Java Security

How Free Software Is Secured Using the Java Language

FOSDEM, Brussels, 2004-02-22

- Sascha Brawer
  brawer@dandelis.ch
- Chris Gray
  chris.gray@kiffer.be
- Dalibor Topić
  robilad@kaffe.org
- Mark Wielaard
  mark@klomp.org

Overview

- Overview
  - Motivation
  - Anatomy of a Java-like system
  - Security in Java (and some problems)
  - Documentation, Quality assurance, Releases
- Current state
  - Class library (GNU Classpath)
  - Virtual Machines (various)
- How you can help

Motivation

- Currently, most free software is written in C

The Good	The Bad	The Ugly
Close to hardware	Easy to make security-related bugs	Libraries not well integrated
Fast execution, even with bad compilers	Lacks “modern” language concepts (“modern” = 1980’ies)	Difficult to learn
Large code base, many libraries	Hard to write portable code	
Ubiquitous		

Motivation

- Java is a good foundation for many projects
  - Type safety and array bounds checking
    - Avoids many typical bugs, crashes, security problems
    - More opportunities for optimizing compilers
  - Modular, object-oriented, concurrent, dynamic
  - Easier to write structured, portable code
  - Very rich library, reasonably well designed (mostly)
  - Large developer base ("COBOL of the 1990'ies")
    - There exist lots of other good (even better?) languages: Oberon, Eiffel, O'Caml, Haskell, Erlang, TOM, Cedar, ...
    - But: They have not many free applications/libraries

Motivation

- But: Java does not solve every problem
  - Over-hyped as a solution to everything
  - Bad reputation for wasting resources (CPU, RAM)
    - It is easy to write inefficient programs
    - Early implementations were very slow
    - Type safety and memory management have their cost
      - Often over-estimated: Array bounds checking (? no buffer overflows) costs ~2% execution time, avoids ~50% security-related bugs
  - Syntax simplified C++ → still not very easy to learn

Motivation

- Java is a compromise
  - Not necessarily ideal, but reasonable
- Plenty of free software is being written in Java
  - Sourceforge.net: 11,032 Java projects

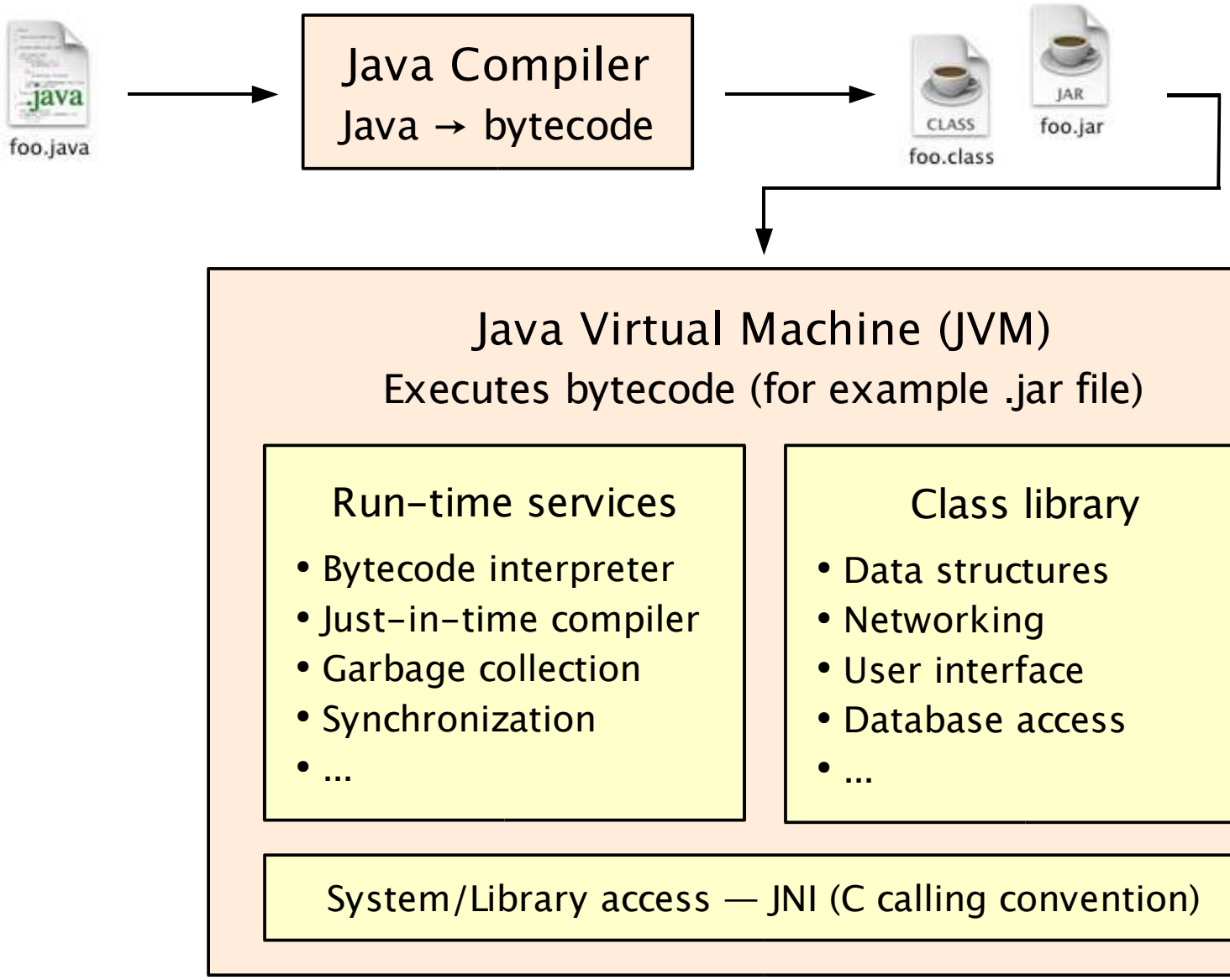
> **Why GNU Will Be Compatible with UNIX**
>
> Unix is not my ideal system, but it is not too bad. The essential features of Unix seem to be good ones, and I think I can fill in what Unix lacks without spoiling them. And a system compatible with Unix would be convenient for many other people to adopt.

