



Universidade Federal de Juiz de Fora - UFJF
Diretoria de Sistemas de Informação - DSI
Centro de Gestão do Conhecimento Organizacional - CGCO

HTML

1. HTML - CONCEITOS BÁSICOS

Baseado no trabalho de Aline R. Alves e Elaine Venson (UFSC)

Este material tem por objetivo estabelecer noções básicas sobre hipertexto e editores HTML. Para melhor entendimento, inicia-se esta seção com o significado das principais terminologias utilizadas neste trabalho.

INTERNET

A Internet, também conhecida como NET, é a rede mais aberta do mundo. Milhares de computadores fornecem recursos que estão disponíveis a qualquer um que tenha acesso à rede. Seu início aconteceu há aproximadamente 20 anos atrás como uma rede experimental do Departamento de Defesa dos Estados Unidos, chamada ARPANET.

Atualmente a Internet é utilizada para várias finalidades, podendo-se destacar:

- Correio eletrônico (e-mail)
- Conversação on-line
- Acesso à informações
- Jogos

WWW

O World Wide Web (WWW) é um sistema de informações distribuídas, baseado em hipermídia, criado por iniciativas de pesquisadores do CERN (Laboratório Europeu de Física de Partículas), na Suíça.

O WWW permite acesso a informações textuais, combinadas com recursos de som, imagem e vídeo, também dispõe suas informações na forma de hipertexto, que permite uma leitura não linear do texto.

Há também os serviços disponíveis por Telnet - arquivos em servidores FTP (File Transfer Protocol), informações servidas por Golpher e Wais (localizar documentos que contém informações que você está procurando) - e informações da Usenet (Newsgroups).

BROWSERS

O programa de computador que permite a visualização de páginas WWW é chamado de Browser. O Browser interpreta documentos escritos em HTML e os reproduz na forma de página WWW.

Os Browsers mais conhecidos são:

- Internet Explorer
- Mozilla
- Opera
- Netscape
- Konqueror

Ainda o browser pode gerenciar correio eletrônico, transferência de arquivos e leitura de newsgroups.

URL - Uniform Resource Locator

URL é a forma padrão de designação de documentos, através da qual estes são acessados na Internet. A URL tem o seguinte formato, como exemplo:

`http://host.empresa.com.br/dir/index.html`

O URL prevê a identificação da máquina (host), do domínio (empresa.com.br), do arquivo que contém o documento (dir/index.html), e do protocolo pelo qual ele será acessado (http).

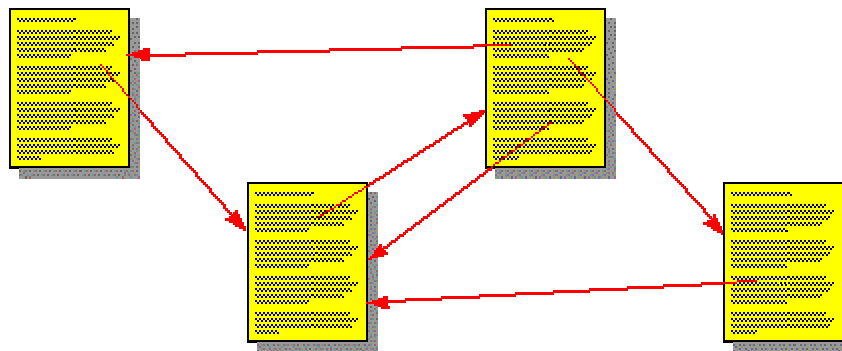
Outros formatos de URL existem para denominar outras formas de acesso a elementos da Internet (por exemplo, arquivos em FTP ou Gopher, artigo na Usenet ou registros de dados em uma base de dados qualquer).

HTTP: HyperText Transfer Protocol

HTTP é um protocolo padronizado para transferência de arquivos que contém documentos hipermídia. Além de um protocolo para transferência de hipertexto, HTTP é um protocolo para acesso à informação com eficiência necessária para realizar saltos de acordo com a exigência dos hipertextos. HTTP transfere principalmente documentos HTML, mas na verdade está aberto para suportar um ilimitado e extensível conjunto de formatos. É o protocolo mais popular entre os suportados por URLs. As URLs do tipo HTTP seguem a forma básica:

`http://host.empresa.com.br`

HTML: HyperText Markup Language



A linguagem HTML(Hyper Text Markup Language) é uma linguagem de marcação simples usada para criar documentos hipertexto que são portáteis de uma plataforma a outra. É uma coleção de estilos usados para definir os vários componentes de um documento. É baseada na SGML (Standard Generalized Markup Language), com semântica genérica e apropriados para a representação de informações de uma ampla gama de aplicações. Um documento HTML pode ser usado para correio eletrônico hipertexto; documentos em geral; hipermídia; menus de opções; sistema de news e em qualquer lugar onde um sistema básico de hipertexto seja necessário.

