

FIA/P GRADUAÇÃO

SISTEMAS DE INFORMAÇÃO

MICROSERVICE AND WEB ENGINEERING

Prof. Dr. Emerson R. Abraham

| Agenda

Aula anterior:

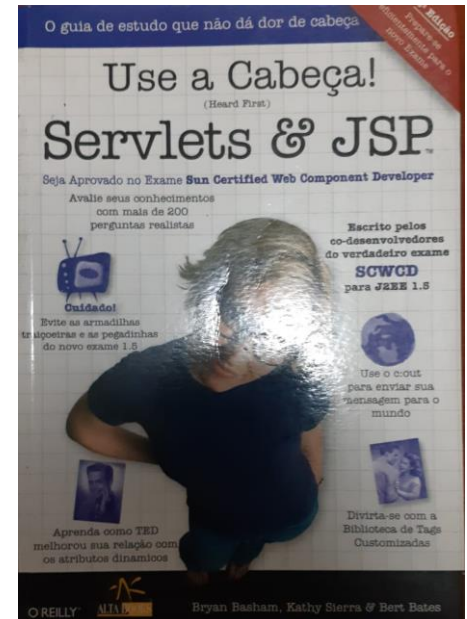
- Tomcat, o que é?
- Configuração e execução

Aula de Hoje:

- Servlet e seu ciclo de vida
- Java Server Pages
- Exercícios JSP e Servlet

Objetivo

- Entender o funcionamento básico do ciclo de vida das requisições (request) e respostas (response) em um projeto web utilizando Servlet e JSP.



Literatura recomendada!

| Servlet

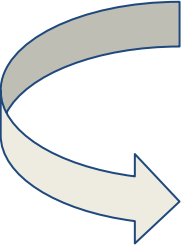
- **Servlet**

- A API Java Servlet (do pacote javax.servlet) proporciona ao desenvolvedor a possibilidade de adicionar conteúdo dinâmico em um servidor web usando a plataforma Java.
- Esta tecnologia disponibiliza ao programador da linguagem Java uma interface para o servidor web (ou servidor de aplicação), através de uma API. As aplicações baseadas no Servlet geram conteúdo dinâmico (normalmente **HTML**) e interagem com os clientes, utilizando o modelo **requisição-resposta**.
- Os servlets normalmente utilizam o protocolo **HTTP**, apesar de não serem restritos a ele. Um Servlet necessita de um Container Web para ser executado.

