Guia Prático de HTML

Este Guia foi escrito por Tiago Daniel de Souza
Email: tiagocopa [at] gmail [dot] com
Site do Autor: http://www.tiagosouza.com

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 2.0 Brazil License.
http://creativecommons.org/licenses/by-nc-sa/2.0/br

SUMÁRIO

1. INTRODUÇÃO

As pessoas imaginam que é muito difícil construir um site. Isto não é verdade!! Qualquer um pode aprender como construir uma página. Se você continuar lendo, estará apto a construir um rapidamente, quando menos esperar.

Outros pensam - *erroneamente* - que será necessário softwares avançados e caros para construir websites. É verdade que existem muitos softwares capazes de criar um website para você. Alguns mais fechados que outros. Mas, se você pretende trilhar o caminho certo, deverá criar você mesmo o website. Felizmente, isto é simples é você já tem a sua disposição todos os softwares que necessita.

O objetivo desta minha apostila é fornecer os conhecimentos básicos que permitirão construir um website de forma correta. A apostila parte da estaca zero e não requer qualquer conhecimento prévio de programação.

Obviamente, a apostila não ensina tudo. Diante disto, será necessário que você se empenhe, pratique e consolide os ensinamentos aqui contidos. Mas, não se aborreça, pois aprender como construir um website é bem divertido e bastante satisfatório quando você trilha o caminho certo do aprendizado.

OK. Chega de conversa. Vamos começar...

2. SOBRE O HTML

O **HTML** é uma **Linguagem de marcação de texto**. Mais especificamente, uma linguagem de marcação de hipertexto. Portanto, antes de começar a falar de HTML, vamos entender o que vem a ser uma linguagem de marcação.

Linguagens de marcação *(markup languages em inglês)* são linguagens que combinam texto com informações extras sobre o texto. Essa informação extra pode ser representada por diversos símbolos ou palavras-chave diferentes, dependendo da linguagem de marcação com que estivermos trabalhando.

O **HTML** não era uma linguagem de formatação de textos qualquer, ela possibilitava ligar textos que estavam num computador a textos que estavam num outro computador, usando como meio a internet. O processador e visualizador de HTML são chamados de navegador, pela característica do hipertexto que permite ao usuário "nadar" na informação.

O navegador *(também chamado de browser)*, nada mais faz do que abrir arquivos de computador e, caso esses arquivos contenham códigos HTML, interpretá-los segundo o padrão do hipertexto e gerar a "página html", que é a manifestação gráfica dos códigos - *aquilo que você usualmente vê quando navega pela internet*. Por convenção, os nomes dos arquivos em HTML terminam com **.html**. **Exemplo: index.html**, **foo.bar.html** e **etc.hml** *(existem também arquivos html que terminam com .shtml e outras extensões malucas)*. Observe que esses arquivos podem estar tanto no computador do usuário que usa o navegador quanto em outros computadores: o navegador é capaz de abrir esses arquivos desde que eles estejam acessíveis - as tais páginas web.

3. SUA PRIMEIRA PÁGINA EM HTML

Uma página HTML é dividida em duas partes, a **cabeça** e o **corpo**. Na **cabeça** *(ou cabeçalho)* são definidos os atributos principais do documento, como o título e as palavras-chave. O **corpo** contém a parte visível do documento, i.e, aquela que você verá processada em seu navegador. Existe ainda uma região da página que está tanto fora da cabeça quanto do corpo *(não! não é a falta de juízo!)*, mas não iremos entrar nesse mérito.

3.1 - O que está dentro de um arquivo em HTML?!

- Um arquivo HTML é constituído por textos que definem os elementos da linguagem HTML usando *"etiquetas de marcação"*;
- As etiquetas de marcação dão instruções ao navegador sobre a estrutura do documento e sobre a forma de como a página deve ser apresentada graficamente;
- Os arquivos HTML podem ser escritos usando um simples editor de textos e seus nomes devem possuir a extensão **.html**

3.2 - Experimente você mesmo

Bom, se você utiliza o Windows *(95, 98, 2000, XP, etc)*, inicie o Bloco de Notas *(Notepad).*

Agora digite o seguinte texto:

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

Salve este arquivo com o nome **index.htm**

Abra o seu navegador. Agora abra o arquivo que você salvou acima, chamado **index.htm** utilizando as opções do menu ou arrastando o ícone do arquivo para dentro da janela do navegador. Observe o resultado.

3.3 - Explicação do Exemplo

A primeira tag em seu documento HTML é **<html>**. Esta tag define o início de um documento HTML e indica ao navegador que todo conteúdo posterior deve ser tratado como uma série de códigos HTML. A última tag em seu documento deverá ser **</html>**. Esta tag indica ao navegador que é o fim de seu documento HTML.

O texto entre as tags **<head>** ... **</head>** é a informação do cabeçalho. Nenhuma informação contida no cabeçalho é exibida na janela do navegador.

O texto entre as tags **<title>** ... **</title>** é o título de seu documento. O título será exibido na legenda do navegador, na parte de cima do browser.

O texto entre as tags **<body>** ... **</body>** são as informações que serão exibidas na página.

O texto entre **** ... **** ficará com o estilo Negrito *(Bold)*

3.4 - Devemos usar a extensão .htm ou .html?

Os nomes dos arquivos escritos em HTML devem ter a extensão **.html**, mas a extensão **.htm** ainda é utilizada. Este fato é uma herança dos tempos *(pré-históricos no que diz respeito à Internet)* do MS-DOS e do Windows 16 bits, em que os nomes dos arquivos tinham no máximo 8 caracteres e as suas extensões não podiam ter mais de 3 caracteres.

Essas deficiências, que no passado obrigaram a usar a extensão **.htm** em vez de **.html** já foram eliminadas. Por isso devemos usar a extensão **.html**, a não ser que exista uma boa razão para você estar utilizando **.htm** *(pouco provável).*

3.5 - Editores "Puros" versus WYSIWYG

Existem duas formas de se criar um texto formatado através de linguagens de marcação. A primeira consiste em escrever o texto, usando as instruções da linguagem, num editor de texto puro. Em seguida, usa-se o processador de texto para produzir o texto formatado.

A outra maneira é usar um editor de textos **WYSIWYG** (*What You See Is What You Get - O Que Você Vê É O Que Você Tem*). Apesar da sigla ser comprida, o conceito é simples: esse tipo de programa é composto por um editor de texto que também é um processador de textos formatados. A diferença aqui é que o texto que o usuário está editando e visualizando não é o texto puro, mas sim o texto já formatado graficamente, ou seja, **o que você vê é o que você tem.**

Você provavelmente já usou editores desse tipo. Os editores como o *Word*, o *AbiWord* e o *OpenOffice Writer* são **WYSIWYG** e os documentos que eles geram utilizam **Linguagens de marcação.**

3.6 - FAQ (Perguntas Freqüentes)

Depois que eu editei meu arquivo HTML, eu não consigo visualizar o resultado em meu navegador. Por quê?!
Verifique se você salvou o arquivo com o nome correto e que sua extensão seja **.htm**. Confira também em sua barra de endereços do navegador, verifique quanto ao diretório se está correto.

Qual navegador eu devo utilizar?!
Você pode utilizar qualquer browser, como o Internet Explorer, Mozilla, Opera, etc... particularmente recomendo o Firefox e o Internet Explorer.

Porque utilizamos tags com letras minúsculas?!
Em HTML os nomes das tags e elementos podem ser escritos tanto com letras maiúsculas quanto com letras minúsculas, tanto que **<B>** é a mesma coisa que **<b>**. Se você observar em tutoriais encontrados pela Web, vai notar que os mais antigos geralmente utilizam letras maiúsculas para escrever os nomes das tags, mas os mais modernos utilizam exclusivamente letras minúsculas. Nesta minha apostila utilizo sempre letras minúsculas porque existe uma razão muito forte para isso.

A **nova geração do HTML** é uma aplicação do **XML** e é designada por **XHTML**. O **XHTML** é a melhor linguagem para se criar páginas para a Web, mas é mais restrita do que o HTML *("rouba" algumas das liberdades que o HTML oferece)*. Ao contrário do que acontece em HTML, em **XML** as etiquetas **<B>** e **<b>** representam elementos diferentes, visto que em **XHTML** foi adotado uma convenção segundo a qual **todas as etiquetas devem ser escritas com letras minúsculas**. Por este motivo é extremamente recomendável que se escreva todas as etiquetas **com letras minúsculas**. Deste modo, você estará adquirindo bons modos e quase não terá trabalho de converter suas páginas HTML para XHTML.

4. ELEMENTOS E ATRIBUTOS

4.1 - Elementos

Um elemento é uma estrutura semântica, composta de **tag de abertura**, **conteúdo** e **tag de fechamento.**

Os documentos HTML são simples arquivos de texto que contêm *"tags de marcação"*. Essas etiquetas definem os elementos da linguagem HTML e os seus conteúdos. A lista seguinte indica algumas de suas características:

- As "tags de marcação" do HTML são usadas para definir os elementos.
- As tags HTML escrevem-se utilizando os caracteres **<** e **>** , entre eles o nome do elemento e os seus atributos.
- A primeira tag do par é a tag de início *(ou de abertura)* e a segunda do par é a tag de fim *(ou de fechamento)*.
- Tudo o que se encontrar entre as tags de início e de fim fazem parte do conteúdo do elemento.
- Em XHTML as tags devem ser escritas sempre com letras minúsculas, e as tags **<b>** e **<B>** não representam o mesmo elemento.

Você se lembra de nosso primeiro exemplo mostrado?!

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

Esta parte abaixo é um elemento HTML:

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

Repare alguns aspectos do código acima:

- Este elemento **inicia** com a tag:
- O **conteúdo** do elemento é este: **Este texto está em negrito**
- O elemento **termina** com a tag final:

O propósito da tag **** é colocar o conteúdo do elemento HTML em negrito *(bold)*

Um exemplo mais complexo

Este exemplo abaixo também é um elemento do HTML *(mais complexo)*:

```
<body>
Esta é minha primeira página. <b>Este texto está em negrito</b>
</body>
```

Este elemento HTML **inicia** com tag **<body>** e **termina** com a tag **</body>**

O propósito da tag **<body>** é definir o conteúdo principal, o corpo do documento.

4.2 – Atributos das Tags

Tag é um código usado para marcar o início e, onde for requerido, o fim de um elemento HTML. Há, como exposto acima, tags de abertura e de fechamento. Uma tag de abertura é representada por **sinal de menor** (**<**), **um nome de elemento HTML**, e um **sinal de maior** (**>**) *(ex.)* e deve ser colocada imediatamente **antes do início** do conteúdo do elemento. Uma **tag de fechamento** se diferencia de uma tag de abertura apenas por uma barra (**/**) **antes do nome** do elemento *(ex.)* e deve ser colocada imediatamente **após o fim** do conteúdo do elemento.

Os **Atributos** servem para definir uma propriedade de um elemento HTML. Os atributos HTML devem ser colocados sempre na **tag de abertura**, logo após o **nome do elemento**, precedido de um **espaço** e é composto de um **nome de atributo**, um **sinal de igual** (**=**) e um **valor de atributo**, cercado por **aspas duplas** (**"**) ou simples (**`**)

A tag **<body>** define o corpo *(body)* de uma página HTML. No exemplo seguinte adicionei o atributo **bgcolor** *(que significa "background color", ou cor de fundo)* para indicarmos que queremos que a página fique com uma cor vermelha em seu plano de fundo.

```
<html>
<body bgcolor="red">
  Esta é a minha primeira página da web.
  <b>Este texto está em negrito</b>
</body>
</html>
```

Vamos considerar agora a tag **<table>**, que define um elemento de uma tabela. Ao juntarmos o atributo **border** *(que significa borda)* com o valor

apresentado a seguir, estamos dizendo ao navegador para não colocar as bordas da tabela *(<table border="0"> indica uma espessura nula para a linha de contorno da tabela):*

```
<html>
<body>
<table border="0">
<tr>
<td>
 Esta é a minha primeira tabela.
</td>
</tr>
</table>
</body>
</html>
```

Já no exemplo abaixo estamos dizendo ao navegador para desenhar uma linha de contorno com espessura 2 *(border="2")*

```
<html>
<body>
<table border="2">
<tr>
<td>
 Esta é a minha segunda tabela.
</td>
</tr>
</table>
</body>
</html>
```

Os atributos sempre entram em pares nome/valor *(name/value),* assim: **name="value"**

Nota 1: Os atributos só podem aparecer nas tags de início. É proibido colocar atributos nas tags de fechamento.

Nota 2: Para garantir a compatibilidade com a linguagem XHTML, utilize letras minúsculas para escrever o nome dos atributos, e **sempre** coloque os valores entre aspas.

4.2.1 - Atributos podem ser aplicados à maioria das tags

Você normalmente usará atributos com mais freqüência em algumas tags, tais como a tag body e raramente usará em outras, como por exemplo, a tag br que é um comando para pular de linha e não precisa de nenhuma informação adicional.

Assim como existem muitas tags, também existem muitos atributos. Alguns atributos são empregados em tags específicas enquanto outros servem para várias tags. E vice-versa: algumas tags podem conter somente um tipo de atributo, enquanto outras podem conter vários tipos.

Isto pode parecer um pouco confuso, mas à medida que você for praticando vai constatar que tudo é fácil e lógico, bem como vai verificar as inúmeras possibilidades que os atributos oferecem.

4.3 - Devemos usar aspas ou plicas/apóstrofos (escrevemos "texto" ou 'texto')?

Acabamos de ver que os valores dos atributos devem ser sempre colocados entre aspas. Normalmente utilizamos as aspas normais (**"**), mas os apóstrofos (**'**) também são permitidos.

Em alguns casos, o valor do atributo contém o próprio caractere aspas. Numa situação dessas, devemos usar apóstrofos *(que aqui funcionam como aspas simples)* para colocar em torno do valor do atributo, assim:

```
alt='Ele disse: "Não!" '
```

4.4 – Notas Rápidas

- As Tags HTML são utilizadas para marcar elementos HTML. Elas estão cercadas pelos dois caracteres **<** *(menor que)* e **>** *(maior que)*
- As tags normalmente aparecem em pares, como **** e ****
- A primeira tag em um par é sempre a tag de início, a segunda tag do par é a tag de fim
- O texto entre o começo e o fim da tag é o conteúdo do elemento.
- HTML não é *sensitive*, ou seja, **** é a mesma coisa que **<B>**
- Se por acaso você escreveu os códigos HTML errado - por exemplo **>b>** invés de **** - não se desespere, pois o máximo que pode acontecer é o seu navegador interpretar sua página de forma diferente do esperado e desenhá-la de um jeito maluco.

5. O CABEÇALHO DE UM DOCUMENTO HTML

5.1 - O Elemento <head> (cabeçalho do documento)

O elemento **<head>** contém informação de caractere geral, também designada por meta-informação, sobre o conteúdo do documento e sobre a forma como ele deve ser apresentado.

Podemos dizer que o termo meta-informação (*ou meta-dados*) significa dados que descrevem outros dados ou informações.

5.1.1 - A Informação Contida no Elemento <head>

Os elementos contidos dentro do elemento **<head>** não são exibidos na tela do navegador.

O padrão HTML estabelece que só um pequeno número de elementos pode aparecer dentro do cabeçalho. Eles são: **<base>**, **<link>**, **<meta>**, **<title>**, **<style>** e **<script>**.

A construção seria o seguinte:

```
<head>
  <p>Aqui temos algum texto</p>
</head>
```

Nesta situação acima, o navegador pode reagir de duas formas:

- Apresentar o texto porque ele se encontra dentro de um elemento <p>
- Esconde o texto porque ele pertence ao cabeçalho.

Se você colocar um elemento não autorizado, como <h2> ou <p>, dentro do cabeçalho, a maioria dos navegadores vai escrever este elemento na página.

