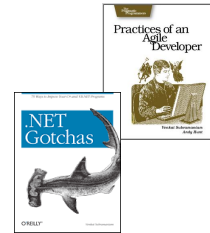


# Agile Web Development with Grails

```
spkr.name = 'Venkat Subramaniam'  
spkr.company = 'Agile Developer, Inc.'  
spkr.credentials = %w{Programmer Trainer Author}  
spkr.blog = 'agiledeveloper.com/blog'  
spkr.email = 'venkats@agiledeveloper.com'
```



## Abstract

- Agile development is all about developing code and seeking feedback from your users to make sure you're developing what's relevant. When they suggest changes, those must be affordable and reliable. Grails, along with its facility to develop test driven, is a killer combination for rapidly developing web applications. In this ZePo (Zero PowerPoint) presentation, we will take a test driven approach to developing a small but fully functional web application in Grails. We will cover the fundamental features of Grails along with utilizing other capabilities like Ajax. At the end of this presentation, you not only be confident, but eager to roll your own web application using Grails.

# Agenda

- ☀ What's Grails?
  - Principles
  - Fundamentals
  - A Quick App
  - Phase I
  - Unit Testing
  - Functional Testing
  - Phase II
  - Templating
  - Ajax
  - Conclusion

## What's Grails?

- Open Source Web Framework
- Inspired by Rails, but not a port
- Uses Groovy and Java
- Built for JVM
- Rapid Development of CRUD Applications
  - But you can leverage Java strengths
- This presentation based on Grail 0.4

# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajax
- Conclusion

## Grails Principles

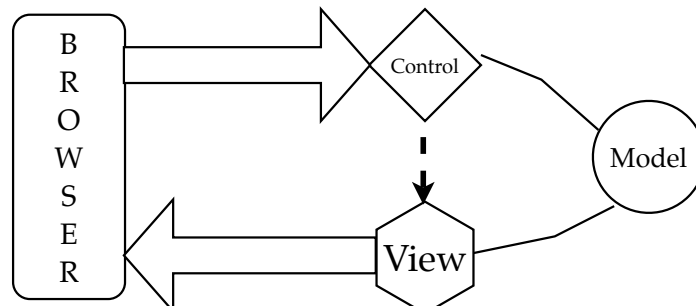
- DRY
- Convention Over Configuration
  - A bit of a problem when you're new
  - Once you get used to it, you're in Paradise
- Keep it Simple
- Leverage Java Platform

# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajax
- Conclusion

## Grails MVC

- Grails built on concept of MVC
- It does not merely recommend that you use MVC
- It tells you to do so, then follows you home, and sits next to you to make sure you do



# Agenda

- What's Grails?
- Principles
- Fundamentals
- ✿ A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajax
- Conclusion

# Build and Learn

- A Contacts Application
- Add name, email, phone, state of residence
- Edit, list, ...

```

> grails help

Welcome to Grails 0.4 - http://grails.org/
Licensed under Apache Standard License 2.0
Grails home is set to: /usr/local/bin/grails-0.4

Base Directory: /Users/venkats/workarea
Environment set to development
Note: No plugin scripts found
Running script /usr/local/bin/grails-0.4/scripts/Help.groovy

Usage (Optionals marked with *):
grails [environment]* [target] [arguments]*

Examples:
grails dev run-app
grails create-app books

Available Targets:
grails bootstrap -- This task will load the Grails application con
d 'ctx'
grails bug-report -- Creates a ZIP containing source artifacts for
grails clean -- Cleans a Grails project
grails compile -- Performs compilation on any source files (Java o
grails console -- Load the Grails interactive Swing console
grails create-app -- Creates a Grails project, including the neces
grails create-controller -- Creates a new controller
grails create-domain-class -- Creates a new domain class
grails create-job -- Creates a new Quartz scheduled job
grails create-job -- Creates a new Quartz scheduled job
grails create-job -- Creates a new Quartz scheduled job
grails create-job -- Creates a new Quartz scheduled job

```

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- ☀ Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templatizing
- Ajaxing
- Conclusion

