

# JRuby: Up and Running!

**Charles Oliver Nutter and  
Thomas Enebo**

The JRuby Guys

Sun Microsystems



Except where otherwise noted, the content of this presentation is licensed under the Creative Commons Attribution-Share Alike 3.0 United States License (<http://creativecommons.org/licenses/by-sa/3.0/us/>).

# Agenda

- JRuby overview
- Java is not a four-letter word
- Real-world JRuby
  - > Swing programming
  - > Tools and IDE support
  - > Graphics and applets (that don't suck)
  - > Web applications
- Interactive: what do you want to know?

# The JRuby Guys

- Charles Oliver Nutter and Thomas Enebo
- Longtime Java developers (10+ yrs each)
- Engineers at Sun Microsystems for 1 yr
- Full-time JRuby developers
- Also working on JVM dynlang support
- Wide range of past experience
  - > C, C++, C#, Perl, Python, Delphi, Lisp, Scheme
  - > Java EE and ME, JINI, WS

# JRuby

- Java implementation of Ruby language
  - > “It's just Ruby!”
- Started in 2002, open source, many contributors
  - > Ola Bini, Marcin Mielzynsky, Nick Sieger, Bill Dortch, Vladimir Sizikov, MenTaLguY
- Aiming for compatibility with current Ruby version
  - > Ruby 1.8.6 patchlevel 111 (114 was just released)
- Improvements on Ruby
  - > Native threading, better performance, many libraries

# JRuby 1.1 Released!

- JRuby 1.1 is finished!
- Massive improvements over 1.0
  - > Full compiler for Ruby code to JVM bytecode
  - > Performance is many times better across the board
  - > New Regexp impl with full Oniguruma features
  - > New rewritten IO subsystem to parallel Ruby behavior
  - > Reduced memory footprint
  - > Best compatibility level ever
- Ready for production use (and already being used)
- Quick series of maintenance releases coming

# Compatibility

- Applications are king
  - > Rails, Rubygems, Rake, Rspec, ...
- Testing rulez (~42,000 expectations/assertions)
  - > Prevents regressions
  - > Helps to better define Ruby
- Prevents fragmenting a community
  - > Sapphire

# Java == A Dirty Word

# Java == A Dirty Word

- I don't like the Java Language

# Java == A Dirty Word

- I had a poor experience with
  - > A Java application
  - > A Java library
  - > Startup Time
  - > Performance

# Java == A Dirty Word

- Poor Advocacy
  - > “The answer is Java. What is the Question?”

# Java != A Dirty Word

# Java != A Dirty Word

- Fantastic Virtual Machine
  - > Tuned for over a decade by an army
  - > Runs on virtually all os/hardware combos
  - > Dynamic optimizations (Hotspot)
  - > Keeps getting faster:

	Java 5	Java 6	
Rexml	10.9s	7.41s	%32
Hpricot	4.06s	2.59s	%36

# DEMO

# JRuby Performance

# Java != A Dirty Word

- Fantastic Garbage Collectors
  - > Compacting
  - > Concurrent
  - > Many tunables and choices

# Java != A Dirty Word

- Threading
- Tools
  - > IDEs(refactoring, debugging)
  - > Profilers (instrumenting, sampling)
  - > JMX (ask VM for stats)
- Libraries
  - > Anything you can think of...
  - > Write `image_science` in 60 lines of Ruby using Java 2D

# **DEMO**

# **NetBeans Profiler**

# Where is JRuby being used?

- Swing GUI development
  - > Makes Swing much nicer to use, easier to handle
- Tooling for IDEs
  - > JRuby's parser enables NetBeans, Eclipse, IntelliJ
- Graphics
  - > Ruby + graphics = cool demos
- **JRuby on Rails**
  - > **Better deployment options, better performance**

# Swing GUI Programming

- Swing API is very large, complex
  - > Ruby magic simplifies most of the tricky bits
- Java is a very verbose language
  - > Ruby makes Swing actually fun
- No consistent cross-platform GUI library for Ruby
  - > Swing works everywhere Java does (everywhere)
- No fire-and-forget execution
  - > No dependencies: any script works on any JRuby install

# DEMO

## Swing in Ruby

# Swing Options

- Cheri - [cheri.rubyforge.org](http://cheri.rubyforge.org)
  - > Builder-like DSL syntax
- Profligacy - [ihate.rubyforge.org/profligacy](http://ihate.rubyforge.org/profligacy)
  - > Rubified layout expression language
  - > Trivial event binding without listeners
- MonkeyBars - [monkeybars.rubyforge.org](http://monkeybars.rubyforge.org)
  - > Leverage GUI builders
  - > MVC structure

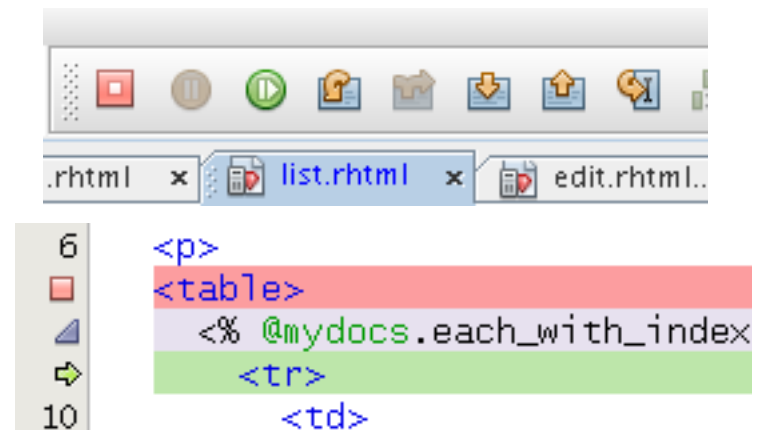
# JRuby Enables Tooling

- JRuby's parser used by most Ruby IDEs
  - > NetBeans Ruby Support
  - > Eclipse RDT/RadRails/Aptana, DLTK, 3<sup>rd</sup> Rail
  - > IntelliJ (not parser, but other areas)
  - > Jedit
- NetBeans is the best right now

```
class Post < ActiveRecord::Migration
  def self.up
    create_table(name, options) ActiveRecord
    create_
  end
end
```