Aparentemente, as pessoas que são responsáveis pela concepção dos browsers acham que este gênero de erros é aceitável. Esta e outras deficiências dos browsers são parcialmente responsáveis pelas más práticas de codificação adquiridas por muitos criadores de páginas HTML. Uma das razões que levaram à criação da linguagem XHTML foi a necessidade de acabar com estas situações de uma vez por todas.

5.1.2 - Elementos de cabeçalho (<head>)

Elemento	Descrição
<head>	Contém informação importante a respeito do documento mas que não deve ser apresentada no corpo da página
<title>	Define o título da página
<base>	Define um URL base comum para todas as ligações relativas da página
<link>	Faz referência a um recurso externo e estabelece a ligação com ele
<meta>	Dá informação sobre aquilo que o documento contém

5.1.3 - A declaração DOCTYPE

A declaração **DOCTYPE** serve para indicar o DTD a usar para validar na página. Quando utilizada, deve aparecer logo no início da página, antes do elemento **<html>**

Declaração	Descrição
<!DOCTYPE>	Define o tipo de documento. Deve ser colocada antes de qualquer elemento pelo que fica mesmo antes do elemento <html>

5.2 - O elemento <meta>

O elemento **<meta>** contém informação de caractere geral *(meta-informação)* sobre o documento e deve ser colocado dentro do elemento **<head>**. Sua finalidade é fornecer informação que descreve o documento.

5.2.1 - Palavras-chave para os motores de pesquisa

Durante alguns anos a utilização mais freqüente da informação fornecida pelo elemento **<meta>** foi a criação de índices para sites de busca. Atualmente somente alguns buscadores ainda usam esta informação para indexar páginas, e os de maior sucesso ignoram este elemento. No entanto, há partes que

continuam a ser tidas em consideração pelos agentes (*robots*) dos buscadores, como por exemplo, as indicações dadas sobre as pastas em que não deve ser feita qualquer indexação.

Alguns sistemas de busca (*não muitos*) usam a informação contida nos elemento meta para indexar as páginas. No fragmento de código seguinte, o elemento **<meta>** contém uma breve descrição da página:

```
<meta name="description" content="Tutoriais e referências técnicas de HTML, CSS, JavaScript, XML, XSLT, SVG">
```

Na parte do código abaixo o elemento **<meta>** contém uma palavras-chave para indexar a página:

```
<meta name="keywords" content="HTML, DHTML, CSS, XML, XHTML, JavaScript, XSLT, SVG">
```

Como acabamos de ver, os nomes dados ao atributo **name** indicam claramente a finalidade da informação contida no elemento **meta**.

Infelizmente, muitos criadores de páginas para a Web abusaram do elemento meta e usaram de forma contrária à sua filosofia para enganar os motores de pesquisa. Em conseqüência disso, os sites de busca passaram a ignorar cada vez mais o elemento **<meta>**, o que acabou dificultando todos aqueles que pretendiam usá-lo de forma correta.

5.2.2 - Valores desconhecidos para o atributo name do elemento <meta>

Algumas vezes encontramos situações em que o atributo **name** do elemento **<meta>** contém um valor desconhecido, como no exemplo seguinte:

```
<meta name="security" content="low">
```

Numa situação destas devemos interpretar o elemento **meta** como contendo informação que é específica do website. Essa informação pode ser importante para o autor da página, mas provavelmente é irrelevante para os visitantes. É possível que essas informações sejam úteis para algum software que leia a página.

6. MODIFICANDO O CORPO DO DOCUMENTO

Como dito anteriormente, o elemento **body** engloba o corpo do seu documento HTML. Ele possui muitos atributos que possibilitam modificar a aparência da sua página, como cor de fundo ou das letras. Esses atributos também podem ser utilizados junto com o atributo **td**. Aqui mostraremos apenas os atributos que interferem nas cores da sua página:

- **bgcolor:** define a cor de fundo de um documento
- **text:** a cor do texto
- **link:** a cor dos links
- **alink:** a cor dos links ativos atualmente *(a página que você está visitando)*
- **vlink:** cor dos links já visitados

Os argumentos são o nome ou o número de uma cor, exatamente como no caso do atributo **color** usado no elemento **font**. Por exemplo:

```
<html>
<head> <title>Isso é uma confusão!</title></head>
<body bgcolor="black" text="gray" link="red" alink="white">

Isso é uma página em html.

</body>
</html>
```

Essa é uma página HTML onde o fundo será preto, com letras cinzas, links em vermelho e links ativos em branco. Essas são as definições globais de cor para seu documento, e a qualquer instante você pode mudar as cores com o elemento **font**.

7. ELEMENTOS BÁSICOS DA LINGUAGEM HTML

7.1 - Cabeçalhos

Os cabeçalhos *(também chamados de Headings)* servem para criar títulos diferenciar as seções da sua página. Eles possuem seis valores diferentes, sendo que a de valor 1 é a que possui a maior fonte e a de valor 6 possui a menor fonte.

Por exemplo, usando o seguinte código:

```
<h1> Título 1 </h1>
<h2> Título 2 </h2>
<h3> Título 3 </h3>
<h4> Título 4 </h4>
<h5> Título 5 </h5>
<h6> Título 6 </h6>
```

No código acima, o HTML automaticamente solta uma linha em branco entre um título e outro.

7.2 - Parágrafos

Os parágrafos são definidos com a tag **<p>**

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

No código acima, o HTML automaticamente acrescenta uma linha em branco antes e depois de um parágrafo.

7.2.1 - Alinhamentos de parágrafo

Você já deve ter visto que em certos documentos o texto aparece ora alinhado à direita, ora à esquerda, no centro ou então ocupando uniformemente o espaço da página *(texto justificado)*. Isso pode ser obtido facilmente em HTML. Veja a tabela abaixo:

ELEMENTO	DESCRIÇÃO
<p align="left">	Alinha o texto à esquerda

<p align="right">	Alinha o texto à direita
<p align="center">	Alinha o texto centralizado
<p align="justify">	Alinha o texto justificado

Note que entre os delimitadores **<** e **>** não encontra-se apenas o elemento **<p>**. Além dele, existe o texto **align="alinhamento"**. Dizemos que **align** é um atributo do elemento **p** e alinhamento é o valor desse atributo. No caso de **<p align="justify">**, o valor do atributo **align** *(que significa alinhamento em inglês)* é **justify** *(justificado)*.

Os atributos, as palavras que seguem o elemento, informam o navegador a respeito das propriedades que os elementos podem assumir. Confuso?! Pode parecer um pouco confuso, mas não é. Como disse a Toya, "*Um disco de banda punk pode ter diversas propriedades ou 'atributos' - a cor do disco, tamanho, velocidade, etc, tudo isso são atributos.*" E eu digo mais: se você coloca o disco num toca-discos, você pode ouvi-lo em duas ou no máximo três velocidades. Às vezes dá pra ouvir ao contrário. Mas se você arremessar seus discos do Fofão, garanto que eles poderão girar em muitas outras velocidades. Assim é o HTML, com muitas opções para você exibir seu texto.

No HTML, os valores dos atributos podem ser definidos da forma **atributo=valor** ou **atributos="valor"** *(como em align="justify")*, que é mais recomendada.
Como você viu, nem todos os códigos em HTML necessitam de atributos, em especial os comandos de modificação de texto como negrito, itálico, sublinhado, etc.

Resumindo, os atributos definem características adicionais aos elementos. Nas seções seguintes veremos elementos com vários atributos. Por exemplo, não adianta dizer ao seu navegador pra por um link em alguma parte do seu texto, é preciso dizer ao navegador, por exemplo, pra que lugar esse link aponta. É isso o que um atributo faz.

7.3 - Criando uma divisão

Existem momentos em que queremos que vários parágrafos possuam um mesmo valor de atributo - *centralizado, por exemplo*. Ao invés de escrevermos **align="justify"** a cada abertura de novo parágrafo, podemos usar o elemento **div**, que cria uma **"divisão"** no documento na qual alguns atributos são preservados. Vejamos o exemplo:

```
<div align="center">
 <p>
 Parágrafo 1
 Parágrafo 1
 Parágrafo 1
 </p>
 <p>
 Parágrafo 2
 Parágrafo 2
 Parágrafo 2
 </p>
 <p>
 Parágrafo 3
 Parágrafo 3
 Parágrafo 3
 </p>
</div>
```

Experimente mudar os atributos do alinhamento pra ver o que acontece!! Na parte de **Folhas de Estilo** mostrarei como utilizar o elemento **div** para criar seções lógicas em documentos.

7.4 – Quebras de Linha

A tag **
** é utilizada quando você quer terminar uma linha, mas não quer começar um novo parágrafo. Com este comando você faz com que ocorra uma quebra de linha, onde você posicionar a tag.

Note que para o elemento **
** não existe o comando **</br>**, isto é, a quebra de linha não age numa região de texto delimitada, mas sim num ponto do texto. Complicado?! Então vamos lá, mais uma vez explicando de uma outra forma: comandos como ****, **** e **<u>** agem sobre uma região do texto e precisam ser fechados com seus respectivos ****, **** e **</u>**, pois do contrário esses comandos agirão até o fim do documento. Já o comando de quebra de linha, **</br>**, e alguns outros atuam somente no ponto onde eles aparecem.

Veja o exemplo abaixo da utilização da tag **
**

```
<p>Este <br> é um pará<br>grafo com quebra de linha</p>
```

A tag **
** é uma etiqueta vazia, ela não possui tag final.

7.5 – Comentários

A tag de comentário é utilizada para inserir comentários no código fonte HTML. Todo tipo de comentário é ignorado pelo navegador, isto é, ele não será exibido na tela. Você pode utilizar esta tag para explicar seu código fonte ou parte dele, que poderá ajudá-lo quando você estiver editando seus códigos posteriormente.

```
<!-- Este é o comentário -->
```

Note que você precisa de um ponto de exclamação depois do parênteses de abertura, e não antes do parênteses final.

7.6 – Informações Úteis

1) Quando você escrever um texto em HTML, esteja ciente de que o texto que você está visualizando não será exibido igualmente em todos os navegadores. Algumas pessoas possuem computadores com resoluções maiores, outros menores, tudo isso influi na questão da exibição da página, o que pode acontecer são os textos e as janelas que podem ser redimensionados.

2) As diferenças nos tamanhos das janelas dos navegadores fazem com que o número de caracteres que cabem numa linha varie muito. Por esse motivo não seremos capazes de controlar nem o número de linhas nem os locais em que acontecem as mudanças de linha. Nunca tente formatar o texto inserindo espaços ou linhas vazias, porque os resultados não serão aqueles que você esperava.

3) O HTML quando encontra dois ou mais espaços seguidos, ele trata-os como se fosse um único espaço. Quando você escreve uma seqüência de espaços seguidos, o resultado é o mesmo de escrever um único espaço. Em HTML, as teclas **Tab** e **Enter** equivalem a um espaço.

4) Sempre que você quiser inserir linhas em branco, utilize a tag **
**. Existem pessoas que utilizam parágrafos vazios para obter o mesmo resultado, mas isso está errado. A tag ** ** deve ser usada apenas para definir parágrafos, e o elemento **
** não deve ser usado, por exemplo, para criar listas, pois existem tags concebidas especificamente para isso. Sempre que você precisar de obter uma formatação especial, você deve usar o elemento que foi criado para esse efeito.

5) Em muitas páginas, os parágrafos estão escritos sem a etiqueta de fechamento *(** **)* Apesar dos navegadores aceitarem esta omissão, tenha sempre atenção para fechar todos os elementos que requerem uma tag de fechamento. Se você tentar validar uma página que contenha estes erros, verá que ela não passará no teste de validação.

6) Os elementos ** ** e **<h1>** ... **<h6>** o navegador adiciona automaticamente uma linha em branco antes do início e outra depois do fim.

7.7 – Lista de elementos básicos do HTML

TAG	DESCRIÇÃO
<html>	Define um documento HTML
<body>	Define o conteúdo principal, o corpo do documento
<h1> até <h6>	Define títulos, de 1 a 6
	Define parágrafo
 	Insere linhas em branco, quebra de linha
<hr>	Insere uma linha horizontal
<!-->	Define um comentário

8. FORMATAÇÃO DE TEXTO

A linguagem HTML define vários elementos para se formatar um texto, como por exemplo, escrever em negrito, itálico, sublinhado, etc. O exemplo abaixo mostra alguns deles:

```
<html>
<body>
  Este exemplo tem texto em <i>itálico</i>, <b>negrito</b>,
  <em>enfatizado</em>, <u>sublinhado</u> e tipo
  <code>código de computador</code>
</body>
</html>
```

8.1 – Elementos para formatação de texto

TAG	DESCRIÇÃO
<b>	Define seu texto em negrito
<big>	Define seu texto grande
<i>	Define seu texto em itálico
<small>	Define seu texto pequeno
	Define seu texto forte

<sub>	Define seu texto subscrito *(ex: H_2O)*
<sup>	Define seu texto sobrescrito *(ex: 15^2)*
<ins>	Define texto inserido
<del>	Define texto apagado
<s>	Desuso. Utilize <del> ao invés
<strike>	Desuso. Utilize <del> ao invés
<u>	Desuso. Utilize <styles> ao invés

8.2 – Elementos para o "output de código de computador"

TAG	DESCRIÇÃO
<code>	Define códigos de texto
<kbd>	Define o texto com uma fonte especial determinada pelo navegador
<samp>	Define código de computador de amostra
<tt>	Define o texto utilizando uma fonte de tipo e largura definidas pelo navegador
<var>	Define variáveis
<pre>	Define o texto pré-formatado
<listing>	Desuso. Utilize <pre> ao invés
<plaintext>	Desuso. Utilize <pre> ao invés
<xmp>	Desuso. Utilize <pré> ao invés

8.3 – Elementos para citações e listas de definições

TAG	DESCRIÇÃO
<abbr>	Define uma abreviação
<acronym>	Define uma sigla
<address>	Define um elemento de endereço
<bdo>	Define a direção do texto
<blockquote>	Define uma longa citação
<q>	Define uma pequena citação
<cite>	Define a citação
<dfn>	Define um termo de citação

9. LIGAÇÕES DE HIPERTEXTO ("LINKS")

Essa é uma das seções mais importantes desta apostila, pois afinal estamos falando de hipertexto. Queremos que uma página HTML possa fazer referência a outras páginas html ou para qualquer outro tipo de arquivo que se encontre em outros lugares da web. No jargão da internet, você quer ligar (link) parte da sua página com outras.

Isso é feito de modo muito simples com o elemento a, como segue:

```
<a href="http://www.midiaindependente.org">CMI Brasil</a>
```

Como você deve ter percebido, o atributo **href** *(Hipertext Reference / Referência de Hipertexto)* indica o endereço da página que você quer "linkar" (ligar) ao seu documento. No exemplo acima, o texto **CMI Brasil** aparece como uma referência para o link. Se você clicar sobre esse texto, seu navegador automaticamente tentará abrir a página http://www.midiaindependente.org

Os endereços são escritos na forma de URL *(Uniform Resource Locators)*, que é uma maneira que inventaram para se especificar o caminho até se chegar num arquivo que esteja nalgum local da internet.

O formato da URL é:

```
protocolo://nome-do-computador/pastas/arquivo
```

O **protocolo** indica como o navegador irá buscar o arquivo. **Nome do computador** é aquele endereço do tipo www.ninguem.com.br e as **pastas** são os diretórios dentro desse computador onde estão os arquivos. Uma URL de um artigo do site da CMI Brasil é, por exemplo, http://www.midiaindependente.org/pt/blue/2004/09/290669.shtml, onde **http://** é o **protocolo**, **pt**, **blue**, **2004** e **09 são pastas** *(uma fica dentro da outra)* e **290669.shtml** é o **nome do arquivo.**

```
http://
  \--- www.midiaindependente.org
              \--- pt
                   \-- blue
                        \--- 2004
                             \--- 09
                                  \--- 290669.shtml
```

Lembre-se também que quando você acessa o endereço http://www.ninguem.com.br, por exemplo, está na verdade vendo o arquivo http://www.ninguem.com.br/index.html. Os arquivos do tipo **index.html** sempre são procurados pelo navegador quando você solicita apenas o nome do site ou uma pasta de um computador.

