Saxon/C - Support #4864

Missing documentation for running Integrated Functions in Java with Saxon-C

2020-12-23 15:56 - sébastien bocahu

Status:	AwaitingInfo	**Start date:**	2020-12-23
Priority:	Low	**Due date:**	
Assignee:	O'Neil Delpratt	**% Done:**	0%
Category:		**Estimated time:**	0:00 hour
Sprint/Milestone:		**Spent time:**	0:00 hour
Found in version:	1.2.1		

Description

Hi,

I'm experimenting writting custom extensions in Java (Java "full API" for integrated functions) I could figure out how to run it from a Java testing app and within Oxygen XML Editor. However, I can't find a way to make it work with Saxon-C called from a PHP script.

I tried to put a jar in saxonpath/lib/ext or saxonpath/lib (where jar files are already present) with no luck.

Any help would be much appreciated.

thanks !

History

#1 - 2020-12-23 23:31 - O'Neil Delpratt

Hi,

It is not possible to integrate a user defined function written in Java into Saxon/C. However it is possible to write your user defined functions in PHP. Please see the following link for more details: https://www.saxonica.com/saxon-c/documentation/index.html#!extensibility

#2 - 2020-12-24 12:11 - sébastien bocahu

Hi,

OK...

I tried that PHP externsion mecanism but it keep to throw that error:

```
  XPST0017: Cannot find a 2-argument function named {http://php.net/xsl}function()
```

Same error with the demo from xsltExamples.php (after commenting it out...)

```
<?php
include('ParseFareConstruction.php'); // function blah()
$saxonProc = new Saxon\SaxonProcessor();
$xsltProc = $saxonProc->newXsltProcessor();
$xsltProc->setProperty("extc", ini_get('extension_dir') . "/saxon");
$saxonProc->registerPHPFunctions(ini_get('extension_dir') . "/saxon");

$xslFile = "test.xsl";
$xmlFile = "input.xml";

$xsltProc->compileFromFile($xslFile);
$xsltProc->setSourceFromFile($xmlFile);
$result = $xsltProc->transformToString();

echo $result;
```

Thanks

#3 - 2020-12-24 12:51 - O'Neil Delpratt

Which Saxon/C product are you using? The extension function feature is only available in the Saxon-PE/C and Saxon-EE/C editions

#4 - 2020-12-24 13:51 - sébastien bocahu

I was using Saxon-C HE.

I just installed PE and requested an evaluation license. (

However, version() still displays "Saxon/C 1.2.1 running with Saxon-HE 9.9.1.5C from Saxonica" and stracing the php cli test shows it does not try to access the license key (but confirms it loads libsaxonpec.so). I even tried

```
$saxonProc->setConfigurationProperty("http://saxon.sf.net/feature/licenseFileLocation", "
/usr/lib/saxon-license.lic");
```

to no avail.

What am I missing ?

#5 - 2020-12-24 19:18 - O'Neil Delpratt

It looks like you are running in a HE configuration. due to the zero argument SaxonProcessor constructor.

Try the following

```
$saxonProc = new Saxon\SaxonProcessor(true);
......
```

#6 - 2020-12-28 13:16 - sébastien bocahu

Thanks, I could get it to work. Could you please explain how to return an XML node from a Java function ?

(I'm trying to create a wrapper to call PHP functions, in that case we'll use paied license for being able to use functions in Saxon-C PE or EE)

```
    @Override
    public SequenceType getResultType(SequenceType[] suppliedArgumentTypes) {
        return SequenceType.SINGLE_NODE;
    }
```

```
    @Override
    public ExtensionFunctionCall makeCallExpression() {
        return new ExtensionFunctionCall() {
            @Override
            public Sequence call(XPathContext context, Sequence[] arguments) throws XPathException {
            String xmlFromWrapper = "<elems></elemens>";
            ...
            return node;
            }
        }
   }
```

I could parse the XML with proc.newDocumentBuilder().build() but I can't find a way to return a XdmNode into a Sequence...

#7 - 2020-12-28 13:46 - O'Neil Delpratt

Hi,

It is not possible to mix Java function with a PHP function in Saxon/C, but you can port what you are doing to a pure PHP function to work within the Saxon/C PHP extension.

The following example shows how to return a xs:string from a PHP function in the XSLT stylesheet:
https://www.saxonica.com/saxon-c/documentation/index.html#!extensibility/extensions_php

It should possible to return some XML node instead. Please can you supply me all the pieces to what you are trying to do?

