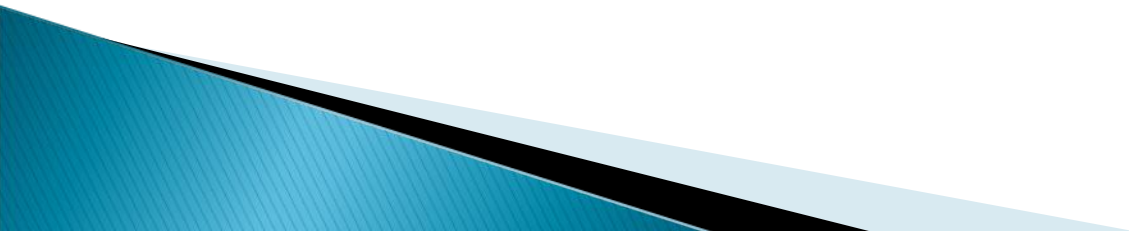
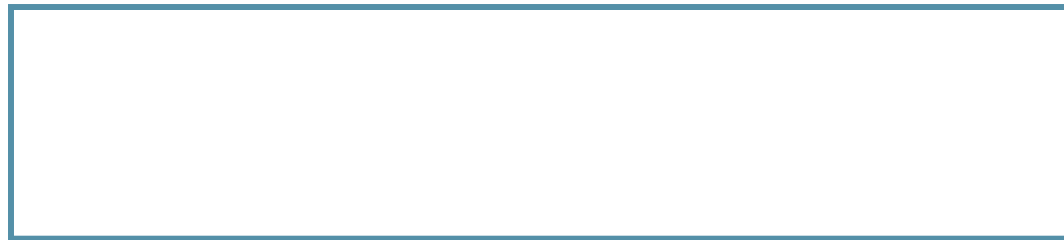
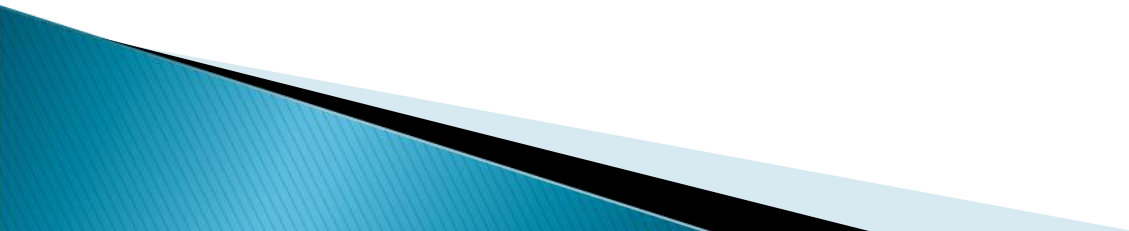


Programação Java para WEB

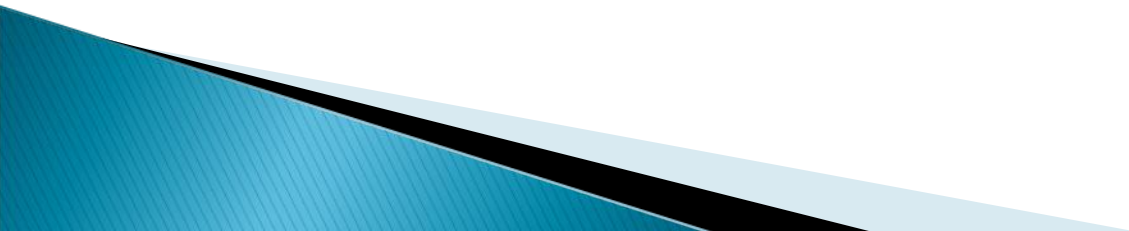


Introdução à Programação Web com Java

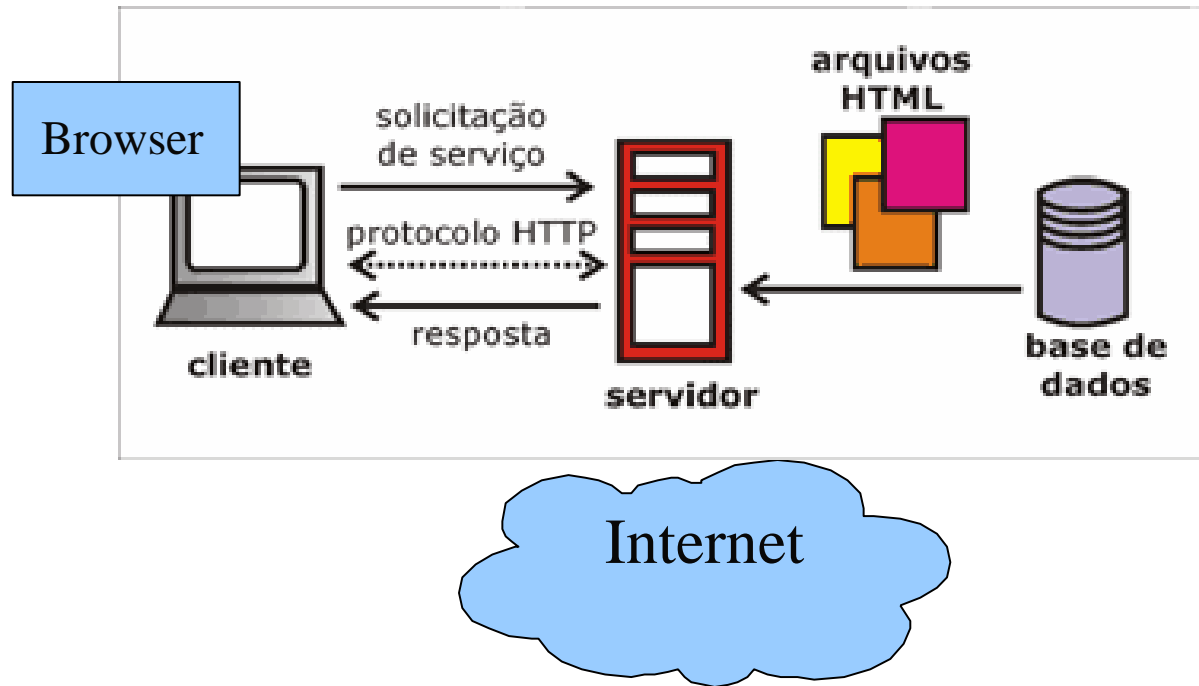


Arquitetura da web

- A arquitetura da web utiliza o modelo cliente-servidor:
 - Servidor web: recebe pedidos e devolve informações
 - Cliente web: elemento que faz requisição a um servidor web e recebe uma resposta de volta



Interação cliente-servidor



A comunicação entre cliente e servidor na web é feita utilizando o protocolo HTTP

HTTP – o que é?

■ HTTP = Hypertext Transfer Protocol

- ☐ O protocolo principal da web
- ☐ O protocolo usado para comunicação entre os browsers e os servidores
- ☐ Permite a transferência de informações multimídia: texto, imagens e sons
- ☐ Não mantém estado: cada nova requisição precisa abrir outra conexão

Conceito de URL

■ URL = Universal Resource Locator

- Termo usado para identificar/localizar recursos de maneira única e uniforme
- Especifica tanto o servidor como o recurso que está sendo requisitado
- Browser especifica a url (servidor+recurso) e recebe o recurso como resposta

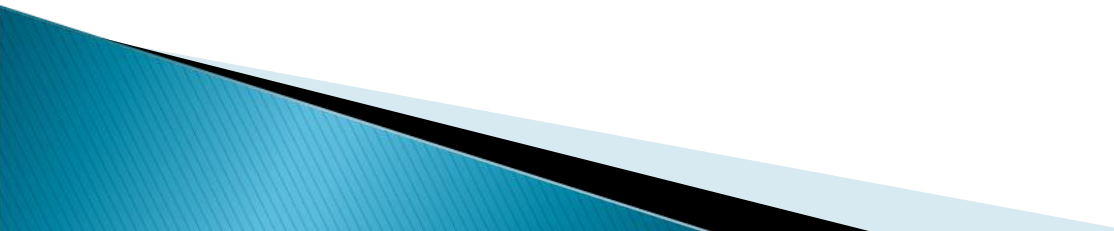
`http://www.jarley.com/index.html`

servidor

recurso

Interação Browser - Servidor

■ Cenário de uso

- 1.Usuário especifica a URL
 - 2.Browser conecta com o servidor especificado na URL
 - 3.Browser prepara e envia o pedido HTTP
 - 4.Servidor busca recurso identificado pela URL
 - 5.Servidor prepara resposta HTTP com o recurso e faz o envio
 - 6.Browser processa a resposta e exibe o recurso solicitado
 - 7.Browser verifica tags e repete o processo para outros recursos especificados (ex.: figuras em uma página HTML)
- 

