Um pouco do resto

GORM
Grails Object Relational Mapping

get()

def album = Album.**get**(params.id)

- Get recebe um id e retorna
  - Ou a instância do objeto correspondente ao id
  - Ou null caso não exista

getAll()

def album = Album.**getAll**(1,2,3)

- Recebe
  - Vários identificadores
- Retorna
  - Uma lista de instâncias

read()

def album = Album.**read**(params.id)

- Recebe
  - Um identificador
- Retorna
  - Um objeto em estado somente leitura

list()

- 4 atributos
  - **max**
    - Número máximo de instâncias a serem retornadas
  - **offset**
    - Qual o 0 relativo da consulta
  - **sort**
    - Nome da propriedade para utilizar no **ORDER BY**
  - **order**
    - Se a ordenação é ASC ou DESC

Classe Album

```
class Album {
     String titulo

     Date lancamento

     static hasMany = [musicas:Musica]

     static belongsTo = [artista:Artista]
}
```


list()

- def allAlbums = Album.list()
  - Recupera todos os albuns
- def top10 = Album.list(
  max:10, sort:'lancamentos', order:'desc')
  - Retorna os 10 albuns mais recentes
- def totalAlbums = Album.count()
  - Recupera o total de albuns

Classe Album

```
class Album {
     String titulo

     Date lancamento

     static hasMany = [musicas:Musica]

     static belongsTo = [artista:Artista]
}
```


listOrderBy*()

def tudoPorLancamento =
Album.listOrderByLancamento()

def tudoPorTitulo =
Album.listOrderByTitulo()

save()
Inserir uma nova instância

def album = new Album(params)
album.save()

save()
Alterar uma instância

```
def album = Album.get(1)

album.titulo = "Titulo Alterado"

album.save()
```


save()
Variação

```
album.save(insert:true)
```


delete()
Excluir do banco

```
album.delete()
```


findBy

- Retorna uma única instância

```
Album.findByTituloAndGenero(
"RATM", "Rock")
```


findBy

- Retorna uma única instância

```
Album.findByTituloOrGenero(
"RATM", "Rock")
```

Mais findBy

```
Album.findByLancamentoBetween(hoje-10,hoje)

Album.findByTituloEquals('RATM')

Album.findByLancamentoGreaterThan(ontem)

Album.findByLancamentoGreaterThanOrEqual(ontem)

Album.findByTituloInList(['123', 'Brasil'])
```


Mais findBy

```
Album.findByGeneroIsNull()

Album.findByGeneroIsNotNull()

Album.findByLancamentoLessThan(ontem)

Album.findByLancamentoLessThanOrEqual(ontem)

Album.findByTituloLike('O tempo não%')

Album.findByTituloNotEqual('Odelay")
```


Métodos "primos"

- **findAllBy***
  - Retorna uma lista baseado nos parâmetros
- **countBy***
  - Retorna um inteiro com o total de instância que satisfazem a consulta

Criteria

```
def c = Album.createCriteria()

def results = c.list {
  eq('genero', 'Alternativo')
  between('lancamento', new Date()-30, new Date())
}
```


Validação sem domínio

CommandObjects

Objetos Command

- Como uma classe de domínio, possui
  - Validação
  - DataBinding
  - Mas não é persistente
  - Exige a definição de uma classe
  - Podem ser criados no pacote dos controles
    - Ou dentro de um controle

Um login command

```
class LoginCommand {
 String login
 String senha

 static constraints = {
  username(blank:false, minSize:6)
  password(blank:false, minSize:6)
 }
}
```


Utilizando o Command

```
class UsuarioController {
   def login = {
   LoginCommand cmd ->
      if(cmd.hasErrors()) {
         redirect(action:'loginForm')
     } else {
          // faz outra coisa}
     }
}
```


Um formulario para o Album

```
<g:form url="[controller: 'album', action: 'save'] ">
  Titulo: <input type="text" name="titulo" /> <br>
  Artista: <input type="text" name="artista" /> <br>
  Musica 1: <input type="text" name="musicas[0]" /> <br>
  Musica 2: <input type="text" name="musicas[1]" /> <br>
      ...
</g:form>
```


AlbumCreateCommand

```
class AlbumCreateCommand {
    String artista
    String titulo
    List musicas = []
    List duracoes = []
    static constraints = {
        artista blank:false
        titulo blank:false
        musicas minSize:1, validator:{ val, obj ->
           if(val.size() != obj.duracoes.size())
           return "musicas.duracoes.not.equal.size"
        }
    }
```


AlbumCreateCommand

```
  Album createAlbum() {
     def artista = Artist.findByNome(artista) ?: new
  Artist(name:artista)
       def album = new Album(titulo:titulo)
       musicas.eachWithIndex { tituloMusica, i ->
          album.addToMusicas(titulo:titulo,
  duracao:duracoes[i])
       }
      return album
     }
}
```


Ajax

Ajax no Grails

- No modo básico
  - Grails dispõe um conjunto de tags que simplificam a criação de componentes Ajax
    - Links
    - Formulários
    - Caixas de texto

Que horas são?

```
class LojaController {
  def index = { }

  def exibirHora = {
     render "A hora é ${new Date()}"
  }
}
```

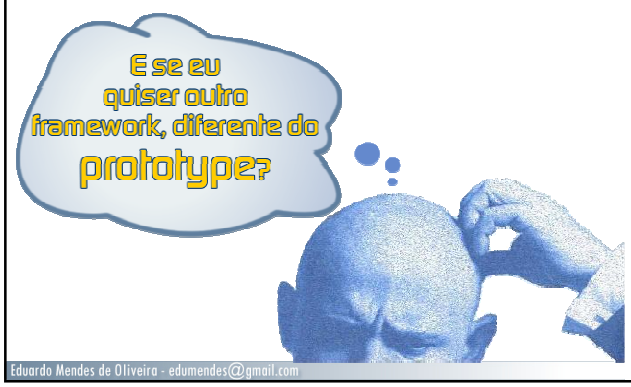

grails-app/views/loja/index.gsp

```
<head>
 <title>Benvindo ao jcTunes</title>
 <meta name="layout" content="main" />
 <g:javascript library="prototype" />
</head>
```


grails-app/views/loja/index.gsp

```
<g:remoteLink
  action="exibirHora" update="hora">
  Que horas são?
</g:remoteLink>

<div id="hora">
</div>
```

grails-app/views/loja/index.gsp

```
<head>
 <title>Benvindo ao jcTunes</title>
 <meta name="layout" content="main" />
 <g:javascript library="yui" />
</head>
```


O que vimos

O que vimos

O que nem vimos

- Mapeamento de URLs
- Controle de Fluxos
- Caching
- Consulta dinâmica com Criteria
- Transações com GORM
- Segurança
- Serviços
- CommandObjects à fundo

Dúvidas???

Obrigado!