

Rank und Schlank

Web-Anwendungen in Java ohne Boilerplate-Code

Mirko Zeibig

IST GmbH Dresden

Was ist das?



Boilerplate?

Boilerplate



JUG saxony – Rank und Schlank

Boilerplate



Übersicht

- Roo
- Technologie
- Installation
- Domainmodell
- CRUD
- Security

Roo

- erste Version 2009 auf Spring One
- 1.1 in Entwicklung
- Ben Alex:

Regarding the name "Roo", we brainstormed about 20 names before SpringOne Europe. We then eliminated those subject to existing trademarks and similar considerations. You'd be surprised how many great names you just can't use due such factors! Anyhow, those names that remained were put to the community to vote and "Roo" was the hands-down winner (as mentioned in my entry above). The name Roo originally came from "Real Object Oriented", but we dropped that given the focus shifted to sustainable productivity when I resumed work on the project. Because I still called the code "Roo" when talking to colleagues, it became the de facto project name. When we saw the names that survived trademark review, we thought it would be nice to at least throw Roo in there – and that's the story of the name!



Roo

- Spring Roo is a lightweight developer tool that makes it fast and easy to deliver instant results.
- Spring Roo is an easy-to-use productivity tool for rapidly building enterprise applications in the Java programming language.
- leichtgewichtig
- schnelle Ergebnisse
- Entwicklungstool
- Java

Technologie

- Roo selbst ist nur ein Werkzeug
- Webanwendung mittels @MVC
- Persistenz durch JPA
- Magie durch AspectJ
- bei Bedarf weitere Teile des Springstack
 - Spring Security
 - Webflow
 - ...

Installation

- JDK 1.5 (1.6)
- Maven 2.0.9 (3.0.3)
- Auspacken
- Environment setzen
 - \$JAVA_HOME
 - \$ROO_HOME
 - \$PATH
 - `ln -s $ROO_HOME/bin/roo.sh /usr/bin/roo`
- Shell starten

Roo Shell

- Eingabe von Befehlen
 - werden in Datei gespeichert
- Überwacht externe Aktivitäten
- Stößt Generierungen an
- ‚hints‘ helfen bei den nötigen Aufgaben
- Tabulatortaste für Vervollständigung

Start eines Projekts

- **Projekt erzeugen**

```
project --topLevelPackage com.istair
```

- **Persistence einstellen**

```
jpa setup --provider HIBERNATE --database HYPERSONIC_IN_MEMORY
```

IDE?

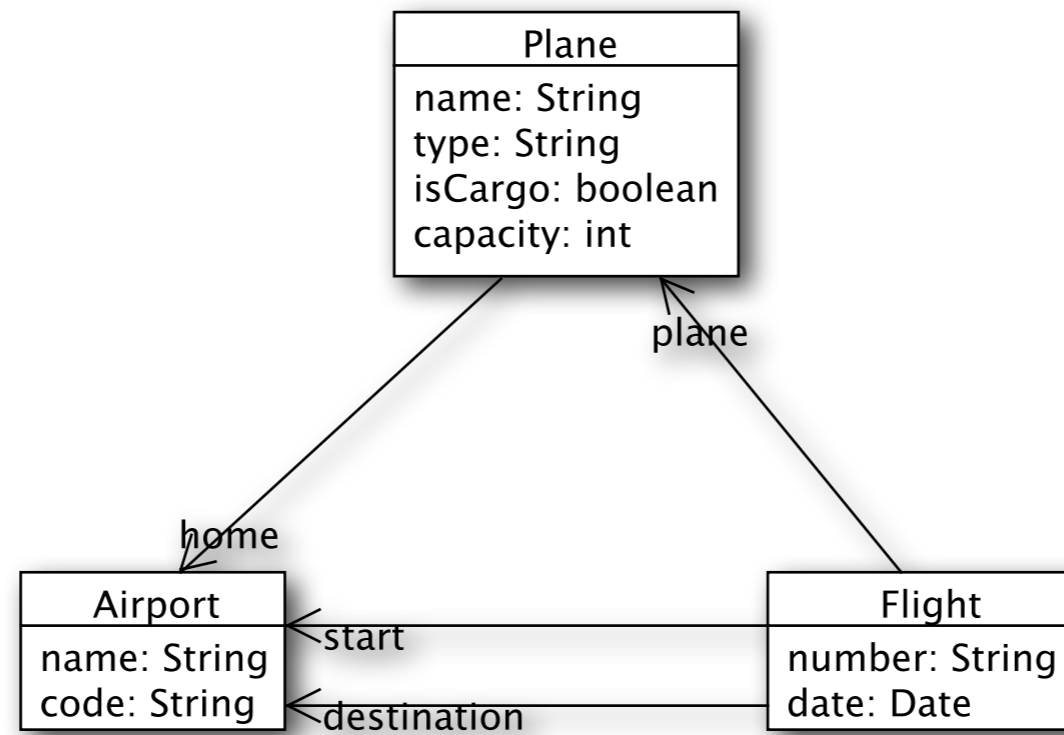
- beliebiger Editor
- besser eine IDE
- Eclipse mit AJDT
- STS mit Spring- und Roo-Plugins

