**AUTORIA WEB**

**Aula 01 - Linguagens e Ferramentas para o Desenvolvimento Web**

- www – É um sistema para compartilhar e interligar documentos hipermídia. Web é um serviço que opera na internet.

- O software woldwideweb foi criado em 1991 por Tim Berners-Lee, com o propósito de interligar documentos de cientista de todo o mundo.

- Mosaic foi o navegador que que liderou o boom da internet no início dos anos 90.

- Página estativa contém conteúdo HTML puro, e seu conteúdo não varia com a interação do usuário.

- Página dinâmica possui conteúdo HTML e código gerados por alguma linguagem de programação, o que permite uma interação do usuário com o conteúdo mostrado e uma dinâmica automática na construção da página.

- W3C (world wide web consortium) é a principal organização da WWW que tem a finalidade de estabelecer padrões (HTML, XHTML, CSS, internacionalização e web móvel) para a criação e interpretação de conteúdos para a Web.

- HTML é uma linguagem de marcação e não uma linguagem de programação. É uma linguagem constituída de tags de marcação que são delimitadas por símbolos de < e >. Exemplo <html> e </html>.

- XML é uma linguagem de marcação que exige uma correta formatação. XML é projetado para descrever dados enquanto a HTML para mostrar dados.

- CSS (cascading style sheets) são estilos escritos em um arquivo com extensão .css, que definem como os elementos html serão mostrados na tela.

- Inserir um comentário em um documento HTML

<!- Comentário html ->

**Aula 02 - HTML: Cabeçalhos, Títulos, Parágrafos e Links**

- HTML significa linguagem de marcação para hiper texto.

- Tags devem ser iniciadas e fechadas, pois delimita os elementos que serão formatados por ela.

- Existem tags vazias que apenas inserem alguma coisa no texto. Ex. <br> , insere quebra de linha.

- Na abertura de tags pode-se ter atributos, que são nomes e seus valores, que oferecem informações adicionais ao elemento. Ex <tag nome = “valor” .... > ... </tag>

- Estrutura de um documento HTML

\* No HTML5 a única declaração necessária e <!DOCTYPE html>

\* Documentos HTML são divididos em duas partes:

1 - Informação sobre a versão do HTML.

2 – O documento delimitado entre as tags <html> </html>. Que tem duas subdivisões:

1. O cabeçalho da página, delimitado pelas tags <head> </head>
2. O corpo do documento, delimitado pelas tags <body> </body>

\* Possíveis tags do cabeçalho:

Tag Descrição

<title> Especifica o título do documento.

<base/> Especifica um endereço ou um alvo para os links do documento.

<link /> Especifica a relação entre o documento e arquivos externos.

<meta/> Especifica metadados.

<script> Especifica scripts.

<style> Especifica o estilo dos elementos do documento.

\* Scheme: provê o contexto para interpretação de um metadado. Por exemplo, na declaração seguinte, o atributo scheme pode ajudar a indicar que o identificador é um número ISBN.

Ex. <meta scheme="ISBN" name="identifier" content="0-8230-2355-9"/>

O atributo scheme não é suportado no HTML 5.

No HTML5, a forma de definir a codificação de caracteres foi simplificada com a adição de um novo atributo, o charset. Ex. <meta charset="UTF-8">

\* O elemento <link> define um relacionamento entre o documento e algum recurso externo.

Ex. <link rel="help" href="/help/">

Ex. <link rel="stylesheet" type="text/css" media="screen" href="css/w3c.css" />

\* tag <base target=”\_blank”/> indica que todos os links devem ser abertos em uma nova janela do navegador.

\* Tags de formatação de textos:

Tag Descrição

<b> Define texto em negrito

<u> Define texto sublinhado

<s> Define texto riscado

<i> Define texto em itálico

<small> Define texto em uma fonte de tamanho menor

<sub> Define texto subescrito

<sup> Define texto superescrito

**Aula 03 - HTML: Figuras, Tabelas, Listas e Formulários**

- A inserção de imagens em páginas HTML é feita na seção de corpo da página body>...</body>), através do elemento <img/> ou <object>.

- A tag <img/> pode possuir basicamente 5 atributos: src, alt, longdesc, width e height.

- src – Indica o caminho do arquivo referente à figura, incluindo o seu formato (pode ser jpg, gif, png, dentre outros tipos).

- alt – Fornece uma descrição curta sobre a imagem quando o cursor move-se sobre a mesma e, por algum motivo, o navegador não consegue mostrá-la.

- longdesc – Especifica um link para uma descrição extensa da imagem. Essa descrição deve suplementar a descrição provida pelo atributo alt.

- Width/Height – Especifica a largura e a altura da imagem, respectivamente. Os valores podem ser em pixels ou proporcionais ao tamanho que a figura ocupará na página.

