

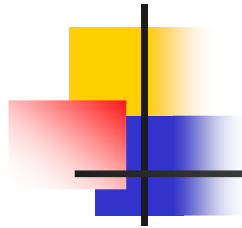
# Java Enterprise Edition - JEE

---

## 11. Struts Framework



Esp. Márcio Palheta  
gtalk: [marcio.palheta@gmail.com](mailto:marcio.palheta@gmail.com)



# Agenda

---

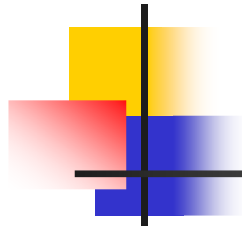
- Utilizar o struts para controlar sua lógica de negócios;
- Criar atalhos para camada de visualização;
- Criar e configurar mapeamentos de ações e templates;
- Utilizar **formbeans** para facilitar a leitura de formulários;
- Validar seu formulário de forma simples;
- Controlar os erros de validação do mesmo.



# Struts

---

- **Struts** é um **framework** do grupo Apache que serve como o **controller** de uma arquitetura **MVC**;
- Documentação e bibliotecas necessárias: <http://struts.apache.org>
- Vamos trabalhar com a versão 1.x do struts, por ser o padrão atual de mercado;



## Struts – continuação

---

- Possui integração com outras tecnologias para suporte ao *Model* e ao *View*:
  - Model: JavaBeans e classes utilitárias
  - View: TagLibs para utilização nas páginas JSP
- Utiliza arquivos de configuração XML para juntar todos os componentes



# Struts – Por quê?

---

- O Struts funciona como um controlador central entre as regras de negócio e a interface com o usuário
- Vantagens:
  - Separação entre lógica de apresentação e lógica de negócio
  - Você pode reaproveitar partes da regra de negócio em diferentes fluxos
  - Um lugar central para controlar a interação entre as camadas

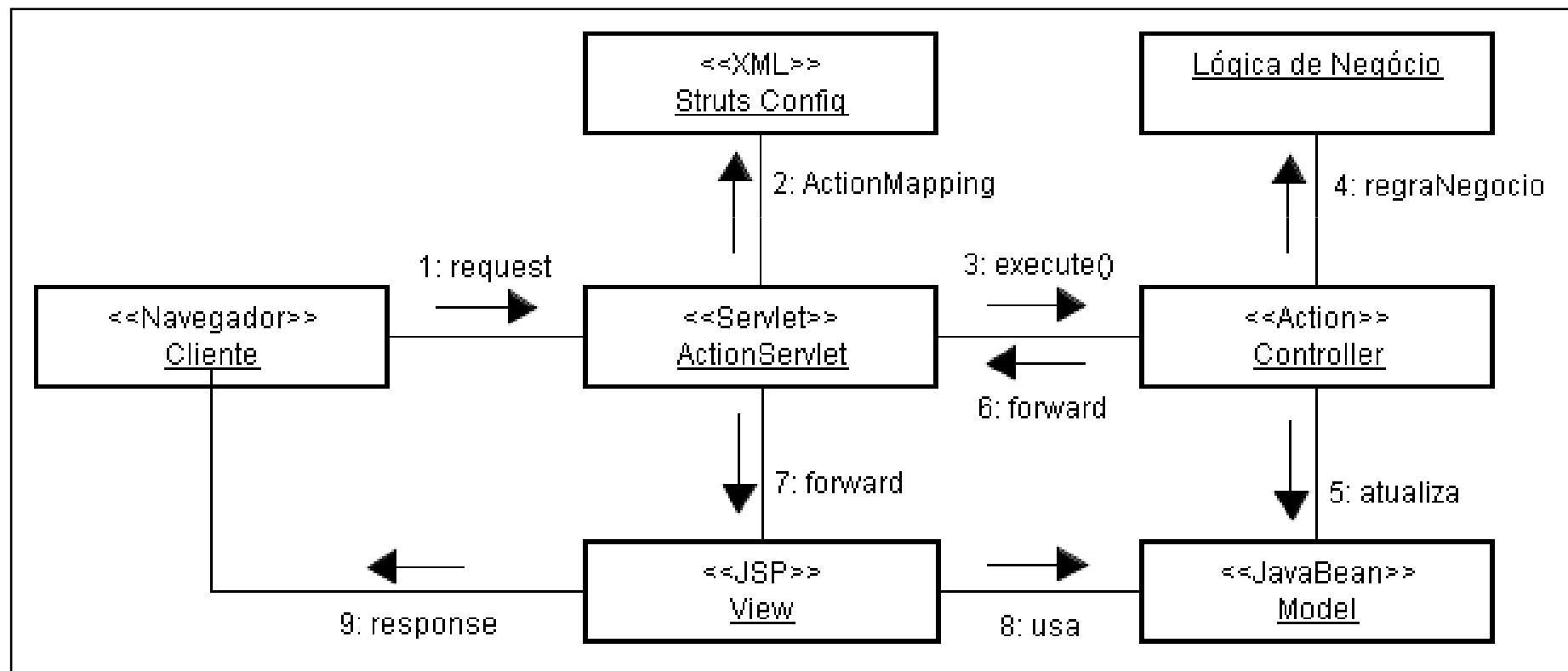


## Ainda tem mais...

---

- Outras funcionalidades:
  - **Internacionalização:** suporte através de *ResourceBundles* e TagLibs
  - **Gerenciamento de erros:** mensagens de erro geradas nas regras de negócio podem ser exibidas para o usuário
  - **Validação de campos:** suporte à validação dos dados de entrada de um *Form*

# Struts – Visão Geral





# Struts - componentes

---

## ■ **ActionServlet**

- Responsável pelo controle do fluxo de navegação
- Esse fluxo é definido no arquivo de configuração (`struts-config.xml`)
- Fluxo pode ser alterado sem recompilar a aplicação

## ■ **Action**

- Classe de controle responsável por:
  - Acessar a camada de negócio
  - Construir a resposta para a requisição
  - Controle e tratamento de erros
- Redireciona o fluxo de execução para uma página JSP (ou outra *Action*)





# Struts - componentes

---

## ■ **ActionForms**

- São JavaBeans
- Coletam informações de formulários para os objetos Action
- Podem ser utilizados para preencher formulários automaticamente

## ■ **Páginas JSP**

- Responsáveis apenas pela apresentação dos resultados
- Tipicamente devem ter muito pouco código Java (devem usar as taglibs do Struts)



# Struts - componentes

---

## ■ **ActionMapping**

- Informa ao *ActionServlet* quais *Action*, *ActionForm* e *ActionForwards* devem ser utilizados para cada URI
- Definidos no arquivo de configuração (`struts-config.xml`)

## ■ **ActionForward**

- Associa um nome lógico para um encaminhamento (URI para um JSP ou outra *Action*)
- Definidos no arquivo de configuração (`struts-config.xml`)
- Geralmente uma requisição é encaminhada primeiramente para uma *Action* e depois para um JSP



# Fluxo de requisição

---

- O servidor recebe uma requisição a um Servlet (ActionServlet);
- O container transfere o controle para o **ActionServlet**;
- O ActionServlet encontra o **ActionMapping** correspondente à URL requisitada;
- O ActionMapping especifica qual classe **Action** deve ser utilizada e o **ActionForm** associado;
- O **ActionForm** é populado com os dados de formulário da requisição



## Fluxo de requisição (Cont.)


- O controle é transferido para *Action* correspondente (através de uma chamada do método `execute()`);
- A *Action* utiliza os dados do *ActionForm* e acessa a camada de negócio (*Model*) para gerar uma resposta para a requisição;
- Uma vez gerada a resposta, a *Action* a armazena em algum contexto do contêiner (tipicamente `request` ou `session`) para acesso pela página JSP;
- A *Action* retorna um *ActionForward* para o *ActionServlet* indicando a página JSP (ou outro recurso) que exibirá a resposta