# Create App

```
Terminal — bash — 115x31
> grails create-app contact

Welcome to Grails 0.4 - http://grails.org/
Licensed under Apache Standard License 2.0
Grails home is set to: /usr/local/bin/grails-0.4

Base Directory: /Users/venkats/workarea
Environment set to production
Note: No plugin scripts found
Running script /usr/local/bin/grails-0.4/scripts/CreateApp.groovy
[mkdir] Created dir: /Users/venkats/workarea/contact/src
[mkdir] Created dir: /Users/venkats/workarea/contact/src/java
[mkdir] Created dir: /Users/venkats/workarea/contact/src/groovy
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/controllers
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/services
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/domain
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/taglib
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/utils
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/views
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/views/layouts
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/i18n
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/conf
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-tests
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/conf
[mkdir] Created dir: /Users/venkats/workarea/contact/grails-app/conf
```

## Directory Structure

- contact
  - build.xml
  - contact.launch
  - grails-app
    - conf
      - ApplicationBootstrap.groovy
      - DevelopmentDataSource.groovy
      - log4j.development.properties
      - log4j.production.properties
      - log4j.test.properties
      - ProductionDataSource.groovy
      - TestDataSource.groovy
    - controllers
    - domain
    - i18n
      - messages.properties
    - services
    - taglib
    - utils
    - views
      - error.gsp
      - layouts
        - main.gsp
    - grails-tests
    - hibernate
    - lib
    - plugins
      - core
        - grails-app
          - taglib
          - utils
    - scripts
    - spring
      - resources.xml
    - src
      - groovy
      - java
    - web-app
      - css
      - images
      - index.jsp
      - js
      - META-INF
      - WEB-INF

# Code Generation

- Grails script generates quite a bit of stuff for you
  - Mostly html
  - Code is mostly synthesized rather than being generated

# Three Configurations

- Creates three configurations
  - Production
  - Development
  - Testing



The image shows two snippets of Groovy code. The first snippet, from `TestDataSource.groovy`, defines a `TestDataSource` class with properties for pooling, database creation, URL, driver class name, username, and password. The second snippet, from `ApplicationBootstrap.groovy`, defines an `ApplicationBootstrap` class with `init` and `destroy` methods.

```
1 class TestDataSource {
2   boolean pooling = true
3   String dbCreate = "update" // one of 'create', 'create-drop', 'update'
4   String url = "jdbc:hsqldb:mem:testDB"
5   String driverClassName = "org.hsqldb.jdbcDriver"
6   String username = "sa"
7   String password = ""
8 }
```

```
1 class ApplicationBootstrap {
2
3   def init = { servletContext ->
4   }
5   def destroy = {
6   }
7 }
```



