

Open APIs
for Open
Minds

All the Cool Kids are using JavaScript

Jason Fox
FIWARE Foundation

Splash Screen Plug-In for DITA-OT

● JavaScript 100.0%

Displays a Random Cat/XKCD Comic whilst compiling DITA

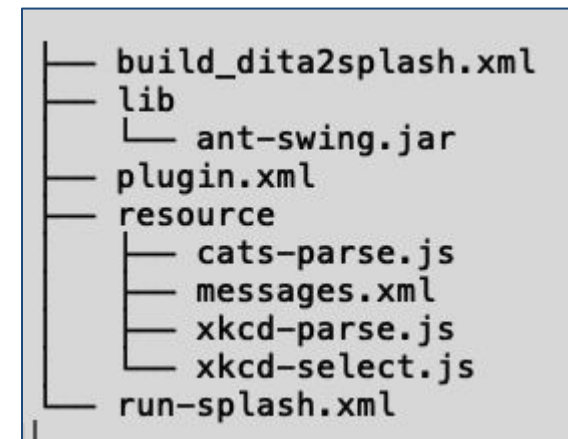
XKCD.com API: <https://xkcd.com/json.html>

CatsAPI <http://thecatapi.com/docs.html>

- **Extension:** depend.preprocess.pre
- **JavaScript:** Simple String Manipulation
Random Number Generation

```
var input = project.getProperty("xkcd.latest");
var pos = input.indexOf('"num": ');
var len = input.indexOf(",", pos);

project.setProperty(
    "num",
    Math.floor(Math.random() * input.substring(pos + 6, len) + 1)
);
```



Splash Screen Plug-In for DITA-OT

● JavaScript 100.0%

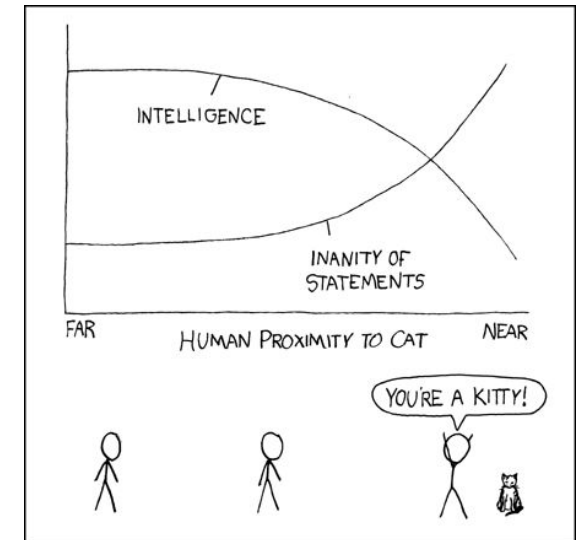
- **Invoke:** Using `<script>` element

```
<script language="javascript" src="${dita.plugin.fox.jason.splash.dir}/resource/xkcd-select.js"/>
```

- Use `<java>` task with Ant Launcher to amend the classpath

```
<java classname="org.apache.tools.ant.launch.Launcher" classpathref="
splash.classpath" fork="true" spawn="true">
  <arg value="-buildfile"/>
  <arg value="${dita.plugin.fox.jason.splash.dir}/run-splash.xml"/>
  <arg value="-Dsplash.image=${splash.image.url}"/>
  <arg value="-Dsplash.title=${splash.title}"/>
  <arg value="-Dsplash.duration=20000"/>
</java>
```

- **GitHub:** fox.jason.splash



ReadTheDocs Plug-In for DITA-OT

● JavaScript 100.0%

Extends the Markdown Plugin to create .yaml file suitable for ReadTheDocs

- **Transform:** build-init plus Markdown plus post-process

```
<target name="readthedocs">
  <create-readthedocs dir="${dita.output.dir}" file="index.md"/>
</target>

<target depends="readthedocs.init, dita2markdown, readthedocs, readthedocs.css"
  name="dita2readthedocs"/>
```

