
WebKit Objective-C Framework Reference

Networking, Internet, & Web: Web Client



2009-07-28



Apple Inc.
© 2009 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

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

Apple, the Apple logo, Carbon, Cocoa, iChat, Logic, Mac, Mac OS, Objective-C, Pages, and Safari are trademarks of Apple Inc., registered in the United States and other countries.

Finder, Spotlight, and WebScript are trademarks of Apple Inc.

Java and all Java-based trademarks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO

THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

Introduction **Introduction 9**

Part I **Classes 11**

Chapter 1 **DOMDocument Additions Reference 13**

Overview 13
Tasks 13
Instance Methods 14

Chapter 2 **DOMElement Additions Reference 15**

Overview 15
Tasks 15
Instance Methods 15

Chapter 3 **DOMHTMLDocument Additions Reference 17**

Overview 17
Tasks 17
Instance Methods 17

Chapter 4 **DOMHTMLFrameElement Additions Reference 19**

Overview 19
Tasks 19
Instance Methods 19

Chapter 5 **DOMHTMLIFrameElement Additions Reference 21**

Overview 21
Tasks 21
Instance Methods 21

Chapter 6 **DOMHTMLObjectElement Additions Reference 23**

Overview 23
Tasks 23
Instance Methods 23

Chapter 7 **DOMNode Additions Reference 25**

Overview 25
Tasks 25
Instance Methods 26

Chapter 8 **DOMRange Additions Reference 27**

Overview 27
Tasks 27
Instance Methods 28

Chapter 9 **WebArchive Class Reference 29**

Overview 29
Tasks 29
Instance Methods 30
Constants 32

Chapter 10 **WebBackForwardList Class Reference 33**

Overview 33
Tasks 33
Instance Methods 35

Chapter 11 **WebDataSource Class Reference 43**

Overview 43
Tasks 43
Instance Methods 45

Chapter 12 **WebDownload Class Reference 53**

Overview 53
Delegate Methods 53

Chapter 13 **WebFrame Class Reference 55**

Overview 55
Tasks 56
Instance Methods 57

Chapter 14 **WebFrameView Class Reference 67**

Overview 67
Tasks 67

Instance Methods 68

Chapter 15 [WebHistory Class Reference](#) 73

Overview 73

Tasks 73

Class Methods 74

Instance Methods 75

Constants 80

Notifications 81

Chapter 16 [WebHistoryItem Class Reference](#) 85

Overview 85

Adopted Protocols 85

Tasks 85

Instance Methods 86

Notifications 89

Chapter 17 [WebPreferences Class Reference](#) 91

Overview 91

Adopted Protocols 92

Tasks 92

Class Methods 96

Instance Methods 96

Constants 119

Notifications 119

Chapter 18 [WebResource Class Reference](#) 121

Overview 121

Tasks 121

Instance Methods 122

Chapter 19 [WebScriptObject Class Reference](#) 125

Overview 125

Tasks 126

Class Methods 126

Instance Methods 127

Chapter 20 [WebUndefined Class Reference](#) 131

Overview 131

Tasks 131

Class Methods 131

Chapter 21 [WebView Class Reference](#) 133

Overview 133
Tasks 134
Class Methods 143
Instance Methods 146
Constants 199
Notifications 200

Part II [Protocols](#) 203

Chapter 22 [WebDocumentRepresentation Protocol Reference](#) 205

Overview 205
Tasks 205
Instance Methods 206

Chapter 23 [WebDocumentSearching Protocol Reference](#) 211

Overview 211
Tasks 211
Instance Methods 211

Chapter 24 [WebDocumentText Protocol Reference](#) 213

Overview 213
Tasks 213
Instance Methods 214

Chapter 25 [WebDocumentView Protocol Reference](#) 217

Overview 217
Tasks 217
Instance Methods 218

Chapter 26 [WebEditingDelegate Protocol Reference](#) 221

Overview 221
Tasks 221
Instance Methods 222
Constants 230

Chapter 27 **WebFrameLoadDelegate Protocol Reference** 231

Overview 231
Tasks 231
Instance Methods 232

Chapter 28 **WebJavaPlugIn Protocol Reference** 241

Overview 241
Tasks 241
Instance Methods 241
Constants 243

Chapter 29 **WebOpenPanelResultListener Protocol Reference** 245

Overview 245
Tasks 245
Instance Methods 246

Chapter 30 **WebPlugIn Protocol Reference** 249

Overview 249
Tasks 249
Instance Methods 250

Chapter 31 **WebPlugInContainer Protocol Reference** 255

Overview 255
Tasks 255
Instance Methods 256

Chapter 32 **WebPlugInViewFactory Protocol Reference** 259

Overview 259
Tasks 259
Class Methods 259
Constants 260

Chapter 33 **WebPolicyDecisionListener Protocol Reference** 263

Overview 263
Tasks 263
Instance Methods 264

Chapter 34 [WebPolicyDelegate Protocol Reference](#) 267

[Overview](#) 267
[Tasks](#) 267
[Instance Methods](#) 268
[Constants](#) 271

Chapter 35 [WebResourceLoadDelegate Protocol Reference](#) 275

[Overview](#) 275
[Tasks](#) 275
[Instance Methods](#) 276

Chapter 36 [WebScripting Protocol Reference](#) 283

[Overview](#) 283
[Tasks](#) 284
[Class Methods](#) 284
[Instance Methods](#) 286

Chapter 37 [WebUIDelegate Protocol Reference](#) 289

[Overview](#) 289
[Tasks](#) 289
[Instance Methods](#) 292
[Constants](#) 315

Part III [Constants](#) 321

Chapter 38 [WebKit Constants Reference](#) 323

[Overview](#) 323
[Constants](#) 323

[Document Revision History](#) 327

Introduction

Framework	/System/Library/Frameworks/WebKit.framework
Header file directories	/System/Library/Frameworks/WebKit.framework/Headers
Declared in	DOMExtensions.h WebArchive.h WebBackForwardList.h WebDOMOperations.h WebDataSource.h WebDocument.h WebDownload.h WebEditingDelegate.h WebFrame.h WebFrameLoadDelegate.h WebFrameView.h WebHistory.h WebHistoryItem.h WebJavaPlugIn.h WebKitErrors.h WebPlugin.h WebPluginContainer.h WebPluginViewFactory.h WebPolicyDelegate.h WebPreferences.h WebResource.h WebResourceLoadDelegate.h WebScriptObject.h WebUIDelegate.h WebView.h

WebKit provides a set of classes to display web content in windows, and implements browser features such as following links when clicked by the user, managing a back-forward list, and managing a history of pages recently visited. WebKit greatly simplifies the complicated process of loading web pages—that is, asynchronously requesting web content from an HTTP server where the response may arrive incrementally, in random order, or partially due to network errors. WebKit also simplifies the process of displaying that content which can contain various MIME types, and compound frame elements each with their own set of scroll bars.

Concurrency Note: The WebKit framework is not thread safe. If you call functions or methods in this framework, you must do so exclusively on the main program thread.

Classes

DOMDocument Additions Reference

Inherits from	DOMNode : DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget (DOMNode) NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDOMOperations.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	WebKit DOM Programming Topics WebKit Objective-C Programming Guide

Overview

Additions to the `DOMDocument` class facilitate communication between the DOM API and WebKit and help convert DOM URL element attributes into web-friendly `NSURL` objects.

Tasks

Getting the Web Frame

- [webFrame](#) (page 14)
Returns the web frame associated with the DOM document.

Constructing URLs

- [URLWithAttributeString:](#) (page 14)
Constructs a URL given an attribute string.

Instance Methods

URLWithAttributeString:

Constructs a URL given an attribute string.

```
- (NSURL *)URLWithAttributeString:(NSString *)string
```

Parameters

string

The HTML attribute string to convert.

Return Value

An `NSURL` object containing an absolute URL derived from the specified attribute string.

Discussion

This method constructs a URL given the string value of an element attribute. Examples include the `href` attribute of a `DOMHTMLAnchorElement` object, or the `src` attribute of a `DOMHTMLImageElement` object. This method only applies to attributes that refer to URLs.

This method is similar to `URLWithString:` in the `NSURL` class, except that `URLWithAttributeString:` handles relative URLs automatically based on the current document's location.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebDOMOperations.h`

webFrame

Returns the web frame associated with the DOM document.

```
- (WebFrame *)webFrame
```

Return Value

The web frame object for the receiver.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebDOMOperations.h`

DOMElement Additions Reference

Inherits from	DOMNode : DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget (DOMNode) NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/DOMExtensions.h
Availability	Available in Mac OS X v10.5 and later.
Companion guides	WebKit DOM Programming Topics WebKit Objective-C Programming Guide

Overview

Additions to the `DOMElement` class to retrieve the image of `DOMHTMLImageElement` objects.

Tasks

Getting Properties

- [image](#) (page 15)
Returns an image associated with the receiver.

Instance Methods

image

Returns an image associated with the receiver.

- (NSImage *)image

Discussion

Returns an `NSImage` for the receiver if it is a `DOMHTMLImageElement` object—a `DOMHTMLObjectElement` object with an image loaded or a `DOMHTMLInputElement` object of type `image`. Returns `nil` if there is an error loading the image or the element does not contain an image.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`DOMExtensions.h`

DOMHTMLDocument Additions Reference

(informal protocol)

Inherits from	DOMDocument : DOMNode : DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget (DOMNode) NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/DOMExtensions.h
Availability	Available in Mac OS X v10.5 and later.
Companion guides	WebKit DOM Programming Topics WebKit Objective-C Programming Guide

Overview

Additions to the `DOMHTMLDocument` class to create document fragments.

Tasks

Creating Document Fragments

- [createDocumentFragmentWithMarkupString:baseURL:](#) (page 17)
Creates a document fragment containing the given HTML markup.
- [createDocumentFragmentWithText:](#) (page 18)
Creates a document fragment containing the given text.

Instance Methods

createDocumentFragmentWithMarkupString:baseURL:

Creates a document fragment containing the given HTML markup.

- (`DOMDocumentFragment *`)`createDocumentFragmentWithMarkupString:(NSString *)markupString baseURL:(NSURL *)baseURL`

Parameters

markupString

The HTML content to parse into a DOM tree.

baseURL

The URI from which the content originated. Used for interpreting relative URLs.

Return Value

A `DOMDocumentFragment` derived from the original markup string.

Discussion

This is a convenience method for the `createDocumentFragment` method in `DOMDocument`. It creates a fragment that has the HTML markup parsed into child nodes of the fragment using the *baseURL* to resolve any relative paths for images or other resources.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`DOMExtensions.h`

createDocumentFragmentWithText:

Creates a document fragment containing the given text.

```
- (DOMDocumentFragment *)createDocumentFragmentWithText:(NSString *)text
```

Parameters

text

The text to convert.

Return Value

A `DOMDocumentFragment` object derived from the source text.

Discussion

This is a convenience method for the `createDocumentFragment` method in `DOMDocument`. This method creates a fragment that contains the supplied plain text.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`DOMExtensions.h`

DOMHTMLFrameElement Additions Reference

Inherits from	DOMHTMLFrameElement : DOMElement : DOMNode : DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget (DOMNode) NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDOMOperations.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	Safari Document Object Model Overview WebKit Objective-C Programming Guide

Overview

Additions to the `DOMHTMLFrameElement` class facilitate communication between the DOM API and WebKit.

Tasks

Getting the Content Frame

- [contentFrame](#) (page 19)
Returns the content frame of the element.

Instance Methods

contentFrame

Returns the content frame of the element.

- (`WebFrame *`)`contentFrame`

Return Value

The element's web frame.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDOMOperations.h

DOMHTMLIFrameElement Additions Reference

Inherits from	DOMHTMLIFrameElement : DOMElement : DOMNode : DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget (DOMNode) NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDOMOperations.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	Safari Document Object Model Overview WebKit Objective-C Programming Guide

Overview

Additions to the `DOMHTMLIFrameElement` class facilitate communication between the DOM API and WebKit.

Tasks

Getting the Content Frame

- [contentFrame](#) (page 21)
Returns the content frame of the element.

Instance Methods

contentFrame

Returns the content frame of the element.

- (`WebFrame *`)contentFrame

Return Value

The element's web frame.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDOMOperations.h

DOMHTMLObjectElement Additions Reference

Inherits from	DOMHTMLObjectElement : DOMElement : DOMNode : DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget (DOMNode) NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDOMOperations.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	Safari Document Object Model Overview WebKit Objective-C Programming Guide

Overview

These additions to the `DOMHTMLObjectElement` class facilitate communication between the DOM API and WebKit.

Tasks

Getting the Content Frame

- [contentFrame](#) (page 23)
Returns the content frame of the element.

Instance Methods

contentFrame

Returns the content frame of the element.

- (`WebFrame *`)contentFrame

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDOMOperations.h

DOMNode Additions Reference

Inherits from	DOMObject : WebScriptObject : NSObject
Conforms to	DOMEventTarget NSObject (NSObject) NSCopying (DOMObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDOMOperations.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	Safari Document Object Model Overview WebKit Objective-C Programming Guide

Overview

Additions to the `DOMNode` class help convert the structured nodes of DOM content into a rich web-viewable form.

Tasks

Creating Archives

- [webArchive](#) (page 26)
Returns a web archive of the content of the node and its children.

Obtaining Layout Rectangles

- [boundingBox](#) (page 26)
Returns a rectangle that bounds the onscreen rendering of the node.
- [lineBoxRects](#) (page 26)
Returns the rectangles that bound each line of text in the node.

Instance Methods

boundingBox

Returns a rectangle that bounds the onscreen rendering of the node.

- (NSRect)boundingBox

Return Value

The rectangle that represents the bounding box of the onscreen rendering of the node.

Availability

Available in Mac OS X v10.5 and later.

Declared In

DOMExtensions.h

lineBoxRects

Returns the rectangles that bound each line of text in the node.

- (NSArray *)lineBoxRects

Return Value

An array of rectangles, in which each rectangle represents the bounding box of a line of text in the node.

Availability

Available in Mac OS X v10.5 and later.

Declared In

DOMExtensions.h

webArchive

Returns a web archive of the content of the node and its children.

- (WebArchive *)webArchive

Return Value

A web archive of the content of the node and its children.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDOMOperations.h

DOMRange Additions Reference

Inherits from	DOMObject : WebScriptObject: NSObject
Conforms to	NSCopying (DOMObject) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDOMOperations.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	WebKit DOM Programming Topics WebKit Objective-C Programming Guide

Overview

Additions to the `DOMRange` class facilitate communication between the DOM API and WebKit and help convert web content into standard markup form.

Tasks

Creating Archives

- [webArchive](#) (page 28)
Returns a web archive of the content in the range.

Formatting Content Ranges

- [markupString](#) (page 28)
Returns a string in markup format corresponding to the content in the range.

Instance Methods

markupString

Returns a string in markup format corresponding to the content in the range.

- (NSString *)markupString

Return Value

An entity-encoded string containing markup for content that falls within the specified DOM range.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDOMOperations.h

webArchive

Returns a web archive of the content in the range.

- (WebArchive *)webArchive

Return Value

A web archive of all content that falls within the specified DOM range.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDOMOperations.h

WebArchive Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebArchive.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

A WebArchive object represents a webpage that can be archived—for example, archived on disk or on the pasteboard. A WebArchive object contains the main resource, as well as the subresources and subframes of the main resource. The main resource can be an entire webpage, a portion of a webpage, or some other kind of data such as an image. Use this class to archive webpages, or place a portion of a webpage on the pasteboard, or to represent rich web content in any application.

Tasks

Initializing

- [initWithMainResource:subresources:subframeArchives:](#) (page 30)
Initializes the receiver with a resource and optional subresources and subframe archives..
- [initWithData:](#) (page 30)
Initializes and returns the receiver, specifying the initial content data.

Getting Attributes

- [mainResource](#) (page 31)
Returns the receiver's main resource.
- [subresources](#) (page 31)
Returns the receiver's subresources, or `nil` if there are none.

- [subframeArchives](#) (page 31)
Returns archives representing the receiver's subresources or `nil` if there are none.
- [data](#) (page 30)
Returns the data representation of the receiver.

Instance Methods

data

Returns the data representation of the receiver.

```
- (NSData *)data
```

Discussion

The data returned can be used to save the web archive to a file, to put it on the pasteboard using the [WebArchivePboardType](#) (page 32) type, or used to initialize another web archive using the [initWithData:](#) (page 30) method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebArchive.h

initWithData:

Initializes and returns the receiver, specifying the initial content data.

```
- (id)initWithData:(NSData *)data
```

Discussion

Use the [data](#) (page 30) method to get the receiver's data.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebArchive.h

initWithMainResource:subresources:subframeArchives:

Initializes the receiver with a resource and optional subresources and subframe archives..

```
- (id)initWithMainResource:(WebResource *)mainResource subresources:(NSArray *)subresources subframeArchives:(NSArray *)subframeArchives
```

Discussion

This method initializes and returns the receiver by setting the main resource to *mainResource*, and setting the subresources and subframe archives if supplied. The *subresources* argument should be an array of WebResource objects or *nil* if none are specified. The *subframeArchives* should be an array of WebArchive objects used by the subframes or *nil* if none are specified.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebArchive.h

mainResource

Returns the receiver's main resource.

- (WebResource *)mainResource

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebArchive.h

subframeArchives

Returns archives representing the receiver's subresources or *nil* if there are none.

- (NSArray *)subframeArchives

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [subresources](#) (page 31)

Declared In

WebArchive.h

subresources

Returns the receiver's subresources, or *nil* if there are none.

- (NSArray *)subresources

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [subframeArchives](#) (page 31)

Declared In

WebArchive.h

Constants

WebArchivePboardType

The pasteboard type for this class.

```
extern NSString *WebArchivePboardType;
```

Constants

WebArchivePboardType

The pasteboard type constant used when adding or accessing a WebArchive on the pasteboard.

Available in Mac OS X v10.3 and later.

Declared in WebArchive.h.

WebBackForwardList Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebBackForwardList.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

A `WebBackForwardList` object maintains a list of visited pages used to go back and forward to the most recent page. A `WebBackForwardList` object maintains only the list data—it does not perform actual page loads (in other words, it does not make any client requests). If you need to perform a page load, see the [loadRequest:](#) (page 62) method in *WebFrame Class Reference* to find out how to do this.

Items are typically inserted in a back-forward list in the order they are visited. A `WebBackForwardList` object also maintains the notion of the current item (which is always at index 0), the preceding item (which is at index -1), and the following item (which is at index 1). The [goBack](#) (page 39) and [goForward](#) (page 39) methods move the current item backward or forward by one. The [goToItem:](#) (page 39) method sets the current item to the specified item. All other methods that return `WebHistoryItem` objects do not change the value of the current item, they just return the requested item or items. You can also limit the number of history items stored in the back-forward list using the [setCapacity:](#) (page 41) method.

`WebBackForwardList` objects also control the number of pages cached. You can turn page caching off by setting the page cache size to 0 using the [pageCacheSize](#) (page 40) method, or limit the number of pages cached by passing a value greater than 0.

Tasks

Adding and Removing Items

- [addItem:](#) (page 35)
Inserts an item into the back-forward list, immediately after the current item.

Moving Backward and Forward

- `goBack` (page 39)
Moves backward one item in the back-forward list.
- `goForward` (page 39)
Moves forward one item in the back-forward list.
- `goToItem:` (page 39)
Makes the specified item in the back-forward list the current item.

Querying the Back-Forward List

- `backItem` (page 35)
Returns the item that precedes the current item in the back-forward list.
- `backListCount` (page 36)
Returns the number of items that precede the current item in the back-forward list.
- `backListWithLimit:` (page 36)
Returns the items that precede the current item in the back-forward list, up to the specified number of items.
- `containsItem:` (page 37)
Returns a Boolean value indicating whether the back-forward list contains the specified item.
- `currentItem` (page 37)
Returns the current item in the back-forward list.
- `itemAtIndex:` (page 40)
Returns the item at the specified index in the back-forward list.
- `forwardItem` (page 37)
Returns the item that follows the current item in the back-forward list.
- `forwardListCount` (page 38)
Returns the number of items that follow the current item in the back-forward list.
- `forwardListWithLimit:` (page 38)
Returns the items that follow the current item in the back-forward list, up to the specified number of items.

Page Caching

- `pageCacheSize` (page 40)
Returns the maximum number of pages that the receiver can cache. (**Deprecated.** Use the `usesPageCache` (page 118) method in `WebPreferences` instead.)
- `setPageCacheSize:` (page 41) **Deprecated in Mac OS X v10.4.11**
Sets the maximum number of pages the receiver can cache. (**Deprecated.** Use the `setUsesPageCache:` (page 116) method in `WebPreferences` instead.)

Setting Attributes

- [capacity](#) (page 36)
Returns the maximum number of items that the back-forward list can contain.
- [setCapacity:](#) (page 41)
Sets the maximum number of items that the back-forward list can contain.

Instance Methods

addItem:

Inserts an item into the back-forward list, immediately after the current item.

```
- (void)addItem:(WebHistoryItem *)item
```

Parameters

item

A web history item that represents a visited webpage. If *item* is `nil`, an `NSInvalidArgumentException` exception is raised.

Discussion

Any items following *item* in the back-forward list are removed. This method also removes items if the capacity of the receiver is exceeded.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebBackForwardList.h

backItem

Returns the item that precedes the current item in the back-forward list.

```
- (WebHistoryItem *)backItem
```

Return Value

The item that precedes the current item in the back-forward list, or `nil` if none precedes it.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [currentItem](#) (page 37)
- [forwardItem](#) (page 37)

Declared In

WebBackForwardList.h

backListCount

Returns the number of items that precede the current item in the back-forward list.

```
- (int)backListCount
```

Return Value

The number of items that precede the current item in the back-forward list.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [forwardListCount](#) (page 38)

Declared In

WebBackForwardList.h

backListWithLimit:

Returns the items that precede the current item in the back-forward list, up to the specified number of items.

```
- (NSArray *)backListWithLimit:(int)limit
```

Parameters

limit

The greatest number of items to return.

Return Value

An array containing (at most) the specified number of items, or `nil` if no items precede the current item.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [forwardListWithLimit:](#) (page 38)

Declared In

WebBackForwardList.h

capacity

Returns the maximum number of items that the back-forward list can contain.

```
- (int)capacity
```

Return Value

The maximum number of items the back-forward list can contain.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setCapacity:](#) (page 41)

Declared In

WebBackForwardList.h

containsItem:

Returns a Boolean value indicating whether the back-forward list contains the specified item.

```
- (BOOL)containsItem:(WebHistoryItem *)item
```

Parameters

item

The item to find in the back-forward list.

Return Value

YES if the specified item is in the back-forward list; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebBackForwardList.h

currentItem

Returns the current item in the back-forward list.

```
- (WebHistoryItem *)currentItem
```

Return Value

The current item, or nil if the back-forward list is empty.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [backItem](#) (page 35)

- [forwardItem](#) (page 37)

Declared In

WebBackForwardList.h

forwardItem

Returns the item that follows the current item in the back-forward list.

```
- (WebHistoryItem *)forwardItem
```

Return Value

The item that follows the current item, or `nil` if none follows it.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [backItem](#) (page 35)
- [currentItem](#) (page 37)

Declared In

WebBackForwardList.h

forwardListCount

Returns the number of items that follow the current item in the back-forward list.

```
- (int)forwardListCount
```

Return Value

The number of items that follow the current item.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [backListCount](#) (page 36)

Declared In

WebBackForwardList.h

forwardListWithLimit:

Returns the items that follow the current item in the back-forward list, up to the specified number of items.

```
- (NSArray *)forwardListWithLimit:(int)limit
```

Parameters

limit

The greatest number of items to return.

Return Value

An array containing (at most) the specified number of items, or `nil` if no items follow the current item.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [backListWithLimit:](#) (page 36)

Declared In

WebBackForwardList.h

goBack

Moves backward one item in the back-forward list.

- (void)goBack

Discussion

This method works by changing the current item to the item that precedes it. This method raises an `NSInternalInconsistencyException` exception if no item precedes the current item.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goForward](#) (page 39)
- [goToItem:](#) (page 39)

Declared In

WebBackForwardList.h

goForward

Moves forward one item in the back-forward list.

- (void)goForward

Discussion

This method works by changing the current item to the item that follows it. This method raises an `NSInternalInconsistencyException` exception if no item follows the current item.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goBack](#) (page 39)
- [goToItem:](#) (page 39)

Declared In

WebBackForwardList.h

goToItem:

Makes the specified item in the back-forward list the current item.

- (void)goToItem:(WebHistoryItem *)item

Parameters*item*

A web history item that represents a visited webpage. If *item* is not in the back-forward list, an `NSInvalidArgumentException` exception is raised.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goBack](#) (page 39)
- [goForward](#) (page 39)

Declared In

WebBackForwardList.h

itemAtIndex:

Returns the item at the specified index in the back-forward list.

```
- (WebHistoryItem *)itemAtIndex:(int)index
```

Parameters*index*

The index of the item to return. The position of the current item is index 0, and the position of any other item is expressed as an offset from index 0. For example, the item preceding the current item is at index -1, and the item following the current item is at index 1.

Return Value

The item at the specified index, or `nil` if *index* exceeds the bounds of the back-forward list (that is, if *index* is greater than the value returned by [forwardListCount](#) (page 38), or less than the negative form of the value returned by [backListCount](#) (page 36)).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebBackForwardList.h

pageCacheSize

Returns the maximum number of pages that the receiver can cache. (Deprecated in Mac OS X v10.4.11. Use the [usesPageCache](#) (page 118) method in `WebPreferences` instead.)

```
- (NSInteger)pageCacheSize
```

Return Value

The maximum number of pages that can be cached.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [setPageCacheSize:](#) (page 41)

Declared In

WebBackForwardList.h

setCapacity:

Sets the maximum number of items that the back-forward list can contain.

- (void)setCapacity:(int)*size*

Parameters

size

The maximum number of items that the list can contain.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [capacity](#) (page 36)

Declared In

WebBackForwardList.h

setPageCacheSize:

Sets the maximum number of pages the receiver can cache. (Deprecated in Mac OS X v10.4.11. Use the [setUsesPageCache:](#) (page 116) method in `WebPreferences` instead.)

- (void)setPageCacheSize:(NSUInteger)*size*

Parameters

size

The maximum number of pages that can be cached.

Discussion

The default page cache size can vary depending on the computer's configuration. Use [pageCacheSize](#) (page 40) to get the current setting.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

Declared In

WebBackForwardList.h

WebDataSource Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDataSource.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide
Related sample code	NewsReader SpecialPictureProtocol

Overview

`WebDataSource` encapsulates the web content to be displayed in a web frame view. A `WebDataSource` object has a representation object, conforming to the `WebDocumentRepresentation` protocol, that holds the data in an appropriate format depending on the MIME type. You can extend WebKit to support new MIME types by implementing your own view and representation classes, and specifying the mapping between them using the `registerViewClass:representationClass:forMIMEType:` (page 145) `WebView` class method.

`WebDataSource` objects have an associated initial request, possibly a modified request, and a response object. Since the data source may be in the process of being loaded, you should check the state of a data source using the `isLoading` (page 47) method before accessing its data. Use the `data` (page 45) method to get the raw data. Use the `representation` (page 48) method to get the actual representation object and query it for more details.

Tasks

Initializing an Instance

- `initWithRequest:` (page 46)
initializes a data source with a URL request.

Querying Page Data and State

- `data` (page 45)
Returns the raw data that represents the data source's content.
- `isLoading` (page 47)
Returns whether the data source is loading its content.
- `pageTitle` (page 47)
Returns the title of the data source's page.
- `representation` (page 48)
Returns the data source's representation depending on its MIME type.
- `textEncodingName` (page 50)
Returns the text encoding for the data source's web view, if set, or the text encoding of the response.

Getting the Request and Response

- `initialRequest` (page 46)
Returns a reference to the original request that was used to load the web content.
- `request` (page 48)
Returns the request that was used to create the data source.
- `response` (page 49)
Returns the associated `WebResourceResponse` object for this data source.

Getting the Web Frame

- `webFrame` (page 51)
Returns the web frame that represents this data source.

Getting an Unreachable URL

- `unreachableURL` (page 50)
Returns the data source's unreachable URL.

Getting a Web Archive

- `webArchive` (page 51)
Returns a web archive representing the data source, its subresources, and subframes.

Accessing Subresources

- `mainResource` (page 47)
Creates and returns a `WebResource` object representing the data source.

- [addSubresource](#): (page 45)
Adds a resource to the data source’s list of subresources.
- [subresourceForURL](#): (page 49)
Returns a subresource for the given URL.
- [subresources](#) (page 50)
Returns the data source’s subresources that have finished downloading.

Instance Methods

addSubresource:

Adds a resource to the data source’s list of subresources.

```
- (void)addSubresource:(WebResource *)subresource
```

Parameters

subresource

The resource to add to the data source.

Discussion

If the data source needs to reload the resource’s URL, it loads the data from *subresource* instead of the network. For example, use this method if you want to use a previously downloaded image rather than accessing the network to reload a resource. If the data source already has a resource with the same URL as *subresource*, then this method replaces it.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [subresourceForURL](#): (page 49)
- [subresources](#) (page 50)

Declared In

WebDataSource.h

data

Returns the raw data that represents the data source’s content.

```
- (NSData *)data
```

Return Value

The data used to display the webpage or `nil` if the data source hasn’t loaded any data. The format of the data is dependent on the data source’s MIME type (obtained from the response).

Discussion

Use the [isLoading](#) (page 47) method to test if a data source is in the process of loading.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [pageTitle](#) (page 47)
- [representation](#) (page 48)

Declared In

WebDataSource.h

initialRequest

Returns a reference to the original request that was used to load the web content.

```
- (NSURLRequest *)initialRequest
```

Return Value

The initial request.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [request](#) (page 48)
- [response](#) (page 49)

Declared In

WebDataSource.h

initWithRequest:

initializes a data source with a URL request.

```
- (id)initWithRequest:(NSURLRequest *)request
```

Parameters

request

The URL request used to load the web content.

Return Value

The initialized web data source.

Discussion

This method is the designated initializer for `WebDataSource` objects. Normally, `WebFrame` objects create their data sources, so you should not invoke this method directly.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDataSource.h

isLoading

Returns whether the data source is loading its content.

- (BOOL)isLoading

Return Value

YES if the data source is in the process of loading its content; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [data](#) (page 45)
- [pageTitle](#) (page 47)
- [representation](#) (page 48)

Declared In

WebDataSource.h

mainResource

Creates and returns a `WebResource` object representing the data source.

- (WebResource *)mainResource

Return Value

A web resource representing the data source. The contents returned are based on the original downloaded data. You can use the returned value to create a `WebArchive` object instead of using the [webArchive](#) (page 51) method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDataSource.h

pageTitle

Returns the title of the data source's page.

- (NSString *)pageTitle

Return Value

The page title or `nil` if the page has no title or the page title hasn't been loaded yet.

Discussion

The associated web view notifies its frame load delegate when the page title is loaded by invoking the [webView:didReceiveTitle:forFrame:](#) (page 237) delegate method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [data](#) (page 45)
- [isLoading](#) (page 47)
- [representation](#) (page 48)

Declared In

WebDataSource.h

representation

Returns the data source's representation depending on its MIME type.

```
- (id < WebDocumentRepresentation >)representation
```

Return Value

The data source's representation or `nil` if the data source is in the process of being loaded and this method is invoked before loading is complete.

Discussion

You can specify the mapping between a representation and MIME type using the [registerViewClass:representationClass:forMIMEType:](#) (page 145) `WebView` class method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [data](#) (page 45)
- [isLoading](#) (page 47)
- [pageTitle](#) (page 47)

Declared In

WebDataSource.h

request

Returns the request that was used to create the data source.

```
- (NSMutableURLRequest *)request
```

Return Value

The request used to load the web content. This URL may be different from the original request returned by the [initialRequest](#) (page 46) method.

Discussion

A web view's resource load delegate may modify requests by implementing the [webView:resource:willSendRequest:redirectResponse:fromDataSource:](#) (page 281) method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [initialRequest](#) (page 46)
- [response](#) (page 49)

Declared In

WebDataSource.h

response

Returns the associated `WebResourceResponse` object for this data source.

- (NSURLResponse *)response

Return Value

The associated `WebResourceResponse` object or `nil` if a response has not been received yet..

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [initialRequest](#) (page 46)
- [request](#) (page 48)

Declared In

WebDataSource.h

subresourceForURL:

Returns a subresource for the given URL.

- (WebResource *)subresourceForURL:(NSURL *)URL

Parameters

URL

The subresource's URL.

Return Value

The subresource for *URL* or `nil` if the data source hasn't finished downloading the subresource.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [addSubresource:](#) (page 45)
- [subresources](#) (page 50)

Declared In

WebDataSource.h

subresources

Returns the data source's subresources that have finished downloading.

- (NSArray *)subresources

Return Value

An array of the data source's loaded subresources.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [addSubresource](#): (page 45)
- [subresourceForURL](#): (page 49)

Declared In

WebDataSource.h

textEncodingName

Returns the text encoding for the data source's web view, if set, or the text encoding of the response.

- (NSString *)textEncodingName

Return Value

The text encoding.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDataSource.h

unreachableURL

Returns the data source's unreachable URL.

- (NSURL *)unreachableURL

Return Value

The unreachable URL if it exists; otherwise, nil.

Discussion

The data source has an unreachable URL if it was created using the

[loadAlternateHTMLString:baseURL:forUnreachableURL](#): (page 60) WebFrame method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebDataSource.h

webArchive

Returns a web archive representing the data source, its subresources, and subframes.

- (WebArchive *)webArchive

Return Value

A web archive representing the data source constructed using the original downloaded data. In the case of HTML, if the current content is preferred, then send `webArchive` to the appropriate DOM object.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [mainResource](#) (page 47)

Declared In

WebDataSource.h

webFrame

Returns the web frame that represents this data source.

- (WebFrame *)webFrame

Return Value

The associated web frame.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDataSource.h

WebDownload Class Reference

Inherits from	NSURLDownload : NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDownload.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guides	WebKit Objective-C Programming Guide URL Loading System Programming Guide

Overview

`WebDownload` objects initiate download client requests on behalf of a delegate. A download request involves loading the data, decoding it (if necessary), and saving it to a file. Instances of this class behave similar to `NSURLDownload` except delegates of `WebDownload` may implement an additional delegate method. The method allows the delegate to specify the window to be used for authentication sheets. If the delegate does not implement this method, the `WebDownload` object will prompt the user for authentication using the standard WebKit authentication panel, as either a sheet or window. There are no additional methods defined in this class.

Delegate Methods

downloadWindowForAuthenticationSheet:

Returns the window to be used by the authentication sheet.

```
- (NSWindow *)downloadWindowForAuthenticationSheet:(WebDownload *)download
```

Parameters

sender

The download object that is requesting the window.

Return Value

An `NSWindow` object into which the `WebDownload` object should draw its authentication sheet.