Os browsers são formadores de HTML. Quando um documento HTML é carregado em um browser, ele lê (passa por um **parser**) as informações HTML e formata o texto e as imagens na tela de acordo com as informações contidas no documento, que podem incluir vários níveis de cabeçalho, listas, menus, formatação de estilos de texto, etc.

A HTML tem sido usada no WWW (World Wide Web) desde 1990. A especificação da mesma vem evoluindo desde então. O HTML + foi apresentado de 1993 e incluiu diversas características inéditas, como tabelas, formulários para preenchimento, figuras com legendas, e propunha as primeiras idéias para suportar fórmulas matemáticas. Em 1994 saiu o HTML 2.0, a especificação foi reescrita para dar consistência e usabilidade. A versão Draft 3.0 de HTML é de março de 1995, mas a diferença entre HTML 2.0 e 3.0 era tão grande que se tornou um obstáculo virtualmente insuperável para implementação. Por este motivo, a versão 3.0 não está sendo mantida. Posteriormente foi apresentada a versão conhecida como HTML 3.2, a qual provê um conjunto de características novas (*applets Java*, fluxo de texto ao redor de imagens e super/subscritos), mantendo, porém, compatibilidade com a versão 2.0 (anterior). A versão mais atualizada é a 4.0 e o W3C, responsável pelas especificação da HTML, está trabalhando na XHTML (Extensible HTML), uma reformulação da HTML para adaptar-se à XML (uma linguagem genérica de marcação).

Formas de criar HTML

Alguns preferem editar HTML diretamente, tratando com seus rótulos (tags), digitando a mão. Outros preferem usar editores visuais WYSIWYG (What You See Is What You Get), tais como o AOLPRESS, Editor Netscape; ou ainda editores não visuais projetados para HTML, como o Hot Dog, HTMLed Pro.

Um documento HTML

O corpo de um documento HTML é constituído por diretivas ou tags, que são comandos de formatação da linguagem. A forma geral de uma tag é <nome_da_tag>. . . </nome_da_tag>. As tags podem ser aninhadas.

A primeira diretiva é <HTML> que indica que o conteúdo deste arquivo está em linguagem HTML.

```
<HTML> . . . . </HTML>.
```

A próxima diretiva encontrada é <HEAD> que contém informações sobre o documento. Há poucas tags que podem ser incluídas dentro de <HEAD>, uma delas é <TITLE>, que contém o título da página e não admite nenhuma tag no seu interior.

```
<HEAD> . . . </HEAD>
```

```
<TITLE> . . . </TITLE>
```

O restante do documento, incluindo todos os textos, ficam dentro da diretiva <BODY>.

```
<BODY> . . . </BODY>
```

Assim a estrutura básica de uma documento HTML que pode ser usada como padrão é mostrada abaixo.

```
<HTML>
```

```
<HEAD>
```

```
<TITLE> Título_da_página </TITLE>
```

```
</HEAD>
```

```
<BODY>
```

```
Corpo_da_homepage . . .
```

```
</BODY>
```

A Formatação do texto

A seguir uma relação de tags que permitem a formatação de um texto

< H1> . . . </H1> <H6> . . . </H6>	Cabeçalhos que dividem as seções do texto em níveis
<P> . . . </P>	Define um parágrafo
<BLOCKQUOTE> . . . </BLOCKQUOTE>	Apresenta o texto como uma citação
<PRE> . . . </PRE>	Insere um texto pré-formatado
 	Permite a quebra de linha
<CENTER> . . . </CENTER>	Deixa o texto centralizado
<P align= > . . . </P>	Parágrafo com alinhamento <P align=center > . . . </P>
<Hn align= > . . . </Hn>	Cabeçalho com alinhamento
<I> . . . </I>	Formata o texto em itálico
 . . . 	Formata o texto em negrito
<U> . . . </U>	Deixa o texto sublinhado
<BIG> . . . </BIG>	Aumenta a fonte e deixa o texto em negrito

Caracteres especiais

HTML permite que caracteres especiais sejam representados por seqüências de escape, indicadas por três partes: um & inicial, um número ou cadeia de caracteres correspondente ao caracter desejado, e um ; final.

Três caracteres ASCII (<, >, &) têm significados especiais em HTML, e são usados dentro de documentos seguindo a correspondência:

Entidade	Caracter
<	<
>	>
&	&

O espaço em branco "não-quebrável" é indicado por

Entidade	Caracter
 	Espaço em branco

Outras seqüências de escape suportam caracteres ISO Latin1. Aqui está uma tabela com os caracteres mais utilizados em Português:

Entidade	Caracter	Entidade	Caracter
á	á	Á	Á
â	â	Â	Â
à	à	À	À
ã	ã	Ã	Ã
ç	ç	Ç	Ç
é	é	É	É
ê	ê	Ê	Ê
í	í	Í	Í
ó	ó	Ó	Ó
ô	ô	Ô	Ô
õ	õ	Õ	Õ
ú	ú	Ú	Ú
ü	ü	Ü	Ü

Como vemos, as seqüências de escape são sensíveis à caixa. Os editores de HTML fazem essa tradução automaticamente

Criando Links entre os documentos

Existem basicamente três tipos de links/âncoras entre documentos:

- Link entre dois documentos
- Link em um mesmo documento
- Link entre um documento e uma URL

1. Link entre dois documentos

Para criar este tipo de link, basta especificar o nome do arquivo de destino e caso não esteja no mesmo diretório, especificar também o caminho.