## Fluxo de requisição (Cont.)

---

- A página JSP constrói o resultado com informações obtidas dos contextos utilizados (`request` ou `session`) ou do *ActionForm*;
- A página JSP encaminha o resultado para o contêiner;
- O contêiner encaminha a resposta para o cliente que iniciou a requisição

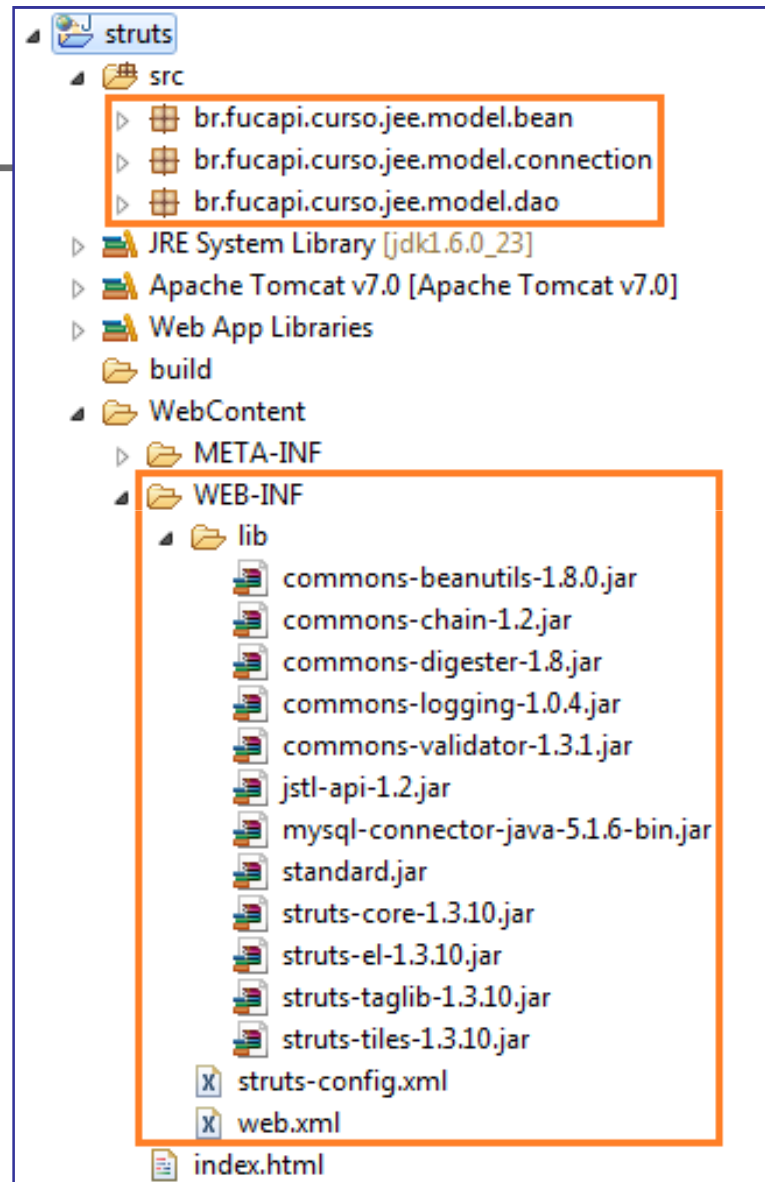


# Configuração do 1º projeto utilizando o struts:

---

- Crie um novo **Dynamic Web Project** chamdo **struts**;
- Copie os drivers utilizados para a pasta WEB-INF/LIB:MySQL, JSTL e Struts;
- Importe a camada do modelo **Empresa**
- Crie os arqs: **index.html** e **struts-config.xml**
- Publique a aplicação no Tomcat;
- Teste a aplicação:  
<http://localhost:8080/struts>

# Estrutura inicial do projeto





# Arquivos de configuração

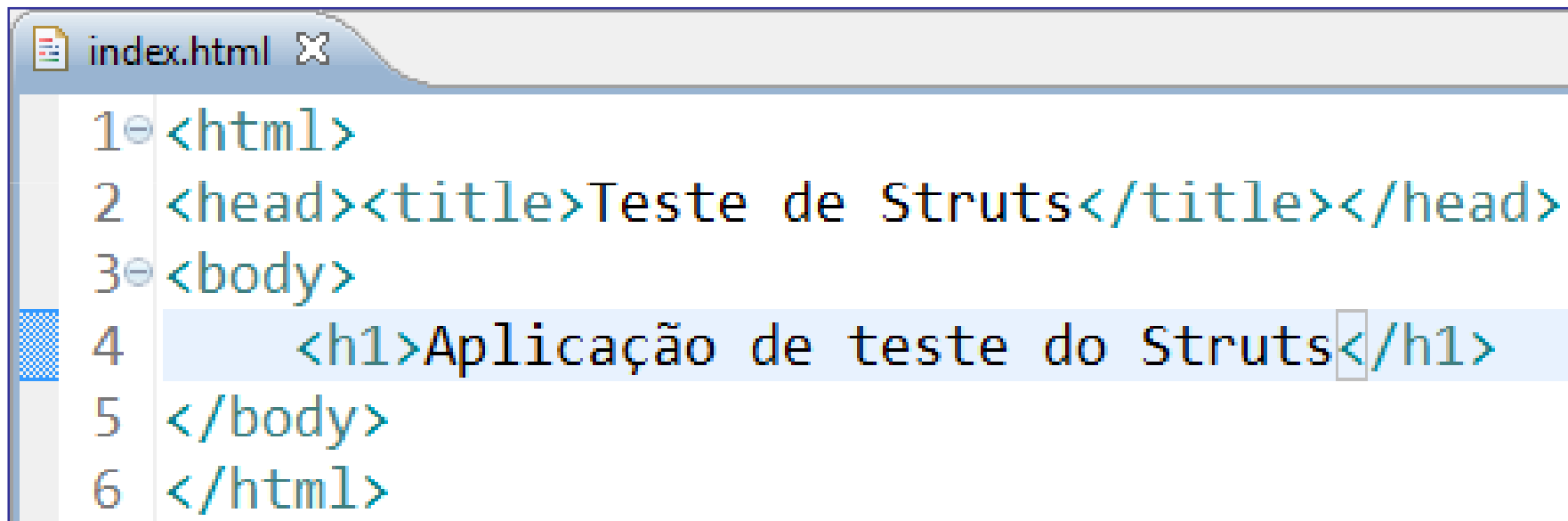
```
<!-- WebContent/WEB-INF/web.xml -->  
<servlet>  
  <servlet-name>struts</servlet-name>  
  <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>  
  <load-on-startup>1</load-on-startup>  
</servlet>  
<servlet-mapping>  
  <servlet-name>struts</servlet-name>  
  <url-pattern>*.do</url-pattern>  
</servlet-mapping>
```

```
<?xml version="1.0" encoding="UTF-8" ?>  
<!-- WebContent/WEB-INF/struts-config.xml -->  
<struts-config>  
  
</struts-config>
```