Recursos Web

■ Os recursos web podem ser de vários formatos:

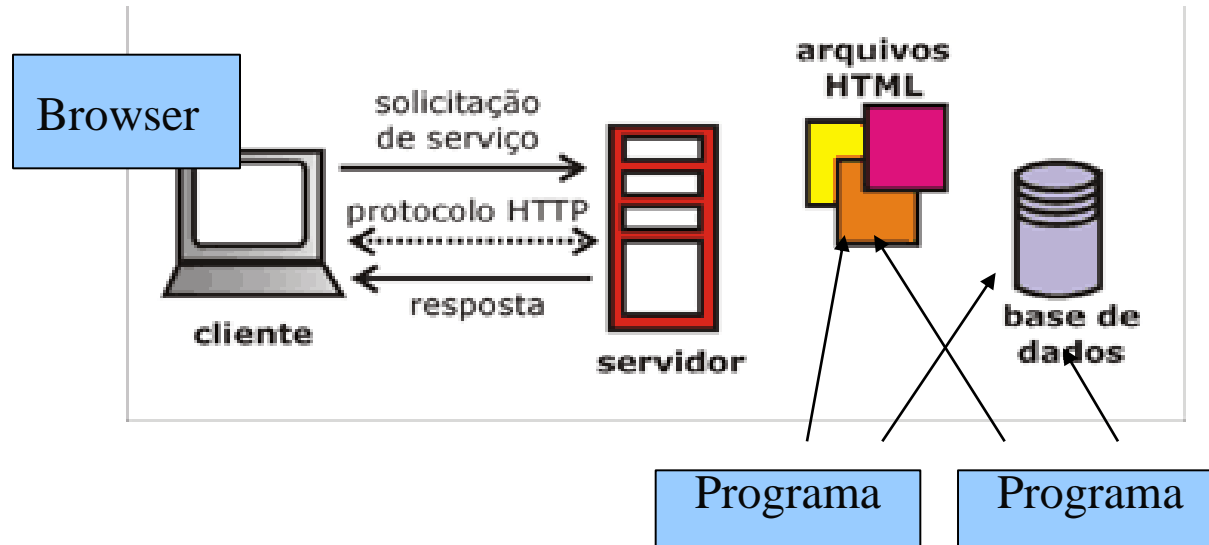
- ☐ Páginas HTML
- ☐ Arquivos texto ou binário
- ☐ Figuras (gif, jpeg, bmp, png, etc.)
- ☐ Programas (CGI, ASP, Servlet, JSP, etc.)

Páginas HTML

- São os recursos mais acessados

- ☐ Podem disparar requisições de outros recursos (links, imagens)
- ☐ Formulários HTML permitem passar dados para programas residentes do servidor
- ☐ Permitem interatividade com o usuário

Acessando recursos web



Os recursos podem estar armazenados no mesmo servidor ou em outro local.

Programas Web

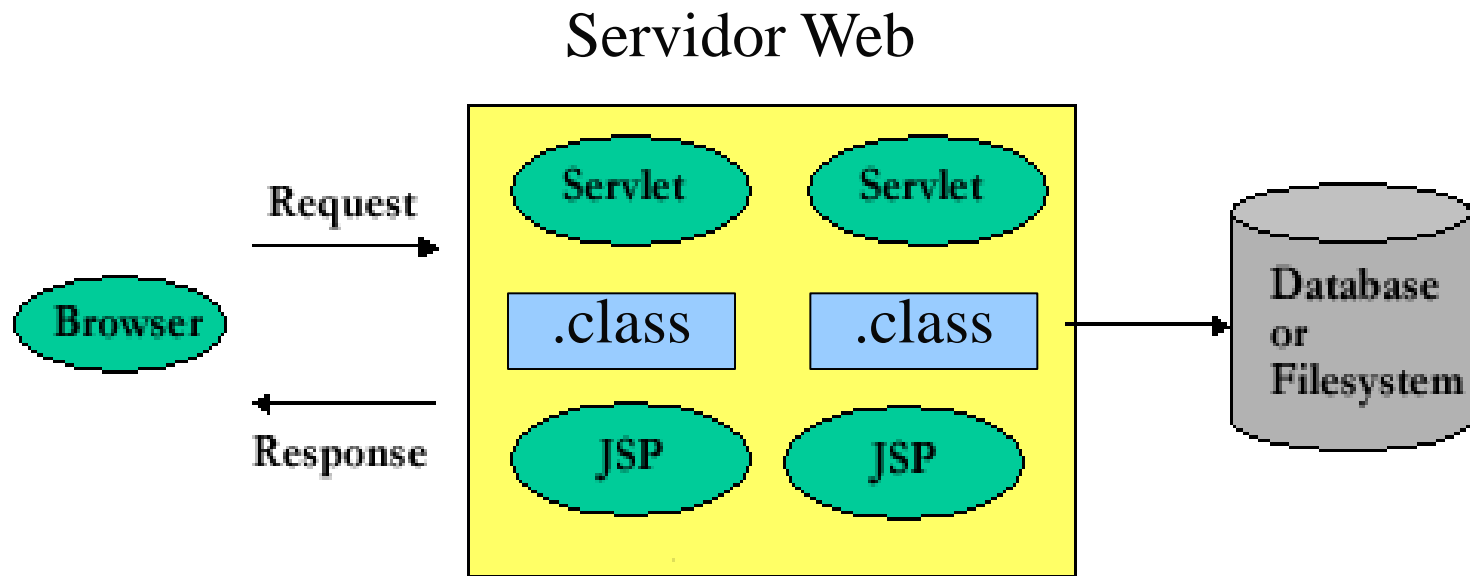
- Elementos residentes no servidor

- ☐ Interativos: usuários passam informações como entrada
- ☐ Dinâmicos: a resposta pode depender das informações passadas
- ☐ Exemplos:
 - Servlets e JavaServer Pages
 - Active Server Pages (ASP)
 - CGI
 - PHP

Servlets e JavaServer Pages

- São programas Java

- Recursos da linguagem para programação web



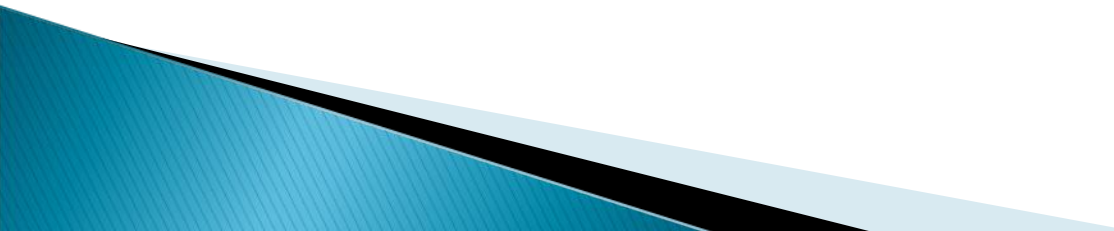
Servlets e JavaServer Pages

■ Recebem requisições e geram respostas

- ☐ Podem acessar outras classes de Java
- ☐ Ambos possuem os mesmos recursos
- ☐ Servlets: mais utilizados para processamento tipicamente "server-side"
- ☐ JSP's: mais utilizados quando existe a necessidade de apresentação de informações para o usuário
- ☐ Boas práticas sugerem o uso de ambos

