

**Iniciativa Open4Education**

# Apresentação da Agenda



- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

## Motivação

- MVC tornou-se um padrão de mercado;
- As interfaces gráficas exigidas são muito complexas para serem desenvolvidas somente com HTML exigindo muito JavaScript;
- Muitos componentes de UI sendo desenvolvidos com Custom Tags ou JavaScript sem padronização;
- Baixa produtividade no desenvolvimento de aplicações Web;

## O que é JavaServer Faces (JSF)?

- Paradigma de programação visual de User-interfaces aplicado à web
- É um framework que permite a criação de aplicações Web com semântica de Swing implementando MVC;
- “Toolability = Ferramentabilidade” ;
- É Uma especificação J2EE – JSR 127;

**J2EE Web Container**

**J2EE EJB  
Container**

## JSF e J2EE

**JavaServer Faces**

**Controller**

**Servlet**

**View**

**Java  
Server  
Pages**

**Custom  
Tag**

**Model**

**JavaBeans**

**EJB**

## Quais os próximos objetivos?

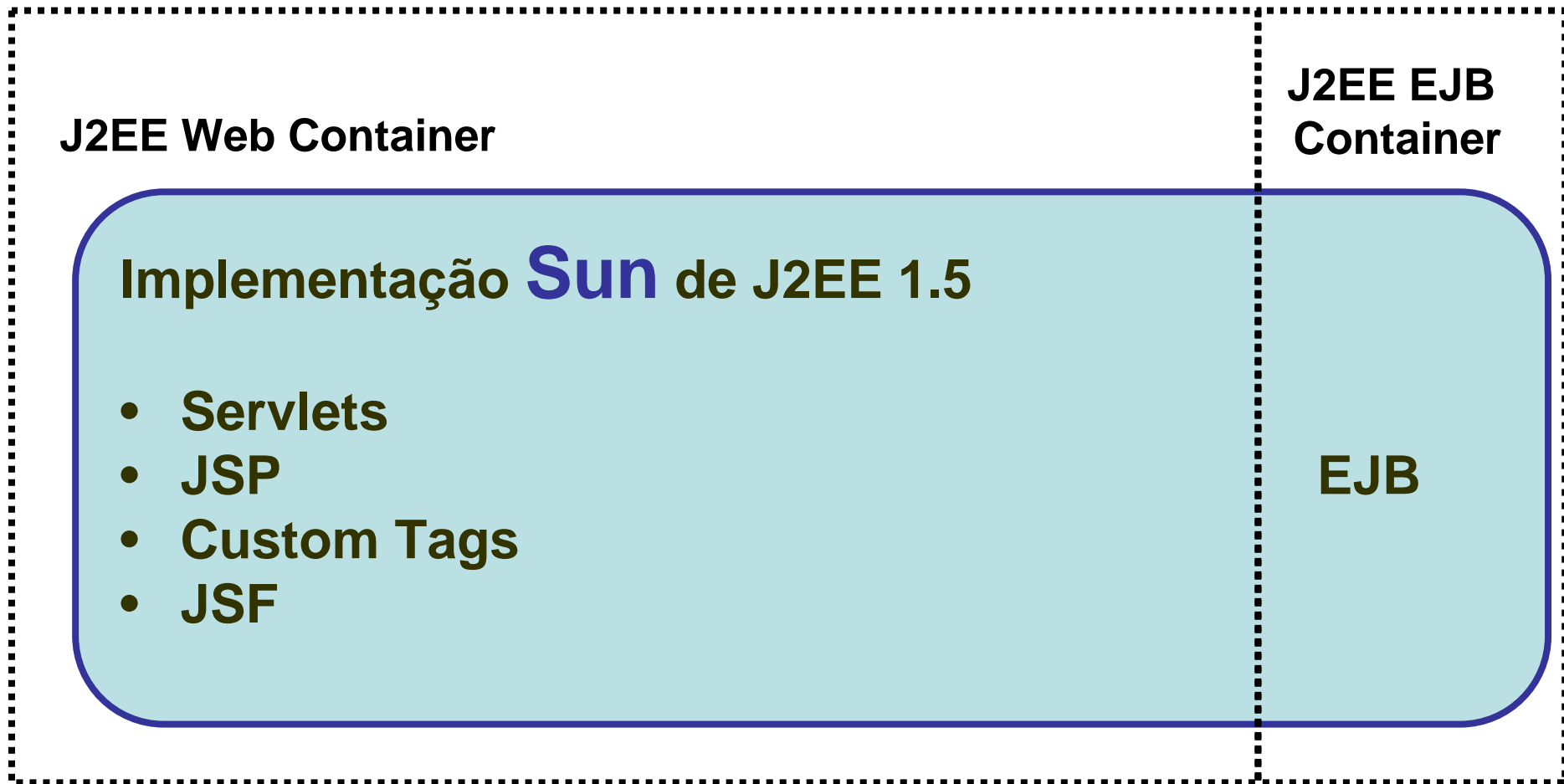
- Todo Web Container compatível com a especificação Java EE 5.0 implementa JSF;
- Criação de componentes de UI compatíveis com JSF por terceiros;
- Suporte a JSF na maioria dos IDEs
- Aumento da produtividade de aplicação J2EE

## OC4J - Oracle





## Sun Application Server



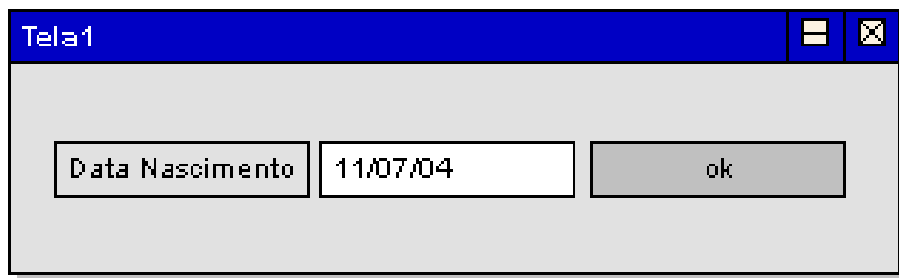
- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

- **Arquitetura Client-Server baseada em HTTP;**
- **Dificuldade em prover o mesmo dinamismo de uma aplicação desktop;**
- **Todo o processamento Java acontece no servidor;**

**GUI e Listener  
são processados pela  
mesma máquina**

## Listener

- Validação
- Conversão de dados
- Integração com a camada Model
- Lógica de negócios, etc...