- aus der Roo Shell
`perform eclipse`
- direkt mit Maven
`mvn eclipse:eclipse`

DEMO

DDD

- Grundlage der Anwendung ist ein Domainmodell.
- Darauf basiert das Scaffolding.
- automatische Persistenz
- generierte Finder



Entities erzeugen

- zur Erinnerung:
`project --topLevelPackage com.istair`
- `topLevelPackage` wird zu `~`
- Erzeugung von Entities mittels `,entity‘`
- Hilfe mit `,hint‘` und TAB

- `entity jpa --class ~.domain.Plane`
- `entity jpa --class ~.domain.Airport`
- `entity jpa --class ~.domain.Flight`

Attribute hinzufügen

- Befehl `field`
- z.B. `field string`, `field boolean`, ...
- JSR303

- `field string --fieldName name --notNull`
- `field string --fieldName typ --notNull`
- `field string --fieldName name --notNull --sizeMin 3`
- `field string --fieldName code --notNull --sizeMin 3 --sizeMax 3`
- `field date --fieldName birthDate --type java.util.Date --notNull`

Beziehungen

- Referenzen werden mit dem Befehl `field reference` angelegt.
- Kardinalitäten mittels Parameter `--cardinality`

```
field reference --class ~.domain.Plane --fieldName home --type ~.domain.Airport
```


Beziehungen

- Referenzen als Menge mittels ‚field set‘
- auch hier: `--cardinality`

```
field set --fieldName foos --element Foo
```

DEMO

Was wurde generiert?

```
package com.istair.domain;

import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import org.springframework.roo.addon.javabean.RooJavaBean;
import org.springframework.roo.addon.jpa.activerecord.RooJpaActiveRecord;
import org.springframework.roo.addon.tostring.RooToString;

@RooJavaBean
@RooToString
@RooJpaActiveRecord
public class Airport {

    @NotNull
    @Size(min = 3)
    private String name;

    @NotNull
    @Size(min = 3, max = 3)
    private String code;
}
```

Inter-type Declarations

- AspectJ
- fügt existierenden Klassen neues hinzu
- z.B. Methoden
- Unterstützung durch AJDT

@RooJavaBean

- Erzeugt Getter und Setter

```
package com.istair.domain;
import com.istair.domain.Airport;
privileged aspect Airport_Roo_JavaBean {

    public String Airport.getName() {
        return this.name;
    }

    public void Airport.setName(String name) {
        this.name = name;
    }

    public String Airport.getCode() {
        return this.code;
    }

    public void Airport.setCode(String code) {
        this.code = code;
    }
}
```

@RooToString

- Erzeugt eine `toString()` Methode

```
package com.istair.domain;

import java.lang.String;

privileged aspect Airport_Roo_ToString {

    public String Airport.toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("Id: ").append(getId()).append(", ");
        sb.append("Version: ").append(getVersion()).append(", ");
        sb.append("Name: ").append(getName()).append(", ");
        sb.append("Code: ").append(getCode());
        return sb.toString();
    }
}
```

