PROJETO ESCOLAS - REFERÊNCIA
Compromisso com a Excelência na Escola Pública

Cadernos de Informatica

CURSO DE CAPACITAÇÃO EM INFORMÁTICA INSTRUMENTAL

CURSO DE MONTAGEM E MANUTENÇÃO DE COMPUTADORES

CURSO SOBRE O SISTEMA OPERACIONAL LINUX

CURSO DE PROGRAMAÇÃO EM JAVA

CURSO DE INTRODUÇÃO A BANCOS DE DADOS

CURSO DE CONSTRUÇÃO DE WEB SITES

CURSO DE EDITORAÇÃO ELETRÔNICA

CURSO DE ILUSTRAÇÃO DIGITAL

CURSO DE PRODUÇÃO FONOGRÁFICA

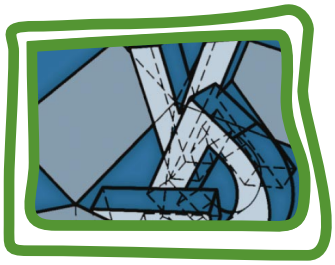
CURSO DE COMPUTAÇÃO GRÁFICA 3D

CURSO DE PROJETO AUXILIADO POR COMPUTADOR

CURSO DE MULTIMÍDIA NA EDUCAÇÃO

Cadernos de Informatica

CURSO DE CONSTRUÇÃO DE WEB SITES



Glayston Pereira Batista
Coordenador
Carlos Eduardo Hermeto de Sá Motta

Apresentação

Os computadores que estão sendo instalados pela SEE nas escolas estaduais deverão ser utilizados para propósitos administrativos e pedagógicos. Para isso, desenvolveu-se um conjunto de cursos destinados a potencializar a utilização desses equipamentos. São doze cursos que estão sendo disponibilizados para as escolas para enriquecimento do seu plano curricular. Esses cursos não são profissionalizantes. São cursos introdutórios, de formação inicial para o trabalho, cujo objetivo é ampliar o horizonte de conhecimentodos alunos para facilitar a futura escolha de uma profissão.

Todos os cursos foram elaborados para serem realizados em 40 módulos-aula, cada um deles podendo ser desenvolvidos em um semestre (com 2 módulos-aula semanais) ou em 10 semanas (com 4 módulos-aula semanais). Em 2006, esses cursos deverão ser oferecidos para os alunos que desejarem cursá-los, em caráter opcional e horário extra-turno.

Em 2007, eles cursos deverão ser incluídos na matriz curricular da escola, na série ou séries por ela definida, integrando a Parte Diversificada do currículo.

Esses cursos foram concebidos para dar aos professores, alunos e funcionários uma dimensão do modo como o computador influencia, hoje, o nosso modo de vida e os meios de produção. Para cada curso selecionado pela escola deverão ser indicados pelo menos dois ou, no máximo, três professores (efetivos, de preferência) para serem capacitados pela SEE. Esses professores irão atuar como multiplicadores, ministrando-os a outros servidores da escola e aos alunos.

Curso de Capacitação em Informática Instrumental

Este curso será implantado obrigatoriamente em todas as escolas estaduais em que for instalado laboratório de informática. Iniciando pelas Escolas-Referência, todos os professores e demais servidores serão capacitados para que possam fazer uso adequado e proveitoso desses equipamentos tanto na administração da escola como nas atividades didáticas.

É um curso voltado para a desmistificação da tecnologia que está sendo implantada. O uso do computador ainda é algo difícil para muitas pessoas que ainda não estão muito familiarizadas com essas novas tecnologias que estão ocupando um espaço cada vez maior na escola e na vida de todos. Este curso vai motivar os participantes para uma aproximação com essas tecnologias, favorecendo a transformação dos recursos de informática em instrumentos de produção e integração entre gestores, professores e demais servidores. As características dos equipamentos e as funcionalidades dos programas serão apresentadas de maneira gradual e num contexto prático. Essas.situações práticas serão apresentadas de maneira que o participante perceba o seu objetivo e o valor de incorporá-las ao seu trabalho cotidiano. Os participantes serão preparados

para navegar e pesquisar na internet; enviar, receber e administrar correspondência eletrônica, além de criar e editar documentos (textos, planilhas e apresentações) de interesse acadêmico e profissional. Esse é um curso fundamental, base e pré-requisito para todos os demais.

Curso de Montagem e Manutenção de Computadores

Este curso será implantado em, pelo menos, uma escola do município sede de cada Superintendência Regional de Ensino. A indicação da escola deverá ser feita pela própria S.R.E, levando-se em conta as condições de infra-estrutura nas Escolas-Referência existentes no município. Nas escolas escolhidas será montado um laboratório de informática especialmente para a oferta desse curso.

O objetivo deste curso é capacitar tecnicamente os alunos de ensino médio que queiram aprender a montar, fazer a manutenção e configurar microcomputadores. Pode ser oferecido para alunos de outras escolas, para professores e demais servidores da escola e para a comunidade, aos finais de semana ou horários em que o laboratório esteja disponível.

Neste curso o participante aprenderá a função de cada um dos componentes do microcomputador. Aprenderá como montar um computador e como configurá-lo, instalando o sistema operacional, particionando e formatando discos rígidos, instalando placas de fax/modem, rede, vídeo, som e outros dispositivos. Conhecerá, ainda, as técnicas de avaliação do funcionamento e configuração de microcomputadores que esteja precisando de manutenção preventiva ou corretiva, além de procedimentos para especificação de um computador para atender as necessidades requeridas por um cliente.

Dos cursos que se seguem, as Escolas-Referência deverão escolher pelo menos dois para implantar em 2006.

No período de 13 a 25 de março/2006, estará disponível no sítio da SEE (www.educacao.mg.gov.br) um formulário eletrônico para que cada diretor das Escolas-Referência possa informar quais os cursos escolhidos pela sua escola e quais os professores que deverão ser capacitados. Durante o período de capacitação, os professores serão substituídos por professores-designados para que as atividades didáticas da escola não sejam prejudicadas.

1. Curso sobre o Sistema Operacional Linux

É destinado àqueles que desejam conhecer ferramentas padrão do ambiente Unix. É um curso voltado para a exploração e organização de conteúdo. São ferramentas tipicamente usadas por usuários avançados do sistema operacional. Tem por finalidade apresentar alguns dos programas mais simples e comuns do ambiente; mostrar que, mesmo com um conjunto pequeno de programas, é possível resolver problemas reais; explicar

a comunicação entre programas via rede e estender o ambiente através de novos programas. O texto didático deste curso apresenta os recursos a serem estudados e propõe exercícios. É um curso para aqueles que gostam de enfrentar desafios.

Ementa: Histórico e desenvolvimento do Unix e Linux. Login no computador. Explorando o computador (processos em execução, conexões abertas). Descrição dos conceitos de arquivo e diretório. Operações simples sobre arquivos e diretórios. Sistema de permissões e quotas.

Procurando arquivos e fazendo backups. Executando e controlando programas. Processamnto de texto. Expressões regulares. Estendendo o ambiente. Trabalho em rede. Um sistema de chat. Comunicação segura no chat (criptografia). Ainda criptografia. Sistema de arquivos como um Banco de Dados. Um programa gráfico. Programando para rede.

2. Curso de Programação Em Java

É um curso de programação introdutório que utiliza a linguagem Java. Essa linguagem se torna, a cada dia, mais popular entre os programadores profissionais. O curso foi desenvolvido em forma de tutorial. O participante vai construir na prática um aplicativo completo (um jogo de batalha naval) que utiliza o sistema gráfico e que pode ser utilizado em qualquer sistema operacional. Os elementos de programação são apresentados em atividades práticas à medida em que se fazem necessários. Aqueles que desejam conhecer os métodos de produção de programas de computadores terão, nesse curso, uma boa visão do processo.

Ementa: Conceitos de linguagem de programação, edição, compilação, depuração e execução de programas. Conceitos fundamentais de linguagens de programação orientada a objetos.

Tipos primitivos da linguagem Java, comandos de atribuição e comandos de repetição. Conceito de herança e programação dirigida por eventos. Tratamento de eventos. Programação da interface gráfica. *Array*s. Números aleatórios.

3. Curso de Introdução ao Bancos de Dados

Este curso mostrará aos participantes os conceitos fundamentais do armazenamento, gerenciamento e pesquisa de dados em computadores. Um banco de dados é um repositório de informações que modelam entidades do mundo real. O Sistema Gerenciador do Banco de Dados permite introduzir, modificar, remover, selecionar e organizar as informações armazenadas. O curso mostra como os bancos de dados são criados e estruturados através de exemplos práticos. Ao final, apresenta os elementos da linguagem SQL (Structured Query Language – Linguagem Estruturada de Pesquisa) que é uma linguagem universal para gerenciamento de informações de bancos de dados e os ele-

mentos básicos da administração desses repositórios de informação..Apesar de ser de nível introdutório, o curso apresenta todos os tópicos de interesse relacionados à área. É um curso voltado para aqueles que desejam conhecer os sistemas que gerenciam volumes grandes e variados de informações, largamente utilizados no mundo empresarial.

Ementa: Modelagem de dados. Normalização. Linguagem SQL. Mecanismos de consulta. Criação e alteração de tabelas. Manipulação e formatação de dados. Organização de resultados de pesquisa. Acesso ao servidor de bancos de dados. Contas de usuários. Segurança. Administração de bancos de dados. Manutenção. Integridade.

4. Curso de Construção de Web Sites

Este curso mostrará aos participantes como construir páginas HTML que forma a estrutura de um "site" na internet. A primeira parte do curso é voltada para a construção de páginas; a segunda parte, para a estruturação do conjunto de páginas que formação o "site", incluindo elementos de programação. Explicará os conceitos elementares da web e mostrará como é que se implementa o conjunto de páginas que forma o "site" num servidor.

Ementa: Linguagem HTML. Apresentação dos principais navegadors disponíveis no mercado.

Construção de uma página HTML simples respeitando os padrões W3C. Recursos de formatação de texto. Recursos de listas, multimídia e navegação. Tabelas e *Frame*s. Folha de Estilo. Elementos de Formulário. Linguagem Javascript. Interação do Javascript com os elementos HTML. Linguagem PHP. Conceitos de Transmissão de Site e critérios para avaliação de servidores.

1. Curso de Editoração Eletrônica

Voltado para a produção de documentos físicos (livros, jornais, revistas) e eletrônicos. Apresenta as ferramentas de produção de texto e as ferramentas de montagem de elementos gráficos numa página. O texto é tratado como elemento de composição gráfica, juntamente com a pintura digital, o desenho digital e outros elementos gráficos utilizados para promover a integração dos elementos gráficos.

O curso explora de maneira extensiva os conceitos relacionados à aparência do texto relativos aos tipos de impressão (fontes). Mostra diversos mecanismos de produção dos mais variados tipos de material impresso, de texto comum às fórmulas matemáticas. Finalmente, discute a metodologia de gerenciamento de documentos.

Ementa: Editor de textos. Formatadores de texto. Tipos e Fontes. Gerenciamento de projetos.

Publicações. Programas para editoração. Programas acessórios. Impressão. Desenvolvimento de um projeto.

2. Curso de Ilustração Digital

Desenvolvido sobre um único aplicativo de tratamento de imagens e pintura digital, o GIMP (GNU Image Manipulation Program – Programa de Manipulação de Imagens GNU).

Este curso ensina, passo a passo, como utilizar ferramentas do programa para produzir ilustrações de qualidade que podem ser utilizadas para qualquer finalidade. A pintura digital é diferente do desenho digital. O desenho se aplica a diagramas e gráficos, por exemplo. A pintura tem um escopo muito mais abrangente e é uma forma de criação mais livre, do ponto de vista formal. É basicamente a diferença que há entre o desenho artístico e o desenho técnico. É, portanto, um curso voltado para aqueles que têm interesses e vocações artísticas.

Ementa: A imagem digita*l.* Espaç*os* de core*s.* Digitalização de imagen*s.* Fotomontagem e colagem digital. Ferramentas de desenh*o.* Ferramentas de pintura. Finalização e saída.

3. Curso de Produção Fonográfica

Curso voltado para aqueles que têm interesse na produção musical. Explica, através de programas, como é que se capturam, modificam e agrupam os sons musicais para produzir arranjos musicais. É um curso introdutório com uma boa visão da totalidade dos procedimentos que levam à produção de um disco.

Ementa: O Fenômeno Sonoro. O Ambiente Sonoro. A Linguagem Musical. Pré-Produção. O Padrão MIDI. A Gravação. A Edição. Pós-processamento. Mixagem. Finalização.

4. Curso de Computação Gráfica

Curso introdutório de modelagem, renderização e animação de objetos tridimensionais.

Esse curso é a base para utilização de animações tridimensionais em filmes. Conduzido como um tutorial do programa BLENDER, apresenta a interface do programa e suas operações elementares. Destinado àqueles que têm ambições de produzir animações de alta qualidade para a educação ou para a mídia.

Ementa: Introdução à Computação Gráfica. Conceitos básicos 2D e 3D. Interface principal do programa Blender. Espaço de trabalho. Navegação em 3D. Modelagem em 3D. Primitivas básicas. Movimentação de objetos. Edição de objetos. Composição de cenas. Materiais e texturas. Aplicação de materiais. UV Mapping. Luzes e Câmeras. Iluminação de cena. Posicionamento e manipulação de câmera. Renderização still frame. Formatos

de saída. Animação básica. Movimentação de câmera e objetos. Renderização da animação. Formatos de saída.

5. Curso de Projeto Auxiliado Por Computador

Os programas de CAD (Computer Aided Design – Projeto Auxiliado por Computador) são utilizados para composição de desenhos técnicos. Diferentemente dos programas de pintura eletrônica (como o GIMP), fornecem ao usuário ferramentas para desenhar com precisão e anotar os desenhos de acordo com as normas técnicas. Além de ensinar ao usuário a utilizar um programa de CAD (QCad), o curso apresenta elementos básicos de desenho técnico e construções geométricas diversas visando preparar o participante para um aprimoramento em áreas típicas das engenharias e da arquitetura..Ementa: Informática aplicada ao desenho técnico. Conceitos básicos: construções geométricas, escalas, dimensionamento, projeções ortográficas e perspectivas. Sistemas de coordenadas cartesiano e polar. Novas entidades geométricas básicas: polígonos e círculos.

Operações geométricas básicas. Tipos de unidades de medida. Criação de um padrão de

formato. Organização de um desenho por níveis. Construções geométricas diversas. A teoria dos conjuntos aplicada ao desenho. Propriedades dos objetos. Edição do desenho.

Movimento, rotação, escalamento e deformação de objetos. Agrupamento de objetos em blocos.

6. Curso de Multimídia na Educação

O curso está dividido em três partes: a) utilização da multimídia no contexto educacional; b) autoria de apresentações multimídia; c) projetos de aprendizagem mediada por tecnologia. Este curso é o fundamento para a criação dos cursos de educação a distância.

Apresenta os elementos que compõem os sistemas de multimídia, as comunidades virtuais de aprendizagem, o planejamento e a preparação de uma apresentação e de uma lição de curso e, finalmente, a tecnologia de objetos de aprendizado multimídia.

Ementa: Introdução à Multimídia e seus componentes. Multimídia na Educação. Comunidades Virtuais de Aprendizagem. "Webquest": Desafios Investigativos baseados na Internet (Web).

Preparação de uma apresentação multimídia.

Sumário

MODULO 1: Introdução ao Webdesign

1.1 - Introdução ao HTML

O que é um arquivo HTML?

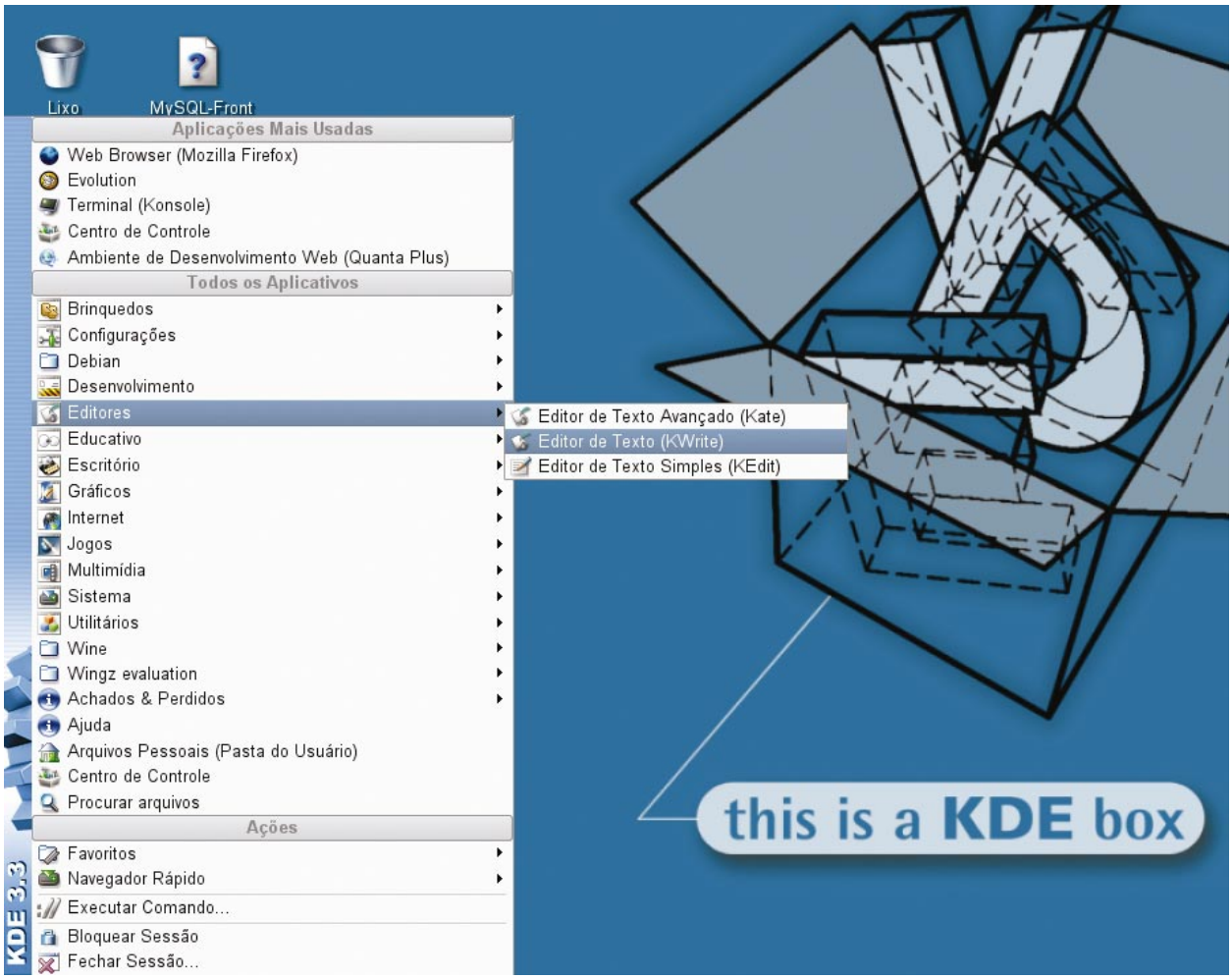
HTML significa Hyper Text Markup Language (Linguagem de marcação de texto).