Em html, existem três formas possíveis de se fazer links: os **links internos**, os **links locais** *(ou relativos)* e os **links externos.**

9.1 - Links internos

Links internos são aqueles que ligam uma seção de uma página com uma outra seção da mesma página. Por exemplo, se eu quiser fazer uma referência para a seção Hipertexto dessa nossa apostila, basta que eu primeiro defina um nome para a seção Hipertexto e então adicione um link pra ela. Isso é feito da seguinte forma:

1 - Vou até o início da seção Hipertexto e adiciono o seguinte código, ao invés de simplesmente escrever o título da seção:

```
<a name="Hipertexto">Hipertexto</a>
```

2 - Adiciono o link pra essa seção com o seguinte código,

```
<a href="#Hipertexto">Vá para a seção hipertexto</a>
```

O atributo **name** serve apenas para dar um nome para uma posição de uma página html. O valor de **name** pode ser qualquer um *(nesse caso eu coloquei "Hipertexto").*

Quando você fizer uma referência a um link interno utilizando o atributo **href**, você precisa necessariamente adicionar o caractere **#** *(jogo da velha)* antes

do nome da sua seção - *no nosso caso,* ***#Hipertexto*** - pois do contrário seu navegador tentará abri-lo como um link local.

9.2 - Links locais ou relativos

Links locais são aqueles que referenciam um arquivo que esteja no mesmo computador que a sua página. **Links locais** não contém o http:// no início do endereço. Por exemplo, se no meu documento html eu quiser fazer um link para o arquivo **links.html** que está na mesma pasta do meu documento, não preciso digital o endereço completo, mas apenas:

```
<a href="links.html">Acesse os links!</a>
```

Esse tipo de argumento para o **href** também é chamado de **links relativos**. Se você quiser linkar uma página que esteja do diretório *(pasta)* superior da sua página, use dois pontos e uma barra antes do nome do arquivo:

```
<a href="../links.html">Acesse os links!</a>
```

A vantagem de você usar links relativos é que você pode mover todo o seu site para um novo endereço na internet e não precisar reeditar todos os links que apontam para páginas do seu próprio site.

9.3 - Links externos

Links externos são aqueles que podem apontar para qualquer arquivo disponível na internet, como por exemplo:

```
<a href="http://pt.wikipedia.org">Acesse a Wikipedia!</a>
```

Os **links externos** precisam necessariamente conter o prefixo http:// e o endereço completo do arquivo.

9.4 - Alvo (target)

Um atributo interessante para o elemento **<a href>** é o **target**, que permite que o link seja aberto numa outra janela do seu navegador.

```
<a href="http://pt.wikipedia.org" target="_blank">Acesse a Wikipedia!</a>
```

O argumento do atributo **target** é o nome da janela do navegador que abrirá o link. Se você quiser que o link seja aberto numa nova janela, simplesmente escolha qualquer nome como argumento.

9.5 – Elementos para fazer ligações

ELEMENTO	DESCRIÇÃO
<a>	Define uma âncora ou uma ligação de hipertexto

10. LISTAS

A linguagem HTML contém elementos que permitem criar **listas de definições**, **listas ordenadas** e **listas não ordenadas.**

10.1 - Listas Não Ordenadas

Uma **lista não ordenada** contém vários itens marcados todos com o mesmo símbolo *(normalmente um círculo pequeno ou um quadrado pequeno).*

Para criar uma lista não ordenada, utilizamos o elemento **<ul>** *("unordered list").* Dentro desse elemento, os vários itens são criados com o elemento **<li>** *("list item").*

O exemplo seguinte mostra uma lista simples:

```
<ul>
 <li>Rum</li>
 <li>Bagaço</li>
</ul>
```

Este é o aspecto de como vai ficar em seu navegador:

- Rum
- Bagaço

Dentro de uma lista não ordenada podemos colocar parágrafos, quebras de linha, imagens, outras listas, etc.

10.2 - Listas Ordenadas

Uma **lista ordenada** contém vários itens numerados e é criada com o elemento **<ol>** *("ordered list")*. Os itens da lista definem-se com o elemento **<li>** *("list item")*.

```
<ol>
 <li>Rum</li>
 <li>Bagaço</li>
</ol>
```

Este é o aspecto de como vai ficar em seu navegador:

1. Rum
2. Bagaço

Dentro de uma lista ordenada podemos colocar parágrafos, quebras de linha, imagens, outras listas, etc.

10.3 – Listas de Definições

Uma **lista de definições** não é constituída por uma série de itens, mas sim por vários termos acompanhados de descrições de seus significados.

As listas de definições são criadas com o elemento **<dl>** *("definition list")* O nome de cada termo fica dentro de um elemento **<dt>** *("definition term")* e a sua descrição fica no elemento **<dd>** *("definition description")*

```
<dl>
  <dt>Rum</dt>
  <dd>Bebida espirituosa muito apreciada pelos piratas do Caribe</dd>
  <dt>Bagaço</dt>
  <dd>Bebida com elevado teor alcoólico. A sabedoria popular
     atribui-lhe fortes propriedades terapêuticas.</dd>
</dl>
```

Este é o aspecto de como vai ficar em seu navegador:

Rum
: Bebida espirituosa muito apreciada pelos piratas do Caribe

Bagaço
: Bebida com elevado teor alcoólico. A sabedoria popular reconhece-lhe fortes propriedades terapêuticas.

Dentro de um elemento **<dd>** podemos colocar parágrafos, quebras de linha, imagens, outras listas, etc.

10.4 – Elementos para Listas

ELEMENTO	DESCRIÇÃO
	Define uma lista ordenada
	Define uma lista não ordenada
	Insere um item na lista
<dl>	Insere uma lista de definições
<dt>	Apresenta a definição de um termo
<dd>	Insere a definição de um termo
<dir>	Desuso. Utilize
<menu>	Desuso. Utilize

11. IMAGENS

Além de manipular texto, o html também serve para exibir imagens de uma maneira muito simples, usando para isso o elemento **img**:

```
<img src="emma.jpg" width="221" height="300">
```

O atributo **src** - *source, fonte em inglês* - dá o **nome do arquivo da imagem** e é o único atributo obrigatório para o elemento **img**. Os atributos para a largura ***(width)*** e a altura ***(height)*** não são necessários, mas serão muito úteis quando as pessoas estiverem navegando em seu site: quando seu navegador abre um arquivo html e encontra o elemento **img**, ele usará o respectivo atributo **src** como o endereço onde está o arquivo de imagem a ser exibido. Em outras palavras, o elemento **img** apenas passa uma referência do arquivo de imagem ao navegador, ou seja, você não está colocando esse arquivo de imagem dentro do seu arquivo html, você está apenas colocando em seu arquivo html uma referência a esse arquivo de imagem. O navegador, por sua vez, começa a baixar essa imagem a partir desse endereço atributo enquanto termina de exibir o documento.

Dependendo do tipo de arquivo de imagem, ela só será exibido pelo navegador quando terminar de ser baixada. Se isso ocorrer e você não utilizou

os atributos **width** e **height** no seu documento, o navegador só reservará espaço na tela do seu computador para a exibição da imagem quando ela estiver sido baixada, e duranter esse intervalo a formatação do seu documento pode ficar diferente. Para evitar isso - *somente por uma questão de estilo* - recomendamos que você sempre use os atributos **width** e **height**, que informam ao navegador o tamanho da imagem, antes mesmo dele começar a baixá-la, e assim o espaço na tela do seu computador será apropriadamente reservado para ela.

11.1 - Tamanho de exibição da imagem

Os atributos **width** e **height** merecem um pouco mais de atenção. Como vimos, são os atributos que determinam o tamanho que a imagem será exibida. Existem duas formas de se fazer isso, utilizando o tamanho em pixels *(tamanho absoluto)* e em valores relativos ao tamanho disponível da tela do computador, isto é, em porcentagem.

Um pixel é a menor unidade luminosa da tela do seu computador que os programas podem manipular, e isso depende muito da resolução da tela do seu computador. Um pixel é um quadradinho luminoso. Atualmente as telas de computador tem uma resolução de 800 pixels de comprimento e 600 de altura, às vezes pode ser de 1024 de comprimento e 768 na altura ou resoluções até maiores, de tal modo que nossos olhos já não conseguem distinguir entre pixels contíguos.

O tamanho das imagens computadorizadas também pode ser medido em pixels, que é o tamanho que a imagem vai ocupar na tela do seu computador quando ela for exibida. Para que você descubra qual é o tamanho da sua imagem, você terá que usar algum programa de edição de imagens ou então abri-la diretamente no seu navegador. Experimente ir no menu Arquivo e depois em Abrir, no seu nagevador, ou então digitar a localização da imagem diretamente na barra de endereços. Quando a imagem for exibida, clique sobre ela com o botão direito do seu mouse e em seguida vá em Propriedades. Uma janela com as informações da imagens - *inclusive seu tamanho* - deverá aparecer. De posse desses valores, basta colocar-los dentro da do img. No exemplo anterior, onde usamos o código **width="221" height="300"**, a imagem foi exibida com **221 pixels de largura** e **300 de altura**, coincidentemente sendo seu tamanho original. Poderíamos ter escrito:

```
<img src="emma.jpg" width="22" height="30">
```

Ou seja, a imagem foi mostrada com um tamanho menor do que o original. Também poderíamos usar tamanhos maiores que original... que tal você experimentar algo como **width="2210" height="3000"?!**

O segundo modo de determinar o tamanho de exibição das imagens no navegador consiste em utilizar porcentagem. Por exemplo:

```
<img src="emma.jpg" width="10%" height="10%">
```

Resultará numa imagem exibida com um décimo **(10%)** do espaço disponível da tela do computador.

Nos dois modos de determinar o tamanho - *em pixels e em porcentagem* - podemos escolher valores independentes para a altura e a largura, de tal forma que a imagem fique distorcida. Olha que engraçado:

```
<img src="emma.jpg" width="50" height="400">
```

11.2 - Texto alternativo

Existem ainda outros atributos interessantes para a exibição de imagens. Você pode incluir uma descrição da imagem para que se, por qualquer razão, alguém não consiga ver a imagem ela possa ler uma descrição no espaço vazio. Essa descrição também aparecerá quando você passar o mouse por sobre a imagem.

Você pode adicionar uma pequena descrição desta maneira:

```
<img src="emma.jpg" width="221" height="300" alt="Emma Goldman!">
```

O atributo **alt** *(texto alternativo)* é utilizado para dar a pequena descrição, neste caso **"Emma Goldman!"**.

11.3 - Borda da imagem

Uma borda *(contorno)* da imagem pode ser adicionada com o atributo border:

```
<img src="emma.jpg" width="10%" height="10%" border="1">
<img src="emma.jpg" width="10%" height="10%" border="2">
<img src="emma.jpg" width="10%" height="10%" border="3">
```

O valor do atributo **border** indicará o tamanho dessa moldura.

11.4 - Alinhamento da imagem

É possível ainda colocar imagens e texto um do lado do outro, de diversas formas. Vamos supor a mais simples:

```
<img src="emma.jpg" width="3%" height="3%" border="1">
```

O texto aparece ao lado da imagem.

Esse código pode funcionar muito bem, mas às vezes acontece do texto aparecer abaixo da imagem. Para fazer o posicionamento da imagens e do texto corretamente, basta utilizar o atributo **align**, que funciona de modo análogo ao do alinhamento de paragráfos:

```
<img src="emma.jpg" width="5%" height="5%" border="1" align="right" border="1">
```

O texto vai aparecer à esquerda da imagem *(o atributo **right** faz que a imagem fique à direita da tela)*, mesmo tendo escrito após o código sobre a imagem. Enquanto o texto puder ficar ao lado da imagem, ele ficará. Quando ele não mais couber nesse espaço, ele começará a ser exibido abaixo da imagem, como você pode ver aqui.

Utilizar o alinhamento de imagens é a melhor maneira de garantir que ela aparecerá no local desejado.

11.5 - Espaçamento da imagem

Notou como o texto do exemplo anterior saiu colado à imagem?! É possível definir um espaçamento entre a imagem e qualquer objeto *(texto, imagem, tabelas)* que esteja ao seu lado, acima ou abaixo, usando para isso os atributos **vspace** *(espaçamento vertical)* e **hspace** *(espaçamento horizontal)*.

O valor desses atributos diz ao navegador qual será o espaço em pixels entre a imagem e qualquer objeto:

```
<img src="emma.jpg" width="3%" height="3%" border="1" align="left" hspace="5"
border="1">
```

O texto vai aparecer à direita da imagem, a cinco pixels de distância dela. Deu pra notar a diferença?! Utilizando todos esses atributos para a exibição de imagens você obtém um texto bem diagramado e bonito de ser lido.

Nota: Ao utilizar imagens, lembre-se de que elas podem aumentar consideravelmente o tempo de carregamento para ver o conteúdo de suas páginas, por isso, use-as com cuidado.

11.6 – Elementos relacionados com imagens

ELEMENTO	DESCRIÇÃO
<img>	Insere uma imagem
<map>	Define um mapeamento sobre a imagem *("image map")*
<área>	Define uma área clicável sobre um mapa de imagem

11.7 – Fundos de Página

Uma cor ou uma imagem de fundo bem escolhido podem melhorar o aspecto da página, mas uma má escolha é capaz de causar danos muito graves na legibilidade e no aspecto geral da página.

11.7.1 - O atributo bgcolor do elemento <body>

O elemento **<body>** possui atributos que nos permitem especificar as cores do texto e a cor de fundo. Também podemos usar uma imagem como padrão de fundo.

O atributo **bgcolor** nos permite escolher a cor de fundo da página. O quadro seguinte mostra três formas de indicar a cor de fundo da página usando um código de cor hexadecimal, um código RGB e um nome de cor *(veja mais à frente o tópico sobre cores).*

```
<body bgcolor="#000000">
<body bgcolor="rgb(0,0,0)">
<body bgcolor="black">
```

As linhas apresentadas mais acima usam formas diferentes *(mas todas válidas)* para dar a cor preta ao fundo da página.

Nota: a forma recomendada para formatar o texto e os fundos de página baseia-se em **estilos CSS**. A utilização do atributo **bgcolor** só se justifica se precisarmos de manter a compatibilidade com navegadores antigos que não suportam CSS *(veremos sobre CSS no fim da apostila).*

11.7.2 - O atributo background

O atributo **background** estabelece que o padrão de fundo da página será uma imagem. O valor deste atributo indica o local onde se encontra a imagem. Se as dimensões da imagem forem inferiores às dimensões da página, o navegador repetirá a imagem *(como num chão de mosaicos ou em uma parede de azulejos),* por forma a ocupar todo o fundo da página.

O exemplo abaixo mostra bem como se faz isso:

```
<body background="imagem.jpg">
<body background="http://www.tiagosouza.com/imagem.jpg">
```

O valor do atributo **background** é uma URL *(site)* que define o local onde se encontra a imagem. Na primeira linha mais acima, demos uma URL relativa e na segunda demos uma URL absoluta.

Nota: a forma recomendada para formatar o texto e os fundos de página baseia-se em **estilos CSS**. A utilização deste atributo só se justifica se precisarmos de manter a compatibilidade com navegadores antigos que não suportam CSS *(veremos sobre CSS no fim da apostila).*

11.8 – Dicas

Sempre que utilizar uma imagem de fundo na página, não se esqueça destes aspectos:

- Certifique-se de que o tamanho da imagem *(em KBytes)* não é muito grande, pois em caso afirmativo, o tempo para carregar a página aumentaria consideravelmente.
- Certifique-se de que a imagem de fundo combina bem com a cor do texto.
- Certifique-se de que o fundo combina bem com as outras imagens da página.
- Verifique se a repetição da imagem de fundo em mosaico resulta em um padrão perfeito. Se você perceber algumas falhas aparentes, utilize outra imagem de fundo ou faça uma edição nesta atual.
- Certifique-se de que a imagem não é incômoda e que não desvia a atenção do texto.