— ***Richard Stallman***
The GNU Manifesto, 1985

Motivation

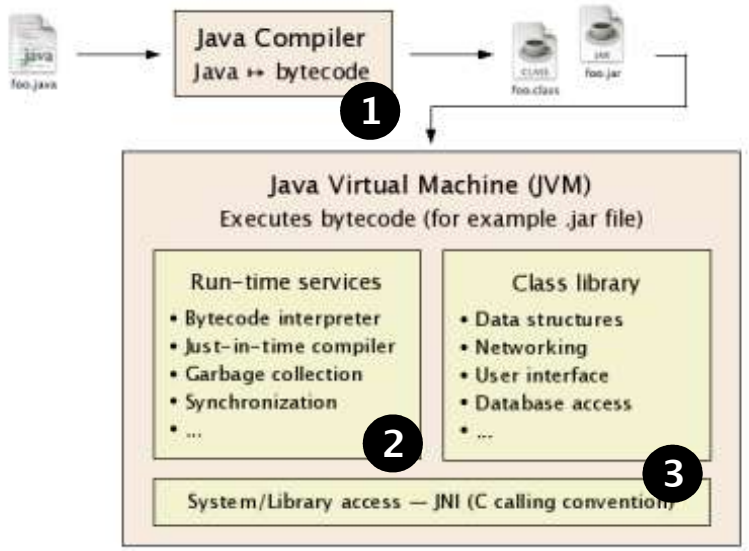
- Freedom!
  - Freedom to use, study, adapt, improve and share
  - Freedom to innovate
  - Not controlled by any single entity
  - Future: Possibility for natural, proven (vs. committee-dictated) standards

