

# 1 System.Net.WebRequest Class

```
2 [ILAsm]  
3 .class public abstract serializable WebRequest extends  
4 System.MarshalByRefObject  
  
5 [C#]  
6 public abstract class WebRequest: MarshalByRefObject
```

## 7 Assembly Info:

- 8 • *Name:* System
- 9 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00]
- 10 • *Version:* 2.0.x.x
- 11 • *Attributes:*
  - 12 ○ CLSCompliantAttribute(true)

## 13 Summary

14 Makes a request to a Uniform Resource Identifier (URI).

## 15 Inherits From: System.MarshalByRefObject

16

17 **Library:** Networking

18

19 **Thread Safety:** All public static members of this type are safe for multithreaded operations.  
20 No instance members are guaranteed to be thread safe.

21

## 22 Description

23 `System.Net.WebRequest` is an abstract class that models the request side of  
24 transactions used for accessing data from the Internet.

25

26 Classes that derive from `System.Net.WebRequest` are required to override the following  
27 members of the `System.Net.WebRequest` class in a protocol-specific manner:

- 28 • `System.Net.WebRequest.Method` -- Gets or sets the protocol method to use in the  
29 current instance.
  
- 30 • `System.Net.WebRequest.RequestUri` -- Gets the `System.Uri` of the resource  
31 associated with the current instance.
  
- 32 • `System.Net.WebRequest.Headers` -- Gets or sets the collection of header  
33 name/value pairs associated with the request.
  
- 34 • `System.Net.WebRequest.ContentLength` -- Gets or sets the content length of the  
35 request data being sent.

- 1     • `System.Net.WebRequest.ContentType` -- Gets or sets the content type of the  
2       request data being sent.
- 3     • `System.Net.WebRequest.Credentials` -- Gets or sets the credentials used for  
4       authenticating the client using the current instance.
- 5     • `System.Net.WebRequest.PreAuthenticate` -- Gets or sets a value that indicates  
6       whether to send authentication information with a request for resources.
- 7     • `System.Net.WebRequest.GetRequestStream` -- Returns a `System.IO.Stream` for  
8       writing data to a resource.
- 9     • `System.Net.WebRequest.BeginGetRequestStream` -- Begins an asynchronous  
10      request for a stream in which to write data to be sent in the current request.
- 11    • `System.Net.WebRequest.EndGetRequestStream` -- Returns a `System.IO.Stream` for  
12      writing data to the resource accessed by the current instance.
- 13    • `System.Net.WebRequest.GetResponse` -- Returns a response to a request.
- 14    • `System.Net.WebRequest.BeginGetResponse` -- Begins an asynchronous request for  
15      a resource.
- 16    • `System.Net.WebRequest.EndGetResponse` -- Returns a `System.Net.WebResponse`  
17      that contains a response to a specified pending request.

18    In addition, derived classes are required to support the `System.Net.IWebRequestCreate`  
19    interface.

20

21    [*Note:* An application that uses the request/response model can request data be sent from  
22    the Internet in a protocol-agnostic manner, in which the application works with instances of  
23    the `System.Net.WebRequest` class while classes that derive from `System.Net.WebRequest`  
24    and implement specific protocols perform the details of the request.

25

26    Requests are sent from an application to a particular Uniform Resource Identifier (URI),  
27    such as a Web page on a server. Using the URI, the `System.Net.WebRequest.Create`  
28    method creates an instance of a type derived from `System.Net.WebRequest` to handle the  
29    request. The type is selected from the set of registered types. Types can be registered to  
30    handle a specific protocol, such as HTTP or FTP, or to handle a request to a specific server  
31    or path on a server. [*Note:* For information on registering types, see  
32    `System.Net.WebRequest.RegisterPrefix`.]

33

34

35

36    The `System.Net.WebRequest` class throws a `System.Net.WebException` exception when an  
37    error occurs while accessing a resource.

38

39    Use the `System.Net.WebRequest.Create` method to initialize a new instance of a class that

1 derives from `System.Net.WebRequest`. Do not use the `System.Net.WebRequest` constructor.  
2  
3 ]

#### 4 **Example**

5 The following example demonstrates using `System.Net.WebRequest.Create` to create  
6 an instance of `System.Net.HttpWebRequest`.

```
7  
8 [C#]  
  
9 using System;  
10 using System.Net;  
11  
12 public class WebRequestExample {  
13     public static void Main() {  
14         // Initialize the WebRequest.  
15         WebRequest myRequest =  
16             WebRequest.Create("http://www.contoso.com");  
17  
18         // Print the type of the request.  
19         Console.WriteLine(myRequest);  
20     }  
21 }  
22 }  
23 }
```

