C:\Documents and Settings\malawito\Mis documentos\LaForjaSVN\trunk\Misc\FaMa-Docs\resources\src-docs\logoBench.jpg

FaMa BenchMarking System

User Guide

# 

Contenido

[1. Introduction 3](#_Toc224994411)

[2. FAMA Benchmark files 3](#_Toc224994412)

[3. Quick Start 3](#_Toc224994413)

[3.1 Creating and configuring a new empty project. 3](#_Toc224994414)

[3.2 Main classes 5](#_Toc224994415)

[4. Metamodel (XML Schema) 6](#_Toc224994416)

[5. References 6](#_Toc224994417)

# 1. Introduction

It is a Benchmark that helps to extract information about the automated analysis of feature models,  we provides more easy to use with FaMa, by this way it's easy   to analyses any reasoner with any feature model that you want.

# 2. FAMA Benchmark files

**Benchmark.jar:** This is the benchmark application. It must be included in the build path.

**FAMAconfig.xml**: This is the main configuration file where we could define the questions, reasoners and criteria we want FAMA Tool Suite to use.

**reasoner.properties:** This is a configuration file were we select the reasoned we want to use in our tests,(it update automatically).

# 3. Quick Start

At first download your FaMa Benchmark distribution that you can find at [www.isa.us.es/fama](http://www.isa.us.es/fama) in zip format.

Once you have downloaded it, the first thing you must do is unzip the file. Inside it, you will find the following folders and files:

* lib: with the jars
* docs: with the user guide
* samples: BenchmarkFacadeTest.java, Benchmarktest.java and BenchmarkAndFamaTest.java. Some example java classes.
* files:
  + version.txt: with the information of the version of different jar files
  + license.txt: with the license terms
  + FAMAconfig.xml: it is the main configuration file. Please do not modify it, unless you are sure about your are doing
  + reasoner.properties; It is a configuration file. Please do not modify it, unless you are sure about your are doing
  + FAMA\_logo.png: the FAMA logo
  + Feature-model-schema.xsd : this is the pattern to create our xml for FAMA

## 3.1 Creating and configuring a new empty project.

Once you have uncompressed FaMaTS, follow these steps to run the first time FaMaTS.

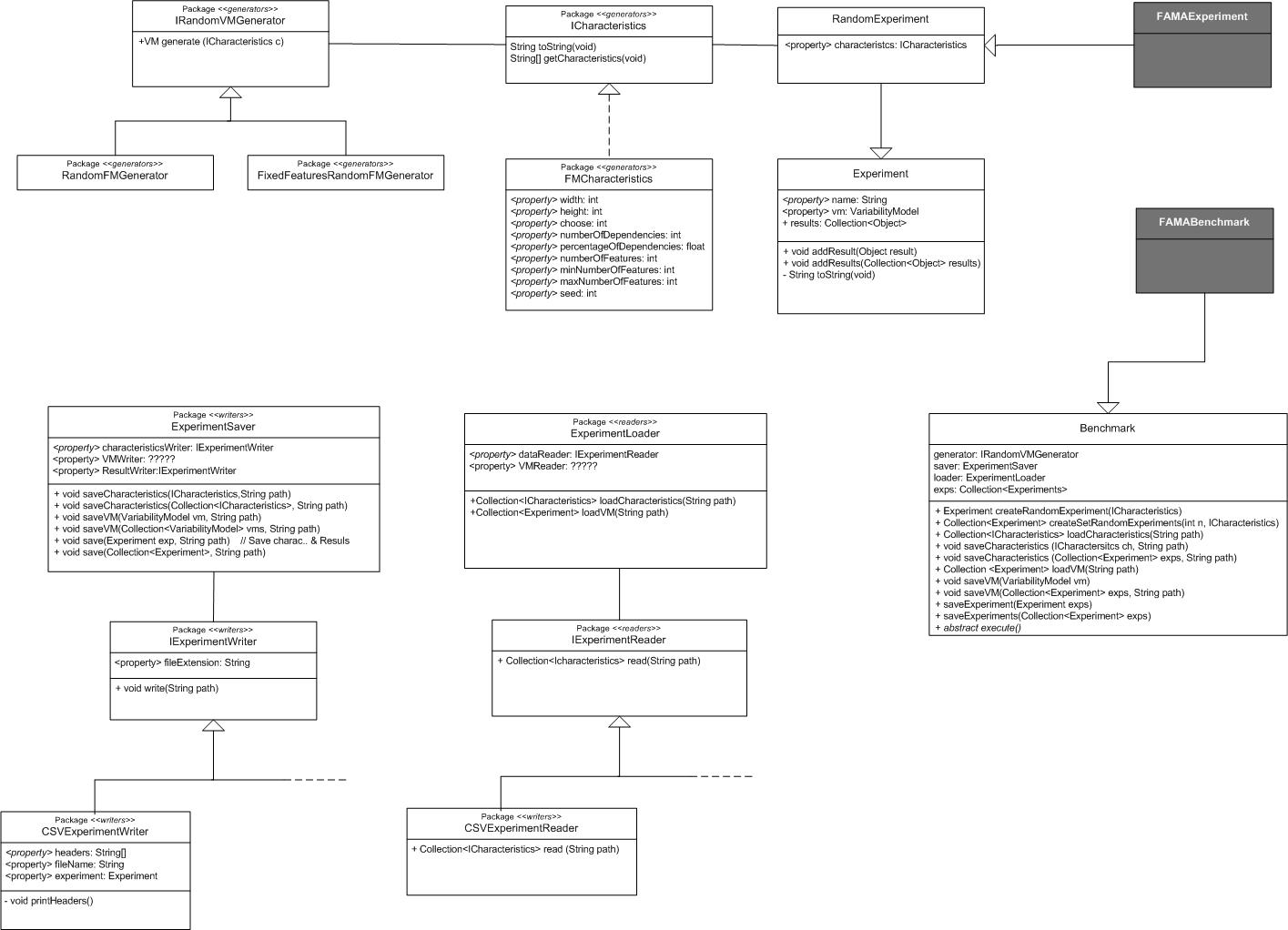
1. Make sure that you have, at least, jre 5.0 (or higher) installed
2. Download FaMa FW
3. Create a new java empty project (with, at least, jre 5.0)
4. Make a new dir at the root folder, called “lib” (if you are on NetBeans, make sure that you are on “files” view)
5. Copy the folder files at the root folder. If you are on NetBeans, make sure that you are on “files” view, also the libs founded in FaMa, <www.isa.us.es/fama>).
6. Copy to this folder all the jars that are on FaMaBenchmark.
7. Add to the project’s build path some files previously copied: FaMaSDK.jar, FaMaQuestions.jar and FaMaModel.jar
8. Copy FaMaConfig.xml and reasoner.properties to the project’s root folder.
9. Copy the folder “fm-samples” to the project’s root folder
10. Create a new package on the project’s source folder (with the name that you want)
11. Copy the java sources that are on “samples” on the new package
12. Run the classes

