Saxon-JS - Support #3163

Supplying base64 encoded source and/or SEF to SaxonJS.transform

2017-03-13 14:42 - Jan-Paul van der Velden

Status:	Closed	**Start date:**	2017-03-13
Priority:	Normal	**Due date:**	
Assignee:	Debbie Lockett	**% Done:**	0%
Category:		**Estimated time:**	0:00 hour
Sprint/Milestone:		**Spent time:**	0:00 hour
Applies to JS Branch:	1.0	**Fixed in JS Release:**	
Fix Committed on JS Branch:		**SEF Generated with:**	

Description

SaxonJS throws an 'Access Denied' in the file SaxonJS.js when the method 'asyncGet' is called. This method uses 'xhr.open' with a 'data:text/xml;charset=utf-8;base64,' URI which causes the Exception.

SaxonJS 1.0 is used.

Platform is IE 11 on Windows 8.1

The web-application runs on a Ubuntu desktop and the tested client runs on Windows 8.1 in Virtualbox.

The URL used to run the web-application was http://10.0.2.2:8081 (gateway URL between HOST and GUEST in Virtualbox).

The Internet Explorer Security Zone for this server was opened to allow cross-site scripting (and every other option available).

Error number: -2147024891

Full error from IE: http://pastebin.com/cv5DbLX1

History

#1 - 2017-03-21 16:12 - Jan-Paul van der Velden

- File ie11.png added

Because of this error no validations are executed which makes it a blocker for us for Edge.

#2 - 2017-03-21 16:19 - Michael Kay

- Assignee set to Debbie Lockett

#3 - 2017-03-21 16:41 - Debbie Lockett

- Status changed from New to In Progress

Apologies for not responding to this bug sooner, especially since it is marked as high priority! Reproducing the problem will be the first challenge, but your analysis helps to see where it arises, to get started towards a solution.

#4 - 2017-03-21 17:02 - Debbie Lockett

Could you tell us what SaxonJS.transform() call you are using?

#5 - 2017-03-22 10:59 - Jan-Paul van der Velden

This is a snippet of the code:

window.SaxonJS.transform(

{

stylesheetLocation: sef,

initialTemplate: formulaCall.id,

destination: "application",

sourceLocation: `data:text/xml,$

{ window.encodeURIComponent(buildInputXML(getFactValueFnOverride || getFactValueFn, // Allow fn override at call time. getCellInfoFn, formulaCall, unitResolver)) }

`

},

Hope this helps..

#6 - 2017-03-22 11:32 - Debbie Lockett

Yes, that is helpful. We must admit we have not tested the use of data URIs for the source location. The Wikipedia page (https://en.wikipedia.org/wiki/Data_URI_scheme) warns that "As of 2015, data URIs are fully supported by most major browsers, and partially supported in Internet Explorer and Microsoft Edge." so this could be a problem specific to IE & Edge which is out of our control. Have you tested your example on other browsers?

It is still hard to tell what it is you are trying to supply as the source. Perhaps you would be better using sourceNode or sourceText rather than sourceLocation (see http://www.saxonica.com/saxon-js/documentation/index.html#!api/transform)?

#7 - 2017-03-22 14:52 - Jan-Paul van der Velden

Thanks for the feedback. Other browser work fine.

We'll refactor the code to use the sourceText.

#8 - 2017-03-30 18:05 - Debbie Lockett

- Status changed from In Progress to Closed

- Priority changed from High to Normal

Marking this bug as closed, since a work around to avoid the problem has been supplied. That is, to use sourceText rather than sourceLocation with a data URI, in the SaxonJS.transform call.

Please let us know (by reopening this bug, or opening a new one) if you have further issues. Thanks.

#9 - 2017-04-07 09:56 - Jan Paul Stegeman

Hello Debbie (and others),

We've switched our codebase to use the sourceText interface, in stead of the sourceLocation interface. And although this did improve the performance somewhat (I'd say around 70% in our workload - which is a-typical), the exact same error message popped up in IE11 and Chrome. So we have been forced to re-open the bug in our tracking software (JIRA), so this issue is still ongoing and i've re-opened this bug as well.