Um arquivo HTML é um arquivo texto contendo pequenas marcações que serão interpretadas por um browser (como o mozilla firefox por exemplo). Estas marcações são chamadas de TAGs (ou etiquetas). As tags mostram ao browser como exibir a página.

Um arquivo HTML deve ter a extensão htm ou html e pode ser criado usando um editor de texto comum.

1.2 - Vamos tentar?

Primeiro vamos abrir nosso editor de texto. No KDE, vá ao menu, e em seguida nas opções Editors (ou Editores), e em Kwrite.



Qualquer outro editor de texto pode ser usado, como o Kate, etc, e e forma de executar o editor pode variar dependendo do programa e da sua distribuição do Linux.

Digite o seguinte texto:

```
<html>
<head>
<title>título da pagina</title>
</head>
<body>
Esta é minha primeira homepage.
</body>
</html>
```

Salve o arquivo como "minhapagina.htm".

Agora vamos visualizar nosso site. Inicie o browser (Mozilla Firefox) e selecione "Open" (ou "Abrir") no menu "File" (ou "Arquivo"). Uma caixa de dialogo irá aparecer. Selecione "Browse" (ou "Escolher arquivo") e localize o arquivo que acabou de criar - "minhapagina.htm" - selecione-o e clique em "Open" (ou "Abrir"). Agora você deve poder ver o caminho do arquivo na caixa de dialogo. Clique em OK, e o browser irá exibir a página.

Explicação do exemplo:

A primeira tag em seu documento HTML é <html>. Note que todas as tags estão delimitadas por "<" e ">". Esta tag indica ao seu browser que é o começo do documento HTML. A ultima tag é </html>. Esta tag indica que é o fim do documento HTML.

Os textos entre a tag <head> e a tag </head> são chamados de informações de cabeçalho. As informações de cabeçalho não são exibidas na janela do browser.

O texto entre as tags <title> e </title> é o título do seu documento. O título é exibido na Barra de título do browser.

É entre a tag <body> e </body> que estará todo o texto a ser exibido na janela do browser. E é nesta "seção" que utilizaremos todos os artifícios de formatação de texto, tabelas e formulários. Veremos mais sobre estes assuntos mais tarde.

Extensão HTM ou HTML?

Quando se salva um documento HTML, você pode usar tanto a extensão htm quanto html. Nós usamos htm nos nossos exemplos. Este é um habito gerado por softwares antigos que só aceitavam extensões com 3 letras.

Com softwares novos é perfeitamente seguro utilizar a extensão html.

Nota sobre os Editores de HTML:

Você pode facilmente editar arquivos HTML com programas do tipo o-que-você-vê-é-o-que-você-leva (WYSIWYG - what you see is what you get), sem precisar ao menos ler o código dos arquivos HTML.

Mas se você quer ser um Desenvolvedor Web habilidoso, é recomendável que comece fazendo você mesmo o arquivo HTML para aprender a dinâmica do HTML. Em tempo poderá utilizar tais editores, sabendo exatamente o que eles estão fazendo e até podendo fazer suas próprias melhorias e otimizações nos arquivos gerados por eles. Para maiores informações a respeito de editor WYSIWYG veja Apêndice"

Elementos HTML

Documentos HTML são arquivos texto feitos de elementos HTML.

Elementos HTML sao definidos usando tags HTML.

Perguntas efetuadas frequentemente (FAQ)

P: Depois que editei o arquivo HTML, eu não pude ver o resultado no meu browser. Porque?

R: Certifique-se que salvou o arquivo com a extensão correta e que está abrindo o arquivo com o mesmo nome no seu browser.

P: Eu tentei editar um arquivo HTML mas as mudanças não aparecem no browser. Porque?

A: O browser mantem uma copia da página para que ele não tenha que abrir a mesma página 2 vezes. Quando você altera o arquivo, seu browser não sabe que fez isso. Use o botão de refresh/reload (Atualizar) do browser para forçar ele a ler a página que editou.

P: Que browser eu devo usar?

A: YouVocê pode fazer todo seu treino em todos os browsers comuns, como Mozilla, Netscape, ou Konkeror. No entanto, alguns exemplos mais avançados de HTML podem exigir browsers em versões mais atualizadas.

1.3 - Tags HTML

- As tags HTML são usadas para marcar elementos HTML;
- As tags são delimitadas por **dois caracteres: < e >**;
- As tags normally **vêm em pares** como e ;
- A primeira tag do par é a **tag inicial**, e a segunda **tag final**;
- O texto entre a tag inicial e final é chamado de **conteúdo do elemento**;
- Tags HTML não são **"case sensitive"**, ou seja, é o mesmo de <B>.

Porquê usamos as tags em minúsculo?

Sabemos que as tags HTML não são "case-sensitive": <B> é o mesmo de .

Quando navegar na web, notará que a maioria dos tutoriais utiliza as tags em maiúsculo em seus exemplos. Nós sempre usamos as tags em minúsculo. Porque?

Se você quer se preparar para as próximas gerações de HTML, você deve começar a usar

tags em minúsculo. Algumas linguagens que surgiram do HTML (como a XHTML – ver apêndice sobre XHTML) exigem as tags em minúsculo, portanto, devemos seguir esta tendência se desejamos nos tornar profissionais desta área.

1.4 - Elementos HTML

Lembre o exemplo da seção 1.2, e adicionamos mais um elemento como a seguir:

```
<html>
<head>
<title>Título da Página</title>
</head>
<body>
Esta é minha segunda homepage.  <b>Este texto está em negrito.</
b>
</body>
</html>
```

Este é um elemento HTML:

```
<b>Este texto está em negrito</b>
```

- O elemento HTML começa com a **tag inicial**:
- O conteúdo do elemento é: este texto está em negrito
- O elemento HTML termina com a tag final:
- O propósito da tag é definir um elemento HTML que deve ser em negrito.

Este também é um elemento HTML:

```
<body>
Esta é minha segunda homepage e <b>Este texto está em negrito.</b>
</body>
```

Este elemento começa com a tag <body>, e termina com a tag </body>.

O propósito da tag <body> é definir o elemento HTML que contém o corpo do documento HTML.

1.5 - Atributos das Tags

Tags pode possuir atributos. Atributos podem dar informações adicionais sobre os elementos da sua página.

Esta tag define o elemento body (corpo) do seu documento: <body>. Com a adição do atributo bgcolor, você pode mostrar ao browser que a cor de fundo (background color) da sua página deve ser vermelha, deste modo: <body bgcolor="red">.

Esta tag define uma tabela: <table>. Com a adição do atributo border, você pode dizer

ao browser que a tabela não deve possuir bordas: <table border="0">

Atributos sempre vêm em pares nome/valor da seguinte forma: name="value".

Atributos são sempre adicionados na **tag inicial** do elemento html.

Estilo das aspas, "red" ou 'red'?

Valores dos atributos devem vir sempre entre aspas. Aspas dupas são mais comuns, mas aspas simples também são permitidas.

Em algumas situações, como quando o próprio valor do atributo contém aspas, é necessário usar aspas simples:

name='João "Webdesigner" da Silva'

MODULO 2: FORMATAÇÃO DE TEXTOS

2.1 - TAGS HTML BÁSICAS

As tags básicas de HTML de presença obrigatórias nas páginas são:

- <HTML> - marcação de entrada do documento
- </HTML> - marcação de saída do documento
- <HEAD> - marcação de entrada do cabeçalho
- </HEAD> - marcação de saída do cabeçalho
- <BODY> - marcação de entrada do corpo
- </BODY> - marcação de saída do corpo

– Apenas o que está escrito entre as marcações body aparecerá na tela maior do seu browser.

– Todas as páginas em HTML são identificadas pela tag <HTML>, que terá obrigatoriamente que estar no início do código. Obviamente, qualquer página irá terminar com </HTML>.

2.1.1 - Cabeçalho

Dentro do cabeçalho (tag "<head>") podemos encontrar:

- <title>: Define o título da página, que é exibido na barra de título dos browsers.
- <style>: Define formatação em CSS
- <script>: Define programação de certas funções em página com scripts, e pode colocar funções de JavaScript.
- <meta>: Define propriedades da página, como codificação de caracteres, descrição da página, autor, etc.

2.1.2 - Corpo

Dentro do corpo (tag "*<body>*") podemos encontrar outras várias tags, como por exemplo:

- <h1>, <h2>,... <h6>: cabeçalhos e títulos no documento em diversos tamanhos.
- <p>: novo parágrafo.
-
: quebra de linha.
- <font>: forma um texto (fonte, cor e tamanho) de um trecho do texto.
- <b>, <i>, <u> e <s>: negrito, itálico, sublinhado e riscado, respectivamente.
- <img>: imagem.
- <a>: hiperlink para uma página, ou para um endereço de E-mail.

O melhor modo de aprender html é trabalhando. Temos exemplos da utilização de algumas destas tags acima. As demais serão vistas ao decorrer do curso.

2.2 - Tente você mesmo – Exemplos

Exemplo 1) Este exemplo é um documento muito simples contendo o mínimo de tags necessárias para se fazer um documento HTML. Ele demonstra como o texto dentro da tag body é exibido no browser.

Código:

```
<html>
<body>
O conteúdo do elemento body é exibido no seu browser.
</body>
</html>
```

Exemplo 2) Este exemplo demonstra como o texto dentro dos elemetos de parágrafo são exibidos no seu browser.

Código:

```
<html>
<body>
<p>Isto é um parágrafo.</p>
<p>Isto é um parágrafo.</p>
<p>Isto é um parágrafo.</p>
<p>Elementos de parágrafo são definidos pela tag p.</p>
</body>
</html>
```

(Você encontrará mais exemplos no final deste módulo.)

2.3 - Subtítulos (Headings)

Subtítulos ou headings são definidos com as tags de <h1> à <h6>. <h1> define o maior Subtítulo e <h6> define o menor Subtítulo.

```
<h1>Isto é um Subtítulo</h1>
<h2>Isto é um Subtítulo</h2>
<h3>Isto é um Subtítulo</h3>
<h4>Isto é um Subtítulo</h4>
<h5>Isto é um Subtítulo</h5>
<h6>Isto é um Subtítulo</h6>
```

HTML adiciona automaticamente uma linha em branco imediatamente antes do título.

2.4 - Parágrafos

Parágrafos são definidos pela tag .

```
<p>Isto é um parágrafo</p>
<p>Este é outro parágrafo</p>
```

HTML automaticamente adiciona uma linha em branco antes e depois de um parágrafo.

2.5 - Quebra de Linha

A tag
 é usada quando você quer terminar uma linha mas não quer começar um novo parágrafo. Esta tag força uma quebra de linha onde você a colocar.

```
<p>Este <br> é um para-<br>grafo com<br>quebra de<br>linha.</p>
A tag <br> é uma "tag vazia", ou seja, não tem tag final ou de fechamento.
```

2.6 - Comentários em HTML

A tag de comentário é usada para inserir um comentário no código fonte do HTML. Um comentário será ignorado pelo browser. Você pode usar comentário para explicar seu código, o que pode ajudar quando você editar o código no futuro.

```
<!-- Isto é um comentário -->
```

Note que você precisa da exclamação na abertura mas não no fechamento da tag.

2.7 - Notas Básicas – Dicas Úteis

Quando escrever um texto em HTML, você nunca terá certeza de como o texto será exibido em outro browser. Algumas pessoas possuem telas grandes e outras pequenas. O texto será reformatado toda vez que o usuário alterar o tamanho da sua janela. Nunca tente formatar o texto em um editor adicionando linhas vazias e espaços ao texto.

HTML irá apagar os espaços extras no seu texto. Qualquer número de espaços contam como um. No HTML uma linha nova conta como um espaço.

Usando Parágrafos vazios para inserir linhas vazia é um hábito ruim. Use a tag
. (Mas não use
 para criar listas. Espere até você aprender a usar listas em HTML.)

Você já deve ter achado que Parágrafos podem ser escritos sem usar a tag de fechamento . Não confie nisso. A próxima versão do HTML não permite que você "pule" nenhuma tag de fechamento, e alguns browsers já notificam como erro ao usuário.

O HTML automaticamente adiciona linhas em branco antes e/ou depois de alguns elementos, como antes e depois do parágrafo e antes do Subtítulo.

Podemos usar a tag chamada de régua horizontal(tag <hr>), para separar seções em nossa página.

2.8 - Formatação do texto:

Agora que sabemos fazer Subtítulos, parágrafos e quebras de linha, podemos começar a ver como se formata um texto.

Vamos tentar?

Abra novamente o editor e digite o texto a seguir:

```
<html>
<body>
<b>Este texto está em negrito</b>
<br>
<strong>Este texto usa a tag strong</strong>
<br>
<i>Este texto está em itálico</i>
<br>
<em>Este texto está enfatizado</em>
<br>
<u>Este texto está em sublinhado</u>
<br>
<big>Este texto é um pouco maior</big>
<br>
<small>Este texto é um pouco menor</small>
<br>
O próximo texto está
<sub>subescrito</sub>
<br>
O próximo texto está
<sup>sobrescrito</sup>
</body>
</html>
```

Explicação do exemplo:

Como você já deve ter notado o texto pode ser colocado em negrito com as tags e . Do mesmo modo temos uma formatação semelhante chamada strong (tags e).

As tags big (<big> e </big>) e small (<small> e </small>) aumentam ou diminuem o tamanho do texto em uma pequena fração.

Para fazer um texto em itálico podemos usar as tags e , ou tags semelhante como e . Para fazê-lo ficar sublinhado usamos <u> e </u>.

Por fim temos as tags para deixar o texto sobrescrito (^e) e subscrito (_e).

2.9 - Modificando a fonte do texto:

A fonte também pode ser alterada, é claro. Para isso usamos a tag <font> e alguns atributos.

Os atributos principais desta tag são:

size: define o tamanho da fonte;

color: define a cor da fonte;

face: define a fonte a ser usada; (ex: face="Times")

2.10 - Exercícios:

Copie os textos dos exemplos a seguir e visualize em um browser.

Parágrafos:

Este exemplo demonstra alguns dos comportamentos dos elementos de parágrafo.

```
<html>
<body>
<p>
Este parágrafo
contém muitas linhas
no código fonte,
mas o browser
as ignora.
</p>

<p>
Já    este
contém      muitos    espaços
no      código       fonte,
mas     o     browser
também    os   ignora.
</p>

<p>
O número de linhas em um parágrafo depende do tamanho da sua
janela do browser. Se você modificar o tamanho da janela do
browser, o número de linhas neste parágrafo irá mudar..
</p>

</body>
</html>
```

Quebras de linha:

Este exemplo demonstra o uso da quebra de linha em um documento HTML.

```
<html>
<body>

<p>
Para fazer<br>quebras<br>de linha<br>em um<br>parágrafo,<br>use
a tag br.
</p>

</body>
</html>
```

Formatação de texto:

Este exemplo demonstra alguns problemas com a formatação do HTML.

```
<html>
<body>

<p>
   Cai cai balão,
    Cai cai balão,
   Na rua do sabão,

   Não cai não,
     Não cai não,
       Não cai não,

Cai aqui na minha mão.
</p>

<p>Note que seu browser simplesmente ignora sua formatação!</p>

</body>
</html>
```

Subtítulos:

Este exemplo demonstra as tags que mostram títulos em um documento HTML.

```
<html>
<body>

<h1>Este é um subtítulo 1</h1>
<h2>Este é um subtítulo 2</h2>
<h3>Este é um subtítulo 3</h3>
```

```
<h4>Este é um subtítulo 4</h4>
<h5>Este é um subtítulo 5</h5>
<h6>Este é um subtítulo 6</h6>
<p>Use tags de Subtítulos somente para Subtítulos. Não use elas
para fazer alguma coisa maior. Use outras tags pra isso.</p>
</body>
</html>
```

Título centralizado:

Este exemplo demonstra um título com alinhamento central.

```
<html>
<body>

<h1 align="center">This is heading 1</h1>

<p>O Subtítulo acima é alinhado ao centro da página.O Subtítulo
acima é alinhado ao centro da página. O Subtítulo acima é ali-
nhado ao centro da página. O Subtítulo acima é alinhado ao
centro da página. </p>

</body>
</html>
```

Régua horizontal:

Este exemplo demonstra como inserir uma régua horizontal.

```
<html>
<body>
<p>A tag hr define uma régua ou linha horizontal:</p>
<hr>
<p>Isto é um parágrafo</p>
<hr>
<p>Isto é um parágrafo</p>
<hr>
<p>Isto é um parágrafo</p>
</body>
</html>
```

Comentários ocultos:

Este exemplo demonstra como inserir um comentário no documento.

```
<html>
<body>
<!--Este comentário não é exibido-->
<p>Este é um parágrafo comum</p>
</body>
</html>
```

Cor de fundo:

Este exemplo demonstra como inserir uma cor de fundo no documento.

```
<html>
<body bgcolor="yellow">
<h2>Olha: Fundo Colorido!</h2>
</body>
</html>
```

2.11 - REVISÃO: TAGS HTML BÁSICAS

Tag	Descrição
<html>	Define um documento HTML
<body>	Define o corpo do documento
<h1> a <h6>	Define título nos tamanhos 1 a 6
<p>	Define um parágrafo
** **	Insere uma única quebra de linha
<hr>	Define uma linha horizontal
<!-->	Define um comentário

Módulo 3: Tags HTML Avançadas

3.1 - Exemplos

Listas não-numeradas

Este exemplo demonstra uma lista não-numerada.

```
<html>
<body>
<h4>Uma lista não-numerada:</h4>
<ul>
  <li>Café</li>
  <li>Chá</li>
  <li>Leite</li>
</ul>
</body>
</html>
```

Lista Numerada

Este exemplo demonstra como construir uma lista numerada.

```
<html>
<body>

<h4>Uma lista numerada:</h4>
<ol>
  <li>Café</li>
  <li>Chá</li>
  <li>Leite</li>
</ol>

</body>
</html>
```

3.2 - Listas

Listas não-numeradas

Uma lista não-numerada é uma lista de itens. Os itens da lista são marcados com "bullets" (normalmente pequenos circulos).

