

JavaServer Pages (JSP) e Servlets

Parte 1: fundamentos

Fernando dos Santos



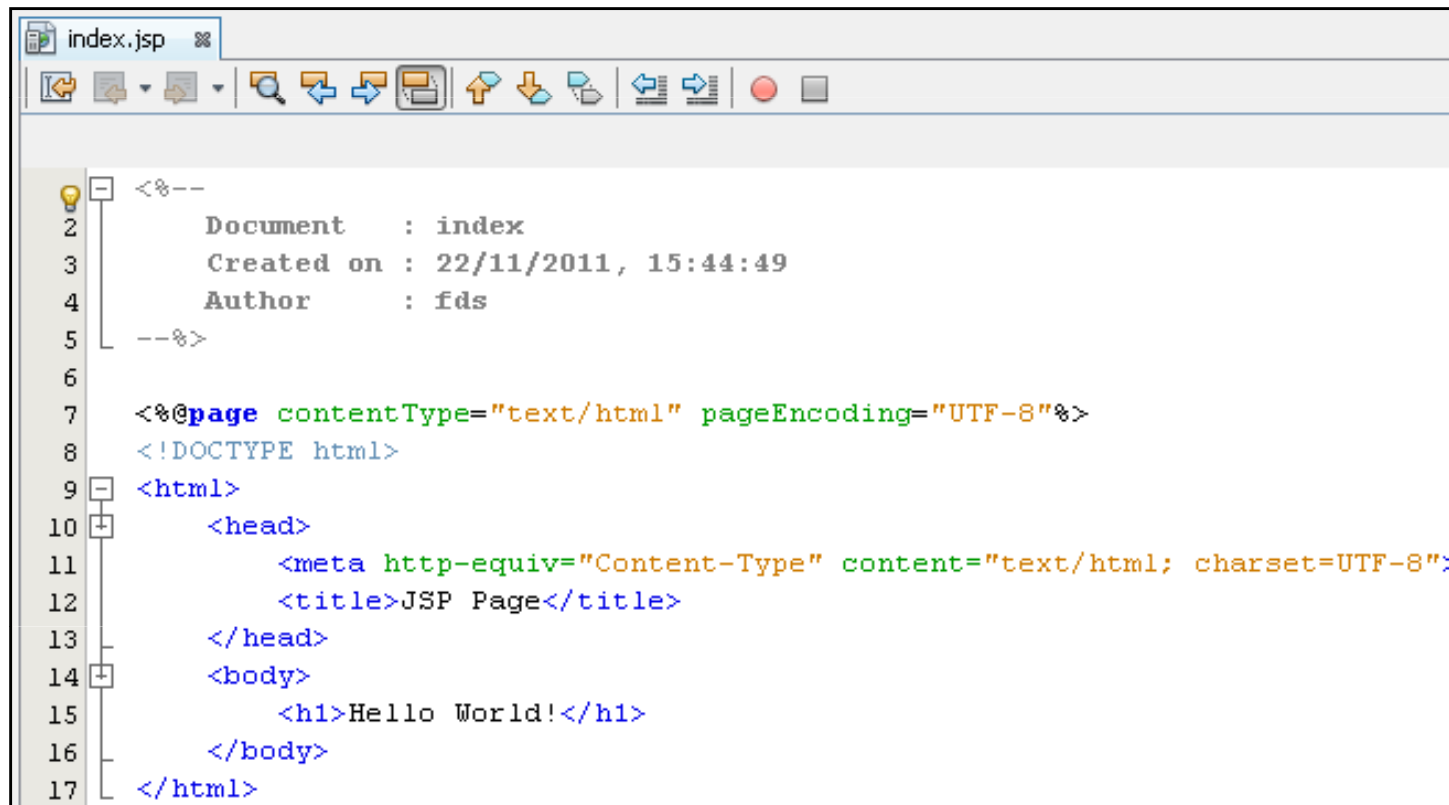
JavaServer Pages (JSP)