Ex. <img src= “url.jpg” alt=“Texto Descritivo” longdesc=”descrição.html” width=“330” height=”20%”/>

- Uma imagem pode ser inserida em um documento HTML através do elemento object. Essa tag é mais genérica e pode inserir figuras, vídeos e applets em um documento HTML.

<object data="URI" type="image/jpg" width="100" height="100" >descricao da imagem </object>

- LISTAS

Existem três tipos de listas: ordenadas, não ordenadas e de definição. Cada tipo de lista é delimitada por uma tag diferente:

<ol>...</ol>– para listas ordenadas (ordered list). Essas listas são utilizadas quando se quer numerar algo; Os itens desta lista são delimitados pelas tags <li> </li>

<ul>...</ul>– para listas não ordenadas (unordered list). Essas listas são utilizadas quando se quer apenas colocar um marcador em seus itens, mas sem numeração; Os itens desta lista são delimitados pelas tags <li> </li>.

<dl>...</dl>– para listas de definição (definition list). Essas listas são usadas quando, além de topificar, ainda se deseja especificar uma definição para cada tópico. Os itens desta lista são delimitados pelas tags <dd> </dd>.

TABELAS

Tabelas são definidas com a tag <table>…</table>.

Elas são divididas em linhas, delimitadas pela tag <tr>...</tr>, e colunas, delimitadas pela tag <td>...</td>, nas quais se determina o conteúdo das células.

As tags <thead>, <tbody> e <tfoot> são usadas para agrupar informações sobre o cabeçalho, o conteúdo e o rodapé da tabela, respectivamente.

FORMULÁRIOS

Formulários são definidos pela tag <form>...</form>.

Três dos importantes atributos da tag <form> são:

\* action =”uri” – especifica para onde enviar as informações inseridas no formulário.

\* enctype – O atributo enctype define como o form-data deve ser codificado quando é submetido para o servidor.

Ex. <form action="mailto:endereco@correio.com" method="post" enctype="text/plain">...</form>

\* method= get|post – especifica qual método HTTP será usado para submeter os dados do formulário. O método "get" anexa as informações na URL do formulário. Já o método post é recomendado quando os dados são privados, tais como senhas.

\* Os elementos de controle de formulários são os elementos que possibilitam a entrada de dados, tais como botões, caixas de texto, check boxes, dentre outros.

- Caixas de texto são usadas para que o usuário possa fornecer texto. Elas podem ser definidas com o elemento <input>, criando uma única linha de texto, ou com o elemento <textarea>, criando múltiplas linhas de entrada de texto.

Para receber textos curtos:

<input type="text" name="nome"> podem ter ainda os atributos size (define o tamanho que a caixa terá na tela), maxlength (define o tamanho máximo do texto que será inserido), value (define um valor para o campo em questão).

Ex. <input type="text" name="nome" value="João"/>

Para receber textos ocultos:

<input type="password" name="senha"/>

Para receber textos longos:

<textarea name="comentário" rows="10" cols="40">Escreva seu comentário....</textarea> rows – define o número de linhas do campo de texto; cols – define o número de colunas do campo de texto. E para escrever um valor inicial para esse campo, deve-se escrevê-lo entre as tags de início e fim de textarea.

Caixas de seleção de arquivo são controles que permitem que o usuário procure e selecione um arquivo em seu computador para ser submetido em um formulário.

Ex. <input type=”file”.../>

BOTÕES

Os botões são definidos pela tag <button> ou <input>. Como a tag <button> pode não ser vazia, então é possível incluir figuras nos botões, enquanto que com a tag <input/> apenas um texto pode ser inserido.

Existem 3 tipos de botões:

Botões submit – servem para enviar um formulário.

Botões reset – servem para incluir em todos os campos seus valores iniciais ou limpar os campos, caso não tenha sido dado nenhum valor inicial para eles.

Botões push – servem para associar algum comportamento através de scripts.

RADIO BUTTON

<input type="radio" name="sex" id="Masculino" checked="checked"/>

<label for="Masculino">Masculino</label><br/>

<input type="radio" name="sex" id="Feminino"/>

<label for="Feminino">Feminino</label>

CHECKBOX

<input type="checkbox" name="areap" value="1" id="areap1"/>

<label for="areap1">Administração </label>

<input type="checkbox" name="areap" value="2" id="areap2"/>

<label for="areap2">Educação</label>

<input type="checkbox" name="areap" value="3" id="areap3" checked=”checked” />

<label for="areap3">Informática</label>

**Aula 04 - XHTML e CSS: definições e particularidades**

- XHTML, linguagem de marcação similar ao HTML criada com base no XML e que serviu como base (além do HTML tradicional) para a criação do HTML5.