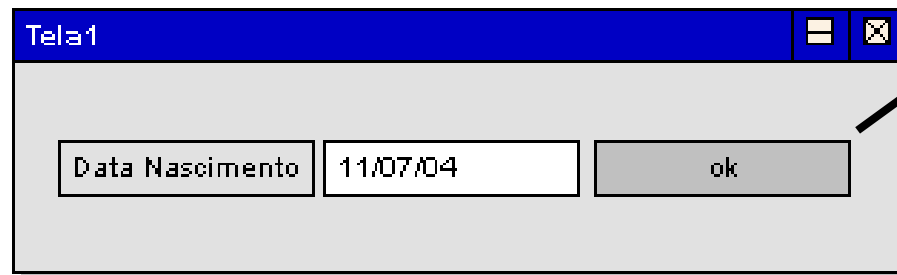


**Java  
Chamada a métodos**

## J2EE Web Container

### Java Servlet

- Validação
- Conversão de dados
- Integração com a camada Model
- Lógica de negócios, etc...



Tela1

Data Nascimento 11/07/04 ok

HTTP  
REQUEST / RESPONSE

**Cliente – Browser: HTML + JavaScript**

## 1. Restore View

- Criação da árvore de componentes

## 2. Apply Request Values

- Atualiza árvore de componentes com

## 3. Process Validators

- Executa todas

## 4. Update Model Values

- Execução do ActionListener padrão, geralmente com

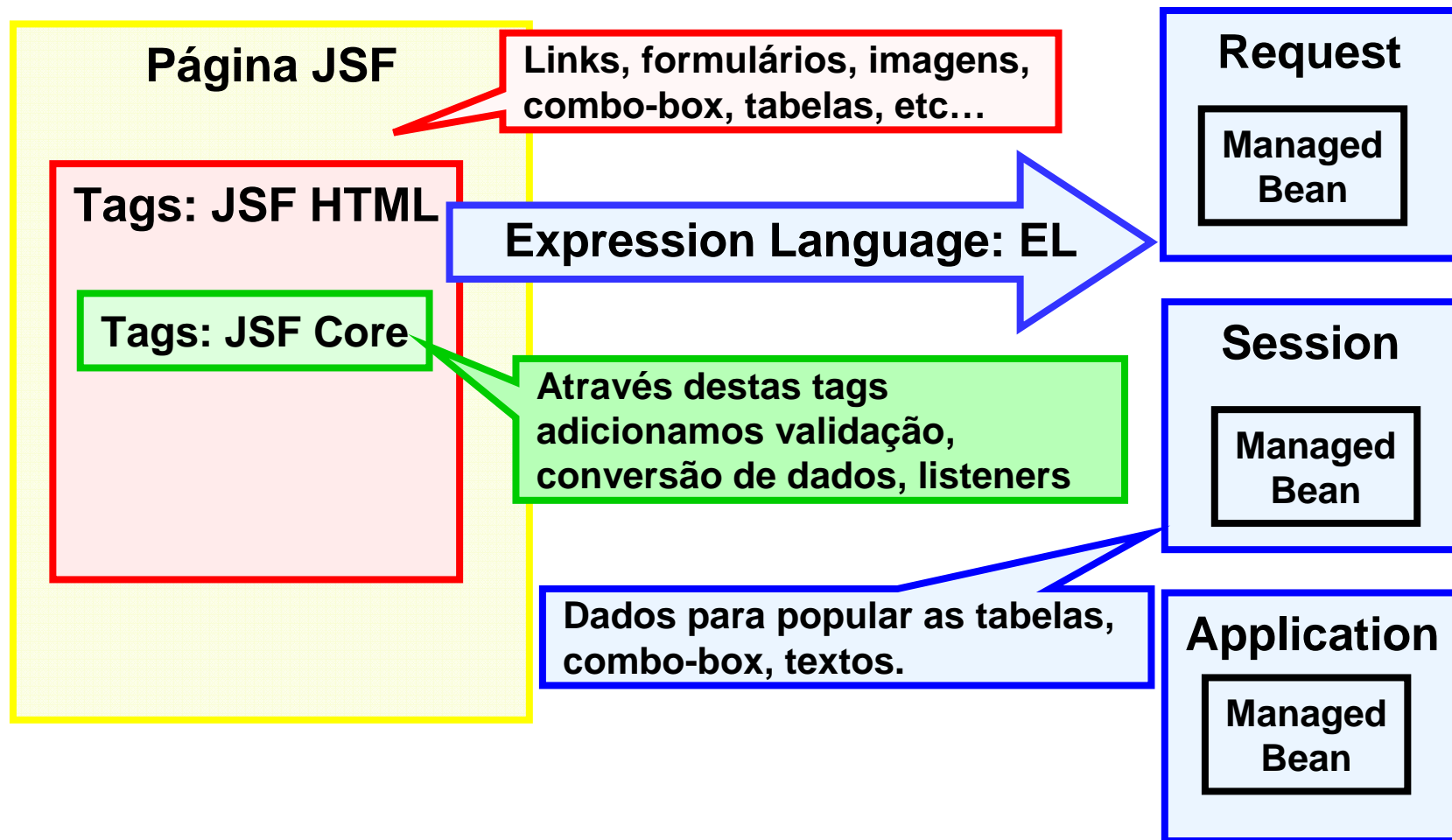
## 5. Invoke Application

## 6. Render Response

- Geração da Response com a árvore de componentes

- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

# Criação de páginas JSF





- **Bibliotecas de Tags**
- **Managed Beans**
- **Componentes de User-interface (UI)**
- **Validadores**
- **Conversores**
- **Eventos**
- **Listeners**
- **Renderizadores**
- **FacesController**