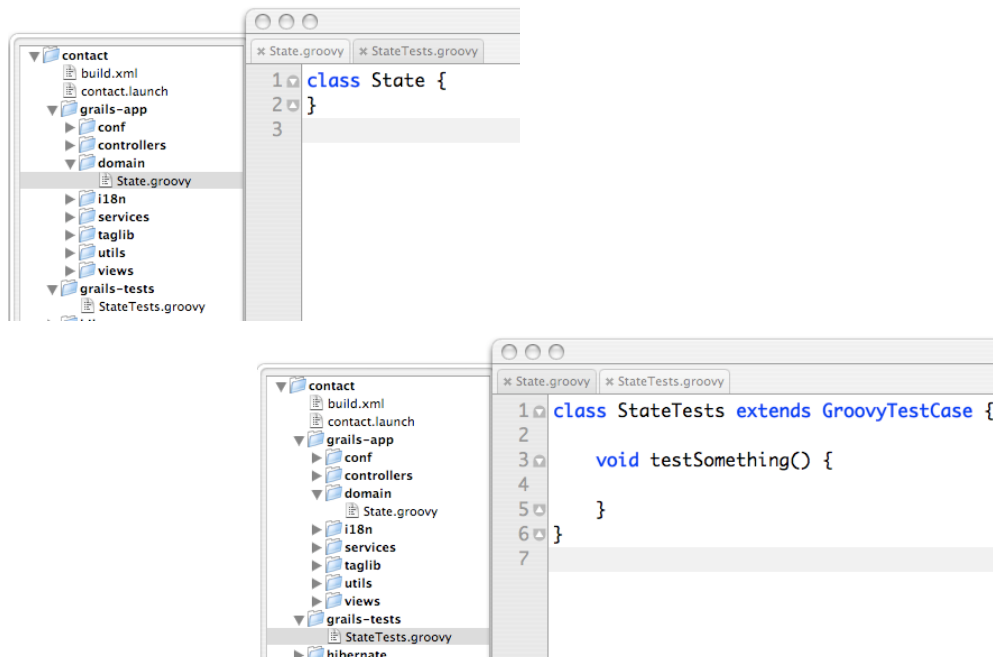
# Create Domain class

```
Terminal — bash — 115x31
> pwd
/Users/venkats/workarea/contact
>

Welcome to Grails 0.4 - http://grails.org/
Licensed under Apache Standard License 2.0
Grails home is set to: /usr/local/bin/grails-0.4

Base Directory: /Users/venkats/workarea/contact
Environment set to development
Running script /usr/local/bin/grails-0.4/scripts/CreateDomainClass.groovy
name not specified. Please enter:
[copy] Copying 1 file to /Users/venkats/workarea/contact/grails-app/domain
Created at /Users/venkats/workarea/contact/grails-app/domain/State.groovy
[copy] Copying 1 file to /Users/venkats/workarea/contact/grails-tests
Created Tests at /Users/venkats/workarea/contact/grails-tests/StateTests.groovy
> 
```

# Generated Code



# Editing Domain Class

```
class State
{
    String twoLetterCode
}
```

# Generating Controller and Views

```
Terminal — bash — 115x39
> grails generate-all State

Welcome to Grails 0.4 - http://grails.org/
Licensed under Apache Standard License 2.0
Grails home is set to: /usr/local/bin/grails-0.4

Base Directory: /Users/venkats/workarea/contact
Environment set to production
Running script /usr/local/bin/grails-0.4/scripts/GenerateAll.groovy
Compiling sources..
[delete] Deleting directory /Users/venkats/workarea/contact/web-app/WEB-INF/grails-app
[mkdir] Created dir: /Users/venkats/workarea/contact/web-app/WEB-INF/grails-app/views
[mkdir] Created dir: /Users/venkats/workarea/contact/web-app/WEB-INF/grails-app/i18n
[copy] Copying 2 files to /Users/venkats/workarea/contact/web-app/WEB-INF/grails-app/views
[native2ascii] Converting 1 file from /Users/venkats/workarea/contact/grails-app/i18n to /Users/venkats/workarea/co
ntact/web-app/WEB-INF/grails-app/i18n
Attempting to load [13] core plugins
Grails plug-in [i18n] with version [0.4] loaded successfully
...
Generating views for domain class [State]
Generating list view for domain class [State]
list view generated at /Users/venkats/workarea/contact/./grails-app/views/state/list.gsp
...
Generating controller for domain class [State]
Controller generated at ./grails-app/controllers/StateController.groovy
>
```

# Controller

```
class StateController {
  def index = { redirect(action:list,params:params) }

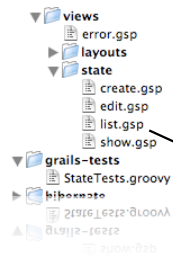
  // the delete, save and update actions only
  // accept POST requests
  def allowedMethods = [delete:'POST',
                        save:'POST',
                        update:'POST']

  def list = {
    if(!params.max)params.max = 10
    [ stateList: State.list( params ) ]
  }

  def show = {
    [ state : State.get( params.id ) ]
  }

  def delete = {
    def state = State.get( params.id )
    if(state) {
```