CSS

- CSS (Cascading Style Sheet) significa, em português, Folhas de Estilo em Cascata. Mas o que significa o termo cascata? Significa que a aplicação de CSS a um documento ocorre como um efeito cascata.

**Aula 05 - Estilizando planos de fundo, fontes, links e listas**

- Referência para adicionar um arquivo CSS externo na página.

<link rel="stylesheet" type="text/css" href="style.css" />

**Aula 06 - CSS: cores, tabelas e formulários**

**Aula 07 - CSS: modelo de caixas, bordas, dimensões e posicionamento**

- O termo caixa é utilizado no contexto de design e layout. Caixas são retângulos a serem exibidos (na tela ou impressos) que consistem de margens, bordas, espaçamentos (padding) e conteúdo.

- Margin – especifica uma área ao redor da borda, ou seja, determina a distância entre caixas.

- Border – como você viu na estilização de tabelas e formulários, uma borda determina um tipo de linha decorativa ao redor de um elemento. Cada caixa pode ter 4 bordas, a saber: de cima, da direita, de baixo e da esquerda. Bordas são, por padrão (default), transparentes e de espessura zero.

- Padding – determina a distância entre a borda e o conteúdo.

- Content – indica o conteúdo da caixa; é onde o texto e imagens aparecem.

**Aula 08 - Layout CSS, arquitetura da informação e conteúdo para web**

- Um layout CSS especifica a estrutura geral da aparência de um site. Nele define-se o posicionamento dos conteúdos, tais como slogan, menus, conteúdo principal, conteúdo secundário, dentre outros.

- Os layouts são classificados dois critérios:

\* Quantidade de colunas: pode ter uma ou mais colunas.

\* Largura da página: a largura pode ser: fixa, liquido, elástico ou hibrido.

- Fixa: a página e suas colunas possuem uma largura fixa, não importando o tamanho da janela do browser nem a resolução do monitor.

- Liquido: as larguras são definidas em porcentagem. Assim, o conteúdo se acomoda à largura da janela do browser (ou resolução do monitor), otimizando o aproveitamento do espaço.

- Elástico: as larguras são definidas com base no tamanho das fontes, usando a unidade “em”. Isso fará com que a largura da página se comporte da mesma forma que o tamanho do texto, aumentando quando a letra aumenta e diminuindo quando a letra diminui.

- Híbrido: a largura da página, colunas e demais elementos são definidos em “ems” e adicionalmente uma largura máxima é definida para eles em porcentagem. Ou definem-se as larguras em porcentagem e as larguras máximas e mínimas em “em”.

- Passos básicos para você conseguir definir o layout de um site:

1 - Escolher e nomear os containers do layout.

2 - Criar a marcação para esses containers.

3 - Escolher o método de posicionamento para os containers.

4 - Inserir e estilizar conteúdos.

5 - Ligar o CSS.

6 - CSS para impressão.

- Passo 1 - Escolher e Nomear os Containers do Layout

\* Um container é uma caixa que geralmente agrupa vários elementos (X)HTML de forma a estruturar o conteúdo da página uniformemente.

- Passo 2 - Criar a marcação para esses containers.

\* Os containers são marcados com a tag <div>. A tag <div> serve para agrupar elementos e, juntamente com os atributos id e class, possibilita definir a estrutura de um documento.

Ex. <div id="container">

<div id="header"></div>

<div id="menu"></div>

<div id="contents"></div>

<div id="footer"></div>

</div>

- Passo 3 - Escolher e Especificar o Método de Posicionamento para os Containers

\* Container principal:

#container{

margin: 10px auto;

width: 650px;

background: #6E934A;

}

\* Container Header:

#header {

background: #8074CB;

height: 50px;

}

\* Container Menu:

#menu {

background: #DBBD4B;

width:200px;

float:right;

height: 500px;

}

\* Container Contents:

#contents {

background: #8074CB;

height:500px;

}

\* Container Footer:

#footer {

background: #4D4197;

height: 30px;

}

- Passo 4 - Inserir e Estilizar Conteúdos.

**Desenvolvimento Web I**

**Aula 01 - Ambiente de desenvolvimento para Web em Java**

- Será utilizado um ambiente de desenvolvimento com Eclipse + Tomcat Server.

- Configuração do servidor no Eclipse é feita pelo menu: Window – preference – Server – Runtime Environments.

- Iniciar uma aplicação web no Eclipse: File – New – other – web – Dynamic Web Project.

- Criar um arquivo para a aplicação: Clique com botão direito na pasta da aplicação: New – JSP File – Web Content.