Uma Aplicação Web com Java

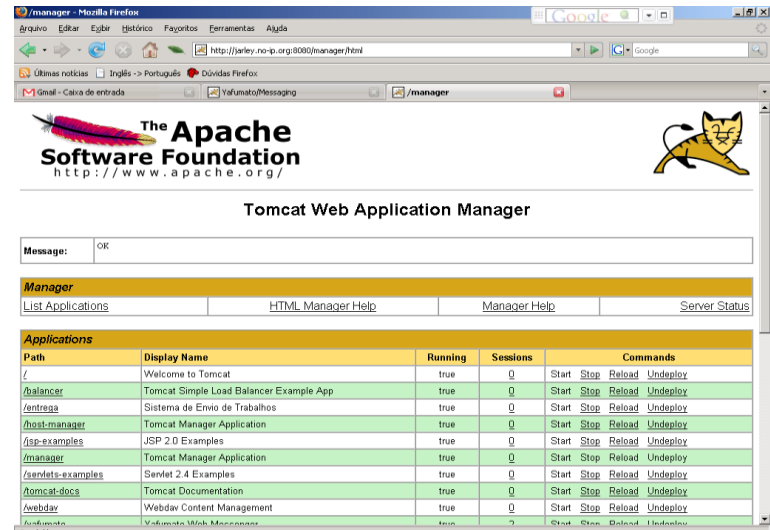
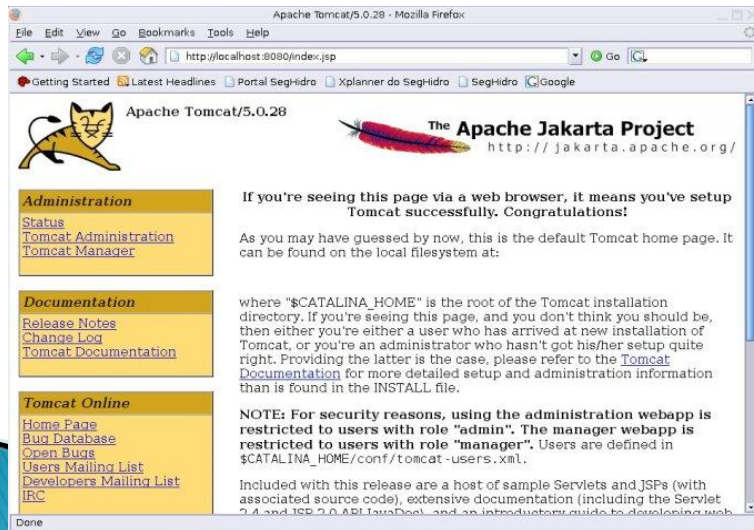
■ Consiste de:

- ☐ Um container
 - ☐ Diretório raiz (contexto)
 - ☐ Deployment Descriptor
 - ☐ Servlets ou JSP's
 - ☐ Outras classes Java
 - ☐ Arquivos auxiliares (properties, XML, HTML, texto, bibliotecas de terceiros)
- 

Uma Aplicação Web com Java

■ Container

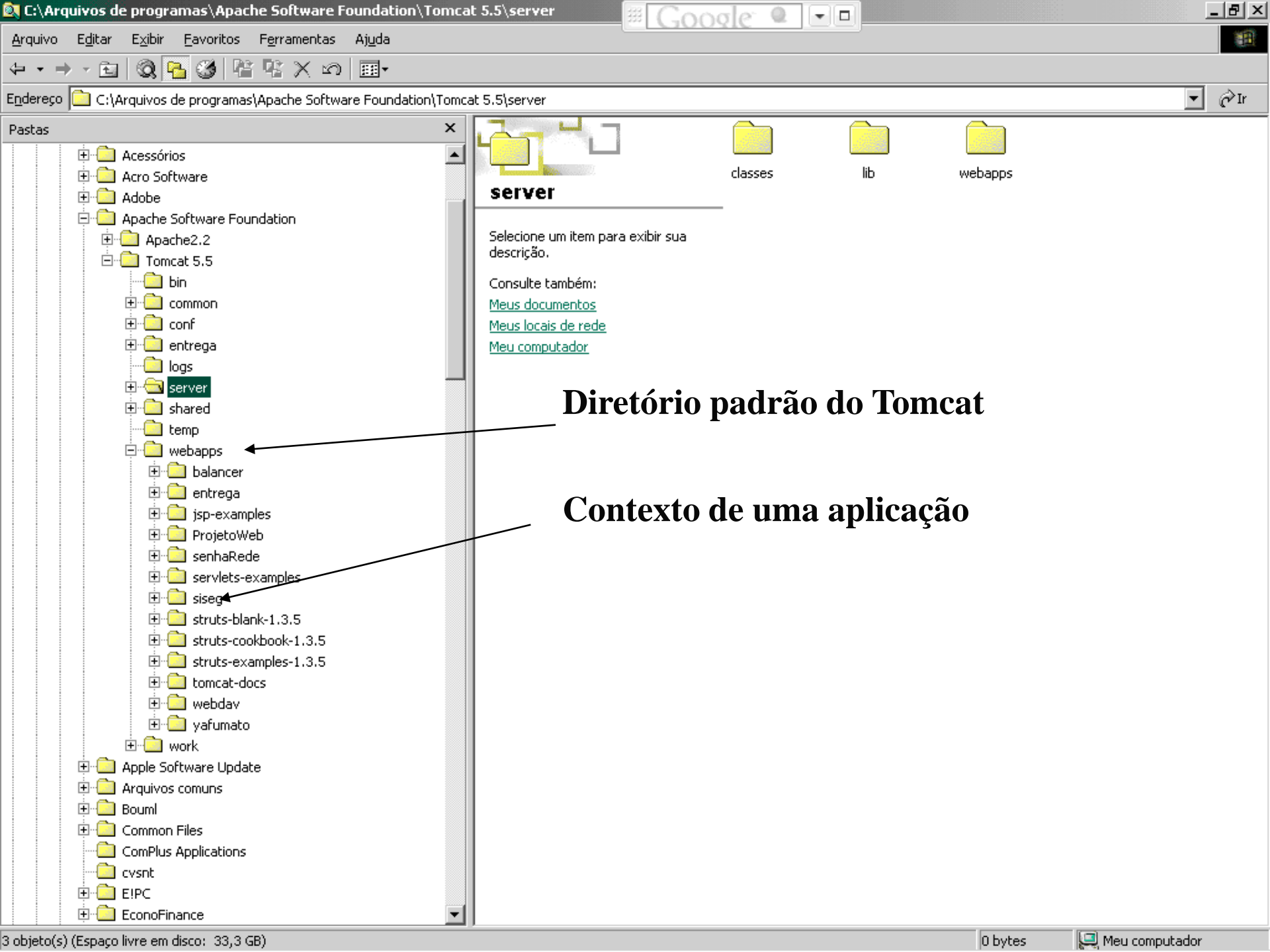
- Nome do software que faz parte do servidor e gerencia as aplicações web
- Toda aplicação existe dentro de um container



Uma Aplicação Web com Java

■ Contexto

- ☐ É o diretório raiz da aplicação
- ☐ Pode ser um diretório padrão do container (ex.:webapps no Tomcat)
- ☐ Apartir do contexto são inseridos todos os elementos de uma aplicação web em Java



Diretório padrão do Tomcat

Contexto de uma aplicação

Uma Aplicação Web com Java

■ Deployment Descriptor

- É um arquivo XML que configura toda a aplicação dentro do servidor:

- Define um “alias” para cada servlet da aplicação

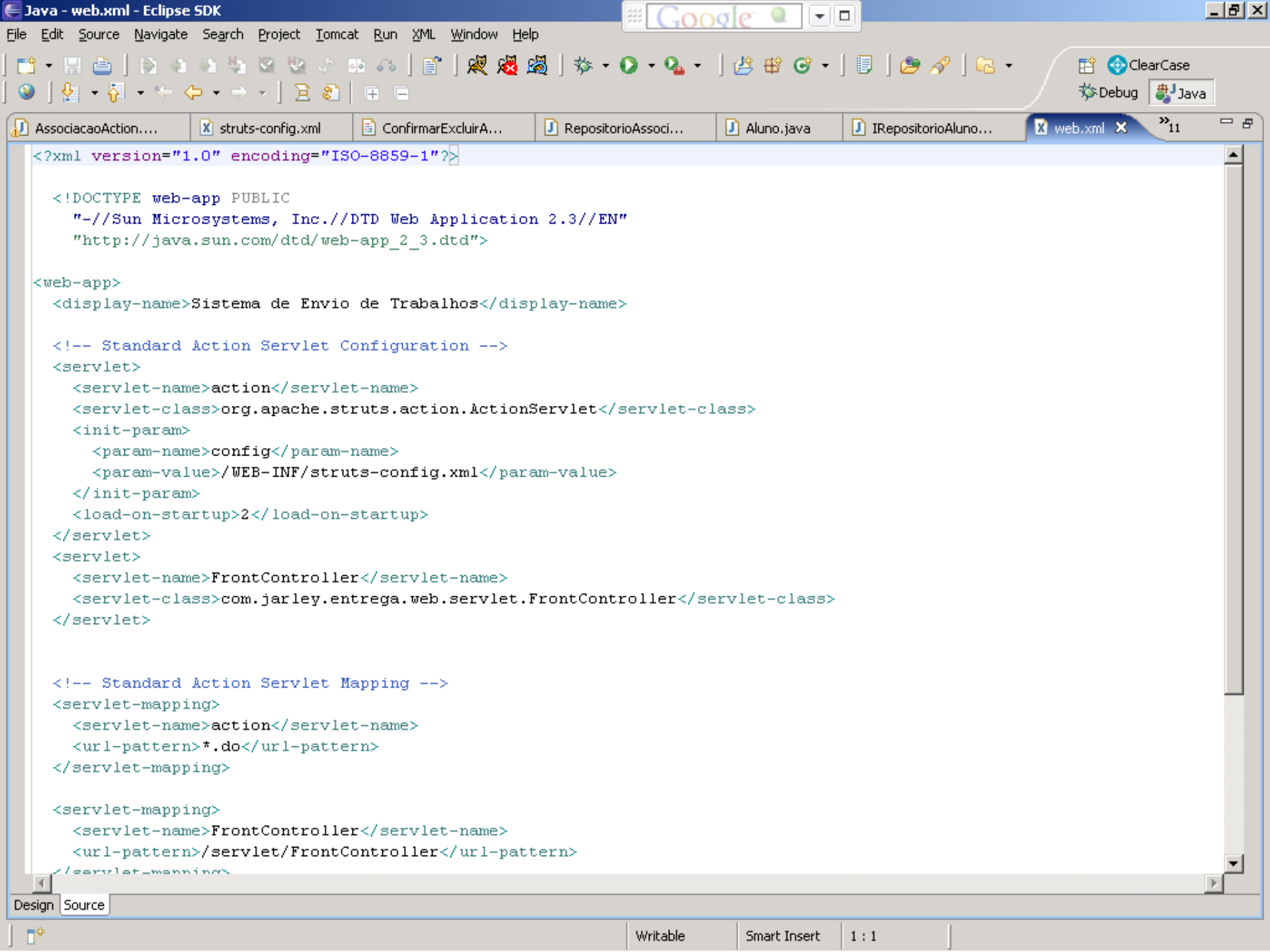
- Filtros de requisição

- Páginas de erro

- Restrições de segurança

- ...

- Nome padrão: web.xml



```
Java - web.xml - Eclipse SDK
File Edit Source Navigate Search Project Tomcat Run XML Window Help

<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app PUBLIC
    "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
  <display-name>Sistema de Envio de Trabalhos</display-name>

  <!-- Standard Action Servlet Configuration -->
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
  </servlet>
  <servlet>
    <servlet-name>FrontController</servlet-name>
    <servlet-class>com.jarley.entrega.web.servlet.FrontController</servlet-class>
  </servlet>

  <!-- Standard Action Servlet Mapping -->
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>

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

Design Source
Writable Smart Insert 1 : 1
```

Estrutura de diretório padrão

- Supondo que o contexto da aplicação seja ENTREGA:

- ENTREGA

- Diretório raiz que disponibiliza arquivos públicos acessíveis via browser (ex.: HTML, JSP's, etc.)

- ENTREGA/WEB-INF

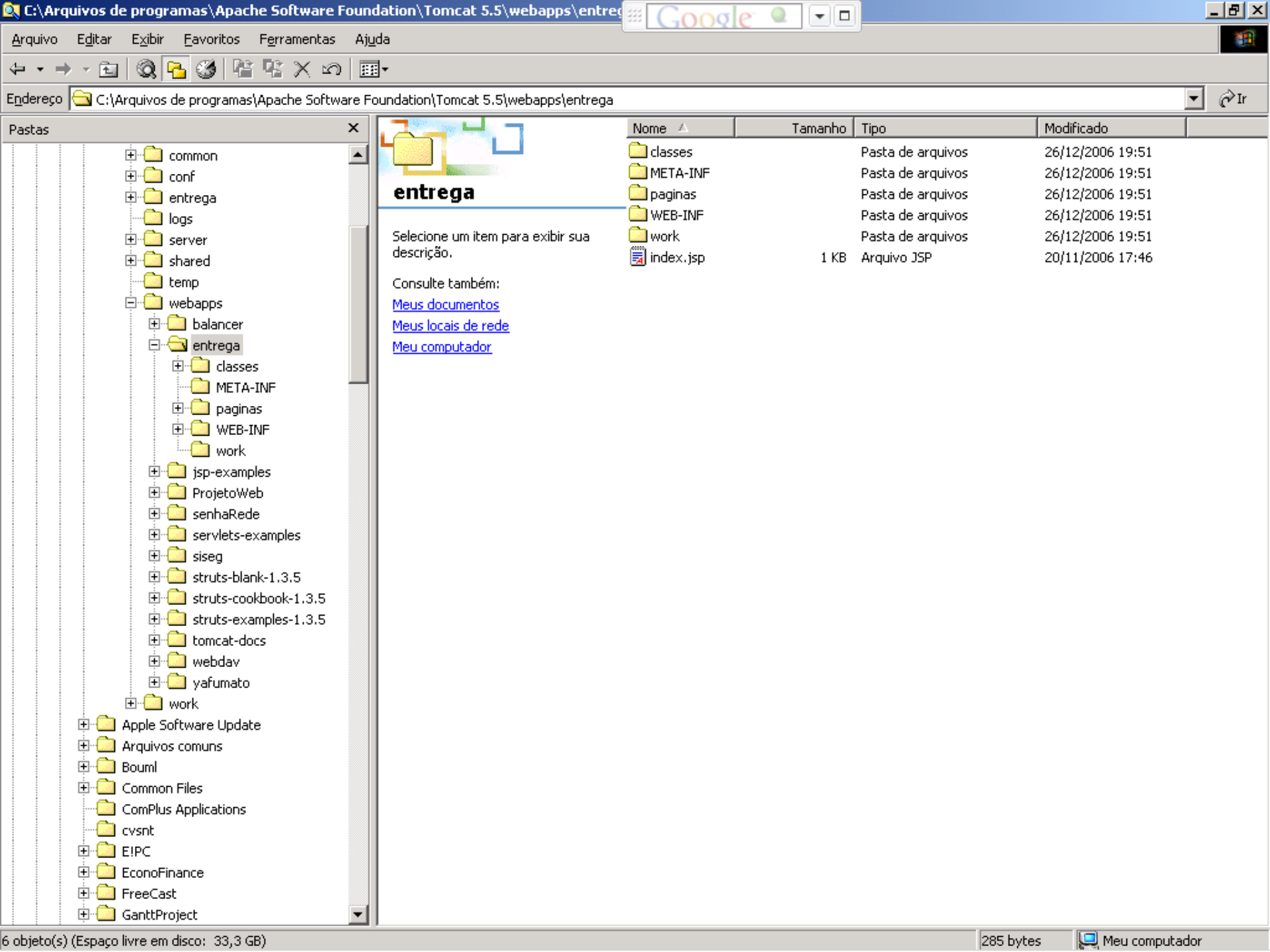
- Diretório que armazena dados protegidos, não acessíveis via browser (ex.: web.xml)

- ENTREGA/WEB-INF/classes

- Classes Java da aplicação, inclusive os servlets

- ENTREGA/WEB-INF/lib

- Bibliotecas usadas pela aplicação (arquivos .jar, .zip, etc.)



