Grails

Desenvolvimento Java Açucarado

Joás Garcia

TADS

asaas.com

me@joasgarcia.com

Agenda

08h30min - 12h

Coffee 10h - 10h15min

20/09 - Groovy e Grails

21/09 - Aplicação de exemplo

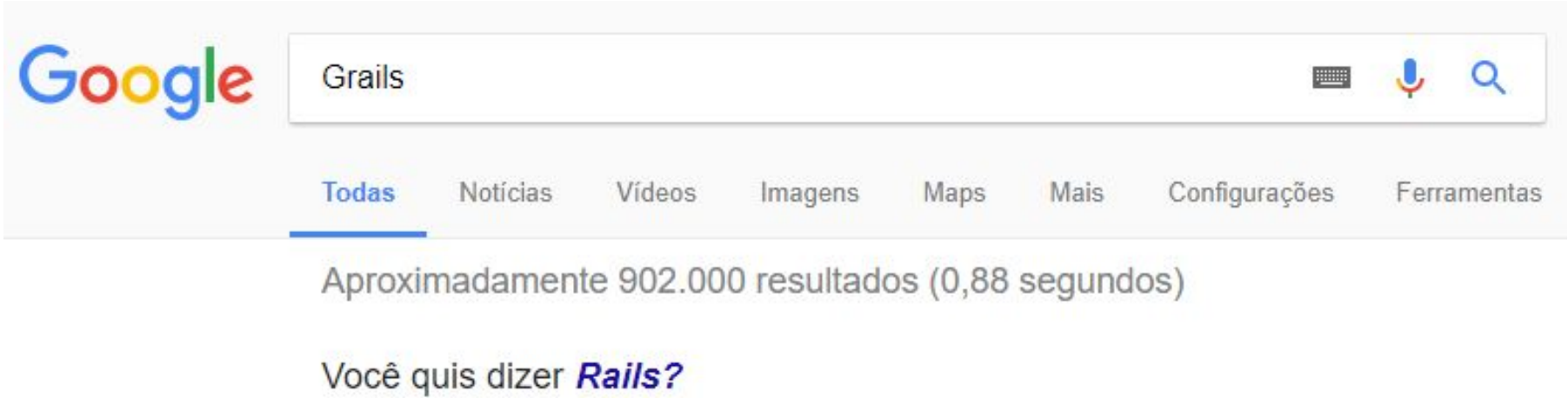
Sorteio de [2] ebooks “Falando de Grails”

Obrigado @casadocodigo
Para concorrer acesse: http://bit.ly/2hdDfuB

Quem conhece Grails?

- **Aplicações Java web**
- **Full-stack**
- **Baseado em Groovy**

Ah sim, conheço Rails!



Foi Groovy on Rails

Fundador do Ruby on Rails pediu para mudar o nome e tornou-se Grails.


Javeiro
RAILS
Struts

Groovy: A linguagem

Grails: O framework

Quem usa?

Joinville

- Informant
- ASAAS
- Dalmark Systems
- Motoboy.com

Brasil

- GuiaBolso

Mundo

- Mercado Livre
- Netflix
- Linkedin

Groovy: O Java açucarado

Java Developer Friendly

- Curva de aprendizagem baixa para quem já sabe Java
- Programação poliglota: Roda código Java dentro do código Groovy

Linguagem dinâmica

- Linguagens dinâmicas visam produtividade
- Mas também possibilidade compilação estática

Scripts

Dar uma olhada no Gradle (https://gradle.org/)
Porém, não se resume a uma linguagem somente para scripts

O nosso playground

https://groovyconsole.appspot.com

Características que deixam o desenvolvimento mais doce

Coisa opcionais

Começando a desburocratizar.

- Ponto e vírgula
- Parênteses
- `return` keyword
- Método `main`

```
println(“O clássico Hello World”)
```

Nem tudo é festa

1. Eu uso sempre o `return` explícito
2. Eu uso sempre parênteses

Legibilidade > Praticidade

Groovy Truth

Se algo é nulo, igual a zero ou vazio é interpretado numa expressão booleana como `false`

Groovy Truth - O conceito de verdade do Groovy

No Java é comum:

```
String foo

if (foo != null && foo != “”) {

    //Faça algo quando não é nulo

    //e nem branco

}
```

No Groovy basta isso:

```
String foo

if (foo) {

    //Faça algo quando não é nulo

    //e nem branco

}
```