Discussion

The default implementation prompts the user for authentication using the standard WebKit authentication panel, as either a sheet or window.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDownload.h

WebFrame Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebFrame.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide
Related sample code	NewsReader

Overview

A `WebFrame` object encapsulates the data displayed in a `WebFrameView` object. There is one `WebFrame` object per frame displayed in a `WebView`. An entire webpage is represented by a hierarchy of `WebFrame` objects in which the root object is called the **main frame**.

Each `WebFrame` also has a `WebDataSource` object that manages the loading of frame content. You use the [loadRequest:](#) (page 62) method to initiate an asynchronous client request which will create a provisional data source. The provisional data source will transition to a committed data source once any data has been received.

There are some special, predefined, frame names that you can use when referring to or finding a `WebFrame`. Some of the predefined frame names are: “_self”, “_current”, “_parent”, and “_top.” See [findFrameNamed:](#) (page 58) for a description of their meaning. Frame names may also be specified in the HTML source, or set by clients.

However, the group name is an arbitrary identifier used to group related frames. For example, JavaScript running in a frame can access any other frame in the same group. It's up to the application how it chooses to scope related frames.

Tasks

Initializing Frames

- [initWithName:webFrameView:webView:](#) (page 60)
Initializes the receiver with a frame name, web frame view, and controlling web view.

Loading Content

- [loadRequest:](#) (page 62)
Connects to a given URL by initiating an asynchronous client request.
- [reload](#) (page 64)
Reloads the initial request passed as an argument to [loadRequest:](#) (page 62).
- [reloadFromOrigin](#) (page 64)
Performs an end-to-end revalidation using cache-validating conditionals if possible.
- [stopLoading](#) (page 65)
Stops any pending loads on the receiver's data source, and those of its children.
- [loadAlternateHTMLString:baseURL:forUnreachableURL:](#) (page 60)
Loads alternate content for a frame whose URL is unreachable.
- [loadHTMLString:baseURL:](#) (page 62)
Sets the main page contents and base URL.
- [loadData:MIMETYPE:textEncodingName:baseURL:](#) (page 61)
Sets the main page contents, MIME type, content encoding, and base URL.
- [loadArchive:](#) (page 61)
Loads an archive into the web frame.

Getting the Data Source

- [dataSource](#) (page 58)
Returns the committed data source.
- [provisionalDataSource](#) (page 64)
Returns the provisional data source.

Getting Related Frames and Views

- [parentFrame](#) (page 63)
Returns the web frame's parent web frame.
- [childFrames](#) (page 57)
Returns the frames of the web frame's immediate children.
- [frameView](#) (page 59)
Returns the web frame's view object.

- [webView](#) (page 65)
Returns the view object that manages the web frame.

Finding Frames

- [findFrameNamed:](#) (page 58)
Returns a web frame that matches the given name.
- [name](#) (page 63)
Returns the web frame's name.

Getting DOM Objects

- [DOMDocument](#) (page 58)
Returns the web frame's DOM document.
- [frameElement](#) (page 59)
Returns the web view's DOM frame element.
- [globalContext](#) (page 60)
Returns the global JavaScript execution context for bridging between the WebKit and JavaScriptCore C API.
- [windowObject](#) (page 66)
Returns the JavaScript window object.

Instance Methods

childFrames

Returns the frames of the web frame's immediate children.

- (NSArray *)childFrames

Return Value

The web frame's immediate children. Each child web frame is an instance of `WebFrame` and corresponds to an HTML frameset or `iframe` element.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [parentFrame](#) (page 63)

Declared In

`WebFrame.h`

dataSource

Returns the committed data source.

- (WebDataSource *)dataSource

Return Value

The committed data source, or `nil` if the provisional data source is not done loading.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [provisionalDataSource](#) (page 64)

Declared In

WebFrame.h

DOMDocument

Returns the web frame's DOM document.

- (DOMDocument *)DOMDocument

Return Value

The web frame's DOM document.

Discussion

Returns `nil` if the receiver doesn't have a DOM document; for example, if it's a standalone image.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebFrame.h

findFrameNamed:

Returns a web frame that matches the given name.

- (WebFrame *)findFrameNamed:(NSString *)name

Parameters

name

The name of a frame.

Return Value

For predefined names, returns the receiver if name is “_self” or “_current”; returns the receiver's parent frame if name is “_parent”; and returns the main frame if name is “_top”. Also returns the receiver if it is the main frame and name is either “_parent” or “_top.” For other names, this method returns the first frame that matches *name*. Returns `nil` if no match is found.

Discussion

This method searches the receiver and its descendents first, then the receiver's parent and its children moving up the hierarchy until a match is found. If no match is found in the receivers hierarchy, this method will search for a matching frame in other main frame hierarchies.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [name](#) (page 63)

Declared In

WebFrame.h

frameElement

Returns the web view's DOM frame element.

- (DOMHTML*Element *)frameElement

Return Value

The web view's DOM frame element. Returns `nil` if the receiver is the main frame.

Discussion

The returned object may be an instance of either `DOMHTMLFrameElement`, `DOMHTMLIFrameElement` or `DOMHTMLObjectElement`.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebFrame.h

frameView

Returns the web frame's view object.

- (WebFrameView *)frameView

Return Value

The web frame's view object.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView](#) (page 65)

Declared In

WebFrame.h

globalContext

Returns the global JavaScript execution context for bridging between the WebKit and JavaScriptCore C API.

- (JSGlobalContextRef)globalContext

Return Value

The global JavaScript execution context.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebFrame.h

initWithName:webFrameView:webView:

Initializes the receiver with a frame name, web frame view, and controlling web view.

```
- (id)initWithName:(NSString *)frameName webFrameView:(WebFrameView *)frameView
    webView:(WebView *)webView
```

Parameters

frameName

The frame name. Typically a custom name or `nil` (if none is specified). It would be inappropriate to use one of the predefined frame names described in [findFrameNamed:](#) (page 58) as they have special meanings.

view

The view that displays this web frame—the view associated with the receiver.

webView

The parent view that manages the main frame and its children.

Return Value

An initialized web frame.

Discussion

Normally, you do not invoke this method directly. `WebView` objects automatically create the main frame and subsequent children when new content is loaded. Send a [loadRequest:](#) (page 62) message to the main frame of a `WebView` to load web content.

This method is the designated initializer for the `WebFrame` class.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrame.h

loadAlternateHTMLString:baseURL:forUnreachableURL:

Loads alternate content for a frame whose URL is unreachable.

```
- (void)loadAlternateHTMLString:(NSString *)string baseURL:(NSURL *)URL
  forUnreachableURL:(NSURL *)unreachableURL
```

Parameters*string*

The string to use as the main page for the document.

URL

A file that is used to resolve relative URLs within the document.

unreachableURL

The URL for the alternate page content.

Discussion

Use this method to display page-level loading errors in a web view. Typically, a `WebFrameLoadDelegate` or `WebPolicyDelegate` object invokes this method from these methods:

[webView:didFailProvisionalLoadWithError:forFrame:](#) (page 235) (`WebFrameLoadDelegate`),

[webView:decidePolicyForMIMEType:request:frame:decisionListener:](#) (page 268)

(`WebPolicyDelegate`), or [webView:unableToImplementPolicyWithError:frame:](#) (page 270)

(`WebPolicyDelegate`). If invoked from one of these methods, the back-forward list is maintained.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebFrame.h

loadArchive:

Loads an archive into the web frame.

```
- (void)loadArchive:(WebArchive *)archive
```

Parameters*archive*

The archive to load.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebFrame.h

loadData:MIMEType:textEncodingName:baseURL:

Sets the main page contents, MIME type, content encoding, and base URL.

```
- (void)loadData:(NSData *)data MIMEType:(NSString *)MIMEType
  textEncodingName:(NSString *)encodingName baseURL:(NSURL *)URL
```

Parameters*data*

The data to use for the main page of the document.

MIMEType

The MIME type of the data.

encodingName

The IANA encoding name (for example, “utf-8” or “utf-16”).

URL

A file that is used to resolve relative URLs within the document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadHTMLString:baseURL:](#) (page 62)

Declared In

WebFrame.h

loadHTMLString:baseURL:

Sets the main page contents and base URL.

```
- (void)loadHTMLString:(NSString *)string baseURL:(NSURL *)URL
```

Parameters*string*

The string to use as the main page for the document.

Since the string is treated as a webpage with UTF-8 encoding, the default encoding for any script elements referenced by the HTML is also UTF-8. To avoid this, include a character set attribute on the script element.

URL

A file that is used to resolve relative URLs within the document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadData:MIMEType:textEncodingName:baseURL:](#) (page 61)

Related Sample Code

NewsReader

Declared In

WebFrame.h

loadRequest:

Connects to a given URL by initiating an asynchronous client request.

```
- (void)loadRequest:(NSURLRequest *)request
```

Parameters

request

A client request.

Discussion

Creates a provisional data source that will transition to a committed data source once any data has been received. Use the [dataSource](#) (page 58) method to check if a committed data source is available, and the [stopLoading](#) (page 65) method to stop the load. This method is typically invoked on the main frame.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [reload](#) (page 64)
- [stopLoading](#) (page 65)

Declared In

WebFrame.h

name

Returns the web frame's name.

- (NSString *)name

Return Value

The web frame's name.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [findFrameNamed:](#) (page 58)

Declared In

WebFrame.h

parentFrame

Returns the web frame's parent web frame.

- (WebFrame *)parentFrame

Return Value

The parent web frame, or `nil` if it has none.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [childFrames](#) (page 57)

Declared In

WebFrame.h

provisionalDataSource

Returns the provisional data source.

- (WebDataSource *)provisionalDataSource

Return Value

The provisional data source, or `nil` if either a load request is not in progress or a load request has completed.

Discussion

Use the [loadRequest:](#) (page 62) method to initiate an asynchronous client request, which creates a provisional data source. The provisional data source transitions to a committed data source once any data is received.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [dataSource](#) (page 58)

Declared In

WebFrame.h

reload

Reloads the initial request passed as an argument to [loadRequest:](#) (page 62).

- (void)reload

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadRequest:](#) (page 62)

- [reloadFromOrigin](#) (page 64)

- [stopLoading](#) (page 65)

Declared In

WebFrame.h

reloadFromOrigin

Performs an end-to-end revalidation using cache-validating conditionals if possible.

- (void)reloadFromOrigin

Discussion

End-to-end reload may be necessary if the cache entry has become corrupted for some reason.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [loadRequest:](#) (page 62)
- [reload](#) (page 64)
- [stopLoading](#) (page 65)

Declared In

WebFrame.h

stopLoading

Stops any pending loads on the receiver's data source, and those of its children.

- (void)stopLoading

Discussion

This method does not change the state of the receiver—whatever content has been loaded is preserved.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadRequest:](#) (page 62)
- [reload](#) (page 64)
- [reloadFromOrigin](#) (page 64)

Declared In

WebFrame.h

webView

Returns the view object that manages the web frame.

- (WebView *)webView

Return Value

The view object that manages the entire hierarchy of web frame objects that contains the receiver.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [frameView](#) (page 59)

Declared In

WebFrame.h

windowObject

Returns the JavaScript window object.

- (WebScriptObject *)windowObject

Return Value

The JavaScript window object.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebFrame.h

WebView Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebView.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

`WebView` objects and their subviews display the web content contained in a frame. You never create instances of `WebView` directly—`WebView` objects create and manage a hierarchy of `WebView` objects, one for each frame. `WebView` objects use a scroll view whose document view conforms to the `WebViewDocumentView` protocol.

Tasks

Getting the Web Frame

- [webView](#) (page 71)
Returns the web frame.

Getting Subviews

- [documentView](#) (page 69)
Returns the subview that displays the web content.

Setting Scrolling Behavior

- [setAllowsScrolling](#): (page 70)
Sets whether the frame view should allow users to scroll.
- [allowsScrolling](#) (page 68)
Returns a Boolean value indicating whether users can scroll.

Printing Views

- [canPrintHeadersAndFooters](#) (page 68)
Returns a Boolean value indicating whether the receiver can print headers and footers.
- [printOperationWithPrintInfo](#): (page 70)
Returns a print operation object to print this frame.
- [documentViewShouldHandlePrint](#) (page 69)
Returns a Boolean value indicating whether the document view should handle a print operation.
- [printDocumentView](#) (page 69)
Prints the receiver.

Instance Methods

allowsScrolling

Returns a Boolean value indicating whether users can scroll.

- (BOOL)allowsScrolling

Return Value

YES if the receiver allows users to scroll; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setAllowsScrolling](#): (page 70)

Declared In

WebFrameView.h

canPrintHeadersAndFooters

Returns a Boolean value indicating whether the receiver can print headers and footers.

- (BOOL)canPrintHeadersAndFooters

Return Value

YES if the receiver can print headers and footers; otherwise, NO.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebFrameView.h

documentView

Returns the subview that displays the web content.

- (NSView < WebDocumentView > *)documentView

Return Value

The subview that displays the web content.

Discussion

Use [setAllowsScrolling:](#) (page 70) to enable scrolling of this view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameView.h

documentViewShouldHandlePrint

Returns a Boolean value indicating whether the document view should handle a print operation.

- (BOOL)documentViewShouldHandlePrint

Return Value

YES if the document view should handle the print operation; otherwise, NO.

Discussion

If this method returns NO, the application terminates its print operation and sends [printDocumentView](#) (page 69) to the web frame view.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebFrameView.h

printDocumentView

Prints the receiver.

- (void)printDocumentView

Discussion

This method is invoked if the `documentViewShouldHandlePrint` (page 69) method returns NO.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

`WebFrameView.h`

printOperationWithPrintInfo:

Returns a print operation object to print this frame.

```
- (NSPrintOperation *)printOperationWithPrintInfo:(NSPrintInfo *)printInfo
```

Parameters

printInfo

Information about the print settings needed to print this frame. See *NSPrintInfo Class Reference* for more information about this object.

Return Value

An *NSPrintOperation* object set up to print this frame. See *NSPrintOperation Class Reference* for more information about this object.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

`WebFrameView.h`

setAllowsScrolling:

Sets whether the frame view should allow users to scroll.

```
- (void)setAllowsScrolling:(BOOL) flag
```

Parameters

flag

If YES, scrolling is allowed; if NO, it is not. If the frame contains a scrolling element, then that value is used as the default; otherwise, the default is YES.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [allowsScrolling](#) (page 68)

Declared In

`WebFrameView.h`

webFrame

Returns the web frame.

- (WebFrame *)webFrame

Return Value

The web frame.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameView.h

WebHistory Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebHistory.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

WebHistory objects are used to maintain the pages visited by users. Visited pages are represented by WebHistoryItem objects. You add and remove history items using the [addItem:](#) (page 75) and [removeItem:](#) (page 79) methods. These methods post appropriate notifications when items are added or removed so you can update the display. WebHistory organizes the WebHistoryItem objects by the day they were visited, ordered from most recent to oldest. You can request all the days that contain history items using the [orderedLastVisitedDays](#) (page 78) method or request the items visited on a particular day using the [orderedItemsLastVisitedOnDay:](#) (page 77) method. WebHistory objects can be loaded and saved by specifying a file URL (see [loadFromURL:error:](#) (page 77)).

Tasks

Accessing Shared History Objects

- + [optionalSharedHistory](#) (page 74)
Returns a shared web history object, if one exists.
- + [setOptionalSharedHistory:](#) (page 75)
Sets the web history object to share.

Adding and Removing History Items

- [addItem:](#) (page 75)
Inserts or updates the specified items in the web history.

- `removeItems:` (page 79)
Removes the specified items from the web history.
- `removeAllItems` (page 78)
Removes all items from the web history.

Getting Web History Items

- `orderedItemsLastVisitedOnDay:` (page 77)
Returns web history items that were last visited on the specified date.
- `orderedLastVisitedDays` (page 78)
Returns all calendar days represented in the web history.
- `itemForURL:` (page 76)
Returns the web history item that corresponds to the specified web location.

Loading and Saving History Information

- `loadFromURL:error:` (page 77)
Loads the contents of the specified web history file.
- `saveToURL:error:` (page 79)
Saves the web history to the specified file.

Getting and Setting Attributes

- `historyAgeInDaysLimit` (page 76)
Returns the maximum age of web history items that can be retrieved.
- `setHistoryAgeInDaysLimit:` (page 80)
Sets the maximum age of web history items that can be retrieved.
- `historyItemLimit` (page 76)
Returns the maximum number of web history items that can be stored.
- `setHistoryItemLimit:` (page 80)
Sets the maximum number of web history items to store.

Class Methods

optionalSharedHistory

Returns a shared web history object, if one exists.

```
+ (WebHistory *)optionalSharedHistory
```

Return Value

A shared web history object initialized with the default web history file, or `nil` if one was not previously specified using the `setOptionalSharedHistory:` (page 75) method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadFromURL:error:](#) (page 77)

Declared In

WebHistory.h

setOptionalSharedHistory:

Sets the web history object to share.

```
+ (void)setOptionalSharedHistory:(WebHistory *)history
```

Parameters

history

The web history object to share.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

+ [optionalSharedHistory](#) (page 74)

- [loadFromURL:error:](#) (page 77)

Declared In

WebHistory.h

Instance Methods

addItem:

Inserts or updates the specified items in the web history.

```
- (void)addItem:(NSArray *)newItems
```

Parameters

newItems

An array of web history items to add. If an item in the array already exists in the web history this method replaces the existing item, so that the last-visited date for the item is updated.

Discussion

When successful, this method posts a notification ([WebHistoryItemsAddedNotification](#) (page 81)).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [removeAllItems](#) (page 78)
- [removeItems:](#) (page 79)

Declared In

WebHistory.h

historyAgeInDaysLimit

Returns the maximum age of web history items that can be retrieved.

- (int)historyAgeInDaysLimit

Return Value

The maximum age, in days, of web history items that can be retrieved.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setHistoryAgeInDaysLimit:](#) (page 80)
- [historyItemLimit](#) (page 76)

Declared In

WebHistory.h

historyItemLimit

Returns the maximum number of web history items that can be stored.

- (int)historyItemLimit

Return Value

The maximum number of web history items that can be stored.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setHistoryItemLimit:](#) (page 80)
- [historyAgeInDaysLimit](#) (page 76)

Declared In

WebHistory.h

itemForURL:

Returns the web history item that corresponds to the specified web location.

- (WebHistoryItem *)itemForURL:(NSURL *)URL

Parameters*URL*

The location, as a URL, of the webpage that was visited.

Return Value

The web history item that represents visits to the specified URL, or `nil` if none was found.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebHistory.h

loadFromURL:error:

Loads the contents of the specified web history file.

```
- (BOOL)loadFromURL:(NSURL *)URL error:(NSError **)error
```

Parameters*URL*

The URL of the file to load. The file should have been created previously by a web history object. Note that the file's format is private and should not be edited directly.

error

On output, `nil` if the load was successful; otherwise, *error*, contains details of the failure.

Return Value

YES if successful; otherwise, NO.

Discussion

When successful, this method posts a notification ([WebHistoryLoadedNotification](#) (page 82)).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [saveToURL:error:](#) (page 79)
- + [setOptionalSharedHistory:](#) (page 75)
- + [optionalSharedHistory](#) (page 74)

Declared In

WebHistory.h

orderedItemsLastVisitedOnDay:

Returns web history items that were last visited on the specified date.

```
- (NSArray *)orderedItemsLastVisitedOnDay:(NSDate *)calendarDate
```

Parameters

calendarDate

The date on which the web history items were last visited.

Return Value

An array of web history items that were last visited on the specified date.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [orderedLastVisitedDays](#) (page 78)

Declared In

WebHistory.h

orderedLastVisitedDays

Returns all calendar days represented in the web history.

- (NSArray *)orderedLastVisitedDays

Return Value

An array of calendar days, in order from most recent to oldest. Each calendar day is associated with at least one web history item.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [orderedItemsLastVisitedOnDay:](#) (page 77)

Declared In

WebHistory.h

removeAllItems

Removes all items from the web history.

- (void)removeAllItems

Discussion

When successful, this method posts a notification ([WebHistoryAllItemsRemovedNotification](#) (page 81)).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [addItem:](#) (page 75)

- [removeItems:](#) (page 79)

Declared In

WebHistory.h

removeItems:

Removes the specified items from the web history.

```
- (void)removeItems:(NSArray *)items
```

Parameters

items

An array of web history items to remove.

Discussion

When successful, this method posts a notification ([WebHistoryItemsRemovedNotification](#) (page 82)).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [addItem:](#) (page 75)

- [removeAllItems](#) (page 78)

Declared In

WebHistory.h

saveToURL:error:

Saves the web history to the specified file.

```
- (BOOL)saveToURL:(NSURL *)URL error:(NSError **)error
```

Parameters

URL

The URL of the file to contain the web history information. The file must be user-writable, but its format is private and should not be edited directly.

error

On output, `nil` if the load was successful; otherwise, `error`, which contains details of the failure.

Return Value

YES if successful; otherwise, NO

Discussion

When successful, this method posts a notification ([WebHistorySavedNotification](#) (page 83)).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadFromURL:error:](#) (page 77)

Declared In

WebHistory.h

setHistoryAgeInDaysLimit:

Sets the maximum age of web history items that can be retrieved.

```
- (void)setHistoryAgeInDaysLimit:(int)limit
```

Parameters

limit

The maximum age, in days, of retrievable web history items.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [historyAgeInDaysLimit](#) (page 76)

Declared In

WebHistory.h

setHistoryItemLimit:

Sets the maximum number of web history items to store.

```
- (void)setHistoryItemLimit:(int)limit
```

Parameters

limit

The maximum number of web history items to store.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [historyItemLimit](#) (page 76)

Declared In

WebHistory.h

Constants

Web History Dictionary Keys

The key for accessing the web history items stored in a notification's user information dictionary.


```
extern NSString *WebHistoryItemsKey;
```

Constants

`WebHistoryItemsKey`

The key to access an array containing the added or removed web history items.

Available in Mac OS X v10.2 and later.

Declared in `WebHistory.h`.

Discussion

This string is used as the key in the `userInfo` dictionary passed as the argument to the [WebHistoryAllItemsRemovedNotification](#) (page 81), [WebHistoryItemsAddedNotification](#) (page 81), and [WebHistoryItemsRemovedNotification](#) (page 82) notifications.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

`WebHistory.h`

Notifications

WebHistoryAllItemsRemovedNotification

Posted when all history items have been removed from the web history.

The notification object is the web history from which the history items were removed. The `userInfo` dictionary contains the following information:

Key	Value
@“WebHistoryItemsKey”	An <code>NSArray</code> object containing the removed items.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [removeAllItems](#) (page 78)

Declared In

`WebHistory.h`

WebHistoryItemsAddedNotification

Posted when history items have been added to a web history.

The notification object is the web history to which the items were added. The `userInfo` dictionary contains the following information:

Key	Value
@“WebHistoryItemsKey”	An NSArray object containing the added items.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [addItem:](#) (page 75)

Declared In

WebHistory.h

WebHistoryItemsRemovedNotification

Posted when items have been removed from the web history.

The notification object is the web history from which the history items were removed. The userInfo dictionary contains the following information:

Key	Value
@“WebHistoryItemsKey”	An NSArray object containing the removed items.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [removeItems:](#) (page 79)

Declared In

WebHistory.h

WebHistoryLoadedNotification

Posted when web history items have been loaded from a URL.

The notification object is the web history that loaded the history items. This notification does not contain a userInfo dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadFromURL:error:](#) (page 77)

Declared In

WebHistory.h

WebHistorySavedNotification

Posted when web history items have been saved to a URL.

The notification object is the web history that saved the history items. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [saveToURL:error:](#) (page 79)

Declared In

WebHistory.h

WebHistoryItem Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebHistoryItem.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

WebHistoryItem objects encapsulate information about visiting a page so that users can return to that page. WebHistory and WebBackForwardList objects manage lists of WebHistoryItem objects. WebHistoryItem objects are created and added to these lists automatically when loading pages, so you do not need to create WebHistoryItem objects directly.

Adopted Protocols

NSCopying
 - copyWithZone:

Tasks

Initializing WebHistoryItem Objects

- initWithURLString:title:lastVisitedTimeInterval: (page 87)
 Initializes the receiver with a URL, *URLString*, a title specified by *title* and the last time this item was visited specified by *time* title, and time last visited.

Getting URL Information

- [URLString](#) (page 89)
Returns the string representation of the URL for the receiver's page.
- [originalURLString](#) (page 87)
Returns the string representation of the original URL for the receiver's page.

Getting and Setting Page Titles

- [title](#) (page 88)
Returns the receiver's original page title.
- [alternateTitle](#) (page 86)
Returns an alternate title that may be used in place of the receiver's page title.
- [setAlternateTitle:](#) (page 88)
Sets an alternate title for a page.

Getting Other Attributes

- [icon](#) (page 87)
Returns the icon for the receiver's page, or `nil` if none exists.
- [lastVisitedTimeInterval](#) (page 87)
Returns the last time and date the receiver's page was visited.

Instance Methods

alternateTitle

Returns an alternate title that may be used in place of the receiver's page title.

```
- (NSString *)alternateTitle
```

Discussion

This method returns `nil` if no alternate title exists.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setAlternateTitle:](#) (page 88)

Declared In

WebHistoryItem.h

icon

Returns the icon for the receiver's page, or `nil` if none exists.

```
- (NSImage *)icon
```

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebHistoryItem.h

initWithURLString:title:lastVisitedTimeInterval:

Initializes the receiver with a URL, *URLString*, a title specified by *title* and the last time this item was visited specified by *time* title, and time last visited.

```
- (id)initWithURLString:(NSString *)URLString title:(NSString *)title
    lastVisitedTimeInterval:(NSTimeInterval)time
```

Discussion

WebKit normally creates WebHistoryItem objects for you but on occasion you might want to create an item and add it to the WebBackForwardList yourself. Note that when an instance is first initialized the strings returned by [URLString](#) (page 89) and [originalURLString](#) (page 87) are the same.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebHistoryItem.h

lastVisitedTimeInterval

Returns the last time and date the receiver's page was visited.

```
- (NSTimeInterval)lastVisitedTimeInterval
```

Discussion

The interval is from a reference date as determined by `NSDate`.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebHistoryItem.h

originalURLString

Returns the string representation of the original URL for the receiver's page.

- (NSString *)originalURLString

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [URLString](#) (page 89)

Declared In

WebHistoryItem.h

setAlternateTitle:

Sets an alternate title for a page.

- (void)setAlternateTitle:(NSString *)*alternateTitle*

Discussion

This is used as a convenience to display or store short versions of the page title.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [alternateTitle](#) (page 86)

- [title](#) (page 88)

Declared In

WebHistoryItem.h

title

Returns the receiver's original page title.

- (NSString *)title

Discussion

The title returned comes from the title HTML tag for HTML documents.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [alternateTitle](#) (page 86)

- [setAlternateTitle:](#) (page 88)

Declared In

WebHistoryItem.h

URLString

Returns the string representation of the URL for the receiver's page.

- (NSString *)URLString

Discussion

This URL may differ from the original URL if the page was, for example, redirected to a new location.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [originalURLString](#) (page 87)

Declared In

WebHistoryItem.h

Notifications

WebHistoryItemChangedNotification

Posted by a WebHistoryItem object when the value of the history item's title, alternate title, URL strings, or last visited interval changes.

This notification does not contain a userInfo dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setAlternateTitle:](#) (page 88)

Declared In

WebHistoryItem.h

WebPreferences Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebPreferences.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

WebPreferences encapsulates the preferences you can change per WebView object. These preferences include font, text encoding, and image settings. Normally a WebView object uses the standard preferences returned by the [standardPreferences](#) (page 96) class method. However, you can modify the preferences for individual WebView instances too. Use the [setPreferencesIdentifier:](#) (page 186) WebView method to change a WebView object’s preferences, or to share preferences between WebView objects. Use the [setAutosaves:](#) (page 106) method to specify if the preferences object should be automatically saved to the user defaults database.

WebPreferences also manages the font preferences for a web view. You can set custom font families for each of the primary web font styles (standard, serif, sans-serif, cursive, and fantasy) as well as their font sizes. The font size preferences alter the display font sizes in a certain way. If the HTML or CSS in the web view’s content specifies font sizes in a relative fashion (such as `font-size=-1` in HTML or `font-size: medium` in CSS), the default font size settings (set by the font size methods prefaced with “default”) have an effect. They do not have an effect for font sizes specified absolutely. The values specified by the minimum font size settings (set by the font size methods prefaced with “minimum”) override all the HTML and CSS font size definitions, and so have an effect on the entirety of the content. The values specified by the minimum logical font size settings (set by the font size methods prefaced with “minimumLogical”) affect all relative font size declarations for HTML and CSS, but also override any CSS font size declarations in the content, whether they are relative or absolute.

The font size for a web view is different than its logical font size. The minimum logical font size, for example, is the absolute minimum size at which the font will display onscreen. This is meant to be a functional boundary and not a style boundary. For example, the default value for a web view’s minimum logical font size is 9 points, because typical web content looks good on Mac OS X at font sizes of 9 point and above. The constraint assures that web content will always look good in a web view. If you know that your content will look good only at 12 points or above, you should change the minimum font size to 12 points and leave the minimum

logical font size alone. This will assure that your content will never display at sizes less than 12 points, but the functional font size boundary of the web view will remain at 9 points to prevent any chance of displaying unnecessarily small text.

Adopted Protocols

NSCoding

```
encodeWithCoder:  
initWithCoder:
```

Tasks

Getting the Standard Preferences

- + [standardPreferences](#) (page 96) **Deprecated in Mac OS X v10.4.11**
Returns the standard set of preferences that may be used by all WebView objects.

Initializing Preferences

- [initWithIdentifier:](#) (page 101)
Returns an initialized WebPreferences object, creating one if it does not exist.

Getting the Identifier

- [identifier](#) (page 100)
Returns the receiver's identifier.

Saving Preferences to the User Defaults Database

- [autosaves](#) (page 97)
Returns whether or not the receiver's attributes are automatically stored in the user defaults database.
- [setAutosaves:](#) (page 106)
Sets whether or not the receiver's attributes are stored in the user defaults database.

Enabling Java

- [setJavaEnabled:](#) (page 110)
Sets whether or not the web view allows Java.
- [isJavaEnabled](#) (page 101)
Returns whether or not Java is enabled for the web view.

Enabling JavaScript

- [setJavaScriptEnabled:](#) (page 110)
Sets whether or not the web view allows JavaScript.
- [isJavaScriptEnabled](#) (page 102)
Returns whether or not JavaScript is enabled for the web view.
- [setJavaScriptCanOpenWindowsAutomatically:](#) (page 110)
Sets whether or not the web view allows JavaScript to open windows automatically.
- [javascriptCanOpenWindowsAutomatically](#) (page 102)
Returns whether or not JavaScript can open windows automatically for the web view.

Enabling Plug-ins

- [setPlugInsEnabled:](#) (page 112)
Sets whether or not the web view allows plug-ins.
- [arePlugInsEnabled](#) (page 97)
Returns whether or not the web view allows plug-ins.

Enabling Style Sheets

- [setUserStyleSheetEnabled:](#) (page 115)
Sets whether or not user style sheets are enabled in the web view.
- [userStyleSheetEnabled](#) (page 118)
Returns whether user style sheets are enabled in the web view.
- [setUserStyleSheetLocation:](#) (page 116)
Sets the location of the user style sheet.
- [userStyleSheetLocation](#) (page 118)
Returns the location of the user style sheet.

Getting and Setting Fonts

- [setCursiveFontFamily:](#) (page 107)
Sets the cursive font family of the web view.
- [cursiveFontFamily](#) (page 98)
Returns the cursive font family for the web view.
- [setFantasyFontFamily:](#) (page 109)
Sets the fantasy font family of the web view.
- [fantasyFontFamily](#) (page 100)
Returns the fantasy font family for the web view.
- [setFixedFontFamily:](#) (page 109)
Sets the fixed font family of the web view.
- [fixedFontFamily](#) (page 100)
Returns the fixed font family for the web view.

- [setSansSerifFontFamily:](#) (page 113)
Sets the sans serif font family of the web view.
- [sansSerifFontFamily](#) (page 104)
Returns the sans serif font family for the web view.
- [setSerifFontFamily:](#) (page 114)
Sets the serif font family of the web view.
- [serifFontFamily](#) (page 105)
Returns the serif font family for the web view.
- [setStandardFontFamily:](#) (page 114)
Sets the standard font family of the web view.
- [standardFontFamily](#) (page 117)
Returns the standard font family used by the web view.

Getting and Setting Font Sizes

- [setDefaultFixedFontSize:](#) (page 107)
Sets the default fixed font size of the web view.
- [defaultFixedFontSize](#) (page 98)
Returns the default fixed font size for the web view.
- [setDefaultFontSize:](#) (page 108)
Sets the default font size of the web view.
- [defaultFontSize](#) (page 99)
Returns the default font size for the web view.
- [setMinimumFontSize:](#) (page 111)
Sets the minimum font size of the web view.
- [minimumFontSize](#) (page 103)
Returns the minimum font size for the web view.
- [setMinimumLogicalFontSize:](#) (page 112)
Sets the minimum logical font size of the web view.
- [minimumLogicalFontSize](#) (page 103)
Returns the minimum logical font size for the web view.

Getting and Setting Text Encoding

- [setDefaultTextEncodingName:](#) (page 108)
Sets the default text encoding of the web view.
- [defaultTextEncodingName](#) (page 99)
Returns the default text encoding for the web view.

Handling Images

- [setAllowsAnimatedImageLooping:](#) (page 105)
Sets whether or not the receiver allows animated images to loop.

- [allowsAnimatedImageLooping](#) (page 96)
Returns whether or not the web view allows animated images to loop.
- [setAllowsAnimatedImages:](#) (page 106)
Sets whether or not the receiver allows animated images.
- [allowsAnimatedImages](#) (page 97)
Returns whether or not the web view allows animated images.
- [setLoadsImagesAutomatically:](#) (page 111)
Sets whether or not the web view allows images to be loaded automatically.
- [loadsImagesAutomatically](#) (page 102)
Returns whether images are loaded automatically for the web view.

Printing Backgrounds

- [setShouldPrintBackgrounds:](#) (page 114)
Sets whether or not the web view should include backgrounds when printing.
- [shouldPrintBackgrounds](#) (page 116)
Returns whether the web view should include backgrounds when printing.

Enabling Private Browsing

- [privateBrowsingEnabled](#) (page 104)
Returns whether or not private browsing is enabled.
- [setPrivateBrowsingEnabled:](#) (page 113)
Sets whether or not private browsing is enabled.

Controlling User Focus

- [tabsToLinks](#) (page 117)
Returns whether or not the tab key will focus links.
- [setTabsToLinks:](#) (page 115)
Sets whether or not the web view will focus control on links when tabbing.

Caching

- [setUsesPageCache:](#) (page 116)
Sets whether the web views associated with the receiver should use the shared page cache.
- [usesPageCache](#) (page 118)
Returns whether the web views associated with the receiver should use the shared page cache.
- [setCacheModel:](#) (page 107)
Sets the cache model for the web views associated with the receiver.
- [cacheModel](#) (page 98)
Returns the cache model for a web view.

