

# **Produtividade com Grails**

Por Rodrigo Urubatan Ferreira  
Jardim

# Sobre o Instrutor

- Rodrigo Urubatan - SCJP 1.4 e SCWCD
- Trabalha com arquitetura de sistemas J2EE e treinamento
- Já desenvolveu projetos utilizando as linguagens Delphi, C++, PHP, ASP, ColdFusion, Leather, Assembly, Perl, ...
- Trabalha com Java/J2EE a 4 anos e com desenvolvimento de sistemas a 9 anos
- Atualmente colabora com pequenas correções a alguns projetos Open Source como o GUV2, Lombok e Veloeclipse e faz parte da coordenação do RSJUG
- Já ministrou palestras em universidades (UCS, ULBRA, UNISC) e diversos eventos (Just Java, FISL, Seminário do RSJUG, Maratona 4 Java, Infosul) tutoriais para o RSJUG e já teve um artigo publicado na revista Mundo Java
- Atualmente trabalha como consultor na ConectaIT, como gerente de tecnologia e qualidade na Tech Office IT, e ministra cursos e alguns pequenos projetos pela USI Informática
- É o principal desenvolvedor do projeto Spring-Annotation

# Groovy

- O que é Groovy
- Quais as vantagens
- Quais as diferenças

# Grails

- O que é grails
- Quais os componentes
- Quais as vantagens

# Estrutura da aplicação

- grails-app
  - conf
  - controllers
  - domain
  - i18n
  - services
  - taglib
  - utils
  - views
- hibernate
- lib
- plugins
- scripts
- spring
- src
  - groovy
  - java
  - templates
    - artifacts
    - scaffolding
  - test
- webapp

# Um cadastro em 3 minutos

```
$ rails create-app exemplo
```

```
$ cd exemplo/
```

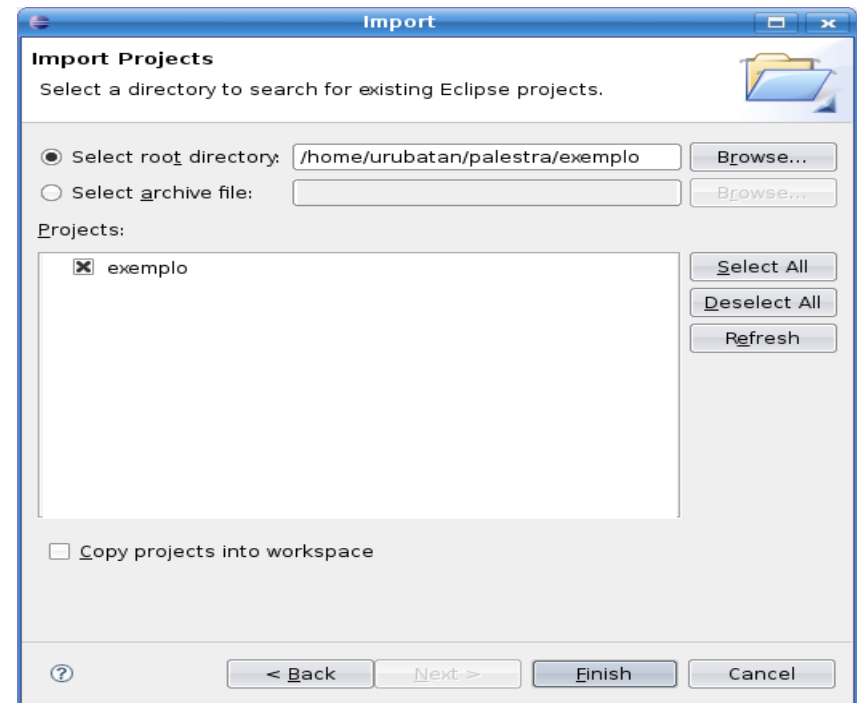
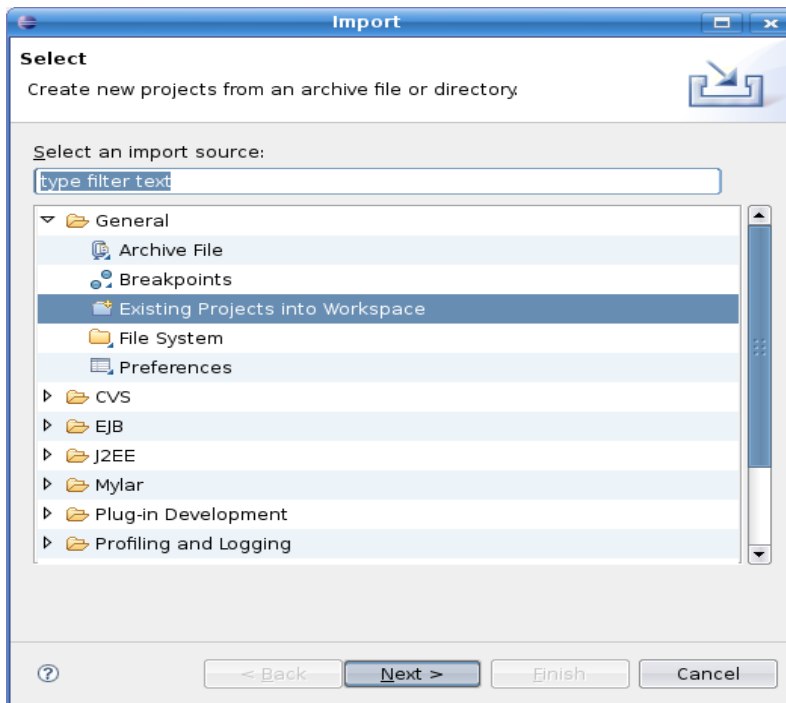
```
$ rails create-domain-class Cliente
```

```
$ rails create-controller Cliente
```

```
$ rails create-controller NotaFiscal
```

```
$ rails run-app
```

