

Trabalhando com Servlets

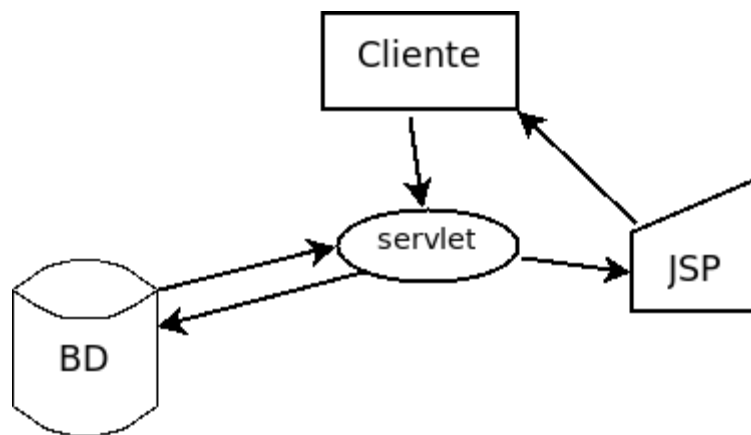
O que é uma Página JSP?

- Tem a forma de uma página HTML com trechos de código Java embutidos e outras tags especiais.
 - A parte dinâmica da página é gerada pelo código Java
- Simplificam a geração de conteúdo dinâmico para Web Designers
- Uma JSP é automaticamente transformada em servlet
- Dois formatos de JSP: padrão ou formato XML

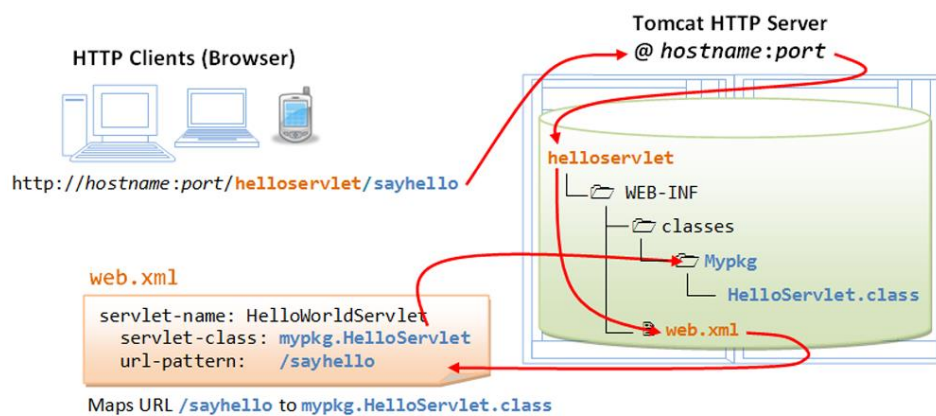
Servlet x JSP

- Servlets
 - Melhor em termos de Eng. Software
 - Mais flexível
 - Não permite independência entre o designer e o programador
- JSP
 - Bem mais fácil de aprender!!!!
 - Um nível maior de abstração pro Servlets
 - O Web Designer pode trabalhar independente do Web Developer e vice-versa