Class Methods

standardPreferences

Returns the standard set of preferences that may be used by all `WebView` objects.

```
+ (WebPreferences *)standardPreferences
```

Return Value

A `WebPreferences` object containing the standard set of preferences that may be used by all `WebView` objects.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

`WebPreferences.h`

Instance Methods

allowsAnimatedImageLooping

Returns whether or not the web view allows animated images to loop.

```
- (BOOL)allowsAnimatedImageLooping
```

Return Value

YES if the web view is configured to loop animated images, NO otherwise.

Discussion

The number of times that an image loops is determined by parameters of the image file itself and cannot be set in the web view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [allowsAnimatedImageLooping](#) (page 96)
- [setAllowsAnimatedImages:](#) (page 106)
- [setAllowsAnimatedImages:](#) (page 106)
- [loadsImagesAutomatically](#) (page 102)
- [setLoadsImagesAutomatically:](#) (page 111)

Declared In

`WebPreferences.h`

allowsAnimatedImages

Returns whether or not the web view allows animated images.

- (BOOL)allowsAnimatedImages

Return Value

YES if the web view allows animated images, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [allowsAnimatedImageLooping](#) (page 96)
- [setAllowsAnimatedImages:](#) (page 106)
- [loadsImagesAutomatically](#) (page 102)
- [setLoadsImagesAutomatically:](#) (page 111)

Declared In

WebPreferences.h

arePlugInsEnabled

Returns whether or not the web view allows plug-ins.

- (BOOL)arePlugInsEnabled

Return Value

YES if the web view allows plug-ins, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setPlugInsEnabled:](#) (page 112)

Declared In

WebPreferences.h

autosaves

Returns whether or not the receiver's attributes are automatically stored in the user defaults database.

- (BOOL)autosaves

Return Value

YES if the receiver's attributes are automatically stored, NO otherwise.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setAutosaves:](#) (page 106)

Declared In

WebPreferences.h

cacheModel

Returns the cache model for a web view.

- (WebCacheModel)cacheModel

Return Value

The cache model for the web views associated with the receiver. Possible values are described in [WebCacheModel](#) (page 119).

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [setCacheModel:](#) (page 107)

Declared In

WebPreferences.h

cursiveFontFamily

Returns the cursive font family for the web view.

- (NSString *)cursiveFontFamily

Return Value

The default font used for cursive text.

Discussion

This preference affects text whose properties indicate that it should be displayed in a cursive font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setCursiveFontFamily:](#) (page 107)

Declared In

WebPreferences.h

defaultFixedFontSize

Returns the default fixed font size for the web view.

- (int)defaultFixedFontSize

Return Value

The default font size used for content displayed in a fixed font if no font size is specified.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setDefaultFixedFontSize:](#) (page 107)

Declared In

WebPreferences.h

defaultFontSize

Returns the default font size for the web view.

- (int)defaultFontSize

Return Value

The default font size used for content displayed in a fixed font if no font size or font family is specified.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setDefaultFontSize:](#) (page 108)

Declared In

WebPreferences.h

defaultTextEncodingName

Returns the default text encoding for the web view.

- (NSString *)defaultTextEncodingName

Return Value

The default text encoding used when interpreting content whose encoding is not specified.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setDefaultTextEncodingName:](#) (page 108)

Declared In

WebPreferences.h

fantasyFontFamily

Returns the fantasy font family for the web view.

- (NSString *)fantasyFontFamily

Return Value

The font family used as the default for content marked with a fantasy font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a fantasy font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setFontFamily:](#) (page 109)

Declared In

WebPreferences.h

fixedFontFamily

Returns the fixed font family for the web view.

- (NSString *)fixedFontFamily

Return Value

The font family used as the default for content marked with a fixed font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a fixed font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setFontFamily:](#) (page 109)

Declared In

WebPreferences.h

identifier

Returns the receiver's identifier.

- (NSString *)identifier

Return Value

The identifier of the WebPreferences object.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [initWithIdentifier:](#) (page 101)

Declared In

WebPreferences.h

initWithIdentifier:

Returns an initialized WebPreferences object, creating one if it does not exist.

```
- (id)initWithIdentifier:(NSString *)anIdentifier
```

Discussion

This method returns either the receiver initialized with *anIdentifier*, or another WebPreferences object matching *anIdentifier* if it exists.

The *anIdentifier* argument should be a unique identifier—it will be prepended to the keys used to store the receiver's attributes in the user defaults database. WebView objects can share instances of WebPreferences by using the same preferences identifier.

Typically, you do not invoke this method directly. Instead, you set the preferences identifier by sending a [setPreferencesIdentifier:](#) (page 186) message to your WebView object. This method is the designated initializer for the WebPreferences class.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [identifier](#) (page 100)

Declared In

WebPreferences.h

isJavaEnabled

Returns whether or not Java is enabled for the web view.

```
- (BOOL)isJavaEnabled
```

Return Value

YES if the view allows Java, NO otherwise.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setJavaEnabled:](#) (page 110)

Declared In

WebPreferences.h

isJavaScriptEnabled

Returns whether or not JavaScript is enabled for the web view.

- (BOOL)isJavaScriptEnabled

Return Value

YES if the web view allows JavaScript, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setJavaScriptEnabled:](#) (page 110)

Declared In

WebPreferences.h

javaScriptCanOpenWindowsAutomatically

Returns whether or not JavaScript can open windows automatically for the web view.

- (BOOL)javaScriptCanOpenWindowsAutomatically

Return Value

YES if the web view allows JavaScript to open windows automatically, otherwise NO.

Discussion

Explicit calls to a JavaScript window opener that are activated by user action (such as a button click) are not affected by this setting.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setJavaScriptCanOpenWindowsAutomatically:](#) (page 110)

Declared In

WebPreferences.h

loadImagesAutomatically

Returns whether images are loaded automatically for the web view.

- (BOOL)loadImagesAutomatically

Return Value

YES if the web view allows images to be loaded automatically, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setLoadsImagesAutomatically:](#) (page 111)
- [allowsAnimatedImages](#) (page 97)
- [setAllowsAnimatedImages:](#) (page 106)
- [allowsAnimatedImageLooping](#) (page 96)
- [setAutosaves:](#) (page 106)

Declared In

WebPreferences.h

minimumFontSize

Returns the minimum font size for the web view.

```
- (int)minimumFontSize
```

Return Value

The minimum font size that will be used for content regardless of the font size specified by that content.

Discussion

The default value is 1, meaning that the minimum font size is constrained only by the minimum logical font size.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setMinimumFontSize:](#) (page 111)

Declared In

WebPreferences.h

minimumLogicalFontSize

Returns the minimum logical font size for the web view.

```
- (int)minimumLogicalFontSize
```

Return Value

The minimum logical font size used by the web view.

Discussion

The minimum logical font size is the smallest font size that will display in a web view when the content's font size is imprecisely specified. This includes content with logical sizes (such as `small`) or with a font size specified as a percentage of the default.

Its default value is a 9-point font size.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setMinimumLogicalFontSize:](#) (page 112)

Declared In

WebPreferences.h

privateBrowsingEnabled

Returns whether or not private browsing is enabled.

- (BOOL)privateBrowsingEnabled

Return Value

YES if private browsing is enabled, otherwise NO.

Discussion

Private browsing prevents the web view from maintaining any history, cache, or AutoFill information for the pages being visited.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setPrivateBrowsingEnabled:](#) (page 113)

Declared In

WebPreferences.h

sansSerifFontFamily

Returns the sans serif font family for the web view.

- (NSString *)sansSerifFontFamily

Return Value

The font family used as the default for content marked with a sans-serif font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a sans-serif font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setSansSerifFontFamily:](#) (page 113)

Declared In

WebPreferences.h

serifFontFamily

Returns the serif font family for the web view.

- (NSString *)serifFontFamily

Return Value

The font family used as the default for content marked with a serif font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a serif font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setSerifFontFamily:](#) (page 114)

Declared In

WebPreferences.h

setAllowsAnimatedImageLooping:

Sets whether or not the receiver allows animated images to loop.

- (void)setAllowsAnimatedImageLooping:(BOOL)flag

Parameters

flag

YES if the web view should loop animated images, NO otherwise.

Discussion

If image looping is disabled, the web view displays it as a static image. The number of times that an image loops is determined by parameters of the image file itself and cannot be set in the web view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [allowsAnimatedImageLooping](#) (page 96)

- [allowsAnimatedImages](#) (page 97)

- [setAllowsAnimatedImages:](#) (page 106)

- [loadsImagesAutomatically](#) (page 102)

- [setLoadsImagesAutomatically:](#) (page 111)

Declared In

WebPreferences.h

setAllowsAnimatedImages:

Sets whether or not the receiver allows animated images.

- (void)setAllowsAnimatedImages:(BOOL)flag

Parameters

flag

Set to YES if the web view should allow animated images, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [allowsAnimatedImages](#) (page 97)
- [setAutosaves:](#) (page 106)
- [allowsAnimatedImageLooping](#) (page 96)
- [loadsImagesAutomatically](#) (page 102)
- [setLoadsImagesAutomatically:](#) (page 111)

Declared In

WebPreferences.h

setAutosaves:

Sets whether or not the receiver's attributes are stored in the user defaults database.

- (void)setAutosaves:(BOOL)flag

Parameters

flag

Set to YES if the receiver's attributes should be automatically stored in the user defaults database, NO otherwise.

Discussion

The default value is NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [autosaves](#) (page 97)

Declared In

WebPreferences.h

setCacheModel:

Sets the cache model for the web views associated with the receiver.

```
- (void)setCacheModel:(WebCacheModel)cacheModel
```

Parameters

cacheModel

The cache model for the web views associated with the receiver. Possible values are described in [WebCacheModel](#) (page 119).

Discussion

Set this property to optimize WebKit's cache footprint (on disk and in memory) to best fit the use of the web view. If a web view is used only for a single webpage, use the [WebCacheModelDocumentViewer](#) (page 119) constant instead.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [cacheModel](#) (page 98)

Declared In

WebPreferences.h

setCursiveFontFamily:

Sets the cursive font family of the web view.

```
- (void)setCursiveFontFamily:(NSString *)family
```

Parameters

family

The font family to use as the default font for cursive text.

Discussion

This preference affects text whose properties indicate that it should be displayed in a cursive font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [cursiveFontFamily](#) (page 98)

Declared In

WebPreferences.h

setDefaultFixedFontSize:

Sets the default fixed font size of the web view.

```
- (void)setDefaultFixedFontSize:(int)size
```

Parameters*size*

The default font size to use for content displayed in a fixed font if no font size is specified.

Discussion

The font size specified by *size* should always be greater than zero.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [defaultFixedFontSize](#) (page 98)

Declared In

WebPreferences.h

setDefaultFontSize:

Sets the default font size of the web view.

```
- (void)setDefaultFontSize:(int)size
```

Parameters*size*

The default font size to use for content displayed in a fixed font if no font size or font family is specified.

Discussion

The font size specified by *size* should always be greater than zero.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [defaultFontSize](#) (page 99)

Declared In

WebPreferences.h

setDefaultTextEncodingName:

Sets the default text encoding of the web view.

```
- (void)setDefaultTextEncodingName:(NSString *)encoding
```

Parameters*encoding*

The default text encoding to use when interpreting content whose encoding is not specified.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [defaultTextEncodingName](#) (page 99)

Declared In

WebPreferences.h

setFantasyFontFamily:

Sets the fantasy font family of the web view.

```
- (void)setFantasyFontFamily:(NSString *)family
```

Parameters

family

The font family to use as the default for content marked with a fantasy font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a fantasy font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [fantasyFontFamily](#) (page 100)

Declared In

WebPreferences.h

setFixedFontFamily:

Sets the fixed font family of the web view.

```
- (void)setFixedFontFamily:(NSString *)family
```

Parameters

family

The font family to use as the default for content marked with a fixed font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a fixed font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [fixedFontFamily](#) (page 100)

Declared In

WebPreferences.h

setJavaEnabled:

Sets whether or not the web view allows Java.

- (void)setJavaEnabled:(BOOL)flag

Parameters

flag

Set to YES to indicate that the web view allows Java, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [isJavaEnabled](#) (page 101)

Declared In

WebPreferences.h

setJavaScriptCanOpenWindowsAutomatically:

Sets whether or not the web view allows JavaScript to open windows automatically.

- (void)setJavaScriptCanOpenWindowsAutomatically:(BOOL)flag

Parameters

flag

Set to YES if the web view should allow JavaScript to open windows automatically, otherwise NO.

Discussion

Explicit calls to a JavaScript window opener that are activated by user action (such as a button click) are not affected by this setting.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [javascriptCanOpenWindowsAutomatically](#) (page 102)

Declared In

WebPreferences.h

setJavaScriptEnabled:

Sets whether or not the web view allows JavaScript.

- (void)setJavaScriptEnabled:(BOOL)flag

Parameters

flag

Set to YES if the web view should allow JavaScript, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [isJavaScriptEnabled](#) (page 102)

Declared In

WebPreferences.h

setLoadsImagesAutomatically:

Sets whether or not the web view allows images to be loaded automatically.

```
- (void)setLoadsImagesAutomatically:(BOOL)flag
```

Parameters

flag

YES if the web view should allow images to be loaded automatically, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadsImagesAutomatically](#) (page 102)
- [allowsAnimatedImages](#) (page 97)
- [setAllowsAnimatedImages:](#) (page 106)
- [allowsAnimatedImageLooping](#) (page 96)
- [setAutosaves:](#) (page 106)

Declared In

WebPreferences.h

setMinimumFontSize:

Sets the minimum font size of the web view.

```
- (void)setMinimumFontSize:(int)size
```

Parameters

size

The minimum font size to use for content regardless of the font size specified by that content.

Discussion

This method sets the minimum display font size for the web view, overriding all content-specified styles, including explicitly specified font sizes.

The font size specified by *size* should always be greater than zero.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [minimumFontSize](#) (page 103)

Declared In

WebPreferences.h

setMinimumLogicalFontSize:

Sets the minimum logical font size of the web view.

```
- (void)setMinimumLogicalFontSize:(int)size
```

Parameters

size

The minimum logical font size that the web view should use.

Discussion

The minimum logical font size is the smallest font size that will display in a web view when the content's font size is imprecisely specified. This includes content with logical sizes (such as `small`) or with a font size specified as a percentage of the default.

Most clients will not want to use this method; rather, explicitly set the minimum display font size using the [setMinimumFontSize:](#) (page 111) method.

The font size specified by *size* should always be greater than zero.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [minimumLogicalFontSize](#) (page 103)

Declared In

WebPreferences.h

setPluginsEnabled:

Sets whether or not the web view allows plug-ins.

```
- (void)setPluginsEnabled:(BOOL)flag
```

Parameters

flag

Set to YES if the web view should allow plug-ins, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [arePluginsEnabled](#) (page 97)

Declared In

WebPreferences.h

setPrivateBrowsingEnabled:

Sets whether or not private browsing is enabled.

```
- (void)setPrivateBrowsingEnabled:(BOOL)flag
```

Return Value

YES if private browsing should be enabled, otherwise NO.

Discussion

If private browsing is enabled, the web view does not store information about the websites the user visits. Private browsing prevents the web view from maintaining any history, cache, or AutoFill information for the pages being visited.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [privateBrowsingEnabled](#) (page 104)

Declared In

WebPreferences.h

setSansSerifFontFamily:

Sets the sans serif font family of the web view.

```
- (void)setSansSerifFontFamily:(NSString *)family
```

Parameters

family

The font family to use as the default for content marked with a sans-serif font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a sans-serif font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [sansSerifFontFamily](#) (page 104)

Declared In

WebPreferences.h

setSerifFontFamily:

Sets the serif font family of the web view.

```
- (void)setSerifFontFamily:(NSString *)family
```

Parameters

family

The font family to use as the default for content marked with a serif font.

Discussion

This preference affects text whose properties indicate that it should be displayed in a serif font but do not specify a particular font.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [serifFontFamily](#) (page 105)

Declared In

WebPreferences.h

setShouldPrintBackgrounds:

Sets whether or not the web view should include backgrounds when printing.

```
- (void)setShouldPrintBackgrounds:(BOOL)flag
```

Parameters

flag

Set to YES if the web view should print backgrounds, otherwise NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [shouldPrintBackgrounds](#) (page 116)

Declared In

WebPreferences.h

setStandardFontFamily:

Sets the standard font family of the web view.

```
- (void)setStandardFontFamily:(NSString *)family
```

Parameters

family

The font family to use as the default for content that does not specify a font.

Discussion

This preference affects text whose properties do not indicate that they should be displayed in a particular font or class of fonts (such as sans-serif).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [standardFontFamily](#) (page 117)

Declared In

WebPreferences.h

setTabsToLinks:

Sets whether or not the web view will focus control on links when tabbing.

- (void)setTabsToLinks:(BOOL)flag

Return Value

YES if the user is able to navigate to links by pressing the tab key, NO otherwise.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [tabsToLinks](#) (page 117)

Declared In

WebPreferences.h

setUserStyleSheetEnabled:

Sets whether or not user style sheets are enabled in the web view.

- (void)setUserStyleSheetEnabled:(BOOL)flag

Parameters

flag

Set to YES if the user style sheets should be enabled in the web view, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [userStyleSheetEnabled](#) (page 118)

Declared In

WebPreferences.h

setUserStyleSheetLocation:

Sets the location of the user style sheet.

```
- (void)setUserStyleSheetLocation:(NSURL *)URL
```

Parameters

URL

An NSURL object containing the location of the user style sheet.

Discussion

The user style sheet will override all existing CSS definitions on the page.

[setUserStyleSheetEnabled:](#) (page 115) must have already been set to YES for this method to have an effect.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [userStyleSheetLocation](#) (page 118)

Declared In

WebPreferences.h

setUsesPageCache:

Sets whether the web views associated with the receiver should use the shared page cache.

```
- (void)setUsesPageCache:(BOOL)usesPageCache
```

Parameters

usesPageCache

YES if the web views should use a page cache; otherwise, NO.

Discussion

Pages are cached when they are added to a back-forward list, and removed from the cache when they are removed from a back-forward list. Because the page cache is global, caching a page in one back-forward list may cause a page in another back-forward list to be removed from the cache.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [usesPageCache](#) (page 118)

Declared In

WebPreferences.h

shouldPrintBackgrounds

Returns whether the web view should include backgrounds when printing.

```
- (BOOL)shouldPrintBackgrounds
```

Return Value

YES if the web view prints backgrounds, otherwise NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setShouldPrintBackgrounds:](#) (page 114)

Declared In

WebPreferences.h

standardFontFamily

Returns the standard font family used by the web view.

- (NSString *)standardFontFamily

Return Value

The font family used as the default for content that does not specify a font.

Discussion

This preference affects text whose properties do not indicate that they should be displayed in a particular font or class of fonts (such as sans-serif).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setStandardFontFamily:](#) (page 114)

Declared In

WebPreferences.h

tabsToLinks

Returns whether or not the tab key will focus links.

- (BOOL)tabsToLinks

Return Value

YES if the user should be able to navigate to links by pressing the tab key, NO otherwise.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setTabsToLinks:](#) (page 115)

Declared In

WebPreferences.h

userStyleSheetEnabled

Returns whether user style sheets are enabled in the web view.

- (BOOL)userStyleSheetEnabled

Return Value

YES if the user style sheets are enabled in the web view, otherwise NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setUserStyleSheetEnabled:](#) (page 115)

Declared In

WebPreferences.h

userStyleSheetLocation

Returns the location of the user style sheet.

- (NSURL *)userStyleSheetLocation

Return Value

An NSURL object containing the location of the user style sheet.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setUserStyleSheetLocation:](#) (page 116)

Declared In

WebPreferences.h

usesPageCache

Returns whether the web views associated with the receiver should use the shared page cache.

- (BOOL)usesPageCache

Return Value

YES if the web views should use a page cache; otherwise, NO.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [setUsesPageCache:](#) (page 116)

Declared In

WebPreferences.h

Constants

WebCacheModel

Specifies the caching model for a web view.

```
enum {
    WebCacheModelDocumentViewer = 0,
    WebCacheModelDocumentBrowser = 1,
    WebCacheModelPrimaryWebBrowser = 2
};
typedef NSUInteger WebCacheModel;
```

Constants**WebCacheModelDocumentViewer**

Releases resources when they are no longer referenced and caches remote resources on disk. This model is appropriate for displaying a static document with no navigation user interface. This is the most memory-efficient model.

Available in Mac OS X v10.5 and later.

Declared in WebPreferences.h.

WebCacheModelDocumentBrowser

Caches a reasonable number of resources and previously viewed documents in memory and on disk. This model is appropriate for displaying and navigating between multiple documents.

Available in Mac OS X v10.5 and later.

Declared in WebPreferences.h.

WebCacheModelPrimaryWebBrowser

Caches a large number of resources and previously viewed documents in memory and on disk. This model is appropriate for a web view that behaves like a web browser.

Available in Mac OS X v10.5 and later.

Declared in WebPreferences.h.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebPreferences.h

Notifications

WebPreferencesChangedNotification

Posted when the web preference settings are changed.

The notification object is the WebPreferences object that changed. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebPreferences.h

WebResource Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebResource.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

A `WebResource` object represents a downloaded URL. It encapsulates the data of the download as well as other resource properties such as the URL, MIME type, and frame name.

Use the `initWithData:URL:MIMETYPE:textEncodingName:frameName:` (page 122) method to initialize a newly created `WebResource` object. Use the other methods in this class to get the properties of a `WebResource` object.

Tasks

Initializing

- `initWithData:URL:MIMETYPE:textEncodingName:frameName:` (page 122)
Initializes and returns a web resource instance.

Getting Attributes

- `data` (page 122)
Returns the receiver's data.
- `URL` (page 124)
Returns the receiver's URL.

- [MIMETYPE](#) (page 123)
Returns the receiver's MIME type.
- [textEncodingName](#) (page 123)
Returns the receiver's text encoding name.
- [frameName](#) (page 122)
Returns the receiver's frame name.

Instance Methods

data

Returns the receiver's data.

```
- (NSData *)data
```

Return Value

The download data.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebResource.h

frameName

Returns the receiver's frame name.

```
- (NSString *)frameName
```

Return Value

The name of the frame. If the receiver does not represent the contents of an entire HTML frame, this method returns `nil`.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebResource.h

initWithData:URL:MIMETYPE:textEncodingName:frameName:

Initializes and returns a web resource instance.

```
- (id)initWithData:(NSData *)dataURL:(NSURL *)URLMIMETYPE:(NSString *)MIMETYPEtextEncodingName:(NSString *)textEncodingNameframeName:(NSString *)frameName
```

Parameters*data*

The download data.

URL

The download URL.

MIMETYPE

The MIME type of the data.

*textEncodingName*The IANA encoding name (for example, “utf-8” or “utf-16”). This parameter may be `nil`.*frameName*The name of the frame. Use this parameter if the resource represents the contents of an entire HTML frame; otherwise pass `nil`.**Return Value**

An initialized web resource.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In`WebResource.h`**MIMETYPE**

Returns the receiver’s MIME type.

- (NSString *)MIMETYPE

Return Value

The MIME type of the data.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In`WebResource.h`**textEncodingName**

Returns the receiver’s text encoding name.

- (NSString *)textEncodingName

Return ValueThe IANA encoding name (for example, “utf-8” or “utf-16”), or `nil` if the name does not exist.**Availability**

Available in Mac OS X v10.3.9 and later.

Declared In`WebResource.h`

URL

Returns the receiver's URL.

- (NSURL *)URL

Return Value

The download URL.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebResource.h

WebScriptObject Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebScriptObject.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide
Related sample code	CallJS Reminders

Overview

A `WebScriptObject` object is an Objective-C wrapper for a scripting object passed to your application from the scripting environment.

You can not create a `WebScriptObject` object directly. You get a window `WebScriptObject` object by sending `windowScriptObject` (page 197) to your `WebView` object.

You can use key-value coding methods—for example, `setValue:forKey:` and `valueForKey:`—to get and set properties of a `WebScriptObject` object. You can also access properties by index using the `setWebScriptValueAtIndex:value:` (page 129) and `webScriptValueAtIndex:` (page 130) methods. Use the `removeWebScriptKey:` (page 128) method to remove a scripting object property.

Not all properties and methods of a class are exported. Use the `setValue:forUndefinedKey:` and `valueForUndefinedKey:` methods to intercept access to properties that are not exported. Similarly, use the `invokeUndefinedMethodFromWebScript:withArguments:` method to intercept method invocations that are not exported.

If you want access to properties and methods defined in your own classes, use the methods in the `WebScripting` informal protocol to specify the properties and methods the class should export to WebKit's JavaScript environment.

Use the `callWebScriptMethod:withArguments:` (page 127) and `evaluateWebScript:` (page 127) methods to execute scripts in the scripting environment.

Tasks

Getting and Setting Properties

- [JSObject](#) (page 128)
Returns the JavaScript object corresponding to the receiver.
- [removeWebScriptKey:](#) (page 128)
Removes a property from a scripting environment.
- [webScriptValueAtIndex:](#) (page 130)
Returns the value of a property at the specified index.
- [setWebScriptValueAtIndex:value:](#) (page 129)
Sets the value of a property at the specified index.

Executing Scripts

- [callWebScriptMethod:withArguments:](#) (page 127)
Returns the result of executing a method in the scripting environment.
- [evaluateWebScript:](#) (page 127)
Returns the result of evaluating a script in the scripting environment.

Raising Exceptions

- + [throwException:](#) (page 126)
Raises an exception in the current script execution context.
- [setException:](#) (page 129)
Raises a scripting environment exception in the context of the current object.

Getting a String Representation

- [stringRepresentation](#) (page 129)
Returns a string representation of the receiver.

Class Methods

throwException:

Raises an exception in the current script execution context.

+ (BOOL)throwException:(NSString *)exceptionMessage

Parameters

exceptionMessage

The exception message.

Return Value

YES if successful, NO otherwise.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setException:](#) (page 129)

Declared In

WebScriptObject.h

Instance Methods

callWebScriptMethod:withArguments:

Returns the result of executing a method in the scripting environment.

```
- (id)callWebScriptMethod:(NSString *)namewithArguments:(NSArray *)args
```

Parameters

name

The name of the method to invoke.

args

The values to pass to the method.

Return Value

The return value of the method. Returns `WebUndefined` if an exception is thrown in the JavaScript environment or the method has no return value.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [evaluateWebScript:](#) (page 127)

Related Sample Code

QT Capture Widget

WebKitPluginWithJavaScript

Declared In

WebScriptObject.h

evaluateWebScript:

Returns the result of evaluating a script in the scripting environment.

```
- (id)evaluateWebScript:(NSString *)script
```

Parameters

script

The script to evaluate.

Return Value

The scripting object. The format of the script is dependent on the target scripting environment. Returns `WebUndefined` if an exception is thrown in the JavaScript environment or there is no return value.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [callWebScriptMethod:withArguments:](#) (page 127)

Related Sample Code

Reminders

Declared In

WebScriptObject.h

JSObject

Returns the JavaScript object corresponding to the receiver.

```
- (JSObjectRef)JSObject
```

Return Value

The JavaScript object corresponding to the receiver in the JavaScriptCore C API.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebScriptObject.h

removeWebScriptKey:

Removes a property from a scripting environment.

```
- (void)removeWebScriptKey:(NSString *)name
```

Parameters

name

Property to remove.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webScriptValueAtIndex:](#) (page 130)

- [setWebScriptValueAtIndex:value:](#) (page 129)

Declared In

WebScriptObject.h

setException:

Raises a scripting environment exception in the context of the current object.

```
- (void)setException:(NSString *)description
```

Parameters

description

Description of the exception.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

+ [throwException:](#) (page 126)

Declared In

WebScriptObject.h

setWebScriptValueAtIndex:value:

Sets the value of a property at the specified index.

```
- (void)setWebScriptValueAtIndex:(unsigned) index value:(id) value
```

Parameters

index

The index of the property.

value

The value of the property.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [removeWebScriptKey:](#) (page 128)

- [webScriptValueAtIndex:](#) (page 130)

Declared In

WebScriptObject.h

stringRepresentation

Returns a string representation of the receiver.

```
- (NSString *)stringRepresentation
```

Return Value

The string representation of the receiver.

Discussion

The coercion of nonstring objects is dependent on the scripting environment.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

webScriptValueAtIndex:

Returns the value of a property at the specified index.

- (id)webScriptValueAtIndex:(unsigned) *index*

Parameters

index

The index of the property.

Return Value

The value of a property at *index*. Returns `WebUndefined` if an exception is thrown in the JavaScript environment.

Discussion

Accessing property values by index is dependent on the scripting environment.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [removeWebScriptKey:](#) (page 128)
- [setWebScriptValueAtIndex:value:](#) (page 129)

Declared In

WebScriptObject.h

WebUndefined Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebScriptObject.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

`WebUndefined` objects are simply used to represent the JavaScript “undefined” value in methods when bridging between JavaScript and Objective-C. For example, if you invoke a JavaScript function that returns the JavaScript “undefined” value, then a `WebUndefined` object is returned to the Objective-C calling context.

Tasks

Getting the Shared Instance

+ `undefined` (page 131)

Returns the shared `WebUndefined` instance.

Class Methods

undefined

Returns the shared `WebUndefined` instance.

+ `(WebUndefined *)undefined`

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

WebView Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebView.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide
Related sample code	CallJS iChatTheater NewsReader SourceView SpecialPictureProtocol

Overview

`WebView` is the core view class in the WebKit framework that manages interactions between the `WebFrame` and `WebFrameView` classes. To embed web content in your application, you just create a `WebView` object, attach it to a window, and send a `loadRequest:` (page 62) message to its main frame.

Behind the scenes, `WebFrame` objects encapsulate the content contained in a single frame element. A hierarchy of `WebFrame` objects is used to model an entire webpage where the root is called the **main frame**. There is a `WebFrameView` object per `WebFrame` object used to display the frame content. Therefore, there is a parallel hierarchy of `WebFrameView` objects used to render an entire page. The `WebView` object is also the parent view of this hierarchy. You do not need to create `WebFrame` and `WebFrameView` objects directly. These objects are automatically created when the page loads, either programmatically or by the user clicking a link.

You customize your embedded web content by implementing `WebView` delegates to handle certain aspects of the process. `WebView` objects have multiple delegates because the process of loading a webpage is asynchronous and complicated if errors occur. All the `WebView` delegates use informal protocols so you only need to implement only the delegates and methods that define the behavior you wish to change—default implementations are already provided.

For example, you might want to implement the frame load and resource load delegates to monitor the load progress and display status messages. Applications that use multiple windows may want to implement a user interface delegate. See the individual informal delegate protocols for more details: *WebFrameLoadDelegate Protocol Reference*, *WebPolicyDelegate Protocol Reference*, *WebResourceLoadDelegate Protocol Reference*, and *WebUIDelegate Protocol Reference*.

Another way to monitor load progress with less control is to observe the [WebViewProgressEstimateChangedNotification](#) (page 201), [WebViewProgressFinishedNotification](#) (page 202), and [WebViewProgressStartedNotification](#) (page 202) notifications. For example, you could observe these notifications to implement a simple progress indicator in your application. You update the progress indicator by invoking the `estimatedProgress` method to get an estimate of the amount of content that is currently loaded.

A `WebView` object is intended to support most features you would expect in a web browser except that it doesn't implement the specific user interface for those features. You are responsible for implementing the user interface objects such as status bars, toolbars, buttons, and text fields. For example, a `WebView` object manages a back-forward list by default, and has [goBack:](#) (page 161) and [goForward:](#) (page 162) action methods. It is your responsibility to create the buttons that would send these action messages. Note, there is some overhead in maintaining a back-forward list and page cache, so you should disable it if your application doesn't use it.

You use a `WebPreferences` object to encapsulate the preferences of a `WebView` object, such as the font, text encoding, and image settings. You can modify the preferences for individual `WebView` objects or specify a shared `WebPreferences` object using the [setPreferencesIdentifier:](#) (page 186) method. Use the [setAutosaves:](#) (page 106) `WebPreferences` method to specify whether the preferences should be automatically saved to the user defaults database.

You can also extend `WebKit` by implementing your own document view and representation classes for specific MIME types. Use the [registerViewClass:representationClass:forMIMEType:](#) (page 145) class method to register your custom classes with a `WebView` object.

Tasks

Registering Document Views and Representations

- + [registerURLSchemeAsLocal:](#) (page 144)
Adds the specified URL scheme to the list of local schemes.
- + [registerViewClass:representationClass:forMIMEType:](#) (page 145)
Specifies the view and representation objects to be used for specific MIME types.

Initializing Views

- [initWithFrame:frameName:groupName:](#) (page 163)
Initializes the receiver with a frame rectangle, frame name, and group name.

Closing the View

- [close](#) (page 154)
Closes the web view when it's no longer needed.
- [shouldCloseWithWindow](#) (page 190)
Returns whether the web view should close when its window or host window closes.
- [setShouldCloseWithWindow:](#) (page 188)
Sets whether the web view should close when its window or host window closes.

Getting the Main Frame

- [mainFrame](#) (page 165)
Returns the main frame, the root of the web frame hierarchy for this page.

Loading Content

- [stopLoading:](#) (page 192)
An action method that stops the loading of any web frame content managed by the receiver.
- [takeStringURLFrom:](#) (page 194)
Sets the receiver's current location by obtaining a URL string from the sender.
- [reload:](#) (page 174)
An action method that reloads the current page.
- [reloadFromOrigin:](#) (page 174)
Action method that performs an end-to-end revalidation using cache-validating conditionals if possible.
- [estimatedProgress](#) (page 159)
Returns an estimate, as a percentage, of the amount of content that is currently loaded.

Drawing

- [drawsBackground](#) (page 158)
Returns whether the web view draws a background.
- [setDrawsBackground:](#) (page 182)
Sets whether a default background is drawn when the webpage has no background set.
- [setShouldUpdateWhileOffscreen:](#) (page 188)
Sets whether the web view should update even when it is not in a window that is currently visible.
- [shouldUpdateWhileOffscreen](#) (page 190)
Returns whether the web view should update even when it is not visible.

Moving Back and Forward

- [setMaintainsBackForwardList:](#) (page 185)
Sets whether to use a back-forward list.

- [backForwardList](#) (page 149)
Returns the receiver's back-forward list.
- [canGoBack](#) (page 149)
Returns whether the previous location can be loaded.
- [goBack](#) (page 160)
Loads the previous location in the back-forward list.
- [goBack:](#) (page 161)
An action method that loads the previous location in the back-forward list.
- [canGoForward](#) (page 150)
Returns whether the next location can be loaded.
- [goForward](#) (page 161)
Loads the next location in the back-forward list.
- [goForward:](#) (page 162)
An action method that loads the next location in the back-forward list.
- [goToBackForwardItem:](#) (page 162)
Loads a specific location from the back-forward list and sets it as the current item.

Changing the Text Size

- [canMakeTextLarger](#) (page 150)
Returns whether the text can be made larger.
- [makeTextLarger:](#) (page 167)
Action method that increases the text size by one unit.
- [canMakeTextSmaller](#) (page 151)
Returns whether the text can be made smaller.
- [makeTextSmaller:](#) (page 167)
Action method that reduces the text size by one unit.