Anatomy of a Java-like system



Anatomy of a Java-like system

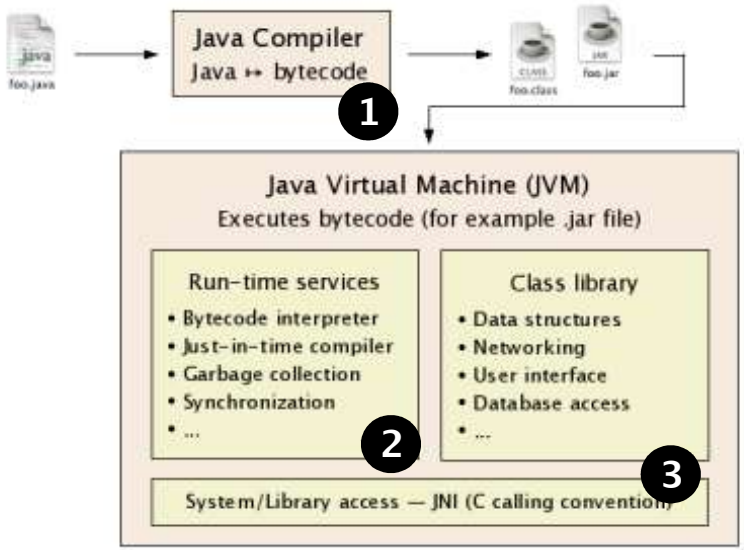
1. Java → bytecode compiler
   - gcj, jikes, kjc, Eclipse, ...
   - Complete up to Java 1.4
     - 1.5 will need some work for generic types: List<A>



2. Run-time services
   - About 15 JVM projects
     - Very diverse goals
     - Almost no common code (yet)

3. Class library
   - java.{lang, math, ...}, javax.{mail, crypto, ...}
   - Some C code for POSIX-like systems

Anatomy of a Java-like system



Very coarse estimates!

Generated using http://dwheeler.com/sloccount/ "Basic COCOMO"

❶ Java compiler

Ex.: Jikes Compiler
71,000 lines of code
~ 17.5 person-years
~ USD 2.4 Mio.

❷ Run-time serv.

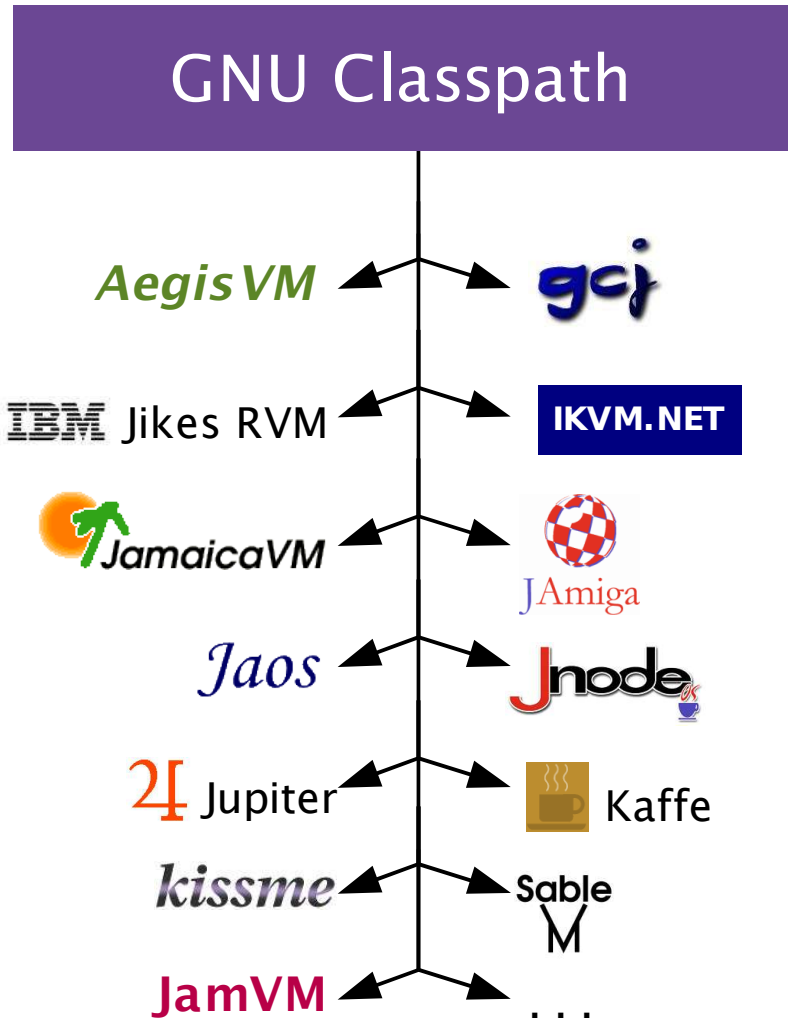
Ex.: Kissme VM
59,000 lines of code
~ 14.5 person-years
~ USD 2.0 Mio.

❸ Class library

Ex.: GNU Classpath
233,000 lines of code
~ 61.5 person-years
~ USD 8.3 Mio.

