


A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light mint green. They are positioned diagonally, with the blue one partially covering the green one.

Stripes

Delivoro Sistemas



Introdução

- Framework MVC2 para Java
- Integra com o Spring
- Se baseia em Actions e uma Taglib Própria (formulários e layout)
- *Code by convention* e Annotations
- Funcionalidades
 - Atribuição automática de Parâmetros da Request a objetos Java
 - Conversões automática de dados (numéricos, datas, horas, etc..)
 - Validação de dados
 - Exibição de mensagens de erro
 - Globais ou atreladas a campos de forms específicos
 - Exibição de mensagens de sucesso
 - Internacionalização



O básico

- ActionBeans e Event Handlers
- URL Binding
- Validação
- Conversão de tipos e Formatadores
- Tags JSP e Layout



ActionBeans

- Todas as requisições recebidas via HTTP (ex: de browsers) passam pelo Stripes Controller
- ActionBeans são objetos criados pelo Controller do Stripes para atender a solicitações dos usuários
- A cada click de mouse ou a cada submit de formulário faz com que um novo objeto ActionBean seja criado para atender aquela requisição vinda da WEB
- ActionBeans devem implementar a interface ActionBean do Stripes
- Uma única ActionBean pode lidar com diversos eventos vindos do navegador

```
public interface ActionBean {  
    public ActionBeanContext getContext();  
    public void setContext(ActionBeanContext context);  
}
```

ActionBeans - Exemplo - Visão Geral

- Action para lidar com o cadastro de bancos
- Para funcionar precisa de uma JSP e um arquivo .properties
- Possui 2 métodos *Event Handlers*
 - Um para exibir o formulário de cadastro para o usuário
 - Um para receber os dados digitados no formulário de cadastro e salvar no banco de dados
- As operações de consulta, remoção, edição e listagem ficariam na mesma classe
 - Bastaria adicionar um Event Handler a mais para cada um deles

```
@UrlBinding("/manut/banco.action") 1
public class BancoActionBean implements ActionBean { 2

    private ActionBeanContext context; 3

    @SpringBean 4
    private BancoMapper bancoMapper;

    @ValidateNestedProperties ({ 5
        @Validate(field="codigo", required=true, maxLength=10, on="salvar"),
        @Validate(field="nome", required=true, maxLength=100, on="salvar")
    })

    private Banco banco;

    @DefaultHandler 6
    public Resolution prepararInserir() { 7
        return new ForwardResolution( path: "/WEB-INF/jsp/editarBanco.jsp"); 8
    }

    public Resolution salvar() throws DadosDuplicadosException {
        bancoMapper.inserirBanco(banco);
        context.getMessages().add(new LocalizableMessage( messageKey: "message.inserted")); 9
        return new RedirectResolution (BancoActionBean.class); 10
    }
}
```

JSP de Conteúdo - Exemplo - Visão Geral

- JSP lida com a interface com o usuário
- Não deve ter código
 - Ou apenas o mínimo necessário
 - Se tiver precisa ser em EL e JSTL
- Stripes facilita a criação de JSPs
 - Utilização de layouts padronizados
 - Lida com a cópia de dados de campos entre JSP e ActionBean
 - Exibição de mensagens informativas
 - Exibição de mensagens de erros