- **Páginas JSF geralmente utilizam duas bibliotecas de Tags:**
  - **JSF Html: renderização de componentes HTML**
  - **JSF Core: integração dos componentes de UI com validadores, conversores**

## A navegação em páginas HTML pode acontecer das seguintes formas:

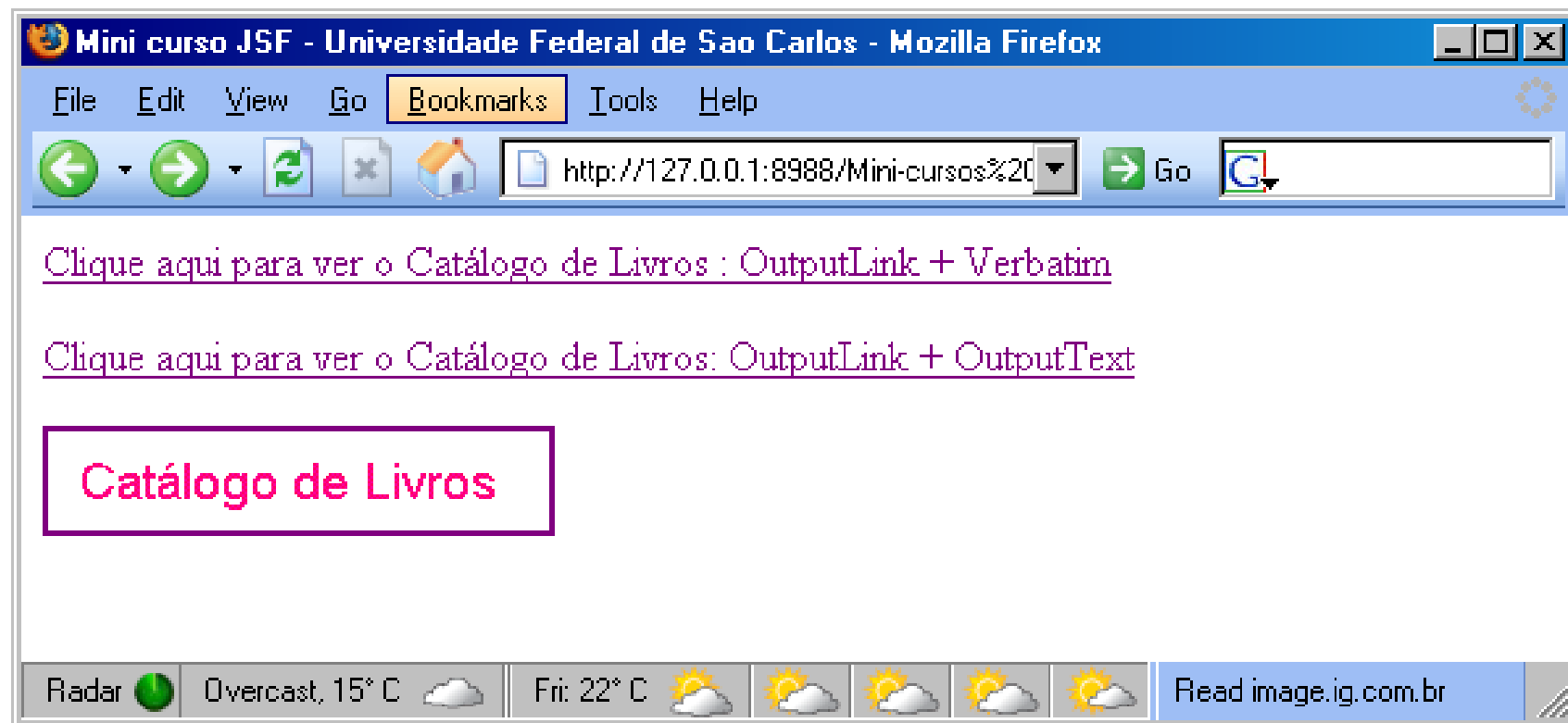
- Um link : `<A HREF="catalogo.jsp"> texto do link </A>`
- Um formulário `<FORM action="catalogo.jsp">`