*SchemaStatements.create\_table(name, opt:*

Creates a new table There are two ways to work with it:  
use the block form or the regular form, like this:



The screenshot shows a web browser window with three tabs: ".rhtml", "list.rhtml", and "edit.rhtml..". The "list.rhtml" tab is active and displays a table. The table has a header row with a red background and a body row with a green background. The code for the table is as follows:

```
6 <p>
7 <table>
8   <% @mydocs.each_with_index
9     <tr>
10       <td>
```

# **DEMO**

## **NetBeans Ruby and Rails**

# Pretty Graphics!

- “Processing is an open source programming language and environment for people who want to program images, animation, and interactions.”
  - > Basically a cool Java library for 2D graphics
- Ruby-Processing wraps Processing with JRuby
  - > Cool, rubified 2D graphics environment for you
  - > Eye-candy demos for us
  - > Thanks to Jeremy Ashkenas for putting these together

# **DEMO**

# **Pretty Graphics!**

# Web applications

- Classic Java web dev is too complicated
  - > Modern frameworks follow Rails' lead
- Over-flexible, over-configured
  - > Conventions trump repetition and configuration
- Rails deployment is still a pain
  - > You shouldn't need N processes!
- Rails performance should be better
  - > JRuby has potential to be much faster

# So, Why JRuby?

- Easier setup
- Better performance
- Easier deployment
- More libraries
- Wider database support
- Platform independence
  - > JRuby on Rails on AS/400 calling MS SQL? No prob!
- Less political resistance

# JRuby Setup

- Download
- Unpack

# JRuby Setup (long version)

- Java installation (if necessary)
- Download JRuby
  - > Includes JRuby, Ruby stdlib, rake and rspec
- Unpack JRuby
  - > Multiple copies on same system work just fine
- Add <jruby dir>/bin to PATH
- Install gems (gem install or jruby -S gem install)

# Performance

- No, it's not all that important
  - > Until it is!
- JRuby 1.0 was about 2x slower than Ruby 1.8.6
- JRuby 1.1 is usually 2-5x faster
  - > Often faster than Ruby 1.9
  - > Lots more we can do
  - > Report performance issues as bugs!
    - <http://jira.codehaus.org/browse/JRUBY>

# Deployment: The Old Way

- Mongrel as server process
  - > Times number of apps
    - Times number of concurrent requests
      - With monitoring to keep alive
        - Hopefully no zombies or memory leaks
          - Better have a big slice
  - > ...And then you need to switch hosts/servers
- Sure, it can work...but why?
  - > These problems have been solved before
  - > It's painful because it's wrong!
    - Mongrel is a ghetto?
- (But yes, Mongrel works on JRuby)

# A Classic Revisited

- Java app server deployment (WAR files)
  - > N apps
  - > N' concurrent requests
  - > N'' database connections
  - > 1 process, to limits of machine
    - Plus full-site management, monitoring, profiling, debugging...
- Designed to scale up and out
  - > ...but at a cost of some agility
- Political standard
  - > Meet them halfway!

# DEMO

## JRuby on GlassFish

# Best of Both Worlds

- Agile development
  - > CLI support, dev time server, no build/deploy
- Simple to use
  - > One-shot execution, both prod and dev
- Scaling, both technical and logistical
  - > Single process for all apps and requests
- Ruby friendly
  - > gem install and go!

# **DEMO**

# **GlassFish Gem**

# Production JRuby on Rails

- Oracle's Mix – digg-like social customer site  
> [mix.oracle.com](http://mix.oracle.com)
- Sun's MediaCast – file distribution portal  
> [mediacast.sun.com](http://mediacast.sun.com)
- ThoughtWorks' Mingle – collaborative project mgmt  
> [mingle.thoughtworks.com](http://mingle.thoughtworks.com)
- Sonar – code/project analysis tool  
> [sonar.hortis.ch](http://sonar.hortis.ch)
- More on the way!

# What Does It All Mean?

- JRuby is raising the bar for Rails all the time
  - > It works today!
  - > Performance improving constantly
  - > Massive collection of high-perf libraries
- GlassFish is not your daddy's app server
  - > An enterprise server in a 3MB gem!
  - > Far easier deployment than other options
- Looking for users, use cases to support
  - > What do you need? Work with us!

# Thank you!

- JRuby - [www.jruby.org](http://www.jruby.org)
  - > [wiki.jruby.org](http://wiki.jruby.org)
- GlassFish - [glassfish.dev.java.net](http://glassfish.dev.java.net)
  - > `gem install glassfish`
  - > Looking for bug reports, feature requests!
- NetBeans - [www.netbeans.org](http://www.netbeans.org)
  - > Try it out, send feedback!
- Charlie's blog: [headius.blogspot.com](http://headius.blogspot.com)
- Tom's blg:  
[www.bloglines.com/blog/ThomasEEnebo](http://www.bloglines.com/blog/ThomasEEnebo)