Link no início da página de destino:

```
< A HREF="destino.htm"> nome_do_link</A>
```

Link em um lugar especificado na página de destino:

```
< A REF="destino.htm"#local_específico>nome_do_link</A>
```

2. Link em um mesmo documento

Para criar este link interno é necessário especificar o ponto de destino.

```
<A HREF="#nome_da_seção">nome_do_link</A>
```

```
< A NAME="nome_da_seção">nome_da_seção</A>
```

3. Link entre documento e URL

Neste tipo de link é necessário especificar o protocolo e o servidor do arquivo

```
<A HREF="protocolo/servidor/arquivo"> link</A>
```

Modificando o Layout da Página

1. Inserindo Imagens

Admite-se alguns padrões de imagens em documentos html, tais como GIF, JPEG e PNG. Às vezes se torna necessária a utilização de imagens com fundo transparente, isto é, o formato GIF89a. Existem programas como LVIEW e PRINT SHOP PRO 3.11 que fazem esse tipo de conversão.

```
<IMG SRC="imagem.gif">
```

Essa diretiva tem como atributos :

- * ALIGN- Alinhamento do texto em relação à figura (top, middle, bottom)

- * ALT - Texto alternativo para leitores que não visualizam a imagem

```
<IMG SRC="imagem.gif" ALT="nova_figura" >
```

- * ISMAP - Identifica a imagem como um image map (imagem mapeada para vários links)

Uma imagem também pode servir como fundo de uma página

```
< BODY BACKGROUND="imagem.gif" >
```

2. Definindo Cores

Pode-se definir as cores dos seguintes ítems em uma página: fundo, texto, link padrão, link visitado, link ativo, inserindo no início da tag <BODY> os correspondentes valores em hexadecimal.

```
<BODY BGCOLOR="#eee0c8" TEXT="#ff0000" LINK="#008000" VLINK="#aa0000"  
ALINK="#b00f00" >
```

Criando Listas

Outro modo de estruturar um documento HTML é utilizando listas. Existem dois tipos de listas: Ordenadas, que atribuem um número a cada item; e as Não-ordenadas que encadeiam uma série de ítems sem enumerá-los.

1. Lista não-ordenada

O comando (unordered list) deve envolver o primeiro e o último item lista.

Para criar uma lista não-ordenada você deve usar uma série de comandos, cuja sintaxe básica é mostrada abaixo:

```
<UL>  
  <LI> texto do item  
  <LI> texto do item  
</UL>
```

2. Lista ordenada

O comando (ordered list) deve envolver o primeiro e o último itens da lista.

Para criar uma lista ordenando seus ítems você deve usar o conjunto de comandos abaixo:

```
<OL>  
  <LI> texto do item  
  <LI> texto do item  
</OL>
```

3. Lista ordenada

O comando (ordered list) deve envolver o primeiro e o último itens da lista.

Para criar uma lista ordenando seus ítems você deve usar o conjunto de comandos abaixo:

```
<OL>  
  <LI> texto do item  
  <LI> texto do item  
</OL>
```

4. Listas aninhadas

Você também pode aninhar diversas listas para criar estruturas hierárquicas. Também é possível misturar listas ordenadas com não-ordenadas. Mas você deve prestar atenção ao casar todos os comandos. Todo comando aberto deve ser fechado com o comando . Se você esquecer um deles a lista ficará incorreta.

Exemplos de listas aninhadas:

Listas ordenadas

```
<B> Listas ordenadas aninhadas</B>  
<OL>  
<LI> este é o primeiro item da lista principal
```

```
<LI> este é o segundo item da lista principal
<OL>
  <LI> este é o primeiro sub item do segundo item da lista principal
  <LI> este é o segundo sub item do segundo item da lista principal
</OL>
</OL>
```

Listas não-ordenadas

```
<B> Listas não-ordenadas aninhadas </B>
<UL>
  <LI> primeiro item da lista principal
  <LI> segundo item da lista principal
  <UL>
    <LI> primeiro sub item do segundo item da lista principal
    <LI> segundo sub item do segundo item da lista principal
  </UL>
</UL>
```

Listas ordenadas e não-ordenadas aninhadas

```
<B> Listas ordenadas aninhadas</B>
<OL>
  <LI> primeiro item da lista principal
  <LI> segundo item da lista principal
  <UL>
    <LI> primeiro sub item do segundo item da lista principal
    <LI> segundo sub item do segundo item da lista principal
  </UL>
  <LI> terceiro item da lista principal
</OL>
```

