

## **Wikiprint Book**

**Title: JPF Inspector**

**Subject: Java Path Finder - projects/jpf-inspector**

**Version: 11**

**Date: 03/15/2013 05:29:14 PM**

## Table of Contents

<b>JPF Inspector</b>	<b>3</b>
Contact	3
Description	3
User guide	3
Repository	3

## JPF Inspector

JPF Inspector is a tool for inspection and control of JPF execution. It supports breakpoints and "single step" execution (forward and backward) at different levels of granularity, and it allows the user to examine program's state (threads, call stacks, and heap), like in a standard debugger (GDB).

Development of this tool started in the scope of the [Google Summer of Code \(GSoC\)](#) program in 2010.

### Contact

Pavel Jancik <Alfik.009@...>, Pavel Parizek <parizek@...>

### Description

Basic features of the JPF Inspector tool include:

- Breakpoints. They can be set to: a specific line of code, instruction at a given position, specific instruction type (invoke, return), read or write access of a variable, object creation and destruction, garbage collection (start and stop), transition boundary (next value choice). Each breakpoint can be in any of the following three states: enabled, disabled, and log. If a breakpoint in the log state is reached, a log message is printed and the execution is not interrupted.
- Single-step execution. It is possible to tell JPF to execute the SuT in a single-step fashion. Currently only forward execution is supported with the following levels of granularity: instruction and source code line (w/o stopping at nested method calls).
- Program state inspection. Inspector provides a mechanism for exploring program state, including values of heap objects and call stacks of individual threads. Navigation over references (object tree) is supported too - a custom expression language is used internally.

Planned advanced features may include some of the following: program state modification, JPF introspection (displaying raw state of choice generators and listeners), gathering of various statistics (profiling information), batch execution (saving and loading command sequences), and configuration management (predefined breakpoints).

JPF Inspector currently provides GUI that is implemented as a new panel for the [JPF shell](#) framework.

### User guide

[User guide](#) describes the currently supported commands and extension points (API) of the Inspector tool. It includes many examples and some screenshots of the Inspector user interface.

### Repository

The Mercurial repository with all the sources is on <http://babelfish.arc.nasa.gov/hg/jpf/jpf-inspector>