**Aula XXII - Desenvolvimento de Web III**

**Páginas Dinâmicas & Servelets**

**Java Server Pages**

**Java Web**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Fontes:***

[*https://www.devmedia.com.br/introducao-ao-desenvolvimento-de-aplicacoes-web/29798*](https://www.devmedia.com.br/introducao-ao-desenvolvimento-de-aplicacoes-web/29798)

[*https://www.devmedia.com.br/artigo-java-magazine-66-desenvolvendo-java-para-web/12289*](https://www.devmedia.com.br/artigo-java-magazine-66-desenvolvendo-java-para-web/12289)

[*https://www.alura.com.br/artigos/criando-uma-aplicacao-java-web-com-servlet*](https://www.alura.com.br/artigos/criando-uma-aplicacao-java-web-com-servlet)

[*https://www.caelum.com.br/apostila-java-web/javaserver-pages/*](https://www.caelum.com.br/apostila-java-web/javaserver-pages/)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Web Dinâmica**

No início a Internet era estática, a falta de interatividade imperava, foi quando surgiu o CGI, a tecnologia CGI criava interatividade ao ambiente, mas não tornava simples o seu desenvolvimento. Entre as tecnologias de CGI mais utilizadas, destaca-se o PERL, linguagem poderosa, utilizada até hoje para administração de servidores. O grande problema de se desenvolver um **CGI**, além do desempenho, era a sua complexidade, a solução para simplificar o desenvolvimento WEB e passar para uma nova fase, surgiram as linguagens server-side scripts como ASP, PHP e JSP. Estas linguagens tornaram o desenvolvimento web mais ágil, simples e prático. Além de possuir um maior desempenho de aplicação, em um meio que a velocidade é essencial, isto garantiu o sucesso das novas linguagens de scripts.

Uma das primeiras ideias para esses "geradores dinâmicos" de páginas HTML foi fazer o servidor

Web Invocar um outro programa externo em cada requisição para gerar o HTML de resposta. Era o famoso CGI que permitia escrever pequenos programas para apresentar páginas dinâmicas usando,por exemplo, Perl, PHP, ASP e té C ou C++.

Na plataforma Java, a primeira e principal tecnologia capaz de gerar páginas dinâmicas são as

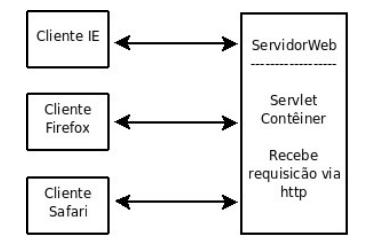
Servlets, que surgiram no ano de 1997. Hoje, a versão mais encontrada no mercado é baseada nas versões 2.x.

**Servelets**

As **Servelets** são uma das principais formas para criar páginas dinâmicas com Java, basta usar a própria linguagem Java para isso por meio da criação de uma classe que terá capacidade de operar no conteúdo HTML.

Uma primeira ideia de **servelet** é que cada uma delas é responsável por uma página, sendo que ela lê dados de requisição do cliente e responde com outros dados (como vimos na aula anterior).

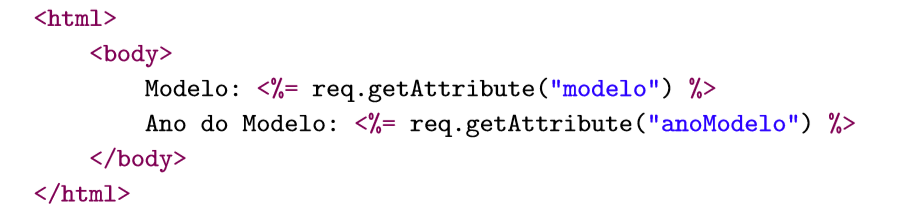
Importante: cada **Servelet é um objeto Java que recebe as requisições (request) e produz uma resposta (response), com uma página HTML dinamicamente gerada.**



