Groovin' with Grails

How to use your favorite frameworks Rails-style in Java.
Really.

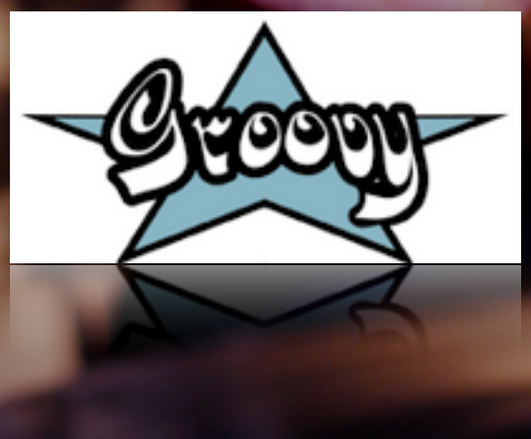
Groovin' with Grails

How to use your favorite frameworks Rails-style in Java.
Really.

Groovin' with Grails

How to use your favorite frameworks Rails-style in Java.
Really.

Groovin' with Grails



+

How to use your favorite frameworks Rails-style in Java.
Really.

Groovin' with Grails

+

How to use your favorite frameworks Rails-style in Java.
Really.

Groovin' with Grails

+

=

How to use your favorite frameworks Rails-style in Java.
Really.

Groovin' with Grails

+

=

How to use your favorite frameworks Rails-style in Java.
Really.

Agenda

- Why Groovy?
- DRY Frameworks
- Grails 10,000 foot view
- Interactive Demo
- Going Further...

Who IS that guy?

- Ken Rimple, Chariot Solutions
  - I am overrun with children and dogs
  - 15+ years in IT Consulting
  - I’ve seen TOO MANY frameworks...
  - My kid’s Dinosaurs scare me...

Why Groovy?

- Dynamically typed language
- Runs natively on the VM as bytecode
- Groovyc compiler compiles both Java and Groovy in one pass...
- Uses a superset of Java syntax and dynamic language ‘syntactic sugar’
- Groovy classes can extend Java classes (and vice-versa)

Really Groovy Features

Really Groovy Features

- Groovy is Java without all that messy typing...

Really Groovy Features

- Groovy is Java without all that messy typing...
  - Closures

Really Groovy Features

- Groovy is Java without all that messy typing...
  - Closures
  - Dynamic Typing

Really Groovy Features

- Groovy is Java without all that messy typing...
  - Closures
  - Dynamic Typing
  - Dynamic Class Redefinition

Really Groovy Features

- Groovy is Java without all that messy typing...
  - Closures
  - Dynamic Typing
  - Dynamic Class Redefinition
  - Easy DSL

Really Groovy Features

- Groovy is Java without all that messy typing...
  - Closures
  - Dynamic Typing
  - Dynamic Class Redefinition
  - Easy DSL
  - Groovy-izes Java Classes

Really Groovy Features

- Groovy is Java without all that messy typing...
  - Closures
  - Dynamic Typing
  - Dynamic Class Redefinition
  - Easy DSL
  - Groovy-izes Java Classes
  - Dirt-simple XML parsing support

Java -vs- Groovy

Java -vs- Groovy

Java -vs- Groovy

Java	
POJOs require Explicit Constructors Explicit get/setters .equals and .hashCode Java is noisy No closures (yet) No dynamic typing	

Java -vs- Groovy

Java	Groovy
POJOs require Explicit Constructors Explicit get/setters .equals and .hashCode Java is noisy No closures (yet) No dynamic typing	POGOs require Definition of members That's it! Groovy Supports Dynamic typing (def) Expanding classes Closures Much more...

Java

```
public class Voter {
  private String ssn;
  private String lastName;
  private String firstName;

  public Voter(String ssn, String firstName,
               String lastname) {
    ...
  }

  public void setSsn() { ...}
  public String getSsn() { ...}

  etc...
}
```

Groovy

Groovy

```
class Voter {
  String ssn
  String lastName
  String firstName
}
```

Java

```
Voter v = new Voter(“123-45-6789”, “Jack”,
                    “Beanstalk”);

// what if we want one with just the SSN?
// write a new constructor!

ArrayList list = new ArrayList();
list.add(new Voter(...));
list.add(....);

for (Voter v : lst) {
   ...
}

(lots of ...)
```

Groovy

Groovy

```
// Groovy provides constructors for free...
def v = new Voter(ssn:”123-45-6789”, firstName,
lastName:”Beanstalk”)
def v2 = new Voter(ssn:”123-45-6789”)
```

Groovy

```
// Groovy provides constructors for free...
def v = new Voter(ssn:”123-45-6789”, firstName,
lastName:”Beanstalk”)
def v2 = new Voter(ssn:”123-45-6789”)
```

```
// arraylists are simple
def voters = [
  new Voter(ssn:”234...”),
  new Voter(ssn:”235...”)]

voters += new Voter(ssn:”234-333-4444”)
```

