Generation of the Kermeta Compiler Documentation

Cyril Faucher

Abstract

This document aims to explain the user interface abilities dedicated to the generation of the Kermeta Compiler Documentation.

Generation of the Kermeta Compiler Documentation
Cyril Faucher
Published Build date: 17-December-2007
Published $Date: $

Table of Contents

CHAPTER

Preface

This plugin is aimed to provide a useful environment to manage Use and Test cases related to the issues in compiling Kermeta. The final result is a generated document (*.html, *.pdf) that summarizes and describes Use and Test cases. There is a status board too, this one indicates design and implementation status for each use case.

Important

Kermeta is an evolving software and despite that we put a lot of attention to this document, it may contain errors (more likely in the code samples). If you find any error or have some information that improves this document, please send it to us using the bug tracker in the forge: **http://gforge.inria.fr/tracker/?group_id=32** or using the developer mailing list (kermeta-developers@lists.gforge.inria.fr) Last check: v0.99.0

CHAPTER 1

Compilation Process

1.1. Required resources

You must check out in your workspace the project ant-docbook-styler from the Kermeta forge.

Plugin dependencies: your installation should contain the following plugins:

- org.kermeta.trek.model
- org.kermeta.trek.model.edit
- org.kermeta.trek.model.editor
- org.kermeta.trek.ui
- org.kermeta.compiler.trek.ui

1.2. How to build the compiler documentation ?

The generation of the final document is done in four steps:

1. edit summary and progress text files for each use cases,
2. generate the Trek Use Kase Models,
3. generate the docbook file,
4. generate the html-single and pdf files.

1. Editing summary and progress text files for each use cases:

Folders organization:

Assumption: a test case verifies an unique use case, then a test case can be contained by a use case.

- The use cases are defined in the folder: *project_dir*/unit_test
- A use case is contained inside a folder named like this: ~/unit_test/comp_*use_case_name* Each use case folder contains a summary file: summary_*use_case_name*.txt and a progress file: progress_*use_case_name*.txt
- The test cases that verify a use case are contained in subfolders as: ~/comp_*use_case_name*/*test_case_name* The test case name is calculated like this: test*number_of_the_test_case* Each test case folder contains a summary file: summary_*test_case_name*.txt Each test case folder contains a folder "input" and "expected_output": "input" contains the resources that are required in order to launch the test case and the "expected_output" contains the resources that are expected after the execution of the test case.

The Summary and progress text files are used during the second step: generation of the Trek models, thus to update information in the final document, the recommended way is to modify summary and progress text files, then restart the second, third and fourth steps (these three steps are press-button operations).

2. Generation of the Trek Use Kase Models:

Right-click on the folder: *project_dir*/unit_test and choose the action:

> Trek for Kermeta Compiler > Generate the Trek Use Kase Models (full process)

Refresh the project (press F5) for avoiding "out of synchronize" mistakes.

Some *.trek files are generated.

3. Generation of the docbook file (*.xml):

Right-click on the Kermeta file: *project_dir*/doc/builder/trek2docbook.kmt and choose the action:

> Run As > Kermeta App

Refresh the project (press F5) for avoiding "out of synchronize" mistakes.

A docbook file is generated: *project_dir*/doc/docbook/unit_test.docb.xml

4. Generation of the html-single and pdf files:

Right-click on the xml file (Ant): *project_dir*/doc/builder/unit_test_build.xml and choose the action:

> Run As > Ant Build

Refresh the project (press F5) for avoiding "out of synchronize" mistakes.

A html-single file is generated: *project_dir*/doc/build/html.single/unit_test/index.html

A pdf file is generated: *project_dir*/doc/build/pdf.fop/unit_test/unit_test.pdf

1.3. How to customize the docbook file generation ?

The generation of the docbook file is built thanks to the tool: KET (more info. at: http://www.kermeta.org/mdk/ket). Thus, some templates have been used to generate the docbook file, these resources are located in the folder: *project_dir*/doc/templates as *.ket files.

The translation/compilation KET to KMT is automatic thanks to a KPM rule that takes in input a *.ket file and transforms it into a *.kmt file. Also if you change the location of the *.ket file, you must update .project.kpm file, i.e.: updating the related name filters.

To conclude, in order to add or customize the docbook templates (*.ket), you have to adapt the *.ket files and maybe adapt the model navigation written in *project_dir*/doc/builder/trek2docbook.kmt.