I'm open, and very reachable, for suggestions as we've run out ;-)

Jan Paul

#10 - 2017-04-07 17:36 - Debbie Lockett

- Status changed from Closed to In Progress

Ok, sorry to hear you still have problems. I'm going to need more details on what you're now supplying as the source, to try to track down the error. What you are now supplying for the sourceText?

Is the Access Denied error message really exactly the same? Could you copy let me know exactly what it says?

Thanks.

#11 - 2017-04-10 11:07 - Jan Paul Stegeman

- File test.sef added

- File test.xml added

Hi Debbie,

Find attached the SEF that we are using, and the xml this SEF is being applied on. The resulting document is a simple "true" or "false", in this case.

Finding the error message on IE11 is a bit more involved since I don't personally have an IE11 (or Edge) browser available. I am able to arrange that, ofcourse, during the day.

Jan Paul

#12 - 2017-04-11 17:42 - Debbie Lockett

Are you now trying to supply the SEF as a data URI? Or has something gone wrong with the upload here? When I open the attached test.sef file it begins "data:text/xml..." Please could you upload it again, after first saving it explicitly as an .xml file?

If the XML you are supplying as source actually is a saved file like this, then you really should be able to use the sourceLocation interface after all. You should supply a string with the relative path to the source from the HTML page (e.g. sourceLocation: "xsl/test.sef").

#13 - 2017-04-12 12:57 - Jan Paul Stegeman

Hi Debbie,

We are seeing that using sourceText gives us much more performance in browser that do work (which is everything except Microsoft browser) so we are going with sourceText anyway. Also, you advised us to use sourceText in the first place since using sourceLocation didn't work in IE/Edge. So neither sourceLocation nor sourceText works. Something else must be going on. Maybe this error message, from an IE11 console, gives you more information?

Transform options supplied: stylesheetLocation, initialTemplate, destination, sourceText,

Unhandled promise rejection Error: Access is denied.

"Unhandled promise rejection"

{

```
  [functions]: ,
```

```
  __proto__: { },
```

```
  description: "Access is denied.
```

",

```
  message: "Access is denied.
```

",

```
  name: "Error",
```

```
  number: -2147024891,
```

```
  stack: "Error: Access is denied.
```

at a.asyncGet (https://abcportaal/SaxonJS.min.js:1:11059)

at a.asyncGetXml (https://abcportaal/SaxonJS.min.js:1:11311)

at a.asyncGetMultipleXml (https://abcportaal/SaxonJS.min.js:1:11516)

at transform (https://abcportaal/SaxonJS.min.js:1:5328)

#14 - 2017-04-12 13:16 - Debbie Lockett

It's good to see the full IE error message, but I'm afraid it does not really tell me much more. I would still like to be able to actually reproduce the problem myself. Can you tell me again exactly what SaxonJS.transform() call you are now using? And attach the test SEF as an XML file (or even the original test XSLT stylesheet and I will generate the SEF myself). Thanks.

#15 - 2017-04-13 09:53 - Jan Paul Stegeman

Debbie,

We're doing this:

```
window.SaxonJS.transform(
            {
              stylesheetLocation: sef,
              initialTemplate: formulaCall.id,
              destination: "application",
              sourceText: inputXml,
            },
            transformResult => {
              resolve(resultParser(transformResult.textContent));
            });
```

Where the **sef** is the *test.sef* I sent you, and the **inputXml** the *test.xml* file.

The *test.sef* file is base64 encoded, you can easily decode it yourself. The SaxonJS code does this for you.

#16 - 2017-04-13 13:13 - Jan Paul Stegeman

Hi Debbie,

To assist you in isolating the problem, I've made a sample project and uploaded it to my GitHub. I called it "saxonable" since I didn't have inspiration for another name.

This is the minimum viable set to have the error reproduced in a browser. Documentation is in the README of the github project.

https://github.com/janpaul/saxonable

Feel free to clone and make changes, I'm happy to handle pull requests!

#17 - 2017-04-13 15:29 - Debbie Lockett

Ok thanks. I had made some progress to make a sample HTML page to run the test SEF and XML, but seeing exactly what you are doing is better.

