

ESCOLA DOM JOÃO BECKER
TÉCNICO EM INFORMÁTICA
PROGRAMAÇÃO PARA INTERNET

APOSTILA

Luciano Juliano Dutra Escobar

Porto Alegre

2023

Sumário

- 1 – Introdução
- 2 – Edição de documentos em HTML
- 3 – Publicação de documentos na internet (sites)
- 4 – Documentos HTML básico e seus componentes
- 5– Seção <head>
- 6– Seção <body>
- 7- Fontes
- 8- Cabeçalhos
- 9- Separadores
- 9.1- Quebra de linha
- 9.2- Parágrafos
- 10- Linha horizontal
- 11 Comentários
- 12 Listas em HTML
 - 12.1 Listas com marcadores
 - 12.2 Listas numeradas
 - 12.3 Listas e sub-listas
- 13 Formatação de frases
- 14 Ligações (uso de links)
 - 14.1 Atributos
 - 14.2 Caminhos
 - 14.3 Indicadores
 - 1.4 Inserindo imagens
 - 1.5 Utilizando imagens para outra página
 - 1.6 Frames
 - 1.7 Frameset, Frame, Noframe
- 15 Formulários eletrônicos
 - 15.1 Atributos para o form

15.2 Get

15.3 Post

15.4 Input

15.5 Elementos type

15.6 Radio

15.7 Password

15.8 Checkbox

15.9 Submit

15.10 Reset

15.11 Textarea

15.12 Select

1. Introdução

É comum as pessoas e principalmente os iniciantes em tecnologia da informação (TI), confundir linguagem de programação com HTML. O primeiro necessita de uma lógica de programação, já o segundo não. A Linguagem de Marcação de Hipertexto (HTML) é uma linguagem de computador que compõe a maior parte das páginas da internet e dos aplicativos online. Um hipertexto é um texto usado para fazer referência a outros textos, enquanto uma linguagem de marcação é composta por uma série de marcações que dizem para os servidores da web qual é o estilo e a estrutura de um documento.

Mas afinal, por que o HTML não é considerado uma linguagem de programação? O HTML não pode criar funcionalidades dinâmicas. Ao invés disso, os usuários podem criar e estruturar seções, parágrafos e links usando elementos, tags e atributos.

É uma linguagem de uso comum para o desenvolvimento web, navegação na internet e documentação. Também vale notar que o HTML agora é considerado um padrão oficial da internet. O World Wide Web Consortium (W3C) mantém e desenvolve especificações do HTML, além de providenciar atualizações regulares.

Esta apostila vai cobrir o básico sobre HTML, incluindo como ele funciona, seus prós e contras, como ele se relaciona com o CSS, o JavaScript e muito mais.

O HTML foi criado em 1991 na Suíça, por Tim Berners-Lee, um físico britânico, cientista da computação e professor do MIT. Inicialmente o HTML foi projetado para interligar instituições de pesquisas próximas, e compartilhar documentos com facilidade.

Documentos HTML são arquivos que terminam com uma extensão .html ou .htm. Um navegador lê o arquivo HTML e renderiza o seu conteúdo para que os usuários da internet possam vê-lo.

Todas as páginas HTML possuem uma série de elementos, que consistem num conjunto de tags e atributos. Os elementos HTML são os tijolos de construção de uma página da internet. Uma tag diz para o navegador onde um elemento começa e termina, enquanto um atributo descreve as características de um elemento.

As três principais partes de um elemento são:

Tag de abertura – usada para dizer onde um elemento começa a ter efeito. A tag é cercada de colchetes angulares para abertura e fechamento. Por exemplo, use a tag de abertura `<p>` para criar um parágrafo.

Conteúdo – essa é a parte que os usuários verão, portanto, é onde se coloca o conteúdo apresentado ao usuário.

Tag de fechamento – igual à tag de abertura, mas com uma barra antes do nome do elemento. Por exemplo, `</p>` para encerrar um parágrafo.

A combinação dessas três partes vai criar um elemento HTML.

Outra parte crucial de um elemento HTML é o seu atributo, que possui duas seções — um nome e um valor de atributo. O nome identifica a informação adicional que um usuário deseja acrescentar, enquanto o valor de atributo fornece mais especificações.

Por exemplo, um elemento de estilo que adiciona a cor roxa e uma fonte da família verdana ficará assim:

```
<p style="color:purple;font-family:verdana">É assim que você adiciona um parágrafo no HTML.</p>
```

A maioria dos elementos possui uma tag de abertura e de fechamento, mas alguns não precisam fechar a tag para funcionar. Esse é o caso dos elementos vazios. Eles não usam uma tag de fechamento pois não têm conteúdo:

```

```

Essa tag de imagem possui dois atributos — um atributo `src` (que é um caminho de imagem) e um atributo `alt` (que é o texto de descrição). Contudo, ele não tem conteúdo nem uma tag de fechamento.

Finalmente, cada documento HTML deve começar com uma declaração `<!DOCTYPE>` para informar ao navegador qual é o tipo de documento. Com o HTML5¹, a declaração doctype HTML pública será:

```
<!DOCTYPE html>
```

¹O maior upgrade da linguagem foi o lançamento do HTML5 em 2014. Diversas novas tags semânticas foram adicionadas que revelam o significado do seu próprio conteúdo, como `<article>`, `<header>`, e `<footer>`.

