NWISWeb Water Services W3C CORS Instructions

David Coyle

December 8, 2022

Version 1

Contents

Introduction

This is a a set of instructions for web developers who are trying to leverage the W3C CORS functionality to use USGS NWISWeb Water Services based on their Javascript based web applications.

Problem Description

The end user developer has a Javascript based web application that uses the NWISWeb Water Services API to call on NWIS Instantaneous Values Service (or similar NWIS web service). Sometimes, these calls return a 403 forbidden error. If I try to repeat the call, this error goes away. The issue is intermittent and not 100% reproduceable.

Analysis

The first question to ask the user is “Do some requests go back more than 120 days?”. If this is the case then see below the section Accessing Data Periods Greater than 120 Days.

W3C CORS Description

The World Wide Web Consortium (W3C) is the standards body that defined Cross-Origin Resource Sharing (CORS). You may already be aware of W3C and CORS. However if you are not familiar or you need to reacquaint yourself with CORS protocol details for your use of NWISWeb Waterservices API scenarios then you can check the references section below at the end of this document (see References).

CORS API Test and Solution

Here is the basic W3C CORS API test and solution. We have generated an example CORS TEST here at the below link. A web developer can use a similar test query. You may already be using CORS but this is just in case it will help to understand and test your NWISWeb Waterservices API scenario. As described above and in the references links, W3C CORS is Cross-Origin Resource Sharing (CORS).

A typical issue we have seen is that is that the W3C CORS specification does not address “web redirects”. So for that use-case, see the section below Accessing Data Periods Greater than 120 Days.

We recommend use of the “Test CORS” web site for your initial application development process. This site is useful in case there is some issue in your code or in the Javascript Library code you are using. The site also provides you with example Javascript code that you can use directly within your application.

Note that in most cases, what looks to you in your F12 screen like an HTTP 403 response has been returned from the server, this is due to the security policy that is embedded in your web browser. You are seeing HTTP 403 returned by the browser based on the browsers’ built-in CORS security policy. So look at your network traffic via F12 or better yet through an external tool like Fiddler.

Generate and run a “Test CORS test” and study the generated Javascript code. Run the CORS TEST at test-cors.org which I sent to you in the link above (This is a slightly different query). Then click on the “Code” tab. Here will you see a sample of properly written Javascript code to deal with CORS requirements.

Sample of properly written Javascript code to deal with CORS

https://www.test-cors.org/#?client_method=GET&client_credentials=false&server_url=https://nwis.waterservices.usgs.gov/nwis/iv?format=json&sites=11123000&startDT=2018-11-20T0:0:00.000&endDT=2019-11-23T23:59:59.000&parameterCd=00060,00065&siteStatus=all

Additional links to Study

- https://www.w3.org/wiki/CORS_Enabled

- https://www.w3.org/wiki/CORS

- https://w3c.github.io/webappsec-cors-for-developers/

CORS example Javascript Code

See the CORS example Javascript below: This is just an example and pertains to another user and site that a user was building:

```
var createCORSRequest = function(method, url) {
  var xhr = new XMLHttpRequest();
  if ("withCredentials" in xhr) {
    // Most browsers.
    xhr.open(method, url, true);
  } else if (typeof XDomainRequest != "undefined") {
    // IE8 & IE9
    xhr = new XDomainRequest();
    xhr.open(method, url);
  } else {
    // CORS not supported.
    xhr = null;
  }
  return xhr;
};

var url = '
https://waterservices.usgs.gov/nwis/iv/?format=json&indent=on&sites=04
119055&parameterCd=00060';
var method = 'GET';
var xhr = createCORSRequest(method, url);

xhr.onload = function() {
  // Success code goes here.
};

xhr.onerror = function() {
  // Error code goes here.
};

xhr.send();
```

Accessing Data Periods Greater than 120 Days

Note that ideally your code and/or library should access waterservices.usgs.gov for all requests up to 120 days. For all requests that are 121 days or greater (e.g., P121D) your code should then use nwis.waterservices.usgs.gov.

If you can make sure these two use-case are implemented in your code, this will provide better workload management for NWISWeb plus faster response times for your end users.

Note: To outside users we have been reluctant to give out the nwis.waterservices address. But we have done so since that has been the only solution to the problem. The use of the nwis.waterservices address has been the only solution in all cases where greater than 120 days of values is needed.

We have copied in the request URL and generated a valid Test CORS test case for you below:

CORS Request URL

https://nwis.waterservices.usgs.gov/nwis/iv/?format=rdb,1.0&period=P365D&sites=390748121231701&parmCd=00065

Valid Test CORS test case

https://www.test-cors.org/#?client_method=GET&client_credentials=false&sites=11123000&startDT=2018-11-20T0%3A0%3A00.000&endDT=2019-11-23T23%3A59%3A59.000&parameterCd=00060%2C00065&siteStatus=all&server_url=https%3A%2F%2Fnwis.waterservices.usgs.gov%2Fnwis%2Fiv%2F%3Fformat%3Drdb%2C1.0%26period%3DP365D%26sites%3D390748121231701%26parmCd%3D00065&server_enable=true&server_status=200&server_credentials=false&server_tabs=remote

Example of a 301 Redirect for a request using P121D (i.e., historical query):

#	Result	Protocol	Host	URL	Body	Caching	Content-Type	Process	Comments	Custom
55	301	HTTPS	waterservices.usgs.gov	/nwis/iv/?format=rdb,1.0&period=P121D&sites=390748121231701&parmCd=00065	0	max-age=900; Expires: Wed, 22 Sep 2021 21:07:56 GMT		fiddler:5424		
56	200	HTTPS	nwis.waterservices.usgs.gov	/nwis/iv/?format=rdb,1.0&period=P121D&sites=390748121231701&parmCd=00065	81	max-age=900; Expires: Wed,	text/plain;charset=UTF-8	fiddler:5424		

22
Sep
202
1
21:0
7:56
GM
T

References

1. World Wide Web Consortium (W3C), https://www.w3.org/Consortium/

2. Cross-Origin Resource Sharing (CORS), https://www.w3.org/Security/wiki/CORS

3. CORS Enabled, https://www.w3.org/wiki/CORS_Enabled

4. CORS Wiki, https://www.w3.org/wiki/CORS

5. CORS Standard, https://www.w3.org/TR/2020/SPSD-cors-20200602/