

Use a Cabeça!

FREEMAN, Eric e Elisabeth. HTML com CSS e XHTML

BASHMAN, Brian / SIERRA Kathy / BATES, Bert. Servlets & JSP

# Introdução à aplicação Web

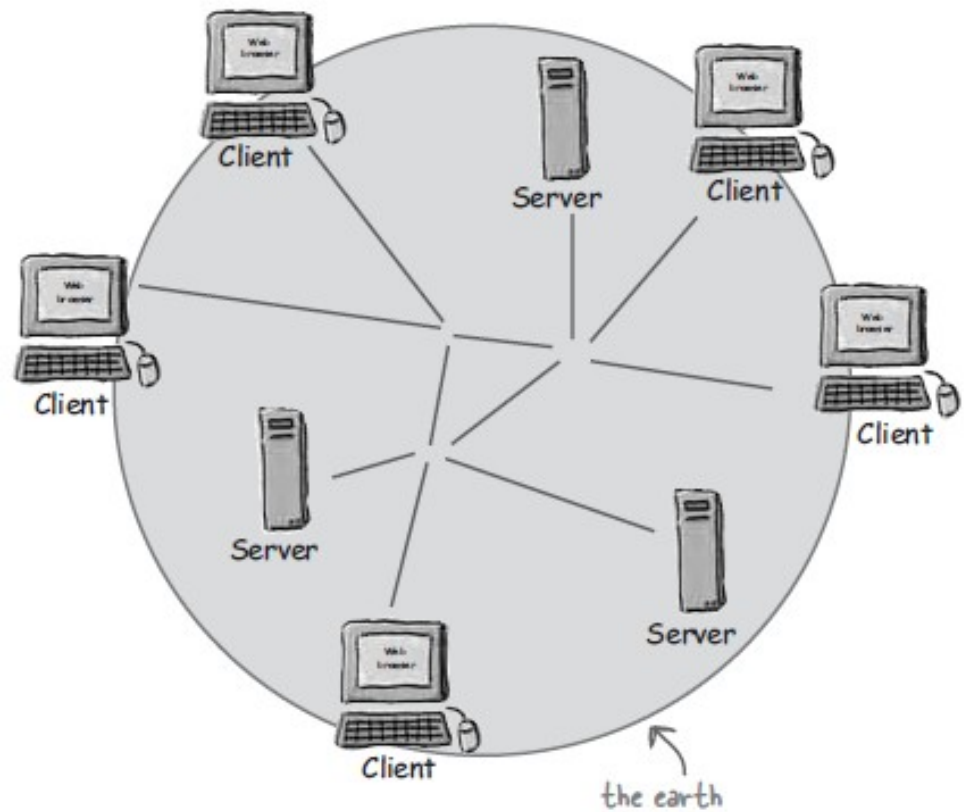
---

# Componentes típicos

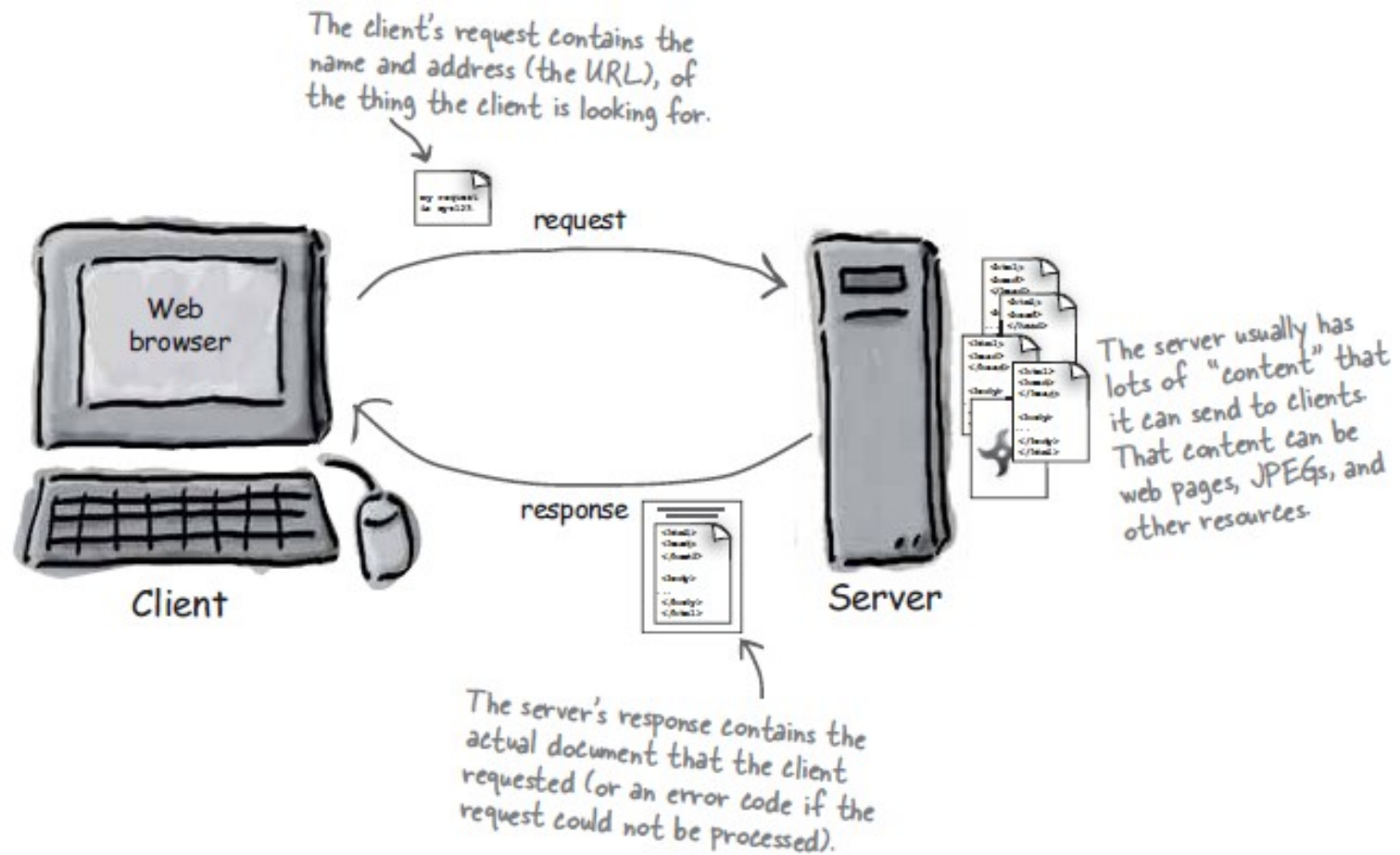
- Software cliente: browser e outros
- Protocolo HTTP
- Infraestrutura de transporte TCP/IP
- Servidor Web
- Aplicações auxiliares
- Mecanismo de persistência, normalmente implementado por meio de uma camada própria (ver exemplo)