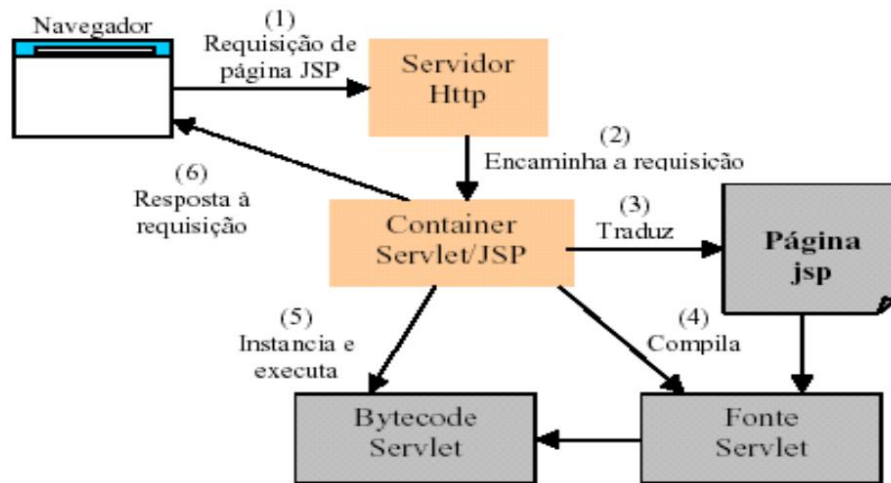
Comportamento de uma página JSP



Estrutura de uma Aplicação Web



Como funciona uma página jsp fazendo a requisição das informações que por sua vez são executadas em containers (onde executam suas aplicações).



Problemas de Servlets

- Servlets forçam o programador a embutido no código HTML dentro de código Java
- Desvantagem: se a maior parte do que tem que ser gerado é texto ou código HTML estático.
- Mistura as coisas: programador tem que ser bom Web Designer e se virar sem ferramentas de Web Design

O desafio tanto para designers e programadores, os dois códigos juntos a serem interpretados na codificação....

```
Date hoje = new Date();
out.println("<body>");
out.println("<p>A data de hoje é "+hoje+".</p>");
out.println("<body>");
HojeServlet.java
```

- Uma solução inteligente é escrever um arquivo de *template*

```
<body>
<p>A data de hoje é <!--#data#-->.</p>
<body>
template.html
```

Podemos dizer então

JSP é uma tecnologia padrão, baseada em templates para servlets.

O mecanismo que a traduz é embutido no servidor

O que são JavaServer Pages?

- Solução do problema anterior usando *templates* JSP

```
<body>
<p>A data de hoje é <%=new Date() %>.</p>
<body>
hoje.jsp
```

- Em um servidor que suporta JSP, processamento de JSP passa por uma camada adicional onde a página é transformada (compilada) em um servlet
- Acesso via URL usa como localizador a própria página

Exemplos de JSP

- A forma mais simples de criar documentos JSP, é
 - 1. Mudar a extensão de um arquivo HTML para .jsp
 - 2. Colocar o documento em um servidor que suporte JSP
- Fazendo isto, a página será transformada em um servlet
 - A compilação é feita no primeiro acesso
 - Nos acessos subsequentes, a requisição é redirecionada ao servlet que foi gerado a página da página

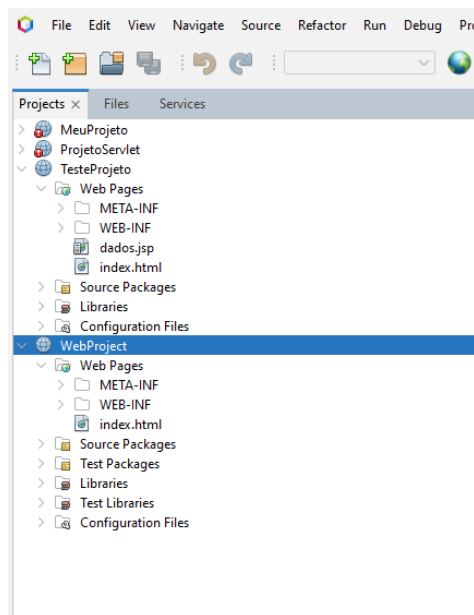
- Transformado em um JSP, um arquivo HTML pode conter blocos de código (scriptlets): `<% ... %>` e expressões `<%= ... %>`