Criando Tabelas

As tabelas têm uma estrutura parecida com a de uma planilha eletrônica, compõem-se de linhas e colunas cujas interseções formam as células. Uma célula pode conter uma imagem, um texto ou link. Uma tabela é criada com os comandos <TABLE> e </TABLE>. Entre eles devem ser especificadas as tags para a criação de linhas e células, títulos, bordas e alinhamento da tabela. Como padrão uma tabela é criada sem borda, isto é, não aparecem as linhas nem as colunas da tabela. Para visualizar as linhas é necessário usar a tag <TR></TR> (table row). Se uma tabela tiver cinco linhas devem ser indicados cinco pares destas tags. As tags <TD></TD> (table data) são usadas para especificar o conteúdo de uma célula e devem ser usadas entre os comandos <TR></TR>. A largura de uma coluna é definida pela largura da maior célula que faz parte da tabela. O conteúdo de uma célula é alinhado pela opção ALIGN. Como padrão o alinhamento é feito horizontal do texto é feito pela esquerda (align=left) e o alinhamento vertical, no meio da célula (VALIGN=middle).

Também há a tag <TH></TH> (table header) usado para criar células da mesma forma que os comandos <TD></TD>, só que o texto fica em negrito e centralizado.

A tag <CAPTION>...</CAPTION> permite a criação de uma legenda para a tabela. A legenda é um texto externo que aparece antes ou depois da tabela. Esses comandos devem ser especificados após o comando Table, antes das tags <TR>. Uma legenda pode ser exibida no topo ou no pé da tabela, dependendo do valor atribuído à opção Align, que pode ser top ou bottom. O padrão é top. A legenda é sempre centralizada horizontalmente.

Algumas opções que podem ser usadas com os comandos descritos:

BORDER = valor - Esta opção pertence ao comando Table. Se for especificado sem nenhum valor, uma linha fina é criada em volta de todas as células. Quanto maior o valor especificado, mais grossa é a borda.

ALIGN - Dentro do comando <CAPTION> pode assumir os valores TOP ou BOTTOM. Nos comandos <TR>,<TH> ou <TD> pode assumir os valores LEFT, CENTER e RIGHT.

VALIGN - É equivalente ao comando Align, só que funciona para alinhar o texto dentro da célula verticalmente. Os valores possíveis são TOP, BOTTOM e BASELINE. O MIDDLE é o padrão.

NOWRAP - Esta opção pode aparecer em qualquer lugar da célula da tabela e indica que o texto não pode ser quebrado para se ajustar ao seu tamanho.

COLSPAN - Esta opção pode aparecer em qualquer lugar da célula e especifica quantas colunas da tabela a célula deve ocupar. O valor padrão é 1.

ROWSPAN - Esta opção funciona como a anterior, só que especifica quantas linhas, para baixo a célula deve ocupar. O padrão é 1.

CELLPADDING - É uma opção que define o espaçamento, em pixels, entre as colunas da tabela.

CELLSPACING - É uma opção que define o espaçamento, em pixels, dentro de uma célula.

Exemplo de tabela

```
<TABLE >
<TR>
<TH> título1</TH>
<TH> título2</TH>
</TR>
<TR>
<TD> linha 1 coluna1 </TD>
<TD> linha 1 coluna 2 </TD>
</TR>
<TR>
<TD> linha 2 coluna1</TD>
<TD> linha2 coluna 2</TD>
</TABLE>
```

Frames

Frames são uma extensão do HTML que permite que a janela do browser seja dividida em várias regiões. Cada uma destas regiões pode conter documentos totalmente distintos e independentes.

Quando utiliza-se frames, é necessário a criação prévia de um documento HTML - documento de *layout*, que irá definir em quantas regiões será dividida a janela do browser, qual será o tamanho e quais serão os documentos carregados em cada uma delas.

O exemplo abaixo ilustra uma página em que a janela do *browser* é dividida em duas regiões - direita e esquerda. Não possui conteúdo, apenas as divisões da janela.

Exemplo de Frame

```
<HTML>
<HEAD>
<TITLE>
</HEAD>
<FRAMESET COLS=200,*>
<FRAME SRC=arquesque.htm>
<FRAME SRC=arqdir.htm NAME=direita>
</FRAMESET>
</HTML>
```

FRAMESET: Substitui a tag <BODY> de um arquivo HTML normal. Possui um argumento que define a divisão desejada - em colunas (vertical) ou linhas (horizontal): COLS ou ROWS e o número e largura de cada linha/coluna, especificado no número existente entre cada vírgula. A quantidade de valores especificados entre vírgulas determina o número de divisões da tela. O * especifica que a divisão terá a largura definida pelo próprio *browser*, ou seja, utilizará o espaço que estiver sobrando na tela.

FRAME: Define os arquivos que serão carregados em cada divisão. Para cada frame definido em <FRAMESET> é necessária a utilização de um tag <FRAME>. O comando NAME é utilizado quando você quiser determinar que futuros documentos serão carregados especificamente naquele frame.

TARGET: Determina em que frame deve ser carregado o arquivo - o frame-alvo.

É possível fazer subdivisões no frame, ou seja, "**frames dentro de frames**". Para isto basta criar um novo frame com nome de um dos arquivos apontados pelo frame inicial, ou seja, definir um novo documento de *layout* com características próprias.

