

---

# NSURLProtocol Class Reference

[Cocoa > Networking](#)



2007-04-01



Apple Inc.  
© 2007 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, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

iPhone is a trademark of Apple Inc.

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

---

## **NSURLProtocol Class Reference 5**

---

Overview 5

Tasks 6

    Creating Protocol Objects 6

    Registering and Unregistering Protocol Classes 6

    Getting and Setting Request Properties 6

    Determining If a Subclass Can Handle a Request 6

    Providing a Canonical Version of a Request 6

    Determining If Requests Are Cache Equivalent 6

    Starting and Stopping Downloads 7

    Getting Protocol Attributes 7

Class Methods 7

    canInitWithRequest: 7

    canonicalRequestForRequest: 7

    propertyForKey:inRequest: 8

    registerClass: 9

    removePropertyForKey:inRequest: 9

    requestIsCacheEquivalent:toRequest: 10

    setProperty:forKey:inRequest: 10

    unregisterClass: 11

Instance Methods 11

    cachedResponse 11

    client 12

    initWithRequest:cachedResponse:client: 12

    request 12

    startLoading 13

    stopLoading 13

---

## **Document Revision History 15**

---

## **Index 17**

---



# NSURLProtocol Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Foundation.framework
<b>Availability</b>	Available in Mac OS X v10.2 with Safari 1.0 installed. Available in Mac OS X v10.2.7 and later.
<b>Companion guide</b>	URL Loading System
<b>Declared in</b>	NSURLProtocol.h

## Overview

`NSURLProtocol` is an abstract class that provides the basic structure for performing protocol-specific loading of URL data. Concrete subclasses handle the specifics associated with one or more protocols or URL schemes.

An application should never need to directly instantiate an `NSURLProtocol` subclass. The instance of the appropriate `NSURLProtocol` subclass for an `NSURLRequest` is created by `NSURLConnection` when a download is started.

The `NSURLProtocolClient` protocol describes the methods an implementation uses to drive the URL loading system from a `NSURLProtocol` subclass.

To support customization of protocol-specific requests, protocol implementors are encouraged to provide categories on `NSURLRequest` and `NSMutableURLRequest`. Protocol implementors who need to extend the capabilities of `NSURLRequest` and `NSMutableURLRequest` in this way can store and retrieve protocol-specific request data by using `NSURLProtocol`'s class methods `propertyForKey:inRequest:` (page 8) and `setProperty:forKey:inRequest:` (page 10).

An essential responsibility for a protocol implementor is creating a `NSURLResponse` for each request it processes successfully. A protocol implementor may wish to create a custom, mutable `NSURLResponse` class to provide protocol specific information.

## Tasks

### Creating Protocol Objects

- [initWithRequest:cachedResponse:client:](#) (page 12)  
Initializes an `NSURLProtocol` object.

### Registering and Unregistering Protocol Classes

- + [registerClass:](#) (page 9)  
Attempts to register a subclass of `NSURLProtocol`, making it visible to the URL loading system.
- + [unregisterClass:](#) (page 11)  
Unregisters the specified subclass of `NSURLProtocol`.

### Getting and Setting Request Properties

- + [propertyForKey:inRequest:](#) (page 8)  
Returns the property associated with the specified key in the specified request.
- + [setProperty:forKey:inRequest:](#) (page 10)  
Sets the property associated with the specified key in the specified request.
- + [removePropertyForKey:inRequest:](#) (page 9)  
Removes the property associated with the specified key in the specified request.

### Determining If a Subclass Can Handle a Request

- + [canInitWithRequest:](#) (page 7)  
Returns whether the protocol subclass can handle the specified request.

### Providing a Canonical Version of a Request

- + [canonicalRequestForRequest:](#) (page 7)  
Returns a canonical version of the specified request.

### Determining If Requests Are Cache Equivalent

- + [requestIsCacheEquivalent:toRequest:](#) (page 10)  
Returns whether two requests are equivalent for cache purposes.

## Starting and Stopping Downloads

- [startLoading](#) (page 13)  
Starts protocol-specific loading of the request.
- [stopLoading](#) (page 13)  
Stops protocol-specific loading of the request.

## Getting Protocol Attributes

- [cachedResponse](#) (page 11)  
Returns the receiver's cached response.
- [client](#) (page 12)  
Returns the object the receiver uses to communicate with the URL loading system.
- [request](#) (page 12)  
Returns the receiver's request.

