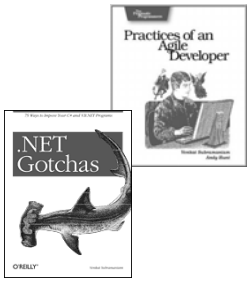
Get Groovier with Grails

Venkat Subramaniam

```
spkr.name = 'Dr. Venkat Subramaniam'
spkr.founder = 'Agile Developer, Inc.'
spkr.affiliated = 'University of Houston'
spkr.associated = 'Rice Univ. Continuing Studies'

spkr.cred = %w{Programmer Speaker Trainer Author}

spkr.nfjs = 'NFJS Speaker since 2002'

spkr.website = 'http://www.agiledeveloper.com'
spkr.email = 'venkats@agiledeveloper.com'
```



Abstract

Inspired by the Ruby on Rails project, Grails brings the ease of web development and "convention over configuration" to the Java platform. We will learn how to create web applications using Grails, how to integrate it with Hibernate, and how to Ajax it, all using the built in features of Grails. This section assumes that you are familiar with Groovy or you have attended the “Groovy for Java Programmers” session. The session will be example driven with live coding where we will build a web application from scratch.

Get Groovier with Grails

- What's Grails
- Grails conventions and Configuration
- Let's Get Coding
- Conclusion

What's Grails

- Open source Web framework
- Groovy Language
- Java Platform
- Effort to bring Ruby-on-Rails (ROR) to Java Platform
- Grails 0.1 release March 2006

Get Groovier with Grails

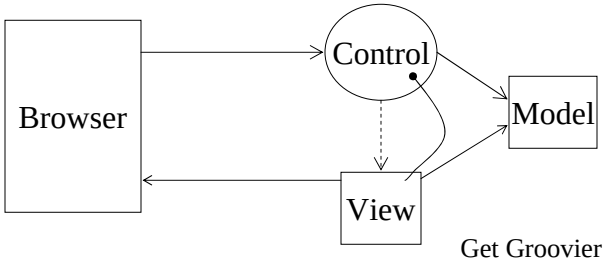
- What's Grails
- Grails conventions and Configuration
- Let's Get Coding
- Conclusion

Grails Conventions & Configuration

- Follows the Ruby and Rails *Convention over configuration* principle
- You name your file in a certain way
- As long as you follow the convention, you are in paradise
  - A bit of a problem for a newbie
  - You love it once you get used to it

Grails MVC

- Grails is built on the concept of MVC
- It actually does not simply promote MVC, it follows you home, sits with you, and makes sure you follow it
- Controller receives request
- You access model to manipulate data
- It transfers control to a view for display



Let's Build and Learn

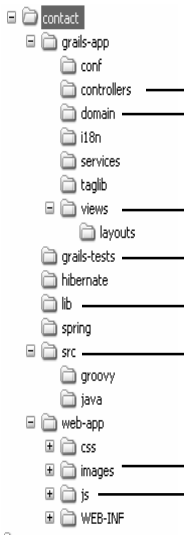
- We will build an application
- We'll learn as we build
- Contact application
  - Gather name, email address, phone number, state in which they live
  - Allow for edit, delete, display list selected based on state (or all)