From my assumptions on what you are doing, I think the PHP function should look like the following:

```
// define an extension function
function userFunction()
{
    $proc = new Saxon\SaxonProcessor();
    $node = $proc->parseXmlFromString("<elems></elemens>");
```

```
     .......
     // not sure what should happen here?
    return $node;
  }
}
```

It would be good to see your XSLT, but again I am assuming the following:

```
<xsl:variable name="args" select="[]"/>
<xsl:variable name="nodeVar" select="php:function('userFunction', $args)"/>
.....
```

Finally for the main PHP script you just connect all the dots as it the link above.

You will find it useful the PHP API:
https://www.saxonica.com/saxon-c/documentation/index.html#!api/saxon_c_php_api/saxon_c_php_saxonprocessor

#8 - 2020-12-28 14:09 - sébastien bocahu

In PHP: I get an error when using the user function:

```
Saxon/C 1.2.1 running with Saxon-PE 9.9.1.5C from SaxonicaStatic error at char 38 in xsl:value-of/@select on l
ine 10 column 110 of test.xsl:
  XPTY0004: The required item type of the second argument of fn:function() is function(*);
  supplied expression ($args) has item type xs:string
```

with XSL:

```
        <xsl:variable name="args" select="string(.)"/>
        <xsl:value-of select="php:function('ParseText', $args)/FareComponent/Airports/Airport" />
```

and a PHP userFunction:

```
function ParseFareConstruction($text)
{
  return "<elem>text</elem>";
}
```

In Java:

I am trying to wrap the PHP user function call by a PHP CLI exec, parse the output as an XML, and return it as XML node. This is for use as testing purpose in the OxygenXML editor UI.

```
    @Override
    public ExtensionFunctionCall makeCallExpression() {
        return new ExtensionFunctionCall() {
            @Override
            public Sequence call(XPathContext context, Sequence[] arguments) throws XPathException {
              NodeInfo ret = null;
              try {
                String v0 = arguments[0].head().toString();
                v0 = v0.substring(1, v0.length()-1);
                String v1 = arguments[1].head().toString();
                v1 = v1.substring(1, v1.length()-1);
                String result = execPHP(v0, v1);
                System.out.println(result);
                //proc.newDocumentBuilder().build(result);
                Processor proc = new Processor(context.getConfiguration());
                StreamSource s = new StreamSource(new StringReader(result));
                s.setSystemId("dontcare");
                XdmNode node = proc.newDocumentBuilder().build(s);
                ret = (NodeInfo) node;
              } catch (SaxonApiException e) {
                System.out.println(e.getMessage());
              }
              return ret;
            }
        };
    }
```

```
public String execPHP(String scriptName, String param) {
    StringBuilder output = new StringBuilder();
    try {
      String line;
      String cmdline = "php -r 'require \"ParseText.php\"; echo " + scriptName + "(\"" + param + "\");'";
      System.out.println(cmdline);
      Process p = Runtime.getRuntime().exec(new String[] { "bash", "-c", cmdline });
      BufferedReader input =
        new BufferedReader
```

```
          (new InputStreamReader(p.getInputStream()));
      while ((line = input.readLine()) != null) {
        System.out.println(line);
          output.append(line);
      }
      input.close();
    }
    catch (Exception err) {
      err.printStackTrace();
    }
    return output.toString();
  }
```

#9 - 2020-12-28 14:40 - O'Neil Delpratt

Thanks for sending your example code.

For the PHP function you are returning a string value instead of an Xdm node.

I would think the following should work:

```
function ParseFareConstruction($text)
{
    $proc = new Saxon\SaxonProcessor();
    $node = $proc->parseXmlFromString("<elem>".$text."</elem>");
    return $node;
}
```

When using the PHP function within your XSLT code the arguments should be specified as an XDM array. For example:

```
<xsl:variable name="args" select="['text']"/>
<xsl:value-of select="php:function('ParseFareConstruction', $args)/FareComponent/Airports/Airport"/>
```

For the above code there seems to be something missing in terms of the XML you are constructing. I hope I am calling the right function in the XSLT code.

I will investigate your issue shortly to try and reproduce the error and provide a solution, but as we are still in holiday season there will be some delay.

#10 - 2020-12-30 11:48 - sébastien bocahu

Thanks, I could achieve what I wanted.

However I experienced weird segmentation faults / jetvm crash that seem to be related to ("either" or "and", I am not sure):

- script being executed from a folder in my home directory (/home/USER/something/script.php whereas same script in /tmp/something/script.php or /opt/something/script.php works like a charm)
- function name (ParseFareConstruction fails, ParseText works)

I'll need some more time to experiment more and give something to reproduce

#11 - 2020-12-30 13:38 - O'Neil Delpratt

- Status changed from New to AwaitingInfo

- Assignee set to O'Neil Delpratt

- Found in version set to 1.2.1

Hi, Glad that it worked and thanks for the feedback. A segmentation faults / jetvm crash is probably a bug. I will look out for a repo from you so that we can investigate further. I suggest creating another bug issue for this problem.