C:\Arquivos de programas\Apache Software Foundation\Tomcat 5.5\webapps\entrega

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Endereço

C:\Arquivos de programas\Apache Software Foundation\Tomcat 5.5\webapps\entrega

Ir

Pastas

+

common

+

conf

+

entrega

logs

+

server

+

shared

temp

-

webapps

+

balancer

-

entrega

+

classes

META-INF

+

paginas

+

WEB-INF

work

+

jsp-examples

+

ProjetoWeb

+

senhaRede

+

servlets-examples

+

siseg

+

struts-blank-1.3.5

+

struts-cookbook-1.3.5

+

struts-examples-1.3.5

+

tomcat-docs

+

webdav

+

yafumato

+

work

+

Apple Software Update

+

Arquivos comuns

+

Bouml

+

Common Files

ComPlus Applications

cvsnt

+

EIPC

+

EconoFinance

+

FreeCast

+

GanttProject

Nome

classesMETA-INFpaginasWEB-INFworkindex.jsp

Tamanho

Tipo

Pasta de arquivosPasta de arquivosPasta de arquivosPasta de arquivosPasta de arquivosArquivo JSP

Modificado

26/12/2006 19:5126/12/2006 19:5126/12/2006 19:5126/12/2006 19:5126/12/2006 19:5120/11/2006 17:46

entrega

Selecione um item para exibir sua descrição.

Consulte também:

[Meus documentos](#)

[Meus locais de rede](#)

[Meu computador](#)

6 objeto(s) (Espaço livre em disco: 33,3 GB)

285 bytes

Meu computador

Introdução à HTML

Introdução à HTML

- Linguagem utilizada para a construção de páginas web
 - Consiste em linhas de programa em forma de texto comum e código especiais
 - Extensão:
 - html ou htm
 - Padrão de nomes para as páginas iniciais:
 - index.html ou index.htm

HTML – HyperText Markup Language

- Linguagem de marcação

- Divide o texto em diferentes pedaços identificados por *tags* (elementos de marcação)

```
<h1>Exemplo de Cabeçalho</h1>
```

- Os browsers encarregam-se de interpretar as tags e formatar o texto adequadamente

Tags – elementos de marcação

- Delimitados por "<" e ">"

- Indiferentes quanto à caixa (maiúsculo e minúsculo)

- <H1> é o mesmo que <h1>

- Forma geral:

`<tag>texto</tag>`

- Algumas tags não exigem a finalização

- Exemplo:
 (insere uma quebra de texto), <hr> (insere uma linha horizontal)

Tags – elementos de marcação

- Podem ser aninhados

```
<h1>Exemplo de tags <b>aninhados</b></h1>
```

- Podem ter atributos

```

```

Estrutura de um documento HTML

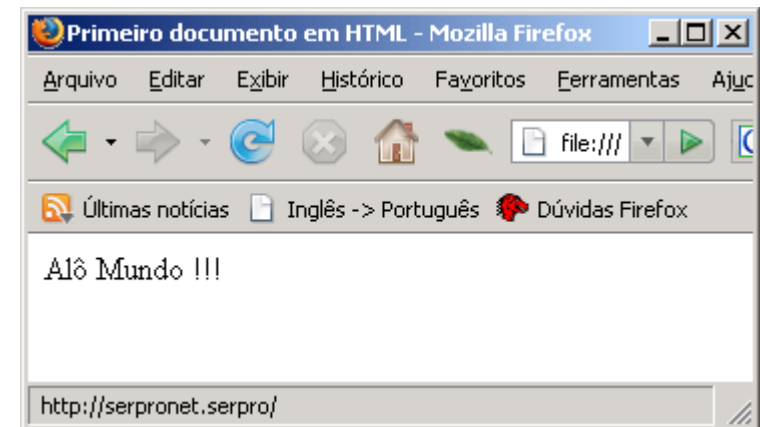
Início

Cabeçalho

Corpo

Fim

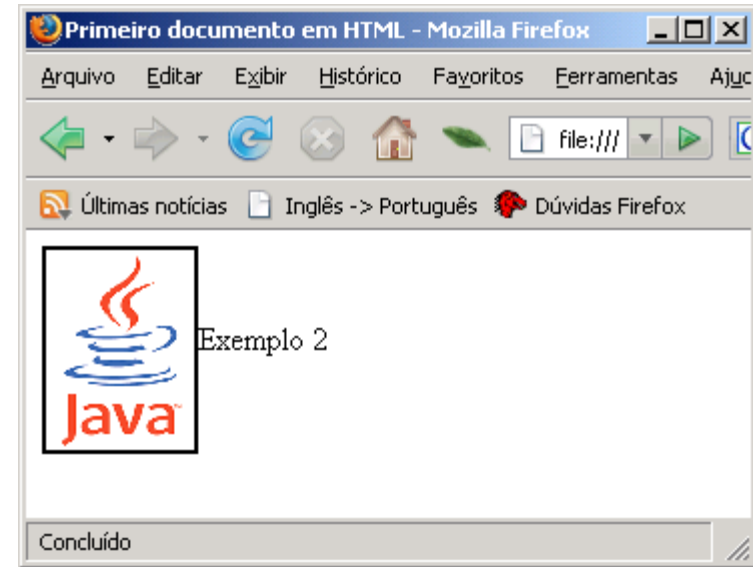
```
1 <html>
2 <head>
3   <meta content="text/html; charset=ISO-8859-1"
4   http-equiv="content-type">
5   <meta name="author" content="Jarley Nóbrega">
6   <meta name="keywords" content="exemplo">
7   <meta name="description" content="Exemplo de página html">
8   <title>Primeiro documento em HTML</title>
9 </head>
10
11 <body>
12 <p>Alô Mundo !!!
13 <br>
14 </p>
15 </body>
16 </html>
```



Inserindo imagens com a tag img

■ Atributos de *img*:

- `src="URL da imagem"`
- `alt="descrição da imagem"`
- `border="tamanho da borda"`
- `align="posição da imagem"`
- `width="largura em pixels"`
- `height="altura em pixels"`



```
Exemplo 2
```

Inserindo links com a tag a

- A tag *a* introduz uma referência para outro documento na rede

- Atributos

- `href="URL do documento referenciado"`

```
<a href="requisitos.html">Requisitos do Projeto</a>
```

- `target="janela onde será carregado o documento"`

```
<a href="http://www.jarley.com"target="frame1">  
Site da Disciplina</a>
```

- URL's relativas x URL's absolutas (qual a melhor?)

Acessando partes de um documento

- É possível carregar o documento destino já em um ponto específico de seu conteúdo:

```
<a href="requisitos.html#funcionais">Requisitos do  
Projeto</a>
```

□ Em "requisitos.html" é preciso definir:

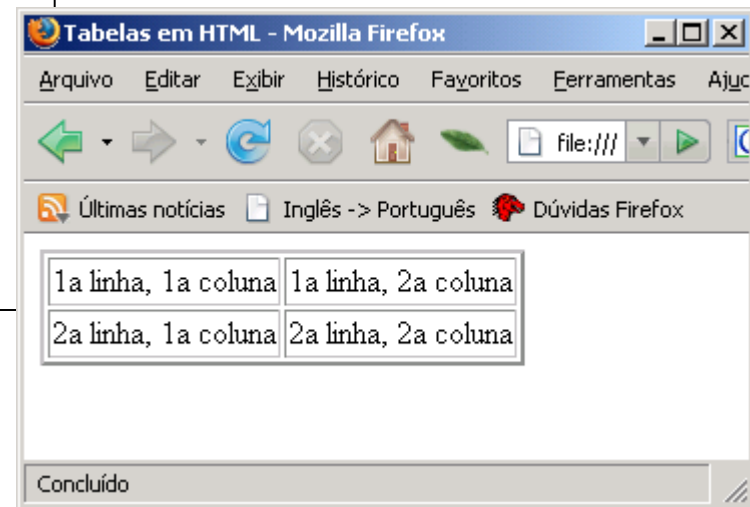
```
<a name="funcionais">Requisitos Funcionais</a>
```

