What's New in Grails 2.0

Burt Beckwith

SpringSource

springsource
A division of vmware

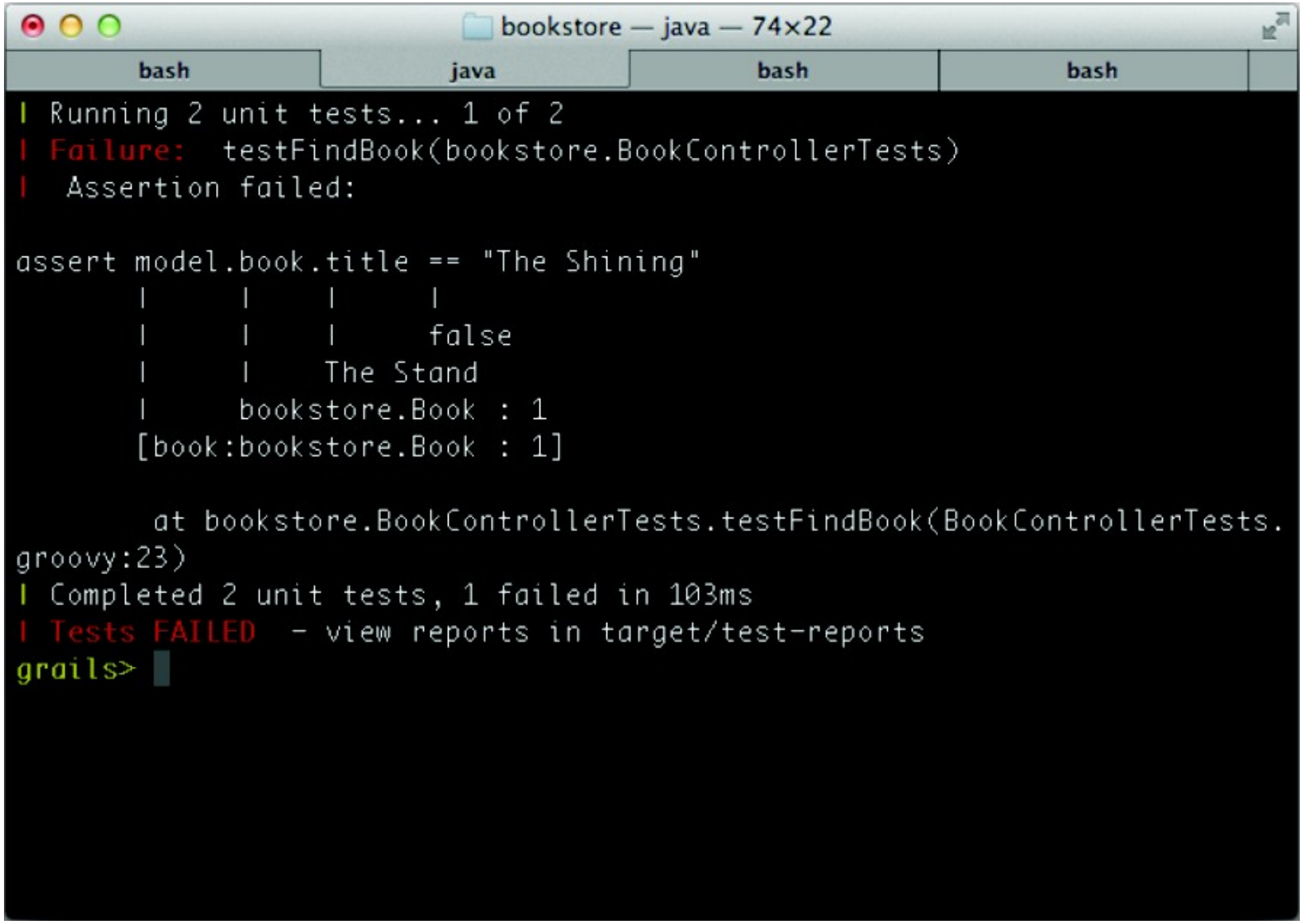
CONFIDENTIAL

Who Am I

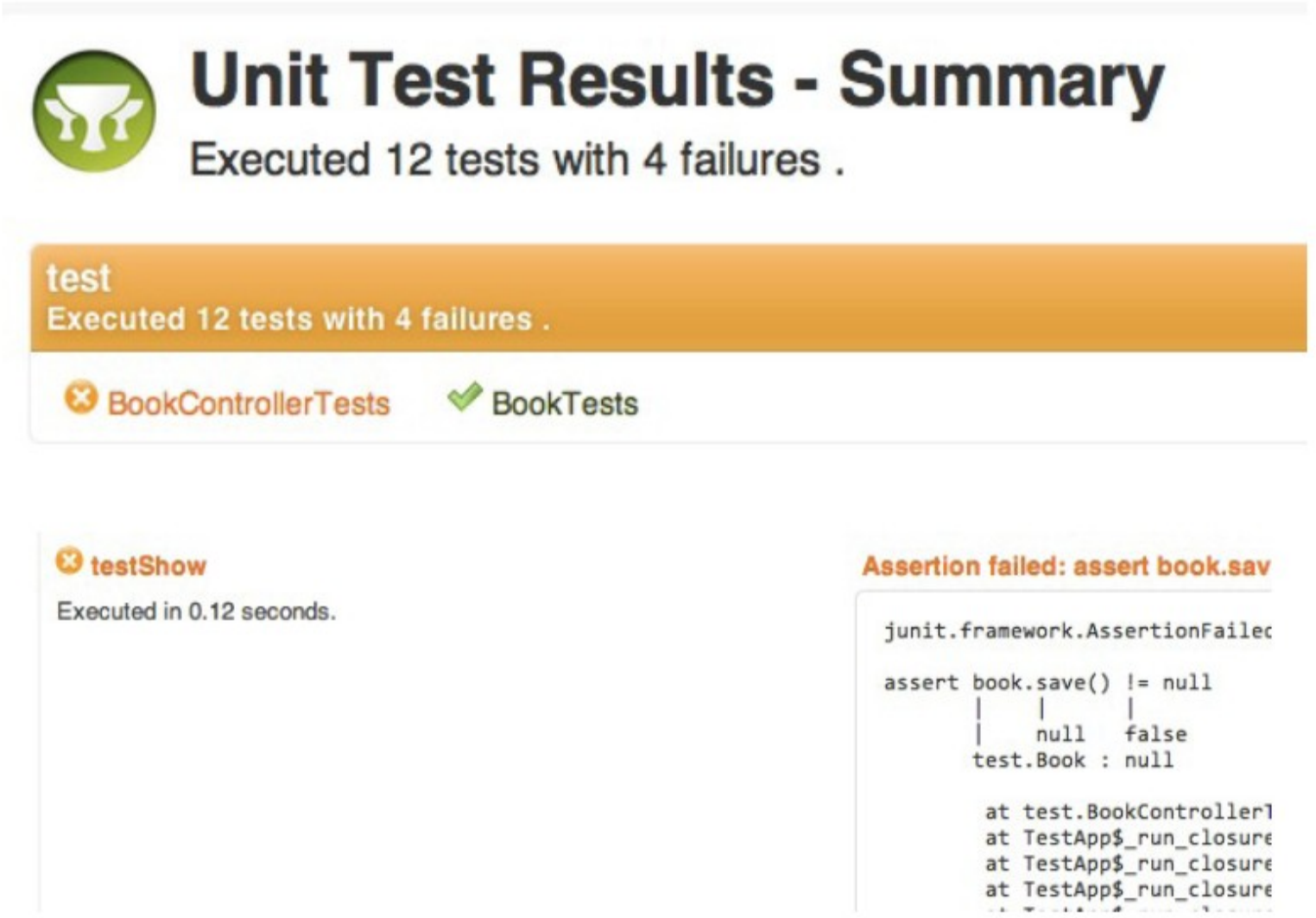
- Java developer for over 12 years
- Background in Spring, Hibernate, Spring Security
- Grails developer for 4 years
- SpringSource employee on the Grails team
- Created or reworked over 35 Grails plugins
- http://burtbeckwith.com/blog/
- https://twitter.com/#!/burtbeckwith