Uma lista não-numerada começa com a tag <ul> (de "**u**nordered **l**ist"). Cada item da lista começa com a tag <li> (de "list item").

```
<ul>
<li>Café</li>
<li>Leite</li>
</ul>
```

Abaixo mostramos como aparecerá no browser:

- Café
- Leite

Dentro de um item da lista você pode colocar Parágrafos, quebras de linha, imagens, links, outras listas, etc.

Listas Numeradas

Uma lista numerada é também uma lista de itens. Os itens da lista são marcados com números ou letras.

Uma lista numerada começa com a tag <ol> (de "**o**rdered **l**ist"). Cada item da lista começa com a tag <li> (de "list item").

```
<ol>
<li>Café</li>
<li>Leite</li>
</ol>
```

Abaixo mostramos como aparecerá no browser:

1. Café
2. Leite

Dentro de um item da lista, podemos colocar Parágrafos, quebras de linha, imagens, links, outras listas, etc.

Listas de Definição

Uma lista de definição não é uma lista de itens. É uma lista de termos e as explicações deles.

Uma lista de definição começa com a tag <dl> (de "**d**efinition **l**ist"). Cada termo da lista de definição começa com a tag <dt> (de "**d**efinition **t**erm"). Cada destrição de um termo da lista começa com a tag <dd>.

```
<dl>
<dt>Café</dt>
<dd>Bebida quente e negra</dd>
<dt>Leite</dt>
<dd>Bebida fria e branca</dd>
</dl>
```

Abaixo mostramos como aparecerá no browser:

Café

Bebida quente e negra

Leite

Bebida fria e branca

Dentro de uma descrição de um termo da lista (tag <dd>) você pode colocar parágrafos, quebra de linha, imagens, links, outras listas, etc.

Mais Exemplos

Tipos diferentes de listas numeradas

Este exemplo demonstra tipos diferentes de listas numeradas.

```
<html>
<body>
<h4>Lista Numerada:</h4>
<ol>
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ol>
<h4>Lista enumaradas por letras (maiusculas):</h4>
<ol type="A">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ol>
<h4>Lista enumaradas por letras (minusculas):</h4>
<ol type="a">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ol>
<h4>Lista enumaradas por números romanos (maiusculas):</h4>
<ol type="I">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ol>
<h4>Lista enumaradas por números romanos (minusculas):</h4>
<ol type="i">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ol>
</body>
</html>
```

Tipos diferentes de listas não-numeradas

Este exemplo demonstra tipos diferentes de listas não-numeradas.

```
<html>
<body>

<h4>Lista marcada por discos:</h4>
<ul type="disc">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ul>

<h4>Lista marcada por circulos:</h4>
<ul type="circle">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ul>

<h4>Lista marcada por quadrados:</h4>
<ul type="square">
 <li>Maça</li>
 <li>Bananas</li>
 <li>Limões</li>
 <li>Laranjas</li>
</ul>

</body>
</html>
```

Listas aninhadas (listas dentro de listas)

Este exemplo demonstra como criar listas aninhadas.

```
<html>
<body>

<h4>Uma lista aninhada:</h4>
<ul>
  <li>Café</li>
  <li>Chá
    <ul>
    <li>Chá preto</li>
    <li>Chá verde</li>
    </ul>
  </li>
  <li>Leite</li>
</ul>

</body>
</html>
```

Listas Aninhadas

Este exemplo demonstra listas aninhadas um pouco mais complexas.

```
<html>
<body>

<h4>Uma lista aninhada:</h4>
<ul>
  <li>Café</li>
  <li>Chá
    <ul>
    <li>Chá preto</li>
    <li>Chá verde
      <ul>
      <li>Chinês</li>
      <li>Africano</li>
      </ul>
    </li>
    </ul>
  </li>
  <li>Leite</li>
</ul>

</body>
</html>
```

Lista de definições

Este exemplo demonstra uma lista de definições.

```
<html>
<body>
<h4>Uma Lista de Definição:</h4>
<dl>
  <dt>Café</dt>
  <dd>Bebida quente e negra</dd>
  <dt>Leite</dt>
  <dd>Bebida fria e branca</dd>
</dl>
</body>
</html>
```

3.3 - Referências em HTML (Links e Hyperlinks)

O HTML usa um hyperlink para referenciar outro documento na web. Por convenção chamamos de link, uma referência a um documento no mesmo site, e de hyperlink, um documento de outro site.

Exemplos

Criando hyperlinks

Este exemplo demonstra como criar referências (links) em um documento HTML.

```
<html>
<body>
<p>
<a href="paginaAnterior.htm">Este Texto</a> e um link para uma
pagina no meu Web Site.
</p>

<p>
<a href="http://www.google.com.br">Este Texto</a> e um link para
uma pagina na Internet.
</p>

</body>
</html>
```

Uma imagem como um link

Este exemplo demonstra como usar uma imagem como um link.

```
<html>
<body>
<p>
Você também pode usar uma imagem como um link:
<a href="proximapagina.htm">
<img border="0" src="botao.gif" width="65" height="38">
</a>
</p>
</body>
</html>
```

A tag de âncora e o atributo Href

O HTML usa a tag <a> (âncora) para criar um link para outro documento.

A âncora pode "apontar" para qualquer recurso na web:: uma página HTML, uma imagem, um arquivo de som, um filme, etc.

A sintaxe para se criar uma âncora é a seguinte:

```
<a href="url">Texto a ser exibido</a>
```

A tag <a> é usada para criar uma âncora para referênciar um documento, que por sua vez é indicado (ou endereçado) pelo atributo Href. As palavras entre as tags de <a> e serão exibidas como um link (ou hyperlink).

Esta âncora define um link para meusite:

```
<a href="http://www.meusite.com.br/">Visite o meu site!</a>
```

A linha acima será exibida no browser da seguinte forma:

Visite o meu site!

O atributo Target

Com o atributo target, você pode indicar onde o documento será aberto.

A linha a seguir irá abrir o link em uma nova janela do browser, e é o procedimento padrão para se abrir páginas de outro site (hyperlink), pois evita que o usuário deixe o seu site para navegar em outro:

```
<a href="http://www.outrosite.com.br/"
target="_blank">Visite o meu outro site!</a>
```

A tag de âncora e o atributo Name

O atributo name é usado para criar uma âncora nomeada. Quando usamos este tipo de âncora, criamos links que nos transferem diretamente para uma seção específica da página, evitando que o usuário use a barra de rolagem para encontrar o que procura.

Abaixo está a sintaxe de uma âncora nomeada:

```
<a name="label">Texto a ser exibido</a>
```

O atributo name é usado para criar uma âncora nomeada. O nome da âncora pode ser qualquer texto que decidir usar.

A linha abaixo define uma âncora nomeada:

```
<a name="dicas">Seção de Dicas Uteis</a>
```

Você irá perceber que a âncora nomeada não é exibida de forma especial.

Para referenciar diretamente para a âncora, adicione um "#" e o nome da âncora como no exemplo abaixo:

```
<a href="http://www.meusite.com.br/html_tutorial.html#dicas">
```

Notas Básicas

Sempre adicione uma barra (“/”) para referências de sub-pastas. Se seu link é: href="http://www.meusite.com.br/html", você irá gerar duas requisições de HTTP para o servidor, porque o servidor irá adicionar a barra ao endereço e criar um requerimento como este: href="http://www.meusite.com.br/html/".

Âncoras nomeadas são frequentemente usadas para criar indices no inicio de grandes documentos. A cada capítudo do documento é dado uma âncora nomeada, e links para cada uma delas são colocados no inicio do documento.

Se um browser não encontrar uma âncora especificada por um link, ele direciona ao inicio do documento, e nenhum erro é gerado.

Mais Exemplos

Abrir um link em uma nova janela do browser

Este exemplo demonstra como abrir um link em uma nova janela do browser, para que o visitante não tenha que sair do seu site.

```
<html>
<body>

<a href="proximapagina.htm" target="_blank">Próxima Pagina</a>

<p>
Se você colocar o atributo de um link como "_blank",
o link abrirá em uma nova janela do browser.
</p>

</body>
</html>
```

Referenciar um local na mesma página

Este exemplo demonstra referenciar um local em outra parte do documento.

```
<html>
<body>

<p>
<a href="#C4">Veja também o Capítulo 4.</a>
</p>

<h2>Capítulo 1</h2>
<p>Este é o capitulo 1</p>
<h2>Capítulo 2</h2>
<p>Este é o capitulo 2</p>
<h2>Capítulo 3</h2>
<p>Este é o capitulo 3</p>
<a name="C4"><h2>Capítulo 4</h2></a>
<p>Este é o capitulo 4</p>
<h2>Capítulo 5</h2>
<p>Este é o capitulo 5</p>
</body>
</html>
```

Criando um link para um e-mail (mailto)

Este exemplo demonstra como criar um link para um e-mail (só funcionará se tem um email configurado).

```
<html>
<body>
<p>
Este é um link para um email:
<a href="mailto:fulano@meusite.com.br?subject=Olá%20denovo">
Enviar e-mail</a>
</p>
<p>
<b>Nota:</b> Espaços entre as palavras devem ser substituidos
por %20 para <b>garantir</b> que o browser exiba o texto corre-
tamente.
</p>
</body>
</html>
Criando um link para um e-mail 2
Este exemplo demonstra como criar um link para um e-mail um
pouco mais complicado.
<html>
<body>
<p>
This is another mailto link:
<a
href="mailto:fulano@meusite.com.br?cc=ciclano@meusite.com.br&bcc=
beltrano@meusite.com.br&subject=Curso%20de%20criacao%de%websites&body=
Você%20esta%20convidado!">Enviar e-mail</a>
</p>
<p>
<b>Nota:</b> Espaços entre as palavras devem ser substituidos
por %20 para <b>garantir</b> que o browser exiba o texto corre-
tamente.
</p>
</body>
</html>
```

3.4 - As Imagens no HTML

Com o HTML você pode exibir imagens em um documento.

Exemplos

Inserir imagens

Este exemplo demonstra como inserir imagens em sua página.

```
<html>
<body>
<p>
Uma imagem:
<img src="constr4.gif"
width="144" height="50">
</p>
<p>
Uma imagem animada:
<img src="hackanm.gif"
width="48" height="48">
</p>
<p>
Note que a sintaxe de inserção da imagem animada é igual a da
imagem estática. A animação depende apenas da imagem inserida.
</p>
</body>
</html>
```

Inserir imagens de locais diferentes

Este exemplo demonstra como inserir imagens de outras pastar ou outros servidores em sua página.

```
<html>
<body>
<p>
Imagem de outra pasta (link):
<img src="/imagens/netscape.gif"
width="33" height="32">
</p>
<p>
Uma imagem de outro site (hyperlink):
<img src="http://www.outrosite.com.br/imagens/mozilla.gif"
width="73" height="68">
</p>
</body>
</html>
```

A tag img e o atributo Src

Em HTML, imagens são definidas com a tag <img>.

A tag <img> é vazia, o que significa que ela contém somente atributos e não possui uma tag de fechamento.

Para exibir uma imagem na página, você precisa usar o atributo src. Src significa "source" ou origem. O valor do atributo src é o URL (endereço) da imagem que você deseja mostrar em sua página.

A sintaxe para definir uma imagem é:

```
<img src="url">
```

O URL indica a localização onde a imagem é armazenada. Uma imagem nomeada "boat.gif" localizada na pasta "imagens" em "www.meusite.com.br" tem o URL: "http://www.meusite.com.br/imagens/boat.gif".

O browser coloca as imagens onde a tag aparece no documento. Se você coloca a tag entre dois parágrafos, o browser mostra o primeiro parágrafo, depois a imagem, e em seguida o segundo parágrafo.

O Atributo Alt

O atributo alt é usado para definir um "texto alternativo" para a imagem. Este valor pode ser qualquer texto definido pelo autor da página:

```
<img src="boat.gif" alt="Barco">
```

O atributo "alt" conta ao leitor o que ele ou ela está perdendo quando o browser não exibe imagens (ou não pôde carrega-las). O browser irá então mostrar o texto alternativo no lugar da imagem. É uma boa prática incluir o atributo "alt" para cada imagem na página, para melhorar a exibição e utilidade da página para pessoas usando browsers que só exibem textos (como os de alguns telefones celulares).

Notas Básicas

Se um arquivo HTML contém dez imagens – onze arquivos são necessários para mostrar as páginas corretamente. Carregando imagens leva tempo, portanto, o melhor conselho é evitar usar muitas imagens diferentes na mesma página.

Mais Exemplos

Background image – Imagem de Fundo

Este exemplo demonstra como adicionar uma imagem de fundo em uma página HTML.

```
<html>
<body background="fundo.jpg">
<h3>Olha: Uma imagem de fundo!</h3>
<p>Ambos os arquivos gif e jpg podem ser usados como imagens de
fundo no HTML.</p>
<p>Se a imagem de fundo é menor que a página, ela se repetirá.</
p>
</body>
</html>
```

Alinhando Imagens

Este exemplo demonstra como alinhar imagens dentro do texto.

```
<html>
<body>
<p>
Uma imagem
<img src="hackanm.gif"
```

```
align="bottom" width="48" height="48">
dentro do texto!
</p>
<p>
Uma imagem
<img src ="hackanm.gif"
align="middle" width="48" height="48">
dentro do texto!
</p>
<p>
Uma imagem
<img src ="hackanm.gif"
align="top" width="48" height="48">
dentro do texto!
</p>
<p>Note que o alinhamento “bottom” é o alinhamento padrão.</p>
<p>
Uma imagem
<img src ="hackanm.gif"
width="48" height="48">
dentro do texto!
</p>
<p>
<img src ="hackanm.gif"
width="48" height="48">
Uma imagem depois do texto!
</p>
<p>
Uma imagem antes do texto!
<img src ="hackanm.gif"
width="48" height="48">
</p>
</body>
</html>
```

Posicionando a imagem no parágrafo

Este exemplo demonstra como deixar a imagem posicionada à esquerda e à direita do parágrafo.

```
<html>
<body>
<p>
<img src ="hackanm.gif"
align ="left" width="48" height="48">
Um parágrafo com uma imagem. O atributo de alinhamento (align)
está como "left". A imagem irá se posicionar à esquerda do
texto.
</p>

<p>
<img src ="hackanm.gif"
```

```
align ="right" width="48" height="48">
Um parágrafo com uma imagem. O atributo de alinhamento (align)
está como "right". A imagem irá se posicionar à direita do
texto.
</p>
</body>
</html>
```

Ajustando a imagem para tamanhos diferentes

Este exemplo demonstra como mudar o tamanho das imagens.

```
<html>
<body>
<p>
<img src="hackanm.gif"
width="20" height="20">
</p>

<p>
<img src="hackanm.gif"
width="45" height="45">
</p>

<p>
<img src="hackanm.gif"
width="70" height="70">
</p>

<p>
Você pode fazer a imagem maior ou menor mudando os valores em
"height" (altura) e em "width" (largura) na tag img.
</p>
</body>
</html>
```

Exibindo um texto alternativo para uma imagem

Este exemplo demonstra como exibir um texto alternativo para uma imagem.

```
<html>
<body>
<img src="paraesquerda.gif" alt="Vá para a esquerda!" width="32"
height="32">
<p>
Browsers que não exibem imagens não podem exibir imagens e irão
somente mostrar o texto alternativo. Aqui, o texto alternativo é
"Vá para a esquerda!".</p>
<p>
Note que se mantiver o ponteiro do mouse acima da imagem, a
maioria dos browsers irá exibir o texto alternativo.
</p>

</body>
</html>
```

MODULO 4: FRAMES

4.1 - FRAMES HTML

A palavra “frame”, significa moldura e é exatamente o que criamos com a utilização deste recurso. Dividimos a página em diversas molduras e cada uma destas molduras abre uma página html diferente, como se fosse uma outra janela do browser.

Exemplos

Conjunto de frames verticais

Este exemplo demonstra como dividir a página verticalmente em três documentos diferentes.

```
<html>
<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
  <frame src="frame_c.htm">
</frameset>
</html>
```

Conjunto de frames horizontais

Este exemplo demonstra como dividir a página horizontalmente em três documentos diferentes.

```
<html>

<frameset rows="25%,50%,25%">

  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
  <frame src="frame_c.htm">

</frameset>

</html>
```

Como usar a tag <noframes>

Este exemplo demonstra como usar a tag <noframes> para exibir uma página diferente em browsers que não suportam frames.

```
<html>
<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
  <frame src="frame_c.htm">
  <noframes>
     <body>
        Seu browser não suporta frames!
     </body>
```

```
</noframes>
</frameset>
</html>
```

Frames

Com os frames você pode exibir mais de um documento HTML na mesma janela do browser. Cada documento HTML é chamado de frame, e cada frame é independente um do outro.

As desvantagens de usar frames são:

- O desenvolvedor Web tem que cuidar de mais documentos HTML;
- É dificil de imprimir uma página completa.

A Tag Frameset

- A tag <frameset> define como dividir a janela em frames;
- Cada frameset define um conjunto de linhas (rows) e colunas(columns);
- Os valores de linhas/colunas indicam a quantidade de tela que cada linha/coluna irá ocupar.

A tag Frame

- A tag <frame> define que documento HTML será exibido em cada frame.

No exemplo abaixo temos um frameset com duas colunas. A primeira coluna irá ocupar 25% da largura da janela do browser. a segunda coluna irá ocupar os outros 75% da largura. O documento HTML "frame_a.htm" é colocado na primeira coluna e o documento "frame_b.htm" será colocado na segunda coluna:

```
<frameset cols="25%,75%">
   <frame src="frame_a.htm">
   <frame src="frame_b.htm">
</frameset>
```

Notas Básicas

Se um frame tem bordas visíveis, o usuário pode mudar o tamanho do frame arrastando a borda. Para previnir o usuário de fazê-lo, você pode adicionar o atributo noresize="noresize" à tag <frame>.

Adicione a tag <noframes> para browsers que não suportam frames.