Groovy Truth - O conceito de verdade do Groovy

No Java é comum:

```
List bar

if (bar != null && bar.size() > 0) {

     //Faça algo quando não é nulo

     //e nem vazio

}
```

No Groovy basta isso:

```
List bar

if (bar) {

     //Faça algo quando não é nulo

     //e nem vazio

}
```

Groovy Beans

Não é necessário declarar getter e setter, o Groovy faz dinamicamente para você :)

Groovy Beans

No Java:

```
Cliente {
     Private String nome
     public String getNome() {
          return this.name
     }
     public void setNome(String nome) {
          return this.nome = nome
     }
}
```

No Groovy basta isso:

```
Cliente {
     String name
}
```

Tipagem dinâmica

```
def a, b

a = 5

b = 6

def c = a + b // C será igual a 11.

d = c + a + b // D será igual a 22
```

Nem tudo é festa

Eu 95% dos casos uso tipagem estática

Closures

O supra-sumo
Tipo de dado que armazena código executável

Açúcar sintático no uso de listas

Açúcar sintático no uso de listas

```
.findAll {}
  .find {}
.collect {}
  .any {}
  .max {}
 .count {}
```

As Strings

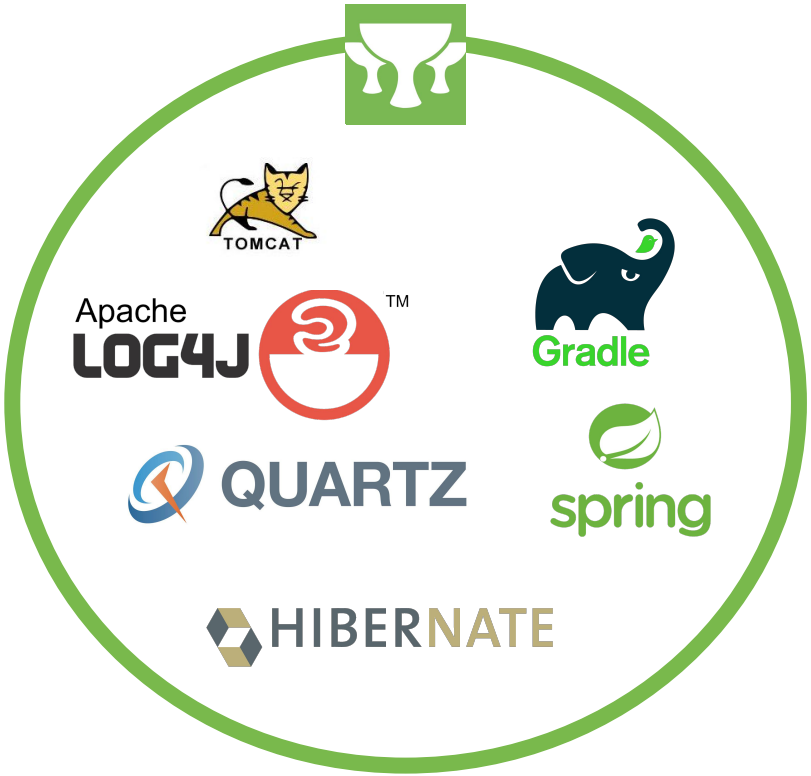
0 Grails

Por que Grails?

Programação por convenção

- Convention over configuration
- Sem infinidade de extensos arquivos XML para configuração
- “Siga as convenções que me viro com o resto”

Um meta-framework



Meta-framework

- Pronto pra uso = Setup rápido
- A integração entre as bibliotecas já vem pré-configurada
- Desenvolvedor se preocupa com o que realmente importa: a lógica de negócio do projeto

A aplicação

Setup | Passo 1: Variáveis de ambiente

Configurar variáveis de ambiente

`JAVA_HOME`

`GRAILS_HOME`

Teste para ver se está tudo certo rodando:

```
javac -version
```

```
grails -version
```

Setup | Passo 2: Configurando o banco

1. Criar um banco de dados “pagamentos”
2. Adicionar a dependência do conector MySQL na aplicação no arquivo build.gradle:

```
dependencies {
    runtime 'mysql:mysql-connector-java:5.1.29'
}
```

Setup | Passo 3: Criando a aplicação

```
grails create-app meuApp
```

Setup | Pass 4: Configurando o dataSource

Acessar o arquivo application.yml

```
dataSource
    driverClassName: com.mysql.jdbc.Driver
    username: root
    password: <SENHA DO USUÁRIO DO BANCO>

environments:
    development:
        dataSource:
            dbCreate: update
            url: "jdbc:mysql://localhost:3306/pagamentos"
```