# Views



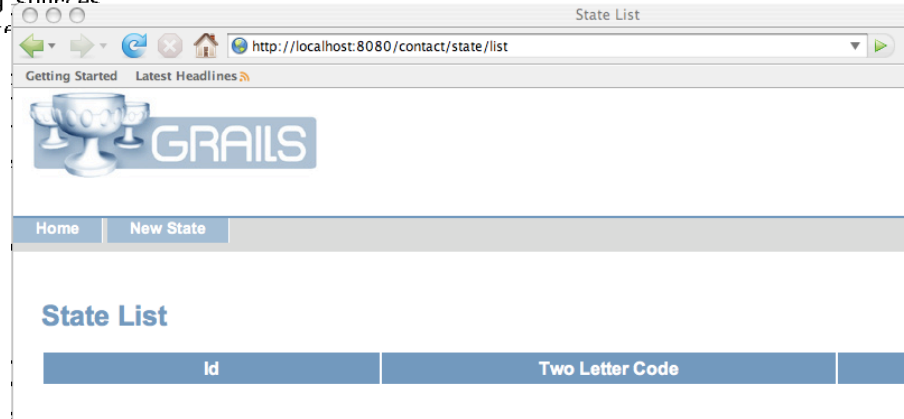
```
<html>
...
<body>
...
<div class="body">
  <h1>State List</h1>
  <g:if test="${flash.message}">
    <div class="message">
      ${flash.message}
    </div>
  </g:if>
  <table>
    <thead>
      <tr>
        <th>Id</th>
        <th>Two Letter Code</th>
      </tr>
    </thead>
    <tbody>
      <g:each in="${stateList}">
        <tr>
          <td>${it.id?.encodeAsHTML()}</td>
          <td>${it.twoLetterCode?.encodeAsHTML()}</td>
          <td class="actionButtons">
            <span class="actionButton"><g:link action="show"
id="${it.id}">Show</g:link></span>
          </td>
        </tr>
      </g:each>
    </tbody>
  </table>
</div>
```

# A Quick Test Drive

```
Terminal — java — 1  
> grails run-app
```

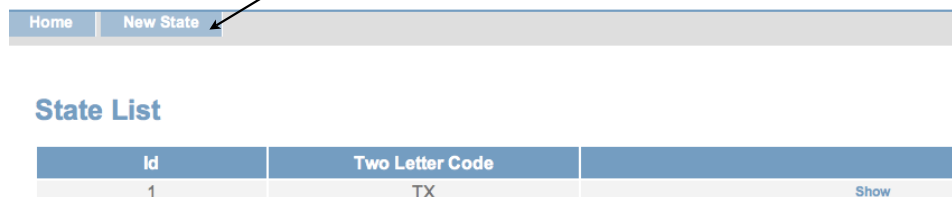
```
Welcome to Grails 0.4 - http://grails.org/  
Licensed under Apache Standard License 2.0  
Grails home is set to: /usr/local/bin/grails-0.4
```

```
Base Directory: /Users/venkats/workarea/contact  
Environment set to development  
Running script /usr/local/bin/grails-0.4/scripts/RunApp.groovy  
Compiling sources  
Deleting
```



# A Quick Test Drive...

Created New State by clicking on this link



# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajax
- Conclusion

## Test Driven Development

- A way to design an application
- Helps find problems quickly
- As system evolves, let us know if we meet the expectations
- Serves as a valuable form of documentation
- Safety net for refactoring
- Write Positive, Negative, and Exception tests

# Test First Domain Logic

```
class State
{
    String twoLetterCode
}

class StateTests extends GroovyTestCase {
    void testStateCodeTwoLetters()
    {
        State state = new State(TwoLetterCode: 'Texas')

        assertFalse state.validate()
    }
}
```

```
>grails test-app
Welcome to Grails 0.4 - http://grails.org/
```

...

Running script /usr/local/bin/grails-0.4/scripts/TestApp.groovy

...

There was 1 failure:

1) testStateCodeTwoLetters(StateTests)junit.framework.AssertionFailedError  
at gjdk.StateTests\_GroovyReflector.invoke(Unknown Source)

...

FAILURES!!!