# Importando o projeto no eclipse



# Editando os Controllers

```
class ClienteController  
{  
static scaffold =  
  Client  
}
```

```
class  
  NotaFiscalController  
  {  
static scaffold =  
  NotaFiscal  
}
```

# Editando as classes de domínio

```
class Cliente {  
    static hasMany = [notas:NotaFiscal]  
    String nome  
}
```

```
class NotaFiscal {  
    static belongsTo = [cliente: Cliente]  
    Cliente cliente  
    Date data  
    Double value  
}
```

# Tudo Pronto!

grails run-app

# O que aconteceu

- O comando scaffold gera dinamicamente todas as views
- O comando run-app monta o um “war” descompactado e executa um servidor jetty para testes
- O banco de dados utilizado é o hsqldb



# Gerando o código para customização

\$ grails generate-all Cliente

\$ grails generate-all NotaFiscal

# Analizando o código Gerado

- O Controller de Clientes
- A view create.gsp
- A view edit.gsp
- A view list.gsp
- A view show.gsp

# Layouts

- Layout global
- Layout por controller

# Alterando o banco de dados

- Copiar o jar do driver do banco para a pasta “lib”
- Alterar o Data Source na pasta grails-app/conf

```
class DevelopmentDataSource {  
    boolean pooling = true  
    String dbCreate = "update" // one of 'create', 'create-drop', 'update'  
    String url = "jdbc:mysql:///teste"  
    String driverClassName = "com.mysql.jdbc.Driver"  
    String username = "root"  
    String password = "admin"  
}
```

# Mudando as listagens para Laszlo

- \$ grails install-plugin  
~/grails\_plugins/grails-Laszlo-0.4.2.zip
- \$ grails generate-laszlo Cliente
- \$ grails generate-laszlo NotaFiscal

# Dicas e Truques

- Customizando os templates do scaffold
- Utilizando o Spring
- Utilizando o Hibernate
- Outras classes Groovy
- Outras Classes Java

# Referencias

- <http://www.springframework.org>
- **<http://www.urubatan.com.br>**
- <http://www.guj.com.br>
- <http://grails.codehaus.org/>
- <http://www.portaljava.com>
- <http://www.hibernate.org>
- <http://www.opensymphony.com/sitemesh>
- <http://groovy.codehaus.org/>