UNIP
UNIVERSIDADE PAULISTA

Aplicações de Linguagem de Programação Orientada a Objetos

MÓDULO 10

Professor: Me. Pablo I. Gandulfo
Data: 01/08/2020
Versão: 1.1

Professor: Me. Pablo I. Gandulfo
Data: 01/08/2020
Versão: 1.1

Introdução a aplicação Web

- Introdução conceitual do JavaEE e de desenvolvimento WEB via JSP e Servlet
  - Conceitos de Programação WEB
  - Servidores (Containers) e Clientes
  - Tecnologias para Desenvolvimento WEB
  - Servlets
- Primeiros Servlets e JSP's
- Exercícios

Conceitos - Método HTTP GET

- Um Navegador emite uma requisição HTTP GET quando:
  - O usuário entra com uma URL no campo Localização ou Endereço do Navegador
  - O usuário clica num *link* na página HTML atual
  - O usuário submete um formulário com o método GET de envio

Estrutura de uma Requisição HTTP

Método / URL / Versão

Linha da Requisição: GET /intermediariojee/OlaMundoServlet HTTP/1.1

Cabeçalhos:

Accept: image/jpeg, application/x-ms-application, image/gif, application/xaml+xml, image/pjpeg, application/x-ms-xbap, */*

Accept-Language: pt-BR,pt;q=0.5

User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; Win64; x64; Trident/7.0)

Accept-Encoding: gzip, deflate

Host: localhost:10001

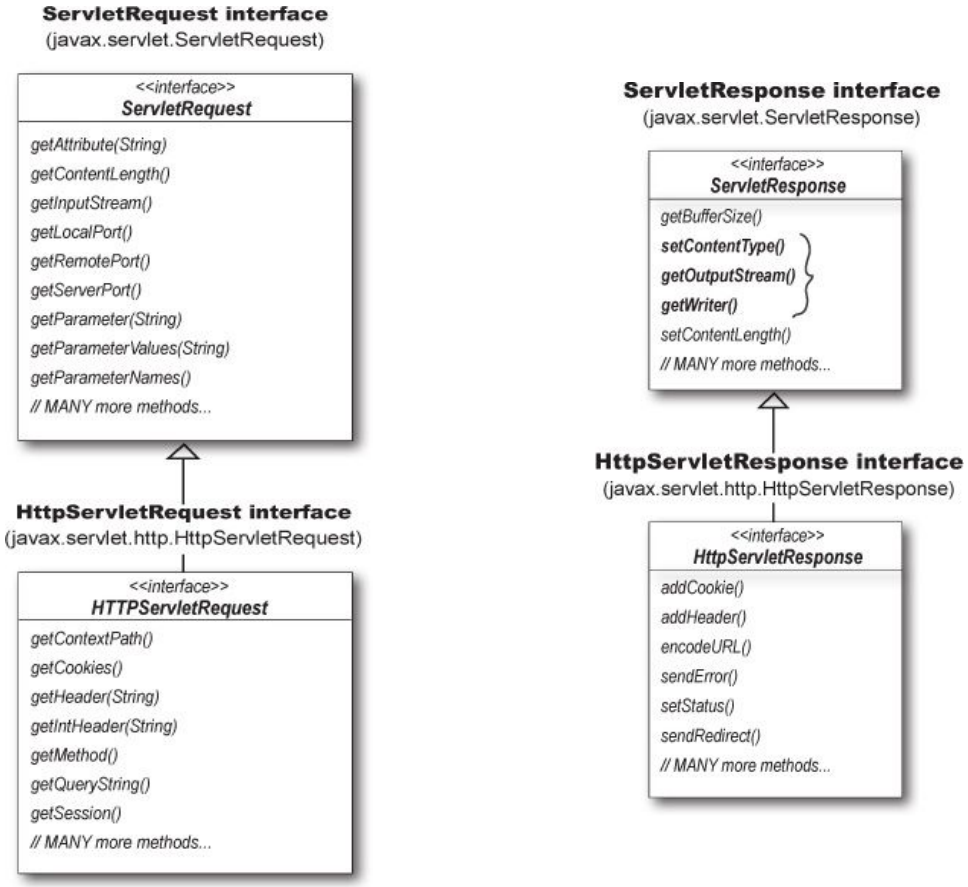
Connection: Keep-Alive

Cookie: JSESSIONID=E4941FBBE271F2C9E5100FCF141A430D; state=SYSEXP%3A; Username=UnknownUser