Os atributos **bgcolor**, **background** e **text** do elemento **<body>** foram **reprovados** nas recomendações mais recentes da W3C para a linguagem HTML *(HTML 4 e XHTML)*. A forma recomendada para obter os mesmos resultados baseia-se na utilização de estilos CSS. A utilização destes atributos só se justifica se precisarmos de manter a compatibilidade com navegadores antigos que não suportam CSS.

São poucos os sites de qualidade que utilizam imagens de fundo, e aqueles que o fazem, usam fundos que não atrapalham a visualização do site.

12 – CORES

Podemos obter qualquer cor que desejarmos combinando proporções corretas de três cores bases: Vermelho *(Red)*, Verde *(Green)* e Azul *(Blue)*.

12.1 - Formas de exprimir cores

Em CSS a forma recomendada para utilizar cores é usando código *(notações)* hexadecimal. Desta forma as cores exprimem usando três números hexadecimais que definem as quantidades de vermelho, verde e azul que entram na composição de uma determinada cor. O valor mais baixo de uma determinada cor é **0** ***(#00** na notação hexadecimal usada em CSS)* e o valor mais alto é **255** ***(#FF** em código hexadecimal).* Assim, a cor preta tem 0 de vermelho, 0 de verde e 0 de azul, escrevendo na forma **#000000**. Já o branco possui 255 de vermelho, 255 de verde e 255 de azul, sendo seu código **#FFFFFF**. O amarelo forte possui 255 de vermelho, 255 de verde e 0 de azul, sendo seu código **#FFFF00**

A tabela abaixo mostra os resultados de diversas combinações de cores:

COR	CÓDIGO HEXADECIMAL	VALOR RGB (DECIMAL)
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

12.2 – Nomes de cores

A tabela seguinte mostra as 16 cores cujos nomes foram definidos oficialmente pelo W3C. Todos os navegadores reconhecem estes nomes pelo que podem usá-los sem qualquer problema:

Cores com Nomes Atribuídos Oficialmente

Aqua (#00FFFF)	Black (#000000)	Blue (#0000FF)	Fuchsia (#FF00FF)
Green (#008000)	Gray (#808080)	Lime (#00FF00)	Maroon (#800000)
Navy (#000080)	Olive (#808000)	Purple (#800080)	Red (#FF0000)
Silver (#C0C0C0)	Teal (#008080)	White (#FFFFFF)	Yellow (#FFFF00)

Apesar de não estarem definidos oficialmente, os nomes das cores apresentados a seguir são reconhecidos por todos os navegadores relevantes, com exceção das versões mais antigas.

Nota: Estes nomes de cores não estão definidos em nenhum padrão do W3C, apesar de serem reconhecidos pelos principais navegadores, esse reconhecimento não é exigido a nenhum navegador para estar conforme com as recomendações oficiais. É pouco provável que nos anos mais próximos os

navegadores para PDA e telefones móveis consigam reconhecer estes nomes de cor. Se quiser garantir a apresentação correta das cores das suas páginas em todos os navegadores conformes com as normas do W3C, em vez dos nomes apresentados na tabela, utilize os códigos hexadecimais apresentados junto das cores.

COR	CÓDIGO HEXADECIMAL	NOME DA COR
	#F0F8FF	AliceBlue
	#FAEBD7	AntiqueWhite
	#7FFFD4	Aquamarine
	#000000	Black
	#0000FF	Blue
	#8A2BE2	BlueViolet
	#A52A2A	Brown

12.3 – Cores seguras da Web

Problemas causados por um número reduzido de cores

Todos os computadores modernos são capazes de mostrar dezenas de milhar ou milhões de cores em simultâneo. Contudo, até meados da década de 1990 muitos sistemas apenas conseguiam apresentar 256 cores diferentes de cada vez. Esta limitação obrigava os navegadores a trabalharem com uma paleta fixa que continha apenas 256 cores.

Os navegadores eram obrigados a usar apenas 256 cores para simular todas as cores que não conseguiam apresentar. Os efeitos destas aproximações eram visíveis na forma pontos adjacentes com cores diferentes e de manchas de cor. Atualmente estas limitações já quase não existem.

Como acabamos de ver, na primeira metade da década de 1990, a maioria dos computadores eram capazes de apresentar apenas 256 cores diferentes de cada vez. Dessas 256 cores, os sistemas operativos *Windows* e *Apple Macintosh* reservavam 20 cores cada um *(40 no total)* para desenhar suas interfaces gráficas. Assim, de um total de 256 cores possíveis, apenas 216 podiam ser escolhidas livremente com a garantia de poderem ser apresentadas tanto numa máquina Windows como em um Mac. Estas 216 cores receberam a designação de **cores seguras da Web.**

A forma encontrada para limitar as conseqüências resultantes da utilização de uma paleta com apenas 216 cores consiste em usar apenas cores cujos códigos hexadecimais usam apenas combinações dos números indicados na tabela seguinte:

RGB	00	51	102	153	204	255
Hex	00	33	66	99	CC	FF

A tabela abaixo mostra todas as 216 combinações de cores que podemos formar com os valores apresentados na tabela anterior. Estas são, portanto, as 216 cores seguras da Web, que mostramos juntamente com seus códigos hexadecimais *(o caractere # no início foi omitido)*:

000000	000033	000066	000099	0000CC	0000FF
003300	003333	003366	003399	0033CC	0033FF
006600	006633	006666	006699	0066CC	0066FF
009900	009933	009966	009999	0099CC	0099FF
00CC00	00CC33	00CC66	00CC99	00CCCC	00CCFF
00FF00	00FF33	00FF66	00FF99	00FFCC	00FFFF
330000	330033	330066	330099	3300CC	3300FF
333300	333333	333366	333399	3333CC	3333FF
336600	336633	336666	336699	3366CC	3366FF
339900	339933	339966	339999	3399CC	3399FF
33CC00	33CC33	33CC66	33CC99	33CCCC	33CCFF
33FF00	33FF33	33FF66	33FF99	33FFCC	33FFFF
660000	660033	660066	660099	6600CC	6600FF
663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF
669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

Atualmente qualquer computador consegue apresentar milhões de cores diferentes ao mesmo tempo, não havendo necessidade de usar apenas cores seguras da Web. Apesar disso, este tema continua a ser focado em quase todas as introduções à construção de páginas.

12.4 - Formas de exprimir os valores das cores

As cores definem-se usando notação hexadecimal que exprime as quantidades de Vermelho *(Red)*, Verde *(Green)* e Azul *(Blue)* que entram em sua composição. A quantidade mínima de uma cor é **0** *(**#00** em código hexadecimal)* e a quantidade máxima é **255** *(**#FF** em código hexadecimal)* Assim, a cor branca escreve-se na forma **#FFFFFF** e a cor preta na forma **#000000**. Uma forma mais antiga que ainda funciona é a forma decimal. Na forma decimal a cor branca exprime-se como **rgb(255,255,255)** e a cor preta exprime-se como **rgb(0,0,0)**. Entre estes dois extremos, temos toda a gama de cores que podem ser apresentadas em um monitor.

Nada de Vermelho

Se desligarmos completamente a cor Vermelha, ficamos com 65536 cores diferentes que resultam de combinar 256 quantidades de verde com 256 quantidades de azul *(65536 = 256 × 256)*.

A tabela seguinte mostra algumas destas combinações:

003300	006600	009900	00CC00	00FF00
003333	006633	009933	00CC33	00FF33
003366	006666	009966	00CC66	00FF66
003399	006699	009999	00CC99	00FF99
0033CC	0066CC	0099CC	00CCCC	00FFCC
0033FF	0066FF	0099FF	00CCFF	00FFFF

Vermelho ao Máximo

Se colocarmos a quantidade de Vermelho com seu valor máximo, que é **255** *(**#FF** em código hexadecimal)* ficamos ainda com 65536 *(256 x 256)* cores diferentes que resultam de combinarmos todos os valores possíveis de Verde com todos os valores possíveis de Azul.

A tabela seguinte mostra algumas destas combinações:

FF3300	FF6600	FF9900	FFCC00	FFFF00
FF3333	FF6633	FF9933	FFCC33	FFFF33
FF3366	FF6666	FF9966	FFCC66	FFFF66
FF3399	FF6699	FF9999	FFCC99	FFFF99
FF33CC	FF66CC	FF99CC	FFCCCC	FFFFCC
FF33FF	FF66FF	FF99FF	FFCCFF	FFFFFF

Tons de Vermelho

A tabela apresentada abaixo mostra o resultado obtido variando a quantidade de cor vermelha de 0 até 255 ao mesmo tempo que se mantêm em zero as quantidades de verde e de azul. Existem 256 tons diferentes de vermelho puro e muitas outras que contêm misturas de outras cores.

TONS DE VERMELHO	HEXADECIMAL	RGB
	#000000	rgb(0,0,0)
	#080000	rgb(8,0,0)
	#100000	rgb(16,0,0)
	#180000	rgb(24,0,0)
	#200000	rgb(32,0,0)
	#280000	rgb(40,0,0)
	#300000	rgb(48,0,0)
	#380000	rgb(56,0,0)
	#400000	rgb(64,0,0)
	#480000	rgb(72,0,0)
	#500000	rgb(80,0,0)
	#580000	rgb(88,0,0)
	#600000	rgb(96,0,0)
	#680000	rgb(104,0,0)
	#700000	rgb(112,0,0)
	#780000	rgb(120,0,0)
	#800000	rgb(128,0,0)
	#880000	rgb(136,0,0)
	#900000	rgb(144,0,0)
	#980000	rgb(152,0,0)
	#A00000	rgb(160,0,0)
	#A80000	rgb(168,0,0)
	#B00000	rgb(176,0,0)
	#B80000	rgb(184,0,0)
	#C00000	rgb(192,0,0)
	#C80000	rgb(200,0,0)
	#D00000	rgb(208,0,0)
	#D80000	rgb(216,0,0)
	#E00000	rgb(224,0,0)
	#E80000	rgb(232,0,0)
	#F00000	rgb(240,0,0)
	#F80000	rgb(248,0,0)
	#FF0000	rgb(255,0,0)

Tons de Cinza

As cores cinzas obtêm-se combinando quantidades iguais de vermelho, verde e azul. A cor branca corresponde ao cinza mais claro de todos e obtém-se juntando 255 de vermelho, 255 de verde e 255 de azul *(**#FFFFFF** em hexadecimal).* O preto é o cinza mais escuro de todos e obtém-se colocando todas as cores em 0 *(**#000000** em hexadecimal).* Entre estes dois valores extremos, temos 254 graus de intensidade possíveis.

A tabela seguinte dá uma idéia dos tons de cinza que podemos obter:

	RGB(0,0,0)	#000000
	RGB(8,8,8)	#080808
	RGB(16,16,16)	#101010
	RGB(24,24,24)	#181818
	RGB(32,32,32)	#202020
	RGB(40,40,40)	#282828
	RGB(48,48,48)	#303030
	RGB(56,56,56)	#383838
	RGB(64,64,64)	#404040
	RGB(72,72,72)	#484848
	RGB(80,80,80)	#505050
	RGB(88,88,88)	#585858
	RGB(96,96,96)	#606060
	RGB(104,104,104)	#686868

	RGB(112,112,112)	#707070
	RGB(120,120,120)	#787878
	RGB(128,128,128)	#808080
	RGB(136,136,136)	#888888
	RGB(144,144,144)	#909090
	RGB(152,152,152)	#989898
	RGB(160,160,160)	#A0A0A0
	RGB(168,168,168)	#A8A8A8
	RGB(176,176,176)	#B0B0B0
	RGB(184,184,184)	#B8B8B8
	RGB(192,192,192)	#C0C0C0
	RGB(200,200,200)	#C8C8C8
	RGB(208,208,208)	#D0D0D0
	RGB(216,216,216)	#D8D8D8
	RGB(224,224,224)	#E0E0E0
	RGB(232,232,232)	#E8E8E8
	RGB(240,240,240)	#F0F0F0
	RGB(248,248,248)	#F8F8F8
	RGB(255,255,255)	#FFFFFF

12.5 - Mais de 16 milhões de cores diferentes

Combinando as 256 cores de Vermelho com as 256 cores de Verde e as 256 cores do Azul, conseguimos criar mais de 16 milhões de cores diferentes *(256×256×256)*.

Praticamente todos os monitores dos computadores modernos estão preparados para apresentar mais de 16 milhões de cores diferentes. No entanto, é preciso ter em mente que os novos sistemas móveis *(celulares, PDA's, etc)*, **geralmente** possuem paletas mais reduzidas. Alguns mostram apenas 256 cores, outros 4096 ou 65536.

12.6 - Mais nomes de cores

As cores mostradas no site abaixo possuem diversos códigos hexadecimais relacionados a cores, todos estão definidos nos padrões do W3C.

http://www.criarweb.com/artigos/exemplos/tallerjs/tabelacores.html

13. ENTIDADES, ACENTUAÇÃO E CARACTERES ESPECIAIS

Alguns caracteres em HTML são chamados de **caracteres reservados** ou **marcadores**, que seriam os **<** *(menor que)* e **>** *(maior que)*, que quando aparecem num documento HTML são interpretados pelo navegador como delimitadores de instrução. Mas e se eu quiser simplesmente que esses caracteres façam parte do meu texto e não sejam interpretados como marcadores?!

No HTML, todos os caracteres têm um nome especial, ou código. Para acessar um caractere, você precisa usar a seguinte convenção:

```
&nome-especial;
```

onde o **nome-especial** é o nome ou um número associado ao símbolo gráfico que você deseja mostrar no seu documento. Isso define algo que podemos chamar de seqüência reservada, que é um grupo de caracteres que quando interpretado pelo navegador produzirá um caractere, que inclusive pode ser um caractere reservado. No HTML, os nomes especiais são chamados de entidades.

Na próxima tabela mostramos alguns nomes especiais de caracteres e suas respectivas seqüências reservadas para acessá-los:

CARACTERES	NOME ESPECIAL	SEQUÊNCIA RESERVADA
á	aacute	&aacute;
Á	Aacute	&Aacute;
é	eacute	&eacute;
ã	atilde	&atilde;
ô	ocirc	&ocirc;
&	amp	&amp;
<	lt	&lt;
>	gt	&gt;
¢	cent	&cent;
£	pound	&pound;
¥	yen	&yen;
§	section	&sect;
©	copyright	&copy;
®	registered trademark	&reg;
×	multiplication	&times;
÷	division	&divide;

Veja a lista completa de entidades HTML no link abaixo:
http://www.w3.org/TR/REC-html40/sgml/entities.html

14. TABELAS

Tabelas auxiliam na visualização de dados ou na divisão de sua página em setores. Entenda por uma **tabela HTML** como sendo semelhante a uma tabela que você desenharia numa folha de papel ou num outro programa de computador. O elemento principal de uma tabela é o **table** e aqui segue um exemplo simples de tabela:

```
<table border="1">
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
    <td>Quarta</td>
  </tr>
</table>
```

Cujo resultado é:

Primeira	Segunda
Terceira	Quarta

Em HTML, as tabelas são divididas em linhas e células. Na tabela acima, a primeira linha contém as células **Primeira** e **Segunda**, enquanto que a segunda linha contém as células **Terceira** e **Quarta**.

Dentro de um bloco **table**, os principais elementos que você utilizará são o **tr**, ou *table row (linha)*, e o **td**, ou *table data (célula)*. Dessa forma, podemos definir as linhas e as células da tabela. É mandatório que um bloco **td** esteja definido dentro de um bloco **tr**, ou seja, que as células estejam dentro de linhas.

