# Curso de Java

## Geração de Páginas WEB


Numa aplicação WEB de ‘conteúdo dinâmico’

- As páginas são construídas a partir  de dados da aplicação (mantidos num banco de dados).
- Usuários interagem com a aplicação através de formulários.
- Os primeiros servidores HTTP permitiam a interação com aplicações externas através do mecanismo conhecido como CGI (‘Common Gateway Interface’). Nesse esquema, cada requisição de página implicava na criação de um processo responsável pelo seu tratamento.


Aplicações WEB feitas em Java requerem o uso de um 'Servidor de Aplicações Java' responsável por

- Oferecer um 'ambiente de execução' para aplicações Java.
- Gerenciar simultaneamente diversas aplicações Java.
- Encaminhar as requisições HTTP às aplicações Java sob seu controle.

*Num servidor de aplicações Java, um objeto da classe Servlet é responsável por atender a uma requisição WEB. O servidor web mantém controle da associação das URL's aos Servlets sob seu controle.*


## Apache TomCat:

- Servidor de aplicações Java desenvolvido dentro do projeto Apache.
- Disponibilizado na modalidade 'Open Source'.
- É a implementação de referência para Servlets, recomendada pela Sun.
- É o servidor de aplicações Java mais utilizado no mercado.
- Utilizado como base em servidores de aplicação JEE (p. ex. JBoss e Geronimo).


Objetivo principal:

- Atender a requisições HTTP através dos métodos ‘get’ e ‘post’ previstos no protocolo.

Implementam os métodos relacionados ao tratamento de cada tipo de requisição:

- doGet(): atende uma requisição feita através do método ‘get’ do protocolo HTTP.
- doPost(): atende a uma requisição feita através de ‘post’.

# Um exemplo


```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class HelloWorld extends HttpServlet {
   public void doGet(HttpServletRequest  request,
                     HttpServletResponse response)
         throws ServletException, IOException {
       PrintWriter out = response.getWriter();
       out.println("Hello World");
   }
}
```


Neste exemplo:

- HttpServlet é a classe base para os servlets que tratam requisições HTTP. Essa classe prevê os métodos **`doGet()`** e **`doPost()`**.
- HttpRequest e HttpResponse são classes que descrevem respectivamente a requisição e a resposta HTTP.
- O comando

  ```
  PrintWriter out=response.getWriter();
  ```

  é usado para acessar o dispositivo de saída para a resposta.
- A resposta à requisição é um texto (sem formatação) escrito no dispositivo de saída.

# Outro exemplo


```
public class ServletExemplo extends HttpServlet {
  protected void doGet( HttpServletRequest  request,
                        HttpServletResponse response)
            throws ServletException, IOException {

    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 " +
                "Transitional//EN\">\n" +
                "<HTML>\n" +
                  "<HEAD><TITLE>Hello WWW</TITLE></HEAD>\n" +
                  "<BODY>\n" +
                      "<H1>Hello WWW</H1>\n" +
                   "</BODY></HTML>");
  }
}
```

# *Outro exemplo*


Neste exemplo:

- O comando

  ```
  response.setContentType("text/html");
  ```

  indica que a resposta será dada como texto HTML.

- A resposta à requisição é uma página HTML escrita no dispositivo de saída.

# Acesso aos dados do formulário


Num formulário HTML cada componente tem uma identificação que é utilizada pelo Servlet para acessar os dados fornecidos pelo usuário. Um exemplo: se na formulário tivermos uma caixa de texto definida como

```
<input type="text" name="entrada" size="20">
```

no Servlet responsável pelo tratamento dessa página temos acesso ao texto digitado nessa caixa através dos comandos

```
String tb = request.getParameter("entrada");
```

Esse mecanismo pode ser utilizado da mesma forma nos métodos **`doGet()`** e **`doPost()`**.