- **Ciclo de Vida e Métodos**


- **init(ServletConfig config)**

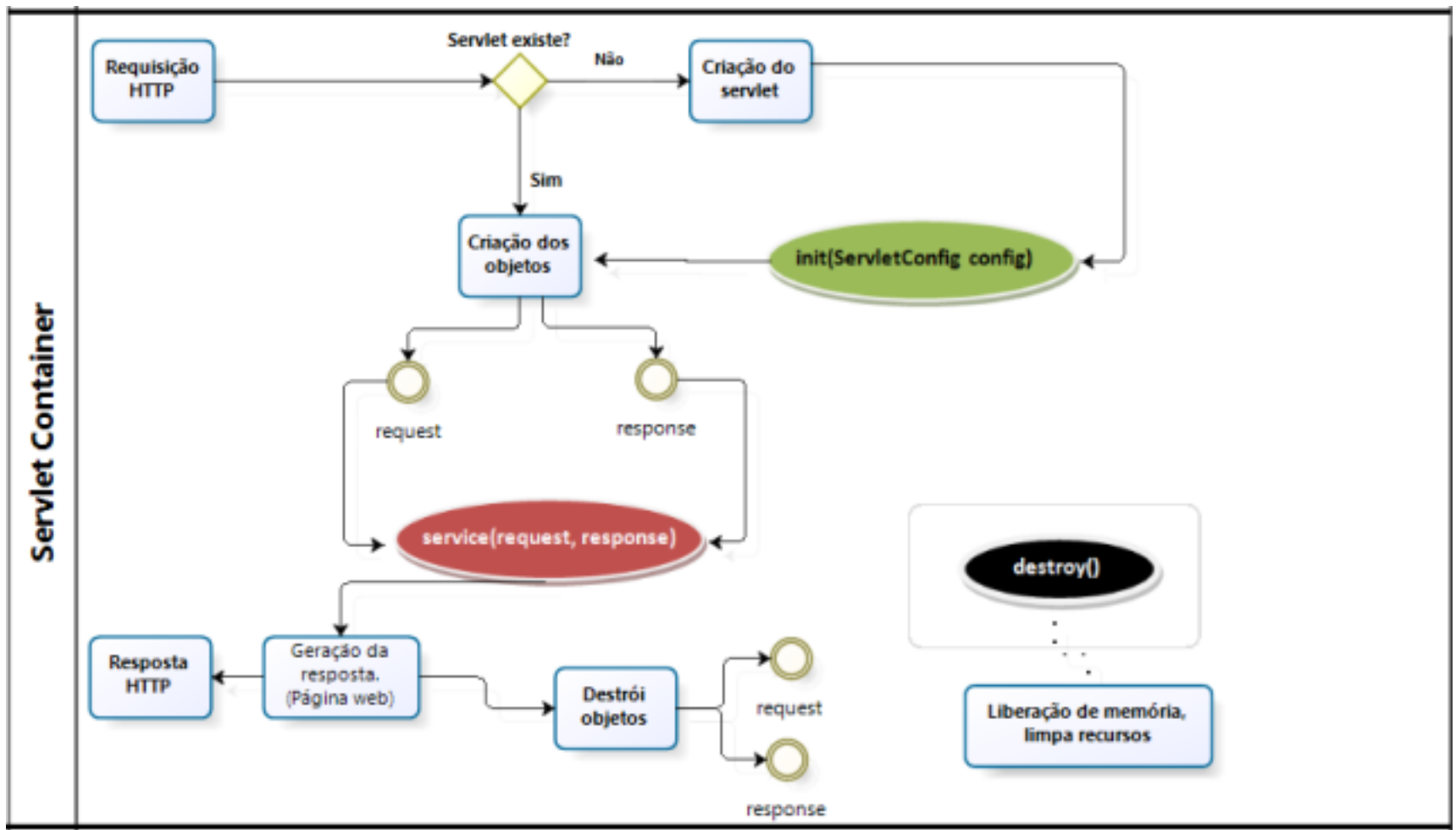
- **service(HttpServletRequest, HttpServletResponse)**

- 
- doGet(HttpServletRequest, HttpServletResponse)
 - doPost(HttpServletRequest, HttpServletResponse)
 - doPut(HttpServletRequest, HttpServletResponse)
 - delete(HttpServletRequest, HttpServletResponse)
 - demais

- **void destroy()**

Se o método service for declarado, os demais métodos não serão executados.





Servlet

- Estrutura do Servlet

```
public class Index extends HttpServlet {  
    public Index() {  
        super();  
    }  
    → public void init(ServletConfig config) throws ServletException {  
        System.out.println("Init");  
    }  
    public void destroy() {  
        System.out.println("destroy");  
    }  
    → protected void doGet(HttpServletRequest request, HttpServletResponse response) throws Servl  
        System.out.println("doGet");  
        response.getWriter().append("Served at: ").append(request.getContextPath());  
    }  
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws Serv  
        System.out.println("doPost");  
        doGet(request, response);  
    }
```


Configuração do Servlet (Servlet 2.5 ou menor)

– web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <display-name>projeto-jsp</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>Index</servlet-name>
    <servlet-class>br.com.fiap.projeto.Index</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>Index</servlet-name>
    <url-pattern>/Index</url-pattern>
  </servlet-mapping>

</web-app>
```

Todos os servlets criados na versão 2.4 ou inferior necessitam estar declarados no arquivo **web.xml** por meio de duas tags xml (servlet e servlet-mapping).

Configuração do Servlet (Servlet 3.0 ou superior)

– Annotation

➔ `import javax.servlet.annotation.WebServlet;`

➔ `@WebServlet("/Index2")`
`public class Index2 extends HttpServlet {`

Apache Tomcat version

Servlet	JSP	Apache
4.0	2.3	9.0.x
3.1	2.3	8.5.x
3.1	2.3	8.0.x
3.0	2.2	7.0.x
2.5	2.1	6.0.x



Java Server Pages (JSP)

- **JSP**

JavaServer Pages (JSP) é uma tecnologia que ajuda os desenvolvedores de software a criarem páginas web geradas dinamicamente baseadas em HTML, XML ou outros tipos de documentos. Lançada em 1999 pela Sun Microsystems, JSP é similar ao PHP, mas usa a linguagem de programação Java.



Java Server Pages (JSP)

- **Sintaxe scriptlet**

- Declaration Tag - criar variáveis e métodos