Now you can run your FaMaFacade test file and verify that everything is working correctly. The output should look like this:

You can run the other examples included too.

## 3.2 Main classes

Now, we are going to detail the classes that appear on the example.



-Benchmark.java: It’s a façade that provides you all the methods that you will need.

|  |
| --- |
| Experiment createRandomExperiment(ICharacteristics ch);  It generates one RandomExperimet with the characteristics that you want, you have to supply the characteristics(only if you are using this facade) whit an cvs file in /files/test that is supplied with the benchmarking System    Experiment createRandomExperiment(String name,ICharacteristics ch);  It generates one RandomExperimet with the characteristics and a name  Collection<Experiment> createSetRandomExperiment(**int** n, ICharacteristics ch);  It creates a set of n experiments;  Collection<ICharacteristics> loadCharacteristics(String path);  It reads the characteristics from the path you want  **void** saveCharacteristics(ICharacteristics ch, String path);  It save the characteristics to the path you want  **void** saveCharacteristics(Collection<? **extends** Experiment> exps, String path)**;**  It save the characteristics from the FMExperimets where you want  Collection<Experiment> loadVariabilityModel(String path);  It generates a collection of experiments with the variability model that you want    **void** saveVariabilityModel(VariabilityModel vm,String path);  It saves the variability model in xml-fama format    **void** saveVariabilityModel(Collection<Experiment> exps, String path);  It saves the variability model in xml-fama format    **void** saveExperiment(Experiment exp,String path);  It saves all the information of the experiment where you want    **void** saveExperiments(Collection<Experiment> exps,String path) **throws** IOException;  It saves all the information of the experiments where you want |

-FaMaBenchmark: to use with FaMa, FaMa Team provides an extremely easy to use façade called FaMaBenchmark.

|  |
| --- |
| This façade only have a method called execute, so when you use it, it will do the entire job for you, but you will need to put the files famaFile.cvs and fixedChars.cvs, in the subdirectory /files/test of your workspace. |

# 4. Metamodel (XML Schema)

The feature model metamodel is written on XML Schema. You can find here at the distribution package, or at the website of FaMa on googlecode (<http://famats.googlecode.com/files/feature-model-schema.xsd> )

# 5. References

FaMa’s site: <http://www.isa.us.es/fama>

Project site on google code: <http://code.google.com/p/famats>

ISA research group: <http://www.isa.us.es>

Sat4j: <http://www.sat4j.org>

JavaBDD: <http://javabdd.sourceforge.net>

Choco-Solver: <http://choco-solver.net>

JavaCSV: <http://sourceforge.net/projects/javacsv/>

Feel free to test it, and give feedback