

# Cours IUT – CSID 2012

An Introduction to Spring Roo

[arnaud.nauwynck@gmail.com](mailto:arnaud.nauwynck@gmail.com)

# Outline

- What is Spring Roo ?
- Self-trained, documentations
- 5 Minutes demo
- Explained design principles
- Eclipse and Roo

# In few Words

- Tool to Create a complete Web application
  - In 2 Minutes
  - Including Database Persistence
  - With standards JAVA libraries only
  - No runtime dependency
- Make Java FUN, FAST, EASY

# <http://www.infoq.com/presentations/Introducing-Spring-Roo>

## What Is Roo?



- Roo is an extensible, text-based RAD tool for Java
- Roo is development-time only (no Roo runtime)

```
balex@lucky:~/helloworld$ roo
```



```
1.0.0.RC2 [rev 321]
```

```
Welcome to Spring Roo. For assistance press TAB or type "hint" then hit ENTER.
```

```
roo> █
```



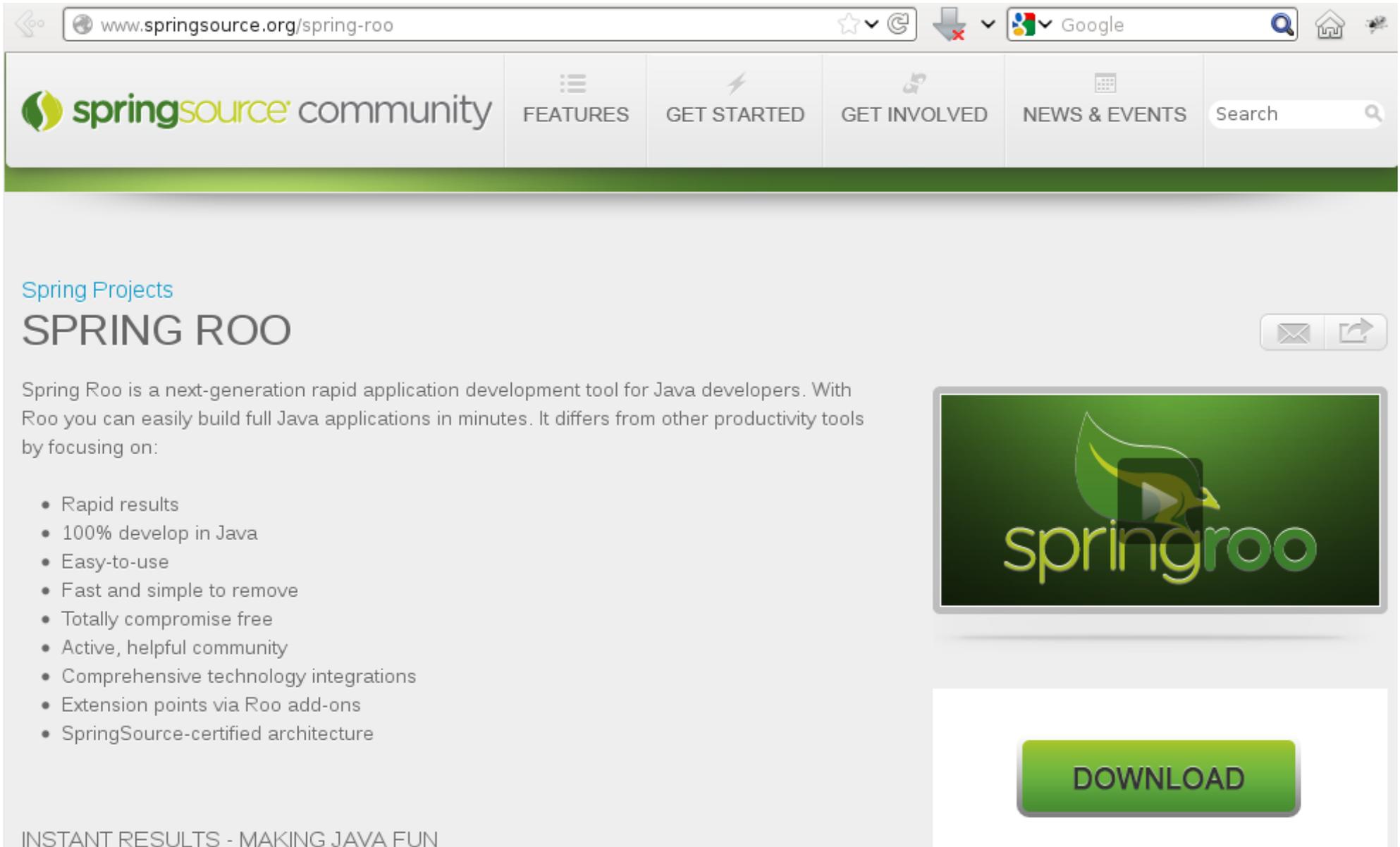
# Why I didn't Heard About It Before ?

- Strange name ...
  - ROO = Real Object Oriented
  - Now “Roo” ... a name by itself
- Recent : 1.0.0.RELEASE in December 2009
- Concurrenced by Grails, Rails, Php
- Roo is “only“ a tool above java standards
  - Many many Java standards (spring, jpa, web, ...)
  - Simple but underlying frameworks are “not”



# Roo Home:

## <http://www.springframework.org/spring-roo>



The image shows a screenshot of a web browser displaying the Spring Roo homepage. The browser's address bar shows the URL [www.springframework.org/spring-roo](http://www.springframework.org/spring-roo). The page features a navigation menu with links for 'FEATURES', 'GET STARTED', 'GET INVOLVED', and 'NEWS & EVENTS'. A search bar is located on the right side of the navigation menu. The main content area includes a 'Spring Projects' section with the heading 'SPRING ROO'. Below this heading, there is a brief description of Spring Roo as a rapid application development tool for Java developers. A list of features is provided, including 'Rapid results', '100% develop in Java', 'Easy-to-use', 'Fast and simple to remove', 'Totally compromise free', 'Active, helpful community', 'Comprehensive technology integrations', 'Extension points via Roo add-ons', and 'SpringSource-certified architecture'. To the right of the text, there is a video player showing a 'springroo' logo. At the bottom right, there is a prominent green 'DOWNLOAD' button. The footer of the page contains the text 'INSTANT RESULTS - MAKING JAVA FUN'.

springsource community

FEATURES GET STARTED GET INVOLVED NEWS & EVENTS

Search

Spring Projects

## SPRING ROO

Spring Roo is a next-generation rapid application development tool for Java developers. With Roo you can easily build full Java applications in minutes. It differs from other productivity tools by focusing on:

- Rapid results
- 100% develop in Java
- Easy-to-use
- Fast and simple to remove
- Totally compromise free
- Active, helpful community
- Comprehensive technology integrations
- Extension points via Roo add-ons
- SpringSource-certified architecture

springroo

DOWNLOAD

INSTANT RESULTS - MAKING JAVA FUN

# Wikipedia - Spring Roo

Create a



WIKIPEDIA  
The Free Encyclopedia

- Main page
- Contents
- Featured content
- Current events
- Random article
- Donate to Wikipedia

## Interaction

- Help
- About Wikipedia
- Community portal
- Recent changes
- Contact Wikipedia

## Toolbox

## Print/export

## Languages

Русский

Article [Talk](#)

Read [Edit](#) [View history](#)

Search

## Spring Roo

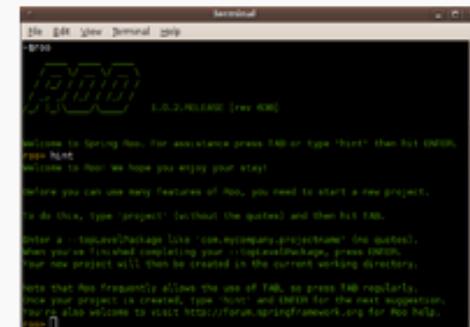
From Wikipedia, the free encyclopedia

**Spring Roo** is an [open source](#) software tool that uses [convention-over-configuration](#) principles to provide [rapid application development](#) of [Java-based enterprise software](#).<sup>[1]</sup> The resulting applications use common Java technologies such as [Spring Framework](#), [Java Persistence API](#), [Java Server Pages](#), [Apache Maven](#) and [AspectJ](#).<sup>[2]</sup> Spring Roo is a member of the [Spring](#) portfolio of projects.

