

JRuby

A Ruby VM in Java

jruby.sourceforge.net

Charles Oliver Nutter, presenting

Who Am I?

- Charles Oliver Nutter: headius@headius.com
- Senior Architect/Technologist at Ventera Corp (gov't, financial, telecom contracting; www.ventera.com)
 - Open source initiatives and architecture team
 - Architect and dev lead for Java-based gov't contracts
- Closed Source:
 - Windows utilities (Hack-It, Deity; www.headius.com)
 - STARS: J2EE app to manage nationwide Food Stamp Program data (Ventera)
- Open Source:
 - LiteStep lead 1998-2000
 - JRuby developer 2004-present, now leading redesign
 - Miscellaneous half-efforts and patch submissions

Part 1: Past and Present

- What Is JRuby?
- Why JRuby?
- Peer Comparison
- The Real World
- How “Ruby” Is It?
- Demos along the way

What is JRuby?

- A “100% Java”-based Ruby Interpreter
- Mostly 1.8 compatible
- Four years and 15 developers; currently 3-5 active and under heavy development
- Originally modeled on Ruby 1.6 code
- Tri-licensed: CPL, GPL, LGPL
- Sun J2SE 1.4 or higher (FOSS Javas in future)
- Java/Ruby integration getting better

Why JRuby?

- JVM provides native threading, generational GC, and extensive networking and database support
- Wealth of libraries and frameworks; large userbase, wide deployment
- Many Javaists would like to use Ruby more
- Java is “just another platform” for Ruby
- JRuby could help grow Ruby the language apart from C Ruby
- Sun, others very interested in dynamic (typed) languages on the JVM
- Javaists (by choice or by force) can help promote Ruby
- Java/Ruby integration merges best of both
- Ruby + J2EE = enterprise Ruby that managers can swallow
- Ruby + J2ME...someday?

Demo 1: Java Integration

- Ruby code mixed into Java class's proxy
 - Enumerable, list behavior added to `java.sql.ResultSet`
- JDBC used for DB access (PostgreSQL 8.0)
- Mostly transparent object marshalling

Peers

- Jython – www.jython.org
 - Pythonists dig it
 - By far the most popular JVM dyn lang
 - Established, stable, feature-complete
 - Helping to formally define/distinguish Python the language from Python the interpreter
 - Interpreted or compiled (runs Python bytecode, or compiles to Java)
 - Widely used
- Groovy – groovy.codehaus.org
 - Ruby-like syntax, some features from Nice
 - Seamless Java integration, sometimes with perf hit
 - First dyn language JSR
 - Lots of momentum
 - Interpreted (JIT) or compiled offline to Java code
- SISC – sisc.sourceforge.net
 - JRuby redesign follows similar patterns
- Many others

The Real World

- RDT: A Ruby IDE for Eclipse;
rubyclipse.sourceforge.net
- jEdit: A Multi-language Code Editor;
www.jedit.org
- DataVision: Java-based Reporting Software;
datavision.sourceforge.net
- Internal projects
- Need more

How “Ruby” is it?

- Of 1049 Rubicon tests, 80% succeed
- Temporary incompatibilities
 - Ruby thread semantics differ from Java’s
 - No continuations
 - Twice as slow (half as fast?) as C Ruby or worse
 - YAML: no up-to-date, working pure Ruby or Java parsers
 - Still missing a few 1.8 methods
- Permanent incompatibilities
 - System calls, C-language Ruby extensions, anything to do with C
 - Platform-specifics: file stats, permissions, process launching, signals, ...

Part Two: The Future

- Continuing to improve compatibility
- Running mainstream Ruby apps
- Improving Java integration
- Speeding up

- **The New JRuby**

What needs to change?

- JRuby deficiencies (as of 0.8.2):
 - Stack depth ($\sim \text{fib}(280)$)
 - Threading and thread semantics
 - Continuation support
 - Speed
 - Consistency, maintainability
 - Compilation
 - Better use of Java's strengths
 - Tighter integration between Java and Ruby
- Ruby deficiencies (as of 1.8):
 - Stack depth ($\sim \text{fib}(1325)$)
 - Native threading
 - Speed
 - Compilation

The New JRuby

- Stackless; Continuation Passing Style (roughly)
- Iterative interpreter
- m:n threading model
- Compilation to Java bytecodes, offline and JIT
- Pluggable architecture
- Seamless, powerful Ruby/Java integration
- Behave in controlled environments
- FAST

Milestones and progress

- Stackless, iterative proof of concept (POC) (Sept 15, complete)
- Redesign, refactoring of POC (Oct)
- Reimpl of interpreter based on POC (Nov)
- Reimpl of built-in classes (Nov-Jan)
- Threading engine (Jan)
- Tail-call optimization (Jan)
- Continuations (Jan)
- Compilation (Feb – Apr)
- Complete for JavaOne 2006

Demo 2: Fibonacci

- Recursive fib algorithm (contrived, I know)
- JRuby 0.8.2: shallow
- Ruby: deeper
- JRuby “stackless” POC: deepest

What Else?

- YARV bytecode execution
- MetaRuby's “Ruby in Ruby” useful to JRuby
- drb proxy to RMI
- ActiveRecord JDBC connector
- WEBrick-mimicking servlets
- Other ideas?

Part Three: What now?

- Redesign is in full swing
- Heavy refactoring of JRuby core
- A better Ruby than ruby?
- Help Wanted!
 - zlib implementation using Ruby-Java integration
 - File locking using Java's NIO (New I/O)
 - Feature-complete YAML support
 - Running mainstream Ruby apps, isolating and reporting errors
 - Help with new design and with refactoring effort
 - Tangibles

Q&A

- JRuby: jruby.sourceforge.net
SF project page: sourceforge.net/projects/jruby
- JRuby mailing lists on SF
- Charles Oliver Nutter: headius@headius.com
- Thanks to:
 - Thomas Enebo: JRuby project manager
 - Kelly Nawrocke: JRuby developer
 - David Corbin: JRuby developer, RDT developer
 - Special thanks to Jan Arne Petersen, original JRubyist