2. Edição de documentos em HTML

A maneira correta de editar o código HTML é usar um editor de texto simples ou um editor específico para trabalhar com código. Sendo que o mais indicado para editar arquivos HTML são editores de código.

Os editores de código não farão qualquer alterações não autorizadas dentro do arquivo HTML, o que significa que não terá quaisquer surpresas desagradáveis depois de gravar as suas alterações.

É possível editar arquivos HTML em um servidor?

Sim, é possível.

Basta que você tenha acesso ao servidor onde está hospedado seu site. Desta forma, você poderá editar arquivos HTML seja em sua máquina ou em um servidor.

Como encontrar uma boa ferramenta para editar arquivos HTML?

Editores de texto podem ser encontrados em qualquer sistema operacional.

O bloco de notas é o editor de texto simples mais conhecido no sistema operacional Windows. Para acessá-lo, basta utilizar o botão direito do mouse em cima do arquivo HTML a ser editado.

O bloco de notas irá abrir o arquivo HTML em formato de código para que possa começar a editar imediatamente.

Mas o ideal para editar arquivos HTML é através de uma ferramenta própria para isso, um editor de código. Existem diversos tipos de editores de código que podem ser usados para te ajudar a como editar arquivos HTML.

A principal diferença entre editores de código e os editores simples, é que um bom editor de código possui marcadores de sintaxe embutidos, enquanto um editor de texto simples exibe o código apenas em preto e branco. O benefício do realce de sintaxe é que ele torna a edição muito mais fácil. Existem diversos editores de textos no mercado, porém irei listar três muito utilizados pelos desenvolvedores e que possuem um suporte muito eficiente na hora de criar e editar arquivos em HTML.

Notepad++

Este é o editor de arquivos HTML mais simples para iniciantes. Você poderá alterar seus arquivos HTML e ter recursos adicionais que não são encontrados em editores simples.

O Notepad++ foi especialmente concebido para editar o código-fonte. Por exemplo, o termo “++” (mais-mais) no nome é uma referência ao operador

incremental em linguagens de programação como C, C++, Java e JavaScript. No entanto, é possível também editar arquivos HTML com este editor.

O Notepad++ é útil sempre que precisar de fazer alterações significativas num arquivo de texto. É o mais indicado e o mais simples editor de código que pode ser utilizado.

Sublime Text

Sublime é um editor desenvolvido por uma companhia de Sydney, este software também se enquadra na categoria de freemium. Freemium quer dizer que você pode usar o Sublime de graça, mas você tem que comprar uma licença para usufruir dos recursos por completo. O Sublime oferece um suporte ótimo que garante que o programa seja constantemente atualizado. Os usuários podem adicionar plugins feito pela comunidade ou criar os seus próprios plugins. Nós acreditamos que a versão gratuita do Sublime é mais que adequada. Porém, se você acha que precisa de mais recursos, você pode adquirir a licença quando quiser.

VS Code

Lançado em 2015 pela Microsoft, o VS Code é um dos principais editores de código fonte da atualidade. Muitos programadores aprendem como usar Visual Studio Code (VS Code) desde o começo dos estudos devido à importância do programa na rotina do profissional de programação.

O Visual Studio Code é um editor de texto conhecido por ser um editor de código aberto muito intuitivo. Além disso, ele também é muito popular por ser multiplataforma e estar disponível para os principais sistemas operacionais, que são Linux, Mac e Windows.

3. Publicação de documento na internet (sites)

Para poder subir o seu site e colocá-lo ele no ar, confira o passo a passo abaixo.

Escolher uma hospedagem de confiança

A criação e a manutenção de um site são coisas que você precisa levar a sério. É crucial escolher uma provedora de hospedagem de primeira linha, que tenha todos os recursos necessários para dar partida no seu projeto.

Existem diversos fatores você deve considerar na hora de escolher um provedor de hospedagem: suporte instantâneo, controle sobre o armazenamento da hospedagem, espaço para crescimento, garantia de reembolso e alguns benefícios extras, tais como, garantir que seu provedor de hospedagem ofereça certificados SSL, recurso essencial para manter seu site, domínio e usuários protegidos.

Escolha o método de upload

O próximo passo é escolher a ferramenta certa para fazer o trabalho. Essas são as 4 ferramentas mais utilizadas para subir um site:

Gerenciador de Arquivos

Um gerenciador de arquivos é uma ferramenta baseada em navegador que possui todos os recursos necessários para cuidar dos arquivos e diretórios do seu site. Esta etapa depende muito de cada servidor de hospedagem e seus planos, mas basicamente o Gerenciador de Arquivos tem um limite de upload. Se os arquivos do seu site tiverem mais de 256 MB, você deve escolher a próxima ferramenta.

Protocolo de Transferência de Arquivos (FTP)

A maioria dos provedores de hospedagem oferece suporte ao FTP — ou Protocolo de Transferência de Arquivos, que pode-se usar para conexão com um cliente FTP, ou seja, é importante que tenha um instalado no seu PC.

