Papyrus Shared Components Descriptor

version 0.0.18

User Guide

Eclipse **2016-10-05**

Table of Contents

1. Table of Contents i
2. Introduction .....
3. User 1
4. SOP Project Component Creation .....
5. SOP Component Release 2
6. SOP Add Missing License 6
7. SOP Target Platform Creation 7
8. FAQ 8

1 User

1.1 Download

The Papyrus Components guide could be downloaded as a pdf here

1.2 Papyrus Components

1.2.1 Context

The Papyrus Components project aims at providing a shared developer environment to help to integrate additional Papyrus Component.

1.3 Standard Operating Procedure

- SOP Project Component Creation
- SOP Target Platform Update
- SOP Release of a component
- SOP Add Missing License to java files
- SOP Add a Gerrit Contributer

2 SOP Component Release

2.1 sop-release

2.1.1 Who is in charge of?

Commiter + a basic developer in order to transfer knowledge and improve the process.

2.1.2 When?

when you need it, on a regular basis, every two month.

2.1.3 How long?

It last at least 120mn

2.2 Prerequisite

2.2.1 Access

Have the Hudson access to the Component tab and a committer with you to make the review and merge.

2.2.2 Green

No critical bugs open for the release you are targeting.

Jobs (master,website,quality,deploy) are green

2.2.3 Version

Initial version of your pom.xml, MANIFEST.MF and category.xml should match qualifier and -SNAPSHOT

2.2.4 Change log

You must fill the changelog to describe the new version feature.

```
/org.eclipse.papyrus-sysml/src/changes/changes.xml
```

Details https://maven.apache.org/plugins/maven-changes-plugin/changes.html

2.3 Pre-Actions

You can send an email to the developers mailing list.

2.4 Steps

2.4.1 Create a bugzilla ticket.

Bug XXXXXX - [YourComponent][release][YourTargetVersion] Release

2.4.2 Upgrade your target platform

Your initial platforms are perhaps using a version setted to 0.0.0. To release your project we advice you to replace this 0.0.0 by the version of the day.

Execute the following command on the target platform you are matching:

```
mvn org.eclipse.tycho.extras:tycho-version-bump-plugin:update-target -Dtarget=/home
```

It should replace all 0.0.0 version by a strick identified version of all needed features.

2.4.3 Upgrade the version of application - version 1

Remove the .qualifier, -SNAPSHOT extension in the different pom.xml, manifest.mf artifact version.

Example of command to upgrade pom.xml, plugin.xml, manifest.mf, category.xml, feature.xml etc..., check if it runs on all plugins depending on your profile configuration

```
mvn org.eclipse.tycho:tycho-versions-plugin:set-version -DnewVersion=1.2.2.qualifie
```

2.4.4 Upgrade the version of application - version 2

Go to the root pom.xml

Update the version (removing the -SNAPSHOT if necessary) of the target.version

Go to the targetPlatform folder

Launch the following maven command using the new version as X.Y.Z (and removing the -SNAPSHOT if necessary):

Use tycho-versions plugin to switch from qualifier to release, and then back from release to qualifier

```
mvn org.eclipse.tycho:tycho-versions-plugin:set-version -DnewVersion=X.Y.Z-SNAPSHOT
```

Then verify the build locally using the following command

```
mvn clean install -Dtycho.localArtifacts=ignore
```

Update the plugin versions by finding and replacing manually the former X.Y.Z-SNAPSHOT and X.Y.Z.qualifier to the new X.Y.Z version. A command can be used, such as at the root:

```
find . -type f -name "file" -exec sed -i 's/X.Y.Z-SNAPSHOT/X.Y.Z/g' {} +
```

check the different pom.xml, feature.xml, category.xml and MANIFEST.MF. Those can be checked either manually or using the following command:

```
find . -name "file to test" | xargs grep -n -e "qualifier" -e "SNAPSHOT"
```

Push on gerrit the different modifications

Check the status of the gerrit job

Add a reviewer

Review and merge the change

2.4.5 Rexecute the job Master and eventually job Website

Who: any

Goal is to use this job version as data for the promotion.

2.4.6 Make the release

Who: commiter action

Go the hudson Component tab

Execute the job papyrus-component-deploy-eclipse

Fill the args

It should tag the release automatically, you copuld check it by looking at the git repository

It should also automatically execute the job papyrus-component-deploy-nexus: to deploy the artifact into the Eclipse Papyrus official Nexus

It should also automatically execute the job and papyrus-component-deploy-website

2.4.7 Deploy the web site

Who: uncommiter action + commiter review

The new web site is available under the target/site-staging directory

Download it as a zip

Unzip it in the papyrus web git repository, under the papyrus/components/YourComponent directort

Update the root index.html with the new version of your component

```
https://git.eclipse.org/c/www.eclipse.org/papyrus-sysml.git/
```

Ask for validation for the review.

Once it is merged, you should see it 3mn here http://www.eclipse.org/papyrus/components/MyComponent/YourTargetVersion/

2.4.8 Upgrade to the next snapshot version of the application

use tycho-versions plugin, use the good profile to execute it

```
mvn org.eclipse.tycho:tycho-versions-plugin:set-version -DnewVersion=X.Y.Z-SNAPSHOT
```

check the different pom.xml and MANIFEST.MF, feature.xml and category.xml files

You have eventually to manually update the category.xml at /org.eclipse.papyrus-sysml/releng/org.eclipse.papyrus.sysml14.p2/category.xml

Push it as new patch and make the review.

2.4.9 Restore eventually the 0.0.0 in the target platform

It could be good to restore the 0.0.0 version in your futur target platform

2.5 Post-Actions

Close the initial ticket.

You can send an email to the developers mailing list

3 SOP Add Missing License

3.1 Context

Sometimes your gerrit failed due to missing license header files. This has been detected thanks to the com.mycila.license-maven-plugin plugin

3.2 How to add license in batch mode?

```
mvn license:format -Dlicense.header=/home/flefevre/gitNeon/org.eclipse.papyrus.tool
```

3.3 Run mvn from any pom directory and enable license check

Add the following property to reference the root directory of the sysml project

```
-Dcomponent.root=
```

4 SOP Target Platform Creation

4.1 Context

All components are based upon a set of targets platform localized at the targetplatform directory. You can update them with the Obeo plugin or directly from a maven command.

It will switch resolve all feature with the latest ones found on the different update site you are refering.

4.2 How to?

4.2.1 Pre-requisite

You have to have a target.file at the root of your target platform plugin. You need to specify with target platform to activate through profile management.

```
mvn validate -DtargetUpdate=true -Declipse.targetrelease=neon-papyrusnightly -Decli
```

the tycho-version-bump-plugin is bound to the validate phase

-DtargetUpdate=true ensure you enforce the taregtUpdate

-Declipse.targetrelease=neon-papyrusnightly -Declipse.release=neon-papyrusnightly: specify with target to update

-f targetplatform/pom.xml: specify the pom to look at

5 FAQ

5.1 User Frequently Asked Questions

General

1. How to clear the different caches?

5.2 General

How to clear the different caches?

You have several options:

Clean the local project artifacts

```
mvn clean install
```

Force update of the downloadded plugins

```
mvn install -U
```

Remove manually downlaoded artifacts, for example

```
rm -rf .m2/repository/org/eclipse/papyrus
```

Ignore local tycho artifacts

```
mvn clean install -Dtycho.localArtifacts=ignore
```

Force the cache of tycho

```
rm -rf .m2/repository/.meta/p2-local-metadata.properties
```