



BUILD YOUR CAREER. GET
FULL ACCESS. **SAVE 770\$**

[Start today](#)



AJAX - The XMLHttpRequest Object

[◀ Previous](#)

[Next ▶](#)

The keystone of AJAX is the XMLHttpRequest object.

1. Create an XMLHttpRequest object
2. Define a callback function
3. Open the XMLHttpRequest object
4. Send a Request to a server

The XMLHttpRequest Object

All modern browsers support the **XMLHttpRequest** object.

The **XMLHttpRequest** object can be used to exchange data with a web server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

Create an XMLHttpRequest Object

All modern browsers (Chrome, Firefox, IE, Edge, Safari, Opera) have a built-in **XMLHttpRequest** object.

Syntax for creating an **XMLHttpRequest** object:

```
variable = new XMLHttpRequest();
```

Define a Callback Function

A callback function is a function passed as a parameter to another function.

In this case, the callback function should contain the code to execute when the response is ready.

```
xhttp.onload = function() {  
    // What to do when the response is ready  
}
```

Send a Request

To send a request to a server, you can use the `open()` and `send()` methods of the `XMLHttpRequest` object:

```
xhttp.open("GET", "ajax_info.txt");  
xhttp.send();
```

Example

```
// Create an XMLHttpRequest object  
const xhttp = new XMLHttpRequest();  
  
// Define a callback function  
xhttp.onload = function() {  
    // Here you can use the Data  
}  
  
// Send a request  
xhttp.open("GET", "ajax_info.txt");  
xhttp.send();
```

[Try it Yourself »](#)

Access Across Domains

For security reasons, modern browsers do not allow access across domains.

This means that both the web page and the XML file it tries to load, must be located on the same server.

The examples on W3Schools all open XML files located on the W3Schools domain.

If you want to use the example above on one of your own web pages, the XML files you load must be located on your own server.

ADVERTISEMENT

XMLHttpRequest Object Methods

Method	Description
<code>new XMLHttpRequest()</code>	Creates a new XMLHttpRequest object
<code>abort()</code>	Cancels the current request
<code>getAllResponseHeaders()</code>	Returns header information
<code>getResponseHeader()</code>	Returns specific header information
<code>open(<i>method</i>, <i>url</i>, <i>async</i>, <i>user</i>, <i>psw</i>)</code>	<p>Specifies the request</p> <p><i>method</i>: the request type GET or POST <i>url</i>: the file location <i>async</i>: true (asynchronous) or false (synchronous) <i>user</i>: optional user name <i>psw</i>: optional password</p>
<code>send()</code>	Sends the request to the server Used for GET requests
<code>send(<i>string</i>)</code>	Sends the request to the server. Used for POST requests
<code>setRequestHeader()</code>	Adds a label/value pair to the header to be sent

XMLHttpRequest Object Properties

Property	Description
onload	Defines a function to be called when the request is received (loaded)
onreadystatechange	Defines a function to be called when the readyState property changes
readyState	Holds the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
responseText	Returns the response data as a string
responseXML	Returns the response data as XML data
status	Returns the status-number of a request 200: "OK" 403: "Forbidden" 404: "Not Found" For a complete list go to the Http Messages Reference
statusText	Returns the status-text (e.g. "OK" or "Not Found")

The onload Property

With the `XMLHttpRequest` object you can define a callback function to be executed when the request receives an answer.

The function is defined in the `onload` property of the `XMLHttpRequest` object:

Example

```
xhttp.onload = function() {
  document.getElementById("demo").innerHTML = this.responseText;
}
```

```
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
```

[Try it Yourself »](#)

Multiple Callback Functions

If you have more than one AJAX task in a website, you should create one function for executing the `XMLHttpRequest` object, and one callback function for each AJAX task.

The function call should contain the URL and what function to call when the response is ready.

Example

```
loadDoc("url-1", myFunction1);

loadDoc("url-2", myFunction2);

function loadDoc(url, cFunction) {
  const xhttp = new XMLHttpRequest();
  xhttp.onload = function() {cFunction(this)};
  xhttp.open("GET", url);
  xhttp.send();
}

function myFunction1(xhttp) {
  // action goes here
}
function myFunction2(xhttp) {
  // action goes here
}
```

The `onreadystatechange` Property

The `readyState` property holds the status of the XMLHttpRequest.

The `onreadystatechange` property defines a callback function to be executed when the readyState changes.

The `status` property and the `statusText` properties hold the status of the XMLHttpRequest object.

