XMLHttpRequest

XMLHttpRequest HTML-5.com is a great guide for web developers. TV Series & Actors and Actresses. Follow TV Series and HTML 5 on Google+.

JavaScript XMLHttpRequest

<script type="text/javascript"> function showProgress(req, xferDirection, ev) { var s; switch (req.readyState) { case 0: s = "UNSENT"; break; case 1: s = "OPENED"; break; case 2: s = "HEADERS_RECEIVED"; break; case 3: s = "LOADING"; break; case 4: s = "DONE"; break; default: s = "unknown state " + req.readyState; break; } var elem = document.getElementById("status"); var node = document.createElement("li"); node.appendChild(document.createTextNode(xferDirection + " " + ev.type + ":" + (ev.type == "readystatechange" ? " readyState=" + req.readyState + " (" + s + ")" : "") + (req.readyState == 4 ? (req.status != 0 ? " status=" + req.status + " (" + req.statusText + ")" : "") : "") + " " + ev + (ev.loaded > 0 || ev.lengthComputable ? " loaded " + ev.loaded + " of " + ev.total + (ev.lengthComputable ? " " : " not ") + "lengthComputable " + (ev.total > 0 ? (Math.round(ev.loaded * 100 / ev.total)) + "%" : "") : ""))); elem.appendChild(node); } function createRequest(name, retry) { var req = new XMLHttpRequest(); req.addEventListener("readystatechange", function(ev) {showProgress(req, name, ev ? ev : window.event);}); req.addEventListener("load", function(ev){ showProgress(req, name + " download", ev ? ev : window.event); var elem = document.getElementById("result"); elem.innerHTML = req.responseText; }); req.addEventListener("loadstart", function(ev){showProgress(req, name + " download", ev ? ev : window.event);}); req.addEventListener("progress", function(ev){showProgress(req, name + " download", ev ? ev : window.event);}); req.addEventListener("loadend", function(ev){showProgress(req, name + " download", ev ? ev : window.event);}); req.addEventListener("error", function(ev){ showProgress(req, name + " download", ev ? ev : window.event); if (retry) retry(); }); req.addEventListener("abort", function(ev) {showProgress(req, name + " download", ev ? ev : window.event);}); if (req.upload) { req.upload.addEventListener("load", function(ev){showProgress(req, name + " upload", ev ? ev : window.event);}); req.upload.addEventListener("loadstart", function(ev){showProgress(req, name + " upload", ev ? ev : window.event);}); req.upload.addEventListener("progress", function(ev){showProgress(req, name + " upload", ev ? ev : window.event);}); req.upload.addEventListener("loadend", function(ev){showProgress(req, name + " upload", ev ? ev : window.event);}); req.upload.addEventListener("error", function(ev){showProgress(req, name + " upload", ev ? ev : window.event);}); req.upload.addEventListener("abort", function(ev) {showProgress(req, name + " upload", ev ? ev : window.event);}); } return req; } function openRequest(req, method, url) { try { req.open(method, url, true); } catch (ex) { var elem = document.getElementById("status"); var node = document.createElement("li"); node.appendChild(document.createTextNode(ex + " (browser does not support cross-site XMLHttpRequest)")); elem.appendChild(node); alert("Browser does not support cross-site XMLHttpRequest: " + ex); return true; } return false; } function sendGetRequest(host, path, data) { var elem = document.getElementById("status"); var query = null; if (data) for (var key in data) { query = (query ? query + "&" : "") + key + (data[key] ? "=" + data[key] : ""); }

var req1 = createRequest("#1", function() { var req2 = createRequest("#2"); var node = document.createElement("li"); node.appendChild(document.createTextNode("GET request #2 - doing retry")); elem.appendChild(node); var url = "http://" + host + path + (query ? "?" + query : ""); if (openRequest(req2, "GET", url)) return; req2.send(null); }); var node = document.createElement("li"); node.appendChild(document.createTextNode("GET request #1 created")); elem.appendChild(node); var url = "https://" + host + path + (query ? "?" + query : ""); if (openRequest(req1, "GET", url)) return; req1.send(null); } function sendPostRequest(host, path, data) { var elem = document.getElementById("status"); if (typeof FormData == "undefined") { var node = document.createElement("li"); node.appendChild(document.createTextNode("POST requires IE 10 or later. Fallback to GET.")); elem.appendChild(node); sendGetRequest(host, path, data); return; } var formdata = new FormData(); if (data) for (var key in data) { formdata.append(key, data[key]); } var req1 = createRequest("#1", function() { var req2 = createRequest("#2"); var node = document.createElement("li"); node.appendChild(document.createTextNode("POST request #2 - doing retry")); elem.appendChild(node); var url = "http://www.ExampleOnly.com/demo/env.php"; if (openRequest(req2, "POST", url)) return; req2.send(formdata); }); var node = document.createElement("li"); node.appendChild(document.createTextNode("POST request #1 created")); elem.appendChild(node); var url = "https://www.ExampleOnly.com/demo/env.php"; if (openRequest(req1, "POST", url)) return; req1.send(formdata); } function doGetRequest() { var elem = document.getElementById("status"); elem.innerHTML = ""; sendGetRequest("www.ExampleOnly.com", "/demo/env.php", { query: null, param: "value" }); } function doPostRequest() { var elem = document.getElementById("status"); elem.innerHTML = ""; sendPostRequest("www.ExampleOnly.com", "/demo/env.php", { query: null, param: "value" }); } </script>
XMLHttpRequest is a JavaScript object type that can be used to retrieve a resource via a URL. Despite the <q>Http</q> in the name of the type, an XMLHttpRequest object can be used to retrieve resources via various protocols including the

- http,
- ftp, and
- file protocols.

First, create an instance of XMLHttpRequest:

```
  var req = new XMLHttpRequest();
```

Add any necessary event listeners to the request:

```
  req.addEventListener("load", function(ev){alert("status=" + req.status);});
  req.addEventListener("error", function(ev){alert("handle error");});
```

- If you attempt to access `status` or `statusText` when `readyState` is not `DONE` you will get an **INVALID_STATE_ERR DOM Exception 11** error. In any event other than `load` or `error`, which are only called when readystate is `DONE`, be sure to do `if (req.readyState == 4)` before checking `req.state` or `req.readyState`.
- If the destination site is a different domain than the origin, you need to add headers at the *destination* site to avoid errors such as **XMLHttpRequest cannot load ...** with **Origin http://*www.origin.com* is not allowed by Access-Control-Allow-Origin** or **Request header field Content-Type is not allowed by Access-Control-Allow-Headers**.

- In addition, when the destination is different from the origin, the *only* status that is returned is 200 OK; otherwise the *req*.status is zero, even when an HTTP error such as 404 Not Found has occurred.

To send a GET request:

```
var url = "http://www.ExampleOnly.com/demo/env.php?query&param=value";
req.open("GET", url, true);
req.send(null);
```

- In the third parameter of `req.open`, specify `true` for an asynchronous request or `false` for a synchronous request

<input type="submit" value=" Try it! " onclick="doGetRequest();return false"></input>
To send a POST request:

```
var url = "http://www.ExampleOnly.com/demo/env.php;
req.open("POST", url, true);
var formData = new FormData();
formData.append("param", "value");
req.send(formData);
```

- In the third parameter of `req.open`, specify `true` for an asynchronous request or `false` for a synchronous request

<input type="submit" value=" Try it! " onclick="doPostRequest();return false"></input>

back to top

THE END