GNU Classpath

- GNU Classpath is shared among many VM projects
  - "Upstream" provider for the class library
    - java.*, most javax.*
    - Mainly for historical reasons, certain javax.* are provided by other projects
  - Common code base, bug tracking, ...
  - Very permissive license



Free software is flexible

- Done with GNU Classpath:
  - Compiling Java source → stand-alone executable
  - Compiling Java bytecode → CIL (bytecode of .NET)
  - Operating Systems with type-safe kernels
  - JVM for multiprocessor clusters (128 CPUs, Myrinet)
  - Embedded systems with real-time guarantees
  - Alternative access to native system/libs (non-JNI)

Free Software can be adapted
to arbitrary needs → innovation

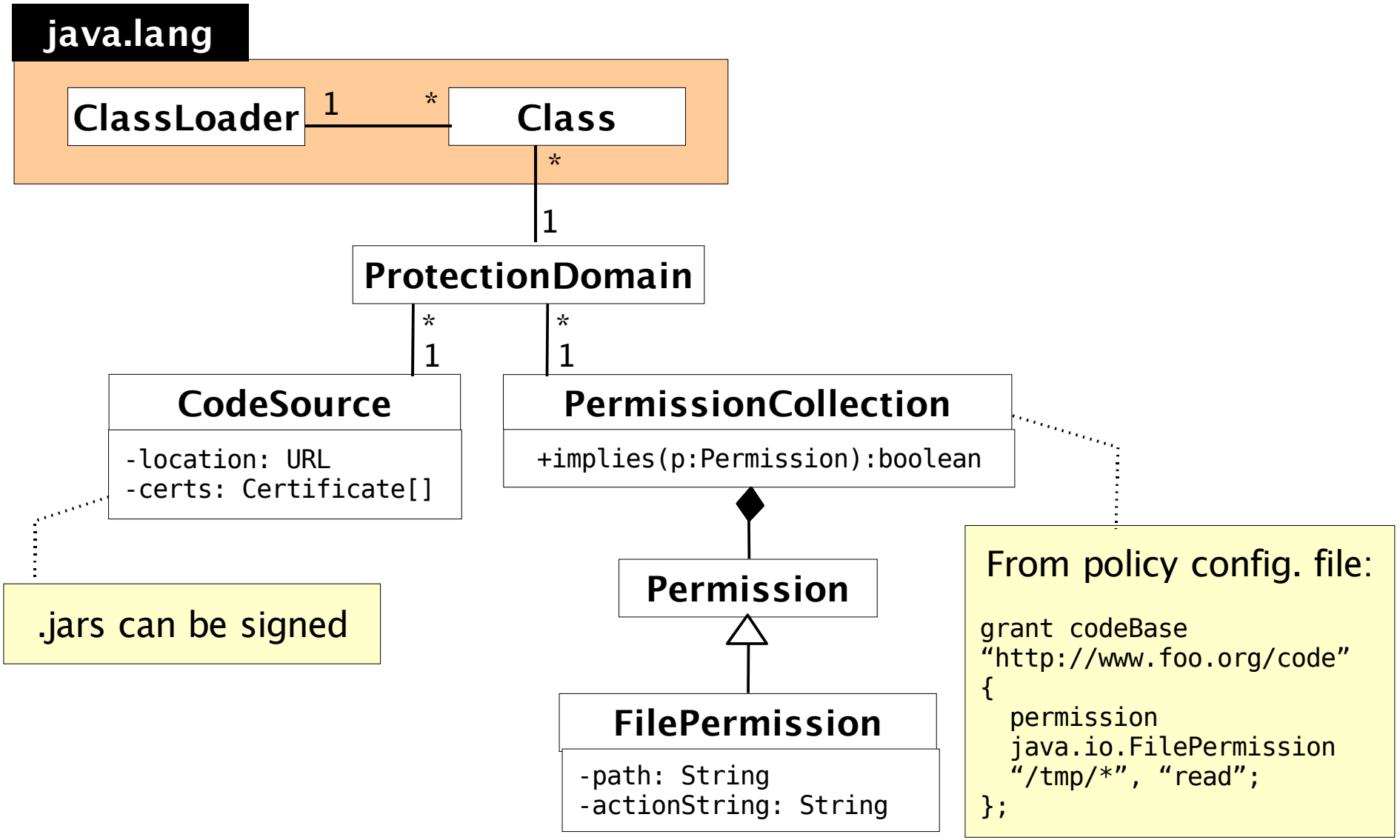
Security in Java: Actors

- Actors in Java security
  - Java-to-Bytecode compiler
    - no `(struct t*)((char*) p + 8)`
  - Verifier
    - bytecode must be well-formed
  - Virtual Machine
    - run-time checks, e.g. dereferencing `null`
  - Class libraries
    - ACL-based security checks

