Fedora Core, Java™ and You

Gary Benson

Software Engineer

What is Java?

The word “Java” is used to describe three things:

- The Java programming language
- The Java virtual machine
- The Java platform

To support Java applications Fedora needs all three.

What Fedora uses: GCJ and ECJ

GCJ is the core of Fedora's Java support:

- GCJ includes `gcj`, a compiler for the Java programming language.
- GCJ also has a runtime and class library, collectively called libgcj.
- The class library is separately known as GNU Classpath.

ECJ is the Eclipse Compiler for Java:

- GCJ's compiler `gcj` is not used for “traditional” Java compilation.
- More on that later...

Why libgcj?

There are many free Java Virtual machines:

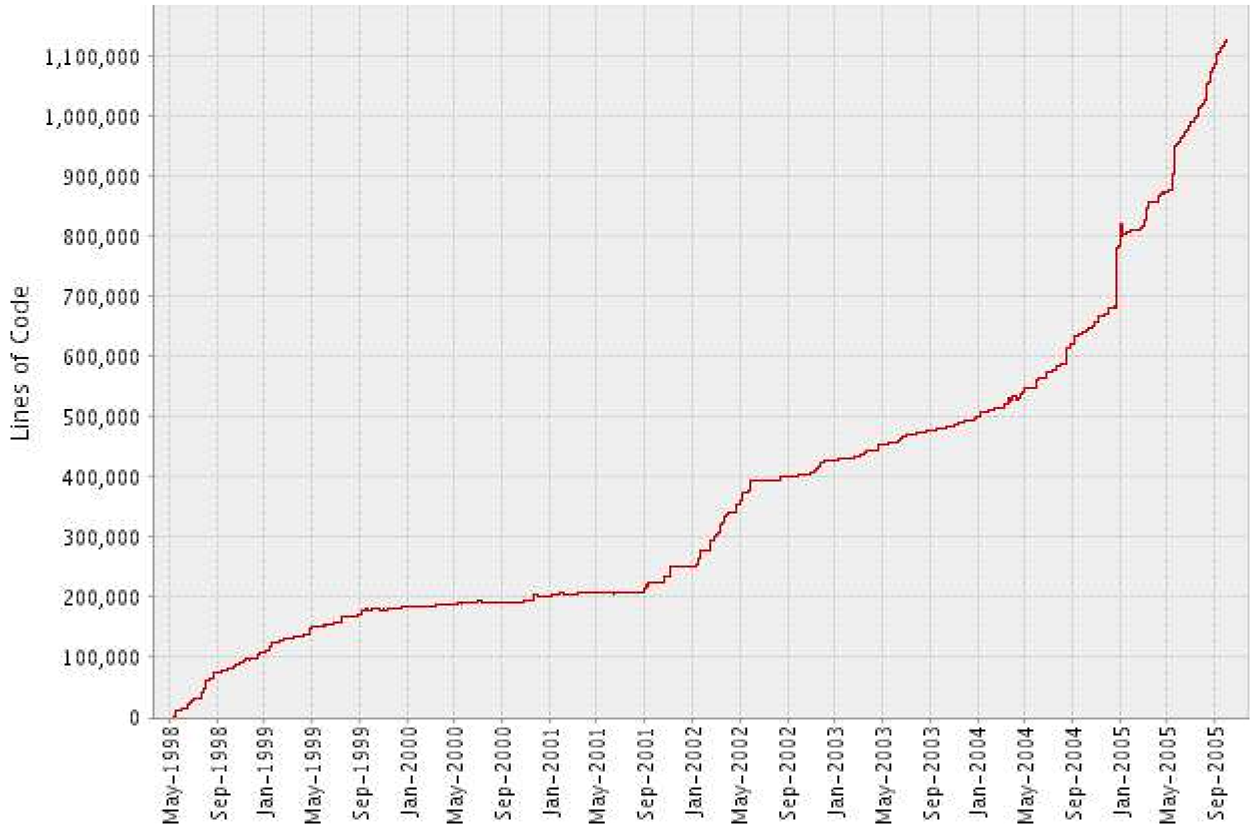
- Cacao, IKVM, JamVM, Jikes RVM, Kaffe, libgcj, Sable VM, ...

