



# An Embedded Java™ Compiler

Arno Unkrig 2005-02-15

# Agenda

- Scenic Route
- The Gory Details
- Inside an Open Source Project

# Setup

- Download `janino-x.y.z.zip` from <http://janino.codehaus.org>
- Extract `janino.jar`
- Put it on the `CLASSPATH`

# ExpressionEvaluator

```
import org.codehaus.janino.*;

// Compile the expression once; relatively slow.
ExpressionEvaluator ee = new ExpressionEvaluator(
    "c > d ? c : d",           // expression
    Integer.TYPE,             // expressionType
    new String[] { "c", "d" }, // parameterNames
    new Class[] { Integer.TYPE, Integer.TYPE } // parameterTypes
);

// Evaluate it with varying parameter values; very fast.
Integer res = (Integer) ee.evaluate(
    new Object[] {           // parameterValues
        new Integer(10),
        new Integer(11),
    }
);
System.out.println("res = " + res);
```

# ScriptEvaluator

```
import org.codehaus.janino.*;

// Create "ScriptEvaluator" object.
ScriptEvaluator se = new ScriptEvaluator(
    ( // script
        "System.out.println(\"Hello world\");\n"
        + "return true;\n"
    ),
    Boolean.TYPE, // returnType
    parameterNames, // parameterNames
    parameterTypes // parameterTypes
);

// Evaluate script with actual parameter values.
Object res = se.evaluate(parameterValues);

// Print script return value.
System.out.println("Result = " + res);
```

# Agenda

- Scenic Route
- **The Gory Details**
- Inside an Open Source Project

# org.codehaus.janino. **Compiler**

- Provides all features of JAVAC:  
Source path, class path, extension directories, boot class path, destination directory, verbose flag, debug options
- Additional features: Warning handle patterns, rebuild flag, source finder
- Wrapper shell script „janinoc“ is a drop-in replacement for JAVAC
- Bindings for ANT and TOMCAT exist

# org.codehaus.janino.**JavaSourceClassLoader**

- Scans, parses, compiles and executes Java source files on-the-fly
- No class files created on the file system
- Wrapper shell script „janino“ implements a combination of JAVAC and JAVA

## **One bizarre example**

Run the JANINO compiler from source to compile itself:

```
$ janino -sourcepath src org.codehaus.janino.Compiler \  
-sourcepath src -d weird-classes src/org/codehaus/janino/Compiler.java  
$
```

# org.codehaus.janino.SimpleCompiler

- Scans, parses, and compiles a single compilation unit from a String/Reader/InputStream into a ClassLoader
- No class files created on the file system

# org.codehaus.janino.**ClassBodyEvaluator**

- Scans, parses and compiles the body of a Java class
- Loads it as a `java.lang.Class`

# org.codehaus.janino.**ScriptEvaluator**

- Scans, parses and compiles the body of a Java method
- Loads it into the JVM and allows for invocation through reflection or an interface

# org.codehaus.janino.ExpressionEvaluator

- Scans, parses and compiles a Java expression
- Loads the resulting class into the JVM
- Allows for evaluation with varying parameters through reflection or an interface

# Debugging JANINO scripts

In order to be able to single-step through JANINO-generated classes:

- **Set**

- Dorg.codehaus.janino.source\_debugging.enable=true

- **Optionally set**

- Dorg.codehaus.janino.source\_debugging.dir=xxx

**to an existing directory**

# Agenda

- Scenic Route
- The Gory Details
- **Inside an Open Source Project**

# How does JANINO work?

- Create an abstract syntax tree (AST) programmatically and/or by parsing Java tokens, rooted at a CompilationUnit object
- Compile the AST into ClassFiles
- During compilation, load referenced classes through the source path and/or the class path
- Store the ClassFiles to disk (Compiler), or
- Load the generated class files through a custom ClassLoader

# History

- Q4/2000: Development of an interpretative expression evaluator for INTERSHOP ENFINITY at Gauss AG
- Q3/2001: Re-implementation with byte code generation
- Q4/2001: Initial LGPL Release
- Continuous addition of language elements
- Q4/2003: Version 1.0 implements all of Java 1.1
- Q2/2004: Version 2.0 adds inner classes and generics (Java 1.2)
- Q1/2005: Janino moved to CODEHAUS.ORG

# Project Structure

- Development: One-Man Team
- CVS, bug tracking tool and web site on [janino.codehaus.org](http://janino.codehaus.org)

# Projects using Janino

Open source projects that use JANINO:

- [Groovy](#) -- an agile dynamic language for the JVM combining lots of great features from languages like Python, Ruby and Smalltalk and making them available to the Java developers using a Java-like syntax
- [Drools](#) -- an augmented implementation of Forgy's Rete algorithm tailored for the Java language
- [Kataba Dynamics](#) -- less verbose, more powerful, more consistent core libraries for Java
- [JINX](#) -- Java multi-user unix-like system

# FAQ

- License – New BSD
- JDK 5.0 language features? – Not planned
- ???

# Off-Topic

- Anybody worked on a JSR?
- Ever used a CODEHAUS project?