Java Server Faces

Introdução

As tecnologias para o desenvolvimento de aplicações WEB têm mudado constantemente. Inicialmente os sites possuíam apenas conteúdo estático. Depois, passaram a oferecer páginas com conteúdos dinâmicos e personalizados. Diversas tecnologias estão envolvidas no desenvolvimento das aplicações WEB como, CGI, Servlets e JSP (Java Server Pages).

A primeira tecnologia voltada para a construção de páginas dinâmicas foi a CGI, os quais são escritos em qualquer linguagem de programação. Eles, porém, apresentam problemas de portabilidade e escalabilidade, além de mesclarem as regras de negócio com a visualização.

Em seguida vieram os servlets, eles são pequenos programas feitos em Java que encapsulam alguma funcionalidade inerente à sua aplicação WEB. No entanto, eles ainda não resolvem o problema da separação das regras de negócio da visualização, dificultando a manutenção.

Posteriormente surgiram as páginas JSP. Elas são facilmente codificadas e produzem conteúdos reutilizáveis. Assim como os servlets, as JSPs também não resolvem o problema da manutenção das aplicações.

Esse problema só foi resolvido quando começou a se aplicar os design patterns no desenvolvimento das páginas. No caso das tecnologias para desenvolvimento WEB usando Java, o design pattern utilizado é o MVC (Model-View-Controller).

Com a grande utilização dos patterns, principalmente no "mundo Java", começaram a surgir diversos frameworks para auxiliar no desenvolvimento de aplicações WEB como Struts, WebWork, JSF.

O que é JavaServer Faces?

JSF é uma tecnologia que incorpora características de um framework MVC para WEB e de um modelo de interfaces gráficas baseado em eventos. Por basear-se no padrão de projeto MVC, uma de suas melhores vantagens é a clara separação entre a visualização e regras de negócio (modelo).

O Padrão MVC segundo JSF

No JSF, o controle é composto por um servlet denominado FacesServlet, por arquivos de configuração e por um conjunto de manipuladores de ações e observadores de eventos. O FacesServlet é responsável por receber requisições da WEB, redirecioná-las para o modelo e então remeter uma resposta. Os arquivos de configuração são responsáveis por realizar associações e mapeamentos de ações e pela definição de regras de navegação. Os manipuladores de eventos são responsáveis por receber os dados vindos da camada de visualização, acessar o modelo, e então devolver o resultado para o FacesServlet.

O modelo representa os objetos de negócio e executa uma lógica de negócio ao receber os dados vindos da camada de visualização. Finalmente, a visualização é composta por component trees (hierarquia de componentes UI), tornando possível unir um componente ao outro para formar interfaces mais complexas. A Figura 1 mostra a arquitetura do JavaServer Faces baseada no modelo MVC.

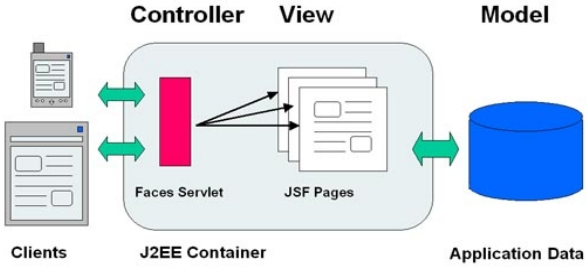


Figura 1 : Arquitetura do JSF baseada no modelo MVC.

Características e Vantagens

- Permite que o desenvolvedor crie UIs através de um conjunto de componentes UIs pré-definidos;
- Fornece um conjunto de tags JSP para acessar os componentes;
- Reusa componentes da página;
- Associa os eventos do lado cliente com os manipuladores dos eventos do lado servidor (os componentes de entrada possuem um valor local representando o estado no lado servidor);
- Fornece separação de funções que envolvem a construção de aplicações WEB.