### Contents [hide]

- Motivation and History
- Standards and Technology Compatibility
- Implementation
  - User Interface
  - Base Add-Ons
  - Roo Core Modules
- Differentiation
- See also
- References
- External links

## Spring Roo



Spring Roo 1.0.2 showing the "hint" command output

**Developer(s)**

VMware (SpringSource)

**Stable release**

1.2.1.RELEASE / February 2012; 5 months ago

# Spring Roo Wikipedia Explained

From Wikipedia, the free encyclopedia

In github  
by springsource

Spring Roo is an **open source** software tool that uses **convention-over-configuration** principles to provide **rapid application development** of **Java-based enterprise software**.<sup>[1]</sup> The resulting applications use common Java technologies such as **Spring Framework, Java Persistence API, Java Server Pages, Apache Maven** and **AspectJ**.<sup>[2]</sup> Spring Roo is a member of the **Spring** portfolio of projects.

shell window  
+ code generator  
.. NO runtime

Simplify code  
(spring annotations ...)

Like Ruby-On-Rails, Grails

Standards JAVA

# Roo = Java Standards

## Standards and Technology Compatibility

---

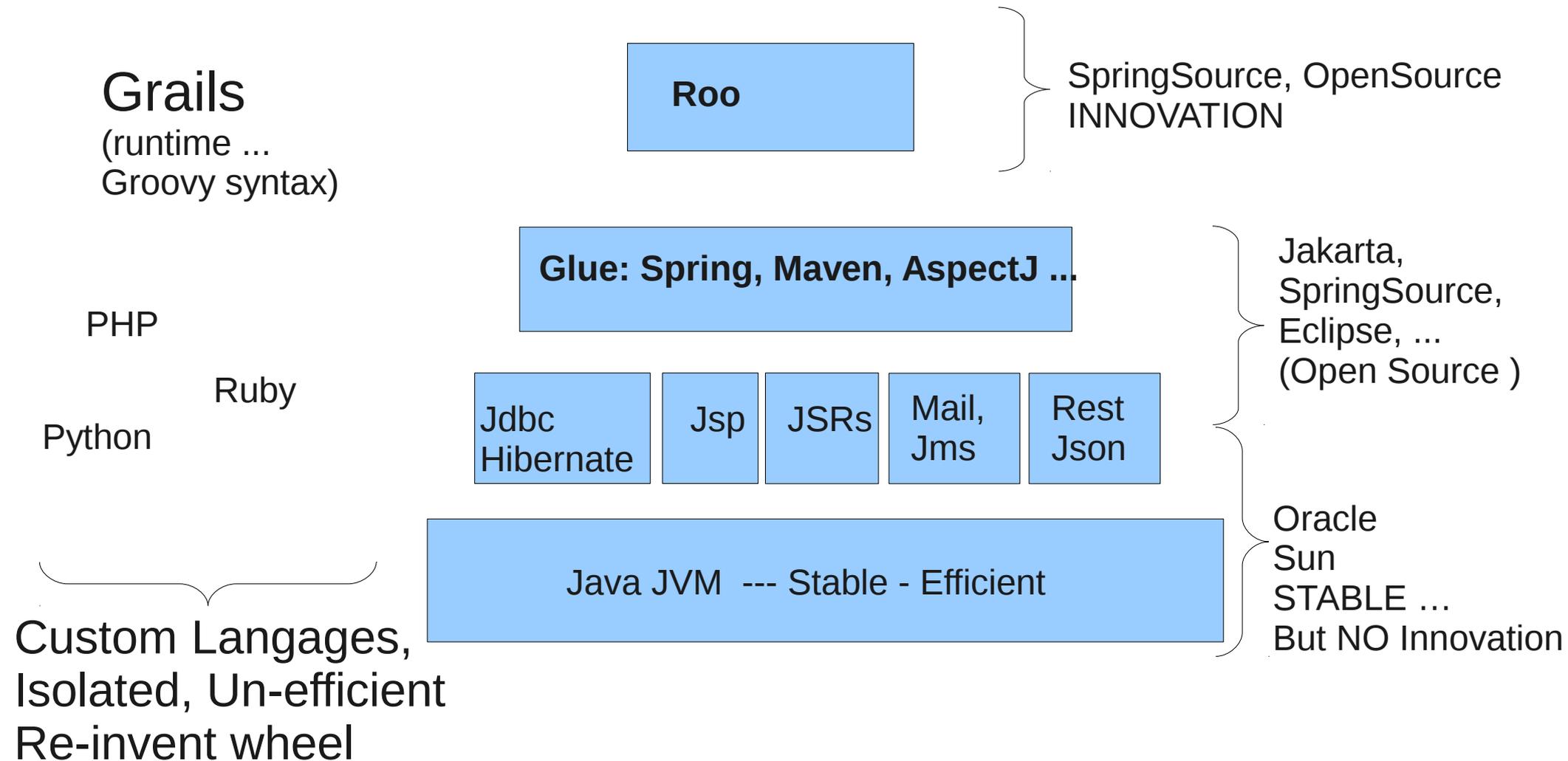
Roo's default installation facilitates the creation of applications that comply with the following standards

- [Apache ActiveMQ](#) (as an embedded [JMS](#) implementation)
- [Adobe Flex](#) (via a [SpringSource](#)-provided plugin)
- [Apache Maven](#) (version 2.2 or above)
- [Apache OpenJPA](#) (as a [JPA](#) implementation)
- [Apache Solr](#) (search server)
- [Apache Tiles](#) [↗](#) (default [MVC](#) views)
- [Apache Tomcat](#) (hosted execution support)
- [AspectJ](#) (used for [AOP](#) plus [mixins](#) to achieve [separation of concerns](#))
- [AspectJ Development Tools](#) [↗](#) (Eclipse plugin)
- [Cloud computing](#) (via [SpringSource Cloud Foundry](#) [↗](#)),
- [Dojo Toolkit](#) (via [Spring JavaScript](#))
- [Eclipse IDE](#) (concurrent execution and project metadata)
- [EclipseLink](#) (as a [JPA](#) implementation)
- [Google Web Toolkit](#) (since GWT 2.1, including GWT 2.1)
- [Hibernate](#) (as a [JPA](#) implementation)
- [Java Bean Validation \(JSR 303\)](#) [↗](#) (including [Hibernate Validator](#))
- [Java Database Connectivity](#) (for [JPA](#) usage)
- [Java Message Service](#) (both message producers and consumers)
- [Java Persistence API](#) (multiple implementations)
- [Java Transaction API](#) (via [Spring](#) transaction abstraction)
- [Java](#) (version 5 or above)
- [Java Server Pages](#) (default [MVC](#) views)
- [Jetty](#) (hosted execution support)
- [JSON](#) (methods in classes for serialization, deserialization and REST)
- [JUnit](#) (automated tests for user projects)
- [Log4j](#) (installation and configuration)
- [OSGi](#) (the Roo tool is built on [OSGi](#))
- [Representational State Transfer \(REST\)](#)
- [Selenium](#) (automated tests for user projects)
- [Spring Framework](#) (version 3 or above)
- [Spring Security](#) (version 3 or above)
- [Spring Web Flow](#) (installation and flow definition)
- [SpringSource Tool Suite](#) [↗](#) (STS has an embedded Roo shell and Roo projects)
- [Web application resource \(WAR file\)](#) (for deployment packaging)