2. HTML AVANÇADO

Baseado no trabalho de Maria Alice Soares de Castro (masc@icmc.usp.br)

Disponível em: <http://www.icmc.usp.br/ensino/material/html/>

Formulários

Um formulário é um modelo para a entrada de um conjunto de dados. O primeiro passo para fazer formulários é aprender as etiquetas que desenham as janelinhas de entrada de dados, para depois trabalharmos com os scripts, que são os programas que tratam esses dados, oferecendo os serviços desejados (acesso a banco de dados, envio de e-mail, etc.).

O elemento `<FORM>` delimita um formulário e contém uma seqüência de elementos de entrada e de formatação do documento.

```
<FORM ACTION="URL_de_script" METHOD=método ENCTYPE="type">...</FORM>
```

1. ACTION

Especifica o URL do *script* ao qual serão enviados os dados do formulário.

2. METHOD

Seleciona um método para acessar o URL de ação. Os métodos usados atualmente são GET e POST. Ambos os métodos transferem dados do browser para o servidor, com a seguinte diferença básica:

- POST
 - os dados entrados fazem parte do corpo da mensagem enviada para o servidor;
 - transfere grande quantidade de dados.
- GET
 - os dados entrados fazem parte do URL (endereço) associado à consulta enviada para o servidor;
 - suporta até 128 caracteres.

3. ENCTYPE

Indica o tipo de codificação dos dados enviados através do formulário. O tipo *default* é `application/x-www-form-urlencoded`. Outro tipo aceito por alguns browsers é `text/plain`.

Os formulários podem conter qualquer formatação - parágrafos, listas, tabelas, imagens - exceto outros formulários. Em especial, colocamos dentro da marcação de `<FORM>` as formatações para campos de entrada de dados, que são três: `<INPUT>`, `<SELECT>` e `<TEXTAREA>`.

Todos os campos de entrada de dados têm um atributo `NAME`, ao qual associamos um nome, que será utilizado posteriormente pelo *script*. São os *scripts* que organizam esses dados de entrada em um conjunto de informações significativas para determinado propósito.

Primeiro vamos ver os tipos de campos para montar um formulário, e depois passaremos aos *scripts*.

INPUT

O campo `<INPUT>` tem um atributo `TYPE`, ao qual atribuímos seis valores diferentes para gerar seis tipos diferentes de entrada de dados.

1. Campo de dados texto

Quando `INPUT` não apresenta atributos, é assumido que `TYPE=TEXT` (*default*); a formatação:

```
Nome : <INPUT TYPE=TEXT NAME="Nome" >
```

ou apenas:

```
Nome : <INPUT NAME="Nome" >
```

2. Campo de dados senha

Entrada de texto na qual os caracteres são escondidos por asteriscos, como se pode ver no exemplo.

```
Login: <INPUT TYPE=TEXT NAME=login><br>
Password: <INPUT TYPE=PASSWORD NAME="senha">
```

Alguns atributos para os campos de tipo TEXT e PASSWORD

VALUE pode ser usado para dar um valor inicial a um campo de tipo texto ou senha. Desse modo, se o usuário não preencher este campo, será adotado este valor padrão. Se o usuário quiser entrar dados, ele simplesmente apaga o que já estiver escrito e escreve suas informações.

```
Nome: <INPUT TYPE=TEXT NAME="nome" VALUE="Seu nome">
```

SIZE especifica o tamanho do espaço no display para o campo do formulário. Só é válido para campos TEXT e PASSWORD. O valor default (padrão) é 20.

```
Endereço: <INPUT TYPE=TEXT SIZE=35>
```

MAXLENGHT é o número de caracteres aceitos em um campo de dados; este atributo só é válido para campos de entrada TEXT e PASSWORD.

```
Dia do mês: <INPUT TYPE=TEXT NAME="ex" MAXLENGHT=2>
```

Apenas 2 caracteres serão lidos pelo formulário, não importa o quanto se escreva neste campo.

3. Múltipla escolha

CHECKBOX insere um botão para escolha de opções. A cada alternativa corresponde um valor a ser manipulado pelo script que processa os dados. Note que o nome do campo (**NAME**) é o mesmo para toda a lista de valores. Pode ser escolhida mais de uma alternativa.

```
<INPUT TYPE=CHECKBOX NAME="esporte" VALUE="basquete">Basquete <br>
<INPUT TYPE=CHECKBOX NAME="esporte" VALUE="bocha">Bocha
```

Uma diretiva **CHECKED** marca uma escolha inicial, isto é, se o usuário não escolher nenhuma opção, o formulário irá considerar a alternativa "pré-escolhida":

```
<INPUT TYPE=CHECKBOX NAME="esporte" VALUE="volei" CHECKED>Vôlei <br>
<INPUT TYPE=CHECKBOX NAME="esporte" VALUE="futebol">Futebol
```