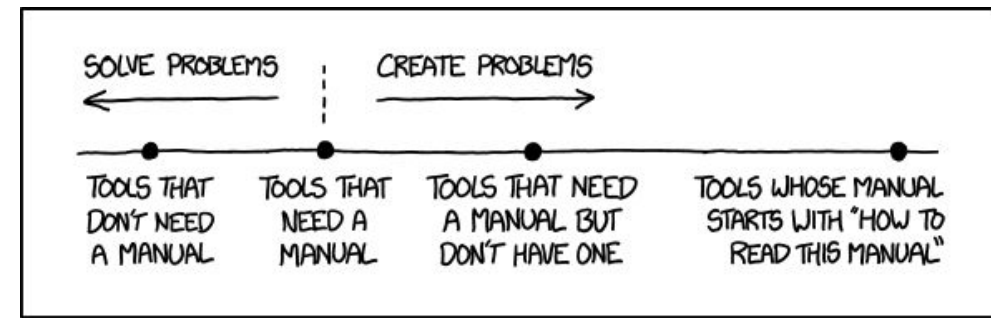
```
├─ build_dita2readthedocs.xml
├─ plugin.xml
├─ resource
│   └─ antlib.xml
│       └─ create-ditamap.js
│           └─ create-readthedocs.js
```

- Use an Init Target to includes AntLib

```
<typedef file="${dita.plugin.fox.jason.readthedocs.dir}/resource/antlib.xml"/>
```

- Define Macro using **<scriptdef>** element

```
<scriptdef language="javascript" name="create-readthedocs" src="
  ${dita.plugin.fox.jason.readthedocs.dir}/resource/create-readthedocs.js">
  <attribute name="file"/>
  <attribute name="dir"/>
</scriptdef>
```



ReadTheDocs Plug-In for DITA-OT

● JavaScript 100.0%

- **JavaScript** : File Manipulation

```
var file = attributes.get("file");
var dir = attributes.get("dir");

function rewriteAsYAML(filename) {

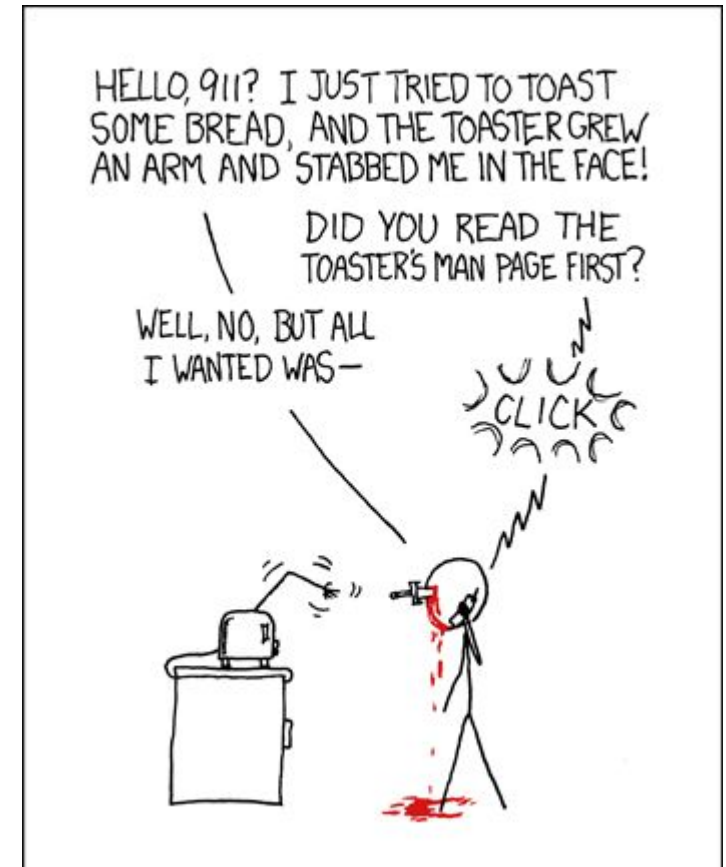
    var indexFile = org.apache.tools.ant.util.FileUtils.readFully(
        new java.io.FileReader(dir + '/' + filename));

    ...etc

    var task = project.createTask('echo');
    task.setFile(new java.io.File(dir + '/mkdocs.yml'));
    task.setMessage(output.join('\n'));
    task.setForce(true);
    task.perform();
}

rewriteAsYAML(file);
```

