKFilterUIView Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKFilterUIView.h
Related sample code	CIRAWFilterSample

Overview

The `IKFilterUIView` class provides a view that contains input parameter controls for a Core Image filter (`CIFilter`). You need to use this class when providing a user interface for a custom filter. The class creates a view that has an object controller for the given filter. It also retains the filter.

Tasks

Creating and Initializing a Filter UI View

`+ viewWithFrame:filter:` (page 26)
Creates a view that contains controls for the input parameters of a filter.

`- initWithFrame:filter:` (page 26)
Initializes a view that contains controls for the input parameters of a filter.

Getting Data from the Filter View

`- filter` (page 26)
Returns the Core Image filter associated with the view.

`- objectController` (page 27)
Returns the object controller for the bindings between the filter and its view.

Class Methods

viewWithFrame:filter:

Creates a view that contains controls for the input parameters of a filter.

```
+ (id)viewWithFrame:(NSRect)frameRect filter:(CIFilter *)inFilter
```

Parameters

frameRect

The rectangle that defines the area of the view.

inFilter

A Core Image filter. The view retains the filter.

Return Value

An `IKFilterUIView` object that contains controls for the input parameters of the provided filter.

Availability

Available in Mac OS X v10.5 and later.

See Also

`- initWithFrame:filter:` (page 26)

Declared In

`IKFilterUIView.h`

Instance Methods

filter

Returns the Core Image filter associated with the view.

```
- (CIFilter *)filter
```

Return Value

The Core Image filter associated with the view.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKFilterUIView.h`

initWithFrame:filter:

Initializes a view that contains controls for the input parameters of a filter.

```
- (id)initWithFrame:(NSRect)frameRect filter:(CIFilter *)inFilter
```

Parameters

frameRect

The rectangle that defines the area of the view.

inFilter

A Core Image filter. The view retains the filter.

Return Value

The `IKFilterUIView` object initialized with controls for the input parameters of the provided filter.

Availability

Available in Mac OS X v10.5 and later.

See Also

`+ viewWithFrame:filter:` (page 26)

Declared In

`IKFilterUIView.h`

objectController

Returns the object controller for the bindings between the filter and its view.

```
- (NSObjectController *)objectController
```

Return Value

The object controller for the bindings between the filter and its view.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKFilterUIView.h`

IKImageBrowserView Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKImageBrowserView.h
Related sample code	AutomatorHandsOn DesktopImage ImageBrowser ImageBrowserViewAppearance ImageKitDemo

Overview

The `IKImageBrowserView` class is a view for displaying and browsing a large amount of images and movies efficiently.

Tasks

Initializing and Setting Up an Image Browser View

`- initWithFrame:` (page 37)
Initializes a newly allocated image browser view with the provided frame rectangle.

`- setDataSource:` (page 42)
Sets the data source of the receiver.

`- dataSource` (page 35)
Returns the data source of the receiver.

`- reloadData` (page 38)
Marks the receiver as needing its data reloaded.

`- setDelegate:` (page 42)
Sets the delegate of the receiver.

- `delegate` (page 35)
  Returns the delegate of the receiver.

Setting the Appearance

- `setCellsStyleMask:` (page 41)
  Defines the appearance style of the cells.
- `cellsStyleMask` (page 33)
  Returns the appearance style mask for the cell.
- `setConstrainsToOriginalSize:` (page 41)
  Sets whether the receiver constrains the cell's image to its original size.
- `constrainsToOriginalSize` (page 34)
  Returns whether the receiver constrains the cell's image to its original size.

Zooming and Resizing

- `setZoomValue:` (page 43)
  Sets the zoom value.
- `zoomValue` (page 44)
  Returns the current zoom value.
- `setContentResizingMask:` (page 41)
  Determines how the receiver resizes its content when zooming.
- `contentResizingMask` (page 34)
  Returns the receiver's content resizing mask, which determines how its content is resized while zooming.

Scrolling

- `scrollIndexToVisible:` (page 38)
  Scrolls the receiver to the item at the specified index.

Setting and Getting Cell Size

- `setCellSize:` (page 40)
  Sets the cell size.
- `cellSize` (page 33)
  Returns the cell size.

Getting Item Information

- `indexOfItemAtPoint:` (page 37)
  Returns the index of the item at the specified location.

- `itemFrameAtIndex:` (page 38)
  Returns the frame rectangle for the item located at the specified index.

Reordering and Groups Items

- `selectionIndexes` (page 39)
  Returns the indexes of the selected cells.
- `setSelectionIndexes:byExtendingSelection:` (page 43)
  Selects cells at the specified indexes.
- `setAllowsMultipleSelection:` (page 39)
  Controls whether the user can select more than one cell at a time.
- `allowsMultipleSelection` (page 32)
  Returns whether multiple selections are allowed.
- `setAllowsEmptySelection:` (page 39)
  Controls whether an empty selection is allowed.
- `allowsEmptySelection` (page 32)
  Returns whether an empty selection is allowed.
- `setAllowsReordering:` (page 40)
  Controls whether the user can reorder items.
- `allowsReordering` (page 32)
  Returns whether the user can reorder items.
- `setAnimates:` (page 40)
  Controls whether the receiver animates reordering and changes of the data source.
- `animates` (page 33)
  Returns whether the receiver animates reordering and changes of the data source.
- `expandGroupAtIndex:` (page 36)
  Expands a group at the specified index.
- `collapseGroupAtIndex:` (page 34)
  Collapses a group at the specified index.
- `isGroupExpandedAtIndex:` (page 37)
  Returns whether the group at the provided index is expanded.

Supporting Drag and Drop

- `setDraggingDestinationDelegate:` (page 42)
  Sets the dragging destination delegate of the receiver.
- `draggingDestinationDelegate` (page 35)
  Returns the dragging destination delegate of the receiver.
- `indexAtLocationOfDroppedItem` (page 36)
  Returns the index of the cell where the drop operation occurred.

Instance Methods

allowsEmptySelection

Returns whether an empty selection is allowed.

```
- (BOOL) allowsEmptySelection;
```

Return Value
`YES` if the receiver allows an empty selection; `NO` otherwise.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setAllowsEmptySelection:` (page 39)

Declared In
`IKImageBrowserView.h`

allowsMultipleSelection

Returns whether multiple selections are allowed.

```
- (BOOL) allowsMultipleSelection;
```

Return Value
`YES` if the receiver allows the user to select more than one cell at a time; `NO` otherwise.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setAllowsEmptySelection:` (page 39)

Declared In
`IKImageBrowserView.h`

allowsReordering

Returns whether the user can reorder items.

```
- (BOOL) allowsReordering;
```

Return Value
`YES` if the user can reorder items; `NO` otherwise.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setAllowsReordering:` (page 40)

Declared In
`IKImageBrowserView.h`

animates

Returns whether the receiver animates reordering and changes of the data source.

```
- (BOOL) animates;
```

Return Value
`YES` if the receiver animates reordering and changes of the data source; `NO` otherwise.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setAnimates:` (page 40)

Declared In
`IKImageBrowserView.h`

cellSize

Returns the cell size.

```
- (NSSize) cellSize;
```

Return Value
The current size for the cells in the image browser view.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

cellsStyleMask

Returns the appearance style mask for the cell.

```
- (NSUInteger) cellsStyleMask;
```

Return Value
The appearance style mask for the cell.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setCellsStyleMask:` (page 41)

Declared In
`IKImageBrowserView.h`

collapseGroupAtIndex:

Collapses a group at the specified index.

```
- (void) collapseGroupAtIndex:(NSUInteger) index;
```

Parameters

index
The index of the group you want to collapse.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `expandGroupAtIndex:` (page 36)
- `isGroupExpandedAtIndex:` (page 37)

Declared In
`IKImageBrowserView.h`

constrainsToOriginalSize

Returns whether the receiver constrains the cell's image to its original size.

```
- (BOOL) constrainsToOriginalSize;
```

Return Value
`NO` if the image is not constrained; otherwise `YES`.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setConstrainsToOriginalSize:` (page 41)

Declared In
`IKImageBrowserView.h`

contentResizingMask

Returns the receiver's content resizing mask, which determines how its content is resized while zooming.

```
- (NSUInteger) contentResizingMask;
```

Return Value
The content resizing mask.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setContentResizingMask:` (page 41)

Declared In
`IKImageBrowserView.h`

dataSource

Returns the data source of the receiver.

```
- (id ) dataSource;
```

Return Value
The data source (`IKImageBrowserDataSource`). The data source is not retained by the receiver.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setDataSource:` (page 42)

Declared In
`IKImageBrowserView.h`

delegate

Returns the delegate of the receiver.

```
- (id) delegate;
```

Return Value
The delegate.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setDelegate:` (page 42)

Declared In
`IKImageBrowserView.h`

draggingDestinationDelegate

Returns the dragging destination delegate of the receiver.

```
- (id) draggingDestinationDelegate;
```

Return Value
The receiver's dragging destination delegate.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setDraggingDestinationDelegate:` (page 42)

Declared In
`IKImageBrowserView.h`

expandGroupAtIndex:

Expands a group at the specified index.

```
- (void) expandGroupAtIndex:(NSUInteger) index;
```

Parameters
index
The index of the group you want to expand.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `collapseGroupAtIndex:` (page 34)
- `isGroupExpandedAtIndex:` (page 37)

Declared In
`IKImageBrowserView.h`

indexAtLocationOfDroppedItem

Returns the index of the cell where the drop operation occurred.

```
- (NSUInteger) indexAtLocationOfDroppedItem;
```

Return Value
The index of the cell where the drop operation occurred.

Discussion
The returned index is valid until the next drop occurs.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

indexOfItemAtPoint:

Returns the index of the item at the specified location.

```
- (NSInteger) indexOfItemAtPoint: (NSPoint)point;
```

Parameters

point

The location of the item.

Return Value

The index of the item or `NSNotFound` if no item at this location.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKImageBrowserView.h`

initWithFrame:

Initializes a newly allocated image browser view with the provided frame rectangle.

```
- (id) initWithFrame:(NSRect) frame;
```

Parameters

frame

The rectangle for the image browser.