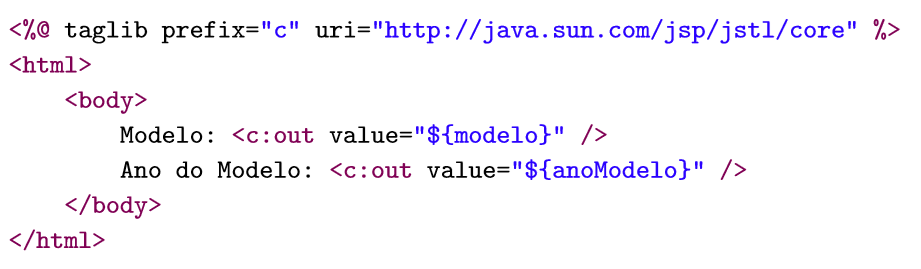
**Java Web**

**Java Server Pages (JSP)**

**As JSPs, são páginas HTML que podem ter dentro de si trechos de código Java.** Como em um projeto real, a quantidade de HTML dentro de uma Servlet seria muito maior do que o visto nos exemplos anteriores, e na prática o código Java de verdade (e não o que escrevia HTML) era exceção. Com as JSPs em vez de ter algo parecido com o servlet que respondia por /visualizar, poderíamos ter uma página como visto abaixo:



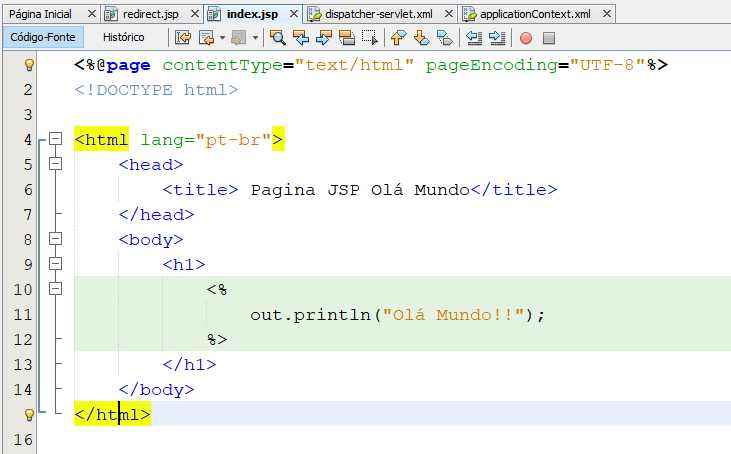
Veja como nosso código agora pareceu melhor. **Agora a proporção parece estar mais correta, sendo o HTML a regra e o Java a exceção.** Nas JSPs temos a possibilidade da escrita direta do HTML, que além de deixar o código mais legível, deixa-o mais fácil e ágil de fazer a manutenção. Apesar da notória melhoria, na mistura entre HTML e Java permaneceram, através dos **Scriptlets**, aqueles estranhos sinais de **<% %>**.Isso provocou o surgimento de técnicas para eliminar esse código Java da visualização do resultado. Foi justamente o papel da **JSTL (JavaStandardTagLibrary) e da EL (ExpressionLanguage)**.

De acordo com a imagem acima é possível escrever HTML com códigos JSP embutidos. **Como o HTML é uma linguagem estática, o JSP será o responsável por criar dinamismo.** Por ser gratuita e possuir especificação aberta possui diversos servidores que suportam a linguagem, entre eles temos: **Tomcat, GlassFish, JBoss,** entre outros. O JSP necessita de servidor para funcionar por ser uma linguagem Server-side script, o usuário não consegue ver a codificação JSP, pois esta é convertida diretamente pelo servidor, sendo apresentado ao usuário apenas codificação HTML.