24 The output is

25  
26 `System.Net.HttpWebRequest`

27

# 1 WebRequest() Constructor

```
2 [ILAsm]  
3 family rtspecialname specialname instance void .ctor()  
4 [C#]  
5 protected WebRequest()
```

## 6 Summary

7 Constructs a new instance of the `System.Net.WebRequest` class.

## 8 Description

9 This constructor is called only by classes that derive from `System.Net.WebRequest`.

10  
11 [*Note:* Use the `System.Net.WebRequest.Create` method to initialize a new instance of a  
12 class that derives from `System.Net.WebRequest`. Do not use this constructor.]

13  
14

15

# 1 WebRequest.Abort() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual void Abort()  
4 [C#]  
5 public virtual void Abort()
```

## 6 Summary

7 Attempts to cancel an asynchronous request made by the current instance to access a  
8 resource.

## 9 Behaviors

10 As described above.

11

## 12 Default

13 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
14 for this method. This method throws `System.NotSupportedException`.

15

## 16 How and When to Override

17 This method must be overridden by classes that inherit from `System.Net.WebRequest` to  
18 provide this functionality.

19

## 20 Usage

21 Use this method to cancel an asynchronous operation started with the  
22 `System.Net.WebRequest.BeginGetResponse` method.

23

## 24 Exceptions

Exception	Condition
<code>System.NotSupportedException</code>	This method is not overridden in the derived class.

25

26

# WebRequest.BeginGetRequestStream(System.AsyncCallback, System.Object) Method

```
[ILAsm]
.method public hidebysig virtual class System.IAsyncResult
BeginGetRequestStream(class System.AsyncCallback callback, object state)

[C#]
public virtual IAsyncResult BeginGetRequestStream(AsyncCallback callback,
object state)
```

## Summary

Begins an asynchronous request for a stream in which to write data to be sent in the current request.

## Parameters

Parameter	Description
<i>callback</i>	A <code>System.AsyncCallback</code> delegate to be called when the stream is available. Can be null.
<i>state</i>	A <code>System.Object</code> containing state information for the asynchronous request.

## Return Value

A `System.IAsyncResult` object that contains information about the asynchronous operation.

## Description

The *state* parameter can be any object that the caller wishes to have available for the duration of the asynchronous operation. This object is available via the `System.IAsyncResult.AsyncState` property of the object returned by this method.

## Behaviors

This method starts an asynchronous operation to obtain a stream used to write data to be sent in the current request. To get the request stream, call the `System.Net.WebRequest.EndGetRequestStream` method and specify the `System.IAsyncResult` object returned by this method.

If the *callback* parameter is not null, the method referenced by *callback* is invoked when the asynchronous operation completes. The `System.IAsyncResult` object

1 returned by this method is passed as the argument to the method referenced by  
2 *callback*.

### 3 **Default**

4 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
5 for this method. This method throws `System.NotSupportedException`.

6

### 7 **How and When to Override**

8 This method must be overridden by classes that inherit from `System.Net.WebRequest` to  
9 provide this functionality.

10

### 11 **Usage**

12 Use this method to start an asynchronous request for a stream used to send data to a  
13 resource. The *callback* delegate can call the  
14 `System.Net.WebRequest.EndGetRequestStream` method to obtain the request stream.

15

### 16 **Exceptions**

Exception	Condition
<b>System.NotSupportedException</b>	This method is not overridden in the derived class.

17

18

# WebRequest.BeginGetResponse(System.AsyncCallback, System.Object) Method

```
[IAsm]
.method public hidebysig virtual class System.IAsyncResult
BeginGetResponse(class System.AsyncCallback callback, object state)