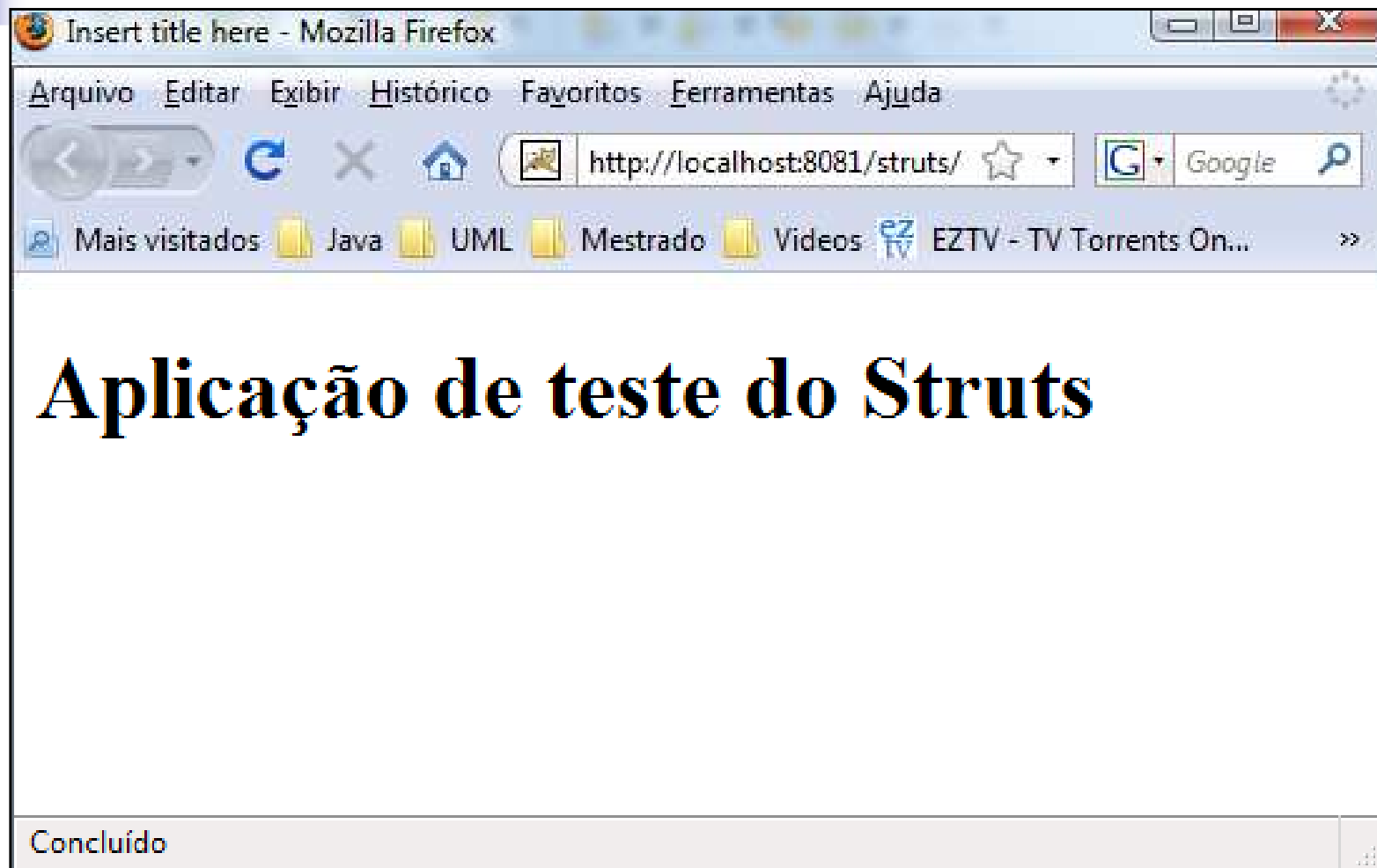


# WebContent/index.html



```
1 <html>
2 <head><title>Teste de Struts</title></head>
3 <body>
4   <h1>Aplicação de teste do Struts</h1>
5 </body>
6 </html>
```

# Teste da aplicação





# Passos para a configuração da aplicação com Struts:

---

- Crie a página que será chamada pela classe do **struts**;
- Crie uma nova classe filha de **Action**;
- Configure o arquivo **struts-config.xml** da aplicação;



# Página exemplo.jsp

```
1<%@ page language="java"
2    contentType="text/html; charset=ISO-8859-1"
3    pageEncoding="ISO-8859-1"%>
4<html>
5<head>
6<title>Página de exemplo</title>
7</head>
8<body>
9    <h1>Minha primeira página com Struts</h1>
10</body>
11</html>
```



## Uma ação no struts

---

- No struts, a lógica de negócio é chamada a partir de classe que estendem `Action`;
- Como nossas classes de controle são filhas da classe `Action`, herdam o método `execute()`;
- Sobrescreva o método citado, com o código necessário à sua aplicação;

# Primeira classe Action

```
package br.fucapi.curso.jee.control.action;

import javax.servlet.http.HttpServletRequest;

//Classe filha da classe Action do struts
public class ActionSimples extends Action{

    @Override
    //Metodo invocado pela classe ActionServlet do Struts
    public ActionForward execute(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {

        //Regras de negócio da aplicacao AQUI
        System.out.println("Executando lógica de negócio");

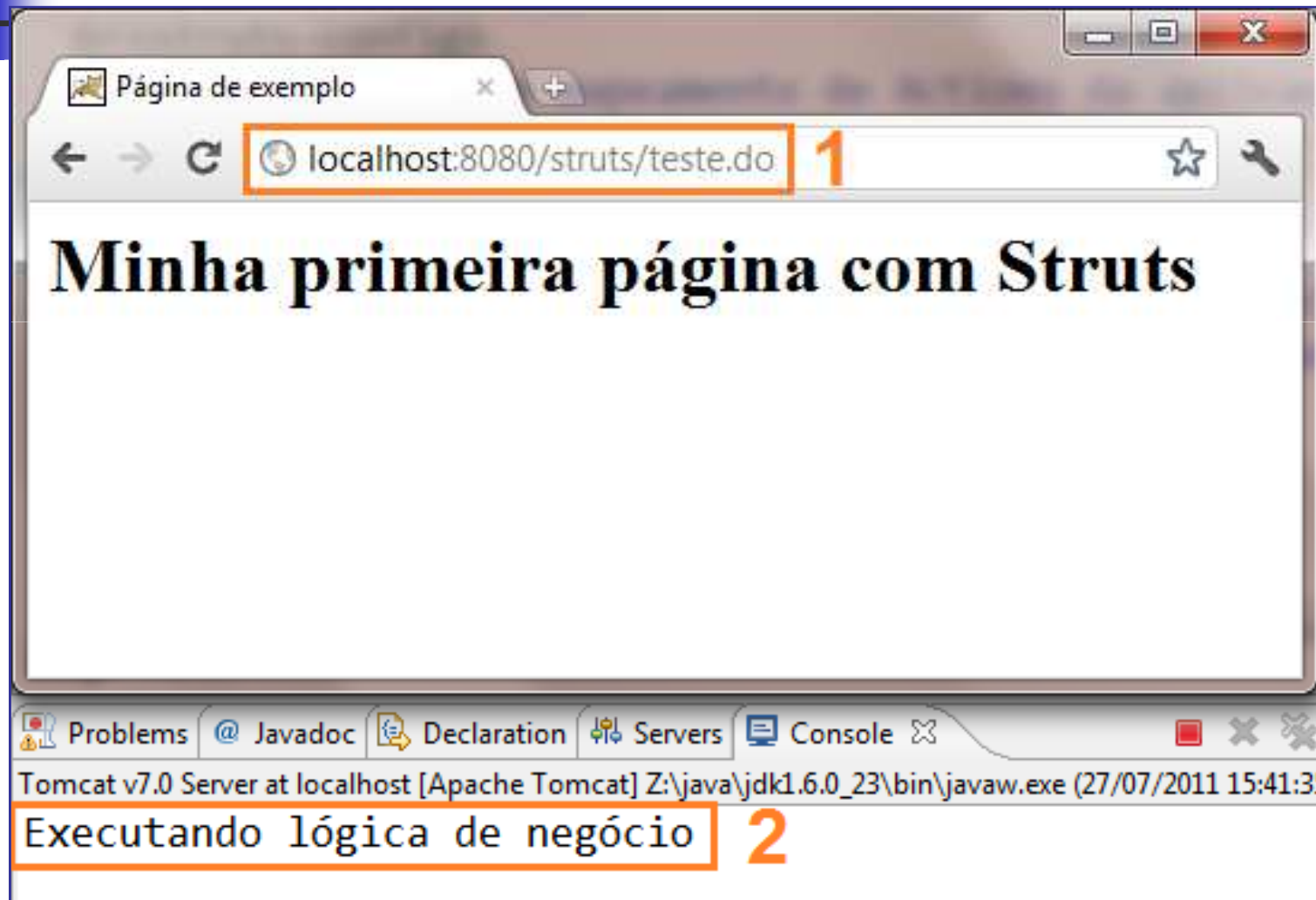
        //Retorno do forward que deve ser executado
        return mapping.findForward("sucesso");
    }
}
```