Return Value

The initialized object.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKImageBrowserView.h`

isGroupExpandedAtIndex:

Returns whether the group at the provided index is expanded.

```
- (BOOL) isGroupExpandedAtIndex:(NSUInteger) index;
```

Parameters

index

The index you want to check.

Return Value

`YES` if the group is expanded; `NO` otherwise.

Availability

Available in Mac OS X v10.5 and later.

See Also
- `expandGroupAtIndex:` (page 36)
- `collapseGroupAtIndex:` (page 34)

Declared In
`IKImageBrowserView.h`

itemFrameAtIndex:

Returns the frame rectangle for the item located at the specified index.

```
- (NSRect) itemFrameAtIndex: (NSInteger)index;
```

Parameters
index
The index of the item whose frame rectangle you want to obtain.

Return Value
The frame rectangle of the item.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

reloadData

Marks the receiver as needing its data reloaded.

```
- (void) reloadData;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

scrollIndexToVisible:

Scrolls the receiver to the item at the specified index.

```
- (void) scrollIndexToVisible:(NSInteger) index;
```

Parameters
index
The index of the item to scroll to.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

selectionIndexes

Returns the indexes of the selected cells.

```
- (NSIndexSet *) selectionIndexes;
```

Return Value
The indexes of the selected cells.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setSelectionIndexes:byExtendingSelection:` (page 43)

Declared In
`IKImageBrowserView.h`

setAllowsEmptySelection:

Controls whether an empty selection is allowed.

```
- (void) setAllowsEmptySelection: (BOOL) flag;
```

Parameters

flag

A `BOOL` value that specifies whether to allow an empty selection.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- allowsEmptySelection` (page 32)

Declared In
`IKImageBrowserView.h`

setAllowsMultipleSelection:

Controls whether the user can select more than one cell at a time.

```
- (void) setAllowsMultipleSelection: (BOOL) flag;
```

Parameters

flag

A `BOOL` value that specifies whether to allow multiple selections.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- allowsMultipleSelection` (page 32)

Declared In
IKImageBrowserView.h

setAllowsReordering:

Controls whether the user can reorder items.

```
- (void) setAllowsReordering: (BOOL) flag;
```

Parameters

flag

A BOOL value that specifies whether the user can reorder items.

Availability
Available in Mac OS X v10.5 and later.

See Also
- allowsReordering (page 32)

Declared In
IKImageBrowserView.h

setAnimates:

Controls whether the receiver animates reordering and changes of the data source.

```
- (void) setAnimates: (BOOL) flag;
```

Parameters

flag

A BOOL value that specifies whether the receiver animates reordering and changes of the data source.

Availability
Available in Mac OS X v10.5 and later.

See Also
- animates (page 33)

Declared In
IKImageBrowserView.h

setCellSize:

Sets the cell size.

```
- (void) setCellSize:(NSSize) size;
```

Parameters

size

The size to set.

Availability
Available in Mac OS X v10.5 and later.

Declared In
IKImageBrowserView.h

setCellsStyleMask:

Defines the appearance style of the cells.

```
- (void)  setCellsStyleMask:(NSUInteger) mask;
```

Parameters

mask

An integer bit mask. A mask can be specified by combining any of the options described in "Cell Appearance Style Masks" (page 44) using the C bitwise OR operator.

Availability
Available in Mac OS X v10.5 and later.

See Also
- cellsStyleMask (page 33)

Declared In
IKImageBrowserView.h

setConstrainsToOriginalSize:

Sets whether the receiver constrains the cell's image to its original size.

```
- (void) setConstrainsToOriginalSize: (BOOL) flag;
```

Parameters

flag

A flag that specifies whether to constrain the image. The default value is NO.

Availability
Available in Mac OS X v10.5 and later.

See Also
- constrainsToOriginalSize (page 34)

Declared In
IKImageBrowserView.h

setContentResizingMask:

Determines how the receiver resizes its content when zooming.

```
- (void) setContentResizingMask:(NSUInteger) mask;
```

Parameters

mask

A resizing mask. You specify a mask by combining any of the following options using the C bitwise OR operator: NSViewWidthSizable, NSViewHeightSizable. Other values are ignored.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `contentResizingMask` (page 34)

Declared In
`IKImageBrowserView.h`

setDataSource:

Sets the data source of the receiver.

```
- (void) setDataSource:(id) source;
```

Parameters

source
A data source (`IKImageBrowserDataSource`).

Availability
Available in Mac OS X v10.5 and later.

See Also
- `dataSource` (page 35)

Declared In
`IKImageBrowserView.h`

setDelegate:

Sets the delegate of the receiver.

```
- (void) setDelegate: (id) aDelegate;
```

Parameters

aDelegate
The delegate must implement the `IKImageBrowserDelegate` informal protocol.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `delegate` (page 35)

Declared In
`IKImageBrowserView.h`

setDraggingDestinationDelegate:

Sets the dragging destination delegate of the receiver.

```
- (void) setDraggingDestinationDelegate:(id) delegate;
```

Parameters

delegate

The delegate (`NSDraggingDestination`) to set.

Availability

Available in Mac OS X v10.5 and later.

See Also

`- draggingDestinationDelegate` (page 35)

Declared In

`IKImageBrowserView.h`

setSelectionIndexes:byExtendingSelection:

Selects cells at the specified indexes.

```
- (void) setSelectionIndexes:(NSIndexSet *) indexes byExtendingSelection:(BOOL)
    extendSelection;
```

Parameters

indexes

The indexes of the cells you want to select.

extendSelection

A `BOOL` value that specifies whether to extend the current selection. Pass `YES` to extends the selection; `NO` replaces the current selection.

Availability

Available in Mac OS X v10.5 and later.

See Also

`- selectionIndexes` (page 39)

Related Sample Code

ImageKitDemo

Declared In

`IKImageBrowserView.h`

setZoomValue:

Sets the zoom value.

```
- (void) setZoomValue:(float)aValue;
```

Parameters

aValue

The zoom value. This value should be greater or equal to zero and less or equal than one. A zoom value of zero corresponds to the minimum size (40x40 pixels). A zoom value of one means images fits the browser bounds. Other values are interpolated.

Availability

Available in Mac OS X v10.5 and later.

See Also
- zoomValue (page 44)

Declared In
IKImageBrowserView.h

zoomValue

Returns the current zoom value.

```
- (float) zoomValue;
```

Return Value
The zoom value.

Availability
Available in Mac OS X v10.5 and later.

See Also
- setZoomValue: (page 43)

Declared In
IKImageBrowserView.h

Constants

Cell Appearance Style Masks

Masks for the appearance style bit field.

```
enum{
    IKCellsStyleNone              =0,
    IKCellsStyleShadowed          =1,
    IKCellsStyleOutlined          =2,
    IKCellsStyleTitled            =4,
    IKCellsStyleSubtitled         =8
};
```

Constants

IKCellsStyleNone
No style.
Available in Mac OS X v10.5 and later.
Declared in IKImageBrowserView.h.

IKCellsStyleShadowed
Cells use shadows.
Available in Mac OS X v10.5 and later.
Declared in IKImageBrowserView.h.

`IKCellsStyleOutlined`
Cells are outlined.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKCellsStyleTitled`
Cells display a title.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKCellsStyleSubtitled`
Cells display a subtitle.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

Declared In
`IKImageBrowserView.h`

Group Style Attributes

Attributes for the group style.

```
enum{
    IKGroupBezelStyle,
    IKGroupDisclosureStyle,
};
```

Constants
`IKGroupBezelStyle`
A bezel style.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKGroupDisclosureStyle`
A disclosure triangle.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

Discussion
These constants affect the appearance of a group.

Declared In
`IKImageBrowserView.h`

View Options Keys

Keys for image browser view options.

```
NSString * const IKImageBrowserBackgroundColorKey;
NSString * const IKImageBrowserSelectionColorKey;
NSString * const IKImageBrowserCellsOutlineColorKey;
NSString * const IKImageBrowserCellsTitleAttributesKey;
NSString * const IKImageBrowserCellsHighlightedTitleAttributesKey;
NSString * const IKImageBrowserCellsSubtitleAttributesKey;
```

Constants

`IKImageBrowserBackgroundColorKey`
A key for the background color of the image browser view. The associated value is an `NSColor` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserSelectionColorKey`
A key for the color that indicates a selection. The associated value is an `NSColor` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserCellsOutlineColorKey`
A key for the outline color for an item in the image browser view. The associated value is an `NSColor` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserCellsTitleAttributesKey`
A key for title attribute of an item in the image browser view. The associated value is an `NSDictionary` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserCellsHighlightedTitleAttributesKey`
A key for the highlighted title attribute for an item in the image browser view. The associated value is an `NSDictionary` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserCellsSubtitleAttributesKey`
A key for a subtitle attribute for an item in the image browser view. The associated value is an `NSDictionary` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

Discussion
You can set and retrieve values for these keys using the methods `setValue:forKey` and `valueForKey:`.

Declared In
`IKImageBrowserView.h`

Group Keys

Keys for group attributes.

```
NSString * const IKImageBrowserGroupRangeKey;
NSString * const IKImageBrowserGroupBackgroundColorKey;
NSString * const IKImageBrowserGroupTitleKey;
NSString * const IKImageBrowserGroupStyleKey;
```

Constants

`IKImageBrowserGroupRangeKey`

A key for the range of a group. The associated value is an `NSValue` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserGroupBackgroundColorKey`

A key for the background color of a group. The associated value is an `NSColor` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserGroupTitleKey`

A key for the title of a group. The associated value is an `NSString` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserGroupStyleKey`

A key for the style of a group. The associated value is one of the constants defined in "Group Style Attributes" (page 45).

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

Declared In

`IKImageBrowserView.h`

IKImageEditPanel Class Reference

Inherits from	NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKImageEditPanel.h

Overview

The `IKImageEditPanel` class provides a panel, that is, a utility window that floats on top of document windows, optimized for image editing.

Tasks

Creating an Image Editing Panel

`+ sharedImageEditPanel` (page 50)
Creates a shared instance of an image editing panel.

Getting the User Adjustments and Effects

`filterArray` (page 50) *property*
Returns the current array of user adjustments to effects. (read-only)

Getting, Setting, and Reloading Data

`dataSource` (page 50) *property*
Specifies the edit panel's dataSource.

`- reloadData` (page 51)
Reloads the data from the data associated with an image editing panel.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

dataSource

Specifies the edit panel's dataSource.