Development Environment Features

New Console UI & Interactive Mode



Better Unit Test Template



Better Documentation Template

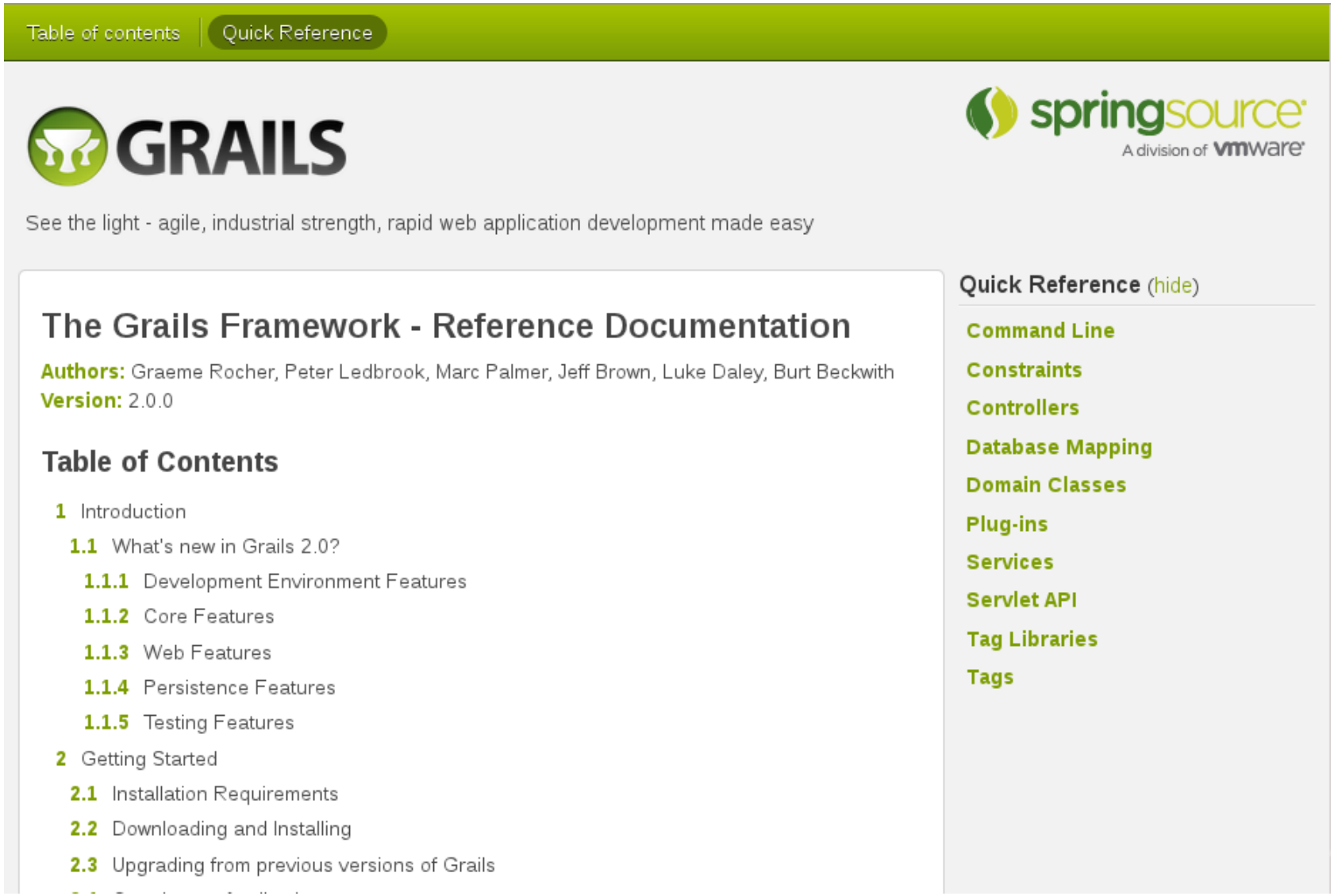


Table of contents | Quick Reference

GRAILS

springsource A division of vmware