Extensibility Plugins

# Comparison Building Blocks – Strength



# Rod Johnson: the Future Of Java Innovation

www.infoq.com/presentations/SpringOne-Keynote-Rod-Johnson

**Keynote: The Future of Java Innovation**  
Presented by **Rod Johnson** on Jun 10, 2009 Length 01:55:18  
Sections **Enterprise Architecture, Operations & Infrastructure, Development** Topics **SpringOne, Spring Roo, Application Servers, Spring, SpringSource, FEATURED Java, Web Servers, Web Frameworks, Enterprise Architecture, SpringOne Europe 2009, SpringSource dm Server**

Share 

How would you like to view the presentation?



**Summary**  
In the opening keynote at SpringOne Europe 2009, Rod Johnson wondered if Java innovation is going to be stifled by latest Oracle acquisition and expressed his belief that Java will continue to evolve outside of Sun as it has done for the last few years. As proof he mentioned: Grails, Roo, a tool for improved developer productivity, a free STS, tc Server and dm Server.

**Bio**  
Rod is one of the world's leading authorities on Java and J2EE development. He is a best-selling author, experienced consultant, and open source developer, as well as a popular conference

## Time to Step Up and Fight Back



- Don't know about you, but *I'm as mad as hell and I'm not going to take it any more*
- We know the JVM is right for enterprise problems
- Time to show Rails/Django/PHP etc. that the JVM can



HEY FELLA! NO MORE PUSHING SMALL GUYS AROUND... I'M AN 'ATLAS MAN' AND I STAND MY GROUND.  
I'M GOTTA BE HERE NOW!

# Install

## Download and Easy Install



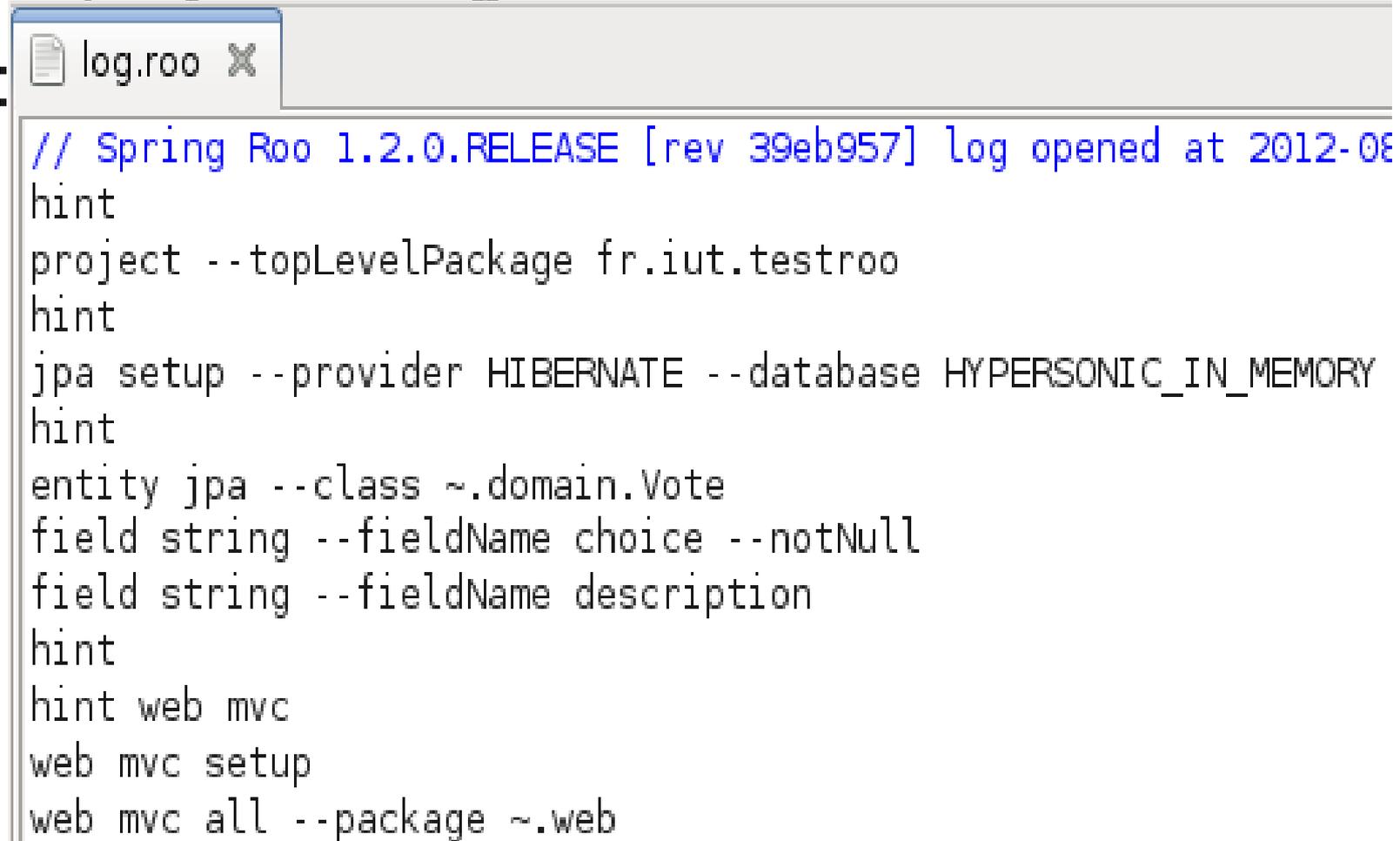
- Download Roo from [www.springsource.org/roo](http://www.springsource.org/roo)
  - Simply unzip and add to your path
- Linux/Apple users would use something like:
  - `sudo ln -s $HOME/spring-roo-1.0.0.RC2/bin/roo.sh /usr/bin/roo`
- Windows
  - Add `$HOME/spring-roo-1.0.0.RC2/bin` to system PATH
- A public Subversion repository is also available
  - See [readme.txt](#) in root of checkout for dev installation

# Getting Started

- Open shell window, Type “roo”
- Interactively :
  - Type “hint” for wizard
  - “tab” for autocompletion on mandatory args
  - “--” then “tab” for optional args
  - “help” : list all commands
  - “help <<command>>”
- Execute tutorial script “script <<file>>”