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-render name="/WEB-INF/jsp/layout.jsp"> 3
  <s:layout-component name="body"> 4
    <h2 id="page-heading"><fmt:message key="label.manutbanco.titulo"/></h2> 5*
    <s:form beanClass="br.com.gourmex.admweb.action.BancoActionBean" method="post" focus="banco.codigo"> 6
      <s:hidden name="banco.idBanco"/> 7
      <fieldset>
        <legend><fmt:message key="label.manutbanco.legenda"/></legend>
        <p>
          <s:label for="banco.codigo"/> * 8
          <s:text name="banco.codigo" size="5" maxlength="10"/> 9
          <s:errors field="banco.codigo"/> 10
        </p>
        <p>
          <s:label for="banco.nome"/> *
          <s:text name="banco.nome" size="25" maxlength="100"/>
          <s:errors field="banco.nome"/>
        </p>
        <s:submit name="salvar" class="register-button"><fmt:message key="label.submit"/></s:submit> 11
        <s:submit name="listar" class="register-button"><fmt:message key="label.back"/></s:submit>
      </fieldset>
    </s:form>
  </s:layout-component>
</s:layout-render>
```

StripesResources.properties - Exemplo - Visão Geral

Onde são salvas:

- Configurações de elementos visuais do Stripes
- Mensagens de Sucesso
- Mensagens de Erro
- Labels de campos (internacionalização)

```
StripesResources.properties x mybatis.properties x build.gradle (gm:GatewayClient) x build.gradle (sistema) x settings.gradle (sistema)

11
12 # Resource strings used by the stripes:errors tag when there are no nested tags
13 stripes.errors.header=<div class="alert alert-danger"> <strong>Verifique os erros abaixo:</strong><but
14 stripes.errors.beforeError=
15 stripes.errors.afterError=<br />
16 stripes.errors.footer=</div>
17
18 stripes.fieldErrors.header=
19 stripes.fieldErrors.beforeError=<span class="error-stripes">
20 stripes.fieldErrors.afterError=</span><br />
21 stripes.fieldErrors.footer=
22 global.validationerror.fielderrors=Verifique os campos abaixo.
23
24 # Resource strings used by the stripes:messages tag
25 stripes.messages.header=<div class="alert alert-warning"><button type="button" class="close" data-dis
26 stripes.messages.beforeMessage=
27 stripes.messages.afterMessage=
28 stripes.messages.footer=</div>
29
30 # Resource strings used by the stripes:messages tag
31 stripes.messages.header=<div class="alert alert-success"><strong>Sucesso!</strong><button type="button
32 stripes.messages.beforeMessage=<li>
33 stripes.messages.afterMessage=</li>
34 stripes.messages.footer=</ul></div>
35
```

JSP de Layout - Exemplo - Visão Geral

Define a estrutura de todas as páginas do sistema

- Inclui JSP que renderizará o topo
- Inclui JSP que renderizará o menu
- Conteúdo (body)
 - Onde as páginas internas do sistema serão exibidas
- Footer
- Mensagens de Sucesso
- Mensagens de Erro
- Labels de campos (internacionalização)