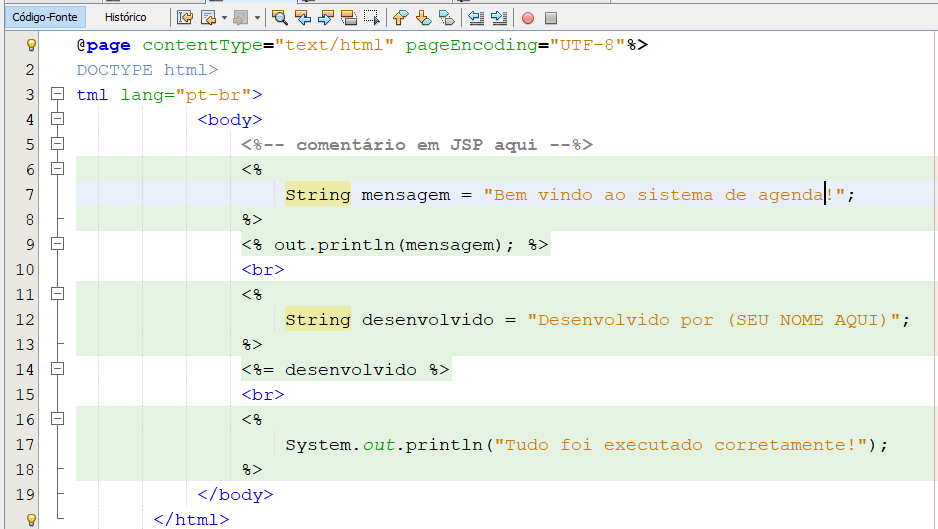
O objetivo da linguagem JSP não é só o desenvolvimento de páginas dinâmicas para Internet. Com ela é possível desenvolver sistemas inteiros para Internet. Além disso, existem diversos benefícios em se utilizar a linguagem JSP.

**JSP é embutido no HTML**

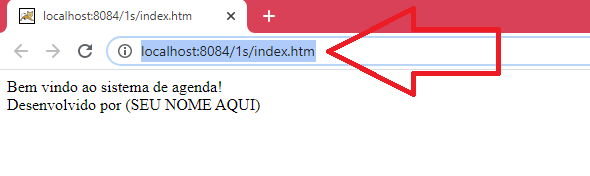
O JSP é simples de se utilizar, podendo gerar o HTML ou ainda estar embutido dentro do HTML, como no exemplo a seguir:



ou



Para ver o que fizemos, basta executar pelo servidor TomCat ou o GlassFish, ambos criam uma URL para que possamos conferir o que declaramos no JSP, um Localhost, usando a porta 8084.



|  |
| --- |
| **Dica Tomcat (DEVMEDIA -** [***https://www.devmedia.com.br/introducao-ao-java-server-pages-jsp/25602***](https://www.devmedia.com.br/introducao-ao-java-server-pages-jsp/25602)**)**  **Tomcat**  O JSP por ser uma linguagem Server-Side como visto anteriormente, necessita de um servidor para o seu funcionamento.  O Apache Tomcat foi desenvolvido pela Apache Software Foundation, o Tomcat é um servlet container de código aberto, ou seja, uma aplicação que interpreta e processa servlets (java servlets) e JSP (Java Server Pages). O servidor está disponível livremente na Internet sem a necessidade de pagamento de licenciamento e está disponível no endereço http://tomcat.apache.org para diversas plataformas, entre elas o Windows, Linux, Solaris etc. Recomenda-se o download da versão 7 do servidor. A instalação é simples, basta seguir o seu passo a passo. O desenvolvedor deve fazer a instalação completa do servidor para seu sistema operacional.  Com o Tomcat instalado pode-se iniciar o desenvolvimento JSP. Para iniciar o desenvolvimento deve-se criar um diretório easyjava em C:\Program Files (x86)\Apache Software Foundation\Tomcat 7.0\webapps (no Windows) e /usr/java/apache-tomcat-7.0 (no Linux)”. No diretório easyjava deve-se criar os diretórios src, web, WEB-INF e WEB-INF/lib.   * src - Diretório onde fica armazenado o código fonte dos servlets; * web - Diretório raiz da aplicação web; * WEB-INF - Diretório que armazena o descritor da aplicação web (web.xml), bem como outros arquivos de configuração. Este diretório é invisível ao usuário; * WEB-INF/lib - Bibliotecas necessárias para a aplicação. |

O JSP, assim como o PHP, pode ser utilizado dentro do HTML, a página JSP, na verdade, é uma página HTML, mas quando for necessário utilizar codificação JSP que deve estar entre as tags HTML:

|  |
| --- |
| **<% código jsp %>** |

O servidor trata de converter para HTML para responder a solicitação do usuário.

Na codificação JSP, têm-se algumas diretivas, as quais são utilizadas para informações especiais dentro de páginas, sendo dividido em três tipos:

* **@include** - Utilizado para inserir os códigos de arquivos à página corrente;
* **@page** - Responsável por trazer informações sobre a página JSP;
* **@taglib** - Responsável por habilitar uma biblioteca de tags personalizada (item que será abordado em outro artigo com mais detalhes).