AsciiDoc & doc-as-code Best Practices

Version 1.18-SNAPSHOT

2018-09-30

Table of Contents

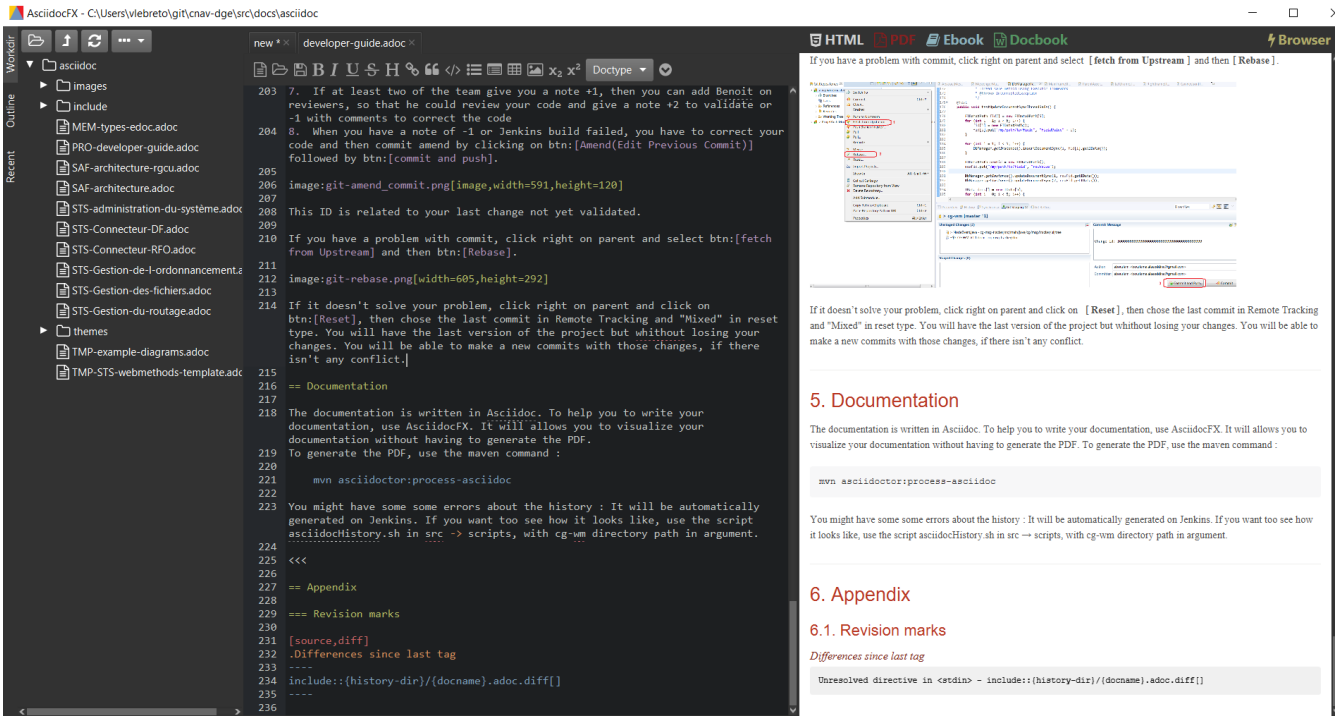
Table 1. History

Date	Author	Detail
2018-09-19	bcouetil	- Sample asciidoctor maven project published on Github - Github & LinkedIn links - Sample project tree - new images + resizing and positioning
2018-08-29	bcouetil	Asciidoc HTML look & feel changes
2018-08-24	bcouetil	Icones added for download + favicon added for webpage
2018-08-23	bcouetil	Initial commit

1. Documentation-as-code with AsciiDoc

All documentations on the project, including this one, are written as part of DevOps principle Documentation-As-Code using AsciiDocFX and compiled with Asciidoctor under maven to produce PDF.

To help you to write your documentation, use AsciidocFX. It will allows you to visualize your documentation without having to generate the PDF. You can consider Visual Studio Code too, it has a nice plugin but with less features.