- **GitHub**: [fox.jason.readthedocs](https://github.com/fox-jason/readthedocs)



DITA Prettier Plug-In for DITA-OT

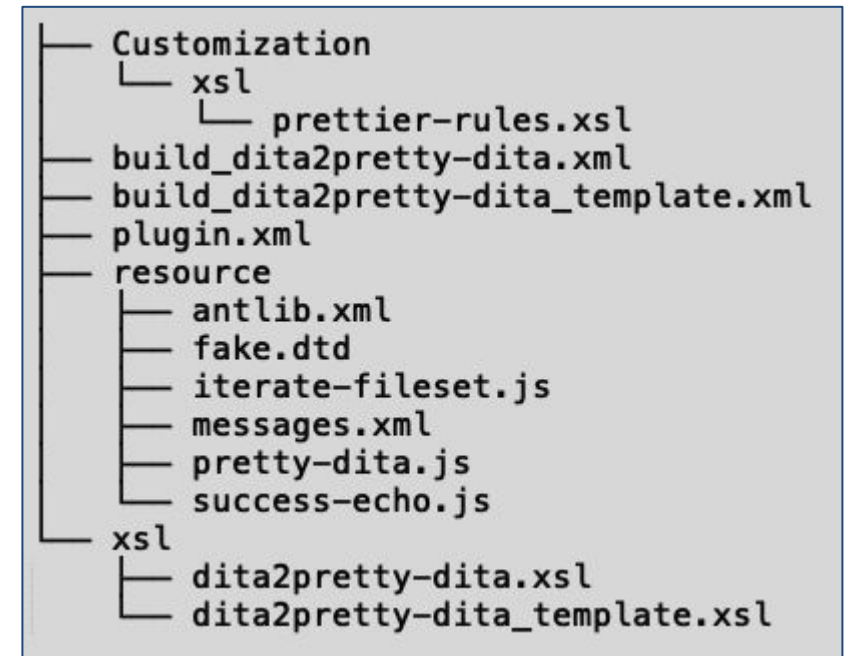
● JavaScript 51.1%

● XSLT 48.9%

Tidies Whitespace and standardizes line length of DITA files

- **Transform:** Full custom XSL Transform
- When processing raw DITA use a fake DTD

```
<xslt unless:set="prettier.ignore" force="true"
  in="@{dir}/@{src}" out="{copy.file}" style="
  ${dita.plugin.fox.jason.pretty-dita.dir}/xsl/dita2pretty-dita.xsl">
  <xmlcatalog refid="fake.dita.catalog"/>
  <param if="args.indent" expression="{args.indent}"
    name="INDENT" type="INT"/>
  <param if="args.style" expression="{args.style}"
    name="STYLE"/>
</xslt>
<tidy-whitespace unless:set="prettier.ignore" file="{copy.file}"
  pragma="{prettier.hasPragma}"/>
```