Getting and Setting Delegates

- [downloadDelegate](#) (page 158)
Return the receiver's download delegate.
- [setDownloadDelegate:](#) (page 181)
Sets the receiver's shared download delegate.
- [frameLoadDelegate](#) (page 160)
Return the receiver's frame load delegate.
- [setFrameLoadDelegate:](#) (page 183)
Sets the receiver's frame load delegate.
- [policyDelegate](#) (page 173)
Returns the receiver's policy delegate.
- [setPolicyDelegate:](#) (page 185)
Sets the receiver's policy delegate.

- [resourceLoadDelegate](#) (page 177)
Returns the receiver's resource load delegate.
- [setResourceLoadDelegate:](#) (page 187)
Sets the receiver's resource load delegate.
- [UIDelegate](#) (page 196)
Returns the receiver's user interface delegate.
- [setUIDelegate:](#) (page 190)
Sets the receiver's user interface delegate.

Getting and Setting the Window

- [hostWindow](#) (page 163)
Returns the receiver's host window.
- [setHostWindow:](#) (page 184)
Sets the receiver's host window.

Getting and Setting Preferences

- [preferences](#) (page 173)
Returns the receiver's preferences.
- [setPreferences:](#) (page 186)
Sets the receiver's preferences.
- [preferencesIdentifier](#) (page 173)
Returns the identifier of the receiver's preferences.
- [setPreferencesIdentifier:](#) (page 186)
Sets the receiver's preferences identifier, creating a preferences object if needed.

Getting and Setting Frame Contents

- [isLoading](#) (page 165)
Returns whether the web view is loading content.
- [selectedFrame](#) (page 178)
Returns the frame with the active selection.
- [setMainFrameURL:](#) (page 184)
Sets the URL that the main frame loads.
- [mainFrameURL](#) (page 166)
Returns the URL that the main frame loads.
- [mainFrameTitle](#) (page 166)
Returns the HTML title of the loaded page.
- [mainFrameIcon](#) (page 166)
Returns the site's favicon.
- [mainFrameDocument](#) (page 165)
Returns the DOM document for the main frame.

Getting and Setting Content Information

- + `canShowMIMETYPE:` (page 143)
Returns whether the receiver can display content of a given MIME type.
- + `MIMETypesShownAsHTML` (page 144)
Returns a list of MIME types that WebKit renders as HTML.
- + `setMIMETypesShownAsHTML:` (page 145)
Sets the MIME types that WebKit attempts to render as HTML.
- + `canShowMIMETYPEAsHTML:` (page 143)
Returns whether the receiver interprets a MIME type as HTML.
- `supportsTextEncoding` (page 194)
Returns whether the document view supports different text encodings.
- `customTextEncodingName` (page 155)
Returns the custom text encoding name.
- `setCustomTextEncodingName:` (page 180)
Sets the custom text encoding name.
- `textSizeMultiplier` (page 195)
Returns the font size multiplier for text displayed in web frame view objects managed by the receiver.
- `setTextSizeMultiplier:` (page 189)
Change the font size multiplier for text displayed in web frame view objects managed by the receiver.

Searching the Document

- `searchFor:direction:caseSensitive:wrap:` (page 177)
Searches a document view for a string and highlights it if it is found.

Getting and Setting the Group Name

- `groupName` (page 162)
Returns the receiver's group name.
- `setGroupName:` (page 183)
Sets the receiver's group name.

Getting and Setting User-agent Strings

- `userAgentForURL:` (page 197)
Returns the appropriate user-agent string for a given URL.
- `applicationNameForUserAgent` (page 148)
Returns the receiver's application name that is used in the user-agent string.
- `setApplicationNameForUserAgent:` (page 179)
Sets the application name used in the user-agent string.
- `customUserAgent` (page 156)
Returns the receiver's custom user-agent string.

- [setCustomUserAgent:](#) (page 181)
Sets the receiver's custom user-agent string.

Processing JavaScript

- [stringByEvaluatingJavaScriptFromString:](#) (page 193)
Returns the result of running a script.

Using the Pasteboard

- + [URLFromPasteboard:](#) (page 145)
Returns a URL from the specified pasteboard.
- + [URLTitleFromPasteboard:](#) (page 146)
Returns the title of a URL from the specified pasteboard.
- [pasteboardTypesForElement:](#) (page 171)
Returns an array of pasteboard types for an element.
- [pasteboardTypesForSelection](#) (page 171)
Returns an array of pasteboard types that can be used for the current selection of the receiver.
- [writeElement:withPasteboardTypes:toPasteboard:](#) (page 198)
Writes an element to the pasteboard using a list of types.
- [writeSelectionWithPasteboardTypes:toPasteboard:](#) (page 198)
Writes the receiver's current selection to a pasteboard using a list of types.

Dragging

- [elementAtPoint:](#) (page 159)
Returns a dictionary description of the element at a given point in the receiver's coordinates.
- [moveDragCaretToPoint:](#) (page 168)
Moves the drag caret that indicates the destination of a drag operation to a given point.
- [removeDragCaret](#) (page 174)
Removes the drag caret that indicates the destination of a drag operation.

Cut, Copy and Paste Action Methods

- [copy:](#) (page 155)
Action method that copies the selected content to the general pasteboard.
- [copyFont:](#) (page 155)
An action method that copies font information onto the font pasteboard.
- [cut:](#) (page 156)
An action method that deletes selected content and puts it on the general pasteboard.
- [delete:](#) (page 157)
An action method that deletes the selected content.

- [paste:](#) (page 170)
An action method that pastes content from the pasteboard at the insertion point or over the selection.
- [pasteFont:](#) (page 172)
An action method that pastes font information from the font pasteboard.
- [pasteAsPlainText:](#) (page 170)
An action method that pastes pasteboard content as plain text.
- [pasteAsRichText:](#) (page 171)
An action method that pastes pasteboard content into the receiver as rich text, maintaining its attributes.

Content Alignment Action Methods

- [alignCenter:](#) (page 146)
An action method that applies center alignment to selected content or all content if there's no selection.
- [alignJustified:](#) (page 147)
An action method that applies full justification to selected content or all content if there's no selection.
- [alignLeft:](#) (page 147)
An action method that applies left justification to selected content or all content if there's no selection.
- [alignRight:](#) (page 148)
An action method that applies right justification to selected content or all content if there is no selection.

Changing the Font, Color and Other Attributes When Editing

- [changeFont:](#) (page 153)
An action method that changes the font of the selection, or all content if there is no selection.
- [changeAttributes:](#) (page 151)
An action method that changes the attributes of the current selection.
- [changeDocumentBackgroundColor:](#) (page 152)
Sets the background color of the selected content.
- [changeColor:](#) (page 152)
Sets the color of the selected content.

Spell-checking Action Methods

- [checkSpelling:](#) (page 153)
An action method that searches for a misspelled word in the receiver.
- [showGuessPanel:](#) (page 191)
An action method that shows a spelling correction panel.

Find Panel Action Method

- [performFindPanelAction:](#) (page 172)
An action method that opens the Find menu and Find panel.

Controlling Speakable Text

- [startSpeaking:](#) (page 192)
An action method that starts speaking the selected text or all text if there's no selection.
- [stopSpeaking:](#) (page 193)
An action method that stops speaking that is in progress.

Getting and Setting Document Editing Attributes

- [isEditable](#) (page 164)
Returns whether the user is allowed to edit the document.
- [setEditable:](#) (page 182)
Sets whether the receiver allows the user to edit its HTML document.
- [smartInsertDeleteEnabled](#) (page 191)
Returns whether smart-space insertion and deletion is enabled.
- [setSmartInsertDeleteEnabled:](#) (page 188)
Sets whether the receiver should insert or delete spaces around selected words to preserve proper spacing and punctuation.
- [isContinuousSpellCheckingEnabled](#) (page 164)
Returns whether the web view has continuous spell-checking enabled.
- [setContinuousSpellCheckingEnabled:](#) (page 180)
Sets whether the web view has continuous spell-checking enabled.
- [spellCheckerDocumentTag](#) (page 191)
Returns the spell-checker document tag for this document.
- [undoManager](#) (page 196)
Returns the receiver's undo manager.
- [editingDelegate](#) (page 159)
Returns the receiver's editing delegate.
- [setEditingDelegate:](#) (page 182)
Sets the receiver's editing delegate.
- [editableDOMRangeForPoint:](#) (page 158)
Returns the editable DOM object located at a given point.

Editing Documents

- [replaceSelectionWithNode:](#) (page 176)
Replaces the receiver's current selection with the specified DOM node.

- [replaceSelectionWithText:](#) (page 176)
Replaces the current selection with a string of text.
- [replaceSelectionWithMarkupString:](#) (page 175)
Replaces the current selection with mixed text and markup.
- [replaceSelectionWithArchive:](#) (page 175)
Replaces the current selection with an archive's contents.
- [deleteSelection](#) (page 157)
Deletes the receiver's current selection unless it's collapsed.
- [moveToBeginningOfSentence:](#) (page 169)
Moves the insertion point to the beginning of the current sentence.
- [moveToBeginningOfSentenceAndModifySelection:](#) (page 169)
Moves the insertion point and extends the selection to the beginning of the current sentence.
- [moveToEndOfSentence:](#) (page 169)
Moves the insertion point to the end of the current sentence.
- [moveToEndOfSentenceAndModifySelection:](#) (page 170)
Moves the insertion point and extends the selection to the end of the current sentence.
- [selectSentence:](#) (page 179)
Selects the entire sentence around the insertion point.
- [toggleContinuousSpellChecking:](#) (page 195)
Toggles whether continuous spell checking is available.
- [toggleSmartInsertDelete:](#) (page 195)
Toggles whether spaces around selected words are inserted or deleted to preserve proper spacing and punctuation.
- [canMakeTextStandardSize](#) (page 151)
Returns whether the current text size is a multiple of 1.
- [makeTextStandardSize:](#) (page 168)
Resets the text size to a multiple of 1.
- [maintainsInactiveSelection](#) (page 166)
Returns whether the selection is maintained when focus is lost.

Selecting Content in the Document

- [selectedDOMRange](#) (page 178)
Returns the range of the current selection.
- [setSelectedDOMRange:affinity:](#) (page 187)
Selects a range of nodes.
- [selectionAffinity](#) (page 179)
Returns the current selection affinity.

Getting and Setting CSS Properties

- [computedStyleForElement:pseudoElement:](#) (page 154)
Returns the computed style of an element and its pseudo element.

- [mediaStyle](#) (page 168)
Returns the receiver's CSS media property.
- [setMediaStyle:](#) (page 185)
Sets the receiver's CSS media property.
- [typingStyle](#) (page 196)
Returns the receiver's CSS typing style.
- [setTypingStyle:](#) (page 189)
Sets the receiver's CSS typing style.
- [styleDeclarationWithText:](#) (page 194)
Returns the CSS style declaration for the specified text.
- [applyStyle:](#) (page 149)
Applies the CSS typing style to the current selection.

Using WebScript

- [windowScriptObject](#) (page 197)
Returns the receiver's window object from the scripting environment.

Class Methods

canShowMIMETYPE:

Returns whether the receiver can display content of a given MIME type.

```
+ (BOOL)canShowMIMETYPE:(NSString *)MIMETYPE
```

Parameters

MIMETYPE

The MIME type of the content.

Return Value

YES if the receiver can display content of the specified MIME type where *MIMETYPE* is one of the standard types like "image/gif"; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebView.h

canShowMIMETYPEAsHTML:

Returns whether the receiver interprets a MIME type as HTML.

```
+ (BOOL)canShowMIMETYPEAsHTML:(NSString *)MIMETYPE
```

Parameters*MIMEType*

The MIME type of the content.

Return ValueYES if the receiver interprets *MIMEType* as HTML; otherwise, NO.**Availability**

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebView.h

MIMETypesShownAsHTML

Returns a list of MIME types that WebKit renders as HTML.

+ (NSArray *)MIMETypesShownAsHTML

Return Value

An array containing NSString objects that represent the MIME types WebKit attempts to render as HTML.

Availability

Available in Mac OS X v10.3.9 and later.

See Also+ [setMIMETypesShownAsHTML:](#) (page 145)**Declared In**

WebView.h

registerURLSchemeAsLocal:

Adds the specified URL scheme to the list of local schemes.

+ (void)registerURLSchemeAsLocal:(NSString *)*scheme***Parameters***scheme*

The scheme to add to the list.

Discussion

You need to register a scheme as local to access resources with file URLs and to have the same security checks as a local file.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

registerViewClass:representationClass:forMIMEType:

Specifies the view and representation objects to be used for specific MIME types.

```
+ (void)registerViewClass:(Class)viewClass
    representationClass:(Class)representationClass forMIMEType:(NSString *)MIMEType
```

Parameters

viewClass

A class conforming to the `WebDocumentView` protocol that displays the specified MIME types.

representationClass

The class conforming to `WebDocumentRepresentation` protocol that represents the specified MIME types.

MIMEType

The MIME type of the content.

This may be a primary MIME type or subtype. For example, if *MIMEType* is “video/” the specified view and representation objects are used for all video types. More specific subtype mappings, such as “image/gif”, takes precedence over primary type matching, such as “image/”.

Discussion

After invoking this method, when *MIMEType* content is encountered, instances of *representationClass* and *viewClass* are created to handle and display it.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebView.h

setMIMETypesShownAsHTML:

Sets the MIME types that WebKit attempts to render as HTML.

```
+ (void)setMIMETypesShownAsHTML:(NSArray *)MIMETypes
```

Parameters

MIMETypes

An array of `NSString` objects representing the MIME types. Typically, you create the *MIMETypes* array by adding additional types to the array returned by the `MIMETypesShownAsHTML` (page 144) class method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

URLFromPasteboard:

Returns a URL from the specified pasteboard.

```
+ (NSURL *)URLFromPasteboard:(NSPasteboard *)pasteboard
```

Parameters

pasteboard

The pasteboard containing a URL.

Return Value

The URL from the specified pasteboard or `nil` if there's no URL on *pasteboard*.

Discussion

This method supports multiple pasteboard types including `NSRULPboardType`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

+ [URLTitleFromPasteboard:](#) (page 146)

Declared In

WebView.h

URLTitleFromPasteboard:

Returns the title of a URL from the specified pasteboard.

```
+ (NSString *)URLTitleFromPasteboard:(NSPasteboard *)pasteboard
```

Parameters

pasteboard

The pasteboard containing the URL.

Return Value

The title of the URL on *pasteboard*. Returns `nil` if there's no URL on *pasteboard* or the URL has no title.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

+ [URLFromPasteboard:](#) (page 145)

Declared In

WebView.h

Instance Methods

alignCenter:

An action method that applies center alignment to selected content or all content if there's no selection.

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

Parameters*sender*

The object that sent this message.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [alignJustified:](#) (page 147)
- [alignLeft:](#) (page 147)
- [alignRight:](#) (page 148)

Declared In

WebView.h

alignJustified:

An action method that applies full justification to selected content or all content if there's no selection.

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

Parameters*sender*

The object that sent this message.

Discussion

This method behaves similar to the `alignJustified:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [alignJustified:](#) (page 147)
- [alignLeft:](#) (page 147)
- [alignRight:](#) (page 148)

Declared In

WebView.h

alignLeft:

An action method that applies left justification to selected content or all content if there's no selection.

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

Parameters*sender*

The object that sent this message.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [alignJustified](#): (page 147)
- [alignCenter](#): (page 146)
- [alignRight](#): (page 148)

Declared In

WebView.h

alignRight:

An action method that applies right justification to selected content or all content if there is no selection.

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

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [alignJustified](#): (page 147)
- [alignLeft](#): (page 147)
- [alignCenter](#): (page 146)

Declared In

WebView.h

applicationNameForUserAgent

Returns the receiver's application name that is used in the user-agent string.

```
- (NSString *)applicationNameForUserAgent
```

Return Value

The application name to use in the user-agent string. The user-agent is used by websites to identify the client browser.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setApplicationNameForUserAgent](#): (page 179)
- [setCustomUserAgent](#): (page 181)
- [customUserAgent](#) (page 156)

Declared In

WebView.h

applyStyle:

Applies the CSS typing style to the current selection.

```
- (void)applyStyle:(DOMCSSStyleDeclaration *)style
```

Parameters

style

The style to apply to the current selection.

Discussion

This method does nothing if there is no current selection or if the current selection is collapsed.

This method hides the complexities of applying styles to elements. If necessary, this method will make multiple passes over the range of the current selection to ensure that the requested style is applied to the elements in that range, and takes into account the complexities of CSS style application rules. This method also simplifies styling attributes so that the minimum number of styling directives are used to yield a given computed style.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setTypingStyle:](#) (page 189)
- [typingStyle](#) (page 196)
- [computedStyleForElement:pseudoElement:](#) (page 154)

Declared In

WebView.h

backForwardList

Returns the receiver's back-forward list.

```
- (WebBackForwardList *)backForwardList
```

Return Value

The receiver's back-forward list.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goBack](#) (page 160)
- [goForward](#) (page 161)
- [goToBackForwardItem:](#) (page 162)

Declared In

WebView.h

canGoBack

Returns whether the previous location can be loaded.

- (BOOL)canGoBack

Return Value

YES if able to move backward; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goBack:](#) (page 161)
- [canGoForward](#) (page 150)
- [goForward:](#) (page 162)

Declared In

WebView.h

canGoForward

Returns whether the next location can be loaded.

- (BOOL)canGoForward

Return Value

YES if able to move forward; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goForward:](#) (page 162)
- [canGoBack](#) (page 149)
- [goBack:](#) (page 161)

Declared In

WebView.h

canMakeTextLarger

Returns whether the text can be made larger.

- (BOOL)canMakeTextLarger

Return Value

YES if able to make the text larger; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [makeTextLarger](#): (page 167)
- [makeTextSmaller](#): (page 167)
- [canMakeTextSmaller](#) (page 151)

Declared In

WebView.h

canMakeTextSmaller

Returns whether the text can be made smaller.

- (BOOL)canMakeTextSmaller

Return Value

YES if able to make the text smaller; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [makeTextSmaller](#): (page 167)
- [canMakeTextLarger](#) (page 150)
- [makeTextLarger](#): (page 167)

Declared In

WebView.h

canMakeTextStandardSize

Returns whether the current text size is a multiple of 1.

- (BOOL)canMakeTextStandardSize

Return Value

YES if the current text size is a multiple of 1; otherwise, NO.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

changeAttributes:

An action method that changes the attributes of the current selection.

- (void)changeAttributes:(id)sender

Parameters*sender*

The object that sent this message.

Discussion

This method behaves similar to the `changeAttributes:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [changeFont:](#) (page 153)

Declared In

WebView.h

changeColor:

Sets the color of the selected content.

- (void)changeColor:(id)sender

Parameters*sender*

The object that sent this message.

Discussion

This method is invoked by the `NSColorPanel` sender and behaves similar to the `changeColor:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [changeDocumentBackgroundColor:](#) (page 152)

Declared In

WebView.h

changeDocumentBackgroundColor:

Sets the background color of the selected content.

- (void)changeDocumentBackgroundColor:(id)sender

Parameters*sender*

The object that sent this message.

Discussion

This method is invoked by the `NSColorPanel` sender and behaves similar to the `changeDocumentBackgroundColor:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [changeColor:](#) (page 152)

Declared In

WebView.h

changeFont:

An action method that changes the font of the selection, or all content if there is no selection.

- (void)changeFont:(id)sender

Parameters

sender

The object that sent this message.

Discussion

If the receiver doesn't use the Fonts panel, this method does nothing.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [changeAttributes:](#) (page 151)

Declared In

WebView.h

checkSpelling:

An action method that searches for a misspelled word in the receiver.

- (void)checkSpelling:(id)sender

Parameters

sender

The object that sent this message.

Discussion

This action method starts a search at the end of the selection and continues until it reaches a word suspected of being misspelled or the end of the content. If a word isn't recognized by the spelling server, a [showGuessPanel:](#) (page 191) message is sent to the receiver which opens the Guess panel and allows the user to make a correction or add the word to the local dictionary.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [showGuessPanel:](#) (page 191)

Declared In

WebView.h

close

Closes the web view when it's no longer needed.

```
- (void)close
```

Discussion

Closes the web view by unloading its webpage and canceling any pending load requests. A closed web view no longer responds to new requests nor sends delegate messages. If the application uses garbage collection, this method needs to be invoked before an instance is collected. It is invoked automatically if the receiver's enclosing window or host window is closed and sending [shouldCloseWithWindow](#) (page 190) to the receiver returns YES. Applications that do not use garbage collection can still use this method to stop the receiver from loading and sending delegate messages.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

computedStyleForElement:pseudoElement:

Returns the computed style of an element and its pseudo element.

```
- (DOMCSSStyleDeclaration *)computedStyleForElement:(DOMElement *)element
    pseudoElement:(NSString *)pseudoElement
```

Parameters

element

The element whose computed style is returned.

pseudoElement

The pseudo element for *element*.

Return Value

An immutable object describing the computed style of *element* and *pseudoElement* according to the Cascading Style Sheets Specification at <http://www.w3.org/TR/CSS21>. Returns nil if the receiver doesn't display *element*.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [typingStyle](#) (page 196)
- [setTypingStyle:](#) (page 189)
- [applyStyle:](#) (page 149)

Declared In

WebView.h

copy:

Action method that copies the selected content to the general pasteboard.

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

Parameters

sender

The object that sent this message.

Discussion

This action method copies the selected content onto the general pasteboard, in as many formats as the receiver supports. For example, a plain text object uses `NSStringPboardType` for plain text, and a rich text object also uses `NSRTFPboardType`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [cut:](#) (page 156)
- [paste:](#) (page 170)

Declared In

WebView.h

copyFont:

An action method that copies font information onto the font pasteboard.

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

Parameters

sender

The object that sent this message.

Discussion

This action method copies the font information for the first character of the selection (or for the insertion point) onto the font pasteboard as `NSFontPboardType`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [pasteFont:](#) (page 172)

Declared In

WebView.h

customTextEncodingName

Returns the custom text encoding name.

```
- (NSString *)customTextEncodingName
```

Return Value

The receiver's custom text encoding name or `nil` if no custom text encoding name was set.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setCustomTextEncodingName:](#) (page 180)

Declared In

WebView.h

customUserAgent

Returns the receiver's custom user-agent string.

- (NSString *)customUserAgent

Return Value

The user-agent string to identify the client browser. The custom user-agent string is used for all URLs.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setApplicationNameForUserAgent:](#) (page 179)

- [applicationNameForUserAgent](#) (page 148)

- [customUserAgent](#) (page 156)

Declared In

WebView.h

cut:

An action method that deletes selected content and puts it on the general pasteboard.

- (void)cut:(id)sender

Parameters

sender

The object that sent this message.

Discussion

This action method deletes the selected content and places it onto the general pasteboard, in as many formats as the receiver supports. For example, a plain text object uses `NSStringPboardType` for plain text, and a rich text object also uses `NSRTFPboardType`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [copy](#): (page 155)
- [paste](#): (page 170)

Declared In

WebView.h

delete:

An action method that deletes the selected content.

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

Parameters

sender

The object that sent this message.

Discussion

The pasteboard is unaffected by invoking this method.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [cut](#): (page 156)

Declared In

WebView.h

deleteSelection

Deletes the receiver's current selection unless it's collapsed.

```
- (void)deleteSelection
```

Discussion

No content is removed if the current selection is collapsed (a range is selected with the same nodes and offsets for the start and end) or if there is no current selection.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [replaceSelectionWithText](#): (page 176)
- [replaceSelectionWithMarkupString](#): (page 175)
- [replaceSelectionWithArchive](#): (page 175)
- [replaceSelectionWithNode](#): (page 176)

Declared In

WebView.h

downloadDelegate

Return the receiver's download delegate.

- (id)downloadDelegate

Return Value

The receiver's download delegate that implements the `WebDownload` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setDownloadDelegate:](#) (page 181)

Declared In

WebView.h

drawsBackground

Returns whether the web view draws a background.

- (BOOL)drawsBackground

Return Value

YES if a background is drawn; otherwise, NO.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

editableDOMRangeForPoint:

Returns the editable DOM object located at a given point.

- (DOMRange *)editableDOMRangeForPoint:(NSPoint)point

Parameters

point

The location of the editable DOM object.

Return Value

A single range object of the editable DOM object located at *point* in the receiver's coordinates.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [selectedDOMRange](#) (page 178)

- [setSelectedDOMRange:affinity:](#) (page 187)

Declared In

WebView.h

editingDelegate

Returns the receiver's editing delegate.

- (id)editingDelegate

Return Value

The receiver's editing delegate.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setEditingDelegate:](#) (page 182)

Declared In

WebView.h

elementAtPoint:

Returns a dictionary description of the element at a given point in the receiver's coordinates.

- (NSDictionary *)elementAtPoint:(NSPoint)point

Parameters

point

The point to represent as a dictionary.

Return Value

A dictionary description of the element at *point* in the receiver's coordinates.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:dragDestinationActionMaskForDraggingInfo:](#) (page 294) (WebUIDelegate)

- [webView:dragSourceActionMaskForPoint:](#) (page 295)

Declared In

WebView.h

estimatedProgress

Returns an estimate, as a percentage, of the amount of content that is currently loaded.

- (double)estimatedProgress

Return Value

A number ranging from 0 to 1.0 and, once a load completes, 1.0 until a new load starts, at which point it resets to 0.

The value is an estimate based on the total number of bytes expected to be received for a document, including all its possible subresources. For more accurate load progress information, implement delegates conforming to the `WebFrameLoadDelegate` and `WebResourceLoadDelegate` informal protocols.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

[WebViewProgressEstimateChangedNotification](#) (page 201)

[WebViewProgressFinishedNotification](#) (page 202)

[WebViewProgressStartedNotification](#) (page 202)

Declared In

WebView.h

frameLoadDelegate

Return the receiver's frame load delegate.

- (id)frameLoadDelegate

Return Value

A frame load delegate that conforms to the `WebFrameLoadDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setFrameLoadDelegate:](#) (page 183)

Declared In

WebView.h

goBack

Loads the previous location in the back-forward list.

- (BOOL)goBack

Return Value

YES if able to move backward; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [backForwardList](#) (page 149)
- [goForward](#) (page 161)
- [goToBackForwardItem:](#) (page 162)

Declared In

WebView.h

goBack:

An action method that loads the previous location in the back-forward list.

- (void)goBack:(id) *sender*

Parameters

sender

The object that sent this message.

Discussion

This method does nothing if it is unable to move backward.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goForward:](#) (page 162)

Declared In

WebView.h

goForward

Loads the next location in the back-forward list.

- (BOOL)goForward

Return Value

YES if able to move forward; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goToBackForwardItem:](#) (page 162)
- [goBack](#) (page 160)

Declared In

WebView.h

goForward:

An action method that loads the next location in the back-forward list.

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

Parameters

sender

The object that sent this message.

Discussion

This method does nothing if it is unable to move forward.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goBack:](#) (page 161)

Declared In

WebView.h

goToBackForwardItem:

Loads a specific location from the back-forward list and sets it as the current item.

```
- (BOOL)goToBackForwardItem:(WebHistoryItem *)item
```

Parameters

item

The index of the location to load. This method sets the current item in the back-forward list to *item*.

Return Value

YES if *item* is in the back-forward list; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [goForward](#) (page 161)

- [goBack](#) (page 160)

- [backForwardList](#) (page 149)

Declared In

WebView.h

groupName

Returns the receiver's group name.

```
- (NSString *)groupName
```

Return Value

The receiver's group name.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setGroupName:](#) (page 183)

Declared In

WebView.h

hostWindow

Returns the receiver's host window.

```
- (NSWindow *)hostWindow
```

Return Value

The receiver's host window.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setHostWindow:](#) (page 184)

Declared In

WebView.h

initWithFrame:frameName:groupName:

Initializes the receiver with a frame rectangle, frame name, and group name.

```
- (id)initWithFrame:(NSRect)frameRect frameName:(NSString *)frameName
    groupName:(NSString *)groupName
```

Parameters

frameRect

The frame rectangle for the created view object.

frameName

The web frame's name. This should not be one of the predefined frame names (see the [WebFrame findFrameNamed:](#) (page 58) method for a description of their meaning), but a custom name or a name used in HTML source. This parameter can be `nil`.

groupName

An arbitrary identifier used to group related frames. For example, JavaScript running in a frame can access any other frame in the same group. It's up to the application how it chooses to scope related frames. This parameter can be `nil`.

Return Value

An initialized view object or `nil` if the object couldn't be created.

Discussion

This method is the designated initializer for the `WebView` class.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

`WebView.h`

isContinuousSpellCheckingEnabled

Returns whether the web view has continuous spell-checking enabled.

- (BOOL)isContinuousSpellCheckingEnabled

Return Value

YES if the object has continuous spell-checking enabled; otherwise, NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setContinuousSpellCheckingEnabled:](#) (page 180)

Declared In

`WebView.h`

isEditable

Returns whether the user is allowed to edit the document.

- (BOOL)isEditable

Return Value

YES if the receiver allows the user to edit the HTML document, NO if it doesn't.

Discussion

You can change the receiver's document programmatically regardless of this setting.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setEditable:](#) (page 182)

Declared In

`WebView.h`

isLoading

Returns whether the web view is loading content.

- (BOOL)isLoading

Return Value

YES if the web view is currently loading any resources; otherwise, NO.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

mainFrame

Returns the main frame, the root of the web frame hierarchy for this page.

- (WebFrame *)mainFrame

Return Value

The main frame.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Related Sample Code

NewsReader

Declared In

WebView.h

mainFrameDocument

Returns the DOM document for the main frame.

- (DOMDocument *)mainFrameDocument

Return Value

The DOM document for the main frame.

Discussion

Invoking this method is equivalent to `[[webView mainFrame] DOMDocument]`.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

mainFrameIcon

Returns the site's favicon.

```
- (NSImage *)mainFrameIcon
```

Return Value

The site's icon. Returns `nil` if no favicon is provided.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

mainFrameTitle

Returns the HTML title of the loaded page.

```
- (NSString *)mainFrameTitle
```

Return Value

The HTML title of the loaded page. Returns @" " if the loaded document is not HTML.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

mainFrameURL

Returns the URL that the main frame loads.

```
- (NSString *)mainFrameURL
```

Return Value

The main frame URL string.

Discussion

This method is functionally equivalent to invoking `[[[[[webView mainFrame] provisionalDataSource] request] URL] absoluteString]`.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

maintainsInactiveSelection

Returns whether the selection is maintained when focus is lost.

- (BOOL)maintainsInactiveSelection

Return Value

YES if the selection is maintained when focus is lost; otherwise, NO.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

makeTextLarger:

Action method that increases the text size by one unit.

- (void)makeTextLarger:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [canMakeTextLarger](#) (page 150)
- [canMakeTextSmaller](#) (page 151)
- [makeTextSmaller:](#) (page 167)

Declared In

WebView.h

makeTextSmaller:

Action method that reduces the text size by one unit.

- (void)makeTextSmaller:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [canMakeTextSmaller](#) (page 151)
- [canMakeTextLarger](#) (page 150)
- [makeTextLarger:](#) (page 167)

Declared In

WebView.h

makeTextStandardSize:

Resets the text size to a multiple of 1.

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

Parameters*sender*

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

mediaStyle

Returns the receiver's CSS media property.

```
- (NSString *)mediaStyle
```

Return ValueThe receiver's CSS media property. `nil` if no media style was set.**Availability**

Available in Mac OS X v10.3.9 and later.

See Also

- [setMediaStyle:](#) (page 185)

Declared In

WebView.h

moveDragCaretToPoint:

Moves the drag caret that indicates the destination of a drag operation to a given point.

```
- (void)moveDragCaretToPoint:(NSPoint)point
```

Parameters*point*

The point to move the drag caret to.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [removeDragCaret](#) (page 174)

Declared In

WebView.h

moveToBeginningOfSentence:

Moves the insertion point to the beginning of the current sentence.

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

Parameters*sender*

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

moveToBeginningOfSentenceAndModifySelection:

Moves the insertion point and extends the selection to the beginning of the current sentence.

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

Parameters*sender*

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

moveToEndOfSentence:

Moves the insertion point to the end of the current sentence.

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

Parameters*sender*

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

moveToEndOfSentenceAndModifySelection:

Moves the insertion point and extends the selection to the end of the current sentence.

- (void)moveToEndOfSentenceAndModifySelection:(id) *sender*

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

paste:

An action method that pastes content from the pasteboard at the insertion point or over the selection.

- (void)paste:(id) *sender*

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [cut:](#) (page 156)

- [copy:](#) (page 155)

Declared In

WebView.h

pasteAsPlainText:

An action method that pastes pasteboard content as plain text.

- (void)pasteAsPlainText:(id) *sender*

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [pasteAsRichText:](#) (page 171)

Declared In

WebView.h

pasteAsRichText:

An action method that pastes pasteboard content into the receiver as rich text, maintaining its attributes.

- (void)pasteAsRichText:(id)sender

Parameters

sender

The object that sent this message.

Discussion

The text is inserted at the insertion point if there is one; otherwise, it replaces the selection.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [pasteAsPlainText:](#) (page 170)

Declared In

WebView.h

pasteboardTypesForElement:

Returns an array of pasteboard types for an element.

- (NSArray *)pasteboardTypesForElement:(NSDictionary *)element

Parameters

element

The element whose pasteboard types you want.

Return Value

An array of pasteboard types for an element.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [pasteboardTypesForSelection](#) (page 171)

+ [URLFromPasteboard:](#) (page 145)

+ [URLTitleFromPasteboard:](#) (page 146)

Declared In

WebView.h

pasteboardTypesForSelection

Returns an array of pasteboard types that can be used for the current selection of the receiver.

- (NSArray *)pasteboardTypesForSelection

Return Value

An array of pasteboard types that can be used for the current selection of the receiver.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [pasteboardTypesForElement:](#) (page 171)
- + [URLFromPasteboard:](#) (page 145)
- + [URLTitleFromPasteboard:](#) (page 146)

Declared In

WebView.h

pasteFont:

An action method that pastes font information from the font pasteboard.

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

Parameters

sender

The object that sent this message.

Discussion

This action method pastes font information from the font pasteboard onto the selected content or insertion point of a rich text object, or over all text of the receiver.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [copyFont:](#) (page 155)

Declared In

WebView.h

performFindPanelAction:

An action method that opens the Find menu and Find panel.

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

Parameters

sender

The object that sent this message.

Discussion

This method behaves similar to the `performFindPanelAction:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

policyDelegate

Returns the receiver's policy delegate.

- (id)policyDelegate

Return Value

A policy delegate that conforms to the `WebPolicyDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setPolicyDelegate:](#) (page 185)

Declared In

WebView.h

preferences

Returns the receiver's preferences.

- (WebPreferences *)preferences

Return Value

The receiver's preferences or the standard preferences, if the preferences were not set using the [setPreferences:](#) (page 186) method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setPreferences:](#) (page 186)

+ [standardPreferences](#) (page 96)

Declared In

WebView.h

preferencesIdentifier

Returns the identifier of the receiver's preferences.

- (NSString *)preferencesIdentifier

Return Value

The preferences identifier.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setPreferencesIdentifier:](#) (page 186)

Declared In

WebView.h

reload:

An action method that reloads the current page.