Ferramenta de Importação Automática de Sites

Você pode usar esse recurso para extrair um arquivo de site para o diretório public_html. A ferramenta suporta os formatos .zip, .tar e .tar.gz, com um limite de upload de 256 MB.

Plugins de Migração do WordPress

Se você planeja usar o WordPress, existem diversas maneiras de fazer o upload do seu site para o CMS. Um dos métodos mais fáceis é usando um plugin de

migração do WordPress como o All in One WP Migration. Depois de instalar e ativar o plugin, o resto dos passos são autoexplicativos.

O limite do tamanho do upload vai depender do seu provedor de hospedagem.

Faça Upload dos Arquivos para o seu Site

Agora você já sabe quais são as melhores ferramentas para fazer upload do seu site, escolha um e bora colocá-lo no ar!

Mova os Arquivos do Site para o Diretório Raiz Principal

Em alguns casos, um diretório adicional será criado quando você fizer upload dos arquivos do seu site. O resultado disso é que alguns visitantes serão direcionados para `seudominio.com/subpasta` ao invés de `seudominio.com`.

Para evitar que isso aconteça, você precisará garantir que os arquivos estão localizados no diretório raiz do seu domínio, que é `public_html`.

Você pode usar o Gerenciador de Arquivos para mover os arquivos do seu site. Tudo que você precisa fazer é clicar com o botão direito na subpasta, selecionar a opção Move (Mover) e definir o `public_html` como destino.

Importe sua Base de Dados

Se o seu site usa uma base de dados, você vai precisar importá-la junto dos seus arquivos:

Crie uma nova base de dados MySQL e um novo usuário.

Acesse seu novo banco de dados e faça upload do arquivo de backup da database.

Use a seção Import (Importar) para fazer upload do arquivo de backup da sua base dados.

Atualize as informações de conexão da MySQL database (nome da base de dados, host, usuário, senha) nos arquivos de configuração.

Confira se o Site Funciona

Assim que tiver finalizado o seu upload, é hora de uma verificação final!

Se o seu domínio já está registrado e apontando para a sua provedora de hospedagem, insira o seu nome de domínio no navegador e veja se ele está direcionando para o seu site.

Tenha em mente que mudanças de DNS podem levar até 48 para serem propagadas pelo mundo inteiro. Por isso, pode necessário esperar algum tempo se você acabou de apontar o domínio para os servidores da sua provedora.

Use uma ferramenta online como o whatsmydns.net para conferir a situação da propagação do seu DNS. Basta inserir o seu domínio na ferramenta e ela vai verificar os registros de DNS em comparação com múltiplos servidores.

4. Documentos HTML básico e seus componentes

O *HTML* é parte fundamental das normas da web, em conjunto com outras tecnologias como o *CSS* e o *JavaScript*. Já que essa linguagem é a base para a construção de páginas, é importante conhecê-la para criarmos o nosso próprio site na internet.

O documento *HTML* sempre inicia com o que chamamos de estrutura básica, esta estrutura é quase que imutável. Sempre será dessa forma e você sempre, sempre começará seu *HTML* começando por esse código. Geralmente os editores já têm atalhos para iniciar os documentos *HTMLs* com essa estrutura, logo, você não precisa se preocupar em decorá-la. Veja abaixo como ela se inicia:

```
1. <!DOCTYPE html>
2. <html lang="pt-br">
3.   <head>
4.     <title>Título da página</title>
5.     <meta charset="utf-8">
6.   </head>
7.   <body>
8.     Aqui vai o código HTML que fará seu site aparecer.
9.   </body>
10.</html>
```

É possível compreender o documento em *HTML* de uma maneira muito simples, através de uma divisão de blocos das *tags* essenciais, conforme a seguinte estrutura:

- Definição do documento (*doctype*)
- Cabeça (*head*)
- Corpo (*body*)
- *Doctype* - Definindo o documento

Uma coisa importante: SEMPRE deve existir o *doctype*, que é este código `<!DOCTYPE html>`.

O *doctype* não é uma *tag HTML*, mas uma instrução para o navegador e outros programas que podem ler seu site, que o código encontrado ali é um código *HTML*. Assim eles sabem o que fazer para mostrar seu site da melhor forma possível. Lembre-se: o *doctype* é OBRIGATÓRIO e deve ser sempre a PRIMEIRA LINHA do seu documento.

4.1. HEAD

Contém informações que não são transpostas visivelmente para o usuário/leitor do documento. São dados implícitos, de uso e controle do documento: vinculação com outros arquivos, aplicação de lógica de programação de *scripts* e metadados. Na prática, todo o conteúdo do cabeçalho fica delimitado entre a abertura e fechamento da *tag head*.

4.2. BODY

Trata-se do documento em si, ou seja, a informação legível para o usuário/leitor do documento. É todo e qualquer texto que se deseja apresentar, assim como toda e qualquer forma de mídia de saída (imagens, sons, miniaplicativos embutidos, conteúdo multimídia, etc). Além disso, toda a apresentação de entrada de dados (formulários) também se aplica nesta seção do documento. Na prática, o corpo do documento é delimitado pelo par de *tags* `<body>` e `</body>`.