Estrutura de uma Resposta HTTP

	Versão / Código e Descrição da Mensagem de Resposta
Linha da Resposta:	HTTP/1.1 200 OK
Cabeçalhos:	Server: Apache-Coyote/1.1
	Content-Type: text/html;charset=ISO-8859-1
	Content-Length: 39
	Date: Mon, 20 Mar 2017 14:35:21 GMT
Corpo:	<html><body>Olá, mundo!</body></html>

Classes HttpServletRequest e HttpServletResponse



Exemplos: Estrutura Geral

```
import javax.servlet.ServletException; import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest; import javax.servlet.http.HttpServletResponse;


@WebServlet("/[NomeServlet]")
public class [NomeServlet] extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head><title>Xxx</title></head>");
        out.println("<body>");
        [Código de lógica do servlet]
        out.println("</body>");
        out.println("</html>");
    }
```

Exemplos: Estrutura Geral (cont.)

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        doGet(request, response);
    }
}
```

Exemplo de Servlet que Apresenta Informações do Cabeçalho da Requisição

```
out.println("<h3>Exemplo de Cabeçalho do Pedido</h3>");
out.println("<table border=\"0\">");
Enumeration<String> cabecalhos = request.getHeaderNames();
while (cabecalhos.hasMoreElements()) {
    String nomeCabecalho = cabecalhos.nextElement();
    String valorCabecalho = request.getHeader(nomeCabecalho);

    out.println("<tr><td bgcolor=\"#CCCCCC\">" + nomeCabecalho + "</td>"
        + "<td>" + valorCabecalho + "</td></tr>");
}
out.println("</table>");
```

Exemplo de Servlet que Apresenta Informações da Requisição

```
out.println("<table border=\"0\">");
out.println("<tr><td>Método:</td><td>" + request.getMethod() + "</td></tr>");
out.println("<tr><td>URI da Requisição:</td><td>" + request.getRequestURI() + "</td></tr>");
out.println("<tr><td>Protocolo:</td><td>" + request.getProtocol() + "</td></tr>");
out.println("<tr><td>Endereço Remoto:</td><td>" + request.getRemoteAddr() + "</td></tr>");
out.println("</table>");
```

Exemplo de Servlet que Apresenta Informações de Parâmetros da Requisição

```
out.println("Parâmetros digitados:<br>");

String primeiroNome = request.getParameter("primeiroNome");
String ultimoNome = request.getParameter("ultimoNome");
if ((primeiroNome != null) || (ultimoNome != null)) {
   out.println("primeiroNome = " + primeiroNome + "<br>");
   out.println("ultimoNome = " + ultimoNome + "<br><br>");
}
else {
   out.println("* Sem parâmetros<br>");
}

out.println("<form action=\"RequestParamExample\" method=\"post\">");
out.println("Primeiro Nome: <input name=\"primeiroNome\" type=\"text\" size=20><br>");
out.println("Último Nome: <input name=\"ultimoNome\" type=\"text\" size=20><br><br>");
out.println("<input value=\"Enviar\" type=\"submit\">");
out.println("</form>");
```

Exercício 2

- Tente desenvolver um código similar ao exercício anterior, mas em JSP
  - Código Java pode ser executado numa página JSP através da escrita:

```
<%
    ... // Código Java
%>
```

- Avalie e descreva vantagens e desvantagens do uso de Servlets em comparação com JSPs, utilizando os Exercícios 1 e 2 como referência

Exercício 3 - Opcional

- Utilize o programa TCPProxy.exe (e TCPProxy.bat) para entender a conversação HTTP realizada entre o navegador e o servidor web, em especial, avaliando os seguintes conceitos:
  - Requisições e Respostas
  - Linha de Chamada
  - Cabeçalhos
  - Corpo

- Exemplo de chamada:
  - > TCPProxy.exe 10001 localhost 8080
    - Escuta na porta 10001 e redireciona a comunicação para a porta 8080
    - Gera o log da conversação em TCPProxy.log

Professor: Me. Pablo I. Gandulfo
Data: 01/08/2020
Versão: 1.1

MVC – Model View Controller

- Apresentação da Arquitetura
- Apresentação de Servlet e JSP
- Roteamento de Requisições HTTP
- Desenvolvimento de Aplicações MVC
- Exercícios

Conceitos de Projeto

- Padrões de Projeto (*Design Patterns*)
- Padrões Arquiteturais (Architectural Patterns)
- Arcabouços Conceituais (*Frameworks*)

Conceitos de Projeto: Padrões Arquiteturais (*Architectural Patterns*)

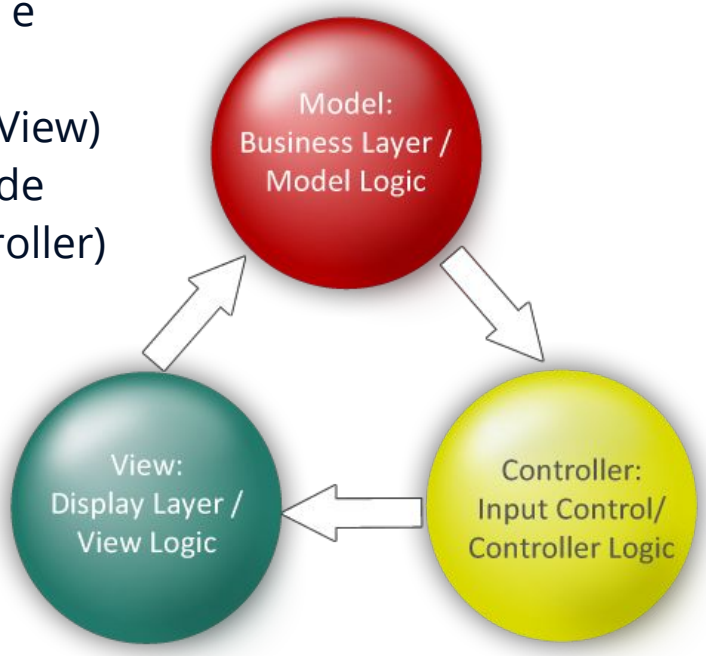
- Padrões Arquiteturais são vistos normalmente como um nível superior aos Padrões de Projeto

