

Aula 12

Aplicações WEB

Visão geral:

- HTML, CSS, JAVASCRIPT
- Container WEB Tomcat
- Primeiro Servlet

Fundamentos

<HTML>

HTML

- HyperText Markup Language
- Arquivo texto contendo marcas padronizadas delimitando elementos que formam um documento hierárquico
- As marcas são do tipo
- `<Marca [atrib1="val1" ...]> [texto] </Marca>`
- A versão atual é a 4.01
- Órgão regulamentador: W3 Consortium
- www.w3.org/MarkUp

HTML: Seções Básicas

```
<html>
```

```
  <head>
```

```
    <title>Título da  
Página</title>
```

```
  </head>
```

```
  <body>
```

```
    Este é o corpo da página!
```

```
  </body>
```

```
</html>
```

HTML: Marcas mais comuns

- Link

`Links`

- Figura

``

- Linha horizontal

`<hr>`

- Quebras de linha

`
 ou <p> </p>`

HTML: Marcas mais comuns

- Formatar letra (tipo, tamanho, cor, ...)

```
<font face="Verdana" size="-1"  
color="#FFFFFF">Alguma coisa</font>
```

- Efeitos nas letras

```
<b>Negrito</b> ou <strong></strong>
```

```
<em>Itálico</em> ou <i></i>
```

```
<u>Sublinhado</u>
```

HTML: Marcas mais comuns

- Alinhamento

`<center>Centralizado</center>`

`<div align= "center|left|right|justify">
 Texto</div>`

HTML: Marcas mais comuns

- Formulário

```
<form name="form" method="post"  
  action="/inscrever">
```

```
<input name="email" type="text" id="email">
```

```
<input name="btnInscrever" type="submit" id="btnInscrever"  
  value="Inscrever">
```

```
</form>
```

HTML: Tabela

```
<table border="1">
```

```
  <tr>
```

```
    <td>Linha 1 Coluna 1</td>
```

```
    <td>Linha 1 Coluna 2</td>
```

```
  </tr>
```

```
  <tr>
```

```
    <td>Linha 2 Coluna 1</td>
```

```
    <td>Linha 2 Coluna 2</td>
```

```
  </tr>
```


```
  <tr>
```

```
    <td colspan="2">Linha 3 Coluna 1</td>
```

```
  </tr>
```

```
</table>
```

Linha 1 Coluna 1	Linha 1 Coluna 2
Linha 2 Coluna 1	Linha 2 Coluna 2
Linha 3 Coluna 1	

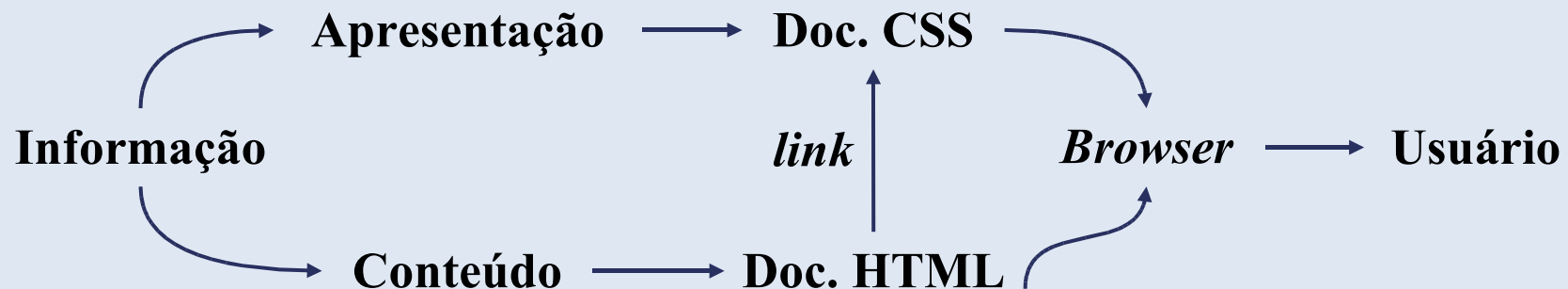


Cascading Style Sheets

Cascading Style Sheets (CSS)

É um mecanismo simples para adicionar estilo (ex.: fontes, cores, espaçamento) aos documentos web

Possibilita um design muito mais sofisticado do que o HTML sozinho, pois separa o conteúdo da apresentação



A Estrutura de um CSS

- O arquivo .css é do tipo texto e pode conter
- diversas declarações que seguem as regras de uma gramática própria
- Cada declaração é formada por um seletor e uma definição

Diagram illustrating the structure of a CSS declaration:

```
seletor → body {  
    nome da propriedade → font-family: Verdana, sans-serif;  
    → font-size: 1em;  
    → text-align: justify;  
    color: #FF0000  
    }  
valor ← (points to the values: Verdana, sans-serif, 1em, justify, #FF0000)
```

A Estrutura de um CSS

- O seletor refere-se a um elemento no documento HTML, que pode ser um elemento padrão ou uma classe criada pelo designer
- A declaração descreve como este elemento deve ser desenhado pelo browser
- `<STYLE type=text/css></STYLE>`

Referenciando um CSS

- Na página HTML que for usar um CSS, coloque a seguinte linha na porção <HEAD>:
- `<link href="arquivo.css" rel="stylesheet" type="text/css">`
- Atenção: normalmente os browsers fazem cache dos arquivos .css
- Isto significa que se você alterar seu arquivo .css, deve forçar uma atualização no browser do usuário, senão as definições permanecerão as mesmas do arquivo antigo



JavaScript

JavaScript

- Parece Java, mas não é!
- Foi criada em 1995 para implementar uma linguagem *script* no lado do cliente (*browser*), em complemento ao HTML, mas também permite processamento no lado do servidor
- É uma linguagem baseada em objetos, multi-plataforma, fracamente tipada e multi-uso
- O código fonte vai junto com a página HTML e é analisado e interpretado pelo *browser* (ou pelo servidor)

Qual a diferença?

- JavaScript é a linguagem original, desenvolvida pela Netscape e Sun
- JScript é a implementação da especificação original pela Microsoft. Também é usada no Windows Script Host (WSH)
- ECMAScript é a especificação padronizada pela ECMA (European Computer Manufacturers Association) em 1997
- Em 1998 foi adotada também pela ISO/IEC

JavaScript: Exemplos

- Escrever no corpo da página
 - `document.write("Alguma coisa.");`
- Abrir uma URL em outra janela
 - `window.open("/links/index.html");`
- Mostrar uma mensagem na tela
 - `alert("Mensagem!");`
- Atenção: JavaScript é case-sensitive!
 - Document é diferente de document !

Objetos, Eventos e Funções

- Existem vários objetos disponíveis:
document, window, form
- Os objetos possuem eventos característicos:
onKeyPress, onChange, onClick, onFocus,
onSubmit, onMouseOver, ...
- Você pode criar suas funções:

```
function nome(param1, param2) {  
    comando;  
  
    return algumValor;  
}
```

Dúvidas?

Aplicações WEB com JAVA

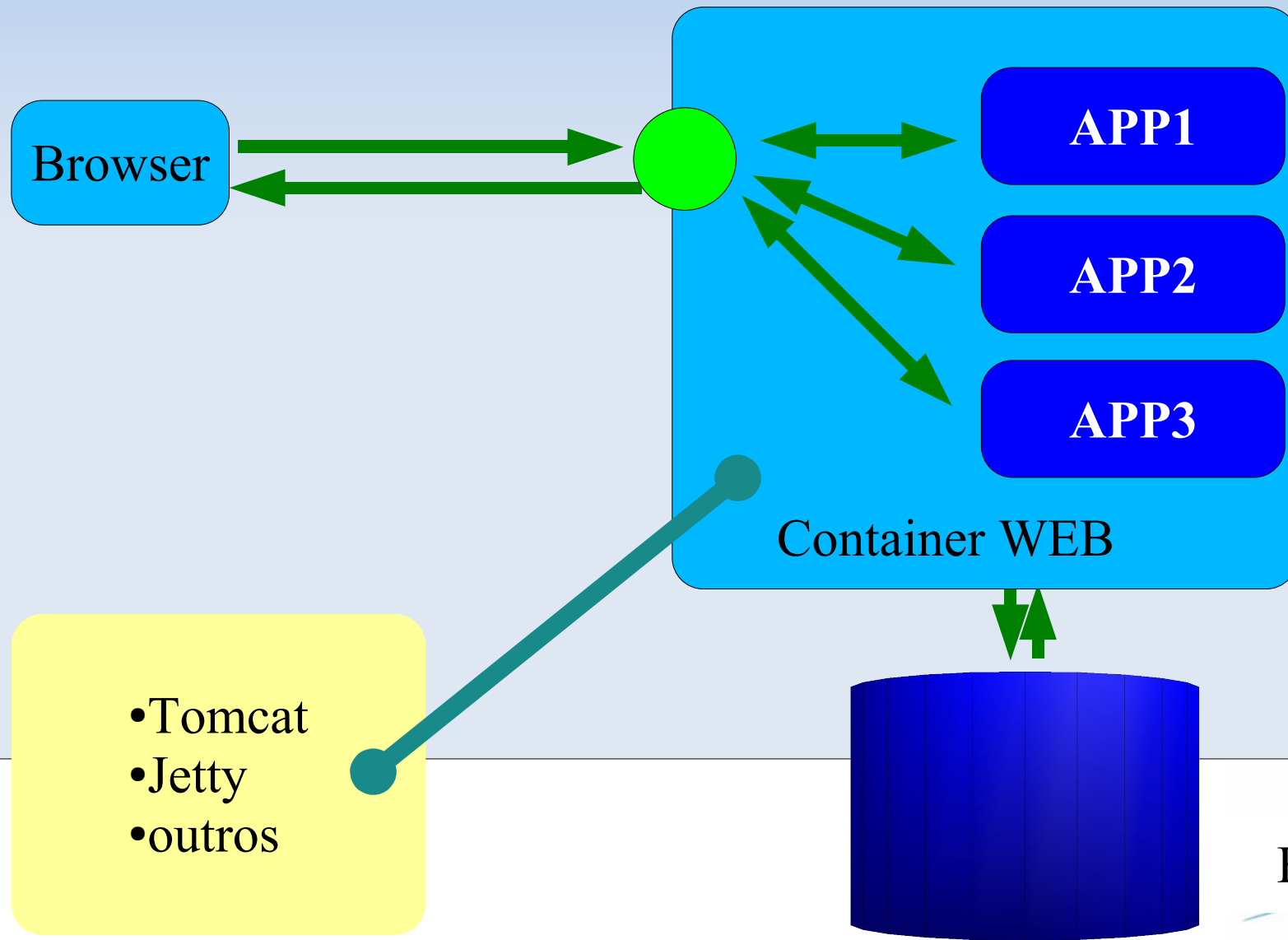
Visão geral:

- Instalação do tomcat
- Configuração do tomcat

Container WEB - tomcat

- <http://www.apache.org>
- Instalando o tomcat;
- Configurando o tomcat

Container WEB



Variáveis de ambiente

- **JAVA_HOME**
 - LOCALIZAÇÃO DO SDK
- **CATALINA_HOME**
 - LOCALIZAÇÃO DO TOMCAT

Inicializando... finalizando...

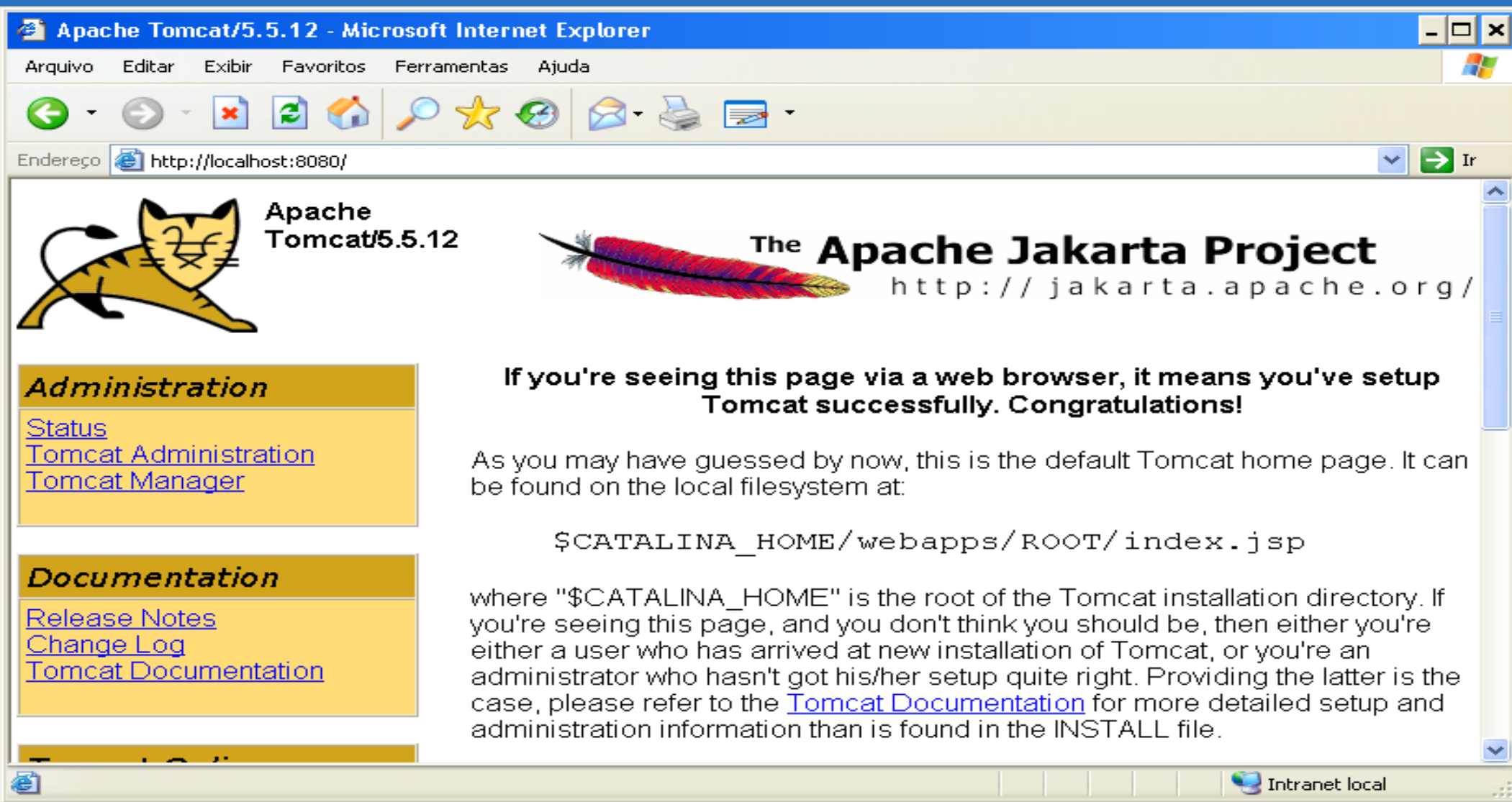
- `bin/startup...`
- `bin/shutdown...`
- `http://localhost:8080`

Inicializando... finalizando...

Tomcat

```
INFO: Initialization processed in 1391 ms
2/Dez/2005 15:48:53 org.apache.catalina.core.StandardService start
INFO: Starting service Catalina
2/Dez/2005 15:48:53 org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/5.5.12
2/Dez/2005 15:48:53 org.apache.catalina.core.StandardHost start
INFO: XML validation disabled
2/Dez/2005 15:48:55 org.apache.catalina.startup.HostConfig deployWAR
INFO: Deploying web application archive loja.war
2/Dez/2005 15:48:57 org.apache.catalina.startup.HostConfig deployWAR
INFO: Deploying web application archive WebModule1.war
- Unable to find config file. Creating new servlet engine config file: /WEB-INF
/server-config.wsdd
- Unable to find required classes (javax.activation.DataHandler and javax.mail.i
nternet.MimeMultipart). Attachment support is disabled.
2/Dez/2005 15:49:00 org.apache.coyote.http11.Http11BaseProtocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
2/Dez/2005 15:49:00 org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
2/Dez/2005 15:49:00 org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/63 config=null
2/Dez/2005 15:49:00 org.apache.catalina.storeconfig.StoreLoader load
INFO: Find registry server-registry.xml at classpath resource
2/Dez/2005 15:49:00 org.apache.catalina.startup.Catalina start
INFO: Server startup in 7703 ms
```

Inicializando... finalizando...



O arquivo conf/server.xml

```
<Connector port="8080" ... />
```

O arquivo conf/tomcat-users.xml

```
<role rolename="manager"/>
```

```
<role rolename="admin"/>
```

```
<user username="admin" password="admin"  
      roles="admin,manager"/>
```

Inicializando... finalizando...

The Apache Jakarta Project
<http://jakarta.apache.org/>



Tomcat Web Application Manager

Message: OK

Manager

[List Applications](#) [HTML Manager Help](#) [Manager Help](#) [Server Status](#)

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/balancer	Tomcat Simple Load Balancer Example App	true	0	Start Stop Reload Undeploy
/host-manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy
/webdav	Webdav Content Management	true	0	Start Stop Reload Undeploy

Deploy

Deploy directory or WAR file located on server

Intranet local

16:17

O diretório webapps

- Diretório onde ficam armazenado as aplicações

Dúvidas??

Exercício

- BAIXAR E INSTALAR O TOMCAT

FIM

Servlets

Objetivos:

- Ser capaz de escrever servlets e executá-los com o tomcat;
- Ser capaz de implantar um servlet no container WEB tomcat;

API Servlet:

- Não é distribuído com JAVA SE;
- Deve ser baixada separadamente;
 - <http://java.sun.com/products/servlet/download.html>
- Documentação:
 - <http://java.sun.com/products/servlet/2.2/javadoc/index.html>

Servlets

- Devem implementar a interface Servlet;
- Alguns métodos são invocados automaticamente pelo servidor;
- Esta interface possui cinco métodos:
 - `void init (ServletConfig config)`
 - `ServletConfig getServletConfig()`
 - `void service(ServletRequest req, ServletResponse resp)`
 - `String getServletInfo()`
 - `void destroy()`

A classe HttpServlet

- Estende GenericServlet que implementa Servlet;
- Define métodos para responder requisições HTTP:
 - doGet, HTTP GET requests
 - doPost, para HTTP POST requests
 - init and destroy, manipular recursos
 - getServletInfo, provê inf. sobre o Servlet

HttpServletRequest:

- Toda chamada para doGet ou doPost recebe um objeto que implementa HttpServletRequest;
- Define métodos para processar requisições HTTP:
 - `String getParameter(String nome)`
 - `Cookie[] getCookies()`
 - `HttpSession getSession()`
 - `String getLocalAddr()`

HttpServletResponse:

- Toda chamada para doGet ou doPost recebe um objeto que implementa HttpServletResponse;
- Define métodos para formular respostas ao cliente HTTP:
 - addCookie(Cookie cookie)
 - ServletOutputStream getOutputStream()
 - PrintWriter getWriter()
 - setContentType(String type)

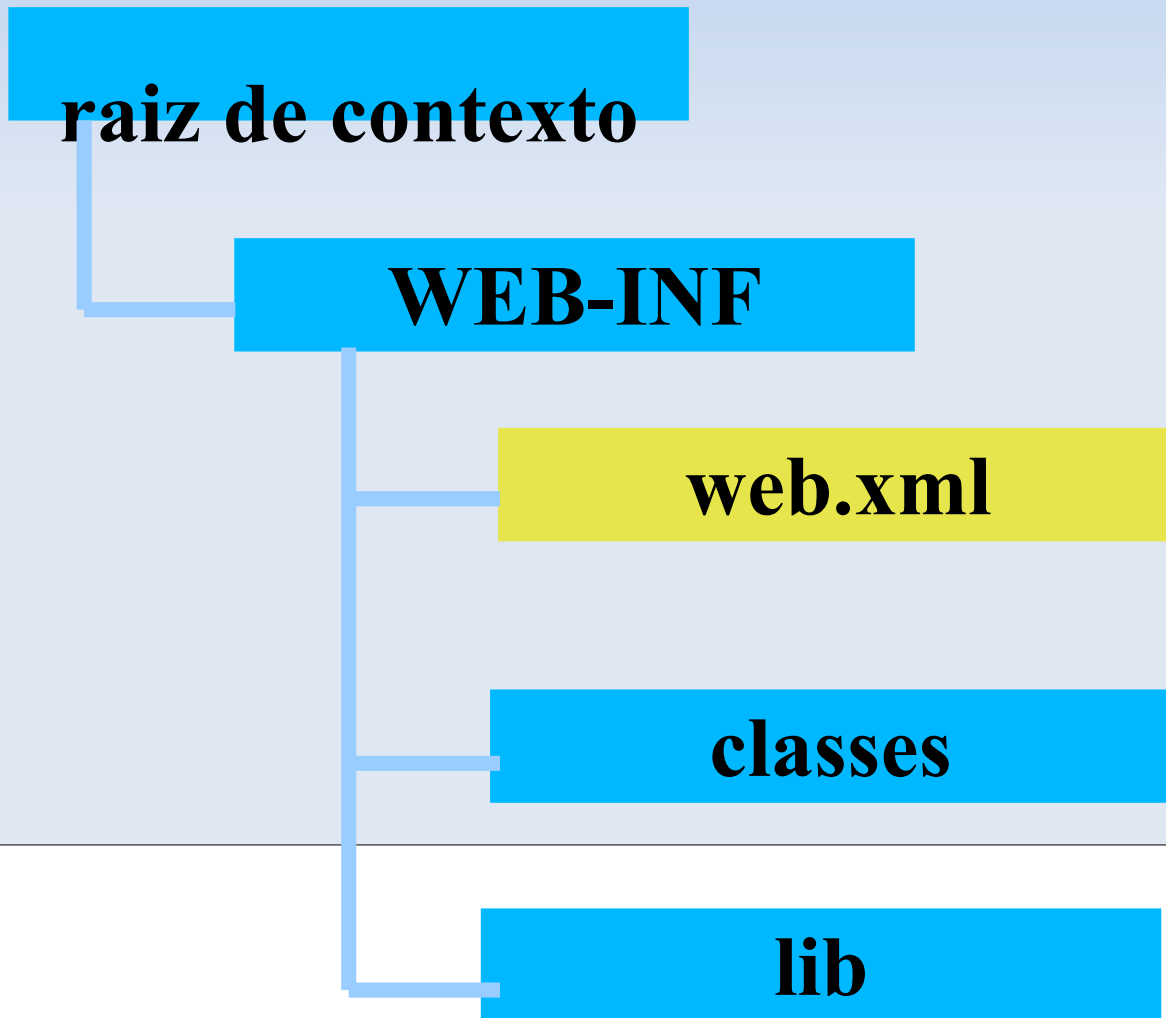
Cabeçalhos de Tipo de Conteúdo:

- MIME (Multipurpose Internet Mail Extensions, extensão de correio da Internet de múltiplos propósitos);
- Alguns tipos de conteúdo MIME:
 - text/html
 - text/plain
 - image/gif
 - application/pdf

O que um app Web deve ter:

- Servlet e suas bibliotecas;
- Um descritor de implantação;
- Seguir a estrutura de diretórios:
 - raiz de contexto
 - o diretório raiz da aplicação (JSP, HTML)
 - WEB-INF
 - web.xml (descritor de implantação)
 - WEB-INF/classes (arquivos .class)
 - WEB-INF/lib (bibliotecas .jar)

Extrutura do diretório:



Um servlet Simples:

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;

public class PrimeiraServlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
        throws ServletException, IOException{
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<HTML>");
        out.println("<HEAD><TITLE>Primeira Servlet</TITLE></HEAD>");
        out.println("<BODY>");
        out.println("<H1>PRIMEIRA SERVLET FUNCIONANDO...</H1>");
        out.println("</BODY>");
        out.println("</HTML>");
        out.close();
    }
}
```

arquivo web.xml

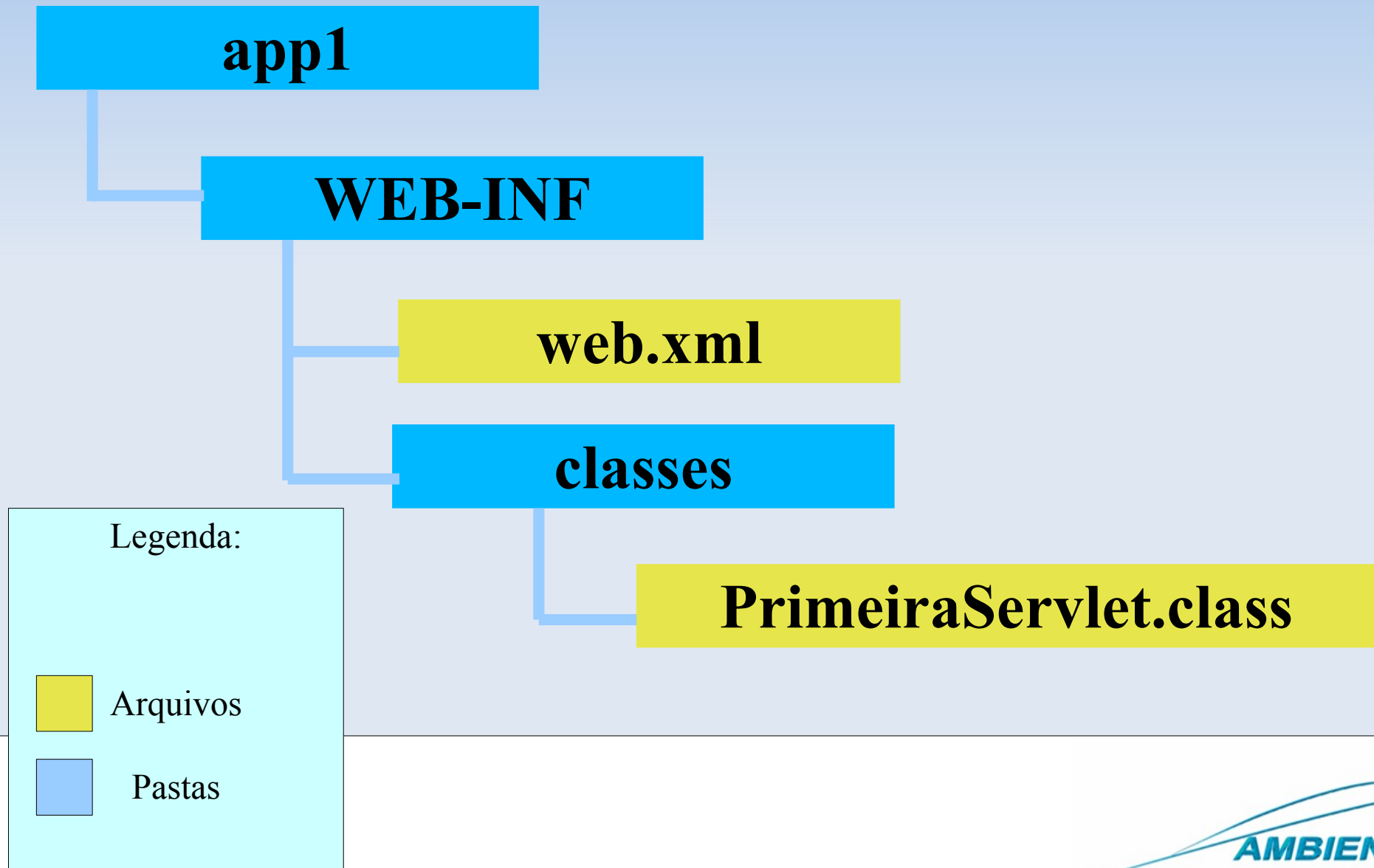
```
<web-app>
  <display-name>
    Exemplo de servlet
  </display-name>

  <servlet>
    <servlet-name>primeira</servlet-name>
    <servlet-class>PrimeiraServlet</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>primeira</servlet-name>
    <url-pattern>/prim</url-pattern>
  </servlet-mapping>

</web-app>
```


Extrutura do diretório:

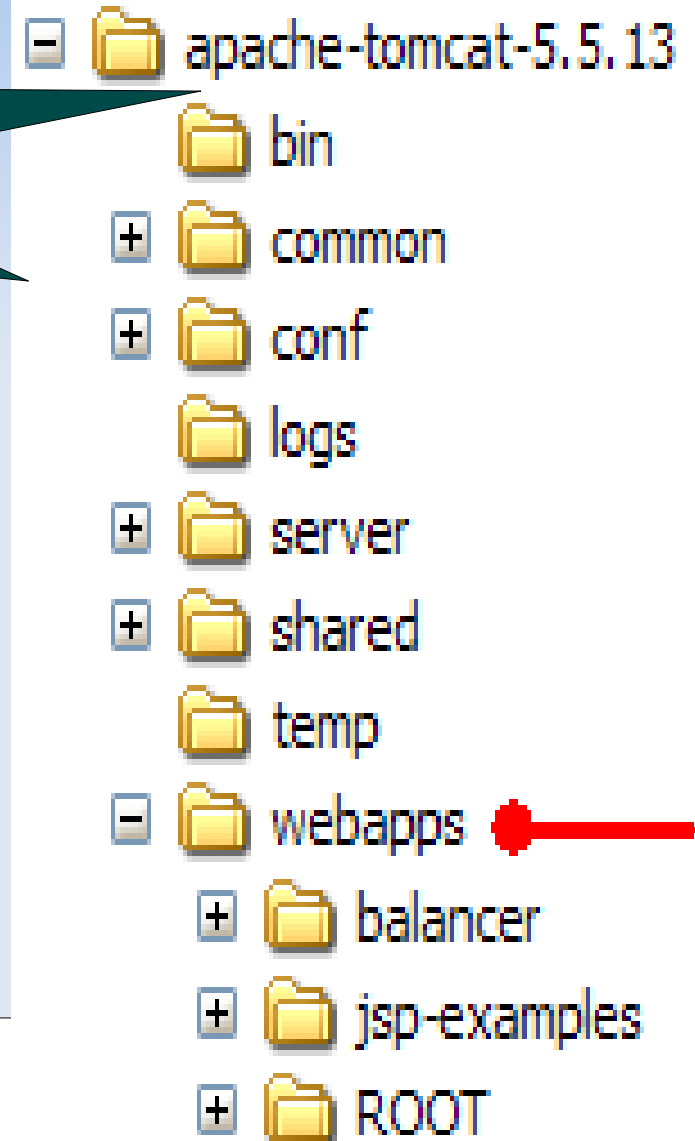


Extrutura do diretório:

**Onde
colocar
isto tudo?**

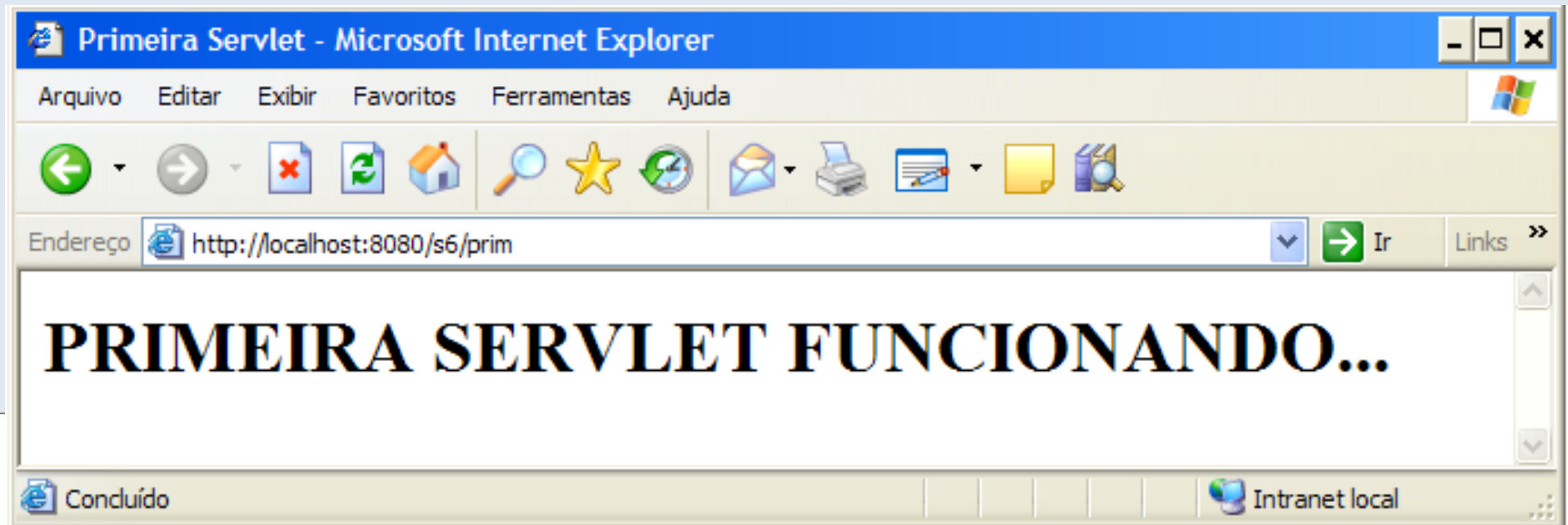
Extrutura do diretório:

webapp



Extrutura do diretório:

- Inicialize o tomcat: bin/startup
- Em um browser digite o endereço:
 - <http://localhost:8080/aula12/prim>



Compactando tudo...

- Costumamos compactar tudo em um arquivo .war para facilitar a distribuição;
- Compacte a pasta primservlet
- Mude seu nome para primservlet.war

Tudo mais fácil...

- Vimos, da forma mais difícil como funciona uma servlet;
- Muitas IDE's no mercado fazem a maior parte do trabalho, deixando para o programador apenas a parte da programação;

Dúvidas?

Exercícios:

- O que é um servlet?
- Cite três métodos das interfaces `HttpServletResponse` e `HttpServletRequest`.
- Qual a estrutura de diretórios de uma aplicação web?
- O que são aplicações `.war`?
- Para que serve o arquivo `web.xml`?

FIM