**Making a Cross Domain Request with JQuery:**

Problem:

* JavaScript follow Same Origin Policy. So you cannot call cross origin directly from JavaScript.

Solutions:

There are many way to solve this issue but this is good options.

1. CORS (Cross-Origin Resource Sharing)

**CORS (Cross-Origin Resource Sharing):**

* <http://www.w3.org/TR/cors/>

This is proper standard document for how cors work.

* From W3C spec,

CORS is a W3C spec that allows cross-domain communication from the browser. By building on top of the XMLHttpRequest object, CORS allows developers to work with the same idioms as same-domain requests.

* To make a CORS request you simply use
  1. XMLHttpRequest in Firefox 3.5+, Safari 4+ & Chrome and
  2. XDomainRequest object in IE8+.
* **withCredentials**
  1. Standard CORS requests do not send or set any cookies by default. In order to include cookies as part of the request, you need to set the XMLHttpRequest’s .withCredentials property to true:
     + xhr.withCredentials = true;
  2. In order for this to work, the server must also enable credentials by setting the Access-Control-Allow-Credentials response header to “true”. See the server section for details.
     + Access-Control-Allow-Credentials: true
  3. **The *.withCredentials* property will include any cookies from the remote domain in the request, and it will also set any cookies from the remote domain**. Note that these cookies still honor same-origin policies, so your JavaScript code can’t access the cookies from document.cookie or the response headers. They can only be controlled by the remote domain.

## CORS from JQuery

* JQuery's [$.ajax()](http://api.jquery.com/jQuery.ajax/) method can be used to make both regular XHR and CORS requests. A few notes about JQuery's implementation:
* JQuery's CORS implementation doesn't support IE's XDomainRequest object. But there are JQuery plugins that enable this. See<http://bugs.jquery.com/ticket/8283> for details.
* The $.support.cors boolean will be set to true if the browser supports CORS (This returns false in IE, see bullet above). This can be a quick way to check for CORS support.

Here's sample code for making a CORS request with JQuery. The comments give more details on how certain properties interact with CORS.

$.ajax({

*// The 'type' property sets the HTTP method.*

*// A value of 'PUT' or 'DELETE' will trigger a preflight request.*

type: 'GET',

*// The URL to make the request to.*

url: 'http://updates.html5rocks.com',

*// The 'contentType' property sets the 'Content-Type' header.*

*// The JQuery default for this property is*

*// 'application/x-www-form-urlencoded; charset=UTF-8', which does not trigger*

*// a preflight. If you set this value to anything other than*

*// application/x-www-form-urlencoded, multipart/form-data, or text/plain,*

*// you will trigger a preflight request.*

contentType: 'text/plain',

xhrFields: {

*// The 'xhrFields' property sets additional fields on the XMLHttpRequest.*

*// This can be used to set the 'withCredentials' property.*

*// Set the value to 'true' if you'd like to pass cookies to the server.*

*// If this is enabled, your server must respond with the header*

*// 'Access-Control-Allow-Credentials: true'.*

withCredentials: false

},

headers: {

*// Set any custom headers here.*

*// If you set any non-simple headers, your server must include these*

*// headers in the 'Access-Control-Allow-Headers' response header.*

},

success: function() {

*// Here's where you handle a successful response.*

},

error: function() {

*// Here's where you handle an error response.*

*// Note that if the error was due to a CORS issue,*

*// this function will still fire, but there won't be any additional*

*// information about the error.*

}

});

**Note:**

* if AJAX request adds any custom HTTP headers or use any method other than GET, POST or HEAD as a request type; browser will make a "preflight" request to check if the server responds with correct headers before sending the actual request. This adds an overhead to your AJAX requests.
* Important links that share excellent knowledge of CROS and other cross origin policies.
  + <http://www.html5rocks.com/en/tutorials/cors/>
  + <http://enable-cors.org/index.html>
  + <http://www.tuicool.com/articles/U7VVBf>
  + <http://cypressnorth.com/programming/cross-domain-ajax-request-with-json-response-for-iefirefoxchrome-safari-jquery/>