Property	Description
<code>onreadystatechange</code>	Defines a function to be called when the readyState property changes
<code>readyState</code>	Holds the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
<code>status</code>	200: "OK" 403: "Forbidden" 404: "Page not found" For a complete list go to the Http Messages Reference
<code>statusText</code>	Returns the status-text (e.g. "OK" or "Not Found")

The `onreadystatechange` function is called every time the readyState changes.

When `readyState` is 4 and `status` is 200, the response is ready:

Example

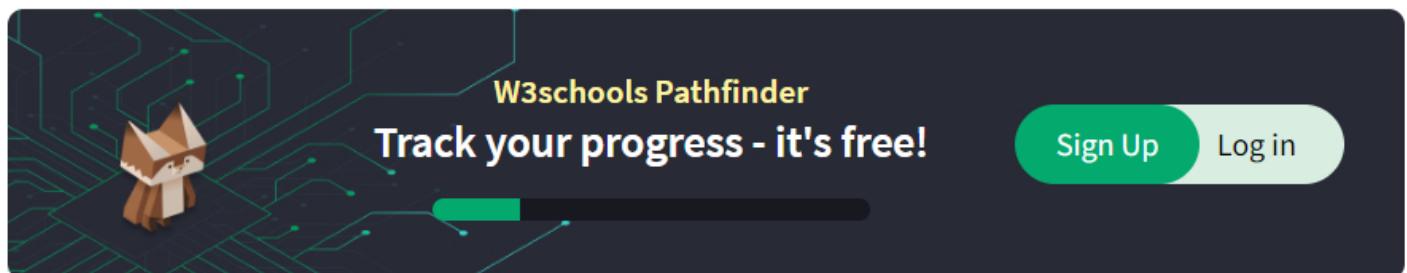
```
function loadDoc() {
  const xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("demo").innerHTML =
        this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
```

[Try it Yourself »](#)

The `onreadystatechange` event is triggered four times (1-4), one time for each change in the readyState.

[« Previous](#)

[Next »](#)



ADVERTISEMENT



BUILD YOUR CAREER. GET
FULL ACCESS. **SAVE 770\$**

[Start today](#)



An advertisement for the Front-End Certification Program. It features the HTML5, CSS3, and JS logos at the top. The text "Front-End Certification Program" is prominently displayed. Below it, it says "Become a certified Front-End Developer! No experience needed". A "Start Today!" button is at the bottom left, and the W3schools logo is at the bottom right.

COLOR PICKER





ADVERTISEMENT

Build your career. Get Full Access.

Lifelong access to all W3Schools courses and certifications!

Start today



SAVE 770\$



SPACES

UPGRADE

AD-FREE

NEWSLETTER

GET CERTIFIED

CONTACT US

Top Tutorials

HTML Tutorial
CSS Tutorial

[CSS Tutorial](#)
[JavaScript Tutorial](#)
[How To Tutorial](#)
[SQL Tutorial](#)
[Python Tutorial](#)
[W3.CSS Tutorial](#)
[Bootstrap Tutorial](#)
[PHP Tutorial](#)
[Java Tutorial](#)
[C++ Tutorial](#)
[jQuery Tutorial](#)

Top References

[HTML Reference](#)
[CSS Reference](#)
[JavaScript Reference](#)
[SQL Reference](#)
[Python Reference](#)
[W3.CSS Reference](#)
[Bootstrap Reference](#)
[PHP Reference](#)
[HTML Colors](#)
[Java Reference](#)
[Angular Reference](#)
[jQuery Reference](#)

Top Examples

[HTML Examples](#)
[CSS Examples](#)
[JavaScript Examples](#)
[How To Examples](#)
[SQL Examples](#)
[Python Examples](#)
[W3.CSS Examples](#)
[Bootstrap Examples](#)
[PHP Examples](#)
[Java Examples](#)
[XML Examples](#)
[jQuery Examples](#)

Get Certified

[HTML Certificate](#)
[CSS Certificate](#)
[JavaScript Certificate](#)
[Front End Certificate](#)

[SQL Certificate](#)
[Python Certificate](#)
[PHP Certificate](#)
[jQuery Certificate](#)
[Java Certificate](#)
[C++ Certificate](#)
[C# Certificate](#)
[XML Certificate](#)



[FORUM](#) [ABOUT](#) [CLASSROOM](#)

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using W3Schools, you agree to have read and accepted our [terms of use](#), [cookie and privacy policy](#).

[Copyright 1999-2024 by Refsnes Data. All Rights Reserved.](#) [W3Schools is Powered by W3.CSS.](#)

