Groovy and Grails

What are they, and what are the benefits and risks?
Who am I?

John Leach, Chief Technical Officer for Syger
Java developer from the start – and still learning
Syger - a small software consultancy in Verona, Italy

Who are you?

Who's heard of Groovy?
Who's heard of Grails?
Who writes web applications?

Pleased to meet you

Warning! Intensive content session
Lots to see, not much time
Questions at the end
Overview of Groovy

Dynamic scripting language
Similar syntax to Java
Evolution not revolution

But...
with closures
and meta programming
and relaxed typing
and lots of syntax sugar
and a silly name - sigh
Overview of

Ruby on Rails philosophy – evolution not revolution
Convention over configuration
Uses 20% Groovy and 80% Java (elegance and power)

But...

with Spring, Hibernate, and SiteMesh
has a plugin architecture
no XML (though it's there when you need it)
no HQL/SQL (though they're there when you need them)
no Ruby, JRuby or Rails
public class HelloWorld {
    private String name;

    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public String greet() {
        return "Hello " + name;
    }
    public static void main(String... args) {
        HelloWorld helloWorld = new HelloWorld();
        helloWorld.setName("Groovy");
        System.out.println(helloWorld.greet());
    }
}
class HelloWorld {
    String name

    String greet() {
        return "Hello " + name
    }

    static void main(String... args) {
        HelloWorld helloWorld = new HelloWorld()
        helloWorld.name = 'Groovy'
        println(helloWorld.greet())
    }
}
class HelloWorld {
    String name

    String greet() {
        "Hello ${name}"
    }

    static void main(String... args) {
        println new HelloWorld(name: 'Groovy').greet()
    }
}

Plain Ordinary Groovy Objects
Reduced clutter
Simple constructors

Adapted from: http://groovy.dzone.com/news/java-groovy-few-easy-steps
Java to Groovy – Check  HelloWorld.class

Groovy:

```
JUGSardegna>groovy HelloWorld.groovy
Hello Groovy
```

Java:

```
JUGSardegna>groovyc HelloWorld.groovy
JUGSardegna>java -cp %GROOVY_HOME%\embeddable\groovy-all-1.5.4.jar;.\HelloWorld
Hello Groovy
```

Javap:

```
JUGSardegna>javap HelloWorld
Compiled from "HelloWorld.groovy"
public class HelloWorld extends java.lang.Object
    implements groovy.lang.GroovyObject {
        public HelloWorld();
        public java.lang.String greet();
        public java.lang.String getName();
        public void setName(java.lang.String);
        public static java.lang.Object main(java.lang.String[]);
        ...
    }
```
import static java.lang.System.out;
import static java.util.Arrays.asList;
import java.util.ArrayList;
import java.util.List;

public class ListTests {
    public static void main(String... args) {
        List<String> names = asList("Ted", "Fred", "Jed", "Ned");
        out.println(names.getClass().toString() + " " + names);
        List<String> shortNames = new ArrayList<String>();
        for(String s : names) {
            if (s.length() < 4) {
                shortNames.add(s);
            }
        }
        out.println(shortNames.size());
        for(String s : shortNames) {
            out.println(s);
        }
    }
}
class ListTests {
    static void main(String... args) {
        List<String> names = ['Ted', 'Fred', 'Jed', 'Ned']
        println "${names.class} ${names}"
        List<String> shortNames = new ArrayList()
        for(String s : names) {
            if (s.length() < 4) {
                shortNames << s
            }
        }
        println shortNames.size()
        shortNames.each { println it }
    }
}
List<String> names = ['Ted', 'Fred', 'Jed', 'Ned']
println "${names.class} ${names}"
List<String> shortNames = names.findAll { it.length() < 4 }
println shortNames.size()
shortNames.each { println it }

JUGSardegna>groovy ListTests.groovy
class java.util.ArrayList ['Ted', 'Fred', 'Jed', 'Ned']
3
Ted
Jed
Ned

