



the
POWER
of
JAVA™



 **OpenLogic®**

JavaOne
Part of the Network for Business Success

Groovy = Java™ Technology + Ruby + Python for the JVM™

Rod Cope

CTO

OpenLogic, Inc.

<http://www.openlogic.com>



TS-3273

Groovy Goal

What You'll Get Out of This Session

Learn what Groovy can do for you and how to start using it today!

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Sample: Java Code and Groovy!

```
public class Filter {  
    public static void main( String[] args ) {  
        List list = new ArrayList();  
        list.add( "Rod" ); list.add( "Neeta" );  
        list.add( "Eric" ); list.add( "Missy" );  
  
        Filter filter = new Filter();  
        List shorts = filter.filterLongerThan( list, 4 );  
        System.out.println( shorts.size() );  
  
        Iterator iter = shorts.iterator();  
        while ( iter.hasNext() ) {  
            System.out.println( iter.next() );  
        }  
    }  
  
    public List filterLongerThan( List list, int length ) {  
        List result = new ArrayList();  
        Iterator iter = list.iterator();  
        while ( iter.hasNext() ) {  
            String item = (String) iter.next();  
            if ( item.length() <= length ) { result.add( item ); }  
        }  
        return result;  
    }  
}
```

Sample: Groovy!

```
def list = ["Rod", "Neeta", "Eric", "Missy"]
def shorts = list.findAll { it.size() <= 4 }
println shorts.size()
shorts.each { println it }
```

```
-> 2
-> Rod
    Eric
```

Sample in Java Code (27 Lines)

```
public class Filter {
    public static void main( String[] args ) {
        List list = new ArrayList();
        list.add( "Rod" ); list.add( "Neeta" );
        list.add( "Eric" ); list.add( "Missy" );

        Filter filter = new Filter();
        List shorts = filter.filterLongerThan( list, 4 );
        System.out.println( shorts.size() );

        Iterator iter = shorts.iterator();
        while ( iter.hasNext() ) {
            System.out.println( iter.next() );
        }
    }

    public List filterLongerThan( List list, int length ) {
        List result = new ArrayList();
        Iterator iter = list.iterator();
        while ( iter.hasNext() ) {
            String item = (String) iter.next();
            if ( item.length() <= length ) { result.add( item ); }
        }
        return result;
    }
}
```

Sample in Groovy (4 lines)

```
def list = ["Rod", "Neeta", "Eric", "Missy"]
def shorts = list.findAll { it.size() <= 4 }
println shorts.size()
shorts.each { println it }
```


Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Java™ Platform Strengths

- Lots of reusable software, components, tools
- VM and binary compatibility
 - Build deployment units (class, jar, jnlp, war, ear, sar, etc.) and run them anywhere
 - Easily reuse libraries and tools
- Allows for innovation at the source code level

Components vs. Scripting

- We reuse far more code than we write
- We spend more time gluing components than writing them
- We write more tests than real code
- The web tier is a perfect example
 - Render business objects as markup (templating/scripting)
 - Glue actions with requests and domain models (MVC)
 - Most of web tier is duct tape with pockets of business logic
- Scripting is a great way to glue components together

Why Another Agile Language?

- Want complete binary compatibility with Java
 - No difference from Java at JVM/bytecode level
 - No wrappers or separate islands of APIs
- Java code-friendly syntax
 - Don't want something totally foreign to Java developers
- Want a scripting language designed:
 - For the Java Platform
 - By Java developers
 - For Java developers

Groovy

- Who/When
 - James Strachan, Bob McWhirter—August 2003
- What
 - Dynamic, object-oriented scripting language for JVM
 - Features of Ruby, Python, and Smalltalk™
- Why
 - JPython, JRuby, BeanShell, etc. all lacking
- How
 - Was hand-written compiler and bytecode generator
 - Now uses ANTLR and ASM

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Groovy Features

- Dynamic and (optional) static typing

```
int a = 2
def str = "Hello"
```

- Native syntax for lists, maps, arrays, beans, etc.

```
def list = ["Rod", 3, new Date()]
def myMap = [Neeta:32, Eric:34]
```

- Closures

```
myMap.each(
{ name, age -> println "$name is $age years old" } )
  -> Eric is 34 years old
  -> Neeta is 32 years old
```

Groovy Features (Cont.)

- Regex built-in

```
if ( "name" ==~ "na.*" ) { println "match!" }  
-> match!
```

- Operator overloading

```
def list = [1, 2, 3] + [4, 5, 6]  
list.each { print it }  
-> 123456
```

- Autoboxing and polymorphism across collection, array, map, bean, String, iterators, etc.

```
String[] array = ['cat', 'dog', 'mouse']  
def str = 'hello'  
println "${array.size()}, ${str.size()}, ${list.size()}"  
-> 3,5,6
```


Groovy-er JDK™ Software

- Adds convenient methods to JDK
- String
 - `contains()`, `count()`, `execute()`, `padLeft()`, `center()`, `padRight()`, `reverse()`, `tokenize()`, `each()`, etc.
- Collection
 - `count()`, `collect()`, `join()`, `each()`, `reverseEach()`, `find/All()`, `min()`, `max()`, `inject()`, `sort()`, etc.
- File
 - `eachFile()`, `eachLine()`, `withPrintWriter()`, `write()`, `getText()`, etc.
- Lots there and growing all the time
- You can add methods programmatically

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Working with Java Technology

- Create Adder.java

```
package com.openlogic.test;
public interface Adder
{
    public int add( int a, int b );
}
```

- Create GAdder.groovy

```
package com.openlogic.test
class GAdder implements Adder
{
    public int add( int a, int b )
    {
        return a + b
    }
}
```

Working with Java Technology (Cont.)