Mais Exemplos

Frameset Misto

Este exemplo demonstra como fazer um frameset com três documentos, e como misturar eles em linhas e colunas.

```
<html>
<frameset rows="50%,50%">
<frame src="frame_a.htm">
<frameset cols="25%,75%">
<frame src="frame_b.htm">
<frame src="frame_c.htm">
</frameset>
</frameset>
</html>
```

Frameset não redimensionável

Este exemplo demonstra o atributo "noresize". Os frames não são redimensionáveis. Mova o mouse sobre a borda entre os frames e note que você não pode movê-la.

```
<html>
<frameset rows="50%,50%">
<frame noresize="noresize" src="frame_a.htm">
<frameset cols="25%,75%">
<frame noresize="noresize" src="frame_b.htm">
<frame noresize="noresize" src="frame_c.htm">
</frameset>
</frameset>
</html>
```

Frame "inline"

Este exemplo demonstra como criar um frame dentro do documento html, diretamente. Você pode usar em uma célula de uma tabela para que a página abra diretamente dentro desta, por exemplo.

```
<html>
<body>
<iframe src="index.htm"></iframe>
<p>Alguns browsers antigos não suportam iframes.</p>
<p>Se o browser não suporta iframes, o frame não será visível.</
p>
</body>
</html>
```

Nota: para abrir um link em um frame específico, devemos dar um nome ao frame usando o atributo "name" e no link (tag <a href...>) referenciamos o nome do frame de destino no atributo "target".

Em HTML, uma das mais importantes estruturas para o layout da sua página é a tabela.

4.2 - Exemplos

Tabelas

Este exemplo demonstra como criar uma tabela em um documento HTML.

```
<html>
<body>
<p>
Toda tabela começa com a tag <b>table</b>.
Toda linha da tabela começa com a tag <b>tr</b>.
Toda coluna da tabela começa com a tag <b>td</b>.
</p>
<h4>Uma linha e uma coluna:</h4>
<table border="1">
<tr>
  <td>100</td>
</tr>
</table>
<h4>Uma linha e tres colunas:</h4>
<table border="1">
<tr>
  <td>100</td>
  <td>200</td>
  <td>300</td>
</tr>
</table>
<h4>Duas linhas e três colunas:</h4>
<table border="1">
<tr>
  <td>100</td>
  <td>200</td>
  <td>300</td>
</tr>
<tr>
  <td>400</td>
  <td>500</td>
  <td>600</td>
</tr>
</table>
</body>
</html>
```

Bordas

Este exemplo demonstra diferentes bordas das tabelas.

```
<html>
<body>
<h4>Borda normal:</h4>
<table border="1">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
```

```
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>

</table>
<h4>Borda grossa:</h4>
<table border="8">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>
<h4>Borda muito grossa:</h4>
<table border="15">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>
</body>
</html>
```

Módulo 5: Tabelas

Tabelas são definidas pela tag <table>. Uma tabela é definida em linhas(com a tag <tr>), e cada linha é dividida em celulas (com a tag <td>). As letras td significam "table data" (dado da tabela), que é o conteúdo de uma celula de dados. Uma célula pode conter texto, imagens, listas, Parágrafos, formularios, réguas horizontais, tabelas, etc.

```
<table border="1">
<tr>
<td>linha 1, celula  1</td>
<td>linha 1, celula 2</td>
</tr>
<tr>
<td>linha  2, celula 1</td>
<td>linha  2, celula 2</td>
</tr>
</table>
```

Este código será exibido no browser como:

linha 1, celula 1	linha 1, celula 2
linha 2, celula 1	linha 2, celula 2

Tabelas e o atributo Border (Borda)

Se você não especificou um atributo de borda a tabela será exibida sem qualquer borda. Algumas vezes isto pode ser util, mas frequentemente queremos que a borda apareça.

Para exibir uma tabela com bordas, você irá ter que usar o atributo Border:

```
<table border="1">
<tr>
<td>linha 1, celula 1</td>
<td>linha 1, celula 2</td>
</tr>
</table>
```

Cabeçalho em uma Tabela

Cabeçalhos em uma tabela são definidos com a tag <th> (table heading).

```
<table border="1">
<tr>
<th>Cabeçalho</th>
<th>Outro Cabeçalho</th>
</tr>
<tr>
<td>linha 1, celula  1</td>
<td>linha 1, celula 2</td>
</tr>
<tr>
<td>linha  2, celula 1</td>
<td>linha  2, celula 2</td>
</tr>

</table>
```

Como aparecerá em um browser:

Cabeçalho	Outro Cabeçalho
linha 1, celula 1	linha 1, celula 2
linha 2, celula 1	linha 2, celula 2

Celulas vazias em uma tabela

Celulas da tabela sem conteúdo não são exibidas em muitos browsers.

```
<table border="1">
<tr>
<td>linha 1, celula 1</td>
<td> linha  1, celula 2</td>
</tr>
<tr>
<td>linha 2, celula 1</td>
<td></td>
</tr>
</table>
```

Este código será exibido por um browser da seguinte forma

linha 1, celula 1	linha 1, celula 2
linha 2, celula 1	

As bordas entorno da celula vazia não aparece em muitos browsers.

Para evitar isto, adicione um “non-breaking space”, cujo o código é “ ” á célula vazia que quer fazer as bordas visíveis, como no exemplo:

```
<table border="1">
<tr>
<td>linha 1, celula 1</td>
<td>linha 1, celula 2</td>
</tr>
<tr>
<td>linha 2, celula 1</td>
<td> </td>
</tr>
</table>
```

Mais Exemplos

Tabela sem borda

Este exemplo demonstra uma tabela sem bordas.

```
<html>
<body>
<h4>Esta tabela não possui bordas:</h4>
<table>
<tr>
  <td>100</td>
  <td>200</td>
  <td>300</td>
</tr>
<tr>
  <td>400</td>

 <td>500</td>
  <td>600</td>
</tr>
</table>
<h4>Esta também não:</h4>
<table border="0">
<tr>
  <td>100</td>
  <td>200</td>
  <td>300</td>
</tr>
<tr>
  <td>400</td>
  <td>500</td>
  <td>600</td>
</tr>
</table>
</body>
</html>
```

Título da tabela

Este exemplo demonstra como exibir um título na tabela.

```
<html>
<body>

<h4>
Esta tabela possui um título
e uma borda grossa:
</h4>

<table border="6">
<caption>Título</caption>
<tr>
  <td>100</td>
  <td>200</td>
  <td>300</td>
</tr>
<tr>
  <td>400</td>
  <td>500</td>
  <td>600</td>
</tr>
</table>

</body>
</html>
```

Tabelas com celulas maiores que o normal.

Este exemplo demonstra como definir celulas ocupando mais de uma linha e/ou coluna.

```
<html>
<body>
<h4>Celula com o tamanho de duas colunas:</h4>
<table border="1">
<tr>
  <th>Nome</th>
  <th colspan="2">Telefone</th>
</tr>
<tr>
  <td>João da Silva</td>
  <td>3123-9876</td>
  <td>9678-9876</td>
</tr>
</table>
<h4>Celula com tamanhho de duas linhas:</h4>
<table border="1">
<tr>
  <th>Nome:</th>
  <td>João da Silva</td>
</tr>
<tr>
  <th rowspan="2">Telefone:</th>
  <td>3123-9876</td>
</tr>
<tr>
  <td>9678-9876</td>
</tr>
</table>
</body>
</html>
```

Tag dentro de uma tabela

Este exemplo demonstra como mostrar elementos dentro de outros.

```
<html>
<body>
<table border="1">
<tr>
  <td>
   <p>Este é um parágrafo dentro da tabela</p>
   <p>Este é outro parágrafo</p>
  </td>
  <td>Tabela dentro de outra:
   <table border="1">
   <tr>
     <td>A</td>
     <td>B</td>
   </tr>
   <tr>
```

```
    <td>C</td>
       <td>D</td>
     </tr>
     </table>
    </td>
  </tr>
  <tr>
    <td>Lista dentro da celula
     <ul>
      <li>A</li>
      <li>B</li>
      <li>C</li>
     </ul>
    </td>
  </tr>
  </table>
  </body>
  </html>
```

Espaçamento na celula

Este exemplo demonstra como usar o "cellpadding" para criar uma margem dentro da tabela entre o conteúdo das celulas e a sua borda.

```
<html>
<body>
<h4>sem cellpadding:</h4>
<table border="1">
<tr>
  <td>Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>linha</td>
</tr>
</table>
<h4>Com cellpadding:</h4>
<table border="1" cellpadding="10">
<tr>
  <td>Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>linha</td>
</tr>
</table>
</body>
</html>
```

Espaçamento entre celulas

Este exemplo demonstra como usar o "cellspacing" para aumentar o espaço entre celulas.

```
<html>
<body>

<h4>Sem cellspacing:</h4>
<table border="1">
<tr>
  <td>Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>linha</td>
</tr>
</table>

<h4>Com cellspacing:</h4>
<table border="1" cellspacing="10">
<tr>
  <td>Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>linha</td>
</tr>
</table>

</body>
</html>
```

Adicionando cor e imagem de fundo da tabela

Este exemplo demonstra como adicionar uma cor/imagem de fundo à tabela.

```
<html>
<body>

<h4>Uma cor de fundo:</h4>
<table border="1" bgcolor="red">
<tr>
  <td>Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>linha</td>
</tr>
</table>
```

```
<h4>Uma imagem de fundo:</h4>
<table border="1" background="bgdesert.jpg">
<tr>
  <td>Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>linha</td>
</tr>
</table>

</body>
</html>
```

Adicionando Cor/Imagem ao fundo de uma celula.

Este exemplo demonstra como adicionar um fundo a uma celula da tabela.

```
<html>
<body>

<h4>Fundo das celulas:</h4>
<table border="1">
<tr>
  <td bgcolor="red">Primeira</td>
  <td>linha</td>
</tr>
<tr>
  <td background="bgdesert.jpg">Segunda</td>
  <td>linha</td>
</tr>
</table>

</body>
</html>
```

Alinhando o conteúdo de uma Celula

Este exemplo demonstra como usar o atributo "align" para alinhar os conteúdos das celulas, para criar uma tabela mais agradável.

```
<html>
<body>

<table width="400" border="1">
 <tr>
  <th align="left">Dinheiro gasto em...</th>
  <th align="right">Janeiro</th>
  <th align="right">Fevereiro</th>
 </tr>
 <tr>
```

```
<td align="center">Roupas</td>
 <td align="right">R$241.10</td>
 <td align="right">R$50.20</td>
</tr>
<tr>
 <td align="center">Comida</td>
 <td align="right">$730.40</td>
 <td align="right">$650.00</td>
</tr>
</table>

</body>
</html>
```

O ATRIBUTO FRAME

Este exemplo demonstra como usar o atributo "frame" para controlar bordas das tabelas.

```
<html>
<body>

<p>
Se nao ve enquadramentos ao redor das tabelas, seu browser não o
suporta.
</p>

<h4>Com Enquadramento="border":</h4>
<table frame="border">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="box":</h4>
<table frame="box">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="void":</h4>
```

```
<table frame="void">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="above":</h4>
<table frame="above">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="below":</h4>
<table frame="below">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="hsides":</h4>
<table frame="hsides">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="vsides":</h4>
<table frame="vsides">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
```

```
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="lhs":</h4>
<table frame="lhs">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

<h4>Com Enquadramento="rhs":</h4>
<table frame="rhs">
<tr>
  <td>Primeira</td>
  <td>Linha</td>
</tr>
<tr>
  <td>Segunda</td>
  <td>Linha</td>
</tr>
</table>

</body>
</html>
```

Módulo 6 - Folhas de Estilo

CSS significa Cascading Style Sheets (Folhas de estilo em cascata). Estilos definem como mostrar os elementos HTML e normalmente são armazenados em folhas de estilo.

Os estilos foram adicionados no HTML 4.0 para facilitar o design de páginas. Folhas de estilo externas à página podem economizar muito trabalho, já que se pode utilizar a mesma programação para estilizar diversas páginas do site.

Estas folhas de estilo externas, são guardadas nos arquivos CSS.

6.1 – Estilos como separação entre conteudo e layout

As tags HTML foram originalmente planejadas para definir o conteúdo de um documento. Elas estão encarregadas a nos dizer "isto é um cabeçalho", "isto é um parágrafo" ou "isto é uma tabela", usando tags como <h1>, <p>, <table>, e assim por diante. O layout do documento era pra ser uma função do browser, sem que fossem usadas nenhuma tag de formatação.

Como os brosers grandes como o Netscape, continuamente adicionavam novas tags e atributos (como a tag <font> e o atributo color) a especificação do HTML original, se tornou mais e mais dificil a criação de websites onde o conteudo do HTML estivesse claramente separado de sua apresentação ou layout.

Para resolver este problema, o World Wide Web Consortium (W3C) – consorcio responsável pela padronização do HTML - criou os estilos (STYLES) em complemento ao HTML 4.0.

Todos os browsers mais utilizados suportam o CSS.

6.2 – Como os estilos economizam trabalho?

As folhas de estilo, definem como os elementos HTML são mostrados. Os estilos são normalmente salvos em arquivos externos (com a extensão .css). Estes arquivos permitem que você mude a aparência e layout de todas as suas páginas apenas editando um arquivo CSS.

CSS é uma inovação no webdesign porque ele permite que desenvolvedores controlem o estilo de diversas páginas de uma só vez. Como um desenvolvedor Web você pode definir um estilo para cada elemento HTML e aplicar a quantas páginas lhe convier. E para fazer uma mudança global, mude o estilo e todos os elementos serão atualizados automaticamente.

6.3 – Sintaxe do CSS

Syntax

A sintaxe do CSS é feita de três partes: um seletor, uma propriedade e um valor:

```
seletor {propriedade: valor}
```

O seletor é normalmente o elemento/tag do HTML a definir, já a propriedade é o atributo que você deseja modificar e cada propriedade pode ter um valor. A propriedade e valor são separados por ":" e ficam ambos entre chaves:

```
body {color: black}
```

Se o valor contem multiplas palavras, são necessárias as aspas ao seu redor:

```
p {font-family: "sans serif"}
```

Nota: Se deseja especificar mais de uma propriedade, separe-as utilizando o ponto-e-vírgula. O exemplo a seguir mostra como definir um parágrafo centralizado com o texto em vermelho:

```
p {text-align:center;color:red}
```

Para deixar as definições de estilo mais amigaveis à nossa leitura, é costume descrever cada propriedade em uma linha, como no exemplo:

```
P
{
text-align: center;
color: black;
font-family: arial
}
```

Agrupando Seletores

Você pode agrupar seletores. Separe cada seletor com uma virgula. No exemplo abaixo, agrupamos todos os elementos de Subtítulos para que sejam exibidos em verde:

```
h1,h2,h3,h4,h5,h6
{
color: green
}
```

O seletor de classe

Com o seletor de classe você pode definir diferentes estilos para o mesmo tipo de elemento HTML. Digamos que você gosta de ter dois tipos de Parágrafos em seu documento: um justificado a direita e outro centralizado. Eis como podemos fazer estes estilos:

```
p.direita {text-align: right}
p.centro {text-align: center}
```

No seu documento HTML, você deve indicar qual tipo de parágrafo deseja usar:

```
<p class="direita">
Este é um parágrafo justificado a direita!
</p>
<p class="centro">
Este é um parágrafo centralizado!
</p>
```

Nota: Somente um atributo de classe pode ser definido por elemento. O exemplo a seguir está **incorreto**:

```
<p class="direita" class="centro">
Isto é um parágrafo!
</p>
```

Você pode omitir o nome da tag no seletor para definir um estilo que irá ser usado por todos os elementos que usarem a classe especificada. No exemplo, todos os elementos com class="centro" serão centralizados:

```
.centro {text-align: center}
```

No código abaixo, tanto o elemento h1 como o elemento p possuem class="centro". Isto significa que ambos utilizarão as regras no seletor ".centro":

```
<h1 class="centro">
Este é um Subtítulo centralizado!
</h1>
<p class="centro">
Este é um parágrafo centralizado!
</p>
```

NÃO inicie o nome de uma classe com um número! Não funcionará em browsers como o Mozilla/Firefox.

O Seletor id

Com o seletor id podemos definir o mesmo estilo para diferentes elementos HTML.
A regra de estilo abaixo será aplicada a qualquer elemento cujo atributo id seja "verde":

```
#verde {color: green}
```

A regra acima será aplicada tanto ao elemento h1 como ao elemento p:

```
<h1 id="verde">texto</h1>
<p id="verde">texto</p>
```

A regra abaixo será aplicada aos elementos de parágrafo P que possuam o atributo id com o valor "para1":

```
p#para1
{
text-align: center;
color: red
}
```

A regra não será aplicada ao elemento h1 abaixo:

```
<h1 id="para1">texto</h1>
```

NÃO inicie o valor de um ID com um número! Não funcionará em browsers como o Mozilla/Firefox.

Comentários no CSS

Podemos inserir comentários no CSS para explicar nosso código, o que pode ajudá-lo a editar seu código em uma data posterior. O comentário será ignorado pelo browser.

Um comentário no CSS começa com "/*", e termina com "*/", como no exemplo:

```
/* Este é um comentário  */
p
{
text-align: center;

/* Este é outro comentário */

color: black;
font-family: arial

}
```

6.4 – Associando um CSS aos seus documentos HTML

Como inserir uma folha de estilos

Quando o browser lê a folha de estilos, ele irá formatar o documento de acordo com ele. Existem três formas de inserir a folha de estilo:

1 - Folha de Estilos Externa

Uma folha de estilos externa é ideal quando os mesmos estilos serão aplicados a diversas páginas. Com a folha de estilos externa, você pode trocar o visual de um website inteiro apenas modificando um arquivo. Cada página precisa referenciar a página de estilos usando a tag <link>. A tag <link> deve ficar na seção head:

```
<head>

<link rel="stylesheet" type="text/css" href="estilos.css" />
</head>
```

O browser irá ler as definições do arquivo estilos.css, e formatar a página de acordo com ele.

Uma folha de estilo externa pode ser escrita em qualquer editor de texto. O arquivo não deve conter tags HTML. Seu arquivo deve usar a extensão .css. Abaixo um exemplo de um arquivo de estilos:

```
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
```

NÃO deixe espaços entre o valor da propriedade e as unidades!

Se usar "margin-left: 20 px" no lugar de "margin-left: 20px" não serão aplicadas as regras em browsers como Mozilla/Firefox ou Netscape.