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

GRAILS

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*

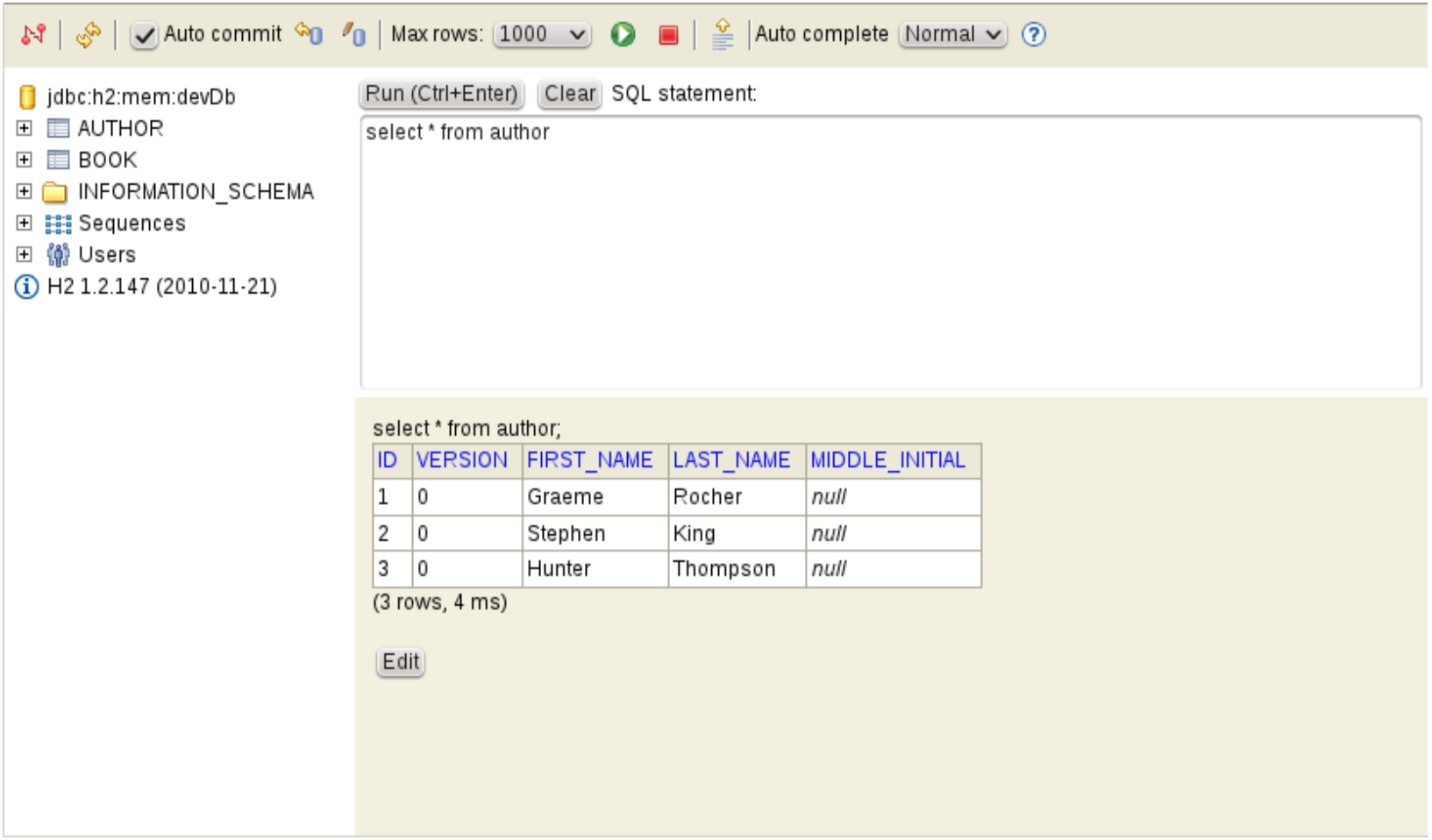
```
3: class BookService {
4:
5:     Book findByTitle(String title) {
6:         Book.findByTitle(titl)
7:     }
8: }
```