Idiomatic Groovy
- Reduced clutter
- Simple concise syntax

Meta Object Protocol - MOP

Introspection:

```java
MyClass.metaClass.methods.each { println it.name }
MyClass.metaClass.properties.each { println it.name }
MyClass.metaClass.respondsTo(obj, 'execute')
MyClass.metaClass.hasProperty(obj, 'status')
```

Dynamic method invocation:

```java
obj."$name"()
```

Dynamic property getters and setters:

```java
Object value = obj."$name"
obj."$name" = value
```
Intercepting (Aspect Oriented Programming)

```java
def invokeMethod = { String name, args -> println "$name invoked" }
def getProperty = { String name -> println "getting $name" }
def setProperty = { String name, value -> println "setting $name" }
```

Changing behaviour at run time:

```java
def methodMissing = { String name, args -> "No $name method" }
def propertyMissing = { String name -> "No $name property" }
Duck.metaClass.quack = { "Quack!" } // duck.quack()
Duck.metaClass.getSpecies = { -> "Canard" } // duck.species
```
Domain Specific Language: AntBuilder

AntBuilder ant = new AntBuilder()
String myDir = 'target/AntTest/'
ant.sequential {
    echo('inside sequential')
    mkdir(dir: myDir)
    copy(todir: myDir) {
        fileset(dir: 'src/test') {
            include(name: '**/*.groovy')
        }
    }
    echo('done')
}
File file = new File('target/AntTest/groovy/util/AntTest.groovy')
assert file.exists() // yes it does

Adapted from: http://groovy.codehaus.org/Using+Ant+from+Groovy

Gant – Groovy, Ant, but no XML: http://gant.codehaus.org/
Domain Specific Language: MarkupBuilder

Groovy snippet:

```groovy
MarkupBuilder xml = new MarkupBuilder(writer)
xml.'rec:records'('xmlns:rec':'http://groovy.codehaus.org') {
  car(name:'HSV Maloo', make:'Holden', year:2006) {
    country('Australia')
    record(type:'speed', ' Truck with speed of 271kph')
  }
}
```

Output snippet:

```xml
<rec:records xmlns:rec='http://groovy.codehaus.org'>
  <car name='HSV Maloo' make='Holden' year='2006'>
    <country>Australia</country>
    <record type='speed'> Truck with speed of 271kph</record>
  </car>
</rec:records>
```

Adapted from: http://groovy.codehaus.org/Creating+XML+using+Groovy's+MarkupBuilder
But Wait, there's More!

Ranges – 0..9, 0..<10
Curried closures
Regular expression syntax sugar - /\d+/  
Extended switch operator – switch(title) { case 'Groovy': ...
Operator overloading – list << item
Elvis operator – value = value ?: defaultValue;
Safe dereferencing - person?.parents?.grandParents
The Expando class – saves writing a 'real' class
Unit testing, the Groovy Mock Library
SwingBuilder
Joint compiler (compile Groovy and Java source code together)
...
But we don't have time for all that now
Groovy – the Good News

Past the version 1.0 barrier (2 Jan 2007)

IDE support is maturing (Eclipse, NetBeans, and IntelliJ IDEA)

Being used in industry

Currently around 30th place according to TIOBE (http://www.tiobe.com)

G2One Inc., support from the Lead developer (Guillaume Laforge)

IBM Project Zero

“Drill down” to Java when you need the speed
Groovy – the Bad News

IDE support is *still* maturing

Slow execution speed (but not s-l-o-w)

Idiomatic Groovy is not Java

Won't get you a Job
Groovy – the Fear, Uncertainty, and Doubt

Interpreted language – no, compiled to Java bytecode

Not a standard – no, JSR 241

Orphan project – no Sun, Oracle, IBM, IntelliJ support

Usurping Java – no, augmenting Java

No Groovy programmers – no, most Java programmers should understand it
Pragmatic Groovy

Start in places where execution speed is less important:

- Build scripts – *AntBuilder*, *Gant*
- Unit testing, and mocking
- Swing User Interfaces – *SwingBuilder*
- Domain Specific Languages
Grails – What's in the Box?