No exemplo anterior, criamos uma linha com **tr** e em seguida, dentro de um bloco **td**, escrevemos o texto da célula (**Primeira**). Abrimos outro bloco **td** nesse mesmo bloco **tr**, onde colocamos o texto da segunda célula (**Segunda**). Depois, fechamos a linha com o **</tr>** para em seguida abrir uma nova linha, onde criamos as células contendo os textos **Terceira** e **Quarta**.

O atributo **border** no elemento **table** serve para dar borda à tabela. Sem o ele não ficaria muito distinto onde começa e termina cada uma das células:

```
<table>
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
    <td>Quarta</td>
  </tr>
</table>
```

Primeira	Segunda
Terceira	Quarta

Dentro de um bloco **td** é possível colocar códigos HTML e inclusive criar até tabelas dentro de células! Veja só:

```
<table border="1">
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
    <td>
      <table border="1">
        <tr>
          <td>Quarta</td>
          <td>Quinta</td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```

Primeira	Segunda	
Terceira	Quarta	Quinta

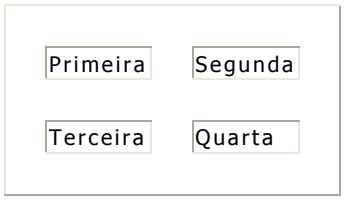
O código fica complicado o tanto quanto se queira.

Agora que você captou a essência das tabelas, vamos mostrar dois três atributos que podem ser utilizados em tabelas.

14.1 - Espaçamento entre células

Para controlarmos a distância entre duas células, utilizamos o atributo **cellspacing**, cujo argumento é dado em pixels:

```
<table cellspacing="20" border="1">
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
    <td>Quarta</td>
  </tr>
</table>
```



Para controlar o espaçamento vertical, utilizamos o **cellpadding**:

```
<table cellspacing="20" cellpadding="35" border="1">
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
    <td>Quarta</td>
  </tr>
</table>
```

14.2 - Preenchimento de células

Pode ser que você queira uma tabela onde a primeira linha tenha duas células, a segunda com apenas uma célula e a terceira com três células. Faremos o seguinte:

```
<table border="1">
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
```

```
  </tr>
  <tr>
    <td>Terceira</td>
  </tr>
  <tr>
    <td>Quarta</td>
    <td>Quinta</td>
    <td>Sexta</td>
  </tr>
</table>
```

Primeira	Segunda
Terceira	
Quarta	Quinta	Sexta

Percebeu como a tabela ficou cheia de buracos?! Para que as células preencham os buracos, utilizaremos o atributo **colspan**:

```
<table border="1">
  <tr>
    <td>Primeira</td>
    <td colspan="2">Segunda</td>
  </tr>
  <tr>
    <td colspan="3">Terceira</td>
  </tr>
  <tr>
    <td>Quarta</td>
    <td>Quinta</td>
    <td>Sexta</td>
  </tr>
</table>
```

Primeira	Segunda	
Terceira		
Quarta	Quinta	Sexta

Da mesma forma, se quisermos uma coluna com apenas uma célula e outra com duas, poderíamos usar o atributo **rowspan**. Veja a diferença entre usarmos e não usarmos o **rowspan**:

```
<table border="1">
  <tr>
    <td>Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
  </tr>
</table>
```

Primeira	Segunda
Terceira	

```
<table border="1">
  <tr>
    <td rowspan="2">Primeira</td>
    <td>Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
  </tr>
</table>
```

Primeira	Segunda
	Terceira

14.3 - Largura de células e tabelas

Para escolher a **largura** de tabelas e células, utiliza-se o já conhecido atributo **width**, cujo argumento é idêntico para o caso de imagens:

```
<table border="1" width="200">
  <tr>
    <td width="30%">Primeira</td>
    <td width="70%">Segunda</td>
  </tr>
  <tr>
    <td>Terceira</td>
    <td>Quarta</td>
  </tr>
</table>
```

Primeira	Segunda
Terceira	Quarta

Note que quando o **width** é utilizado com o elemento **table**, a largura refere-se à **largura** da página, enquanto que quando usado dentro de um **td** a largura é referente ao **tamando da tabela**. Isso tudo desde que o argumento seja dado em **porcentagem** e não em pixels.

14.4 - Uso de tabelas

Aqui exploramos poucas possibilidades das tabelas HTML. Você pode utilizá-las para construir uma simples caixa para exibição de informações como fazer uma página inteira utilizando tabelas. Existem muitos outros elementos e atributos para ajudá-lo nessa tarefa. Consulte as referências para mais informações sobre tabelas.

14.5 - Dicas

Os elementos **<thead>**, **<tbody>** e **<tfoot>** ainda são pouco usados devido ao suporte deficiente oferecido pelos navegadores antigos. Os navegadores modernos já suportam bem estes elementos.

14.6 – Elementos relativos a tabela

ELEMENTO	DESCRIÇÃO
<table>	Define uma tabela
<th>	Define um cabeçalho para uma tabela
<tr>	Insere uma nova linha numa tabela
<td>	Insere uma célula numa tabela
<caption>	Define uma legenda para uma tabela
<colgroup>	Agrupa colunas numa tabela
<col>	Define os valores dos atributos para uma ou mais colunas da tabela
<thead>	Define um cabeçalho de uma tabela
<tbody>	Define um corpo numa tabela
<tfoot>	Define o rodapé de uma tabela

15. FORMULÁRIOS

Os formulários servem para recolher dados introduzidos pelos visitantes e enviá-las para você, através de seu servidor.

15.1 – Criar Formulários

Um formulário é uma seção da página HTML que contém elementos que permitem ao visitante inserirem dados *(elementos **<tetxarea>** e vários tipos de elementos **<input>**, **<option>** e **<select>**).* Estes elementos permitem inserir dados numéricos, textos pequenos, textos longos, selecionar elementos em uma lista com várias escolhas, responder facilmente com respostas do tipo *"sim"* ou *"não"*, selecionar rapidamente uma opção em um pequeno grupo, etc.

Os formulários são criados com o elemento **<form>**. Dentro desse elemento principal colocamos diversos elementos para a inserção dos dados.

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

O elemento **<form>** por si só não faz com que o navegador desenhe nada na página nem permite inserir dados. Ele contém elementos que recolhem os dados e possui atributos que "dizem" ao navegador como e para onde devem ser enviados os dados inseridos pelo visitante.

15.2 - input

O elemento que encontramos com maior freqüência em formulários é o elemento **<input>**. O exemplo abaixo mostra um formulário simples com dois elementos input:

```
<form action="processar.php" method="post">
  Primeiro nome:
  <input type="text" name="primeiro_nome">
  <br>
  Último nome:
  <input type="text" name="ultimo_nome">
</form>
```

O formulário acima ficará sendo exibido desta forma em seu navegador:

Primeiro nome:

Último nome:

O elemento **<input>** pode assumir diversas formas com finalidades diferentes.

15.3 - "Radio Buttons"

Os "**Radio Buttons**" são utilizados para se criar um grupo pequeno de opções em que apenas podemos selecionar uma de cada vez.

```
<form action="processamento.asp">
  <input type="radio" name="sexo" value="masculino"> Masculino
  <br>
  <input type="radio" name="sexo" value="feminino"> Feminino
<form>
```

O exemplo acima ficará sendo exibido desta maneira em seu browser:

 Masculino
 Feminino

Repare que só podemos selecionar uma das opções acima.

15.4 – Checkboxes

As caixas de validação *("checkboxes")* devem ser usadas sempre que queremos que o visitante aceite *(ou não)* os itens dentro de um pequeno grupo. É permitido validar mais de uma opção simultaneamente.

```
<form>
 <input type="checkbox" name="carro">Eu tenho um carro
 <br>
 <input type="checkbox" name="surf">Eu tenho uma prancha de surf
<form>
```

O exemplo acima ficará sendo exibido desta forma em seu browser:

 Eu tenho um carro
 Eu tenho uma prancha de surf

Repare que podemos selecionar várias opções ao mesmo tempo.

15.5 - O atributo action e o botão de submissão

Quando o visitante clica sobre o botão **"Submeter"** *(ou "**Submit**")*, as opções marcadas e o texto que foram inseridos no formulário são enviados para você *(ou para seu servidor)*. O atributo **action** do elemento **<form>** contém o endereço (URL) do recurso da Web (*site, download, email, etc*) que está encarregado de realizar este processamento. É para lá que o conteúdo do formulário será enviado.

```
<form name="input" action="exemplos/action.html" method="get">
 Nome de usuário:
 <input type="text" name="usuario">
 <input type="submit" value="Submeter">
<form>
```

O exemplo acima ficará sendo exibido da seguinte forma em seu navegador:

Nome de usuário: Submeter

15.6 - Elementos para Formulários

ELEMENTO	DESCRIÇÃO
<form>	Define um formulário para recolher dados inseridos pelo utilizador
<input>	Insere um campo para introduzir dados
<textarea>	Define uma área de texto *(permite inserir texto com várias linhas e um número ilimitado de caracteres)*
<label>	Define um nome para um elemento
<fieldset>	Agrupa elementos num formulário
<legend>	Define uma legenda para um grupo de elementos do formulário
<select>	Define uma lista com várias opções selecionáveis
<optgroup>	Define um grupo de opções
<option>	Insere mais uma opção em uma lista com várias opções selecionáveis
<button>	Define um botão que pode ser pressionado
<isindex>	Desuso. Utilize **<input>** com o atributo **type="button"**

16. FRAMES HTML

As molduras ("*frames*") são subjanelas definidas sobre a janela principal do browser. Estas subjanelas são criadas dividindo a janela em várias partes. Cada uma dessas partes pode apresentar uma página da Web diferente. As subjanelas são habitualmente designadas por molduras, ou "*frames*".

16.1 - Vantagens e desvantagens das molduras

As molduras ("*frames*") nos permitem apresentar mais do que uma página HTML numa única janela do browser. Cada página está dentro da sua própria moldura (*subjanela*) e é independente das restantes páginas. Apesar de oferecerem alguma liberdade ao facilitarem bastante a criação de barras de navegação em conjuntos de documentos com muitas páginas e de tornarem bastante mais rápido o carregamento das páginas, as molduras também podem dar origem a algumas dificuldades, tais como:

- O criador de páginas vê-se obrigado a lidar com um número maior de páginas ao mesmo tempo.
- A impressão do conteúdo do navegador fica mais difícil.
- Algumas vezes será necessário a utilização de um único elemento **<a>**, para que ele realize a ligação de duas ou mais páginas, o que obriga a utilizar JavaScript.

16.2 - O Elemento frameset

- O elemento **<frameset>** define a forma como a janela do browser se subdivide para acomodar as molduras.
- Este elemento divide a janela do browser em linhas e colunas.

- Os valores atribuídos às linhas e às colunas indicam a quantidade de área de écran que cada linha e cada coluna devem ocupar.

16.3 – O Elemento frame

O elemento **<frame>** define qual o documento HTML a colocar numa determinada moldura.

No exemplo apresentado abaixo, temos um conjunto de molduras com duas colunas. A primeira coluna ocupa 25% da largura da janela do navegador, enquanto a segunda coluna ocupa 75% da largura. O documento **moldura_a.html** ocupa a primeira coluna e o documento **moldura_b.html** ocupa a segunda coluna:

```
<frameset cols="25%,75%">
   <frame src="moldura_a.html">
   <frame src="moldura_b.html">
<frameset>
```

16.4 – Dicas

Quando uma moldura possui linhas de contorno visíveis, você pode alterar suas dimensões arrastando as linhas de limite com o mouse. Para impedir que isso aconteça, utilize o atributo **noresize="noresize"** dentro da tag **<frame>**

Utilize a tag **<noframes>** para que os navegadores que não suportam frames possam mostrar um aviso na janela de quem está acessando o site.

```
<a href ="pagina_1.html" target="principal">Página 1</a><br>
<a href="pagina_2.html" target ="principal">Página 2</a><br>
<a href="pagina_3.html" target ="principal">Página 3</a>
```

Ao clicar num link dentro da moldura de navegação a nova página abre-se na segunda moldura (*à direita*), que tem por nome "**principal**".

16.5 – Elementos para Frames

ELEMENTO	DESCRIÇÃO
<frameset>	Define um conjunto de molduras
<frame>	Define o conteúdo de uma subjanela *(moldura, ou "frame")*
<noframes>	Define uma seção "noframes" para ser usada pelos browsers que não suportam molduras
<iframe>	Define uma subjanela *(moldura)* interior *("inline frame")*

17. INSERÇÃO DE SCRIPTS

A inserção de scripts em suas páginas escritas em HTML podem fazer com que elas sejam capazes de reagir de forma dinâmica e interagir com seus visitantes.

17.1 - Inserir um script numa página HTML

A inserção de um script em HTML faz-se através do elemento **<script>**

```
<html>
 <head>
 </head>
<body>
  <script type="text/javascript">
    document.write("Olá Mundo!");
  </script>
</body>
</html>
```

O script acima fará o seguinte resultado:

```
Olá Mundo!
```

17.1.1 - Um exemplo prático

```
<html>
<body>
<script type="text/javascript">
<!--
document.write("Olá mundo!");
-->
</script>
</body>
</html>
```

17.2 - Como lidar com os browsers antigos

Um navegador que não reconhece o elemento **<script>** é muito antigo e a sua utilidade na Web atual é muito reduzida. Estes browsers, ao encontrarem um elemento **<script>** (*cujo significado desconhecem*), limitam-se a apresentar o texto que está dentro do elemento como se tratasse de conteúdo normal. Para impedir que isso aconteça, deve-se colocar o conteúdo do elemento **<script>** dentro de um comentário. Deste modo, os navegadores que não suportam scripts ignoram-nos, mas os outros navegadores reconhecem os scripts e executam-nos (*apesar dos comentários*). Mesmo com browsers modernos, a utilização de comentários nos scripts é uma prática recomendável porque evita muitos problemas que surgem quando utilizamos scripts na linguagem XHTML.

17.2.1 - O Elemento <noscript>

Além de escondermos o código dentro de um comentário, uma outra forma de ajudar os browsers muito antigos seria utilizar o elemento **<noscript>** para oferecer conteúdos alternativos.

O elemento **<noscript>** é usado para compensar de alguma forma a falta de execução de um script. O conteúdo deste elemento será apresentado pelos browsers que não reconhecem scripts, mas será ignorado pelos browsers mais modernos e não interfere na apresentação da página.

17.2.2 - Um exemplo

```
<html>
<body>
<script type="text/javascript">
<!--
document.write("Olá mundo!")
```

