

## Configuração do VRaptor

### Bem vindo ao curso de VRaptor3 da Alura

Oi pessoal, recomendamos o curso de VRaptor3 apenas se você trabalha com algum projeto legado que utiliza do mesmo. Caso você deseje aprender o VRaptor, recomendamos que comece na última versão dele, o [VRaptor4](https://www.alura.com.br/curso-online-desenvolvimento-web-com-vraptor-4) (<https://www.alura.com.br/curso-online-desenvolvimento-web-com-vraptor-4>).

### Frameworks web Java

Desenvolver aplicações web usando Java é uma das opções mais procuradas pelos programadores, devido a todas as vantagens da plataforma. Muitos \*frameworks\* facilitam esse trabalho, e os mais conhecidos são o Struts, o Spring MVC, o JSF, entre outros.

Para facilitar ainda mais o desenvolvimento web, a Caelum criou o **VRaptor**, um framework que faz uso de todas as boas práticas de desenvolvimento recomendadas.



### Importação do projeto

Antes de começar, precisamos instalar o projeto que será utilizado durante todo o curso. O projeto em estudo será um cadastro de produtos. Nesse exemplo, o aluno será capaz de usar grande parte dos recursos oferecidos pela ferramenta.

Você pode baixar o projeto pelo link [vraptor-produtos.zip](http://s3.amazonaws.com/caelum-online-public/FJ-23/vraptor-produtos.zip) (<http://s3.amazonaws.com/caelum-online-public/FJ-23/vraptor-produtos.zip>).

Depois extraia e importe o projeto no Eclipse. Veja que este é um projeto web java padrão. O filtro do VRaptor está configurado dentro do `web.xml` :

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app ....>
  <display-name>vraptor-produtos</display-name>
  <filter>
    <filter-name>vraptor</filter-name>
    <filter-class>br.com.caelum.vraptor.VRaptor</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>vraptor</filter-name>
    <url-pattern>/*</url-pattern>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>REQUEST</dispatcher>
```

```
</filter-mapping>
</web-app>
```

Note que uma classe de domínio já existe no projeto: o modelo chamado `Produto` contém `nome`, `descrição`, `preço`:

```
public class Produto {

    private Long id;
    private String nome;
    private String descricao;
    private double preco;

    //construtores

    //getters e setters
}
```

Também já há uma classe `ProdutoDAO`, que simula o acesso ao banco de dados.

## Instalação do Tomcat

Vamos usar o Tomcat na versão 7 para rodar a aplicação. Você pode baixar o servidor pelo link: [Apache Tomcat 7 \(http://tomcat.apache.org/download-70.cgi\)](http://tomcat.apache.org/download-70.cgi). Depois extraia o ZIP e configure o Tomcat dentro do Eclipse. Para tal, utilizaremos a aba "*view Server*". Caso essa janela tenha sido fechada, podemos reabri-la usando o atalho `Ctrl + 3` e digitando *server*.

Vamos, então, preparar o Tomcat definindo uma nova configuração do servidor. Na caixa de diálogo *New server*, escolheremos **Apache Tomcat v7.0 Server**. Em seguida, apertando *next*, definimos o lugar onde o Tomcat foi instalado (ou descompactado). No nosso caso, como já temos uma distribuição do Tomcat baixada e extraída, vamos apontar para ela. Na próxima aba associe o projeto ao Tomcat. Ao finalizar, aparecerá o Tomcat na *view Server*.

Já podemos subir o Tomcat e acessar a aplicação. No navegador, digite a seguinte URI:

<http://localhost:8080/vraptor-produtos> (<http://localhost:8080/vraptor-produtos>)

## O primeiro \*controller\*

Para respondermos a uma requisição HTTP, usamos os controllers do VRaptor. Nesse caso, abrindo a classe `IndexController` com o `CTRL+SHIFT+T`, temos o método `root` que está mapeado para a URI `/`. Vamos adicionar uma chamada ao `System.out.println` para imprimir a mensagem *Meu sistema com VRaptor*. Lembre de usar o `CTRL+ESPAÇO` para auto-preenchimento.

```
@Resource
public class IndexController {

    @Path("/")
    public void root() {
        System.out.println("Meu sistema com vraptor");
    }
}
```

Depois de reiniciar o servidor basta acessar a URI e verificar o resultado no console.

## Mapeamento de URIs para métodos

Para dizer que este controller representa algo disponibilizado na web, a anotação `@Resource` é necessária em todos os controladores. Como vimos, o mapeamento é feito através do `@Path`. Nesse caso, indica que a ação responderá por "/", ou seja, a raiz do projeto.

Após a execução desse método, o VRaptor vai, por padrão, para o arquivo `WEB-INF/jsp/index/root.jsp`. Note como o nome da pasta reflete o nome do controller (Index) e o nome do arquivo reflete o nome do método.

```
<html>
```

```
<body>
```

Parabéns, o VRaptor está funcionando, e você já pode passar para a próxima seção!

Essa aqui será a futura página de entrada da loja!

```
</body>
```

```
</html>
```