# Pepper

## Developer's Guide

**Florian Zipser** <saltnpepper@lists.hu-berlin.de>
**INRIA**
**SFB 632 Information Structure / D1 Linguistic Database**
**Humboldt-Universität zu Berlin**
**Universität Potsdam**

# Pepper: Developer's Guide

by Florian Zipser, , , , and

Version 1.1.4-SNAPSHOT

# Table of Contents

# Foreword

The intention of this document is first to give a guide to the user of how to use the here mentioned pepper modules and how to utilize a mapping performed by them. Second this document shall give a closer view in the details of such a mapping in a declarative way, to give the user a chance to understand how specific data will be mapped by the presented pepper modules.

# Chapter 1. Subprojects

Pepper is a mavenized (see: ) Java project basing on the plugIn System OSGi (see: ). It is separated into several sub projects.

- pepper-exceptions Including a hierarchie of several java exceptions derived from the super class RuntimeException (see).
- pepper-modules Contains the three pepper-module classes PepperImporter, PepperExporter and PepperManipulator. These are the classes manipulating the salt model and doing the linguistic work. We also provide a set of modules to import several formats into salt, to export salt to several formats and to manipulate a salt model, for instance by enhancing the annotations.
- pepper-workflow This sub project contains the model of the workflow specification (.pepperparams files).
- pepper-framework This project is the core of pepper and provides the architecture of Pepper to convert data.
- pepper-starter This is a rather simple project, just for starting the pepper-framework. Both was separated, because of the use of OSGi. Pepper-starter is not an OSGi project, but pepper-framework is.
- pepper-testSuite With the project pepper-testSuite, we provide a set of sub-projects, which are very useful, when creating own pepper-modules. This project is also a maven parent project and itself contains the sub-projects:
- pepper-moduleTests This project contains three test-classes PepperImporterTest, PepperExporterTest and PepperManipulatorTest. All are derived from the class TestCase coming from Junit(see: ). With these classes you can use a set of predefined tests, your new module has to pass. It is very simple, to use these classes, just Create a test-class for your module and derive it from one of the three classes. Now you can enhance your class by own tests, which are specific for your module. For testing, just start your test class in a Junit environment, or let them test by maven. The derived tests will be launched automatically.
- pepper-testEnvironment The project pepper-testEnvironment provides the class TestRunner. This class is an OSGi bundle having an activator (see: ) and realizes an entry point to run pepper and to test your new module without compiling all sources. This is a nice way for developing and debugging single modules. When you use an IDE, supporting OSGi for instance like Eclipse (see: ) just start this class in Run configurations → OSGi. Select all OSGi bundles, you need (especially your new module) and run it.