Is there a particular reason that you are using a base64 encoded version of the SEF? I think this is where problems are arising. You are supplying the full string containing this base64 encoded SEF as the stylesheetLocation value in the SaxonJS.transform call. The stylesheetLocation is expected to be a string containing a relative path to the SEF (XML file). Somewhat surprisingly, what you are supplying does work for some browsers - which treat this string as a data URI, and the XMLHttpRequest does actually return the XML as the responseText. But, as previously noted data URIs do not work on IE/Edge.

If you can use an actual XML file instead (say testSEF.xml in the data folder), then in your SaxonJS.transform call you should be able to replace the option stylesheetLocation:sef with stylesheetLocation:"data/testSEF.xml". Then you should not need to do any of the external retrieval of the SEF which you are currently doing, in the doSomeSaxonJS function in app.js.

(Note to self: I would also like to look into the performance difference observed for sourceText versus sourceLocation, since supplying the sourceLocation seems simpler to me, but have not done this yet.)

#18 - 2017-04-14 09:41 - Jan Paul Stegeman

Hi Debbie,

I hear what you're saying. But our application doesn't work that way. The SEF that we are using is not a file in the filesystem that we can access, it's embedded in another file (a JSON, actually) that we supply to SaxonJS through the location mechanism. Agreed, this works in Chrome/Firefox/Safari. I'm not so sure about the "suprisingly" that you used, what we do **should** be completely valid in a modern browser. So I would say that "somewhat suprisingly it does NOT work on IE and Edge" ;-)

Performance-wise, I do believe that sourceText should be faster than using sourceLocation, since in the first case we simply inject a string into SaxonJS, whereas in the other case a file has to be read from the filesystem. Aside from other processing that you may do.

The sample repository I built for you is as close to our specific usecase as possible.

#19 - 2017-04-19 11:53 - Debbie Lockett

- Subject changed from SaxonJS throws an 'Access Denied' in the file SaxonJS.js when the method 'asyncGet' is called. to Supplying base64 encoded source and/or SEF to SaxonJS.transform

Ok, so rather than supplying the SEF as a data URI via the stylesheetLocation option of SaxonJS.transform, let's aim to supply the SEF as an XML DOM node, via the stylesheetNode option. Here's a solution to do that:

Change the processSaxon function to use the option 'stylesheetNode: sef' rather than 'stylesheetLocation: sef'. And within the doSomeSaxonJS function, call

```
processSaxon(parseXml(decodeUTF8(sef)), xml)
```

rather than

```
processSaxon(sef, xml)
```

Add the following functions to app.js:

```
function decodeUTF8(sef) {
    var index = sef.indexOf('base64,', 0) + 7;
    var octets =
        window.SaxonJS.U.Atomic.base64Binary.fromString(sef.substring(index)).value;
    var utf8 = "";
    for (var i = 0; i < octets.length; i++) {
        var c = octets.charCodeAt(i);
        if (c <= 127) {
            utf8 += String.fromCodePoint(c);
        } else {
            utf8 += '%' + c.toString(16);
        }
    }
    return decodeURIComponent(utf8);
}

function parseXml(xmlString) {
    return ( new window.DOMParser() ).parseFromString(xmlString, "application/xml");
}
```

The complicated part is in converting the base64 encoded version of the SEF to a suitable string version that can be supplied to DOMParser.parseFromSting(). This is especially complicated due to the fact that we use a non-ASCII character ("Greek capital letter Sigma") in the SEF checksum. There is some code within Saxon-JS which decodes base64 to a string of octets, which the above solution uses. We will look at making this simpler and more accessible in future releases.

#20 - 2017-05-11 17:44 - Debbie Lockett

- Status changed from In Progress to Resolved

Marking this as resolved.

I've added a new 'Feature' issue to track further developments - i.e. making the work around solution produced more accessible.

See https://saxonica.plan.io/issues/3218

#21 - 2017-08-10 12:54 - Debbie Lockett

- Tracker changed from Bug to Support

- Status changed from Resolved to Closed

- Applies to JS Branch 1.0 added

Files

ie11.png	90.5 KB	2017-03-21	Jan-Paul van der Velden
test.sef	103 KB	2017-04-10	Jan Paul Stegeman
test.xml	864 Bytes	2017-04-10	Jan Paul Stegeman