- Eles possuem terminologia e conceitos similares aos Padrões de Projeto, mas se concentram em fornecer modelos e métodos reutilizáveis especificamente para a arquitetura global dos sistemas de informação

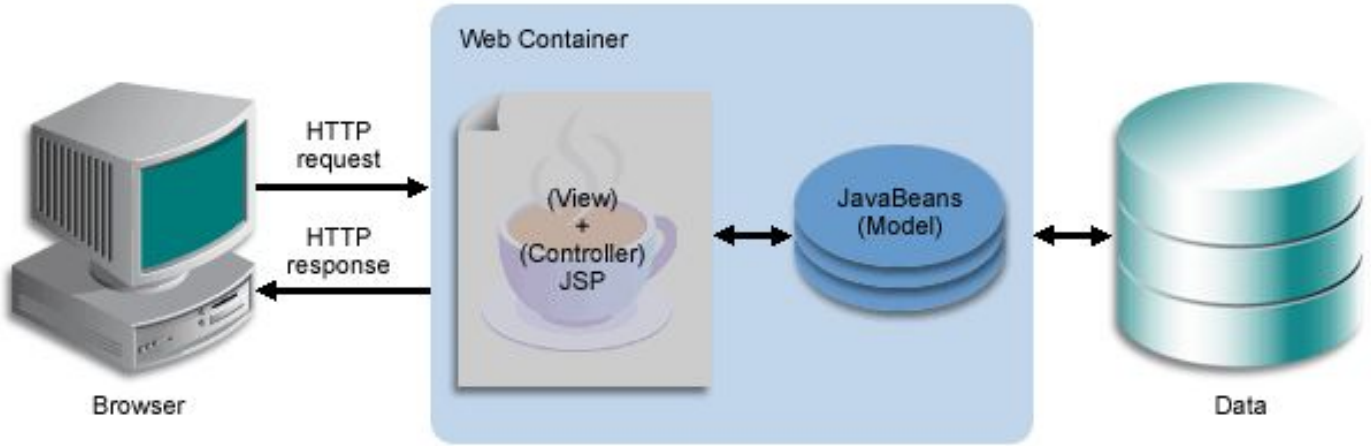
- Padrões Arquiteturais são estratégias de alto nível que dizem respeito a componentes de larga escala, que lidam com propriedades e mecanismos globais de um sistema

MVC: Model-View-Controller - Arquitetura

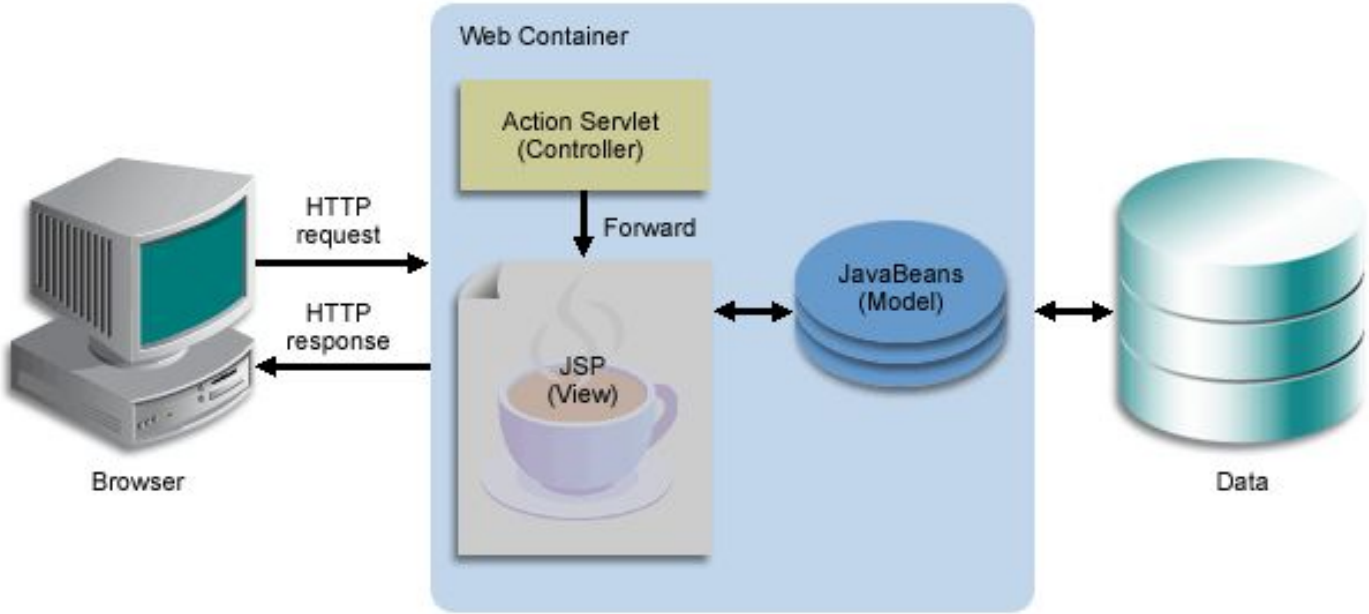
- MVC é denominado por vários autores como um Padrão Arquitetural
- Ele descreve uma forma de estruturar uma aplicação, as responsabilidades e interações, separando os componentes da aplicação em três camadas:
  - Camada de regras de negócio e dados (Model – Modelo)
  - Camada de interface (Visão – View)
  - Camada de controle e regras de interface (Controlador – Controller)



MVC: Model-View-Controller - Arquitetura - Modelo 1

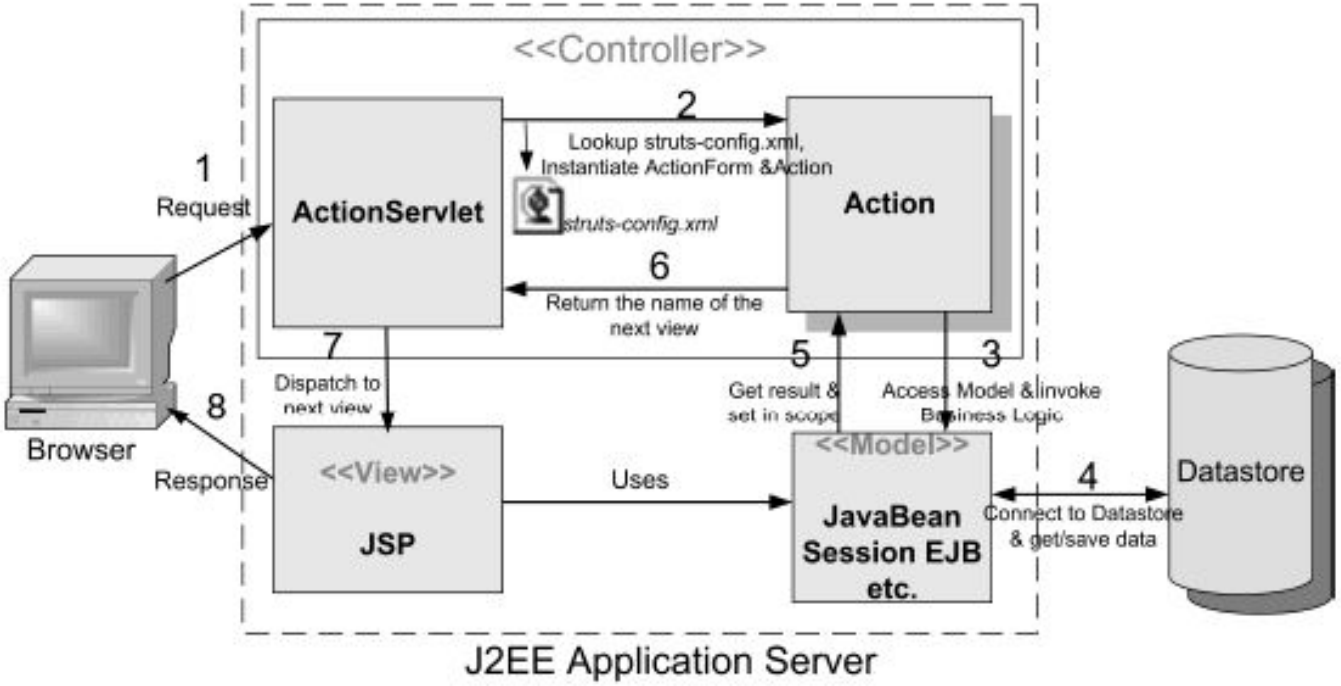


MVC: Model-View-Controller - Arquitetura - Modelo 2



Framework Struts

- Struts é um *Framework* e uma das implementações do MVC



Apresentação de Servlet e JSP

- Numa aplicação MVC os papéis previstos são desenvolvidos da seguinte forma:
  - VIEW’s => Páginas JSP
  - CONTROLLER’s => Servlets Java
  - MODEL’s => Classes Java

Roteamento de Requisições HTTP

- Para que seja possível implementar o MVC, é necessário rotear requisições HTTP de Views para Controllers e de Controllers para Views

- Forma de roteamento:
  - View => Controller: submeter a requisição do navegador ao servidor WEB simplesmente apontando para a URL correta

  - Controller => View: esse roteamento é feito internamente no servidor WEB, normalmente pelo código:

```
request.getRequestDispatcher("[URL do JSP]").forward(request, response)
```

Desenvolvimento de Aplicações MVC - Exemplo

- Desenvolvimento de um conjunto de fontes que ilustrem o conceito de um *framework* MVC:
  - Página JSP representando a VIEW
  - Classe Java representando o CONTROLLER
  - Classe Java representando o MODEL

- Roteamento de requisições Controller => View e View => Controller

- Fluxo: [1] VIEW => [2] CONTROLLER => [3] MODEL => [1] VIEW => [2] CONTROLLER => [3] MODEL => ...

Exercício 1

- Descreva o que é MVC e caracterize o conceito de padrão arquitetural e arcabouço conceitual (*framework*) envolvido

- Quais são as vantagens de se utilizar o MVC numa aplicação?