Generators
Predefined application layout (folders)
Model View Controller pattern - surprise!
GORM – Hibernate made easy
Spring and Spring MVC under the covers
SiteMesh powering the views
Groovy Server Pages (GSP)
Tag Libraries but no XML
Plug-in architecture
Testing – unit, integration, web
Excellent, concise documentation
Generators

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>grails create-app</td>
<td>Creates (and populates) the application directories</td>
</tr>
<tr>
<td>grails create-domain-class</td>
<td>Creates an empty domain (model) class</td>
</tr>
<tr>
<td>grails create-service</td>
<td>Creates a transactional business logic class</td>
</tr>
<tr>
<td>grails create-tag-lib</td>
<td>Creates an empty tag library class</td>
</tr>
<tr>
<td>grails create-unit-test</td>
<td>Creates a unit test class</td>
</tr>
<tr>
<td>grails generate-controller</td>
<td>Generates a CRUD controller class for a domain class</td>
</tr>
<tr>
<td>grails generate-views</td>
<td>Generates the four CRUD views for a domain class</td>
</tr>
<tr>
<td>grails run-app</td>
<td>Runs the web application in Jetty</td>
</tr>
<tr>
<td>grails test-app</td>
<td>Runs the unit tests</td>
</tr>
<tr>
<td>grails console</td>
<td>Runs the Grails Swing interactive console</td>
</tr>
<tr>
<td>grails shell</td>
<td>Runs the Grails interactive shell</td>
</tr>
<tr>
<td>grails war</td>
<td>Creates a war file for JEE deployment</td>
</tr>
</tbody>
</table>

Run `grails create-app`, then `grails run-app`, and you've got an (empty) running web application, in less than 30 seconds.

You'll still have to write *some* code yourself.
Domain Models

class Album {
    User user
    String caption
    String description
    Set pictures

    static belongsTo = User
    static hasMany = [ pictures:Picture ]
    static constraints = {
        caption(size:1..40, blank:false)
    }
    static mapping = {
        pictures cascade:'all', inverse:true
        description type:'text'
        user index:'user_idx', unique:false
    }
}

class Picture {
    User user
    Album album
    Set images
    String file
    String caption

    static belongsTo = [ User, Album ]
    static hasMany = [ images:Image ]
    static transients = [ 'file' ]
    static mapping = {
        images cascade:'all', inverse:true
    }
}

Associations

Validation

Transient properties

Object Relational Mapping

POGO

Syger
Views

views/layouts/main.gsp

```html
<!DOCTYPE html ... >
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title><g:layoutTitle default="WebAlbum" /></title>
  <link rel="stylesheet" type="text/css"
    href="${createLinkTo(dir:'css', file:'main.css')}"/>
  <g:layoutHead />
</head>
<body>
  <div class='title'>WebAlbum</div>
  <div class='content'>
    <g:layoutBody />
  </div>
</body>
</html>
```
Views

```html
<html>
<head>
<meta name="layout" content="main" />
<title>WebAlbum : Create Picture</title>
<script type="text/javascript"> ... </script>
</head>
<body>
<h1>Create Picture</h1>
<g:uploadForm action="save">
   ... 
   <td valign="top" class="name">
      <label for="caption">Caption:</label>
   </td>
   <td valign="top" class="value">
      <input type="text" size="40" maxlength="40"
         id="caption" name="caption"
         value="${fieldValue(bean: picture, field: 'caption')}"/>
   </td>
   ... 
</g:uploadForm>
</body>
</html>
```

domain/Picture.groovy
class PictureController {

    def list = {
        [list:Picture.list(params), paginateCount:Picture.count()]
    }

    def show = {
        Picture picture = Picture.get(params.id)
        if (!picture) {
            flash.message = "Picture not found"
            redirect(action:list)
        } else {
            return [picture:picture]
        }
    }

