

PROGRAMAÇÃO WEB

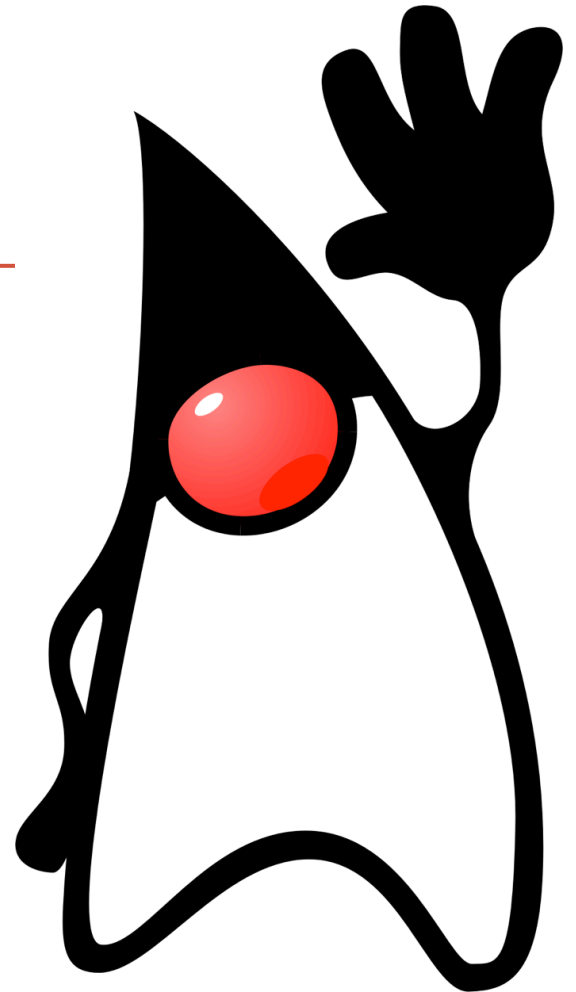


- Java Server Pages Avançado

Prof. Luiz Carlos Querino Filho
luiz.querino@fatec.sp.gov.br

Fatec Garça – 2017

Parte 04





Java Server Pages

- A tecnologia JSP (Java Server Pages) possibilita a “mesclagem” de código Java com código HTML diretamente.
- Páginas JSP devem possuir a extensão **.jsp**
- O código Java é colocado “no meio” do código HTML, sendo delimitado pelas tags **<%** e **%>**.
- Em uma página JSP, tudo que estiver fora das tags **<%** e **%>** é tratado como HTML e texto puro. Tudo que estiver dentro, será processado como código Java.
- Dentro do código Java também é possível imprimir texto e HTML, bastando para isso usar os métodos **print** e **println** do objeto **out**:

```
out.println(“Isto é <b>HTML</b>.”);
```



Java Server Pages

- Uma possibilidade interessante do uso do JSP é que o código em HTML também pode ficar “subordinado” ao código Java (no caso de **ifs** e repetições):

```
<%  
for (int i = 0; i < 10; i++) {  
%>
```

Este texto será repetido 10 vezes!


```
<%  
}  
%>
```



Java Server Pages

- Como alternar entre “modo HTML” e “modo Java” é bastante comum, existe um atalho JSP para simplesmente imprimir uma variável ou resultado de um comando no navegador. Basta colocar a variável ou comando entre as tags **<%=** e **%>**:

<%

int soma = valor1 + valor2;

%>

<h1>Resultado da operação:</h1>

<h2>

<%= valor1 %> + <%= valor2 %> = <%= soma %>

</h2>



Java Server Pages

- **Objetos implícitos:** são criados automaticamente pelo servidor de aplicações e estão disponíveis em todas as páginas JSP. Entre parênteses, as classes originais destes objetos:
- **out** (`javax.servlet.jsp.JspWriter`)
- **request**
(`javax.servlet.http.HttpServletRequest`)
- **session** (`javax.servlet.http.HttpSession`)
- **application** (objeto da classe `ServletContext`)
- **response**
(`javax.servlet.http.HttpServletResponse`)