Serviços do JSF

JSF possui dois principais componentes: Java APIs para a representação de componentes UI e o gerenciamento de seus estados, manipulação/observação de eventos, validação de entrada, conversão de dados, internacionalização e acessibilidade; e taglibs JSP que expressam a interface JSF em uma página JSP e que realizam a conexão dos objetos no lado servidor.

O framework JSF é responsável pela interação com dispositivos clientes e fornece as ferramentas para unir a apresentação visual, a lógica da aplicação e a lógica de negócios de uma aplicação web. Entretanto, o escopo do JSF se restringe à camada da apresentação. Consistência de bancos de dados, serviços web e outras conexões de backend (de servidores) estão fora do escopo do JSF.

Aqui estão os serviços mais importantes que o framework JSF oferece:

- Arquitetura Modelo-Visão-Controlador

  Todos as aplicações em software permiten aos usuários manipular certos dados, tais como carrinhos de compras virtuais, itinetários de viajens ou quaisquer dados que sejam necessários para um determinado contexto. Esses dados são chamados de modelo. Da mesma forma como um artista cria uma pintura de um modelo em seu estúdio, um desenvolvedor de software produz visões do modelo do dados. Em uma aplicação web, o HTML (ou alguma tecnología de renderização semelhante) é usando para tomar a aplicação amigável ao usuário a partir dessas visões.

  O JSF conecta a visão com o modelo. Como você já viu, um componente de visão pode ser ligado a uma propiedade bean de um objeto modelo, como por exemplo>

  <h:inputText value="#{user.name}"/>

  Além disso, o JSF opera como o controlador que reage ao usuário ao processar eventos de ação de mudança de valores, encadeando-os para o código que atualiza o modelo ou a visão. Por exemplo, você pode desejar invocar um método para checar se o usuário está autorizado para fazer login. Use a seguinte tag JSF:

  <h:commandButton value="Login" action="#{user.check}"/>

  Quando o botão é clicado e o formulario é submtido ao servidor, a implementação JSF invoca o método check do bean user. Esse método pode realizar ações arbitrárias para atualizar o modelo e ele retorna a ID de navegação da próxima página a ser exibida. Nós discutiremos esse mecanismo em mais detalhes depois.

  Assim o JSF implementa a clássica arquitetura modelo-visão-controlador.

- Conversão de Dados

  Os usuários digitam dados em formulários web em forma de texto. Negócios requerem dados em forma de números, datas ou outros tipos de dados, o JSF facilita a tarefa de especificar curtomizar regras de conversão.

- Validação e Manipulação de Erros

O JSF falicita a tarefa de vincular regras de validação a campos do tipo "este campo é obrigatório" ou "este campo debe ser um número". É claro que, quando os usuários digitam dados inválidos, você precisa exhibir mensagens de erro apropiadas. O JSF elimina boa parte do tédio dessa tarefa de programação.

- Internacionalização

  O JSF gerencia questões de internacionalização tais como a codificação de caracteres e a seleção de resource bundles.

- Componentes Customizados

  Os desenvolvedores de componentes podem desenvolver componentes sofisticados que os designers de páginas podem simplesmente utilizar em suas páginas. Por exemplo, suponha que um desenvolvedor de componentes produza um componente calendário customizado. Você simplesmente o inclui na sua página, com um comando do tipo:

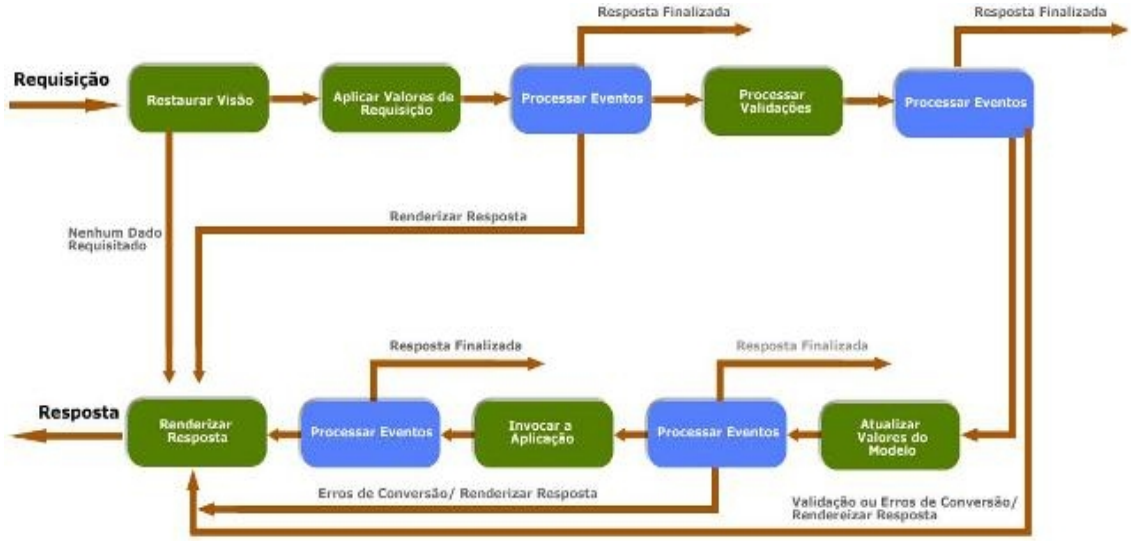
  <acme:calendar value="#{flight.departure}" startOfWeek="Mon"/>

- Renderizadores Alternativos

  Por padrão, o JSF gera comandos para páginas HTML. Mas é facil estender o framework JSF para produzir comandos para outra linguagem de descrição de páginas, tal como o WML ou o XUL.

- Suporte a Ferramentas

Ciclo de Vida



Define 6 fases diferentes

A fase "Restaurar visão" recupera a árvore de componentes para uma página requisada, caos ela tenha sido exibida anteriormente, ou constrói uma nova árvore de componentes caso esteja sendo exhibida pela primeira vez. Se a página tiver sido exhibida anteriormente, todos os componentes são retornadosao seu estado prévio. Isso significa que o JSF automáticamente recupera informações de formulários. Por exemplo, quando um usuário posta dados ilegaisque são rejeitados durante a codificação, as informações inseridas anteriormente são reexibidas para que o usuário as possa corrigir.

Se a requisisão não possuir dados requisitados, a implementação JSF passa para a fase "Renderizar Resposta". Isso acontece quando a página é exhibida pela primeira vez.
Do contrário, a próxima fazer é "Aplicar Valores de Requisição". Nessa fase, a implementação JSF atua sobre os objetos componentes da árvore de componentes. Cada objeto de componente verifica quais valores requisitados pertenecem ao objeto em questão e os armazena.

Na fase de "Processar Validações", a string de valores submetida é principalmente convertida em valores locais, que

podem ser objetos de qualquer tipo. Quando você elabora uma página JSF, você pode utilizar validadores que realizam correções nos valores locais. Se os dados estiverem válidos, o ciclo de vida do JSF procede normalmente. Entretanto, quando ocorrem erros de conversão ou de validação, a implementação JSF invoca a fase "Renderizar Resposta" diretamente, reexibindo a página atual para que o usuário tenha uma segunda chance de fornecer os dados corretos.

Depois que os conversores e validadores tiverem feito o seu trabalho, assume-se que é seguro atualizar os dados do modelo. Durante a fase "Atualizar Valores do Modelo" os valores locais são usados para atualizar os beans que estão vinculados aos componentes.

Na fase "Invocar Aplicação", é executado o método action do componente botão ou link que causou a submissão do formulário. Esse método pode executar processamentos de aplicação arbitrários. Ele retorna uma string de resultado que é passada para o handler de navegação. O handler de navegação procura a página seguinte. Finalmente, a fase "Renderizar Resposta" codifica a resposta e a envia para o navegador. Quando o usuário submete um formulario, clica em um link ou gera de qualquer outra forma uma nova requisição, o ciclo começa de zero.

Exemplo

O arquivo WebApplication.zip, apresenta o exemplo da palestra.