Ciclo de Vida

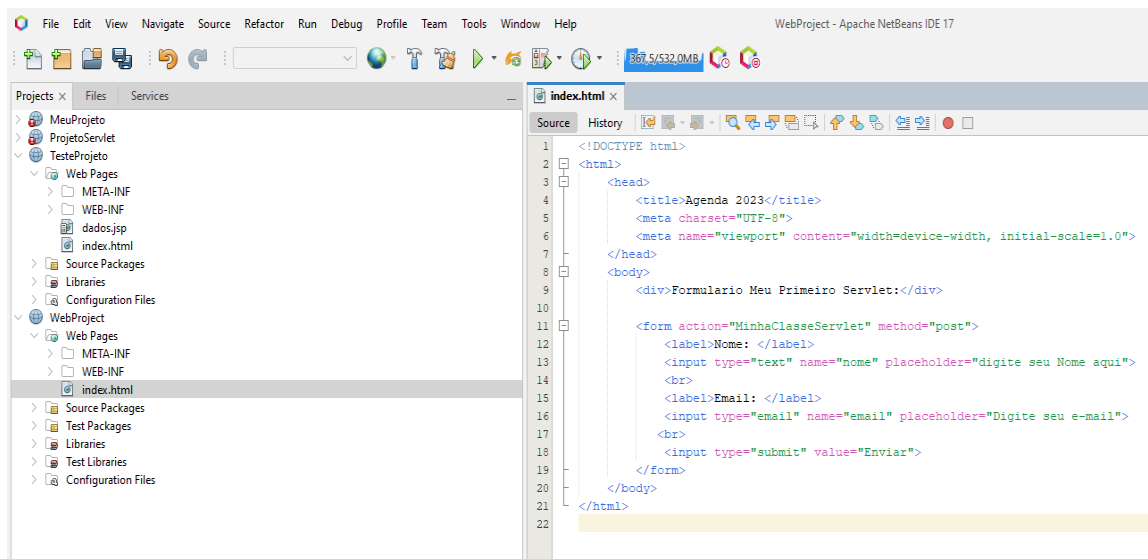
1. Quando uma requisição é mapeada em uma JSP, o container
 - i. Verifica se o servlet correspondente à página é mais antigo que a página (ou se não existe)
 - ii. Se o servlet não existe ou é mais antigo, a página JSP será compilada para gerar novo servlet
 - iii. Com o servlet atualizado, a requisição é redirecionada para ele
2. Deste ponto em diante, o comportamento **equivale** ao ciclo de vida do **servlet**, mas os métodos são diferentes
 - i. Se o servlet ainda não estiver na memória, ele é instanciado, carregado e seu método `jspInit()` é chamado
 - ii. Para cada requisição, seu método `_jspService(req, res)` é chamado. Ele é resultado da compilação do corpo da página JSP
 - iii. No fim da vida, o método `jspDestroy()` é chamado

Minha Primeira Classe Servlet.

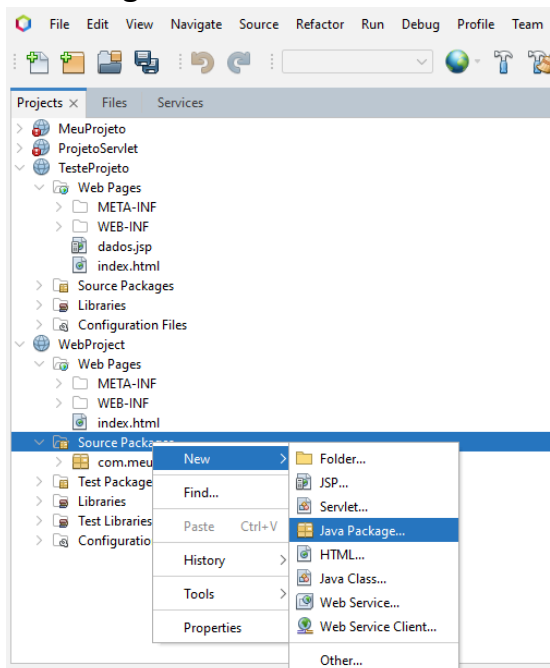
- 1) Primeiro Passo Criar uma Aplicação Web com o nome WebProject.
- 2) File – New Project – Web Application.