Java Server Pages

- **Objetos implícitos:**
- **out** (objeto da classe `JspWriter`)
 - Possui métodos para imprimir conteúdo no navegador.
 - O conteúdo pode ser tanto colocado diretamente em código HTML, “saindo” do código JSP, quanto por meio dos métodos deste objeto:

```
out.print("Texto puro <b>com código HTML</b>");
```

```
out.println("Linha com quebra no final");
```



Java Server Pages

Objetos implícitos:

- request (objeto da classe `HttpServletRequest`)
- Usado principalmente para obtenção de valores passados por um formulário (usando método GET ou POST):

```
String nome = request.getParameter("nomeNoFormulario");
```

- Também pode ser usado para obter um **RequestDispatcher**, que pode ser usado para redirecionar a uma outra página:

```
request.getRequestDispatcher("saida.jsp").forward(request  
, response);
```

O que é uma sessão no servidor?

- Uma **sessão** (*session*) em um servidor Web indica uma conexão entre um usuário (usando um navegador na sua máquina) e o servidor.
- Além de guardar o estado da conexão, a sessão pode armazenar (enquanto o usuário permanece acessando o site) valores de variáveis específicas para esse usuário.
- Isso é interessante, pois em JSP as variáveis criadas em um arquivo não são acessíveis nos demais arquivos da aplicação web.
- **Por exemplo:** supondo que você possui um formulário para realizar o login do usuário no site. Após validar o acesso em um arquivo JSP, como fazer para “lembrar” nas outras páginas da aplicação que usuário já fez o login?

O que é uma sessão no servidor?

- Uma forma de fazer isso seria usar **cookies**, que são arquivos de texto que guardam valores **no navegador do usuário**.
- Mas existe uma outra maneira, mais poderosa e flexível: guardar as informações de login dentro da **sessão do usuário**.
- Dessa forma, em qualquer página JSP será possível recuperar esses valores, modificá-los e até mesmo atualizá-los dentro da sessão. Quando o usuário desconectar do site, a sessão e suas variáveis são automaticamente destruídas.
- Com esse recurso, resolvemos o problema de manter o estado do usuário no seu acesso entre várias páginas.
- Para acessar os dados da sessão (que são específicos para cada usuário conectado ao site), usamos o objeto JSP **session**.



Java Server Pages

Objetos implícitos:

- session (objeto da classe HttpSession)
- **session.setAttribute(String chave, Object valor);**
- Guarda na sessão o objeto passado no parâmetro **valor**, usando a String chave.
- Este objeto poderá então ser recuperado em outras páginas JSP na mesma sessão usando:
- **Object objeto = session.getAttribute(String chave);**
- Você pode **armazener diversas variáveis**, de diferentes tipos (incluindo vetores e objetos) na mesma sessão.



Java Server Pages

Objetos implícitos:

- session (objeto da classe HttpSession)
- Como qualquer objeto pode ser armazenado, deve-se fazer o casting para o tipo específico ao recuperar o mesmo:

```
String usuario = (String)session.getAttribute("usuario");  
int acessos =  
(Integer)session.getAttribute("acessos");
```



Java Server Pages

- **Objetos implícitos:**
- `session` (objeto da classe `HttpSession`)
 - Contém dados da sessão do usuário. Dados de sessão são específicos para cada usuário do site (simultâneos ou não)
 - **`session.isNew()`**
 - Retorna **`true`** se for uma nova sessão. Método interessante para ser usado na criação de objetos iniciais para a sessão:

```
if (session.isNew()) {  
    String login = "Visitante";  
    session.setAttribute("login", login);  
}
```

Exercícios

- **Contador:** faça uma aplicação web que conte quantas vezes o usuário carregou a página e exiba esse valor. Use uma variável de sessão do tipo **int** para guardar e atualizar a contagem de acessos.
- **Login:** monte um aplicativo Web com um formulário de login, com nome de usuário e senha (utilize o campo `<input type="password">` para a senha).
 - Crie um JSP para validar o login (utilize um **if** simples para testar se o usuário informou um login e senha corretos).
 - Caso o login esteja certo, guarde na **sessão** que o usuário está logado, assim como seu nome de usuário.
 - Crie uma outra página JSP. Nela, verifique pela sessão se o usuário está conectado. Se estiver, exiba uma mensagem de boas-vindas, junto do **seu nome de usuário**. **Se não estiver, mostre ACESSO NEGADO** em negrito e vermelho.