# struts-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- WebContent/WEB-INF/struts-config.xml -->
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.2//EN"
    "http://jakarta.apache.org/struts/dtds/struts-config_1_2.dtd">
<struts-config>
    <!-- Lista de mapeamento de Actions da aplicação -->
    1 <action-mappings>
        <!-- Mapeamento da classe Action -->
        2 <action path="/teste"
            type="br.fucapi.curso.jee.control.action.ActionSimples">
            <forward name="sucesso" path="/exemplo.jsp"/>
        </action>
    </action-mappings>
</struts-config>
```

# Teste da aplicação struts



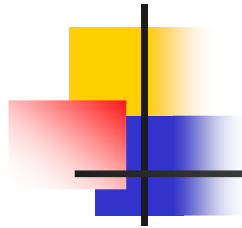




# Erros comuns

---

- O erro mais famoso nos primeiros exemplos de uma Action do Struts é colocar o **nome** do forward de **maneira inválida**, por exemplo, em minúsculo no struts-config.xml e em maiúsculo na sua classe.
- Como o Struts não encontra um redirecionamento com tal chave, o método **findForward** retorna null, resultado: uma **tela em branco**.



## Erros comuns 2

---

- Outro erro comum é esquecer de colocar a barra antes do nome do redirecionamento.
- Todo path de **forward** deve começar com uma barra.
- Se você colocar somente **exemplo.jsp** o erro diz que faltou uma Barra;




# Criação da aplicação em três camadas, utilizando struts

---

- No exemplo a seguir, vamos criar a estrutura necessária para usar JSP, Struts, DAO e JDBC;
- Criação do filtro de conexões;
- A Action recebe uma requisição;
- Acessa a classe DAO;
- E redireciona para a tela de listagem;

# Filtro de conexões



```
package br.fucapi.curso.jee.control.filter;

import java.io.IOException;

@WebFilter("/*")
public class ConnectionFilter implements Filter {
    public void init(FilterConfig fConfig) throws ServletException {
        System.out.println("Inicio do filtro de conexoes"); }
    public void destroy() {
        System.out.println("Fim do filtro de conexoes"); }
    public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
        Connection connection = ConnectionFactory.getConnection();
        // Colocando a connection na requisicao
        request.setAttribute("connection", connection);
        // Execucao da requisicao
        chain.doFilter(request, response);
        // Fechamento da conexao
        try {
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

# A classe ListarEmpresasAction

```
package br.fucapi.curso.jee.control.action; 1
import java.sql.Connection;
public class ListarEmpresasAction extends Action {
    @Override
    public ActionForward execute(
        ActionMapping mapping, ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response) throws Exception {
        //Executa a regra de negócio
        Connection connection =
            (Connection) request.getAttribute("connection");
        2 EmpresaDAO dao = new EmpresaDAO(connection);
        List<Empresa> listaEmpresas = dao.listar();
        request.setAttribute("colecacaoDeEmpresas", listaEmpresas);

        //redireciona para a próxima tela
        3 return mapping.findForward("listar");
    }
}
```



# Como serão as telas?

---

- Scriptlets, struts-logic ou JSTL?
- A **JSTL** é o padrão atual de mercado, utilizada, inclusive pelo grupo Apache;
- Resultado: **JSTL**
- A partir deste ponto, a estrutura das páginas JSP será baseada em JSTL;

# WebContent/listagem.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
  <head><title>Tela de listagem de empresas</title></head>
  <body>
    <table border="1">
      <c:forEach var="empresa" items="${colecaoDeEmpresas}" >
        <tr>
          <td>${empresa.cnpj}</td>
          <td>${empresa.razaoSocial}</td>
        </tr>
      </c:forEach>
    </table>
  </body>
</html>
```

# Atualização do arquivo struts-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- WebContent/WEB-INF/struts-config.xml -->
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.2//EN"
    "http://jakarta.apache.org/struts/dtds/struts-config_1_2.dtd">
<struts-config>
    <action-mappings>
        <action path="/teste"
            type="br.fucapi.curso.jee.control.action.ActionSimples">
            <forward name="sucesso" path="/exemplo.jsp"/>
        </action>
        <action path="/listarEmpresas"
            type="br.fucapi.curso.jee.control.action.ListarEmpresasAction">
            <forward name="listar" path="/listagem.jsp"/>
        </action>
    </action-mappings>
</struts-config>
```

