JavaServer Faces

Conceitos-chave de JSF

Ciclo de vida da requisição

Alteracoes e atualizacoes: Gustavo Wagner © *2008*
gugawag@gmail.com

Slides originais gentilmente cedidos por: Frederico Costa Guedes Pereira © 2007
fredguedespereira@gmail.com

O que é?

- Um framework padrão da plataforma JEE para desenvolvimento RAD na web usando MVC
- Versão atual: 1.2
- Provê três fundamentos:
  - Uma arquitetura de componentes
  - Um conjunto padrão de elementos de GUI (widgets)
  - Um modelo de programação orientado a eventos

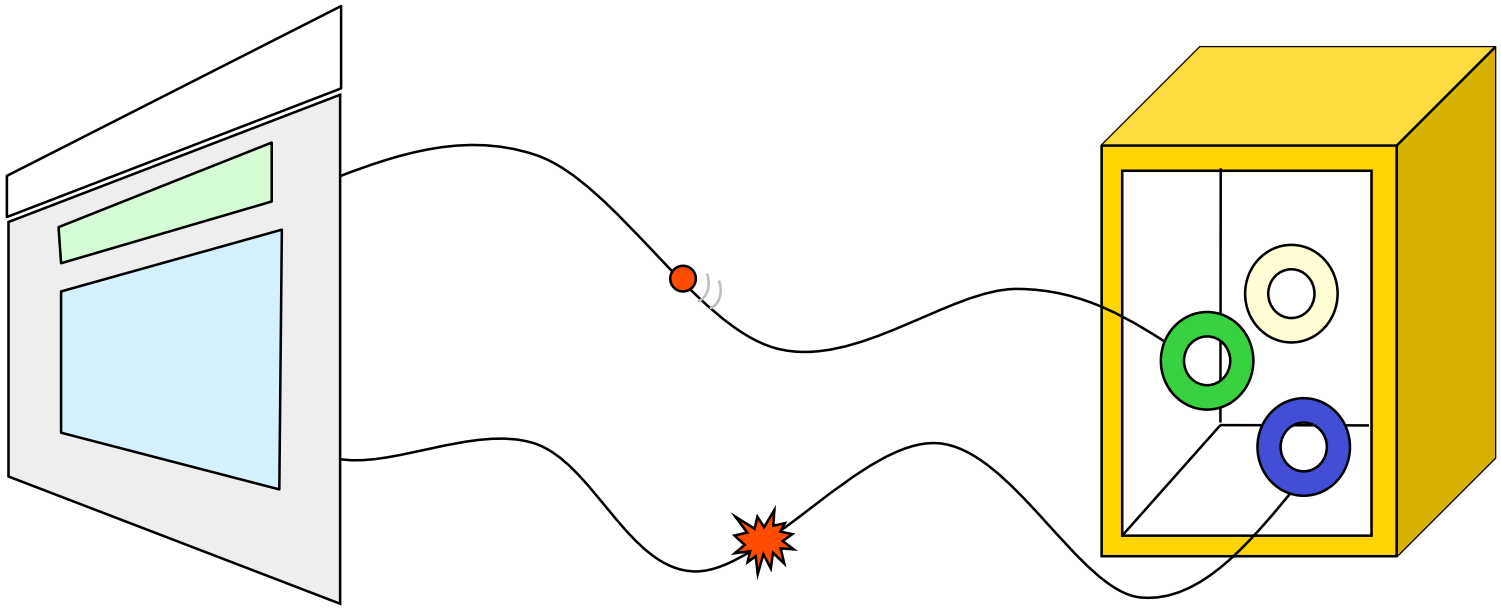
Arquitetura de componentes

- Uma arquitetura de componentes possibilita que um software possa ser “montado” a partir de pedaços (componentes) mais simples e plugáveis
- A arquitetura do JSF possui componentes pré-definidos e é extensível (crie seus próprios componentes, visuais ou não)
- Componentes possuem propriedades e geram eventos
  - visualização → browser, tratamento do evento→ servidor

Arquitetura de componentes

- A visualização dos componentes pode ser renderizada em diferentes dispositivos/formatos (html, celular, pda, etc..)
  - Componentes possuem facilidades de validação
  - Componentes podem manter-se sincronizados com objetos Java (*backing beans*)
  - Componentes têm suporte para internacionalização
- JSF possui facilidades para definir a navegação entre as páginas