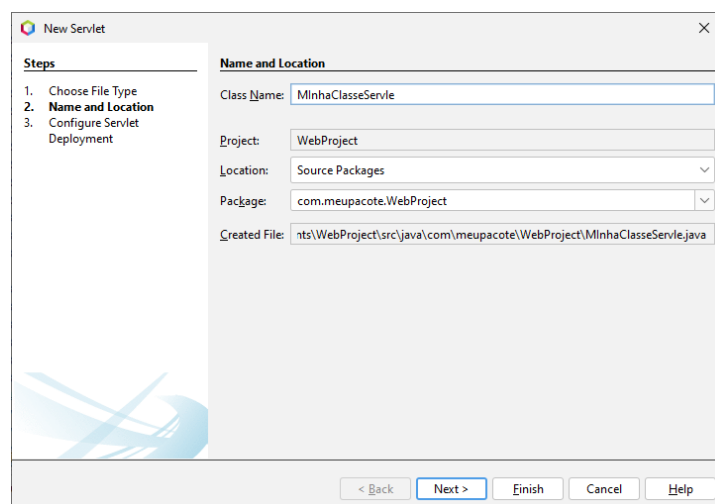
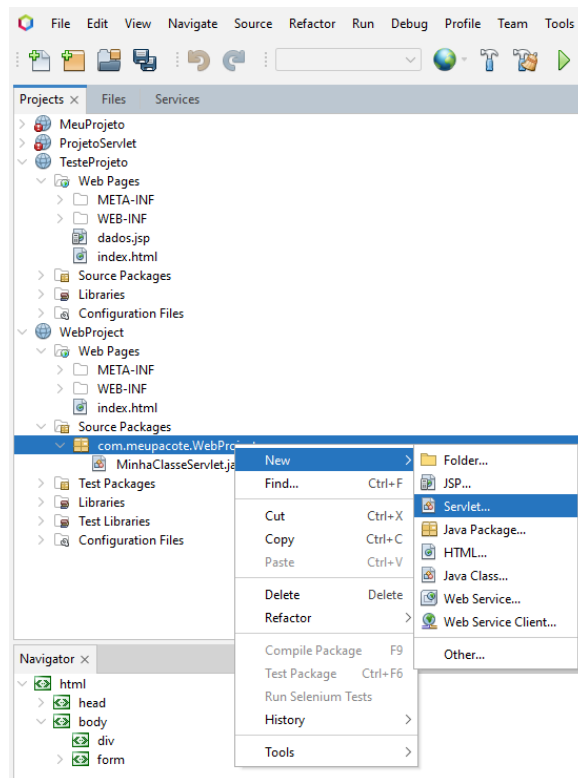
- 3) Ao criar o projeto a estrutura já conta uma pagina html simples, podemos utilizá-la ou não , mas neste modelo faremos que esta página redirecione para um servlet.



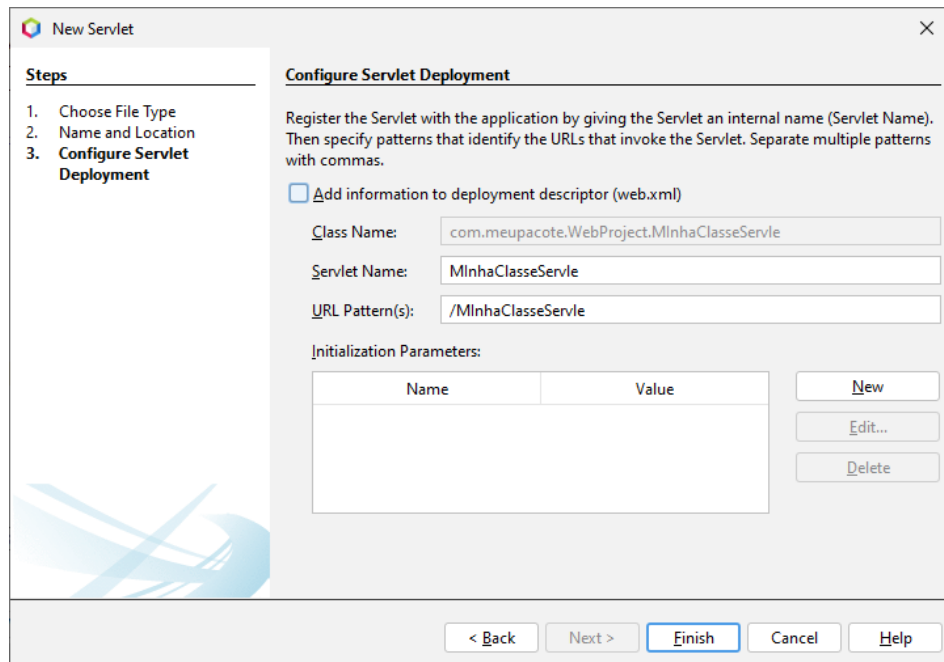
- 4) Crie um formulário simples com dois campos nome e email, e utilize o método post para envio das informações para a outra página. E no action de seu formulário o nome deverá ser **MinhaClasseServlet**.
- 5) No diretório Source Packages – botão direito – novo –