Tabelas em HTML

■ Estrutura geral

```
<table border="2">
<tr>
  <td>1a linha, 1a coluna</td>
  <td>1a linha, 2a coluna</td>
</tr>

<tr>
  <td>2a linha, 1a coluna</td>
  <td>2a linha, 2a coluna</td>
</tr>
</table>
```



Tabelas em HTML

- Existe também a tag <th> para definir um cabeçalho.

```
<table border="2">
<th>
  <td>1a linha, 1a coluna</td>
  <td>1a linha, 2a coluna</td>
</th>

<tr>
  <td>2a linha, 1a coluna</td>
  <td>2a linha, 2a coluna</td>
</tr>
</table>
```


Atributos de Tabelas

- Existem atributos específicos do elemento `table`

- ex.: `border="2"`

- Existem atributos específicos do elemento `td`

- ex.: `nowrap` (impede que o texto de uma célula seja dividido em várias linhas)

- Outros atributos podem ser usados com `table`, `tr` ou `td`

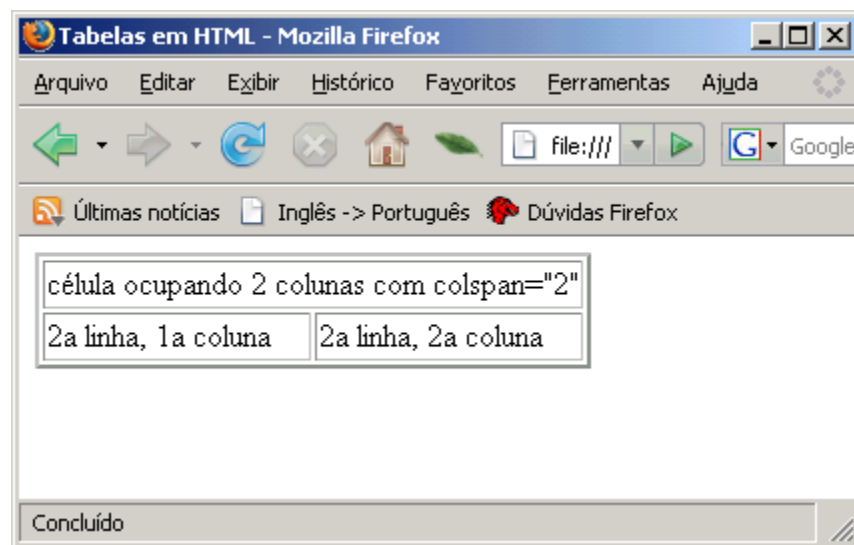
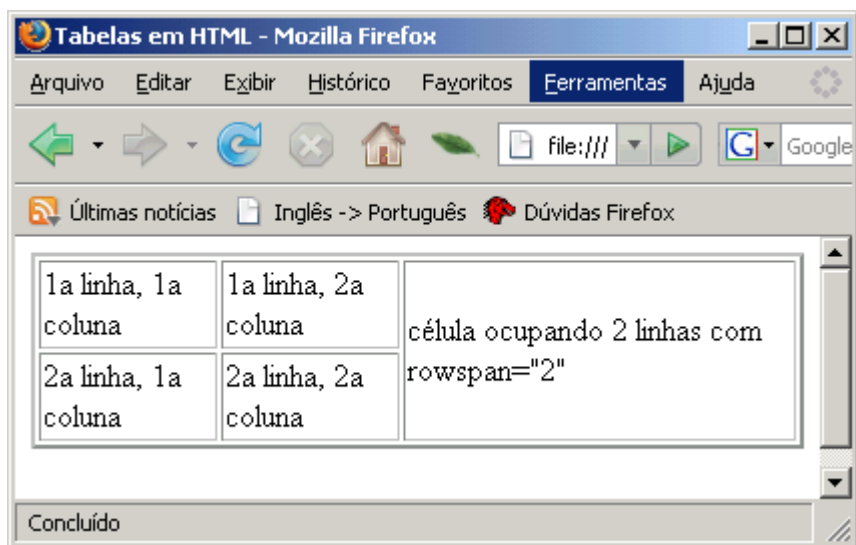
- `width="largura em pixels ou percentual"`

- `bgcolor="#RRGGBB"`

- `align="left | right | center"`

Os atributos colspan e rowspan

- Os atributos de `td`, que fazem uma célula ocupar mais de uma coluna ou mais de uma linha



Especificando o tamanho com width

- O atributo `width` pode ser usado para especificar
 - O tamanho total da tabela com relação à página HTML onde ela está
 - O tamanho de uma determinada célula com relação à tabela onde ela está
- O tamanho pode ser especificado em pixels ou em um valor percentual
 - Exemplos:
 - `<table width="250">` (define o comprimento da tabela em 250 pixels)
 - `<table width="50%">` (define o comprimento da tabela como 50% do comprimento da página)
 - `<td width="50%">` (define o comprimento da célula como 50% do comprimento da tabela)

Formulários

- São utilizados para coletar dados, por exemplo, através de campos de texto
 - Os dados coletados podem ser submetidos para processamento em algum servidor
- Criados com a *tag* `<form></form>`
- Não podem ser aninhados
- Estrutura geral

```
<form name="idFormulario" method="GET ou POST"
      action="URL">
    Campos e botões do formulário
</form>
```

Submissão de formulários

- O que acontece quando submetemos um formulário?
 1. O browser envia todos os dados dos campos do formulário para o servidor web
 - O servidor ativa a URL especificada em `action` para tratar os dados
 - O método (`GET` ou `POST`) especificado em `method` determina como os dados enviados para o servidor serão acessados
 - Se nada for especificado, o padrão é `GET`

Diferença entre GET e POST

- Os dados em um método `GET` são enviados dentro da URL
 - É limitado em relação ao tamanho dos dados enviados
- Os dados em um método `POST` trafegam dentro da mensagem HTTP
 - Os dados não são visíveis na URL
 - O tamanho dos dados enviados é ilimitado
- `POST` normalmente é o método mais adequado para passar parâmetros para o servidor

Campos de formulários

- Campos texto
- Campos de senha
- Campos escondidos
- Radio buttons
- Check boxes
- Botões
 - ☐ simples
 - ☐ de submissão
 - ☐ de cancelamento (reset)

—————→ Inseridos usando a tag `input`

■ Áreas de texto —————→

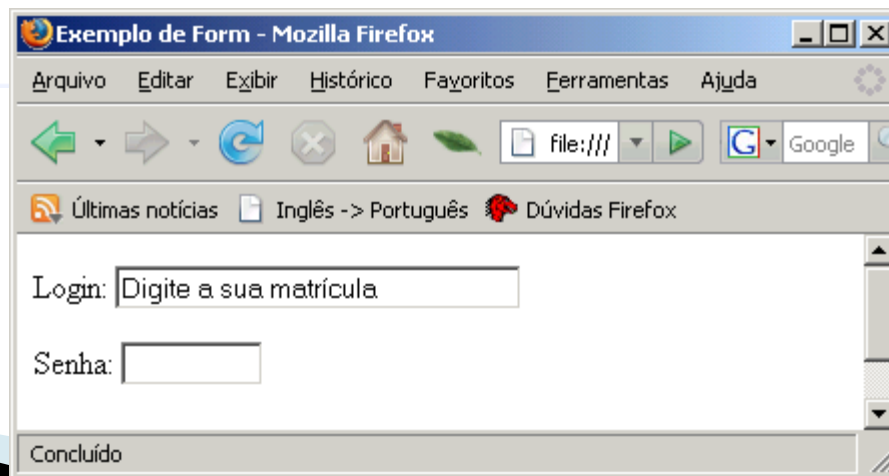
Inseridos usando a tag `textarea`

■ Listas (combo boxes) —————→

Inseridos usando a tag `select`

Campos texto, de senha e escondidos

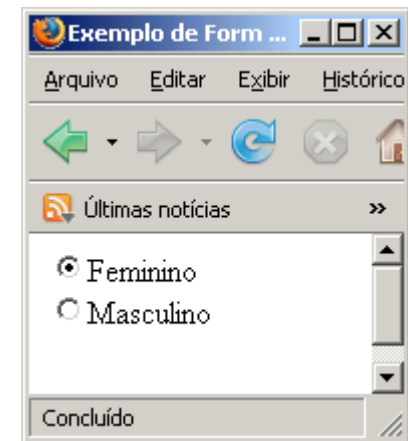
```
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1"
  http-equiv="content-type">
  <title>Exemplo de Form</title>
</head>
<body>
<form name="exemplo" method="GET"
  action="/processaExemplo">
  <p>Login: <input type="text" name="login" size="30"
    maxlength="15" value="Digite a sua matrícula">
  </p>
  <p>Senha: <input type="password" name="senha" size="8"
    maxlength="8">
  </p>
  <p><input type="hidden" name="sistema" value="teste">
</form>
<br>
</body>
</html>
```