# A Web

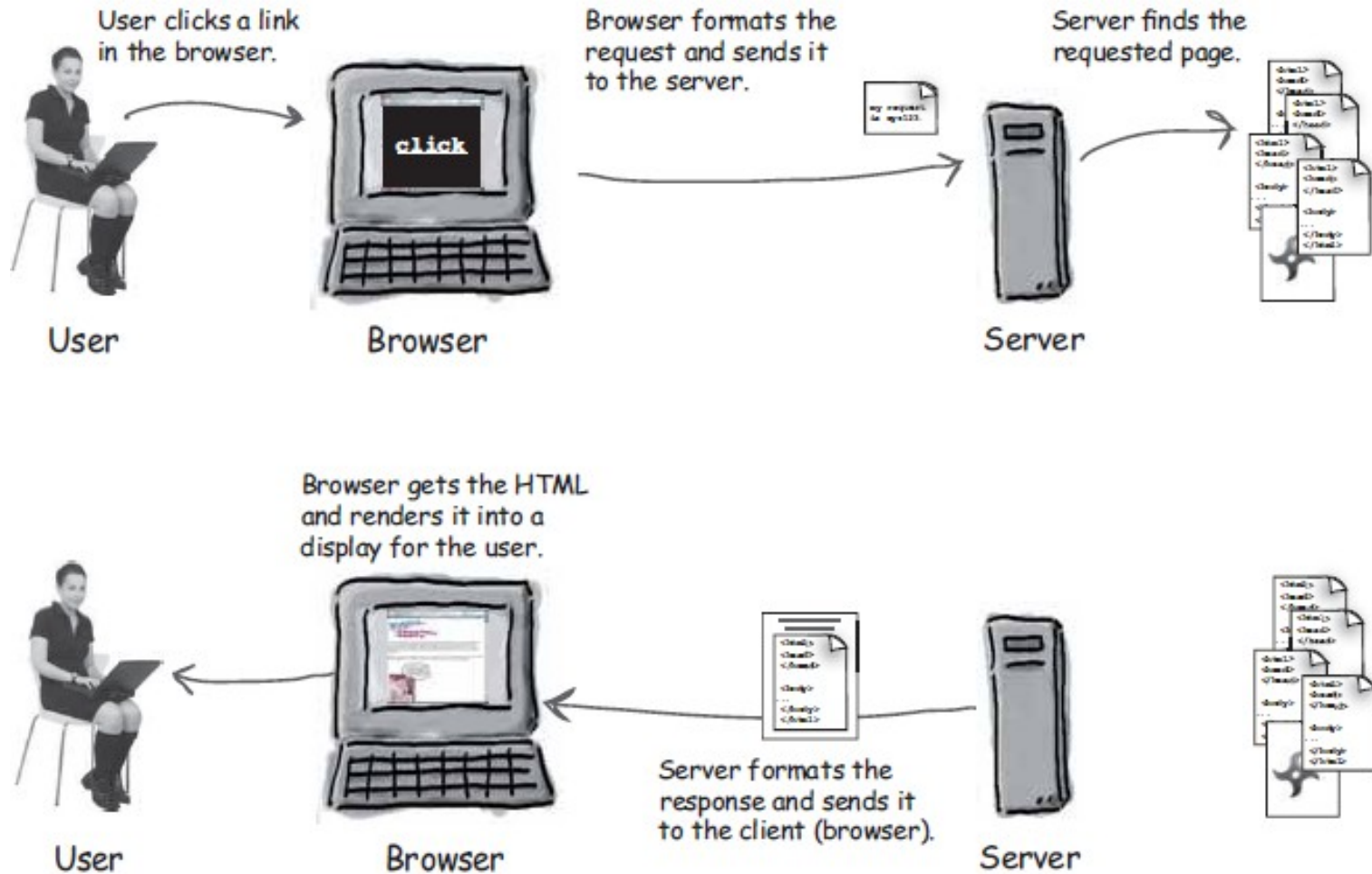
Vários clientes  
Web tomando  
serviços de  
diversos  
servidores Web



# Cliente e servidor Web



# Cliente e servidor Web



# Tecnologias típicas do cliente

- Marcação: XML, XHTML, HTML
- Folhas de Estilos: CSS
- Script: Javascript, VBScript
- Applet Java
- Flash
- AJAX – Assicronos Javascript and XML

# Tecnologias típicas do servidor\*

- JSP e Servlets
- PHP
- Cold Fusion
- Microsoft Dot Net
- Common Gateway Interface – CGI