Este é o preceito básico que deve estar muito bem claro para você, onde as marcações se aplicam, e quais são os resultados deste modelo. Por exemplo: se desejar informar um conteúdo textual para saída legível ao usuário do seu sistema web, esta marcação deverá obrigatoriamente estar no bloco do corpo da página. Ainda, para definir qual o tipo de codificação da página (uma meta informação do documento), esta deve obrigatoriamente estar marcada no cabeçalho do mesmo documento.

Dentro do elemento *BODY* sua estrutura de página terá os elementos semânticos da construção da sua página, onde serão declarados e identificados cabeçalhos, rodapé, conteúdo principal, etc.

Agora que você já sabe o que é o *HTML*, vamos ver os principais marcadores que definem a função dos elementos que fazem parte da página.

4.3. Parágrafo (`<p>`)

A *tag* `<p>` deve ser aberta e fechada para definir um parágrafo.

`<p>`Este é um parágrafo em HTML.`</p>`

4.4. Cabeçalho (`<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>`)

As *heading tags*, ou *tags* de cabeçalho, ajudam a criar uma hierarquia entre as partes do texto, separando-o em seções. O `<h1>` é o cabeçalho de maior relevância, seguido pelo `<h2>`, `<h3>` e assim sucessivamente até o `<h6>`, de menor importância e assim por diante até o `<h6>` se for o caso.

1. `<h1>Título principal</h1>`
2. `<p>Parágrafo de introdução.</p>`

3. <h2>1. Primeiro intertítulo</h2>
4. <p>Parágrafo da primeira seção.</p>
5. <p>Mais um parágrafo da primeira seção.</p>

4.5. Imagem ()

A *tag* de imagem não precisa ser fechada, já que o seu conteúdo é definido na própria *tag* com o atributo “src”.

```

```

4.6. Âncora (<a>)

A *tag* de âncora define um link para outra página. O atributo “href” determina o destino desse link.

```
<a href="https://rockcontent.com/br/blog/arquitetura-da-informacao/">Arquitetura da Informação</a>
```

4.7. Lista ordenada ()

Uma *ordered list*, ou lista ordenada, é exibida com elementos numerados. Cada elemento deve ser marcado com a *tag* (*list item*).

```
<ol>
  <li>SEO</li>
  <li>Links patrocinados</li>
  <li>Redes sociais</li>
</ol>
```

4.8. Lista não ordenada ()

A *tag unordered list*, ou lista não ordenada, mostra os elementos como *bulletpoints*.

```
<ul>
  <li>Automação de marketing</li>
  <li>Geração de leads</li>
  <li>E-mail marketing</li>
</ul>
```

4.9. Tabela (<table>)

A *tag* de tabela exibe informações no formato de linhas e colunas. Dentro da *tag* <table>, usamos as *tags* <tr> para representar as linhas, <th> para representar os cabeçalhos e <td> para representar cada célula.

```
<table>
  <tr>
    <th>Nome</th>
    <th>Sobrenome</th>
  </tr>
  <tr>
    <td>Jonah</td>
```

```

        <td>Berger</td>
    </tr>
    <tr>
        <td>Ann</td>
        <td>Handley</td>
    </tr>
    <tr>
        <td>Neil</td>
        <td>Patel</td>
    </tr>
</table>

```

4.10. Comentário (<!-- -->)

Usamos esta *tag* para escrever algum comentário que ajude no entendimento do código. Tudo que estiver escrito entre os marcadores <!-- e --> não será mostrado ao usuário.

```

<p>O HTML é fundamental para quem deseja criar um site na web.</p> <!-- Este
é um parágrafo em HTML -->

```

5. Seção `<head>`

O *head* de um documento *HTML* é a parte que não é exibida no navegador da *Web* quando a página é carregada. Ele contém informações como *title*, *links* para `<css>` (se você deseja modelar seu conteúdo *HTML* com *CSS*), *links* para *favicons* personalizados e outros metadados (dados sobre o *HTML*, como quem o escreveu, e palavras-chave importantes que descrevem o documento.)

O cabeçalho *HTML* é o conteúdo do elemento `<head>` — ao contrário do conteúdo do elemento `<body>` (que são exibidos na página quando carregados no navegador), o conteúdo do cabeçalho não é exibido na página, em vez disso, o trabalho do cabeçalho é conter metadados sobre o documento. No exemplo seguinte, o cabeçalho é bem simples:

```
<head>
  <meta charset="utf-8">
  <title>Minha página de teste</title>
</head>
```

Em páginas maiores, o cabeçalho pode ter mais conteúdo.

5.1. Adicionando um título

Nós já vimos o elemento `<title>` em ação — ele pode ser usado para adicionar um título ao documento, mas pode ser confundido com o elemento `<h1>`, que é usado para adicionar um título de nível superior ao conteúdo do *body*, as vezes também é associado como o título da página. Mas são coisas diferentes!

O elemento `<h1>` aparece na página quando é carregado no navegador, geralmente isso deve ser usado uma vez por página, para marcar o título do conteúdo da sua página, (o título da história, ou da notícia, ou o que quer que seja apropriado para o uso).