4. Escolha única

RADIO insere um botão de escolha de valores para uma opção, isto é, somente uma alternativa pode ser escolhida. Note que o nome do campo (**NAME**) é o mesmo para toda a lista de valores.

```
<INPUT TYPE=RADIO NAME="time" VALUE="palm">Palmeiras <br>
<INPUT TYPE=RADIO NAME="time" VALUE="inte">Internacional
```

Uma diretiva **CHECKED** marca uma escolha inicial - se o usuário não escolher nenhuma opção, o formulário irá considerar a alternativa "pré-escolhida":

```
<INPUT TYPE=RADIO NAME="time" VALUE="aea" CHECKED>AEA <br>
<INPUT TYPE=RADIO NAME="time" VALUE="naut">Náutico
```

5. Botões de ação - SUBMIT

SUBMIT apresenta o botão que causa o envio dos dados de entrada para o servidor:

```
<INPUT TYPE=SUBMIT>
```

É possível modificar o rótulo desse botão através do atributo **VALUE**

```
<INPUT TYPE=SUBMIT VALUE="Envia mensagem">
```

6. Botões de ação - RESET

RESET restaura os valores iniciais das entradas de dados.

```
<INPUT TYPE=RESET>
```

É possível modificar o rótulo desse botão através do atributo **VALUE**

```
<INPUT TYPE=RESET VALUE="Apaga tudo!">
```

SELECT

<SELECT> apresenta uma lista de valores, através de campos OPTION.

```
<SELECT NAME="sabor" >
<OPTION>Abacaxi
<OPTION>Creme
<OPTION>Morango
<OPTION>Chocolate
</SELECT>
```

Também é possível estabelecer uma escolha-padrão, através do atributo SELECTED

```
<SELECT NAME="sabor" >
<OPTION>Abacaxi
<OPTION SELECTED>Creme
<OPTION>Morango
<OPTION>Chocolate
</SELECT>
```

Com o atributo SIZE, escolhe-se quantos itens da lista serão mostrados (no exemplo, **SIZE=4**):

```
<SELECT NAME="sabor" SIZE=4 >
<OPTION>Abacaxi
<OPTION SELECTED>Creme
<OPTION>Morango
<OPTION>Chocolate
</SELECT>
```

Com o atributo MULTIPLE, define-se que se pode selecionar mais de um item (pressionando a tecla Shift do teclado enquanto se selecionam os itens):

```
<SELECT NAME="sabor" SIZE=4 MULTIPLE >
<OPTION>Abacaxi
<OPTION SELECTED>Creme
<OPTION>Morango
<OPTION>Chocolate
</SELECT>
```

TEXTAREA

<TEXTAREA> abre uma área para entrada de texto, de acordo com atributos para número de colunas, linhas, e - se for o caso - um valor inicial.

```
<TEXTAREA COLS=40 ROWS=5 NAME="comentario"> Deixe seu comentário </TEXTAREA>
```

CGI Scripts

CGI, ou *Common Gateway Interface*, é uma interface definida de maneira a possibilitar a execução de programas - "gateways" - sob um servidor HTTP. Neste contexto, os "gateways" são programas ou scripts (também chamados "cgi-bin") que recebem requisições de informação, retornando um documento com os resultados correspondentes. Esse documento retornado pode existir previamente, ou pode ser gerado pelo script especialmente para atender àquela requisição específica do usuário (diz-se que o documento é gerado "*on the fly*").

Exemplos de aplicação de CGI incluem:

- processamento de dados submetidos através de formulários: consulta a banco de dados, cadastramento em listas, livros de visita, pesquisas de opinião;
- criação de documentos personalizados *on the fly*;
- gerenciamento de contadores de acesso;
- processamento de mapas.

Tais scripts podem ser escritos em qualquer linguagem que possa ler variáveis, processar dados e retornar respostas - ou seja, qualquer linguagem de programação! A linguagem é determinada de acordo com a plataforma do servidor e da aplicação a ser implementada.

Visão Geral

Os scripts têm uma forma geral comum:

1. leitura de dados entrados pelo usuário (e/ou campos de informação de um pacote HTTP);
2. processamento dos dados (armazenamento dos dados em um banco de dados, realização de cálculos, recuperação de dados etc.);
3. montagem de uma página Web ou geração de uma imagem com os resultados produzidos.

Submissão de formulários

Suponhamos um formulário cuja marcação principal seja:

```
<FORM ACTION="/cgi-bin/teste.cgi" METHOD=método>  
...  
</FORM>
```

onde ACTION indica o URL do script que receberá os dados (ainda não vamos nos preocupar com o campo METHOD)

Quando submetemos os dados de um formulário, o browser organiza os dados entrados pelo usuário, assegurando que as informações chegarão em ordem e serão compreendidas pelo script. Vejamos alguns exemplos.

Um campo de texto simples, tal como:

```
<INPUT TYPE=TEXT NAME="nome" >
```

cuja entrada tenha sido "Prof. Achille Bassi"

será organizado pelo browser da seguinte forma:

```
nome=Prof.+Achille+Bassi
```

Isto é, passa a assumir um formato `nome = valor`, onde:

nome foi definido nas etiquetas do formulário (pelo atributo NAME de cada campo) e

valor é a entrada oferecida pelo usuário.