```
<%! private int contador= 0 ; %>
```

- Expression Tag - mostrar algum resultado

```
<%= contador %>
```

- Directive Tag – diretivas, libs, etc

```
<%@ include file="pagina.jsp" %>
```

- Scriptlet tag - código comum

```
<% for (int i=1; i<4; i++) { %>  
    <p>Este número é <%= i %>.</p>  
<% } %>
```

Java Server Pages (JSP)

- **Sintaxe Expression Language e JSTL**

```
<c:if test="${not empty usuario}">  
    Usuário ${usuario.nome} cadastrado com sucesso!  
</c:if>
```

- O objetivo da EL é simplificar o trabalho do programador, entretanto, não permite expressões lógicas, condicionais, laços de repetição, etc.
- JSTL é uma coleção de tags que visa suprir esta lacuna.

Exercício 1

Criar um formulario JSP

I Formulário JSP

- Insira um formulário com **nome**, **senha** e botão **enviar** na página **index.jsp**.
- O formulário da página **index.jsp** deverá ter sua propriedade **action** apontada para a página **home.jsp**.
- Crie um novo jsp com o nome **home.jsp** (pasta WebContent da aplicação).
- Na página **home.jsp** insira um **scriptlet** para validar se a senha digitada é igual a **123456** e imprimir uma mensagem de sucesso ou erro.

I Formulário JSP

Formulário `index.jsp`.

```
<form name="form" action="home.jsp" method="post">
  <label for="nome">Nome:</label>
  <input type="text" id="nome" name="nome" value="" size="30">

  <label for="senha">Senha:</label>
  <input type="password" id="senha" name="senha" value=""
size="20">

  <input type="submit" value="Entrar">
</form>
```

Formulário JSP

Página `home.jsp`.

```
<body>

Nome: <%= request.getParameter("nome") %> <br>

<%
String senha = request.getParameter("senha");
if ( senha.equals("123456") ) { %>
    <b> Login efetuado com sucesso.</b>
<% } else { %>
    <b> Senha inválida.</b>
<% } %>

</body>
```

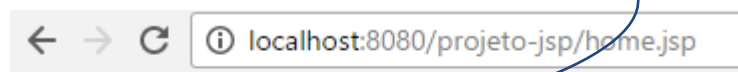
Formulário JSP

Executando a solução!



Projeto JSP

Nome: Senha:



Nome: teste
Senha inválida.



Nome: teste
Login efetuado com sucesso.

Exercício 2

Criar um servlet para o formulário

Formulário Servlet

- **Criar a classe Index.java e configurá-la para ser um servlet:**

```
@WebServlet("/index")
```

```
public class Index extends HttpServlet
```

- **Criar o método doPost e realizar o redirect para a home:**

```
request.getRequestDispatcher("home").forward(request, response);
```

- Alterar o action do formulário index.jsp para acessar o servlet index

```
<form name="form" action="index" method="post">
```

- **Criar a classe Home.java e configurá-la para ser um servlet**
- **Criar o método doPost e fazer a lógica de validação de senha que antes estava no JSP**

Formulário Servlet

- **Escrever o HTML de resposta no método doPost:**

```
String nome = request.getParameter("nome");  
String senha = request.getParameter("senha");  
String mensagem = "Logado com sucesso!";
```

```
if (!"123456".equals(senha)) {mensagem = "Login inválido";}
```

```
response.setContentType("text/html");  
PrintWriter out = response.getWriter();  
out.println("<html>");  
out.println("<head>");  
out.println("<meta http-equiv='Content-Type' content='text/html; charset=ISO-8859-1'>");  
out.println("<title>Insert title here</title>");  
out.println("</head>");  
out.println("<body>");  
out.println(" Nome: " + nome + " <br>");  
out.println(" <b> " + mensagem + "</b>");  
out.println("</body>");  
out.println("</html>");
```



| Homework

1 – Rever os exercícios de Servlet e JSP



Referências

Apache Tomcat disponível em: <http://tomcat.apache.org/>

Profº. Pedro Ivo Correia

Profº. Lucas Furlaneto

Copyright © 2021

Todos direitos reservados. Reprodução ou divulgação total / parcial deste documento é proibido sem o consentimento formal.