O elemento `<title>` é um metadado que representa o título de todo o documento *HTML* (não o conteúdo do documento).

5.2. Metadados: o elemento `<meta>`

Metadados é dado que descreve dado, e *HTML* possui uma maneira "oficial" de adicionar metadados a um documento, o elemento `<meta>`. Existem muitos tipos diferentes de elementos `<meta>` que podem ser incluídos no `<head>` da sua página, mas não falaremos de todos eles agora, pois seria muito confuso. Em vez disso, explicaremos algumas coisas que você pode ver comumente, apenas para lhe dar uma ideia.

Muitos elementos `<meta>` incluem atributos de *name* e *content*:

O *name* especifica o tipo de elemento meta que é; que tipo de informação contém.

O *content* especifica o conteúdo real da *tag meta*.

Dois desses meta-elementos que são úteis para incluir na sua página definem o autor da página e fornecem uma descrição concisa da página. Vejamos um exemplo:

```
<meta name="autor" content="Luciano J. D. Escobar">
<meta name="descricao" content="Esta apostila foi elaborada pensando no
aprendizado de quem nunca ouviu falar em HTML e para consultas rápidas no caso de
um desenvolvedor com mais prática.">
```

Especificar um autor é útil de muitas maneiras: é útil para poder descobrir quem escreveu a página, se quiser enviar perguntas sobre o conteúdo que você gostaria de contatá-la. Alguns sistemas de gerenciamento de conteúdo possuem ferramentas para extrair automaticamente as informações do autor da página e disponibilizá-las para seus propósitos.

Especificar uma descrição que inclua palavras-chave relacionadas ao conteúdo da sua página é útil porque tem potencial para tornar sua página mais alta nas pesquisas relevantes realizadas nos mecanismos de busca (tais atividades são denominadas *Search Engine Optimization* ou *SEO*).

5.2.1. Especificando a codificação de caracteres do seu documento

Observe o exemplo abaixo:

```
<meta charset="utf-8">
```

Este elemento simplesmente especifica a codificação de caracteres do documento, o conjunto de caracteres que o documento está autorizado a usar “*utf-8*” é um conjunto de caracteres universal que inclui praticamente qualquer caractere de qualquer linguagem humana. Isso significa que sua página *web* poderá lidar com a exibição de qualquer idioma; portanto, é uma boa ideia configurar isso em todas as páginas *web* que você criar.

5.2.2. Outros tipos de metadados

Ao navegar pela web, você também encontrará outros tipos de metadados. Muitos dos recursos que você verá em sites são criações proprietárias, projetados para fornecer a determinados sites (como sites de redes sociais) informações específicas que eles podem usar.

Por exemplo, *Open Graph Data* é um protocolo de metadados que o *Facebook* inventou para fornecer metadados mais ricos para sites.


```

<meta property = "og:image" content=
"https://developer.mozilla.org/static/img/opengraph-logo.png">
<meta property="og:description" content="A Mozilla Developer Network (MDN)
fornece informações sobre tecnologias Open Web, incluindo HTML, CSS e APIs para
ambos os sites da Web e aplicativos HTML5. Ele também documenta produtos Mozilla,
como o sistema operacional Firefox.">
<meta property = "og:title" content="Mozilla Developer Network">

```

Um efeito disso é que, quando você liga o site *MDN* no *facebook*, o *link* aparece junto com uma imagem e descrição: uma experiência mais rica para usuários.

O *Twitter* também possui seus próprios metadados proprietários, o que tem um efeito semelhante quando o *URL* do *site* é exibido no *twitter.com*. Por exemplo:

```

<meta name="twitter:title" content="Mozilla Developer Network">

```

5.3. Adicionando ícones personalizados ao seu *site*

Para enriquecer ainda mais o design do seu site, você pode adicionar referências a ícones personalizados em seus metadados, e estes, serão exibidos em determinados contextos. O mais usado é o *favicon* (abreviação de "*favorites icon*", referindo-se ao seu uso nas listas "favoritos" nos navegadores).

O humilde *favicon* existe há muitos anos. É o primeiro ícone desse tipo: um ícone de 16 *pixels*² usado em vários lugares. Você pode ver (dependendo do navegador) ícones favoritos exibidos na guia do navegador que contém cada página aberta e ao lado de páginas marcadas no painel de favoritos.

Um *favicon* pode ser adicionado à sua página, salvando-o no mesmo diretório que a página de índice do site, salvo no formato *“.ico”* (a maioria dos navegadores suportará *favicons* em formatos mais comuns como *“.gif* ou *.png”*).

Adicionando a seguinte linha ao *HTML* *<head>* para fazer referência a ele:

```

<link rel="shortcut icon" href="favicon.ico" type="image/x-icon">

```

Há muitos outros tipos de ícones para considerar nestes dias também. Por exemplo:

```

<!-- iPhone não-Retina, iPod Touch e dispositivos Android 2.1+: -->
<link rel="apple-touch-icon-precomposed"
href="https://developer.mozilla.org/static/img/favicon57.png">
<!-- favicon básico -->
<link rel="shortcut icon"
href="https://developer.mozilla.org/static/img/favicon32.png">

```

Os comentários explicam onde cada ícone é usado, esses elementos cobrem coisas como fornecer um ícone de alta resolução agradável para usar quando o site é salvo na tela inicial do iPad.