    ...
}
class PictureController {

    def beforeInterceptor = [ action:this.&intercept, only:['create']]

    def create = {
        ...
    }

    def intercept() {
        User user = sessionUser()
        if (!user || user.albumsCount == 0) {
            flash.warning = "You must create an album first!"
            redirect(controller: 'album', action: 'create')
            return false
        }
        true
    }
}
Tag Libraries

views/picture/show.gsp

...<tr class="prop">
    <td valign="top" class="name">Caption:</td>
    <td valign="top" class="value">
        <wa:pictureAnchor picture="${picture}" size="${Image.Original}">
            ${picture.caption ?: '...'}
        </wa:pictureAnchor>
    </td>
</tr>...
class WebAlbumTagLib {

    static namespace = "wa"

    def pictureAnchor = { attrs, body ->
        Picture picture = attrs.remove('picture')
        def size = attrs.remove('size')
        String link = createPictureLink(picture.id, size).encodeAsHTML()
        out << "<a href="${link}""
        attrs.each { key, value ->
            out << " $key="$value""
        }
        out << '>
        out << body()
        out << '</a>'
    }

    ...
}

taglib/WebAlbumTagLib.groovy
But Wait, there's More!

Filters – conf/WebAlbumFilter.groovy
Create Gant scripts – scripts/CompileSources.groovy
GORM many-to-many, composition, inheritance, eager fetching, ...
GORM dynamic finders – findByFirstNameAndLastName(…)
GORM transactions – User.withTransaction { status -> … }
Controller chaining
Shared templates
URL mappings - "/sale" (controller:'product', action:'sale')
Multiple request conversations – Web Flow
Ajax support – Prototype, Dojo, Yahoo UI, GWT
Content negotiation
Web Services – REST and SOAP

But we don't have time for all that now
Past the version 1.0 barrier (4 Feb 2008)

IDE support is maturing (Eclipse, NetBeans, and IntelliJ IDEA)

Being used in industry

G2One Inc., support from the Lead developer (Graeme Rocher)

“Drill down” to Java when you need to
Grails – the Bad News

IDE support is *still* maturing

Slow execution speed (but not s-l-o-w)

Won't get you a Job
Grails – the Fear, Uncertainty, and Doubt

Another Rails clone – no, uses the philosophy in a Groovy/Java way

Built with an interpreted language (Groovy) – no, 20% Groovy which compiles to bytecode anyway

No Grails programmers – no, see no Groovy programmers

Only good for CRUD applications – no, you can do any full stack JEE application, SOAP and REST included

Much slower than JEE – no, Sun engineers results showed JEE to be 2 to 4 times faster with 100 to 500 concurrent users

Start in places where execution speed is less important:

- In-house web applications
- “Long tail” applications (10 – 50 concurrent users)
- Prototyping a JEE web application
What's Next?

Groovy: http://groovy.codehaus.org/
Grails: http://grails.org/
About Groovy: http://aboutgroovy.com/
Groovy Zone: http://groovy.dzone.com/
InfoQ Groovy: http://www.infoq.com/groovy
InfoQ Grails: http://www.infoq.com/grails
Graeme Rocher's blog: http://graemerocher.blogspot.com/
Guillaume Laforge's blog: http://glaforge.free.fr/weblog/
G2One Inc.: http://www.g2one.com/index.html
Read the Books, Watch the Movies

Books:
- Groovy Recipes: http://pragprog.com/titles/sdgrvr
- Programming Groovy: http://pragprog.com/titles/vslg
- Groovy in Action: http://www.manning.com/koenig/
- The Definitive Guide to Grails:

Films:
- Grails eXchange 2007: http://grails-exchange.com/
Thank You, any Questions?

Syger: http://www.syger.it/

Grails WebAlbum:
http://www.syger.it/Tutorials/GrailsWebAlbum.html

Ruby on Rails WebAlbum (a comparison, written first):

My personal site: http://www.jhl.it/

Contact: john.leach@syger.it