[C#]
public virtual IAsyncResult BeginGetResponse(AsyncCallback callback,
object state)
```

## Summary

Begins sending the current request asynchronously.

## Parameters

Parameter	Description
<i>callback</i>	A <code>System.AsyncCallback</code> delegate to be called when the response from the server is available.
<i>state</i>	A <code>System.Object</code> containing state information for the asynchronous request.

## Return Value

A `System.IAsyncResult` object that contains information about the asynchronous operation.

## Description

The *state* parameter can be any object that the caller wishes to have available for the duration of the asynchronous operation. This object is available via the `System.IAsyncResult.AsyncState` property of the object returned by this method.

## Behaviors

This method starts an asynchronous operation to send the current request and receive the response from the server that processed the request. To get the response, call the `System.Net.WebRequest.EndGetResponse` method and specify the `System.IAsyncResult` object returned by this method.

If the *callback* parameter is not null, the method referenced by *callback* is invoked when the asynchronous operation completes. The `System.IAsyncResult` object returned by this method is passed as the argument to the method referenced by *callback*.

1 **Default**

2 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
3 for this method. This method throws `System.NotSupportedException`.

4

5 **How and When to Override**

6 This method must be overridden by classes that inherit from `System.Net.WebRequest` to  
7 provide this functionality.

8

9 **Usage**

10 The `System.Net.WebRequest.BeginGetResponse` method starts an asynchronous  
11 request for a response. The callback delegate can call the  
12 `System.Net.WebRequest.EndGetResponse` method to return the  
13 `System.Net.WebResponse` received from the resource.

14 **Exceptions**

Exception	Condition
<b>System.NotSupportedException</b>	This method is not overridden in the derived class.

15

16

# 1 WebRequest.Create(System.String) Method

```
2 [ILAsm]  
3 .method public hidebysig static class System.Net.WebRequest Create(string  
4 requestUriString)  
5 [C#]  
6 public static WebRequest Create(string requestUriString)
```

## 7 Summary

8 Constructs a new instance of a class derived from `System.Net.WebRequest`. The new  
9 instance is of the type registered for the scheme of the specified URI.

## 10 Parameters

Parameter	Description
<i>requestUriString</i>	A <code>System.String</code> that contains a URI.

## 12 Return Value

13 A new instance of a class that derived from `System.Net.WebRequest` and is registered to  
14 handle the scheme of *requestUriString*.

## 15 Description

16 [Note: This method returns a new instance of a class that derived from  
17 `System.Net.WebRequest`. The `System.Type` of this new instance is determined at run  
18 time by the scheme of the URI in *requestUriString*. For example, when a URI beginning  
19 with `http://` is passed in *requestUriString*, a `System.Net.HttpWebRequest` instance is  
20 returned.

21 Classes that derive from `System.Net.WebRequest` that are created to handle other  
22 requests are registered with the `System.Net.WebRequest.RegisterPrefix` method.

24 ]  
25

## 26 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>requestUriString</i> is null.
<b>System.NotSupportedException</b>	The request scheme specified in <i>requestUri</i> is not registered.

<b>System.UriFormatException</b>	The URI specified in <i>requestUriString</i> is not a valid URI.
<b>System.Security.SecurityException</b>	The caller does not have permission to connect to the requested URI or a URI that the request is redirected to.

1

2 **Permissions**

Permission	Description
<b>System.Security.Permissions.WebPermission</b>	Requires permission to connect to the requested URI. See <code>System.Net.NetworkAccess.Connect</code> .

3

4

# WebRequest.Create(System.Uri) Method

```
[ILAsm]  
.method public hidebysig static class System.Net.WebRequest Create(class  
System.Uri requestUri)  
  
[C#]  
public static WebRequest Create(Uri requestUri)
```

## Summary

Constructs a new instance of a class derived from `System.Net.WebRequest`.

## Parameters

Parameter	Description
<i>requestUri</i>	A <code>System.Uri</code> containing the URI of the requested resource.

## Return Value

A new instance of a class derived from `System.Net.WebRequest` that is registered to handle the closest registered match for *requestUri*.

## Description

To determine the closest match, this method checks the registered URIs for the longest URI prefix that matches *requestUri*.

[*Note:* For an example that demonstrates this method, see `System.Net.WebRequest.CreateDefault`.

]

## Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<i>requestUri</i> is null.
<code>System.NotSupportedException</code>	The request scheme specified in <i>requestUri</i> is not registered.
<code>System.Security.SecurityException</code>	The caller does not have permission to connect to the requested URI or a URI that the request is

	redirected to.
--	----------------

1

## 2 Permissions

Permission	Description
<b>System.Security.Permissions. WebPermission</b>	Requires permission to connect to the requested URI. See <code>System.Net.NetworkAccess.Connect</code> .

3

4

# 1 WebRequest.CreateDefault(System.Uri)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static class System.Net.WebRequest  
5 CreateDefault(class System.Uri requestUri)  
  
6 [C#]  
7 public static WebRequest CreateDefault(Uri requestUri)
```

### 8 Summary

9 Constructs a new instance of a class derived from `System.Net.WebRequest`. The new  
10 instance is of the type registered for the scheme of the specified URI.

### 11 Parameters

Parameter	Description
<i>requestUri</i>	A <code>System.Uri</code> containing the URI of the requested resource.

### 12 Return Value

14 A new instance of the type derived from `System.Net.WebRequest` that is registered for  
15 the scheme of the specified `System.Uri`.