# Sample Script

- Cf in roo/examples/⟨⟨project⟩⟩.roo
- Cf in your project log.roo
- Sample:



```
// Spring Roo 1.2.0.RELEASE [rev 39eb957] log opened at 2012-08-08 10:10:10
hint
project --topLevelPackage fr.iut.testroo
hint
jpa setup --provider HIBERNATE --database HYPERSONIC_IN_MEMORY
hint
entity jpa --class ~.domain.Vote
field string --fieldName choice --notNull
field string --fieldName description
hint
hint web mvc
web mvc setup
web mvc all --package ~.web
```

# Compile, Test & Run It

- Compile:
  - Use maven from shell: `$ mvn install`
  - From roo: `roo> perform package`
- Test:
  - `$ mvn tomcat:run`
  - open <http://localhost:8080/test>
- Import in Eclipse:
  - Using m2eclipse ... import as maven project
  - (old: “perform eclipse” or “mvn eclipse:eclipse” )

# Running CRUD !

**ROO** **Spring**

**VOTE**

- Create new Vote
- List all Votes

Welcome to Testroo

Welcome to Testroo

Spring Roo provides interactive, lightweight and user customizable tooling that enables rapid delivery of high performance enterprise Java applications.

Home | Language: | Theme: [standard](#)

List all Votes Show, edit, delete

Choice	Description			
java	the good choice			
php	game over			
grails	fast develop but untyped			
python	silly tabs			
lisp	Lot of Insane Stupid Paranthesis			
dotnet	good choice, java clone by Micro\$soft			

Create new Vote

Choice :

Description :

**SAVE**

Show Vote

Choice : java

Description : the good choice

Update Vote

Choice :

Description :

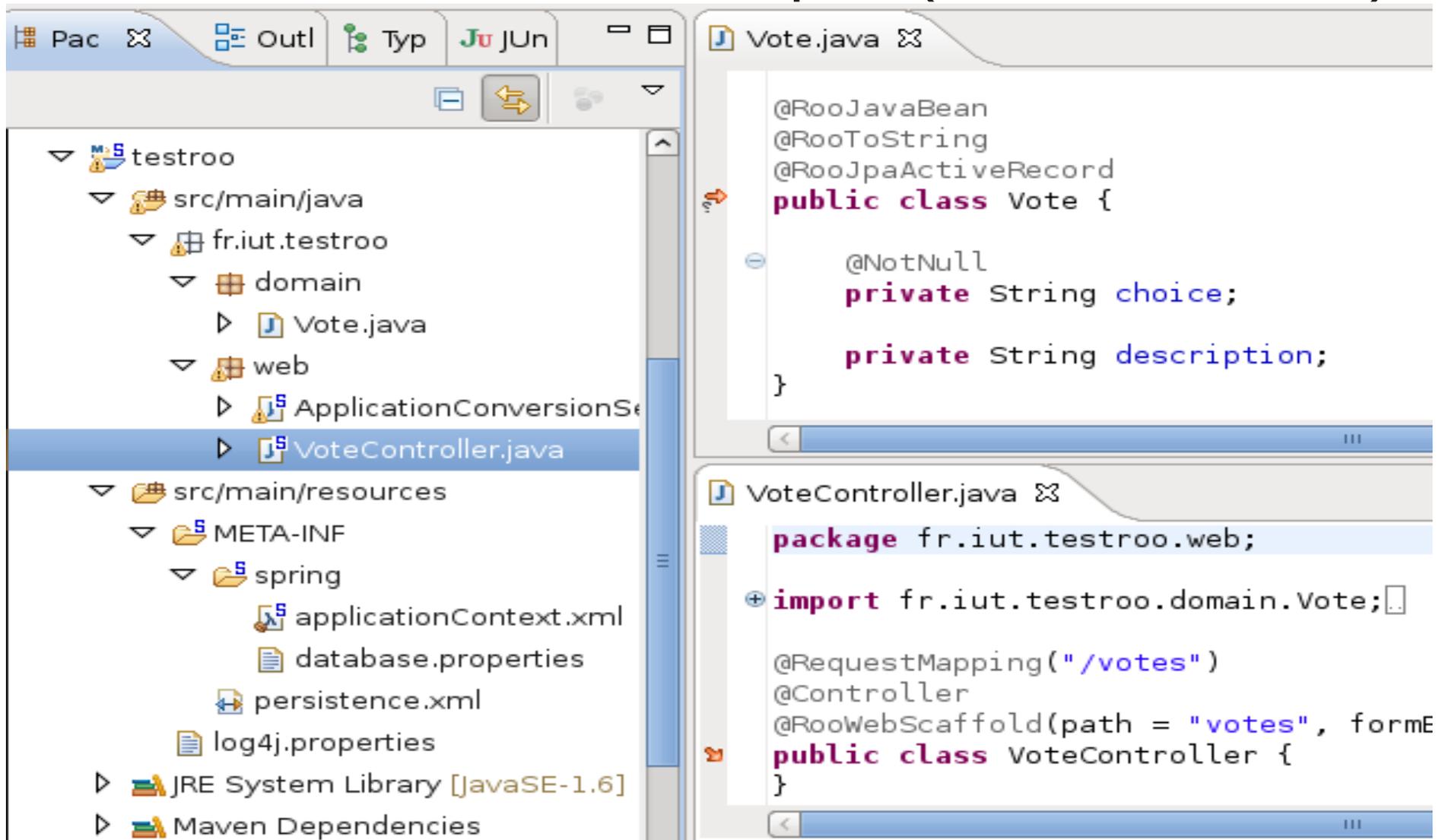
**SAVE**

**Annotations:**

- create**: points to 'Create new Vote' in the menu.
- list**: points to 'List all Votes' in the menu and the 'List' button in the table.
- Show, edit, delete**: points to the action icons in the table.
- Update**: points to the 'Update Vote' form.
- Show**: points to the 'Show Vote' form.
- Create**: points to the 'Create new Vote' form.
- list, new, edit, delete**: points to the action icons at the bottom of the 'Show Vote' form.

# Small Is Beautiful ... KISS

- See ALL User Code in Eclipse (not Generated )



The screenshot displays the Eclipse IDE interface. On the left, the Project Explorer shows the project structure for 'testroo'. The 'src/main/java' directory is expanded, showing the 'fr.iut.testroo' package. Underneath, there are 'domain' and 'web' sub-packages. The 'Vote.java' file is visible in the 'domain' package, and 'VoteController.java' is visible in the 'web' package. The 'src/main/resources' directory is also expanded, showing configuration files like 'applicationContext.xml', 'database.properties', 'persistence.xml', and 'log4j.properties'. The 'JRE System Library [JavaSE-1.6]' and 'Maven Dependencies' are also visible at the bottom of the Project Explorer.