```
<%@page pageEncoding="UTF-8" contentType="text/html; charset=UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-definition> 3
  <fmt:bundle basename="StripesResources"> 4
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title><fmt:message key="label.topbanner.title"/*>/title> 5*
      </head>
      <body>
        <div class="container_16">
          <s:layout-component name="top"> 6
            <jsp:include page="/incl/topbanner.jsp"/> 7*
          </s:layout-component>
          <s:layout-component name="menu">
            <jsp:include page="/incl/navigator.jsp"/>
          </s:layout-component>
          <strong><s:errors globalErrorsOnly="true"/></strong> 8
          <strong><s:messages/></strong> 9
          <s:layout-component name="body" /> 10
          <s:layout-component name="footer">
            <jsp:include page="/WEB-INF/jsp/footer.jsp"/>
          </s:layout-component>
        </div>
      </body>
    </html>
  </fmt:bundle>
</s:layout-definition>
```


JSP Include - Exemplo - Visão Geral

Define uma parte/pedaco de todas as páginas do sistema

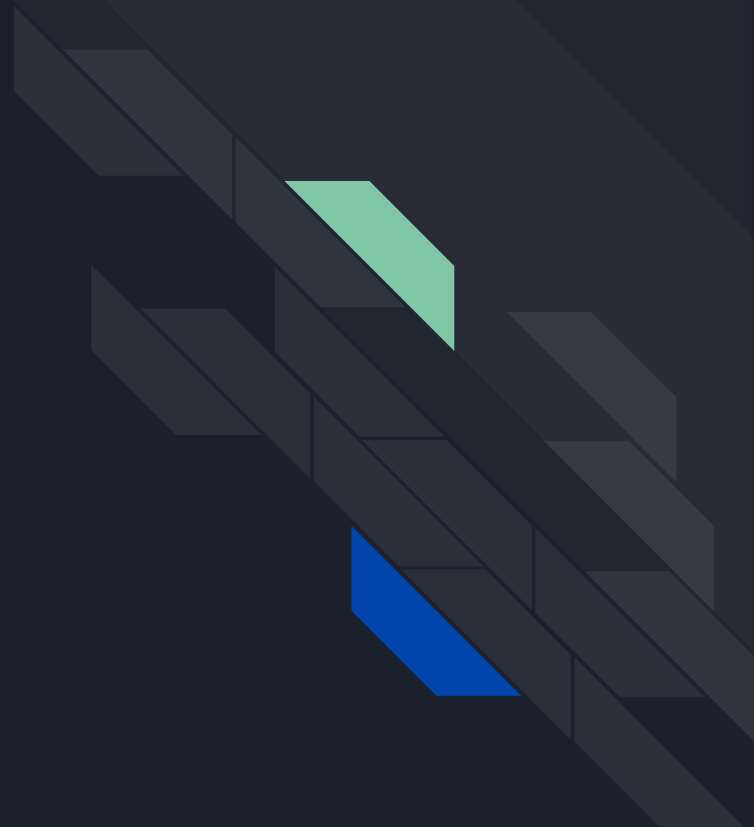
- Nesse caso é o topo de todas as páginas do sistema
- Usa Expression Language para exibir
 - Dados obtidos da Request
 - Dados obtidos da Action
- Usa tags JSTL para
 - Lógica básica de exibição
 - Exibição de mensagens internacionalizadas

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %> 1*
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*

<div class="grid_4" style="margin-top: 10px;" >
    <a href="/">
        " />
    </a>
</div>

<div class="grid_12" style="margin-top: 25px;" >
    <c:if test="${sessionScope.SESSION_CONTEXT != null}" > 3*
        <strong>
            <fmt:message key="topbanner.usuario" />
            <c:out value="${actionBean.gmxContext.usuario.nome}" /> 4*
            Seu IP: <c:out value="${pageContext.request.remoteAddr}" /></strong> 5*
        </c:if>
    </div>
```

ActionBean



ActionBeans - Detalhamento

1. Define a URL para acessar essa action
2. Toda Action precisa implementar essa interface
 - Para o Stripes enxergar e usar a Action
3. Injetado pelo Stripes para action poder ter acesso à request e response
4. Integração entre Stripes e Spring
 - Spring injeta objeto que irá interagir com BD
5. Validação de dados antes de cadastrar
 - Se falhar na validação, mensagens de erro são exibidas na JSP por tags s:error
6. Event handler padrão da Action
 - Ponto de entrada caso um evento não seja informado pelo navegador

```
@UrlBinding("/manut/banco.action") 1
public class BancoActionBean implements ActionBean { 2

    private ActionBeanContext context; 3

    @SpringBean 4
    private BancoMapper bancoMapper;

    @ValidateNestedProperties ({ 5
        @Validate(field="codigo", required=true, maxLength=10, on="salvar"),
        @Validate(field="nome", required=true, maxLength=100, on="salvar")
    })

    private Banco banco;

    @DefaultHandler 6
    public Resolution prepararInserir() { 7
        return new ForwardResolution( path: "/WEB-INF/jsp/editarBanco.jsp"); 8
    }

    public Resolution salvar() throws DadosDuplicadosException {
        bancoMapper.inserirBanco(banco);
        context.getMessages().add(new LocalizableMessage( messageKey: "message.inserted")); 9
        return new RedirectResolution (BancoActionBean.class); 10
    }
}
```