### 16 Description

17 [Note: When this method is invoked, only the scheme portion of *requestUri* is checked  
18 against the list of URIs registered for the current instance. Conversely, when  
19 `System.Net.WebRequest.Create` is invoked, the entire URI is checked against the list of  
20 registered URIs.]  
21  
22

### 23 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<i>requestUri</i> is null.
<code>System.NotSupportedException</code>	The request scheme specified in <i>requestUri</i> is not registered.
<code>System.Security.SecurityException</code>	The caller does not have permission to connect to the requested URI or a URI that the request is

redirected to.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40

## Example

This example demonstrates the use of the `System.Net.WebRequest.Create` and `System.Net.WebRequest.CreateDefault` methods.

[C#]

```
using System;
using System.Net;

public class ContosoTextRequest: WebRequest, IWebRequestCreate
{
    public new WebRequest Create(Uri uri)
    {
        return new ContosoTextRequest();
    }
}

public class CreateDefaultExample
{
    public static void Main()
    {
        ContosoTextRequest contoso = new ContosoTextRequest();
        Uri contosoUri = new Uri("http://www.contoso.com/text");
        WebRequest.RegisterPrefix("http://www.contoso.com/text", contoso);

        WebRequest httpContoso = WebRequest.CreateDefault(contosoUri);
        Console.WriteLine("CreateDefault --> {0}", httpContoso);

        WebRequest textContoso = WebRequest.Create(contosoUri);
        Console.WriteLine("Create --> {0}", textContoso);
    }
}
```

The output is

CreateDefault --> System.Net.HttpWebRequest

Create --> ContosoTextRequest

## Permissions

Permission	Description
<b>System.Security.Permissions.WebPermission</b>	Requires permission to connect to the requested URI. See <code>System.Net.NetworkAccess.Connect</code> .

1

2

1  
2 **WebRequest.EndGetRequestStream(System.I**  
3 **AsyncResult) Method**

```
4 [ILAsm]  
5 .method public hidebysig virtual class System.IO.Stream  
6 EndGetRequestStream(class System.IAsyncResult asyncResult)  
7 [C#]  
8 public virtual Stream EndGetRequestStream(IAsyncResult asyncResult)
```

9 **Summary**

10 Returns a `System.IO.Stream` for writing data to the resource identified by the  
11 `System.Net.WebRequest.RequestUri` property of the current instance.

12 **Parameters**

Parameter	Description
<i>asyncResult</i>	A <code>System.IAsyncResult</code> object that references a request for a <code>System.IO.Stream</code> started with <code>System.Net.WebRequest.BeginGetRequestStream</code> .

13  
14 **Return Value**

15 A `System.IO.Stream` to write data to.

16 **Description**

17 This method completes an asynchronous request for a stream that was started by the  
18 `System.Net.WebRequest.BeginGetRequestStream` method.

19 **Behaviors**

20 As described above.

21  
22 **Default**

23 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
24 for this method. This method throws `System.NotSupportedException`.

25

1 **How and When to Override**

2 This method must be overridden by classes that inherit from `System.Net.WebRequest` to  
3 provide this functionality.

4

5 **Usage**

6 Use this method to complete an asynchronous request for a stream that was started  
7 with the `System.Net.WebRequest.BeginGetRequestStream` method.

8 **Exceptions**

Exception	Condition
<b>System.NotSupportedException</b>	This method is not overridden in the derived class.
<b>System.ArgumentException</b>	<i>asyncResult</i> was not returned by a call to <code>System.Net.WebRequest.BeginGetRequestStream</code> .
<b>System.ArgumentNullException</b>	<i>asyncResult</i> is a null reference.
<b>System.InvalidOperationException</b>	This method was called previously using <i>asyncResult</i> . -or- No stream is available.
<b>System.Net.WebException</b>	An error occurred while processing the request.

9

10

1  
2 **WebRequest.EndGetResponse(System.IAsyncResult)** Method  
3

```
4 [IAsm]  
5 .method public hidebysig virtual class System.Net.WebResponse  
6 EndGetResponse(class System.IAsyncResult asyncResult)  
7 [C#]  
8 public virtual WebResponse EndGetResponse(IAsyncResult asyncResult)
```

9 **Summary**

10 Returns a `System.Net.WebResponse` that contains a response to a specified pending  
11 request.

12 **Parameters**

Parameter	Description
<i>asyncResult</i>	A <code>System.IAsyncResult</code> object that references a pending request that was started with <code>System.Net.WebRequest.BeginGetResponse</code> .