Security in Java: Levels

- 0. Type safety, array bounds checking
  - ensured by: Compiler, Verifier
- 1. Encapsulation (private, protected, etc.)
  - ensured by: Compiler, Verifier
- 2. Fine-grained access control
  - Access Control Lists
  - based on code source

Security in Java: Architecture



Security in Java: Architecture

```
package java.io;
public class File
{
  private String path;
  public void delete()
  {
    Permission p = new FilePermission(path, “delete”);
    System.getSecurityManager().checkPermission(p);
  }
}
```

java.lang.SecurityManager.checkPermiss
ion
java.io.File.delete
org.foo.Wombat.someMethod
org.foo.Wombat.main

Call Stack

For each method on the call stack:

Does the PermissionCollection of its ProtectionDomain imply p?

If not, throw SecurityException

Security in Java: Problems

- Denial-of-service attacks

```
import java.awt.*;

class CPUAttacker
  extends java.awt.Component
{
  public void paint(Graphics g)
  {
    while (true)
      ;
  }
}
```

DoS on AWT (GUI) thread:
Excessive CPU consumption

```
class MemoryAttacker
{
  public void foo()
  {
    List x = new LinkedList();
    while (true)
      x.add("abc");
  }
}
```

DoS on memory management:
Excessive heap allocation

Security in Java: Problems

- Denial-of-service attacks

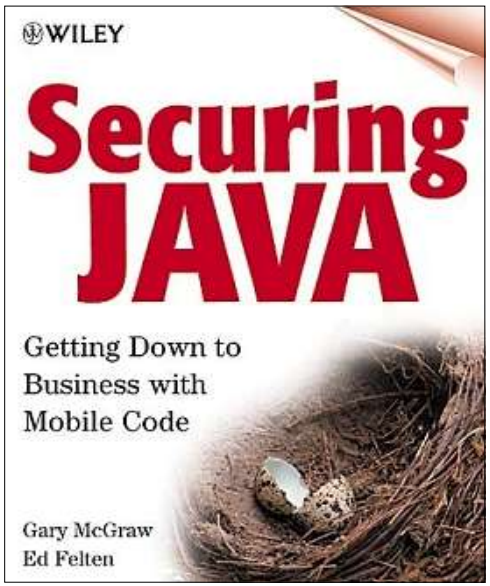
```
class ClassLoaderAttacker
{
  public void attack()
  {
    synchronized (foo
      .getClass()
      .getClassLoader())
    {
      while (true)
        ;
    }
  }
}
```

DoS on Virtual Machine:
Deadlock upon class loading

```
class FinalizationAttacker
{
  public void finalize()
  {
    while (true)
      ;
  }
}
```

DoS on class finalization thread

Security in Java: Further reading

- 

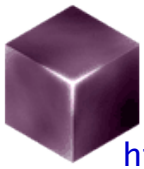
Gary McGraw and Ed Felten. Securing Java: Getting Down to Business with Mobile Code. John Wiley & Sons, 1999. 324 pages.

Also available online at http://www.securingjava.com/

- ECOOP '02 Workshop on Resource Management for Safe Languages

http://www.ovmj.org/workshops/resman/

Quality assurance



http://sources.redhat.com/mauve/

- Test suite
  - ~100,000 tests
  - We need many more!
- Key to reliability
- Tests are very easy to write