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]
```

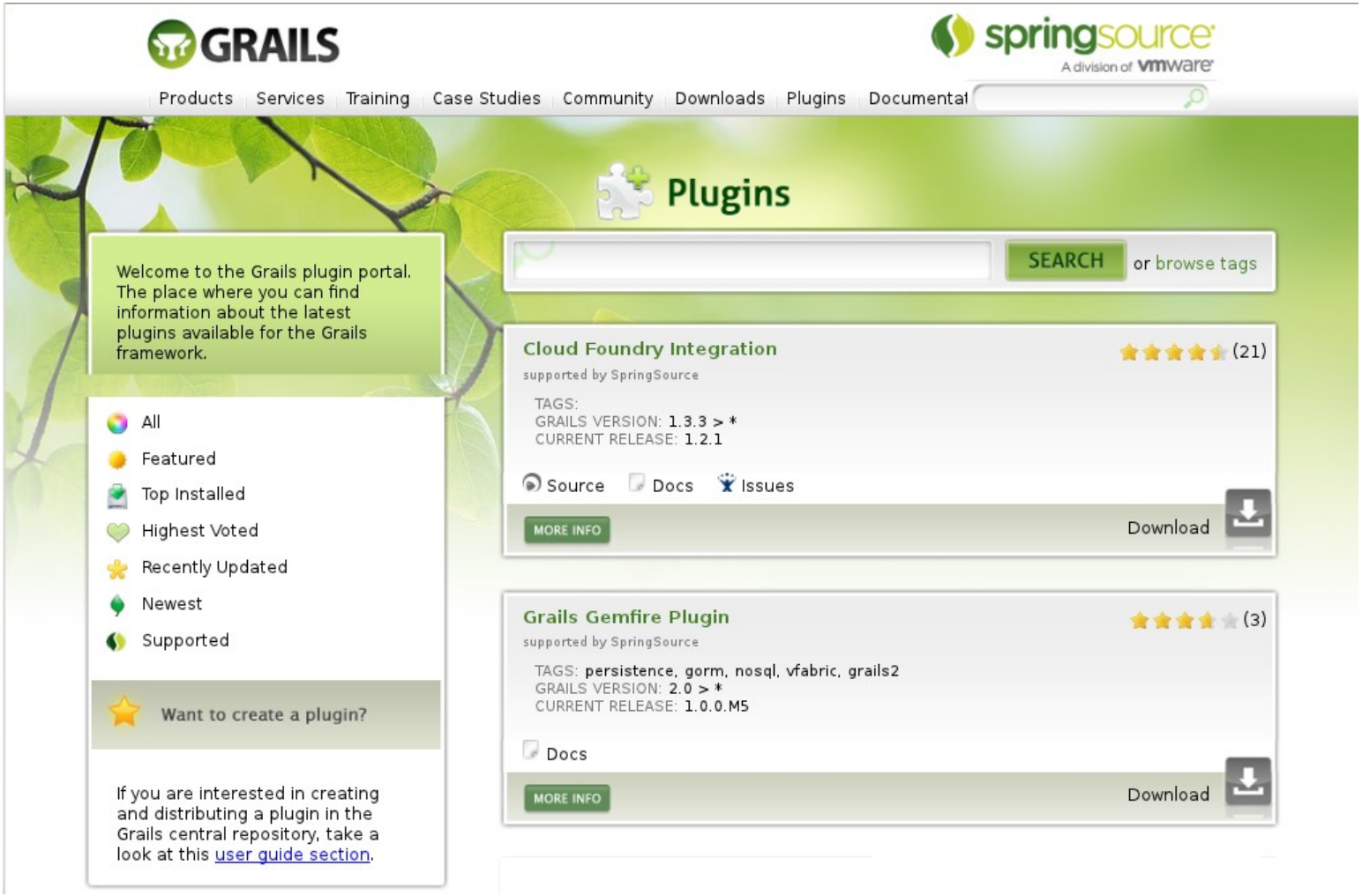
Database Console

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



Plugin Portal Usage Tracking

- Opt-in usage tracking of plugins



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
- GPars 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

GPars bundled

- GPars 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

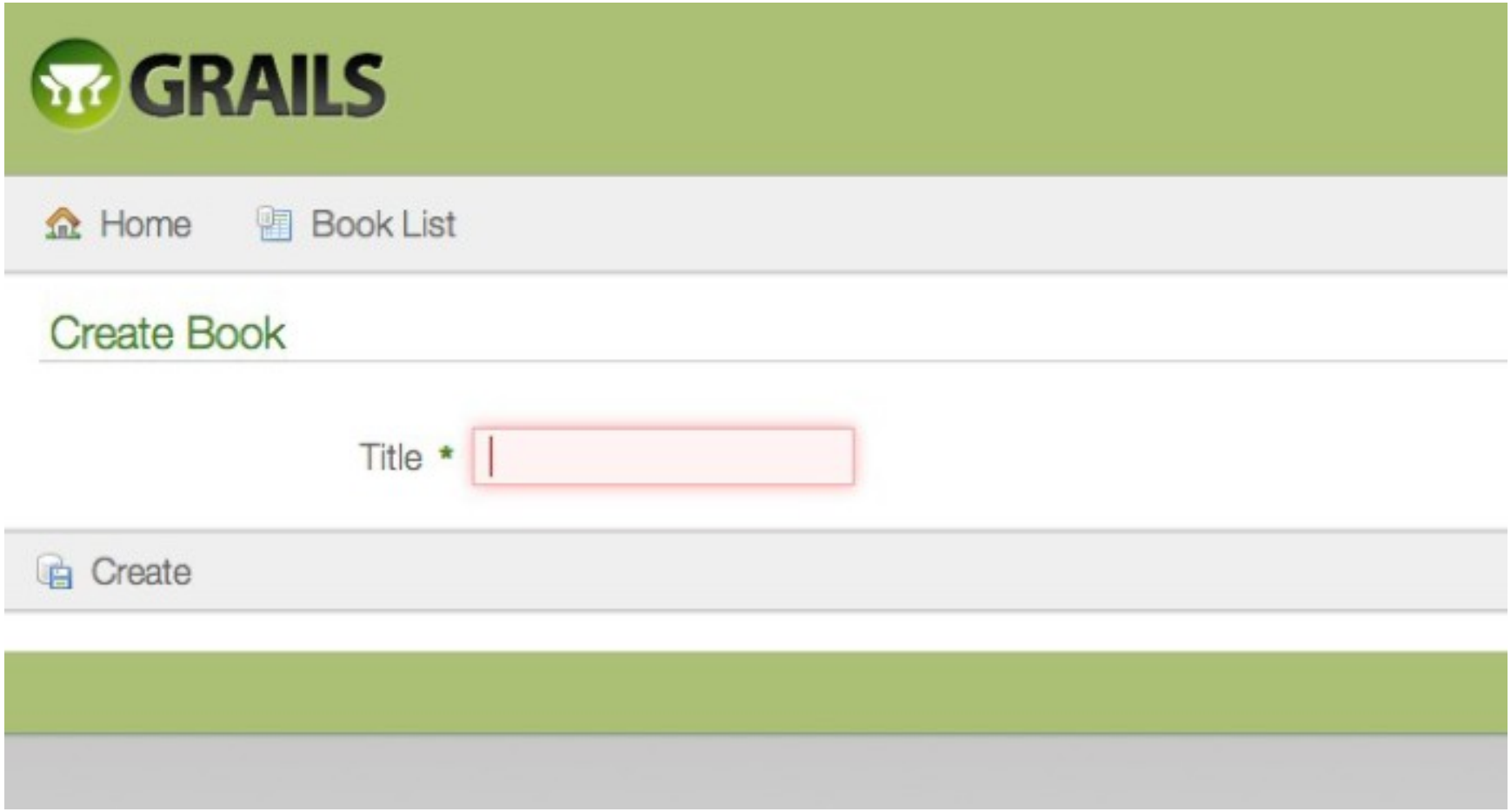
Binary Plugins

- 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



New APIs

- Page Rendering

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

- Link Generation

```
LinkGenerator grailsLinkGenerator
String generateLink() {
   grailsLinkGenerator.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

GORM API

GORM API

GORM API

- 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)
```

SQL Database Migration

Hibernate 'update'

+

Production data

=

?

SQL Database Migration

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/

SQL 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

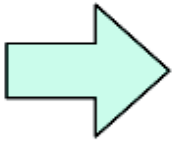
SQL Reverse Engineering

```
$ 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()
- findOrSaveWhere()

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

BETA

CLOUD FOUNDRY™

DEPLOY & SCALE YOUR APPLICATIONS IN SECONDS

For the Future

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

Demo

Thank You

Questions?