Estrutura de diretórios

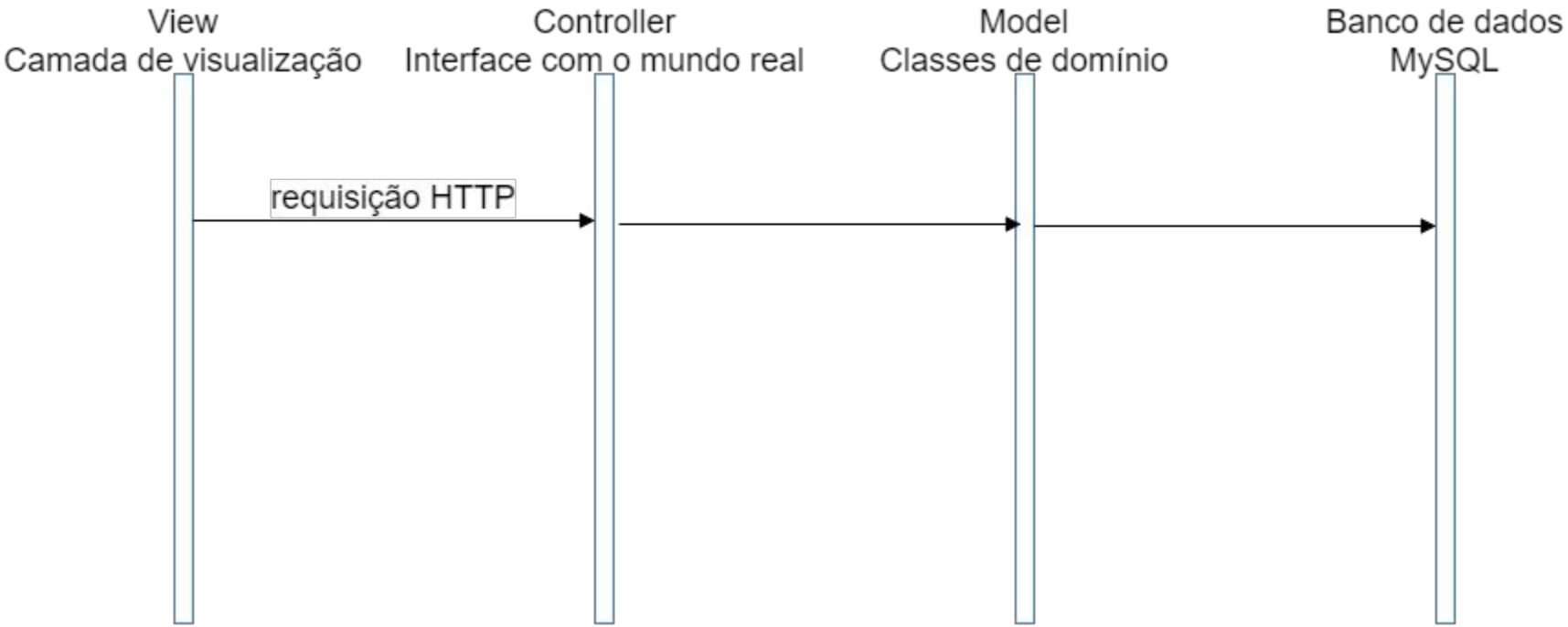
Fica claro a programação por convenção

O nosso modelo de dados

Pagamento
vencimento: Date formaDePagamento : FormaDePagamento valor: BigDecimal situacao: SituacaoPagamento descricao: String cliente: Cliente

Cliente
nome: String email: String cpf: String endereco: String

Como funciona



Contraints

Não pode criar uma cobrança vencida

Não pode criar uma cobrança com valor menor que zero

Não pode criar um Cliente com mesmo CPF

Não pode criar uma cobrança com a descrição maior que 200 caracteres

Scaffolding

O andaime para os seus CRUDs

E os pontos negativos do Grails?

Comunidade pouco ativa

Aplicação consome bastante memória como típica aplicação Java

Próximos passos

Tópicos avançados que valem a pena conferir

Metaprogramação

Multithreading

Criação de plugins Grails

Programação orientada a aspectos (Spring AOP)

Sitemesh

Referências

http://groovy-lang.org/

http://docs.grails.org/latest/

Grails: um guia rápido e indireto

Livro Falando de Grails

http://grailsbrasil.com.br/

http://www.itexto.net/devkico/ - Blog referência em Grails no Brasil