Generics and Java 1.4 with one codebase

By James Lemieux and Jesse Wilson

Glazed Lists was in a unique position with the introduction of generics in JDK 1.5. While Generics make sense almost anywhere in a codebase, it is extremely appropriate for use with the Collections API. Glazed Lists sits atop java.util.List and so we were bound almost by moral fiber to support generics. The project however, has an army of users that aren't moving off of JDK 1.4, the platform on which Glazed Lists was born.

Our options were limited:

- Fork the codebase for each version of the source we are interested in publishing and repeat all development and bug fixes to all codebases.
- Introduce runtime dependency on Retroweaver. This library supports Java 1.5 language features, while still retaining total binary compatibility with 1.4 virtual machines.
- Find a clever way to downgrade JDK 1.5 source to JDK 1.4 source

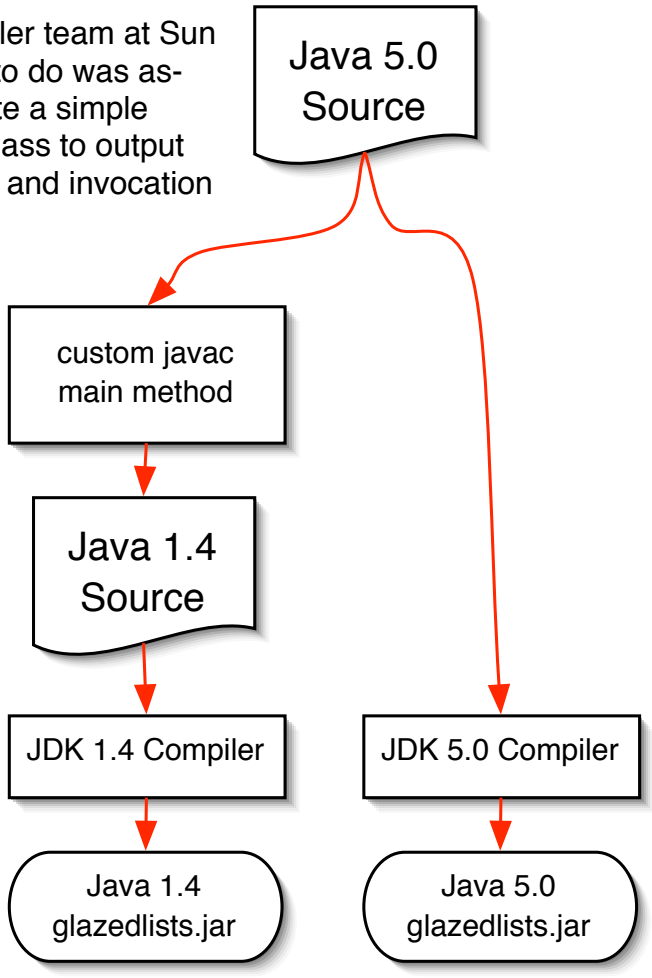
Being devout engineers in quest of “the best way,” we vowed to exhaust all attempts to keep a single codebase free of external dependencies before relying on alternatives.

To succeed in converting JDK 1.5 to 1.4 source it would have to be at the parse tree level. Generics are implemented by the compiler rendering source code structurally equivalent between the two versions.

The Java language is wrought with complexity, which makes source code translation tools very difficult to write. The lack of an open source tool for Java language manipulation is confirmed by this fact. Even parsing source code to a tree and writing that back out as source was an extremely difficult task. Further searching turned up a solution in the most unlikely of places: our own backyard.

Inspecting tools.jar which ships with the JDK is like opening all of the Christmas presents you'll ever get at once. It includes the java compiler, its parser, and many other tools for language manipulation including something called a TreeTranslator. This visitor traverses the parse tree and makes uniform changes to it. Our luck didn't end there as there is a subclass of TreeTranslator called Lower which is responsible for removing JDK 1.5 syntactic sugar (generics, inner classes, class literals, assertions, foreach loops, etc.)

We couldn't believe our good fortune! The compiler team at Sun had already completed our work and all we had to do was assemble the tools that they had given us. We wrote a simple wrapper around JavaCompiler that caused the class to output source code rather than bytecode. Configuration and invocation of JavaCompiler required only 10 lines of code.



With Ant and our new generics-stripping tool, we can use our Java 1.5 source tree to create a Java 1.5 .jar and a Java 1.4 .jar. By converting to source first, we can compile using JDK 1.4. This guarantees that we have no dependencies on classes or methods found only in Java 1.5.

This solution works well for us but it has some limitations:

- The source code emitter in tools.jar doesn't support anonymous inner classes or overridden return types
- We don't get to use the Java 1.5 class library such as the new concurrency package or Formattable interface

See also:

Glazed Lists: http://publicobject.com/glazedlists/

Antlr: http://www.antlr.org

Retroweaver: http://retroweaver.sf.net/

Java 1.5 Generics Tutorial: http://java.sun.com/j2se/1.5/pdf/generics-tutorial.pdf