MOCKDOMAIN()

Testando uma classe de domínio
mockDomain

```
void testDuracaoMinima() {
        mockDomain(Musica)

        def musica = new Musica(duracao: 0)

        assertFalse 'Validacao deve falhar', musica.validate()

        assertEquals "min", musica.errors.duracao
}
```


Entendendo os Controles

Introdução

- Os controles são classes responsáveis por **manipular requisições** que chegam à aplicação
  - O controle **recebe** a requisição
  - Geralmente, **realiza alguma tarefa** com a requisição
  - E finalmente decide **o que deve acontecer em seguida**
    - **Executar uma outra ação** dele ou de outro controle
    - Renderizar uma **visão**
    - Renderizar **informações** diretamente para uma visão

Introdução

- Existe um controle para cada requisição
- Não precisam herdar ou implementar uma interface específica

Definindo um controle

- Localização
  - grails-app/controllers
- Convenção
  - O nome da classe deve terminar com "Controller"

Ações de um controle

```
class
  SampleController {
  def first = {...}
  def second = {...}
  def third = {...}
  def fourth = {...}
}
```

- Ações são definidas como um campo
- Cada campo é atribuído a bloco de código utilizando uma closure Groovy
- Controller pode definir um número qualquer de ações

Mapeamentos, urls no Grails

```
class SampleController {
  def first = {...}
  ...
}
```

- A primeira parte de uma url representa qual controle está sendo acessado
- A segunda parte representa a ação a ser executada

nomeAplicacao/sample/first

Configurando a ação default

- Se o controle
  - Tem apenas 1 ação
    - Tal ação torna-se a default
  - Possui uma ação chamada index
    - A ação index será a default
  - Define uma propriedade defaultAction
    - Seu valor será a ação default

Controle
Tem apenas 1 ação

```
class SampleController {
  def list = {}
}
```

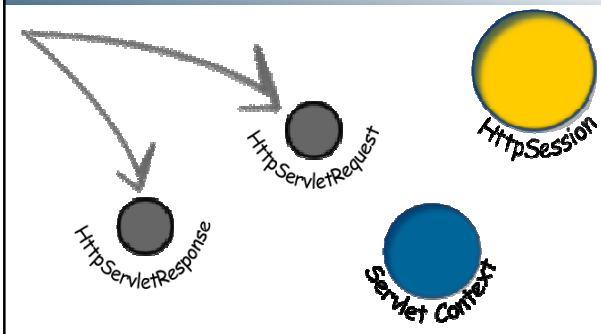

Controle
Possui uma ação chamada index

```
class SampleController {
  def list = {}
  def index = {}
}
```


Controle
com propriedade defaultAction

```
class SampleController {
  def defaultAction = 'list'
  def list = {}
  def index = {}
}
```

Na API Servlet



Na API Servlet

Objetos implícitos no controle

actionName	Nome da ação que está sendo executada
actionUri	Uri relativa a ação em execução
controllerName	Nome do atual controle em execução
controllerUri	Uri do controle em execução
flash	Objeto para acessar o escopo flash
log	Instância de org.apache.commons.logging.Log
params	Mapa de parâmetros de requisição
request	O objeto HttpServletRequest
response	O objeto HttpServletResponse
session	O objeto HttpSession object
servletContext	O objeto ServletContext

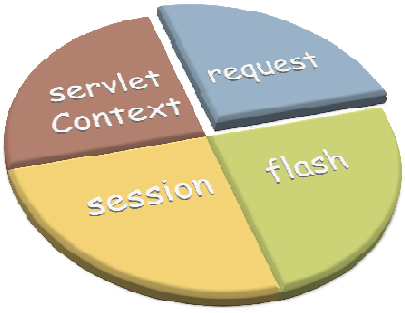
Comparação com Grails

- Servlet
  - request.getAttribute("myAttr");
  - request.setAttribute("myAttr", "myValue");
- Grails

  request.myAttr

  request.myAttr = "myValue"