The main editor area shows the code for 'Vote.java' and 'VoteController.java'. The 'Vote.java' code is as follows:

```
@RooJavaBean
@RooToString
@RooJpaActiveRecord
public class Vote {

    @NotNull
    private String choice;

    private String description;
}
```

The 'VoteController.java' code is as follows:

```
package fr.iut.testroo.web;

import fr.iut.testroo.domain.Vote;

@RequestMapping("/votes")
@Controller
@RooWebScaffold(path = "votes", formE
public class VoteController {
}
```

# Where Is The Magic ?

The image illustrates the configuration of a filter in an IDE to hide generated Spring Roo ITDs. On the left, the 'Java Element Filters' dialog is open, showing the 'Package Pr' tab. The checkbox for 'Hide generated Spring Roo ITDs' is checked. Below this, the filter description reads: 'Hides ITDs that are generated by Spring Roo'. On the right, the package explorer shows a tree structure with two main packages: 'domain' and 'web'. Under 'domain', several files are listed, including 'Vote\_Roo\_Configurable.aj', 'Vote\_Roo\_Jpa\_ActiveRecord.aj', and 'Vote\_Roo\_Jpa\_Entity.aj'. Under 'web', files like 'ApplicationConversionServiceFactor' and 'VoteController\_Roo\_Controller.aj' are visible. Red arrows originate from the 'Hide generated Spring Roo ITDs' checkbox in the dialog and point to the corresponding files in the package explorer, demonstrating the effect of the filter.

Top Level Elements >

Configure Working Sets...

Hide generated Spring Roo ITDs

Filters...

Package Pr

Name filter patterns (matching names will be hidden):

The patterns are separated by comma, where  
\* = any string, ? = any character, , = ,

Select the elements to exclude from the view:

Hide generated Spring Roo ITDs

Filter description:  
Hides ITDs that are generated by Spring Roo

Select All Deselect All

Cancel OK

domain

- Vote\_Roo\_Configurable.aj
- Vote\_Roo\_JavaBean.aj
- Vote\_Roo\_Jpa\_ActiveRecord.aj
- Vote\_Roo\_Jpa\_Entity.aj
- Vote\_Roo\_ToString.aj
- Vote.java

web

- ApplicationConversionServiceFactor
- ApplicationConversionServiceFactor
- VoteController\_Roo\_Controller.aj
- VoteController.java

# User Code + ITD ==AspectJ==> Bytecode

The screenshot displays an IDE environment with the following components:

- Project Tree (Left):** Shows a project structure with packages like `fr.iut.testroo` and `domain`. The file `Vote_Roo_jpa_ActiveRecord.aj` is selected under the `domain` package.
- Top Editor (Vote.java):** Shows the source code of the `Vote` class. It includes an `@Transactional` annotation and a `persist()` method that checks for a null `entityManager` before persisting the instance.
- Bottom Editor (VoteController.java):** Shows the source code of the `VoteController` class. It includes an `@RequestMapping` annotation for a POST method and a `create()` method that handles form validation and persistence.

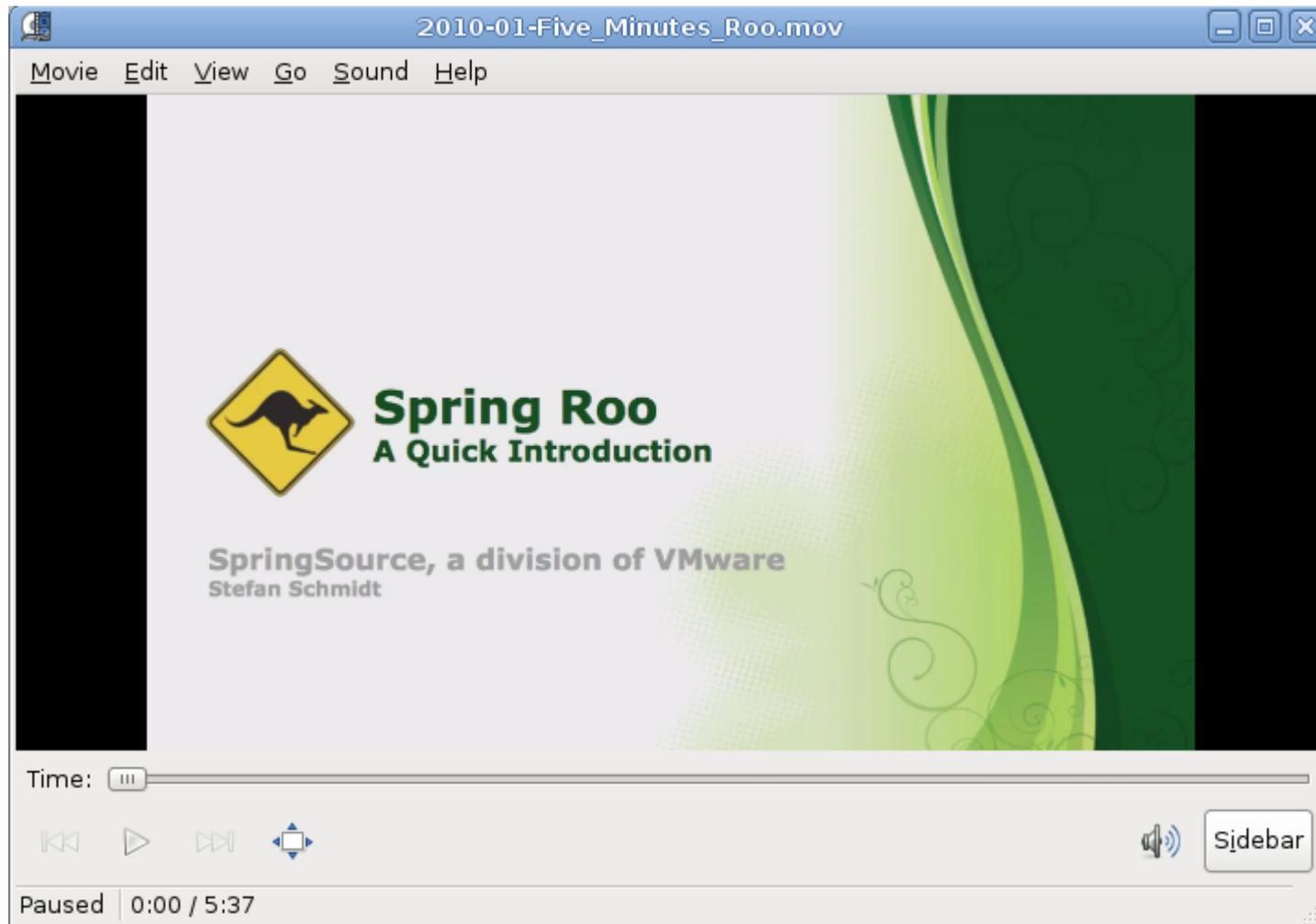
```
Vote.java
@Transactional
public void Vote.persist() {
    if (this.entityManager == null) this.entityManager = entityManager;
    this.entityManager.persist(this);
}

VoteController.java
@RequestMapping(method = RequestMethod.POST, produces = "text/html")
public String VoteController.create(@Valid Vote vote, BindingResult bindingResult) {
    if (bindingResult.hasErrors()) {
        populateEditForm(uiModel, vote);
        return "votes/create";
    }
    uiModel.asMap().clear();
    vote.persist();
    return "redirect:/votes/" + encodeUrlPathSegment(vote.getId().toString());
}
```

# Video – Demo Tutorial

Cf from Home Page :

[http://s3.springsource.com/MRKT/roo/2010-01-Five\\_Minutes\\_Roo.mov](http://s3.springsource.com/MRKT/roo/2010-01-Five_Minutes_Roo.mov)



# Video from Ben Alex

<http://www.infoq.com/presentations/Introducing-Spring-Roo>

Presentation My Bookmarks

## Introducing Spring Roo - Extreme Productivity in 10 Minutes

Presented by **Ben Alex** on Jan 07, 2010 Length 01:29:01 Download: **MP3**

Sections **Development** Topics **Spring Roo**, **Spring**, **FEATURED Java**, **SpringSource**, **VMWare**, **SpringSource Tool Suite**, **SpringOne 2009**

Share

Recorded at: **springOne**  
[Bookmark this!](#)

How would you like to view the presentation? horizontal vertical Maximize



**Summary**  
Ben Alex, the Roo's founder, explains what Roo is, how to get started, and creates a project from scratch demonstrating some of Roo's features: code assist, visual error reporting, JPA-based persistence, bean validation support, automated JUnit integration tests, entity finders, scripting support, messaging support, round-trip support, Eclipse and Spring Tool Suite integration and others.