ActionBeans - Detalhamento

7. Todo Evento Handler deve ser público e retornar Resolution
 - Para o Stripes saber qual JSP ou Action será invocada depois
8. Encaminhamento de continuidade
 - Normalmente para uma JSP
 - Torna variáveis da Action visíveis para JSP
 - Continua o processamento iniciado pela Action antes de retornar para navegador
 - Quando a JSP terminar, é retornado o conteúdo HTML a ser exibido e o código HTTP 200 (sucesso)
 - É o mais comum

```
@UrlBinding("/manut/banco.action") 1
public class BancoActionBean implements ActionBean { 2

    private ActionBeanContext context; 3

    @SpringBean 4
    private BancoMapper bancoMapper;

    @ValidateNestedProperties ({ 5
        @Validate(field="codigo", required=true, maxLength=10, on="salvar"),
        @Validate(field="nome", required=true, maxLength=100, on="salvar")
    })
    private Banco banco;

    @DefaultHandler 6
    public Resolution prepararInserir() { 7
        return new ForwardResolution( path: "/WEB-INF/jsp/editarBanco.jsp"); 8
    }

    public Resolution salvar() throws DadosDuplicadosException {
        bancoMapper.inserirBanco(banco);
        context.getMessages().add(new LocalizableMessage( messageKey: "message.inserted")); 9
        return new RedirectResolution (BancoActionBean.class); 10
    }
}
```

ActionBeans - Detalhamento

9. Adiciona mensagem a ser exibida pela JSP
 - Mensagem será consultada no StripesResources.properties
 - Adequado para internacionalização
10. Encaminhamento de reinício
 - Retorna código HTTP 308 para navegador
 - Navegador irá solicitar página indicada pelo Redirect
 - Evita efeitos colaterais do refresh de página após um cadastro

```
@UrlBinding("/manut/banco.action") 1
public class BancoActionBean implements ActionBean { 2

    private ActionBeanContext context; 3

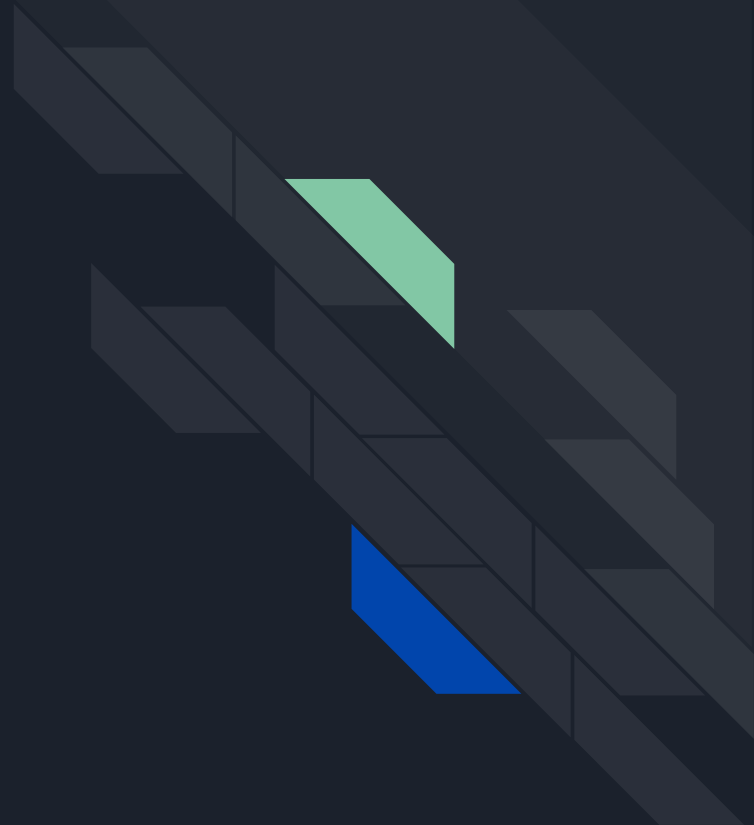
    @SpringBean 4
    private BancoMapper bancoMapper;

    @ValidateNestedProperties ({ 5
        @Validate(field="codigo", required=true, maxLength=10, on="salvar"),
        @Validate(field="nome", required=true, maxLength=100, on="salvar")
    })
    private Banco banco;

    @DefaultHandler 6
    public Resolution prepararInserir() { 7
        return new ForwardResolution( path: "/WEB-INF/jsp/editarBanco.jsp"); 8
    }

    public Resolution salvar() throws DadosDuplicadosException {
        bancoMapper.inserirBanco(banco);
        context.getMessages().add(new LocalizableMessage( messageKey: "message.inserted")); 9
        return new RedirectResolution (BancoActionBean.class); 10
    }
}
```

