

# What's New in Grails 2.0

*Burt Beckwith*  
*SpringSource*



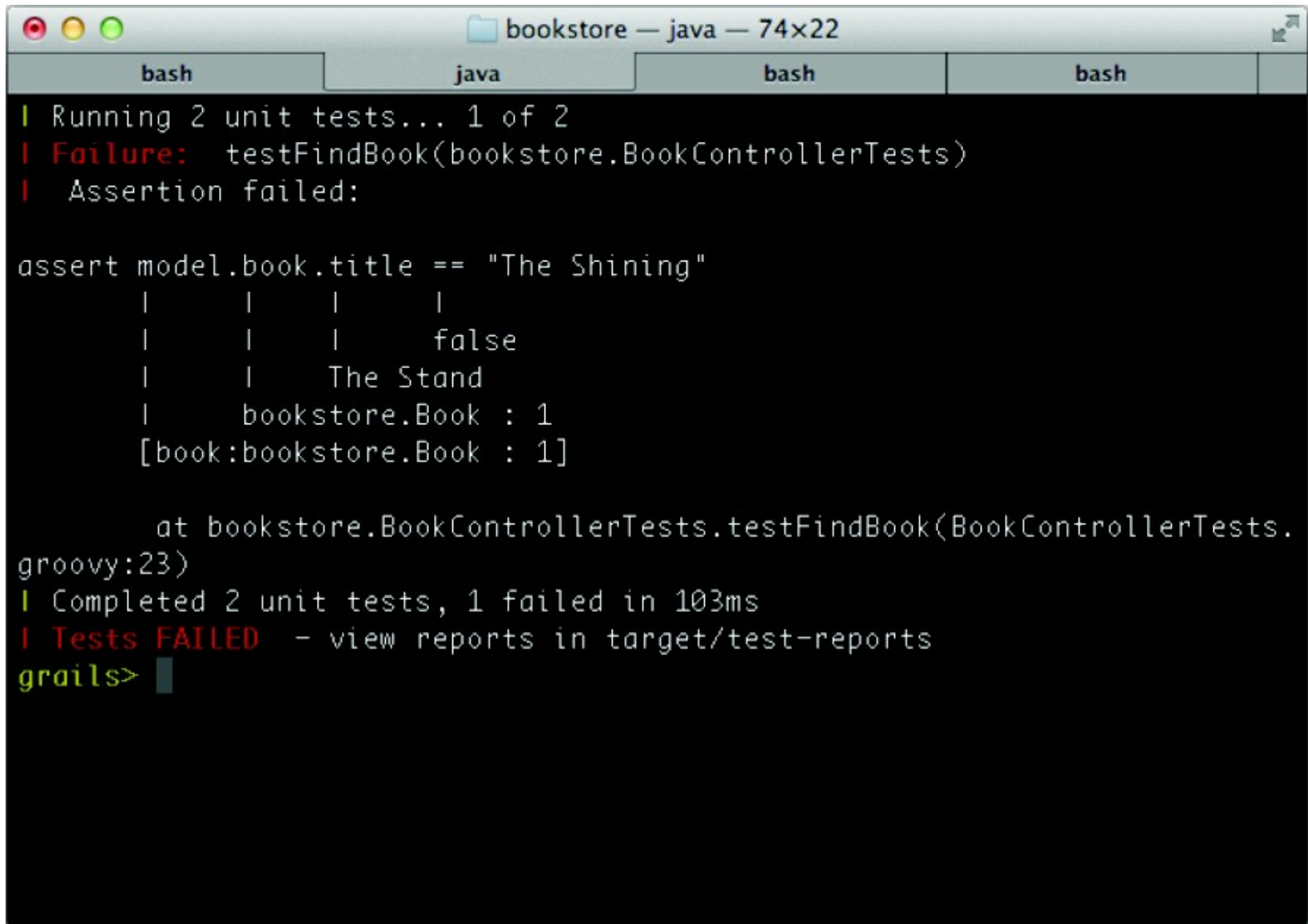
# Who Am I

---

- Java developer for over 12 years
- Background in Spring, Hibernate, Spring Security
- Grails developer for 4 years
- SpringSource employee on the Grails team
- Created or reworked over 35 Grails plugins
- <http://burtbeckwith.com/blog/>
- <https://twitter.com/#!/burtbeckwith>