## Class Methods

### **canInitWithRequest:**

Returns whether the protocol subclass can handle the specified request.

```
+ (BOOL)canInitWithRequest:(NSURLRequest *)request
```

#### **Parameters**

*request*

The request to be handled.

#### **Return Value**

YES if the protocol subclass can handle *request*, otherwise NO.

#### **Discussion**

A subclass should inspect *request* and determine whether or not the implementation can perform a load with that request.

This is an abstract method and subclasses must provide an implementation.

#### **Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

#### **Declared In**

NSURLProtocol.h

### **canonicalRequestForRequest:**

Returns a canonical version of the specified request.

```
+ (NSURLRequest *)canonicalRequestForRequest:(NSURLRequest *)request
```

**Parameters**

*request*

The request whose canonical version is desired.

**Return Value**

The canonical form of *request*.

**Discussion**

It is up to each concrete protocol implementation to define what “canonical” means. A protocol should guarantee that the same input request always yields the same canonical form.

Special consideration should be given when implementing this method, because the canonical form of a request is used to lookup objects in the URL cache, a process which performs equality checks between `NSURLRequest` objects.

This is an abstract method and subclasses must provide an implementation.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**Declared In**

`NSURLProtocol.h`

**propertyForKey:inRequest:**

Returns the property associated with the specified key in the specified request.

```
+ (id)propertyForKey:(NSString *)key inRequest:(NSURLRequest *)request
```

**Parameters**

*key*

The key of the desired property.

*request*

The request whose properties are to be queried.

**Return Value**

The property associated with *key*, or `nil` if no property has been stored for *key*.

**Discussion**

This method provides an interface for protocol implementors to access protocol-specific information associated with `NSURLRequest` objects.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**See Also**

[+ setProperty:forKey:inRequest:](#) (page 10)

**Declared In**

`NSURLProtocol.h`



## registerClass:

Attempts to register a subclass of `NSURLProtocol`, making it visible to the URL loading system.

```
+ (BOOL)registerClass:(Class)protocolClass
```

### Parameters

*protocolClass*

The subclass of `NSURLProtocol` to register.

### Return Value

YES if the registration is successful, NO otherwise. The only failure condition is if *protocolClass* is not a subclass of `NSURLProtocol`.

### Discussion

When the URL loading system begins to load a request, each registered protocol class is consulted in turn to see if it can be initialized with the specified request. The first `NSURLProtocol` subclass to return YES when sent a `canInitWithRequest:` (page 7) message is used to perform the URL load. There is no guarantee that all registered protocol classes will be consulted.

Classes are consulted in the reverse order of their registration. A similar design governs the process to create the canonical form of a request with `canonicalRequestForRequest:` (page 7).

### Availability

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

### See Also

+ [unregisterClass:](#) (page 11)

### Declared In

`NSURLProtocol.h`

## removePropertyForKey:inRequest:

Removes the property associated with the specified key in the specified request.

```
+ (void)removePropertyForKey:(NSString *)key inRequest:(NSMutableURLRequest *)request
```

### Parameters

*key*

The key whose value should be removed.

*request*

The request from which to remove the property value.

### Discussion

This method is used to provide an interface for protocol implementors to customize protocol-specific information associated with `NSMutableURLRequest` objects.

### Availability

Available in Mac OS X v10.5 and later.

### See Also

+ [propertyForKey:inRequest:](#) (page 8)

**Declared In**

NSURLProtocol.h

**requestIsCacheEquivalent:toRequest:**

Returns whether two requests are equivalent for cache purposes.

```
+ (BOOL)requestIsCacheEquivalent:(NSURLRequest *)aRequest toRequest:(NSURLRequest *)bRequest
```

**Parameters***aRequest*The request to compare with *bRequest*.*bRequest*The request to compare with *aRequest*.**Return Value**

YES if *aRequest* and *bRequest* are equivalent for cache purposes, NO otherwise. Requests are considered equivalent for cache purposes if and only if they would be handled by the same protocol and that protocol declares them equivalent after performing implementation-specific checks.