**Aula 02 - Introdução aos Servlets**

- Para gerarmos páginas dinâmicas no servidor de aplicações web, fazemos uso de componentes de software chamados Servlets.

- Servlets são nada mais que classes em Java que herdam código (atributos e métodos) direta ou indiretamente de uma classe especial chamada HttpServlet existente no pacote javax.servlet.http.

- A resposta de Servlet é dado por:

\* O instanciamento de uma variável de saída através do método response.getWriter().

\* O uso do método write(), da variável instanciada, para fazer descrever a resposta.

\* O uso do método close() para encerrar a construção da resposta e o navegador apresenta-la.

Ex: @WebServlet("/ServletExemplo")

public class ServletExemplo extends HttpServlet {

private static final long serialVersionUID = 1L;

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// TODO Auto-generated method stub

PrintWriter saida = response.getWriter();

saida.write("<HTML><BODY>Olá!</BODY></HTML>");

saida.close();

- Para informar que dada aplicação é um Servlet. Isto pode ser feito através do aquivo web.xml.

<servlet>

<description></description>

<display-name>ServletExemplo</display-name>

<servlet-name>ServletExemplo</servlet-name>

<servlet-class>aula02.ServletExemplo</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ServletExemplo</servlet-name>

<url-pattern>/ServletExemplo</url-pattern>

</servlet-mapping>

Ou fazendo a anotação @WebServlet("/ServletExemplo") que tira a necessidade de o mapeamento do Servlet pelo arquivo web.xml.

- Ciclo de Vida dos Servlet

1- Carregar a classe Servlet e instanciar.

2 – Executar o método init().

3 – Executar o método service().

4 – Executar o método destroy(), quando o Servlet não for mais necessário.

**Aula 03 - Manipulando requisições e respostas HTML com Servlets**

- As requisições a serem trafegados via métodos HTTP diferentes, GET ou POST, têm acessos aos parâmetros que são o mesmo: através do uso da classe javax.servlet.http.HttpServletRequest.

- O método getParameter() da classe HttpServletRequest, o qual recebe como parâmetro uma String que representa o nome do parâmetro da requisição web cujo valor você deseja ter acesso.

**Aula 04 - Trabalhando com sessões**

- Para manter o estado do cliente no lado do servidor, precisamos de uma classe Java para representar esse estado. O nome dessa classe (na verdade, interface) é javax.servlet.http.HttpSession.

**Aula 05 - Introdução e Recursos JavaServer Pages**

-

**Aula 06 - MVC, Servlets e JavaServer Pages**

-

**Desenvolvimento Web II**

**Aula 01 - Introdução ao Javaserver Faces**

- O JSF é um framework para o desenvolvimento de interfaces de usuário baseada em componentes para aplicações Web utilizando a linguagem Java.

- O JSF é um framework MVC e contém diversos serviços que facilitam a vida do desenvolvedor.

- Em uma aplicação típica JSF encontraremos as seguintes partes:

\* Facelets: conjunto de partes XHTML, representando o view do MVC.

\* Managed Beans: conjunto de classes Java, representando o controller do MVC.

\* Conjunto de classes de domínio referente ao negócio da aplicação. Model do MVC.

\* Arquivo XML: descritor de implementação web.

**Aula 02 - Conhecendo as Tags do JSF**

- JSF possui tags disponibilizadas que representam componentes HTML. Que são agrupadas em dois grupos: Core e HTML.

Ex: <h:inputText value = “Oi!” renderiza <input type="text" value="Oi!" />

- Para usar essas tags devemos informar os namespace utilizado na página

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:f="http://java.sun.com/jsf/core">

</html>

**Aula 03 - Validação de Campos no JSF**

- Validações de campo refletem as regras de negócio das aplicações.

- Portanto, validar informações envolve duas tarefas principais:

1 - Verificar se as informações necessárias para continuação da funcionalidade estão presentes, ou seja, foram informadas pelo usuário, e estão no formato correto;

2 - Exibir mensagens de validação quando alguma informação estiver faltando (não informada pelo usuário) ou no formato incorreto. Ao exibir essas mensagens, é importante mantermos os dados já informados pelo usuário que estão corretos.

- o JSF possui duas formas de validação, as quais veremos mais detalhadamente a seguir:

* Declarativa: utilizando os validadores padrões do JSF;
* Imperativa: método de validação no managed bean e classes validadoras que implementam a interface javax.faces.validator.Validator.

- A validação de campo obrigatório verifica se um determinado campo não está vazio durante a submissão do formulário.

Ex: <h:inputText id="nome" value="#{pessoaMBean.pessoa.nome}" title="Nome" required="true" requiredMessage="Nome: campo obrigatório!"/>

**Aula 04 - Desenvolvendo um Sistema de Informação Web com JSF**

-