# Development Environment Features

# New Console UI & Interactive Mode



The screenshot shows a terminal window titled "bookstore — java — 74x22". The terminal output is as follows:

```
bash
java
bash
bash

| Running 2 unit tests... 1 of 2
| Failure: testFindBook(bookstore.BookControllerTests)
| Assertion failed:

assert model.book.title == "The Shining"
|   |   |   |
|   |   |   false
|   |   The Stand
|   bookstore.Book : 1
| [book:bookstore.Book : 1]

      at bookstore.BookControllerTests.testFindBook(BookControllerTests.
groovy:23)
| Completed 2 unit tests, 1 failed in 103ms
| Tests FAILED - view reports in target/test-reports
grails> █
```



## Unit Test Results - Summary

Executed 12 tests with 4 failures .

test

Executed 12 tests with 4 failures .



BookControllerTests



BookTests



testShow

Executed in 0.12 seconds.

### Assertion failed: assert book.sav

```
junit.framework.AssertionFailedError
assert book.save() != null
|      |      |
|      null  false
test.Book : null

at test.BookController1
at TestApp$_run_closure
at TestApp$_run_closure
at TestApp$_run_closure
```

# Better Documentation Template

[Table of contents](#)

[Quick Reference](#)



See the light - agile, industrial strength, rapid web application development made easy

## The Grails Framework - Reference Documentation

**Authors:** Graeme Rocher, Peter Ledbrook, Marc Palmer, Jeff Brown, Luke Daley, Burt Beckwith

**Version:** 2.0.0

### Table of Contents

- 1** Introduction
  - 1.1** What's new in Grails 2.0?
    - 1.1.1** Development Environment Features
    - 1.1.2** Core Features
    - 1.1.3** Web Features
    - 1.1.4** Persistence Features
    - 1.1.5** Testing Features
- 2** Getting Started
  - 2.1** Installation Requirements
  - 2.2** Downloading and Installing
  - 2.3** Upgrading from previous versions of Grails

### Quick Reference [\(hide\)](#)

- [Command Line](#)
- [Constraints](#)
- [Controllers](#)
- [Database Mapping](#)
- [Domain Classes](#)
- [Plug-ins](#)
- [Services](#)
- [Servlet API](#)
- [Tag Libraries](#)
- [Tags](#)

