

# **Linguagem de Programação 2**

## **Aula 1**

### **Desenvolvimento de Aplicações Web JavaServer Pages (JSP)**

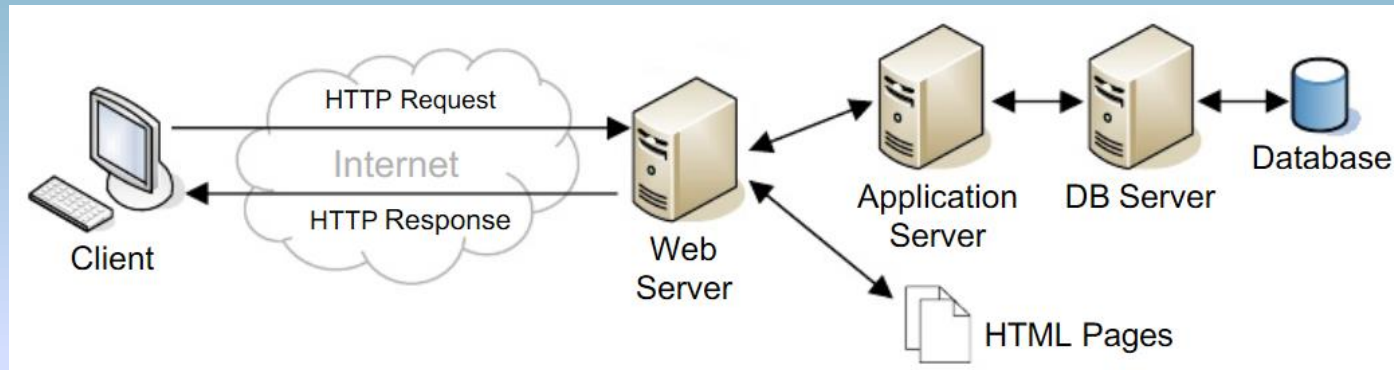
[l.bertholdo@ifsp.edu.br](mailto:l.bertholdo@ifsp.edu.br)

# Conteúdo

- Desenvolvimento de Aplicações Web
- Ambiente de Desenvolvimento Web
- JavaServer Pages (JSP)
- Elementos Sintáticos do JSP
- Exemplo – Aplicação JSP
- Formulários
- Recuperação de Dados
- Exemplo – Aplicação JSP com Formulário

# Desenvolvimento de Aplicações Web

- Normalmente, aplicações web são desenvolvidas sob uma arquitetura **cliente-servidor**, sendo o lado servidor dividido em diferentes níveis.



**1º Nível:** O **servidor Web** é responsável por gerenciar as páginas HTML (Hypertext Transfer Protocol) armazenadas e apresentá-las no navegador do cliente quando requisitadas.

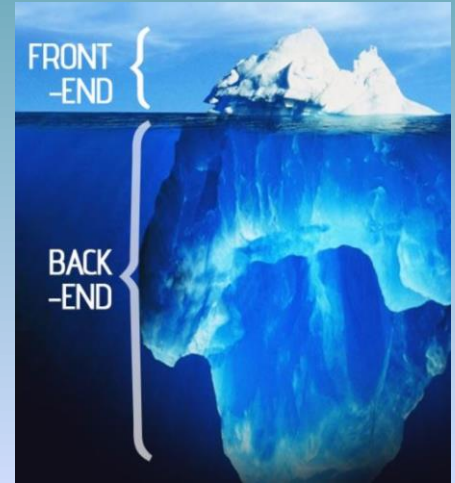
**2º Nível:** Se o cliente solicitar um serviço a um sistema corporativo (consulta aos dados dos clientes inadimplentes, por exemplo), a solicitação é repassada a um **servidor de aplicação**.

**3º Nível:** O **servidor de banco de dados** acessa diretamente o banco de dados do sistema para repassar ao servidor de aplicação os dados solicitados.

- Nesse contexto, o **desenvolvimento** da aplicação pode ser classificado em dois tipos: **Client-Side** (lado cliente) e **Server-Side** (lado servidor).

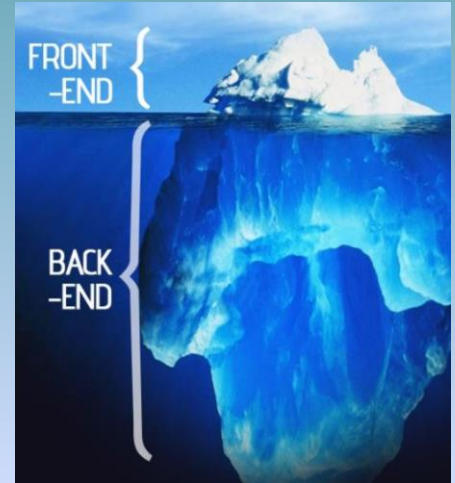
# Desenvolvimento de Aplicações Web

- Desenvolvimento *Client-Side* (ou *Front-End*)
  - As linguagens utilizadas são interpretadas e executadas usando os recursos computacionais do navegador do usuário.
  - Estas linguagens tem o objetivo de estruturar, formatar e estilizar a interface de usuário, além de fornecer funções básicas que permitam a interação do usuário com a aplicação.
  - No desenvolvimento *front-end*, a construção das interfaces de usuário leva em conta questões básicas como estética, validações de dados e segurança, mas também usabilidade, experiência do usuário, acessibilidade e comunicabilidade.



# Desenvolvimento de Aplicações Web

- Desenvolvimento *Server-Side* (ou *Back-End*)
  - As linguagens de programação utilizadas são executadas nos servidores.
  - São linguagens que têm o objetivo de prover à aplicação funções mais complexas, que demandam maior capacidade, robustez e segurança durante o processamento, como consultas a bancos de dados e cálculos matemáticos pesados, por exemplo.
  - O foco do desenvolvimento *back-end* está essencialmente na implementação da lógica de negócios e do modelo de dados da aplicação, como requisitos de sistema, regras de negócio, estrutura de classes e de bancos de dados.



# Desenvolvimento de Aplicações Web

- Linguagens

**Linguagem de marcação** utilizada para definir a estrutura de páginas de hipertexto, por meio de *tags*, que servem de *containers* para os elementos da página (textos, imagens, botões, campos etc).

## Linguagens do Lado Cliente (front-end)



**Cascading Style Sheets (CSS)** é uma **linguagem de estilização** cujo objetivo é separar os elementos de apresentação dos elementos de estruturação em uma página de hipertexto.

## Linguagens de Programação do Lado Servidor (back-end)



**Linguagem de script/programação** usada para executar *scripts* e adicionar características dinâmicas e interatividade às páginas. Atualmente, é usada tanto no desenvolvimento *front-end* como no *back-end*.

# Desenvolvimento de Aplicações Web

- Além de **linguagens** e **navegadores** para executar e depurar seus códigos, um desenvolvedor web necessita de diversas **ferramentas** para auxiliar na construção de uma aplicação.

## **IDEs e Editores de Código**

As **IDEs** são usadas principalmente na programação **back-end** e incluem, além de editor de código-fonte, compilador, depurador e outros recursos. Exemplos: Eclipse, IntelliJ IDEA, Visual Studio. Já os **editores** são voltados para o **front-end** e visam facilitar a escrita de código. Exemplos: Atom, Sublime Text, Notepad++.

## **Frameworks de Estilização**

Também conhecidos como "Frameworks CSS" são pacotes de código padronizado que fornecem uma base para construir e estilizar uma aplicação, melhorando a interatividade e a estética do produto final. Exemplos: Bootstrap, Foundation, Materialize.

## **Frameworks Web App**

São usados para simplificar o desenvolvimento das camadas lógica, de negócios e de dados de aplicações web. Normalmente, fornecem recursos para criação de funções básicas em aplicações, como operações de CRUD. Exemplos: Spring, Hibernate, ASP.NET, Entity Framework, Laravel, Angular, Django, Ruby on Rails.

## **Terminais de Linha de Comando**

Algumas ferramentas usadas durante o desenvolvimento, como o gerenciador de versões Git, fazem uso de terminais de linha de comando como o Prompt de Comando ou o PowerShell do Windows, e o Bash do Linux.

# Ambiente de Desenvolvimento Web

- O ambiente de desenvolvimento Web para a linguagem Java requer três componentes básicos:
  - **JDK (Java Development Kit)** – Conjunto de ferramentas para o desenvolvimento de aplicações em Java, que inclui o compilador Java e a máquina virtual (*Java Runtime Environment* – JRE).
  - **Apache Tomcat** – Servidor Web e *container* de *servlets* padrão usado para executar aplicações Web escritas em Java.
  - **IDE (*Integrated Development Environment* ou Ambiente Integrado de Desenvolvimento)** – Software que reúne ferramentas para programação de aplicações, além de recursos para configuração e gerenciamento de projetos. **Exemplos:** Eclipse, IntelliJ IDEA, NetBeans e Visual Studio Code.



# Ambiente de Desenvolvimento Web

- Download e Instalação do Java Development Kit (JDK)  
<https://www.oracle.com/java/technologies/downloads>

## Java 22, Java 21, and Java 17 available now

JDK 21 is the latest long-term support release of Java SE Platform.

[Learn about Java SE Subscription](#)

[JDK 22](#)   [JDK 21](#)   [JDK 17](#)   [GraalVM for JDK 22](#)   [GraalVM for JDK 21](#)   [GraalVM for JDK 17](#)

## JDK Development Kit 22.0.2 downloads

JDK 22 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions \(NFTC\)](#).

JDK 22 will receive updates under these terms, until September 2024, when it will be superseded by JDK 23.

[Linux](#)   [macOS](#)   [Windows](#)

Product/file description	File size	Download
x64 Compressed Archive	184.16 MB	<a href="https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.zip">https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.zip</a> (sha256)
x64 Installer	164.35 MB	<a href="https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.exe">https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.exe</a> (sha256)
x64 MSI Installer	163.09 MB	<a href="https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.msi">https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.msi</a> (sha256)

# Ambiente de Desenvolvimento Web

- Configuração de variáveis de ambiente (após instalação do JDK)\*
  1. Acesse **Computador >> Propriedades >> Configurações Avançadas do Sistema**.
  2. Na aba **Avançado**, clique em **Variáveis de Ambiente**.
  3. Em **Variáveis do Sistema**, clique em **Novo....** Em **Nome da Variável**, digite "JAVA\_HOME" e em **Valor da Variável**, digite o caminho onde o JDK está instalado. Por exemplo: "C:\Program Files\Java\jdk1.8.0\_241".
  4. Na mesma janela, em **Variáveis de Usuário**, clique em **Novo....** Em **Nome da Variável**, digite "PATH" e em **Valor da Variável**, digite "%JAVA\_HOME%\bin". Se a variável PATH já existir, clique em **Editar...** e inclua este novo caminho. Clique em **OK** para fechar as janelas.
  5. Abra um prompt de comando e execute o comando "javac -version" e verifique a versão do JDK apresentada. No exemplo acima, será apresentado "javac 1.8.0\_241".

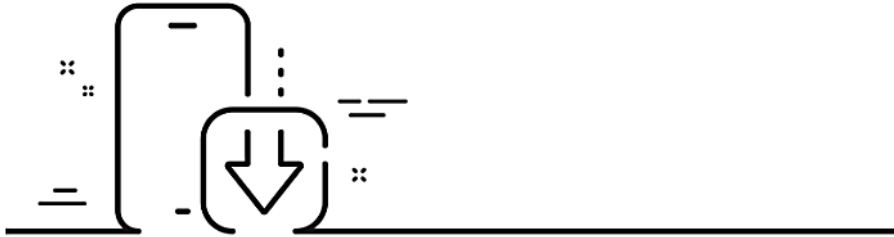
```
C:\Users\Leonardo>javac -version
javac 1.8.0_241
C:\Users\Leonardo>_
```

\* No ambiente Windows.

# Ambiente de Desenvolvimento Web

- Download e Instalação da IDE Eclipse

<http://www.eclipse.org/downloads>



OCX 2024: Early Bird Pricing Ends Soon!

The Eclipse Foundation's flagship developer conference is coming soon, featuring collocated events for Java, and more. Register by 23 September for the best price!



Install your favorite desktop IDE packages

[Learn More](#)

[Download x86\\_64](#)

[Download Packages](#)

[Need Help](#)

Após baixar o instalador, execute-o como Administrador.

Download Eclipse Technology that is right for you

Eclipse IDE 2024-09 R Packages



Eclipse IDE for Java Developers

331 MB 579 DOWNLOADS

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration



Windows x86\_64  
macOS x86\_64 | AArch64  
Linux x86\_64 | AArch64



Eclipse IDE for Enterprise Java and Web Developers

530 MB 333 DOWNLOADS

Tools for developers working with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer Pages and Faces, Yaml, Markdown, Web Services, JPA and Data Tools, Maven and Gradle, Git, and more.



Windows x86\_64  
macOS x86\_64 | AArch64  
Linux x86\_64 | AArch64

[Click here to raise an issue with the Eclipse Web Tools Platform. Maintainers will move opened issues to the right place.](#)

[Click here to raise an issue with the Eclipse Platform.](#)

[Click here to raise an issue with Maven integration for web projects.](#)

[Click here to raise an issue with Eclipse Wild Web Developer \(incubating\).](#)

TEMURIN  
by ADOPTIUM

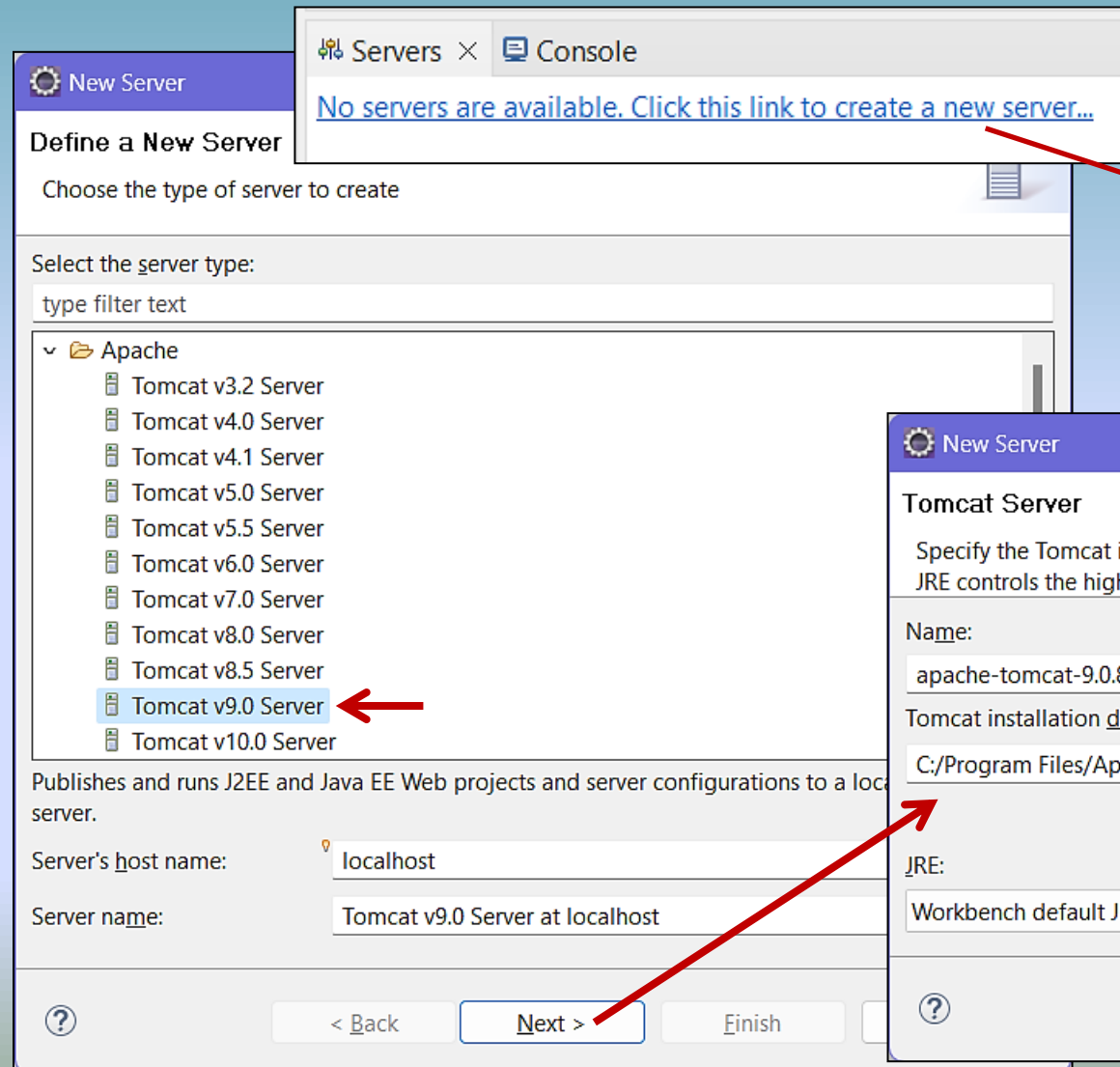
The Eclipse Temurin™ project provides high-quality, TCK certified OpenJDK runtimes and associated technology for use across the Java™ ecosystem.

[Learn More](#)

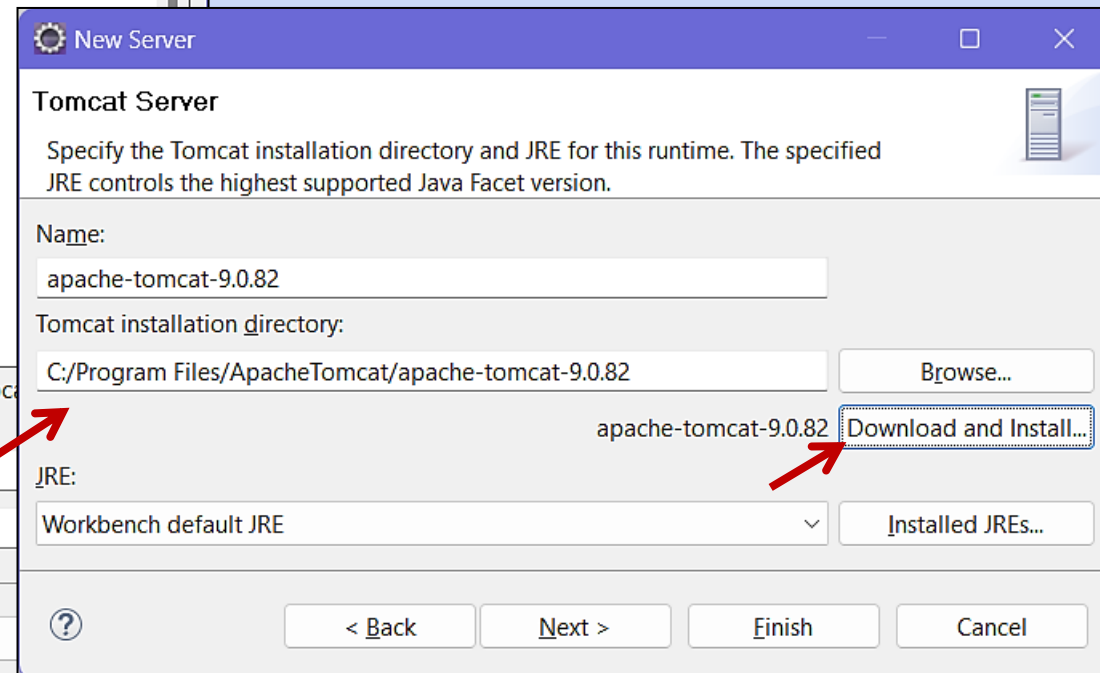
[Download](#)

# Ambiente de Desenvolvimento Web

- Download e Instalação do Apache Tomcat



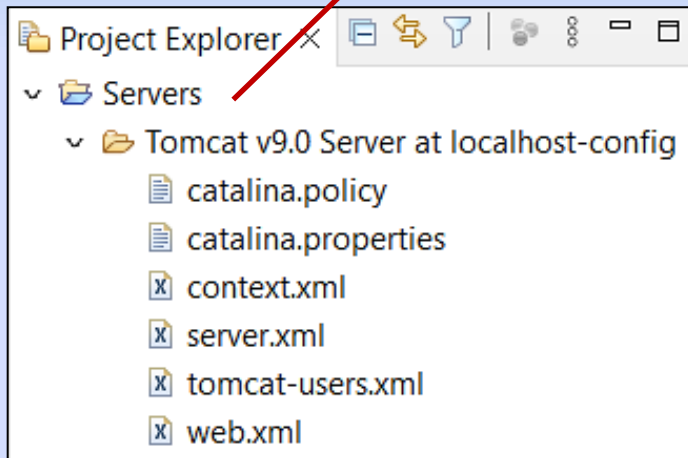
No Eclipse, habilite a *view Servers* (menu **Window >> Show View >> Servers**). Na *view* aberta, clique no link **No servers....** Selecione uma versão de servidor Tomcat e clique em **Next**. Na janela aberta, clique em **Download and Install**, aceite os termos e selecione o diretório de instalação. Nesse exemplo, ele sugeriu a versão 9.0.82. Clique em **Finish**.



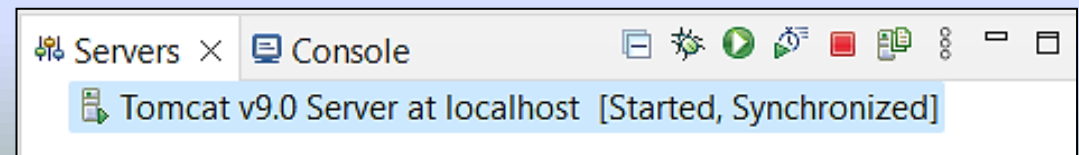
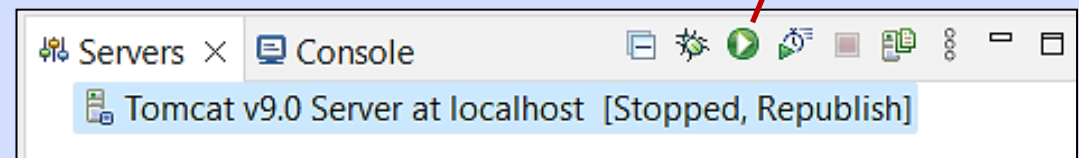
# Ambiente de Desenvolvimento Web

- Download e Instalação do Apache Tomcat

Após instalar o **Apache Tomcat**, o explorador de projetos exibirá um projeto chamado **Servers**, que contém os arquivos de configuração do Tomcat, necessários para que o servidor execute as aplicações Web a serem criadas.



Na view **Servers**, inicie o servidor Tomcat instalado clicando neste botão.



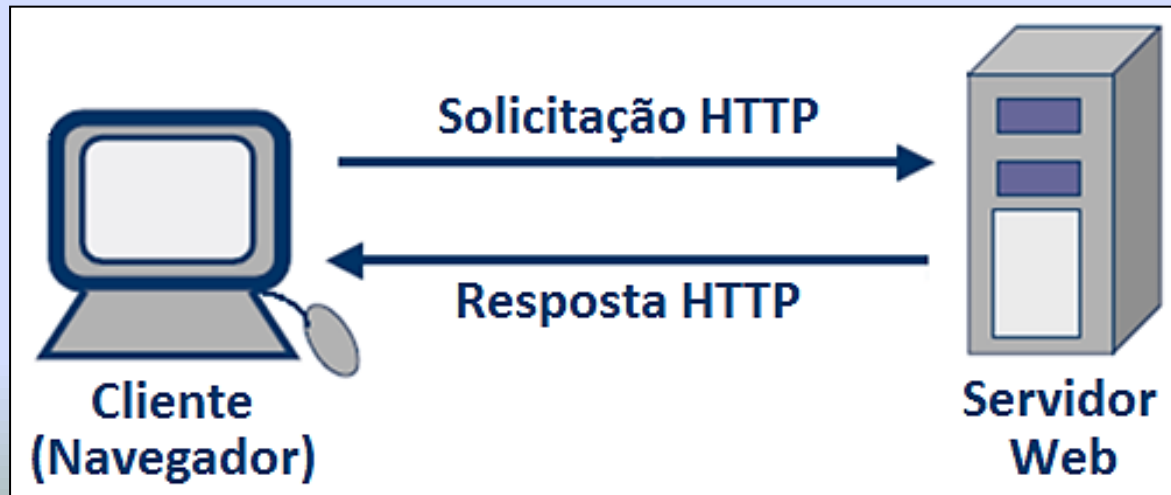
# JavaServer Pages



- Tecnologia desenvolvida pela Sun Microsystems em meados dos anos 1990, voltada ao desenvolvimento de aplicações baseadas na linguagem de programação Java.
- Para gerar páginas de forma dinâmica, a tecnologia se utiliza de **servidores Web** e **servidores de aplicação**.
- Páginas escritas com código JSP nada mais são do que páginas HTML que contêm códigos escritos em **Java**, os quais definem a lógica de apresentação da aplicação.

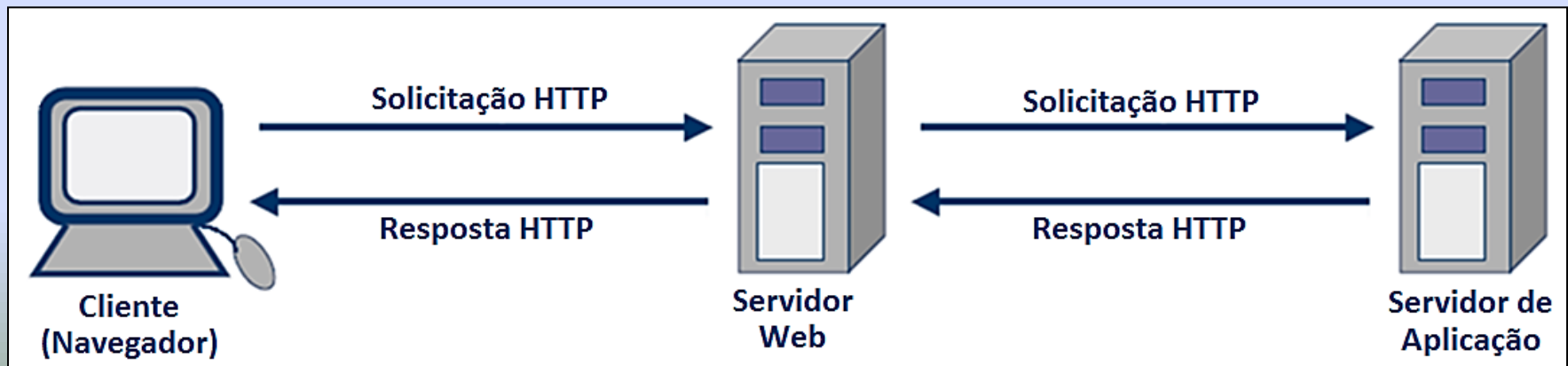
# JavaServer Pages

- Em uma página composta apenas por HTML, o conteúdo é estático e não pode ser alterado pelo usuário. Para apresentá-la, o navegador (cliente) envia uma solicitação HTTP ao servidor Web.
- O servidor processa então a solicitação e envia uma resposta HTTP contendo o documento HTML referente à página solicitada, apresentando-a no navegador.



# JavaServer Pages

- Em páginas HTML que possuem código JSP, o conteúdo pode ser gerado dinamicamente, ou seja, as páginas podem ser modificadas ou construídas em tempo real.
- O cliente envia uma solicitação HTTP ao **servidor Web**. Este, por sua vez, envia outra solicitação ao **servidor de aplicação**, que se encarrega de executar o código JSP que vai gerar o conteúdo dinâmico da página.
- O conteúdo gerado é enviado ao **servidor Web**, que, por sua vez, envia outra resposta HTTP com o documento HTML ao cliente, no qual a página é exibida com todo o seu conteúdo (estático e dinâmico).





# Elementos Sintáticos do JSP

- Existem três tipos principais de códigos Java que podem ser inseridos em páginas HTML, cada uma apresentando um comportamento diferente durante a execução do código.
  - **Scripting** – Scripts usados declarar variáveis, invocar métodos, processar dados e exibir valores ao usuário.
  - **Diretivas** – Informações necessárias para que o Tomcat processe a página JSP de forma correta.
  - **Ações** – Tarefas que o Tomcat deve executar quando uma página é solicitada por um cliente.

# Elementos Sintáticos do JSP


- **Scripting**

- **Scriptlet** – O código é executado toda vez que a página for solicitada. Exemplo:

```
<% out.println("Olá!") %>
```


- **Declaração** – É usado para declarar variáveis e métodos quando a página é executada. Por este motivo, o código é executado somente na primeira vez em que a página é solicitada. Exemplo:

```
<%! int quantAlunos = 25; %>
```



- **Expressão** – É usada para exibir valores de variáveis ao usuário. Exemplo:

```
<p>Quantidade de Alunos: <%= quantAlunos %></p>
```



# Elementos Sintáticos do JSP

- **Diretivas**

- **page** – Permite incluir bibliotecas de classes na página JSP, além de informar alguns atributos da página. Exemplos:

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<%@page import="java.util.Date"%>
```



- **include** – Permite incluir na página o conteúdo de um arquivo-texto. Exemplo:

```
<%@include file="rodapePagina.txt"%>
```



- **taglib** – Permite incluir na página bibliotecas de tags JSP criadas pelo próprio desenvolvedor ou uma biblioteca padrão como a JSLT (*Java Standard Tag Library*). Exemplo:

```
<%@taglib file="minhasTags.jsp"%>
```




# Elementos Sintáticos do JSP

- **Ações**

- Existem oito tipos de **ações** que o Tomcat pode executar quando uma página é solicitada por um cliente: element, forward, include, plugin, text, useBean, getProperty, setProperty.
- Também existem cinco **subações** que podem ser usadas somente dentro do corpo de uma ação: attribute, body, fallback, param e params.
- Exemplo:

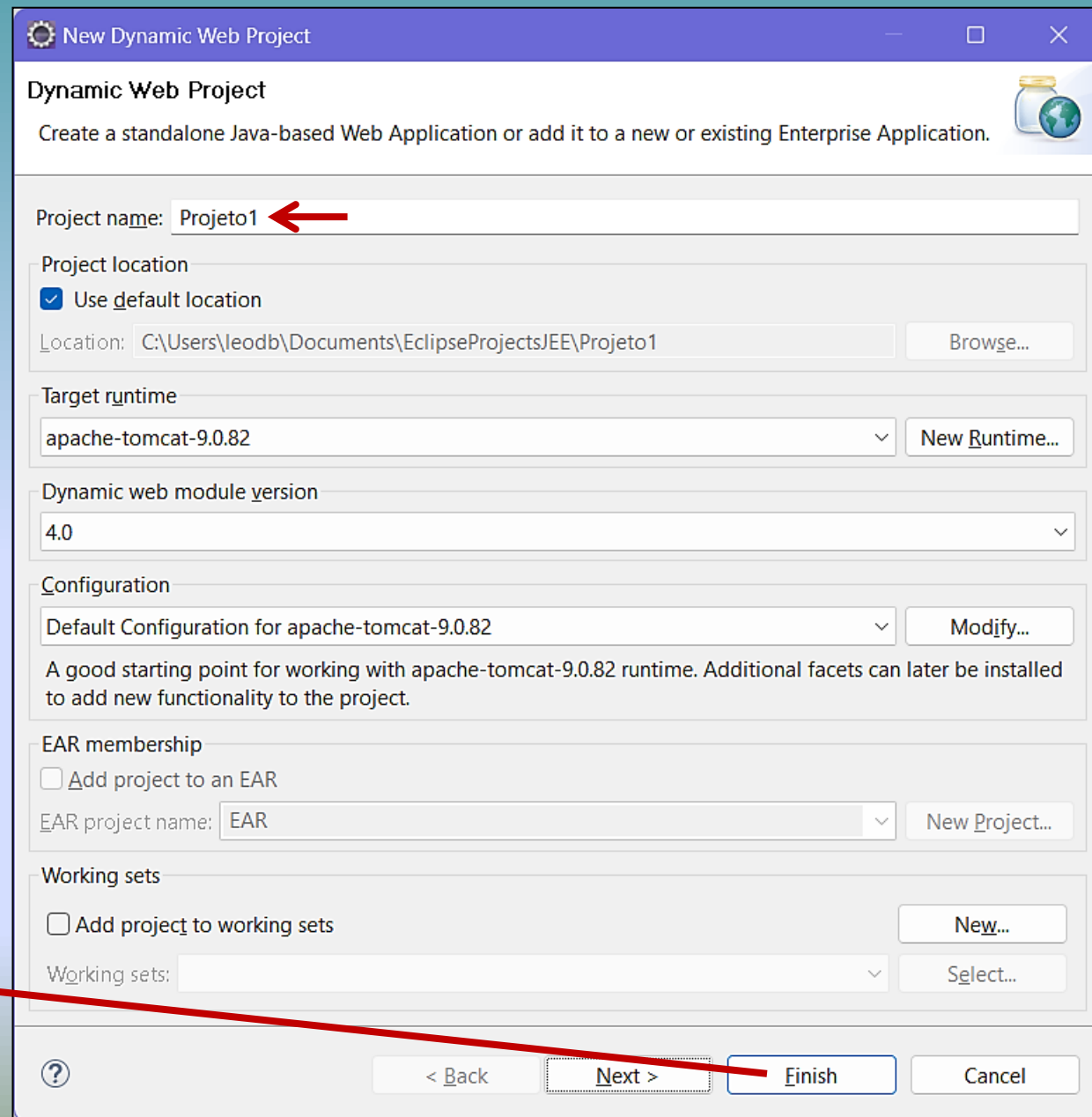
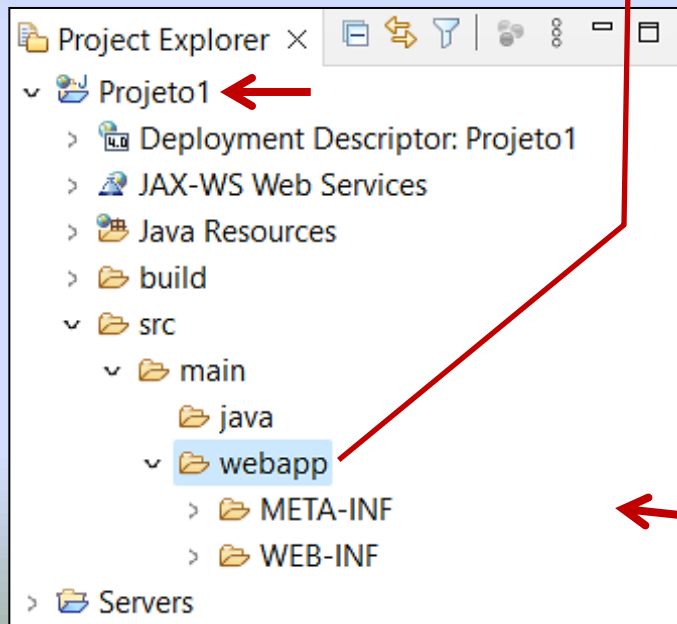
`<jsp:include page="calculo.jsp"/>`

The diagram shows the JSP code snippet `<jsp:include page="calculo.jsp"/>`. Below the opening tag `<jsp:include`, there is a red arrow pointing upwards. Similarly, below the closing tag `/>`, there is another red arrow pointing upwards. This highlights the start and end of the action element.

# Exemplo – Aplicação JSP

- Criando um Projeto  
Menu **File >> New >> Dynamic Web Project**

A pasta **webapp** armazena os arquivos **HTML** e **JSP** do projeto.



# Exemplo – Aplicação JSP

- Criando um Arquivo JSP

**Projeto >> New >> JSP File**

The screenshot shows the Eclipse IDE interface. On the right, the 'Project Explorer' shows a project named 'Projeto1' with a folder structure: 'Deployment Descriptor: Projeto1', 'JAX-WS Web Services', 'Java Resources', 'build', 'src', 'main' (containing 'java' and 'webapp'), and 'META-INF' and 'WEB-INF' under 'webapp'. A red arrow points from the 'webapp' folder in the Project Explorer to the 'New JSP File' wizard.

The 'New JSP File' wizard is open, showing the 'JSP' tab. The 'Enter or select the parent folder:' field is set to 'Projeto1/src/main/webapp'. The 'File name:' field is set to 'exemplo1.jsp'. A red arrow points to the 'File name:' field. The 'Advanced >>' button is visible. At the bottom, there are buttons for '< Back', 'Next >', 'Finish', and 'Cancel'. A red arrow points to the 'Finish' button.

The 'exemplo1.jsp' file is open in the editor, showing the following code:

```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2   pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Insert here title</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

A red arrow points from the code editor to a yellow box at the bottom right containing the text 'Página JSP gerada'.

# Exemplo – Aplicação JSP

- Alterando o código da página JSP gerada

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@page import="java.util.Date"%>
<%@page import="java.text.SimpleDateFormat"%>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Minha Página JSP</title>
</head>
<body>
    <%!int contAcessos = 0;%>
    <%out.println("<h3>Informações de Data e Hora:</h3>");;%>
    <%
        Date DataAtual = new Date();
        String data = new SimpleDateFormat("dd/MM/yyyy").format(DataAtual);
        String hora = new SimpleDateFormat("hh:mm").format(DataAtual);
    %>
    <%contAcessos++;;%>
    <h4>Data Atual: <%=data%></h4>
    <h4>Hora Atual: <%=hora%></h4>
    <h4>Número de Acessos: <%=contAcessos%></h4>
</body>
</html>
```

Diretivas

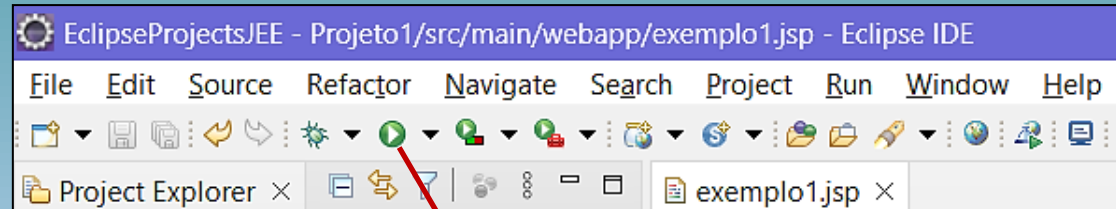
Declaração  
de variável

Scriptlets

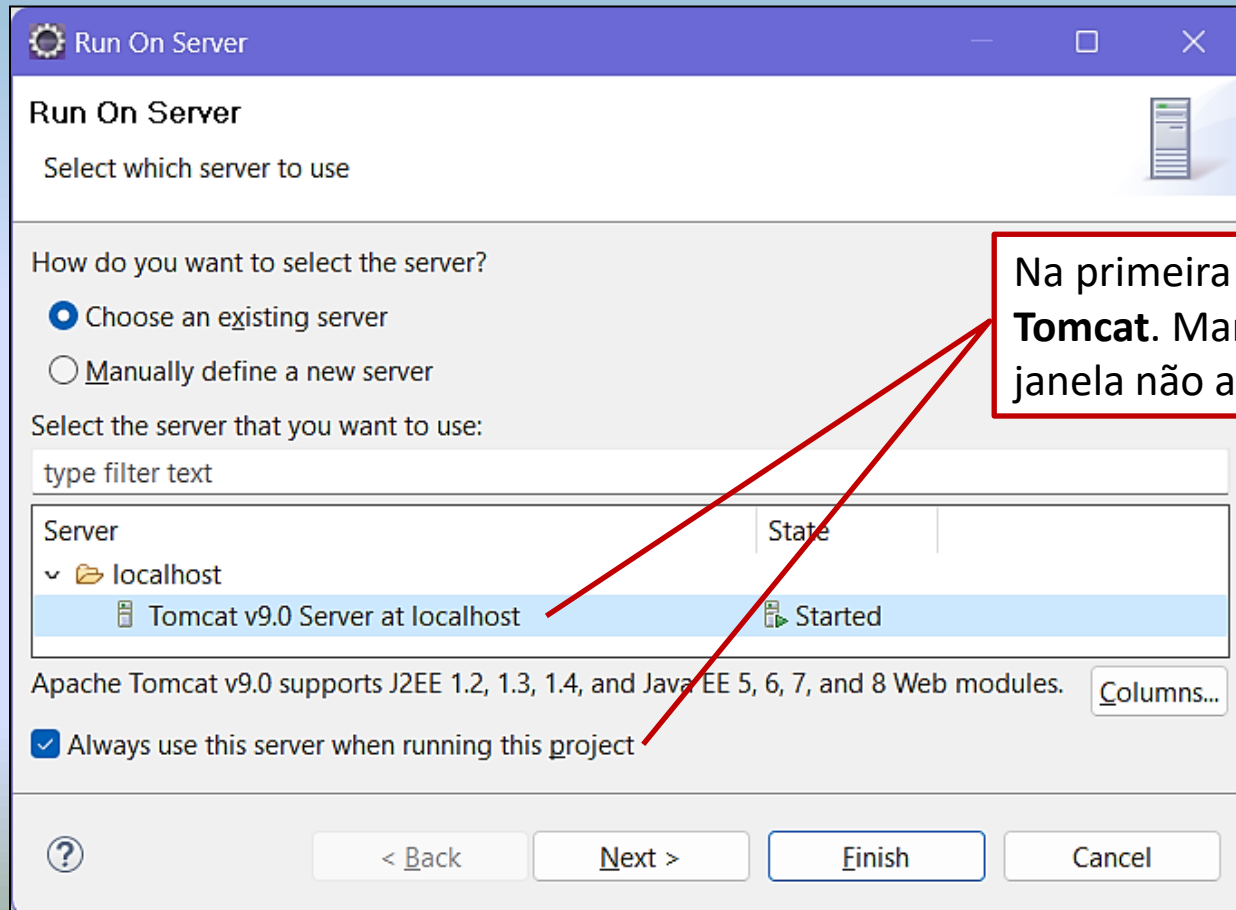
Expressões

# Exemplo – Aplicação JSP

- Executando a página JSP



Para executar a página, clique no botão **Run** da barra de ferramentas.

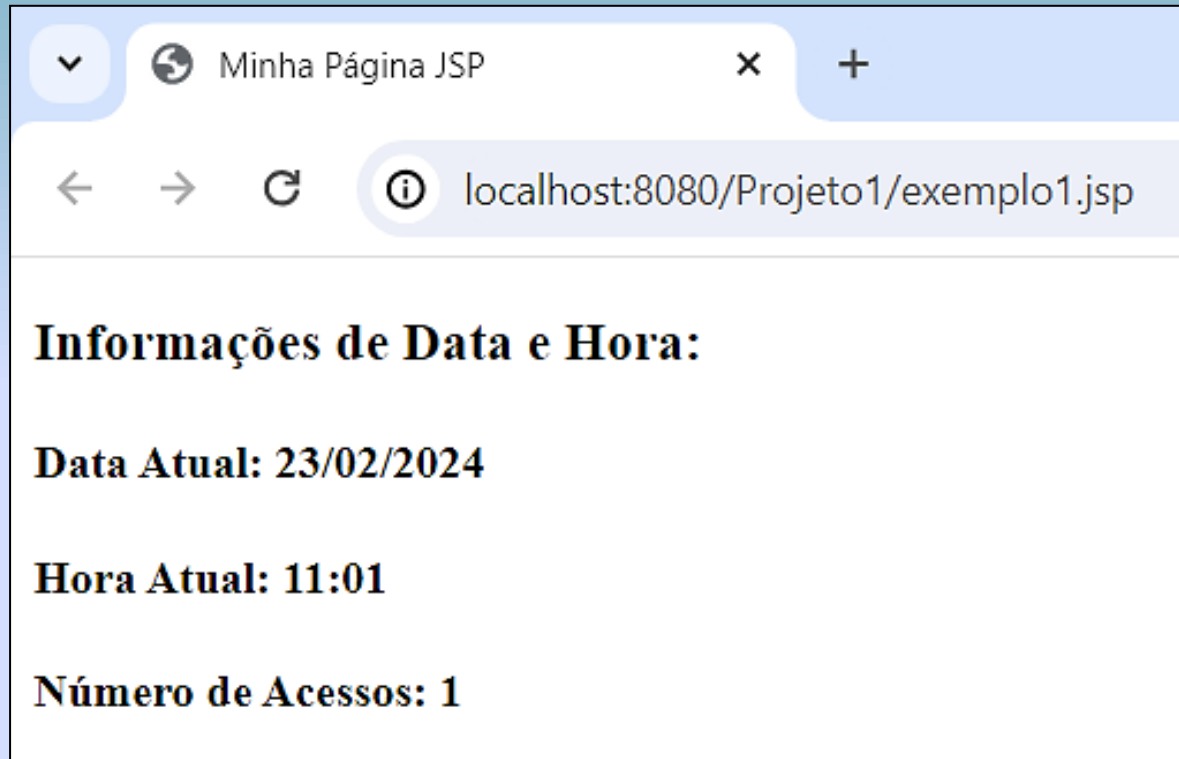


Na primeira execução, selecione o servidor **Tomcat**. Marque esta opção para que essa janela não apareça nas próximas execuções.



# Exemplo – Aplicação JSP

- Página JSP exibida no navegador

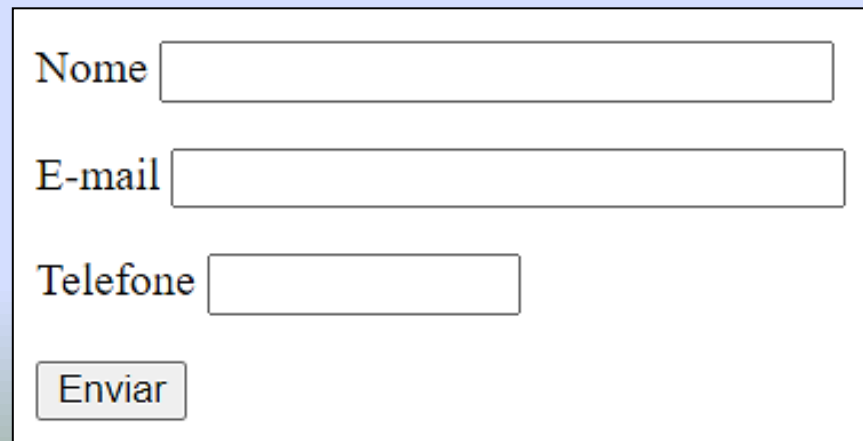


*O navegador padrão do **Eclipse** pode ser alterado no menu **Window >> Preferences >> General >> Web Browser**.*

# Formulários

- Em aplicações Web, os formulários podem ser criados por meio da tag **<form>** da linguagem HTML, a qual agrupa diversos componentes, como: campos de texto, rótulos de campos, caixas de seleção e botões.

```
<form>  
  <p>Nome <input type="text" name="nome" size="30"></p>  
  <p>E-mail <input type="text" name="email" size="30"></p>  
  <p>Telefone <input type="text" name="telefone" size="11"></p>  
  <p><input type="submit" value="Enviar" name="enviar"></p>  
</form>
```



Nome

E-mail

Telefone

# Formulários

- Para que seja possível enviar os dados de um formulário para uma página JSP, é preciso informá-la no atributo **action** da tag **<form>**.
- O atributo **action** especifica o arquivo de código responsável, no lado servidor, pela tarefa de receber e processar os dados informados no formulário.



```
<form action="recebeDados.jsp">
  <p>Nome <input type="text" name="nome" size="30"></p>
  <p>E-mail <input type="text" name="email" size="30"></p>
  <p>Telefone <input type="text" name="telefone" size="11"></p>
  <p><input type="submit" value="Enviar" name="enviar"></p>
</form>
```

# Formulários

- Para enviar os dados de um formulário para o servidor, também é preciso definir o método de envio por meio do atributo **method** da tag **<form>**.
- Existem dois métodos principais para envio de dados: **GET** (padrão, caso não seja especificado nenhum método) e **POST**.



```
<form method="POST" action="recebeDados.jsp">
  <p>Nome <input type="text" name="nome" size="30"></p>
  <p>E-mail <input type="text" name="email" size="30"></p>
  <p>Telefone <input type="text" name="telefone" size="11"></p>
  <p><input type="submit" value="Enviar" name="enviar"></p>
</form>
```

# Formulários

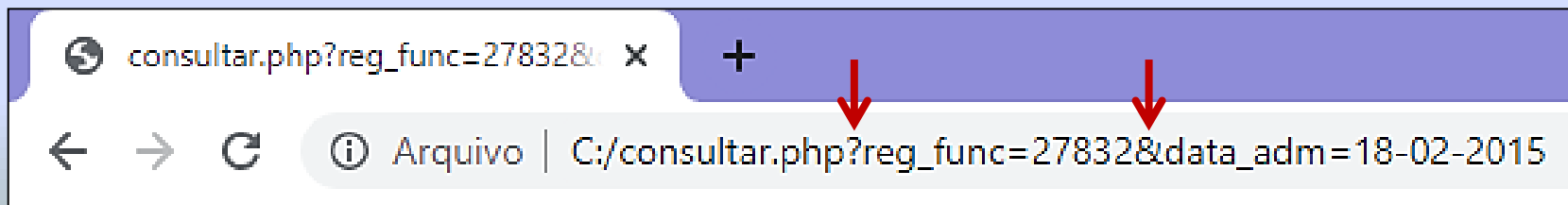
- Método **GET**

*Ao enviar o formulário, seus dados são enviados na URL do endereço de destino, logo após o nome do arquivo de código. O caractere **?** representa o início do conjunto de campos do formulário enviado e o caractere **&** separa cada par campo-valor.*

Registro Funcional

Data de Admissão

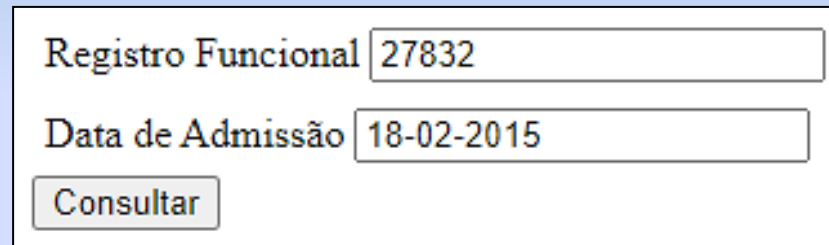
*O método **GET** deve ser usado apenas em consultas, e que não envolvam dados confidenciais. Para outros tipos de operação (login, inserção, alteração, exclusão de dados etc) ou para qualquer operação que envolva dados sigilosos, deve-se usar o método **POST**.*



# Formulários

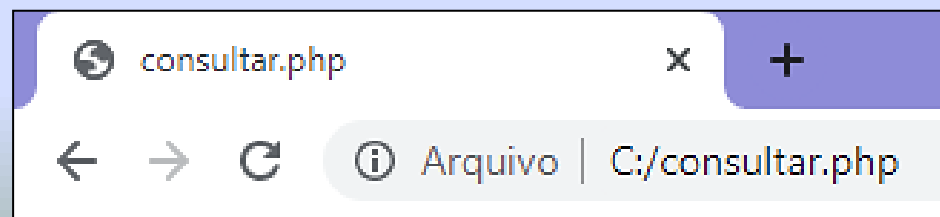
- Método **POST**

*Ao enviar o formulário, seus dados não são enviados na URL do endereço de destino. Em vez disso, os dados são encapsulados no corpo da requisição HTTP enviada ao servidor.*



Registro Funcional

Data de Admissão



# Formulários

- Características GET x POST

Características	Método GET	Método POST
Limite de caracteres na URL	Possui limite de caracteres que podem ser enviados na URL, o qual varia conforme o navegador.	Não possui limitação, pois os dados são encapsulados no corpo da requisição HTTP.
Armazenamento em cache	A URL do endereço é armazenada no <i>cache</i> do navegador ou de um <i>proxy</i> (servidor intermediário). Com isso, a requisição HTTP pode não chegar ao servidor, já que o navegador pode recuperar a página que está em seu <i>cache</i> ou no <i>cache</i> de um <i>proxy</i> .	A URL do endereço não é armazenada em <i>cache</i> . Por isso, se for necessário que uma requisição sempre chegue no servidor, deve-se usar este método.
Visibilidade dos dados	Os dados ficam visíveis na URL, podendo ser capturados por programas maliciosos.	Os dados não ficam visíveis na URL, pois os dados são encapsulados no corpo da requisição HTTP.
Tipos de dados	Permite enviar apenas caracteres.	Permite enviar qualquer tipo de dado, como arquivos de texto e de imagem.

# Recuperação de Dados

- Ao enviar um formulário para o servidor, os dados recebidos podem ser recuperados através do objeto implícito\* **request**.
- Este objeto possui o método **getParameter**, que permite acessar os valores informados no formulário por meio do respectivo atributo **name** de cada campo.

*Os valores retornados pelo método **getParameter** são do tipo **String**. Por isso, se estes valores forem usados em cálculos matemáticos, torna-se ser necessário convertê-los para um tipo de dado compatível (int, double etc).*

\* **Objetos implícitos** são objetos que não precisam ser declarados, pois são inicializados implicitamente quando a página JSP é carregada.



# Recuperação de Dados

index.html

```
<form method="POST" action="recebeDados.jsp">
  <p>Nome <input type="text" name="nome" size="30"></p>
  <p>E-mail <input type="text" name="email" size="30"></p>
  <p>Telefone <input type="text" name="telefone" size="11"></p>
  <p><input type="submit" value="Enviar" name="enviar"></p>
</form>
```

Para criar um arquivo HTML, acesse:  
**Projeto >> New >> HTML File**

recebeDados.jsp (seção <body>)

```
<body>
  <%
    String nome, email, telefone;
    nome = request.getParameter("nome");
    email = request.getParameter("email");
    telefone = request.getParameter("telefone");
  %>
  <h4>Nome: <%=nome%></h4>
  <h4>E-mail: <%=email%></h4>
  <h4>Telefone: <%=telefone%></h4>
</body>
```

Nome

E-mail

Telefone

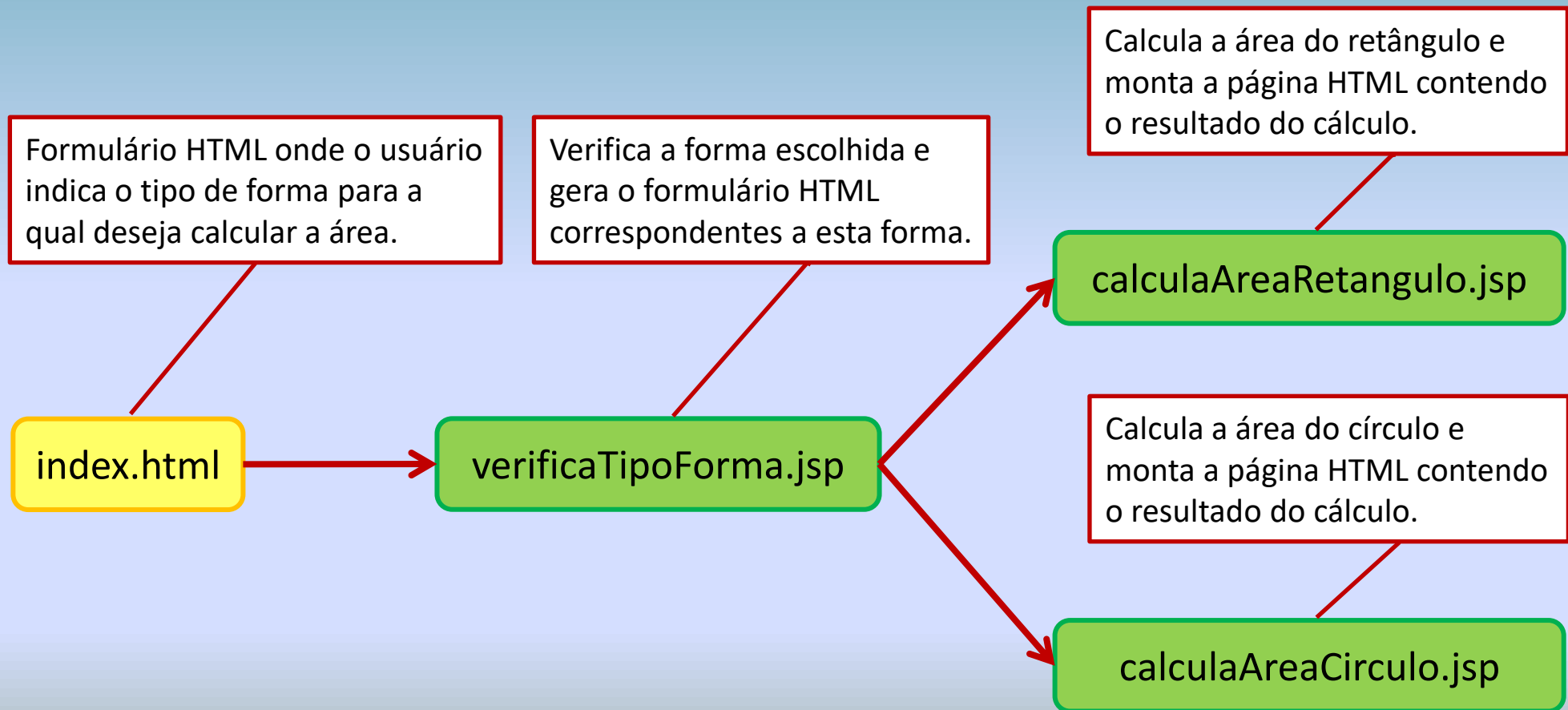
**Nome: Marie Curie**

**E-mail: mariecurie@gmail.com**

**Telefone: 1133334444**

# Exemplo – Aplicação JSP com Formulário

- Cálculo de Áreas de Formas Geométricas



# Exemplo – Aplicação JSP com Formulário

index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Cálculo de Áreas</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
<div>
<h3>Cálculo de Áreas de Formas Geométricas</h3>
<form action="verificaTipoForma.jsp">
  <p>Informe seu nome <input type="text" name="txNome" size="30"></p>
  <p><b>Tipos de Formas Geométricas:</b></p>
  <p><input type="radio" name="rbForma" value="retangulo" checked> Retângulo</p>
  <p><input type="radio" name="rbForma" value="circulo"> Círculo</p>
  <p><input type="submit" name="btProximo" value="Próximo"></p>
</form>
</div>
</body>
</html>
```

## Cálculo de Áreas de Formas Geométricas

Informe seu nome

### Tipos de Formas Geométricas:

☒ Retângulo

☐ Círculo

# Exemplo – Aplicação JSP com Formulário

## verificaTipoForma.jsp (seção <body>)

```
<body>
<%
// Verifica qual botão "radio" (Retângulo ou Círculo) foi selecionado, para
// montar o formulário contendo os respectivos dados de entrada da forma.
if (request.getParameter("rbForma").equals("retangulo")) {
    out.println("<form action='calculaAreaRetangulo.jsp'>");
    out.println("<h3>Cálculo da área do retângulo</h3>");
    out.println("<p>Base do retângulo <input type='text' name='txBase' size='10'></p>");
    out.println("<p>Altura do retângulo <input type='text' name='txAltura' size='10'></p>");
} else if (request.getParameter("rbForma").equals("circulo")) {
    out.println("<form action='calculaAreaCirculo.jsp'>");
    out.println("<h3>Cálculo da área do círculo</h3>");
    out.println("<p>Raio do círculo <input type='text' name='txRaio' size='10'></p>");
}

// Verifica se o nome do usuário foi preenchido na tela inicial:
// Se sim, inclui o nome em um campo oculto (hidden) do formulário.
// Se não, inclui o texto "Usuário anônimo" neste campo oculto.
if (!(request.getParameter("txNome").equals(""))
    out.println("<input type='hidden' name='hdNome' value='" + request.getParameter("txNome") + "'>");
else
    out.println("<input type='hidden' name='hdNome' value='Usuário anônimo'>");

// Inclui o botão Calcular no formulário.
out.println("<input type='submit' name='btCalcular' value='Calcular'>");
out.println("</form>"); // Fecha o formulário.
%>
</body>
```

### Cálculo da área do retângulo

Base do retângulo

Altura do retângulo

Calcular

### Cálculo da área do círculo

Raio do círculo

Calcular

# Exemplo – Aplicação JSP com Formulário

## calculaAreaRetangulo.jsp (seção <body>)

```
<body>
  <%!double base = 0, altura = 0, area = 0;%>
  <%
    // Verifica se os campos Base e Altura foram preenchidos.
    if (!(request.getParameter("txBase").equals("")) && !(request.getParameter("txAltura").equals(""))) {
      base = Double.parseDouble(request.getParameter("txBase"));
      altura = Double.parseDouble(request.getParameter("txAltura"));
      area = base * altura;
      // Recupera o valor do campo oculto da página verificaTipoForma.jsp e apresenta o resultado.
      out.println("<h4>" + request.getParameter("hdNome") + ", a área do retângulo é: " + area + "</h4>");
    } else
      out.println("Preencha todos os campos!");
  %>
</body>
```

## calculaAreaCirculo.jsp (seção <body>)

```
<body>
  <%!double raio = 0, area = 0;%>
  <%
    // Verifica se o campo Raio foi preenchido.
    if (!(request.getParameter("txRaio").equals(""))) {
      raio = Double.parseDouble(request.getParameter("txRaio"));
      area = Math.pow(raio, 2) * Math.PI;
      // Recupera o valor do campo oculto da página verificaTipoForma.jsp e apresenta o resultado.
      out.println("<h4>" + request.getParameter("hdNome") + ", a área do círculo é: " + area + "</h4>");
    } else
      out.println("Preencha todos os campos!");
  %>
</body>
```

# Exemplo – Aplicação JSP com Formulário

**Cálculo de Áreas de Formas Geométricas**

Informe seu nome

**Tipos de Formas Geométricas:**

☒ Retângulo

☐ Círculo

**Cálculo da área do retângulo**

Base do retângulo

Altura do retângulo

**João, a área do retângulo é: 28.0**

**Cálculo de Áreas de Formas Geométricas**

Informe seu nome

**Tipos de Formas Geométricas:**

☐ Retângulo

☒ Círculo

**Cálculo da área do círculo**

Raio do círculo

**Usuário anônimo, a área do círculo é: 78.53981633974483**

# Referências

- ALVES, William Pereira. **Java para Web**: Desenvolvimento de Aplicações. São Paulo: Érica, 2015.