Developer Guide

Table of Contents

Contribution

Contribution

There are several ways in which you may contribute to this project.

- File issues
- Submit a pull requests

Found a bug or missing feature?

Please file an issue in our issue tracking system.

Submit a Pull Request

If you found a solution to an open issue and implemented it, we would be happy to add your contribution in the code base. For doing so, please create a pull request. Prior to that, please make sure you

- rebased against the `develop` branch
- stick to project coding conventions
- added test cases for the problem you are solving
- added docs, describing the change
- generally comply with codeacy report

Project Setup

Project Setup

If you are interested in developing and building the project please read the following the instructions carefully.

Version control

To get sources of the project, please execute:

```
git clone https://github.com/camunda/camunda-rest-client-spring-boot.git
cd camunda-rest-client-spring-boot
```

We are using gitflow in our git SCM for naming brnaches. That means that you should start from `develop` branch, create a `feature/<name>` out of it and once it is completed create a pull request containing it. Please squash your commits before submitting and use semantic commit messages, if possible.

Project Build

Perform the following steps to get a development setup up and running.

```
./mvnw clean install
```

Integration Tests

By default, the build command will ignore the run of `failsafe` Maven plugin executing the integration tests (usual JUnit tests with class names ending with ITest). In order to run integration tests, please call from your command line:

```
./mvnw -Pitest
```

Project build modes and profiles

Camunda Version

You can choose the used Camunda version by specifying the profile `camunda-ee` or `camunda-ce`. The default version is a Community Edition. Specify `-Pcamunda-ee` to switch to Camunda Enterprise edition. This will require a valid Camunda license. You can put it into a file `~/.camunda/license.txt` and it will be detected automatically.

Documentation

We are using JavaEden Orchid for generation of a static site documentation and rely on AsciiDoc as much as possible.

Tip If you want to develop your docs in 'live' mode, run `./mvnw -f docs -Pserve-docs` and access the http://localhost:8080/ from your browser.

For creation of documentation, please run:

```
./mvnw -f docs orchid:build
```

This operation requires special permissions. You need to replace `GITHUB_TOKEN` by the token

Warning of the github pages repository, allowing to publish the pages.

In order to publish documentation to github pages, please run from command line

```
./mvnw -f docs -Pdeploy-docs -DgithubToken=GITHUB_TOKEN
```

Generation of JavaDoc and Sources

By default, the sources and javadoc API documentation are not generated from the source code. To enable this:

```
./mvnw clean install -Prelease
```

Starting example application

To start applications, either use your IDE and create run configuration for the class:

- `org.camunda.bpm.extension.rest.example.CamundaBpmFeignExampleApplication`

Alternatively, you can run them from the command line:

```
./mvn spring-boot:run -f example
```

Continuous Integration

Travis CI is building all branches on commit hook. In addition, a private-hosted Jenkins CI by Camunda is used to build the releases from `master` branch.

Release Management

Release management has been set-up for use of Sonatype Nexus (= Maven Central) and are produced by Camunda Jenkins.

What modules get deployed to repository

Every Maven module is enabled by default. If you want to change this, please provide the property

```
<maven.deploy.skip>true</maven.deploy.skip>
```

inside the corresponding `pom.xml`. Currently, all examples are *EXCLUDED* from publication into Maven Central.