Creating Grails App

- Grails runs off an ant script

```
C:\workarea\projects>grails create-app
Buildfile: C:\programs\grails\grails-0.1\src\grails\build.xml

init-props:
   [groovy] statements executed successfully

create-app:
     [input] Enter application name:
contact

init-props:
   [groovy] statements executed successfully

init:
    [mkdir] Created dir: C:\workarea\projects\contact\src
    [mkdir] Created dir: C:\workarea\projects\contact\src\java
    [mkdir] Created dir: C:\workarea\projects\contact\src\groovy
    [mkdir] Created dir: C:\workarea\projects\contact\grails-app
    [mkdir] Created dir: C:\workarea\projects\contact\grails-app\controllers
    [mkdir] Created dir: C:\workarea\projects\contact\grails-app\services
    [mkdir] Created dir: C:\workarea\projects\contact\grails-app\domain
    [mkdir] Created dir: C:\workarea\projects\contact\grails-app\tag
    [mkdir] Created dir: C:\workarea\projects\contact\grails-ap
    [mkdir] Created dir: C:\workarea\projects\contact\grail
    [mkdir] Created dir: C:\workarea\projects\contact\
    [mkdir] Created dir: C:\workarea\projects\contac
    [mkdir] Created dir: C:\workarea\projects\cont
    [mkdir] Created dir: C:\workarea\projects\co
    [mkdir] Created dir: C:\workarea\projects\
    [mkdir] Created dir: C:\workarea\projec
    [mkdir] Created dir: C:\workarea\projec
    [mkdir] Created dir: C:\workarea\proje
    [mkdir] Created dir: C:\workarea\pro
    [mkdir] Created dir: C:\workarea\pr
    [mkdir] Created dir: C:\workare
     [copy] Copying 1 file to C:\wo
```


Grails Directory Structure



Controllers with actions
Model classes
One sub-directory per controller
Unit and functional tests
Dependent components
Project specific source
Your logo, etc.
Javascript (for AJAX, etc.)

Code Generation

- Grails uses extensive code generation
- Driven by ant scripts
- Not a whole lot of code being generated however; mostly html generation
- Lot of behind the scene stuff

Domain Class Generation

- Create start up code for class quickly

```
C:\workarea\projects\contact>grails create-domain-class
Buildfile: C:\programs\grails\grails-0.1\src\grails\build.xml

init-props:
   [groovy] statements executed successfully

create-domain-class:
    [input] Enter domain class name:
State
   [groovy] statements executed successfully
     [copy] Copying 1 file to C:\workarea\projects\contact\grails-app\domain
     [echo] Domain class created: grails-app/domain/State.groovy

internal-create-test-suite:
   [groovy] statements executed successfully
     [copy] Copying 1 file to C:\workarea\projects\contact\grails-tests
     [echo] Created test suite: grails-tests/StateTests.groovy

BUILD SUCCESSFUL
Total time: 9 seconds
C:\workarea\projects\contact>_
```

//../contact/grails-app/domain/State.groovy

```
1 class State {
2     @Property Long id
3     @Property Long version
4
5     String toString() { "${this.class.name} :  $id" }
6 }
```

Adding fields

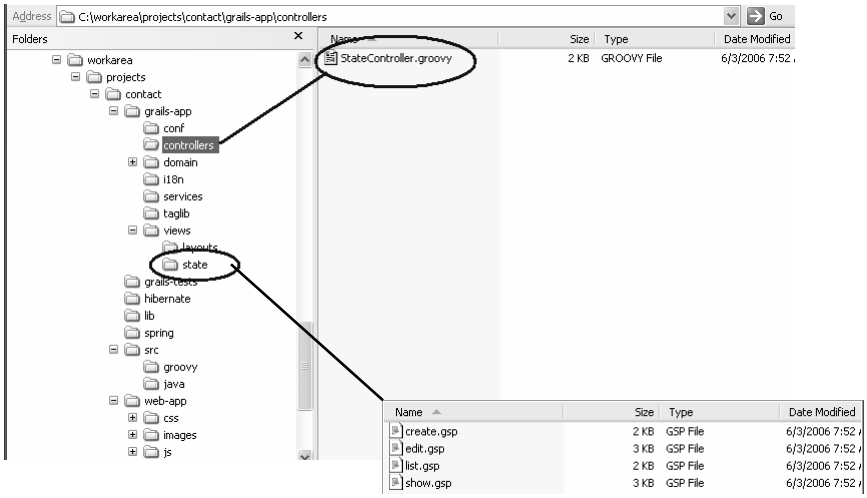
```
class State {
    @Property Long id
    @Property Long version

    @Property String name

    String toString() { "${this.class.name} :  $id" }
}
```