Tests run: 1, Failures: 1, Errors: 0

Test Failures!!!

# Code Next Domain Logic

```
class State
{
    String twoLetterCode

    static constraints = {
        twoLetterCode(size: 2..2)
    }
}
```

```
>grails test-app
```

...

.

Time: 0.031

OK (1 test)

# After Two More Tests...

```
class StateTests extends GroovyTestCase {  
  
    void testStateCodeTwoLetters()  
    {  
        def state = new State(TwoLetterCode: 'Texas')  
  
        assertFalse state.validate()  
    }  
  
    void testStateCodeNotEmpty()  
    {  
        def state = new State(TwoLetterCode: '')  
        assertFalse state.validate()  
    }  
  
    void testStateCodeUnique()  
    {  
        def state = new State(TwoLetterCode: 'TX')  
        state.save()  
  
        def anotherState = new State(TwoLetterCode: 'TX')  
        assertFalse anotherState.validate()  
    }  
}
```

```
class State  
{  
    String twoLetterCode  
  
    static constraints = {  
        twoLetterCode(size: 2..2, blank: false, unique: true)  
    }  
}
```

>grails test-app  
...  
...  
Time: 0.082  
OK (3 tests)

## Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajax
- Conclusion

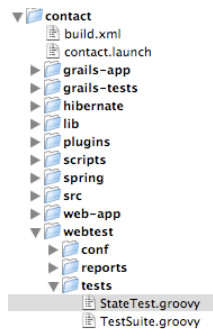
# Integration Testing

- Functional Testing or Integration Testing is automated testing of functionality
- Very helpful when done in addition to unit testing
- Grails uses Canoo for this

## Generate Web Test

```
> grails generate-webtest
```

```
...
Environment set to development
Running script /usr/local/bin/grails-0.4/scripts/GenerateWebtest.groovy
Domain class name not specified. Please enter:
State
[get] Getting: http://webtest.canoo.com/webtest/build.zip
[get] To: /usr/local/bin/grails-0.4/downloads/webtest.zip
[get] local file date : Thu Feb 08 14:11:24 CST 2007
.....
[unzip] Expanding: /usr/local/bin/grails-0.4/downloads/webtest.zip into /usr/local/bin/grails-0.4/downloads/web
test
[copy] Copying 3 files to /Users/venkats/workarea/contact
[copy] Copying 1 file to /Users/venkats/workarea/contact/webtest/tests
Web Test generated at webtest/tests/StateTest.groovy
```





# Generated Web Test

```
class StateTest extends grails.util.WebTest {

    // Unlike unit tests, functional tests are often sequence dependent.
    // Specify that sequence here.
    void suite() {
        testStateListNewDelete()
        // add tests for more operations here
    }

    def testStateListNewDelete() {
        webtest('State basic operations: view list, create new entry, view, edit, delete, view'){
            invoke(url:'state')
            verifyText(text:'Home')

            verifyListPage(0)

            clickLink(label:'New State')
            verifyText(text:'Create State')
            clickButton(label:'Create')
            verifyText(text:'Show State', description:'Detail page')
            clickLink(label:'List', description:'Back to list view')

            verifyListPage(1)
            verifyListPage(2)

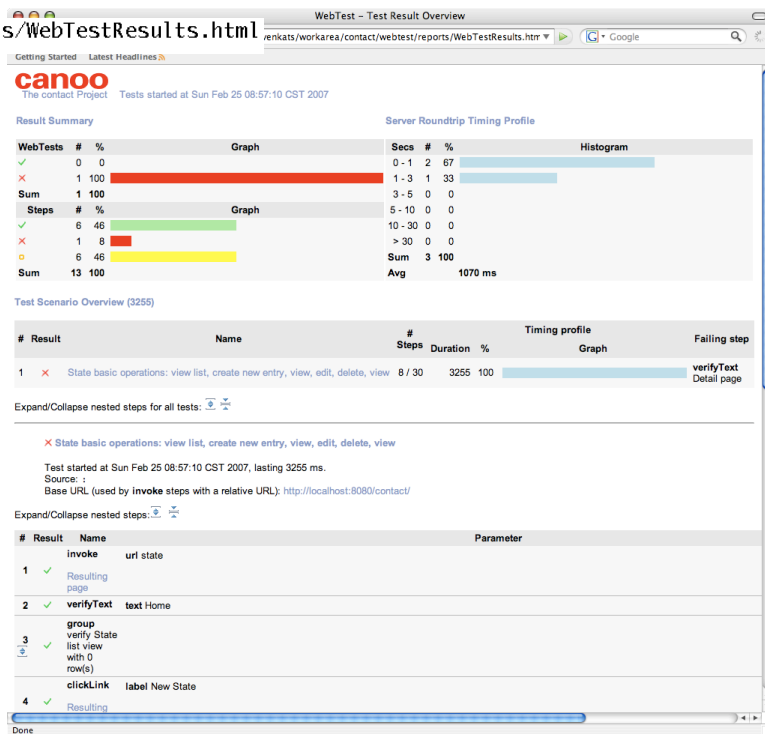
            clickLink(label:'List', description:'Back to list view')
            verifyText(text:'Home')
            clickLink(label:'List', description:'Back to list view')
        }
    }
}
```

