ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

J. Hoenicke
J. Christ

01.02.2012
Hand in solutions via email to
`christj@informatik.uni-freiburg.de`
until 08.02.2012 (only Java sources, KeY proofs, and PDFs accepted).
Paper submissions possible after the lecture.

# Tutorials for "Formal methods for Java"
# Exercise sheet 12

## Exercise 1: Installing JPF

Install a Mercurial client for your operating system. Get a copy of the JPF repository either with

```
hg clone http://babelfish.arc.nasa.gov/hg/jpf/jpf-core
```

or a similar command from your Mercurial client.
Compile the downloaded version, e.g., using ant from the cloned repository:

```
bin/ant
```

Instructions for Eclipse or NetBeans can be found on the JPF wiki at `http://babelfish.arc.nasa.gov/trac/jpf/wiki`

## Exercise 2: Configuring JPF

Create the directory `.jpf` in your home directory. Inside this directory create the main configuration file `site.properties` containing only the lines

```
jpf.home=<where you cloned jpf-core to>
jpf-core=${jpf.home}/jpf-core

extensions=${jpf-core}
```

where the `jpf.home` is set appropriately.

## Exercise 3: Create a JPF Project

Get a copy of the jpf-template project, e.g., with

```
hg clone http://babelfish.arc.nasa.gov/hg/jpf/jpf-template
```

and compile it.
Create a new JPF project with the following command line

```
<path-to-jpf-template>/bin/create_project <Project-Name> <path-to-jpf-core>
```

where the parts within < and > are set appropriately.
This step creates a new folder containing your project. The folder contains a `bin` directory that you can use to run JPF, a `src` directory with many subdirectories that you can use for development and examples, and a ant build script to build your project.

**Exercise 4: Configuring a SUT**
Create a subfolder `src/main/exercises` in your new JPF project. Download the file `NonNullChecker.java` from the website of the lecture and place it in this folder. Download the file `NonNull.java` and place it in the folder `src/examples` of your JPF project. Run `ant` from the root folder of your JPF project to compile both the listener and the example.
Now, write a configuration file for `NonNull.java`. JPF should use the `NonNullChecker` listener which should check the fields `t1`, `t2`, and `gObj` of class `NonNull`, and the variable `test` in the main method. The corresponding configuration elements for `NonNullChecker` are `nnc.fields` and `nnc.vars`.
Test your configuration on the example using the `jpf` script from the `bin` directory of your project. Note that you need to rerun `ant` if you change anything but the configuration file.

**Exercise 5: Listener Notification**
The `NonNullChecker` from the previous exercise uses notifications after the instruction has been executed. Transform this listener such that it checks before the instruction is executed.
*Hint:* You can look at the elements on the stack with the ThreadInfo.peek(**int**) member function.
*Hint 2:* Look into the source file of the `ACONST_NULL` instruction to see what the stack value of a `null` pointer is in JPF.

**Exercise 6: Bonus: NonNull Invariant**
Unfortunately, the `NonNullChecker` cannot be used to realize the invariant

> Field x in class Y is never null!”

Explain why this invariant cannot be established and give a possible solution to the problem.