Groovy

```
// Groovy provides constructors for free...
def v = new Voter(ssn:”123-45-6789”, firstName,
lastName:”Beanstalk”)
def v2 = new Voter(ssn:”123-45-6789”)
```

```
// arraylists are simple
def voters = [
  new Voter(ssn:”234...”),
  new Voter(ssn:”235...”)]

voters += new Voter(ssn:”234-333-4444”)
```

```
// An example closure...
voters.each {
  println(“Voter: ${it.ssn}”)
}
```

Groovy “Is” Java

- Groovy is Java without all the noise and with added flexibility
- Groovy compiles to byte code
- Java classes can extend Groovy classes
- Groovy classes can extend Java classes
- Do not have to create an interpreter to use a Groovy class (just add the groovy jar)

Groovy has Elvis!

- With Java:
  - int myVal = somevar != null ? somevar : 0;
- The Elvis Operator
  - int myVal = somevar :? 0
- ELVIS!!!!

- Groovy is java, saying less...

What is Grails?

What is Grails?

- An agile application framework, written in Java and Groovy

What is Grails?

- An agile application framework, written in Java and Groovy
- A rich set of plugins

What is Grails?

- An agile application framework, written in Java and Groovy
- A rich set of plugins
- An easy to understand set of components

What is Grails?

- An agile application framework, written in Java and Groovy
- A rich set of plugins
- An easy to understand set of components
- Can be deployed to a web server as a web application

What is Grails?

- An agile application framework, written in Java and Groovy
- A rich set of plugins
- An easy to understand set of components
- Can be deployed to a web server as a web application
- Able to execute **any major java library or service** on the VM natively

Don’t Repeat Yourself!

- Grails is a DRY platform
- Groovy and Grails aim to remove duplication of effort
- Grails favors convention over configuration where possible

Grails: DRYing out Java

Grails: DRYing out Java

- Code backed by industry standard APIs (Spring, Hibernate, SiteMesh, ACEGI, etc...)

Grails: DRYing out Java

- Code backed by industry standard APIs (Spring, Hibernate, SiteMesh, ACEGI, etc...)
- However, the configuration handled by convention or by simple DSLs

Grails: DRYing out Java

- Code backed by industry standard APIs (Spring, Hibernate, SiteMesh, ACEGI, etc...)
- However, the configuration handled by convention or by simple DSLs
- Do the same work without all that messy typing!!!

Grails: DRYing out Java

- Code backed by industry standard APIs (Spring, Hibernate, SiteMesh, ACEGI, etc...)
- However, the configuration handled by convention or by simple DSLs
- Do the same work without all that messy typing!!!
- AND, to use any Java library, drop it in ./lib and access from Groovy OR Java

Creating a Grails App

- Download Grails from grails.org
- Unzip the files
- Set the GRAILS_HOME path variable
- add $GRAILS_HOME/bin to the path
- type: grails create-app and follow the instructions...

Key Grails Classes

Key Grails Classes

- Domain Class - A class representing an object in your domain (database)

Key Grails Classes

- Domain Class - A class representing an object in your domain (database)
- Controller - A class that operates on URLs submitted to the web site

Key Grails Classes

- Domain Class - A class representing an object in your domain (database)
- Controller - A class that operates on URLs submitted to the web site
- View - A Groovy Server Page (GSP) designed to render the content based on a specific request

Domain Class

- Represent data backed by a datastore
- Backed by Hibernate
- Validated by Spring Validation

- Grails will create tables automatically if configured in DataSource.groovy
- Grails uses Domain Class information to build mappings automatically
- Full Hibernate settings are available if needed using mappings

Sample Domain Class

```
class Party {

    static constraints = {
        name(blank:false)
        description(size:1..5000)
    }

    static hasMany = [candidates: Candidate]

    String name
    String description

    String toString() {
        “Party Name: ${name}”
    }

}
```

Controller

- Analogous to a Struts Action
- Backed by Spring Controllers
- Each method handled by the Controller is a closure

View

- Represents the data that results from a Controller action
- Default view name resolution
  - /grails-app/views/controllername/closure
- Written as a Groovy Server Page (gsp)
- Dirt-simple tag libraries

Creating Grails Classes

- Grails has creation scripts to build the base objects (domains, controllers, views, taglibs, tests, services). Example:

```
grails create-domain-class
grails create-controller
grails create-view
```

- Will prompt for object names if not specified

The Scaffold

The Scaffold

- Sometimes, you just don't know what you want yet...

The Scaffold

- Sometimes, you just don't know what you want yet...
- Why define a page before you nail down the data model?