```
-->
</script>
<noscript>
        O seu browser não suporta
        JavaScript!
</noscript>
</body>
</html>
```

17.3 - Elementos para inserir scripts e código

Elemento	Descrição
<script>	Define um bloco que contém um script
<noscript>	Define texto alternativo para ser apresentado sempre que o script não é executado
<object>	Insere um objeto na página
<param>	Define parâmetros para controlar o objeto
<applet>	Desuso. Use <object>

18. OUTRAS POSSIBILIDADES

O princípio de funcionamento do HTML já deve estar sólido em sua mente, se é que você leu tudo o que antecede esta seção. Para que este texto não se torne enfadonho, passaremos para a Especificação HTM, desenvolvida pelo consórcio da web, a tarefa de detalhar cada um dos tópicos listados a seguir.

Especificação HTML *(em inglês)*→ http://www.w3.org/TR/html4
Especificação HTML *(em espanhol)*→ http://html.conclase.net/w3c/html401-es/cover.html
Consórcio da web → docs.indymedia.org/view/Sysadmin/GuiaHtml#O_cons_rcio_da_Web

- **Texto estruturado:** blocos que permitem estruturar lógica e visualmente seu texto
  http://www.w3.org/TR/html4/struct/text.html
- **Listas:** blocos para a criação de listagens
  http://www.w3.org/TR/html4/struct/lists.html
- **Frames:** permite que a página exibida no navegador seja formada por mais de um arquivo html *(não recomendado, veja em **Acessibilidade**)*
  http://www.w3.org/TR/html4/present/frames.html
- **Formulários:** possibilita a interação do seu html com aplicações web.
  http://www.w3.org/TR/html4/interact/forms.html
- **Scripting:** permite a automatização de certas seções do seu documento.
  http://www.w3.org/TR/html4/interact/scripts.html
- **Meta-informações:** fornecem dados a respeito do próprio documento, como data, conjunto de caracteres usado, data de validade, etc
  http://www.w3.org/TR/html4/struct/global.html#h-7.4.4

Essas funcionalidades permitirão que você faça documentos HTML bem completos. Prosseguiremos agora a parte final desta apostila, onde serão abordados alguns temas bem bacanas ;-)

19. COLOCANDO SEU SITE NO AR

Até agora somente você conseguiu visualizar suas páginas. Chegou a hora de mostrá-las para o mundo todo.

Para publicar seu trabalho na Internet, você precisa somente de espaço em um servidor e um programa FTP gratuito.

Se você costuma acessar a Internet já deve ter visto que existem vários serviços gratuitos de hospedagem de sites. Seu endereço no servidor será alguma coisa parecida com http://home.servidor.com/~nomedousuario. Você tem que ativar o serviço. Informe-se como fazer isto na documentação fornecida pelo serviço de hospedagem.

Outra opção é obter um espaço gratuito de um servidor na Internet. Isto é igual a registrar uma conta de e-mail (como por exemplo, obter um endereço de e-mail do hotmail) **você pode registrar-se gratuitamente para obter um espaço em um servidor na Internet**. Existem várias companhias que oferecem este serviço gratuito - entre elas a Angelfire *(clique em "Sign Up" e escolha membro gratuito - ou veja mais abaixo uma lista com sérvios gratuitos de hospedagem)* - isto é um processo bem rápido.
Angelfire → http://angelfire.lycos.com

Para acessar o servidor você precisa conhecer o "**Nome do seu servidor**" *(por exemplo, ftp.angelfire.com)* e ter um nome de usuário e senha.

19.1 – Isto é tudo que eu preciso?!

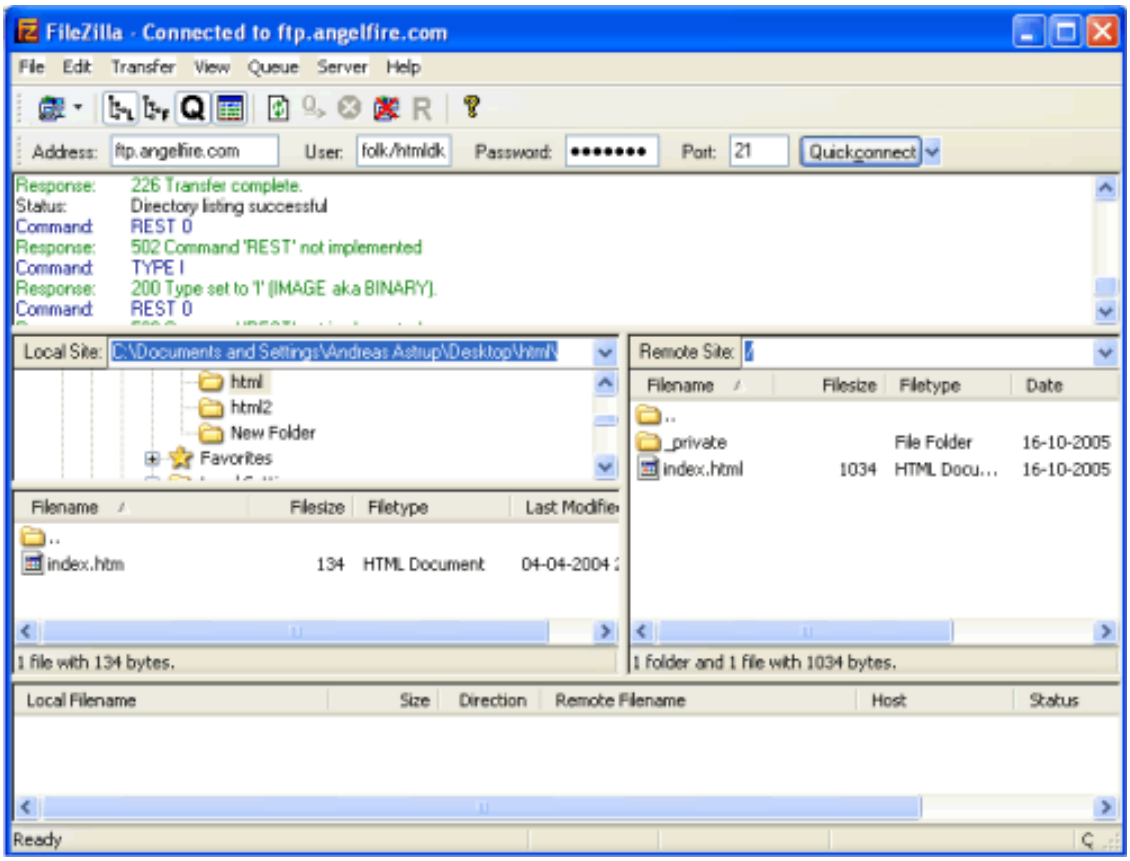
Para acessar o servidor e publicar as páginas você precisa de um programa FTP. Você não deve ter um programa destes ainda, mas existem vários deles na Internet para download e são gratuitos.

Existem muitos programas FTP. Um dos melhores é o FileZilla, e é gratuito . Você pode obter o FileZilla em http://filezilla.sourceforge.net

19.2 - Como eu faço envio os arquivos do meu site?!

Abaixo fornecemos um exemplo de como fazer isto usando o servidor Angelfire e o programa FileZilla. Este procedimento é mais ou menos igual para qualquer programa FTP.

Conecte-se à Internet e abra o programa FTP. Insira "**Host Name**" *("ftp.angelfire.com" no "Address")*, **nome do usuário** *(em "User")* e **senha** *(em "Password")* clique **"Connect"**. Você agora tem acesso ao servidor. Em um lado da janela do programa você verá os diretórios e arquivos do seu computador *("Local Site")*, e no outro o do servidor *("Remote Site"):*



Ache os seus documentos HTML e imagens a serem publicadas *(no "Local site")* e transfira para o servidor *("Remote site")* simplesmente dando um clique duplo nos arquivos. Agora o mundo todo poderá ver seu site.

Dê a uma das páginas o nome de "**index.htm**" *(ou "index.html")* e ela será automaticamente a página de entrada no site, ou seja, basta digitar http://www.angelfire.com/folk/htmlnet *(sem qualquer nome de arquivo)* e abrirá http://www.angelfire.com/folk/htmlnet/index.htm

Para mais adiante será uma boa idéia comprar um domínio *(espaço no servidor)* só para você *(www.seu-nome.com)* e assim ter um nome do site bem mais curto e fácil de guardar que aqueles fornecidos por um serviço gratuito da Internet. Você pode encontrar na Internet várias firmas que vendem domínio. Faça uma busca do Google.

LISTA DE SITES COM INFORMAÇÕES SOBRE HOSPEDAGEM GRATUITA
Link 01
http://neosite.ilogic.com.br/artigos/hospedagemgratis.htm

Link 02
http://criandosite.virgula.com.br/

Link 03
http://www.babooforum.com.br/idealbb/view.asp?topicID=425835

20. HTML AVANÇADO

Esta parte assume que você já treinou HTML o suficiente para estar familiarizado com a linguagem e fazer sites na internet. Daqui em diante, este artigo apenas fará um *tour* por outras linguagens e tecnologias que podem ser

utilizadas para organizar seus documentos, separar a aparência do conteúdo, facilitar buscas e criar documentos que, sobretudo não utilizem formatos que venham a ser tornar obsoletos.

Estarei dando aqui uma visão geral de todas as tecnologias usadas atualmente que podem ser integradas ou tem alguma relação com o HTML e linguagens de marcação:

- Folhas de Estilo (CSS)
- Templates
- Acessibilidade
- Sítios dinâmicos ou automatizados
- Web standards e validação

20.1 - Folhas de Estilo (CSS)

Folhas de estilo permitem que você separe o conteúdo do seu documento da sua representação gráfica. Como exemplo, ao invés de uma expressão do tipo:

```
<font face="fixed" size="+2" color="red">Estilo misturado com o texto</font>
```

Você utiliza algo mais organizado, como:

```
<div class="estilosa">Estilo separado do texto</div>
```

E então você definiria uma classe de nome estilosa onde os atributos, como tipo de fonte, tamanho e cor seriam definidos. Essa definição pode inclusive estar armazenada num arquivo separado do seu documento, sendo que neste último é necessário apenas uma referência ao arquivo que contenha os estilos.

O mecanismo mais utilizado para criar folhas de estilo para um arquivo HTML é o *Cascading Style Sheet*, ou CSS. Em CSS, o estilo estilosa para o exemplo acima poderia ser definido como:

```
div.estilosa {
  color : red;
  font  : fixed;
  font-size: 14;
}
```

As vantagens de usar folhas de estilo são:

- Separação entre a informação que o seu texto pretende passar e sua apresentação
- Permite que toda a apresentação do seu texto seja mudada sem que seja preciso alterar o html, bastando apenas que o arquivo que contenha a folha de estilo seja alterado
- Mais organização para seus documentos
- Facilita a criação de Templates

Pela clareza e organização resultante do uso de folhas de estilo, muitas tags e atributos do HTML são consideradas obsoletas ou seu uso é depreciado, inclusive muitas das quais ensinadas neste tutorial. Sempre que possível, utilize CSS nos seus documentos.

Para mais informações sobre Folhas de Estilo em HTML, consulte os links que anexei abaixo:

Materiais de CSS → http://www.codigofonte.com/css
Especificação HTML → http://www.w3.org/TR/html4/present/styles.html
Cascading Style Sheets, level 1 → http://www.w3.org/TR/1999/REC-CSS1-19990111

20.2 – Templates

Quem usa HTML provavelmente não o faz para escrever um único documento e sim um grande número deles. Se você está redigindo um texto ou fazendo um site composto por muitos arquivos, provavelmente você deseja que todos seus documentos HTML tenham a mesma aparência ou o mesmo estilo.

Para uniformizar o estilo e a apresentação, você pode criar um modelo, também conhecido como **Template**, que contenha o esqueleto de qualquer página de um site ou texto que seja dividido em vários arquivos. Esse template pode conter menus, tabelas e todos os campos que você for utilizar.

Existem ainda os motores de template, que são softwares que juntam o conteúdo com os templates e produzem páginas HTML. Criar um template depende do gosto de cada um, porém se você pretende utilizar um sítio automatizado existem vários motores de template que você pode utilizar, como por exemplo:

- FreeMarker: é um template engine. Ele facilita a geração de textual *(HTML, PostScript, TeX, source code, etc)* e ajuda a separar edição de design da lógica. Integrado com servlets, XML, Python e mais.

- Smarty: é uma classe de templates. Funciona de uma forma que separe interface da lógica de programação e tem por objetivo, facilitar e melhorar o desenvolvimento de qualquer aplicação em PHP.

20.3 – Acessibilidade

Uma consideração importante ao escrever seus documentos HTML é o quão usável e acessível ele é, tanto em termos visuais quanto na organização do conteúdo. Lembre-se que todos os tipos de pessoas podem acessar seu site, desde as leigas até as especialistas no assunto que você trata, desde aquelas que já conhecem o site até as que nunca ouviram falar dele. Seu conteúdo também precisa estar organizado para que ao longo do tempo as coisas não se percam.

Para ter um bom nível de usabilidade em seu site, siga as seguintes recomendações:

- Use esquemas simples
- A navegação do seu sítio deve ser intuitiva
- Crie seções do tipo "Sobre este sítio", "Quem somos", etc
- Use índices na medida do possível
- Escreva resumos
- Use referências
- Use uma lógica para os nomes de arquivos e pastas
- Evite usar frames, elas bloqueiam o modo natural de se navegar pela web

- Escolha mais de um estilo para seu sítio e evite usar combinações de cores que dificultem a visualização
- Disponibilize, na medida do possível, seu conteúdo em mais de um formato: texto simples, documento HTML, etc
- Disponibilize seu conteúdo em copyleft ou em outras licenças
- Valide seu HTML: verifique se seu código HTML está válido ou se existem incorreções

A acessibilidade de um site também leva em consideração os portadores de necessidades especiais.

O W3C possui linhas gerais de recomendação quanto à acessibilidade do conteúdo de documentos que usam linguagem de marcação.

20.4 – Sites dinâmicos ou automatizados

Seu texto ou site pode ser composto simplesmente por meia dúzia de documentos HTML e ter pouca ou nenhuma atualização ao longo da sua existência. Caso nada disso seja verdade e você está fazendo algo grande ou então você não tem tempo para escrever seu código HTML na mão, talvez seja o momento de considerar a adoção de um sistema automatizado. Destaco três tipos deles:

- Gerenciadores de conteúdo
- Blogs
- Wikis

Gerenciadores de conteúdo são softwares que rodam em servidores e que administram informações disponíveis em sites. Muitos sites na web são apenas arquivos html, mas também existem sites automatizados que permitem armazenar o conteúdo das páginas num banco de dados e a partir desse banco criar muitas páginas html e ainda manter o sistema de busca e a publicação sem que para isso o usuário precise saber html ou qualquer outra linguagem de marcação. Alguns gerenciadores de conteúdo permitem que existam usuários capazes de adicionar comentários em publicações existentes ou até mesmo criar suas próprias páginas através do preenchimento de formulários.

Os **Blogs** são casos especiais de gerenciadores de conteúdo que funcionam como diários ou coluna de editorial, onde o dono do blog adiciona texto, imagens e código HTML no site apenas preenchendo um formulário.

Já os **Wikis** são ferramentas que permitem qualquer pessoa criar ou editar um texto utilizando uma linguagem de marcação especial e mais simples de ser utilizada do que o HTML. Duas características principais dos sistemas Wiki são: você pode criar facilmente uma nova página simplesmente escolhendo um nome especial para ela e escrever na página já existente. A outra característica dos wikis é a preservação das modificações: o wiki possui um histórico de todas as modificações feitas num documento.

Os gerenciadores de conteúdo, blogs e wikis são escritos nalguma linguagem de programação como PHP, Perl, Python, Java, Ruby e muitas vezes utilizam um banco de dados para armazenar o conteúdo. Com um pouco de estudo, você pode escrever seu próprio sistema ou então utilizar uma solução pronta, como por exemplo:

- **Gerenciadores de conteúdo:** Drupal, SPIP, etc
- **Wikis:** PmWiki, PhpWiki, MediaWiki, TWiki, etc
- **Blogs:** b2, WordPress, etc

20.5 - Web standards e validação

Nesta lição você aprenderá mais alguns conceitos teóricos do HTML.

20.5.1 - O que mais há para conhecer sobre HTML?!

HTML pode ser escrito de várias maneiras. O navegador está apto a ler HTML escrito de várias maneiras. Podemos dizer que HTML tem muitos dialetos. Esta é a razão porque alguns websites são apresentados de formas diversas em diferentes navegadores.

Desde o aparecimento da Internet tem sido feitas várias tentativas para se normatizar o HTML notadamente através do *World Wide Web Consortium* (W3C) fundado por Tim Berners-Lee *(yep! o grande sujeito que inventou o HTML)*. Mas, este tem sido um árduo e longo caminho.

No passado - *quando você tinha que comprar um navegador* – o Netscape dominava o mercado de navegadores. Àquela época as normas para o HTML estavam nas suas versões 2.0 e 3.2. Mas pelo fato de dominar 90% do mercado a Netscape não teria - e não teve - que se preocupar muito com normas. Pelo contrário, a Netscape inventava seus próprios elementos de marcação que não funcionavam em outros navegadores.

Por muitos anos a Microsoft ignorou completamente a Internet. Em determinado momento, resolveu competir com a Netscape e lançou seu navegador próprio. A primeira versão do navegador da Microsoft's, o Internet Explorer, não era melhor do que o Netscape no suporte às normas do HTML. Mas, a Microsoft resolveu distribuir seu navegador gratuitamente (isto sempre agrada a todos) e o Internet Explorer em pouco tempo tornou-se o navegador mais usado e mais popular.

A partir das versões 4 e 5 a Microsoft anunciava que seus navegadores ofereciam cada vez maior suporte às normas HTML do W3C. A Netscape não se movimentou para atualizar seu navegador e continuou a colocar no mercado a velha e desatualizada versão 4.

O que vem a seguir é história. Nos dias atuais as normas HTML estão na sua versão 4.01 e no XHTML. Hoje em dia é o Internet Explorer que detém quase 90% do mercado. O Internet Explorer ainda tem seus elementos próprios, mas oferece suporte para as normas HTML do W3C. Os navegadores Mozilla, Opera e Netscape também suportam as normas.

Então, quando você codifica HTML de acordo com as normas do W3C, você está construindo um website para ser visualizado em todos os navegadores - não só agora mas também no futuro. E felizmente, tudo o que você aprendeu neste tutorial está de acordo com a mais nova versão do HTML, que é o XHTML.

20.5.1.1 - Legal!! Posso anunciar?!

Devido à existência de diferentes tipos de HTML, você precisa informar ao navegador qual é o "dialeto" do HTML e no seu caso você aprendeu XHTML. Para informar ao navegador, você usa o *Document Type Definition*. O *Document Type Definition* deve ser escrito sempre no topo do documento:

Exemplo 1:

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

<html xmlns="http://www.w3.org/1999/xhtml" lang="pt-br">

<head>
<title>Título</title>
</head>

<body>
<p>texto texto</p>
</body>

</html>
```