2 - Folha de Estilos Interna

Uma folha de estilo interna deve ser usada quando um documento tem um estilo único. Podemos definir estilos internos na seção head usando a tag <style>, da seguinte forma:

```
<head>
<style type="text/css">
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
</style>
</head>
```

Nota: Os browsers normalmente ignoram tags desconhecidas. Isto significa que um browser antigo que não suporta estilos irá ignorar a tag <style>, mas o conteudo da tag será exibido na página. É possivel prevenir que um browser antigo eiba o conteudo escondendo-o dentro de um comentário HTML:

```
<head>
<style type="text/css">
<!--
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
-->
</style>
</head>
```

3 - Estilos "Inline"

Um estilo inline perde muitas das vantagens de se usar estilos, pelo fato de misturar o conteúdo com o layout. Use este método com cuidado, em casos em que um estilo somente aparece em uma única ocorrencia em um documento.

Para usar estilos inline usamos o atributo style na tag relevante. O atributo do estilo pode conter qualquer propriedade CSS:

```
<p style="color: sienna; margin-left: 20px">
Isto é um parágrafo!
</p>
```

Veja no Apendice A as diversas propriedades e valores que podem ser usados no CSS.

MODULO 7: Formulários HTML (Forms) e Entradas de Dados (Input)

Formulários HTML são usados para selecionar diferentes formas de entradas de dados pelo usuário.

7.1 - Exemplos

Campos de Texto

Este exemplo demonstra como criar campos de texto em uma página HTML. O usuário pode escrever textos em um campo de texto.

```
<html>
<body>

<form>
Nome:
<input type="text" name="nome">
<br>
Sobrenome:
<input type="text" name="sobrenome">
</form>
</body>
</html>
```

Campos de Senha (password)

Este exemplo demonstra como criar um campo de senha em uma página HTML.

```
<html>
<body>

<form>
Login:
<input type="text" name="login">
<br>
senha:
<input type="password" name="senha">
</form>
<p>
Note que quando você digita os caracteres em um campo de senha,
o browser mostra asteriscos ('*') no lugar dos caracteres.
</p>
</body>
</html>
```

(Veja mais exemplos no final deste módulo.)

7.2 - Formulários (Forms)

Um formulário é uma aread que pode conter elementos de formulário.

Elementos de formulário são elementos que permitem ao usuário entrar informações (como campos de texto, caixas de seleção "drop-down", botões "radio", caixas "checkboxes", etc.) em um formulário.

Um formulário é definido com a tag <form>.

```
<form>
  <input>
  <input>
</form>
```

7.3 - Entrada de Dados (Input)

A tag mais usada em um formulário é a tag <input>. O tipo de entrada é especificada pelo atributo "type". Os tipos de entrada mais comuns serão explicados abaixo.

Campos de Texto (ou Text Fields)

Campos de texto são usados quanto queremos que o usuário entre com letras ou numeros, etc. em um formulário.

```
<form>
Nome:
<input type="text" name="nome">
<br>
Sobrenome:
<input type="text" name="sobrenome">
</form>
```

Este código será exibido por um browser da seguinte forma:

Nome: [ ]

Sobrenome: [ ]

Note que o formulário em si não é exibido. Note também que na maioria dos browsers, a largura do campo de texto é de 20 caracteres por padrão.

7.4 - Botões "Radio"

Botões radio são usados quando se deseja que o usuário faça uma seleção em uma lista limitada de opções.

```
<form>
<input type="radio" name="genero" value="M"> Masculino
<br>
<input type="radio" name="genero" value="F"> Feminino
</form>
```

Este código será exibido por um browser da seguinte forma:

Masculino

Feminino

Note que apenas uma opção pode ser selecionada.

7.5 - Caixas Checkbox

As Checkbox são usadas quando se deseja que o usuário selecione uma ou mais opções em uma lista limitada de opções.

```
<form>
<input type="checkbox" name="bicicleta">
Eu tenho uma bicicleta
<br>
<input type="checkbox" name="carro">
Eu tenho um carro
</form>
```

Este código será exibido por um browser da seguinte forma:

- Eu tenho uma bicicleta
- Eu tenho um carro

7.6 - O atributo "Action" e o botão "Submit" de um formulário

Quando ousuário clica no botão "submit", o conteúdo do formulário é enviado para outro arquivo. O atributo "action" define o nome do arquivo para o qual os dados serão enviados. O arquivo especificado no atributo action geralmente faz alguma operação com os dados recebidos.

```
<form name="input" action="html_form_action.php"
method="get">
Login:
<input type="text" name="login">
<input type="submit" value="Submit">
</form>
```

Este código será exibido por um browser da seguinte forma:

Login: Submit

Os dados digitados na caixa de texto serão enviados para a página indicada no atributo action.

Estes dados serão inicializados às variaveis cujo nome são indicados pelo atributo "name" das respectivas tags Input. Veremos como acessar estas variáveis, atravéz do Javascript, no próximo capítulo. Da mesma forma estes dados podem ser acessados com o PHP, ASP etc. Veremos uma introdução ao PHP mais adiante no nosso curso.

7.7 - Mais Exemplos

Caixas Checkbox

Este exemplo demonstra como criar check-boxes em uma página HTML. A user can select or unselect a checkbox.

```
<html>
<body>

<form>
<input type="checkbox" name="bicicleta">Eu tenho uma bicicleta
<br>
<input type="checkbox" name="carro">Eu tenho um carro
</form>
</body>
</html>
```

Botão "radio"

Este exemplo demonstra como criar radio-buttons em uma página HTML.

```
<html>
<body>

<form>
<input type="radio" name="genero" checked="checked" value="M">
Masculino
<br>
<input type="radio" name="genero" value="F"> Feminino
</form>

<p>
Quando o usuário clica em um botão "radio", o botão é marcado,
e todos os outros desmarcados.
</p>

</body>
</html>
```

Caixa dropdown simples

Este exemplo demonstra como criar uma caixa drop-down em uma página HTML. Uma caixa drop-down é uma lista de seleção (select).

```
<html>
<body>

<form>
<select name="carros">
<option value="volvo">Volvo
<option value="bmw">BMW
<option value="fiat">Fiat
```

```
<option value="audi">Audi
</select>
</form>

</body>
</html>
```

Outra Caixa drop-down

Este exemplo demonstra como criar uma caixa drop-down box com um valor pré-selecionado.

```
<html>
<body>

<form>
<select name="cars">
<option value="volvo">Volvo
<option value="bmw">BMW
<option value="fiat" selected="selected">Fiat
<option value="audi">Audi
</select>
</form>

</body>
</html>
```

Area de Texto

Este exemplo demonstra como criar uma area de texto (um campo de texto com multiplas linhas). Um usuário pode escrever textos mais longos em uma área de texto.

```
<html>
<body>
<form>
<textarea rows="10" cols="30">
O gato brincava no jardim.
</textarea>
</form>
</body>
</html>
```

Botão

Este exemplo demonstra como criar um botão. Em um botão podemos definir nosso próprio texto.

```
<html>
<body>

<form>
```

```
<input type="button" value="Meu primeiro Botao">
</form>

</body>
</html>
```

Borda no formulário

Este exemplo demonstra como desenhar uma borda com um título ao redor dos seus dados.

```
<html>
<body>

<fieldset>
<legend>
Dados de Saude:
</legend>
<form>
Largura <input type="text" size="3">
Altura <input type="text" size="3">
</form>
</fieldset>

<p>

Se não foi exibida uma borda ao redor do formulário, seu browser
é muito antigo.

</p>

</body>
</html>
```

Exemplo de Formulário

Este exemplo demonstra como adicionar um formulário à uma página. Este formulário contém diversas formas de entrada de dados.

```
<html>
<body>

<fieldset>
<legend>
Novo Usuario:
</legend>
<form name="input" action="cadastrar.php" method="get">
Nome: <input type="text" name="nome"><br>
Sobrenome: <input type="text" name="sobrenome"><br>
Nome de usuario: <input type="text" name="login"><br>
e-mail: <input type="text" name="email"><br>
```

```
Genero:<br>
<input type="radio" name="genero" value="M"> Masculino<br>
<input type="radio" name="genero" value="F"> Feminino<br>
<input type="checkbox" name="newsletter">Gostaria de receber
emails do administrador<br>
Pequena descrição pessoal:
<textarea name="desc" rows="10" cols="30">
Insira a descrição aqui.
</textarea><br>
<input type="submit" value="Enviar">
</form>
</fieldset>
</body>
</html>
```

Tags de Formulário

Tag	Descrição
<form>	Define um formulário para inserção de dados
<input>	Define um campo de inserção de dados
<textarea>	Define uma area de texto
<label>	Define um rótulo
<fieldset>	Define um bloco de campos
<legend>	Define um título ou legenda para um bloco de campos
<select>	Define uma lista de seleção (caixa dropdown)
<optgroup>	Define um grupo de opções
<option>	Define uma opção em uma caixa dropdown
<button>	Define um botão

Modulo 8: Introdução ao JavaScript

JavaScript é uma linguagem usada em milhões de páginas da web para melhorar o design, validar formulários, detectar browser, criar cookies e muito mais.

O JavaScript é a mais popular linguagem de script na internet, e funciona em todos os maiores browsers, como Mozilla, Firefox, Netscape e Opera.

8.1 - O que é o JavaScript?

- JavaScript foi criado para adicionar interatividade às páginas HTML;
- JavaScript é uma linguagem de script (uma linguagem de script é uma linguagem de programação leve que é interpretada por programas como os browsers);
- O JavaScript consiste em linhas de código executáveis pelo computador;
- O JavaScript é normalmente inserido diretamente no código da página HTML;
- JavaScript é uma linguagem interpretada (scripts sem um compilação prévia, ou seja, sem tradução para linguagem de maquina);
- Todos podem usar o JavaScript sem comprar uma licensa;

Java e JavaScript são a mesma coisa?

NÃO!

Java and JavaScript são duas linguagems muito diferentes tanto em conceito como design!

Java (desenvolvido pela Sun Microsystems) é uma poderosa e muito mais complexa linguagem de programação – na mesma categoria das linguagens C e C++.

O que um JavaScript pode fazer?

- **JavaScript dá aos webdesigners uma ferramenta de programação –** Autores HTML não são normalmente programadores, mas o JavaScript é uma linguagem com a sintaxe muito simples! Quase todos podem colocar pequenos pedaços de código ("snippets") em suas páginas HTML;
- **JavaScript pode colocar texto dinâmico em uma página HTML –** Um comando JavaScript como este: document.write("<h1>" + nome + "</h1>") pode escrever um texto variavel em uma página HTML;
- **JavaScript pode reagir a eventos -** O JavaScript pode ser preparado para executar quanto alguma coisa acontece, como quando a página é carregada ou quando o usuário clica em um elemento HTML;
- **JavaScript pode ler e escrever elementos HTML -** O JavaScript pode ler e mudar o conteudo de um elemento HTML;
- **JavaScript pode ser usado para validar dados -** O JavaScript pode ser usado para

validar um formulário antes que este seja enviado ao servidor, economizando processamento para o servidor;

- **JavaScript pode detectar o browser do visitante -** O JavaScript pode ser usado para detectar o browser do usuário, e -dependendo do browser – carregar uma página específicamente feita para o browser do usuário;
- **JavaScript pode criar cookies -** O JavaScript pode ser usado para guardar dados e recuperar informações do computador do usuário.

A tag HTML <script> é usada para inserir um código JavaScript em um documento HTML.

8.2 - Exemplos

Como escrever na página:

```
<html>
<body>

<script type="text/javascript">
document.write("Este texto foi escrito pelo JavaScript!")
</script>

</body>
</html>
```

Como formatar o texto com tags HTML usando o Javascript:

```
<html>
<body>

<script type="text/javascript">
document.write("<h1>Este texto foi escrito e formatado pelo
<B>JavaScript</B>!</h1>")
</script>

</body>
</html>
```

Como colocar um código JavaScript em uma página HTML

```
<html>
<body>
<script type="text/javascript">
document.write("Este texto foi escrito pelo JavaScript!")
</script>
</body>
</html>
```

O código acima irá produzir esta saida no browser:

Este texto foi escrito pelo JavaScript!

8.3 - Entendendo o Exemplo

Para inserir o código JavaScript em um documento, usamos a tag <script> (também usamos o atributo "type" para definir a linguagem do script).

Então, as tags <script type="text/javascript"> e </script> contam ao browser onde começa e termina o código JavaScript:

```
<html>
<body>
<script type="text/javascript">
...
</script>
</body>
</html>
```

O comando **"document.write"** é o comando padrão de saida de texto para a página.

Inserindo o comando "document.write" entre as tags <script type="text/javascript"> e </script>, fará com que o broswer o reconheça como um comando JavaScript e executará o comando. Neste caso o browser irá escrever a frase "Este texto foi escrito pelo JavaScript!" na página:

```
<html>
<body>
<script type="text/javascript">
document.write("Este texto foi escrito pelo JavaScript!")
</script>
</body>
</html>
```

Nota: Se não colocarmos a tag <script>, o browser irá tratar o comando "document.write" como texto puro, e assim irá escrever a linha inteira à página.

Terminar linhas com ";"?

Pela tradição das linguagens de programação, como C++ e Java, cada comando deve ser seguido de ponto-e-vírgula.

Muitos programadores continuam com este hábito quando programando em JavaScript, mas em geral, o ponto-e-vírgula é opcional! No entanto, ponto-e-vírgula são necessários se você quer adicionar mais de um comando por linha.

Como fazer com os Browsers antigos?

Browsers que não suportam o JavaScript mostrarão o script como conteúdo da página. Para evitar que isso aconteça, podemos usar a tag de comentário HTML:

```
<script type="text/javascript">
<!--
document.write("Este texto foi escrito pelo JavaScript!")
//-->
</script>
```

As duas barras antes do fechamento da tag de comentário ("//") representam o simbolo de comentário do JavaScript. Isto evita que o JavaScript analise a linha.

8.4 - JavaScript, Onde colocar...

JavaScripts na seção "body" irão ser executados ENQUANTO a página carrega.

JavaScripts na seção "head" irão ser executados quando chamados.

8.5 - Exemplos

Cabeçalho

Scripts que contem funções podem ser colocados no cabeçalho do documento. Assim podemos garantir que o script é carregado antes da função ser chamada.

```
<html>
<head>
<script type="text/javascript">
function message()
{
alert("Esta caixa de alerta foi chamada pelo evento onload")
}
</script>
</head>

<body onload="message()">

</body>
</html>
```

Body section

Executa um script que foi colocado na seção "body".

```
<html>
<head>
</head>
```

```
<body>

<script type="text/javascript">
document.write("Esta mensagem será escrita quando a página for
carregada")
</script>

</body>
</html>
```

External script

Como acessar um script externo a página HTML.

```
<html>
<head>
</head>
<body>
<script src="xxx.js"></script>
<p>
Os scripts estão dentro do arquivo chamado "xxx.js".
</p>
</body>
</html>
```

Onde colocar o JavaScript

JavaScripts em uma página será executado imediatamente enquanto a página carrega no browser. Isto não é o que queremos sempre. Algumas vezes queremos executar o script quando a página carrega, outras vezes quando acontece o que chamamo de evento.

Scripts na seção “head”: Scripts para serem executados quando são chamados, ou quando um evento é acionado, são colocados na seção “head”. Quando colocamos um script na seção “head”, garantimos que estes são carregados antes que qualquer um os utilize.

```
<html>
<head>
<script type="text/javascript">
....
</script>
</head>
```

Scripts na seção “body”: Scripts para serem executados quando a página é carregada, são colocados na seção “body”. Quando se coloca um script na seção body ele é geralmente conteúdo da página.

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
....
</script>
</body>
```

Scripts tanto na seção “body” quanto na seção “head”: Você pode colocar um número ilimitado de scripts em seu documento, então você pode ter scripts nas duas seções.

```
<html>
<head>
<script type="text/javascript">
....
</script>
</head>
<body>
<script type="text/javascript">
....
</script>
</body>
```

Usando um JavaScript Externo

Algumas vezes você pode querer executar um script em diversas páginas, sem ter que escrever o mesmo script em cada uma delas.

Para simplificar isto, você pode escrever JavaScript em um arquivo externo. Salve o arquivo JavaScript externo com a extensão “.js”.

Nota: O script externo não pode conter a tag <script>!

Para usar o script externo, indique o arquivo .js com o atributo "src" da tag <script>:

```
<html>
<head>
<script src="xxx.js"></script>
</head>
<body>
</body>
</html>
```

Nota: Lembre de colocar o script exatamente onde você normalmente colocaria o script caso não fosse usar o arquivo externo!

8.6 - Variaveis JavaScript

Uma variavel é um "local de armazenamento" para a informação.

Exemplos

Variaveis

Variaveis são usadas para armazenar dados. Este exemplo mostra como.

```
<html>
<body>

<script type="text/javascript">
var name = "Joao"
document.write(name)
document.write("<h1>"+name+"</h1>")
</script>

<p>Este exemplo declara uma variavel, guarda um valor nesta, e
entao,
mostra o valor da variavel.</p>

<p>Entao, a variavel é mostrada novamente, mas como Subtítulo.</
p>

</body>
</html>
```

Variáveis

Uma variável contém a informação que você deseja guardar. O valor da variavel pode mudar durante a execução do script. Você pode se referir a uma variável pelo nome para ver seu valor ou para mudar seu valor.

Regras para os nomes das variáveis:

- Nomes de Variáveis SÃO “case-sensitive”;
- Eles devem começar com uma letra ou com o caracter underscore “_”.

IMPORTANTE! JavaScript é “case-sensitive”! Uma variável nomeada “nome” não é a mesma que uma nomeada “NOME”!

Declarando uma Variável

Você pode criar uma variavel com o comando “var”:

```
var nome = algum valor
```

Você também pode criar uma variável sem o comando "var":

```
nome = algum valor
```

Atribuindo um Valor para uma Variável

Você pode atribuir um valor para uma variável desta forma:

```
var nome = "joao"
```

Ou assim:

```
nome = "joao"
```

A variável "nome" está ao lado esquerdo da expressão e o valor que você quer atribuir ao lado direito. Agora a variável "nome" tem o valor "joao".

Tempo de vida das Variáveis

Quando você declara uma variavel dentro de uma função, a variavel pode apenas ser acessada dentro da função. Quando você sai da funcão, esta variavel é destruida. Estas variaveis são chamadas variaveis locais. Você pode ter variaveis locais com o mesmo nome em funções diferentes, porque cada uma é reconhecida apenas no escopo da função a ser executada no momento.

Se você declara uma variável fora da função, todas as funções na sua página podem acessá-la. O tempo de vida desta variável inicia quando ela é declarada e termina quando a página é fechada.

Comando If...Else (Se...senao)

Comandos condicionais no JavaScript são usados para executar operações diferentes baseadas em condições diferentes.

8.7 - Exemplos

Comando "if" (Se...)

Como escrever um comando if.