- (void)reload:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setResourceLoadDelegate:](#) (page 187)

Declared In

WebView.h

reloadFromOrigin:

Action method that performs an end-to-end revalidation using cache-validating conditionals if possible.

- (void)reloadFromOrigin:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [reloadFromOrigin](#) (page 64) (WebView)

- [reload:](#) (page 174)

Declared In

WebView.h

removeDragCaret

Removes the drag caret that indicates the destination of a drag operation.

- (void)removeDragCaret

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [moveDragCaretToPoint:](#) (page 168)

Declared In

WebView.h

replaceSelectionWithArchive:

Replaces the current selection with an archive's contents.

```
- (void)replaceSelectionWithArchive:(WebArchive *)archive
```

Parameters

archive

The archive that replaces the current selection.

Discussion

If the current selection is collapsed (a range is selected with the same nodes and offsets for the start and end) then no content is removed when inserting the archive, and the selection is collapsed and moved to the end of the inserted content. If no content is selected, the archive is not inserted.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [replaceSelectionWithText:](#) (page 176)
- [replaceSelectionWithMarkupString:](#) (page 175)
- [replaceSelectionWithNode:](#) (page 176)
- [deleteSelection](#) (page 157)

Declared In

WebView.h

replaceSelectionWithMarkupString:

Replaces the current selection with mixed text and markup.

```
- (void)replaceSelectionWithMarkupString:(NSString *)markupString
```

Parameters

markupString

The markup string that replaces the current selection.

Discussion

If the current selection is collapsed (a range is selected with the same nodes and offsets for the start and end) then no content is removed when inserting the markup, and the selection is collapsed and moved to the end of the inserted content. If no content is selected, the markup is not inserted.

See [HTML Clipboard Format](#) for a specification of the supported HTML markup.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [replaceSelectionWithText:](#) (page 176)
- [replaceSelectionWithNode:](#) (page 176)
- [replaceSelectionWithArchive:](#) (page 175)
- [deleteSelection](#) (page 157)

Declared In

WebView.h

replaceSelectionWithNode:

Replaces the receiver's current selection with the specified DOM node.

```
- (void)replaceSelectionWithNode:(DOMNode *)node
```

Parameters

node

The node that replaces the current selection. If *nil*, a `NOT_FOUND_ERR` DOM error is thrown as an exception. Use the [deleteSelection](#) (page 157) method to delete the selection.

Discussion

If the current selection is collapsed (a range is selected with the same nodes and offsets for the start and end) then no content is removed when inserting the node, and the selection is collapsed and moved to the end of the inserted content. If no content is selected, the node is not inserted.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [replaceSelectionWithText:](#) (page 176)
- [replaceSelectionWithMarkupString:](#) (page 175)
- [replaceSelectionWithArchive:](#) (page 175)
- [deleteSelection](#) (page 157)

Declared In

WebView.h

replaceSelectionWithText:

Replaces the current selection with a string of text.

```
- (void)replaceSelectionWithText:(NSString *)text
```

Parameters

text

The text that replaces the current selection.

Discussion

If the current selection is collapsed (a range is selected with the same nodes and offsets for the start and end) then no content is removed when inserting the text, and the selection is collapsed and moved to the end of the inserted content. If no content is selected, the text is not inserted.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [replaceSelectionWithNode:](#) (page 176)
- [replaceSelectionWithMarkupString:](#) (page 175)
- [replaceSelectionWithArchive:](#) (page 175)
- [deleteSelection](#) (page 157)

Declared In

WebView.h

resourceLoadDelegate

Returns the receiver's resource load delegate.

```
- (id)resourceLoadDelegate
```

Return Value

A resource load delegate that conforms to the `WebResourceLoadDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setResourceLoadDelegate:](#) (page 187)

Declared In

WebView.h

searchFor:direction:caseSensitive:wrap:

Searches a document view for a string and highlights it if it is found.

```
- (BOOL)searchFor:(NSString *)string direction:(BOOL)forward
    caseSensitive:(BOOL)caseFlag wrap:(BOOL)wrapFlag
```

Parameters

string

The search string.

forward

If YES the direction of the search is forward; if NO, the direction is backward.

caseFlag

If YES if the search is case sensitive; otherwise, it is not.

wrapFlag

If YES if the search wraps; otherwise, it does not.

Return Value

YES if the search is successful; otherwise, NO.

Discussion

The search for *string* begins from the current selection and continues in the direction specified by *forward*. The search continues across all frames.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [applicationNameForUserAgent](#) (page 148)
- [setCustomUserAgent:](#) (page 181)
- [customUserAgent](#) (page 156)

Declared In

WebView.h

selectedDOMRange

Returns the range of the current selection.

- (DOMRange *)selectedDOMRange

Return Value

The range of the current selection. *nil* if nothing is selected.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [selectionAffinity](#) (page 179)
- [setSelectedDOMRange:affinity:](#) (page 187)
- [editableDOMRangeForPoint:](#) (page 158)

Declared In

WebView.h

selectedFrame

Returns the frame with the active selection.

- (WebFrame *)selectedFrame

Return Value

The frame that contains the first responder. If it doesn't exist, the frame that contains a non-zero-length selection; otherwise, *nil*.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

selectionAffinity

Returns the current selection affinity.

- (NSSelectionAffinity)selectionAffinity

Return Value

The preferred direction of selection—upward or downward—of the receiver’s current selection. For example, if text wraps across line boundaries, the value returned by this method indicates whether or not the insertion point appears after the last character of the first line or before the first character of the following line.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [selectedDOMRange](#) (page 178)
- [setSelectedDOMRange:affinity:](#) (page 187)

Declared In

WebView.h

selectSentence:

Selects the entire sentence around the insertion point.

- (void)selectSentence:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

setApplicationNameForUserAgent:

Sets the application name used in the user-agent string.

- (void)setApplicationNameForUserAgent:(NSString *)applicationName

Parameters*applicationName*

The application name to use in the user-agent string. The user-agent is used by websites to identify the client browser.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [applicationNameForUserAgent](#) (page 148)
- [setCustomUserAgent:](#) (page 181)
- [customUserAgent](#) (page 156)

Declared In

WebView.h

setContinuousSpellCheckingEnabled:

Sets whether the web view has continuous spell-checking enabled.

```
- (void)setContinuousSpellCheckingEnabled:(BOOL)flag
```

Parameters*flag*

YES if the object should have continuous spell-checking enabled; otherwise, NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [isContinuousSpellCheckingEnabled](#) (page 164)

Declared In

WebView.h

setCustomTextEncodingName:

Sets the custom text encoding name.

```
- (void)setCustomTextEncodingName:(NSString *)encodingName
```

Parameters*encodingName*

A text encoding name. If `nil`, the default encoding is restored.

Discussion

This method overrides the default text encoding, including any encoding that is specified in the webpage header or HTTP response. Invoking this method stops any load in progress. The default encoding is restored when the main frame changes to a new location, or if *encodingName* is `nil`.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [customTextEncodingName](#) (page 155)

Declared In

WebView.h

setCustomUserAgent:

Sets the receiver's custom user-agent string.

```
- (void)setCustomUserAgent:(NSString *)userAgentString
```

Parameters

userAgentString

The custom user-agent string. The user-agent string is used by websites to identify the client browser. The custom user-agent string is used for all URLs. If `nil`, then the receiver constructs a user-agent string that produces the best rendering results for each URL.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setApplicationNameForUserAgent:](#) (page 179)

- [applicationNameForUserAgent](#) (page 148)

- [customUserAgent](#) (page 156)

Declared In

WebView.h

setDownloadDelegate:

Sets the receiver's shared download delegate.

```
- (void)setDownloadDelegate:(id)delegate
```

Parameters

delegate

The download delegate that implements the `WebDownload` protocol.

Discussion

WebKit may create `WebDownload` objects automatically to handle downloads that start with a webpage or link.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [downloadDelegate](#) (page 158)

Declared In

WebView.h

setDrawsBackground:

Sets whether a default background is drawn when the webpage has no background set.

```
- (void)setDrawsBackground:(BOOL)drawsBackground
```

Parameters

drawsBackground

If YES, a default background is drawn; if NO, it is not.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

setEditable:

Sets whether the receiver allows the user to edit its HTML document.

```
- (void)setEditable:(BOOL)flag
```

Parameters

flag

YES if the receiver allows the user to edit the document. NO if an element in the receiver's document can be edited only if the `CONTENTEDITABLE` attribute has been set on the element or one of its parent elements.

Discussion

You can change the receiver's document programmatically regardless of this setting. By default a `WebView` object is not editable.

Normally, an HTML document is not editable unless the elements within the document are editable. This method provides a low-level way to make the contents of a `WebView` object editable without altering the document or DOM structure.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [isEditable](#) (page 164)

Declared In

WebView.h

setEditingDelegate:

Sets the receiver's editing delegate.

- (void)setEditingDelegate:(id)delegate

Parameters

delegate

The editing delegate for the web view that conforms to the `WebEditingDelegate` protocol.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [editingDelegate](#) (page 159)

Declared In

WebView.h

setFrameLoadDelegate:

Sets the receiver's frame load delegate.

- (void)setFrameLoadDelegate:(id)delegate

Parameters

delegate

A frame load delegate that conforms to the `WebFrameLoadDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [frameLoadDelegate](#) (page 160)

Declared In

WebView.h

setGroupName:

Sets the receiver's group name.

- (void)setGroupName:(NSString *)groupName

Parameters

groupName

An arbitrary identifier used to group related frames.

Discussion

You might use this method to set the group name of a `WebView` object after it is loaded from a nib file.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [initWithFrame:frameName:groupName:](#) (page 163)

- [groupName](#) (page 162)

Declared In

WebView.h

setHostWindow:

Sets the receiver's host window.

```
- (void)setHostWindow:(NSWindow *)hostWindow
```

Parameters

hostWindow

A host window.

Discussion

This method sets the receiver's host window to *hostWindow*. Your application should only use this method if a web view is going to be removed from its window temporarily, and you want the web view to continue operating (for example, you don't want to interrupt a load in progress). Since the receiver retains *hostWindow*, it is your responsibility to set the host window to `nil` before closing the window to avoid a retain loop.

For example, you might invoke this method if you attach a web view to an `NSTabView` object (as in a tabbed browser implementation). The `NSTabView` object takes views out of the window when they are not in the active tab, so you need to invoke this method before the web view is removed from its window. If you don't invoke this method, plug-ins will stop operating when the web view is removed from its window.

Note: Plug-ins and JavaScript depend on a window to function properly even if the web view is not in an actual window.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [hostWindow](#) (page 163)

Declared In

WebView.h

setMainFrameURL:

Sets the URL that the main frame loads.

```
- (void)setMainFrameURL:(NSString *)URLString
```

Parameters

URLString

The main frame URL string.

Discussion

This method is functionally equivalent to invoking `[[webView mainFrame] loadRequest:[NSURLRequest requestWithURL:[NSURL URLWithString:URLString]]]`.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

setMaintainsBackForwardList:

Sets whether to use a back-forward list.

```
- (void)setMaintainsBackForwardList:(BOOL)flag
```

Parameters

flag

If NO, clears the back-forward list and release the page cache; otherwise, it does not.

Discussion

The back-forward list maintains a page cache, so applications that do not use the [goForward](#) (page 161) or [goBack](#) (page 160) methods should disable it.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [backForwardList](#) (page 149)

Declared In

WebView.h

setMediaStyle:

Sets the receiver's CSS media property.

```
- (void)setMediaStyle:(NSString *)mediaStyle
```

Parameters

mediaStyle

The CSS media property for the receiver.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [mediaStyle](#) (page 168)

Declared In

WebView.h

setPolicyDelegate:

Sets the receiver's policy delegate.

```
- (void)setPolicyDelegate:(id)delegate
```

Parameters

delegate

A policy delegate that conforms to the `WebPolicyDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [policyDelegate](#) (page 173)

Declared In

WebView.h

setPreferences:

Sets the receiver's preferences.

```
- (void)setPreferences:(WebPreferences *)preferences
```

Parameters

preferences

The web view's preferences.

Discussion

Typically, you do not invoke this method directly. Use the [setPreferencesIdentifier:](#) (page 186) method to change the receiver's preferences.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [preferences](#) (page 173)

Declared In

WebView.h

setPreferencesIdentifier:

Sets the receiver's preferences identifier, creating a preferences object if needed.

```
- (void)setPreferencesIdentifier:(NSString *)anIdentifier
```

Parameters

anIdentifier

The unique identifier for the preferences—it is fixed to the keys used to store the receiver's preferences in the user defaults database. `WebView` objects can share instances of the `WebPreferences` class by using the same preferences identifier.

Discussion

This method sets the receiver's preferences to the specified preferences object if it exists. Otherwise, this method creates a new `WebPreferences` object for the receiver initialized with `anIdentifier`. Use this method to change the preferences used by the receiver's `WebFrameView` objects. If you do not directly set the preferences, `WebFrameView` objects use the preferences returned by the `standardPreferences` (page 96) class method of `WebPreferences`.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [preferencesIdentifier](#) (page 173)
- [setAutosaves:](#) (page 106)

Declared In

WebView.h

setResourceLoadDelegate:

Sets the receiver's resource load delegate.

```
- (void)setResourceLoadDelegate:(id)delegate
```

Parameters

delegate

A resource load delegate that conforms to the `WebResourceLoadDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [resourceLoadDelegate](#) (page 177)

Declared In

WebView.h

setSelectedDOMRange:affinity:

Selects a range of nodes.

```
- (void)setSelectedDOMRange:(DOMRange *)range
    affinity:(NSSelectionAffinity)selectionAffinity
```

Parameters

range

The range of nodes to select. If *range* is `nil`, the current selection is cleared. This method raises a `DOMRangeException` if the range has been detached or refers to nodes not displayed by the receiver.

selectionAffinity

See the [selectionAffinity](#) (page 179) method for information on selection affinity.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [selectedDOMRange](#) (page 178)
- [selectionAffinity](#) (page 179)
- [editableDOMRangeForPoint:](#) (page 158)

Declared In

WebView.h

setShouldCloseWithWindow:

Sets whether the web view should close when its window or host window closes.

```
- (void)setShouldCloseWithWindow:(BOOL)close
```

Parameters

close

If YES, the web view should close; otherwise, it should not.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

setShouldUpdateWhileOffscreen:

Sets whether the web view should update even when it is not in a window that is currently visible.

```
- (void)setShouldUpdateWhileOffscreen:(BOOL)updateWhileOffscreen
```

Parameters

updateWhileOffscreen

If YES, the web view updates regardless if it is visible. If NO, it updates only if it is visible, possibly improving performance, and then updates automatically when it becomes visible. The default value is YES.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [shouldUpdateWhileOffscreen](#) (page 190)

Declared In

WebView.h

setSmartInsertDeleteEnabled:

Sets whether the receiver should insert or delete spaces around selected words to preserve proper spacing and punctuation.

- (void)setSmartInsertDeleteEnabled:(BOOL)*flag*

Parameters

flag

If YES, the receiver performs smart insert and delete; if NO, it inserts and deletes exactly what's selected.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [smartInsertDeleteEnabled](#) (page 191)

Declared In

WebView.h

setTextSizeMultiplier:

Change the font size multiplier for text displayed in web frame view objects managed by the receiver.

- (void)setTextSizeMultiplier:(float)*multiplier*

Parameters

multiplier

A fractional percentage value where 1.0 denotes 100%.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [textSizeMultiplier](#) (page 195)

Declared In

WebView.h

setTypingStyle:

Sets the receiver's CSS typing style.

- (void)setTypingStyle:(DOMCSSStyleDeclaration *)*style*

Parameters

style

The receiver's CSS typing style.

Discussion

The typing style is reset automatically when the receiver's selection changes.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [typingStyle](#) (page 196)

- [computedStyleForElement:pseudoElement:](#) (page 154)

- [applyStyle:](#) (page 149)

Declared In

WebView.h

setUIDelegate:

Sets the receiver's user interface delegate.

- (void)setUIDelegate:(id)delegate

Parameters

delegate

A user interface delegate that conforms to the `WebUIDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [UIDelegate](#) (page 196)

Declared In

WebView.h

shouldCloseWithWindow

Returns whether the web view should close when its window or host window closes.

- (BOOL)shouldCloseWithWindow

Return Value

If YES, the web view should close; otherwise, it should not.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

shouldUpdateWhileOffscreen

Returns whether the web view should update even when it is not visible.

- (BOOL)shouldUpdateWhileOffscreen

Return Value

YES if it should update even when it is not visible; otherwise, NO.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [setShouldUpdateWhileOffscreen:](#) (page 188)

Declared In

WebView.h

showGuessPanel:

An action method that shows a spelling correction panel.

- (void)showGuessPanel:(id)sender

Parameters

sender

The object that sent this message.

Discussion

This action method opens the Spelling panel, allowing the user to make a correction during spell checking.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [checkSpelling:](#) (page 153)

Declared In

WebView.h

smartInsertDeleteEnabled

Returns whether smart-space insertion and deletion is enabled.

- (BOOL)smartInsertDeleteEnabled

Return Value

YES if the receiver inserts or deletes space around selected words so as to preserve proper spacing and punctuation. NO if it inserts and deletes exactly what's selected.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setSmartInsertDeleteEnabled:](#) (page 188)

Declared In

WebView.h

spellCheckerDocumentTag

Returns the spell-checker document tag for this document.

- (NSInteger)spellCheckerDocumentTag

Return Value

The document tag for this web view. A tag identifying the receiver's text as a document for the spell-checker server. See the `NSSpellChecker` and `NSSpellServer` class specifications for more information on how this tag is used.

The return value changed from `unsigned int` to a `NSUInteger` in Mac OS X v10.5.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

`spellCheckerDocumentTag` (`NSTextView`)

Declared In

`WebView.h`

startSpeaking:

An action method that starts speaking the selected text or all text if there's no selection.

- (void)startSpeaking:(id) *sender*

Parameters

sender

The object that sent this message.

Discussion

Speech continues asynchronously until the end of the text or until terminated by invoking the `stopSpeaking:` (page 193) method. This method behaves similar to the `startSpeaking:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- `stopSpeaking:` (page 193)

Declared In

`WebView.h`

stopLoading:

An action method that stops the loading of any web frame content managed by the receiver.

- (void)stopLoading:(id) *sender*

Parameters

sender

The object that sent this message.

Discussion

Stops any content in the process of being loaded by the main frame or any of its children frames. Does nothing if no content is being loaded.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebView.h

stopSpeaking:

An action method that stops speaking that is in progress.

- (void)stopSpeaking:(id) *sender*

Parameters

sender

The object that sent this message.

Discussion

This action method stops speech that was previously started with [startSpeaking:](#) (page 192). This method behaves similar to the `stopSpeaking:` method in `NSTextView`.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [startSpeaking:](#) (page 192)

Declared In

WebView.h

stringByEvaluatingJavaScriptFromString:

Returns the result of running a script.

- (NSString *)stringByEvaluatingJavaScriptFromString:(NSString *) *script*

Parameters

script

The script to run.

Return Value

The result of running a JavaScript specified by *script*, or an empty string if the script failed.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Related Sample Code

WebKitPluginWithSimpleGUI

Declared In

WebView.h

styleDeclarationWithText:

Returns the CSS style declaration for the specified text.

```
- (DOMCSSStyleDeclaration *)styleDeclarationWithText:(NSString *)text
```

Parameters

text

The text whose style declaration is returned.

Return Value

The style declaration for *text*.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

supportsTextEncoding

Returns whether the document view supports different text encodings.

```
- (BOOL)supportsTextEncoding
```

Return Value

YES if the receiver's document view can support different text encodings; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebView.h

takeStringURLFrom:

Sets the receiver's current location by obtaining a URL string from the sender.

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

Parameters

sender

The object that sent this message.

Discussion

This method sets the receiver's current location to the value obtained by sending a `stringValue` message to *sender*, then starts loading the URL returned by *sender*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [loadRequest:](#) (page 62) (WebFrame)

Declared In

WebView.h

textSizeMultiplier

Returns the font size multiplier for text displayed in web frame view objects managed by the receiver.

- (float)textSizeMultiplier

Return Value

The font size multiplier, a fractional percentage value where 1.0 denotes 100%.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setTextSizeMultiplier:](#) (page 189)

Declared In

WebView.h

toggleContinuousSpellChecking:

Toggles whether continuous spell checking is available.

- (void)toggleContinuousSpellChecking:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

toggleSmartInsertDelete:

Toggles whether spaces around selected words are inserted or deleted to preserve proper spacing and punctuation.

- (void)toggleSmartInsertDelete:(id)sender

Parameters

sender

The object that sent this message.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebView.h

typingStyle

Returns the receiver's CSS typing style.

- (DOMCSSStyleDeclaration *)typingStyle

Return Value

The receiver's CSS typing style.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setTypingStyle:](#) (page 189)
- [computedStyleForElement:pseudoElement:](#) (page 154)
- [applyStyle:](#) (page 149)

Declared In

WebView.h

UIDelegate

Returns the receiver's user interface delegate.

- (id)UIDelegate

Return Value

A user interface delegate that conforms to the `WebUIDelegate` protocol.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setUIDelegate:](#) (page 190)

Declared In

WebView.h

undoManager

Returns the receiver's undo manager.

- (NSUndoManager *)undoManager

Return Value

The receiver's undo manager.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

userAgentForURL:

Returns the appropriate user-agent string for a given URL.

```
- (NSString *)userAgentForURL:(NSURL *)URL
```

Parameters

URL

The URL that you need the user-agent string for.

Return Value

The user-agent string for a given URL. The user-agent string is used by websites to identify the client browser.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setCustomUserAgent:](#) (page 181)

- [customUserAgent](#) (page 156)

Declared In

WebView.h

windowScriptObject

Returns the receiver's window object from the scripting environment.

```
- (WebScriptObject *)windowScriptObject
```

Return Value

The receiver's window object.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [objectForWebScript](#) (page 250) (WebPlugIn) (WebFrameLoadDelegate)

- [webView>windowScriptObjectAvailable:](#) (page 239)

Related Sample Code

QT Capture Widget

WebKitPluginWithJavaScript

Declared In

WebView.h

writeElement:withPasteboardTypes:toPasteboard:

Writes an element to the pasteboard using a list of types.

```
- (void)writeElement:(NSDictionary *)element withPasteboardTypes:(NSArray *)types  
    toPasteboard:(NSPasteboard *)pasteboard
```

Parameters*element*

The element to write to the pasteboard.

types

The pasteboard types to use for the element.

pasteboard

The pasteboard to use for writing.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [writeSelectionWithPasteboardTypes:toPasteboard:](#) (page 198)

Declared In

WebView.h

writeSelectionWithPasteboardTypes:toPasteboard:

Writes the receiver's current selection to a pasteboard using a list of types.

```
- (void)writeSelectionWithPasteboardTypes:(NSArray *)types toPasteboard:(NSPasteboard  
    *)pasteboard
```

Parameters*types*

The pasteboard types to use for the selection.

pasteboard

The pasteboard to use for writing.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [writeElement:withPasteboardTypes:toPasteboard:](#) (page 198)

Declared In

WebView.h

Constants

Element Dictionary Keys

Predefined keys used to access an element dictionary.

```
extern NSString *WebElementDOMNodeKey;
extern NSString *WebElementFrameKey;
extern NSString *WebElementImageAltStringKey;
extern NSString *WebElementImageKey;
extern NSString *WebElementImageRectKey;
extern NSString *WebElementImageURLKey;
extern NSString *WebElementIsSelectedKey;
extern NSString *WebElementLinkURLKey;
extern NSString *WebElementLinkTargetFrameKey;
extern NSString *WebElementLinkTitleKey;
extern NSString *WebElementLinkLabelKey;
```

Constants

`WebElementDOMNodeKey`

The `DOMNode` for this element.

Available in Mac OS X v10.3 and later.

Declared in `WebView.h`.

`WebElementFrameKey`

The `WebFrame` object associated with this element.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

`WebElementImageAltStringKey`

An `NSString` of the ALT attribute of an image element.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

`WebElementImageKey`

An `NSImage` representing an image element.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

`WebElementImageRectKey`

An `NSValue` containing an `NSRect`, the size of an image element.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

`WebElementImageURLKey`

An `NSURL` containing the location of an image element.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

WebElementIsSelectedKey

An NSNumber used as a BOOL value to indicate whether a text element is selected or not. Zero value indicates false, true otherwise.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

WebElementLinkURLKey

An NSURL containing the location of a link if the element is within an anchor.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

WebElementLinkTargetFrameKey

The WebFrame object associated with the target of the anchor.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

WebElementLinkTitleKey

An NSString containing the title of an anchor.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

WebElementLinkLabelKey

An NSString containing the text within an anchor.

Available in Mac OS X v10.2 and later.

Declared in `WebView.h`.

Discussion

These constants represent predefined keys used to access an element dictionary. An element dictionary is an NSDictionary representation of an HTML element, as in a clicked or selected element. Some methods in the `WebPolicyDelegate` informal protocol have an element dictionary argument. The descriptions below describe the dictionary value for the key.

Notifications

WebViewDidBeginEditingNotification

Posted when a web view begins any operation that changes its contents in response to user editing. The notification object is the WebView object that the user is editing. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebView.h`

WebViewDidChangeNotification

Posted when a web view performs any operation that changes its contents in response to user editing. The notification object is the WebView object that the user is editing. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

WebViewDidChangeSelectionNotification

Posted when a web view changes its typing selection. The notification object is the WebView that changed its typing selection. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

WebViewDidChangeTypingStyleNotification

Posted when a web view changes its typing style. The notification object is the WebView that changed its typing style. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [setTypingStyle:](#) (page 189)

Declared In

WebView.h

WebViewDidEndEditingNotification

Posted when a web view ends any operation that changes its contents in response to user editing. The notification object is the WebView that the user is editing. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebView.h

WebViewProgressEstimateChangedNotification

Posted by a WebView object when the estimated progress value of a load changes. This notification may be posted zero or more times after a [WebViewProgressStartedNotification](#) (page 202) notification is posted. The notification object is the WebView for which the progress value has changed. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [estimatedProgress](#) (page 159)

Declared In

WebView.h

WebViewProgressFinishedNotification

Posted by a WebView object when the load has finished. The notification object is the WebView that finished loading. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [estimatedProgress](#) (page 159)

Declared In

WebView.h

WebViewProgressStartedNotification

Posted by a WebView object when a load begins, including a load that is initiated in a subframe. The notification object is the WebView that began loading. This notification does not contain a `userInfo` dictionary.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [estimatedProgress](#) (page 159)

Declared In

WebView.h

Protocols

WebDocumentRepresentation Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDocument.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

This protocol is adopted by document representation classes that handle specific MIME types. You can implement your own document view classes and document representation classes to render data for specific MIME types, and register those classes using the `WebFrame` `registerViewClass:representationClass:forMIMEType:` (page 145) method.

Tasks

Setting the Data Source

- `setDataSource:` (page 208)
Sets the receiver's data source.

Loading Content

- `receivedData:withDataSource:` (page 207)
Invoked when a data source has received some data.
- `receivedError:withDataSource:` (page 208)
Invoked when a data source receives an error loading its content.
- `finishedLoadingWithDataSource:` (page 207)
Invoked when a data source finishes loading its content.

Getting Document Source

- [canProvideDocumentSource](#) (page 206)
Returns whether the receiver can provide content source.
- [documentSource](#) (page 206)
Returns the receiver's source as text.

Getting the Document Title

- [title](#) (page 209)
Returns the receiver's document title.

Instance Methods

canProvideDocumentSource

Returns whether the receiver can provide content source.

- (BOOL)canProvideDocumentSource

Return Value

YES if the receiver can provide source for the document content (for example, HTML source), NO otherwise.

Discussion

The receiver should return YES only if it makes sense for someone to view the source of the document in question. For example, a web view returns NO if the content is an image, was produced by a plug-in, or contains text content already.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [documentSource](#) (page 206)

Declared In

WebDocument.h

documentSource

Returns the receiver's source as text.

- (NSString *)documentSource

Return Value

Returns the document source associated with the receiver or nil if the source cannot be provided.

Discussion

For example, for HTML documents, the receiver should return the HTML source.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [canProvideDataSource](#) (page 206)

Declared In

WebDocument.h

finishedLoadingWithDataSource:

Invoked when a data source finishes loading its content.

```
- (void)finishedLoadingWithDataSource:(WebDataSource *)dataSource
```

Parameters

dataSource

A `WebDataSource` object that identifies the request that finished loading.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setDataSource:](#) (page 208)

Declared In

WebDocument.h

receivedData:withDataSource:

Invoked when a data source has received some data.

```
- (void)receivedData:(NSData *)data withDataSource:(WebDataSource *)dataSource
```

Parameters

data

An `NSData` object containing the data received.

dataSource

A `WebDataSource` object that identifies the request that generated this data.

Discussion

Data is loaded incrementally, so this method may be invoked multiple times. The receiver is responsible for accumulating this data.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [receivedError:withDataSource:](#) (page 208)

Declared In

WebDocument.h

receivedError:withDataSource:

Invoked when a data source receives an error loading its content.

```
- (void)receivedError:(NSError *)error withDataSource:(WebDataSource *)dataSource
```

Parameters

error

An `NSError` object that indicates what error occurred.

dataSource

A `WebDataSource` object that identifies the request that caused this error.

Discussion

The *error* argument contains details on the error that occurred.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [receivedData:withDataSource:](#) (page 207)

Declared In

WebDocument.h

setDataSource:

Sets the receiver's data source.

```
- (void)setDataSource:(WebDataSource *)dataSource
```

Parameters

dataSource

The data source to associate with the object.

Discussion

This method is invoked soon after the document representation is created.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [finishedLoadingWithDataSource:](#) (page 207)

Declared In

WebDocument.h

title

Returns the receiver's document title.

```
- (NSString *)title
```

Return Value

The title associated with the receiving document object.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDocument.h

WebDocumentSearching Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDocument.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

`WebDocumentSearching` is an optional protocol for document view objects that support searching. Classes that adopt this protocol should also adopt `WebDocumentView` and inherit from `NSView`.

Tasks

Searching a Document

- [searchFor:direction:caseSensitive:wrap:](#) (page 211)
Searches for a string in a given direction from the current position.

Instance Methods

searchFor:direction:caseSensitive:wrap:

Searches for a string in a given direction from the current position.

- (BOOL)searchFor:(NSString *)*string* direction:(BOOL)*directionFlag*
caseSensitive:(BOOL)*caseFlag* wrap:(BOOL)*wrapFlag*

Parameters

string

The string to search for.

directionFlag

If YES, the search is in the forward direction from the current location; otherwise, the search is in the backward direction.

caseFlag

If YES then the search is case sensitive; otherwise, it is not.

wrapFlag

If YES, the search continues from the end of the document to the current location; otherwise, it stops at the end of the document.

Return Value

YES if the receiver contains *string* in the specified direction; otherwise, NO.

Discussion

The receiver should select the string if it is found.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDocument.h

WebDocumentText Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDocument.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

`WebDocumentText` is an optional protocol for document view objects that display text. This protocol defines methods for accessing document content as strings, and methods for text selection. Classes that adopt this protocol should also adopt `WebDocumentView` and inherit from `NSView`.

Tasks

Getting Document Content

- [string](#) (page 216)
Returns the entire content of the web document as a string.
- [attributedString](#) (page 214)
Returns the entire content of the web document as an attributed string.

Selecting and Deselecting Text

- [selectAll](#) (page 215)
Selects all the text in the web document.
- [deselectAll](#) (page 214)
Deselects the currently selected text in the web document.
- [selectedString](#) (page 215)
Returns the currently selected text in the web document as a string.

- [selectedAttributedString](#) (page 215)
Returns the currently selected text in the web document as an attributed string.

Text Encoding

- [supportsTextEncoding](#) (page 216)
Returns a Boolean value that indicates whether the web document supports text encoding.

Instance Methods

attributedString

Returns the entire content of the web document as an attributed string.

- (NSAttributedString *)attributedString

Return Value

An attributed string containing the entire content of the web document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [string](#) (page 216)

Declared In

WebDocument.h

deselectAll

Deselects the currently selected text in the web document.

- (void)deselectAll

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [selectAll](#) (page 215)

Declared In

WebDocument.h

selectAll

Selects all the text in the web document.

- (void)selectAll

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [deselectAll](#) (page 214)

Declared In

WebDocument.h

selectedAttributedString

Returns the currently selected text in the web document as an attributed string.

- (NSAttributedString *)selectedAttributedString

Return Value

An attributed string containing the currently selected text in the web document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [selectedString](#) (page 215)

Declared In

WebDocument.h

selectedString

Returns the currently selected text in the web document as a string.

- (NSString *)selectedString

Return Value

The currently selected text in the web document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [selectedAttributedString](#) (page 215)

Declared In

WebDocument.h

string

Returns the entire content of the web document as a string.

- (NSString *)string

Return Value

The entire content of the web document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [attributedString](#) (page 214)

Declared In

WebDocument.h

supportsTextEncoding

Returns a Boolean value that indicates whether the web document supports text encoding.

- (BOOL)supportsTextEncoding

Return Value

YES if the web document supports text encoding; otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDocument.h

WebDocumentView Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebDocument.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

This protocol is adopted by the document view of a `WebFrameView`. You can extend WebKit to support additional MIME types by implementing your own document view and document representation classes to render data for specific MIME types. You register those classes using the `WebFrame.registerViewClass:representationClass:forMIMEType:` (page 145) method. Classes that adopt this protocol are expected to be subclasses of `NSView`.

Tasks

Setting the Data Source

- `setDataSource:` (page 219)
Invoked when the data source for this document has been changed.
- `dataSourceUpdated:` (page 218)
Invoked when additional data has been received.

Controlling the Layout

- `setNeedsLayout:` (page 219)
Sets whether or not the receiver should change its layout.
- `layout` (page 218)
Invoked when the receiver should change its layout immediately.

Attaching to a Window

- [viewDidMoveToHostWindow](#) (page 219)
Invoked when a web view's host window is set.
- [viewWillMoveToHostWindow:](#) (page 220)
Invoked when a web view's host window is about to change.

Instance Methods

dataSourceUpdated:

Invoked when additional data has been received.

```
- (void)dataSourceUpdated:(WebDataSource *)dataSource
```

Parameters

dataSource

The source of the new data.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setDataSource:](#) (page 219)

Declared In

WebDocument.h

layout

Invoked when the receiver should change its layout immediately.

```
- (void)layout
```

Discussion

This message is sent to the view as a hint to perform any calculations and update rendering information. For example, at a minimum, the receiver might set the frame rectangle. This method should not perform any drawing operations.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [setNeedsLayout:](#) (page 219)

Declared In

WebDocument.h

setDataSource:

Invoked when the data source for this document has been changed.

```
- (void)setDataSource:(WebDataSource *)dataSource
```

Parameters

dataSource

The new data source for the document.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [dataSourceUpdated:](#) (page 218)

Declared In

WebDocument.h

setNeedsLayout:

Sets whether or not the receiver should change its layout.

```
- (void)setNeedsLayout:(BOOL)flag
```

Parameters

flag

Sets whether the receiver needs to update its layout in the next call to its `drawRect:` method.