The Scaffold

- Sometimes, you just don't know what you want yet...
- Why define a page before you nail down the data model?
- Just use a Grails Scaffold

The Scaffold

- Sometimes, you just don't know what you want yet...
- Why define a page before you nail down the data model?
- Just use a Grails Scaffold

```
class VoteController {
    def scaffold = Vote.class
}
```

Grails Demo

- Topics
  - Creating a Grails App
  - The Domain Model
  - Controllers and Scaffolding
  - Generating and modifying views

The Generator Script

- Builds code with default behaviors based on other classes
- Similar to rails' rake task, use the grails generate task to build your elements
  - **`grails generate-views`** *`domain-class`*
  - **`grails generate-all`** *`domain-class`*
- Usually used once the domain model is fleshed out a bit

Grails Workflow - Domain Driven

- Build a domain class for each domain object
- Build a controller for each domain class, but scaffold it to the domain class itself
- Model away, making sure the data mappings are complete
- Finish by generating or coding all views/ controllers/tests
- This helps focus on the data, not the UI, first!

GORM

- Grails Object Relational Mapping API
- Uses domain classes written in Groovy
- Backed by Hibernate and Spring
- Binds validations to the UI and backend
- Write Hibernate objects without all of the messy XML!

GORM Benefits

- Write your domain classes as POGOS
- Define your validation in terms of constraints and get validation for free
- Define your relationships using constraints and get hibernate mapping for free
- And...

GORM Dynamic Finders

- All GORM objects get a findAll() method, and ability to generate queries on the fly. Just type:
  - `def result = domObj.findById(234)`
  - `def results = domObj.findAllByNameOrderByPrice("name")`
- You could also use a GORM DSL for the query...

GORM DSL

- More direct use of Hibernate Criteria:
- Bring back a list of voters who registered within the last 30 days, and are in the Whig party, ordered by last name.

```
def results =
    Voter.withCriteria {
        def now = new Date()
        between('registrationDate', now-30, now)
        party {
          eq(name, 'Whig')
        }
        order('lastName')
    }
```

Just scratching the surface!

- Only so much to cover in one hour...

So, Why Grails?

- Grails is Java App Development on Steroids
- Grails ORM is the proven Hibernate framework, but much easier to stomach
- Grails UI is Spring MVC, including WebFlow, but EASY
- Grails makes writing web pages easier
- Grails can use any Java framework by dropping it in the lib directory. Done!

Grails Plugins

Over 56 At Last Count...	
Rich UI: Ajax, GWT, Echo2, YUI, OpenLazlo, DWR, Flex, etc..)	Graphing with Google, JreeChart, OpenFlash Charting
Testing: Canoo WebTest, Selenium, Coverage	Security with ACEGI, JSecurity, capcha plugins, etc...
JMS, RSS/Atom feed generators, Searching with Compass/Lucene	Remoting and Web Service plugins
Performance and Caching Plugins (S3, ehcache, Static Resources Plugin), Scheduling with Quartz	This is an open community: www.grails.org/plugins

Grails applications in Production Today

- Grails is a 1.0.x release, but...
  - Numerous insurance, financial institutions are using Groovy in applications
  - Grails has been used in production applications since version 0.6
  - Sky launched showbiz.sky.com on Grails this year, 186 million hits / month
- Grails is evolving, but feature rich today

Grails Issues

- Many to Many relationships are not scaffolded today
  - Have to do two many-to-one relationships or customize your GORM models
- Grails only handles a single datasource for GORM at the moment (but you can use Groovy SQL and Hibernate with other datasources)
- Migrations support is lacking in core product (although plugins exist)

On Deck for Grails 1.1

- From Graeme Rocher's talk at G2One
  - Potential JPA Support
  - Portlet support
  - Built-in DB Migrations ala Rails
  - Java Content Repository support (map a domain class to a JCR)
  - Vendor API support (to help with IDE tooling)

Books on Groovy and

- Getting Started with Grails (free e-book, Jason Rudolph) -- available at InfoQ
- Groovy Recipes: Greasing the Wheels of Java (Scott Davis)
- Programming Groovy (Venkat Subramanium)
- The Definitive Guide to Grails (Grame Rocher, but this is out of date)

Great sample app

- Gravl - Glen Smith's Grails-based Blog
  - http://code.google.com/p/gravl/downloads/list
- Excellent example of
- AJAX
- Rich UI (tag clouds, date picker, timeline, more)
- RSS Feed generation
- Searching with the searchable plugin
- Yahoo UI page layout

Resources

- Chariot Resources for ETE, Blogs and Podcasts
  - http://www.chariotsolutions.com/java_lab/podcasts
- Me: krimple@chariotsolutions.com
  - http://www.rimple.com/tech and Chariot TechCast