# *Páginas dinâmicas – outros mecanismos*

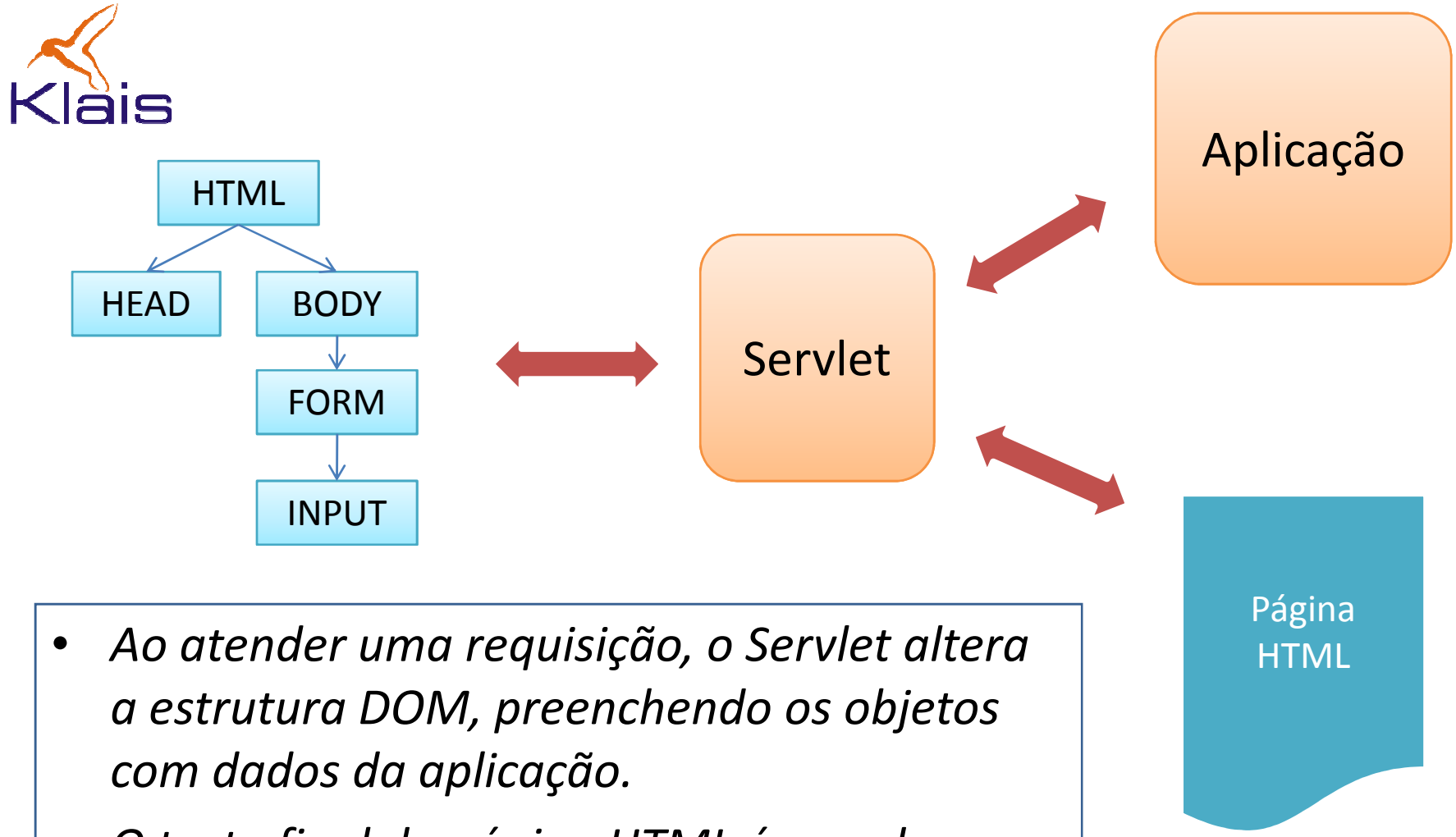

- JSP: Java Server Pages – Combinam código Java com texto HTML que são pré-compiladas pelo servidor de aplicações quando é feito o primeiro acesso. A 'compilação' de uma página JSP gera um Servlet que é executado da mesma forma que um Servlet normal.
- Manipulação do *DOM*(*Document Object Model*): a página HTML sendo gerada é representada como uma estrutura de objetos ao qual o Servlet agrega conteúdo dinâmico de acordo com as necessidades da aplicação.

*A geração de páginas dinâmicas é um dos principais problemas enfrentados ao se desenvolver aplicações WEB. No caso de Java, existem atualmente diversos frameworks que se propõem a atacar esse problema.*


```
<HTML>
  <BODY>
     <%
       // Entre comentários podem ser colocados
       // que são executados na apresentação da página.
       System.out.println( "Evaluating date now" );
       java.util.Date date = new java.util.Date();
     %>
     Hello!  The time is now <%= date %>
  </BODY>
</HTML>
```

*Ao iniciar a aplicação o servidor de aplicações 'compila' o texto JSP, gerando um Servlet equivalente.*

# Manipulação do DOM




- *Ao atender uma requisição, o Servlet altera a estrutura DOM, preenchendo os objetos com dados da aplicação.*
- *O texto final da página HTML é gerado a partir do DOM alterado.*


## Klais Forms

Um Servlet chamado TemplateServlet  constrói a página a partir de

- um modelo (*template*)  representado como uma estrutura de objetos e que é base para a página a ser gerada. Esse modelo define 'regiões' da página (como p. ex. 'cabeçalho', 'rodapé', 'corpo', 'menu', etc) e cada região tem uma identificação única.  A estrutura de objetos é construída a partir de uma página HTML padrão, que pode usar uma folha de estilo e código Javascript.
- Um mapa (*ScreenData*) contendo os componentes visuais e dados a serem associados a cada região da página.
- Um mapa que associa dados da aplicação aos componentes visuais associados à página. Esse mapa pode ser construído dinamicamente pela aplicação.

Com base  no modelo e nos mapas, o TemplateServlet gera as páginas HTML dinâmicas da aplicação.