There are two main reasons Fedora uses libgcj:

- Availability on many platforms.
- Ability to use precompiled native code.

GNU Classpath

Free core class library for Java virtual machines and compilers.



The JPackage Project

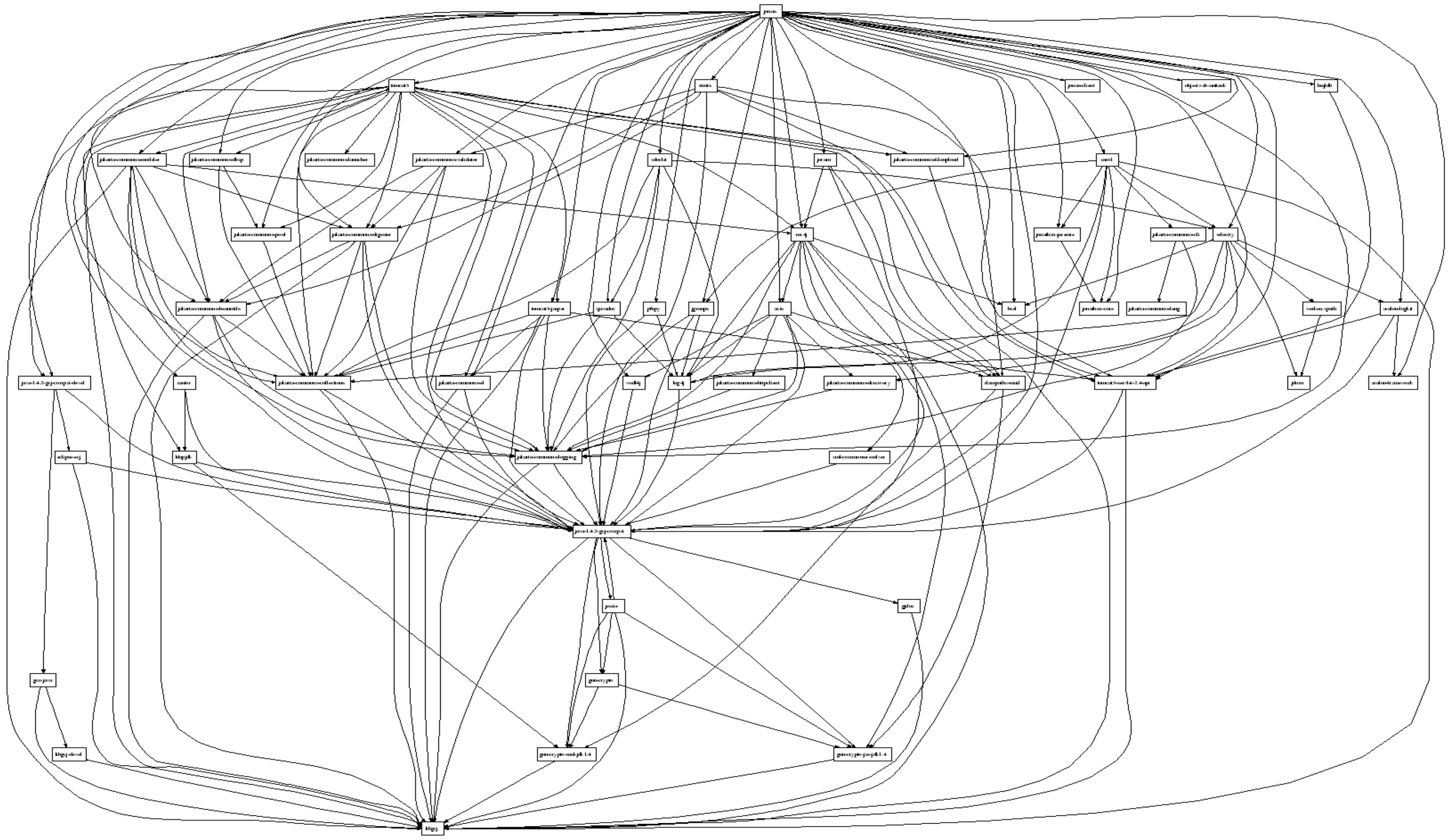
A collection of some 1,600 Java software packages for Linux:

- Distribution-agnostic RPM packages.
- Both runtimes/development kits and applications.
- Segregation between free and non-free packages.
- All free packages built entirely from source.
- Multiple runtimes/development kits may be installed.

Fedora includes:

- JPackage-compatible runtime and development kit packages.
- A whole bunch of applications.

JPackage JOnAS



Fedora's Java Compilers

`gcj` can operate in several modes:

- Java source (`.java`) to Java bytecode (`.class`)
- Java source (`.java`) to native machine code (`.o`)
- Java bytecode (`.class`, `.jar`) to native machine code (`.o`)

In Fedora:

- ECJ compiles Java source to bytecode.
- `gcj` compiles that bytecode to native machine code.

But, how does the native code fit in?

The original ABI

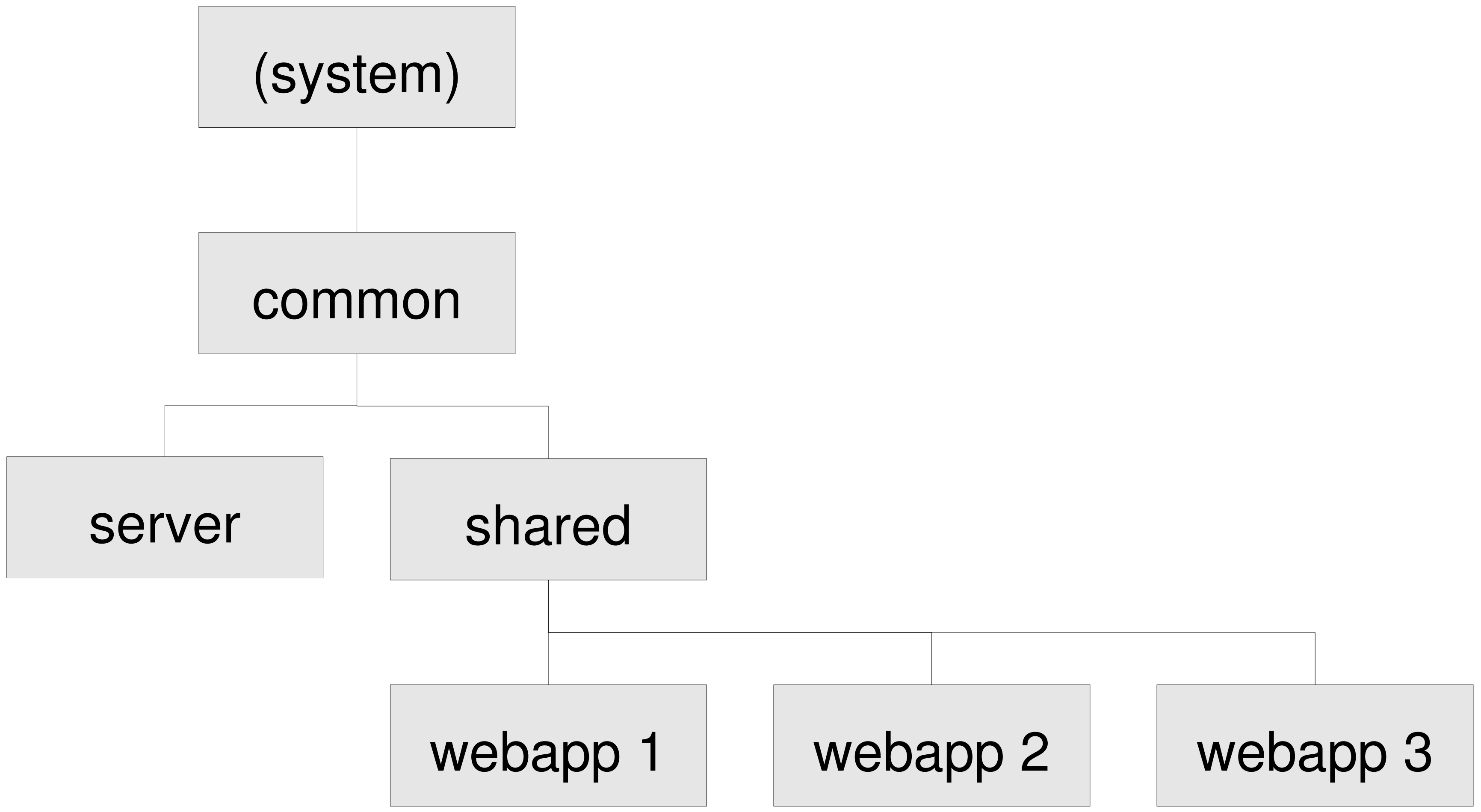
At first, native Java libraries were much the same as C++ ones:

- Standard parameter passing mechanisms used.
- Static fields and methods resolved by the linker.
- Instance fields looked up by offset.
- Instance methods looked up in vtable.

Class names were transformed to locate libraries:

- To find `org.apache.tools.ant.Main`:
    - check `/usr/lib/lib-org-apache-tools-ant-Main.so`
    - check `/usr/lib/lib-org-apache-tools-ant.so`
    - check `/usr/lib/lib-org-apache-tools.so`
    - ...etc

Tomcat's class loaders



Binary Compatibility

Chapter 13 of *“The Java Language Specification”:*

- Methods may be added
- Fields may be added
- Inheritance tree may change

The C++ linking model is not suitable:

- Fixed offsets
- Fixed object sizes
- Fixed inheritance tree

Supporting class loaders

At build time, for each class:

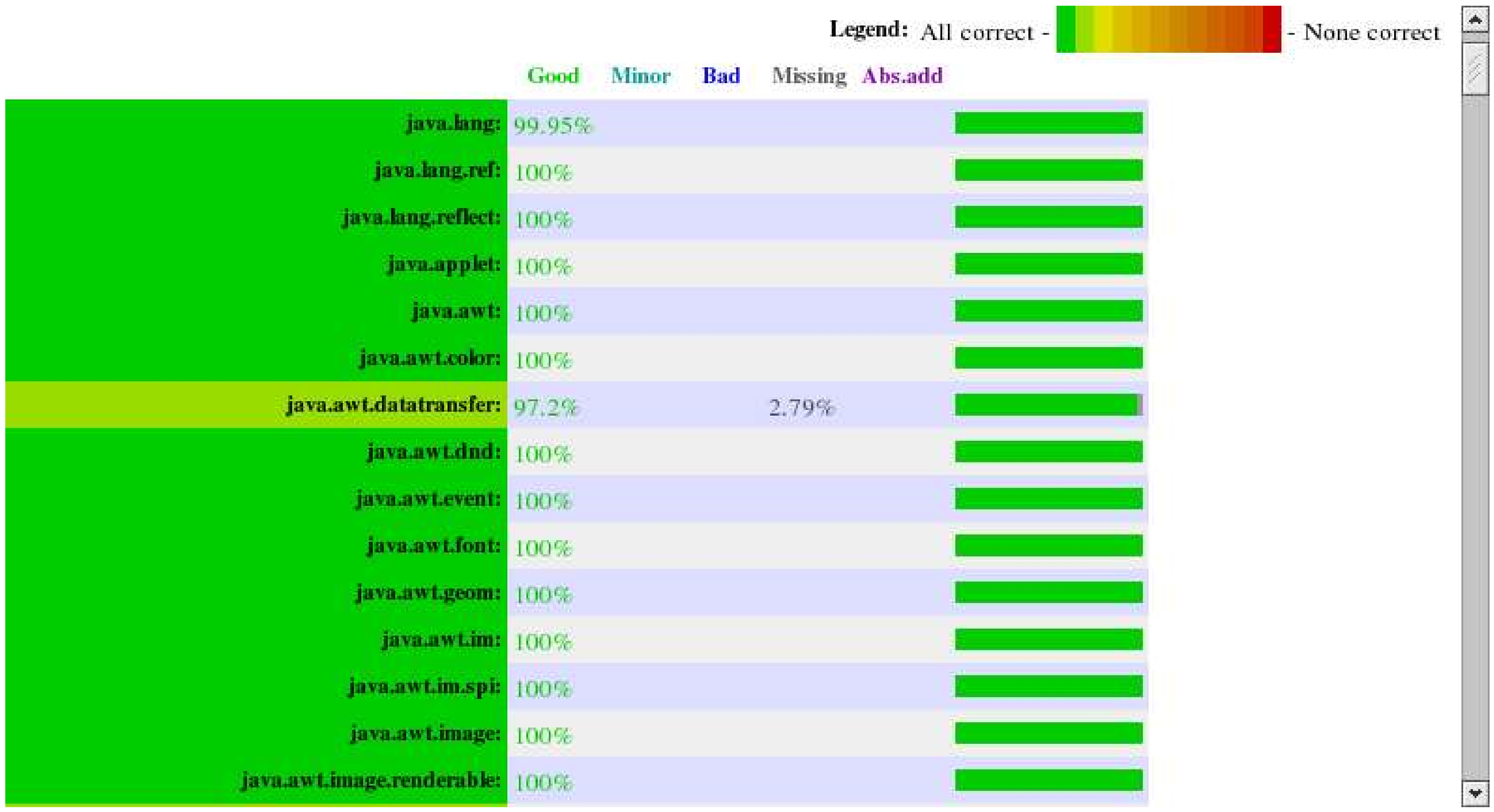
- Generate MD5 digest of bytes.
- Store MD5 → shared library mapping in database.

Whenever `ClassLoader.defineClass(byte[])` is called:

- Generate MD5 digest of bytes.
    - If found in database, load corresponding shared library.
    - If not in database, interpret.

This precisely matches the model used by other JVMs.

Future work: API coverage