Exercícios

- **AdivinhaNumero**
- Sorteie um número entre 1 e 50. Guarde ele na sessão.
- Peça um palpite ao usuário.
- Se ele errou, indique se o número sorteado é maior ou menor que o informado pelo usuário. Armazene na sessão que o usuário fez uma nova tentativa (um contador do tipo `int` chamado `tentativas`). Mostre um “emoticon” triste e peça para ele tentar de novo.
- Se ele acertou, dê os parabéns! Mostre um “emoticon” feliz e exiba quantas vezes ele tentou até acertar. Mostre um link para ele jogar de novo. Para que ele possa jogar de novo, é preciso zerar a sessão, executando o método **`session.invalidate()`**;

Vetores em sessão

- Você também pode guardar **vetores** em uma sessão:

```
String[] produtos = new String[] {"Pen drive", "Impressora",  
"Tablet", "Teclado", "Mouse"};
```

- Para vetores de valores numéricos (**int** ou **double**), use a classe ao invés do tipo primitivo (**Integer** ou **Double**):

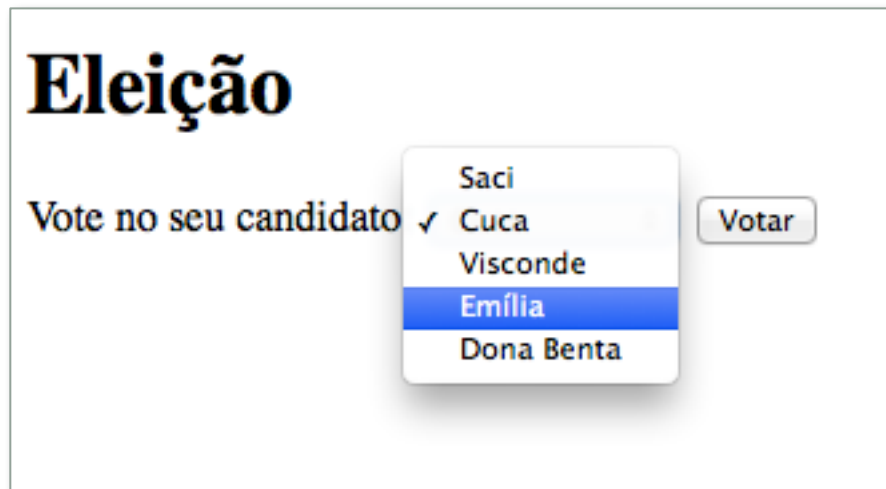
```
Integer[] estoque = new Integer[]{0, 5, 1, 2, 0};  
session.setAttribute("produtos", produtos);  
session.setAttribute("estoque", estoque);
```

- Para recuperá-los:

```
String[] produtos = (String[])session.getAttribute("produtos");  
Integer[] estoque = (Integer[])session.getAttribute("estoque");
```

Exercício - Eleição

- Crie uma aplicação web JSP para uma eleição.
- Na página inicial, use um formulário para que o usuário escolha entre cinco candidatos, usando um campo `<select>`:



Eleição

Vote no seu candidato

✓ Saci
Cuca
Visconde
Emília
Dona Benta

Votar

Exercício - Eleição

- Envie o voto para um arquivo JSP. Guarde um voto em uma variável de sessão contadora. DICA: use um vetor de **Integer[]**, para armazenar os votos de cada candidato.
- Exiba o resultado parcial, e um link para votar novamente:

Resultado	
Saci	2
Cuca	0
Visconde	1
Emília	1
Dona Benta	0
Novo voto	

BIBLIOGRAFIA

- HEFFELFINGER, D. W. Java EE 6 Development with NetBeans 7. Birmingham: Packt Publishing, 2011.
- PANDA, D.; RAHMAN, R.; LANE, D. EJB 3 in Action. Greenwich: Manning Publications, 2007.
- BASHAM, Bryan. Use A Cabeça! Servlets e JSP. Alta Books, 2008.
- KURNIAWAN, B. Java para Web com Servlets, JSP e EJB. São Paulo: Ciência Moderna, 2002.