```
@property(assign) id<IKImageEditPanelDataSource> dataSource
```

Availability
Available in Mac OS X v10.6 and later.

Declared In
`IKImageEditPanel.h`

filterArray

Returns the current array of user adjustments to effects. (read-only)

```
@property(readonly) NSArray *filterArray
```

Availability
Available in Mac OS X v10.6 and later.

Declared In
`IKImageEditPanel.h`

Class Methods

sharedImageEditPanel

Creates a shared instance of an image editing panel.

```
+ (IKImageEditPanel *)sharedImageEditPanel
```

Return Value
An `IKImageEditPanel` object.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageEditPanel.h`

Instance Methods

reloadData

Reloads the data from the data associated with an image editing panel.

```
- (void)reloadData
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageEditPanel.h`

IKImageView Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKImageView.h
Related sample code	ImageKitDemo

Overview

The `IKImageView` class provides an efficient way to display images in a view while at the same time supporting a number of image editing operations such as rotating, zooming, and cropping. It supports drag and drop, so that the user can drag an image to the view. If possible, image rendering uses hardware acceleration to achieve optimal performance. The `IKImageView` class is implemented as a subclass of `NSView`. Similar to `NSImageView`, the `IKImageView` class is used to display a single image.

You can provide an images for the view in any of these formats:

- File reference (`NSURL`, `CFURLRef`, or a path)
- `CGImageSourceRef`
- Data (`NSData` or `CFDataRef`)
- Image (`NSImage`, `CGImageRef`, or `CIImage`)

Providing a file reference is the preferred way to set the the image for a view because in addition to the actual image data, `IKImageView` also handles the image metadata embedded in the file. The image view automatically fetches the metadata from a file reference, whereas for the other sources (except for a `CGImageSourceRef` source), it cannot. For images set from other sources, you need to set the metadata separately.

`IKImageView` supports multi-frame images (TIFF, GIF, and so forth) and animated images.

Tasks

Getting and Setting Image View Characteristics

`delegate` (page 57) *property*
Specifies the delegate object of the receiver.

`zoomFactor` (page 59) *property*
Specifies the zoom factor for the image view.

`rotationAngle` (page 58) *property*
Specifies the rotation angle for the image view.

`currentToolMode` (page 56) *property*
Specifies the current tool mode for the image view.

`autoresizes` (page 56) *property*
Specifies the automatic resizing state for the image view.

`hasHorizontalScroller` (page 57) *property*
Specifies the horizontal scroll bar state for the image view.

`hasVerticalScroller` (page 58) *property*
Specifies the vertical scroll bar state for the image view.

`autohidesScrollers` (page 56) *property*
Specifies the automatic-hiding scroll bar state for the image view.

`supportsDragAndDrop` (page 58) *property*
Specifies the drag-and-drop support state for the image view.

`editable` (page 57) *property*
Specifies the editable state for the image view.

`doubleClickOpensImageEditPanel` (page 57) *property*
Specifies the image-opening state of the editing pane in the image view.

`imageCorrection` (page 58) *property*
Specifies a Core Image filter for image correction.

`backgroundColor` (page 56) *property*
Specifies the background color for the image view.

- `imageSize` (page 62)
Returns the size of the image in the image view.

- `imageProperties` (page 62)
Returns the metadata for the image in the view.

Getting and Setting Images

- `image` (page 61)
Returns the image associated with the view, after any image corrections.

- `setImage:imageProperties:` (page 64)
Sets the image to display in an image view.

- `setImageWithURL:` (page 64)
  Initializes an image view with the image specified by a URL.

Manipulating the Image in a View

- `setRotationAngle:centerPoint:` (page 65)
  Sets the rotation angle at the provided origin.
- `setImageZoomFactor:centerPoint:` (page 64)
  Sets the zoom factor at the provided origin.
- `zoomImageToFit:` (page 66)
  Zooms the image so that it fits in the image view.
- `zoomImageToActualSize:` (page 66)
  Zooms the image so that it is displayed using its true size.
- `zoomImageToRect:` (page 66)
  Zooms the image so that it fits in the specified rectangle.
- `flipImageHorizontal:` (page 61)
  Flips an image along the horizontal axis.
- `flipImageVertical:` (page 61)
  Flips an image along the vertical axis.

Working With Core Animation

- `setOverlay:forType:` (page 65)
  Sets an overlay type for a Core Animation layer.
- `overlayForType:` (page 62)
  Returns the Core Animation layer associated with a layer type.

Scrolling

- `scrollToPoint:` (page 63)
  Scrolls the view to the specified point.
- `scrollToRect:` (page 63)
  Scrolls the view so that it includes the provided rectangular area.

Converting Points and Rectangles

- `convertViewPointToImagePoint:` (page 60)
  Converts an image view coordinate to an image coordinate.
- `convertViewRectToImageRect:` (page 60)
  Converts an image view rectangle to an image rectangle.
- `convertImagePointToViewPoint:` (page 59)
  Converts an image coordinate to an image view coordinate.

- convertImageRectToViewRect: (page 59)
  Converts an image rectangle to an image view rectangle.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

autohidesScrollers

Specifies the automatic-hiding scroll bar state for the image view.

```
@property BOOL autohidesScrollers;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
IKImageView.h

autoresizes

Specifies the automatic resizing state for the image view.

```
@property BOOL autoresizes;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
IKImageView.h

backgroundColor

Specifies the background color for the image view.

```
@property NSColor *  backgroundColor;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
IKImageView.h

currentToolMode

Specifies the current tool mode for the image view.

```
@property NSString*  currentToolMode;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

delegate

Specifies the delegate object of the receiver.

```
@property id delegate;
```

Discussion
An IKImageView object's delegate is inserted in the responder chain after the image view itself and is informed of various actions by the image view through delegation messages.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

doubleClickOpensImageEditPanel

Specifies the image-opening state of the editing pane in the image view.

```
@property BOOL doubleClickOpensImageEditPanel;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

editable

Specifies the editable state for the image view.

```
@property BOOL editable;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

hasHorizontalScroller

Specifies the horizontal scroll bar state for the image view.

```
@property BOOL hasHorizontalScroller;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

hasVerticalScroller

Specifies the vertical scroll bar state for the image view.

```
@property BOOL hasVerticalScroller;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

imageCorrection

Specifies a Core Image filter for image correction.

```
@property CIFilter * imageCorrection;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

rotationAngle

Specifies the rotation angle for the image view.

```
@property CGFloat rotationAngle;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

supportsDragAndDrop

Specifies the drag-and-drop support state for the image view.

```
@property BOOL supportsDragAndDrop;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

zoomFactor

Specifies the zoom factor for the image view.

```
@property CGFloat zoomFactor;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

Instance Methods

convertImagePointToViewPoint:

Converts an image coordinate to an image view coordinate.

```
- (NSPoint)convertImagePointToViewPoint: (NSPoint)imagePoint;
```

Parameters

imagePoint
A point specified in coordinates relative to the image.

Return Value
A point specified in coordinates relative to the image view.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- convertViewPointToImagePoint:` (page 60)

Declared In
`IKImageView.h`

convertImageRectToViewRect:

Converts an image rectangle to an image view rectangle.

```
- (NSRect)convertImageRectToViewRect: (NSRect)imageRect;
```

Parameters

imageRect

An rectangle specified in coordinates relative to the image.

Return Value

An rectangle specified in coordinates relative to the image view.

Availability

Available in Mac OS X v10.5 and later.

See Also

- convertViewRectToImageRect: (page 60)

Declared In

IKImageView.h

convertViewPointToImagePoint:

Converts an image view coordinate to an image coordinate.

```
- (NSPoint)convertViewPointToImagePoint: (NSPoint)viewPoint;
```

Parameters

viewPoint

A point specified in coordinates relative to the image view.

Return Value

The point specified in coordinates relative to the image.

Availability

Available in Mac OS X v10.5 and later.

See Also

- convertImagePointToViewPoint: (page 59)

Declared In

IKImageView.h

convertViewRectToImageRect:

Converts an image view rectangle to an image rectangle.

```
- (NSRect)convertViewRectToImageRect: (NSRect)viewRect;
```

Parameters

viewRect

An rectangle specified in coordinates relative to the image view.

Return Value

The rectangle specified in coordinates relative to the image.

Availability

Available in Mac OS X v10.5 and later.

See Also
- convertImageRectToViewRect: (page 59)

Declared In
IKImageView.h

flipImageHorizontal:

Flips an image along the horizontal axis.

```
- (void)flipImageHorizontal: (id)sender;
```

Parameters

sender
The object initiating the action.

Availability
Available in Mac OS X v10.5 and later.

See Also
- flipImageVertical: (page 61)

Declared In
IKImageView.h

flipImageVertical:

Flips an image along the vertical axis.

```
- (void)flipImageVertical: (id)sender;
```

Parameters

sender
The object initiating the action.

Availability
Available in Mac OS X v10.5 and later.

See Also
- flipImageHorizontal: (page 61)

Declared In
IKImageView.h

image

Returns the image associated with the view, after any image corrections.

```
- (CGImageRef)image;
```

Return Value
The image.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setImage:imageProperties:` (page 64)
- `setImageWithURL:` (page 64)

Declared In
`IKImageView.h`

imageProperties

Returns the metadata for the image in the view.

```
- (NSDictionary*)imageProperties;
```

Return Value
A dictionary of metadata that specifies the image properties.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

imageSize

Returns the size of the image in the image view.

```
- (NSSize)imageSize;
```

Return Value
The size of the image.