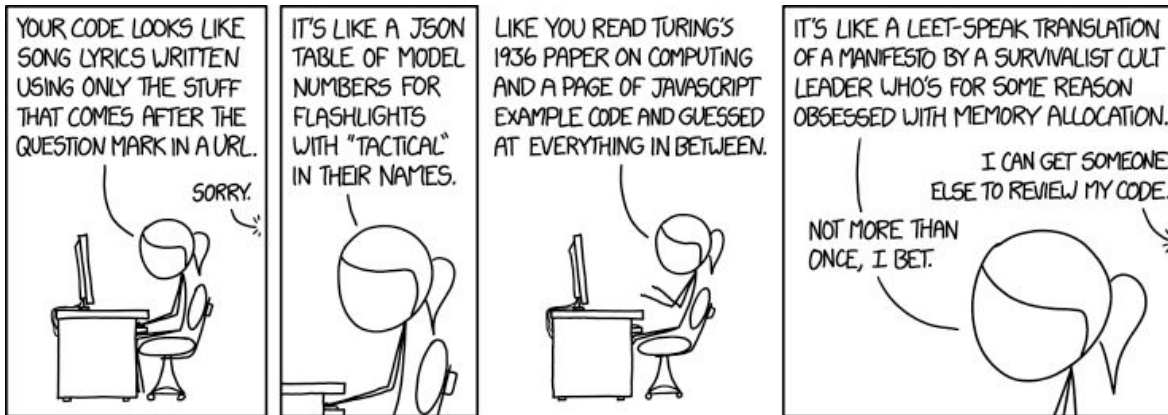


DITA Prettier Plug-In for DITA-OT

● JavaScript 51.1%

● XSLT 48.9%

- **JavaScript:**
 - String Manipulation
 - File Manipulation
 - File Iteration
- Macros can call Macros



- **GitHub:** [fox.jason.pretty-dita](https://github.com/fox.jason.pretty-dita)

```
var dir = attributes.get("dir");
var toDir = attributes.get("todir");
var filesets = elements.get("fileset");
var macro = attributes.get("macro");

for (var i = 0; i < filesets.size(); ++i) {
  var fileset = filesets.get(i);
  var scanner = fileset.getDirectoryScanner(project);
  scanner.scan();
  var files = scanner.getIncludedFiles();
  for (var j = 0; j < files.length; j++) {
    var task = project.createTask(macro);
    if (files[i] !== "") {
      try {
        task.setDynamicAttribute("src", files[j]);
        task.setDynamicAttribute("dir", dir);
        task.setDynamicAttribute("toDir", toDir);
        task.execute();
      } catch (err) {
        task.log("Execution error: " + err.message);
      }
    }
  }
}
```

