TracNav

- JPFWiki - Welcome Page

Introduction...

Installing JPF...

User Guide...

Developer Guide...

Modules

- jpf-core
- jpf-actor
- jpf-awt
- jpf-awt-shell
- jpf-concurrent
- jpf-cv
- jpf-delayed
- jpf-guided-test
- jpf-mango
- jpf-racefinder
- jpf-rtembed
- jpf-statechart
- net-locache
- jpf-aprop
- jpf-numeric
- jpf-symbc
- jpf-concolic
- jpf-symbc-load?
- jpf-extended-test-gen
- jpf-parallel-spf?
- eclipse-jpf
- netbeans-jpf
- jpf-inspector
- jpf-shell
- jpf-template
- jpf-trace-server

Summer Projects
- External Projects
- Change(B)log

About...

- Events
- Presentations
- Papers
- FAQ
- History?
- Support
- Playground
- Table of Context

JPF Modules
JPF is divided into separate runtime modules that are runtime configurable. The `jpf-core` module is always required, but all others are optional "JPF extensions", implementing features such as different execution modes, special properties or alternative library models.

Each module has its own repository and/or binary snapshots as attached files to respective wiki pages.

Here is the list of modules that are available from this server, each with more details about purpose, installation and use.

**JPF Core module:**
- `jpf-core` - the VM and core mechanisms used by all other projects - explicit state model checking for Java bytecode

**Model Checking Modules:**
- `jpf-actor` - Tool and framework for systematic testing of actor programs (e.g. Scala)
- `jpf-aprop` - Java annotation based properties and their corresponding checkers
- `jpf-awt` - JPF specific library implementations for `java.awt` and `javax.swing`
- `jpf-awt-shell` - specialized JPF shell for model checking `java.awt` and `javax.swing` applications
- `jpf-concurrent` - optimized `java.util.concurrent` library implementation for JPF
- `jpf-delayed` - postpones non-deterministic choice of values until they are used
- `jpf-guided-test` - Framework for guiding the search using heuristics and static analysis
- `jpf-mango` - specification and proof artifact generation
- `jpf-numeric` - an alternative bytecode set for inspection of numeric programs
- `jpf-racefinder` - a precise data race detector in a relaxed Java memory model
- `jpf-rtembed` - programs for real-time and embedded platforms (e.g., RTSJ and SCJ)
- `jpf-statechart` - UML statechart modeling
- `net-iocache` - I/O cache extension to handle network communication
- `jpf-cv` - compositional verification using JPF
- `jpf-nas` - model checking of distributed multithreaded programs

**Symbolic Execution Modules:**
- `jpf-symbc` - Symbolic PathFinder - symbolic execution for Java bytecode
- `jpf-concolic` - concolic (mixed concrete/symbolic) execution for Java bytecode
- `jpf-symbc-load?` - Symbolic Load Test Generation
- `jpf-extended-test-gen` - Using JPF and SPF for generating tests with respect to MC/DC coverage
- `jpf-parallel-spf?` - Generic framework for parallelizing SPF (needs `jpf-extended-test-gen`, `jpf-symbc`, `jpf-core`)
- `jpf-regression` - Directed Incremental Symbolic Execution (needs `jpf-guided-test`, `jpf-symbc`, `jpf-core`)
- `jpf-abstraction` - Abstract PathFinder

**JPF Tool Modules:**
- `eclipse-jpf` - JPF launcher plugin for Eclipse IDE
- `netbeans-jpf` - JPF launcher plugin for NetBeans IDE
- `jpf-inspector` - a tool for inspection and control of JPF execution
- `jpf-shell` - the generic user interface support for JPF
- `jpf-template` - a tool for creating new JPF projects
- `jpf-trace-server` - enables storing, querying and analysis of the execution trace