Discussion
The image size changes whenever an image is rotates or cropped.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageView.h`

overlayForType:

Returns the Core Animation layer associated with a layer type.

```
- (CALayer*)overlayForType: (NSString*)layerType;
```

Parameters
layerType
A layer type. See "Overlay Types" (page 68).

Return Value
The Core Animation layer.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setOverlay:forType:` (page 65)

Declared In
`IKImageView.h`

scrollToPoint:

Scrolls the view to the specified point.

```
- (void)scrollToPoint:(NSPoint)point;
```

Parameters

point
The point to scroll to.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `scrollToRect:` (page 63)

Declared In
`IKImageView.h`

scrollToRect:

Scrolls the view so that it includes the provided rectangular area.

```
- (void)scrollToRect:(NSRect)rect;
```

Parameters

rect
The rectangular area to include in the view.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `scrollToPoint:` (page 63)

Declared In
`IKImageView.h`

setImage:imageProperties:

Sets the image to display in an image view.

```
- (void)setImage: (CGImageRef)image  imageProperties: (NSDictionary*)metaData;
```

Parameters

image

The image to set.

metaData

A dictionary that contains metadata that describes the image.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `image` (page 61)
- `imageProperties` (page 62)
- `setImageWithURL:` (page 64)

Declared In
`IKImageView.h`

setImageWithURL:

Initializes an image view with the image specified by a URL.

```
- (void)setImageWithURL: (NSURL*)url;
```

Parameters

url

The URL that specifies the location of the image.

Discussion
This method is the preferred initializer for RAW images. If you use this method for a TIFF file that contains multiple images, only the first image is displayed.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setImage:imageProperties:` (page 64)

Declared In
`IKImageView.h`

setImageZoomFactor:centerPoint:

Sets the zoom factor at the provided origin.

```
- (void)setImageZoomFactor: (CGFloat)zoomFactor centerPoint: (NSPoint)centerPoint;
```

Parameters

zoomFactor

The zoom factor to apply to the image.

centerPoint

The point that specifies the origin of the zoom factor.

Availability
Available in Mac OS X v10.5 and later.

See Also
`@property zoomFactor` (page 59)

Declared In
`IKImageView.h`

setOverlay:forType:

Sets an overlay type for a Core Animation layer.

```
- (void)setOverlay: (CALayer*)layer forType: (NSString*)layerType;
```

Parameters

layer

A Core Animation layer object.

layerType

A layer type. See "Overlay Types" (page 68).

Availability
Available in Mac OS X v10.5 and later.

See Also
`- overlayForType:` (page 62)

Declared In
`IKImageView.h`

setRotationAngle:centerPoint:

Sets the rotation angle at the provided origin.

```
- (void)setRotationAngle: (CGFloat)rotationAngle  centerPoint: (NSPoint)centerPoint;
```

Parameters

rotationAngle

The rotation angle to apply to the image.

centerPoint

The point that specifies the origin of the rotation angle.

Availability
Available in Mac OS X v10.5 and later.

See Also
`@property rotationAngle` (page 58)

Declared In
`IKImageView.h`

zoomImageToActualSize:

Zooms the image so that it is displayed using its true size.

```
- (void)zoomImageToActualSize: (id)sender;
```

Parameters

sender
The object initiating the action.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- zoomImageToFit:` (page 66)
`- zoomImageToRect:` (page 66)

Declared In
`IKImageView.h`

zoomImageToFit:

Zooms the image so that it fits in the image view.

```
- (void)zoomImageToFit: (id)sender;
```

Parameters

sender
The object initiating the action.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- zoomImageToActualSize:` (page 66)
`- zoomImageToRect:` (page 66)

Declared In
`IKImageView.h`

zoomImageToRect:

Zooms the image so that it fits in the specified rectangle.

```
- (void)zoomImageToRect: (NSRect)rect;
```

Parameters

rect

The rectangle to fit the image in.

Availability

Available in Mac OS X v10.5 and later.

See Also

- `zoomImageToFit:` (page 66)
- `zoomImageToActualSize:` (page 66)

Declared In

`IKImageView.h`

Constants

Tool Modes

Image Kit tools modes.

```
NSString *const IKToolModeMove;
NSString *const IKToolModeSelect;
NSString *const IKToolModeCrop;
NSString *const IKToolModeRotate;
NSString *const IKToolModeAnnotate;
```

Constants

`IKToolModeMove`

The move tool.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

`IKToolModeSelect`

The selection tool.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

`IKToolModeCrop`

The crop tool.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

`IKToolModeRotate`

The rotation tool.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

`IKToolModeAnnotate`

The annotation tool.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

Declared In
`IKImageView.h`

Overlay Types

A layer level.

```
NSString *const IKOverlayTypeBackground;
NSString *const IKOverlayTypeImage;
```

Constants

`IKOverlayTypeBackground`

A background.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

`IKOverlayTypeImage`

An image.

Available in Mac OS X v10.5 and later.

Declared in `IKImageView.h`.

Declared In
`IKImageView.h`

IKPictureTaker Class Reference

Inherits from	NSPanel : NSWindow : NSResponder : NSObject
Conforms to	NSUserInterfaceValidations (NSWindow) NSAnimatablePropertyContainer (NSWindow) NSCoding (NSResponder) NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKPictureTaker.h
Related sample code	ImageKitDemo

Overview

The `IKPictureTaker` class represents a panel that allows users to choose images by browsing the file system. The picture taker panel provides an Open Recent menu, supports image cropping, and supports taking snapshots from an iSight or other digital camera.

Tasks

Getting and Setting Images

`- setInputImage:` (page 74)
Set the image input for the picture taker.

`- inputImage` (page 72)
Returns the input image associated with the picture taker.

`- outputImage` (page 72)
Returns the edited image.

Managing the Picture Taker

`+ pictureTaker` (page 70)
Returns a shared `IKPictureTaker` instance, creating it if necessary.

- `beginPictureTakerSheetForWindow:withDelegate:didEndSelector:contextInfo:` (page 70)
  Opens a picture taker as a sheet whose parent is the specified window.
- `beginPictureTakerWithDelegate:didEndSelector:contextInfo:` (page 71)
  Opens a picture taker pane.
- `popUpRecentsMenuForView:withDelegate:didEndSelector:contextInfo:` (page 73)
  Displays the Open Recent popup menu associated with the picture taker.
- `runModal` (page 73)
  Opens a modal picture taker dialog.

Getting and Setting Mirroring

- `setMirroring:` (page 74)
  Controls whether the receiver enables video mirroring during snapshots.
- `mirroring` (page 72)
  Returns whether video mirroring is enabled during snapshots.

Class Methods

pictureTaker

Returns a shared `IKPictureTaker` instance, creating it if necessary.

```
+ (IKPictureTaker *) pictureTaker;
```

Return Value
An `IKPictureTaker` object.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
ImageKitDemo

Declared In
`IKPictureTaker.h`

Instance Methods

beginPictureTakerSheetForWindow:withDelegate:didEndSelector:contextInfo:

Opens a picture taker as a sheet whose parent is the specified window.

```
- (void) beginPictureTakerSheetForWindow:(NSWindow *)aWindow withDelegate:(id)
    delegate didEndSelector:(SEL) didEndSelector contextInfo:(void *) contextInfo;
```

Parameters

aWindow

The parent window of the picture taker sheet.

delegate

The object that will invoke the selector `didEndSelector` when the picture taker session terminates.

didEndSelector

The selector to invoke when the picture taker session terminates.

contextInfo

Any data that must be passed as an argument to the delegate through `didEndSelector` after the picture taker session terminates.

Discussion

The `didEndSelector` method should have the following signature:

```
- (void)pictureTakerDidEnd:(IKPictureTaker *)sheet returnCode:(NSInteger)returnCode
contextInfo:(void *)contextInfo;
```

The `returnCode` value is set to `NSOKButton` if the user validates, or to `NSCancelButton` if the user cancels.

Availability

Available in Mac OS X v10.5 and later.

See Also

`- beginPictureTakerWithDelegate:didEndSelector:contextInfo:` (page 71)

Related Sample Code

ImageKitDemo

Declared In

`IKPictureTaker.h`

beginPictureTakerWithDelegate:didEndSelector:contextInfo:

Opens a picture taker pane.

```
- (void) beginPictureTakerWithDelegate:(id) delegate didEndSelector:(SEL)
    didEndSelector contextInfo:(void *) contextInfo;
```

Parameters

delegate

The object that will invoke the selector `didEndSelector` when the picture taker session terminates.

didEndSelector

The selector to invoke when the picture taker session terminates.

contextInfo

Any data that must be passed as an argument to the delegate through `didEndSelector` after the picture taker session terminates.

Discussion

The `didEndSelector` method should have the following signature:

```
- (void)pictureTakerDidEnd:(IKPictureTaker *)sheet returnCode:(NSInteger)returnCode
contextInfo:(void *)contextInfo;
```

The `returnCode` value is set to `NSOKButton` if the user validates, or to `NSCancelButton` if the user cancels.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- beginPictureTakerSheetForWindow:withDelegate:didEndSelector:contextInfo:` (page 70)

Declared In
`IKPictureTaker.h`

inputImage

Returns the input image associated with the picture taker.

```
- (NSImage*) inputImage;
```

Return Value
The input image.

Discussion
The input image is never modified by the picture taker.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setInputImage:` (page 74)

Declared In
`IKPictureTaker.h`

mirroring

Returns whether video mirroring is enabled during snapshots.

```
- (BOOL) mirroring;
```

Return Value
Returns YES if video mirroring is enabled, NO otherwise.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKPictureTaker.h`

outputImage

Returns the edited image.

```
- (NSImage*) outputImage;
```

Return Value
The edited image.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
ImageKitDemo
ImagePicker

Declared In
`IKPictureTaker.h`

popUpRecentsMenuForView:withDelegate:didEndSelector:contextInfo:

Displays the Open Recent popup menu associated with the picture taker.

```
- (void) popUpRecentsMenuForView:(NSView *) aView withDelegate:(id) delegate
    didEndSelector:(SEL) didEndSelector contextInfo:(void *) contextInfo;
```

Parameters

`delegate`
The object that will invoke the selector `didEndSelector` when the picture taker session terminates.