- JavaServer Pages (JSP) é uma tecnologia Java para desenvolvimento de páginas dinâmicas. Permite colocar trechos de código Java em uma página, para serem processados no servidor
- Uma página JSP contém basicamente dois tipos de elementos:
 - estáticos: correspondem a qualquer texto, como HTML ou XML;
 - dinâmicos: correspondem aos trechos de código Java, que constróem o conteúdo dinâmico.
- Nesta primeira parte, usaremos só elementos estáticos HTML



JSP: exemplo

- Qualquer elemento (tag) HTML pode ser inserida em um JSP.



```
index.jsp
1  <%--
2      Document      : index
3      Created on    : 22/11/2011, 15:44:49
4      Author       : fds
5  --%>
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10     <head>
11         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12         <title>JSP Page</title>
13     </head>
14     <body>
15         <h1>Hello World!</h1>
16     </body>
17 </html>
```

- Você utilizará as *tags* HTML que pesquisou ao realizar a atividade no AAGI 2



Exercício – Projeto ClinicaWeb

- Criar uma página JSP: cadastroPaciente.jsp
- Implementar os componentes HTML para preencher os dados do paciente, conforme exemplo abaixo:

The screenshot shows a Firefox browser window with the title 'Cadastro de Paciente'. The address bar shows 'localhost:8084/ClinicaWeb/cadastroPaciente.jsp'. The form is titled 'Informe os dados do paciente:' and contains the following fields:

CPF:	<input type="text"/>
Nome:	<input type="text"/>
Endereço:	<input type="text"/>
Sexo:	<input type="text"/>
Data de Nasc.:	<input type="text"/>
Telefone:	<input type="text"/>
Peso:	<input type="text"/>
Altura:	<input type="text"/>

At the bottom of the form is a 'Salvar' button.

Dicas:

- Os componentes devem ficar em um form HTML, com as seguintes configurações:
method = **“POST”**
action = **“/ClinicaWeb/ClinicaServlet/incluirPaciente”**
- Para alinhar os textos e inputs em forma de duas colunas, use um componente <table>
- **Cada input text deve ter um name**, conforme exemplo a seguir:

```
<input type="text" name="edCPF" />
```



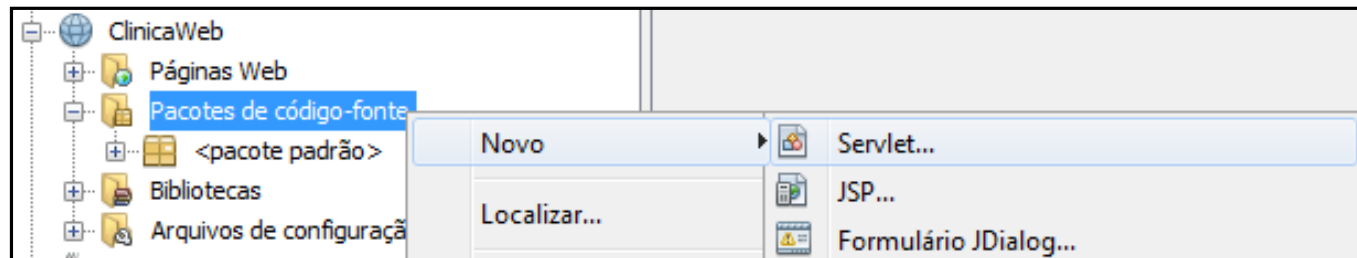
Servlet

- Servlet é um **programa Java**, que compõe uma **Aplicação Web**.
- É executado em um **Servidor de Aplicação Web**, que roda em uma **Máquina Servidora**.
- Servlet recebe e processa as requisições, gerando conteúdo dinâmico para a resposta.
- Desenvolveremos um servlet para receber os dados do paciente quando o usuário clicar em “Salvar” na tela de cadastro. Este servlet, de posse dos dados do paciente, deverá persistir o paciente no banco de dados.