@RooJpaActiveRecord

- Definiert alles zur Persistenz
- PK und Version Felder
- Methoden
 - `persist()`
 - `flush()`
 - `remove()`
 - `merge()`
- EntityManager
- `countAirports()`
- Finder
 - `findAllAirports()`
 - `findAirport(Long id)`
 - `findAirportEntries(int firstResult, int maxResults)`

Finder

- nicht so dynamisch wie Grails
- müssen zur Entwicklungszeit definiert werden
- sind ‚unsichtbar‘ in den Domainklassen
 - ITD mit AspectJ
- Verknüpfung von beliebig vielen Feldern
- ‚finder list‘ zeigt die Kombinationen an
 - `finder list --depth`
 - `finder list --filter`
- ‚finder add‘ erzeugt sie

finder list

```
live-roo-test2 — Spring Roo 1.2.2.RELEASE [rev 7d75659] — java — 80x24
zeiban2:live-roo-test2 mzeibig$ roo.sh

  -----
 /  _  \  _  \  _  \
//  /  /  /  /  /  /
/_ , _/ /  /  /  /
// | | \  \  \  \  \ 1.2.2.RELEASE [rev 7d75659]

Welcome to Spring Roo. For assistance press TAB or type "hint" then hit ENTER.
@rooinaction: @SpringRoo Second blog post on testing Roo Add-ons is here - Mocki
to and the Argument Captor. http://t.co/S4wfLyAI
roo> finder list --class ~.domain.Airport
findAirportsByCodeEquals(String code)
findAirportsByCodeIsNotNull()
findAirportsByCodeIsNull()
findAirportsByCodeLike(String code)
findAirportsByCodeNotEquals(String code)
findAirportsByNameEquals(String name)
findAirportsByNameIsNotNull()
findAirportsByNameIsNull()
findAirportsByNameLike(String name)
findAirportsByNameNotEquals(String name)
~.domain.Airport roo>
```

DEMO

CRUD Scaffolding

- Einfache Funktionalität zur Pflege des Domainmodells
- Ausgangspunkt für weitere Entwicklung
- Vorlage für andere Funktionalität
- auch zum Lernen geeignet
- erzeugt wird @MVC Code

```
web mvc setup  
web mvc all --package ~.web
```

Starten der Anwendung

```
mvn tomcat:run
```



DEMO

Security

- SpringSecurity
- erzeugt Infrastruktur
- einfache Konfiguration
- Benutzer stehen in der Konfiguration
- Kenntnis des Frameworks nötig

`security setup`

DEMO

Vielen Dank!

Mirko Zeibig

IST GmbH Dresden

IST GmbH Dresden

- Softwareentwicklung, Consulting
- Konzeption, Entwicklung, Test
- bei uns und bei Ihnen
- Leichtgewichtige Architekturen
- EAI und Frontend (Portale, CMS)
- Architekturreviews, Training

<http://www.ist-dresden.de/blog/>



```
project --topLevelPackage com.istair
jpa setup --provider HIBERNATE --database HYPERSONIC_IN_MEMORY
perform eclipse
entity jpa --class ~.domain.Plane
field string --fieldName name --notNull
field string --fieldName typ --notNull
field boolean --fieldName isCargo --notNull
field number --fieldName capacity --type java.lang.Integer --notNull
entity jpa --class ~.domain.Airport
field string --fieldName name --notNull --sizeMin 3
field string --fieldName code --notNull --sizeMin 3 --sizeMax 3
entity jpa --class ~.domain.Flight
field string --fieldName nummer --notNull
field date --fieldName startDate --type java.util.Date --notNull
field reference --class ~.domain.Plane --fieldName home --type ~.domain.Airport
field reference --class ~.domain.Flight --fieldName plane --type ~.domain.Plane
field reference --fieldName start --type ~.domain.Airport --permitReservedWords
field reference --fieldName destination --type ~.domain.Airport
finder list --class ~.domain.Airport
finder add findAirportsByCodeLike
web mvc setup
web mvc all --package ~.web
security setup
quit
```