`didEndSelector`
The selector to invoke when the picture taker session terminates.

`contextInfo`
Any data that must be passed as an argument to the delegate through `didEndSelector` after the picture taker session terminates.

Discussion
The `didEndSelector` method should have the following signature:

```
- (void)pictureTakerDidEnd:(IKPictureTaker *)sheet returnCode:(NSInteger)returnCode
contextInfo:(void *)contextInfo;
```

The `returnCode` value is set to `NSOKButton` if the user validates, or to `NSCancelButton` if the user cancels.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKPictureTaker.h`

runModal

Opens a modal picture taker dialog.

```
- (NSInteger) runModal;
```

Return Value
Returns `NSOKButton` if the user edits or chooses an image; `NSCancelButton` if the user cancels or does not change the default image.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKPictureTaker.h`

setInputImage:

Set the image input for the picture taker.

```
- (void) setInputImage:(NSImage *) image;
```

Parameters

image

An `NSImage` object.

Discussion
The input image is never modified by the picture taker.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- inputImage` (page 72)

Related Sample Code
ImagePicker

Declared In
`IKPictureTaker.h`

setMirroring:

Controls whether the receiver enables video mirroring during snapshots.

```
- (void) setMirroring:(BOOL)b;
```

Parameters

b

The default setting is `YES`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKPictureTaker.h`

Constants

Picture Taker Keys

Keys for customizing the picture taker appearance and behavior.

```
NSString *const IKPictureTakerAllowsVideoCaptureKey;
NSString *const IKPictureTakerAllowsFileChoosingKey;
NSString *const IKPictureTakerShowRecentPictureKey;
NSString *const IKPictureTakerUpdateRecentPictureKey;
NSString *const IKPictureTakerAllowsEditingKey;
NSString *const IKPictureTakerShowEffectsKey;
NSString *const IKPictureTakerInformationalTextKey;
NSString *const IKPictureTakerImageTransformsKey;
NSString *const IKPictureTakerOutputImageMaxSizeKey;
NSString *const IKPictureTakerCropAreaSizeKey;
NSString *const IKPictureTakerShowAddressBookPictureKey;
NSString *const IKPictureTakerShowEmptyPictureKey;
```

Constants

`IKPictureTakerAllowsVideoCaptureKey`

A key for allowing video capture. The associated value is an `NSNumber` value (`BOOL`) whose default value is `YES`.

Available in Mac OS X v10.5 and later.

Declared in `IKPictureTaker.h`.

`IKPictureTakerAllowsFileChoosingKey`

A key for allowing the user to choose a file. The associated value is an `NSNumber` object that contains a `BOOL` value whose default value is `YES`.

Available in Mac OS X v10.5 and later.

Declared in `IKPictureTaker.h`.

`IKPictureTakerUpdateRecentPictureKey`

A key for allowing a recent picture to be updated. The associated value is an `NSNumber` object that contains a `BOOL` value whose default value is `YES`.

Available in Mac OS X v10.5 and later.

Declared in `IKPictureTaker.h`.

`IKPictureTakerAllowsEditingKey`

A key for allowing image editing. The associated value is an `NSNumber` object that contains a `BOOL` value whose default value is `YES`.

Available in Mac OS X v10.5 and later.

Declared in `IKPictureTaker.h`.

`IKPictureTakerShowEffectsKey`

A key for showing effects. The associated value is an `NSNumber` object that contains a `BOOL` value whose default value is `NO`.

Available in Mac OS X v10.5 and later.

Declared in `IKPictureTaker.h`.

`IKPictureTakerInformationalTextKey`
: A key for informational text. The associated value is an `NSString` or `NSAttributedString` object whose default value is `"Drag Image Here"`.

  Available in Mac OS X v10.5 and later.

  Declared in `IKPictureTaker.h`.

`IKPictureTakerImageTransformsKey`
: A n image transformation key. The associated value is an `NSDictionary` object that can be serialized.

  Available in Mac OS X v10.5 and later.

  Declared in `IKPictureTaker.h`.

`IKPictureTakerOutputImageMaxSizeKey`
: A key for the maximum size of the output image. The associated value is an `NSValue` object (`NSSize`).

  Available in Mac OS X v10.5 and later.

  Declared in `IKPictureTaker.h`.

`IKPictureTakerCropAreaSizeKey`
: A key for the cropping area size. The associated value is an `NSValue` object (`NSSize`).

  Available in Mac OS X v10.5 and later.

  Declared in `IKPictureTaker.h`.

`IKPictureTakerShowAddressBookPictureKey`
: A key for showing the address book picture. The associated value is a Boolean value packages as an `NSNumber` object. The default value is `NO`. If set to YES, the picture taker automatically adds the address book image for the Me user at the end of the Recent Pictures pop-up menu.

  Available in Mac OS X v10.5 and later.

  Declared in `IKPictureTaker.h`.

`IKPictureTakerShowEmptyPictureKey`
: A key for showing an empty picture. The associated value is an `NSImage` object. The default value is `nil`. If set to an image, the picture taker automatically shows an image at the end of the Recent Pictures pop-up menu. that means "no picture."

  Available in Mac OS X v10.5 and later.

  Declared in `IKPictureTaker.h`.

Discussion
You can set picture taker options using `setValue:forKey` (`NSKeyValueCoding`).

Declared In
`IKPictureTaker.h`

IKSaveOptions Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKSaveOptions.h
Related sample code	CIRAWFilterSample DispatchFractal ImageKitDemo

Overview

The `IKSaveOptions` class initializes, adds, and manages user interface options for saving image data.

Tasks

Initializing and Adding Options

`- initWithImageProperties:imageUTType:` (page 80)
Initializes a save options accessory pane for the provided image properties and uniform type identifier.

Retrieving User Settings

`imageProperties` (page 78) *property*
Returns a dictionary of updated image properties that reflects the user's selection. (read-only)

`imageUTType` (page 78) *property*
Returns the uniform type identifier that reflects the user's selection. (read-only)

`userSelection` (page 79) *property*
Returns a dictionary that contains the save options selected by the user. (read-only)

Filtering Save Options

`- addSaveOptionsToView:` (page 79)
Adds the save options interface to the specified view.

`- addSaveOptionsAccessoryViewToSavePanel:` (page 79)
Adds IKSaveOptions UI to a NSSavePanel.

Filtering Save Types

`delegate` (page 78) *property*
Returns the delegate object. (read-only)

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

delegate

Returns the delegate object. (read-only)

```
@property(readonly) id delegate
```

Availability
Available in Mac OS X v10.6 and later.

Declared In
`IKSaveOptions.h`

imageProperties

Returns a dictionary of updated image properties that reflects the user's selection. (read-only)

```
@property(readonly) NSDictionary *imageProperties
```

Availability
Available in Mac OS X v10.6 and later.

Related Sample Code
DispatchFractal

Declared In
`IKSaveOptions.h`

imageUTType

Returns the uniform type identifier that reflects the user's selection. (read-only)

```
@property(readonly) NSString *imageUTType
```

Availability
Available in Mac OS X v10.6 and later.

Related Sample Code
DispatchFractal

Declared In
`IKSaveOptions.h`

userSelection

Returns a dictionary that contains the save options selected by the user. (read-only)

```
@property(readonly) NSDictionary *userSelection
```

Availability
Available in Mac OS X v10.6 and later.

Declared In
`IKSaveOptions.h`

Instance Methods

addSaveOptionsAccessoryViewToSavePanel:

Adds IKSaveOptions UI to a NSSavePanel.

```
- (void)addSaveOptionsAccessoryViewToSavePanel:(NSSavePanel *)savePanel
```

Parameters
savePanel
The save panel to add the IKSaveOptions to.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
DispatchFractal

Declared In
`IKSaveOptions.h`

addSaveOptionsToView:

Adds the save options interface to the specified view.

```
- (void)addSaveOptionsToView:(NSView *)view
```

Parameters

view

The view that the interface is added to.

Availability
Available in Mac OS X v10.6 and later.

Declared In
IKSaveOptions.h

initWithImageProperties:imageUTType:

Initializes a save options accessory pane for the provided image properties and uniform type identifier.

```
- (id)initWithImageProperties:(NSDictionary *)imageProperties
    imageUTType:(NSString *)imageUTType
```

Parameters

imageProperties

A dictionary of image properties.

imageUTType

A string that specifies a uniform type identifier, such as JPEG. See *Uniform Type Identifiers Overview*.

Return Value
The initialized object.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CIRAWFilterSample
DispatchFractal
ImageKitDemo

Declared In
IKSaveOptions.h

IKSlideshow Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKSlideshow.h
Related sample code	ImageKitDemo

Overview

The `IKSlideshow` class encapsulates a data source and options for a slideshow.

Tasks

Creating a Shared Instance of a Slideshow

`+ sharedSlideshow` (page 83)
Returns a shared instance of a slideshow.

Running and Stopping a Slideshow

`- runSlideshowWithDataSource:inMode:options:` (page 85)
Runs a slideshow that contains the specified kind of items, provided from a data source.

`- stopSlideshow:` (page 85)
Stops a slideshow.

`autoPlayDelay` (page 82) *property*
Controls the interval of time before a slideshow starts to play automatically.

Getting Slideshow Data

`- indexOfCurrentSlideshowItem` (page 84)
Returns the index of the current slideshow item.

Reloading Data

`- reloadData` (page 84)
Reloads the data for a slideshow.

`- reloadSlideshowItemAtIndex:` (page 84)
Reloads the data for a slideshow, starting at the specified index.

Exporting Slideshow Items

`+ canExportToApplication:` (page 82)
Finds out whether the slideshow can export its contents to an application.

