

AJAX

Ajax: Asynchronous JavaScript and XML*

AJAX is a developer's dream, because you can:

- Read data from a web server - after the page has loaded
- Update a web page without reloading the page
- Send data to a web server - in the background

* Now **JSON: JavaScript Object Notation** is more popular than XML

EXAMPLE

https://www.w3schools.com/js/tryit.asp?filename=tryjs_ajax_first

```
<html>
<body>

<div id="demo">
<button type="button" onclick="loadDoc()">Change Content</button>
</div>

<script>
function loadDoc() {
  var 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", true);
  xhttp.send();
}
</script>
```

```
</body>  
</html>
```

CREATE AN XMLHttpRequest OBJECT

Syntax for creating an XMLHttpRequest object:

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

XMLHttpRequest METHODS

Method	Description
<code>getAllResponseHeaders()</code>	Returns header information
<code>getResponseHeader()</code>	Returns specific header information
<code>open(<i>method</i>, <i>url</i>, <i>async</i>)</code>	Specifies the request <i>method</i> : the request type GET or POST <i>url</i> : the file location <i>async</i> : true (asynchronous)
<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

GET REQUESTS

```
| xmlhttp.open("GET", "demo_get1.html", true);  
| xmlhttp.send();
```

```
| xmlhttp.open("GET", "demo_get2.php?fname=Henry&lname=Ford", true);  
| xmlhttp.send();
```

POST REQUESTS

```
xhttp.open("POST", "ajax_test.php", true);  
xhttp.setRequestHeader("Content-type", "application/x-www-form-  
urlencoded");  
xhttp.send("fname=Henry&lname=Ford");
```

ONREADYSTATECHANGE PROPERTY

```
xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
  
document.getElementById("demo").innerHTML = this.responseText;  
    }  
};
```

onreadystatechange	Defines a function to be called when the readyState property changes
--------------------	--

READYSTATE & STATUS PROPERTIES

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
status	200: "OK" 403: "Forbidden" 404: "Page not found" For a complete list go to the Http Messages Reference
statusText	Returns the status-text (e.g. "OK" or "Not Found")

AJAX EXAMPLE #1

https://www.w3schools.com/js/tryit.asp?filename=tryjs_ajax_suggest_php

```
<html><head>
<script>
function showHint(str) {
    if (str.length == 0) {
        document.getElementById("txtHint").innerHTML = "";
        return;
    } else {
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
document.getElementById("txtHint").innerHTML =this.responseText;
                }
            };
        xmlhttp.open("GET", "gethint.php?q=" + str, true);
        xmlhttp.send();
    }
}
</script> </head>
```

AJAX EXAMPLE #2

https://www.w3schools.com/js/tryit.asp?filename=tryjs_ajax_suggest_php

```
<body>

<p><b>Start typing a name in the input field below:</b></p>
<form>
First name: <input type="text" onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>

</body>
</html>
```

AXIOS

```
$ npm install axios
```

```
import axios from "axios";
```

- Automatic request body serialization to:
 - JSON (`application/json`)
 - Multipart / FormData (`multipart/form-data`)
 - URL encoded form (`application/x-www-form-urlencoded`)
- Posting HTML forms as JSON
- Automatic JSON data handling in response

Axios GET

```
async function getUser() {  
  try {  
    const response = await axios.get('/user?ID=12345');  
    console.log(response);  
  } catch (error) {  
    console.error(error);  
  }  
}
```

Axios POST

```
async function addUser(user) {
  try {
    const response = await axios.post('/user', user,
    {
      headers: {
        'Content-Type': 'application/json'
      }
    })
    console.log(response);
  } catch (error) {
    console.error(error);
  }
}
```

JSON

JSON: **J**ava**S**cript **O**bject **N**otation.

JSON is a syntax for storing and exchanging data.

JSON is text, written with JavaScript object notation.

JSON VALID DATA TYPES

In JSON, values must be one of the following data types:

- a string
- a number
- an array
- a boolean
- *null*

JSON OBJECTS

```
{ "name": "John", "age": 30, "car": null };
```

JSON ARRAYS

```
[ "Ford", "BMW", "Fiat" ]
```

JSON ARRAYS OF OBJECTS

[

```
{ "name": "Jose Bastos", "micropost_id": "70", "user_id": "74", "content": "mais um", "created_at": "2017-11-12 00:39:57", "total": "2" },
```

```
{ "name": "Jose Bastos", "micropost_id": "69", "user_id": "74", "content": "vamos a ver", "created_at": "2017-11-12 00:06:08", "total": "2" },
```

```
{ "name": "Jose Bastos", "micropost_id": "68", "user_id": "74", "content": "\r\ntest", "created_at": "2017-11-11 23:44:44", "total": "2" }
```

]

JAVASCRIPT PARSING JSON

Imagine we received this text from a web server:

```
' { "name": "John", "age": 30, "city": "New York" } '
```

Use the JavaScript function `JSON.parse()` to convert text into a JavaScript object:

```
var obj = JSON.parse(' { "name": "John", "age": 30, "city": "New York" } ');
```

STRINGIFY A JAVASCRIPT OBJECT

If we have this object in JavaScript:

```
var obj = { "name": "John", "age": 30, "city": "New York" };
```

Use the JavaScript function `JSON.stringify()` to convert it into a string.

```
var myJSON = JSON.stringify(obj);
```

JSON PHP #1

Objects in PHP can be converted into JSON by using the PHP function `json_encode()`:

```
<?php
$myObj->name = "John";
$myObj->age = 30;
$myObj->city = "New York";

$myJSON = json_encode($myObj);

echo $myJSON;
?>

{"age":30,"city":"New York","name":"John"}
```

JSON PHP #2

Associative Arrays in PHP can be converted into JSON by using the PHP function `json_encode()`:

```
<?php
```

```
$age = array("Peter"=>"35", "Ben"=>"37",  
"Joe"=>"43");
```

```
$myJSON = json_encode($age);
```

```
echo $myJSON;
```

```
?>
```

```
{"Peter": "35", "Ben": "37", "Joe": "43" }
```