Surveying the JVM Landscape

Scott Battaglia & Drew Wills

Jasig Spring Conference, March 10th, 2010

© Copyright Unicon, Inc., 2006. This work is the intellectual property of Unicon, Inc. Permission is granted for this material to be shared for non-commercial, educational purposes, provided that this copyright statement appears on the reproduced materials and notice is given that the copying is by permission of Unicon, Inc. To disseminate otherwise or to republish requires written permission from Unicon, Inc.

THE STATE UNIVERSITY OF NEW JERSEY RUTGERS

UNICON

1. JVM Languages Roll Call
2. Scala Quick Tour
3. Groovy Quick Tour
4. JavaScript Examples

JVM Languages Roll Call

About the JVM & the variety of languages in use on it

What is the JVM?

- The Java Virtual Machine is a crucial component of the Java Platform
- It provides a *virtual* machine model for software programs & data structures to execute other programs & scripts
- Available for Windows, *nix, Solaris, Mac OS, & Mobile Devices
- Sun Microsystems claims over 4.5 billion JVM-enabled devices

What is the JVM? (cont.)

- The JVM runs an intermediate language called *Java bytecode*
- Java bytecode is normally generated from Java source code, but not always
- JVMs have been released from Sun, as well as IBM & BEA
- JVM runtime can execute .class & .jar files

What Can Run on the JVM?

- Designed for the JVM:
  - Java (duh!)
  - Groovy*
  - Clojure
  - Scala*
- Non-JVM languages with JVM versions:
  - Ruby (jRuby)
  - JavaScript (Rhino)
  - Python (Jython)
  - PHP (Quercus)
- More complete list: http://en.wikipedia.org/wiki/List_of_JVM_languages

Java Pros & Cons

- Pros
  - Platform independent
  - Small footprint
  - Pre-built libraries
  - Rich 3rd party lib & IDE support
  - Object-oriented
  - Strictly-typed
- Cons
  - Community process makes for slow changes
  - Poor implementation of generics, *etc.*
  - Lacks newer language features (closures, *etc.*)
  - Not completely object-oriented
  - Verbose syntax (especially after generics)
  - No operator overloading, *etc.*

New Focus on Alternate Languages

- JDK 6
  - JSR-223: Scripting for the Java Platform
  - Mozilla Rhino **ScriptEngine** implementation
- JDK 7
  - JSR-292: Supporting Dynamically Typed Languages on the Java Platform
  - Project Coin
    - Switch on Strings
    - Automatic resource management
    - Type inference for generic instance creation
    - Language support for collections

Scala Quick Tour

Introduction, features, & examples

What is Scala?

- Begun by Martin Odersky in 2001
  - Professor, School of Computer & Communication Sciences at EPFL
  - Worked on reference implementation of javac
- Current version is 2.7.7; 2.8 is in the works

http://www.scala-lang.org/

What is Scala? (cont.)

- Statically-typed language
- Object-oriented programming
- Functional programming
- Available for the JVM and .NET CLR
- Succinct syntax
- Sophisticated type system
- Scales from scripts to large, distributed applications
- Benefits from JVM improvements, profiling, & optimization tools

Scala Features

- Object vs. Class
- Compiler is *smart*
  - Semicolons optional
  - Can detect line wrapping
- Variable declarations
  - **val** vs. **var**
- Type inference

```
1 val intToStringMap Map[Integer, String] = new HashMap
2
3
```

- Option class vs. null
  - None, Some
- Tuples

Scala Features (cont.)

- Actor API for concurrency
- No checked exceptions
- Operator overloading
- Support for DSLs (domain-specific languages)
- Filtering & yielding in for loops

```
1  val filteredBreeds = for {
2      breed <- dogBreeds
3      if breed.contains("Terrier")
4      if !breed.startsWith("Yorkshire")
5  } yield breed
6
7
```

Scala Features (cont.)

- Pattern matching...
  - On types
    - **case i: Int => println("Got an Integer: " + i)**
  - On sequences
    - **case List(_,3,_) => println("Four elements")**
  - On Tuples
    - **case (thingOne, thingTwo) if thingOne == "Good" => ...**
  - On case classes
    - **case Person("Alice", 25) => println("Hi Alice!")**
    - **case Person(name, age) => println("Who are you, " + age + "-year-old person named " + name + "?")**
  - Applies to try/catch blocks also

Scala Features (cont.)