`+ exportSlideshowItem:toApplication:` (page 83)
Exports a slideshow item to the application that has the provided bundle identifier.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

autoPlayDelay

Controls the interval of time before a slideshow starts to play automatically.

```
@property NSTimeInterval autoPlayDelay
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKSlideshow.h`

Class Methods

canExportToApplication:

Finds out whether the slideshow can export its contents to an application.

```
+ (BOOL)canExportToApplication:(NSString *)applicationBundleIdentifier
```

Parameters

applicationBundleIdentifier

The bundle identifier of the application that you want to export the slideshow to. See "Bundle Identifiers" (page 86).

Return Value

YES if the slideshow can be exported to the specified application; NO otherwise.

Availability

Available in Mac OS X v10.5 and later.

See Also

+ exportSlideshowItem:toApplication: (page 83)

Declared In

IKSlideshow.h

exportSlideshowItem:toApplication:

Exports a slideshow item to the application that has the provided bundle identifier.

```
+ (void)exportSlideshowItem:(id)item toApplication:(NSString
    *)applicationBundleIdentifier
```

Parameters

item

The item to export

applicationBundleIdentifier

The bundle identifier of the application that you want to export the item to.

Availability

Available in Mac OS X v10.5 and later.

See Also

+ canExportToApplication: (page 82)

Declared In

IKSlideshow.h

sharedSlideshow

Returns a shared instance of a slideshow.

```
+ (IKSlideshow *)sharedSlideshow
```

Return Value

A slideshow object.

Availability

Available in Mac OS X v10.5 and later.

Related Sample Code

ImageKitDemo

Declared In
IKSlideshow.h

Instance Methods

indexOfCurrentSlideshowItem

Returns the index of the current slideshow item.

```
- (NSUInteger)indexOfCurrentSlideshowItem
```

Return Value
The index of the current item in the slideshow.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
ImageKitDemo

Declared In
IKSlideshow.h

reloadData

Reloads the data for a slideshow.

```
- (void)reloadData
```

Availability
Available in Mac OS X v10.5 and later.

See Also
- reloadSlideshowItemAtIndex: (page 84)

Declared In
IKSlideshow.h

reloadSlideshowItemAtIndex:

Reloads the data for a slideshow, starting at the specified index.

```
- (void)reloadSlideshowItemAtIndex:(NSUInteger)index
```

Parameters
index
The index that species where to reload the slideshow data.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `reloadData` (page 84)

Declared In
`IKSlideshow.h`

runSlideshowWithDataSource:inMode:options:

Runs a slideshow that contains the specified kind of items, provided from a data source.

```
- (void)runSlideshowWithDataSource:(id < IKSlideshowDataSource >)dataSource
    inMode:(NSString *)slideshowMode options:(NSDictionary *)slideshowOptions
```

Parameters

dataSource
The data source to use for the slideshow.

slideshowMode
A constant that indicate what kind of items are in the slideshow—`IKSlideshowModeImages`, `IKSlideshowModePDF`, or `IKSlideshowModeQuickLook`. See "Slideshow Modes" (page 86).

slideshowOptions
A dictionary of slideshow options. See "Slideshow Option Keys" (page 87).

Availability
Available in Mac OS X v10.5 and later.

See Also
- `stopSlideshow:` (page 85)

Related Sample Code
ImageKitDemo

Declared In
`IKSlideshow.h`

stopSlideshow:

Stops a slideshow.

```
- (void)stopSlideshow:(id)sender
```

Parameters

sender
The object sending the message to stop the slideshow.

Discussion
This method is invoked when the user clicks a button or issues a stop command.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `runSlideshowWithDataSource:inMode:options:` (page 85)

Declared In
`IKSlideshow.h`

Constants

Bundle Identifiers

Identifiers for exporting slideshow items to an application.

```
NSString *const IK_iPhotoBundleIdentifier;
NSString *const IK_ApertureBundleIdentifier;
NSString *const IK_MailBundleIdentifier;
```

Constants

`IK_iPhotoBundleIdentifier`
The iPhoto application—`com.apple.iPhoto`.
Available in Mac OS X v10.5 and later.
Declared in `IKSlideshow.h`.

`IK_ApertureBundleIdentifier`
The Aperature application—`com.apple.Aperture`.
Available in Mac OS X v10.6 and later.
Declared in `IKSlideshow.h`.

`IK_MailBundleIdentifier`
The Mail application—`com.apple.mail`.
Available in Mac OS X v10.6 and later.
Declared in `IKSlideshow.h`.

Slideshow Modes

The kind of items in the slideshow.

```
NSString *const IKSlideshowModeImages;
NSString *const IKSlideshowModePDF;
NSString *const IKSlideshowModeOther;
```

Constants

`IKSlideshowModeImages`
All items in the slideshow are images.
Available in Mac OS X v10.5 and later.
Declared in `IKSlideshow.h`.

`IKSlideshowModePDF`
All items in the slideshow are PDF documents.
Available in Mac OS X v10.5 and later.
Declared in `IKSlideshow.h`.

`IKSlideshowModeOther`
There are a mixture of items in the slideshow (image, PDF, text, HTML, and so on).

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

Slideshow Option Keys

Keys for slideshow options.

```
NSString *const IKSlideshowWrapAround;
NSString *const IKSlideshowStartPaused;
NSString *const IKSlideshowStartIndex;
NSString *const IKSlideshowPDFDisplayBox;
NSString *const IKSlideshowPDFDisplayMode;
NSString *const IKSlideshowPDFDisplaysAsBook;
NSString *const IKSlideshowScreen;
NSString *const IKSlideshowAudioFile;
```

Constants

`IKSlideshowWrapAround`
A key for starting the slideshow over after the last slide shows. The associated value is a `Boolean` data type.

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowStartPaused`
A key for starting in a paused state. The associated value is a `Boolean` data type.

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowStartIndex`
A key for the slideshow item index. The associated value is an index.

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowPDFDisplayBox`
A key for the PDF display box. The associated value is a type of display box, such as `kPDFDisplayBoxMediaBox` or `kPDFDisplayBoxMediaBox`. See *PDFPage Class Reference* for more information.

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowPDFDisplayMode`
A key for the PDF display mode. The associated value is a PDF display mode constant, such as `kPDFDisplaySinglePage` or `kPDFDisplayTwoUp`. See *PDFView Class Reference* for more information.

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowPDFDisplaysAsBook`
A key for displaying the slideshow as a book. The associated value is a `Boolean` data type.

Available in Mac OS X v10.5 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowScreen`

A key specifying the screen on which the slideshow is displayed. The associated value is an `NSScreen` object. By default `mainScreen` is used.

Available in Mac OS X v10.6 and later.

Declared in `IKSlideshow.h`.

`IKSlideshowAudioFile`

A key specifying the audio file played during the slideshow. The associated value is an `NSURL` object.

Available in Mac OS X v10.6 and later.

Declared in `IKSlideshow.h`.

Protocols

IKFilterCustomUIProvider Protocol Reference

Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKFilterUI.h

Overview

The `IKFilterCustomUIProvider` protocol is an addition to the `CIFilter` class that defines a method for providing a view for a filter. This protocol is implemented by any filter that provides its own user interface.

Tasks

Providing a Custom View

`- provideViewForUIConfiguration:excludedKeys:` (page 91) *required method*
Provides a custom view for a filter. (required)

Instance Methods

provideViewForUIConfiguration:excludedKeys:

Provides a custom view for a filter. (required)

```
-(IKFilterUIView*)provideViewForUIConfiguration:(NSDictionary*)inUIConfiguration
    excludedKeys:(NSArray*)inKeys
```

Parameters

inUIConfiguration
A dictionary that specifies the size of the controls. Provide the key `IKUISizeFlavor` and one of the following values: `IKUISizeMini`, `IKUISizeSmall`, or `IKUISizeRegular`. For more information on these constants, see *User Interface Options* in *CIFilter Image Kit Additions*.

inKeys
An array of the input keys for which you do *not* want to provide a user interface. Pass `nil` if you want all input keys to be represented in the user interface.

Return Value
An `IKFilterUIView` object or `nil` if the filter is unable to provide a view. If `nil`, the Image Kit framework will attempt to provide a user interface.

Discussion
This method overrides the method `viewForUIConfiguration:excludedKeys:` (page 12).

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKFilterUI.h`

IKImageBrowserDataSource Protocol Reference

(informal protocol)

Adopted by	IKImageBrowserView
Framework	/System/Library/Frameworks/Quartz.framework/ImageKit.framework
Declared in	ImageKit/IKImageBrowserView.h

Overview

The `IKImageBrowserDataSource` informal protocol declares the methods that an instance of the `IKImageBrowserView` class uses to access the contents of its data source object.

Tasks

Providing Information About Items (Required)

- `numberOfItemsInImageBrowser:` (page 97) *required method*
  Returns the number of records managed by the data source object. (required)
- `imageBrowser:itemAtIndex:` (page 94) *required method*
  Returns an object for the item in an image browser view that corresponds to the specified index. (required)

Supporting Item Editing (Optional)

- `imageBrowser:removeItemsAtIndexes:` (page 95) *required method*
  Signals that a remove operation should be applied to the specified items. (required)
- `imageBrowser:moveItemsAtIndexes:toIndex:` (page 95) *required method*
  Signals that the specified items should be moved to the specified destination. (required)
- `imageBrowser:writeItemsAtIndexes:toPasteboard:` (page 96) *required method*
  Signals that a drag should begin. (required)

Providing Information About Groups (Optional)

- `numberOfGroupsInImageBrowser:` (page 96) *required method*
  Returns the number of groups in an image browser view. (required)

Instance Methods

imageBrowser:groupAtIndex:

Returns the group at the specified index. (required)

```
- (NSDictionary *) imageBrowser:(IKImageBrowserView *) aBrowser
    groupAtIndex:(NSUInteger) index;
```

Parameters

aBrowser

An image browser view.

index

The index of the group you want to retrieve.

Return Value

A dictionary that defines the group. The keys in this dictionary can be any of the following constants: `IKImageBrowserGroupStyle`, `IKImageBrowserGroupBackgroundColorKey`, `IKImageBrowserGroupTitleKey`, and `IKImageBrowserGroupRangeKey`. For more information on these constants, see *IKImageBrowserView Class Reference*.

Discussion

This method is optional.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKImageBrowserView.h`

imageBrowser:itemAtIndex:

Returns an object for the item in an image browser view that corresponds to the specified index. (required)