JSP de Conteúdo



JSP de Conteúdo - Detalhamento

1. Inclui Taglib do Stripes - prefixo s
2. Inclui Taglib fmt
 - Nesse caso sendo usada para internacionalização
3. Herda Layout de layout.jsp
 - Note a ausência de código HTML contendo
 - Includes de CSS e Javascript
 - Topo da página
 - Menu da aplicação
 - Rodapé
4. Sobrescreve component body do layout
 - É onde deve ir o conteúdo de cada página da aplicação
 - O layout não define essa parte

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-render name="/WEB-INF/jsp/layout.jsp"> 3
  <s:layout-component name="body"> 4
    <h2 id="page-heading"><fmt:message key="label.manutbanco.titulo"/></h2> 5*
    <s:form beanClass="br.com.gourmex.admweb.action.BancoActionBean" method="post" focus="banco.codigo"> 6
      <s:hidden name="banco.idBanco"/> 7
      <fieldset>
        <legend><fmt:message key="label.manutbanco.legenda"/></legend>
        <p>
          <s:label for="banco.codigo"/> * 8
          <s:text name="banco.codigo" size="5" maxLength="10"/> 9
          <s:errors field="banco.codigo"/> 10
        </p>
        <p>
          <s:label for="banco.nome"/> *
          <s:text name="banco.nome" size="25" maxLength="100"/>
          <s:errors field="banco.nome"/>
        </p>
        <s:submit name="salvar" class="register-button"><fmt:message key="label.submit"/></s:submit> 11
        <s:submit name="listar" class="register-button"><fmt:message key="label.back"/></s:submit>
      </fieldset>
    </s:form>
  </s:layout-component>
</s:layout-render>
```

JSP de Conteúdo - Detalhamento

5. Exibe título da página usando mensagem internacionalizada
 - Será consultada em StripesResources.properties
6. Tag do Stripes para renderizar formulário
 - Ao invés de colocar URL coloca-se o nome da Action que irá ser chamada
 - focus é útil para colocar o cursor no campo desejado quando o form for exibido
7. Tag do Stripes para renderizar campo hidden (oculto)
 - Caso o atributo código do objeto Banco definido na Action possua um valor ele será automaticamente preenchido

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-render name="/WEB-INF/jsp/layout.jsp"> 3
    <s:layout-component name="body"> 4
        <h2 id="page-heading"><fmt:message key="label.manutbanco.titulo"/></h2> 5*
        <s:form beanClass="br.com.gourmex.admweb.action.BancoActionBean" method="post" focus="banco.codigo"> 6
            <s:hidden name="banco.idBanco"/> 7
            <fieldset>
                <legend><fmt:message key="label.manutbanco.legenda"/></legend>
                <p>
                    <s:label for="banco.codigo"/> * 8
                    <s:text name="banco.codigo" size="5" maxLength="10"/> 9
                    <s:errors field="banco.codigo"/> 10
                </p>
                <p>
                    <s:label for="banco.nome"/> *
                    <s:text name="banco.nome" size="25" maxLength="100"/>
                    <s:errors field="banco.nome"/>
                </p>
                <s:submit name="salvar" class="register-button"><fmt:message key="label.submit"/></s:submit> 11
                <s:submit name="listar" class="register-button"><fmt:message key="label.back"/></s:submit>
            </fieldset>
        </s:form>
    </s:layout-component>
</s:layout-render>
```