O objetivo principal aqui é permitir o entendimento do que são essas coisas, no caso de você encontrá-las enquanto navega no código-fonte dos outros sites.

Nota: se o seu site usa uma Política de Segurança de Conteúdo (CSP) para aumentar sua segurança, a política se aplica ao *favicon*. Se você encontrar problemas com o *favicon* não carregando, verifique se a diretiva *img-src* (pt-br) do cabeçalho *Content-Security-Policy* não está impedindo o acesso a ele.

6. Seção <body>

Todos nós conhecemos. Mas será que estamos utilizando *de verdade* ele?

```
<body>
<!-- todo o conteudo do site aqui dentro -->
</body>
```

O body é o elemento no nosso documento responsável por ser o container de tudo que é conteúdo do site.

Diferente do <head>, o que está dentro do <body> é o que fica ou ficará disponível para visualização do visitante.

As informações que queremos transmitir. Títulos, textos, imagens e etc. O *body* é uma *tag HTML*, e como tal, aceita atributos.

http://www.w3schools.com/tags/tag_body.asp

Os mais interessantes, são o *id* e a *class*.

```
<body id="home">
```

6.1. Usar o body com um ID

É uma proposta interessante, e nos ajuda a economizar alguns elementos de estruturação de layout de vez em quando. Por exemplo, aqueles layouts onde a *home* do site difere por poucas coisas das páginas internas, como: topo mais alto, rodapé inexistente e tal.

Esse tipo de situação, podemos facilmente resolver usando o atributo *ID* no *body*.

Sendo a *home*, estilizamos o topo diferente do que ele seria nas internas.

```
body#home #header { height: 200px; }
#header { height: 100px; }
```

A marcação *html* fica intacta, evitamos, coisas como: *#header_home*.

E isso nos levará a escrever um *css* mais bonito também. Afinal, fizemos um condicional no *css*. Mas isto é um assunto para outro momento.

6.2. Usar o body com uma ou mais classes

Começamos falando do *ID*, porém existe também o *class*.

```
<body class="home">
```

Um bom motivo de uso é o fato de poder usar mais de uma vez. Com classes, temos todos os benefícios que foram citados do *ID*:

```
body.home #header { height: 200px; }
#header { height: 100px; }
```

E é possível colocar mais de uma, veja:

```
<body class="cursos html">
```

A motivação disso, é imaginemos um menu *dropdown*.

```
<li>Cursos
  <ul>
    <li>HTML</li>
    <li>CSS</li>
    <li>JavaScript</li>
  </ul>
</li>
```

Assim cada página interna, teria uma combinação de *class* para o *body* diferente:

```
class="cursos html", class="cursos css", class="cursos javascript"..
```

A vantagem disso, é que a *class="cursos"*, é comum a todos eles, e no *css* podemos fazer facilmente um destaque naquele item de menu, se estivermos dentro de um dos filhos. Mais uma vez, levamos o poder do *css* à outro nível.

Tendo a *class html*, *css*.. e sabendo que cada página interna terá somente um título H1, podemos por exemplo, fazer o H1 ser azul na página de *HTML*, ser vermelho na página de *css*, e ser de qualquer outra cor por padrão, nas páginas em que não definirmos nada.

```
.html h1 { color: #00f; }
.css h1 { color: #f00; }
h1 { color: #000; }
```

A grande sacada disso é não precisarmos adicionar nada mais no *html*, além das respectivas classes no *body*.

7. Fontes

Existem diversos tipos de fontes HTML disponíveis para se usar na construção de um site. Uma fonte que se encaixa com a proposta do seu projeto online vai melhorar o web design, fazendo com que os conteúdos fiquem mais fáceis de serem lidos.

Porém, escolher a opção certa pode ser difícil. Justamente porque existem muitas opções por aí. E até mesmo algumas fontes são muito parecidas com as outras para determinadas pessoas.

Conhecendo ou não, uma fonte HTML é uma peça fundamental do seu site. Ela não só permite que seus visitantes leiam os conteúdos mais facilmente, como também melhoram a estética de uma página.

Você precisa escolher a tipologia do seu site com bastante cuidado. Usar tipos muito pequenos ou muito grandes pode ter um impacto negativo no jeito como seu site se apresenta para os leitores.

Além disso, algumas fontes são seguras de usar em sites. Enquanto outras nem tanto. Vamos abordar isso em maiores detalhes mais abaixo.

Para mudar uma fonte de texto para HTML, você pode usar o atributo de estilo dentro da tag `<p>` que define o uso de parágrafos. Aqui vai um exemplo:

```
<p style = "font-family:courier,arial,helvetica;">  
Escreva seu texto aqui.  
</p>
```

Existem três tipos de fontes no exemplo acima (Courier, Arial e Helvética). A segunda e a terceira servem como backup caso a primeira não seja encontrada na página ou não esteja instalada.