```
- (id) imageBrowser:(IKImageBrowserView *) aBrowser itemAtIndex:(NSUInteger)index;
```

Parameters

aBrowser

An image browser view.

index

The index of the item you want to retrieve.

Return Value

An `IKImageBrowserItem` object.

Discussion
Your data source must implement this method. The returned object must implement the required methods of the `IKImageBrowserItem` protocol.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

imageBrowser:moveItemsAtIndexes:toIndex:

Signals that the specified items should be moved to the specified destination. (required)

```
- (BOOL) imageBrowser:(IKImageBrowserView *) aBrowser moveItemsAtIndexes: (NSIndexSet
    *)indexes toIndex:(NSUInteger)destinationIndex;
```

Parameters

aBrowser
An image browser view.

indexes
The indexes of the items that should be reordered.

destinationIndex
The starting index of the destination the items should be moved to.

Return Value
`YES` if successful; `NO` otherwise.

Discussion
This method is optional. It is invoked by the image browser view after Image Kit determines that a reordering operation should be applied. The data source should update itself by reordering its elements.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setAllowsReordering:` (page 40)

Declared In
`IKImageBrowserView.h`

imageBrowser:removeItemsAtIndexes:

Signals that a remove operation should be applied to the specified items. (required)

```
- (void) imageBrowser:(IKImageBrowserView *) aBrowser
    removeItemsAtIndexes:(NSIndexSet *) indexes;
```

Parameters

aBrowser
An image browser view.

indexes
The indexes of the items that should be removed.

Discussion
This method is optional. It is invoked by the image browser after Image Kit determines that a remove operation should be applied. In response, the data source should update itself by removing the specified items.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

imageBrowser:writeItemsAtIndexes:toPasteboard:

Signals that a drag should begin. (required)

```
- (NSUInteger) imageBrowser:(IKImageBrowserView *) aBrowser
    writeItemsAtIndexes:(NSIndexSet *) itemIndexes toPasteboard:(NSPasteboard
    *)pasteboard;
```

Parameters
aBrowser
An image browser view.
itemIndexes
The indexes of the items that should be dragged.
pasteboard
The pasteboard to copy the items to.

Return Value
The number of items written to the pasteboard.

Discussion
This method is optional. It is invoked after Image Kit determines that a drag should begin, but before the drag has been started.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

numberOfGroupsInImageBrowser:

Returns the number of groups in an image browser view. (required)

```
- (NSUInteger) numberOfGroupsInImageBrowser:(IKImageBrowserView *) aBrowser;
```

Parameters
aBrowser
An image browser view.

Return Value
The number of groups.

Discussion
This method is optional.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

numberOfItemsInImageBrowser:

Returns the number of records managed by the data source object. (required)

```
- (NSUInteger) numberOfItemsInImageBrowser:(IKImageBrowserView *) aBrowser;
```

Parameters

aBrowser
An image browser view.

Return Value
The number of records managed by the image browser view.

Discussion
Your data source must implement this method. An `IKImageView` object uses this method to determine how many cells it should create and display.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

IKImageBrowserDelegate Protocol Reference

(informal protocol)

Adopted by	IKImageBrowserView
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Declared in	ImageKit/IKImageBrowserView.h

Overview

The `IKImageBrowserDelegate` is an informal protocol for the delegate of an `IKImageBrowserView` object. You can implement these methods to perform custom tasks when in response to events in the image browser view.

Tasks

Performing Custom Tasks in Response to User Events

- `imageBrowser:backgroundWasRightClickedWithEvent:` (page 99) *required method*
  Performs custom tasks when the user right-clicks the image browser view background. (required)
- `imageBrowser:cellWasRightClickedAtIndex:withEvent:` (page 100) *required method*
  Performs custom tasks when the user right-clicks an item in the image browser view. (required)
- `imageBrowser:cellWasDoubleClickedAtIndex:` (page 100) *required method*
  Performs custom tasks when the user double-clicks an item in the image browser view. (required)
- `imageBrowserSelectionDidChange:` (page 101) *required method*
  Performs custom tasks when the selection changes. (required)

Instance Methods

imageBrowser:backgroundWasRightClickedWithEvent:

Performs custom tasks when the user right-clicks the image browser view background. (required)

```
- (void) imageBrowser:(IKImageBrowserView *) aBrowser
    backgroundWasRightClickedWithEvent:(NSEvent *) event;
```

Parameters

aBrowser

An image browser view.

event

The event that invoked the method.

Discussion

This method signals that the user either right-clicked the background or left-clicked it with the Alt key pressed. You can implement this method if you want to perform custom tasks at that time.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKImageBrowserView.h`

imageBrowser:cellWasDoubleClickedAtIndex:

Performs custom tasks when the user double-clicks an item in the image browser view. (required)

```
- (void) imageBrowser:(IKImageBrowserView *) aBrowser
    cellWasDoubleClickedAtIndex:(NSUInteger) index;
```

Parameters

aBrowser

An image browser view.

index

The index of the cell.

Discussion

This method signals that the user double-clicked an item in the image browser view. You can implement this method if you want to perform custom tasks at that time.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`IKImageBrowserView.h`

imageBrowser:cellWasRightClickedAtIndex:withEvent:

Performs custom tasks when the user right-clicks an item in the image browser view. (required)

```
- (void) imageBrowser:(IKImageBrowserView *) aBrowser
    cellWasRightClickedAtIndex:(NSUInteger) index withEvent:(NSEvent *) event;
```

Parameters

aBrowser

An image browser view.

index

The index of the cell.

event
The event that invoked the method.

Discussion
This method signals that the user either right-clicked an item in the browser or left-clicked the item with the Alt key pressed. You can implement this method if you want to perform custom tasks at that time.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

imageBrowserSelectionDidChange:

Performs custom tasks when the selection changes. (required)

```
- (void) imageBrowserSelectionDidChange:(IKImageBrowserView *) aBrowser;
```

Parameters
aBrowser
An image browser view.

Discussion
This method signals that the user changes the selection in the image browser view. You can implement this method if you want to perform custom tasks at that time.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

IKImageBrowserItem Protocol Reference

(informal protocol)

Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Declared in	ImageKit/IKImageBrowserView.h

Overview

The `IKImageBrowserItem` informal protocol declares the methods that an instance of the `IKImageBrowserView` class uses to access the contents of its data source for a given item. Some of the methods in this protocol are needed frequently, so you should implement them efficiently.

Tasks

Providing Required Information for an Image

- `imageUID` (page 105) *required method*
  Returns a unique string that identifies the data source item. (required)
- `imageRepresentationType` (page 104) *required method*
  Returns the representation type of the image to display. (required)
- `imageRepresentation` (page 104) *required method*
  Returns the image to display. (required)

Providing Optional Information for an Image

- `imageVersion` (page 106) *required method*
  Returns the version of the item. (required)
- `imageTitle` (page 105) *required method*
  Returns the display title of the image. (required)
- `imageSubtitle` (page 105) *required method*
  Returns the display subtitle of the image. (required)
- `isSelectable` (page 106) *required method*
  Returns whether this item is selectable. (required)

Instance Methods

imageRepresentation

Returns the image to display. (required)

```
- (id) imageRepresentation;
```

Return Value
The image to display; can return `nil` if the item has no image to display.

Discussion
Your data source must implement this method. This method is called frequently, so the receiver should cache the returned instance.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CocoaCreateMovie
DesktopImage
ImageBrowser
ImageBrowserViewAppearance
ImageKitDemo

Declared In
`IKImageBrowserView.h`

imageRepresentationType

Returns the representation type of the image to display. (required)

```
- (NSString *) imageRepresentationType;
```

Return Value
A string that specifies the image representation type. The string can be any of the constants defined in "Image Representation Types" (page 107).

Discussion
Your data source must implement this method.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
DesktopImage
Image Kit with Core Data
ImageBrowser
ImageBrowserViewAppearance
ImageKitDemo

Declared In
`IKImageBrowserView.h`

imageSubtitle

Returns the display subtitle of the image. (required)

```
- (NSString *) imageSubtitle
```

Return Value
The display subtitle of the image.

Discussion
This method is optional.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
ImageBrowserViewAppearance

Declared In
`IKImageBrowserView.h`

imageTitle

Returns the display title of the image. (required)

```
- (NSString *) imageTitle;
```

Return Value
The display title of the image.

Discussion
This method is optional.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
DesktopImage
ImageBrowserViewAppearance
ImageKitDemo

Declared In
`IKImageBrowserView.h`

imageUID

Returns a unique string that identifies the data source item. (required)

```
- (NSString *)  imageUID;
```

Return Value
The string that identifies the data source item

Discussion
Your data source must implement this method. The image browser view uses this identifier to associate the data source item and its cache.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
DesktopImage
Image Kit with Core Data
ImageBrowser
ImageBrowserViewAppearance
ImageKitDemo

Declared In
`IKImageBrowserView.h`

imageVersion

Returns the version of the item. (required)

```
- (NSUInteger) imageVersion;
```

Return Value
The version of the item.

Discussion
This method is optional. The receiver can return a new version to let the image browser knows that it should not use its cache for the item.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
Image Kit with Core Data

Declared In
`IKImageBrowserView.h`

isSelectable

Returns whether this item is selectable. (required)

```
- (BOOL) isSelectable;
```

Return Value
`YES` if the item is selectable; `NO` otherwise.

Discussion
This method is optional. You can prevent selection of this item by returning `NO`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`IKImageBrowserView.h`

Constants

Image Representation Types

Representation types for images.

