

# Wikiprint Book

**Title:** The Attribute System

**Subject:** Java Path Finder - devel/attributes

**Version:** 7

**Date:** 03/15/2013 05:33:16 PM

## Table of Contents

TracNav	3
Introduction...	3
Installing JPF...	3
User Guide...	3
Developer Guide	3
MJI...	3
Projects...	3
About...	3
<b>The Attribute System</b>	<b>3</b>
Usage	4
<b>Note:</b>	<b>4</b>
<b>Note:</b>	<b>4</b>

## TracNav

- [JPFWiki - Welcome Page](#)

### Introduction...

### Installing JPF...

### User Guide...

### Developer Guide

- [Design](#)
- [Choice Generator](#)
- [Partial Order Reduction](#)
- [Attributes](#)
- [Listener](#)

### MJI...

- [Bytecode Factory](#)
- [Logging](#)
- [Report](#)
- [Embedded](#)
- [JPF tests](#)
- [JPF project layout](#)
- [Create a JPF project](#)
- [Coding Conventions](#)
- [Hosting update site](#)

### Projects...

- [Summer Projects](#)
- [External Projects](#)
- [Change\(B\)log](#)

### About...

- [Events](#)
- [Presentations](#)
- [Papers](#)
- [FAQ](#)
- [History?](#)
- [Support](#)
- [People?](#)
- [Playground](#)
- [Table of Context](#)

## **The Attribute System**

While JPF stores values for operands, local variables and fields very similar to a normal VM, it also features a storage extension mechanism that lets you associate arbitrary objects with stack slots (operands and locals), fields, and whole objects (ElementInfos). The attribute objects can be set/used in [native peers](#) or [listeners](#) to add state stored/restored information that automatically follows the data flow.

Note that JPF does not restore attribute object values upon backtracking per default, only attribute references. If you need to make sure attribute values are restored, you have to use copy-on-write and then store back when accessing and modifying such attributes.

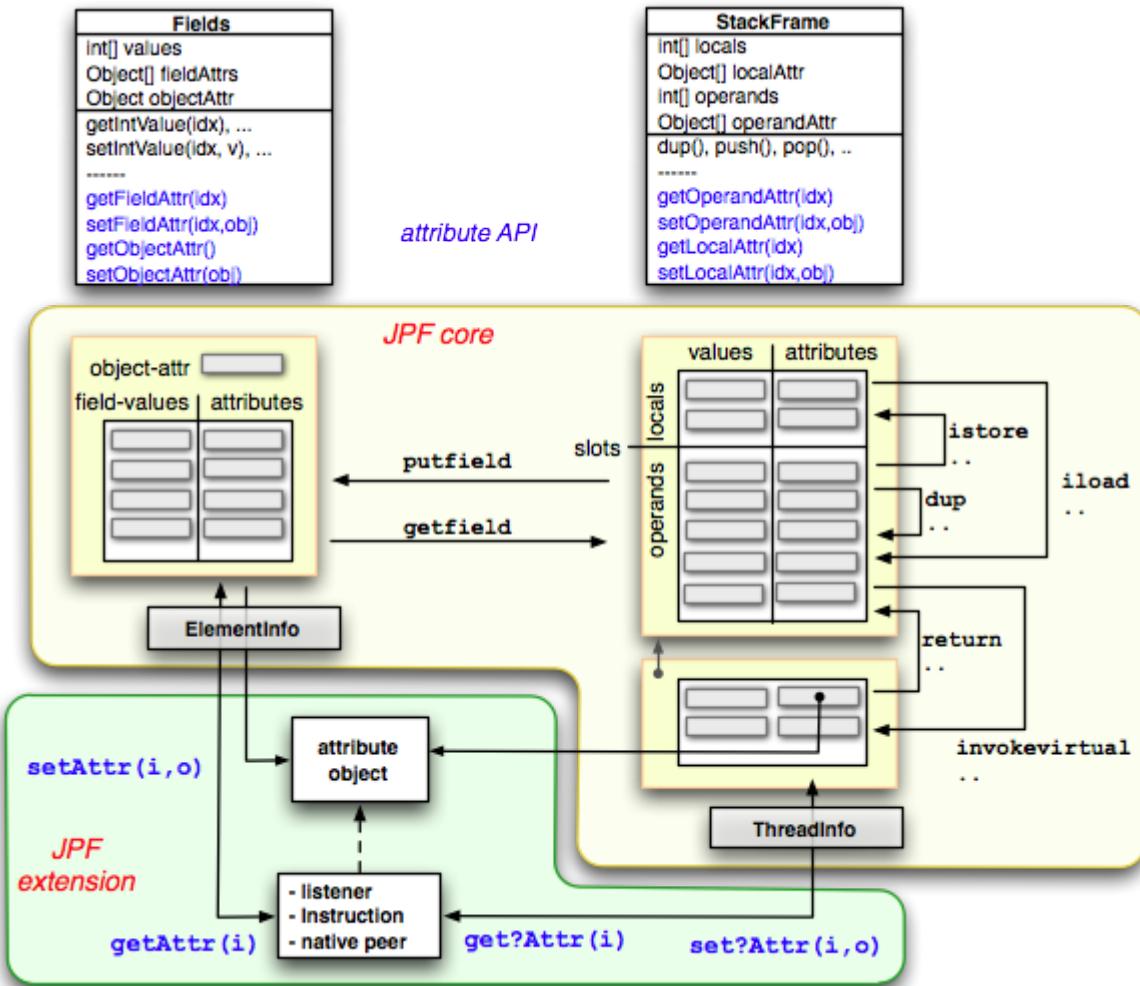


Figure: JPF Attribute System

JPF provides an API to set/access these attributes, which is located in `gov.nasa.jpf.jvm.Fields` (for field attributes) and `gov.nasa.jpf.jvm.StackFrame` (for local variables and operands). Once set, the JVM copies the attributes each time it reads/writes the associated field or stackframe slot.

## Usage

For example, such attributes can be used to represent symbolic values or numeric error bounds. It should be noted though that attributes impose additional runtime costs, which is also why we don't treat normal, concrete values just as a special kind of attribute (normal values are still stored separately as builtin types like `int`). The upside of this is that your attributes coexist with normal, concrete values, which for instance allows things like mixed symbolic and concrete execution.

## Note:

JPF now can associate attributes not only with fields of an object, but with the object as a whole. See the `gov.nasa.jpf.jvm.ElementInfo` API for details

## Note:

while there is an API to set/retrieve attributes based on type, there is no implementation yet that allows multiple attributes to be stored