JRuby
Enterprise 2.0

James Crisp    Josh Price
ThoughtWorks
Agenda

- What are Ruby and JRuby?
- Benefits and Drawbacks
- Where to use?
- Demos
- Case studies
Ruby

• Created by Yukihiro Matsumoto in 1993
• Open Source
• Vibrant community
• Dynamically typed
• Pure OO
• Syntactically flexible for DSLs
• Advanced meta programming
Ruby Runtimes

- **MRI** Matz’s Ruby Interpreter in C (Ruby 1.8.6)
- **YARV** Yet another Ruby VM (Ruby 1.9)
- **IronRuby** .NET, DLR (alpha)
- **Rubinius** Alternative VM
- **xRuby** Compiler in Java
- **JRuby** on the JVM
Java platform
Today's Java Platform
JRuby

- Java implementation of Ruby
- Open Source
- Full time developers from Sun and ThoughtWorks
  - Charles Nutter, Thomas Enebo, Ola Bini et al
- Allows Ruby to call Java code and vice versa
Timeline

- Started around 2001 by Jan Arne Petersen
- JRuby 1.0 released in June 2007
- JRuby 1.1 RC2 just released
- 1.1 release out soon
Why JRuby?

• Ruby / Rails productivity & tools
• Java libraries and interoperability
• Run on any OS / JVM / App server
• Leverage existing infrastructure & ops skills
• Politics – it's just a WAR
JRuby advantages

- Native vs Green Threading
- World class garbage collection
- Unicode
- Runs the same on all platforms
- JIT & AOT compilation
JRuby disadvantages

- Large memory footprint
- Longer startup times
- No native C extensions
- Not technically complete
Ruby Demo
A Simple Ruby Class

class RubyDemo < Demo

  def say_hello
    5.times { puts “Hi from Ruby!” }  
  end

  def greet(guests)
    guests.collect { |g| “Welcome #{g.name}” }  
  end

end

>> RubyDemo.new.greet(josh, james)

=> [“Welcome Josh”, “Welcome James”]
import javax.swing.JFrame
import javax.swing.JButton

f = JFrame.new('Swing Demo')
f.set_size 300, 300
f.layout = java.awt.FlowLayout.new
button = JButton.new('Hi World!')
f.add(button)
f.show
require 'cheri/swing'
include Cheri::Swing

f = swing.frame('Hello') {
  size 500, 500
  flow_layout
  button('Hello!') {
    on_click { puts 'Hi from Swing!' }
  }
}
Rails

- Originally written by DHH extracted from Basecamp
- Released as Open Source in 2004

Complete web framework:
- MVC framework with built-in REST / AJAX support
- O-R mapping framework
- Standard directory structure
- Simple templating language for views
- Plug-in framework
JRuby on Rails demo

- GoldSpike
  - Packages a WAR for any Java app server
  - Automatically creates extra runtimes
- Warbler
  - Easy to use wrapper around GoldSpike
- Glassfish Rails gem
  - Gem containing Glassfish v3 and Grizzly
Where to use?

- Enterprise Glue
- Web 1.0 / 2.0 Apps
- Ruby tools for Java
JRuby as Enterprise Glue

- Aggregating, synchronising and displaying data from different systems
- Productivity benefits of using Ruby / Rails
- Leverage existing Java code
- Better JMS integration
- Best of both worlds
Case Study

- Enterprise glue in a large investment bank
- Different systems with partial information
- JRuby webapp aggregates and displays data
- Built in 4 months
- Connects to different systems
  - JDBC (Sybase, Oracle)
  - XML over HTTPS (Ruby + Java)
  - REST over HTTPS (Ruby + Java)
  - SOAP (xFire)
Mingle

• ThoughtWorks Studios first product
• Agile Project Management application
• Built with Ruby on Rails, deployed on JRuby
• v1.0 built in < 1 year
• v1.1 released 3 months later
• v2.0 due for release soon
JRuby and Mingle

- Developed in MRI deployed on JRuby
- Stable deployment
- Speed of development
- Java libraries
  - charting, search, PDF, encryption
- Obfuscating ruby code
- Large memory footprint / speed
Ruby Tools

- **Gems**
  - Simple package management
    - `gem install activescaffold`
- **Rake**
  - Build tool
    - `rake db:migrate`
- **Capistrano**
  - Flexible deployment tool
    - `cap production deploy`
desc "Generate application code"
task :code_gen do
  # do the code generation
end

desc "Compile code"
task :compile => [:code_gen] do
  # do the compilation
end

desc "Test application"
task :test => [:compile] do
  # run the tests
end
desc "Start ferret server"
  task :start, :role => :app do
    run "cd #{current_path}; script/ferret_server -e #{stage} start"
  end

desc "Stop ferret server"
  task :stop, :role => :app do
    run "cd #{current_path}; script/ferret_server -e #{stage} stop"
  end

desc "Restart ferret server"
  task :restart, :role => :app do
    ferret.server.stop
    sleep(5)
    ferret.server.start
  end
require 'test/unit'
require 'java'

import java.util.ArrayList

class ArrayListTest < Test::Unit::TestCase
  def test_adding_object_to_arraylist
    list = ArrayList.new
    list.add("element")
    assert_equal(1, list.size)
  end

end
RSpec Java code

```ruby
require 'spec'
require 'java'
import java.util.ArrayList

describe "ArrayList with no elements" do
  it "should have one element after adding a string" do
    list = ArrayList.new
    list.add("element")
    list.size.should == 1
  end
end
```
Adoption issues

- Few developers in the market
- Smaller community than Ruby or Java
- Less books, google hits, documentation
- Less training and support options
- Diverging from the “one true language”
Adoption

- Ola's book
- Growing community
- Mailing lists & blogs
Summary

• Ruby / Rails productivity
• Web apps and system integration
• Leverage your Java investment
• Deploy to mature Java production environment
• Already used in production
Questions?
References

**JRuby**
http://www.jruby.org

**Ola Bini's blog & book**
http://ola-bini.blogspot.com/

**Charles Nutter's blog**
http://headius.blogspot.com/

**James' blog (for slides)**
http://jamescrisp.org/