- 6) O nome do pacote deverá ser com.meupacote.WebProject.
- 7) Seguindo o modelo anterior botão direito em cima do pacote criado e devera ser criada a classe com o nome MinhaClasseServlet.

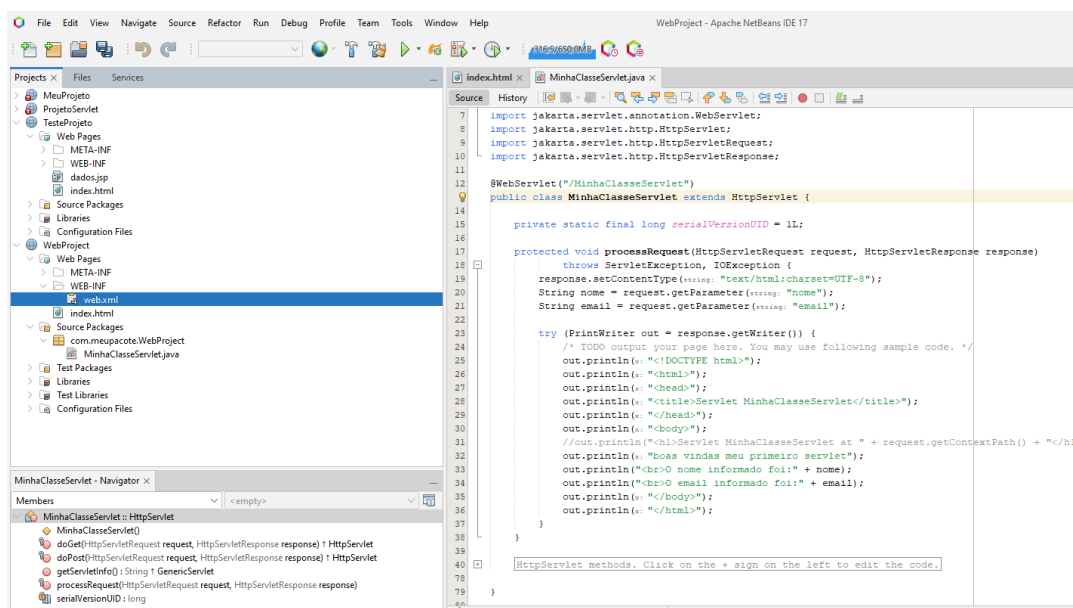


Observe que o pacote devera ser incluso na classe..que esta sendo criada.



O nome da classe servlet e url /MinhaClasseServlet... e finish

8) Observe que a Classe é criada automaticamente.



9) Iremos acrescentar alguns parâmetros abaixo como a passagem de parâmetros dos campos nome e email..para uma variável string. Logo em seguida utilize um try.. catch para incluir o códigos html..de retorno..do servlet..


```

import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

@WebServlet("/MinhaClasseServlet")
public class MinhaClasseServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        String nome = request.getParameter("nome");
        String email = request.getParameter("email");

        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet MinhaClasseServlet</title>");
            out.println("</head>");
            out.println("<body>");
            //out.println("<h1>Servlet MinhaClasseServlet at " + request.getContextPath() + "</h1>");
            out.println("<br>boas vindas meu primeiro servlet");
            out.println("<br>O nome informado foi: " + nome);
            out.println("<br>O email informado foi: " + email);
            out.println("</body>");
            out.println("</html>");
        }
    }
}

```

- 10) Feita esta etapa estamos pronto para mapear nossa estrutura do servlet pelo arquivo xml.
- 11) O arquivo web.xml ira mapear nossa estrutura do projeto.

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="6.0" xmlns="https://jakarta.ee/xml/ns/jakartaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee
        https://jakarta.ee/xml/ns/jakartaee/web-app 6.0.xsd">

    <servlet>
        <servlet-name>MinhaClasseServlet</servlet-name>
        <servlet-class>com.meupacote.WebProject.MinhaClasseServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>MinhaClasseServlet</servlet-name>
        <url-pattern>/meuservlet</url-pattern>
    </servlet-mapping>
</web-app>

```

Ficando assim dentro do xml..na parte de cima não altera nada inclua somente o mapeamento sendo pelo:

servlet-name(Nomeda Classe)
servlet-class: onde esta pacote.nomeprojeto.nomeclasse
dentro do servlet-mapping:
servlet-name(Nomeda Classe)..novamente.
url-pattern: /nome ou apelido a ser chamado o projeto.

<servlet>

<servlet-name>MinhaClasseServlet</servlet-name>

<servlet-

class>com.meupacote.WebProject.MinhaClasseServlet</servlet-class>

</servlet>

<servlet-mapping>

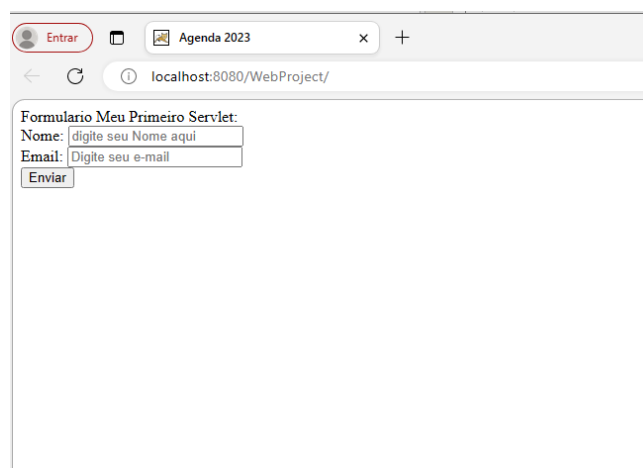
<servlet-name>MinhaClasseServlet</servlet-name>

<url-pattern>/meuservlet</url-pattern>

</servlet-mapping>

Pronto agora está criado nosso primeiro servlet.

Vamos compilar e logo em seguida no navegador encontraremos:



Entrar Agenda 2023 x +

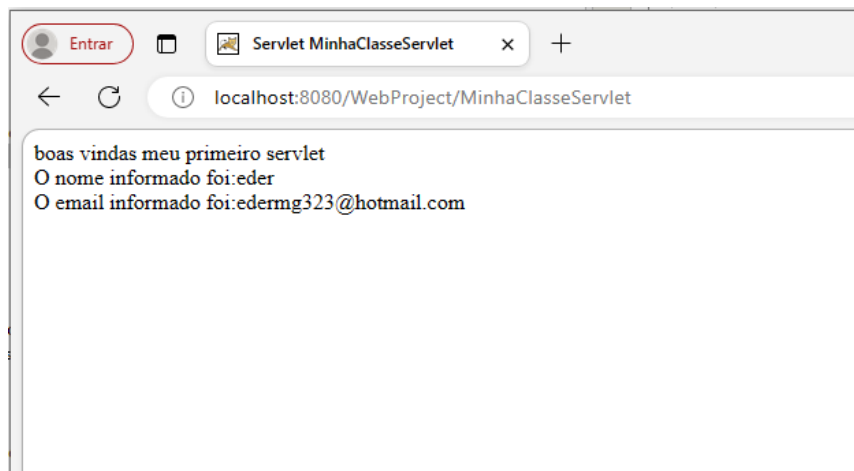
localhost:8080/WebProject/

Formulario Meu Primeiro Servlet:

Nome:

Email:

**Preencha os dados e clique em enviar a pagina seguinte irá processar
nosso servlet as informações retornando assim:**



Parabéns seu primeiro Servlet, foi criado com sucesso..