```
<html>
<body>

<script type="text/javascript">
var data = new Date()
var hora = data.getHours()

if (hora < 12)
```

```
{
document.write("<b>Bom Dia</b>")
}
</script>

<p>
Este exemplo demonstra o comando If.
</p>

<p>
Se a hora do seu browser é menor que 12,
você receberá a mensagem “Bom Dia”.
</p>

</body>
</html>
```

If...else statement

Como escrever um comando “if...else” (Se...Senao).

```
<html>
<body>

<script type="text/javascript">
var data = new Date()
var hora = data.getHours()

if (hora < 12)
{
document.write("<b>Bom Dia</b>")
}
else
{
document.write("<b>Boa Tarde</b>")
}
</script>

<p>
Este exemplo demonstra o comando “If...Else”.
</p>

<p>
Se a hora do seu browser é menor que 12,
você receberá a mensagem "Bom Dia".
Senão receberá  a mensagem "Boa Tarde".
</p>

</body>
</html>
```

Comando "If..elseif...else"

Como escrever um comando "if..else if...else" (Se...Senão Se...Senão).

```
<html>
<body>

<script type="text/javascript">
var data= new Date()
var hora = data.getHours()
if (hora<12)
{
document.write("<b>Bom Dia</b>")
}
else if (hora>=12 && hora<=18)
{
document.write("<b>Boa Tarde</b>")
}
else
{
document.write("<b>Boa Noite</b>")
}
</script>

<p>
Este exemplo demonstra o comando "if..else if...else".
</p>

</body>
</html>
```

Link Aleatório

Este exemplo demonstra um link, que te levará ao site "www.google.com" ou ao site "www.cade.com.br" aleatoriamente com 50% de chance para cada um deles.

```
<html>
<body>
<script type="text/javascript">
var r=Math.random()
if (r>0.5)
{
document.write("<a href='http://www.google.com'>Ferramenta de
Busca</a>")
}
else
{
document.write("<a href='http://www.cade.com.br'>Ferramenta de
Busca</a>")
}
</script>

</body>
</html>
```

8.8 - Comandos Condicionais

Frequentemente quando escrevemos um código JavaScripts, queremos executar diferentes ações para desisões diferentes. Você pode os comandos condicionais para fazer isto.

No JavaScript podemos ter os seguintes comandos condicionais:

- **if** - Use este comando se deseja executar alguma operação somente se uma condição for verdadeira;
- **if...else** - Use estes comandos se deseja executar uma operação se uma condição for verdadeira ou outra operação se esta condição for falsa;
- **if...else if....else** - Use estes comandos se deseja selecionar um bloco entre varios, dependendo das condições indicadas;
- **witch** - Use este comando se você quer selecionar uma opção entre varias dentro de um conjunto conhecido de opções, para indicar as operações a serem executadas.

O comando "If"

Você deve usar o comando "if" se deseja que uma ou mais operações sejam executadas somente se uma opção for verdadeira.

Sintaxe

```
if (condição)
{
codigo a ser executado se a condição for verdadeira
}
```

Note que "if" é escrito em letras minúsculas. Usando letras maiúsculas (IF) resultará em um erro de JavaScript!

Exemplo 1

```
<script type="text/javascript">
//Escreve uma mensagem "Bom Dia" se
//a hora é menor que 12
var d=new Date()
var hora=d.getHours()

if (hora<12)
{
document.write("<b>Bom Dia</b>")
}
</script>
```

Exemplo 2

```
<script type="text/javascript">
//Escreve "Hora do almoço!" se a hora é igual a 11
var d=new Date()
var hora=d.getHours()

if (hora==11)
{
document.write("<b>Hora do almoço!</b>")
}
</script>
```

Nota: Quando comparando variaveis, devemos usar dois sinais de igualdade ("==")! Perceba também que não existe "else" nesta sintaxe. Você apenas indica o código a executar somente se a condição for verdadeira.

Os Comandos "If...else"

Se você deseja executar uma operação se uma condição for verdadeira e ou outra operação caso esta condição seja falsa, use o comando "If...else"

Sintaxe

```
if (condição)
{
código a ser executado se a condição for verdadeira
}
else
{
código a ser executado se a condição for falsa
}
```

Exemplo

```
<script type="text/javascript">
//Se a hora é menor que 12 exibe "Bom Dia",
//Senão exibe "Boa Tarde".
var d = new Date()
var hora = d.getHours()

if (hora < 10)
{
document.write("Bom Dia!")
}
else
{
document.write("Boa Tarde!")
}
</script>
```

Os comandos "If...else if...else"

Você deve usar os comando "if....else if...else" se você tem mais de duas opções de operações a serem executadas de acordo com as condições dadas.

Sintaxe

```
if (condição1)
{
código a ser executado se a condição1 for verdadeira
}
else if (condição2)
{
código a ser executado se a condição2 for verdadeira
}
else
{
código a ser executado se a condição1 e a condição2 forem falsas
}
```

Exemplo

```
<script type="text/javascript">
var d = new Date()
var time = d.getHours()
if (time<12)
{
document.write("<b>Bom Dia</b>")
}
else if (time>11 && time<19)
{
document.write("<b>Boa Tarde</b>")
}
else
{
document.write("<b>Boa Noite</b>")
}
</script>
```

Comando Switch

O comando switch, funciona como uma chave que comuta entre diversas operações a serem executadas de acordo com uma opção. Ele foi feito para diminuir o uso dos comandos "if...else if...else" e para simplificar o código.

Exemplos

Comando Switch

Como escrever o comando switch:

```
<html>
<body>
<script type="text/javascript">
var d = new Date()
Dia=d.getDay()
switch (Dia)
{
case 5:
  document.write("<b>Sexta-feira</b>")
  break
case 6:

document.write("<b>Sabado</b>")
  break
case 0:
  document.write("<b>Domingo</b>")
  break
default:
  document.write("<b>Quando será o final de semana?</b>")
}
</script>

<p>Este código JavaScript irá gerar mensagens diferentes basean-
do-se no dia da semana atual. Note que Domingo=0, Segunda-
feira=1, Terça-feira=2, etc.</p>

</body>
</html>
```

Quando Usar o Comando Switch do JavaScript

Você deve usar o comando switch quando existem muitas opções de código para serem escolhidas.

Sintaxe

```
switch(n)
{
case 1:
  executa bloco de código 1
  break
case 2:
  executa bloco de código 2
  break
case 3:
  executa bloco de código 3
  break
case 4:
  executa bloco de código 4
  break
...
default:
  codigo a ser executado se nenhuma das opções acima forem
executadas
}
```

É assim que funciona: Primeiro temos uma unica expressão n (frequentemente apenas uma variavel), esta é avaliada apenas uma vez. O resultado desta expressão é comparado com cada caso dentro do bloco do switch. Em caso um caso ser igual à expressão, entao este bloco será executado. Use o "break" para evitar que o código continue executando para o próximo caso automaticamente. Se nenhum caso for igual à expressão, o caso "default" é executado.

Exemplo

```
<script type="text/javascript">
//Voce receberá diferentes mensagens
//dependendo do dia da semana atual.
//Domingo=0, Segunda=1, Terça=2, etc.
var d=new Date()
Dia=d.getDay()
switch (Dia)
{
case 5:
  document.write("Sexta-feira")
  break
case 6:
  document.write("Sabado")
  break
case 0:
  document.write("Domingo")
  break
default:
  document.write("Quando chegará o final de semana?")
}
</script>
```

8.9 - Operadores em JavaScript

Operadores Aritiméticos

Operador	Descrição	Exemplo	Resultado
+	Adição	x=2 y=2	4
-	Subtração	x=5 y=2 x-y	3
*	Multiplicação	x=5 y=4 x*y	20
/	Divisão	15/5 5/2	3 2.5
%	Modulo (resto da divisão)	5%2 10%8 10%2	1 2 0
++	Incrementar (em 1)	x=5 x++	x=6
--	Decrementar (em 1)x--	x=5	x=4

Operadores de Atribuição

Operador	Exemplo	É o mesmo de...
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Operadores de Comparação

Operador	Descrição	Exemplo
==	É igual a	5==8 retorna falso
===	É igual a	(checa valor e tipo) x=5 y="5" x==y retorna verdadeiro x===y retorna falso
!=	Diferente	5!=8 retorna verdadeiro
>	Maior que	5>8 retorna falso
<	Menor que	5<8 retorna verdadeiro
>=	Maior ou Igual a	5>=8 retorna falso
<=	Menor ou Igual a	5<=8 retorna verdadeiro

Operadores Logicos

Operador	Descrição	Exemplo
&&	E	x=6 y=3 (x < 10 && y > 1) retorna verdadeiro
||	OU	x=6 y=3 (x==5 || y==5) retorna falso
!	NÃO	x=6 y=3 !(x==y) retorna verdadeiro

Operador de String (cadeia de texto)

Uma string é geralmente texto, por exemplo "Texto do JavaScript!". Para concatenar uma ou mais strings em uma usamos o operador +.

```
txt1="Mais que"
txt2="otimo dia!"
txt3=txt1+txt2
```

A variavel txt3 agora contém "Mais queotimo dia!".

Para adicionar um espaço entre duas strings, insira o espaço na expressão ou em uma das strings.

```
txt1="Mais que"
txt2="otimo dia!"
txt3=txt1+" "+txt2
ou
txt1="Mais que "
txt2="otimo dia!"
txt3=txt1+txt2
```

A variavel txt3 agora contém "Mais que otimo dia!".

8.10 - Caixa Popup

Em JavaScript podemos criar três tipos de caixa popup: Alert box, Confirm box, e Prompt box.

Exemplos

Alert box

```
<html>
<head>
<script type="text/javascript">
function disp_alert()
{
alert("Esta é uma caixa de alerta!!")
```

```
}
</script>
</head>

<body>
<form>
<input type="button" onclick="disp_alert()" value="Mostra uma
caixa de alerta">
</form>
</body>

</html>
```

Alert box com quebra de linha

```
<html>
<head>
<script type="text/javascript">
function disp_alert()
{
alert("Olá denovo! É assim que adicionamos uma" + '\n' + "quebra
de linha em uma caixa de alerta!")
}
</script>
</head>

<body>
<form>
<input type="button" onclick="disp_alert()" value="Display alert box">
</form>
</body>

</html>
```

Confirm box

```
<html>
<head>
<script type="text/javascript">
function disp_confirm()
{
var name=confirm("Aperte um botao")
if (name==true)
{
document.write("Você apertou OK!")
}
else
{
document.write("Você cancelou a janela!")
}
}
</script>
</head>

<body>
```

```
<form>
<input type="button" onclick="disp_confirm()" value="Mostra uma
caixa de confirmacao">
</form>
</body>

</html>
```

Prompt box

```
<html>
<head>
<script type="text/javascript">
function disp_prompt()
{
var nome=prompt("Qual o seu nome?","")
if (nome!=null && nome!="")
{
document.write("Olá " + nome + "! Como vai?")
}
}
</script>
</head>

<body>
<form>
<input type="button" onclick="disp_prompt()" value="Mostra uma
caixa de pergunta">
</form>
</body>

</html>
```

Alert Box

Uma “alert box” é usada se você deseja ter certeza que uma informação é lida pelo usuário.

Quando uma “alert box” aparece, o usuário terá que precionar "OK" para continuar.

Sintaxe:

```
alert("texto")
```

Confirm Box

Uma “confirm box” é usada para confirmar ou aceitar alguma coisa.

Quando uma “confirm box” aparece, o usuário terá que clicar em "OK" ou "Cancel" para continuar.

Se o usuário clica em "OK", a caixa retorna verdadeiro, se ele clicar em "Cancel" (ou “Cancelar”, dependendo do browser), a caixa retorna falso.

Sintaxe:

```
confirm("texto")
```

Prompt Box

Uma “prompt box” é usada quando precisamos de um valor antes de entrar na página.

Quando a caixa aparece, o usuário precisará precionar "OK" ou "Cancel" para continuar depois de entrar com o valor.

Se o usuário clicar em "OK" a caixa retorna o valor inserido. Se ele clicar em "Cancel" a caixa retorna null.

Sintaxe:

```
prompt("texto","valor padrao")
```

8.11 - Funções JavaScript

Uma função é um bloco de código reutilizável que será executado em um evento ou quando chamada.

Exemplos

```
Function
Como chamar uma função:
<html>
<head>

<script type="text/javascript">
function minhafuncao()
{
alert("Oi")
}
</script>

</head>
<body>

<form>
<input type="button"
onclick="minhafuncao()"
value="Chamar funcao">
</form>

<p>Apertando o botao, a funcao será chamada. A funcao ira mos-
trar a mensagem.</p>

</body>
</html>
```

Função com parâmetros

Como passar uma variavel como parâmetro para uma função:

```
<html>
<head>

<script type="text/javascript">
function minhafuncao(txt)
{
alert(txt)
}
</script>

</head>
<body>

<form>
<input type="button"
onclick="minhafuncao('Oi')"
value="Chamar funcao">
</form>

<p>Apertando o botao, a funcao será chamada. A funcao ira mos-
trar a o parâmetro.</p>

</body>
</html>
```

Função com parâmetro (varias chamadas)

A função pode ser chamada com diferentes parâmetros a cada chamada.

```
<html>
<head>
<script type="text/javascript">
function minhafuncao(txt)
{
alert(txt)
}
</script>
</head>

<body>
<form>
<input type="button"
onclick="minhafuncao('Bom Dia!')"
value="De manha">

<input type="button"
onclick="minhafuncao('Boa Tarde!')"
value="De tarde">
</form>

<p>
Dependendo do botao que clicar, uma mensagem diferente aparecera,
executando a mesma funcao.
</p>

</body>
</html>
```

Função que retorna um valor

Como fazer a função retornar um valor:

```
<html>
<head>

<script type="text/javascript">
function minhafuncao()
{
return ("Oi, tenha um bom dia!")
}
</script>

</head>
<body>

<script type="text/javascript">
document.write(minhafuncao())
</script>

<p>O script na seção body chama a função.</p>

<p>A função retorna o texto.</p>

</body>
</html>
```

Uma função com parametros que retorna um valor

Como fazer uma função que retorna o produto de dois valores:

```
<html>
<head>
<script type="text/javascript">
function produto(a,b)
{
return a*b
}
</script>
</head>

<body>
<script type="text/javascript">
document.write(produto(4,3))
</script>
<p>O script na seção body chama a função
com os parametros  (4 e 3).</p>
<p>A função retorna o produto destes dois parametros.</p>
</body>
</html>
```

Para evitar que o browser execute um script quando a página carrega, você deve escrever este código como uma função.

Uma function contém algum código que será executado somente por um evento ou por uma chamada à função.

Você pode chamar uma função de qualquer lugar da sua página (ou até de outras pági-

nas se a função está em um arquivo ".js" externo).

Funções são definidas no inicio da página, na seção <head>.

Exemplo

```
<html>
<head>
<script type="text/javascript">
functionmostramensagem ()
{
alert("Este texto foi escrito pelo JavaScript!")
}
</script>
</head>
<body>
<form>

<input type="button" value="Clique aqui!"
onclick="mostramensagem()" >
</form>
</body>
</html>
```

Se a linha: alert("Este texto foi escrito pelo JavaScript!!"), no exemplo acima, não fosse escrita na função esta seria executada assim que a página carregasse. Agora, o script não será executado antes que o usuário aperte o botão. Adicionamos um evento "onClick" ao botão que executará a função "mostramensagem()" quando o botão for clicado.

Como definir uma Função

A sintaxe para criar uma função é:

```
function nome(var1,var2,...,varX)
{
código a ser executado
}
```

var1, var2, etc. São variaveis ou valores passados para função (parêmetros). O "{" e o "}" definem o inicio e o fim da função.

Nota: Uma função sem parâmetros devem incluir os parenteses "()" depois do nome da função:

```
function nome()
{
código a ser executado
}
```

Nota: Não esqueça da importancia das letras maiúsculas e minúsculas do JavaScript! A palavra function precisa ser escrita em letras minúsculas, ou ocorrerá um erro JavaScript! Também lembre que você precisa chamar a função com o nome exato que foi declarado como nome da função (incluindo letras maiúsculas e minúsculas).

O Comando return

O comando return é usado para especificar o fvalor que é retornado pela função.

Então, funções que irão retornar algum valor dever usar o comando return.

Exemplo

A função abaixo deve retornar o produto de dois números (“a” e “b”):

```
function produto(a,b)
{
x=a*b
return x
}
```

Quando você chama a função acima, você deve passar também os dois parâmetros:

```
resultado=produto(2,3)
```

O valor retornado da função produto() é 6, e irá ser guardado na variavel chamada de resultado.

8.12 - Laços ou Repetições

Laços (ou “repetições”) em JavaScript são usados para executar o mesmo bloco de programação um número de vezes ou enquanto uma condição é verdadeira.

Exemplos

O laço “For”

Como escrever um laço. Use um laço “For” para executar o mesmo bloco de programação varias vezes.

```
<html>
<body>

<script type="text/javascript">
for (i = 0; i <= 5; i++)
{
document.write("O numero e " + i)
document.write("<br>")
}
</script>

<p>Explicacao:</p>

<p>Este laco começa com i=0 (i vem de indice).</p>

<p>Enquanto <b>i</b> é menor, ou igual a 5, o laço continuará a
executar.</p>

<p><b>i</b> será incrementado em 1 a cada “volta” do laço.</p>

</body>
</html>
```

Frequentemente quando escrevemos um código, queremos que o mesmo bloco execute varias vezes seguidas. Para não termos que adicionar o mesmo código dezenas ou as vezes centenas de vezes podemos usar o laço para fazer estas operações.

Em JavaScript existem dois tipos de laços diferentes:

- **for –** repete um bloco um numero específico de vezes;
- **while –** repete um bloco enquanto uma condição é atendida (verdadeira)

O laço "For"

O laço for é usado quando se sabe o número de vezes em que um script será executado.

Sintaxe

```
for
(variavel=valorinicial;variavel<=valorfinal;variavel=variavel+incremento)
{
    codigo a ser executado
}
```

Exemplo

Explicação: O exemplo abaixo define um laço que inicia com i=0. O laço irá continuar a executar enquanto i for menor ou igual a 10. i será incrementado em 1 à cada volta.

Nota: O parâmetro incremento pode também ser negativo e a condição pode ser qualquer outra operação lógica.