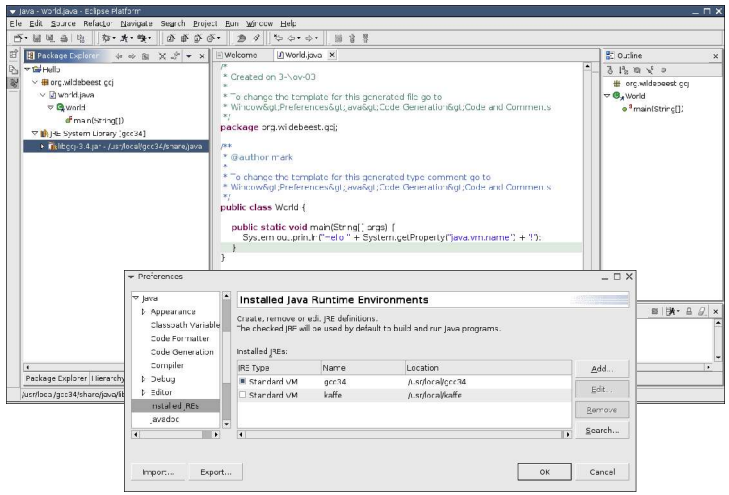
```
// Tags: JDK1.0
package gnu.testlet.java.util.Stack;

import java.util.Stack;
import gnu.testlet.*;

public class empty
  extends Testlet
{
  public void test(TestHarness h)
  {
    Stack stack = new Stack();

    // Check #1.
    h.check(stack.empty());

    // Check #2.
    stack.push("abc");
    h.check(!stack.empty());
  }
}
```

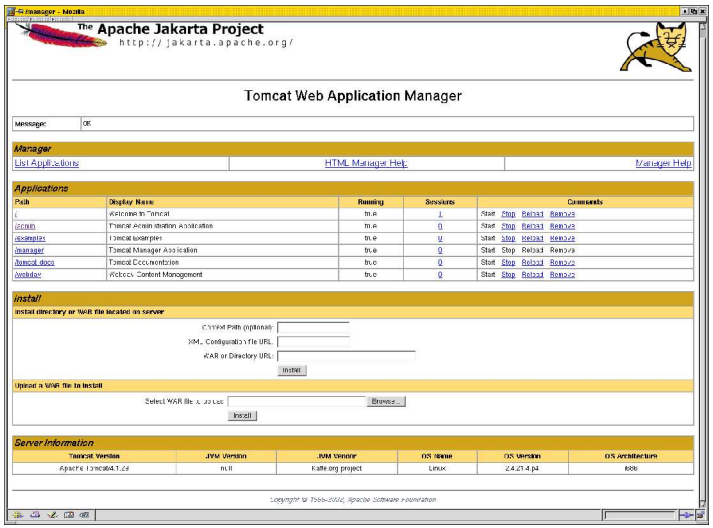
Test for java.util.Stack.empty()