# Enhanced Error Reporting



## Error 500: Internal Server Error

**URI:** /bookstore/book/find

**Class:** groovy.lang.MissingPropertyException

**Message:** No such property: titl for class: bookstore.BookService

### Around line 6 of *grails-app/services/bookstore/BookService.groovy*

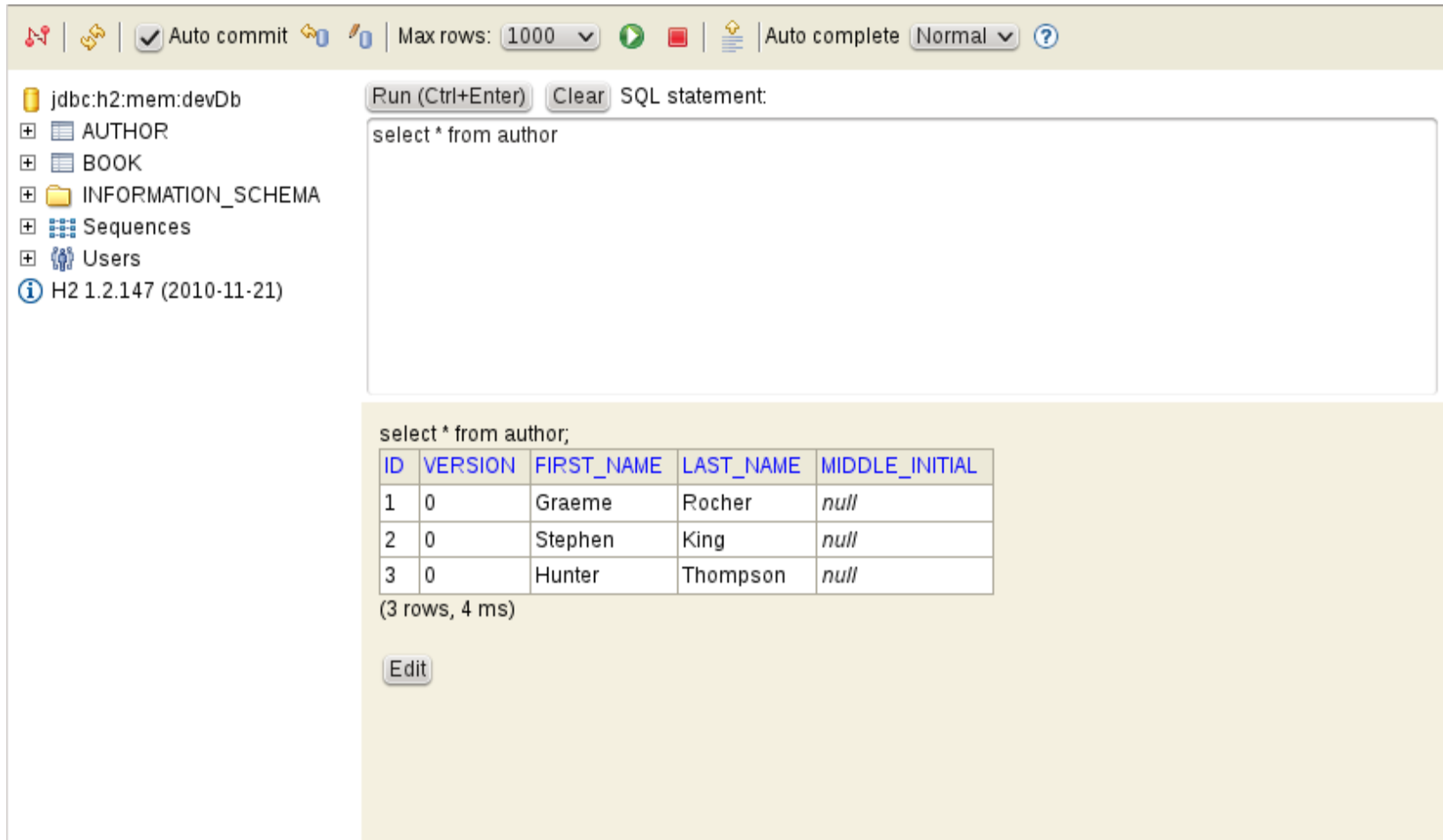
```
3: class BookService {
4:
5:     Book findByTitle(String title) {
6:         Book.findByTitle(titl)
7:     }
8: }
```

### Around line 10 of *grails-app/controllers/bookstore/BookController.groovy*

```
7:     def bookService
8:     def find() {
9:
10:         def b = bookService.findByTitle(params.title)
11:
12:         [book:b]
```

# Database Console

- Available at <http://localhost:8080/appname/dbconsole> in dev environment; can be enabled in other environments



The screenshot shows the Database Console interface. At the top, there is a toolbar with icons for refresh, save, auto-commit, max rows (set to 1000), and auto-complete (set to Normal). Below the toolbar, the left sidebar shows the database structure for 'jdbc:h2:mem:devDb', including tables 'AUTHOR' and 'BOOK', a folder 'INFORMATION\_SCHEMA', 'Sequences', and 'Users'. The main area displays the SQL statement 'select \* from author' in a text input field. Below the input field, the results of the query are shown in a table format. The table has five columns: ID, VERSION, FIRST\_NAME, LAST\_NAME, and MIDDLE\_INITIAL. There are three rows of data, each with a version of 0 and a null middle initial. Below the table, it indicates '(3 rows, 4 ms)' and an 'Edit' button.