- Compile the Java code

```
javac -d target Adder.java
```

- Compile the Groovy code

```
groovyc --classpath target GAdder.groovy
```

- Use the Groovy code inside Java code with the interface

```
Adder adder = new com.openlogic.test.GAdder();  
int answer = adder.add( 1, 2 );  
System.out.println( "Answer = " + answer );  
-> Answer = 3
```

Working with Ant

- BSF-compliant

```
<script language="groovy">println 'hi' * 2</script>
```

- Groovy task

```
<taskdef name="groovy"
    classname="org.codehaus.groovy.ant.Groovy"
    classpath="groovy-all-1.0-jsr-05.jar"/>

<groovy>
    def x = 2
    for ( count in 1..x ) {
        ant.echo( "hello world ($count)" )
    }
    ant.jar( destfile: "c:/stuff.jar", basedir: "." )
</groovy>
```

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Groovy Markup

- Native support for hierarchical structures in code
 - XML
 - XHTML
 - Ant
 - Swing
 - SWT
- Relatively easy to add your own

Groovy Markup

XML Generation

```
<people>
```

```
  <person name='Rod' >
```

```
    <pet name='Bowie' age='4' />
```

```
    <pet name='Misha' age='10' />
```

```
  </person>
```

```
  <person name='Eric' >
```

```
    <pet name='Poe' age='6' />
```

```
    <pet name='Doc' age='5' />
```

```
  </person>
```

```
</people>
```


Groovy Markup

XML Generation

```

data = [Rod:  [Misha:10, Bowie:4],
        Eric: [Poe:6,   Doc:5] ]

xml = new groovy.xml.MarkupBuilder()

doc = xml.people() {
    data.each { name, pets ->
        person( name: name ) {
            pets.each { dog, age ->
                pet( name: dog, age: age )
            }
        }
    }
}
  
```

Groovy Markup

XML Parsing

```
xml = """<people>
  <person name="Rod">
    <pet name="Misha" age="10"/>
    <pet name="Bowie" age="4"/>
  </person>
  <person name="Eric">
    <pet name="Poe" age="6"/>
    <pet name="Doc" age="5"/>
  </person></people>"""

people = new XmlParser().parseText( xml )
println people.person.pet['@name']
-> ['Misha', 'Bowie', 'Poe', 'Doc']
```

Groovy Markup

Swing

```
import java.awt.*
swing = new groovy.swing.SwingBuilder()
frame = swing.frame(title:'My Frame', size:[800,400]) {
    menuBar {
        menu(text:'File') {
            menuItem() {
                action(name:'New', closure:{ println("New") })
            }
        }
    }
    panel(layout:new BorderLayout()) {
        label(text:'Name', constraints:BorderLayout.WEST,
            tooltipText:'This is the name field')
        button(text:'Click me!',
            constraints:BorderLayout.SOUTH,
            actionPerformed: { println("Click!") } )
    }
}
frame.show()
```

Groovy SQL

```
sql = new groovy.sql.Sql( dataSource )
sql.execute( "create table person
             ( name varchar, age integer)" )

people = sql.dataSet( "person" )
people.add( name: "Rod", age: 35 )
people.add( name: "Neeta", age: 32 )

sql.eachRow( "select * from person" ) { person ->
    println "$person.name is $person.age years old"
}
```

-> Rod is 35 years old

-> Neeta is 32 years old

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Extras

- Processes: `println "cmd /c dir".execute().text`
- Threading: `Thread.start { any code }`
- Testing: `GroovyTestCase`, `GroovyMock`
- SWT: full support for SWT building, like `SwingBuilder`
- Groovy pages/template engine: `GSP`, `Groovlets`, etc.
- UNIX[®] Scripting: Groovy API for `pipe`, `cat`, `grep`, etc.
- Eclipse, IntelliJ, JEdit: Groovy plug-ins available
- XML-RPC, Groovy SOAP
- Dynamic and programmatic language extensions
- Grails (conceptually similar to Rails, but with Spring and Hibernate)
- ActiveX Proxy: control over Microsoft Windows (IE, Excel, etc.)

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

DEMO

Groovy Automation

Agenda

Groovy Sample

Genesis of Groovy

Groovy Features

Working with Java Technology

Groovy Markup

Extras

Demo

Conclusion

Conclusion

- Status
 - 1.0 release expected later this summer
- Development time
 - Less than half that of Java technology
- Performance
 - 20–90% of Java technology, depending on usage
 - Very little tuning so far, waiting for 1.0 release
- Recommendations
 - Ready for small, non-mission-critical projects
 - Try it! Very easy to learn and lots of fun!

For More Information

- Groovy home page
 - <http://groovy.codehaus.org>
- GDK Javadoc™
 - <http://groovy.codehaus.org/groovy-jdk.html>
- JSR-241: The Groovy Programming Language
 - <http://www.jcp.org/en/jsr/detail?id=241>

Q&A

Rod Cope, CTO and Founder
OpenLogic, Inc.



the
POWER
of
JAVA™



 **OpenLogic®**

JavaOne
Part of the Network for Business Success

Groovy = Java™ Technology + Ruby + Python for the JVM™

Rod Cope

CTO

OpenLogic, Inc.

<http://www.openlogic.com>



TS-3273