# Running Generated Tests

```
>grails run-webtest
...
Environment set to test
Running WebTest!
Testing contact
[delete] Deleting 5 files from /Users/venkats/workarea/contact/webtest/reports
[style] Transforming into /Users/venkats/workarea/contact/webtest/reports
[style] Processing /Users/venkats/workarea/contact/webtest/reports/WebTestResults.xml to /Users/venkats/workarea/contact/webtest/reports/WebTestResults.html
[style] Loading stylesheet /usr/local/bin/grails-0.4/downloads/webtest/resources/WebTestReport.xsl
Closing application context [org.codehaus.groovy.grails.commons.spring.GrailsWebApplicationContext;hashCode=10224031]
Closing Hibernate SessionFactory
```

# Viewing Test Results

> open webtest/reports/WebTestResults.html



## Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajax
- Conclusion

# Building Second Domain Class

- We will continue to build the next Domain class—Person
- Steps similar to what we've done earlier
  - Create Domain Class
  - Generate Controller and Views

## Person Class

```
class Person
{
  String name
  String email
  String phoneNumber
  State residenceState

  def belongsTo = State
}
```

# Test Driving

Where's our State?

Was in the in-memory database  
Missing State won't give confidence to our customer

# Bootstrap Data

```
class ApplicationBootstrap {  
  
    def init = { servletContext ->  
        new State(TwoLetterCode: 'TX').save()  
        new State(TwoLetterCode: 'GA').save()  
        new State(TwoLetterCode: 'CA').save()  
        new State(TwoLetterCode: 'MA').save()  
        new State(TwoLetterCode: 'CO').save()  
        new State(TwoLetterCode: 'IL').save()  
    }  
    def destroy = {  
    }  
}
```

# Revisit Create Person

Home | Person List

### Create Person

Email:

Name:

Phone:

Residence State: 

State : 1  
State : 2  
State : 3  
State : 4  
State : 5

Create

???

## A Quick Fix

```
<tr class='prop'><td valign='top' class='name'><label  
for='residenceState'>Residence State:</label></td><td valign='top' class='value  
${hasErrors(bean:person,field:'residenceState','errors')}><g:select optionKey="id" from="${State.list()}"  
name='residenceState.id' value="${person?.residenceState?.id}" optionValue="name"></g:select></td></tr>
```

Add optionValue element  
to the select in create.gsp

Home | Person List

### Create Person

Email:

Name:

Phone:

Residence State: 