Run (Ctrl+Enter) Clear SQL statement:

```
select * from author
```

ID	VERSION	FIRST_NAME	LAST_NAME	MIDDLE_INITIAL
1	0	Graeme	Rocher	<i>null</i>
2	0	Stephen	King	<i>null</i>
3	0	Hunter	Thompson	<i>null</i>

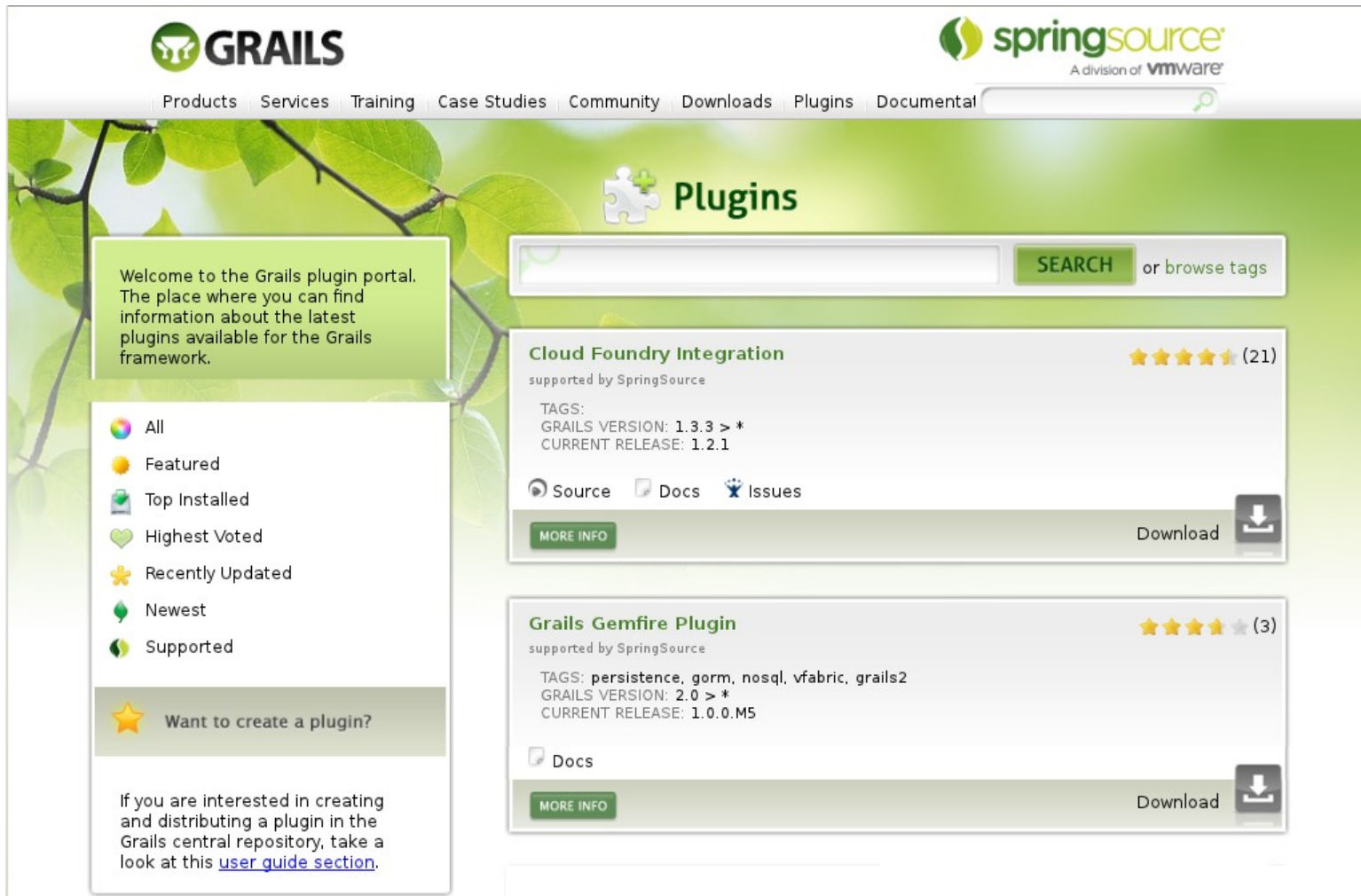
(3 rows, 4 ms)

Edit



# Plugin Portal Usage Tracking

## ■ Opt-in usage tracking of plugins



The screenshot displays the Grails Plugin Portal interface. At the top, the Grails logo is on the left and the SpringSource logo (A division of VMware) is on the right. A navigation menu includes links for Products, Services, Training, Case Studies, Community, Downloads, Plugins, and Documental. A search bar is located to the right of the menu.