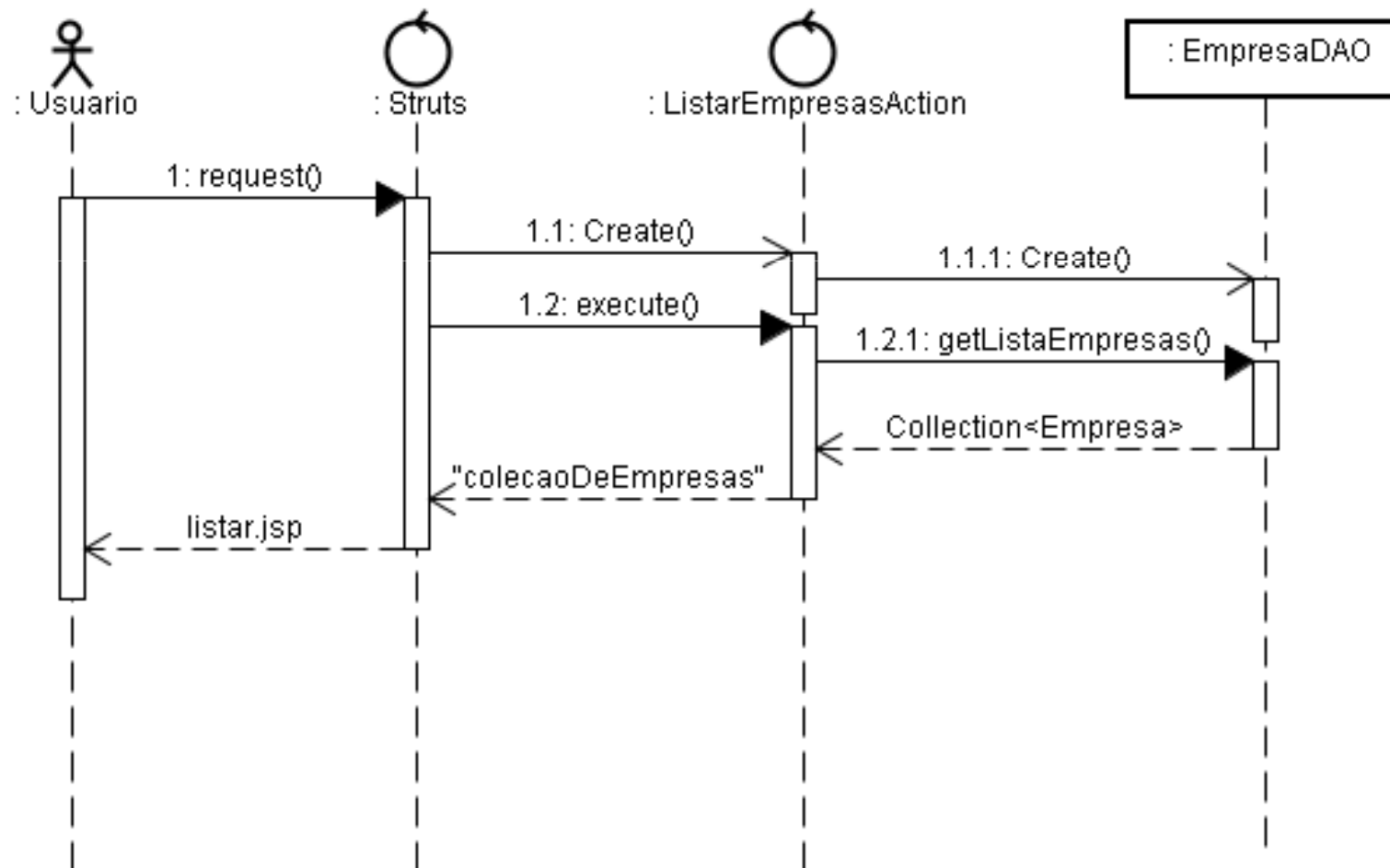


# Acesso à aplicação

- <http://localhost:8080/struts/listarEmpresas.do>



# Diagrama de sequencia do STRUTS





# Mapeamento de resultado condicional

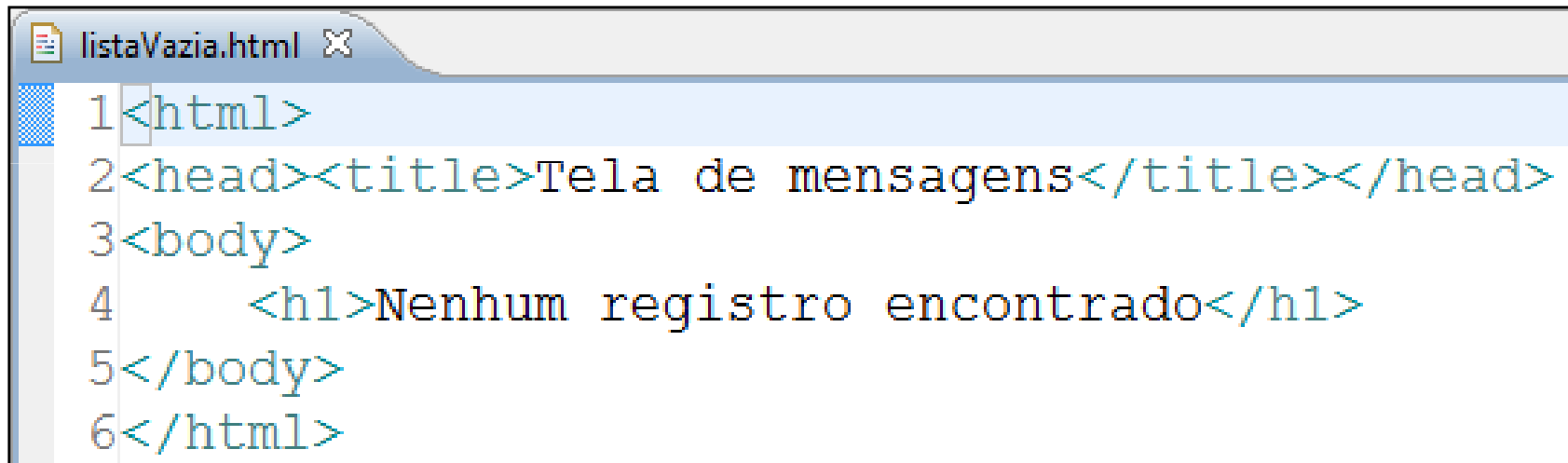
---

- Precisamos, agora, mostrar a mensagem “**Lista vazia**” quando nenhum registro for encontrado;
- Siga os passos a seguir:
- Crie a página **listaVazia.html**;
- Altere a ListarEmpresasAction.java
- Altere o struts-config.xml;



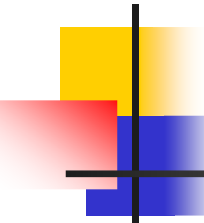
# listaVazia.html

---



```
listaVazia.html ✕  
1<html>  
2<head><title>Tela de mensagens</title></head>  
3<body>  
4    <h1>Nenhum registro encontrado</h1>  
5</body>  
6</html>
```

# ListarEmpresasAction.java



```
package br.fucapi.curso.jee.control.action;
import java.sql.Connection;
public class ListarEmpresasAction extends Action {
    @Override
    public ActionForward execute(
        ActionMapping mapping, ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response) throws Exception {
        //Executa a regra de negócio
        Connection connection =
            (Connection) request.getAttribute("connection");
        EmpresaDAO dao = new EmpresaDAO(connection);
        List<Empresa> listaEmpresas = dao.listar();
        request.setAttribute("colecãoDeEmpresas", listaEmpresas);
        //redireciona para a próxima tela
        if(listaEmpresas.size()>0){
            return mapping.findForward("listar");
        }else{
            return mapping.findForward("semRegistros");
        }
    }
}
```



# struts-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- WebContent/WEB-INF/struts-config.xml -->
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.2//EN"
    "http://jakarta.apache.org/struts/dtds/struts-config_1_2.dtd">
<struts-config>
    <action-mappings>
        <action path="/teste"
            type="br.fucapi.curso.jee.control.action.ActionSimples">
            <forward name="sucesso" path="/exemplo.jsp"/>
        </action>
        <action path="/listarEmpresas"
            type="br.fucapi.curso.jee.control.action.ListarEmpresasAction">
            <forward name="listar" path="/listagem.jsp"/>
            <forward name="semRegistror" path="/listaVazia.html"/>
        </action>
    </action-mappings>
</struts-config>
```

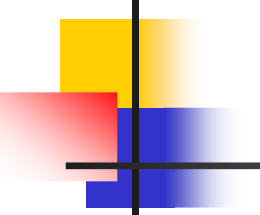


# Passos para o cadastro de novas empresas

---

- 1) Criar a lógica de negócios;
- 2) Criar a JSP de visualização;
- 3) Criar o mapeamento da lógica para a visualização;
- E depois os passos opcionais:
- 4) Criar a validação do formulário na lógica de negócios
- 5) Criar o controle de erro na visualização

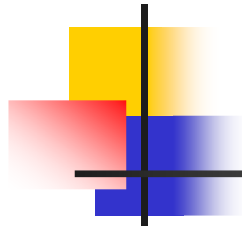
# Atualização da classe Empresa