**Discussion**

The `NSURLProtocol` implementation of this method compares the URLs of the requests to determine if the requests should be considered equivalent. Subclasses can override this method to provide protocol-specific comparisons.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**Declared In**

NSURLProtocol.h

**setProperty:forKey:inRequest:**

Sets the property associated with the specified key in the specified request.

```
+ (void)setProperty:(id)value forKey:(NSString *)key inRequest:(NSMutableURLRequest *)request
```

**Parameters***value*

The value to set for the specified property.

*key*

The key for the specified property.

*request*

The request for which to create the property.

**Discussion**

This method is used to provide an interface for protocol implementors to customize protocol-specific information associated with `NSMutableURLRequest` objects.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**See Also**

+ [propertyForKey:inRequest:](#) (page 8)

**Declared In**

NSURLProtocol.h

**unregisterClass:**

Unregisters the specified subclass of NSURLProtocol.

```
+ (void)unregisterClass:(Class)protocolClass
```

**Parameters**

*protocolClass*

The subclass of NSURLProtocol to unregister.

**Discussion**

After this method is invoked, *protocolClass* is no longer consulted by the URL loading system.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**See Also**

+ [registerClass:](#) (page 9)

**Declared In**

NSURLProtocol.h

## Instance Methods

**cachedResponse**

Returns the receiver's cached response.

```
- (NSCachedURLResponse *)cachedResponse
```

**Return Value**

The receiver's cached response.

**Discussion**

Subclasses must implement this method.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**Declared In**

NSURLProtocol.h

**client**

Returns the object the receiver uses to communicate with the URL loading system.

```
- (id < NSURLConnectionClient >)client
```

**Return Value**

The object the receiver uses to communicate with the URL loading system.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**Declared In**

NSURLProtocol.h

**initWithRequest:cachedResponse:client:**

Initializes an NSURLConnection object.

```
- (id)initWithRequest:(NSURLRequest *)request cachedResponse:(NSCachedURLResponse *)cachedResponse client:(id < NSURLConnectionClient >)client
```

**Parameters**

*request*

The URL request for the URL protocol object.

*cachedResponse*

A cached response for the request; may be `nil` if there is no existing cached response for the request.

*client*

An object that provides an implementation of the NSURLConnectionClient protocol that the receiver uses to communicate with the URL loading system.

**Discussion**

Subclasses should override this method to do any custom initialization. An application should never explicitly call this method.

This is the designated initializer for NSURLConnection.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**Declared In**

NSURLConnection.h

**request**

Returns the receiver's request.

- (NSURLRequest \*)request

**Return Value**

The receiver's request.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**Declared In**

NSURLProtocol.h

## startLoading

Starts protocol-specific loading of the request.

- (void)startLoading

**Discussion**

When this method is called, the subclass implementation should start loading the request, providing feedback to the URL loading system via the `NSURLProtocolClient` protocol.

Subclasses must implement this method.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**See Also**

- [stopLoading](#) (page 13)

**Declared In**

NSURLProtocol.h

## stopLoading

Stops protocol-specific loading of the request.

- (void)stopLoading

**Discussion**

When this method is called, the subclass implementation should stop loading a request. This could be in response to a cancel operation, so protocol implementations must be able to handle this call while a load is in progress.

Subclasses must implement this method.

**Availability**

Available in Mac OS X v10.2 with Safari 1.0 installed.

Available in Mac OS X v10.2.7 and later.

**See Also**

- [startLoading](#) (page 13)

**Declared In**

NSURLProtocol.h

# Document Revision History

---

This table describes the changes to *NSURLProtocol Class Reference*.

Date	Notes
2007-04-01	Updated for Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History



# Index

---

## C

---

cachedResponse **instance method** [11](#)  
canInitWithRequest: **class method** [7](#)  
canonicalRequestForRequest: **class method** [7](#)  
client **instance method** [12](#)

## I

---

initWithRequest:cachedResponse:client: **instance method** [12](#)

## P

---

propertyForKey:inRequest: **class method** [8](#)

## R

---

registerClass: **class method** [9](#)  
removePropertyForKey:inRequest: **class method** [9](#)  
request **instance method** [12](#)  
requestIsCacheEquivalent:toRequest: **class method** [10](#)

## S

---

setProperty:forKey:inRequest: **class method** [10](#)  
startLoading **instance method** [13](#)  
stopLoading **instance method** [13](#)

## U

---

unregisterClass: **class method** [11](#)