The main content area is titled "Plugins" and features a search bar with a "SEARCH" button and the text "or browse tags". Below this, there are two plugin listings:

- Cloud Foundry Integration**: supported by SpringSource, 5 stars (21). TAGS: GRAILS VERSION: 1.3.3 > \*, CURRENT RELEASE: 1.2.1. Includes links for Source, Docs, and Issues. Buttons for "MORE INFO" and "Download" are present.
- Grails Gemfire Plugin**: supported by SpringSource, 5 stars (3). TAGS: persistence, gorm, nosql, vfabric, grails2. GRAILS VERSION: 2.0 > \*, CURRENT RELEASE: 1.0.0.M5. Includes a link for Docs. Buttons for "MORE INFO" and "Download" are present.

A sidebar on the left contains a welcome message: "Welcome to the Grails plugin portal. The place where you can find information about the latest plugins available for the Grails framework." Below this is a list of filters: All, Featured, Top Installed, Highest Voted, Recently Updated, Newest, and Supported. At the bottom of the sidebar, there is a star icon and the text "Want to create a plugin?". Below that, a message states: "If you are interested in creating and distributing a plugin in the Grails central repository, take a look at this [user guide section](#)."

# Upgraded Libraries

---



# What's new in Groovy 1.8?

# What's new in Groovy 1.8?

---

- Nicer DSLs with command chains expressions
- Runtime performance improvements
- GParas bundled
- Closure enhancements
- Builtin JSON support
- New AST transformations

# Command Chains Expressions

---

- Simple example:
  - **turn left then right**
  - Was **turn(left).then(right)**
  
- These are equivalent:
  - **take 2.pills, of: chloroquinine, after: 6.hours**
  - **take(2).pills(of).chloroquinine(after).6(hours)**
  - **take 2 pills of chloroquinine after 6 hours**

# Runtime performance improvements

---

- Significant runtime improvements for primitive type operations
  - Classical Fibonacci example 13x faster!
  - almost as fast as Java
- Some direct method calls
- Current work being done for static type checking
  - <http://blackdragsview.blogspot.com/2011/10/feeling-grumpy.html>

- GPar is bundled in the Groovy distribution
- Covers a wide range of parallel and concurrent paradigms:
  - Actors
  - Fork/join
  - Map/filter/reduce
  - Dataflow
  - Agents
  - STM
  - Parallel arrays
  - Executors
  - and more ...

# Closure enhancements

---

- Closure annotation parameters
- Some more functional flavor
  - composition
  - trampoline
  - memoization
- Currying improvements



## Closure annotation parameters

---

```
@Retention(RetentionPolicy.RUNTIME)
@interface Invariant {
    Class value() // a closure class
}

@Invariant({number >= 0 })
class Distance {
    float number
    String unit
}

def d = new Distance(number: 10, unit: "meters")
def anno = Distance.getAnnotation(Invariant)
def check = anno.value().newInstance(d, d)
assert check(d)
```

# Built in JSON support

- Consuming
- Producing
- Pretty-printing

```
import groovy.json.*

def json = new JsonBuilder()
json.person {
    name 'Guillaume'
    age 33
    pets 'Hector', 'Felix'
}
println json.toString()
println json.toPrettyString()
```

```
{"person":{"name":"Guillaume","age":33,"pets":["Hector","Felix"]}}
```

```
{
  "person": {
    "name": "Guillaume",
    "age": 33,
    "pets": [
      "Hector",
      "Felix"
    ]
  }
}
```

# New AST transformations

---

- @Log
- @Field
- @AutoClone
- @AutoExternalizable
- @ThreadInterrupt,  
@TimedInterrupt,  
@ConditionalInterrupt
- @InheritConstructor
- @Canonical
  - @ToString
  - @EqualsAndHashCode
  - @TupleConstructor
- @WithReadLock
- @WithWriteLock

# New Automatic Reloading

---

# New Automatic Reloading

---

- Reloading in run-app works with
  - Typed service references
  - Domain classes
  - src/groovy, src/java

# New Automatic Reloading

---

- Reloading in run-app works with
  - Typed service references
  - Domain classes
  - src/groovy, src/java
- Any command with -reloading