```
public class Empresa {
    private Long id;
    private String cnpj;
    private String razaoSocial;
    private String endereco;
    private String telefone;
    private String site;
    private String email;
    private Calendar dataCriacao;

    public void setDataCriacao(String dataSTR) {
        try {
            SimpleDateFormat format;
            format = new SimpleDateFormat("dd/MM/yyyy");
            Date date = format.parse(dataSTR);
            dataCriacao = Calendar.getInstance();
            dataCriacao.setTime(date);
        } catch (ParseException e) {
            e.printStackTrace();
        }
    }
    // Metodos de get e set aqui
}
```





# Utilização de formulários

---

- O Struts possui uma classe chamada **ActionForm** que, ao ser estendida, permite a leitura dos parâmetros do **request**, sem nos preocuparmos com o comando **request.getParameter()**;
- Para cada formulário HTML que existe em nosso site, devemos criar uma classe em Java para representar seus campos.



# EmpresaForm.java

```
package br.fucapi.curso.jee.control.form;
import org.apache.struts.action.ActionForm;
import br.fucapi.curso.jee.model.bean.Empresa;

public class EmpresaForm extends ActionForm{
    private static final long serialVersionUID = 1L;
    //Atributos equivalentes aos campos de html:form
    private String dataCriacaoSTR;
    private Empresa empresa = new Empresa();
    public Empresa getEmpresa() {
        return empresa;
    }
    public String getDataCriacaoSTR() {
        return dataCriacaoSTR;
    }
    public void setDataCriacaoSTR(String dataCriacaoSTR) {
        this.dataCriacaoSTR = dataCriacaoSTR;
    }
}
```



# Mapeamento do Form no struts-config.xml

---

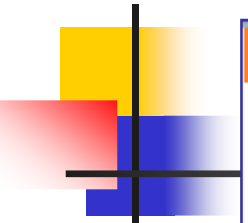
- Assim como a **Action**, devemos configurar nosso **ActionForm** no arquivo **struts-config**.
- Para isso, usamos a tag chamada **form-bean**, com os seguintes atributos:
  - **name**: Nome ou apelido da classe;
  - **type**: Classe que representa o formulário;
- A seguir, veja a atualização do arquivo:

# Atualização do formulário no struts-config.xml

- A tag **form-beans** antes de action-mappings

```
<struts-config>
  <form-beans>
    <form-bean name="EmpresaForm"
               type="br.fucapi.curso.jee.control.form.EmpresaForm" />
  </form-beans>
  <action-mappings>
    <action path="/teste"
           type="br.fucapi.curso.jee.control.action.ActionSimples">
      <forward name="sucesso" path="/exemplo.jsp"/>
    </action>
    <action path="/listarEmpresas"
           type="br.fucapi.curso.jee.control.action.ListarEmpresasAction">
      <forward name="listar" path="/listagem.jsp"/>
      <forward name="semRegistor" path="/listaVazia.html"/>
    </action>
  </action-mappings>
</struts-config>
```

# Action para inclusão



```
package br.fucapi.curso.jee.control.action;
import java.sql.Connection;
public class AdicionarEmpresaAction extends Action {
    @Override
    public ActionForward execute (
        ActionMapping mapping, ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
        //Conversao para o formulario definido
        EmpresaForm formulario = (EmpresaForm)form;

        //Recuperacao dos dados do formulario
        Empresa empresa = formulario.getEmpresa();
        empresa.setDataCriacao(formulario.getDataCriacaoSTR());
        //Envio para o banco de dados
        Connection connection =
            (Connection) request.getAttribute("connection");
        EmpresaDAO dao = new EmpresaDAO(connection);
        dao.cadastrar(empresa);
        //Retorno do forward que indica a proxima pagina
        return mapping.findForward("sucesso");
    }
}
```

# Atualização do struts-config

```
<struts-config>
  <form-beans>
    <form-bean name="EmpresaForm"
               type="br.fucapi.curso.jee.control.form.EmpresaForm" />
  </form-beans>
  <action-mappings>
    <action path="/teste"
            type="br.fucapi.curso.jee.control.action.ActionSimples">
      <forward name="sucesso" path="/exemplo.jsp"/>
    </action>
    <action path="/listarEmpresas"
            type="br.fucapi.curso.jee.control.action.ListarEmpresasAction">
      <forward name="listar" path="/listagem.jsp"/>
      <forward name="semRegistor" path="/listaVazia.html"/>
    </action>
    <action path="/adicionarEmpresa" name="EmpresaForm"
            type="br.fucapi.curso.jee.control.action.AdicionarEmpresaAction">
      <forward name="sucesso" path="/listarEmpresas.do"/>
    </action>
  </action-mappings>
</struts-config>
```

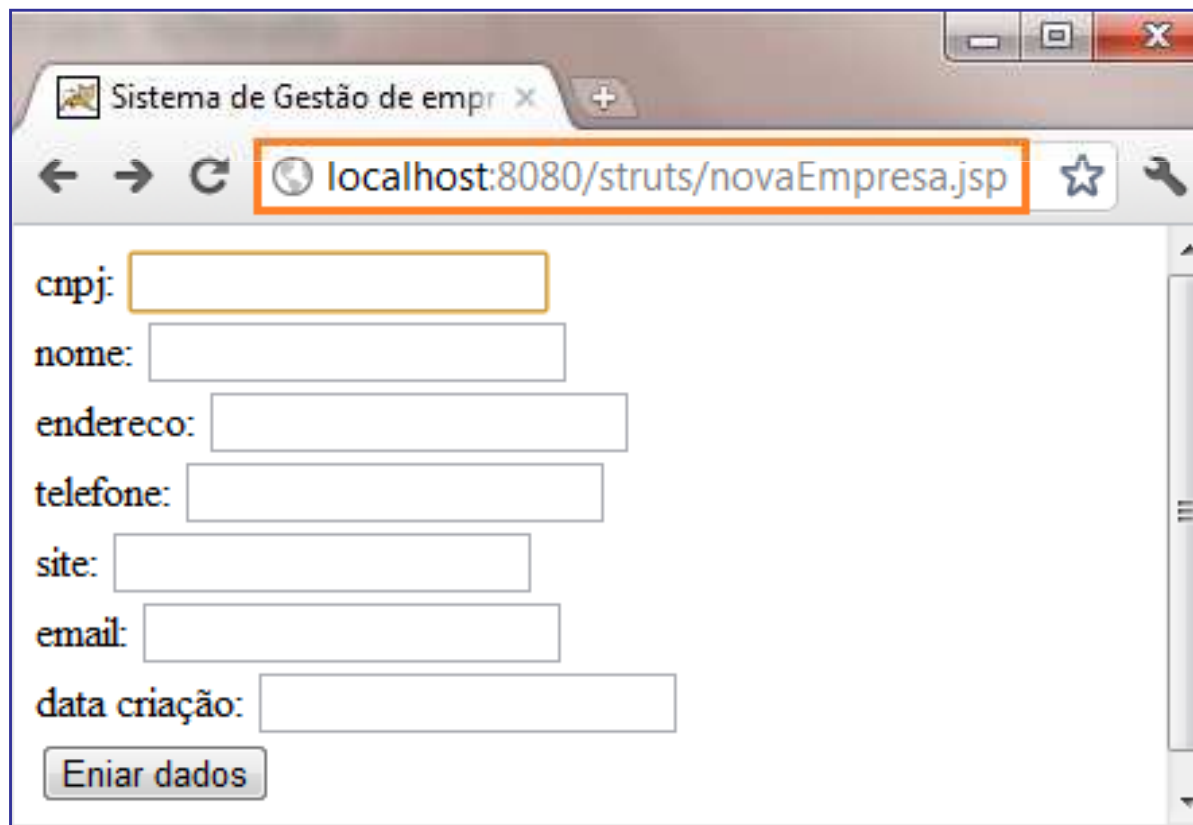


# Cadastro - novaEmpresa.jsp