```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=10;i++)
{
document.write("O numero e " + i)
document.write("<br />")
}
</script>
</body>
</html>
```

Resultado

```
O numero e 0
O numero e 1
O numero e 2
O numero e 3
O numero e 4
O numero e 5
O numero e 6
O numero e 7
O numero e 8
O numero e 9
O numero e 10
```

O laço "While"

Laços em JavaScript também podem ser usados para executar o mesmo bloco de código mesmo se não sabemos quantas vezes este bloco será executado. Esta é a principal função do While.

Exemplos

Laço While

Como fazer o laço "while" executar a mesma operação que o laço "for".

```
<html>
<body>

<script type="text/javascript">
i = 0
while (i <= 5)

{
document.write("O numero e " + i)
document.write("<br>")
i++
}
</script>

<p>Explicacao:</p>

<p>Este laco começa com i=0 (i vem de indice).</p>

<p>Enquanto <b>i</b> é menor, ou igual a 5, o laço continuará a
executar.</p>

<p><b>i</b> será incrementado em 1 a cada "volta" do laço.</p>

</body>
</html>
```

O laço While para verificação de senhas

Pedindo uma senha usando o laço while. O resto da página não será mostrado enquanto a senha não for digitada.

```
<html>
<body>

<script type="text/javascript">
senha=prompt("Senha","")
while (senha!="senha")
{
senha=prompt("Senha","")
}
```

```
</script>

</body>
</html>
```

O laço “while”

O laço while é usado quando queremos que um bloco de código continue sendo executado enquanto uma certa condição é atendida (verdadeira).

```
while (condicao)
{
    codigo a ser executado
}
```

Exemplo

Explicação: O exemplo abaixo define um laço que inicia com i=0. O laço continua repetindo ENQUANTO i for menor ou igual a 10. i será incrementado em um a cada “volta”.

```
<html>
<body>
<script type="text/javascript">
var i=0
while (i<=10)
{
document.write("O numero e " + i)
document.write("<br />")
i=i+1
}
</script>
</body>
</html>
```

Resultado

```
O numero e 0
O numero e 1
O numero e 2
O numero e 3
O numero e 4
O numero e 5
O numero e 6
O numero e 7
O numero e 8
O numero e 9
O numero e 10
```

8.13 - Eventos em Javascript

Eventos são ações que podem ser detectadas pelo JavaScript.

Eventos

Usando o JavaScript, temos a habilidade de criar páginas dinâmicas.

Cada elemento da página web tem certos eventos que podem ser acionados pelas funções JavaScript. Por exemplo, podemos usar o evento "onClick" de um botao para indicar que uma função será executada quando um usuário clicar no botão. Definimos os eventos em tags HTML.

Exemplos de eventos:

- Um clique do mouse
- Uma imagem ou página sendo carregada
- Mouse movendo sobre certas partes da página
- Selecionando uma caixa de entrada de dados de um formulário
- Enviando um formulario HTML
- Apertando uma tecla

No apendice B temos uma tabela com eventos do JavaScript

Nota: Eventos são normalmente associados a funções. E funções não são executadas antes de um evento ocorrer.

"onload" e "onUnload"

Os eventos onload e onUnload são acionados quando o usuário entra ou sai da página.

O evento "onload" é geralmente usado para checar o browser (e versão do browser) do visitante, para que sejam carregados os elementos corretos da página.

Tanto o evento onload e o onUnload são usados, também, para manipular cookies que devem ser acionados quando o usuário entra ou sai da página. Por exemplo, você poderia ter um popup perguntando pelo nome do usuário mas apenas na primeira vez que ele visita a página. O nome é armazenado em um cookie. Da próxima vez que o usuário visita a página, você ter outro popup dizendo "Benvindo Fulano de Tal!", por exemplo.

onFocus, onBlur and onChange

Os eventos onFocus, onBlur and onChange são geralmente usados em combinação com funções de validação de formulário.

Abaixo está um exemplo de como usar o evento onChanget. A função checaEmail() irá ser chamada sempre que o usuário modifica o conteúdo do campo:

```
<input type="text" size="30"
id="email" onchange="checaEmail()">;
```

onSubmit

O evento onSubmit é usado para validar TODOS os campos do formulário antes de enviá-los.

Abaixo está um exemplo de como usar o evento onSubmit. A função checaForm() será chamada quando o usuário clicar no botão de envio (submit) no formulário. Se os valores dos campos não são aceitáveis, o envio é cancelado. A funçao checaForm() retorna verdadeiro ou falso. Se ela retornar verdadeiro o formulário será enviado, senão o envio será cancelado:

```
<form method="post" action="xxx.htm"
onsubmit="return checkForm()">
```

onMouseOver e onMouseOut

onMouseOver e onMouseOut são geralmente usados em botões "animados".

Veja um exemplo de um evento “onMouseOver” abaixo. Uma “alert box” aparece quando o mouse está acima do link:

```
<a href="http://www.google.com"
onmouseover="alert('An onMouseOver event');return false">
<img src="w3schools.gif" width="100" height="30">
</a>
```

MODULO 9: Introdução ao PHP

Um arquivo PHP pode conter text, tags HTML e scripts. Scripts em um arquivo PHP são executados no servidor. Isto diferencia o PHP do Javascript, dá algumas novas possibilidades ao webdesigner.

9.1 - O que é PHP?

- PHP significa PHP: Hypertext Preprocessor;
- PHP é uma linguagem de script "server-side" (que executa no servidor);
- PHP suporta varios bancos de dados (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)
- PHP é um software open source (OSS);
- PHP é gratis para fazer o download e usar.

O que é um arquivo PHP?

- Arquivos PHP podem conter texto, tags HTML e scripts;
- Arquivos PHP são retornados aos browsers como HTML puro;
- Arquivos PHP têm a extensão ".php", ".php3", ou ".phtml".

Porquê PHP?

- PHP "roda" em diferentes plataformas (Windows, Linux, Unix, etc.);
- PHP é compatível com quase todos os servidores usados atualmente (Apache, etc.)
- PHP é GRATIS, e o download pode ser feito do site oficial PHP: www.php.net;
- PHP é fácil de aprender e roda eficientemente no servidor, não exigindo nenhum download por parte do usuário.

9.2 - Instalando o PHP (no servidor)

Do que precisamos?

Se seu servidor suporta o PHP – você não precisa fazer nada! Não é necessário compilar nada ou instalar ferramentas extras – apenas crie alguns arquivos .php na sua pasta web (ou envie para seu servidor) – e o servidor fará todo o trabalho para você. A maioria dos servidores da web oferecem suporte ao PHP.

No entanto, se seu servidor não suporta PHP, você precisa instalá-lo. Abaixo está um link para um bom tutorial de como instalar o PHP 4:

http://hotwired.lycos.com/webmonkey/00/44/index4a.html?tw=programming

Download do PHP

Download do PHP (gratis): http://www.php.net/downloads.php

9.3 - Sintaxe do PHP

Você não pode ver o código do PHP selecionando a opção "View source" ou "Exibir Código Fonte" em seu browser – você apenas verá o arquivo HTML resultante do PHP. Isto acontece porque os scripts são executados no servidor e enviados como arquivos HTML ao seu browser.

Neste capítulo, veremos que a sintaxe do PHP é bastante semelhante à do JavaScript.

Sintaxe básica do PHP

Um arquivo PHP normalmente contém tags HTML, do mesmo modo que um arquivo HTML, e alguns códigos de script PHP.

Abaixo, temos um exemplo de um simples script PHP, que envia uma mensagem "Isto é um texto PHP" ao browser:

```
<html>
<body>
<?php echo "Isto é um texto PHP"; ?>
</body>
</html>
```

Os blocos de script PHP sempre começam com "<?php" e terminam com "?>". Um script PHP pode ser colocado em qualquer lugar no documento.

Cada linha de código em PHP precisa do ponto-e-vírgula (";") no final. O ponto-e-vírgula é um separador e é usado para distinguir um conjunto de instruções de outro.

Existem dois comandos básicos de exibição no PHP: "echo" e "print". No exemplo acima usamos o comando echo para exibir o texto.

9.4 - Variaveis in PHP

Uma das principais diferenças entre o PHP e o JavaScript é que em PHP todas as variaveis começam com o simbolo "$". Assim como no JavaScript as variaveis podem conter strings, números, etc.

Abaixo, o script PHP atribui a string "Isto é um texto PHP" à variavelb chamada $txt:

```
<html>
<body>
<?php
$txt="Isto é um texto PHP";
echo $txt;
?>
</body>
</html>
```

Para concatenar duas ou mais strings usamos o operador ponto (.):

```
<html>
<body>
<?php
$txt1="Isto é um texto";
$txt2="PHP";
```

```
echo $txt1 . " " . $txt2 ;
?>
</body>
</html>
```

A saída do script acima será: "Isto é um texto PHP".

9.5 - Comentários em PHP

Em PHP, usamos o "//" para fazer comentários em uma linha (como no JavaScript) ou "/*" e "*/" para fazer grandes blocos de comentários (mais de uma linha).

```
<html>
<body>
<?php
// Isto é um comentário
/*
Este é
um bloco
de comentario
*/
?>
</body>
</html>
```

9.6 - Operadores do PHP

Esta seção lista os diferentes operadores do PHP.

Operadores Aritiméticos

Operador	Descrição	Exemplo	Resultado
+	Adição	x=2 x+2	4
-	Subtração	x=2 5-x	3
*	Multiplicação	x=4 x*5	20
/	Divisão	15/5 5/2	3 2.5
%	Modulo (resto da divisão)	5%2 10%8 10%2	1 2 0
++	Incremento (em 1)	x=5 x++	x=6
--	Decremento (em 1)	x=5 x--	x=4

Operadores de Atribuição

Operador	Exemplo	É o mesmo de...
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Operadores de Comparação

Operador	Descrição	Exemplo
==	Igual à	5==8 retorna Falso
!=	Diferente de	5!=8 retorna Verdadeiro
>	Maior que	5>8 retorna Falso
<	Menor que	5<8 retorna Verdadeiro
>=	Maior ou Igual a	5>=8 retorna Falso
<=	Menor ou Igual a	5<=8 retorna Verdadeiro

Operadores Lógicos

Operator	Description	Example
&&	E	x=6 y=3 (x < 10 && y > 1) retorna Verdadeiro
||	Ou	x=6 y=3 (x==5 || y==5) retorna Falso
!	Não	x=6 y=3 !(x==y) retorna Verdadeiro

9.7 - Comandos condicionais em PHP

Assim como no JavaScript, temos os mesmos comandos condicionais no PHP.

No PHP os seguintes comandos condicionais:

- **if (...else) –** usamos este comando quando queremos executar um conjunto de código quando uma condição é verdadeira (e outra quando a condição é falsa);
- **switch –** usamos este comando quando queremos selecionar um bloco de comandos dentre varios dependendo de uma opção.

O comando If

Se queremos que algum código seja executado somente quando uma condição é verdadeira ou falsa, usamos o comando if....else.

Sintaxe

```
if (condição)
codigo a ser executado se a condição é verdadeira;
else
codigo a ser executado se a condição é falsa;
```

Exemplo

O exemplo irá exibir "Tenha um bom final de semana!" se o dia é sexta feira, senão irá exibir "Tenha um bom dia!":

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
echo "Have a nice weekend!";
else
echo "Have a nice day!";
?>
</body>
</html>
```

Assim como no JavaScript, se mais de uma linha deve ser executada quando a condição é verdadeira (ou falsa), estas linhas devem estar dentro das chaves “{” e “}”:

```
<html>
<body>
<?php
$x=10;
if ($x==10)
{
echo "Ola<br>";
echo "Bom dia<br>";
}
?>
</body>
</html>
```

O comando Switch

Se você deseja selecionar um bloco de código entre vários, deve usar o comando Switch.

Sintaxe

```
switch (expressão)
{
case rotulo1:
  codigo a ser executado se a expressão  = label1;
  break;
case rotulo2:
  codigo a ser executado se a expressão = label2;
  break;
default:
  codigo a ser executado se
  a expressão é diferente
  tanto de label1, quanto de label2;
}
```

Exemplo

```
<html>
<body>
<?php
switch ($x)
{
case 1:
  echo "Number 1";
  break;
case 2:
  echo "Number 2";
  break;
case 3:
  echo "Number 3";
  break;
default:
  echo "No number between 1 and 3";
}
?>
</body>
</html>
```

9.8 - Laços em PHP

Em PHP temos os seguintes laços:

- **while –** repete um bloco de texto enquanto uma condição é verdadeira;
- **for –** repete um bloco de texto um número específico de vezes;

O comando While

O comando while irá executar um bloco de código enquanto uma condição for verdadeira.

Sintaxe

```
while (condição)
código a ser executado;
```

Exemplo

O seguinte exemplo demonstra um loop que irá continuar a executar enquanto a variavel i é menor ou igual a 5. i será incrementado em um a cada volta:

```
<html>
<body>
<?php
$i=1;
while($i<=5)
{
echo "O numero e " . $i . "<br>";
$i++;
}
?>
</body>
</html>
```

O comando For

O comando For é usado quando você sabe quantas vezes você quer executar um bloco de comandos.

Sintaxe

```
for (initialização; condição; incremento)
{
    código a ser executado;
}
```

Exemplo

O exemplo seguinte exibe o texto "Isto é um texto PHP!" cinco vezes:

```
<html>
<body>
<?php
for ($i=1; $i<=5; $i++)
{
    echo "Isto é um texto PHP!<br>";
}
?>
</body>
</html>
```

9.9 - Formularios e PHP

Uma grande utilidade do PHP é como ele manipula os formulários HTML.

Manipulando formulários com PHP

O a coisa mais importante quando manipulamos formulários com o PHP, é que qualquer elemento do formulário estará automaticamente disponível no nosso script PHP.

Veja este exemplo de formulário HTML:

```
<html>
<body>
<form action="benvindo.php" method="POST">
Digite seu nome: <input type="text" name="nome" /><br>
Digite sua idade: <input type="text" name="idade" /><br>
<input type="submit" />
</form>
</body>
</html>
```

O exemplo acima contém duas caixas de texto e um botão de envio. Quando o usuário clica no botão, o arquivo “welcome.php” é chamado.

Vejamos agora como o arquivo “welcome.php” manipula o formulário:

```
<html>
<body>
Benvindo <?php echo $_POST["nome"]; ?>.<br />
Voce tem <?php echo $_POST["idade"]; ?> anos!
</body>
</html>
```

A saída deste script será:

```
Benvindo Joao.
Voce tem 28 anos!
```

As variaveis “$_POST["name"]” e “$_POST["age"]” são automáticamente atribuidas para você pelo PHP. A variavel “$_POST” contém TODOS os dados do formulário.

Nota: Se o atributo “method” do formulário é GET, então os dados do formulário estarão na variável “$_GET” e não na variavel “$_POST”.

Modulo 10: Enviando seu site

10.1 - O que é FTP?

FTP (File Transfer Protocol ou protocolo de transmissão de arquivos) é o mais simples e seguro modo para transmitir e receber arquivos na internet. Sabendo ou não, você provavelmente usa o ftp frequentemente.

O uso mais comum do FTP é para baixar arquivos (download) na internet. Por isso, FTP é vital para a maioria dos leilões online, jogos e etc. Além disso, a habilidade de transferir arquivos (enviar e receber) torna o FTP essencial para qualquer um criando um Web site, tanto amadores como profissionais.

Quando se faz o download de um arquivo da Internet você está transferindo um arquivo para o seu computador de outro computador na Internet. Você pode ou não saber onde está o computador com o arquivo, mas você sabe o seu URL (ou endereço).

Um endereço FTP parece com um endereço HTTP (Protocolo de transmissão de Hypertexto), ou endereço do, exceto que ele usa o prefixo ftp:// ao invéz de http://.

Exemplo de endereço de Website:	http://www.meusite.com.br
Exemplo de endereço FTP:	ftp://ftp.meusite.com.br

Frequentemente, um computador com um endereço FTP é dedicado em receber uma conecção FTP. Assim como um computador dedicado a hospedar sites é chamado de Web Server, um computador dedicado a receber uma conecção FTP é chamado de “FTP server”.

10.2 - O que é um FTP Server?

Um FTP server é como um grande armário de arquivos. Ele armazena os arquivos, com opções de organização e rótulos. Os arquivos podem ser definidos como públicos ou privados e um usuário pode ter acesso a arquivos que outro não possui.

Para acessar um servidor FTP precisamos de um Login e uma Senha. Se o criador do FTP server deseja liberar o acesso ao público, então, o login é 'anonymous' e a senha é seu endereço de email. Se o FTP server não é público, existirá um login único (e senha) para cada usuário.

Quando conectando a um FTP server que permite login anônimo, não será exigido um login e uma senha. Portanto, quando fazendo um download da Internet, você poderá estar usando um FTP anônimo e nem se dar conta disso.

Para fazer uma conexão FTP você pode usar um Web browser comum (Mozilla, Firefox, Netscape, etc.) ou uma aplicação FTP dedicada, que chamamos de “FTP Client”.

Quando usando um Web browser para uma conexão FTP, uploads (envio) FTP são difíceis, ou muitas vezes impossível, e os downloads não são protegidos (não é recomendado para upload ou download de arquivos grandes).

Quando usamos um FTP Client, uploads e downloads não podem ser mais fáceis, e você ainda ganha em segurança e outras ferramentas. Por exemplo, você pode pausar um download, ou continuar um que não foi totalmente recebido, que é uma opção muito util para pessoas usando conexões discadas ou instáveis (que frequentemente caem, interrompendo o download).

10.3 - O que é um FTP Client?

Um FTP Client é uma aplicação (ou programa) que foi projetada para transferir arquivos usando o protocolo FTP. Ele precisa ser instalado no computador e só pode ser usado com uma conexão ativa à internet.

O design padrão para um FTP Client consiste em dois paineis, sendo o da esquerda para o computador local e o da direita para o computador remoto. Nesses paineis, são exibidos os arquivos de cada um dos computadores (local e remoto). Os arquivos podem ser facilmente transferidos usando botões nomeados Upload (ou ">>") e botões de Download (ou "<<"). Alguns FTP clients usam o sistema de drag-and-drop para "arrastar" arquivos de um painel para o outro.