# New Automatic Reloading

---

- Reloading in run-app works with
  - Typed service references
  - Domain classes
  - src/groovy, src/java
- Any command with -reloading
- Interactive mode and integration tests

# Binary Plugins

---

- Package pre-compiled plugins into JAR files
- Deployable as standard JARs to Maven repositories
- Declared as JAR dependencies
- Commercial plugins more viable
- No special IDE integration needed


**\$ grails package-plugin –binary**




# Web Features

# HTML5 Scaffolding




 Home

 Book List

## Create Book

Title \*

 Create

# New APIs

---

## ■ Page Rendering

```
PageRenderer groovyPageRenderer
void welcomeUser(User user) {
    def contents = groovyPageRenderer.render(
        view: "/emails/welcome",
        model: [user: user])
    ...
}
```

## ■ Link Generation

```
LinkGenerator railsLinkGenerator
String generateLink() {
    railsLinkGenerator.link(
        controller: "book", action: "list")
}
```

# Advanced Static Resource Handling

---

- Integrated resource plugin into core
  - <http://grails.org/plugin/resources>
- Tuning static resources no longer a headache
  - gzip (<http://grails.org/plugin/zipped-resources>)
  - cache (<http://grails.org/plugin/cached-resources>)
  - De-duplication
  - Bundling
- New tags to ease integration
  - img
  - external
  - javascript

# Bundling Static Resources

---

```
modules = {
  core {
    dependsOn 'utils'
    resource url: '/js/core.js', disposition: 'head'
    resource url: '/js/ui.js'
    resource url: '/css/main.css'
    resource url: '/css/branding.css'
    resource url: '/css/print.css', attrs: [media: 'print']
  }
  utils {
    dependsOn 'jquery'
    resource url: '/js/utils.js'
  }
}
```

# Zipping and Caching

```
$ grails install-plugin cached-resources
$ grails install-plugin zipped-resources
```

Or add dependency in BuildConfig.groovy:

```
plugins {
    runtime ":hibernate:$grailsVersion"
    runtime ":jquery:1.7.1"
    runtime ":resources:1.1.5"
    build ":tomcat:$grailsVersion"

    runtime ":cached-resources:1.0"
    runtime ":zipped-resources:1.0"
}
```

# Persistence Features







mongoDB



***Cassandra***

redis



HIBERNATE



- Plugins should not assume Hibernate is available

# GORM Plugins

---

- Redis - <http://grails.org/plugin/redis-gorm>
- MongoDB - <http://grails.org/plugin/mongodb>
- Amazon SimpleDB - <http://grails.org/plugin/simpledb>
- Neo4j - <http://grails.org/plugin/neo4j>
- Riak - <http://grails.org/plugin/riak>
- GORM JPA - <http://grails.org/plugin/gorm-jpa>
- Hibernate - <http://grails.org/plugin/hibernate>

# Where Queries

---

# Where Queries

---

- New, compile-time checked query DSL

```
def query = Person.where {  
    firstName == "Bart"  
}  
Person bart = query.find()
```

# Where Queries

---

- New, compile-time checked query DSL

```
def query = Person.where {  
    firstName == "Bart"  
}  
Person bart = query.find()
```

- Uses native Groovy operators ==, !=, >, <, <=, >= etc.

```
def query = Person.where {  
    firstName == "Fred" && !(lastName == 'Simpson')  
}
```

# Where Queries

- New, compile-time checked query DSL

```
def query = Person.where {  
    firstName == "Bart"  
}  
Person bart = query.find()
```

- Uses native Groovy operators ==, !=, >, <, <=, >= etc.

```
def query = Person.where {  
    firstName == "Fred" && !(lastName == 'Simpson')  
}
```

- Aggregate functions supported avg, sum, max, min etc.

```
def query = Person.where {  
    age > avg(age)  
}
```

# Multiple Data Sources

---

- Support for defining multiple scoped data sources

```
class ZipCode {  
    String code  
    static mapping = {  
        datasource 'auditing'  
    }  
}
```



# Multiple Data Sources

---

- Support for defining multiple scoped data sources

```
class ZipCode {  
    String code  
    static mapping = {  
        datasource 'auditing'  
    }  
}
```

- Each data source accessible via static property

```
def zipCode = ZipCode.auditing.get(42)
```

Hibernate 'update'  
+  
Production data  
=  
?