Download and install :

- AsciidocFX, to create and edit your documents
- Graphviz, to add the extension for your AsciidocFX installation

Rules to produce AsciiDoc files :

- file names are in kebab-case, except the type at the beginning
- use **kbd:[myShortcut]** for keyboard shortcuts
- use **\btn[myButton]** for buttons
- High level title should be on new pages (use <<<)
- don't forget init/closure includes at beginning/end of the adoc
- For STS, use TMP-STS-webmethods-template.adoc as a template

To show a in progress diagram to colleagues, you can generate a web link using PlantUml generator.

1.1. Inline icons

- See https://asciidoctor.org/docs/user-manual/#icons
- Font Awesome 4.6.3 Class Explorer : https://lab.artlung.com/font-awesome-sample/
  - no need for :icons: font to use font awesome-font for icons

1.2. Custom highlighter

- See the maven configuration below, it speaks for itself
- Highlight.js demo with all languages and all styles
- Default highlight.js processor is too basic, instructions here to change : https://asciidoctor.org/docs/user-manual/#highlight-js

1.3. Generation using Maven

You will find a sample Maven project generating both PDF and HTML, with current doc's layout, on my Github.

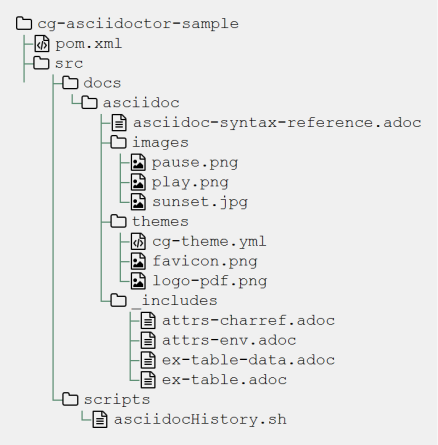


Figure 1. File system tree

AsciiDoctor maven plugin configuration

```
<!-- to generate asciidoc PDF + HTML documents -->
<!-- single usage : mvn generate-resources -Dadoc.skip=false [- -non-recursive] -->
<plugin>
    <groupId>org.asciidoctor</groupId>
    <artifactId>asciidoctor-maven-plugin</artifactId>
    <version>1.5.6</version>
    <dependencies>
        <dependency>
            <groupId>org.asciidoctor</groupId>
            <artifactId>asciidoctorj-pdf</artifactId>
            <!-- alpha 15+16 have an annoying message on multiple lines : -->
            <!-- warning: already initialized constant XXXX -->
            <version>1.5.0-alpha.14</version>
        </dependency>
        <dependency>
            <groupId>org.asciidoctor</groupId>
            <artifactId>asciidoctorj-diagram</artifactId>
            <version>1.5.9</version>
        </dependency>
    </dependencies>
```

```
    <configuration>
        <skip>${adoc.skip}</skip>
        <sourceDirectory>src/docs/asciidoc</sourceDirectory>
        <requires>
            <require>asciidoctor-diagram</require>
        </requires>
        <attributes>
            <icons>font</icons>
            <idseparator>-</idseparator>
            <idprefix />
            <!-- custom -->
            <source-dir>../../main/java</source-dir>
            <test-dir>../../test/java</test-dir>
            <project-version>${project.version}</project-version>
            <root-project-dir>${user.dir}</root-project-dir>
            <history-dir>${project.build.directory}/generated-docs/history</history-dir>
            <project-images-dir>${project.basedir}/src/main/resources/images</project-images-dir>
        </attributes>
    </configuration>
    <executions>
        <execution>
            <id>asciidoc-to-html</id>
            <phase>generate-resources</phase>
            <goals>
                <goal>process-asciidoc</goal>
            </goals>
            <configuration>
                <backend>html5</backend>
                <!-- coderay, highlight.js, prettify -->
                <!-- coderay/prettify have only a bright theme -->
                <!-- prettify is nice but comments are red -->
                <!-- pygments/rouge is not integrated yet : https://github.com/asciidoctor/asciidoctor/issues/1040 -->
                <sourceHighlighter>highlight.js</sourceHighlighter>
                <attributes>
                    <!-- local version to go beyond basic languages (for ex groovy) -->
                    <highlightjsdir>highlight</highlightjsdir>
                    <!-- explore here https://highlightjs.org/static/demo/ -->