13  
14 **Return Value**

15 A `System.Net.WebResponse` that contains a response to the request referenced by  
16 *asyncResult*.

17 **Behaviors**

18 As described above.

19

20 **Default**

21 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
22 for this method. This method throws `System.NotSupportedException`.

23

24 **How and When to Override**

25 This method must be overridden by classes that inherit from `System.Net.WebRequest` to  
26 provide this functionality.

1

## 2 Usage

3 Use this method to complete an asynchronous request for an Internet resource that was  
4 started with the `System.Net.WebRequest.BeginGetResponse` method.

5

## 6 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This method is not overridden in the derived class.
<b>System.ArgumentException</b>	<i>asyncResult</i> was not returned by a call to <code>System.Net.WebRequest.BeginGetResponse</code> .
<b>System.ArgumentNullException</b>	<i>asyncResult</i> is a null reference.
<b>System.InvalidOperationException</b>	The <code>System.Net.WebRequest.ContentLength</code> property of the current instance is greater than zero but no data has been written to the request stream. -or- This method was called previously using <i>asyncResult</i> .
<b>System.Net.WebException</b>	An error occurred while processing the request.

7

8

# 1 WebRequest.GetRequestStream() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual class System.IO.Stream GetRequestStream()  
4 [C#]  
5 public virtual Stream GetRequestStream()
```

## 6 Summary

7 Returns a `System.IO.Stream` for writing data to a resource.

## 8 Return Value

9 A `System.IO.Stream` for writing data to a resource.

## 10 Behaviors

11 As described above.

12

## 13 Default

14 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
15 for this method. This method throws `System.NotSupportedException`.

16

## 17 How and When to Override

18 This method is required to be overridden by classes that inherit from  
19 `System.Net.WebRequest`.

20

## 21 Usage

22 Use this method to initiate a request to send data to a resource and obtain a  
23 `System.IO.Stream` instance for sending data to that resource.

24

25 The `System.Net.WebRequest.GetRequestStream` method provides synchronous access  
26 to the `System.IO.Stream`. For asynchronous access, use the  
27 `System.Net.WebRequest.BeginGetRequestStream` and  
28 `System.Net.WebRequest.EndGetRequestStream` methods.

## 29 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This method is not overridden in the derived class.

1

2

# 1 WebRequest.GetResponse() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual class System.Net.WebResponse  
4 GetResponse()  
5 [C#]  
6 public virtual WebResponse GetResponse()
```

## 7 Summary

8 Returns a response to a request.

## 9 Return Value

10 A `System.Net.WebResponse` containing the response to the request.

## 11 Behaviors

12 This method returns an instance of a type derived from `System.Net.WebResponse` that  
13 is registered for the `System.Net.WebRequest.RequestUri` property of the current  
14 instance. This new instance is required to contain a response from the resource to the  
15 current request.

16  
17 If the timeout period for the request expires, or an error occurs while processing the  
18 request, this method is required to throw a `System.Net.WebException` exception.

## 19 Default

20 The `System.Net.WebRequest` class is abstract and does not provide an implementation  
21 for this method. This method throws `System.NotSupportedException`.

## 23 How and When to Override

24 This method must be overridden by classes that inherit from `System.Net.WebRequest` to  
25 provide this functionality.

## 27 Usage

28 Use this method for synchronous access to a resource. For asynchronous access, use the  
29 `System.Net.WebRequest.BeginGetResponse` and  
30 `System.Net.WebRequest.EndGetResponse` methods.

1 **Exceptions**

<b>Exception</b>	<b>Condition</b>
<b>System.NotSupportedException</b>	This method is not overridden in the derived class.
<b>System.Net.WebException</b>	The request timed out.  -or-  An error occurred while processing the request.

2

3

# 1 WebRequest.RegisterPrefix(System.String, 2 System.Net.IWebRequestCreate) Method

```
3 [ILAsm]  
4 .method public hidebysig static bool RegisterPrefix(string prefix, class  
5 System.Net.IWebRequestCreate creator)  
  
6 [C#]  
7 public static bool RegisterPrefix(string prefix, IWebRequestCreate  
8 creator)
```

## 9 Summary

10 Registers a type derived from `System.Net.WebRequest`, and associates the type with  
11 the specified URI.

## 12 Parameters

Parameter	Description
<i>prefix</i>	A <code>System.String</code> containing the URI that the derived type services. Can specify a scheme or a complete URI.
<i>creator</i>	An instance of a type that implements the <code>System.Net.IWebRequestCreate</code> interface.

## 14 Return Value

15 `true` if registration is successful; `false`, if *prefix* is already registered.