Radio buttons

- Um conjunto de radio buttons tem o mesmo valor para o atributo `name`
 - Para seleccionar previamente um deles, usa-se o atributo `checked`

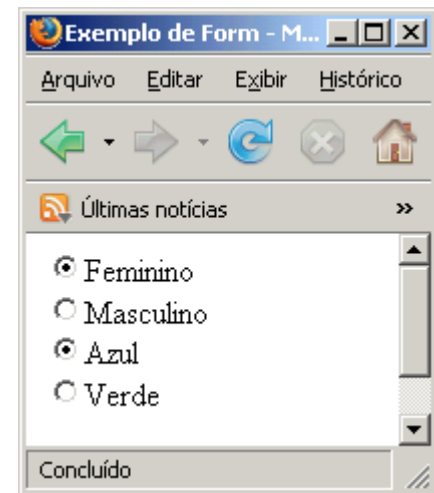
```
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1"
  http-equiv="content-type">
  <title>Exemplo de Form</title>
</head>
<body>
<form name="exemplo" method="GET"
  action="/processaExemplo">
  <input type="radio" name="sexo" value="F"
    checked>Feminino<br>
  <input type="radio" name="sexo" value="M">Masculino<br>
</form>
<br>
</body>
</html>
```



Radio buttons

- Radio buttons com nomes diferentes são considerados campos diferentes
- No exemplo abaixo, sexo e olhos são submetidos ao servidor

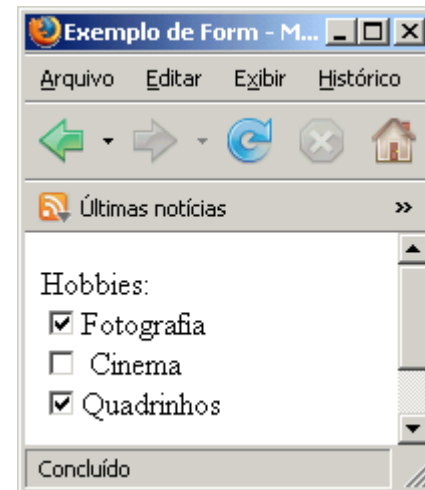
```
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1"
  http-equiv="content-type">
  <title>Exemplo de Form</title>
</head>
<body>
<form name="exemplo" method="GET"
  action="/processaExemplo">
  <input type="radio" name="sexo" value="F"
    checked>Feminino<br>
  <input type="radio" name="sexo" value="M">Masculino<br>
  <input type="radio" name="olhos" value="azul"
    checked>Azul<br>
  <input type="radio" name="olhos" value="verde">Verde<br>
</form>
<br>
</body>
</html>
```



Check boxes

- Para seleccionar previamente um deles, usa-se o atributo checked

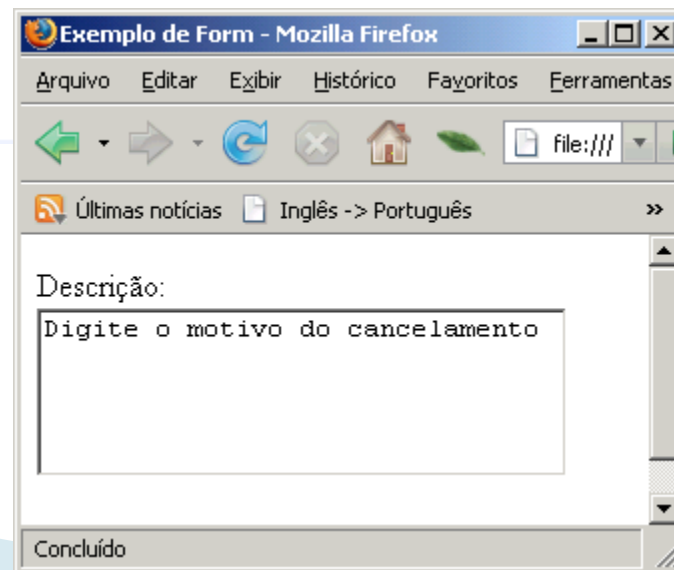
```
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1"
  http-equiv="content-type">
  <title>Exemplo de Form</title>
</head>
<body>
<form name="exemplo" method="GET"
  action="/processaExemplo">
  <p>Hobbies:<br>
  <input type="checkbox" name="hobby" value="1"
    checked>Fotografia<br>
  <input type="checkbox" name="hobby" value="2">
    Cinema<br>
  <input type="checkbox" name="hobby" value="3"
    checked>Quadrinhos<br></p>
</form>
<br>
</body>
</html>
```



Áreas de texto

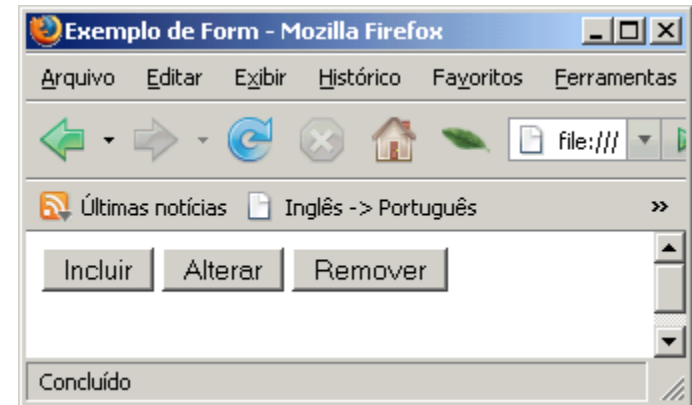
- Para campos de texto maiores que 1 linha

```
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1"
  http-equiv="content-type">
  <title>Exemplo de Form</title>
</head>
<body>
<form name="exemplo" method="GET"
  action="/processaExemplo">
  <p>Descrição:<br>
  <textarea name="descricao" rows="4" cols="30">Digite o motivo do cancelamento
  </textarea></p>
</form>
<br>
</body>
</html>
```



Botões

```
<input type="button" value="Incluir">  
<input type="button" value="Alterar">  
<input type="button" value="Remover">
```



- Podem ser específicos para submissão dos dados ou para “limpar” o formulário

```
<input type="submit" value="Enviar">  
<input type="reset" value="Limpar">
```

- O botão de reset recoloca os campos com os seus valores iniciais

Botões

■ Diferenças entre os tipos "button" e "submit"

□ Button:

- Define um botão simples. Para usá-lo precisamos definir um evento e uma ação
- ex.: `onClick="processaFormulario()"`
- A ação pode ser uma chamada a um código em JavaScript

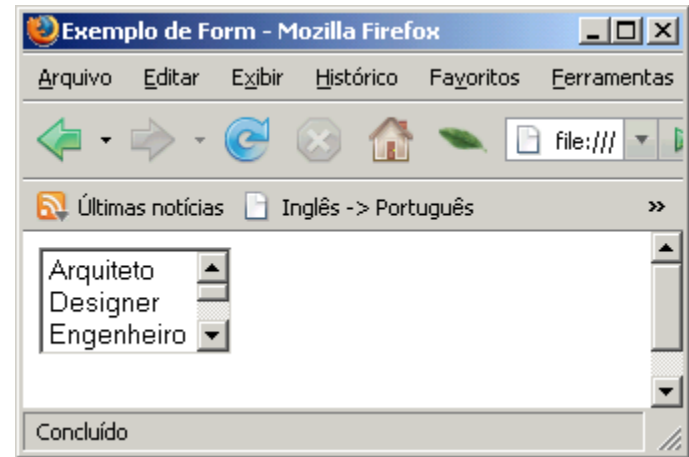
□ Submit

- Submete o formulário para a URL definida no atributo `action` da tag `<form>`

Listas (combo-boxes)