Scaffold

- What about Controller and View
- You can have them created in one shot
- Provides CRUD for Model
  - Create, Read, Update, Delete

```
C:\workarea\projects\contact>grails generate-all
Buildfile: C:\programs\grails\grails-0.1\src\grails\build.xml

init-props:
   [groovy] statements executed successfully

build:

package:
    [mkdir] Created dir: C:\workarea\projects\contact\tmp\war
     [copy] Copying 89 files to C:\workarea\projects\contact\tmp\war

input-domain-class:
    [input] Enter domain class name:
State

generate-all:

generate-controller:impl:
     [java] Loading XML bean definitions from class path resource [applicationCo
ntext.xml]
     [java] Loading XML bean definitions from class path resource [spring/resour
```

Controller and Views Generated



Controller class

```
1  class StateController {
2      @Property index = { redirect(action:list,params:params) }
3
4      @Property list = {
5          if(!params['max']) params['max'] = 10
6          [ stateList: State.list( params ) ]
7      }
8
9      @Property show = {
10         [ state : State.get( params['id'] ) ]
11     }
12
13     @Property delete = {
14         def state = State.get( params['id'] )
15         if(state) {
16             state.delete()
17             flash['message'] = "State ${params['id']} deleted."
18             redirect(action:list)
19         }
           ...
```

Used by View

list View

```
25                          <th>Id</th>
26
27                          <th>Name</th>
28
29                      <th></th>
30                  </tr>
31                  <g:each in="${stateList}">
32                      <tr>
33
34                              <td>${it.id}</td>
35
36                              <td>${it.name}</td>
37
38                          <td class="actionButtons">
39                              <span class="actionButton"><g:link action="show"
    id="${it.id}">Show</g:link></span>
40                          </td>
41                      </tr>
42                  </g:each>
```