\* Aplicações auxiliares acionadas pelo servidor Web

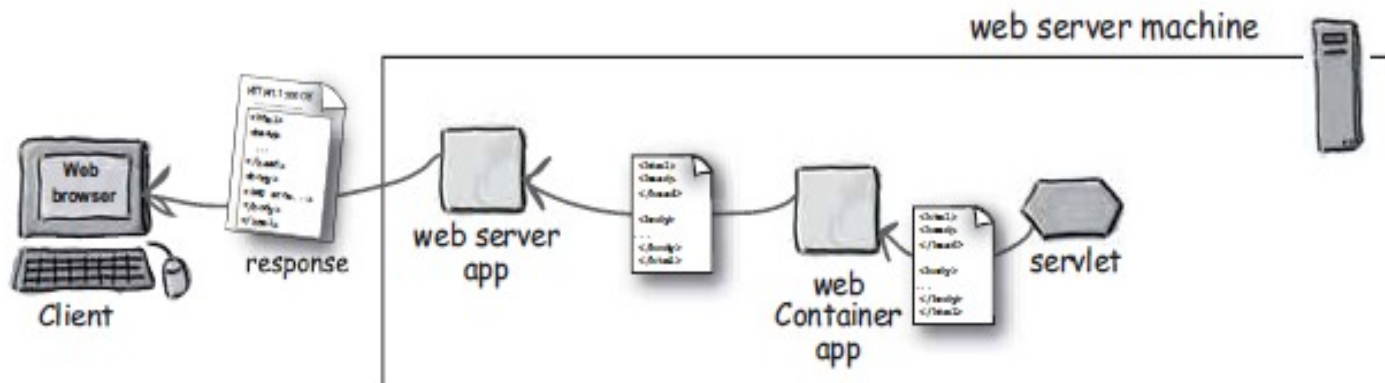
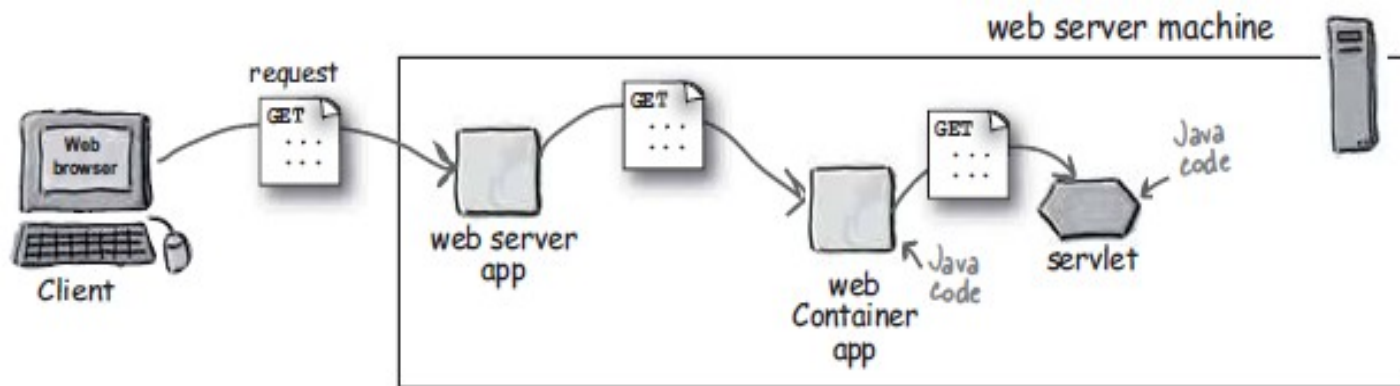
# Funcionamento Básico

1. Cliente realiza solicitação
2. Servidor Web recebe a solicitação
3. Solicitações para páginas estáticas são servidas diretamente pelo servidor Web
4. Solicitações para conteúdo a ser gerado (dinâmico), como JSP e Servlets, são encaminhadas a aplicações auxiliares
5. Aplicações auxiliares devolvem ao servidor Web o conteúdo gerado

# Funcionamento Básico

1. O servidor Web gera uma resposta ao cliente
2. O cliente recebe a resposta, interpreta o conteúdo e renderiza a página
3. A página apresentada normalmente contém elementos de hipertexto e hipermedia por meio dos quais o usuário pode solicitar ao navegador que gere uma nova solicitação
4. O processo se repete