## Podemos utilizar os seguintes componentes JSF para gerar hyper links:

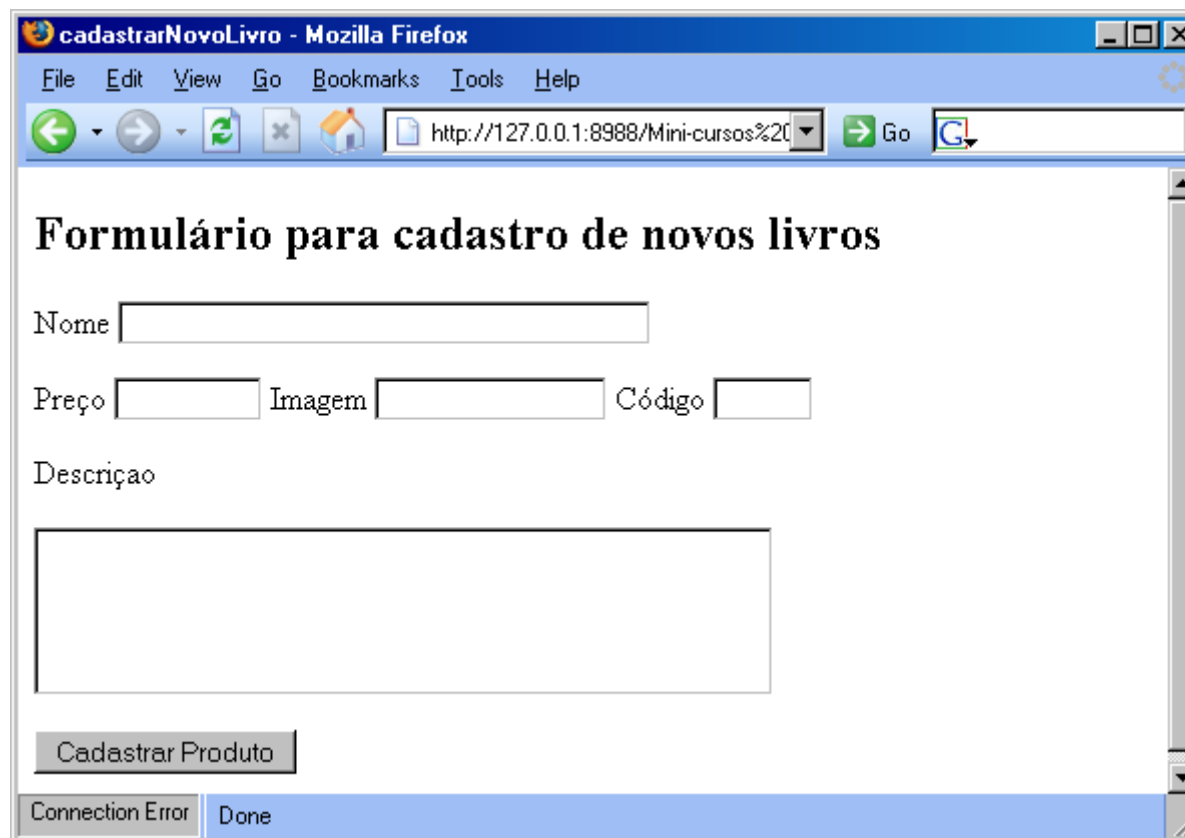
- `OutputLink`
- `CommandButton`
- `CommandLink`

1. **OutputLink:** Utilizamos OutputLink quando nenhuma ação deve ser realizada quando o link for clicado, ou seja, somente o redirecionamento nos interessa.
2. **CommandLink e CommandButton:** são utilizados para gerar links e submeter formulários, e podem ser configurados para executar uma determinada ação antes de enviar a Request para a página configurada.
3. **CommandLink:** utiliza JavaScript para submissão da Request.

# Tags de navegação



# Criação de páginas JSF



**cadastrarNovoLivro - Mozilla Firefox**

File Edit View Go Bookmarks Tools Help

http://127.0.0.1:8988/Mini-cursos%20 Go

## Formulário para cadastro de novos livros

Nome

Preço  Imagem  Código

Descrição

Cadastrar Produto

Connection Error Done

- Bibliotecas de Tags
- **Managed Beans**
- Componentes de User-interface (UI)
- Validadores
- Conversores
- Eventos
- Listeners
- Renderizadores
- FacesController

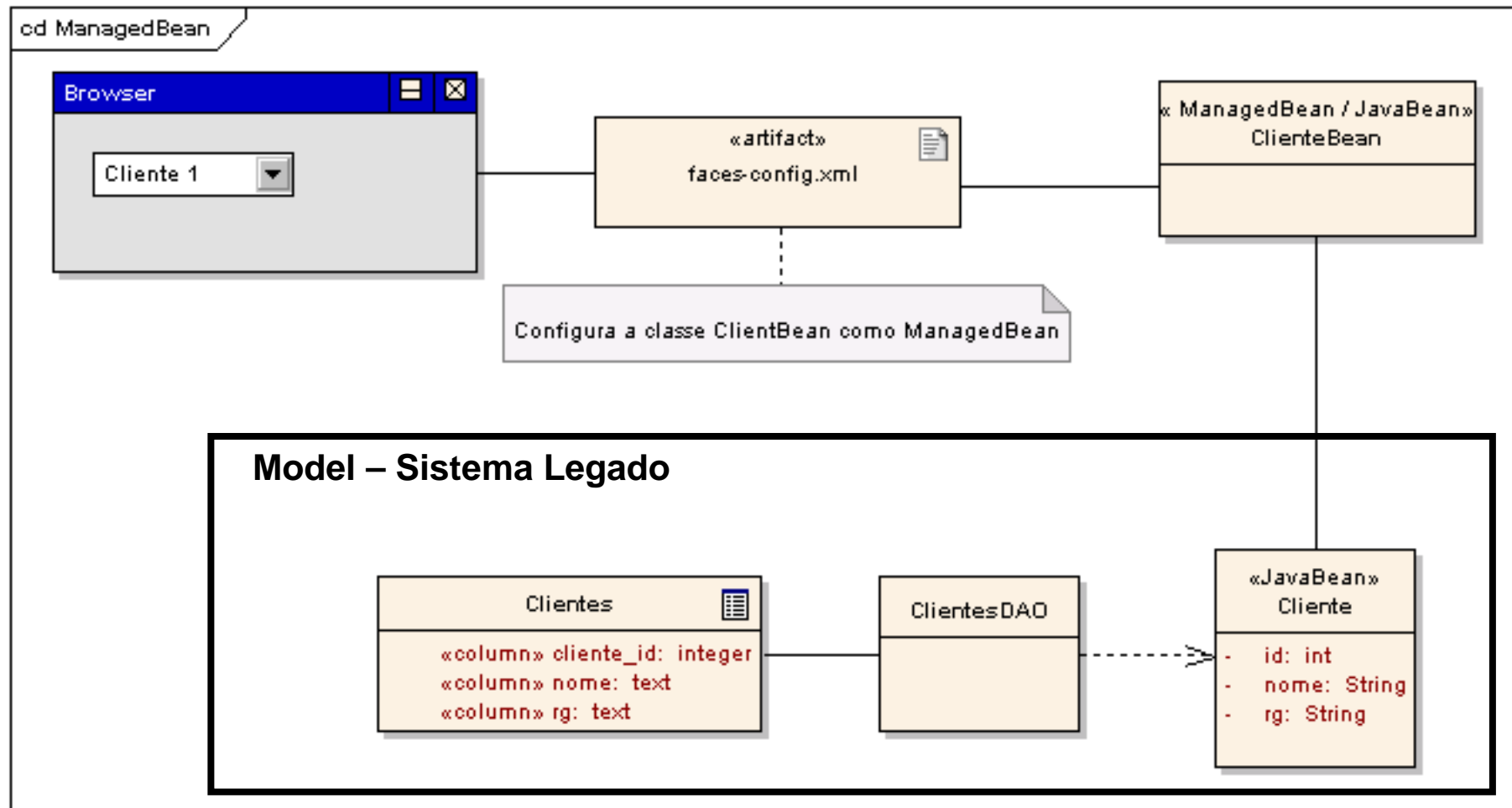
Um Managed Bean é um JavaBean gerenciado pelo framework JSF, ou seja, ele é instanciado, e colocado no escopo de acordo com as configurações encontradas no `faces-config.xml`