JSP de Conteúdo - Detalhamento

8. Tag do Stripes para exibir label de um campo

- Será consultado em `StripesResources.properties`

9. Tag do Stripes para renderizar campo texto

- Caso o atributo nome do objeto Banco definido na Action possua um valor, ele será automaticamente preenchido e exibido para o usuário

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-render name="/WEB-INF/jsp/layout.jsp"> 3
  <s:layout-component name="body"> 4
    <h2 id="page-heading"><fmt:message key="label.manutbanco.titulo"/></h2> 5*
    <s:form beanClass="br.com.gourmex.admweb.action.BancoActionBean" method="post" focus="banco.codigo"> 6
      <s:hidden name="banco.idBanco"/> 7
      <fieldset>
        <legend><fmt:message key="label.manutbanco.legenda"/></legend>
        <p>
          <s:label for="banco.codigo"/> * 8
          <s:text name="banco.codigo" size="5" maxLength="10"/> 9
          <s:errors field="banco.codigo"/> 10
        </p>
        <p>
          <s:label for="banco.nome"/> *
          <s:text name="banco.nome" size="25" maxLength="100"/>
          <s:errors field="banco.nome"/>
        </p>
        <s:submit name="salvar" class="register-button"><fmt:message key="label.submit"/></s:submit> 11
        <s:submit name="listar" class="register-button"><fmt:message key="label.back"/></s:submit>
      </fieldset>
    </s:form>
  </s:layout-component>
</s:layout-render>
```

JSP de Conteúdo - Detalhamento

10. Tag do Stripes para definir onde serão exibidas mensagens de erro do campo "banco.nome"

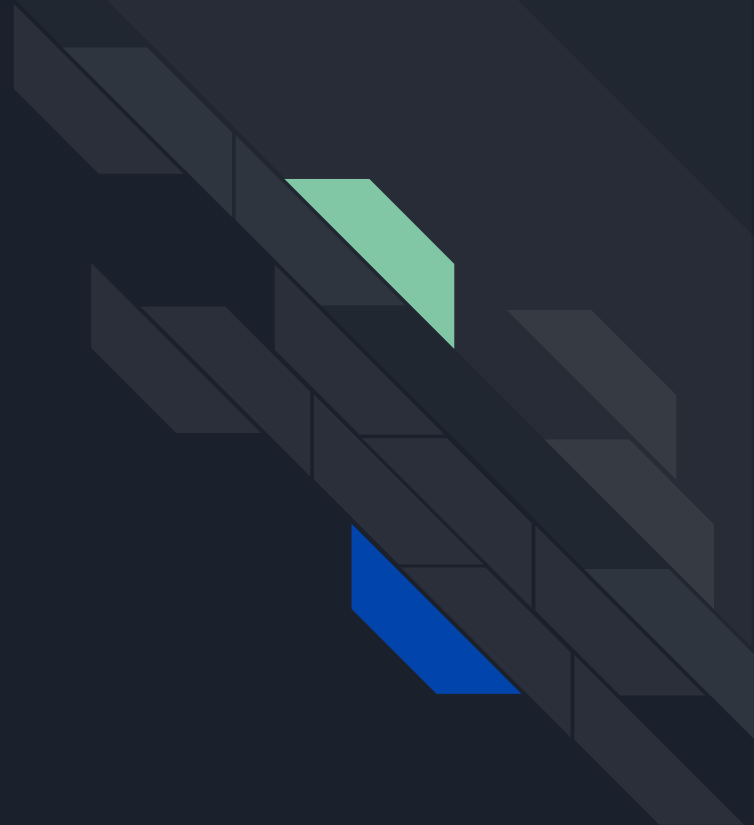
- Geralmente geradas pelo validador do Stripes

11. Tag do Stripes para gerar o botão submit do formulário

- Quando clicado irá invocar a ActionBean
- O campo name define qual Event Handler da Action será invocado
 - Nessa caso será 'salvar'
 - Dê atenção especial a isso!

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-render name="/WEB-INF/jsp/layout.jsp"> 3
  <s:layout-component name="body"> 4
    <h2 id="page-heading"><fmt:message key="label.manutbanco.titulo"/></h2> 5*
    <s:form beanClass="br.com.gourmex.admweb.action.BancoActionBean" method="post" focus="banco.codigo"> 6
      <s:hidden name="banco.idBanco"/> 7
      <fieldset>
        <legend><fmt:message key="label.manutbanco.legenda"/></legend>
        <p>
          <s:label for="banco.codigo"/> * 8
          <s:text name="banco.codigo" size="5" maxLength="10"/> 9
          <s:errors field="banco.codigo"/> 10
        </p>
        <p>
          <s:label for="banco.nome"/> *
          <s:text name="banco.nome" size="25" maxLength="100"/>
          <s:errors field="banco.nome"/>
        </p>
        <s:submit name="salvar" class="register-button"><fmt:message key="label.submit"/></s:submit> 11
        <s:submit name="listar" class="register-button"><fmt:message key="label.back"/></s:submit>
      </fieldset>
    </s:form>
  </s:layout-component>
</s:layout-render>
```