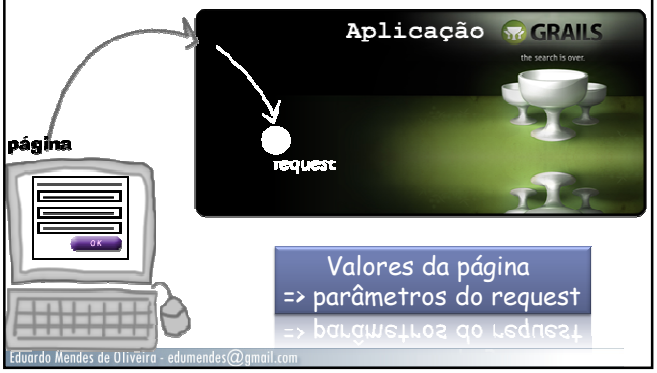
Escopos dos controles



Escopos dos controle

- request
  - Objetos colocados dentro de **request** estarão disponíveis durante a execução da requisição corrente
- flash
  - Objetos colocados dentro de **flash** estarão disponíveis para a requisição atual e para a próxima
- session
  - Objetos colocados no **session** serão mantidos até que a sessão do usuário seja invalidada, ou manualmente ou por expiração
- servletContext
  - Objetos colocados no servletContext são compartilhados por toda a aplicação e mantidos durante todo o tempo de vida da aplicação

Acessando parâmetros da requisição



Valores da página
=> parâmetros do request

Acessando parâmetros da requisição

- Via Servlet API

```
def userName =
  request.getParameter('userName')
log.info("User Name: ${userName}")
```

- Via propriedade params do Grails

```
def userName = params.userName
log.info("User Name: ${userName}")
```


Redirect
Redirecionando a requisição

Às vezes é necessário redirecionar o controle para outra ação de controle

Todo controle tem acesso a um método chamado **redirect**, que recebe um mapa como argumento

Redirecionando a requisição

```
class SampleController {
  def first = {
    // redireciona para a acao "second"
    redirect(action: "second")
  }
  def second = {
    // ...
  }
}
```


Redirecionando para outro controle

```
class SampleController {
 def first = {
  //redirecionar para a acao list de Loja
  redirect(action: "list", controller: "Loja")
 }
}
```


Argumentos do redirect

action • O nome da ação para redirecionamento	**controller** • O nome do controle para redirecionamento	**id** • Valor do parâmetro id para ser enviado no redirecionamento
params • Um mapa de parâmetros	**uri** • Um endereço relativo para redirecionar	**url** • Um endereço absoluto

Informações que chegam às visões

MODELO

Criando um modelo

- Uma das atividades fundamentais do controle
  - **Reunir dados** para serem renderizados na visão
- O controle pode realizar esta tarefa
  - **Ou delegar** para uma classe de serviço ou outros componentes
- Quando o controle faz esta tarefa, os dados são disponibilizados diretamente para as visões através de **um mapa**, o MODELO
- O mapa é retornado pelas ações de um controle

Retornando dados

```
class MusicaController {
  def show = {
    [ musica: Musica.get(params.id) ]
  }
}
```

- return é opcional
- Como a última expressão avaliada pela ação foi um mapa, então o mapa é o valor retornado desta ação
- O mapa pode ter qualquer quantidade de valores

class MusicaController {
  def show = {
    [musica: Musica.get(params.id)]
  }
}

Renderizando a Visão

- O controle MusicaController
  - 1 método chamado show
    - Retorna um modelo contendo uma chave e um objeto
- Onde está a referência para qual visão ele irá despachar o fluxo?
- Por convenção
  - grails-app/views/musica/show.gsp

E se eu quiser alterar a View de destino?

- Utilize o método **render**

```
class MusicaController {
  def show = {
      render(view:"exibir",
             model:[musica: Musica.get(params.id)])
  }
}
```

- Por convenção
  - grails-app/views/musica/exibir.gsp

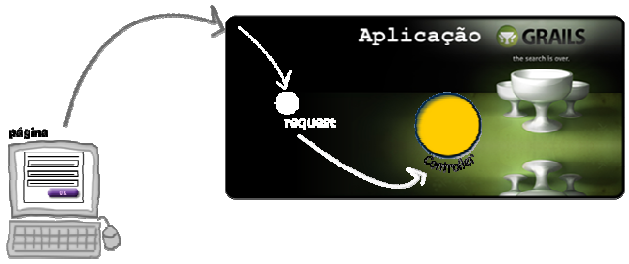
E para mudar o diretório?

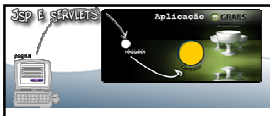
- Utilize um caminho absoluto

```
class MusicaController {
  def show = {
      render(view:"/outrapasta/exibir",
             model:[musica: Musica.get(params.id)])
  }
}
```

  - grails-app/views/outrapasta/exibir.gsp

Ligando dados da visão para o controle





Ligando dados

- Controles também precisam
  - Criar **novos objetos** de domínio
  - **Popular** as propriedades destes objetos com valores recebidos como parâmetros na requisição