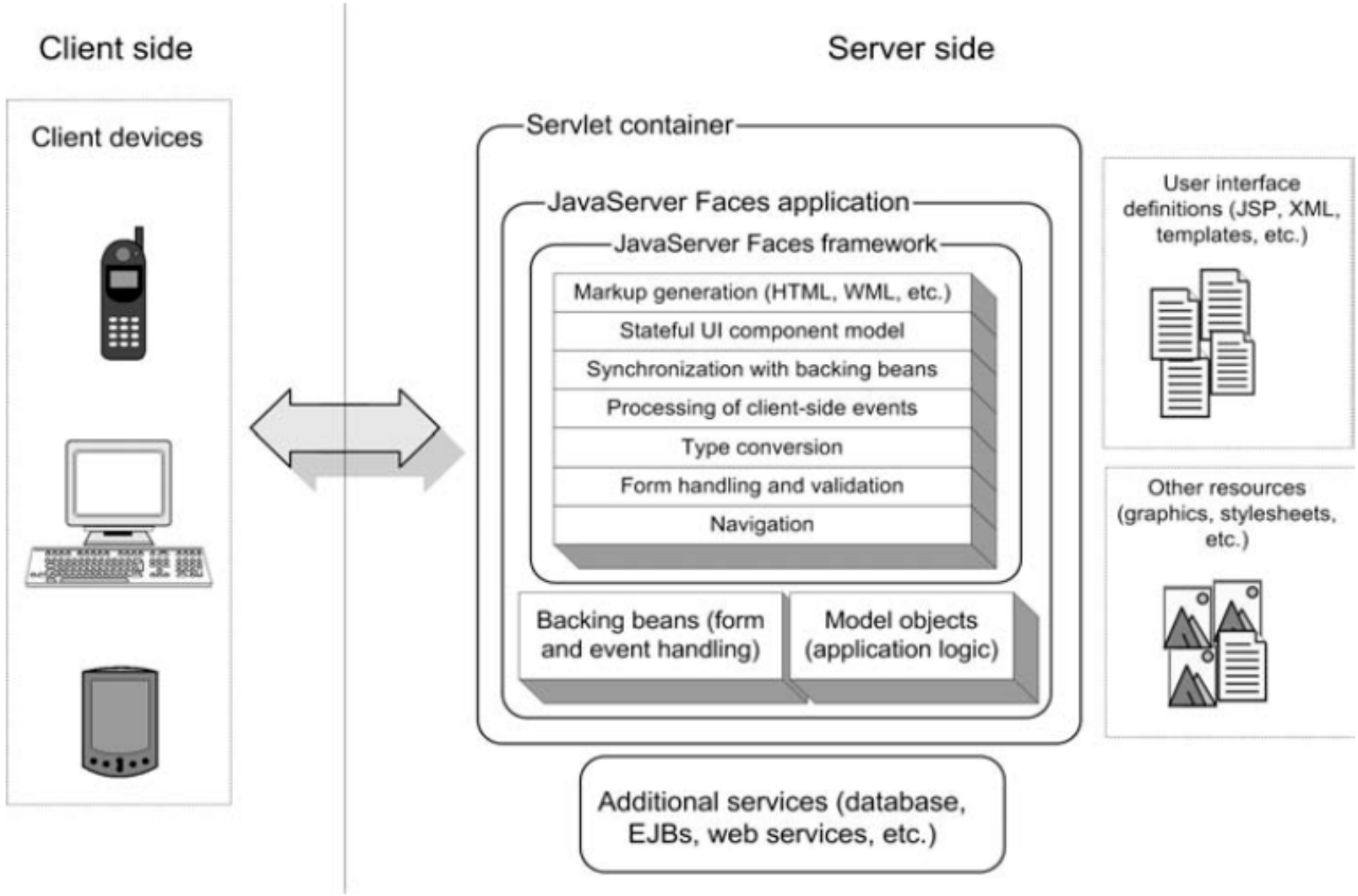
JavaServer Faces



Browser

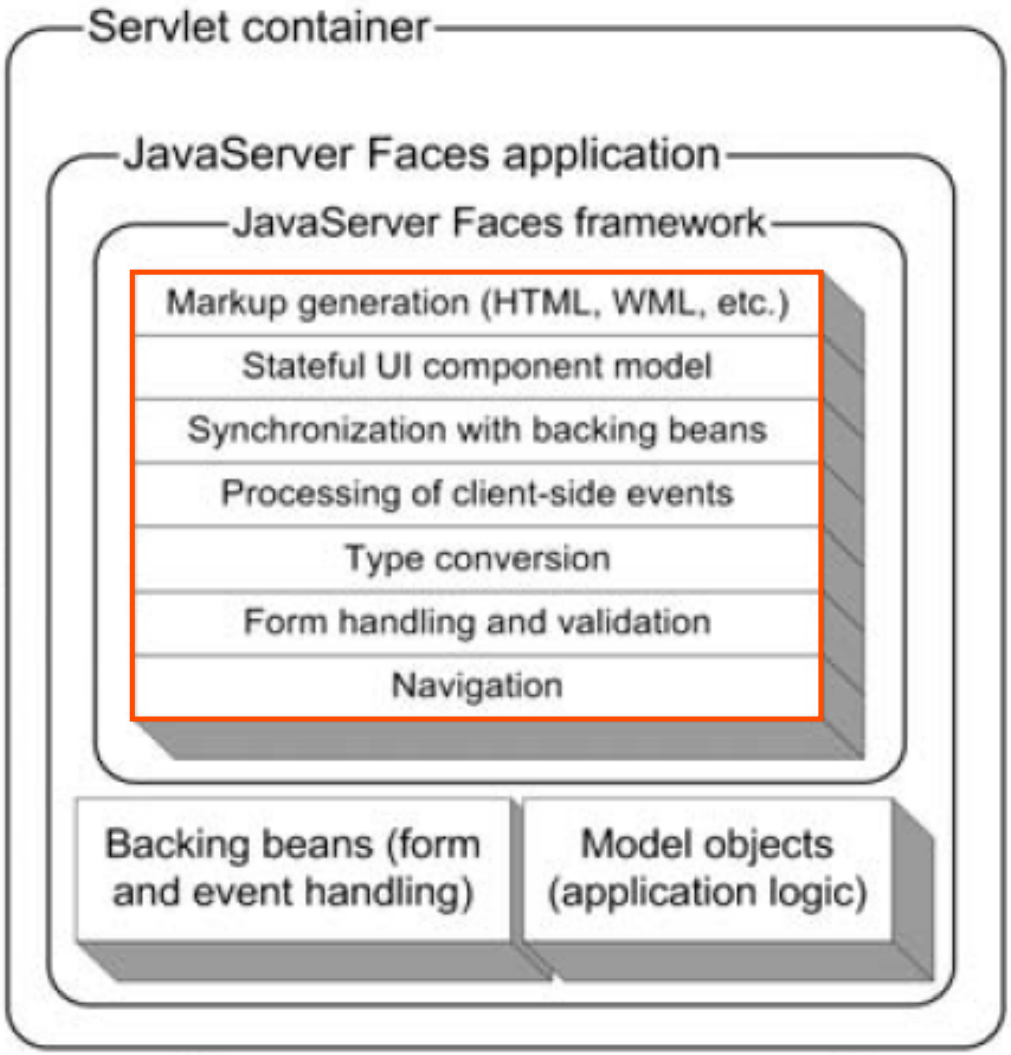
Servidor

JavaServer Faces



Tarefas do framework

Retirado do livro JSF in action

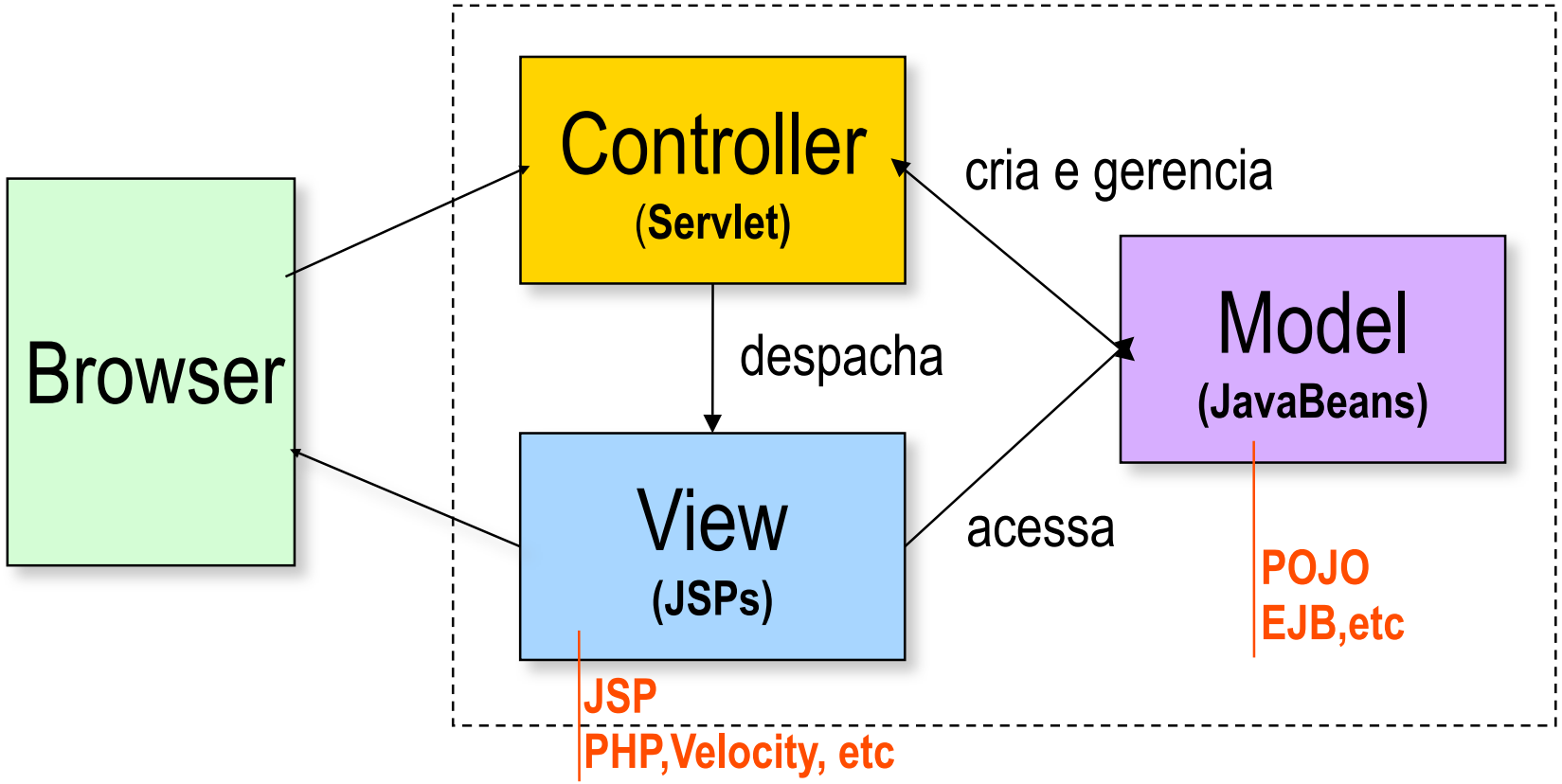


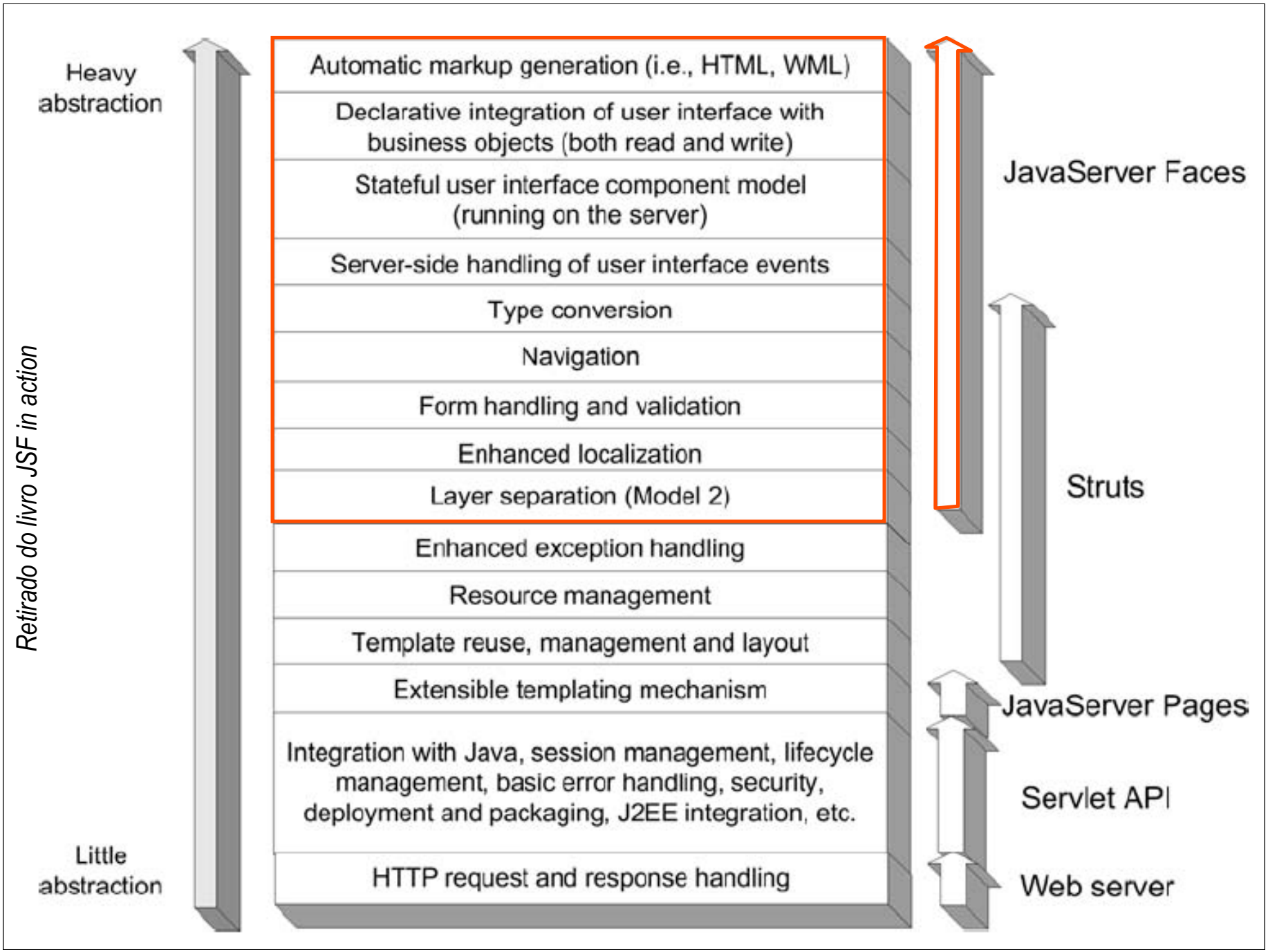
JavaServer Faces

- **Objetivo**: facilitar o desenvolvimento de aplicações web
- **Como procura alcançar**:
  - Fazendo o desenvolvedor pensar numa aplicação web como sendo composta por **componentes**, **backing beans** e **eventos** ao invés de requests, responses e marcações (html);
  - Utilizando a separação de papéis na aplicação em Model, View e Controller
  - Abstraindo os detalhes de baixo nível de uma aplicação web (ciclo do protocolo HTTP)

JavaServer Faces

- JSF é um *framework* MVC para a web:




Heavy abstraction
Little abstraction
Retirado do livro JSF in action
Automatic markup generation (i.e., HTML, WML)
Declarative integration of user interface with business objects (both read and write)
Stateful user interface component model (running on the server)
Server-side handling of user interface events
Type conversion
Navigation
Form handling and validation
Enhanced localization
Layer separation (Model 2)
Enhanced exception handling
Resource management
Template reuse, management and layout
Extensible templating mechanism
Integration with Java, session management, lifecycle management, basic error handling, security, deployment and packaging, J2EE integration, etc.
HTTP request and response handling
JavaServer Faces
Struts
JavaServer Pages
Servlet API
Web server

Conceitos-chave de JSF

- ***Componente de UI***
  - São JavaBeans mantidos no servidor e que oferecem funcionalidades específicas de interatividade com o usuário. São usados nas visões, estruturados como uma árvore (como o DOM de uma página num browser).
- ***Renderer***
  - Sabem como apresentar um componente de UI nos dispositivos de E/S específicos (palm, desktops, etc.)
- ***Validador***
  - Sabem como realizar validações nos valores fornecidos pelo usuário

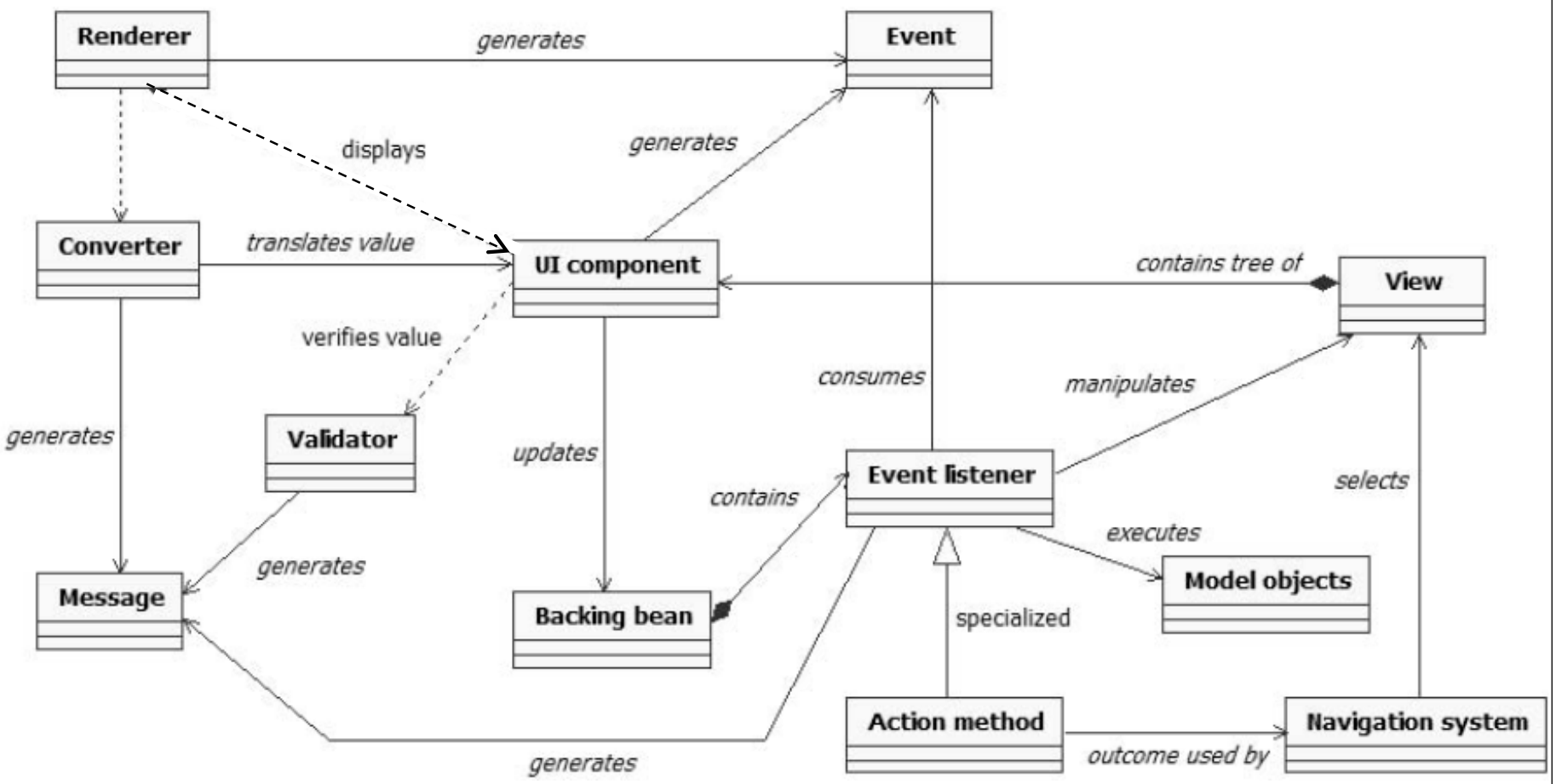
Conceitos-chave de JSF

- ***Backing beans***
  - JavaBeans que recebem os valores dos componentes e implementam métodos tratadores de eventos. Podem referenciar um componente de UI.
- ***Conversor***
  - Convertem valores de componentes UI em Strings e vice-versa. (ex.: converter a String 15/10/2008 num objeto Date)
- ***Eventos*** e ***tratadores*** (listeners)
  - JSF segue um modelo semelhante ao Swing para lidar com eventos de componentes de UI.