Além do **Document Type Definition**, que informa ao navegador que você está codificando XHTML, você precisa ainda adicionar informação extra na tag html, usando os atributos xmlns e lang.

xmlns é abreviação de "XML-Name-Space" e deve ter sempre o valor http://www.w3.org/1999/xhtml. Isto é tudo o que você precisa saber. Se você tem o hábito ou gosta de se aprofundar e conhecer coisas complicadas poderá ler mais sobre **namespaces** no site do W3C.
namespaces no site do W3C - http://www.w3.org/TR/1999/REC-xml-names-19990114

No atributo **lang** você especifica em que língua *(aqui trata-se de linguagem humana)* o documento é escrito. As abreviaturas para as línguas existentes no mundo todo, estão nas *ISO 639 standard* . No exemplo acima a língua definida no atributo é o português do Brasil *("pt-br")*.

Uma DTD informará ao navegador como deve ser lido e renderizado o HTML. Use o exemplo mostrado, como um template para todos os seus futuros documentos HTML.

O DTD é importante ainda, para a validação da página.

20.5.1.2 - Validação?! Porquê deveria eu fazer isto?!

Insira o DTD nas suas páginas e poderá verificar erros no seu HTML, usando o validador gratuito do W3C. (http://validator.w3.org)

Para testar o validador faça o seguinte: crie uma página e publique na Internet. A seguir entre em http://validator.w3.org e lá digite o endereço *(a URL)* da sua página, depois clique no botão validar. Se seu HTML estiver correto, vai aparecer uma mensagem de congratulações. Se não, será apresentada uma lista de erros, informando o quê está errado e onde. Cometa alguns erros propositais no seu código para verificar o que acontece.

O validador não é útil somente no encontro de erros. Alguns navegadores tentam interpretar os erros cometidos pelos desenvolvedores e consertar o

código mostrando a página corretamente. Em navegadores assim você nunca encontrará erros no próprio navegador. Já outros navegadores não aceitam o erro e apresentam a página arruinada ou mesmo nem apresentam. O validador então ajuda você a encontrar erros que você não tenha nem idéia de que existiam

Sempre valide suas páginas, para ter certeza que elas serão mostradas corretamente em todos os navegadores.

21 - GUIA DE REFERÊNCIA RÁPIDA

Como resumo de todo o código HTML apresentado até agora, deixo algumas tabelas de referência rápida que podem ser até impressas e guardadas no bolso.

Características gerais de um documento HTML

CÓDIGO	FUNÇÃO	EXIGE FECHAMENTO
html	bloco interno é considerado documento html	sim
head	bloco que define a cabeça do documento	sim
title	bloco que define o título do documento	sim
body	bloco interno é considerado o corpo do html	sim

Aparência do documento

CÓDIGO	FUNÇÃO	EXIGE FECHAMENTO
b	bloco em negrito	sim
i	bloco em itálico	sim
u	bloco sublinhado	sim
font	muda as características da fonte	sim
pre	mantém o texto pré-formatado	sim

Quebra de linha, parágrafos e divisões

CÓDIGO	FUNÇÃO	EXIGE FECHAMENTO
br	quebra de linha	não
center	texto centralizado	sim
p	parágrafos	recomendado
div	cria uma divisão	sim

Atributos diversos

CÓDIGO	ATRIBUTO	FUNÇÃO	VALORES POSSÍVEIS
p, div	align	alinhamento	letf, right, justify, center
font	face	fonte	nome da fonte
font	size	tamanho	-7 a +7
font	color	cor	ver no tópico de cores

Espaçamento e entidades

CÓDIGO	FUNÇÃO	EXIGE FECHAMENTO
	espaço simples	não é uma tag

O documento HTML básico

```
<html>
<head>
   <title>Aqui colocamos o título</title>
</head>
<body>
   Aqui colocamos os conteúdos visíveis
</body>
```

```
<html>
```

Elementos de cabeçalho *(para capítulos ou secções)*

```
<h1>Cabeçalho maior</h1>
<h2>. . . </h2>
<h3>. . . </h3>
<h4>. . . </h4>
<h5>. . .</h5>
<h6>Cabeçalho menor</h6>
```

Elementos para texto

```
<p>Isto é um parágrafo</p>
<br> (mudança forçada de linha)
<hr> (linha horizontal)
<pre>Isto é texto pré-formatado</pre>
```

Estilos lógicos

```
<em>Isto é texto enfatizado</em>
 <strong>Isto é texto forte</strong>
 <code>Isto é código de computador</code>
```

Estilos físicos

```
<b>Isto é texto em negrito</b>
 <i>Isto é texto em itálico</i>
```

Ligações e âncoras

```
<a href="http://www.tiagosouza.com/">Isto é uma Ligação</a>
<a href=" http://www.tiagosouza.com/"><img src="URL" alt="Texto alternativo"></a>
<a href="mailto:tiagocopa@gmail.com ">Enviar e-mail</a>
```

Uma âncora com nome:

```
<a name="dicas" id="dicas">Dicas Úteis</a>
<a href="#dicas">Saltar para a Secção de Dicas</a>
```

Lista não ordenada

```
<ul>
   <li>Primeiro item</li>
   <li>Item seguinte</li>
 </ul>
```

Lista ordenada

```
<ol>
  <li>Primeiro item</li>
  <li>Item seguinte</li>
 </ol>
```

Lista de definições

```
<dl>
  <dt>Primeiro termo</dt>
  <dd>Definição</dd>
  <dt>Termo seguinte</dt>
  <dd>Definição</dd>
 </dl>
```

Tabelas

```
<table border="1">
<tr>
  <th>um cabeçalho</th>
  <th>outro cabeçalho</th>
</tr>
<tr>
 <td>algum texto</td>
 <td>mais texto</td>
</tr>
</table>
```

Subjanelas *(molduras, ou "frames")*

```
<frameset cols="25%,75%">
  <frame src="pagina1.html">
  <frame src="pagina2.html">
<frameset>
```

Formulários

```
<form action="http://www.tiagosouza.com/processar.php" method="post/get">
<input type="text" name="lastname" value="Nabo" size="30" maxlength="50">
<input type="password">
<input type="checkbox" checked="checked">
<input type="radio" checked="checked">
<input type="submit">
<input type="reset">
<input type="hidden">

<select>
<option>Rabanetes
<option selected>Alhos
<option>Cebolas
</select>
<textarea name="Comentario" rows="60" cols="20"></textarea>
</form>
```

Entidades

< representa o mesmo que **<**
> representa o mesmo que **>**
© representa o mesmo que **©**

Outros Elementos

```
<!-- Isto é um comentário -->
```

```
<blockquote>
   Texto citado a partir de uma fonte externa.
</blockquote>
```

```
<address>
Endereço (1ª linha)<br>
Endereço (2ª linha)<br>
Cidade<br>
</address>
```

22 – REFERÊNCIAS COMPLETAS DE HTML 4.01

Todos os elementos ordenados alfabeticamente:

ELEMENTO	DESCRIÇÃO
<!--...-->	Permite inserir um comentário
<!DOCTYPE>	Indica o tipo de documento usado na página
<a>	Insere uma âncora *(marca que identifica o local do documento em que se encontra)*
<abbr>	Insere uma abreviação
<acronym>	Insere um acrônimo
<address>	Insere um endereço *(postal)*
<applet>	Insere um applet *(miniaplicação em Java)* Desuso *(utilize <**object**>)*
<area>	Define uma área sobre uma imagem
<b>	Insere texto carregado *(negrito ou "bold")*
<base>	Define o URL base de onde partem todas as ligações relativas da página
<basefont>	Define o tipo de letra base para a página. Desuso *(usar CSS)*
<bdo>	Define a direção em que o texto é apresentado
<big>	Texto com letra maior
<blockquote>	Define uma citação extensa
<body>	Elemento que contém o corpo *(conteúdo visível)* da página

 	Provoca uma mudança de linha forçada
<button>	Insere um botão num formulário
<caption>	Define a legenda de uma tabela
	Texto alinhado ao centro. Desuso.
<cite>	Insere uma citação
<code>	Define o texto que é código de computador
<col>	Define os atributos para as colunas de uma tabela
<colgroup>	Agrupa colunas numa tabela
<dd>	Insere texto que descreve uma definição
<del>	Define texto que foi apagado *("deleted")*
<dir>	Mostra uma lista de diretório. Desuso.
<dfn>	Insere a definição de um termo
	Insere uma divisão *(ou seção)* dentro da página
<dl>	Insere uma lista de definições *("definition list")*
<dt>	Insere a definição de um termo
	Insere texto enfatizado *(escreve-se em itálico)*
<fieldset>	Elemento que contém um grupo de campos de um formulário
<font>	Define o tipo de letra, o tamanho e a cor a aplicar ao texto. Desuso. *(usar CSS)*
<form>	Insere um formulário
<frame>	Define uma subjanela *(moldura)* dentro da janela principal do browser. A subjanela contém a sua própria página da Web
<frameset>	Insere um conjunto de subjanelas *("frames")*
<h1> a <h6>	Define cabeçalhos *("headers")* desde o nível 1 *(mais importante)* até ao nível 6 *(menos importante)*
<head>	Contém informação importante a respeito do documento que não deve ser apresentada no corpo da página
<hr>	Desenha uma linha horizontal
<html>	Elemento dentro do qual são colocados todos os restantes elementos da página
<i>	Insere texto para ser escrito com caracteres itálicos
<iframe>	Insere no interior da página da Web uma subjanela *("frame")* contendo a sua própria página da Web
<img>	Insere uma imagem
<input>	Define uma caixa para inserção de texto num formulário
<ins>	Define texto que foi inserido em substituição de outro mais antigo
<isindex>	
<kbd>	Define texto inserido através do teclado
<label>	Define uma marca que será associada a um controlo. Ao clicar nessa marca, o controle que estiver associado deverá ser acionado.
<legend>	Define um título num elemento <fieldset>
	Define um item de uma lista
<link>	Faz referência a um recurso externo e estabelece a ligação com ele
<map>	Define um mapeamento sobre uma imagem
<menu>	Define uma lista de menu. Desuso.
<meta>	Dá informação sobre aquilo que o documento contém
<noframes>	Define um bloco noframes, o qual só será apresentado se o browser não suportar os elementos <frameset> e <frame>
<noscript>	Define um bloco noscript, o qual só será apresentado se o browser não suportar o elemento <script>
<object>	Insere um objeto dentro da página *(como um filme em Flash, por exemplo)*
	Define uma lista ordenada
<optgroup>	Define um grupo de opções
<option>	Define uma opção numa lista de um formulário
<p>	Insere um parágrafo
<param>	Define um parâmetro para o elemento <object>
<pre>	Define texto pré-formatado
<q>	Define uma citação curta
<s>	Define texto que se escreve com um traço horizontal sobreposto *("strikethrough")* Desuso. (usar CSS)

<samp>	Define uma amostra *("sample")* de código de computador
<script>	Insere um script
<select>	Define uma lista com itens selecionáveis
<small>	Define texto menor *("small")*
	Insere uma divisão (ou secção) dentro da página
<strike>	Define texto que se escreve com um traço horizontal sobreposto *("strikethrough")* Desuso (usar CSS).
	Define texto mais forte *(será escrito em negrito)*
<style>	Define estilos CSS a aplicar na página
<sub>	Define texto que fica alinhado um pouco mais abaixo *("subscript")*
<sup>	Define texto que fica alinhado um pouco mais acima *("superscript")*
<table>	Define uma tabela
<tbody>	Define um corpo *("body")* numa tabela
<td>	Define uma divisão, ou célula, numa tabela
<textarea>	Define uma área para inserção de texto num formulário
<tfoot>	Define o rodapé de uma tabela
<th>	Define o cabeçalho de uma coluna numa tabela
<thead>	Define o cabeçalho de uma célula numa tabela
<title>	Define o título da página
<tr>	Define uma linha de células numa tabela
<tt>	Define texto que imita o de uma máquina de escrever antiga *("teletype text")*
<u>	Define texto sublinhado *("underlined")* Desuso (usar CSS)
	Define uma lista não ordenada *("unordered list")*
<var>	Define uma variável

23. ATRIBUTOS ESPECIAIS DE HTML 4

23.1 - Atributos intrínsecos

Quase todos os elementos do HTML possuem atributos. Os atributos específicos de cada elemento são descritos junto à descrição do próprio elemento na **Referência de HTML** *(veja mais acima)*. Os atributos apresentados na lista seguinte fazem parte do núcleo *("core")* da linguagem HTML e são raras as exceções que podem ser usadas em todos os elementos do HTML 4, porque são atributos intrínsecos da linguagem.

23.2 - Atributos nucleares ou intrínsecos ("*Core Attributes*")

Não podem ser usados com os elementos **base**, **head**, **html**, **meta**, **param**, **script**, **style**, e **title**.

ATRIBUTO	VALOR	DESCRIÇÃO
class	*class_rule* ou *style_rule*	A classe (CSS) a que pertence o elemento
id	*id_name*	Um nome único *(não deve ser repetido no mesmo documento)* para o elemento
style	*style_definition*	Definição de um estilo dentro do próprio corpo do documento *("inline style definition")*
title	*tooltip_text*	Texto a apresentar como dica

23.3 - Atributos lingüísticos

Não pode ser usado com os elementos **base**, **br**, **frame**, **frameset**, **hr**, **iframe, param** e **script**.

ATRIBUTO	VALOR	DESCRIÇÃO
dir	ltr | rtl	Define a direção do texto
lang	*language_code*	Define o código da língua usada na escrita dos textos

23.4 - Atributos de teclado

ATRIBUTO	VALOR	DESCRIÇÃO
accesskey	*carácter (corresponde a uma tecla)*	Define um atalho do teclado *(seqüência de teclas)* que permite aceder mais rapidamente a um elemento da página
tabindex	*número*	Define o número de ordem *("tab order")* do elemento no acesso através da tecla tab

23.5 - Eventos de janela

A partir da versão 4 da linguagem HTML praticamente todos os elementos podem desencadear eventos que têm como resposta a execução de ações por parte do navegador, como por exemplo, executar códigos JavaScript quando o visitante clicar num determinado elemento. A lista apresentada mais abaixo mostra os atributos que podemos inserir nos elementos do HTML para definir ações de resposta a eventos.