Discussion

A view conforming to this protocol should store the most recent value of this flag in an internal variable.

Then, in its `drawRect` method, if the most recent value of this flag was YES, it should invoke [layout](#) (page 218) and reset the internal variable before updating the contents of the view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebDocument.h

viewDidMoveToHostWindow

Invoked when a web view's host window is set.

```
- (void)viewDidMoveToHostWindow
```

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [viewWillMoveToHostWindow:](#) (page 220)

- [setHostWindow:](#) (page 184)

Declared In

WebDocument.h

viewWillMoveToHostWindow:

Invoked when a web view's host window is about to change.

- (void)viewWillMoveToHostWindow:(NSWindow *)*hostWindow*

Parameters

hostWindow

The new host window for the view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [viewDidMoveToHostWindow](#) (page 219)
- [setHostWindow:](#) (page 184)

Declared In

WebDocument.h

WebEditingDelegate Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebEditingDelegate.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

You use `WebEditingDelegate` to control or augment the editing behavior of a `WebView` object. Objects conforming to the `WebEditingDelegate` informal protocol may receive *should* messages before or *did* messages after an editing action. Typically, you implement an editing delegate if you want to change the default editing behavior.

Tasks

Controlling Editing Behavior

- [webView:shouldApplyStyle:toElementsInDOMRange:](#) (page 223)
Returns whether the user should be allowed to apply a style to a range of content.
- [webView:shouldBeginEditingInDOMRange:](#) (page 223)
Returns whether the user is allowed to edit a range of content in a web view.
- [webView:shouldChangeSelectedDOMRange:toDOMRange:affinity:stillSelecting:](#) (page 224)
Returns whether the user should be allowed to change the selected range.
- [webView:shouldChangeTypingStyle:toStyle:](#) (page 225)
Returns whether the user should be allowed to change the typing style in a web view.
- [webView:shouldDeleteDOMRange:](#) (page 225)
Returns whether the user should be allowed to delete a range of content.
- [webView:shouldEndEditingInDOMRange:](#) (page 226)
Returns whether the user should be allowed to end editing.
- [webView:shouldInsertNode:replacingDOMRange:givenAction:](#) (page 226)
Returns whether the user should be allowed to insert a node in place of a range of content.
- [webView:shouldInsertText:replacingDOMRange:givenAction:](#) (page 227)
Returns whether a user should be allowed to insert text in place of a range of content.

Responding to Notifications

- [webViewDidBeginEditing:](#) (page 228)
Sent by the default notification center when the user begins editing the web view.
- [webViewDidChange:](#) (page 228)
Sent by the default notification center when the user changes content in the web view.
- [webViewDidChangeSelection:](#) (page 228)
Sent by the default notification center when the user changes the selection in the web view.
- [webViewDidChangeTypingStyle:](#) (page 229)
Sent by the default notification center when the user changes the typing style in the web view.
- [webViewDidEndEditing:](#) (page 229)
Sent by the default notification center when the user stops editing the web view.

Performing Commands

- [webView:doCommandBySelector:](#) (page 223)
Returns whether the receiver performs a command instead of the web view.

Getting the Undo Manager

- [undoManagerForWebView:](#) (page 222)
Returns the undo manager to be used by a web view.

Instance Methods

undoManagerForWebView:

Returns the undo manager to be used by a web view.

```
- (NSUndoManager *)undoManagerForWebView:(WebView *)webView
```

Parameters

webView

The web view that the user is editing.

Return Value

The undo manager.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebEditingDelegate.h

webView:doCommandBySelector:

Returns whether the receiver performs a command instead of the web view.

```
- (BOOL)webView:(WebView *)webView doCommandBySelector:(SEL)command
```

Parameters

webView

The web view that the user is editing.

command

The command to perform.

Return Value

YES if the receiver will perform *command*; otherwise, NO.

Discussion

Implement this method if you want to perform *command* instead of letting the web view perform *command*.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebEditingDelegate.h

webView:shouldApplyStyle:toElementsInDOMRange:

Returns whether the user should be allowed to apply a style to a range of content.

```
- (BOOL)webView:(WebView *)webView shouldApplyStyle:(DOMCSSStyleDeclaration *)style
toElementsInDOMRange:(DOMRange *)range
```

Parameters

webView

The web view that the user is editing.

style

The style to apply.

range

The range of the content.

Return Value

YES if the user should be allowed to apply the style to the content range; otherwise, NO.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebEditingDelegate.h

webView:shouldBeginEditingInDOMRange:

Returns whether the user is allowed to edit a range of content in a web view.

```
- (BOOL)webView:(WebView *)webView shouldBeginEditingInDOMRange:(DOMRange *)range
```

Parameters*webView*

The web view that the user is editing.

*range*The section of the begin-editing request; used to determine if editing is allowed. Typically, *range* is not the current selection but may become the current selection if this method returns YES.**Return Value**YES if the user is allowed to edit *webView*; otherwise, NO.**Discussion**

This method is invoked when a web view attempts to become the first responder or when the user drops an object on it.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldEndEditingInDOMRange:](#) (page 226)
- [webViewDidBeginEditing:](#) (page 228)

Declared In

WebEditingDelegate.h

webView:shouldChangeSelectedDOMRange:toDOMRange:affinity:stillSelecting:

Returns whether the user should be allowed to change the selected range.

```
- (BOOL)webView:(WebView *)webView shouldChangeSelectedDOMRange:(DOMRange *)currentRange toDOMRange:(DOMRange *)proposedRange
    affinity:(NSSelectionAffinity)selectionAffinity stillSelecting:(BOOL)flag
```

Parameters*webView*

The web view that the user is editing.

currentRange

The old range the user wants to change.

proposedRange

The new range the user wants to select.

selectionAffinity

The direction of the selection.

flag

YES if the user is still selecting; otherwise, NO.

Return Value

YES if the user is allowed to change the selected range; otherwise, NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webViewDidChangeSelection:](#) (page 228)

Declared In

WebEditingDelegate.h

webView:shouldChangeTypingStyle:toStyle:

Returns whether the user should be allowed to change the typing style in a web view.

```
- (BOOL)webView:(WebView *)webView shouldChangeTypingStyle:(DOMCSSStyleDeclaration *)currentStyle toStyle:(DOMCSSStyleDeclaration *)proposedStyle
```

Parameters*webView*

The web view that the user is editing.

currentStyle

The old style the user wants to change.

proposedStyle

The new style the user wants to set.

Return ValueYES if the user should be allowed to change the typing style in *webView* to *proposedStyle*; otherwise, NO.**Discussion**

You can implement this method to take some other action—for example, set the typing style to a different style—and return NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also- [webViewDidChangeTypingStyle:](#) (page 229)**Declared In**

WebEditingDelegate.h

webView:shouldDeleteDOMRange:

Returns whether the user should be allowed to delete a range of content.

```
- (BOOL)webView:(WebView *)webView shouldDeleteDOMRange:(DOMRange *)range
```

Parameters*webView*

The web view that the user is editing.

range

The range of the content to delete.

Return ValueYES if the user should be allowed to delete the content specified by *range*; otherwise, NO.**Discussion**

This method may perform an alternate action—for example, delete a different range—and return NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webViewDidChange:](#) (page 228)

Declared In

WebEditingDelegate.h

webView:shouldEndEditingInDOMRange:

Returns whether the user should be allowed to end editing.

```
- (BOOL)webView:(WebView *)webView shouldEndEditingInDOMRange:(DOMRange *)range
```

Parameters

webView

The web view that the user is editing.

range

Typically, the current selection, although it might not be. Use the *range* parameter to help determine whether the user can end editing.

Return Value

YES if the user should be allowed to end editing *webView*; otherwise, NO. If this method returns YES, *webView* ends editing and resigns as the first responder.

Discussion

This method is invoked when a web view attempts to resign as the first responder.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldBeginEditingInDOMRange:](#) (page 223)

- [webViewDidEndEditing:](#) (page 229)

Declared In

WebEditingDelegate.h

webView:shouldInsertNode:replacingDOMRange:givenAction:

Returns whether the user should be allowed to insert a node in place of a range of content.

```
- (BOOL)webView:(WebView *)webView shouldInsertNode:(DOMNode *)node
    replacingDOMRange:(DOMRange *)range givenAction:(WebViewInsertAction)action
```

Parameters

webView

The web view that the user is editing.

node

The content to insert.

range

The portion of the content that is replaced with *node*.

action

Indicates the type of user action that initiated the insertion.

Return Value

YES if the user should be allowed to insert *node* in *webView*; otherwise, NO.

Discussion

This method may perform an alternate action—for example, insert a different node—and return NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldInsertText:replacingDOMRange:givenAction:](#) (page 227)
- [webViewDidChange:](#) (page 228)

Declared In

WebEditingDelegate.h

webView:shouldInsertText:replacingDOMRange:givenAction:

Returns whether a user should be allowed to insert text in place of a range of content.

```
- (BOOL)webView:(WebView *)webView shouldInsertText:(NSString *)text
    replacingDOMRange:(DOMRange *)range givenAction:(WebViewInsertAction)action
```

Parameters

webView

The web view that the user is editing.

text

The text to insert.

range

The portion of the document that will be replaced with *text*.

action

Indicates the type of user action that initiated the insertion.

Return Value

YES if the user should be allowed to insert *text* in *webView*; otherwise, NO.

Discussion

This method may perform an alternate action—for example, insert different text—and return NO.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldInsertNode:replacingDOMRange:givenAction:](#) (page 226)
- [webViewDidChange:](#) (page 228)

Declared In

WebEditingDelegate.h

webViewDidBeginEditing:

Sent by the default notification center when the user begins editing the web view.

```
- (void)webViewDidBeginEditing:(NSNotification *)notification
```

Parameters

notification

Always set to [WebViewDidBeginEditingNotification](#) (page 200). You can retrieve the `WebView` object by sending `object` to *notification*.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webViewDidEndEditing:](#) (page 229)
- [webView:shouldBeginEditingInDOMRange:](#) (page 223)

Declared In

WebEditingDelegate.h

webViewDidChange:

Sent by the default notification center when the user changes content in the web view.

```
- (void)webViewDidChange:(NSNotification *)notification
```

Parameters

notification

Always set to [WebViewDidChangeNotification](#) (page 200). You can retrieve the `WebView` object by sending `object` to *notification*.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldDeleteDOMRange:](#) (page 225)
- [webView:shouldInsertNode:replacingDOMRange:givenAction:](#) (page 226)
- [webView:shouldInsertText:replacingDOMRange:givenAction:](#) (page 227)
- [webView:shouldDeleteDOMRange:](#) (page 225)

Declared In

WebEditingDelegate.h

webViewDidChangeSelection:

Sent by the default notification center when the user changes the selection in the web view.

```
- (void)webViewDidChangeSelection:(NSNotification *)notification
```

Parameters

notification

Always set to [WebViewDidChangeSelectionNotification](#) (page 201). You can retrieve the `WebView` object by sending `object` to *notification*.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldChangeSelectedDOMRange:toDOMRange:affinity:stillSelecting:](#) (page 224)

Declared In

WebEditingDelegate.h

webViewDidChangeTypingStyle:

Sent by the default notification center when the user changes the typing style in the web view.

- (void)webViewDidChangeTypingStyle:(NSNotification *)*notification*

Parameters

notification

Always set to [WebViewDidChangeTypingStyleNotification](#) (page 201). You can retrieve the `WebView` object by sending `object` to *notification*.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webView:shouldChangeTypingStyle:toStyle:](#) (page 225)

Declared In

WebEditingDelegate.h

webViewDidEndEditing:

Sent by the default notification center when the user stops editing the web view.

- (void)webViewDidEndEditing:(NSNotification *)*notification*

Parameters

notification

Always set to [WebViewDidEndEditingNotification](#) (page 201). You can retrieve the `WebView` object by sending `object` to *notification*.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webViewDidBeginEditing:](#) (page 228)

- [webView:shouldEndEditingInDOMRange:](#) (page 226)

Declared In

WebEditingDelegate.h

Constants

WebViewInsertAction

The type of user action that initiated a delegate message.

```
typedef enum {  
    WebViewInsertActionTyped,  
    WebViewInsertActionPasted,  
    WebViewInsertActionDropped,  
} WebViewInsertAction;
```

Constants

WebViewInsertActionTyped

Indicates the user inserted content by typing.

Available in Mac OS X v10.3 and later.

Declared in WebEditingDelegate.h.

WebViewInsertActionPasted

Indicates the user inserted content by pasting.

Available in Mac OS X v10.3 and later.

Declared in WebEditingDelegate.h.

WebViewInsertActionDropped

Indicates the user inserted content by dropping.

Available in Mac OS X v10.3 and later.

Declared in WebEditingDelegate.h.

Discussion

These constants are described in WebEditingDelegate.

Availability

Available in Mac OS X v10.3.9 and later.

WebFrameLoadDelegate Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebFrameLoadDelegate.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

WebView frame load delegates implement this informal protocol to be notified while frame loads are in progress. Delegates are notified when a frame load starts, when a page title or icon is loaded, when a redirect occurs, when a data source is committed, and when the change is complete. The [webView:didStartProvisionalLoadForFrame:](#) (page 238) method is invoked when a frame load starts, and the [webView:didFinishLoadForFrame:](#) (page 236) method is invoked when the change is done. However, depending on the content being loaded, some of the other methods defined in this protocol may be invoked multiple times. All the methods in this protocol are optional.

Tasks

State Change Messages

- [webView:didStartProvisionalLoadForFrame:](#) (page 238)
Invoked when a page load is in progress in a given frame.
- [webView:didFinishLoadForFrame:](#) (page 236)
Invoked when a page load completes.
- [webView:didCommitLoadForFrame:](#) (page 234)
Invoked when content starts arriving for a page load.
- [webView:willCloseFrame:](#) (page 238)
Invoked when a frame will be closed.
- [webView:didChangeLocationWithinPageForFrame:](#) (page 233)
Invoked when the scroll position within a frame changes.

Data Received Messages

- `webView:didReceiveTitle:forFrame:` (page 237)
Invoked when the page title of a frame loads or changes.
- `webView:didReceiveIcon:forFrame:` (page 236)
Invoked when a page icon changes.

Error Messages

- `webView:didFailProvisionalLoadWithError:forFrame:` (page 235)
Invoked if an error occurs when starting to load data for a page.
- `webView:didFailLoadWithError:forFrame:` (page 234)
Invoked when an error occurs loading a committed data source.

Client and Server Redirect Messages

- `webView:didCancelClientRedirectForFrame:` (page 232)
Invoked when a client redirect is cancelled.
- `webView:willPerformClientRedirectToURL:delay:fireDate:forFrame:` (page 239)
Invoked when a frame receives a client redirect and before it is fired.
- `webView:didReceiveServerRedirectForProvisionalLoadForFrame:` (page 237)
Invoked when a provisional data source for a frame receives a server redirect.

WebScript Messages

- `webView:didClearWindowObject:forFrame:` (page 233)
Invoked when the JavaScript window object in a frame is ready for loading.
- `webView>windowScriptObjectAvailable:` (page 239) **Deprecated in Mac OS X v10.4.11**
Invoked when a frame's scripting object for a page is available. (Use the `webView:didClearWindowObject:forFrame:` (page 233) method instead.)

Instance Methods

webView:didCancelClientRedirectForFrame:

Invoked when a client redirect is cancelled.

```
- (void)webView:(WebView *)sender didCancelClientRedirectForFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

This might happen if a frame changes locations before a pending client redirect is fired. The client redirect occurred in *frame*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:willPerformClientRedirectToURL:delay:fireDate:forFrame:](#) (page 239)

Declared In

WebFrameLoadDelegate.h

webView:didChangeLocationWithinPageForFrame:

Invoked when the scroll position within a frame changes.

```
- (void)webView:(WebView *)sender didChangeLocationWithinPageForFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

Typically, invoked when the user clicks on an anchor within a page. Additional information about the request can be obtained from the data source of *frame*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameLoadDelegate.h

webView:didClearWindowObject:forFrame:

Invoked when the JavaScript window object in a frame is ready for loading.

```
- (void)webView:(WebView *)sender didClearWindowObject:(WebScriptObject *)windowObject forFrame:(WebFrame *)frame
```

Parameters

sender

The web view sending this message.

windowObject

The cleared JavaScript window object.

frame

The frame containing the JavaScript window object.

Discussion

Use this method to set custom properties on the window object before the page is actually loaded. Every time a frame loads or is reloaded all DOM properties are cleared from the window object so the new page has a fresh window object to use. If the page you are loading depends on specific window object properties to exist, they should be added at this point before any scripts are executed.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebFrameLoadDelegat.e.h

webView:didCommitLoadForFrame:

Invoked when content starts arriving for a page load.

```
- (void)webView:(WebView *)senderdidCommitLoadForFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

This method is invoked when a data source transitions from a provisional to committed state—that is, once the data source of *frame* has received one byte or more of data. This method is invoked after a [webView:didStartProvisionalLoadForFrame:](#) (page 238) message but before a [webView:didFinishLoadForFrame:](#) (page 236) message is sent to the delegate.

In some cases, a single frame load may be committed more than once. This happens in the case of multipart/x-mixed-replace, also known as a “server push.” In this case, a single frame load results in multiple documents loaded in sequence. This method is invoked once for each document that is successfully loaded.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameLoadDelegat.e.h

webView:didFailLoadWithError:forFrame:

Invoked when an error occurs loading a committed data source.

```
- (void)webView:(WebView *)senderdidFailLoadWithError:(NSError *)errorforFrame:(WebFrame *)frame
```

Parameters*sender*

The web view containing the frame.

error

The type of error that occurred during the load.

frame

The frame being loaded.

Discussion

This method is called after the data source has been committed but resulted in an error.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also- [webView:didFinishLoadForFrame:](#) (page 236)**Declared In**

WebFrameLoadDelegate.h

webView:didFailProvisionalLoadWithError:forFrame:

Invoked if an error occurs when starting to load data for a page.

```
- (void)webView:(WebView *)senderdidFailProvisionalLoadWithError:(NSError *)errorforFrame:(WebFrame *)frame
```

Parameters*sender*

The web view containing the frame.

error

Specifies the type of error that occurred during the load.

frame

The frame being loaded.

Discussion

The frame continues to display the committed data source if there is one.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also- [webView:didFinishLoadForFrame:](#) (page 236)**Declared In**

WebFrameLoadDelegate.h

webView:didFinishLoadForFrame:

Invoked when a page load completes.

```
- (void)webView:(WebView *)senderdidFinishLoadForFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

This method is invoked when a location request for *frame* has completed; that is, when all the resources are done loading. Additional information about the request can be obtained from the data source of *frame*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:didStartProvisionalLoadForFrame:](#) (page 238)

Declared In

WebFrameLoadDelegate.h

webView:didReceiveIcon:forFrame:

Invoked when a page icon changes.

```
- (void)webView:(WebView *)senderdidReceiveIcon:(NSImage *)imageforFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

image

The page icon for a data source.

frame

The frame being loaded.

Discussion

This method may be invoked multiple times before all resources for *frame* are completely loaded. Sometimes a page uses a default icon or stored image that changes when the actual images is loaded.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameLoadDelegate.h

webView:didReceiveServerRedirectForProvisionalLoadForFrame:

Invoked when a provisional data source for a frame receives a server redirect.

```
- (void)webView:(WebView *)senderdidReceiveServerRedirectForProvisionalLoadForFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

A **server redirect** is when one URL location is redirected to another. Additional information about the new request can be obtained from the data source of *frame*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameLoadDelegate.h

webView:didReceiveTitle:forFrame:

Invoked when the page title of a frame loads or changes.

```
- (void)webView:(WebView *)senderdidReceiveTitle:(NSString *)titleforFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

title

The newly loaded title.

frame

The frame being loaded.

Discussion

This method may be invoked multiple times before all resources for *frame* are completely loaded. Delegates might implement this message to display the page title to the user.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebFrameLoadDelegate.h

webView:didStartProvisionalLoadForFrame:

Invoked when a page load is in progress in a given frame.

```
- (void)webView:(WebView *)senderdidStartProvisionalLoadForFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

This method is invoked when a new client request is made by *sender* to load a provisional data source for *frame*. This method may be invoked after sending [loadRequest:](#) (page 62) to a `WebFrame` object or as a consequence of the user clicking a link displayed in a web frame view. Delegates might implement this method to notify the user that a request is in progress. Additional information about the request can be obtained from the data source of *frame*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:didFinishLoadForFrame:](#) (page 236)

Declared In

WebFrameLoadDelegate.h

webView:willCloseFrame:

Invoked when a frame will be closed.

```
- (void)webView:(WebView *)senderwillCloseFrame:(WebFrame *)frame
```

Parameters

sender

The web view containing the frame.

frame

The frame being loaded.

Discussion

Invoked right before WebKit is done with *frame* and the objects it owns.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:willPerformClientRedirectToURL:delay:fireDate:forFrame:](#) (page 239)

Declared In

WebFrameLoadDelegate.h

webView:willPerformClientRedirectToURL:delay:fireDate:forFrame:

Invoked when a frame receives a client redirect and before it is fired.

```
- (void)webView:(WebView *)senderwillPerformClientRedirectToURL:(NSURL
    *)URLdelay:(NSTimeInterval)secondsfireDate:(NSDate *)dateforFrame:(WebFrame
    *)frame
```

Parameters

sender

The web view containing the frame.

URL

The redirect location.

seconds

The number of seconds from *date* before the redirect will be fired.

date

The date and time to fire the redirect.

frame

The frame where the redirect occurred.

Discussion

Delegates might implement this method to display progress while a client redirect is pending. If a client redirect is cancelled the [webView:didCancelClientRedirectForFrame:](#) (page 232) delegate method is invoked.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:didCancelClientRedirectForFrame:](#) (page 232)

Declared In

WebFrameLoadDelegate.h

webView>windowScriptObjectAvailable:

Invoked when a frame's scripting object for a page is available. (Use the [webView:didClearWindowObject:forFrame:](#) (page 233) method instead.) (Deprecated in Mac OS X v10.4.11.)

```
- (void)webView:(WebView *)sender>windowScriptObjectAvailable:(WebScriptObject
    *)windowScriptObject
```

Parameters

sender

The web view containing the frame.

windowScriptObject

The window object in the scripting environment.

Discussion

This method is invoked before the page is actually loaded.

Availability

Available in Mac OS X v10.3.9 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [windowScriptObject](#) (page 197)

Declared In

WebFrameLoadDelegate.h

WebJavaPlugIn Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebJavaPlugIn.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

The `WebJavaPlugIn` protocol provides methods to facilitate JNI access to the Java virtual machine via the plug-in.

Tasks

Getting and Setting Java Applets

- `webPlugInCallJava:isStatic:returnType:method:arguments:callingURL:exceptionDescription:` (page 241)
Sends a message directly to a Java object in a plug-in.
- `webPlugInGetApplet` (page 242)
Returns a `JObject` that represents a Java applet in a `WebPlugInContainer`.

Instance Methods

`webPlugInCallJava:isStatic:returnType:method:arguments:callingURL:exceptionDescription:`

Sends a message directly to a Java object in a plug-in.

```
- (jvalue)webPlugInCallJava:(jobject)object isStatic:(BOOL)isStatic returnType:(WebJNIReturnType)returnType method:(jmethodID)method arguments:(jvalue *)args callingURL:(NSURL *)url exceptionDescription:(NSString **)exceptionString
```

Parameters*object*

The Java instance receiving the message.

*isStatic*If YES, *method* is expected to be a class method.*returnType*

The return type of the Java method.

method

The Java method being called.

*args*The arguments for the method specified by *method*.*url*

The URL for the page that contains the JavaScript that is interacting with Java.

*exceptionString*A string for describing any exceptions thrown by Java. Pass `nil` if you do not want an exception description.**Return Value**

The return value of the Java method.

Discussion

This method is preferred over using JNI to send messages to Java applets, and is required to guarantee the correct thread will receive the message. Always invoke this method from within the main thread.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webPlugInGetApplet](#) (page 242)

Declared In

WebJavaPlugIn.h

webPlugInGetAppletReturns a `jobject` that represents a Java applet in a `WebPlugInContainer`.

```
- (jobject)webPlugInGetApplet
```

Return ValueA `jobject` that represents the applet.**Discussion**

Always invoke this method from within the main thread.

Availability

Available in Mac OS X v10.3.9 and later.

See Also

- [webPlugInCallJava:isStatic:returnType:method:arguments:callingURL:exceptionDescription:](#) (page 241)

Declared In

WebJavaPlugIn.h

Constants

WebJNIReturnType

The return type of the Java method.

```
typedef enum {  
    WebJNIReturnTypeInvalid = 0,  
    WebJNIReturnTypeVoid,  
    WebJNIReturnTypeObject,  
    WebJNIReturnTypeBoolean,  
    WebJNIReturnTypeByte,  
    WebJNIReturnTypeChar,  
    WebJNIReturnTypeShort,  
    WebJNIReturnTypeInt,  
    WebJNIReturnTypeLong,  
    WebJNIReturnTypeFloat,  
    WebJNIReturnTypeDouble  
} WebJNIReturnType;
```

Constants

WebJNIReturnTypeInvalid

The return type is invalid.

Available in Mac OS X v10.3 and later.

Declared in WebJavaPlugIn.h.

WebJNIReturnTypeVoid

The return type is void.

Available in Mac OS X v10.3 and later.

Declared in WebJavaPlugIn.h.

WebJNIReturnTypeObject

The return type is an object.

Available in Mac OS X v10.3 and later.

Declared in WebJavaPlugIn.h.

WebJNIReturnTypeBoolean

The return type is a Boolean value.

Available in Mac OS X v10.3 and later.

Declared in WebJavaPlugIn.h.

WebJNIReturnTypeByte

The return type is a byte value.

Available in Mac OS X v10.3 and later.

Declared in WebJavaPlugIn.h.

WebJNIReturnTypeChar

The return type is a `char` value.

Available in Mac OS X v10.3 and later.

Declared in `WebJavaPlugIn.h`.

WebJNIReturnTypeShort

The return type is a `short` value.

Available in Mac OS X v10.3 and later.

Declared in `WebJavaPlugIn.h`.

WebJNIReturnTypeInt

The return type is an `int` value.

Available in Mac OS X v10.3 and later.

Declared in `WebJavaPlugIn.h`.

WebJNIReturnTypeLong

The return type is a `long` value.

Available in Mac OS X v10.3 and later.

Declared in `WebJavaPlugIn.h`.

WebJNIReturnTypeFloat

The return type is a `float` value.

Available in Mac OS X v10.3 and later.

Declared in `WebJavaPlugIn.h`.

WebJNIReturnTypeDouble

The return type is a `double` value.

Available in Mac OS X v10.3 and later.

Declared in `WebJavaPlugIn.h`.

WebOpenPanelResultListener Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebUIDelegate.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

WebView user interface delegates that implement the [webView:runOpenPanelForFileButtonWithResultListener:](#) (page 302) method use the methods defined in this protocol to communicate with the listener object. The methods allow the delegate to send a cancel message, or set the selected file name.

Tasks

Setting a File Name

- [chooseFilename:](#) (page 246)
Displays a file open panel and returns the selected filename.
- [chooseFileNames:](#) (page 246)
Displays a file open panel and returns the multiple selected filenames.

Cancelling a File Open Operation

- [cancel](#) (page 246)
Invoked when a file open operation was cancelled.

Instance Methods

cancel

Invoked when a file open operation was cancelled.

- (void)cancel

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

chooseFilename:

Displays a file open panel and returns the selected filename.

- (void)chooseFilename:(NSString *)*fileName*

Parameters

fileName

The selected file name.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [chooseFileNames:](#) (page 246)

Declared In

WebUIDelegate.h

chooseFileNames:

Displays a file open panel and returns the multiple selected filenames.

- (void)chooseFileNames:(NSArray *)*fileNames*

Parameters

fileNames

An array of selected file names.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [chooseFilename:](#) (page 246)

Declared In

WebUIDelegate.h

WebPlugIn Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebPlugIn.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	WebKit Plug-In Programming Topics WebKit Objective-C Programming Guide

Overview

The `WebPlugIn` informal protocol defines methods that enable interaction between an application using the WebKit framework and any WebKit-based plug-ins it may use.

Tasks

Accessing the Scripting Environment

- [objectForWebScript](#) (page 250)
Returns an object that exposes the plug-in's scripting interface.

Using Plug-in State Information

- [webPlugInSetIsSelected:](#) (page 253)
Controls plug-in behavior based on its selection.

Controlling the Plug-in

- [webPlugInDestroy](#) (page 250)
Prepares the plug-in for deallocation.
- [webPlugInInitialize](#) (page 251)
Initializes the plug-in.
- [webPlugInStart](#) (page 253)
Tells the plug-in to start normal operation.

- [webPluginStop](#) (page 254)
Tells the plug-in to stop normal operation.

Main Resource Messages

- [webPluginMainResourceDidFailWithError:](#) (page 251)
Invoked when an error occurs loading the main resource.
- [webPluginMainResourceDidFinishLoading](#) (page 252)
Invoked when the connection successfully finishes loading data.
- [webPluginMainResourceDidReceiveData:](#) (page 252)
Invoked when the connection loads data incrementally.
- [webPluginMainResourceDidReceiveResponse:](#) (page 253)
Invoked when the connection receives sufficient data to construct the URL response for its request.

Instance Methods

objectForWebScript

Returns an object that exposes the plug-in's scripting interface.

- (id)objectForWebScript

Return Value

An object representing the plug-in's scripting interface.

Discussion

The methods of the object are exposed to the script environment. Messages sent to the returned object will be invoked in the scripting environment. See the [WebScripting Protocol Reference](#) (page 283) informal protocol for more details.

Availability

Available in Mac OS X v10.3.9 and later.

Related Sample Code

QT Capture Widget

SayIt

WebKitCIPlugin

WebKitPluginWithJavaScript

Declared In

WebPlugin.h

webPluginDestroy

Prepares the plug-in for deallocation.

- (void)webPluginDestroy

Discussion

Typically, this method releases the memory and other resources used by the plug-in. For example, if the plug-in retained a `WebPluginContainer` object, this method should release that object. Do not send any other messages to the plug-in after invoking this method, because calling this method destroys the plug-in. No other methods in this interface may be called after the application has called this method.

Availability

Available in Mac OS X v10.3.9 and later.

Related Sample Code

SayIt
WebKitCIPlugin

Declared In

`WebPlugin.h`

webPluginInitialize

Initializes the plug-in.

```
- (void)webPluginInitialize
```

Discussion

Tells the plug-in to perform one-time initialization. This method must be called only once per instance of the plug-in object, before any other methods in the protocol are called.

Availability

Available in Mac OS X v10.3.9 and later.

Related Sample Code

QT Capture Widget
SayIt
WebKitCIPlugin
WebKitPluginWithSimpleGUI

Declared In

`WebPlugin.h`

webPluginMainResourceDidFailWithError:

Invoked when an error occurs loading the main resource.

```
- (void)webPluginMainResourceDidFailWithError:(NSError *)error
```

Parameters

error

An error object containing details of why the connection failed to load the request successfully.

Discussion

This message is invoked when the underlying `NSURLConnection` object for the main resource sends the `connection:didFailWithError: message to its delegate`.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [webPluginMainResourceDidFinishLoading](#) (page 252)

Declared In

WebPlugin.h

webPluginMainResourceDidFinishLoading

Invoked when the connection successfully finishes loading data.

- (void)webPluginMainResourceDidFinishLoading

Discussion

This message is invoked when the `WebPluginShouldLoadMainResourceKey` plug-in command-line argument is set to `NO` and the underlying `NSURLConnection` object for the main resource sends the `connectionDidFinishLoading:` message to its delegate.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [webPluginMainResourceDidFailWithError:](#) (page 251)

Declared In

WebPlugin.h

webPluginMainResourceDidReceiveData:

Invoked when the connection loads data incrementally.

- (void)webPluginMainResourceDidReceiveData:(NSData *)data

Parameters

data

The newly available data.

Discussion

This message is invoked when the `WebPluginShouldLoadMainResourceKey` plug-in command-line argument is set to `NO` and the underlying `NSURLConnection` object for the main resource sends the `connection:didReceiveData:` message to its delegate.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [webPluginMainResourceDidReceiveResponse:](#) (page 253)

Declared In

WebPlugin.h

webPlugInMainResourceDidReceiveResponse:

Invoked when the connection receives sufficient data to construct the URL response for its request.

- (void)webPlugInMainResourceDidReceiveResponse:(NSURLResponse *)*response*

Parameters

response

The URL response for the connection's request.

Discussion

This message is invoked when the `WebPlugInShouldLoadMainResourceKey` plug-in command-line argument is set to `NO` and the underlying `NSURLConnection` object for the main resource sends the `connection:didReceiveResponse:` message to its delegate.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [webPlugInMainResourceDidReceiveData:](#) (page 252)

Declared In

WebPlugIn.h

webPlugInSetIsSelected:

Controls plug-in behavior based on its selection.

- (void)webPlugInSetIsSelected:(BOOL)*selected*

Parameters

isSelected

If `YES`, the plug-in is currently selected. Otherwise, it is not selected.

Discussion

This may be used, for example, to change the plug-in's appearance when it is selected by the user.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebPlugIn.h

webPlugInStart

Tells the plug-in to start normal operation.

- (void)webPlugInStart

Discussion

The plug-in usually begins its primary task (such as drawing, playing sounds, or animating) in this method. This method may be called more than once, provided that the application has already called [webPlugInInitialize](#) (page 251) and that each call to this method is followed later by a call to [webPlugInStop](#) (page 254).

Availability

Available in Mac OS X v10.3.9 and later.

Related Sample Code

WebKitCIPugin

Declared In

WebPlugin.h

webPluginStop

Tells the plug-in to stop normal operation.

- (void)webPluginStop

Discussion

This method may be called more than once, provided that the application has already called [webPluginInitialize](#) (page 251) and that each call to this method is preceded by a call to [webPluginStart](#) (page 253).

Availability

Available in Mac OS X v10.3.9 and later.

Related Sample Code

WebKitCIPugin

Declared In

WebPlugin.h

WebPlugInContainer Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebPlugInContainer.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	WebKit Plug-In Programming Topics WebKit Objective-C Programming Guide

Overview

`WebPlugInContainer` is an informal protocol that enables a plug-in to send messages to the application.

Tasks

Performing Actions on the Enclosing Container

- [webPlugInContainerLoadRequest:inFrame:](#) (page 256)
Loads a URL into a web frame.
- [webPlugInContainerShowStatus:](#) (page 257)
Tells the container to show a status message.

Obtaining Information About the Container

- [webFrame](#) (page 256)
Returns the `WebFrame` that contains the plug-in.
- [webPlugInContainerSelectionColor](#) (page 256)
Returns the plug-in selection color.

Instance Methods

webFrame

Returns the `WebFrame` that contains the plug-in.

```
- (WebFrame *)webFrame
```

Return Value

The `WebFrame` that contains the plug-in.