Um ManagedBean também é chamado de backing bean, pois contém os dados e os métodos que serão executados quando algum dos componentes da página JSF tiver que executar uma ação.



Chamamos de *binding* o vínculo entre um componente da página JSF e o seu backing model / managed bean.

# Managed Beans



- Utilizamos Taglibs e EL (Expression Language) para associar (fazer o binding) de um componente de UI com um ManagedBean;

**<h:outputText value="#{clienteBean.nome}"/>**

**A String clienteBean está associada a classe ClienteBean no faces-config.xml.**

## Declaração de um ManagedBean no faces-config.xml

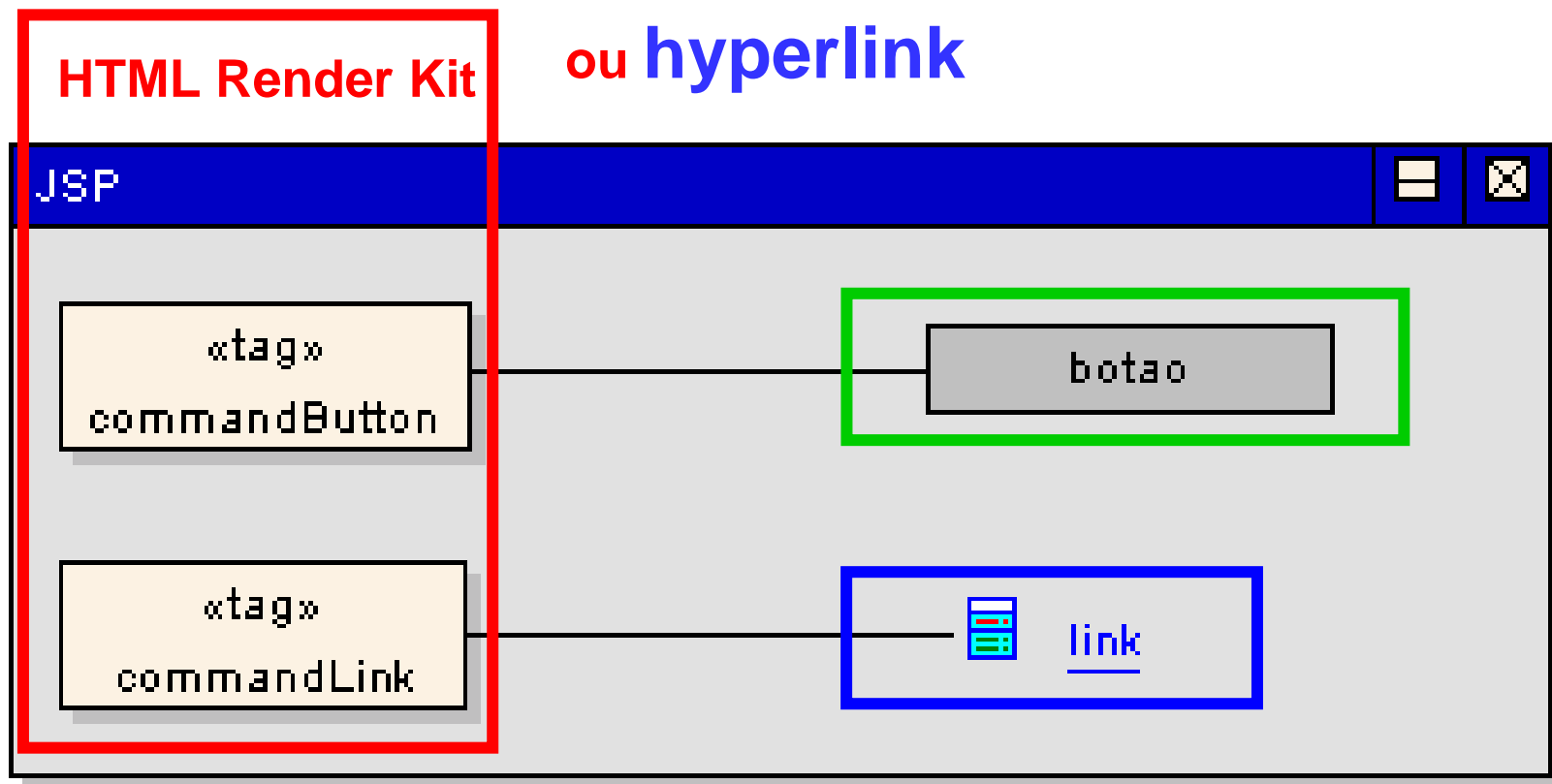
```
<managed-bean>  
  <managed-bean-name>clienteBean</managed-bean-name>  
  <managed-bean-class>ClienteBean</managed-bean-class>  
  <managed-bean-scope>session</managed-bean-scope>  
</managed-bean>
```