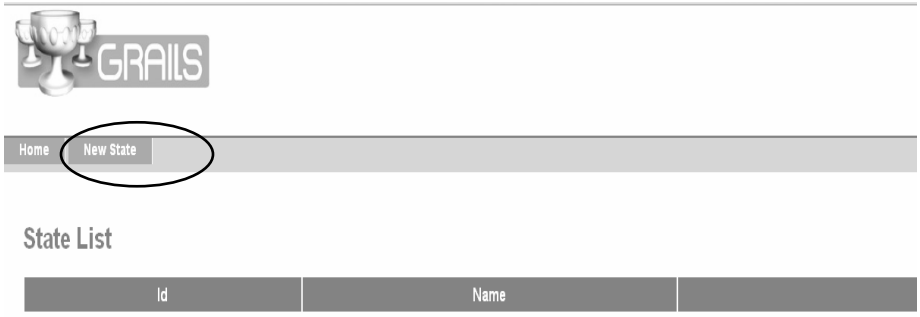

Take it for a test drive

- Run the application to see what we've done so far

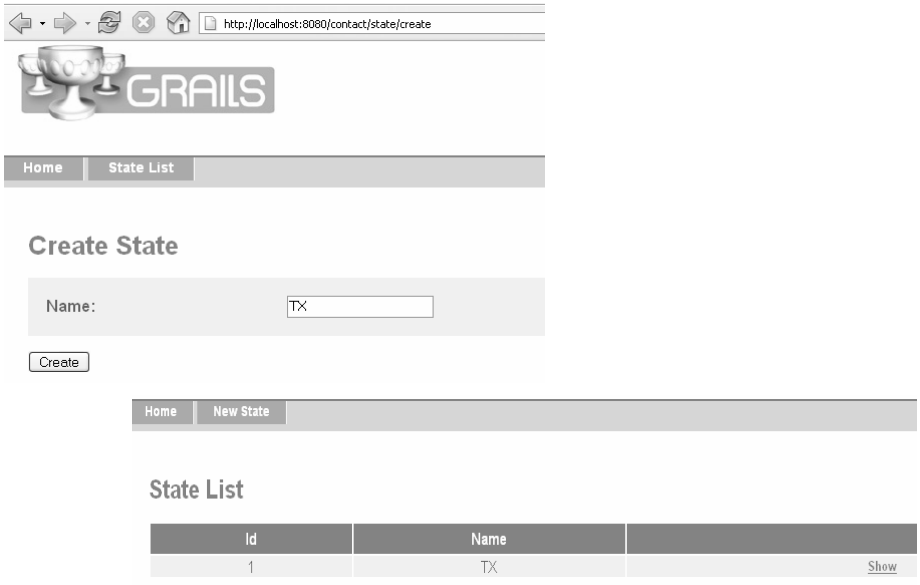
```
C:\workarea\projects\contact>grails run-app
Buildfile: C:\programs\grails\grails-0.1\src\grails\build.xml

dev:

init-props:
   [groovy] statements executed successfully

build:

package:
   [groovy] statements executed successfully

package:checkj5:

package:java5:

run-app:

run-app:impl:
   [groovy] log4j:WARN No appenders could be found for logger (org.mortbay.util.
Container).
   [groovy] log4j:WARN Please initialize the log4j system properly.
   [groovy] Root WebApplicationContext: initialization started
   [groovy] Loading Spring root WebApplicationContext
   [groovy] Loading XML bean definitions from ServletContext resource [/WEB-INF/
```

URL to Controller/Action Mapping

- http://localhost:8080/contact/state/list
- Calls ***list*** action property of ***StateController***
- Routes the view processing to ***list.gsp***



Creating a State



Initializing Data

- By default data is stored in in-memory database
- You can configure to real database
- For test purpose, helps to initialize some data

../contact/grails-app/conf/ApplicationBootStrap.groovy

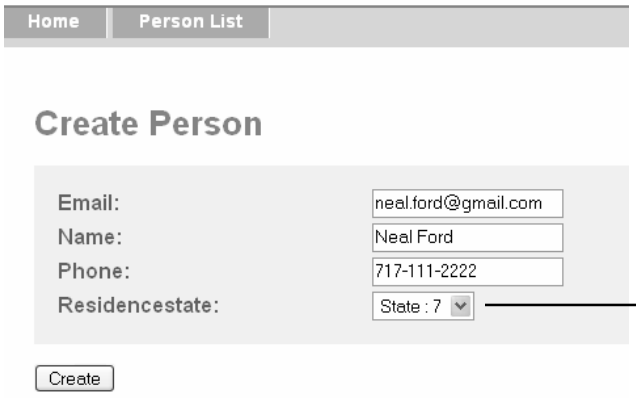
```
1  class ApplicationBootStrap {
2
3       @Property Closure init = { servletContext ->
4          new State(name: 'TX').save()
5          new State(name: 'CA').save()
6          new State(name: 'MA').save()
7          new State(name: 'PA').save()
8          new State(name: 'LA').save()
9          new State(name: 'AZ').save()
10         new State(name: 'FL').save()
```


Linking Models

- grails create-domain-class
  - Person

```
1  class Person {
2      @Property Long id
3      @Property Long version
4
5      @Property belongsTo = State
6
7      @Property String name
8      @Property String email
9      @Property String phone
10     @Property State residence_state
11
12     String toString() { "${this.class.name} :  $id" }
13 }
```

Almost Good Generation



What's that?

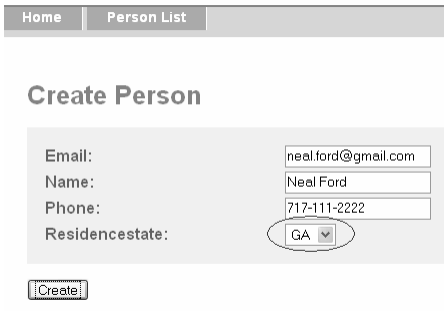
In the View – Calls State's toString() method

```
<g:select optionKey="id" from="${State.list()}"
    name='residence_state.id' value='${person?.residence_state?.id}'>
</g:select>
```