**Bio**  
Dr Ben Alex is a Principal Software Engineer with SpringSource, and has been working

## Agenda



---

- Introducing Roo
- Capability Areas
- Using Roo
- Roadmap and Resources



# Tutorial

 [static.springsource.org/spring-roo/reference/html/beginning.html](http://static.springsource.org/spring-roo/reference/html/beginning.html)

## Spring Framework

### Chapter 2. Beginning With Roo: The Tutorial

In this chapter we'll build an app step-by-step together in a relatively fast side-steps to other sections of this manual.

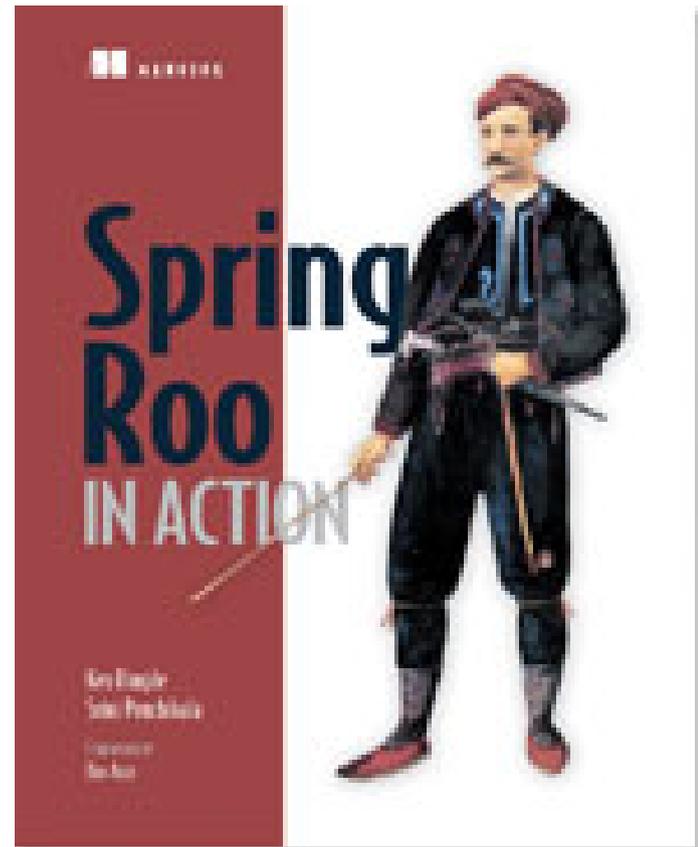
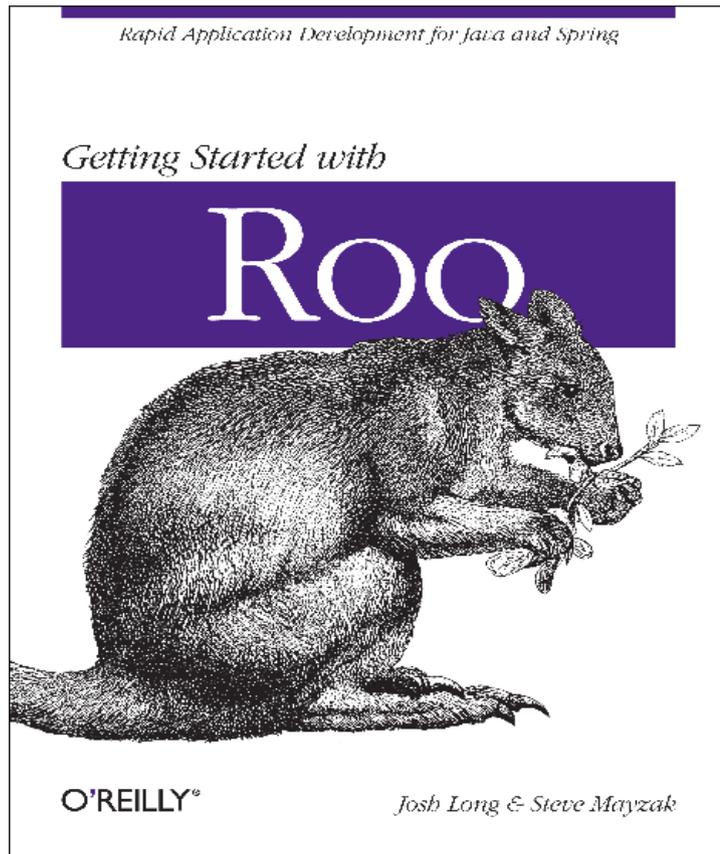
#### 2.1. What You'll Learn

In this tutorial you will learn to create a complete Web application from scratch with Roo. In particular you will learn how to use the Roo shell for:

- project creation
- creation and development of domain objects (JPA entities)
- adding fields of different types to the domain objects
- creating relationships between domain objects
- automatic creation of integration tests
- creating workspace artifacts to import the project into your IDE
- automatic scaffolding of a Web tier
- running the application in a Web container
- controlling and securing access to different views in the application
- customizing the look and feel of the Web UI for our business domain
- creating and running Selenium tests
- deployment and backup of your application

# Books

[http://spring-roo-repository.springsource.org/Getting\\_Started\\_with\\_Roo.pdf](http://spring-roo-repository.springsource.org/Getting_Started_with_Roo.pdf)



# Source Code

GitHub, Inc. [US] <https://github.com/springsource/spring-roo>

github

[Signup and Pricing](#) [Explore GitHub](#) [Features](#) [Blog](#) [Sign in](#)

PUBLIC



SpringSource / **spring-roo**

★ Star

202

Fork

37

Code

Network

Pull Requests 1

Graphs

Spring Roo is a next-generation rapid application development tool for Java developers. It focuses on higher productivity, stock-standard Java APIs, high usability, avoiding engineering trade-offs and facilitating easy Roo removal. — [Read more](#)

<http://www.springsource.org/roo>



ZIP

HTTP

Git Read-Only

<https://github.com/SpringSource/spring-roo.git>



Read-Only access



branch: **master**

Files

Commits

Branches 3

Tags 23

Downloads

Latest commit to the **master** branch

ROO-3197: Data On Demand could be not private in the Roo Integration ...