- Enumerations are just classes
- Traits: "interfaces with optional implementations"
- Primary constructors vs. auxiliary constructors

Scala Examples

- Hello World

```
1  Object HelloWorld {
2      def main(args: Array[String]) {
3          println("Hello, world!")
4      }
5  }
6
```

- Java interoperability

```
1  import java.lang.Double;
2
3  val lower = Integer.parseInt(args(0))
4  val upper = Integer.parseInt(args(1))
5
6  val number = lower + new Double((Math.random * (upper - lower))).intValue();
7  Console.println("A number between " + lower + " and " + upper + " is: " + number);
8
```

Scala Collections Example

```
1  object Maps {
2
3      val colors = Map("red" -> 0xFF0000,
4                       "turquoise" -> 0x00FFFF,
5                       "orange" -> 0xFF8040,
6                       "brown" -> 0x804000)
7
8      def main(args: Array[String]) {
9          for (name <- args) println(
10             colors.get(name) match {
11                 case Some(code) => name + " has code:  " + code
12                 case None => "Unknown color:  " + name
13             }
14         }
15     }
16
17 }
18
19
```

Groovy Quick Tour

Introduction, features, & examples

What is Groovy?

- *"An agile dynamic language for the Java Platform"*
- JSR-241: The Groovy Programming Language
- Pre-JSR releases 2004 – 2006
- Version 1.0 released January 2, 2007
- Version 1.7.1 released February 19, 2010
- G2One Inc. acquired by SpringSource Nov. 11, 2008

What is Groovy? (cont.)

- Object-oriented
- *Almost* a Java language & platform superset
- Compiles to standard Java bytecode at build time
- Supports static *and dynamic* typing
- Functional programming
- Succinct syntax
- Metaprogramming

Groovy Features

- Better Defaults than Java
  - 6 packages + 2 classes imported automatically
  - Classes & methods are public by default
  - No checked exceptions
- Boilerplate items are optional
  - Semicolons
  - **return** Keyword
  - Getters & setters
  - Class & method declarations

Groovy Features (cont.)

- Groovy Truth
- GStrings, multi-line strings, & slashy strings
- Language-level regex support
- Enhanced switch syntax
  - Case statements based on collections, ranges, classes, regex, equals()
- Language-level support for Lists & Maps
- Closures

Groovy Features (cont.)

- JDK library enhancements
  - http://groovy.codehaus.org/groovy-jdk/
  - New methods added to String, Map, List, URL, *etc.*
- >groovy (Groovy)
- >groovysh (Groovy Shell)
- >groovyConsole (Groovy Console)

Groovy Examples

- Hello World

```
1  println 'Hello World!'
2
```

- Java interoperability

```
1  def lower = args[0].toInteger();
2  def upper = args[1].toInteger();
3
4  def number = lower + (int) (Math.random() * (upper - lower));
5  println "A number between ${lower} and ${upper} is:  ${number}";
6
```

Groovy Collections Example

```
1  def colors = [
2      red:0xFF0000,
3      turquoise:0x00FFFF,
4      orange:0xFF8040,
5      brown:0x804000
6  ];
7
8  args.each {
9      switch (it) {
10         case colors.keySet():
11             println it + ' has code:  ' + colors[it];
12             break;
13         default:
14             println 'Unknown color ' + it;
15             break;
16     }
17 }
18
```

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Users\awills>groovy example.groovy red foo bar orange
red has code:  16711680
Unknown color foo
Unknown color bar
orange has code:  16744512

C:\Users\awills>
```

Groovy Seminar

- Introduction to Groovy
  - Andrew Wills & Lennard Fuller
  - Wednesday, 1:00 PM – 4:30 PM
  - Supplemental cost
- Broad introduction to the Groovy Language & platform tools
- Aimed at Java developers who are new to Groovy

JavaScript Examples

A look at a non-JVM language

JavaScript in a Nutshell

- Mozilla Rhino: JavaScript for Java
- Object-based
- Dynamic-typing
- Functional programming
- Java-like syntax
- De-facto scripting language of the browser
- No Java types imported by default – not even **java.lang**!

JavaScript Examples

- Hello World

```
1  print('Hello World!');
2
3
```

- Java interoperability

```
1
2  var lower = parseInt(arguments[0]);
3  var upper = parseInt(arguments[1]);
4  var number = lower + Math.round(Math.random() * (upper - lower));
5
6  importPackage(java.lang);
7  var buffer = new StringBuilder();
8  buffer.append("A number between ").append(lower).append(" and ")
9                .append(upper).append(" is:  ").append(number);
10 print(buffer.toString());
11
12
```

JavaScript Collections Example

```
var colors = {
    red:0xFF0000,
    turquoise:0x00FFFF,
    orange:0xFF8040,
    brown:0x804000
};

for (a in arguments) {
    var name = arguments[a];
    colors[name] ? print(name + ' has code:  ' + colors[name])
                 : print('Unknown color:  ' + name);
}
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Users\awills>js test.js red foo bar orange

C:\Users\awills>java -cp c:\rhino1_7R2\js.jar org.mozilla.javascript.tools.shell
.Main test.js red foo bar orange
red has code:  16711680
Unknown color:  foo
Unknown color:  bar
orange has code:  16744512

C:\Users\awills>
```

Questions?

Scott Battaglia

scott_battaglia@rutgers.edu

Drew Wills

drew@unicon.net