Discussion

Only implemented by containers that are based on the WebKit's plug-in architecture.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebPluginContainer.h`

webPluginContainerLoadRequest:inFrame:

Loads a URL into a web frame.

```
- (void)webPluginContainerLoadRequest:(NSURLRequest *)request inFrame:(NSString *)target
```

Parameters

request

The request that specifies the URL.

target

The frame into which the URL is loaded.

Discussion

If the frame specified by *target* is not found, a new window is opened, loaded with the URL request, and given the specified frame name. If *target* is `nil`, the frame enclosing the plug-in is loaded with the URL request.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebPluginContainer.h`

webPluginContainerSelectionColor

Returns the plug-in selection color.

```
- (NSColor *)webPluginContainerSelectionColor
```


Return Value

The plug-in selection color.

Discussion

The color should be used for any special drawing when the plug-in is selected.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebPluginContainer.h

webPluginContainerShowStatus:

Tells the container to show a status message.

```
- (void)webPluginContainerShowStatus:(NSString *)message
```

Parameters

message

The status message to be displayed.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebPluginContainer.h

WebPluginViewFactory Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebPluginViewFactory.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guides	WebKit Plug-In Programming Topics WebKit Objective-C Programming Guide
Related sample code	QT Capture Widget WebKitPluginStarter WebKitPluginWithSimpleGUI

Overview

A `WebPluginViewFactory` object is used to create an `NSView` for a plug-in. The principal class in a plug-in bundle must conform to this protocol.

Tasks

Creating the Plug-in View

+ [pluginViewWithArguments:](#) (page 259)
Creates a new plug-in view.

Class Methods

pluginViewWithArguments:

Creates a new plug-in view.

```
+ (NSView *)pluginViewWithArguments:(NSDictionary *)arguments
```

Parameters*arguments*

Arguments used in creating the view.

Return Value

The created view.

Discussion

This method returns an `NSView` object that conforms to the `WebPlugin` informal protocol. The arguments dictionary should be specified by the keys and objects described in “Constants” (page 260). This method is required.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In`WebPluginViewFactory.h`

Constants

Plug-in View Dictionary Keys

The following constants define the keys used to access the values in the *arguments* dictionary passed in to the `pluginViewWithArguments:` (page 259) method. Note that `WebPluginBaseURLKey` (page 260) and `WebPluginAttributesKey` will always correspond to data, the others may be `nil`.

```
extern NSString *WebPluginBaseURLKey;
extern NSString *WebPluginAttributesKey;
extern NSString *WebPluginContainerKey;
extern NSString *WebPluginContainingElementKey;
extern NSString *WebPluginShouldLoadMainResourceKey;
```

Constants`WebPluginBaseURLKey`The base URL of the document containing the plug-in's view. *Required key.*

Available in Mac OS X v10.3 and later.

Declared in `WebPluginViewFactory.h`.`WebPluginAttributesKey`

The `NSDictionary` object containing all names and values of all attributes of the plug-in's associated HTML element, as well as all names and values of the parameters to be passed to the plug-in. For example, this dictionary will contain all `PARAM` elements within an `APPLET` element. If attribute and parameter names conflict, the attributes of an element take precedence over any of its parameters. All keys and values in this dictionary must be of type `NSString`. *Required key.*

Available in Mac OS X v10.3 and later.

Declared in `WebPluginViewFactory.h`.

WebPluginContainerKey

An object that conforms to the `WebPluginContainer` informal protocol. This object is used for callbacks from the plug-in to the enclosing application. If `WebPluginContainerKey` is `nil`, no callbacks will occur.

Available in Mac OS X v10.3 and later.

Declared in `WebPluginViewFactory.h`.

WebPluginContainingElementKey

If an element of the page's Document Object Model was used to specify the plug-in, this will contain that element. Otherwise, it will be `nil`.

Available in Mac OS X v10.3 and later.

Declared in `WebPluginViewFactory.h`.

WebPluginShouldLoadMainResourceKey

A Boolean value indicating whether the plug-in should load its own main resource (the `src` URL, in most cases). If `YES`, the plug-in should load its own main resource. If `NO`, the plug-in uses the data provided by WebKit and the `webPluginMainResourceDidReceiveData:` message is sent to the plug-in when data is received. The default value is `NO`.

Available in Mac OS X v10.6 and later.

Declared in `WebPluginViewFactory.h`.

WebPolicyDecisionListener Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebPolicyDelegate.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

This protocol enables `WebView` policy delegates to communicate with listener objects. A listener object conforming to this protocol is passed as one of the arguments to web view policy delegate methods.

This protocol allows delegates to handle download decisions asynchronously. For example, the policy delegate may display a sheet, and the listener object gets notified only after the user clicks an OK or Cancel button. You do not directly create objects that conform to this protocol.

Tasks

Making Resource-Usage Decisions

- [download](#) (page 264)
Tells the listener to download the resource instead of displaying it.
- [ignore](#) (page 264)
Tells the listener to ignore the resource.
- [use](#) (page 264)
Tells the listener to use the resource.

Instance Methods

download

Tells the listener to download the resource instead of displaying it.

```
- (void)download
```

Discussion

This method converts a location change that may be in progress to a download operation without having to stop and restart the download. You might invoke this method based on the content's MIME type.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebPolicyDelegate.h

ignore

Tells the listener to ignore the resource.

```
- (void)ignore
```

Discussion

You might invoke this method to handle the resource request yourself. For example, you might want to open a new window, open a window behind the current window, open a URL in an external application, or show a file URL location in the Finder.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebPolicyDelegate.h

use

Tells the listener to use the resource.

```
- (void)use
```

Discussion

If there are pending policy decisions, the next policy delegate method has the opportunity to decide what to do with the resource. This will be either the next navigation policy delegate (if there is a redirect), or the content policy delegate. If there are no pending policy decisions, the resource will be displayed if possible. If there is no document view available to display the resource, then the [webView:unableToImplementPolicyWithError:frame:](#) (page 270) message will be sent to the web view policy delegate with an appropriate error. Invoking this method creates any new windows needed to handle the resource.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

+ [registerViewClass:representationClass:forMIMETYPE:](#) (page 145)

Declared In

WebPolicyDelegate.h

WebPolicyDelegate Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebPolicyDelegate.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

The `WebPolicyDelegate` informal protocol works with the `WebPolicyDecisionListener` protocol to modify the policy decisions that the `WebView` class makes when handling URLs or the data objects they represent. The methods in this protocol are typically invoked in the following order.

1. The `webView:decidePolicyForNewWindowAction:request:newFrameName:decisionListener:` (page 270) method is invoked once for every load.
2. The `webView:decidePolicyForNavigationAction:request:frame:decisionListener:` (page 269) method may be invoked zero or more times after a load started. This method is invoked every time a server redirect is encountered unless blocked by an earlier policy decision.
3. The `webView:decidePolicyForMIMEType:request:frame:decisionListener:` (page 268) method is invoked after the MIME type of the content is known unless this method is blocked by an earlier policy decision.
4. The `webView:unableToImplementPolicyWithError:frame:` (page 270) method is invoked when an error occurs implementing a policy decision.

Tasks

Making Content Decisions

- `webView:decidePolicyForMIMEType:request:frame:decisionListener:` (page 268)
Decides whether to display content with a given MIME type.

Making Navigation Decisions

- [webView:decidePolicyForNavigationAction:request:frame:decisionListener:](#) (page 269)
Routes a navigation action internally or to an external viewer.

Making New Window Decisions

- [webView:decidePolicyForNewWindowAction:request:newFrameName:decisionListener:](#) (page 270)
Decides whether to allow a targeted navigation event, such as opening a link in a new window.

Handling Errors

- [webView:unableToImplementPolicyWithError:frame:](#) (page 270)
Handles or drops events that were rejected by a policy maker.

Instance Methods

webView:decidePolicyForMIMETYPE:request:frame:decisionListener:

Decides whether to display content with a given MIME type.

```
- (void)webView:(WebView *)webView decidePolicyForMIMETYPE:(NSString *)type
  request:(NSURLRequest *)request frame:(WebFrame *)frame decisionListener:(id
  < WebPolicyDecisionListener >)listener
```

Parameters

webView

The associated web view.

type

The MIME type of the content.

request

The request to load the content.

frame

The frame for displaying the content.

listener

The object that receives the policy decision.

Discussion

This method is invoked during the process of loading content for *request* after the [webView:didStartProvisionalLoadForFrame:](#) (page 238) method in the `WebFrameLoadDelegate` informal protocol is called by the `WebView` object. The web view implements a policy decision by sending one of the `WebPolicyDecisionListener` protocol messages to *listener*.

If you do not implement this method, the default behavior is used. The listener is told to ignore the MIME type unless *webView* specifies it can handle the type in its [canShowMIMETYPE:](#) (page 143) method.

In some rare cases, multiple responses may be received for a single resource. This happens in the case of multipart/x-mixed-replace, also known as a “server push.” In this case, this method will be invoked multiple times.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebPolicyDelegate.h

webView:decidePolicyForNavigationAction:request:frame:decisionListener:

Routes a navigation action internally or to an external viewer.

```
- (void)webView:(WebView *)webView decidePolicyForNavigationAction:(NSDictionary *)actionInformation request:(NSURLRequest *)request frame:(WebFrame *)frame decisionListener:(id < WebPolicyDecisionListener >)listener
```

Parameters

webView

The `WebView` object for which this object is the policy delegate.

actionInformation

A description of the action that triggered the navigation request. The possible key-value pairs in this dictionary are defined in “[Action Dictionary Keys](#)” (page 271).

request

The request for which the navigation is made.

frame

The `WebFrame` object in which the action occurred.

listener

The `WebPolicyDecisionListener` object that receives the policy decision.

Discussion

This method is invoked when a navigation decision needs to be made. The web view implements a policy decision by sending one of the `WebPolicyDecisionListener` protocol messages to *listener*. This method is invoked whenever a server redirect is encountered, and before loading starts.

If you do not implement this method, the default behavior is used. The listener handles the navigation internally if the request is for an error page or if the `canHandleRequest:` method of the `NSURLConnection` class returns YES when passed *request*. Otherwise, the listener ignores the navigation, and it is handled externally.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebPolicyDelegate.h

webView:decidePolicyForNewWindowAction:request:newFrameName:decisionListener:

Decides whether to allow a targeted navigation event, such as opening a link in a new window.

```
- (void)webView:(WebView *)webView decidePolicyForNewWindowAction:(NSDictionary *)actionInformation request:(NSURLRequest *)request newFrameName:(NSString *)frameName decisionListener:(id < WebPolicyDecisionListener >)listener
```

Parameters

webView

The `WebView` object for which this object is the policy delegate.

actionInformation

A description of the action that triggered the navigation request. The possible key-value pairs in this dictionary are defined in [“Making content decisions”](#) (page 267).

request

The request for which the new window action is performed.

frameName

The name of the new frame that contains the content returned from the request.

listener

The `WebPolicyDecisionListener` object that receives the policy decision.

Discussion

This method is invoked when a targeted navigation decision needs to be made. A targeted navigation typically opens a new window to display content. The receiver implements a policy decision by sending one of the `WebPolicyDecisionListener` protocol messages to *listener*. This method allows delegates to modify the behavior of targeted links which normally open a new window. Delegates might do something else, such as download or present the content in a special way. If this method sends [use](#) (page 264) to *listener* then the new window will be opened, and

[webView:decidePolicyForNavigationAction:request:frame:decisionListener:](#) (page 269) will be invoked with a `WebNavigationTypeOther` (page 273) as the value for the `WebActionNavigationTypeKey` (page 271) key in the action dictionary.

The default behavior sends [use](#) (page 264) to *listener*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

`WebPolicyDelegate.h`

webView:unableToImplementPolicyWithError:frame:

Handles or drops events that were rejected by a policy maker.

```
- (void)webView:(WebView *)webView unableToImplementPolicyWithError:(NSError *)error frame:(WebFrame *)frame
```

Parameters

webView

The `WebView` object for which this object is the policy delegate.

error

The error that occurred.

frame

The frame in which the error occurred.

Discussion

Delegates might implement this method to display or log an error message. If you do not implement this method, no action is taken.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebPolicyDelegate.h

Constants

Action Dictionary Keys

Keys that might appear in a dictionary passed as the `actionInformation` parameter to the `webView:decidePolicyForNavigationAction:request:frame:decisionListener:` (page 269) and `webView:decidePolicyForNewWindowAction:request:newFrameName:decisionListener:` (page 270) methods.

```
extern NSString *WebActionNavigationTypeKey;
extern NSString *WebActionElementKey;
extern NSString *WebActionButtonKey;
extern NSString *WebActionModifierFlagsKey;
extern NSString *WebActionOriginalURLKey;
```

Constants

`WebActionNavigationTypeKey`

The navigation type of the action. Can be any of the values defined in “WebNavigationType” below.

Available in Mac OS X v10.2 and later.

Declared in `WebPolicyDelegate.h`.

`WebActionElementKey`

A dictionary containing element information. See *WebView Class Reference* for a description of the key-value pairs in this dictionary.

Available in Mac OS X v10.2 and later.

Declared in `WebPolicyDelegate.h`.

`WebActionButtonKey`

An `NSNumber` object where 0 indicates the left button, 1 indicates the middle button, and 2 indicates the right button.

Available in Mac OS X v10.2 and later.

Declared in `WebPolicyDelegate.h`.

WebActionModifierFlagsKey

An unsigned number that indicates the modifier flag.

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

WebActionOriginalURLKey

The URL that initiated the action.

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

WebNavigationType

Possible values for the [WebActionNavigationTypeKey](#) (page 271) key that appears in an action dictionary.

```
typedef enum {
    WebNavigationTypeLinkClicked,
    WebNavigationTypeFormSubmitted,
    WebNavigationTypeBackForward,
    WebNavigationTypeReload,
    WebNavigationTypeFormResubmitted,
    WebNavigationTypeOther
} WebNavigationType;
```

Constants

WebNavigationTypeLinkClicked

A link (an href) was clicked.

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

WebNavigationTypeFormSubmitted

A form was submitted.

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

WebNavigationTypeBackForward

The user clicked back or forward button.

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

WebNavigationTypeReload

The user hit the reload button.

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

WebNavigationTypeFormResubmitted

A form was resubmitted (through a back, forward or reload action).

Available in Mac OS X v10.2 and later.

Declared in WebPolicyDelegate.h.

`WebNavigationTypeOther`

Navigation is taking place for some other reason.

Available in Mac OS X v10.2 and later.

Declared in `WebPolicyDelegate.h`.

WebResourceLoadDelegate Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebResourceLoadDelegate.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

Web view resource load delegates implement this informal protocol to be notified on the progress of loading individual resources. Note that there can be hundreds of resources, such as images and other media, per page. So, if you just want to get page loading status see the `WebFrameLoadDelegate` protocol.

There's a separate client request and server response made for each resource on a page. By implementing the `webView:identifierForInitialRequest:fromDataSource:` (page 276) method, resource load delegates provide a tracking object used to identify individual resources in subsequent calls to delegate methods. Delegates are then notified when resource loading starts, when data is incrementally received, when any load errors occur, and when the load is complete. Delegates may also change a request before it is sent. In some cases, depending on the page content and server redirects, methods defined in this protocol may be invoked multiple times (see individual method descriptions for more details). All the methods in this protocol are optional.

Tasks

Setting Identifiers

- `webView:identifierForInitialRequest:fromDataSource:` (page 276)
Returns an identifier object used to track the progress of loading a single resource.

Loading Content

- `webView:resource:willSendRequest:redirectResponse:fromDataSource:` (page 281)
Invoked before a request is initiated for a resource and returns a possibly modified request.
- `webView:resource:didFinishLoadingFromDataSource:` (page 278)
Invoked when all of the data for a given resource is loaded.

- [webView:resource:didReceiveResponse:fromDataSource:](#) (page 280)
Invoked after a resource has been loaded.
- [webView:resource:didReceiveContentLength:fromDataSource:](#) (page 280)
Invoked when some of the data for a given resource has arrived.
- [webView:resource:didFailLoadingWithError:fromDataSource:](#) (page 278)
Invoked when a resource failed to load.
- [webView:pluginFailedWithError:dataSource:](#) (page 277)
Invoked when a plug-in fails to load.

Authenticating Resources

- [webView:resource:didReceiveAuthenticationChallenge:fromDataSource:](#) (page 279)
Invoked when an authentication challenge has been received for a resource.
- [webView:resource:didCancelAuthenticationChallenge:fromDataSource:](#) (page 277)
Invoked when an authentication challenge for a resource was canceled.

Instance Methods

webView:identifierForInitialRequest:fromDataSource:

Returns an identifier object used to track the progress of loading a single resource.

```
- (id)webView:(WebView *)sender identifierForInitialRequest:(NSURLRequest *)request
fromDataSource:(WebDataSource *)dataSource
```

Parameters

sender

The web view that sent this message.

request

The request that initiated this load for *dataSource*.

dataSource

The data source for this web view.

Return Value

An identifier object that is retained by *sender* and passed as an parameter to all other delegate messages pertaining to this resource.

Discussion

Delegates might implement this method to begin tracking the progress of loading an individual resource. Note that this method is invoked once per load where as the [webView:resource:willSendRequest:redirectResponse:fromDataSource:](#) (page 281) method may be invoked multiple times.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebResourceLoadDelegate.h

webView:pluginFailedWithError:dataSource:

Invoked when a plug-in fails to load.

```
- (void)webView:(WebView *)sender pluginFailedWithError:(NSError *)error
    dataSource:(WebDataSource *)dataSource
```

Parameters*sender*

The web view that sent this message.

error

The error that occurred during the process of loading that resource.

The *userInfo* dictionary of *error* may contain additional information about the failure. If the *userInfo* dictionary is not *nil*, it may contain some or all of these key-value pairs. The value of the `NSErrorFailingURLKey` key is a URL string of the SRC attribute. The value of the `WebKitErrorPlugInNameKey` (page 324) key is a string containing the plug-in's name. The value for the `WebKitErrorPlugInPageURLStringKey` (page 325) key is a URL string of the PLUGINSPAGE attribute. The value of the `WebKitErrorMIMETYPEKey` (page 324) key is a string of the TYPE attribute.

dataSource

The data source for this web view.

Discussion

This method might be invoked if a plug-in is not found, fails to load, or is not available for some reason. Delegates might implement this method to display or log a detailed error message. If you do not implement this method, no action is taken.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also- [webView:resource:didFailLoadingWithError:fromDataSource:](#) (page 278)**Declared In**

WebResourceLoadDelegate.h

webView:resource:didCancelAuthenticationChallenge:fromDataSource:

Invoked when an authentication challenge for a resource was canceled.

```
- (void)webView:(WebView *)sender resource:(id)identifier
    didCancelAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge
    fromDataSource:(WebDataSource *)dataSource
```

Parameters*sender*

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

challenge

The authentication challenge that was canceled.

dataSource

The data source for this web view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:resource:didReceiveAuthenticationChallenge:fromDataSource:](#) (page 279)

Declared In

WebResourceLoadDelegate.h

webView:resource:didFailLoadingWithError:fromDataSource:

Invoked when a resource failed to load.

```
- (void)webView:(WebView *)sender resource:(id)identifier
  didFailLoadingWithError:(NSError *)error fromDataSource:(WebDataSource
  *)dataSource
```

Parameters

sender

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

error

The error that occurred loading that resource.

dataSource

The data source for this web view.

Discussion

Delegates might implement this method to display or log a detailed error message.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:pluginFailedWithError:dataSource:](#) (page 277)

Declared In

WebResourceLoadDelegate.h

webView:resource:didFinishLoadingFromDataSource:

Invoked when all of the data for a given resource is loaded.

```
- (void)webView:(WebView *)sender resource:(id)identifier
  didFinishLoadingFromDataSource:(WebDataSource *)dataSource
```

Parameters*sender*

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

dataSource

The data source for this web view.

Discussion

The *identifier* parameter is used to track the resource being loaded by *dataSource*. Delegates might implement this method to update the load status of an individual resource.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:resource:willSendRequest:redirectResponse:fromDataSource:](#) (page 281)

Declared In

WebResourceLoadDelegate.h

webView:resource:didReceiveAuthenticationChallenge:fromDataSource:

Invoked when an authentication challenge has been received for a resource.

```
- (void)webView:(WebView *)sender resource:(id)identifier
  didReceiveAuthenticationChallenge:(NSURLAuthenticationChallenge *)challenge
  fromDataSource:(WebDataSource *)dataSource
```

Parameters*sender*

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

challenge

The authentication challenge that was received.

dataSource

The data source for this web view.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:resource:didCancelAuthenticationChallenge:fromDataSource:](#) (page 277)

Declared In

WebResourceLoadDelegate.h

webView:resource:didReceiveContentLength:fromDataSource:

Invoked when some of the data for a given resource has arrived.

```
- (void)webView:(WebView *)sender resource:(id)identifier
    didReceiveContentLength:(NSUInteger)length fromDataSource:(WebDataSource
    *)dataSource
```

Parameters

sender

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

length

The amount of incremental data received for this resource—the amount of data loaded since the last time this method was invoked for this resource, not the total amount received for this resource.

The *length* parameter type was changed from type `unsigned int` to type `NSUInteger` in Mac OS X v10.5.

dataSource

The data source for this web view.

Discussion

Delegates might implement this method to update the load status of an individual resource.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebResourceLoadDelegate.h

webView:resource:didReceiveResponse:fromDataSource:

Invoked after a resource has been loaded.

```
- (void)webView:(WebView *)sender resource:(id)identifier
    didReceiveResponse:(NSURLResponse *)response fromDataSource:(WebDataSource
    *)dataSource
```

Parameters

sender

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

response

The reply that was received.

dataSource

The data source for this web view.

Discussion

In some rare cases, multiple responses may be received for a single resource. This happens in the case of multipart/x-mixed-replace, also known as a *server push*. In this case, delegates should assume that the progress of loading this resource restarts, and the expected content length may change.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebResourceLoadDelegate.h

webView:resource:willSendRequest:redirectResponse:fromDataSource:

Invoked before a request is initiated for a resource and returns a possibly modified request.

```
- (NSURLRequest *)webView:(WebView *)sender resource:(id)identifier
  willSendRequest:(NSURLRequest *)request redirectResponse:(NSURLResponse
  *)redirectResponse fromDataSource:(WebDataSource *)dataSource
```

Parameters

sender

The web view that sent this message.

identifier

An identifier object used to track the resource being loaded by *dataSource*.

request

The request that is sent.

redirectResponse

The redirect server response. If *nil*, there is no redirect in progress.

dataSource

The data source for this web view.

Return Value

A possibly modified request.

Discussion

Delegates might implement this method to modify resource requests before they are sent. Note that this method might be invoked multiple times per load (as a result of a server redirect) whereas the [webView:identifierForInitialRequest:fromDataSource:](#) (page 276) method is invoked once.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:resource:didFinishLoadingFromDataSource:](#) (page 278)

Declared In

WebResourceLoadDelegate.h

WebScripting Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebScriptObject.h
Availability	Available in Mac OS X v10.3.9 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

`WebScripting` is an informal protocol that defines methods that classes can implement to export their interfaces to a WebScript environment such as JavaScript.

Not all properties and methods are exported to JavaScript by default. The object needs to implement the class methods described below to specify the properties and methods to export. Furthermore, a method is not exported if its return type and all its parameters are not Objective-C objects or scalars.

Method argument and return types that are Objective-C objects will be converted to appropriate types for the scripting environment. For example:

- `nil` is converted to undefined.
- `NSNumber` objects will be converted to JavaScript numbers.
- `NSString` objects will be converted to JavaScript strings.
- `NSArray` objects will be mapped to special read-only arrays.
- `NSNull` will be converted to JavaScript's `null`.
- `WebUndefined` will be converted to undefined.
- `WebScriptObject` instances will be unwrapped for the scripting environment.

Instances of all other classes will be wrapped before being passed to the script, and unwrapped as they return to Objective-C. Primitive types such as `int` and `char` are cast to a numeric in JavaScript.

Access to an object's attributes, such as instance variables, is managed by key-value coding (KVC). The KVC methods `setValue:forKey:` and `valueForKey:` are used to access the attributes of an object from the scripting environment. Additionally, the scripting environment can attempt any number of attribute requests or method invocations that are not exported by your class. You can manage these requests by overriding the `setValue:forUndefinedKey:` and `valueForUndefinedKey:` methods from the key-value coding protocol.

Exceptions can be raised from the scripting environment by sending a `throwException:` (page 126) message to the relevant `WebScriptObject` instance. The method raising the exception must be within the scope of the script invocation.

Tasks

Getting Attributes

- + `webScriptNameForKey:` (page 285)
Returns the scripting environment name for an attribute specified by a key.
- + `webScriptNameForSelector:` (page 286)
Returns the scripting environment name for a selector.
- + `isSelectorExcludedFromWebScript:` (page 285)
Returns whether a selector should be hidden from the scripting environment.
- + `isKeyExcludedFromWebScript:` (page 284)
Returns whether a key should be hidden from the scripting environment.

Invoking Methods

- `invokeDefaultMethodWithArguments:` (page 287)
Executes when a script attempts to invoke a method on an exposed object directly.
- `invokeUndefinedMethodFromWebScript:withArguments:` (page 287)
Handles undefined method invocation from the scripting environment.

Finalizing

- `finalizeForWebScript` (page 286)
Performs cleanup when the scripting environment is reset.

Class Methods

isKeyExcludedFromWebScript:

Returns whether a key should be hidden from the scripting environment.

```
+ (BOOL)isKeyExcludedFromWebScript:(const char *)name
```

Parameters

name

The name of the attribute.

Return Value

YES if the attribute specified by *name* should be hidden from the scripting environment; otherwise, NO.

Discussion

The default value is YES.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

isSelectorExcludedFromWebScript:

Returns whether a selector should be hidden from the scripting environment.

```
+ (BOOL)isSelectorExcludedFromWebScript:(SEL)aSelector
```

Parameters

aSelector

The selector.

Return Value

YES if the selector specified by *aSelector* should be hidden from the scripting environment; otherwise, NO.

Discussion

Only methods with valid parameters and return types are exported to the WebKit JavaScript environment. The valid types are Objective-C objects and scalars. The default value is YES.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

webScriptNameForKey:

Returns the scripting environment name for an attribute specified by a key.

```
+ (NSString *)webScriptNameForKey:(const char *)name
```

Parameters

name

The name of the attribute.

Return Value

The name used to represent the attribute in the scripting environment.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

webViewScriptNameForSelector:

Returns the scripting environment name for a selector.

```
+ (NSString *)webViewScriptNameForSelector:(SEL)aSelector
```

Parameters

aSelector

The selector.

Return Value

The name used to represent the selector in the scripting environment.

Discussion

It is your responsibility to ensure that the returned name is unique to the script invoking this method. If this method returns `nil` or you do not implement it, the default name for the selector is constructed as follows:

- A colon (":") in the Objective-C selector is replaced by an underscore ("_").
- An underscore in the Objective-C selector is prefixed with a dollar sign ("\$").
- A dollar sign in the Objective-C selector is prefixed with another dollar sign.

The following table shows examples of how the default name is constructed:

Objective-C selector	Default script name for selector
setFlag:	setFlag_
setFlag:forKey: withAttributes:	setFlag_forKey_withAttributes_
propertiesForExample_Object:	propertiesForExample\$_Object_
set_\$_forKey: withDictionary:	set_\$_\$_forKey_withDictionary_

Since the default construction for a method name can be confusing depending on its Objective-C name, you should implement this method and return a more human-readable name.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

Instance Methods

finalizeForWebScript

Performs cleanup when the scripting environment is reset.

```
- (void)finalizeForWebScript
```

Discussion

This method is invoked on objects exposed to the scripting environment just before the scripting environment is reset. After invocation, the receiving object will no longer be referenced by the scripting environment. Further references to `WebScriptObject` instances created by the exposed object will be invalid and may produce unpredictable results.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebScriptObject.h`

invokeDefaultMethodWithArguments:

Executes when a script attempts to invoke a method on an exposed object directly.

```
- (id)invokeDefaultMethodWithArguments:(NSArray *)args
```

Parameters

args

The arguments to be passed to the default method.

Return Value

The result of invoking the default method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

`WebScriptObject.h`

invokeUndefinedMethodFromWebScript:withArguments:

Handles undefined method invocation from the scripting environment.

```
- (id)invokeUndefinedMethodFromWebScript:(NSString *)namewithArguments:(NSArray *)args
```

Parameters

name

The name of the undefined method.

args

The arguments passed to the undefined method.

Return Value

The result of invoking the undefined method.

Discussion

This method is invoked when a script attempts to invoke a method not directly exported to the scripting environment. You should return the result of the invocation, converted appropriately for the scripting environment.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebScriptObject.h

WebUIDelegate Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/WebKit.framework
Declared in	WebKit/WebUIDelegate.h
Availability	Available in Mac OS X v10.2 with Safari 1.0 and later. Available in Mac OS X v10.2.7 and later.
Companion guide	WebKit Objective-C Programming Guide

Overview

Web view user interface delegates implement this informal protocol to control the opening of new windows, augment the behavior of default menu items displayed when the user clicks elements, and perform other user interface–related tasks. These methods can be invoked as a result of handling JavaScript or other plug-in content. Delegates that display more than one web view per window, for example, need to implement some of these methods to handle that case. The default implementation assumes one window per web view, so non-conventional user interfaces might implement a user interface delegate.

Tasks

Creating and Closing Windows

- [webView:createWebViewModalDialogWithRequest:](#) (page 293)
Creates a modal window containing a web view that loads the specified request.
- [webViewRunModal:](#) (page 313)
Displays a web view in a modal window.
- [webView:createWebViewWithRequest:](#) (page 294)
Creates a window containing a web view to load the specified request.
- [webViewClose:](#) (page 308)
Closes a web view in a window.

Moving and Resizing Windows

- [webViewIsResizable:](#) (page 312)
Returns a Boolean value indicating whether a web view’s window can be resized.

- `webView:setResizable:` (page 304)
Sets whether a web view's window can be resized.
- `webView:setFrame:` (page 303)
Sets the frame rectangle of a web view's window to the specified frame size.
- `webViewFrame:` (page 311)
Returns the frame rectangle of a web view's window.

Moving and Resizing Content Views

- `webView:setContentRect:` (page 303) **Deprecated in Mac OS X v10.4.11**
Sets the window's content view frame to the specified content rectangle. (**Deprecated.** Content rectangle calculations are automatic.)
- `webViewContentRect:` (page 309) **Deprecated in Mac OS X v10.4.11**
Returns a web view window's content rectangle. (**Deprecated.** Content rectangle calculations are automatic.)

Making Windows Key and Main

- `webViewFocus:` (page 310)
Brings a web view's window to the front and makes it the active window.
- `webViewUnfocus:` (page 314)
Relinquishes focus on a web view's window.

Ordering Windows

- `webViewShow:` (page 313)
Displays a web view's window and moves it to the front.

Working with the Responder Chain

- `webViewFirstResponder:` (page 309)
Returns the first responder of the web view's window.
- `webView:makeFirstResponder:` (page 296)
Sets the first responder of a web view's window to the specified view.

Handling Mouse Events

- `webView:mouseDidMoveOverElement:modifierFlags:` (page 297)
Updates information about the element the user is mousing over.
- `webView:contextMenuItemsForElement:defaultMenuItems:` (page 292)
Returns menu items to display in an element's contextual menu.

Opening Panels

- [webView:runJavaScriptAlertPanelWithMessage:initiatedByFrame:](#) (page 299)
Displays a JavaScript alert panel containing the specified message.
- [webView:runJavaScriptConfirmPanelWithMessage:initiatedByFrame:](#) (page 300)
Displays a JavaScript confirmation panel with the specified message.
- [webView:runJavaScriptTextInputPanelWithPrompt:defaultText:initiatedByFrame:](#) (page 301)
Displays a JavaScript text input panel and returns the entered text.
- [webView:runOpenPanelForFileButtonWithResultListener:](#) (page 302)
Displays an open panel for a file input control.
- [webView:runOpenPanelForFileButtonWithResultListener:allowMultipleFiles:](#) (page 302)
Displays an open panel for a file input control.
- [webView:runBeforeUnloadConfirmPanelWithMessage:initiatedByFrame:](#) (page 298)
Displays a confirmation panel containing the specified message before a window closes.
- [webView:runJavaScriptAlertPanelWithMessage:](#) (page 298) **Deprecated in Mac OS X v10.4.11**
Displays a JavaScript alert panel. (**Deprecated.** Use [webView:runJavaScriptAlertPanelWithMessage:initiatedByFrame:](#) (page 299) instead.)
- [webView:runJavaScriptConfirmPanelWithMessage:](#) (page 299) **Deprecated in Mac OS X v10.4.11**
Displays a JavaScript confirm panel. (**Deprecated.** Use [webView:runJavaScriptConfirmPanelWithMessage:initiatedByFrame:](#) (page 300) instead.)
- [webView:runJavaScriptTextInputPanelWithPrompt:defaultText:](#) (page 300) **Deprecated in Mac OS X v10.4.11**
Displays a JavaScript text input panel and returns the entered text. (**Deprecated.** Use [webView:runJavaScriptTextInputPanelWithPrompt:defaultText:initiatedByFrame:](#) (page 301) instead.)

Displaying Status Messages

- [webView:setStatusText:](#) (page 305)
Sets the status message displayed by a web view's window, if any, to the specified text.
- [webViewStatusText:](#) (page 314)
Returns the current status message from a web view's window.

Managing Toolbars and the Status Bar

- [webViewAreToolbarsVisible:](#) (page 308)
Returns a Boolean value indicating whether any toolbars are visible in a web view's window.
- [webView:setToolbarsVisible:](#) (page 305)
Sets whether a web view's toolbars should be visible.
- [webViewIsStatusBarVisible:](#) (page 312)
Returns a Boolean value indicating whether the status bar in a web view's window is visible.
- [webView:setStatusBarVisible:](#) (page 304)
Sets the visibility of the status bar in a web view's window.

Controlling Drag Behavior

- [webView:dragDestinationActionMaskForDraggingInfo:](#) (page 294)
Returns a mask indicating which drag operations are allowed by the sender.
- [webView:dragSourceActionMaskForPoint:](#) (page 295)
Returns a mask indicating which drag-source actions are allowed for a drag that begins at the specified location.
- [webView:willPerformDragDestinationAction:forDraggingInfo:](#) (page 307)
Tells the receiver that the sending web view will perform the specified drag-destination action.
- [webView:willPerformDragSourceAction:fromPoint:withPasteboard:](#) (page 307)
Tells the receiver that the sending web view will perform the specified drag-source action.

Controlling Other Behaviors

- [webView:shouldPerformAction:fromSender:](#) (page 306)
Returns a Boolean value that indicates whether the action sent by the specified object should be performed.
- [webView:validateUserInterfaceItem:defaultValidation:](#) (page 306)
Returns a Boolean value that indicates whether the specified user interface item is valid.

Printing

- [webView:printFrameView:](#) (page 297)
Prints the contents of a web frame view.
- [webViewHeaderHeight:](#) (page 311)
Returns the height of the web view's printed page header.
- [webViewFooterHeight:](#) (page 310)
Returns the height of the web view's printed page footer.
- [webView:drawHeaderInRect:](#) (page 296)
Draws the web view's header in the specified rectangle.
- [webView:drawFooterInRect:](#) (page 295)
Draws the web view's footer in the specified rectangle.