Criação de um Servlet (1)

- O Servlet é uma classe Java que estende HttpServlet e implementa métodos para processamento de requisições.
- Para criar um servlet, botão direito sobre “Pacotes de código fonte” → **Novo** → **Servlet**



- Se a opção Servlet não aparecer, escolha a opção **Outro**. Na janela que abrir, escolha a categoria **Web** e o tipo de arquivo **Servlet**



Criação de um Servlet (2)

- Defina o nome da classe: **ClinicaServlet**

Novo Servlet

Passos

1. Escolha o tipo de arquivo
- 2. Nome e local**
3. Configurar implementação do servlet

Nome e local

Nome da classe:

Projeto:

Localização:

Paquete:

Arquivo criado:

⚠ Aviso: é altamente recomendado que você NÃO coloque classe:

< Voltar Próximo > Finalizar Cancelar Ajuda



Criação de um Servlet (3)

- Marque a opção:
 - “Adicionar informações ao descritor de implementação (web.xml)”
- Em padrões de URL, informe: **/ClinicaServlet/incluirPaciente**

Novo Servlet

Passos

1. Escolha o tipo de arquivo
2. Nome e local
3. **Configurar implementação**

Configurar implementação do servlet

Registre o servlet com o aplicativo atribuindo ao servlet um nome interno (Nome do servlet). Depois especifique os padrões que identifiquem as URLs que invocam o servlet. Separar múltiplos padrões com vírgulas.

☒ Adicionar informação ao descritor de implementação (web.xml)

Nome da classe:

Nome do servlet:

Padrão(ões) de URL:

Parâmetros de inicialização:

Nome	Valor
------	-------

< Voltar Próximo > **Finalizar** Cancelar Ajuda

Você notou a relação entre este padrão de URL e a action do form utilizado na página JSP?

É através dos padrões de URL que a requisição enviada pelo navegador é direcionada para o Servlet que possui aquele padrão de URL configurado.

Especifique novos padrões de URL (separados por virgula) quando quiser que o Servlet trate diferentes requisições (ex: editar, excluir, etc)



Criação de um Servlet (4)

- Ao finalizar, será criada a classe ClinicaServlet.
- É no método **processRequest** que será implementado o código que recupera os dados do paciente e insere no banco de dados.

```
ClinicaServlet.java
17 public class ClinicaServlet extends HttpServlet {
18
19     /**
20      * Processes requests for both HTTP GET and POST methods.
21      * @param request servlet request
22      * @param response servlet response
23      * @throws ServletException if a servlet-specific error occurs
24      * @throws IOException if an I/O error occurs
25      */
26     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
27         throws ServletException, IOException {
28         response.setContentType("text/html;charset=UTF-8");
29         PrintWriter out = response.getWriter();
30         try {
31             /* TODO output your page here
32              out.println("<html>");
33              out.println("<head>");
34              out.println("<title>Servlet ClinicaServlet</title>");
35              out.println("</head>");
36              out.println("<body>");
37              out.println("<h1>Servlet ClinicaServlet at " + request.getContextPath () + "</h1>");
38              out.println("</body>");
39              out.println("</html>");
40              */
41         } finally {
42             out.close();
43         }
44     }
```



Objetos Request e Response

- Observe que o método `processRequest` recebe dois parâmetros: `request` e `response`.



```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
```

- Estes objetos são criados pelo servidor (TomCat) e repassados ao Servlet, para que tenha acesso aos dados **recebidos** do navegador, e aos dados que serão **enviados** ao navegador.
- `request` é do tipo **HttpServletRequest**
 - contém os dados GET e POST enviados pelo navegador
 - parâmetros da URL
 - campos de formulário
- `response` é do tipo **HttpServletResponse**
 - contém a resposta que será devolvida para o navegador.
 - cabeçalho de página
 - conteúdo de página:
 - `PrintWriter out = response.getWriter();`
 - `out.println("Alo pagina dinamica");`