- Bibliotecas de Tags
- Managed Beans
- **Componentes de User-interface (UI)**
- Validadores
- Conversores
- Eventos
- Listeners
- Renderizadores
- FacesController

## Tecnicamente:

- Todos os componentes de UI são subclasses da classe abstrata `UIComponent`;
- A classe `UIComponentBase` já implementa os métodos necessários de `UIComponent` podendo ser estendida diretamente;

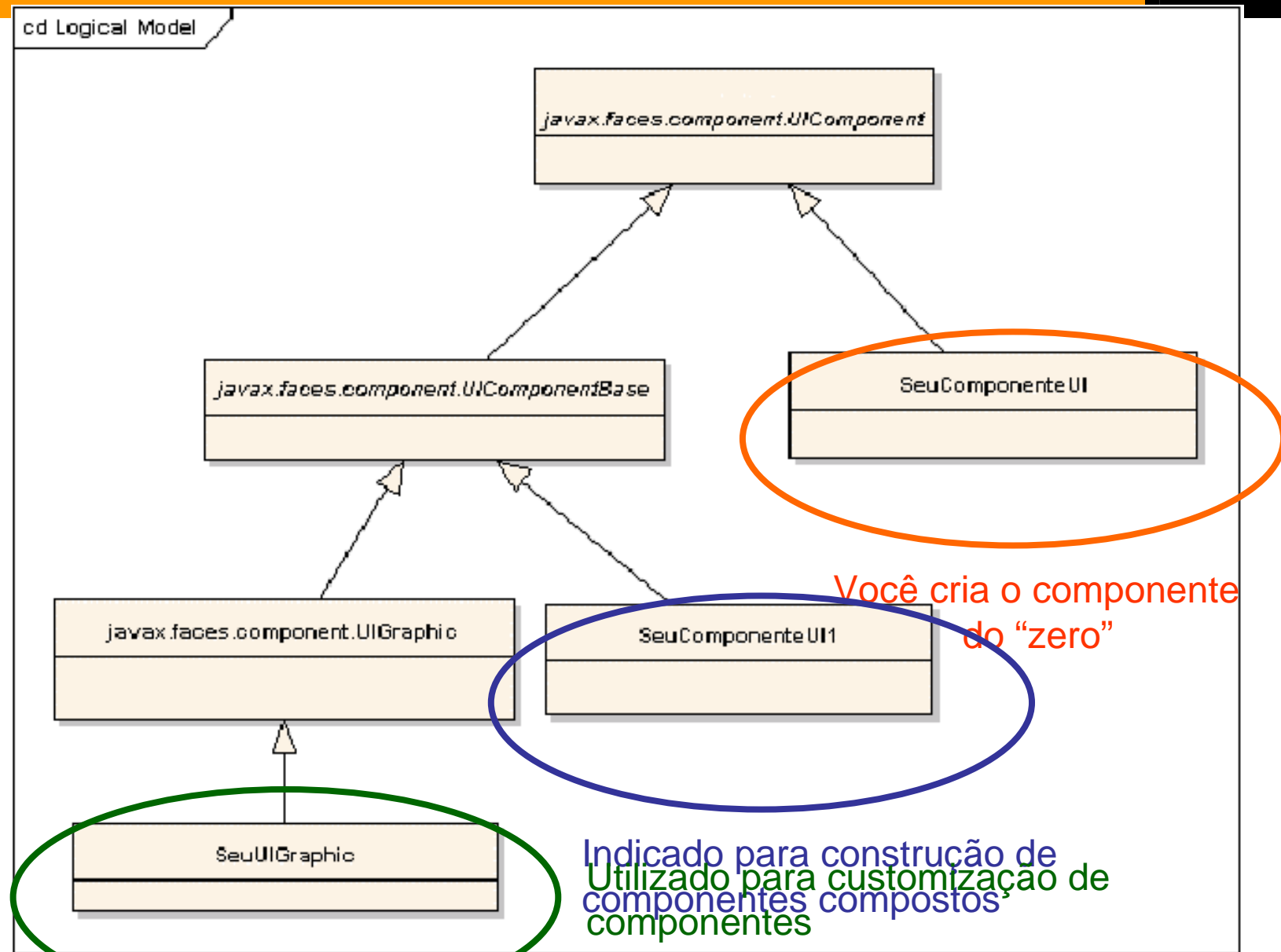
Custom Tag renderiza um **UICommand**  
em forma de **botão**  
ou **hyperlink**



**Componente de UI não é responsável pela renderização;**



# Componentes de User-Interface



## Standard UI Components

- `UICommand` (botões, hyperlinks, menus)
- `UIForm`, `UIInput`, `UIParameter`,
- `UIGraphic` (imagens)
- `UIMessage`, `UIMessages` (msgs de erro)
- `UIOutput`, `UIPanel`

## Standard UI Components

- `UISelectBoolean` (check box);
- `UISelectItems`, `UISelectItem`, `UISelectMany`  
`UISelectOne` (combo box, listas, conjunto de check boxes)
- `UIData`, `UIColumn` (tabelas, listas e árvores)