Tweaking the View

- Use the optionValue attribute

```
<g:select optionKey="id" from="${State.list()}"
    name='residence_state.id' value='${person?.residence_state?.id}'
    optionValue="name">
</g:select>
```



Creating your own View

- You can create a Controller and its Views separately

- grails generate-controller
- grails generate-views

Creating Sharable View

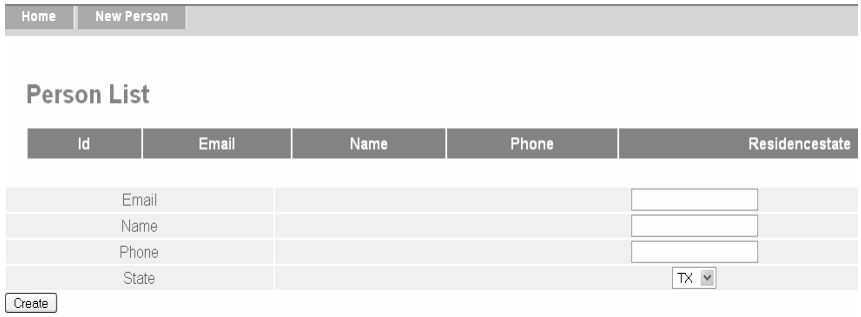
- You can ask grails to render partial content on a page
- Move contents within form from create.gsp in view/person to _addperson.gsp
- Modify create.gsp to

```
<g:form action="save" method="post">
 <g:render template="addperson" bean="Person.new" />
</g:form>
```

Convention: Refer to **_addperson.gsp**

Using partial template

- Modify list.gsp to use the partial template



Move population of list to template

```
<table>
    <tr>
            <th>Id</th>
            <th>Email</th>
            <th>Name</th>
            <th>Phone</th>
            <th>Residencestate</th>
        <th></th>
    </tr>
    <g:render template="listperson" collection="${personList}"/>
</table>
</div>
```

Where are we heading?

- By breaking this into smaller pieces we can promote more reuse of code
- But, what's the real advantage?
- We can update part of the page
- But, why do we want that?
  - AJAX

AJAXing your application

- Grails comes with Prototype and scriptaculous
- May provide alternates later on
- Easy to provide Asynchronous updates and AJAX effects

Using Prototype

```
<g:javascript library="prototype" />
```

```
<g:formRemote url="[controller:'person', action: 'saveperson']"
    method="post" update="mylist" >
    <g:render template="addperson" bean="${new Person()}" />
</g:formRemote>
```


Using Effect

```
<g:javascript library="prototype" />
<g:javascript library="scriptaculous" />
<g:javascript library="effect" />
```

```
<g:formRemote url="[controller:'person', action: 'saveperson']"
    method="post" update="mylist"
    onComplete="new Effect.BlindDown('mylist')">
    <g:render template="addperson" bean="${new Person()}" />
</g:formRemote>
```

Get Groovier with Grails

- What's Grails
- Grails conventions and Configuration
- Let's Get Coding
- Conclusion

Coding a Grails App

- Enough of talking, let's get coding
- A full fledge application using Grails
  - Creating application
  - Creating domain class
  - Creating controller
  - Creating views
  - AJAXing the application
- Download the code example from http://www.agiledeveloper.com
  - Click on download

Quiz Time

Get Groovier with Grails

- What's Grails
- Grails conventions and Configuration
- Let's Get Coding
- Conclusion

Conclusion

- +
  - Grails based on ROR
  - Convention over Configuration
  - Very cool to bring Rail concept to Java platform
  - Leverages your Java investments
- -
  - Honestly, does not fully feel like Rails, but close enough
  - Relatively new, pretty early in its release

References

1. http://grails.org/
2. Agile Web Development with Rails
   http://pragmaticprogrammer.com/titles/rails/index.html
3. Download examples/slides from

http://www.agiledeveloper.com/download.aspx

Thanks!

Please fill out your evaluations!