**alankstewart** authored 2 months ago

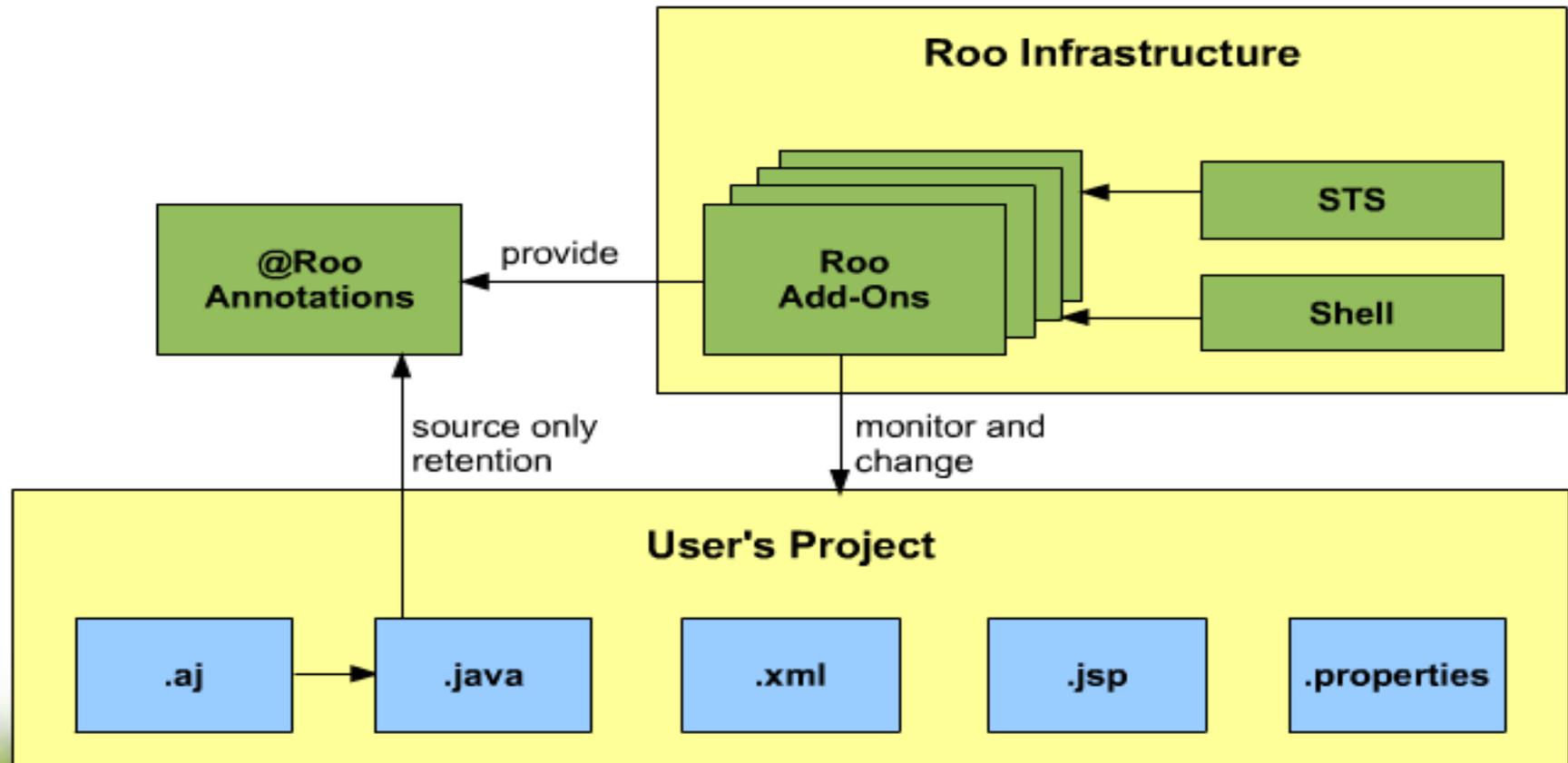


commit **ba84b7e530**

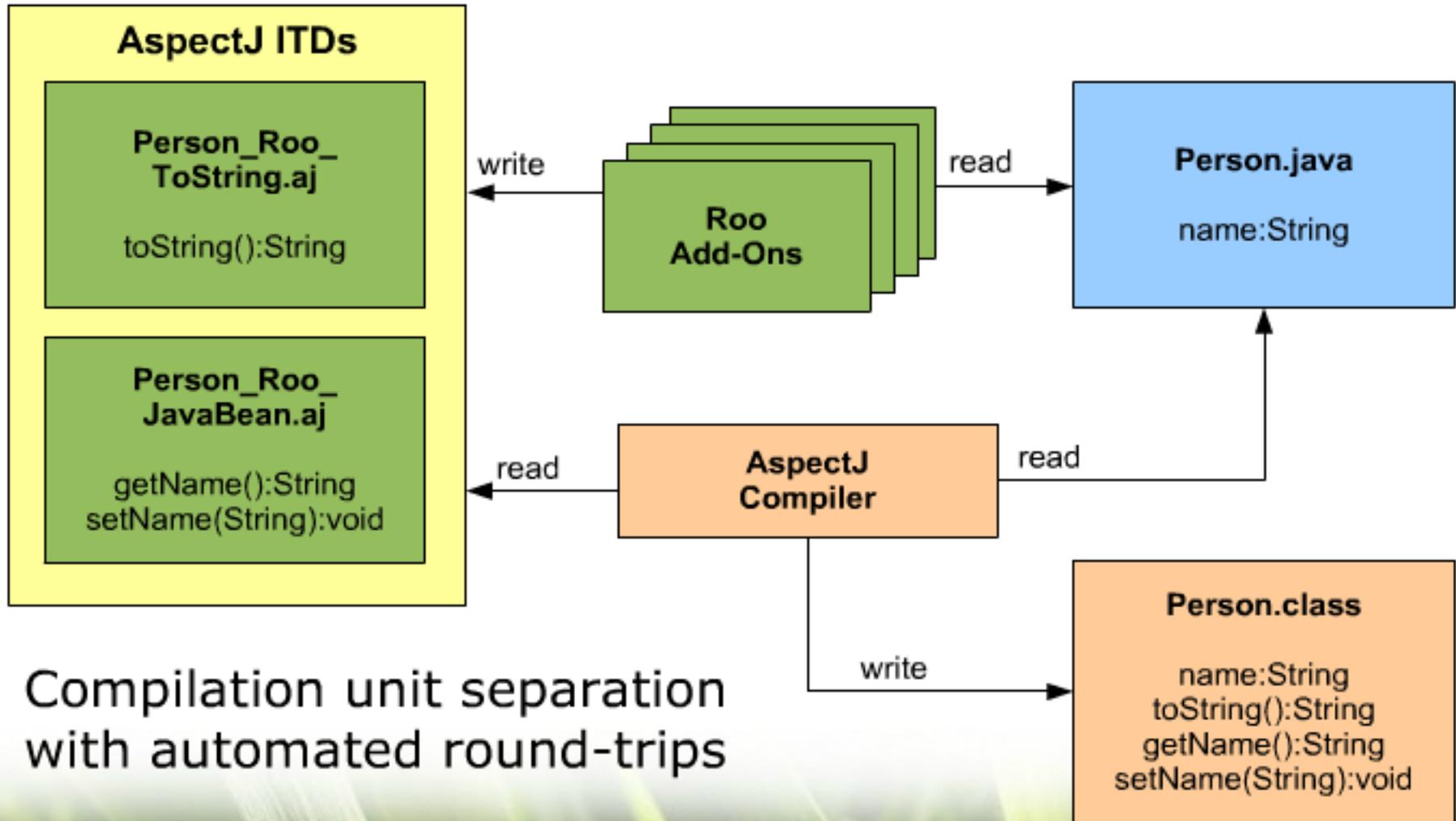
**spring-roo** /

name	age	message	history
<a href="#">addon-backup</a>	3 months ago	ROO-3162: Release Spring Roo 1.2.2.RELEASE - change to new 1.2.3.BUIL... [alankstewart]	
<a href="#">addon-cloud-foundry</a>	3 months ago	ROO-3162: Release Spring Roo 1.2.2.RELEASE - change to new 1.2.3.BUIL... [alankstewart]	