Outro recurso útil, principalmente para os desenvolvedores Web, é o recurso de sincronização. Este recurso permite que se mantenha sempre as duas pastas (a pasta do seu site localmente e o seu site no computador remoto) sincronizadas. Este recurso, mantém as duas pastas sempre atualizadas e idênticas, automaticamente.

10.4 - Servidores para Hospedagem de Web Sites

Existem centenas de serviços de hospedagem na web. É importante então saber o que diferencia os servidores, e o que esperar do serviço.

O primeiro ponto que se deve olhar em um servidor, é se ele é compatível com seu site. Se eu site é um pequeno site, sem um banco de dados ou uma linguagem de script como o PHP, então provavelmente qualquer servidor poderá hospedar seu site. Caso contrário esses pontos devem ser observados:

- **Tamanho do espaço disponível para o site:** seu site deve ocupar um espaço menor ou igual ao oferecido pelo serviço. Isso incluindo todos os arquivos, e considerando o crescimento do site a longo prazo;
- **Linguagens de script (server-side):** se seu site usa uma linguagem como o PHP, este servidor precisa oferecer este serviço, para que sua página funcione corretamente. (Note que JavaScript, não é server-side, portanto não depende do servidor);
- **Banco de Dados:** se seu site possui algum tipo de banco de dados, o servidor deve oferecer este serviço, assim como a linguagem de script para manipular o banco;
- **Preço:** alguns servidores são gratúitos, porém, exigem que algum tipo de propaganda seja inserido no site, ou outra forma de pagamento semelhante; outros servidores possuem formas de pagamento mensais, que devem ser compatíveis ao orçamento do site.

APÊNDICE A: REFERÊNCIA DE CSS

PROPRIEDADE BACKGROUND DO CSS

A propriedade background permite a você controlar a cor do fundo de um elemento, usar uma imagem como fundo, repetir uma imagem verticalmente ou horizontalmente, e posicionar uma imagem na página.

PROPRIEDADE	DESCRIÇÃO	VALORES
background	Uma propriedade para se setar todas as propriedades de fundo em uma declaração *background-position*	*background-color* *background-image* *background-repeat* *background-attachment*
background-attachment	Indica se a imagem de fundo é fixa ou ela pode se movimentar com o resto da página (“scroll”).	*scroll* *fixed*
background-color	Modifica a cor de fundo de um elemento	*color-rgb* *color-hex* *color-name* *transparent*
background-image	Seta a imagem de fundo	*url* *none*
background-position	Indica a posição inicial da imagem de fundo	*top left* *top center* *top right* *center left* *center center* *center right* *bottom left* *bottom center* *bottom right* *x-% y-%* *x-pos y-pos*
background-repeat	Indica se (ou como) a imagem de fundo irá se repetir	repeat *repeat-x* *repeat-y* *no-repeat*

PROPRIEDADES DE TEXTO CSS

A propriedade de texto CSS permite a você controlar a aparência do texto. É possível mudar a cor do texto, aumentar ou diminuir o espaço entre caracteres, alinhar, decorar, indentar a primeira linha, etc.

Property	Description	Values
color	Modifica a cor do texto	color
direction	Modifica a direção do texto	ltr rtl
letter-spacing	Aumenta ou diminui o espaçamento entre caracteres	normal length
text-align	Alinha o texto de um elemento	left right center justify
text-decoration	Adiciona uma decoração no texto	none underline overline line-through blink
text-indent	Indenta a primeira linha de um texto em um elemento	length %
text-transform	Transforma as letras em um elemento	none capitalize uppercase lowercase
white-space	Modifica a forma como o espaço é tratado em um elemento	normal pre nowrap
word-spacing	Aumenta ou diminui o espaço entre palavras	normal length

Propriedades de Fonte CSS

A propriedade de fonte permite a você mudar a familia da fonte, e outros estilos do texto.

Propriedade	Descrição	Valores
font	Propriedade para modificar varias propriedades do texto	*font-style* *font-variant* *font-weight* *font-size/line-height* *font-family* caption icon menu message-box small-caption status-bar
font-family	Lista de fontes em uma familia	family-name generic-family
font-size	Modifica o tamanho da fonte	xx-small x-small small medium large x-large xx-large smaller larger length %

Propriedade	Descrição	Valores
font-size-adjust	Especifica um valor de aspectopara um elemento que irá preservar o x-height da fonte	none *number*
font-stretch	Condensa ou espande a fonte atual	normal wider narrower ultra-condensed extra-condensed condensed semi-condensed semi-expanded expanded extra-expanded ultra-expanded
font-style	Muda o estilo da fonte	normal italic oblique
font-variant	Mostra o texto em small-caps ou não	normal small-caps
font-weight	Muda a largura da fonte	normal bold bolder lighter 100 200 300 400 500 600 700 800 900

Propriedade de Borda

As propriedades de borda no CSS permitem que você especifique o estilo e cor da borda de um elemento. Em HTML usamos tabelas para criar bordas ao redor do texto, mas com as propriedades de borda do CSS podemos criar bordas com efeitos, e elas podem ser aplicadas a qualquer elemento.

Property	Description	Values
border	Propriedade básica de borda	border-width border-style border-color
border-bottom	Propriedade para a borda inferior	border-bottom-width border-style border-color
border-bottom-color	Propriedade para a cor da borda inferior	border-color
border-bottom-style	Propriedade para o estilo da borda inferior	border-style
border-bottom-width	Largura da borda inferior	thin medium thick length
border-color	Cor de todas as bordas (superior, inferior e laterais)	color

Property	Description	Values
border-left	Propriedade para a borda esquerda	border-left-width border-style border-color
border-left-color	Cor da borda esquerda	border-color
border-left-style	Estilo da borda esquerda	border-style
border-left-width	Largura da borda esquerda	thin medium thick length
border-right	Propriedade para a borda direita	border-right-width border-style border-color
border-right-color	Cor da borda direita	border-color
border-right-style	Estilo da borda direita	border-style
border-right-width	Largura da borda direita	thin medium thick length
border-style	Estilos para todas as bordas	none hidden dotted dashed solid double groove ridge inset outset
border-top	Propriedade para a borda superior	border-top-width border-style border-color
border-top-color	Cor da borda superior	border-color
border-top-style	Estilo da borda superior	border-style
border-top-width	Largura da borda superior	thin medium thick length
border-width	Largura de todas as bordas	thin medium thick length

Apêndice B: Referência de Eventos JavaScript

Eventos são normalmente usados em combinação com funções, e as funções não serão executadas antes que um evento ocorra!

Manipulação de Eventos

Uma novidade do HTML 4.0 foi a habilidade de deixar eventos HTML acionarem ações no browser, como acionar uma função Javascript quando o usuário clica em um elemento HTML. Abaixo está uma lista de atributos que podem ser inseridos em tags HTML para definir as ações dos eventos.

Atributo	O evento ocorre quando...
onabort	O carregamento da imagem é cancelado
onblur	Um elemento perde o "foco"
onchange	O conteúdo de um campo muda
onclick	Usuário clica em um objeto com o mouse
ondblclick	Usuário dá um duplo-click em um object com o mouse
onerror	Um erro ocorre quando carregando um documento ou imagem
onfocus	Um elemento ganha o "foco"
onkeydown	Uma tecla é apertada (ocorre antes de soltar a tecla)
onkeypress	Uma tecla é apertada e solta
onkeyup	Uma tecla é solta (ocorre depois de soltar a tecla)
onload	Uma página ou imagem é carregada
onmousedown	O botão do mouse é precionado
onmousemove	O mouse é movido (sobre o objeto)
onmouseout	O mouse é movido para fora de um objeto
onmouseover	O mouse é movido para dentro de um objeto
onmouseup	O botão do mouse é solto
onreset	O botão reset é precionado
onresize	A janela ou frame é redimensionado
onselect	O texto é selecionado
onsubmit	O botão de envio (submit) é precionado
onunload	O usuário sai da página

Apendice C: Caracteres especiais no JavaScript

Em JavaScript você pode adicionar caracteres especiais à uma string de texto usando a barra inversa.

Inserindo Caracteres Especiais

A barra inversa (\) é usada para inserir apostrofes, quebras de linha, aspas, e outros caracteres especiais para uma string de texto.

Olhe o seguinte código JavaScript:

```
var txt="Somos os "Vikings" vindos do norte."
document.write(txt)
```

No JavaScript, uma string é iniciada e terminada com aspas simples ou duplas. Isto significa que a string acima será cortada em “Somos os ”.

Para resolver isso, você precisa usar a barra inversa (\) antes de cada aspa na palavra "Vikings". Isto transforma cada aspa em uma string literal:

```
var txt="Somos os \"Vikings\" vindos do norte."
document.write(txt)
```

JavasScript irá agora exibir a string corretamente: Somos os "Vikings" vindos do norte.

Aqui está outro exemplo:

```
document.write ("Voce \& eu estamos cantando!")
```

O exemplo acima irá produzir a seguinte saída:

```
Voce & eu estamos cantando!
```

A tabela abaixo lista outros caracteres especiais que podem ser adicionados à uma string de texto com a barra inversa:

Código	Saídas
\'	Aspas simples
\"	Aspas
\&	“e” comercial (“&”)
\\	Barra inversa (“\”)
\n	Quebra de linha
\r	Retorno de linha
\t	Tabulação
\b	Backspace
\f	Alimentação do formulario

Bibliografia

Internet:

http://linux.about.com/

http://www.php.net/

http://www.google.com/

http://www.nvu.com

http://pt.wikipedia.org/

Livros:

HTML 4: Guia de Consulta Rápida

Autores...: RUBENS PRATES e MARCELO SILVEIRA

Editora...: Novatec

Ano/Edição: 2001 / 1

JavaScript: Guia de Consulta Rápida

Autor.....: EDGARD B. DAMIANI

Editora...: Novatec

Ano/Edição: 2001 / 1

HTML 4: Guia de Consulta Rápida

Autores...: RUBENS PRATES e MARCELO SILVEIRA

Editora...: Novatec

Ano/Edição: 2001 / 1

Desenvolvendo Websites com PHP

Autor.....: JULIANO NIEDERAUER

Editora...: Novatec

Ano/Edição: 2004 / 2

JavaScript: a Bíblia

Autor.....: DANNY GOODMAN

Editora...: Campus

Ano?Edição: 2001 / 1

APÊNDICE: EDITORES WYSIWYG

WYSIWYG é o acrónimo da expressão em inglês "What You See Is What You Get", que pode ser traduzido para "O que você vê é o que você recebe" (OQVVEOQVR). Trata-se de um método de edição no qual o usuário vê o resultado no momento da edição na tela do computador já com a aparência do produto final. Um exemplo clássico de editor WYSIWYG é o Microsoft Word, no qual o documento é mostrado na tela da mesma forma que será impresso.

Dirigido a criação de Web Sites, o mais conhecido atualmente para Windows é o Macromedia Dreamweaver, no qual qualquer pessoa, com o mínimo de conhecimento em HTML pode fazer muito rapidamente uma página ou até um site inteiro para internet. Porem o alto custo da licença o torna acessível apenas as grandes empresas de desenvolvimento.

O Nvu (pronuncia-se Nview) é um editor HTML, estilo WYSIWYG, de código livre produzido pela Linspire. Disponível para Linux, MacOS e Windows, tem como objetivo criar uma alternativa livre diante dos softwares proprietários como Macromedia Dreamweaver e Microsoft FrontPage. Ele é baseado no Composer, um editor HTML integrante do Mozilla e Netscape.

A Cobra Tecnologia produziu uma apostila em português do Nvu que será distribuída junto com o software no programa do Governo Federal "Computador Para Todos". A apostila tem 21 páginas e destina-se a desenvolvedores e web designers que ainda não tiveram contato com a ferramenta.

A apostila é licenciada pela Creative Commons e pode ser copiada, alterada, ampliada e redistribuida livremente, desde que mantido o crédito da autoria. Uma copia pode ser obtida facilmente fazendo uma busca através no google (www.google.com.br) ou ainda pelo endereço: "www.nvu.com/guide/apostila_nvu_cobra1-1.0.pdf".

APÊNDICE: XHTML

De onde veio XHTML?

As tags e atributos do XHTML foram criadas com base nas tags e atributos do HTML. Logo o XHTML é uma linguagem de marcação que estende o HTML e a transformação de um documento existente de HTML para XHTML é uma tarefa bem simples.

Qual a finalidade do XHTML ?

XHTML é a sigla em inglês para EXtensible HyperText Markup Language que em português resulta em Linguagem Extensível para Marcação de Hipertexto, escrita para substituir o HTML e nada mais é do que um HTML "puro, claro e limpo".

Vantagens de se usar XHTML

- A compatibilidade da linguagem XHTML com as futuras aplicações de usuários, garantindo que as criações XHTML conservarão se estáveis por longos anos.
- A edição de um código XHTML existente é uma tarefa bem simples por se tratar de uma escrita limpa.
- O tempo de carregamento de uma página XHTML é mais rápido pois os browsers tem a interpretar uma página limpa sem ter que interpretar e decidir sobre renderização de erros de código.
- Uma página XHTML é compatível com os principais browsers evitando assim um comportamento não previsto ao se trocar de browser.

As diferenças entre XHTML e HTML

As principais diferenças são:

- todas as tags devem ser escritas em letras minúsculas;
- os elementos (tags) devem estar convenientemente aninhados;
- os documentos devem ser bem formados;
- o uso de tags de fechamento é obrigatório;
- elementos vazios devem ser fechados;
- diferenças para os atributos.

Todas as tags devem ser escritas em letras minúsculas

A XHTML é case-sensitive (sensível ao tamanho da letra).

Errado: `<DIV><P>Aqui um texto</P></DIV>`

Certo: `<div><p>Aqui um texto</p></div>`

Os elementos (tags) devem estar convenientemente aninhados

Errado: Aqui um texto negrito

Certo: Aqui um texto negrito

Os documentos devem ser bem formados

Um documento é bem formado quando todos os elementos XHTML devem estão corretamente aninhados dentro do elemento raiz <html>.

A estrutura básica do documento deve ser conforme abaixo:

```
<html>
<head>
...
</head>
<body>
...
</body>
</html>
```

O uso de tags de fechamento é obrigatório

Em HTML é permitido para determinados elementos, omitir-se a tag de fechamento. O XHTML não permite omissão de qualquer tag de fechamento.

Errado: Um parágrafo. Outro parágrafo.

Certo: Um parágrafo. Outro parágrafo.

Elementos vazios devem ser fechados

Elementos vazios devem ter uma tag de fechamento ou a tag de abertura deve terminar com />. Como exemplo,
 ou <hr></hr>.

Errado: Elementos vazios sem terminação

```
<br>
<hr>
<img src="imagem.gif" alt="minha imagem ">
```

Certo: Elementos vazios com terminação

```
<br />
<hr />
<img src="imagem.gif" alt="minhaimagem " />
```

Diferenças para os atributos

Nomes de atributos

Assim como as tags, os atributos também são case-sensitive então deve-se escrever nomes de atributos em minúsculas;

Errado: <td ROWSPAN=”3">

Certo: <td rowspan=”3">

Valores de atributos

Os valores de atributos devem estar entre “aspas”;

Errado: <td rowspan=3>

Certo: <td rowspan=”3">

Sintaxe dos atributos

A sintaxe para atributos deve ser escrita por completo;

Errado: <input checked />

Certo: <input checked=”checked” />

Abaixo uma relação dos atributos que se enquadram nesta recomendação

Errado:	**Certo:**
compact	compact=”compact”
checked	checked=”checked”
declare	declare=”declare”
readonly	readonly=”readonly”
disabled	disabled=”disabled”
selected	selected=”selected”
defer	defer=”defer”
ismap	ismap=”ismap”
nohref	nohref=”nohref”
noshade	noshade=”noshade”
nowrap	nowrap=”nowrap”
multiple	multiple=”multiple”
noresize	noresize=”noresize”

Os atributos id e name:

O HTML 4 define o atributo name para os elementos a, applet, form, frame, iframe, img, e map. HTML 4 também introduziu o atribute id. Ambos estes atributos foram projetados para serem usados como identificadores.

Em XHTML, os identificadores são do tipo ID, e poderá existir somente um atributo do tipo ID por elemento. Documentos XHTML DEVEM usar o atributo id ao definir identificadores para os elementos listados acima.

Em XHTML o atributo name destes elementos está formalmente em desuso e será excluído nas versões futuras de XHTML.

Errado:

Certo:

Nota: Por razões de compatibilidade com browsers antigos você pode usar ambos os atributos como abaixo:

Pontos de âncoras

Em HTML para criar um ponto de âncora, associamos um nome ao elemento <a>:

<a name="topo" >Início do parágrafo..bla...

Em XHTML adicione o atributo id:

<a id="topo" name="topo" >Início do parágrafo..bla...

O atributo alt para imagens

Em XHTML o uso do atributo alt para imagens é obrigatório

Se tratar-se de uma imagem decorativa pode-se usar o atributo alt vazio:

Elementos obrigatórios em um documento XHTML

É obrigatório a declaração do DOCTYPE assim como a existências dos elementos <html> <head> <title> e <body>

Um modelo mínimo de documento XHTML é conforme abaixo:

<!DOCTYPE bla..bla..bla>

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

<head>

<title>Título do odcumento</title>

```
</head>
<body>
Conteúdo do documento
</body>
</html>
```

A declaração DOCTYPE não faz parte da marcação XHTML e como tal não é também um elemento, razão pela qual não há necessidade de tag de fechamento.

Para que serve o DOCTYPE ?

A Definição do tipo de documento especifica qual é a sintaxe usada no documento. ELA é usada para validar o documento XHTML. O DOCTYPE deve ser sempre a primeira declaração em um documento web.

Os tipos de DOCTYPE

São três os tipos de DOCTYPE para XHTML:

- STRICT
- TRANSITIONAL
- FRAMESET

<XHTML; 1.0 Strict <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML; 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

Esta é a mais rígida das declarações. Os documentos XHTML no modo Strict não admitem qualquer item de formatação dentro dos elementos e nem elementos em desuso "deprecated" segundo as recomendações do W3C. São indicados para uso com folhas de estilo

<XHTML; 1.0 Transitional <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML; 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Esta declaração permite uma maior flexibilidade e é indicada para documentos que ainda utilizem elementos em desuso ("deprecated"), regras de apresentação embutidas em tags e também para documentos destinados a exibição em browsers sem suporte para CSS. Não admite qualquer tipo de marcação para frames.

<XHTML; 1.0 Frameset <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML; 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

Esta declaração permite tudo da declaração transational e mais os elementos especifico para frames.