TX  
MA  
CO  
NY  
CA

Create

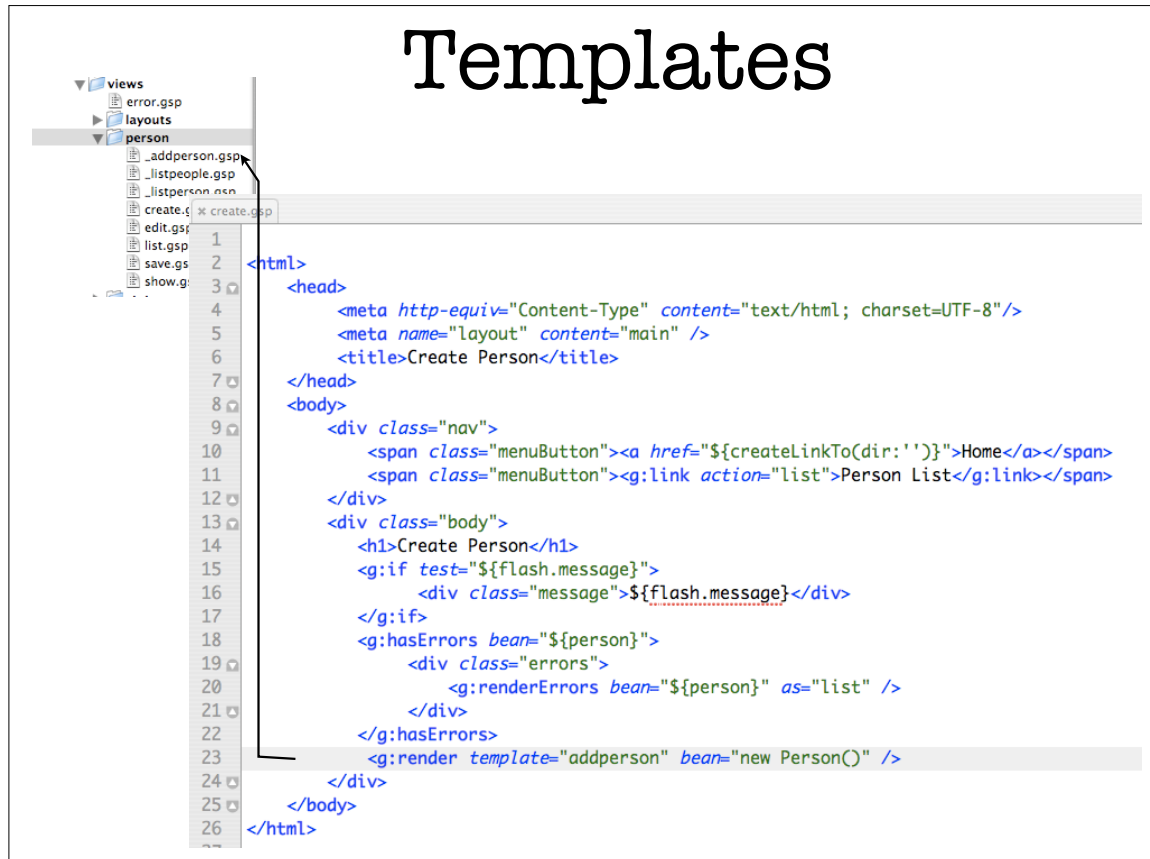
# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templatizing
- Ajaxing
- Conclusion

# Templates

- Reusable fragments
- File naming convention starts with \_
- You can render as a bean or collection
  - bean results in one expansion
  - collection results in one expansion per element of collection

# Templates



The screenshot shows a web application editor. On the left is a file tree with the following structure:

- views
  - error.gsp
  - layouts
    - person
      - \_addperson.gsp
      - \_listpeople.gsp
      - \_listperson.nos
      - create.gsp
      - edit.gsp
      - list.gsp
      - save.gsp
      - show.gsp

The main editor displays the content of `create.gsp`, which is a GSP template. The code is as follows:

```
1 <html>
2 <head>
3   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
4   <meta name="layout" content="main" />
5   <title>Create Person</title>
6 </head>
7 <body>
8   <div class="nav">
9     <span class="menuButton"><a href="${createLinkTo(dir: '')}">Home</a></span>
10    <span class="menuButton"><g:link action="list">Person List</g:link></span>
11  </div>
12  <div class="body">
13    <h1>Create Person</h1>
14    <g:if test="${flash.message}">
15      <div class="message">${flash.message}</div>
16    </g:if>
17    <g:hasErrors bean="${person}">
18      <div class="errors">
19        <g:renderErrors bean="${person}" as="list" />
20      </div>
21    </g:hasErrors>
22    <g:render template="addperson" bean="new Person()" />
23  </div>
24 </body>
25 </html>
```

# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templatizing
- Ajaxing
- Conclusion

# Ajaxing Frameworks

- Very easy to integrate with Ajax frameworks
- Ships with Prototype
- You may use other frameworks as well

## Ajaxing Create

```
1  <html>
2
3  <head>
4      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
5      <meta name="layout" content="main" />
6      <title>Person List</title>
7  </head>
8  <body>
9      <div class="nav">
10         <span class="menuButton"><a href="${createLinkTo(dir:')}">Home</a></span>
11         <span class="menuButton"><g:link action="create">New Person</g:link></span>
12     </div>
13     <div class="body">
14         <h1>Person List</h1>
15         <g:if test="${flash.message}">
16             <div class="message">
17                 ${flash.message}
18             </div>
19         </g:if>
20         <div id="persondiv">
21             <g:render template="listpeople" bean="personList"></g:render>
22         </div>
23         <g:render template="addperson" bean="new Person()" />
24     </body>
25 </html>
```



# Ajaxing Create...

```
1 <table id="persontable">
2   <thead>
3     <tr>
4       <th>Id</th>
5
6       <th>Email</th>
7
8       <th>Name</th>
9
10      <th>Phone</th>
11
12      <th>Residence State</th>
13
14    </th></tr>
15  </thead>
16  <tbody>
17    <g:render template="listperson" collection="{personList}" />
18  </tbody>
19 </table>
20
21 <div class="paginateButtons">
22   <g:paginate total="{Person.count()}" />
23 </div>
```

# Ajaxing Create...

```
1 <tr>
2   <td>${it.id?.encodeAsHTML()}</td>
3
4   <td>${it.email?.encodeAsHTML()}</td>
5
6   <td>${it.name?.encodeAsHTML()}</td>
7
8   <td>${it.phone?.encodeAsHTML()}</td>
9
10  <td>${it.residenceState?.encodeAsHTML()}</td>
11
12  <td class="actionButtons">
13    <span class="actionButton"><g:link action="show" id="${it.id}>Show</g:link></span>
14  </td>
15 </tr>
```

# Ajaxing Create...



```

1 <g:javascript library="prototype" />
2 <g:javascript library="scriptaculous" />
3 <g:javascript library="effects" />
4
5 <g:formRemote uri="[controller: 'person', action: 'save']" name="saveperson" method="post"
6 update="persondiv"
7 onComplete="alert(\$(persontable).rows.length); new Effect.Pulsate(\$(persontable).rows[\$(persontable).rows.length
8 - 1]);">
9 <div class="dialog">
10 <table>
11 <tbody>
12
13
14

```

# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajaxing
- Conclusion

# Agenda

- What's Grails?
- Principles
- Fundamentals
- A Quick App
- Phase I
- Unit Testing
- Functional Testing
- Phase II
- Templating
- Ajaxing
- Conclusion

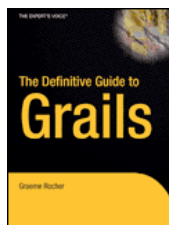
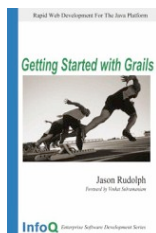
- # Agenda
- What's Grails?
  - Principles
  - Fundamentals
  - A Quick App
  - Phase I
  - Unit Testing
  - Functional Testing
  - Phase II
  - Templating
  - Ajaxing
  - Conclusion

# Quiz Time



## References

- <http://grails.codehaus.org>
- <http://grails.codehaus.org/Download>
- <http://groovy.codehaus.org>
- <http://groovy.codehaus.org/Download>
- <http://aboutgroovy.com>



# Thank You!

<http://www.agiledeveloper.com> — download