

gentoo

Gentoo



www.gentoo.org

Terminology



- OBS!
- This representation has Gentoo terminology in it.
 - I will explain where necessary.

Some Gentoo ideals



- Source based
 - Java is no exception
- Choice
 - Leave it up to the user to know what he wants
 - Provide sensible defaults

Gentoo Java Team



- 9 developers in the java herd
 - Only a couple of active ones at the moment
- Lead by Karl Trygve Kalleberg
- Active users helping
 - Some eventually become developers
- ~350 packages maintained by the java herd
- ~200 open bugs (counting java-related dev-util stuff, etc)
- ~140 experimental packages

Why build java from source?



- Users may want to manually patch or tweak the sources between `src_unpack` and `src_compile`.
- The USE flags to control optional features.
- When security flaws are found, we want to issue a hotfix immediately. It is not always feasible to wait for the upstream project to make a new release. With binary-only packages, the only fix we can offer is disabling the software entirely and masking it.
- We guarantee to our users that the source exists and compiles properly, the basis of regular open-source development.

Continued



- Gentoo-specific tweaks and intermediary patches when upstream takes a long time to issue a new release.
- There is an obvious correspondence between the source code and the resulting binaries, so the user is guaranteed that there are no trojans hiding in the binaries (we don't guarantee the lack of trojans in the source code itself, but at least it's easier to find there).
- Users or ebuilds can run javadoc on the source code.

trees



- ~10000 packages in Portage at the moment
 - ~350 maintained by the java herd
- Experimental tree for testing stuff
- Other existing overlays
 - Eclipse
 - free-java-overlay
 - axxo-overlay (Support for Java 1.5)

What we provide



- We provide Sun, Blackdown, JRockit, IBM, Compaq, Kaffe, Kissme, JamVM, SableVM, gcj
- We provide most common libraries: Xalan, Junit, commons-*, JMS, JMX, log4j, wsdl4j, ...
- We provide development tools: Jikes, Eclipse, NetBeans,
- Jython, JRuby, JRexx, AspectJ, Maven, Ant, Scala, ...

Where are we now?



- Blackdown is the default
- User can choose what he wants
 - No official support for every jdk
 - Other arches have different defaults
 - Virtual jdk and jre packages
- 1.5 only supported as the user jdk and system jre

Where are we going



- Robust compilation of all packages with 1.5
 - Switching the jdk while emerging
 - build.xml rewriting
- Requires the new improved java-config tool
- A totally free java environment
 - Ecj will become the default compiler
- We have few requests for GCJ; consider a thing of the future.

Problems



- Packed jars
 - Only one copy of a jar can be installed at any time
 - The actual version of libraries a project is compiled against is sometimes extremely difficult to determine. Many projects take a CVS snapshot of another project, compile it and put in a .jar with no version information.
- The binary compatibility between releases is very bad
- Bootstrapping

More ...



- Many open-source projects depend heavily on closed-source libraries
 - Netbeans
- Java build systems can be pathetic
- Source is not always available separately from the project.

Questions?



- Introduction to a java ebuild if wanted