Conceitos-chave de JSF

- ***Mensagens***
  - Fornecem uma forma padronizada dos outros elementos enviarem uma mensagem de erro para o usuário
- ***Navegação***
  - JSF tem um mecanismo para definir a navegação entre as páginas da aplicação, ao invés dela ser definida dentro de cada página. Isso facilita a manutenção.

Relação entre os elementos



Retirado do livro JSF in action

Componentes de UI

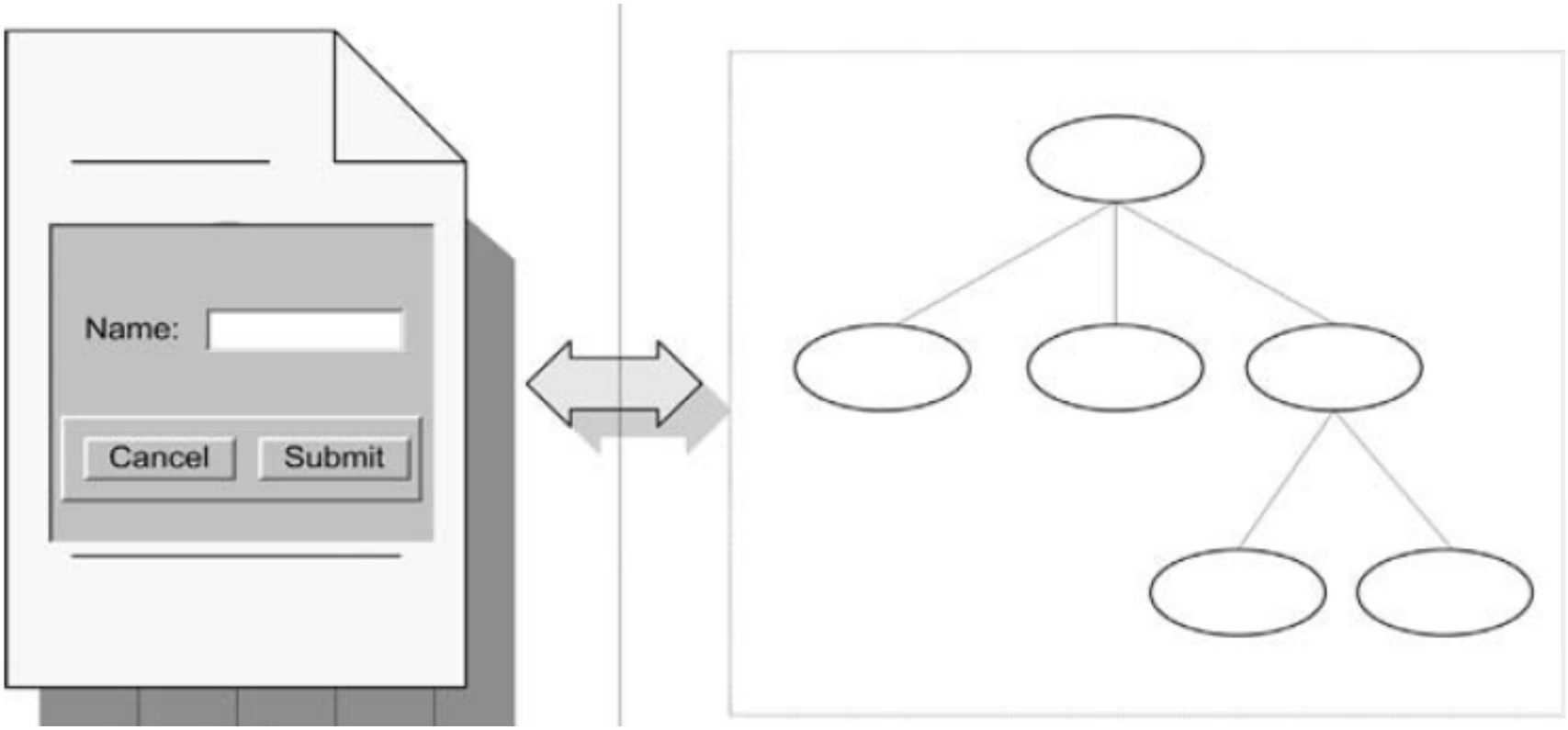
- São **objetos** (java beans) **que vivem no servidor** e não no cliente
  - Uma representação visual deles é que vai para o cliente!
- São fáceis de **reutilizar**
  - Encapsulam várias tecnologias, oferecendo para o usuário apenas um conjunto de propriedades
- Podem ser **renderizados** de diferentes formas
  - Um componente Faces pode ser exibido num documento HTML, num visor de celular ou de palm ou noutra forma qualquer de exibição, desde que exista um *render* para ele