- Criadas através das tags `select` e `option`

```
<select name="profissao" size="3">
  <option value="1">Arquiteto</option>
  <option value="2">Designer</option>
  <option value="3">Engenheiro</option>
  <option value="4">Médico</option>
  <option value="5">Veterinário</option>
</select>
```



- Quando `size` não é especificado, a lista é apresentada como um `ComboBox`, que é uma lista com `size` igual a 1

Listas (combo-boxes)

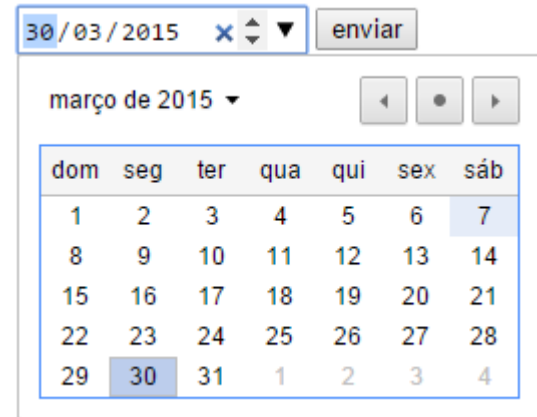
- Pode-se permitir a seleção de mais de uma opção com o atributo `multiple`
- Pode-se selecionar previamente alguma opção com o atributo `selected`

Campo Date

- Elemento utilizado para inserir datas.

`<input type="date" name="data" />`

- A data é enviada ao servidor no formato
 - AAAA-MM-dd
 - 2015-03-11



The image shows a date picker interface. At the top, there is a text input field containing '30/03/2015' with a clear button (X) and a dropdown arrow. To the right of the input field is a button labeled 'enviar'. Below the input field, there is a dropdown menu showing 'março de 2015' with a downward arrow. To the right of the dropdown menu are three navigation buttons: a left arrow, a center dot, and a right arrow. Below these elements is a calendar table with days of the week as headers and dates as content.

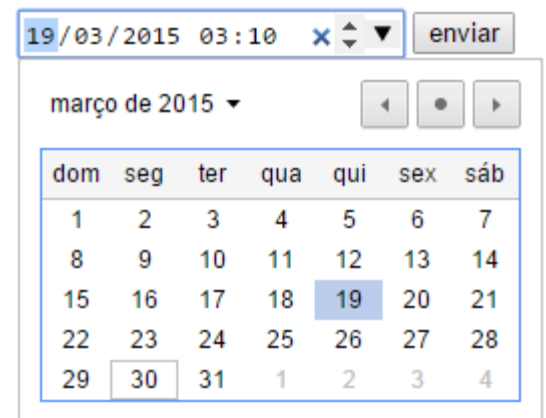
dom	seg	ter	qua	qui	sex	sáb
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Campo Datetime-local

- Elemento utilizado para inserir data e hora.

`<input type="datetime-local" name="data" />`

- Permite a inserção automática da data, porém a hora deve ser digitada pelo usuário.
- Envia o dado para o servidor da seguinte forma:
 - AAAA-MM-ddTHH:mm
 - 2015-03-21T03:10



The screenshot shows a web form with a datetime-local input field. The field contains the text "19/03/2015 03:10". To the right of the field is a button labeled "enviar". Below the field is a date picker calendar for March 2015. The calendar shows the days of the week (dom, seg, ter, qua, qui, sex, sáb) and the dates (1 to 31). The date 19 is highlighted in blue.

dom	seg	ter	qua	qui	sex	sáb
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Campo email

- Impede o envio dos dados ao servidor caso o valor passado não represente um email.

```
<input type="email" name="mail" />
```

- Formato:

- email@dominio

! Inclua um "@" no endereço de e-mail. "asd" está com um "@" faltando.

Campo file

- Permite enviar arquivos ao servidor.
- Para enviar arquivos ao servidor, é preciso definir o formulário como mostrado na figura.

```
<form action="NewServlet1" enctype="multipart/form-data" method="post">  
    <input type="file" name="arquivo" />  
    <input type="submit" value="enviar"/>  
</form>
```

Campo Month

- Permite especificar o mês e o ano;

```
<input type="month" name="mes" />
```

- A data é passada no seguinte formato:

- AAAA-MM

- 2014-03

março de 2015 x ▼ enviar

março de 2015 ◀ ● ▶

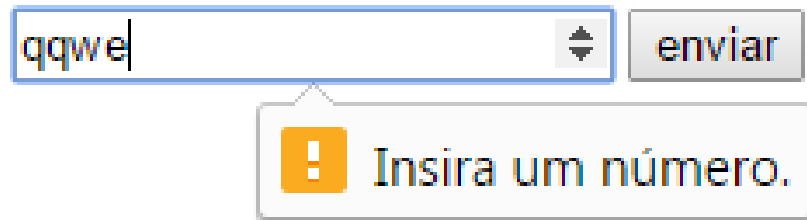
dom	seg	ter	qua	qui	sex	sáb
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Campo Number

- Permite inserir apenas números;

```
<input type="number" name="numero" />
```

- Impede que o usuário digite letras.



qqwe

enviar

! Insira um número.

Campo Time

- Permite inserir Horas e Minutos

```
<input type="time" name="hora" />
```

- Os dados são passados no formato:

- HH:mm

- 12:12



A screenshot of a web browser interface. It features a time input field with a blue border, containing the text '12:12' in orange and blue. To the right of the time is a small 'x' icon and a vertical double-headed arrow icon. To the right of the input field is a gray button with the text 'enviar' in blue.

Campo Semana

- Permite inserir uma determinada semana do ano.

```
<input type="week" name="semana" />
```

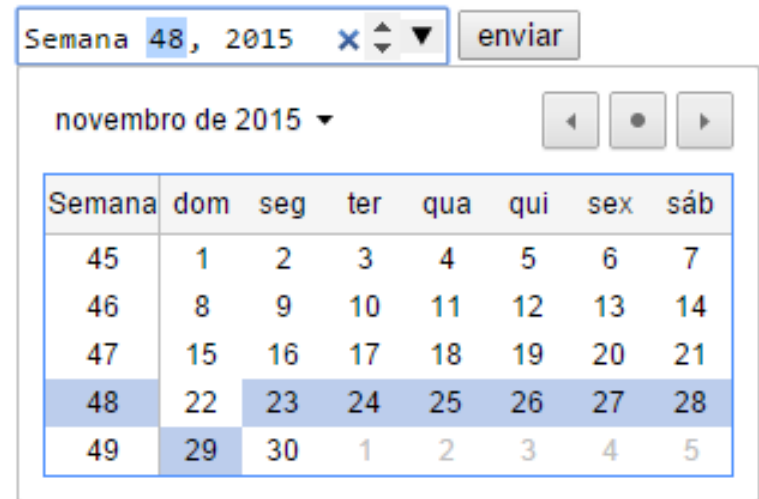
- Os dados são passados no formato:

- AAAA-WSS

- 2015-W48

- O valor 48

Representa a
Quadragésima oitava
Semana do ano.



Semana 48, 2015 x ▾ enviar

novembro de 2015 ▾ ◀ ● ▶

Semana	dom	seg	ter	qua	qui	sex	sáb
45	1	2	3	4	5	6	7
46	8	9	10	11	12	13	14
47	15	16	17	18	19	20	21
48	22	23	24	25	26	27	28
49	29	30	1	2	3	4	5

Campo range

- Apresenta uma barra horizontal com um intervalo de valores que podem ser selecionados pelo desenvolvedor.

```
<input type="range" name="intervalor" min=0  
max=10/>
```