```
NSString * const IKImageBrowserPathRepresentationType;
NSString * const IKImageBrowserNSURLRepresentationType;
NSString * const IKImageBrowserNSImageRepresentationType;
NSString * const IKImageBrowserCGImageRepresentationType;
NSString * const IKImageBrowserCGImageSourceRepresentationType;
NSString * const IKImageBrowserNSDataRepresentationType;
NSString * const IKImageBrowserNSBitmapImageRepresentationType;
NSString * const IKImageBrowserQTMovieRepresentationType;
NSString * const IKImageBrowserQTMoviePathRepresentationType;
NSString * const IKImageBrowserQCCompositionRepresentationType;
NSString * const IKImageBrowserQCCompositionPathRepresentationType;
NSString * const IKImageBrowserQuickLookPathRepresentationType;
NSString * const IKImageBrowserIconRefPathRepresentationType;
NSString * const IKImageBrowserIconRefRepresentationType;
```

Constants

`IKImageBrowserPathRepresentationType`
A path representation (`NSString`).
Available in Mac OS X v10.5 and later.
Declared in `IKImageBrowserView.h`.

`IKImageBrowserNSURLRepresentationType`
An `NSURL`object.
Available in Mac OS X v10.5 and later.
Declared in `IKImageBrowserView.h`.

`IKImageBrowserNSImageRepresentationType`
An `NSImage` object.
Available in Mac OS X v10.5 and later.
Declared in `IKImageBrowserView.h`.

`IKImageBrowserCGImageRepresentationType`
A `CGImageRef` object.
Available in Mac OS X v10.5 and later.
Declared in `IKImageBrowserView.h`.

`IKImageBrowserCGImageSourceRepresentationType`
A `CGImageSourceRef` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserNSDataRepresentationType`
An `NSData` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserNSBitmapImageRepresentationType`
An `NSBitmapImageRep` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserQTMovieRepresentationType`
A `QTMovie` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserQTMoviePathRepresentationType`
A path (NSString) or URL (NSURL) to a QuickTime movie.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserQCCompositionRepresentationType`
A `QCComposition` object.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserQCCompositionPathRepresentationType`
A path (NSString) or URL (NSURL) to a Quartz Composer composition.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserQuickLookPathRepresentationType`
A path (NSString) or URL (NSURL) to load data using QuickLook.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserIconRefPathRepresentationType`
A path to an icon.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

`IKImageBrowserIconRefRepresentationType`
An icon.

Available in Mac OS X v10.5 and later.

Declared in `IKImageBrowserView.h`.

Declared In
`IKImageBrowserView.h`

IKImageEditPanelDataSource Protocol Reference

Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKImageEditPanel.h
Companion guide	Image Kit Programming Guide

Overview

The `IKImageEditPanelDataSource` protocol describes the methods that an `IKImageEditPanel` object uses to access the contents of its data source object.

Tasks

Getting and Setting Image Properties

- `imageProperties` (page 111)
  Returns a dictionary of the image properties associated with the image in the image edit panel.
- `setImage:imageProperties:` (page 111) *required method*
  Sets an image with the specified properties. (required)

Getting Images From the Data Source

- `image` (page 111) *required method*
  Returns an image. (required)
- `thumbnailWithMaximumSize:` (page 112)
  Returns a thumbnail image whose size is no larger than the specified size.

New Methods

- `hasAdjustMode` (page 110)
  Returns whether the adjust mode view tab should be displayed.

- `hasDetailsMode` (page 110)
  Returns whether the details mode view tab should be displayed.
- `hasEffectsMode` (page 110)
  Returns whether the effects mode view tab should be displayed.

Instance Methods

hasAdjustMode

Returns whether the adjust mode view tab should be displayed.

```
- (BOOL)hasAdjustMode
```

Return Value
`YES` if the tab should be displayed, otherwise NO.

Availability
Available in Mac OS X v10.6 and later.

Declared In
`IKImageEditPanel.h`

hasDetailsMode

Returns whether the details mode view tab should be displayed.

```
- (BOOL)hasDetailsMode
```

Return Value
`YES` if the tab should be displayed, otherwise NO.

Availability
Available in Mac OS X v10.6 and later.

Declared In
`IKImageEditPanel.h`

hasEffectsMode

Returns whether the effects mode view tab should be displayed.

```
- (BOOL)hasEffectsMode
```

Return Value
`YES` if the tab should be displayed, otherwise NO.

Availability
Available in Mac OS X v10.6 and later.

Declared In
IKImageEditPanel.h

image

Returns an image. (required)

```
- (CGImageRef)image
```

Return Value
An image.

Discussion
Your data source must implement this method.

Availability
Available in Mac OS X v 10.5 and later.

Declared In
IKImageEditPanel.h

imageProperties

Returns a dictionary of the image properties associated with the image in the image edit panel.

```
- (NSDictionary *)imageProperties
```

Return Value
A dictionary that contains the properties of the image.

Availability
Available in Mac OS X v 10.5 and later.

See Also
- setImage:imageProperties (page ?)

Declared In
IKImageEditPanel.h

setImage:imageProperties:

Sets an image with the specified properties. (required)

```
- (void)setImage:(CGImageRef)image imageProperties:(NSDictionary *)metaData
```

Discussion
Your data source must implement this method.

Availability
Available in Mac OS X v 10.5 and later.

See Also
- imageProperties (page ?)

Declared In
`IKImageEditPanel.h`

thumbnailWithMaximumSize:

Returns a thumbnail image whose size is no larger than the specified size.

```
- (CGImageRef)thumbnailWithMaximumSize:(NSSize)size
```

Return Value
An image.

Availability
Available in Mac OS X v 10.5 and later.

Declared In
`IKImageEditPanel.h`

IKSlideshowDataSource Protocol Reference

Adopted by	IKSlideshow
Framework	System/Library/Frameworks/Quartz.framework/ImageKit.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	ImageKit/IKSlideShow.h

Overview

The `IKSlideshowDataSource` protocol describes the methods that an `IKSlideshow` object uses to access the contents of its data source object.

Important: Slide show data source methods may be called on secondary threads. When you implement these methods, you must ensure that they are safe to run on threads other than the main thread.

Tasks

Providing Slideshow Information

- `numberOfSlideshowItems` (page 114) *required method*
  Returns the number of items in a slideshow. (required)
- `slideshowItemAtIndex:` (page 116) *required method*
  Returns the item for a given index (required)
- `nameOfSlideshowItemAtIndex:` (page 114)
  Returns the display name for item at the specified index.
- `canExportSlideshowItemAtIndex:toApplication:` (page 114)
  Reports whether the export button should be enabled for a a slideshow item.

Performing Custom Tasks

- `slideshowWillStart` (page 116)
  Performs custom tasks when the slideshow is about to start.

- `slideshowDidStop` (page 115)
  Performs custom tasks when the slideshow stops.
- `slideshowDidChangeCurrentIndex:` (page 115)
  Performs custom tasks when the slideshow changes to the item at the specified index.

Instance Methods

canExportSlideshowItemAtIndex:toApplication:

Reports whether the export button should be enabled for a a slideshow item.

```
- (BOOL)canExportSlideshowItemAtIndex:(NSUInteger)index toApplication:(NSString
    *)applicationBundleIdentifier
```

Return Value
`YES` if the export button should be enabled for an item; otherwise `NO`.

Availability
Available in Mac OS X v 10.5 and later.

Declared In
`IKSlideshow.h`

nameOfSlideshowItemAtIndex:

Returns the display name for item at the specified index.

```
- (NSString *)nameOfSlideshowItemAtIndex:(NSUInteger)index
```

Parameters
index
The index for a slideshow item.

Return Value
The display name. For the best user experience, you should provide the localized name, because this string appears in the user interface.

Discussion
This method is optional.

Availability
Available in Mac OS X v 10.5 and later.

Declared In
`IKSlideshow.h`

numberOfSlideshowItems

Returns the number of items in a slideshow. (required)

```
- (NSUInteger)numberOfSlideshowItems
```

Return Value
The number of items in the slideshow.

Discussion
Your data source must implement this method.

Availability
Available in Mac OS X v 10.5 and later.

Declared In
`IKSlideshow.h`

slideshowDidChangeCurrentIndex:

Performs custom tasks when the slideshow changes to the item at the specified index.

```
- (void)slideshowDidChangeCurrentIndex:(NSUInteger)newIndex
```

Parameters
newIndex
The index of the current item.

Discussion
Image Kit invokes this method when the slideshow changes to the specified item. Implement this method to perform custom tasks at that time.

Availability
Available in Mac OS X v 10.5 and later.

Declared In
`IKSlideshow.h`

slideshowDidStop

Performs custom tasks when the slideshow stops.

```
- (void)slideshowDidStop
```

Discussion
TImage Kit invokes this method when the slideshow stops. Implement this method to perform custom tasks at that time.

Availability
Available in Mac OS X v 10.5 and later.

See Also
`- slideshowWillStart` (page ?)

Declared In
`IKSlideshow.h`

slideshowItemAtIndex:

Returns the item for a given index (required)

```
- (id)slideshowItemAtIndex:(NSUInteger)index
```

Parameters

index

An index of an item in the slideshow.

Return Value

The object that corresponds to the item at the specified index. The item can be any of the following objects: `NSImage`, `NSString` (to specify a path name), `NSURL`, `NSFileWrapper`, `CGImageRef`, or `PDFPage`.

Discussion

Your data source must implement this method.

Availability

Available in Mac OS X v 10.5 and later.

Declared In

`IKSlideshow.h`

slideshowWillStart

Performs custom tasks when the slideshow is about to start.

```
- (void)slideshowWillStart
```

Discussion

Image Kit invokes this method when the slideshow is about to start. Implement this method to perform custom tasks at that time.

Availability

Available in Mac OS X v 10.5 and later.

See Also

`- slideshowDidStop` (page ?)

Declared In

`IKSlideshow.h`

Document Revision History

This table describes the changes to *Image Kit Reference Collection*.

Date	Notes
2006-12-06	New collection that describes the Objective-C API for providing a user interface for images, image editing, and image processing.

Document Revision History

Index

G

H

I

M

N

O

P

R

S

T

U

V

Z