Note que os espaços em branco são substituídos por sinais de +. Se o formulário tiver mais algum campo de informação, por exemplo:

```
<INPUT TYPE=TEXT NAME="email" >
```

com uma entrada "bassi@icmc.usp.br"

o browser estará gerando uma linha única com todos esses dados, desta forma:

```
nome=Prof.+Achille+Bassi&email=bassi@icmc.usp.br
```

Isto é, os campos de informação são separados por &. Também todos os acentos e símbolos especiais são codificados em hexadecimal para o envio dos dados. Esta codificação dos dados de um formulário é padrão.

Feita essa codificação, o browser envia uma requisição para o URL indicado em ACTION, que está associado a um script. Esse envio está sujeito a um método específico, que definirá como o script irá recuperar essas informações para processá-las.

Antes de ver os métodos, é bom conhecer as variáveis de ambiente, que serão muito úteis no tratamento de informações.

Variáveis de ambiente

Variáveis de ambiente são variáveis estabelecidas pelo servidor HTTP no momento de atender requisições de documentos. Os scripts têm acesso a essas variáveis através da interface CGI.

A variáveis geralmente mais usadas são estas:

SERVER_SOFTWARE : nome e versão do software do servidor

SERVER_NAME : nome do servidor

SERVER_PROTOCOL : versão do protocolo HTTP usado pelo servidor

SERVER_PORT : porta TCP pela qual o servidor atende as requisições

QUERY_STRING : cadeia de caracteres que contém uma consulta ou entrada de formulário (ela está presente de acordo com o método e o script utilizado para tratar os dados do formulário)

REMOTE_HOST

máquina que solicita a execução do script (máquina do usuário)

CONTENT_TYPE

tipo MIME do conteúdo da mensagem enviada

CONTENT_LENGTH

comprimento da mensagem enviada

HTTP_USER_AGENT

nome e versão do browser utilizado pelo usuário

Métodos de HTTP

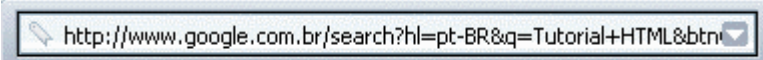
O protocolo HTTP utiliza vários métodos de manipulação e organização dos dados. Atualmente, dois métodos são mais utilizados para submeter dados de formulários: `POST` e `GET`. A diferença entre eles é a seguinte:

1. `POST`

- com `POST`, os dados entrados pelo formulário fazem parte do corpo da mensagem enviada para o servidor;
- a cadeia de caracteres com os dados do formulário é lida como uma entrada de dados padrão, de comprimento `CONTENT_LENGTH`;
- é possível transferir grande quantidade de dados.

2. `GET`

- com `GET`, os dados de entrada do script fazem parte do URL associado à consulta enviada para o servidor - por exemplo, nas consultas a catálogos e buscadores do tipo Yahoo e Google:



- a cadeia de entrada é colocada na variável de ambiente `QUERY_STRING`;
- suporta somente até 128 caracteres;
- como os dados da consulta fazem parte do URL, ela pode ser encapsulada em um URL de link ou bookmark;
- um detalhe é que os dados de entrada assim submetidos são copiados no registro de acessos ao site; portanto, *informações que exigem segurança não devem ser manipuladas por este método.*

Codificação de URLs

Quando os dados são trocados entre o servidor Web e o navegador, como no caso de formulários que usam o método `GET`, a codificação da URL garante que caracteres com significado especial sejam mascarados e não provoquem erros. Esta codificação é feita automaticamente pelo navegador.

Caracter	Código Hexadecimal	Código Decimal
Dollar ("\$")	%24	36
Ampersand ("&")	%26	38
Mais ("+")	%2B	43
Vírgula (",")	%2C	44
Barra ("/")	%2F	47
Dois-pontos (":")	%3A	58
Ponto-vírgula (";")	%3B	59
Igual ("=")	%3D	61
Interrogação ("?")	%3F	63
'At' ("@")	%40	64
Espaço	%20	32
Aspas ("")	%22	34
Menor que ("<")	%3C	60
Maior que (">")	%3E	62
'Pound' ("#")	%23	35
Porcento ("%")	%25	37
Chave esquerda ("{"	%7B	123
Chave direita ("}"	%7D	125
Pipe (" ")	%7C	124
Barra invertida ("\"	%5C	92
Circunflexo ("^")	%5E	94
Til ("~")	%7E	126
Colchete esquerdo ("["	%5B	91
Colchete direito ("]")	%5D	93
Acento Grave ("`")	%60	96

Folhas de Estilo

Um avanço interessante na linguagem HTML após a versão 3.2 foi a introdução das *Style Sheets* ou *Cascading Style Sheets*. Essas *folhas de estilo* permitem o uso de formatações uniformes em um site, de maneira bastante "econômica".