Releases

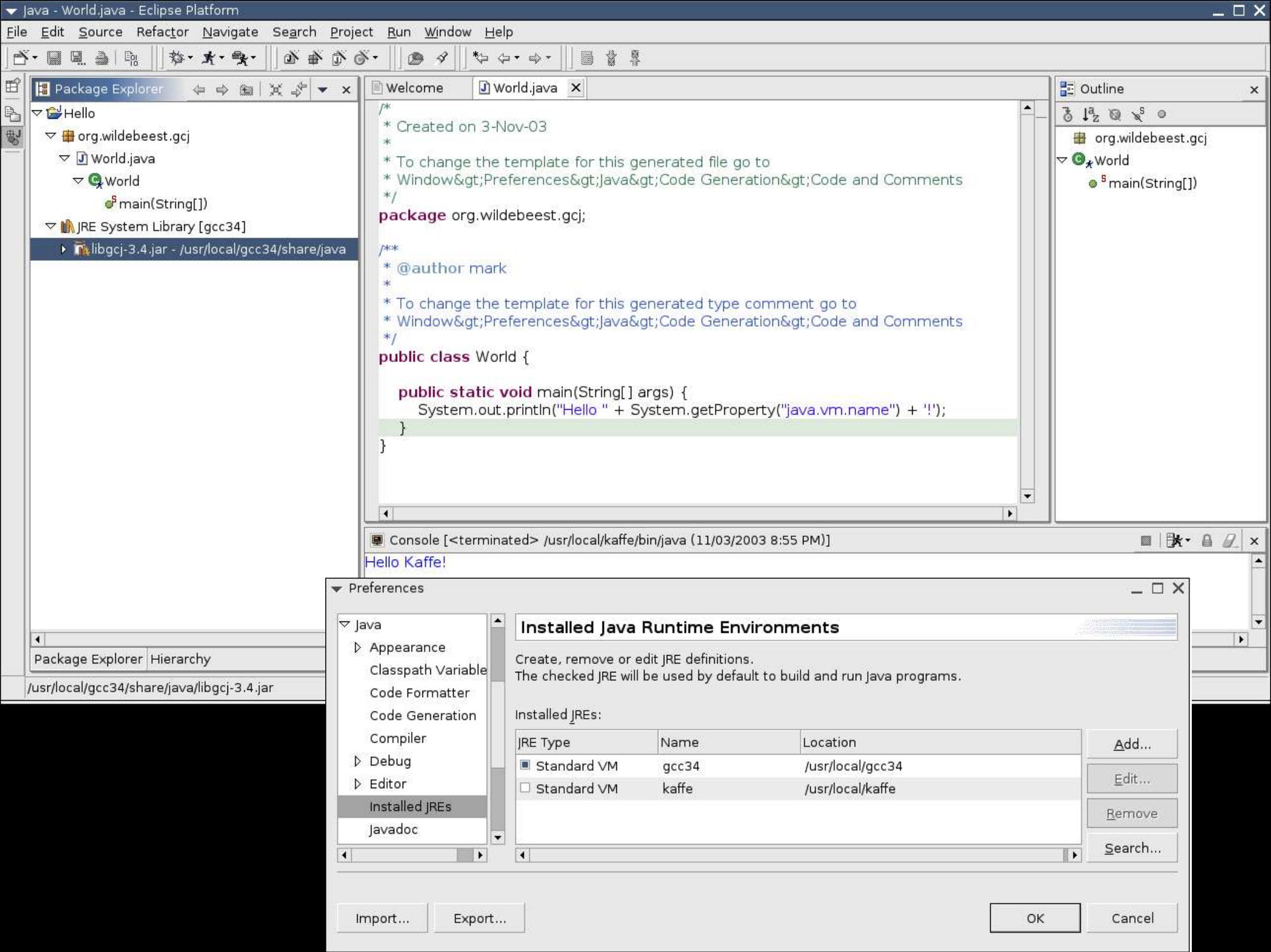
- Now: Version 0.08
- Time-based release schedule
  - Every three months, a new 0.0x release
- Already quite usable for real applications
  - “0.08” may be too modest a name
- For v1.0, we need:
  - Fixed VM interface
  - Complete implementation of supported packages
    - Define “supported”!
  - Full test coverage
  - Full documentation of the API
    - Start of a real manual?

What's working?





- Eclipse IDE
  - UI: SWT (not AWT)
  - eclipse.org
- Tomcat
  - Container for Servlets and JavaServer Pages
  - jakarta.apache.org


Java - World.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Hello
org.wildebeest.gcj
World.java
World
main(String[])
JRE System Library [gcc34]
libgcj-3.4.jar - /usr/local/gcc34/share/java
Welcome
World.java
/*
* Created on 3-Nov-03
*
* To change the template for this generated file go to
* Window>Preferences>Java>Code Generation>Code and Comments
*/
package org.wildebeest.gcj;
/**
* @author mark
*
* To change the template for this generated type comment go to
* Window>Preferences>Java>Code Generation>Code and Comments
*/
public class World {
public static void main(String[] args) {
System.out.println("Hello " + System.getProperty("java.vm.name") + '!');
}
}
Outline
org.wildebeest.gcj
World
main(String[])
Console [<terminated> /usr/local/kaffe/bin/java (11/03/2003 8:55 PM)]
Hello Kaffe!
Package Explorer
Hierarchy
/usr/local/gcc34/share/java/libgcj-3.4.jar
Preferences
Java
Appearance
Classpath Variable
Code Formatter
Code Generation
Compiler
Debug
Editor
Installed JREs
Javadoc
Installed Java Runtime Environments
Create, remove or edit JRE definitions.
The checked JRE will be used by default to build and run Java programs.
Installed JREs:
JRE Type
Name
Location
Standard VM
gcc34
/usr/local/gcc34
Standard VM
kaffe
/usr/local/kaffe
Add...
Edit...
Remove
Search...
Import...
Export...
OK
Cancel

/manager - Mozilla

The Apache Jakarta Project
http://jakarta.apache.org/

Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	1	Start Stop Reload Remove
/admin	Tomcat Administration Application	true	0	Start Stop Reload Remove
/examples	Tomcat Examples	true	0	Start Stop Reload Remove
/manager	Tomcat Manager Application	true	0	Start Stop Reload Remove
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Remove
/webdav	Webdav Content Management	true	0	Start Stop Reload Remove

Install

Install directory or WAR file located on server

Context Path (optional):

XML Configuration file URL:

WAR or Directory URL:

Install

Upload a WAR file to install

Select WAR file to upload Browse...

