Saxon-JS - Feature #4961

use-when in SEF compilation phase

2021-04-02 15:17 - Martynas Jusevicius

Status:	New	**Start date:**	2021-04-02
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Command line	**Estimated time:**	0:00 hour
Sprint/Milestone:		**Spent time:**	0:00 hour
Applies to JS Branch:	2	**Fixed in JS Release:**	
Fix Committed on JS Branch:		**SEF Generated with:**	

Description

ac:uuid is my custom extension function which is obviously not available client-side. But using use-when as follows allowed me to differentiate between function calls on the server and in the browser:

```
<xsl:value-of use-when="system-property('xsl:product-name') eq 'Saxon-JS'" select="
resolve-uri(concat('/', ixsl:call(ixsl:window(), 'generateUUID', [])), $ac:uri)"/>
<xsl:value-of use-when="system-property('xsl:product-name') = 'SAXON'" select="
resolve-uri(concat('/', ac:uuid()), $ac:uri)"/>
```

That worked fine when compiling SEF with Saxon-JS. Now I'm trying Saxon-EE for that, and a different use-when kicks in and I get an error about the missing ac:uuid function:

```
Error near {...ve-uri(concat('/', ac:uuid(...} at char 24 in xsl:value-of/@select on line 460 colu
mn 175 of default.xsl:
  XPST0017  Cannot find a 0-argument function named Q{https://w3id.org/atomgraph/client#}uuid()
Error near {...ac:forClass, $ldt:base) els...} at char 24 in xsl:param/@select on line 230 column
161 of resource.xsl:
  XPST0017  Cannot find a 3-argument function named
  Q{https://w3id.org/atomgraph/client#}construct-doc()
```

Is there a workaround here? Can some property be used to differentiate between compilation and execution phases in use-when?

History

#1 - 2021-04-04 11:36 - Martynas Jusevicius

To clarify: ac:uuid is registered as an extension function in Java, that is why Saxon-EE cannot find it.

#2 - 2021-04-05 13:20 - Martynas Jusevicius

Worked around this by adding function stubs as suggested by Michael on xml.com Slack:

```
    <xsl:function name="ac:construct-doc" as="document-node()*" override-extension-function="no">
        <xsl:message use-when="system-property('xsl:product-name') = 'SAXON'" terminate="yes">
            Not implemented -- com.atomgraph.client.writer.function.ConstructDocument needs to be registered a
s an extension function
        </xsl:message>
    </xsl:function>
```

Now the stylesheet SEF compiles with both Saxon-JS 2.1 and Saxon-EE 10.3

#3 - 2021-04-05 15:19 - Martynas Jusevicius

Now I have a problem with execution on Saxon-HE however. Since I've removed use-when, IXSL functions ixsl:call/ixsl:eval/ixsl:window are not recognized:

```
Error near {...'limit', [ $limit ]), 'offs...} at char 20 in xsl:sequence/@select on line 180 column 355 of de
fault.xsl:
  XPST0017  Cannot find a 3-argument function named Q{http://saxonica.com/ns/interactiveXSLT}call()
Error near {...'limit', [ $limit ]), 'offs...} at char 10 in xsl:sequence/@select on line 183 column 123 of de
fault.xsl:
  XPST0017  Cannot find a 3-argument function named Q{http://saxonica.com/ns/interactiveXSLT}call()
Error near {...ing($js-statement/@statemen...} at char 0 in xsl:sequence/@select on line 225 column 77 of defa
```

```
ult.xsl:
  XPST0017  Cannot find a 1-argument function named Q{http://saxonica.com/ns/interactiveXSLT}eval()
Error near {...ing($js-statement/@statemen...} at char 0 in xsl:sequence/@select on line 244 column 77 of defa
ult.xsl:
  XPST0017  Cannot find a 1-argument function named Q{http://saxonica.com/ns/interactiveXSLT}eval()
Error near {...ing($js-statement/@statemen...} at char 0 in xsl:sequence/@select on line 276 column 85 of defa
ult.xsl:
  XPST0017  Cannot find a 1-argument function named Q{http://saxonica.com/ns/interactiveXSLT}eval()
Error near {...ing($js-statement/@statemen...} at char 0 in xsl:sequence/@select on line 282 column 85 of defa
ult.xsl:
  XPST0017  Cannot find a 1-argument function named Q{http://saxonica.com/ns/interactiveXSLT}eval()
Error near {...ing($js-statement/@statemen...} at char 0 in xsl:sequence/@select on line 288 column 85 of defa
ult.xsl:
  XPST0017  Cannot find a 1-argument function named Q{http://saxonica.com/ns/interactiveXSLT}eval()
Error in {ixsl:call(ixsl:window(), 'aler...} at char 10 in xsl:value-of/@select on line 195 column 89 of sparq
l.xsl:
  XPST0017  Cannot find a 0-argument function named Q{http://saxonica.com/ns/interactiveXSLT}window()
15:05:08,546 [http-nio-8080-exec-1] ERROR Application:618 - System XSLT stylesheet error
net.sf.saxon.s9api.SaxonApiException: Errors were reported during stylesheet compilation
```

Now I'm lost again.

#4 - 2021-04-07 16:35 - Martynas Jusevicius

I think a property like saxon:phase which would return Compilation/Execution would address my use-when use case.

#5 - 2021-04-07 17:28 - Michael Kay

It's a bit more tricky than that. We have to evaluate xsl:use-when at compile time, and there's a question of how much we know at that stage about the target environment for execution. We can certainly add a system property such as saxon:target which tells you what the compilation target environment is. But we don't know (and don't want to know) the version of the target environment, and we don't know what extension functions are available in that environment -- we only know what stubs are available in the compile time environment. So this all requires some rather careful thought.

But apart from xsl:use-when (and other static expressions) there's also the question of calls to system-property(), function-available(), etc appearing in dynamic expressions. Currently I suspect the compiler will evaluate these eagerly, assuming that the answer is the same for the execution environment as for the compiler environment. If that's the case, it doesn't seem the right thing to do.

#7 - 2021-04-27 17:21 - Michael Kay

Perhaps we want rules something like the following:

(a) system-property() and XX-available() (whether called statically or dynamically) are intended to return properties of the target (run-time) environment.

(b) the target (run-time) environment is presumed to be the same as the compile-time environment unless otherwise specified.

(c) one way it can be "specified otherwise" is by use of -target on the command line. (But does this give us enough information, e.g. do we need to know whether it's JS-in-the-browser vs JS-on-node? And what do we do about xsl:product-version?)

(d) perhaps we want other ways of "specifying otherwise" e.g. a declaration in the stylesheet itself?

In static expressions we have to evaluate the function at compile time, and therefore with incomplete knowledge of the run-time environment.

In dynamic expressions we should behave as if we evaluated the function dynamically; we should only do eager/early evaluation if we are sure we know the answer (e.g. if not generating a SEF).

#8 - 2021-04-28 12:50 - Martynas Jusevicius

(c) also relates to https://saxonica.plan.io/issues/4962, I think