## 16 Description

17 `System.Net.HttpWebRequest` is registered to service requests for HTTP and HTTPS  
18 schemes. Attempts to register a different type for these schemes will fail.

19  
20 [Note: This method registers types that derive from `System.Net.WebRequest` to service  
21 requests. These derived types are typically registered to handle a specific protocol, such  
22 HTTP or FTP, but can be registered to handle a request to a specific server or path on a  
23 server. Therefore, *prefix* can be either a scheme or a complete URI.

24  
25 The `System.Net.WebRequest` class calls the `System.Net.IWebRequestCreate.Create`  
26 method to create additional instances of the same type as *creator*.

27 ]  
28

## 29 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>prefix</i> is null or <i>creator</i> is null.

1

## 2 Example

3 The following example demonstrates how to register a new scheme.

4

5 [C#]

6 using System;

7 using System.Net;

8

```
9 public class ftpWebRequest: WebRequest {
10     //implement ftp-specific protocol methods and properties
11 }
12
```

```
13 public class ftpCreator: IWebRequestCreate
14 {
15     public WebRequest Create(Uri uri)
16     {
17         return new ftpWebRequest();
18     }
19 }
20
```

```
21 public class RegisterPrefixExample
22 {
23
24     public static void Main()
25     {
26
27         ftpCreator creator = new ftpCreator();
28         WebRequest.RegisterPrefix("ftp://", creator);
29         WebRequest wr = WebRequest.Create("ftp://testFile");
30         Console.WriteLine(wr);
31     }
32 }
33
```

34 The output is

35

36 ftpWebRequest

37

# 1 WebRequest.ConnectionGroupName Property

```
2 [ILAsm]  
3 .property string ConnectionGroupName { public hidebysig virtual  
4 specialname string get_ConnectionGroupName() public hidebysig virtual  
5 specialname void set_ConnectionGroupName(string value) }  
  
6 [C#]  
7 public virtual string ConnectionGroupName { get; set; }
```

## 8 Summary

9 Gets or sets the name of the connection group for the current instance.

## 10 Property Value

11 A `System.String` that contains the name of the connection group for the current  
12 instance.

## 13 Description

14 This property associates specific requests within an application with a  
15 `System.Net.ServicePoint`.

## 16 Behaviors

17 As described above.

## 19 Default

20 This property throws `System.NotSupportedException`.

## 22 How and When to Override

23 This property is required to be overridden by classes that inherit from  
24 `System.Net.WebRequest`. The `System.Net.WebRequest.ConnectionGroupName` property  
25 typically associates a group of requests that share a set of credentials with a connection  
26 to an Internet resource to avoid potential security failures.

## 28 Usage

29 Use this property to get or set the name of the connection group for the current instance.

1 **Exceptions**

<b>Exception</b>	<b>Condition</b>
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

2

3

# 1 WebRequest.ContentLength Property

```
2 [ILAsm]  
3 .property int64 ContentLength { public hidebysig virtual specialname int64  
4 get_ContentLength() public hidebysig virtual specialname void  
5 set_ContentLength(int64 value) }  
6 [C#]  
7 public virtual long ContentLength { get; set; }
```

## 8 Summary

9 Gets or sets the content length of the request data being sent.

## 10 Property Value

11 A `System.Int64` containing the number of bytes of request data being sent.

## 12 Behaviors

13 This property is required to throw a `System.InvalidOperationException` exception if  
14 data has already been written to the request stream, and a  
15 `System.ArgumentOutOfRangeException` exception if the property is being set to a value  
16 less than zero.

## 18 Default

19 This property throws `System.NotSupportedException`.

## 21 How and When to Override

22 This property is required to be overridden by classes that inherit from  
23 `System.Net.WebRequest`.

## 25 Usage

26 Use this property to get the number of bytes sent to the resource.

## 28 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.
<b>System.InvalidOperationException</b>	Data has already been written to the request stream.
<b>System.ArgumentOutOfRangeException</b>	This property is being set to a value less than zero.