7.1. Fontes *WEB* seguras

Fonte *Web Segura* é um termo usado para descrever uma fonte que é universalmente instalada em todos os computadores. Seu site vai carregar mais rápido quando você usar uma fonte *web* segura que já está armazenada na sua máquina.

E isso vai afetar o *SEO* do seu site, já que os mecanismos de busca contam o carregamento de uma página como um dos fatores de “ranqueamento” mais importantes.

É por isso que recomendamos que você sempre use uma fonte *web* segura. Isso garante que seus visitantes poderão vê-la facilmente enquanto mantém o desempenho do seu site lá em cima.

Tenha em mente que existem alternativas para fontes *web* seguras e elas geralmente compartilham algumas características das mais populares, como *Helvética*, *Futura*, etc. Por exemplo, as alternativas para *Sans Serif* como *Futura* são a *Noir Pro* e a *Lorin*.

Fora isso, diferente das fontes *web* seguras, as fontes alternativas não costumam estar instaladas nos Sistemas Operacionais.

Porém, pode ser que você queira usar alguma delas. Você pode achar, por exemplo, que determinada fonte é usada tão excessivamente que é melhor escolher outra para se diferenciar.

Apenas lembre-se de que, escolhendo usar uma fonte alternativa, pode ser que velocidade do seu site seja um pouco comprometida.

7.2. As 5 famílias de fontes existentes

Em tipografia, cada fonte é membro de uma destas 5 famílias que vamos mostrar abaixo. Essa semelhança é baseada puramente em design.

7.2.1. *Cursive* (exemplo: *Zapf-Chancery*)

As fontes na família *Cursive* imitam a caligrafia humana. É por isso que as letras são todas unidas, juntas e sem separações.

7.2.2. *Fantasy* (exemplo: *Star Wars*)

A família *Fantasy* geralmente tem elementos decorativos em cada letra, mas que ainda representam caracteres. Muitos livros de ficção ou filmes usam esse tipo de fonte nos seus títulos.

7.2.3. *Serif* (exemplo: *Times New Roman*)

A característica mais marcante dessa família de fonte é uma pequena linha nas extremidades de cada letra ou símbolo. Vários sites usam *Serif* para o corpo dos seus textos em conteúdos escritos.

7.2.4. *Sans-serif* (exemplo: *Helvética*)

Diferente da *Serif*, a *Sans-serif* não tem a pequena linha atrelada em cada letra ou símbolo. Além disso, a fonte geralmente tem uma largura parecida, apresentando um aspecto minimalista e moderno.

7.2.5. *Monospace* (exemplo: *Courier*)

Cada letra e símbolo da família de fontes *Monospace* ocupa exatamente o mesmo espaço horizontalmente. Ela é frequentemente usada em máquinas de escrever ou terminais de computadores.

7.6. Exemplos de fontes em HTML

As fontes que mencionadas abaixo são todas do tipo “*web seguras*”. Por conta disso, elas são compatíveis em todos os sistemas e numa variedade de dispositivos.

Cada uma delas têm características exclusivas. Então, é preciso considerar o estilo do seu site e a tonalidade da sua escrita antes de escolher a fonte certa.

- Arial

Arial é uma fonte da família *Sans-serif* e é uma das mais usadas no mundo. Muitos críticos dizem que essa fonte é uma aposta segura. Então tanto textos na internet quanto em jornais costumam apresentar a fonte Arial.

- Times New Roman

Essa fonte *HTML* é uma variação da velha fonte *Times* do grupo *Serif*. A Times New Roman é a escolha favorita para conteúdos publicados em livros e publicações impressas. Essa fonte até mesmo foi recomendada pela APA (Associação Americana de Psicologia).

- Helvética

Designers geralmente adoram a Helvética. E por isso que as pessoas sempre acertam ao escolhê-la como fonte. Sem surpresas, grandes marcas mundiais, como Jeep, Kawasaki, Motorola e BMW usam essa fonte nos seus logos.

- Courier New

Essa fonte é uma das alternativas à Courier. Ela é mais fina e visualmente mais atraente aos olhos quando a vemos numa tela. É por isso que aparelhos eletrônicos quase sempre usam Courier New.

- Verdana

Essa fonte pode ser lida em tamanhos pequenos em telas com resoluções menores. Muitas pessoas identificam a Verdana como uma fonte para telas, ainda que isso não seja de todo verdade. Por exemplo, o IKEA usa essa fonte não apenas para o site, mas também para catálogos.

7.7. Manipulando fontes com a tag ``

Para alterar o tipo de fonte de um texto em *HTML*, usa-se o atributo *face*, da *tag* ``. A sintaxe desta *tag* e do atributo é:

```
<font face="NOME DA FONTE"> Aqui é seu texto que está com a fonte "NOME DA FONTE" </font>
```

Para mudar a cor de um determinado texto, devemos colocar tal trecho entre as *tags* `` e ``, porém, devemos adicionar o atributo *color*, que vai definir a cor que você queira, baseado na Tabela de Cores.