```
<%@ taglib uri="http://struts.apache.org/tags-html" prefix="html" %>
<html:html>
  <head><title>Sistema de Gestão de empresas</title> </head>
  <html:errors/>
  <html:form action="/adicionarEmpresa" focus="empresa.cnpj">
    cnpj: <html:text property="empresa.cnpj"/><br>
    nome: <html:text property="empresa.razaoSocial"/><br>
    endereco: <html:text property="empresa.endereco"/><br>
    telefone: <html:text property="empresa.telefone"/><br>
    site: <html:text property="empresa.site"/><br>
    email: <html:text property="empresa.email"/><br>
    data criação: <html:text property="dataCriacaoSTR"/><br>
    <html:submit>Enviar dados</html:submit>
  </html:form>
</html:html>
```

# Página de cadastro

- <http://localhost:8080/struts/novaEmpresa.jsp>



A screenshot of a web browser window displaying a registration form. The browser's address bar shows the URL `localhost:8080/struts/novaEmpresa.jsp`, which is highlighted with an orange border. The page title is "Sistema de Gestão de empr". The form contains several input fields for company information: "cnpj:", "nome:", "endereco:", "telefone:", "site:", "email:", and "data criação:". Each field is represented by a text input box. At the bottom of the form is a button labeled "Eniar dados".



# Internacionalização e mensagens

- No struts, podemos centralizar as mensagens exibidas em apenas um arquivo de configuração, chamado `src/MessageResources.properties`
- Passos para a configuração:
- Criar o arquivo de configuração;
- Mapear o arquivo no struts-config;
- Utilizar as mensagens em páginas JSP;



# Arquivo de mensagens

---

- Na pasta SRC do seu projeto, crie o arquivo MessageResources.properties, com o texto a seguir:

```
1# comentario de um arquivo .properties
2menu.nome = Nome do menu
3menu.arquivo = Escolher Arquivo
4menu.editar = Editar Arquivo
5menu.sair = Sair da aplicação
6site.titulo = Sistema de teste do Struts
```

# Atualização do struts-config

```
<struts-config>
  <form-beans>
    <form-bean name="EmpresaForm"
               type="br.fucapi.curso.jee.control.form.EmpresaForm" />
  </form-beans>
  <action-mappings>
    <action path="/teste"
           type="br.fucapi.curso.jee.control.action.ActionSimples">
      <forward name="sucesso" path="/exemplo.jsp"/>
    </action>
    <action path="/listarEmpresas"
           type="br.fucapi.curso.jee.control.action.ListarEmpresasAction">
      <forward name="listar" path="/listagem.jsp"/>
      <forward name="semRegistror" path="/listaVazia.html"/>
    </action>
    <action path="/adicionarEmpresa" name="EmpresaForm"
           type="br.fucapi.curso.jee.control.action.AdicionarEmpresaAction">
      <forward name="sucesso" path="/listarEmpresas.do"/>
    </action>
  </action-mappings>
  <!-- Configuracao do arquivo de mensagens -->
  <message-resources parameter="MessageResources" />
</struts-config>
```



# Pagina testeMensagem.jsp

---

```
<%@ taglib uri="http://struts.apache.org/tags-bean" prefix="bean" %>
<html>
<head><title><bean:message key="site.titulo"/></title></head>
<body>
    <bean:message key="menu.nome"/><br>
    <bean:message key="menu.arquivo"/><br>
    <bean:message key="menu.editar"/><br>
    <bean:message key="menu.sair"/><br>
</body>
</html>
```

- **Acesse:**  
<http://localhost:8080/struts/testeMensagem.jsp>



# Validação de campos de um formulário

---

- Outro benefício que o struts apresenta é seu **framework de validação**, chamado de **Struts Validation**;
- Com esse framework, podemos capturar, de forma automática, problemas como a falta de preenchimento de um campo **requerido**;
- Utilizando o arquivo **MessageResources**, podemos devolver mensagens padrões, para cada tipo de erro;



## Passo necessários à validação

---

- Criação de um formbean que estenda a classe **ValidatorForm**;
- Inclusão do plugin de validação no arquivo struts-config.xml;
- Atualização do mapeamento da Action no struts config.xml;
- Criação dos arquivos validation.xml e validator-rules.xml;
- Atualizar o MessageResource.properties



# Passo 01 – ValidatorForm

```
package br.fucapi.curso.jee.control.form;
import org.apache.struts.validator.ValidatorForm;

import br.fucapi.curso.jee.model.bean.Empresa;

public class EmpresaForm extends ValidatorForm {
    private static final long serialVersionUID = 1L;
    //Atributos equivalentes aos campos de html:form
    private String dataCriacaoSTR;
    private Empresa empresa = new Empresa();
    public Empresa getEmpresa() {
        return empresa;
    }
    public String getDataCriacaoSTR() {
        return dataCriacaoSTR;
    }
    public void setDataCriacaoSTR(String dataCriacaoSTR) {
        this.dataCriacaoSTR = dataCriacaoSTR;
    }
}
```

# Passos 02 e 03: Atualizações do struts-config.xml

```
<!-- Outras ACTIONS aqui -->
<action path="/adicionarEmpresa" name="EmpresaForm"
        input="/novaEmpresa.jsp" validate="true"
        type="br.fucapi.curso.jee.control.action.AdicionarEmpresaAction">
    <forward name="sucesso" path="/listarEmpresas.do"/>
</action>
</action-mappings>
<!-- Configuracao do arquivo de mensagens -->
<message-resources parameter="MessageResources" />

<!-- Configuracao do Validator Plugin -->
<plug-in className="org.apache.struts.validator.ValidatorPlugIn">
    <set-property
        property="pathnames"
        value="/WEB-INF/validator-rules.xml,/WEB-INF/validation.xml"/>
</plug-in>
</struts-config>
```



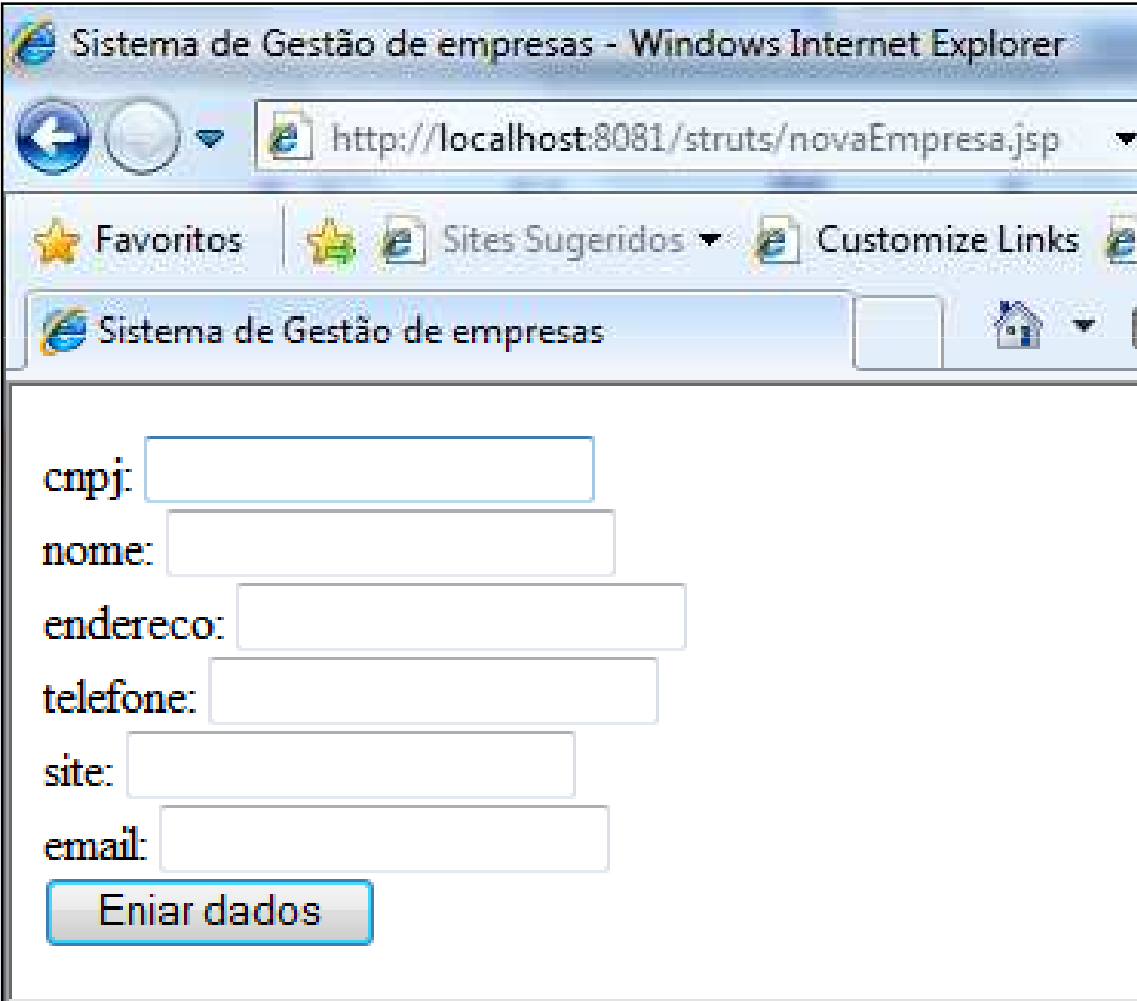
# Passo 04 – Criação do arquivo validation.xml