1

2

# 1 WebRequest.ContentType Property

```
2 [ILAsm]  
3 .property string ContentType { public hidebysig virtual specialname string  
4 get_ContentType() public hidebysig virtual specialname void  
5 set_ContentType(string value) }  
6 [C#]  
7 public virtual string ContentType { get; set; }
```

## 8 Summary

9 Gets or sets the content type of the request data being sent.

## 10 Property Value

11 A `System.String` that represents the content type of the request data.

## 12 Description

13 The `System.Net.WebRequest.ContentType` property contains the media type of the  
14 request.

15  
16 [*Note:* This is typically the MIME encoding of the content.]  
17  
18

## 19 Behaviors

20 As described above.  
21

## 22 Default

23 This property throws `System.NotSupportedException`.  
24

## 25 How and When to Override

26 This property is required to be overridden by classes that inherit from  
27 `System.Net.WebRequest`.  
28

## 29 Usage

30 Use this property to get the media type of request.

1

## 2 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

3

4

# 1 WebRequest.Credentials Property

```
2 [ILAsm]
3 .property class System.Net.ICredentials Credentials { public hidebysig
4 virtual specialname class System.Net.ICredentials get_Credentials() public
5 hidebysig virtual specialname void set_Credentials(class
6 System.Net.ICredentials value) }
7
8 [C#]
9 public virtual ICredentials Credentials { get; set; }
```

## 9 Summary

10 Gets or sets the credentials used for authenticating the client using the current instance.

## 11 Property Value

12 A `System.Net.ICredentials` object containing the authentication credentials associated  
13 with the request. The default is `null`.

## 14 Behaviors

15 As described above.

16

## 17 Default

18 This property throws `System.NotSupportedException`.

19

## 20 How and When to Override

21 This property is required to be overridden by classes that inherit from  
22 `System.Net.WebRequest`.

23

## 24 Usage

25 Use this property to store or access the user, password, and domain information of the  
26 current instance.

27

## 28 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

1

2

# 1 WebRequest.Headers Property

```
2 [ILAsm]  
3 .property class System.Net.WebHeaderCollection Headers { public hidebysig  
4 virtual specialname class System.Net.WebHeaderCollection get_Headers()  
5 public hidebysig virtual specialname void set_Headers(class  
6 System.Net.WebHeaderCollection value) }  
7 [C#]  
8 public virtual WebHeaderCollection Headers { get; set; }
```

## 9 Summary

10 Gets or sets the collection of header name/value pairs associated with the request.

## 11 Property Value

12 A `System.Net.WebHeaderCollection` containing the header name/value pairs  
13 associated with the current instance.

## 14 Description

15 This property contains a `System.Net.WebHeaderCollection` instance containing the  
16 header information to send to resources.

## 17 Behaviors

18 As described above.

## 20 Default

21 This property throws a `System.NotSupportedException` exception.

## 23 How and When to Override

24 This property must be overridden by classes that inherit from `System.Net.WebRequest`.

## 26 Usage

27 Use this property to determine the header information of a request.

1 **Exceptions**

<b>Exception</b>	<b>Condition</b>
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

2

3

# 1 WebRequest.Method Property

```
2 [ILAsm]  
3 .property string Method { public hidebysig virtual specialname string  
4 get_Method() public hidebysig virtual specialname void set_Method(string  
5 value) }  
6 [C#]  
7 public virtual string Method { get; set; }
```

## 8 Summary

9 Gets or sets the protocol method to use in the current instance.

## 10 Property Value

11 A `System.String` containing the protocol method to use in the current instance.

## 12 Behaviors

13 The default value of this property is required to be a protocol method that does not  
14 require protocol-specific properties to be set. For the HTTP protocol, this value is GET.

15

## 16 Default

17 This property throws `System.NotSupportedException`.

18

## 19 How and When to Override

20 This property must be overridden by classes that inherit from `System.Net.WebRequest`  
21 to provide this functionality.

22

## 23 Usage

24 Use this property to set the protocol-specific method that will be used to make a  
25 request.

26

## 27 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

1

2

# 1 WebRequest.PreAuthenticate Property

```
2 [ILAsm]  
3 .property bool PreAuthenticate { public hidebysig virtual specialname bool  
4 get_PreAuthenticate() public hidebysig virtual specialname void  
5 set_PreAuthenticate(bool value) }  
6 [C#]  
7 public virtual bool PreAuthenticate { get; set; }
```

## 8 Summary

9 Gets or sets a `System.Boolean` value that determines whether to send authentication  
10 information with the current request instead of waiting for an authentication challenge  
11 from the requested resource.

## 12 Property Value

13 `true` if authentication information will be sent with the current request without waiting  
14 for an authentication challenge from the requested resource; otherwise, `false`.

## 15 Behaviors

16 If `System.Net.WebRequest.PreAuthenticate` is `true`, the current instance sends  
17 authentication credentials without waiting to be challenged by the server specified by  
18 the `System.Net.WebRequest.RequestUri` property of the current instance. When this  
19 property is `false`, the current instance waits for a challenge from the server before  
20 sending credentials.