Hibernate 'update'

+

Production data

=



# GORM Plugins

---

- Install the Database Migration plugin:

```
$ grails install-plugin database-migration
```

- Official Docs at:

- <http://grails-plugins.github.com/grails-database-migration/>

Pre-production, Hibernate 'update' or 'create-drop'

# SQL Database Migration

---

Pre-production, Hibernate 'update' or 'create-drop'



dbm-generate-changelog  
dbm-changelog-sync

# SQL Database Migration

---

Pre-production, Hibernate 'update' or 'create-drop'



dbm-generate-changelog  
dbm-changelog-sync



Change domain model



dbm-gorm-diff  
dbm-update

# SQL Reverse Engineering

---



# SQL Reverse Engineering

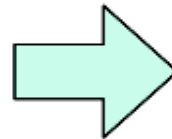
---

```
$ grails install-plugin db-reverse-engineer  
$ grails db-reverse-engineer
```

# SQL Reverse Engineering

```
$ grails install-plugin db-reverse-engineer  
$ grails db-reverse-engineer
```

PERSON	
ID	INT
VERSION	INT
NAME	VARCHAR
AGE	INT
DATE_CREATED	TIMESTAMP



```
class Person {  
    String name  
    Integer age  
    Date dateCreated  
    ...  
}
```

# Other GORM Improvements

---

- Abstract base domain classes
  - These now result in a table

# Other GORM Improvements

---

- Abstract base domain classes
  - These now result in a table
- `findOrCreateWhere()`
- `findOrCreateSaveWhere()`

# Other GORM Improvements

---

- Abstract base domain classes
  - These now result in a table
- findOrCreateWhere()
- findOrSaveWhere()

```
def user = User.findByLogin('admin')
if (!user) {
    user = new User(login: 'admin')
    user.save(failOnError: true)
}
```

# Other GORM Improvements

---

- Abstract base domain classes
  - These now result in a table
- findOrCreateWhere()
- findOrSaveWhere()

```
def user = User.findByLogin('admin')
if (!user) {
    user = new User(login: 'admin')
    user.save(failOnError: true)
}
```

```
def user = User.findOrSaveWhere(login: 'admin')
```

# Better Unit Testing

# Unit Testing Pre-2.0

---



# Unit Testing Pre-2.0

---

- • mockDomain() had only partial GORM support
  - always lagged changes in GORM

# Unit Testing Pre-2.0

---

- • mockDomain() had only partial GORM support
  - always lagged changes in GORM
- Inheritance-based
  - hierarchy duplicated for Spock
  - difficult to extend

# Unit Testing Pre-2.0

---

- • mockDomain() had only partial GORM support
  - always lagged changes in GORM
- Inheritance-based
  - hierarchy duplicated for Spock
  - difficult to extend
- Weak support for web-related testing
  - controllers
  - tag libraries

# The Mixin Approach

---

```
@TestFor(MyController)  
@Mock(Person)  
class MyControllerUnitTests {  
    protected void setUp() {  
        new Person(...).save()  
        new Person(...).save()  
    }  
  
    void testIndex() {  
        def model = controller.index()  
        ...  
    }  
}
```

# Support for testing...

---

# Support for testing...

---

- Tag libraries

# Support for testing...

---

- Tag libraries
- Command objects

# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses



# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses
- File upload

# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses
- File upload
- View and template rendering

# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses
- File upload
- View and template rendering
- Filters

# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses
- File upload
- View and template rendering
- Filters
- URL mappings

# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses
- File upload
- View and template rendering
- Filters
- URL mappings
- Criteria queries

# Support for testing...

---

- Tag libraries
- Command objects
- XML & JSON responses
- File upload
- View and template rendering
- Filters
- URL mappings
- Criteria queries
- and more!

# Grails in the Cloud

---



**CLOUD FOUNDRY**<sup>TM</sup> BETA  
DEPLOY & SCALE YOUR APPLICATIONS IN SECONDS



# For the Future

---

- A continued focus on
  - Reliability
  - User experience
  - Modularity
  - More cloud



# Demo

**Thank You**

**Questions?**