# Implementation Overview



# Active Generation



Compilation unit separation with automated round-trips

# Eclipse AspectJ – Push In...

The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer displays a project named 'testroo' with a package structure: 'src/main/java' containing 'fr.iut.testroo', which has sub-packages 'domain' and 'web'. The 'src/main/resources' folder is also visible. The main editor area shows a context menu for the 'fr.iut.testroo' package. The 'Refactor' option is selected, and its sub-menu is open, showing 'Push In...' as the active option. Other options in the sub-menu include 'Rename...', 'Move...', and 'Infer Generic Type Arguments...'. The 'Push In...' option is highlighted in blue.

The 'Push In Intertype Declaration' dialog box is open in the foreground. It contains the following text and table:

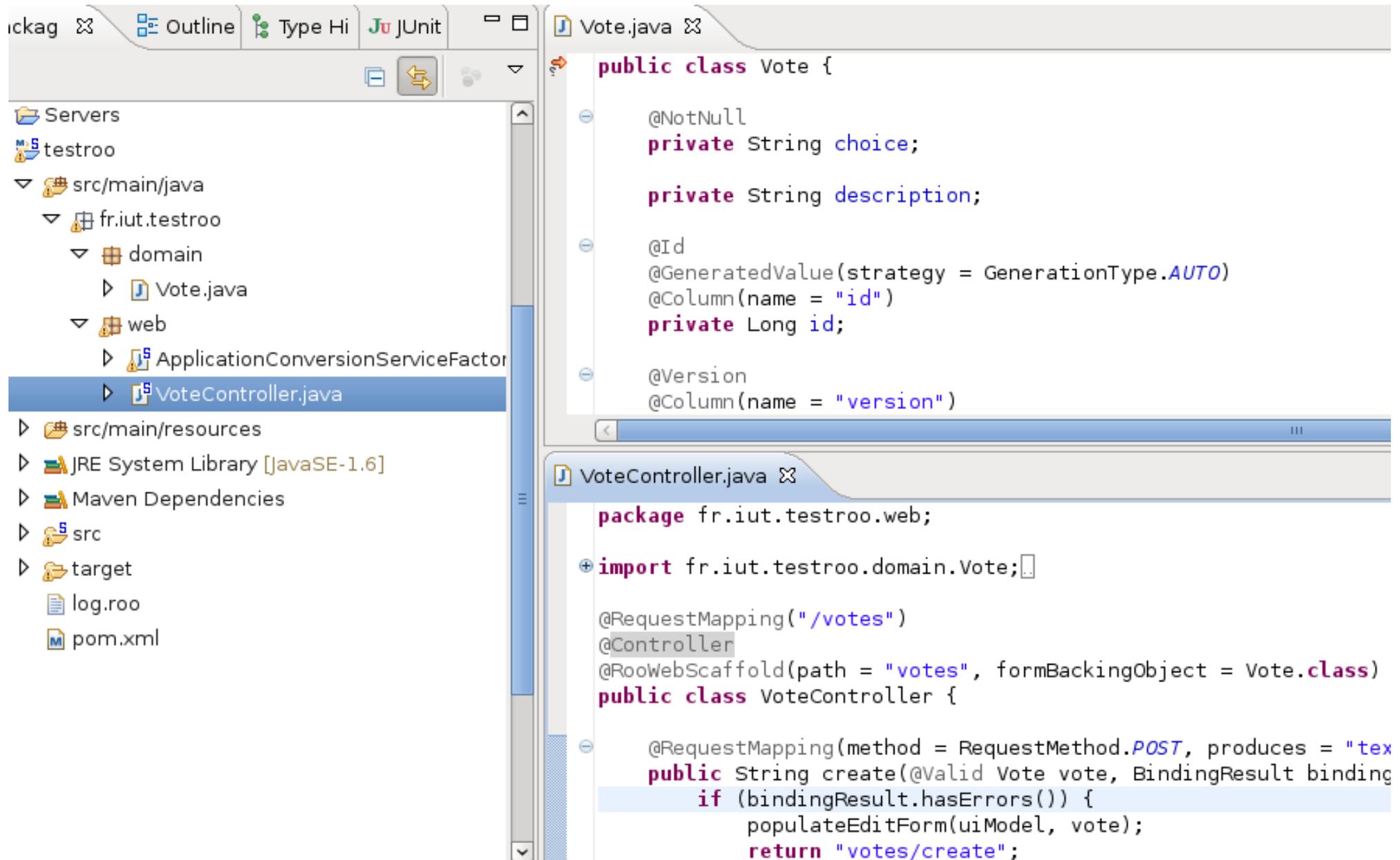
The following intertype declarations will be pushed into their target types:

Declaring aspect	Intertype Name	Target type
Vote_Roo_JavaBean	Vote.getChoice	Vote
Vote_Roo_JavaBean	Vote.setChoice	Vote
Vote_Roo_JavaBean	Vote.getDescription	Vote
Vote_Roo_JavaBean	Vote.setDescription	Vote
Vote_Roo_Jpa_Entity	Vote.version	Vote
Vote_Roo_Jpa_Entity	Vote.getId	Vote

To change the set of intertype declarations to be pushed in, click cancel and reselect only the desired AspectJ e

Buttons: Preview > OK Cancel

# Happy Runtime End – Roo is gone



The screenshot shows an IDE interface with a project explorer on the left and two code editors on the right. The project explorer shows a project named 'testroo' with a package structure: 'src/main/java/fr.iut.testroo/domain/Vote.java' and 'src/main/web/VoteController.java'. The 'VoteController.java' file is selected and highlighted in blue.

The 'Vote.java' editor shows the following code:

```
public class Vote {  
  
    @NotNull  
    private String choice;  
  
    private String description;  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    @Column(name = "id")  
    private Long id;  
  
    @Version  
    @Column(name = "version")  
}
```

The 'VoteController.java' editor shows the following code:

```
package fr.iut.testroo.web;  
  
import fr.iut.testroo.domain.Vote;  
  
@RequestMapping("/votes")  
@Controller  
@RooWebScaffold(path = "votes", formBackingObject = Vote.class)  
public class VoteController {  
  
    @RequestMapping(method = RequestMethod.POST, produces = "text/html")  
    public String create(@Valid Vote vote, BindingResult bindingResult) {  
        if (bindingResult.hasErrors()) {  
            populateEditForm(uiModel, vote);  
            return "votes/create";  
        }  
    }  
}
```

# Conclusion

Spring Roo is Beautifull !!

Questions ?

[arnaud.nauwynck@gmail.com](mailto:arnaud.nauwynck@gmail.com)