# Funcionamiento Básico



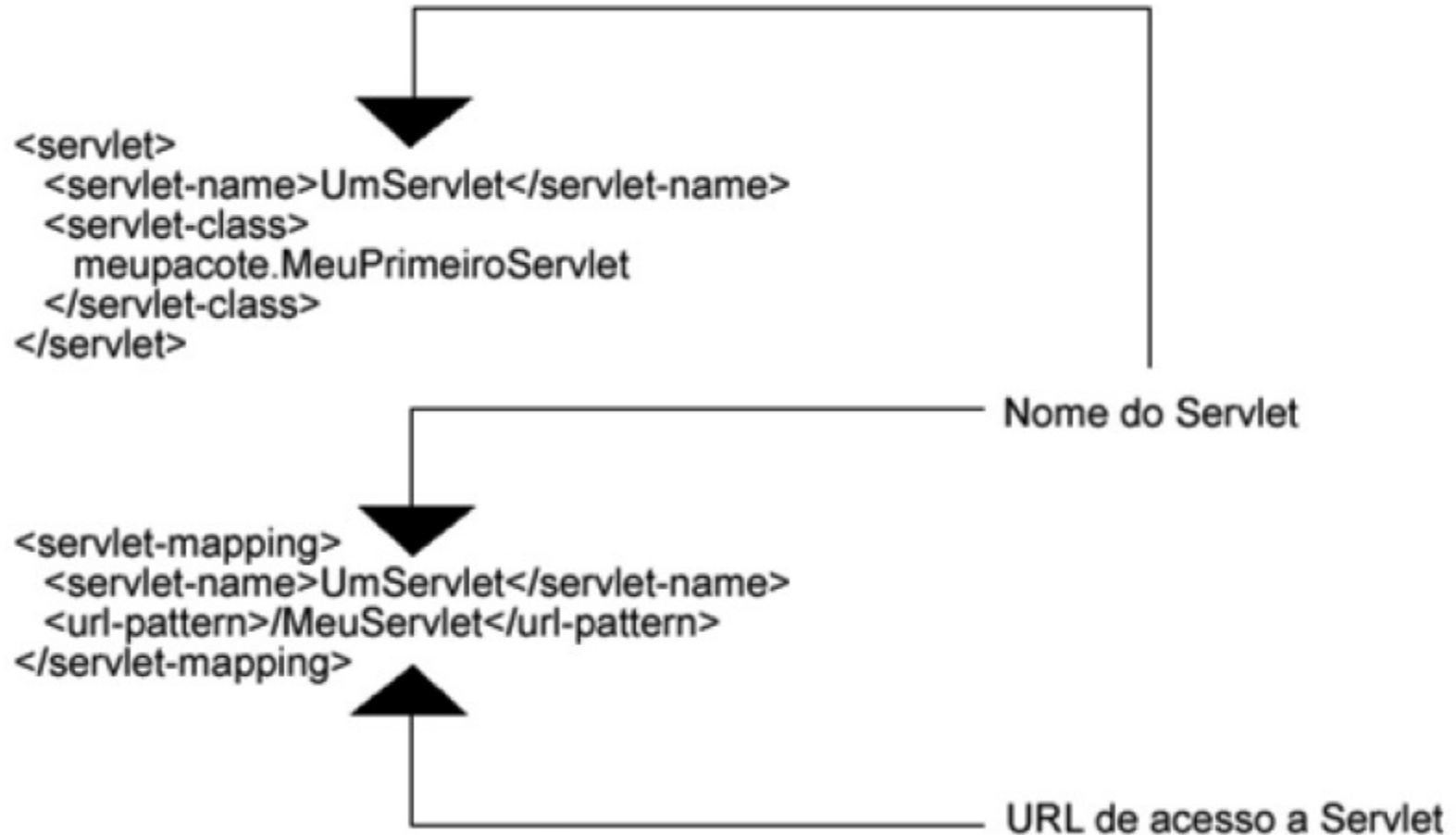
# Servlets

- Na arquitetura **MVC**, destinam-se principalmente ao desenvolvimento da camada ***Control***
- São classes **gerenciadas** por um **contêiner Servlet/JSP** para atender a solicitações da aplicação Web (ciclo de vida)
- Possuem um ou mais **mapeamentos**, definidos no **descritor de distribuição web.xml**, onde devem estar relacionadas

# Servlets: ciclo de vida

```
public class DemoCicloDeVidaServlet extends HttpServlet {  
    public DemoCicloDeVidaServlet() { ... }  
    public void init() throws ServletException { ... }  
    public void destroy() { ... }  
    protected void doGet(  
        HttpServletRequest request,  
        HttpServletResponse response)  
        throws ServletException { ... }  
    protected void doPost(  
        HttpServletRequest request,  
        HttpServletResponse response)  
        throws ServletException { ... }  
}
```

# web.xml: registro e mapeamento dos servlets



# Contêiner JSP/Servlet

- Implementação da **especificação** JSP/Servlet (Tomcat, Glassfish, Jboss)
- Gerencia as **requisições** e os **servlets**, encaminhando a estes as requisições

# Servlets: funções básicas

- Recebimentos de **parâmetros** do cliente
- Disponibilização de atributos na **página**, na **requisição**, na **sessão** e na **aplicação** (escopos)
- **Encaminhamento** da requisição para outro recurso
- **Instanciação** de outras classes para tomada de serviços

# Prática de Laboratório 1

1. Desenvolva um formulário XHTML que envie dados a um servlet. Este deve exibir os dados enviados pelo formulário.
2. Desenvolva um formulário XHTML que envie dados a um servlet. Este deve retirar os parâmetros da requisição enviados pelo formulário, definir esses parâmetros como atributos da requisição e encaminhar a requisição para que outro servlet acesse os atributos e os exiba em uma página XHTML.
3. Desenvolva uma aplicação Web composta por:
  - Um formulário que defina alguns campos, inclusive um campo oculto (tipo hidden), e envie os dados a um servlet.
  - Um servlet controlador que receba os dados e encaminhe para um dos servlets de exibição, dependendo do valor do campo oculto do formulário.
  - Dois servlets de exibição que serão acionados pelo controlador, dependendo do valor do campo oculto do formulário.