Só podem ser usados com os elementos **<body>** e **<frameset>**

ATRIBUTO	VALOR	DESCRIÇÃO
onload	*script*	Script a executar quando o documento acabar de ser carregado
onunload	*script*	Script a executar quando o documento for abandonado

23.6 - Eventos para formulários

Só podem ser usados com elementos associados a um formulário:

ATRIBUTO	VALOR	DESCRIÇÃO
onchange	*script*	Script a executar quando o valor contido no elemento sofrer uma alteração
onsubmit	*script*	Script a executar quando o formulário for submetido
onreset	*script*	Script a executar quando o conteúdo do formulário for reposto nos valores iniciais
onselect	*script*	Script a executar quando o elemento for selecionado
onblur	*script*	Script a executar quando o elemento perder o foco
onfocus	*script*	Script a executar quando o elemento ganhar o foco

23.7 - Eventos de teclado

Não podem ser usados com os elemento base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, e title.

ATRIBUTO	VALOR	DESCRIÇÃO
onkeydown	*script*	Script a executar quando uma tecla é pressionada
onkeypress	*script*	Script a executar quando uma tecla é pressionada e seguidamente libertada
onkeyup	*script*	Script a executar quando uma tecla é libertada

23.8 - Eventos do mouse

Não podem ser usados com os elementos base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style e title.

ATRIBUTO	VALOR	DESCRIÇÃO
onclick	*script*	Script a executar quando é detectado um clique no rato
ondblclick	*script*	Script a executar quando é detectado um clique duplo no rato
onmousedown	*script*	Script a executar quando o botão do rato é pressionado
onmousemove	*script*	Script a executar quando o ponteiro do rato muda de posição
onmouseout	*script*	Script a executar quando o ponteiro do rato deixa de estar sobre um elemento
onmouseover	*script*	Script a executar quando o ponteiro do rato passa a estar sobre um elemento
onmouseup	*script*	Script a executar quando o botão do rato é libertado

24. O PRESENTE E O FUTURO DO MARKUP

As linguagens de marcação se mostraram formas poderosíssimas para armazenar e categorizar conteúdo, tanto é que elas estão sendo desenvolvidas para acomodar dados com diferentes propósitos, incluindo representações de roteamento de chamadas telefônicas, fórmulas matemáticas e esquemas de classificação de remédios. Nos próximos tópicos veremos como essa generalização está tomando forma.

24.1 - O consórcio da Web

O HTML e demais linguagens de marcação são atualmente regulamentados pelo World Wide Web Consortium, o Consórcio da Web ou simplesmente *W3C*, uma iniciativa criada para padronizar muitas das tecnologias de informação surgidas com o advento da Internet.

O sucesso do HTML levou a diversos pesquisadores a explorar mais as possibilidades das linguagens de marcação que:

- Acomodem bem o tipo de informação que pretendem armazenar
- Separem o conteúdo da visualização
- Possam ser interpretados por diversos programas

A abordagem que as pesquisas na área escolheram foi a busca por uma generalização total das linguagens e duas metalinguagens *(linguagens usadas para definir linguagens)* foram definidas: o **SGML** e o **XML**.

24.2 - SGML e HTML

O **SGML** *(ou Standard Generalized Markup Language, Linguagem de Marcação Genérica Padrão)* foi a primeira generalização de linguagem de marcação a ser largamente adotada. Como metalinguagem, o **SGML** é utilizado para definir novas linguagens de marcação e atualmente o HTML é definido através de SGML.

24.3 – XML

O **XML** também é uma linguagem definida sobre SGML, mas que serve para definir linguagens de marcação com propósitos específicos e com uma sintaxe bem estrita. O **XML** é mais simples de ser interpretado via software e por isso tem ganho bastante visibilidade. Exemplos de linguagens de marcação criadas em XML são:

- **MathML:** usada para notação matemática.
- **RDF:** usado principalmente para armazenar notícias.
- **VoiceXML:** projetada para esquematizar conversas entre um ser humano e um computador.
- **XHTML:** uma versão do HTML mais simples e escrita em XML.

24.4 - RDF e Syndication

Em especial, o **RDF** *(Resource Description Framework)* é uma linguagem utilizada para criar uma outra sopa de letrinhas conhecida como **RSS** *(RDF Site Summary, ou Conteúdo de Sítios em RDF)*, que é um contêiner para armazenar informações de sites na internet.

Por exemplo, se você quisesse divulgar notícias veiculadas do site iMasters dentro de seu próprio website, você **TERIA** que fazer um programa bem complicado para extrair essa informação do HTML, mas o site iMasters também oferece seu conteúdo em **RSS**, que nada mais é do que um arquivo escrito num dialeto XML que contém os títulos, as datas e os/as autores das notícias numa estrutura bem rígida, que pode ser facilmente interpretada por outros sites e programas. Então, se alguém quiser divulgar algum tipo de conteúdo publicado no site iMasters dentro de seu site, basta utilizar o **RSS** do iMasters.
* iMasters é um site voltado a desenvolvedores web, muito bem conceituado. www.imasters.com.br

O RSS resolveu o problema de juntar informações de diferentes sites, cada um deles escrito em um determinado software ou usando determinado template HTML. Além disso, o RSS permite que a informação do site seja obtida por qualquer sistema de tratamento de informações (outro site, um programa que faz a leitura de notícias, um aparelho móvel, etc).

O RSS é suficientemente completo para permitir também que informações de conteúdos multimídia sejam distribuídas. Exemplos disso são os podcasts e o vodcast. O **podcast** é o RSS que contém informações de arquivos de áudio disponíveis em um site *(por exemplo, áudios de notícias, músicas ou programas de rádio)*. Um tocador de áudio pode ser constantemente alimentado com um podcast e com as informações contidas nele baixar todos os áudios automaticamente, sem intervenção do usuário. Com um **vodcast** é similar e consiste num RSS para arquivos de vídeo.

25. DICAS FINAIS

Se você está lendo isto aqui neste momento, parabéns, isso significa que você certamente está apto com o conteúdo aqui transmitido.

Então, agora eu já sei tudo?!

Você aprendeu um bocado de coisas e já está em condições de construir seu website. No entanto o que você aprendeu é o básico e ainda há muita coisa a

ser aperfeiçoada. Podemos dizer que você construiu uma base sólida para continuar e se aprofundar no assunto.

Nesta última lição, deixo algumas dicas finais:

- Para começar, aconselhamos a escrever seus documentos HTML de forma ordenada e estruturada. Assim fazendo você estará não só mostrando aos outros que possui uma base sólida de conhecimento, mas também estará facilitando a leitura, interpretação e manutenção do código.

- Siga as normas e valide seu código. Mas, não faça disto uma fonte de stress. Escreva um XHTML claro, use o DTD e valide suas páginas no http://validator.w3c.org

- Coloque conteúdos nas suas páginas. Lembre-se que HTML é apenas a ferramenta que possibilita apresentar informação na Internet, assim é necessário que haja a informação a ser apresentada ou seja, o conteúdo. Páginas lindas e bem desenhadas são ótimas, mas as pessoas buscam informação na Internet.

- Evite encher suas páginas com imagens pesadas e outros "balangandans" que você encontra na Internet. Isto faz com suas páginas demorem a carregar e é frustrante para o usuário. Páginas que demoram mais de 20 segundos para carregar podem perder até 50% dos seus visitantes.

- Lembre-se de cadastrar seu site nos sites de busca, de modo a que outras pessoas, além da sua família, possam encontrá-las e visitá-las. Na página de entrada dos sites para cadastro em mecanismos de busca você encontrará um link para adicionar seu site *(o mais importante é o Google mas, existem outros tais como, DMOZ, Yahoo, AltaVista, AlltheWeb e Lycos).*

- Nesta apostila você aprendeu a usar o Bloco de Notas, que é um simples e fácil editor de textos, contudo talvez você agora possa pensar em usar um editor mais sofisticado com mais possibilidades e ferramentas. Você encontra uma listagem e sumário de diferentes editores em Superdownloads.com.br ou pelo próprio Google.

Como eu aprendo mais?!

Antes de mais nada é muito importante que você continue a trabalhar e experimentar com tudo que você aprendeu neste tutorial. Quando encontrar algum site que contenha uma coisa que você ache interessante, estude o código do site. No seu navegador vá ao menu *"View"* - *"Ver"* e escolha *"Source"* - *"Código Fonte"* para ver o código do site.

Só nos resta desejar a você que passe horas agradáveis ao lado do seu novo amigo, o HTML.

O que pretende divulgar?!

Que tipo de conteúdo você pode apresentar na Web?! Praticamente o que quiser. Eis aqui alguns tipos de conteúdo mais comuns na Web, no momento:

- **Informações pessoais:** informações sobre você, por exemplo.
- **Hobbies ou interesses especiais:** filmes, motocicletas, etc.
- **Publicações:** como jornais, revistas.

- **Perfis de empresa:** o que uma empresa faz ou vende, etc
- **Documentação On-line:** desde manuais, guias de treinamento, dicionários, etc.
- **Catálogos de compras:** comercialização de artigos.
- **Lojas online.**
- **Pesquisas de opinião:** a interatividade com o usuário através de formulários, caixas de sugestões, etc.
- **Educação online:** numerosas universidades, escolas e empresas particulares oferecem o ensino a distância através da Web.

O único limite da Web é a sua própria vontade. Por isso, se a sua idéia não estiver nesta lista ou parecer meio maluca ou ainda não estiver amadurecida, pare e navegue um pouco pela Internet. Com certeza encontra excelentes idéias e poderá amadurecer as suas e ter muitas outras.

Estabeleça seus objetivos

Você deve se perguntar os que seus leitores procuram?! O que deseja realizar com sua apresentação?! Eles lerão a página inteira ou apenas uma parte dela?!

Antes de começar entrar com códigos ou imagens você deve pensar o que quero colocar em minha página?! Como será estruturada?! Ela está adequada ou não ao meu público alvo?!

Os objetivos não precisam ser grandiosos, mas a determinação irá ajudá-lo a elaborar, organizar e codificar suas páginas com uma maior probabilidade de sucesso.

Caso vá desenvolver uma apresentação Web para uma empresa ou pessoas é importante que você colha junto ao seu cliente seus objetivos, idéias, a forma que imagina sua página, etc. Assim, ficará bem mais fácil de começar seu trabalho.

Divida seu conteúdo em tópicos

Crie uma lista com os principais tópicos, a princípio não importa a ordem. Esta é uma forma de começar a se organizar.

Sua lista poderá ter quantos tópicos desejar, mas se perca listando uma quantidade enorme de tópicos *(seu leitor poderá se cansar e se perder em meio a tantas opções).*

Organização e Navegação

Aqui descreverei algumas das estruturas e navegação e suas características e ainda considerações importantes como:

- Os tipos de informação que se adaptam melhor a cada estrutura.
- Como os leitores conseguem se deslocar pelo conteúdo de cada tipo de estrutura para encontrar as informações de que precisam
- Como Ter certeza de que os leitores conseguem se localizar nos seus documentos *(contexto)* e achar o caminho de volta até uma posição conhecida.

Ao ler esta parte reflita como suas informações se encaixaria em cada uma. Você poderá combinar, até mesmo, duas estruturas e criar uma nova forma de navegação.

Webdesign e Visão do Projeto

A maioria das pessoas associam Design unicamente ao projeto gráfico de um WebSite, o que é um erro.

O conceito de Web Design envolve todos os aspectos da construção de um site, desde o desenho de sua estrutura de navegação e forma de disponibilização da informação até o desenvolvimento do projeto gráfico.

A construção de um Website deve, antes de mais nada, ser entendida como um projeto, composto de fases e desenvolvido por pessoas de backgrounds diferentes.

Conteúdo e Forma

O que faz uma pessoa entrar em um site?! O faz a pessoa querer retornar?!

Estas perguntas devem sempre estar na cabeça das pessoas de uma equipe de construção de sites.

A primeira muito relaciona-se ao público alvo. Para qual tipo de público você está disponibilizando informação?! Quais são as prioridades e interesses deste público? Que tipo de conexão é mais usada por ele?!

As respostas para a segunda pergunta certamente envolvem três pontos: o 1º é o conteúdo. É o elemento central. O 2º ponto é o design do site *(estrutura de navegação e projeto gráfico).*

Estruturar a informação de forma que a navegação seja o mais intuitiva possível faz com que sua apresentação Web tenha muito mais chance de sucesso, considerando o tipo de informação disponibilizada conforme seu público alvo. Outro ponto é a atualização para que não perca uma das principais características da Internet: a dinamicidade.

1. Na criação do projeto gráfico, tente sempre optar por soluções que melhor se adaptem a plataforma mais usada. Por exemplo: monitor 14 polegadas com resolução de 640x480, faça com a melhor diagramação das páginas aconteça quando o site é visto nesse tipo de monitor e resolução. Caso o seu site deva ser diagramado para uma resolução ou tamanho de monitor diferentes do mais usado, informe no site.

2. Num projeto gráfico deve sempre se perguntar se sua apresentação ficou adequada ou não ao que seu cliente desejava e ao seu público alvo.

3. Para ter uma garantia que suas imagens serão sempre bem visualizadas, procure trabalhar com imagens paletizadas *(formato .gif)*. O melhor resultado para isso será trabalhar suas imagens em RGB e depois indexá-las com o menor número de cores possível. Caso precise utilizar imagens True Color, salve-as em formato .jpeg, que resultará em arquivos pequenos.

4. Sempre indique em que browser sua página será melhor visualizada.

5. A página de entrada de um site é muito importante. Ela deve ser projetada de forma que diga a que o site se destina, seja de forma textual, visual ou estrutural.

6. O conteúdo e forma como ele é apresentado serão os principais atrativos de suas páginas, a não ser que o ponto chave sejam as imagens. Por isso, não pense que a net é igual a outras mídias.

7. Direitos autorais: A proteção de direitos autorais não costuma ser muito evidente na Web. O desejo original de uma página não é protegido. Mas qualquer texto, gráfico, vídeo ou áudio originais deve ser protegido. Um link ou URL não é protegido, mas uma lista de links *(como no caso das bibliotecas virtuais)* deve ser protegido pelos direitos autorais.

8. A netiqueta e Domínio público: A Web foi criada com base em ser capaz de atar links de hipertexto de qualquer outra localização da rede. Consequentemente, quando você disponibiliza um site, implica em você permitir a outros "linkar" sua página de Web, porém a netiqueta recomenda que se peça permissão aos Webmasters sempre que possível.

26. CONCLUSÃO

O volume de informações no mundo do Desenvolvimento Web é extremamente grande, e dificilmente uma pessoa domina todos os aspectos e particularidades. Por isso é bastante comum ouvirmos termos como "Webdesigner" e "Programador Web". Os Webdesigners geralmente dominam a arte da criação de layouts, interfaces com o usuário, tipografia, CSS e programas de edição de imagens. Os Programadores Web por sua vez são responsáveis pela estruturação dos códigos das páginas, pela programação das Linguagens de Servidor e Navegador e pela criação de banco de dados.

Apesar de excitante, a discussão sobre esses temas começa a fugir muito do objetivo inicial desta apostila, que é a introdução prática do leitor ou da leitora à linguagem de marcação conhecida como HTML e também dar uma noção sobre as novas tecnologias e o futuro da linguagem de marcação.

Ao avançar nos estudos logo se percebe essa ramificação e cabe ao estudante desenvolver a área que mais lhe atrai.

Espero que você tenha gostado.

Um abraço,

Tiago Souza
http://www.tiagosouza.com
http://useclick.blogspot.com