- Bibliotecas de Tags
- Managed Beans
- Componentes de User-interface (UI)
- **Validadores**
- Conversores
- Eventos
- Listeners
- Renderizadores
- FacesController

- Todos os componentes de UI derivados de **UIInput** podem ser validados;

## A validação pode ser feita das seguintes formas:

- Delegar a validação para um método de um JavaBean que estiver no escopo;
- Utilizar um Standard Validator
- Criar um Validator que implemente a interface `javax.faces.validator.Validator`

**Todas as implementações de JSF devem ter os seguintes validadores:**

- DoubleRangeValidator
- LengthValidator
- LongRangeValidator

- **Bibliotecas de Tags**
- **Managed Beans**
- **Componentes de User-interface (UI)**
- **Validadores**
- **Conversores**
- **Eventos**
- **Listeners**
- **Renderizadores**
- **FacesController**



## Problemas comum:

- Dados digitados pelo usuário são Strings;
- Dados apresentados podem estar formatados ou ter representação gráfica, como um calendário;
- É necessário converter estas Strings para Date, long, char, int, Cliente ou qualquer outro tipo e vice-versa.

- Todos os componentes de UI derivados de `UIOutput` podem ter conversores associados;
- Um conversor deve implementar a interface `javax.faces.convert.Converter`;

**Todas as implementações de JSF devem ter os seguintes conversores:**

- DateTime
- Number (moeda, porcentagem, inteiros, etc...)

- Bibliotecas de Tags
- Managed Beans
- Componentes de User-interface (UI)
- Validadores
- Conversores
- **Eventos**
- Listeners
- Renderizadores
- FacesController

- Modelo de eventos muito parecido com AWT e Swing
- Os Eventos são responsáveis pela propagação das ações sobre a interface com o usuário;
- Cada componente de UI pode disparar quantos eventos forem necessários;
- Um evento deve implementar a interface `javax.faces.event.FacesEvent`

- **A especificação do JSF padroniza duas interfaces de eventos:**
  - ActionEvent (geradas através da interação com UICommand)
  - ValueChangeEvent (geradas através da interação com UIInput)
- **A especificação JavaBeans recomenda que todas as classe que representam eventos sejam pós-fixadas com a palavra Event.**

- **Bibliotecas de Tags**
- **Managed Beans**
- **Componentes de User-interface (UI)**
- **Validadores**
- **Conversores**
- **Eventos**
- **Listeners**
- **Renderizadores**
- **FacesController**

- Cada tipo de evento tem uma interface listener correspondente;
- Toda interface listener deve estender a interface `javax.faces.event.FacesListener` ;
- A especificação JavaBeans recomenda que toda a classe que representa um listener tenha seu nome baseado no evento que está associada e seja pós-fixada com Listener ;



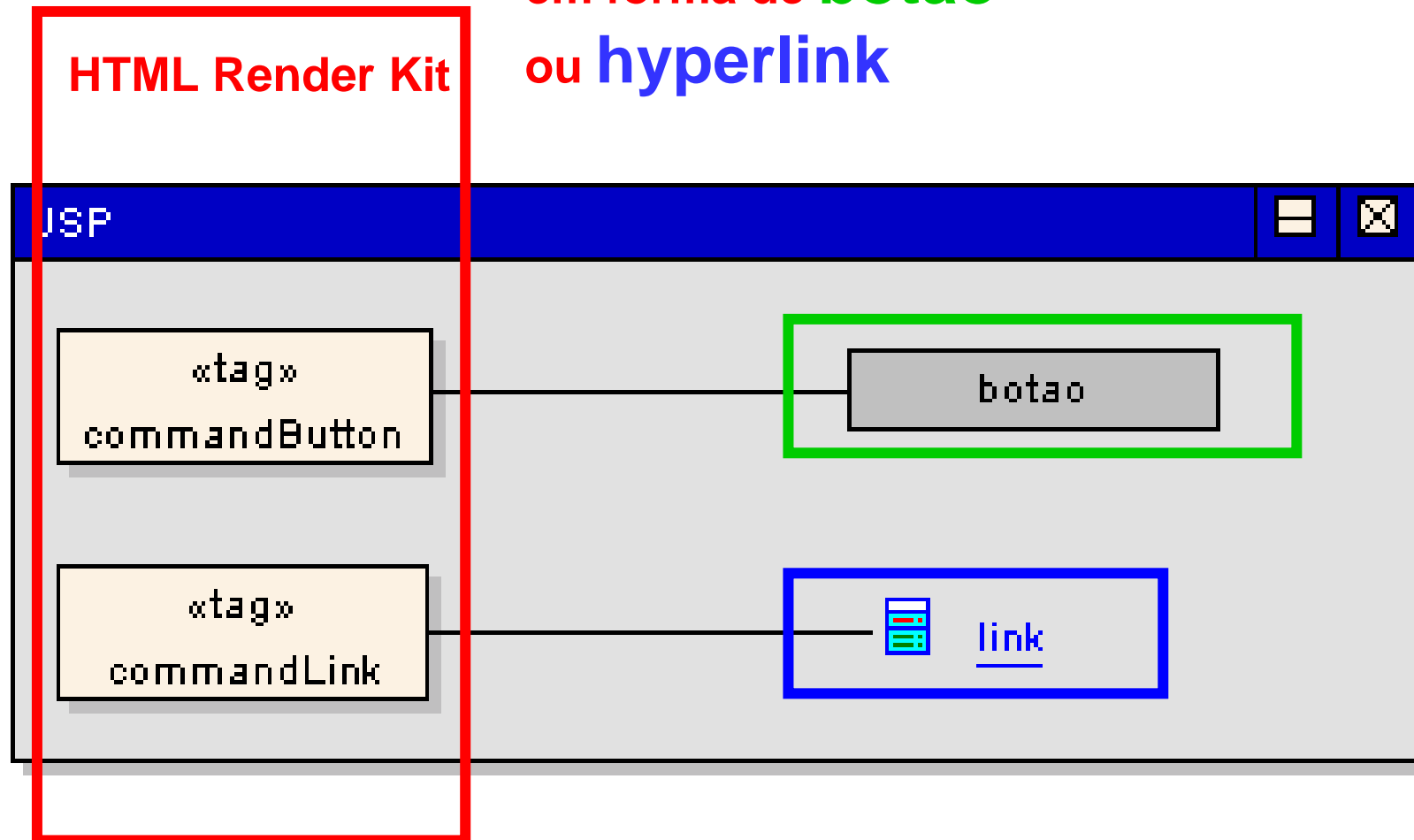
- **A especificação do JSF define duas interfaces de Listeners padrão:**
  - ActionListener (UICommand)
  - ValueChangeListener (UIInput)

- Cada componente de UI pode ter quantos listeners forem necessários;
- Um Componente de UI também pode ser um listener;

- **Os componentes de UI tem métodos para registrar e remover listeners. Exemplos:**
  - addActionListener
  - removeActionListener
- **O registro de listeners também pode ser feito através das custom tags encontradas na Standard JSF HTML tag library**

- **Bibliotecas de Tags**
- **Managed Beans**
- **Componentes de User-interface (UI)**
- **Validadores**
- **Conversores**
- **Eventos**
- **Listeners**
- **Renderizadores**
- **FacesController**

Custom Tag renderiza um **UICommand**  
em forma de **botão**  
ou **hyperlink**



- A renderização do componente pode ser feita pelo próprio componente ou delegada para um Renderizador.
- A renderização através de um Renderizador provê maior flexibilidade e reusabilidade para aplicação.

- A aparência fica separada da funcionalidade. O componente provê a funcionalidade e o renderizador a aparência;
- Utilizamos estes componentes de UI através de custom tags definidas no HTML RenderKit;
- O HTML RenderKit deve obrigatoriamente ser oferecido por todas as implementações de JSF;

- Um RenderKit é uma coleção de renderizadores (`javax.faces.render.Renderer`)
- Um RenderKit deve implementar a interface `javax.faces.render.RenderKit`



## Standard JSF HTML Renderers:

- Button
- Web Link
- Table
- Form
- Image
- Hidden
- Secret
- Input Text
- TextArea
- Label
- Output Link
- Output Text
- Grid
- Group
- Checkbox
- Listbox
- Menu
- Radio

- **Bibliotecas de Tags**
- **Managed Beans**
- **Componentes de User-interface (UI)**
- **Validadores**
- **Conversores**
- **Eventos**
- **Listeners**
- **Renderizadores**
- **FacesController**

- Um dos principais componentes de controle da aplicação;
- Implementação de FrontController conveniente para gerenciamento de segurança e acesso a aplicações Web;
- Responsável por grande parte do ciclo de vida de uma aplicação JSF, pois “inicia” vários processos;

- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

## JARs necessários:

- jsf-api.jar
- jsf-ri.jar
- jstl.jar
- standard.jar
- commons-beansutils.jar
- commons-digester.jar
- commons-collections.jar
- commons-logging.jar

## Deployment Descriptor: web.xml

- Fazer o mapeamento do FacesController
- Configurar o tipo de persistência dos dados da tela (no cliente ou servidor)

## **faces-config.xml**

Neste arquivo são feitas todas as configurações da aplicação JSF, como por exemplo:

- Navegação
- Managed Beans
- Validators

## A navegação é configurada no faces-config.xml

```
<navigation-rule>  
  <from-view-id>/index.jsp</from-view-id>  
  
  <navigation-case>  
    <description>Descricao</description>  
    <from-outcome>viewClienteData</from-outcome>  
    <to-view-id>/viewClienteData.jsp</to-view-id>  
  </navigation-case>  
  
</navigation-rule>
```



- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

- JSP não é a única forma de construir interfaces para JSF;
- A integração é feita através de TagLibs;
- As TagLibs “ligam” os componentes server-side aos client-side ( tipicamente HTML)

- **Core Tag Library:** gerenciamento de listeners, configuração de componentes, validação, entre outros;
- **HTML Tag Library:** Definem o renderizador do componente de UI, utilizam EL para integração com os Managed Beans;
- Existe uma tag para cada combinação entre renderizador e componente;  
Por exemplo, um **UIInput** pode ser renderizado em forma de **inputText** ou de **inputSecret**;

- Configurações necessárias:

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
```

```
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="h" %>
```

- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

- Java Studio Creator
- JDeveloper
- JBuilder
- MyEclipseIDE
- Red Hat Developer Studio
- WSAD
- Eclipse 3.3
- NetBeans Visual Web Pack

- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Perguntas e Respostas**

- [http://java.sun.com/developer/technicalArticles/J2EE/intro\\_ee5](http://java.sun.com/developer/technicalArticles/J2EE/intro_ee5)
- <http://java.sun.com/j2ee/javaxserverfaces>
- Mastering JavaServer Faces – Editora Wiley
- <http://www.jcp.org/en/jsr/detail?id=127>



- 🌀 **Introdução**
- 🌀 **Arquitetura JSF**
- 🌀 **Principais componentes JSF**
- 🌀 **Configuração**
- 🌀 **Integração JSF e JSP**
- 🌀 **Ferramentas**
- 🌀 **Links**
- 🌀 **Como usar AJAX**
- 🌀 **AJAX e JSF**
- 🌀 **Perguntas e Respostas**