JSP de Layout



JSP de Layout - Detalhamento

1. Importação da Taglib do Stripes
2. Importação da Taglib fmt da JSTL
3. Definição do Layout
4. Importação do arquivo properties
 - a. Onde as mensagens do fmt:message estão armazenadas
5. Exibição de mensagem i18n usando fmt:message
6. Definição de componente do layout
 - a. Definir como um componente permite que JSP que herda o layout altere essa parte do layout
7. Inclusão da JSP topbanner

```
<%@page pageEncoding="UTF-8" contentType="text/html; charset=UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-definition> 3
  <fmt:bundle basename="StripesResources"> 4
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title><fmt:message key="label.topbanner.title"/>*> 5*
      </head>
      <body>
        <div class="container_16">
          <s:layout-component name="top"> 6
            <jsp:include page="/incl/topbanner.jsp"/> 7*
          </s:layout-component>
          <s:layout-component name="menu">
            <jsp:include page="/incl/navigator.jsp"/>
          </s:layout-component>
          <strong><s:errors globalErrorsOnly="true"/></strong> 8
          <strong><s:messages/></strong> 9
          <s:layout-component name="body" /> 10
          <s:layout-component name="footer">
            <jsp:include page="/WEB-INF/jsp/footer.jsp"/>
          </s:layout-component>
        </div>
      </body>
    </html>
  </fmt:bundle>
</s:layout-definition>
```

JSP de Layout - Detalhamento

8. Tag do Stripes para exibir erros

- Define onde no layout as mensagens de erro serão exibidas
- Nesse caso exibirá apenas erros globais
- Erros globais são aqueles não relacionados a campos de um form

9. Tag do Stripes para exibir mensagens

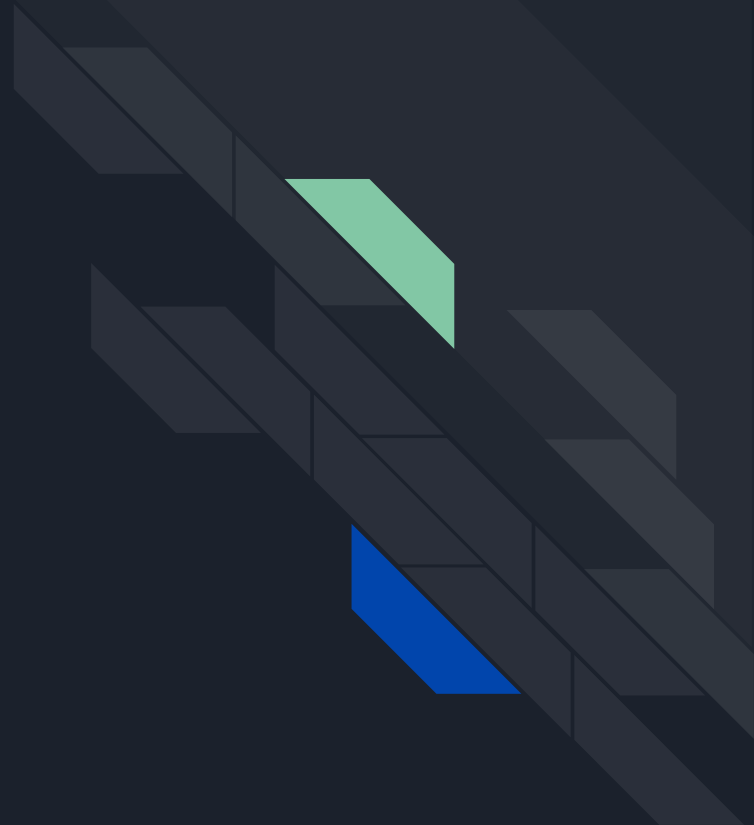
- Define onde no layout as mensagens de sucesso ou avisos serão exibidas

10. Define componente chamado body porém sem nenhum conteúdo

- Deverá ser definido nas páginas filhas para exibição de conteúdo dinâmico da aplicação

```
<%@page pageEncoding="UTF-8" contentType="text/html; charset=UTF-8" %>
<%@ taglib prefix="s" uri="http://stripes.sourceforge.net/stripes-dynattr.tld" %> 1
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*
<s:layout-definition> 3
  <fmt:bundle basename="StripesResources"> 4
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title><fmt:message key="label.topbanner.title"/>*</title> 5*
      </head>
      <body>
        <div class="container_16">
          <s:layout-component name="top"> 6
            <jsp:include page="/incl/topbanner.jsp"/> 7*
          </s:layout-component>
          <s:layout-component name="menu">
            <jsp:include page="/incl/navigator.jsp"/>
          </s:layout-component>
          <strong><s:errors globalErrorsOnly="true"/></strong> 8
          <strong><s:messages/></strong> 9
          <s:layout-component name="body" /> 10
          <s:layout-component name="footer">
            <jsp:include page="/WEB-INF/jsp/footer.jsp"/>
          </s:layout-component>
        </div>
      </body>
    </html>
  </fmt:bundle>
</s:layout-definition>
```

JSP de Include



JSP Include - Exemplo - Visão Geral

1. Importação da Taglib c da JSTL
2. Importação da Taglib fmt da JSTL
3. Usando comando if da JSTL
4. Exibindo informação armazenada em objeto da ActionBean
5. Exibindo informação armazenada na requisição do Servlet Container

```
<%@page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %> 1*
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %> 2*

<div class="grid_4" style="margin-top: 10px;" >
    <a href="/">
        " />
    </a>
</div>

<div class="grid_12" style="margin-top: 25px;" >
    <c:if test="${sessionScope.SESSION_CONTEXT != null}" > 3*
        <strong>
            <fmt:message key="topbanner.usuario" />
            <c:out value="${actionBean.gmxContext.usuario.nome}" /> 4*
            Seu IP: <c:out value="${pageContext.request.remoteAddr}" /></strong> 5*
        </c:if>
    </div>
```



Resumo

- Stripes é o nosso framework MVC
- Contém lógica de navegação e lida com recebimento de dados do usuário e exibição de dados do usuário
- Não pode conter lógica de negócio - apenas de 'tela'
 - Interfaces com o usuário mudam frequentemente
 - Lógica de negócio muda menos frequentemente
 - Se eu trocar a interface, eu não preciso reescrever lógica de negócio