                    <highlightjs-theme>gruvbox-dark</highlightjs-theme>
                    <imagesdir>./images</imagesdir>
                    <toclevels>5</toclevels>
                    <sectanchors>true</sectanchors>
                    <docinfo1>true</docinfo1>
                    <favicon>themes/favicon.png</favicon>
                    <!-- custom style if default is not suitable -->
                    <!-- <stylesheet>rubygems.css</stylesheet> -->
                    <!-- <stylesdir>themes</stylesdir> -->
                </attributes>
            </configuration>
        </execution>
        <execution>
            <id>asciidoc-to-revealjs</id>
            <!-- after html because it's also an html -->
            <phase>generate-resources</phase>
            <goals>
                <goal>process-asciidoc</goal>
            </goals>
            <configuration>
                <backend>revealjs</backend>
                <sourceDocumentName>PRES-asciidoc.adoc</sourceDocumentName>
                        <outputDirectory>${project.slides.directory}</outputDirectory>
                        <templateDir>${project.build.directory}/asciidoctor-reveal.js-${asciidoctor-
revealjs.version}/templates/slim</templateDir>
                        <sourceHighlighter>coderay</sourceHighlighter>
                        <attributes>
                            <revealjsdir>reveal.js-${revealjs.version}</revealjsdir>
                            <revealjs_theme>black</revealjs_theme>
                            <revealjs_transition>linear</revealjs_transition>
                        </attributes>
                    </configuration>
        </execution>
```

```
        <execution>
            <id>asciidoc-to-pdf</id>
            <phase>generate-resources</phase>
            <goals>
                <goal>process-asciidoc</goal>
            </goals>
            <configuration>
                <backend>pdf</backend>
                <sourceHighlighter>rouge</sourceHighlighter>
                <attributes>
                    <imagesdir>${project.build.directory}/generated-docs/images</imagesdir>
                    <toc />
                    <toclevels>3</toclevels>
                    <pdf-style>${user.dir}/src/docs/asciidoc/themes/cg-theme.yml</pdf-style>
                    <pagenums />
                </attributes>
            </configuration>
        </execution>
    </executions>
</plugin>
```

To generate the PDF locally, use the maven command :

```
mvn generate-resources --non-recursive
```

1.4. .doc to .adoc

If the initial documentation is of Microsoft Word, you have to first migrate it to AsciiDoc file.

Download and install pandoc.

To migrate a document, use it in command line

```
pandoc --from=docx --to=asciidoc --wrap=none --atx-headers --normalize --extract-media=images monDoc.docx > monDoc.adoc
```

1.5. Publish HTML & PDF to github

1.5.1. Initialize Github space

```
mkdir docs
cd docs
git init
git checkout --orphan gh-pages
```

Copy a first version of the site in the directory, then :

```
git add *
git commit -m "initial site content"
git remote add origin "https://github.com/NeVraX182/docs.git"
git push --set-upstream origin gh-pages
```

1.5.2. Configure Maven plugin

Maven SCM plugin configuration

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-scm-publish-plugin</artifactId>
    <version>3.0.0</version>
    <inherited>false</inherited>
    <configuration>
        <scmBranch>gh-pages</scmBranch>
        <!-- token generated from github > settings > Developer settings > Personal access tokens > public_repo -->
        <pubScmUrl>scm:git:https://USER:TOKEN@github.com/USER/docs.git</pubScmUrl>
        <content>target/generated-docs</content>
    </configuration>
</plugin>
```

1.5.3. Use in pipelines

To use in Jenkins Pipelines, see Jenkins Best Practices

To use in Gitlab Pipelines, see Gitlab Best Practices

2. Appendix

2.1. Revision marks

Differences since last tag