Já vimos que para poder formatar um texto, era preciso escrever uma marcação parecida com:

```
<h3><font face="Arial" color="#4A7D7B">Um título genérico</font></h3>
<p><font face="Arial" size="2">Um texto genérico com algum </font><font
face="Arial" size="2" color="red">destaque qualquer</font><font face="Arial"
size="2"> junto, após um título genérico, etc.</font></p>
```

para ter o resultado:

Um título genérico

Um texto genérico com algum destaque qualquer junto, após um título genérico, etc.

Acontece que, de um documento para outro, pode acontecer de esquecermos o tamanho da fonte usada no texto, qual a fonte ou a cor dos títulos de determinada subseção, e a uniformidade de apresentação das páginas acaba ficando prejudicada.

Com as folhas de estilo, podemos declarar estilos apropriados para seções de texto, aplicando esses estilos sem nos preocuparmos com detalhes de fontes, cor e tamanhos.

E mais: se for necessário modificar algum dia as cores de todos os títulos ou dos destaques ao longo dos textos, simplesmente alteramos a folha de estilo, atualizando imediatamente a apresentação de todos os documentos que fazem referência a ela.

Para o exemplo acima, poderíamos criar a seguinte folha de estilo:

```
BODY { font: 10pt Arial }
H3 { color:#4A7D7B }
.destaque { color: red }
```

E assim, para ter o mesmo resultado do exemplo acima, a formatação seria muito mais simples que a primeira:

```
<h3>Um título genérico</h3>
<p>Um texto genérico com algum <span class="destaque">destaque qualquer</span>
junto, após um título genérico, etc.</p>
```

A definição da folha de estilo deve ficar dentro de <HEAD>, da seguinte forma, se a folha for definida dentro do mesmo documento em que está sendo usada:

```
<HEAD>
...
<STYLE TYPE="text/css">
  BODY { font: 10pt Arial }
  H3 { color:#4A7D7B }
  .destaque { color: red }
</STYLE>
...
</HEAD>
```

Ou então, quando a folha de estilo é definida externamente:

```
<HEAD>
...
<LINK REL="stylesheet" HREF="folha_de_estilo.css">
...
</HEAD>
```

Neste caso, uma mesma folha de estilo pode ser usada por vários documentos HTML, que também poderão ter suas próprias folhas de estilo internas.

Duas formatações muito usadas com o uso de folhas de estilo são as que definem a apresentação geral do documento (fundo, cor e fonte de texto) e as que modificam a apresentação dos links.

Folha de estilo para BODY

São várias as opções de configuração, mas aqui estão somente as mais comuns:

Definindo a cor de fundo da página:

```
body { background-color: #D4CCC4 }
```

Definindo a cor de fundo e a imagem de fundo da página:

```
body { background-color: #D4CCC4;
      background-image: url("/icones/fundopag.jpg") }
```

Definindo a cor de fundo e fixando a imagem de fundo da página (não funciona com Netscape 4 e versões anteriores):

```
body { background-color: #D4CCC4;
      background-image: url("/icones/fundopag.jpg");
      background-repeat: no-repeat;
      background-attachment: fixed }
```

Com essas definições acima, não é preciso repetir a declaração de cor e imagem de fundo na etiqueta <BODY> da página.

Em particular, o Internet Explorer reconhece formatações para a barra de rolagem das páginas, com uma folha de estilo que define as cores de cada componente:

```
body { scrollbar-face-color: white;
      scrollbar-highlight-color: yellow;
      scrollbar-shadow-color: blue;
      scrollbar-darkshadow-color: black;
      scrollbar-3dlight-color: red;
      scrollbar-arrow-color: gray;
      scrollbar-track-color: green }
```

Obs.: Essas cores podem ser definidas em RGB, como #ffff00; o exemplo acima apenas serve para que o efeito de cada componente fique bem definido (experimente!).

Folha de estilo para os links

A definição geral de estilo para links inclui três elementos:

```
a:link { color:#0000ff }
a:hover { color:#ffff00 }
a:visited { color:#c0c0c0 }
```

Além da cor, podemos indicar se queremos o link sublinhado ou não; abaixo, definimos o link não sublinhado, o efeito de 'mouse-over' sublinhado e o link visitado com efeito de texto riscado:

```
a:link { color:#0000ff; text-decoration: none }
a:hover { color:#ffff00; text-decoration: underline }
a:visited { color:#c0c0c0; text-decoration: line-through }
```

Outra opção é definir uma cor de fundo para o link, por exemplo:

```
a:hover { color:#ffff00; text-decoration: none; background:#0000aa }
```

Uma folha de estilo com definições para o documento e para os links seria declarada, dentro de um documento, desta forma:

```
<style type="text/css">
  body { background-color: #D4CCC4;
        background-image: url("/icones/fundopag.jpg") }
  a:link { color:#0000ff; text-decoration: none }
  a:hover { color:#ffff00; text-decoration: underline }
  a:visited { color:#c0c0c0; text-decoration: line-through }
</style>
```