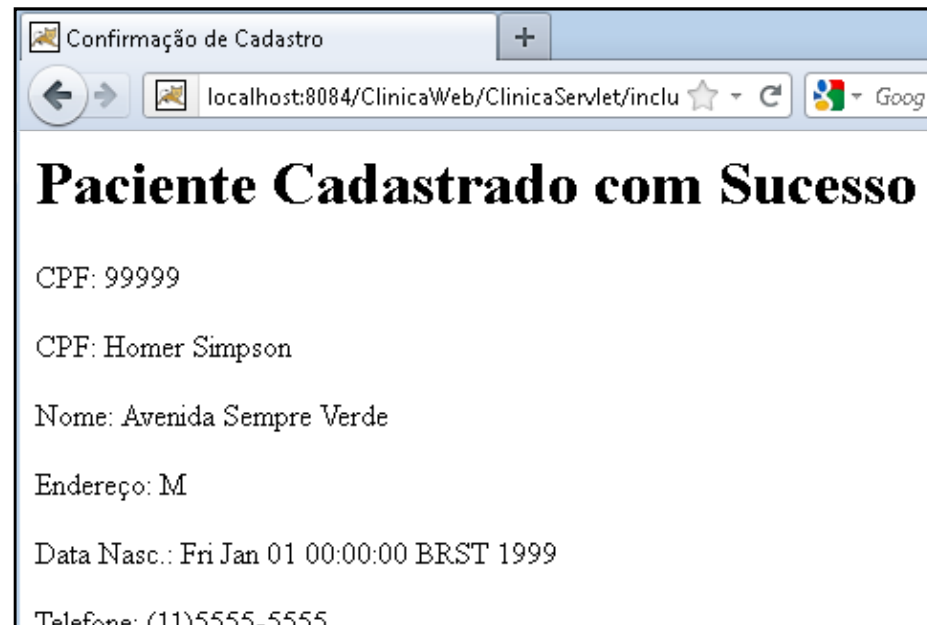
Recuperando dados no Servlet

- Todos os parâmetros (campos de formulário) enviados pelo navegador estão armazenados no objeto `HttpServletRequest`
- O valor de um parâmetro é recuperado através de seu nome. No caso de **input text**, o nome é definido na propriedade **name**:
 - `<input type="text" name="campo1"/>` // no JSP
 - `request.getParameter("campo1");` // no Servlet
- Todos os valores de parâmetros são tratados como `String`:
 - `String nome = request.getParameter("edNome");`
 - `String sobreNome = request.getParameter("edSobrenome");`
- Se necessário, pode-se converter o parâmetro para um tipo Java:
 - `int idade = Integer.parseInt(request.getParameter("edIdade"));`
 - `float salario = Float.parseFloat(request.getParameter("edSalario"));`



Exercício – Projeto ClinicaWeb

1. Implementar o método **processRequest** do ClinicaServlet para que ele faça o seguinte:
 - recuperar os dados do paciente a partir do objeto request
 - instanciar um objeto Paciente
 - setar os dados do paciente no objeto
 - salvar o paciente no banco de dados (JPA)
 - gerar uma resposta HTML para ser enviada ao navegador, utilizando o objeto response e o PrintWriter out, conforme exemplo abaixo:





Bibliografia

- TODD, N. **JavaServer pages :o guia do desenvolvedor**. Rio de Janeiro: Elsevier: 2003.
- BASHAM, Brian; SIERRA, Kathy; BATES, Bert. **Use a cabeça!:** Servlets & JSP. Rio de Janeiro : Alta Books, c2005. 534 p, il.
 - Recomendado! Tem na biblioteca do CEAVI.
- BERGSTEN, Hans. **JavaServer Pages**. Beijing : OReilly, 2001. xviii, 552p, il. (The Java series).