Legend: All correct - 🟩🟥 - None correct

	Good	Minor	Bad	Missing	Abs.add
java.lang:	99.95%				
java.lang.ref:	100%				
java.lang.reflect:	100%				
java.applet:	100%				
java.awt:	100%				
java.awt.color:	100%				
java.awt.datatransfer:	97.2%			2.79%	
java.awt.dnd:	100%				
java.awt.event:	100%				
java.awt.font:	100%				
java.awt.geom:	100%				
java.awt.im:	100%				
java.awt.im.spi:	100%				
java.awt.image:	100%				
java.awt.image.renderable:	100%				

Future work: performance

libgcj has lost much of its speed advantage:

- JITs have improved.
- The new ABI hurts performance.

It also uses too much memory.

Now we can run real applications we can profile:

- Many useful optimizations have already identified.
- Improving IO libraries should pay dividends.
- Some methods could be rewritten in C++.
- Lack of JIT not expected to be a particular problem.

Other future work

Security audit:

- Sandbox
- Security manager
- gcjwebplugin

Support for 1.5:

- Virtual machine
- Platform
- Compiler

Summary

Fedora includes a free Java implementation that can run huge applications.

If you'd like to use it but are having problems:

- fedora-devel-java-list@redhat.com
- #fedora-java on irc.freenode.net