## 22 Default

23 This property throws `System.NotSupportedException`.

## 25 How and When to Override

26 This property must be overridden by classes that inherit from `System.Net.WebRequest`  
27 to provide this functionality.

## 29 Usage

30 Use this property to ensure that authentication information is sent with every request.  
31 Setting this property to `true` allows clients to improve server efficiency by avoiding  
32 extra round trips caused by authentication challenges.

1

2 **Exceptions**

<b>Exception</b>	<b>Condition</b>
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

3

4

# 1 WebRequest.Proxy Property

```
2 [ILAsm]  
3 .property class System.Net.IWebProxy Proxy { public hidebysig virtual  
4 specialname class System.Net.IWebProxy get_Proxy() public hidebysig  
5 virtual specialname void set_Proxy(class System.Net.IWebProxy value) }  
6 [C#]  
7 public virtual IWebProxy Proxy { get; set; }
```

## 8 Summary

9 Gets or sets the network proxy to use to access resources.

## 10 Property Value

11 A System.Net.IWebProxy to use to access resources.

## 12 Description

13 The System.Net.WebRequest.Proxy property identifies the network proxy that the  
14 request uses to access resources. The request is made through the proxy server rather  
15 than directly to the server hosting the resource.

## 16 Behaviors

17 If the System.Net.WebRequest.Proxy property of the current instance has not been set,  
18 the value of this property is required to be null.  
19

20 If the property is being set to null, it is required to throw a  
21 System.ArgumentNullException exception.

## 22 Default

23 This property throws System.NotSupportedException.  
24

## 25 How and When to Override

26 This property must be overridden by classes that inherit from System.Net.WebRequest  
27 to provide this functionality.  
28

## 29 Usage

30 Use this method to obtain a System.Net.IWebProxy instance that represents the proxy  
31 server used by the current instance.

1

## 2 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

3

4

# 1 WebRequest.RequestUri Property

```
2 [ILAsm]  
3 .property class System.Uri RequestUri { public hidebysig virtual  
4 specialname class System.Uri get_RequestUri() }  
5 [C#]  
6 public virtual Uri RequestUri { get; }
```

## 7 Summary

8 Gets the `System.Uri` of the resource associated with the current instance.

## 9 Property Value

10 A `System.Uri` containing the URI of the resource associated with the current instance

## 11 Description

12 This property is read-only.

## 13 Behaviors

14 `System.Net.WebRequest.RequestUri` is required to contain the URI passed to the  
15 `System.Net.WebRequest.Create` methods. If the protocol implemented by a derived  
16 class supports redirection, the derived class is required to provide a property to contain  
17 the URI that actually services the request.

18

## 19 Default

20 This property throws a `System.NotSupportedException` exception.

21

## 22 How and When to Override

23 This property must be overridden by classes that inherit from `System.Net.WebRequest`  
24 to provide this functionality.

25 ]

## 26 Usage

27 Use this property to determine the URI that the request was addressed to. For  
28 information about the URI that actually serviced the request, see  
29 `System.Net.WebResponse.ResponseUri`.

1

## 2 Exceptions

Exception	Condition
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

3

4

# 1 WebRequest.Timeout Property

```
2  [ILAsm]  
3  .property int32 Timeout { public hidebysig virtual specialname int32  
4  get_Timeout() public hidebysig virtual specialname void set_Timeout(int32  
5  value) }  
6  [C#]  
7  public virtual int Timeout { get; set; }
```

## 8 Summary

9 Gets or sets the length of time before requests for resources time out.

## 10 Property Value

11 A `System.Int32` containing the length of time, in milliseconds, before the current  
12 request will time out, or `System.Threading.Timeout.Infinite` to indicate that the  
13 request does not time out.

## 14 Behaviors

15 Classes that derive from `System.Net.WebRequest` are required to indicate a timeout by  
16 throwing a `System.Net.WebException` with the `System.Net.WebException.Status` field  
17 set to `System.Net.WebExceptionStatus.Timeout` if a request times out.

## 19 Default

20 This property throws a `System.NotSupportedException` exception.

## 22 How and When to Override

23 This property must be overridden by classes that inherit from `System.Net.WebRequest`  
24 to provide this functionality.

## 26 Usage

27 Use this property to set the timeout period for requests for resources.

28  
29 The `System.Net.WebRequest.Timeout` property affects only synchronous requests made  
30 with the `System.Net.WebRequest.GetResponse` method. To time out asynchronous  
31 requests, use the `System.Net.WebRequest.Abort` method.

1 **Exceptions**

<b>Exception</b>	<b>Condition</b>
<b>System.NotSupportedException</b>	This property is not implemented in the derived class.

2  
3