```
<form-validation>
<global>
  <constant>
    <constant-name>telephoneFormat</constant-name>
    <constant-value>^\d{5,10}$</constant-value>
  </constant>
</global>
<formset>
  <form name="EmpresaForm">
    <field property="empresa.cnpj" depends="required">
      <arg key="label.CNPJ" />
    </field>
    <field property="empresa.razaoSocial" depends="required">
      <arg key="label.razaoSocial" />
    </field>
  </form>
</formset>
</form-validation>
```

# Atualização do arquivos de mensagens

```
1# comentario de um arquivo .properties
2menu.nome = Nome do menu
3menu.arquivo = Escolher Arquivo
4menu.editar = Editar Arquivo
5menu.sair = Sair da aplicação
6site.titulo = Sistema de teste do Struts
7
8#Lista de titulos de campos
9label.CNPJ = CNPJ
10label.razaoSocial= Razão Social
11
12# general error msgs
13errors.header=<font size="5"><UL>
14errors.prefix=<LI><span style="color: red">
15errors.suffix=</span></LI>
16errors.footer=</UL></font>
17errors.required=Campo "{0}" não informado.
```

[http://localhost:8080/struts/  
novaEmpresa.jsp](http://localhost:8080/struts/novaEmpresa.jsp)



Sistema de Gestão de empresas - Windows Internet Explorer

http://localhost:8081/struts/novaEmpresa.jsp

Favoritos Sites Sugeridos Customize Links

Sistema de Gestão de empresas

cnpj:

nome:

endereço:

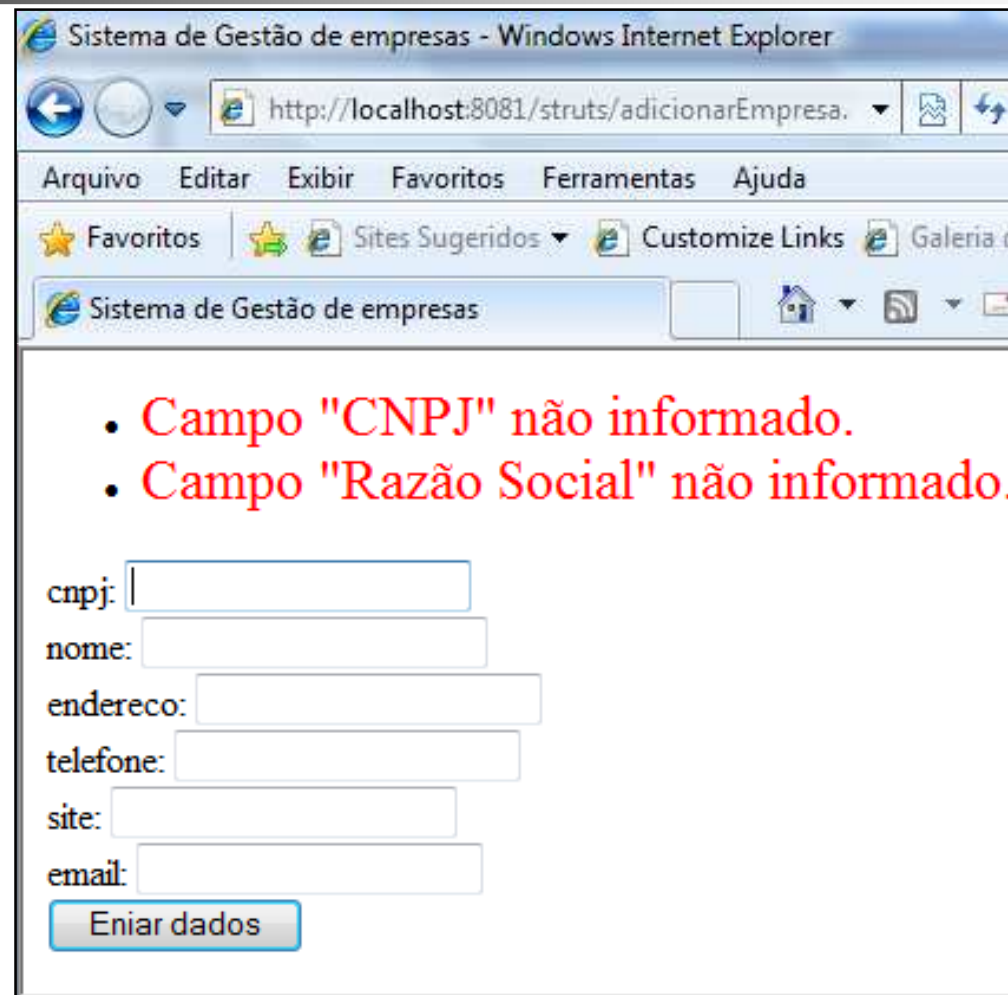
telefone:

site:

email:

Enviar dados

# Resultado é exibido na mesma página



Sistema de Gestão de empresas - Windows Internet Explorer

http://localhost:8081/struts/adicionarEmpresa.

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Favoritos Sites Sugeridos Customize Links Galeria d

Sistema de Gestão de empresas

- Campo "CNPJ" não informado.
- Campo "Razão Social" não informado.

cnpj:

nome:

endereço:

telefone:

site:

email:

Enviar dados



## O que vem a seguir?

---

- Revisão da camada de modelo;
- JDBC, Factory e DAO;
- Persistência com Hibernate;



# Referências

---

- [www.caelum.com.br](http://www.caelum.com.br)
- Hall, Marty, "Core Servlets and Java Server Pages", Janeiro 2002, Sun Microsystems Press;
- <http://java.sun.com/j2ee/1.6/docs/tutorial/doc/index.html>
- <http://java.sun.com/products/jndi/docs.html>
- <http://java.sun.com/blueprints/corej2eepatterns/Patterns/index.html>

# Java Enterprise Edition - JEE

---

## 11. Struts Framework



Esp. Márcio Palheta  
gtalk: [marcio.palheta@gmail.com](mailto:marcio.palheta@gmail.com)