Exemplo: <http://www.vqv.com.br/java/aulaLPVServlet.zip>

# Prática de Laboratório 2

1. Usando a apostila fornecida como apoio, desenvolva um formulário XHTML que envie dados a um servlet controlador, conforme layout e observações a seguir.
2. O servlet controlador deve retirar os parâmetros da requisição, criar um objeto de uma classe auxiliar ao formulário (esta deve conter uma variável de instância para campo do formulário, devidamente encapsuladas), configurar as variáveis de instância com os valores dos parâmetros recebidos do formulário XHTML, colocar o objeto criado como atributo da requisição e encaminhar esse objeto a um servlet exibidor de dados.
3. O servlet exibidor de dados deve ler o objeto que foi posto como atributo da requisição e montar uma página XHTML para exibir, de forma amigável ao usuário, todos os dados postados pelo formulário.
4. Acrescente no formulário um novo botão submit gravar e, no servlet controlador, analise qual dos dois botões foi pressionado na submissão, encaminhado em cada caso a requisição para um servlet diferente.

# Prática de Laboratório 2

Nome do candidato:

Endereço:

CEP:

-- Cidade --



-- UF --



Telefones:

RG:

CPF:

Sexo:  Masc.

Fem.

Escolaridade:

-- Selecione --



email:

Curso:

-- Selecione --



Senha de acompanhamento:

Incluir

# Prática de Laboratório 2

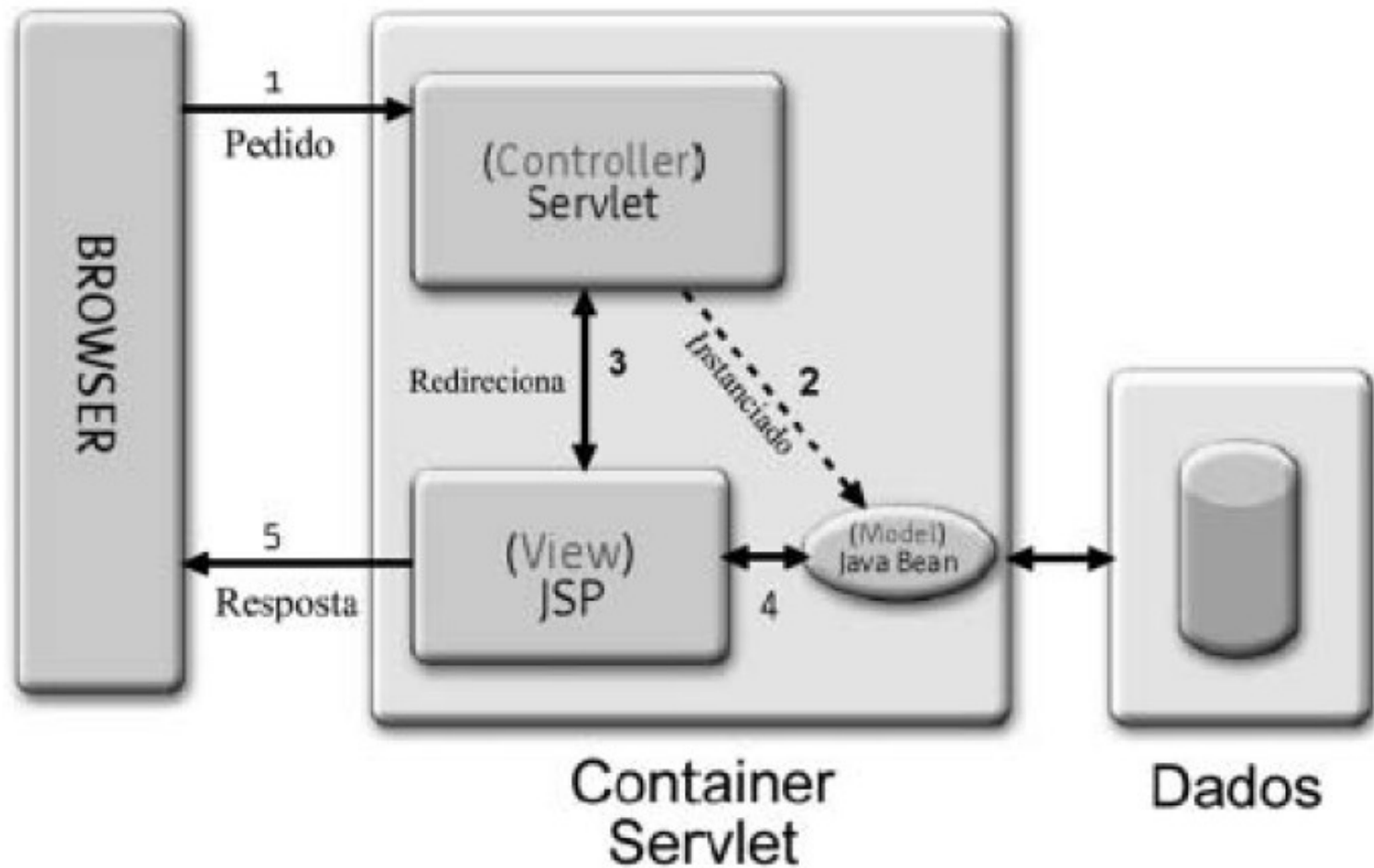
Observações:

<p>Cidades:</p> <ul style="list-style-type: none"><li>- Brasília</li><li>- Taguatinga</li><li>- Sobradinho</li><li>- Águas Claras</li></ul>	<p>Escolaridade:</p> <ul style="list-style-type: none"><li>- Ensino Fundamental</li><li>- Ensino Médio</li><li>- Superior Incompleto</li><li>- Superior Completo</li><li>- Pós-Graduação</li></ul>
<p>UF:</p> <ul style="list-style-type: none"><li>- DF</li><li>- GO</li></ul>	<p>Curso:</p> <ul style="list-style-type: none"><li>- Web Design</li><li>- Lógica de Programação</li><li>- Java</li><li>- Delphi</li></ul>

# JSP - Java Server Pages

- Na arquitetura **MVC**, destina-se principalmente ao desenvolvimento da camada ***View***
- Seus componentes são normalmente **inseridos** em um documento do tipo **HTML**
- Trata-se de uma interface de nível mais alto para a **geração de servlets**

# MVC



# Componentes típicos de JSP

- `<%-- --%>` Comentário JSP
- `<%@ %>` Diretiva
- `<%= %>` Tag de expressão
- `<% %>` Scriptlet
- `<%! %>` Declarações
- `${ }` Expression Language – EL
- `<c:forEach>` JSTL

Analisar os componentes nos códigos de exemplo!

# Trabalho de Pesquisa (parte I)

- Discorra sobre o uso da linguagem Java para desenvolvimento de aplicações Web, adotando arquitetura MVC. Fale sobre como construir uma aplicação Web, separando seus componentes em camadas (modelo, visão e controle) e sobre as formas de uso mais adequadas de JSP, Servlets e classes Java convencionais, POJO. Use a Web, a biblioteca e outras fontes, citando-as no trabalho.

# Trabalho de Pesquisa (parte II)

Pesquise e descreva cada um dos recursos JSP a seguir, apresentando exemplos:

- `out.print` e `out.println`
- `request.getParameter()`
- `request.setAttribute()` e `request.getAttribute()`
- `response.sendRedirect()`
- Tags da JSTL Core: `forEach`, `set` e `if`
- Expression Language – EL
- Scriptlets `<%...%>`, expressões `<%=...%>` e declarações `<%!...%>`

# Dicas ao programador iniciante

- Organizar desde o início do desenvolvimento os componentes da aplicação
- Testar cada alteração, certificando-se da correção de seu resultado
- Quando parecer que as alterações não surtiram efeito algum, reinicie o servidor e redistribua (re-deploy) a aplicação e seus componentes externos, como projeto de negócio, persistência, etc.

# Aplicação de exemplo

- Cadastro de Empregados
- CRUD (Create, Read, Update, Delete)
- Arquitetura MVC (Model, View, Control)
- Primeira versão com JSP
- Versão final com Servlets
- Ver diagrama de sequência para edição

<<control>>  
controleEmpregado.jsp

<<DAO>>  
EmpregaoDAO

<<view>>  
listaEmpregado.jsp

<<control>>  
controleEdita.jsp

<<view>>  
editaEmpregado.jsp

<<control>>  
processaEdita.jsp

<<view helper>>  
FormEmpregado.java

