JRuby Basics

Sang Shin
www.javapassion.com
“Learning is fun!”

Topics

- What is and what is not JRuby?
- Why JRuby (over Ruby) & JRuby on Rails (Ruby on Rails)?
- Why use (J)Ruby with Java?
- Calling Java from JRuby
- Calling (J)Ruby from Java

We are going to cover Rails deployment over Java platform in another presentation later on.

What is &
What is not JRuby?

What is JRuby?

- Java implementation of Ruby language
- Started in 2002, open source, many contributors
- Releases
  - June 2007: 1.0 release, focus on compatibility
  - April 2008: 1.1 release, focus on performance
  - May 2010: 1.5.0 release
  - August 2010: 1.5.2 release
- Aiming for compatibility with current Ruby version
  - JRuby 1.5.x is compatible with Ruby 1.8.7 and soon

What is NOT JRuby?

- JRuby is NOT an attempt to pollute Ruby
- JRuby is NOT an attempt to alter/add incompatibility
- JRuby is NOT a silver bullet to all Ruby issues
- JRuby is NOT an “admit that Java sucks”
- JRuby is NOT slow

JRuby Design Goals

- Bring beauty of Ruby to Java world
- Bring Java ecosystem to Ruby world
- Enable highly scalable and highly performing Ruby implementation

Why JRuby (over Ruby) & JRuby on Rails (over Ruby on Rails)?

Why JRuby (over Ruby)

- With JRuby you get the best of both worlds: Ruby applications and libraries, plus Java libraries. And you can access those libraries with Ruby syntax (or Java syntax, if you want).
- On average JRuby, runs 2 and a half times faster than Ruby, except at startup
- In addition to native threads, JRuby supports Unicode natively
- Code can be fully compiled ahead of time or just in time

Why JRuby on Rails (over Ruby on Rails)

- A JRuby on Rails application can be packaged into a WAR, and then deployed onto any compliant server.
  - The packaging of creating a war file from Rails can be done with Goldspike, or with the new kid on the block: Warbler
- Can use vast array of Java libraries
  - JPA, JTA, JMS, EJB, JDBC, JMX, JSF, etc.
  - Database connections are made through JDBC, which provides broader, more scalable database support.
- Glassfish--the Java web app server that scales well--is available as a JRuby gem.

Why (J)Ruby over Java?

Why (J)Ruby over Java?

- Ruby is fun, beautiful, and powerful language
- Ruby has features missing from Java
  - Closure (blocks)
  - Open classes
  - Meta programming
  - Duck-typing
- Domain Specific Language (DSL)
  - Through (J)Ruby's capability of allowing expressive method names and cushy semantics

Where is JRuby being used?

- Graphics and Games
  - Ruby + graphics = cool
- GUI development
  - Makes Swing much nicer to use, easier to handle
- JRuby on Rails
  - Better deployment options, better performance

Calling Java From JRuby

include Java & import

- *include Java* statement will give you access to the bundled Java libraries. However, this will not give you access to non-bundled libraries.
- The *import* statement is used to import a Java Class.

```
include Java

import java.util.ArrayList
import javax.swing.JFrame

list = ArrayList.new
frame = javax.swing.JFrame.new("Passion!")
list << frame
```

include_package within a Ruby Module

- Use *include_package "<package_name>"* in a Ruby Module to support namespaced access to the Java classes in the package.

```
include Java

module JavaLang
  include_package "java.lang"
end

s = JavaLang::String.new("This is my string from java.lang
    package")
```

include_class

- Use include_class "<class_name>" to include unbundled Java classes
- The unbundled Java classes (in the form of jar file) should be in the classpath

```
include Java

include_class 'mypackage.Hello'
h = Hello.new

puts "----Invoke a method of from Hello object"
s = h.sayHello("Message from Hello Java Class!")
```

Java Classes Naming in JRuby

- Imported classes are mapped into Ruby name space as follows:
  - *Java Convention: org.foo.department.Widget*
  - *JRuby convention: Java::OrgFooDepartment::Widget*

```
# The following two are equivalent
frame = javax.swing.JFrame.new("Passion!")
frame = Java::JavaxSwing::JFrame.new("Passion!")
```

Java Method Naming in JRuby

- Ruby method naming convention “lower case with _” can be used for Java method

```
version =
   System.getProperties["java.runtime.version"]
version =
   System.get_properties["java.runtime.version"]
```

Calling Ruby from Java

Calling Ruby From Java

- Leveraging scripting framework (JSR 223)

```
import javax.script.*;
public class EvalScript {
    public static void main(String[] args) throws Exception {
      ScriptEngineManager factory = new ScriptEngineManager();
      // Create a JRuby engine.
      ScriptEngine engine = factory.getEngineByName("jruby");
      // Evaluate JRuby code from string.
      try {
        engine.eval("puts('Hello')");
      } catch (ScriptException exception) {
        exception.printStackTrace();
      }
}
```

Thank you!

Sang Shin
http://www.javapassion.com
“Learning is fun!”