Componentes de UI

- Suas representações gráficas são **atualizadas automaticamente** pelo JSF entre requisições da mesma visão onde se encontram;
  - JSF mantém os valores dos *widgtes* sincronizados com o estado do componente UI
- JSF mantém uma **árvore de componentes UI** que pertencem a uma visão
  - Representação interna do JSF para a visão
  - Cada componente na árvore possui um ID
    - Definido pelo desenvolvedor ou gerado automaticamente
    - exemplo de ID: para a pagina http://localhost:8080/olahmundo/faces/index.jsp ID serah: /index.jsp

Componentes de UI

- Árvore de componentes de UI e sua visão

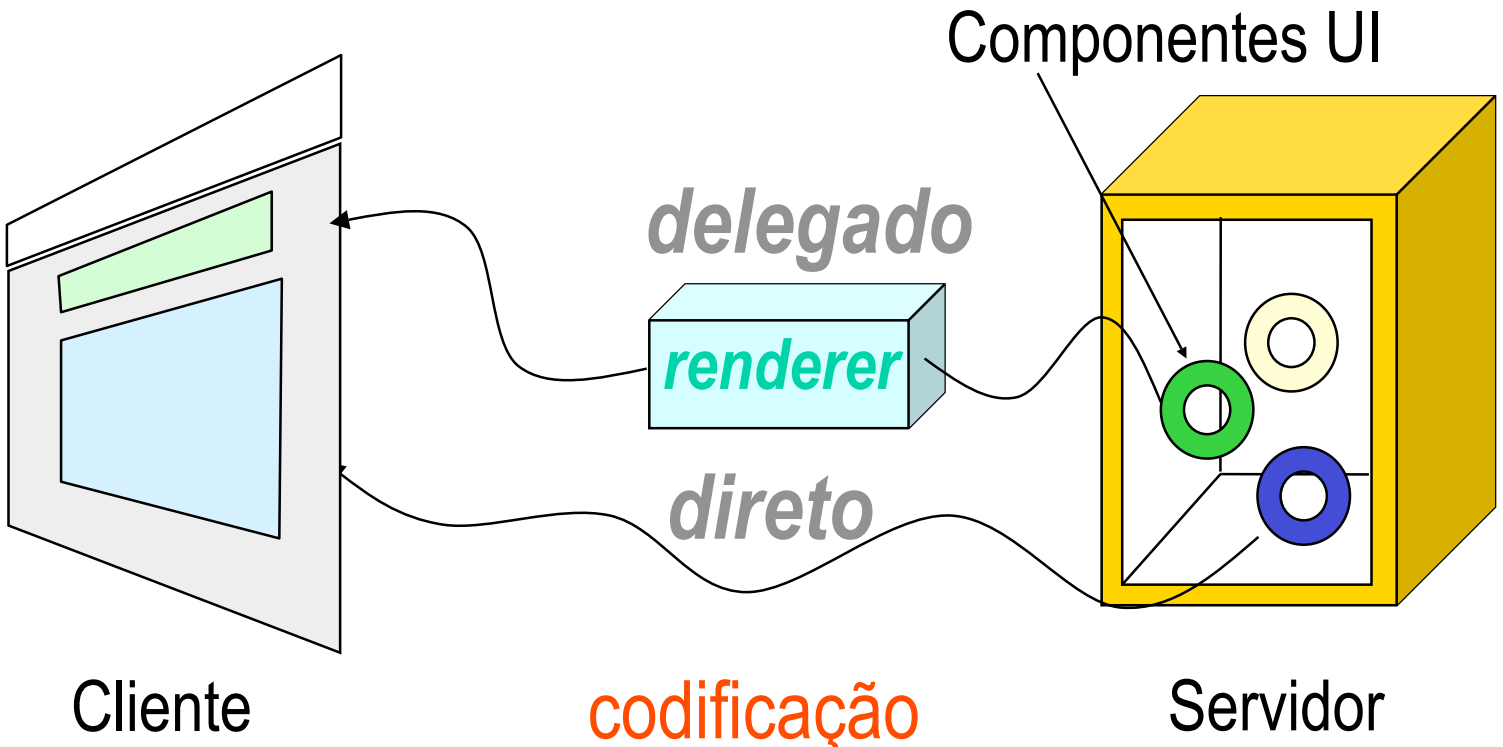


Retirado do livro JSF in action

Renderizadores

- Dois modelos:
  - **Implementação direta**: componentes de UI são responsáveis pela própria renderização.
  - **Implementação delegada**: o componente de UI delega para um renderizador (um objeto) esta tarefa.