Instance Methods

webView:contextMenuItemsForElement:defaultMenuItems:

Returns menu items to display in an element's contextual menu.

- (NSArray *)**webView:(WebView *)sender contextMenuItemsForElement:(NSDictionary *)element defaultMenuItems:(NSArray *)defaultMenuItems**

Parameters*sender*

The web view that sent the message.

element

A dictionary that describes the element that was clicked. See “Constants” in *WebView Class Reference* for information about the key-value pairs in this dictionary.

defaultMenuItems

The menu items included by default in the element’s contextual menu. See “Menu Item Tags” (page 315) for values you can use to differentiate among specific menu items.

Return Value

An array of menu items to display in the element’s contextual menu.

Discussion

This method is invoked every time the user clicks the right mouse button, or control-clicks, on an element to reveal a contextual menu. The receiver typically returns a modified copy of the default menu items dictionary, adding and removing menu items as appropriate for this type of element. You can use this mechanism to remove items that are not appropriate for a particular environment or task, such as saving files to the desktop in a web kiosk. You do not need to set the actions and targets of the default items.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

webView:createWebViewModalDialogWithRequest:

Creates a modal window containing a web view that loads the specified request.

```
- (WebView *)webView:(WebView *)sender
  createWebViewModalDialogWithRequest:(NSURLRequest *)request
```

Parameters*sender*

The web view that sent the message.

request

The request to load.

Return Value

The web view that is loading the specified request.

Discussion

This method is invoked when JavaScript calls `window.showModalDialog`. It should create a new modal window containing the web view and initially hide the window. The `webViewRunModal:` (page 313) message is sent to the delegate to display the web view.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webView:createWebViewWithRequest:

Creates a window containing a web view to load the specified request.

```
- (WebView *)webView:(WebView *)sender createWebViewWithRequest:(NSURLRequest *)request
```

Parameters

sender

The web view that sent the message.

request

The request to load.

Return Value

The web view that is loading the request.

Discussion

This method should begin loading the content for the specified request by sending `loadRequest:` (page 62) to its main frame. The new window should initially be hidden. Later, a `webViewShow:` (page 313) message is sent to the delegate of the new web view. By default, this method returns `nil`.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

webView:dragDestinationActionMaskForDraggingInfo:

Returns a mask indicating which drag operations are allowed by the sender.

```
- (NSUInteger)webView:(WebView *)sender dragDestinationActionMaskForDraggingInfo:(id <NSDraggingInfo>)draggingInfo
```

Parameters

sender

The web view that sent the message.

draggingInfo

The information object for the dragging operation.

Return Value

A mask that indicates which drag operations are allowed when content is dragged over the sending web view. (Note that the return value changed from an `unsigned int` to an `NSUInteger` in Mac OS X v10.5.) See “[Drag-Destination Actions](#)” (page 319) for a list of return values.

Discussion

This method can be invoked multiple times while content is dragged over the sending web view. When the content is dropped, the web view sends a notification (`webView:willPerformDragDestinationAction:forDraggingInfo:` (page 307)) to the receiver.

If you do not implement this method, it returns `WebDragDestinationActionAny` (page 319) by default.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

webView:dragSourceActionMaskForPoint:

Returns a mask indicating which drag-source actions are allowed for a drag that begins at the specified location.

```
- (NSUInteger)webView:(WebView *)sender dragSourceActionMaskForPoint:(NSPoint)point
```

Parameters*sender*

The web view that sent the message.

point

The point at which the drag began, specified in the coordinates of the web view.

Return Value

A mask indicating which drag-source actions are allowed. (Note that the return value changed from an unsigned int to an NSUInteger in Mac OS X v10.5.) See “[Drag-Source Actions](#)” (page 319) for a list of return values.

Discussion

This method is called after the user has begun a drag from a point in a web view. This method can be invoked multiple times while content is dragged from the sending web view. When the content is dropped, the sender sends [webView:willPerformDragSourceAction:fromPoint:withPasteboard:](#) (page 307) to the receiver.

If you do not implement this method, it returns (WebDragSourceActionAny & ~WebDragSourceActionLink) if the cursor is in an editable part of the web view; otherwise, it returns [WebDragDestinationActionAny](#) (page 319).

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

webView:drawFooterInRect:

Draws the web view’s footer in the specified rectangle.

```
- (void)webView:(WebView *)sender drawFooterInRect:(NSRect)rect
```

Parameters*sender*

The web view that sent the message.

rect

The rectangle reserved for drawing the footer.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [webViewFooterHeight:](#) (page 310)

Declared In

WebUIDelegate.h

webView:drawHeaderInRect:

Draws the web view's header in the specified rectangle.

```
- (void)webView:(WebView *)sender drawHeaderInRect:(NSRect)rect
```

Parameters

sender

The web view that sent the message.

rect

The rectangle reserved for drawing the header.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [webViewHeaderHeight:](#) (page 311)

Declared In

WebUIDelegate.h

webView:makeFirstResponder:

Sets the first responder of a web view's window to the specified view.

```
- (void)webView:(WebView *)sender makeFirstResponder:(NSResponder *)responder
```

Parameters

sender

The web view that sent the message.

responder

A view in the web view's hierarchy.

Discussion

You can ignore this message if *sender* is not yet attached to a window.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewFirstResponder:](#) (page 309)

Declared In

WebUIDelegate.h

webView:mouseDidMoveOverElement:modifierFlags:

Updates information about the element the user is mousing over.

```
- (void)webView:(WebView *)sender mouseDidMoveOverElement:(NSDictionary
*)elementInformation modifierFlags:(NSUInteger)modifierFlags
```

Parameters

sender

The web view that sent the message.

elementInformation

A dictionary that describes the element under the mouse, or `nil`. See “Constants” in *WebView Class Reference* for information about the key-value pairs in this dictionary.

modifierFlags

An integer bit field that indicates the modifier keys in effect during the event. See “Modifier Flags” in *NSEvent Class Reference* for information about possible modifiers. Note that this parameter was changed from an unsigned `int` to an `NSUInteger` in Mac OS X v10.5.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

webView:printFrameView:

Prints the contents of a web frame view.

```
- (void)webView:(WebView *)sender printFrameView:(WebFrameView *)frameView
```

Parameters

sender

The web view that sent the message.

frameView

The web frame view whose contents to print.

Discussion

This method is invoked when a script or a user wants to print a webpage. Typically, the delegate implements this method to prepare the web frame view content for printing. The web frame view can handle some content without intervention by the delegate. Send the [documentViewShouldHandlePrint](#) (page 69) message to the web frame view to determine if it can handle printing. If this method returns `YES`, then the delegate can print the content by sending the [printDocumentView](#) (page 69) message to the web frame view. Otherwise, the delegate can use [printOperationWithPrintInfo:](#) (page 70) to get an `NSPrintOperation` object to print the web frame view.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webView:runBeforeUnloadConfirmPanelWithMessage:initiatedByFrame:

Displays a confirmation panel containing the specified message before a window closes.

```
- (BOOL)webView:(WebView *)sender runBeforeUnloadConfirmPanelWithMessage:(NSString *)message initiatedByFrame:(WebFrame *)frame
```

Parameters

sender

The web view that sent the message.

message

The message to display in the panel.

frame

The web frame whose JavaScript initiated this call.

Return Value

YES if the user clicked the OK button; otherwise, NO.

Discussion

Use this method to include a message in the confirmation panel in addition to the message supplied by the webpage. The confirmation panel should contain OK and Cancel buttons.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webView:runJavaScriptAlertPanelWithMessage:

Displays a JavaScript alert panel. (Deprecated in Mac OS X v10.4.11. Use [webView:runJavaScriptAlertPanelWithMessage:initiatedByFrame:](#) (page 299) instead.)

```
- (void)webView:(WebView *)sender runJavaScriptAlertPanelWithMessage:(NSString *)message
```

Parameters

sender

The web view that sent the message.

message

The message to display in the alert panel.

Discussion

This method is used to display a panel when JavaScript code calls `alert`. Delegates should visually indicate that this panel comes from JavaScript. The panel should have, for example, a single OK button. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [webView:runJavaScriptConfirmPanelWithMessage:](#) (page 299)

- [webView:runJavaScriptTextInputPanelWithPrompt:defaultText:](#) (page 300)

Declared In

WebUIDelegate.h

webView:runJavaScriptAlertPanelWithMessage:initiatedByFrame:

Displays a JavaScript alert panel containing the specified message.

```
- (void)webView:(WebView *)sender runJavaScriptAlertPanelWithMessage:(NSString *)message initiatedByFrame:(WebFrame *)frame
```

Parameters

sender

The web view that sent the message.

message

The message to display in the alert panel.

frame

The web frame whose JavaScript initiated this call.

Discussion

This method displays an alert panel when JavaScript code calls `alert`. Delegates should visually indicate that this panel comes from JavaScript. The panel should contain a single OK button. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webView:runJavaScriptConfirmPanelWithMessage:

Displays a JavaScript confirm panel. (Deprecated in Mac OS X v10.4.11. Use [webView:runJavaScriptConfirmPanelWithMessage:initiatedByFrame:](#) (page 300) instead.)

```
- (BOOL)webView:(WebView *)sender runJavaScriptConfirmPanelWithMessage:(NSString *)message
```

Parameters

sender

The web view that sent the message.

message

The message to display in the confirmation panel.

Discussion

This method is used to display a confirmation panel when JavaScript code calls `confirm`. It returns YES if confirmed, NO otherwise. Delegates should visually indicate that this panel comes from JavaScript. The panel should have, for example, an OK and Cancel button. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [webView:runJavaScriptAlertPanelWithMessage:](#) (page 298)
- [webView:runJavaScriptTextInputPanelWithPrompt:defaultText:](#) (page 300)

Declared In

WebUIDelegate.h

webView:runJavaScriptConfirmPanelWithMessage:initiatedByFrame:

Displays a JavaScript confirmation panel with the specified message.

```
- (BOOL)webView:(WebView *)sender runJavaScriptConfirmPanelWithMessage:(NSString *)message initiatedByFrame:(WebFrame *)frame
```

Parameters

sender

The web view that sent the message.

message

The message to display in the confirmation panel.

frame

The web frame whose JavaScript initiated this call.

Return Value

YES if the user clicks OK; otherwise, NO.

Discussion

This method displays a confirmation panel when JavaScript code calls `confirm`. Delegates should visually indicate that this panel comes from JavaScript. The panel should contain an OK and a Cancel button. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webView:runJavaScriptTextInputPanelWithPrompt:defaultText:

Displays a JavaScript text input panel and returns the entered text. (Deprecated in Mac OS X v10.4.11. Use [webView:runJavaScriptTextInputPanelWithPrompt:defaultText:initiatedByFrame:](#) (page 301) instead.)

```
- (NSString *)webView:(WebView *)sender runJavaScriptTextInputPanelWithPrompt:(NSString *)prompt defaultText:(NSString *)defaultText
```

Parameters

sender

The web view that sent the message.

prompt

The message to display in the text input panel.

defaultText

Default placeholder text to display in the text field.

Discussion

This method is used to provide an alternative prompt panel when JavaScript code calls `prompt`. Delegates should visually indicate that this panel comes from JavaScript. The panel should have an OK and Cancel button, and an editable text field. If you do not implement this method, a JavaScript text input panel is displayed.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [webView:runJavaScriptAlertPanelWithMessage:](#) (page 298)

- [webView:runJavaScriptConfirmPanelWithMessage:](#) (page 299)

Declared In

WebUIDelegate.h

webView:runJavaScriptTextInputPanelWithPrompt:defaultText:initiatedByFrame:

Displays a JavaScript text input panel and returns the entered text.

```
- (NSString *)webView:(WebView *)sender
  runJavaScriptTextInputPanelWithPrompt:(NSString *)prompt defaultText:(NSString *)defaultText
  initiatedByFrame:(WebFrame *)frame
```

Parameters

sender

The web view that sent the message.

prompt

The message to display in the text input panel.

defaultText

Default placeholder text to display in the text field.

frame

The web frame whose JavaScript initiated this call.

Return Value

The text entered by the user if the user clicks OK; otherwise, `nil`.

Discussion

This method is used to provide an alternate text input panel when JavaScript code calls `prompt`. Delegates should visually indicate that this panel comes from JavaScript. The panel should contain an OK and a Cancel button, and an editable text field. If you do not implement this method, a JavaScript text input panel is displayed.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webView:runOpenPanelForFileButtonWithResultListener:

Displays an open panel for a file input control.

```
- (void)webView:(WebView *)sender runOpenPanelForFileButtonWithResultListener:(id
    < WebOpenPanelResultListener >)resultListener
```

Parameters*sender*

The web view that sent the message.

resultListener

See the WebOpenPanelResultListener protocol for how to set these values.

Discussion

This method uses a listener object to set the results of the open panel, instead of returning the value directly. This approach allows delegates to implement the open panel as a modal dialog. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

webView:runOpenPanelForFileButtonWithResultListener:allowMultipleFiles:

Displays an open panel for a file input control.

```
- (void)webView:(WebView *)sender runOpenPanelForFileButtonWithResultListener:(id
    < WebOpenPanelResultListener >)resultListener
    allowMultipleFiles:(BOOL)allowMultipleFiles
```

Parameters*sender*

The web view that sent the message.

resultListener

See the WebOpenPanelResultListener protocol for how to set these values.

allowMultipleFiles

If YES, the open panel should allow multiple files to be selected; otherwise, it should not.

Availability

Available in Mac OS X v10.6 and later.

Declared In

WebUIDelegate.h

webView:setContentRect:

Sets the window's content view frame to the specified content rectangle. (Deprecated in Mac OS X v10.4.11. Content rectangle calculations are automatic.)

```
- (void)webView:(WebView *)sender setContentRect:(NSRect)contentRect
```

Parameters

sender

The web view that sent the message.

contentRect

The location and size of the window's content area.

Discussion

The content view is the highest accessible `NSView` object in the view hierarchy displayed in the window. A web view invokes this method instead of setting the content view's frame directly, allowing delegates to augment the behavior by, for example, avoiding auto-saving of the size.

If this method is not implemented by the delegate, then `webView:setFrame:` (page 303) is invoked with the rectangle returned by sending the `NSWindow` method `frameRectForContentRect:styleMask:` to the window.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [webViewContentRect:](#) (page 309)

Declared In

WebUIDelegate.h

webView:setFrame:

Sets the frame rectangle of a web view's window to the specified frame size.

```
- (void)webView:(WebView *)sender setFrame:(NSRect)frame
```

Parameters

sender

The web view that sent the message.

frame

The frame size.

Discussion

The sender invokes this method instead of setting the window's frame directly, allowing delegates to augment the behavior by, for example, saving the original window size before resizing as a result of JavaScript running. If you do not implement this method, the `NSWindow` method `setFrame:display:` is sent to the window that contains *sender*, with `YES` passed as the display argument.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewFrame:](#) (page 311)

Declared In

WebUIDelegate.h

webView:setResizable:

Sets whether a web view's window can be resized.

```
- (void)webView:(WebView *)sender setResizable:(BOOL)resizable
```

Parameters

sender

The web view that sent the message.

resizable

If YES, the web view's window can be resized; if NO, the window is not resizable.

Discussion

By default, this method sets the window containing a web view to be resizable. If you display multiple web views in a window then your user interface delegate should implement this method to handle this special case. If you do not implement this method, the `NSWindow` method `setShowsResizeIndicator:` is sent to the window that contains *sender*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewIsResizable:](#) (page 312)

Declared In

WebUIDelegate.h

webView:setStatusBarVisible:

Sets the visibility of the status bar in a web view's window.

```
- (void)webView:(WebView *)sender setStatusBarVisible:(BOOL)visible
```

Parameters

sender

The web view that sent the message.

visible

If YES, the delegate should display the status bar (if any); if NO, the delegate should hide the status bar.

Discussion

No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewIsStatusBarVisible:](#) (page 312)

Declared In

WebUIDelegate.h

webView:setStatusText:

Sets the status message displayed by a web view's window, if any, to the specified text.

```
- (void)webView:(WebView *)sender setStatusText:(NSString *)text
```

Parameters

sender

The web view that sent the message.

text

The status message to display.

Discussion

The delegate receives this message when a JavaScript function in the web view explicitly sets the status text. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewStatusText:](#) (page 314)

Declared In

WebUIDelegate.h

webView:setToolbarsVisible:

Sets whether a web view's toolbars should be visible.

```
- (void)webView:(WebView *)sender setToolbarsVisible:(BOOL)visible
```

Parameters

sender

The web view that sent the message.

visible

If YES, all toolbars (with the exception of the status bar) are shown; otherwise, all toolbars (with the exception of the status bar) are removed.

Discussion

No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewAreToolbarsVisible:](#) (page 308)

Declared In

WebUIDelegate.h

webView:shouldPerformAction:fromSender:

Returns a Boolean value that indicates whether the action sent by the specified object should be performed.

```
- (BOOL)webView:(WebView *)sender shouldPerformAction:(SEL)action
    fromSender:(id)fromObject
```

Parameters

sender

The web view that sent the message.

action

The action to perform. See *WebView Class Reference* for information on actions a web view can perform.

fromObject

The object that sent the action.

Return Value

YES if the action should be performed; otherwise, NO.

Discussion

This method allows the delegate to control the web view's behavior when action methods are invoked. For example, if the action is `copy:`, the delegate can return `NO` to perform a copy in some other way than the default.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

webView:validateUserInterfaceItem:defaultValidation:

Returns a Boolean value that indicates whether the specified user interface item is valid.

```
- (BOOL)webView:(WebView *)sender validateUserInterfaceItem:(id <
    NSValidatedUserInterfaceItem >)item defaultValidation:(BOOL)defaultValidation
```

Parameters

sender

The web view that sent the message.

item

The user interface item being validated.

defaultValidation

YES if the web view believes the user interface item is valid; otherwise, NO.

Return Value

YES if the specified user interface item is valid; otherwise, NO.

Discussion

See *NSUserInterfaceValidations Protocol Reference* and *NSValidatedUserInterfaceItem Protocol Reference* for more information about user interface validation. If you do not implement this method, the value of *defaultValidation* is used.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

webView:willPerformDragDestinationAction:forDraggingInfo:

Tells the receiver that the sending web view will perform the specified drag-destination action.

```
- (void)webView:(WebView *)sender
  willPerformDragDestinationAction:(WebDragDestinationAction)action
  forDraggingInfo:(id < NSDraggingInfo >)draggingInfo
```

Parameters

sender

The web view that sent the message.

action

The drag-destination action to perform. See “[Drag-Destination Actions](#)” (page 319) for a list of actions.

draggingInfo

The information object for the dragging operation.

Discussion

This method is invoked after the last invocation of the [webView:dragDestinationActionMaskForDraggingInfo:](#) (page 294) method, when the dragged content is dropped and the sender is about to perform the destination action. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

webView:willPerformDragSourceAction:fromPoint:withPasteboard:

Tells the receiver that the sending web view will perform the specified drag-source action.

```
- (void)webView:(WebView *)sender
  willPerformDragSourceAction:(WebDragSourceAction)action fromPoint:(NSPoint)point
  withPasteboard:(NSPasteboard *)pasteboard
```

Parameters*sender*

The web view that sent the message.

*action*The drag-source action to perform. See “[Drag-Source Actions](#)” (page 319) for a list of actions.*point*

The point at which the drag began, specified in the coordinates of the web view.

pasteboard

The drag pasteboard.

Discussion

This method is invoked after the last invocation of the `webView:dragSourceActionMaskForPoint:` (page 295) method, when the dragged content is dropped and the sender is about to perform the drag-source action. The delegate has the opportunity to modify the contents of the object on the pasteboard before completing the drag-source action. No action is taken if you do not implement this method.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

webViewAreToolbarsVisible:

Returns a Boolean value indicating whether any toolbars are visible in a web view's window.

- (BOOL)webViewAreToolbarsVisible:(WebView *)sender

Parameters*sender*

The web view that sent the message.

Return Value

YES if a web view's window has any toolbars that are currently visible (other than the status bar); otherwise, NO.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also- [webView:setToolbarsVisible:](#) (page 305)**Declared In**

WebUIDelegate.h

webViewClose:

Closes a web view in a window.

- (void)webViewClose:(WebView *)sender

Parameters*sender*

The web view that sent the message.

Discussion

If you display multiple web views in a window then you might want to close only *sender* in your implementation. By default, this method sends the `close` method to the `NSWindow` object that contains *sender*.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

webViewContentRect:

Returns a web view window's content rectangle. (Deprecated in Mac OS X v10.4.11. Content rectangle calculations are automatic.)

```
- (NSRect)webViewContentRect:(WebView *)sender
```

Parameters*sender*

The web view that sent the message.

Return Value

The content rectangle of the window that contains the web view.

Discussion

The content view is the highest accessible `NSView` object in the view hierarchy displayed in the window. A web view invokes this method instead of setting the content view's frame directly, allowing delegates to alter the size that is returned.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Deprecated in Mac OS X v10.4.11.

See Also

- [webView:setContentRect:](#) (page 303)

Declared In

WebUIDelegate.h

webViewFirstResponder:

Returns the first responder of the web view's window.

```
- (NSResponder *)webViewFirstResponder:(WebView *)sender
```

Parameters*sender*

The web view that sent the message.

Return Value

The view or subview that currently has the input focus. It can return `nil` or the default first responder if the sender is not attached to a window or if another view (not in the window) has the focus.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:makeFirstResponder:](#) (page 296)

Declared In

WebUIDelegate.h

webViewFocus:

Brings a web view's window to the front and makes it the active window.

```
- (void)webViewFocus:(WebView *)sender
```

Parameters*sender*

The web view that sent the message.

Discussion

By default, this method brings a web view's window into focus. If you display multiple web views in a window then you might also want to focus the input on *sender*, using [webView:makeFirstResponder:](#) (page 296).

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewUnfocus:](#) (page 314)

Declared In

WebUIDelegate.h

webViewFooterHeight:

Returns the height of the web view's printed page footer.

```
- (float)webViewFooterHeight:(WebView *)sender
```

Parameters*sender*

The web view that sent the message.

Return Value

The height of the web view's printed page footer. Returns 0.0 if no space is reserved for the footer.

Discussion

The height returned by this method is used to calculate the rectangle passed to the [webView:drawFooterInRect:](#) (page 295) method.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [webView:drawFooterInRect:](#) (page 295)

Declared In

WebUIDelegate.h

webViewFrame:

Returns the frame rectangle of a web view's window.

```
- (NSRect)webViewFrame:(WebView *)sender
```

Parameters

sender

The web view that sent the message.

Return Value

The frame rectangle of the web view's window.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:setFrame:](#) (page 303)

Declared In

WebUIDelegate.h

webViewHeaderHeight:

Returns the height of the web view's printed page header.

```
- (float)webViewHeaderHeight:(WebView *)sender
```

Parameters

sender

The web view that sent the message.

Return Value

The height of the web view's printed page header. Returns 0.0 if no space is reserved for the header.

Discussion

The height returned by this method is used to calculate the rectangle passed to the [webView:drawHeaderInRect:](#) (page 296) method.

Availability

Available in Mac OS X v10.4.11 and later.

See Also

- [webView:drawHeaderInRect:](#) (page 296)

Declared In

WebUIDelegate.h

webViewIsResizable:

Returns a Boolean value indicating whether a web view's window can be resized.

- (BOOL)webViewIsResizable:(WebView *)sender

Parameters

sender

The web view that sent the message.

Return Value

YES if the web view's window can be resized; otherwise, NO.

Discussion

If you display multiple web views in a window then your user interface delegate should implement this method to handle this special case.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:setResizable:](#) (page 304)

Declared In

WebUIDelegate.h

webViewIsStatusBarVisible:

Returns a Boolean value indicating whether the status bar in a web view's window is visible.

- (BOOL)webViewIsStatusBarVisible:(WebView *)sender

Parameters

sender

The web view that sent the message.

Return Value

YES if a web view's status bar (if any) is visible; otherwise, NO.

Discussion

If you do not implement this method, it returns NO by default.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:setStatusBarVisible:](#) (page 304)

Declared In

WebUIDelegate.h

webViewRunModal:

Displays a web view in a modal window.

```
- (void)webViewRunModal:(WebView *)sender
```

Parameters

sender

The web view that sent the message.

Discussion

This method should display and order front a modal window containing the specified web view. This method is invoked after the [webView:createWebViewModalDialogWithRequest:](#) (page 293) method is used to create a new window.

Availability

Available in Mac OS X v10.4.11 and later.

Declared In

WebUIDelegate.h

webViewShow:

Displays a web view's window and moves it to the front.

```
- (void)webViewShow:(WebView *)sender
```

Parameters

sender

The web view that sent the message.

Discussion

This method is typically used after a call to [webView:createWebViewWithRequest:](#) (page 294), which creates a new window. The new window is not ordered to the front (or even shown) unless you implement this method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

Declared In

WebUIDelegate.h

webViewStatusText:

Returns the current status message from a web view's window.

```
- (NSString *)webViewStatusText:(WebView *)sender
```

Parameters

sender

The web view that sent the message.

Return Value

The status message displayed in the web view's window if one has been set with the [webView:setStatusText:](#) (page 305) method; otherwise, `nil`.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webView:setStatusText:](#) (page 305)

Declared In

WebUIDelegate.h

webViewUnfocus:

Relinquishes focus on a web view's window.

```
- (void)webViewUnfocus:(WebView *)sender
```

Parameters

sender

The web view that sent the message.

Discussion

This method releases focus for the entire window. If you display multiple web views in a window, you might instead want to change the input focus to another view, using the [webView:makeFirstResponder:](#) (page 296) method.

Availability

Available in Mac OS X v10.2 with Safari 1.0 and later.

Available in Mac OS X v10.2.7 and later.

See Also

- [webViewFocus:](#) (page 310)

Declared In

WebUIDelegate.h

Constants

Menu Item Tags

Tags that define the types of default menu items passed to the `webView:contextMenuItemsForElement:defaultMenuItems:` (page 292) method.

```
enum {
    WebMenuItemTagOpenLinkInNewWindow = 1,
    WebMenuItemTagDownloadLinkToDisk,
    WebMenuItemTagCopyLinkToClipboard,
    WebMenuItemTagOpenImageInNewWindow,
    WebMenuItemTagDownloadImageToDisk,
    WebMenuItemTagCopyImageToClipboard,
    WebMenuItemTagOpenFrameInNewWindow,
    WebMenuItemTagCopy,
    WebMenuItemTagGoBack,
    WebMenuItemTagGoForward,
    WebMenuItemTagStop,
    WebMenuItemTagReload,
    WebMenuItemTagCut,
    WebMenuItemTagPaste,
    WebMenuItemTagSpellingGuess,
    WebMenuItemTagNoGuessesFound,
    WebMenuItemTagIgnoreSpelling,
    WebMenuItemTagLearnSpelling,
    WebMenuItemTagOther,
    WebMenuItemTagSearchInSpotlight,
    WebMenuItemTagSearchWeb,
    WebMenuItemTagLookUpInDictionary,
    WebMenuItemTagOpenWithDefaultApplication,
    WebMenuItemPDFActualSize,
    WebMenuItemPDFZoomIn,
    WebMenuItemPDFZoomOut,
    WebMenuItemPDFAutoSize,
    WebMenuItemPDFSinglePage,
    WebMenuItemPDFFacingPages,
    WebMenuItemPDFContinuous,
    WebMenuItemPDFNextPage,
    WebMenuItemPDFPreviousPage,
};
```

Constants

`WebMenuItemTagOpenLinkInNewWindow`
Open the link in a new window.
 Available in Mac OS X v10.2 and later.
 Declared in `WebUIDelegate.h`.

`WebMenuItemTagDownloadLinkToDisk`
Download the link to a disk.
 Available in Mac OS X v10.2 and later.
 Declared in `WebUIDelegate.h`.

WebMenuItemTagCopyLinkToClipboard

Copy the link to the clipboard.

Available in Mac OS X v10.2 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagOpenImageInNewWindow

Open the image in a new window.

Available in Mac OS X v10.2 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagDownloadImageToDisk

Download the image to disk.

Available in Mac OS X v10.2 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagCopyImageToClipboard

Copy the image to the clipboard.

Available in Mac OS X v10.2 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagOpenFrameInNewWindow

Open the frame in a new window.

Available in Mac OS X v10.2 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagCopy

Copy the element to the clipboard.

Available in Mac OS X v10.2 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagGoBack

Load the previous page.

Available in Mac OS X v10.3.9 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagGoForward

Load the next page.

Available in Mac OS X v10.3.9 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagStop

Stop loading the current page.

Available in Mac OS X v10.3.9 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagReload

Reload the current page.

Available in Mac OS X v10.3.9 and later.

Declared in WebUIDelegate.h.

WebMenuItemTagCut

Cut the currently selected content.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagPaste

Paste the content on the clipboard onto the current selection.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagSpellingGuess

Suggest spellings for the misspelled word.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagNoGuessesFound

Indicate whether any suggested spellings for the misspelled word could be found.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagIgnoreSpelling

Ignore the misspelled word.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagLearnSpelling

Add the misspelled word to the user's list of acceptable words.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagOther

Used when a tag for an item in the context menu can't be determined.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagSearchInSpotlight

Search Spotlight for the current selection.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagSearchWeb

Search the web for the current selection.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

WebMenuItemTagLookUpInDictionary

Look up the current selection in the Dictionary.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemTagOpenWithDefaultApplication`

Open the current selection using the default application.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFActualSize`

Display a PDF document at its original size.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFZoomIn`

Scale up a PDF document.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFZoomOut`

Scale down a PDF document.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFAutoSize`

Display a PDF document at a user-specified size.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFSinglePage`

Display a PDF document one page at a time.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFFacingPages`

Display a PDF document two pages at a time.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFContinuous`

Display all pages in a PDF document continuously, using a vertical scroll bar, if necessary.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFNextPage`

Display the next page of a PDF document.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

`WebMenuItemPDFPreviousPage`

Display the previous page of a PDF document.

Available in Mac OS X v10.4.11 and later.

Declared in `WebUIDelegate.h`.

Discussion

These tags define common actions a user might want to take with elements in a page. You can use the tags to differentiate between the different types of menu items.

Declared In

WebUIDelegate.h

Drag-Destination Actions

Actions that the destination object of a drag operation can perform.

```
typedef enum {
    WebDragDestinationActionNone      = 0,
    WebDragDestinationActionDHTML    = 1,
    WebDragDestinationActionEdit     = 2,
    WebDragDestinationActionLoad     = 4,
    WebDragDestinationActionAny      = UINT_MAX
} WebDragDestinationAction;
```

Constants

WebDragDestinationActionNone

No action.

Available in Mac OS X v10.3 and later.

Declared in WebUIDelegate.h.

WebDragDestinationActionDHTML

Allows DHTML (such as JavaScript) to handle the drag.

Available in Mac OS X v10.3 and later.

Declared in WebUIDelegate.h.

WebDragDestinationActionEdit

Allows editable documents to be changed by the drag operation.

Available in Mac OS X v10.3 and later.

Declared in WebUIDelegate.h.

WebDragDestinationActionLoad

Allows the drag operation to change the location.

Available in Mac OS X v10.3 and later.

Declared in WebUIDelegate.h.

WebDragDestinationActionAny

Allows any defined action to occur.

Available in Mac OS X v10.3 and later.

Declared in WebUIDelegate.h.

Availability

Available in Mac OS X v10.3.9 and later.

Declared In

WebUIDelegate.h

Drag-Source Actions

Actions that the source object of a drag operation can perform.

```
typedef enum {
    WebDragSourceActionNone           = 0,
    WebDragSourceActionDHTML         = 1,
    WebDragSourceActionImage          = 2,
    WebDragSourceActionLink           = 4,
    WebDragSourceActionSelection      = 8,
    WebDragSourceActionAny            = UINT_MAX
} WebDragSourceAction;
```

Constants

`WebDragSourceActionNone`

No action.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

`WebDragSourceActionDHTML`

Allows DHTML (such as JavaScript) in the source object to initiate a drag operation.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

`WebDragSourceActionImage`

Allows the user to drag an image in the source object.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

`WebDragSourceActionLink`

Allows the user to drag a link in the source object.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

`WebDragSourceActionSelection`

Allows the user to drag a selection in the source object.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

`WebDragSourceActionAny`

Allows any defined action to occur.

Available in Mac OS X v10.3 and later.

Declared in `WebUIDelegate.h`.

Availability

Available in Mac OS X v10.3.9 and later.

Constants

WebKit Constants Reference

Framework: WebKit/WebKit.h

Overview

This document describes the types and constants found in the WebKit.

Constants

WebKit Policy Errors

Policy errors that can occur.

```
typedef enum {
    WebKitErrorCannotShowMIMETYPE = 100,
    WebKitErrorCannotShowURL = 101,
    WebKitErrorFrameLoadInterruptedByPolicyChange = 102,
};
```

Constants

`WebKitErrorCannotShowMIMETYPE`
Indicates that a MIME type is not supported.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorCannotShowURL`
Indicates a failure in changing a location.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorFrameLoadInterruptedByPolicyChange`
Indicates that a frame load was interrupted by a policy change.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

Discussion

These errors occur when applying policy decisions.

WebKit Plug-in and Java Errors

Plug-in and Java errors that can occur.

```
typedef enum {
    WebKitErrorCannotFindPlugIn = 200,
    WebKitErrorCannotLoadPlugIn = 201,
    WebKitErrorJavaUnavailable = 202,
};
```

Constants

`WebKitErrorCannotFindPlugIn`
 Indicates a plug-in could not be found.
 Available in Mac OS X v10.3 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorCannotLoadPlugIn`
 Indicates a plug-in could not be loaded.
 Available in Mac OS X v10.3 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorJavaUnavailable`
 Indicates that Java is unavailable.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

Discussion

These errors occur when loading pages containing plug-in or Java content.

Other WebKit Errors

Other WebKit errors that can occur.

```
extern NSString *WebKitErrorDomain;
extern NSString * const WebKitErrorMIMETYPEKey;
extern NSString * const WebKitErrorPlugInNameKey;
extern NSString * const WebKitErrorPlugInPageURLStringKey;
```

Constants

`WebKitErrorDomain`
 A string used by `NSError` to indicate that the error was originated by a WebKit class.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorMIMETYPEKey`
 A dictionary key whose value is a string of the `TYPE` attribute.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorPlugInNameKey`
 A dictionary key whose value is a string containing the plug-in's name.
 Available in Mac OS X v10.2 and later.
 Declared in `WebKitErrors.h`.

`WebKitErrorPlugInPageURLStringKey`

A dictionary key whose value is a URL string of the `PLUGINSPAGE` attribute.

Available in Mac OS X v10.2 and later.

Declared in `WebKitErrors.h`.

Discussion

These errors occur while loading content.

Document Revision History

This table describes the changes to *WebKit Objective-C Framework Reference*.

Date	Notes
2009-07-28	Added concurrency information.
2009-06-17	Minor edits throughout.
2008-10-15	Minor edits throughout.
2006-05-23	First publication of this content as a collection of separate documents.

REVISION HISTORY

Document Revision History