Code Highlighting Plug-Ins for DITA-OT

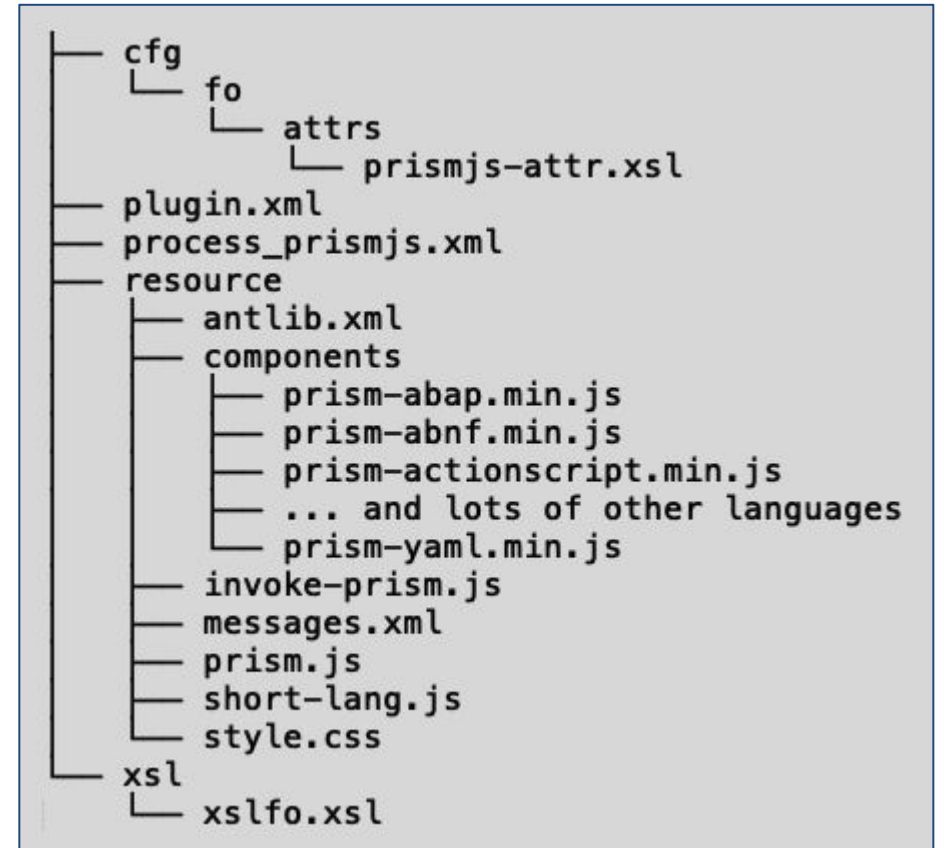
● XSLT 64.2%

● JavaScript 19.4%

● CSS 16.4%

An integration of **Prism-JS JavaScript Library** into the DITA Open Toolkit engine, enabling static HTML and PDF syntax highlighting.

- **Extensions:**
 - depend.preprocess.post
 - extend.css.process to inject CSS
- Use **<XMLTask>** to manipulate DITA
- **GitHub:**
 - fox.jason.prismjs
 - fox.jason.prismjs.dark-theme



Code Highlighting Plug-Ins for DITA-OT

● XSLT 64.2%

● JavaScript 19.4%

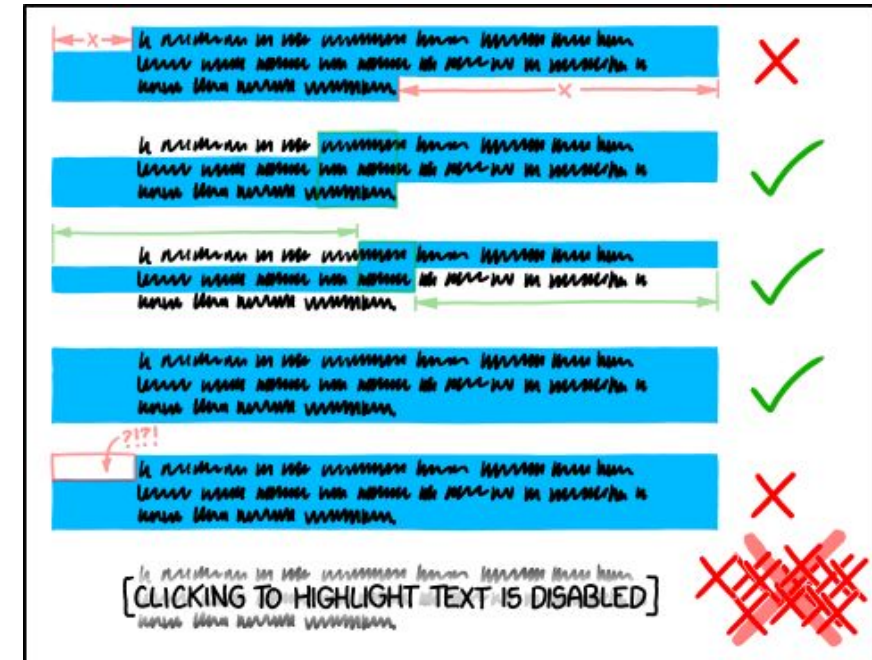
● CSS 16.4%

JavaScript:

- String Manipulation
- Dynamic Library Loading

```
eval(  
    "" +  
    org.apache.tools.ant.util.FileUtils.readFully(  
        new java.io.FileReader(project.getProperty("prismjs.temp.file"))  
    )  
);
```

Dynamic loading is slow, avoid if unnecessary.
Can update the list of loaded languages on the fly



Thank you!

<http://fiware.org>

Follow @FIWARE on Twitter