Renderizadores



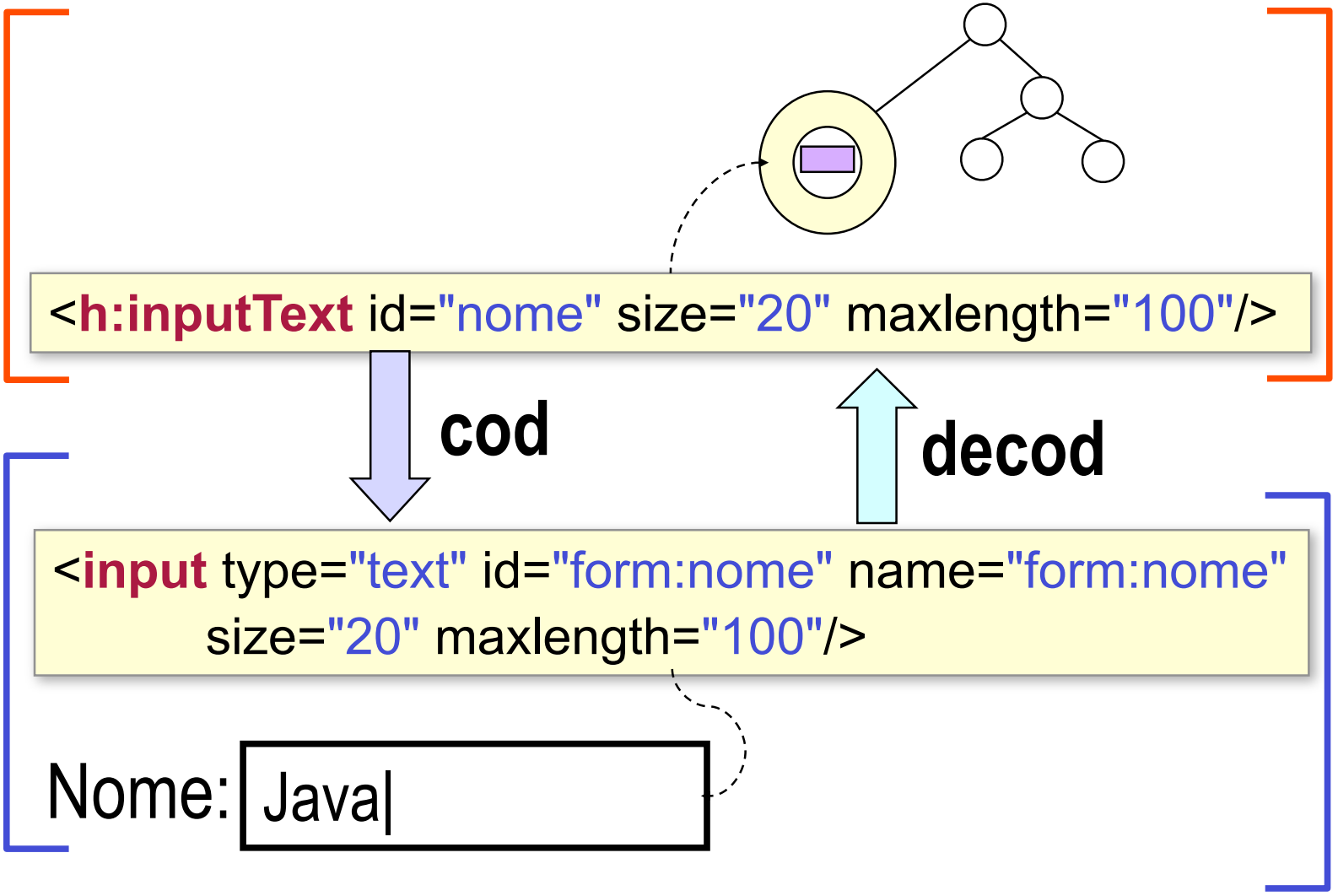
codificação

criar uma representação gráfica para o componente

decodificação

atualizar o componente com os valores fornecidos na UI

Renderizadores



Validadores

- Três níveis de validação
  - No ***componente de UI***: simples e específicas ao componente
  - Em ***métodos de um backing bean***: quando envolvem mais de um campo do formulário
  - Em ***classes de validadores***: plugáveis e gerais
- Toda a validação é realizada no servidor!

```
<h:inputText>
    <f:validateLength minimum="2" maximum="10"/>
</h:inputText>
```

Backing beans

- São o "C" do MVC do JSF!
  - Mantêm valores de componentes de UI
  - Possuem métodos para tratamento de eventos
- Componentes podem obter dados de propriedades dos *back beans* via JSF-**EL**

```
<h:outputText id="helloBeanOutput"
        value="#{helloBean.numControls}"/>
```

O valor do componente HtmlOutputText vem diretamente da propriedade do **helloBean**

Backing beans

- Um bean também pode referenciar um objeto de um componente visual (que fica no servidor)

```
<h:panelGrid id="controlPanel"
        binding="#{helloBean.controlPanel}"/>
```

O objeto **helloBean** possui uma propriedade do tipo **HtmlPanelGrid**, podendo acessá-lo programaticamente.

Backing beans

- Como os *backing beans* são criados?

managed bean

```
<managed-bean>
    <managed-bean-name>
        helloBean
    </managed-bean-name>
    <managed-bean-class>
        com.virtua.jsf.sample.hello.HelloBean
    </managed-bean-class>
    <managed-bean-scope>
        session
    </managed-bean-scope>
</managed-bean>
```

Conversores

- Convertem objetos (tipos) em String para serem apresentados ao cliente (nas visões) e vice-versa, no sentindo inverso.
  - Cuidam da **formatação** e **localização** também
- São utilizados internamente pelos renderizadores ou pelos componentes visuais

```
<h:outputText value="#{user.dateOfBirth}">
      <f:convert_datetime type="both" dateStyle="short" />
</h:outputText>
```

Conversores

- Esquematicamente: