Globalcode®
The Developers Company

Globalcode Community - Brazil

Research and Education

Agenda – Parte Teórica

1. Introdução
2. Conceitos básicos de Web
3. Desafios no desenvolvimento Web
4. Histórico do desenvolvimento para Internet
5. Plataforma Java 2 Enterprise Edition
6. Arquiteturas Web com J2EE

Agenda – Parte Prática

1. Instalação de um servidor Web Java;
2. Criando um aplicativo Web;
3. Java Server Pages;
4. Java Servlet;
5. Demo;

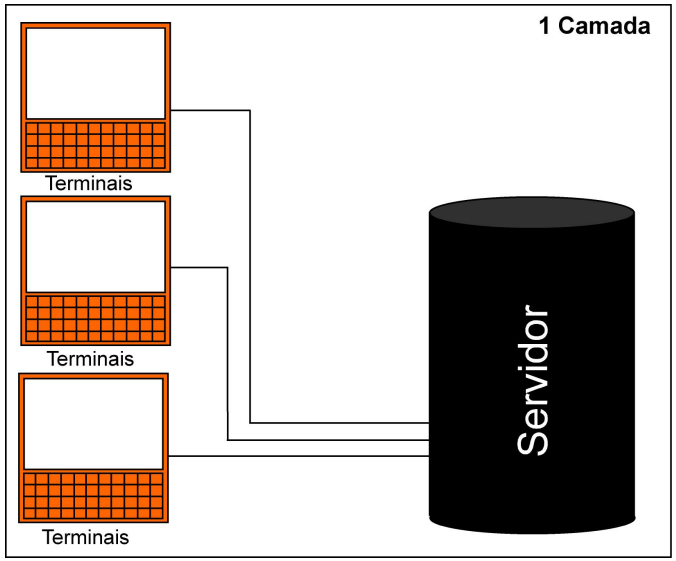
Palestrante

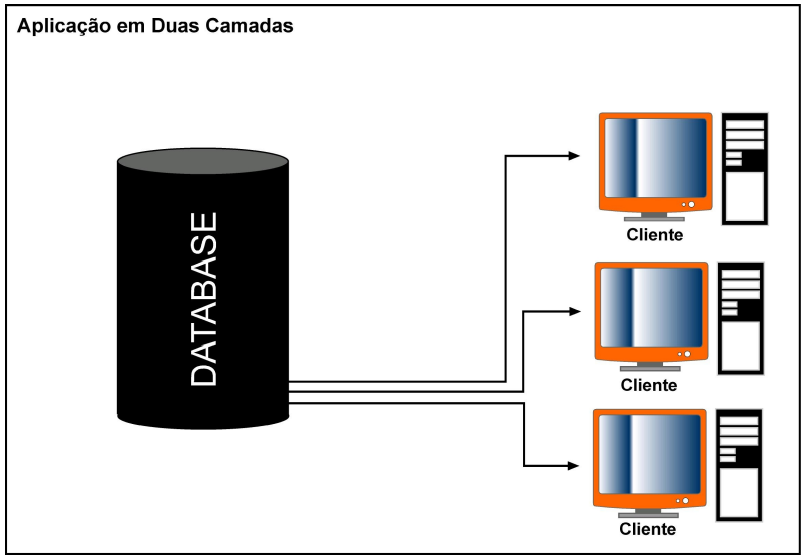
Vinicius Senger – vinicius@globalcode.com.br

- Sócio e fundador da Globalcode, foi instrutor e consultor da Sun e Oracle no Brasil;
- Trabalhou em projetos de grande porte em bancos. Começou a programar com 8 anos e trabalha com desenvolvimento de softwares profissionalmente desde os 13 anos;
- Certificações: Sun Certified Java Programmer e Sun Enterprise Architect – p1;

Introdução

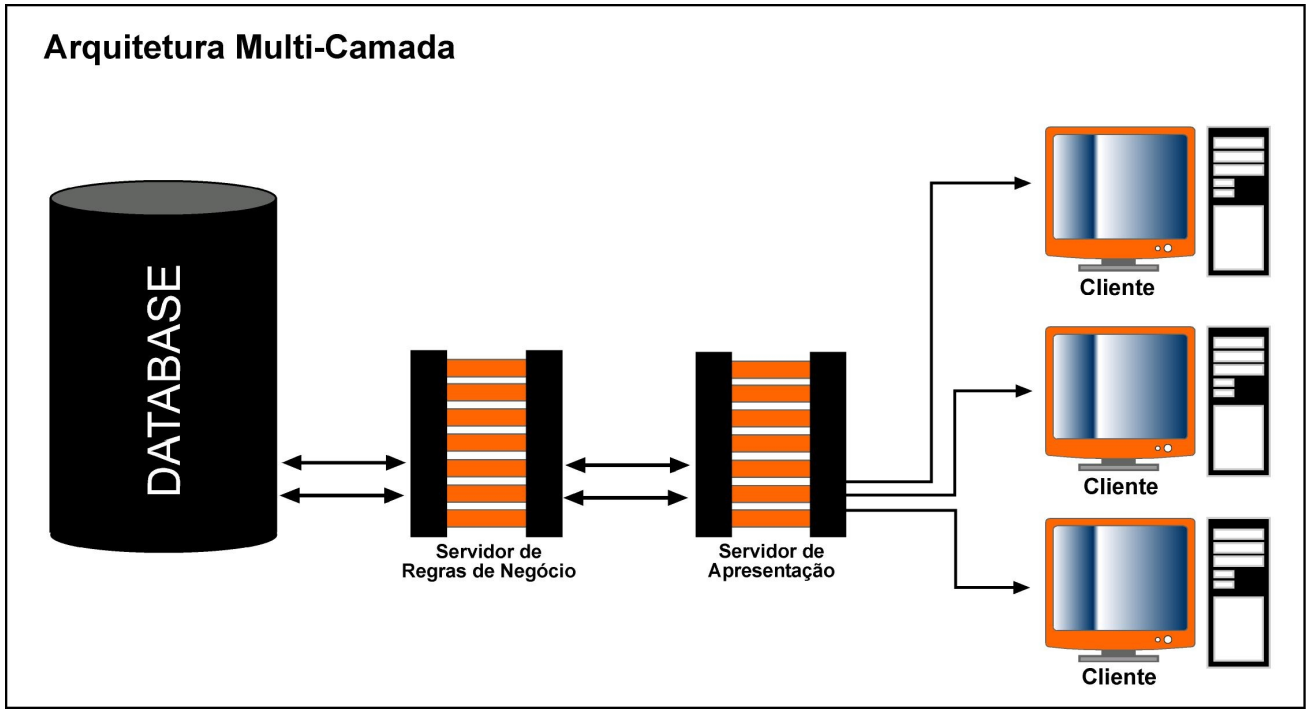
- A primeira arquitetura popular foi a arquitetura mono-camada / monolítica;
- Com o downsizing, aplicativos passaram a ser desenvolvidos em duas camadas: **client-server**;





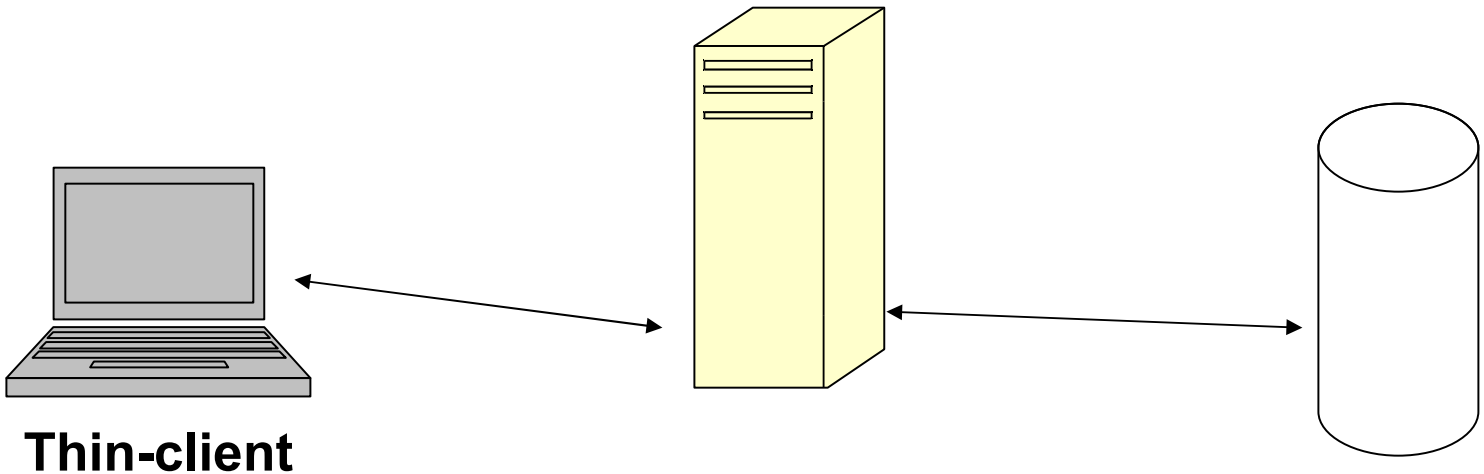
Introdução

- A partir da década de 90, aplicativos passaram a ser desenvolvidos em múltiplas camadas (3, 4 ou mais):



Introdução

- A forma mais popular atualmente para desenvolvimento em três camadas, é utilizando um servidor Web:



Thin-client
Pequenas validações
Terminal burro colorido

Servidor Web
Regras de negócio
Regras de apresentação

Database Server
Regras de armazenamento
Regras de negócio

Introdução

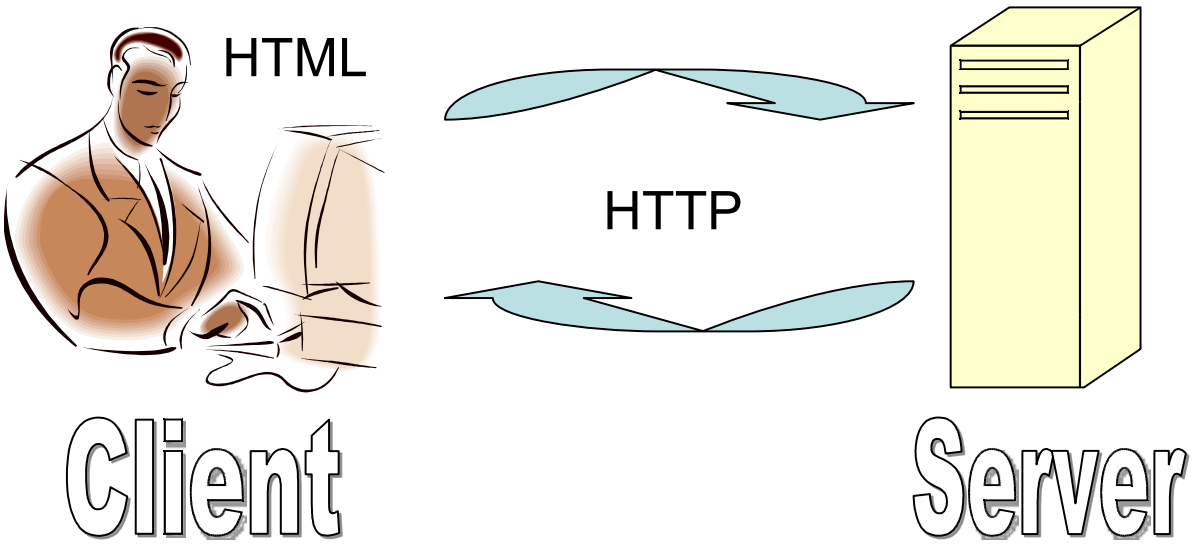
- A arquitetura Web é muito dinâmica para mudanças, pois não temos um aplicativo client nas máquinas dos usuários;
- A arquitetura Web é acessível, pois pode ser disponibilizada na Internet;
- Arquiteturas Web podem atender a grandes demandas;
- Podemos acessar um aplicativo Web com diferentes computadores e dispositivos: celular, PDA's wireless, PC's, Apple, Unix, Torradeiras (TORRADEIRAS?!?!?);

Agenda – Parte Teórica

1. Introdução
2. Conceitos básicos de Web
3. Desafios no desenvolvimento Web
4. Histórico do desenvolvimento para Internet
5. Plataforma Java 2 Enterprise Edition
6. Arquiteturas Web com J2EE

Conceitos Web

- Comunicação client-server ocorre através do protocolo HTTP (**H**yper**T**ext **T**ransfer **P**rotocol);
- HTTP é um protocolo de alto nível baseado no TCP/IP;
- Tipicamente trafegamos arquivos / conteúdo HTML (**H**yper **T**ext **M**arkup **L**anguage)



HTTP

- O protocolo HTTP foi criado para interligar universidades americanas;
- Podemos **enviar dados** (formulários e arquivos) por um browser através de HTTP
- Podemos **receber dados** de um servidor HTTP
- Podemos receber dados em diferentes formatos: HTML, TXT, PDF, Excel, GIF, JPG, BMP, etc.

HTML

- É uma linguagem de marcação interpretada por browsers;
- Com as tags de marcação podemos criar páginas com textos, tabelas, imagens e hyperlinks.
- Páginas HTML são documentos texto que contém código HTML, com a extensão .html ou .htm
- HTML pode ser utilizado para construção de páginas estáticas ou em conjunto com outra linguagem (JSP, ASP, PHP, entre outras) para construção de páginas dinâmicas.

HTML

Exemplo: exemploHTML.html

```
<HTML>
  <HEAD><TITLE> Exemplo documento HTML</TITLE></HEAD>
<BODY>
  Tabela de preços <br> <br>
  <TABLE border="1">
    <TR> <TD> Televisão </TD><TD> R$ 400,00</TD></TR>
    <TR> <TD> Coca-cola </TD><TD> R$ 2,00   </TD> </TR>
  </TABLE>
  <BR><BR>
  <A HREF="http://www.globalcode.com.br">Globalcode </A>
</BODY>
</HTML>
```

CSS

CSS = **C**ascade **S**tyle **S**heet

- Quando utilizamos HTML todas as configurações como fontes, cores e tabelas são colocadas dentro da página HTML.
- Em aplicações com muitas páginas HTML a alteração de estilos tende a ser muito trabalhosa.
- O CSS é utilizado para tirar estas configurações de dentro das páginas HTML, colocando-as dentro de um arquivo com a extensão .css

CSS

- Podemos redefinir as tags existentes no HTML, como a tag para hyper link, tabelas entre outros, ou podemos definir nossas próprias tags.
- Vamos analisar o exemplo exemplo.css que altera a tag para hyper link.

```
a
{
   text-decoration: none;
   font-family: Arial, Helvetica, sans-serif;
   font-weight: bold;
   color: #FFCC33
}
```

CSS

```
<HTML>
  <HEAD>
    <LINK rel="stylesheet" type="text/css"href="exemplo.css" />
    <TITLE> Exemplo documento HTML</TITLE>
  </HEAD>
<BODY>
  Tabela de preços <br> <br>
  <TABLE border="1">
     <TR> <TD> Televisão </TD><TD> R$ 400,00</TD></TR>
     <TR> <TD> Coca-cola </TD><TD> R$ 2,00   </TD> </TR>
  </TABLE>
  <BR><BR>
  <A HREF="http://www.globalcode.com.br">Globalcode </A>
</BODY>
</HTML>
```

JavaScript

- JavaScript é uma linguagem propriamente dita, tem tipos de dados, condicionais, loops e funções;

- O JavaScript foi desenvolvido pela Netscape.

JavaScript

- JavaScript é diferente de Java conceitualmente.
- JavaScript parecido “sintaxamente”;
- JavaScript é utilizada em muitas páginas Web para:
  - melhorar o design das páginas
  - fazer validação de campos em formulários /
  - esconder e exibir campos
  - identificar qual o browser está sendo utilizado
  - entre outras coisas

JavaScript

- O código JavaScript pode ser colocado dentro do HTML ou em um arquivo separado com a extensão .js
- Exemplo testeJS.html
- Vamos analisar o código do testeJS

```
<SCRIPT type="text/javascript">
function validate(){
  if (document.form1.nome.value.length<1){
   alert("Por favor digite seu nome!");
   return false;
  } else return true;
}
</SCRIPT>
```

JavaScript

```
<BODY>
  <FORM name="form1"
              action="exemploHTML.html"
              onsubmit="return validate()">
    Nome
      <INPUT type="text" name="nome" >
      <INPUT type="submit" value="Enviar">
</FORM>
</BODY>
```

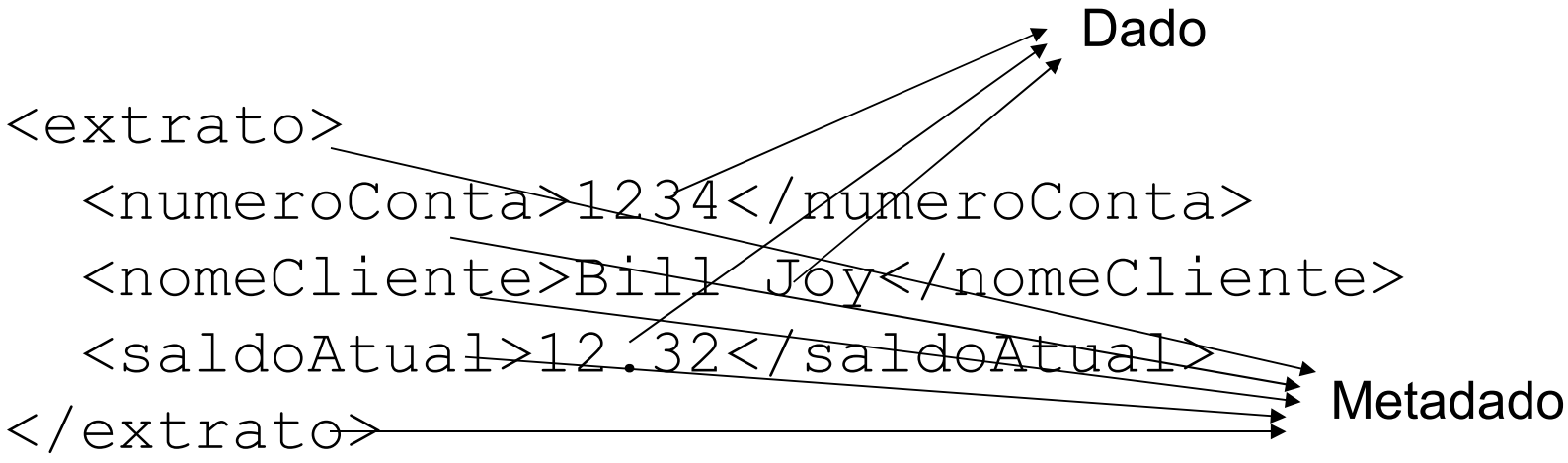
XML

- XML = eXtensible Markup Language
- É um documento texto, assim como HTML também é demarcado por tags;
- É a evolução do documento texto, pois inclui não só o dado texto, mas também o que é aquele dado;
- Suporte a múltiplos idiomas;
- Pode ser validado por um documento de regras de formatação (DTD Data Type Definition)

XML

- Tipicamente utilizado para: troca de documentos entre instituições (governo com bancos), documentos de configuração de aplicativos, geração de PDF's, entre outros.

- Contém dado e meta-dado:



Agenda – Parte Teórica

1. Introdução
2. Conceitos básicos de Web
3. Desafios no desenvolvimento Web
4. Histórico do desenvolvimento para Internet
5. Plataforma Java 2 Enterprise Edition
6. Arquiteturas Web com J2EE

Desafios da Web

- Mescla de tecnologias: Java, JavaScript, CSS, HTML, JSP, SQL, HTTP, XML, etc.
- As telas no browser são pouco dinâmicas;
- O protocolo HTTP é “sessionless”, ou seja, não mantém conexão TCP/IP ativa;
- O cliente tem que requisitar dados para o servidor HTTP;
- O servidor HTTP não pode “empurrar” dados para o cliente sem que ele peça;

Desafios da Web

- Telas com muito dinamismo exigem muito JavaScript, que por sua vez, tem problemas de compatibilidade entre os diversos navegadores do mercado;
- Conclusão: é melhor adotar **mais uma** tecnologia para quando a tela for complexa (tela de pedido, preenchimento de trouble ticket);
- As alternativas para interfaces Web ricas (RIA - Rich Internet Application) são:
  - Java Applet / Java Swing;
  - Flash

Agenda – Parte Teórica

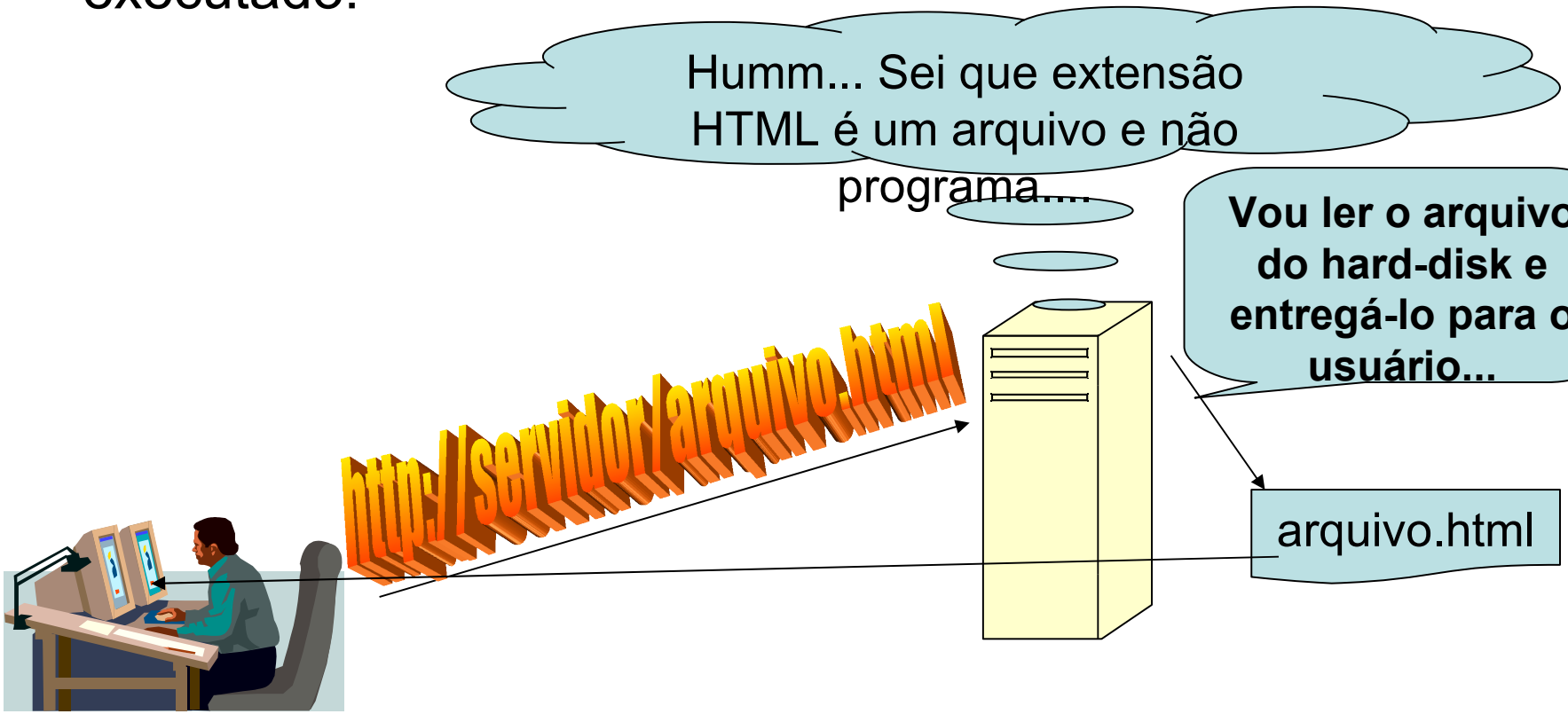
1. Introdução
2. Conceitos básicos de Web
3. Desafios no desenvolvimento Web
4. Histórico do desenvolvimento para Internet
5. Plataforma Java 2 Enterprise Edition
6. Arquiteturas Web com J2EE

Histórico

- O protocolo HTTP foi especificado para trabalhar com arquivos estáticos, ou seja, atua como um protocolo de transferência de arquivos;
- Para aplicativos, tipicamente precisamos de conteúdo HTML com conteúdo dinâmico provido por um banco de dados;
- Foi “adaptado” para servir dados de aplicativos;

Histórico

- Para servir arquivos por HTTP, o seguinte processo é executado:



Histórico

- Para servir arquivos por HTTP, o seguinte p
executado:

Histórico

- Esta técnica é chamada de CGI – Common Gateway Interface;
- Foi a primeira técnica utilizada no desenvolvimento de aplicativos Web;
- O seu problema é que os programas não são residentes, ou seja;
- A cada solicitação, o programa é carregado e em seguida descarregado;

Histórico

- Com Java, o mesmo conceito é utilizado porém os programas (máquina virtual + classes) ficam residentes na memória do servidor em questão;

- Podemos concluir que a arquitetura inicial oferece um menor desempenho e consome mais hardware que Java;

Histórico

- Com Java podemos criar programas que são executados no servidor HTTP através de JavaServlet e JavaServer Page;

- Com Java podemos atender uma demanda maior consumindo menor quantidade de recursos se comparado com “CGI’s puros”;

Agenda – Parte Teórica

1. Introdução
2. Conceitos básicos de Web
3. Desafios no desenvolvimento Web
4. Histórico do desenvolvimento para Internet
5. Plataforma Java 2 Enterprise Edition
6. Arquiteturas Web com J2EE

J2EE para Web

- É a plataforma Java para desenvolvimento Web;
- Tomcat, Websphere, Weblogic, JRun são servidores Web compatíveis com J2EE;
- Para que trabalha com Microsoft: é o IIS do Java;
- Podemos adaptar o Internet Information Server para que adicionar suporta à Java na plataforma Microsoft;

J2EE para Web

- A plataforma J2EE para Web Plataforma é consagrada e de baixo risco;
- Muitos casos de sucesso;
- Desenvolvemos aplicativos Web com Java e J2EE através de:
  - Classes Java comunicam com banco de dados;
  - Classes Java que representam as entidades em objetos;
  - Classes Java que processam dados das entidades;
  - Páginas JSP's

J2EE para Web

- Desenvolvemos aplicativos Web com Java e J2EE através de:
  - Classes Java comunicam com banco de dados;
  - Classes Java que representam as entidades em objetos;
  - Classes Java que processam dados das entidades;
  - Arquivos Web: HTML / CSS / JavaScript
  - Páginas JSP's
  - Componentes JavaServet (classes Java que podem ser acionadas por HTTP)
  - Arquivo de configuração XML
- Um aplitivo Web pode ser compactado em um único arquivo com extensão .war;

Agenda – Parte Teórica

1. Introdução
2. Conceitos básicos de Web
3. Desafios no desenvolvimento Web
4. Histórico do desenvolvimento para Internet
5. Plataforma Java 2 Enterprise Edition
6. Arquiteturas Web com J2EE

Arquiteturas Web

- Podemos trabalhar com arquiteturas simples, onde codificamos tudo em páginas JSP's;
- Podemos trabalhar arquiteturas que dividem as responsabilidades da solução em diversas classes;
- Uma forma tradicional de dividir é conhecida por M.V.C. (Model View and Controller);
- Com MVC podemos reaproveitar uma entidade em diferentes visualizações / "telas";

Parte Prática

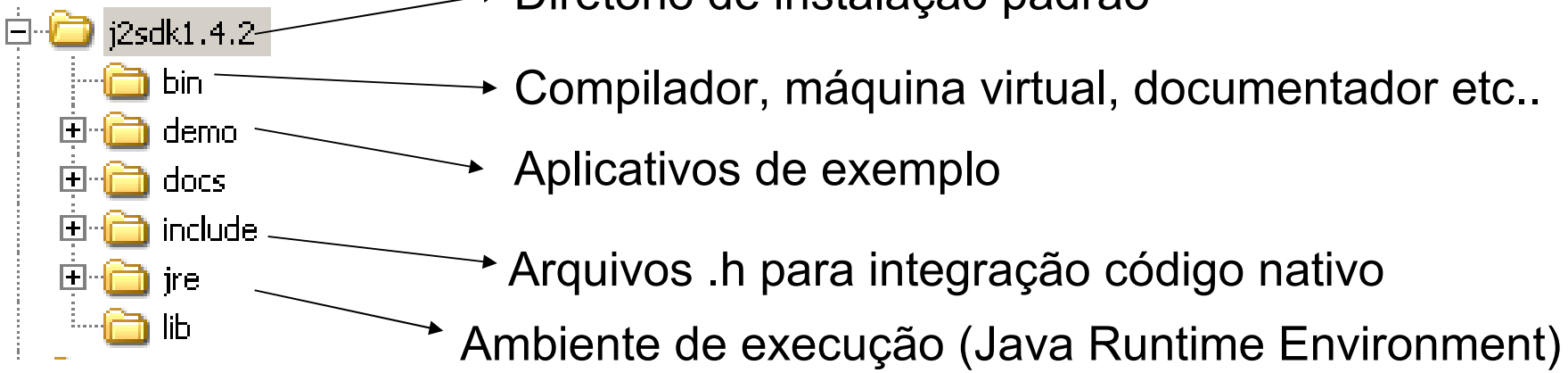
Agenda – Parte Prática

1. Instalação de um servidor Web Java;
2. Criando um aplicativo Web;
3. Java Server Pages;
4. Java Servlet;
5. Demo;

Instalação

- Para instalar um servidor Web Java, antes precisamos instalar o JDK Standard Edition;
- Download a partir da URL: http://java.sun.com/j2se/1.4.2/download.html
- Recomendamos utilizar a versão 1.4.2;
- Pode ser feito o download com o Netbeans;
- No Windows é um executável installshield;
- A instalação é simples, Next até Finish...

Instalação JDK



Instalação JDK

JAVA_HOME & PATH

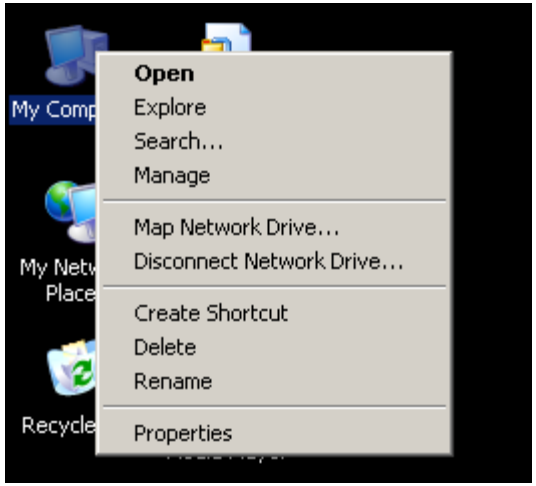
- Para facilitar o uso do compilador e interpretador, recomendamos colocar o diretório c:\j2sdk1.4.2\bin no **PATH**;

- Devemos criar uma variável de ambiente chamada JAVA_HOME, indicando o local de instalação do Kit;

- Esta variável é utilizada para aplicativos que dependem de Java poderem localizar a máquina virtual;

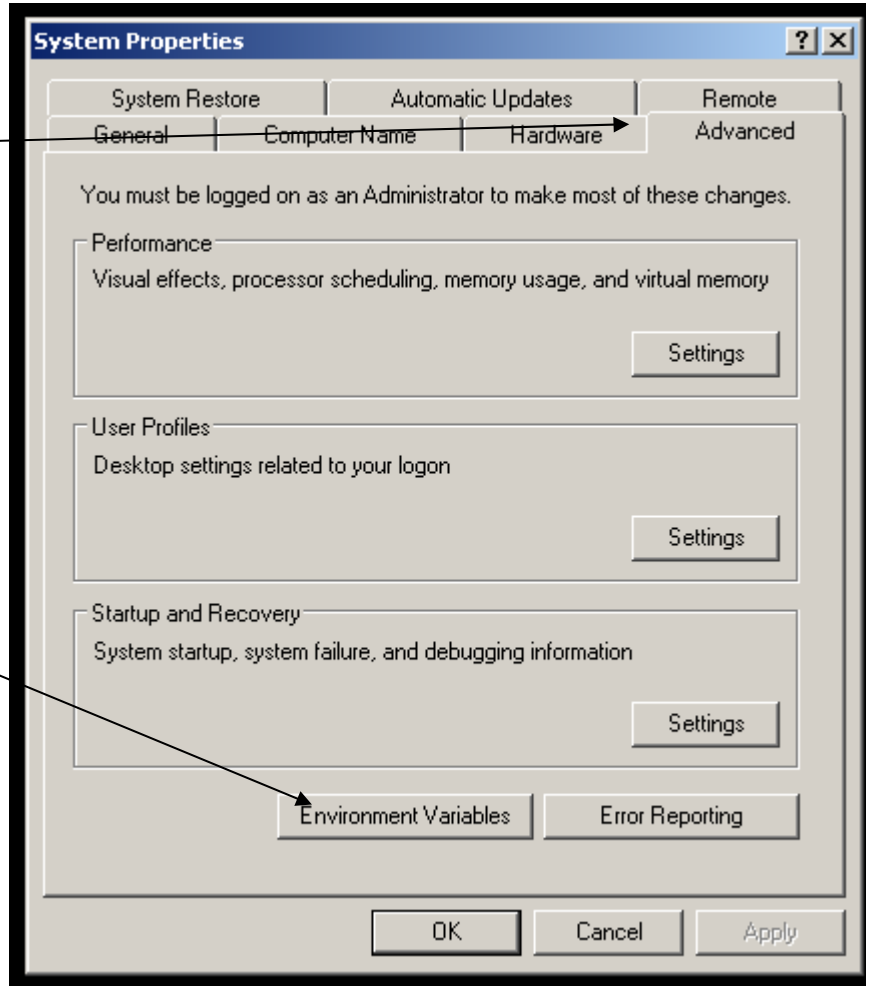
Instalação JDK

1. Clicar com botão direito no “Meu Computador”, em seguida clique em propriedades:



Instalação JDK

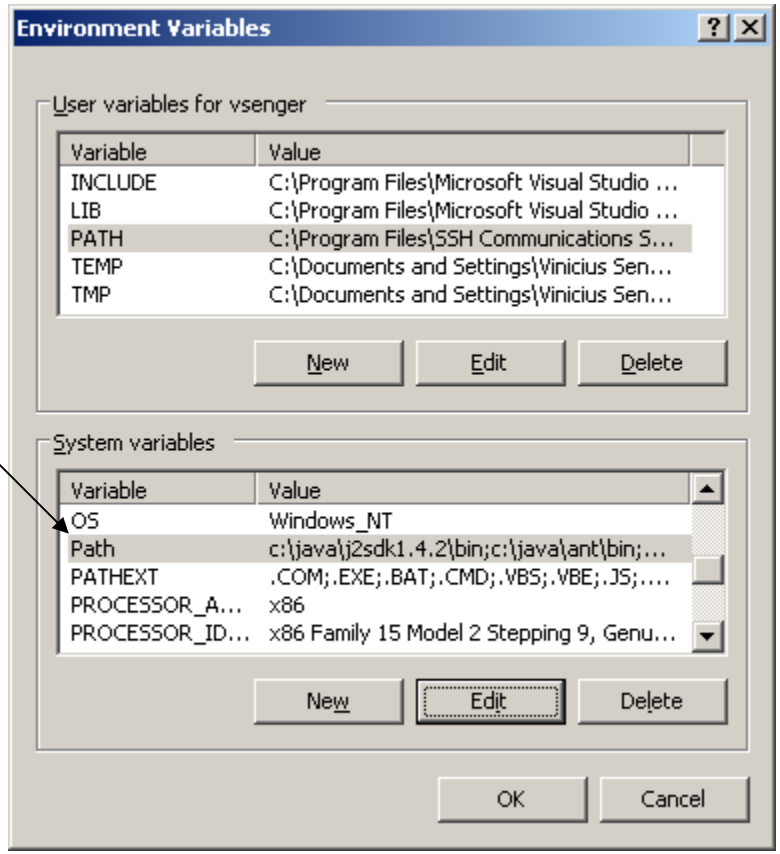
2. Clique em “Avançado” / “Advanced”
3. Clique em “Variáveis de Ambiente” / “Environment Variables”

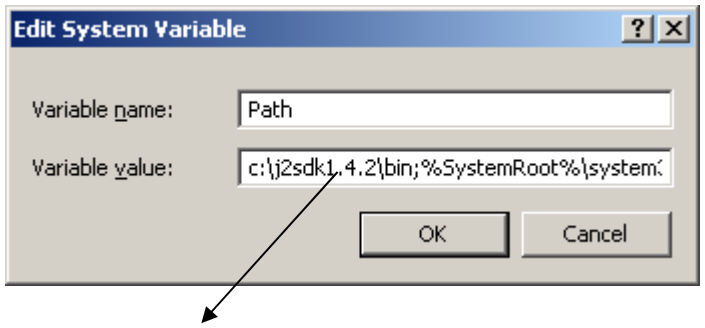


Instalação JDK

4. Clique em “PATH”, em seguida “Edit” / “Editar”

5. Preencha os campos com os seguintes valores:



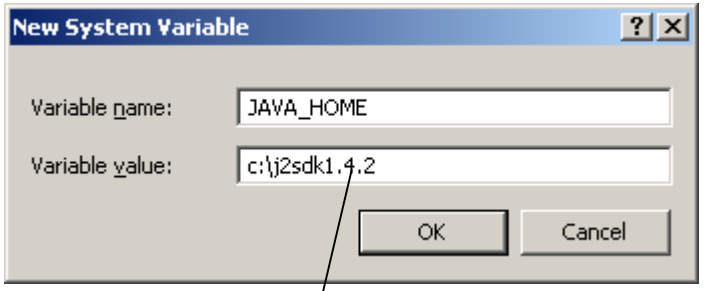


Se você instalou em um diretório diferente do default, indique-o aqui!

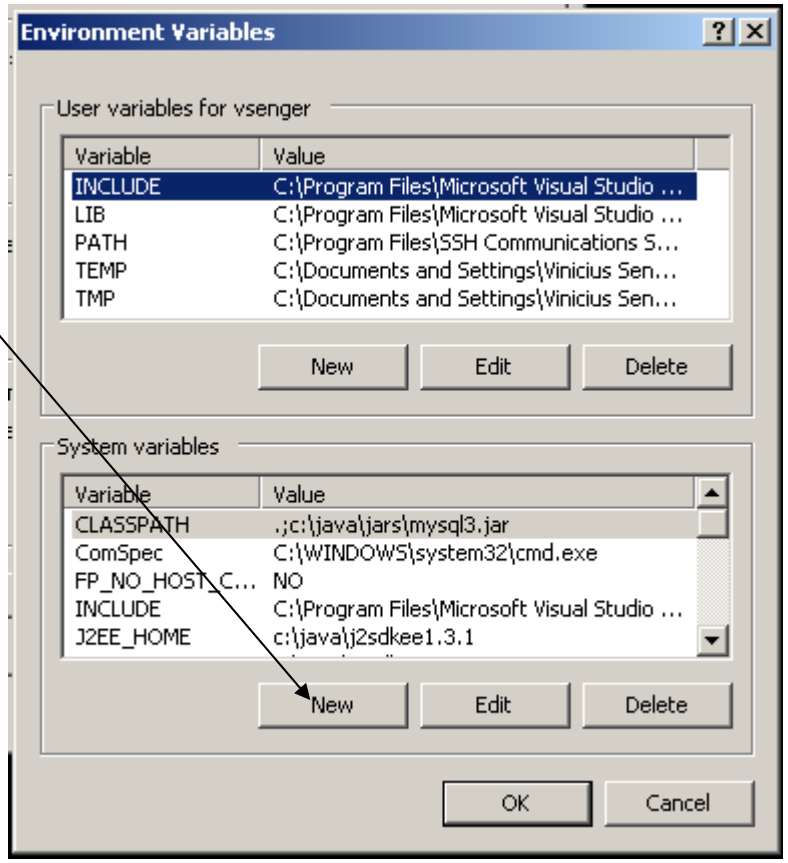
Instalação JDK

6. Clique em “New” / “Novo”

7. Preencha os campos com os seguintes valores:



Atenção: não colocar \bin



Instalação Web Server

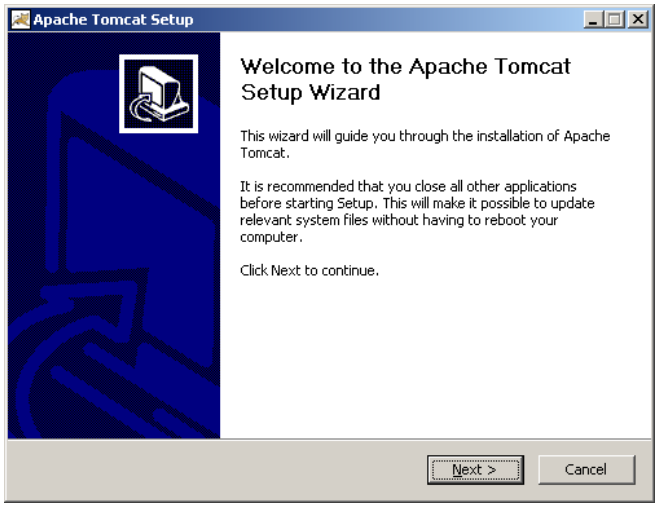
- Temos diversos servidores Web para Java:
  - Apache Tomcat (gratuito / referência)
  - WebSphere
  - JRun
  - WebLogic
  - Oracle OC4J
  - Jetty
  - Sybase Application Server
  - E mais n opções...
- Vamos utilizar Tomcat, tudo que funciona no Tomcat funciona em qualquer servidor Web Java;

Instalação Web Server

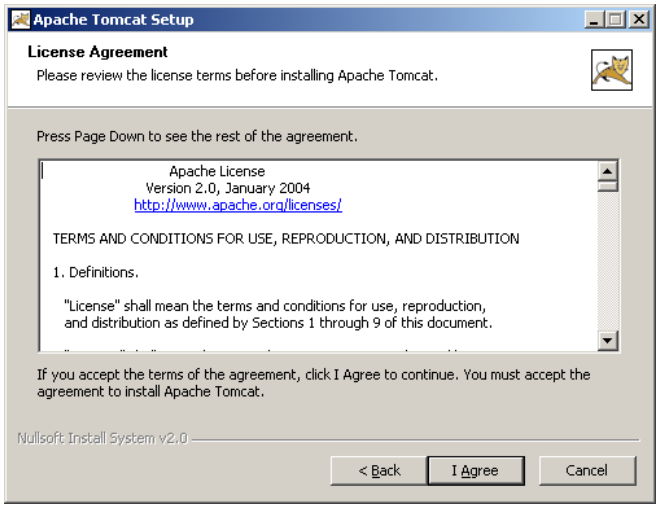
- Download do Tomcat pode ser feito através da URL http://jakarta.apache.org/site/binindex.cgi
- Recomendamos o uso da versão 5.x, evitar beta;
- Para Windows, tem uma versão EXE com installshield;
- Outras plataformas, é um ZIP;
- Não esquecer do pré-requisito de instalar o JDK Standard Edition...

Instalação Web Server

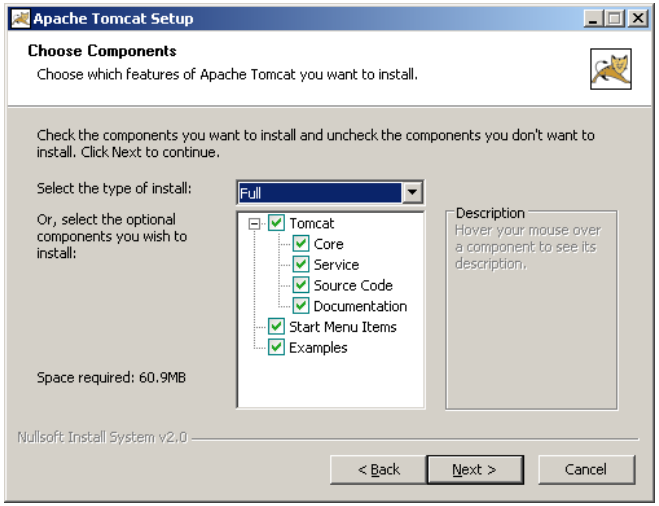
1. Ao disparar a instalação ...



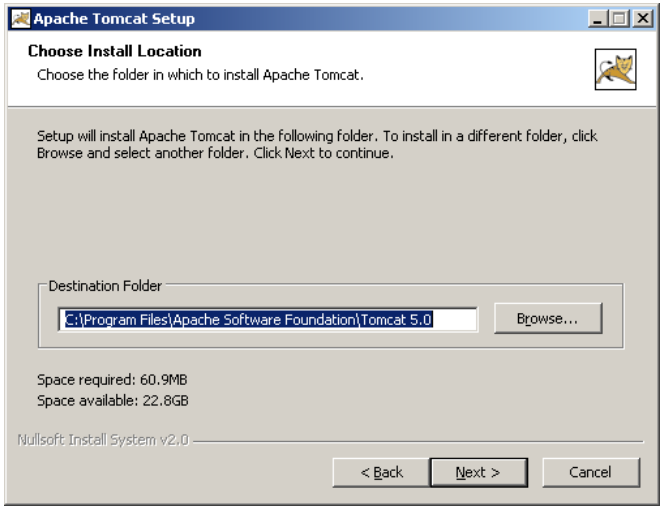
2. License Agreement



3. Tipo da instalação

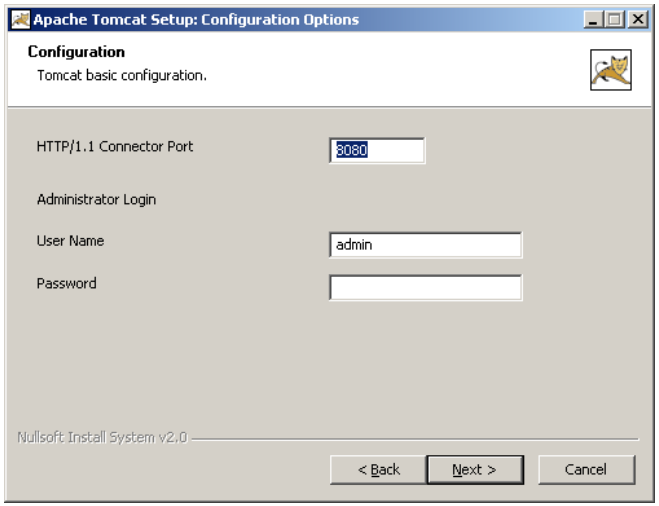


4. Local da instalação

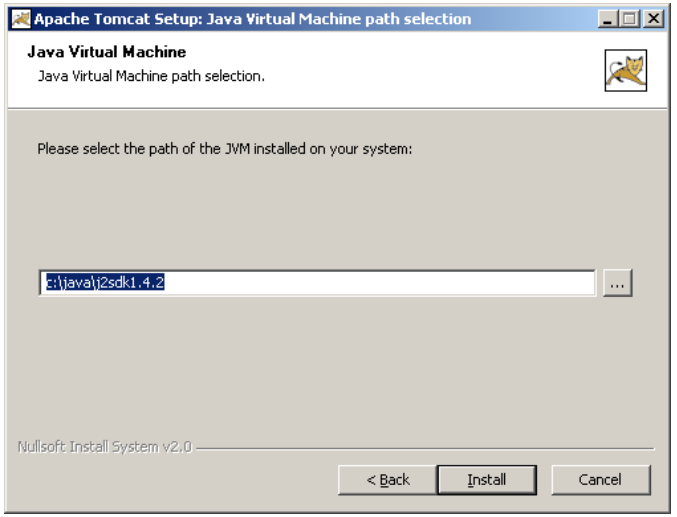


Instalação Web Server

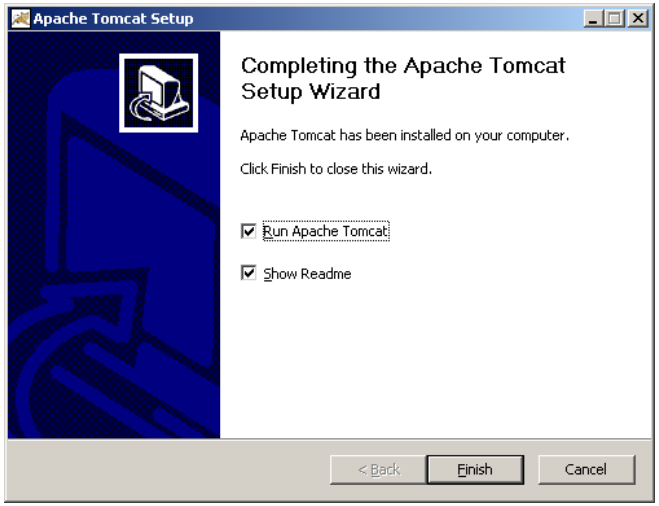
5. Senha de administrador



6. Local do JDK

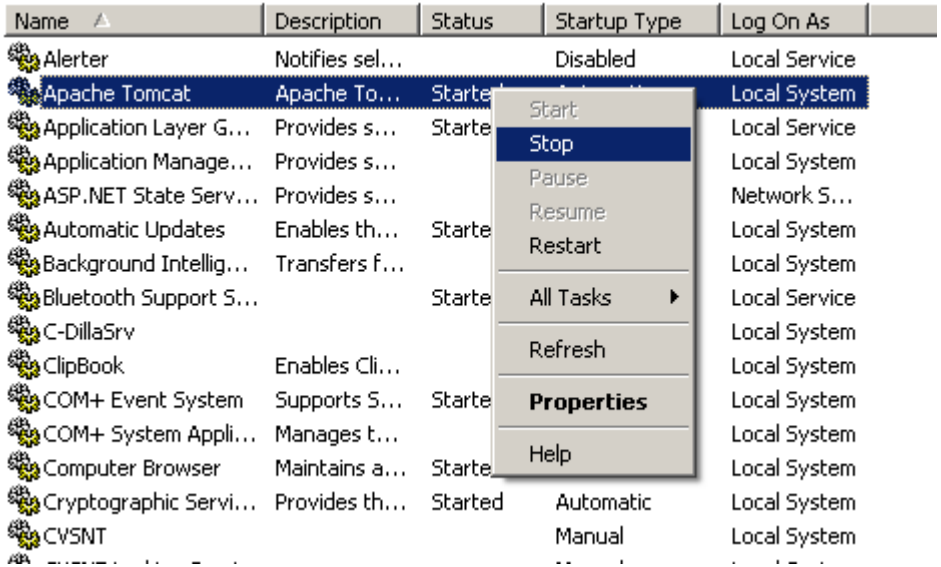


7. Instalação concluida



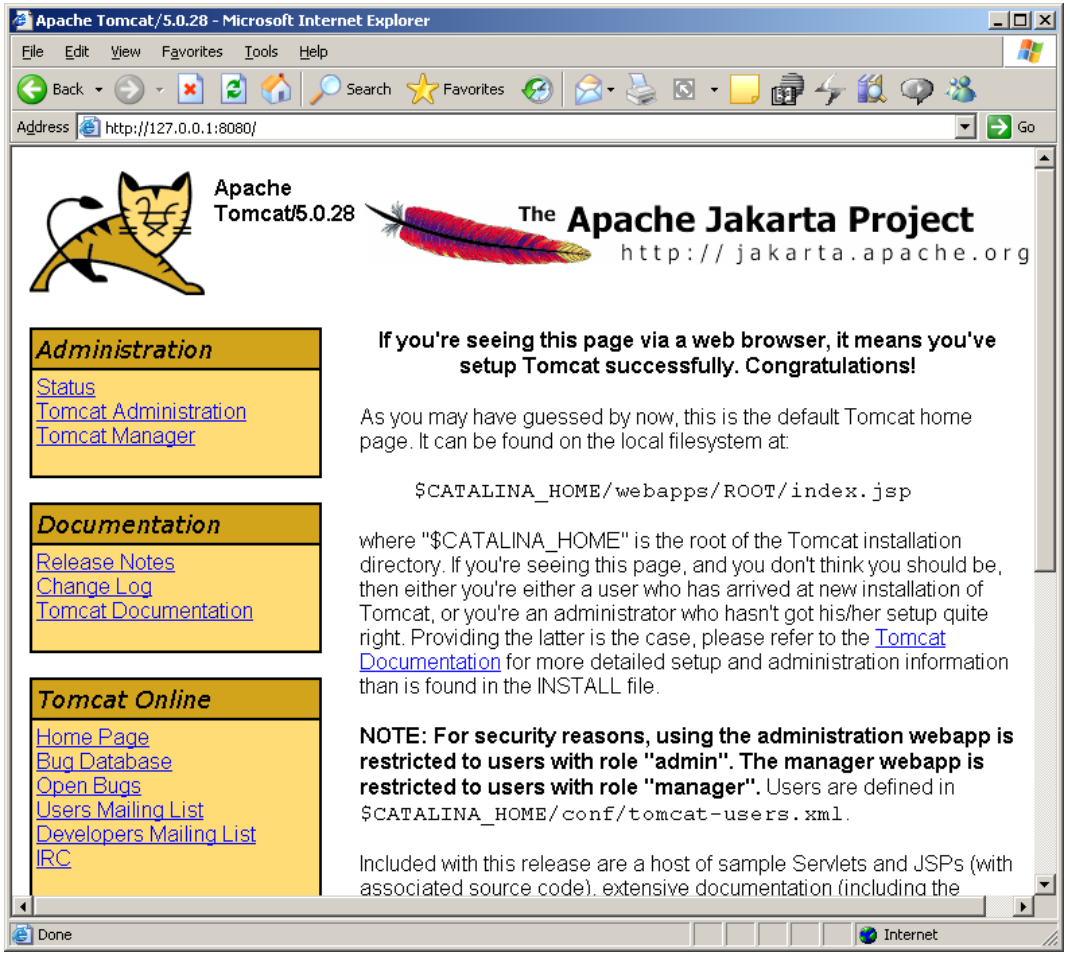
Instalação Web Server

- Para iniciar o Tomcat utilize os serviços do Windows:



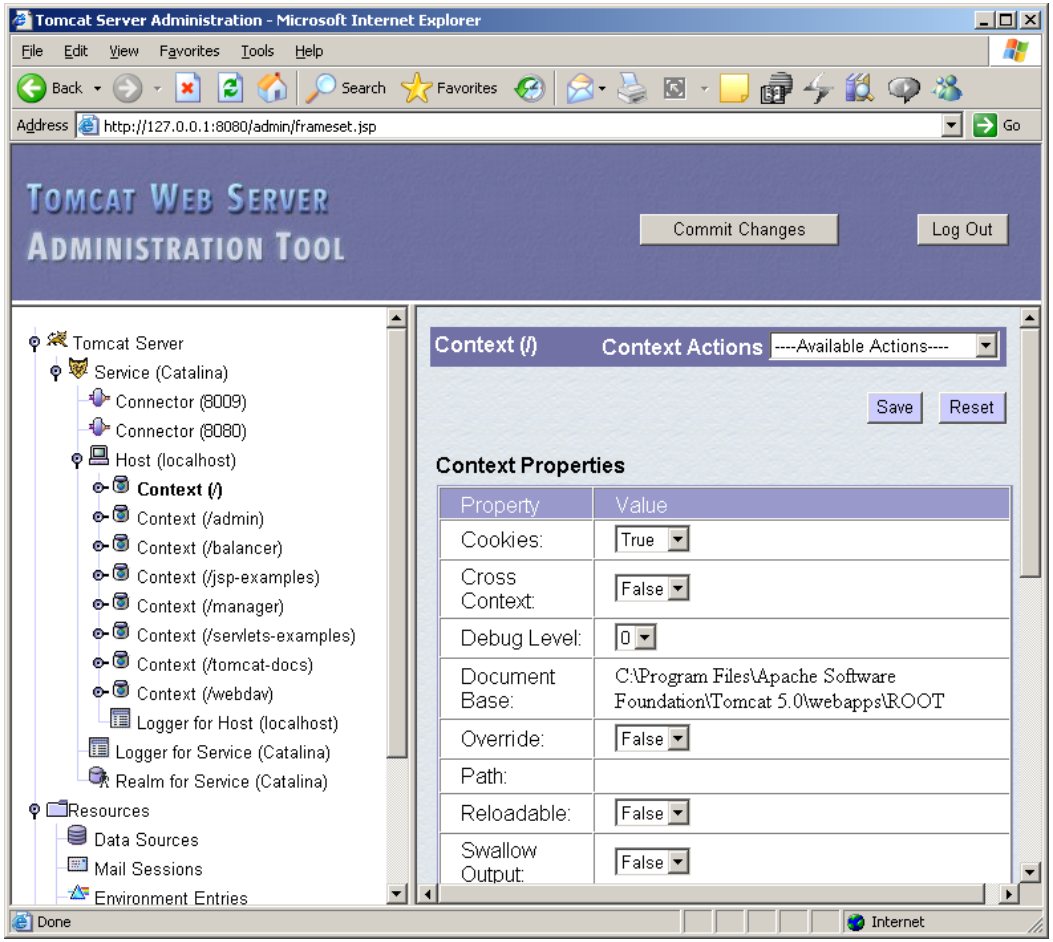
Instalação Web Server

- Por padrão o Tomcat utiliza a porta 8080, portanto para testarmos se está funcionando, acesse a URL: http://localhost:8080



Instalação Web Server

- Na URL: *http://localhost:8080/a* você pode administrar e configurar características do Tomcat



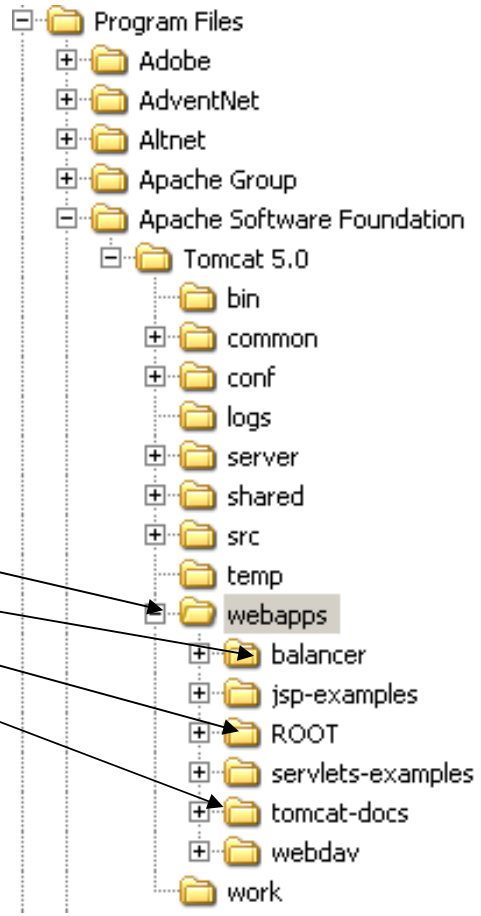
Agenda – Parte Prática

1. Instalação de um servidor Web Java;
2. Criando um aplicativo Web;
3. Java Server Pages;
4. Java Servlet;
5. Demo;

Criando um Aplicativo Web

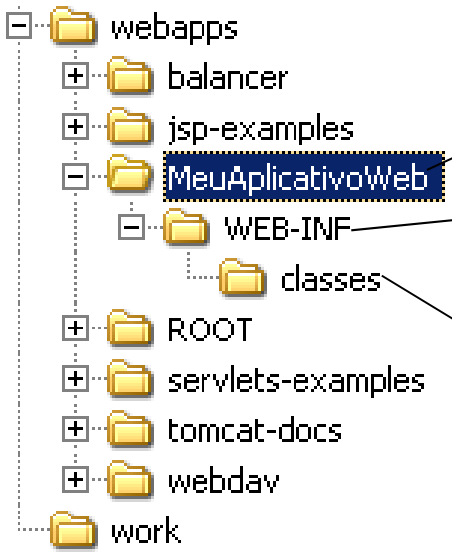
- As aplicações são instaladas (por padrão) no diretório webapps do Tomcat

- Cada sub-diretório representa um aplicativo Web



Criando um Aplicativo Web

- Para criar um novo aplicativo, criar a seguinte estrutura de diretórios:



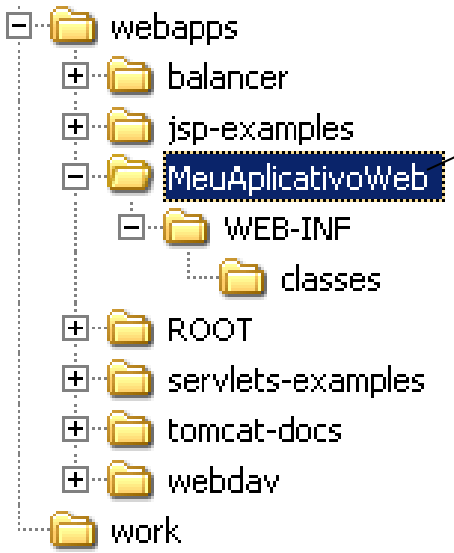
Diretório do aplicativo onde colocaremos Web File JSP's, HTML's, GIF's, CSS, etc.

Diretório especial de configuração e classes Java,, conterá um arquivo chamado web.xml. Configura o aplicativo web.

Colocaremos as classes Java de processamento, acesso a banco de dados, servlets, etc.

Criando um Aplicativo Web

- Vamos colocar um arquivo index.html que servirá como página de entrada:

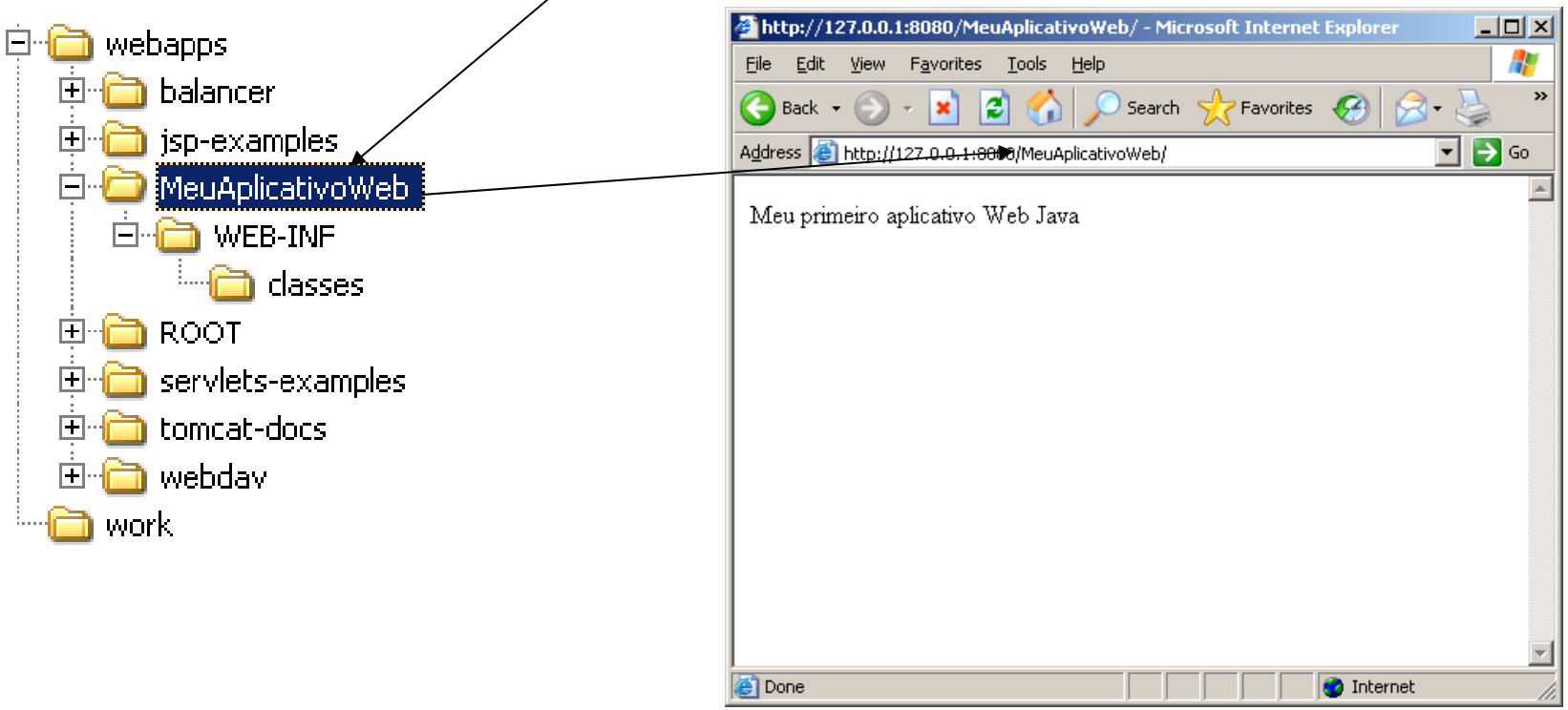


index.html

```
<html>
<body>
<p>Meu primeiro aplicativo
Web Java </p>
</body>
</html>
```

Criando um Aplicativo Web

- Agora podemos acessar nosso aplicativo através da URL: http://localhost:8080/MeuAplicativoWeb



Agenda – Parte Prática

1. Instalação de um servidor Web Java;
2. Criando um aplicativo Web;
3. Java Server Pages;
4. Java Servlet;
5. Demo;

Java Server Pages

- É uma tecnologia Java para geração de conteúdo Web dinâmico (estático + banco de dados, por exemplo);
- Permite mesclar código HTML com código Java:

```
<html>
  <head>
    <title>Olá Mundo</title>
  </head>

  <body>
       Código HTML puro <br>
       <% for(int x=0;x<100;x++) {        %>
         <p>Olá Mundo com laço: <%= x %></p>
       <%}%>
  </body>
</html>
```

Java Server Pages

- Parecido com ASP, porém no lugar de Basic utilizamos Java;
- Robusto, flexível e componentizável;
- Permite a criação de aplicativos simples, inteiramente escritos dentro dos JSP's (model one);
- Permite a criação de aplicativos profissionais, utilizando técnicas avançadas de modelagem como M.V.C.;
- Rápido;
- É transformado em Servlet antes da sua execução;

Agenda – Parte Prática

1. Instalação de um servidor Web Java;
2. Criando um aplicativo Web;
3. Java Server Pages;
4. Java Servlet;
5. Demo;

Java Servlet

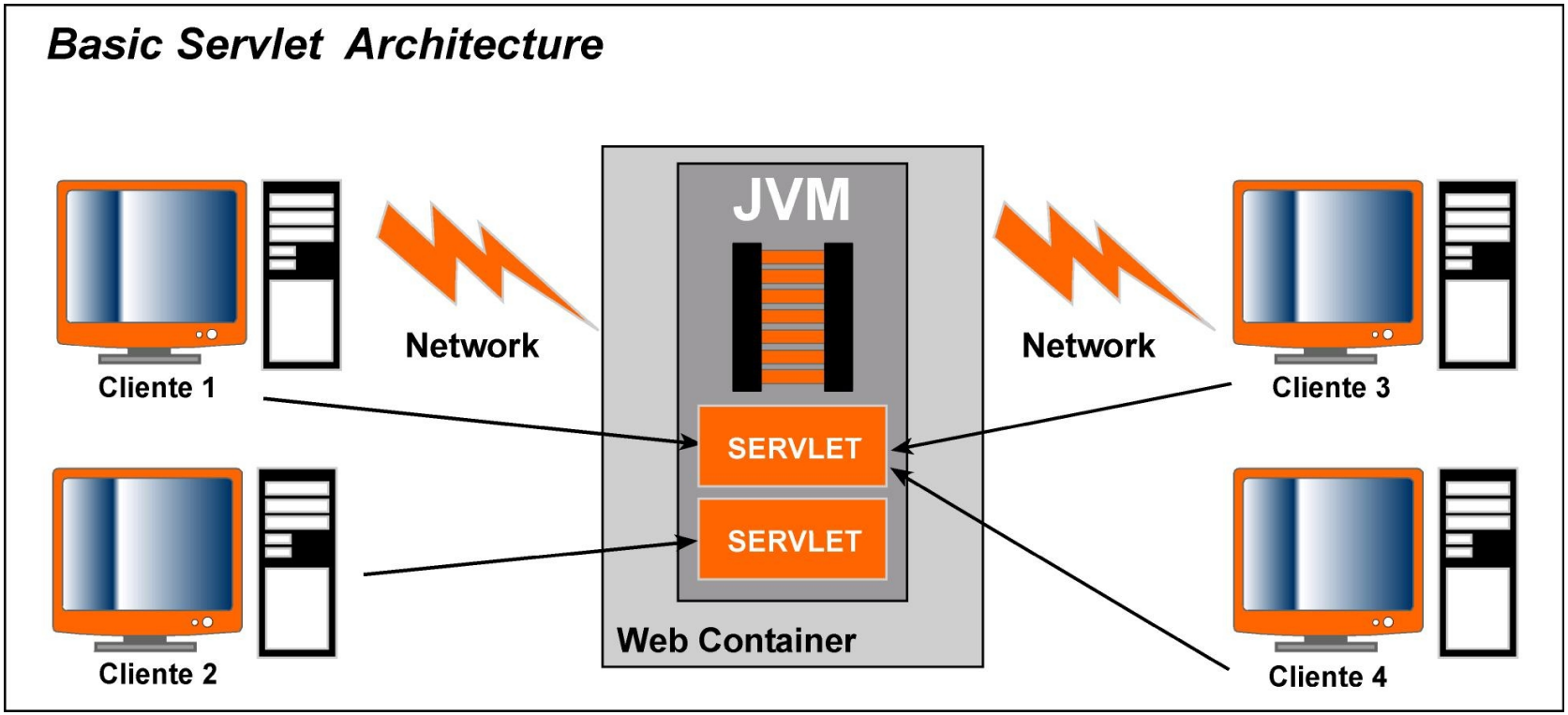
- Tecnologia que “mãe” do JSP;
- Todo JSP é transformado em um Servlet;
- É uma classe Java que estende HttpServlet;
- Deve ser declarado e configurado no web.xml;
- Servlet Vs. JSP:
  - Servet = mais utilizado para recepção e controle de requisições Web;
  - JSP = mais utilizado para geração de conteúdo;
- Exemplo:

Java Servlet

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

public class OlaMundo extends HttpServlet {
    public void doGet(HttpServletRequest request,
HttpServletResponse   response) throws IOException,
ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<body>");
        out.println("<head>");
        out.println("<title>Hello World!</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Hello World!</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

Java Servlet



Agenda – Parte Prática

1. Instalação de um servidor Web Java;
2. Criando um aplicativo Web;
3. Java Server Pages;
4. Java Servlet;
5. Demo;

DEMO

Globalcode®
The Developers Company

Open-source Education

Programação Web com Java

Iniciativa Globalcode