Install

Server Information

Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture
Apache Tomcat/4.1.29	null	Kaffe.org project	Linux	2.4.21.4.p4	i686

Current state

- J2SE 1.3/1.4 (Desktop)
  - Source: GNU Classpath
  - License: GPL + “Compiler exception”
    - Allows static linking with proprietary closed-source code and into embedded systems → very permissive
  - No formal compliance with any specification

java.beans[.beancontext], java.io, java.lang[.ref, .reflect], java.math, java.net, java.nio.charset[.spi] (only SPI), java.rmi[.activation, .dgc, .registry, .server], java.security[.acl, .cert, .interfaces, .spec], java.sql, java.text, java.util[.jar, .logging, .zip]

javax.naming[.directory, .ldap, .spi], javax.swing.[border, plaf], javax.transaction[.xa]

Current state

- J2SE 1.4 (Desktop), partial implementation

java.applet, java.awt, java.awt.event, java.awt.geom,
java.awt.image[.renderable], java.nio[.channels, .spi], java.util.prefs
javax.accessibility, javax.print[.*] (only SPI), javax.swing.undo

- J2SE 1.4 (Desktop), lots of work/nothing there

java.awt.color, java.awt.font, java.awt.im[.spi],
java.nio.charset (service providers)

javax.imageio[.*], javax.print (service providers), javax.rmi[.CORBA],
java.security.auth[.*], javax.swing[.colorchooser, .event,
.filechooser, .plaf.basic, .plaf.metal, .plaf.multi, .table,
.text, .text.html, .text.html.parser, .text.rtf, .tree]

org.ietf.jgss, org.omg[.*]

Current state

- Various standard packages
  - Licenses: Various (but all “free”)
  - No formal compliance with any specification

javax.crypto
Service providers for many crypto algorithms
GNU Crypto

java.util.regex
GNU regex

javax.sound
Service providers for ALSA, esd, lame, ogg
Tritonus Project

javax.net
javax.net.ssl
javax.security.cert
SSL/TLS Support
Jessie

javax.activation
javax.infobus
javax.mail
javax.servlet
GNU Classpath Extensions

javax.speech
Text-to-Speech (no recognition)
FreeTTS

Current state

- J2EE 1.4 (Server)
  - Multi-tier stack: web services/enterprise computing
  - Multiple free implementations: JBoss.org, Jonas
  - Needs some (not much) work to combine it all
  - Conformance: JBoss.org has recently licensed the J2EE1.4 compliance kit from Sun microsystems
    - Now making sure that JBoss.org satisfies the test suite
    - Will afterwards be certified as compliant

javax.activation, javax.ejb[.spi], javax.enterprise[.*], javax.jms, javax.mail[.*], javax.management[.*], javax.resource[.*], javax.security.jacc, javax.servlet[.*], javax.transaction[.xa], javax.xml.*

Current state

- Plenty of free software is being written in Java
  - sourceforge.net: 11,032 Java projects
  - Support for multiple other programming languages
    - GNU KAWA: Scheme, Common Lisp, eLisp, ECMAScript, ...
    - Jython, JRuby
- Traditional free libraries with Java bindings
  - GTK+, GNOME, glade, gconf, vte, ...
    - Will be part of the official language bindings in GNOME 2.6
  - GNU gettext, getopt, readline, ...

How you can help

- Any help is greatly appreciated
  - Test Java programs on free Virtual Machines
  - Write test cases for Mauve
  - Write intellegible documentation
  - Implement library classes
  - Fund development
- How to proceed
  - Look at the task list
    - http://www.gnu.org/software/classpath/
    - Both easy and tricky tasks
  - Ask us for details: classpath@gnu.org
  - Copyrights must get assigned to FSF
  - Happy hacking!

How you can help with security

- Security in Classpath
  - We have 90% of the Java security framework
    - ... and need **you** for the second 90%!
  - Code audit
    - do we call checkPermission() where needed?
  - GNU Crypto
- Security in VMs
  - Type safety proved by bytecode verification
    - VM-specific
    - no free VMs do it yet in full
  - Code audit