Há 3 sintaxes para definir a cor do texto:

```
<font color="#XXXXXX"> Cor definida por 6 dígitos hexadecimais </font>
<font color="rgb(x,x,x)"> Cor definida por 3 números do RGB </font>
<font color="NOME DA COR"> Cor definida pelo nome </font>
```

Porém, assim no caso do atributo *bgcolor* da *tag* `<head>`, que muda a cor do fundo do site, o RGB só funciona em CSS.

Finalizando este tópico sobre manipulação e formatação das fontes de um texto, vamos ensinar como mudar o tamanho de uma fonte. Isso é feito através do atributo *size* (tamanho, em inglês) da *tag* ``. A sintaxe de uso desse atributo é a seguinte:

```
<font size="numero"> O tamanho da fonte deste texto eh "numero" </font>
```

Onde este "numero" pode variar de 1 até 7, onde o número 1 é o menor tamanho de fonte, e o número 7 é o maior tamanho possível de fonte.

7.8. Conclusão

Agora você já pode decidir qual é a melhor fonte para o estilo e o tom do seu site. Apenas preste atenção na usabilidade de um tipo em particular e avalie se ele se encaixa com o que você precisa.

Fica aqui uma dica de programador, os usuários de internet compartilham de alguns gostos definitivos, eles odeiam a fonte *Comic Sans*. As pessoas consideram essa fonte infantil, nada profissional e até mesmo esquisita.

Isso até se tornou um meme ao longo dos anos. E é por isso que não recomendamos que você a use. Existem opções que vão fazer seu site ficar muito mais atrativo visualmente.



NÃO ME USE

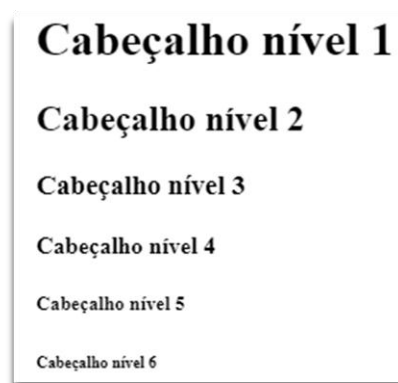
8. Cabeçalho

Os elementos *HTML* `<h1>` – `<h6>` representam seis níveis de título de seção, na qual o `<h1>` é o nível de seção mais alto e `<h6>` é o mais baixo.

Elementos de cabeçalho são implementados em seis níveis, `<h1>` é o mais importante e `<h6>` é o de menor importância. Um elemento de cabeçalho descreve brevemente o tópico da seção em que ele está. As informações de cabeçalho podem ser usadas por agentes de usuário, por exemplo, para construir uma tabela de conteúdos para um documento automaticamente.

O seguinte código mostra, em uso, todos os níveis de cabeçalho e ao lado um exemplo da saída mostrada em tela para o leitor.

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



Dica de programador: não use níveis menores para diminuir o tamanho da fonte do cabeçalho: use a propriedade *CSS* *font-size*. Evite pular níveis de cabeçalhos: sempre comece com `<h1>`, depois `<h2>` e assim por diante. Tente, também, ter pelo menos um cabeçalho de primeiro nível em uma página.

Em *HTML5*, use o elemento `<section>` para definir o *outline* de um documento. Cabeçalhos fornecem títulos para seções e subseções. Você também pode agrupar um cabeçalho e seu conteúdo usando o elemento `<div>`².

² É um container genérico para conteúdo de fluxo, que de certa forma não representa nada. Ele pode ser utilizado para agrupar elementos para fins de estilos (usando *class* ou *id*), ou porque eles compartilham valores de atributos, como *lang*.

9. Separadores

Para organizarmos melhor o nosso site, precisamos de separadores em *HTML*. Iremos falar um pouco sobre cada um.

9.1. Quebra de linhas

O elemento *HTML* *quebra-de-linha* `
` produz uma quebra de linha em um texto (*carriage-return*). É útil para escrever poemas ou um endereço, onde a divisão de linha é significativa.

Atenção: não use `
` para aumentar o espaço entre as linhas de texto; para isso use a propriedade *CSS* *margin* ou o elemento `<p>`.

Exemplo:

```
Mozilla Foundation<br>
1981 Landings Drive<br>
Building K<br>
Mountain View, CA 94043-0801<br>
USA
```

Resultado:

A rectangular box containing the text: Mozilla Foundation, 1981 Landings Drive, Building K, Mountain View, CA 94043-0801, USA. The text is arranged in five lines, with each line starting at the same left margin, demonstrating the effect of the
 tag.

9.2. Parágrafos

O elemento *HTML* `<p>` representa um parágrafo. Em mídias visuais, parágrafos são representados como blocos indentados de texto com a primeira letra avançada e separados por linhas em branco. Já em *HTML*, parágrafos são usados para agrupar conteúdos relacionados de qualquer tipo, como imagens e campos de um formulário.

Parágrafos são elementos *block-level*, e fecharão automaticamente caso outro elemento *block-level* inicie antes da *tag* de fechamento `</p>`.

Como exemplo:

```
<p>Este é o primeiro parágrafo do texto. Este é o primeiro parágrafo do texto.
Este é o primeiro parágrafo do texto. Este é o primeiro parágrafo do texto.</p>

<p>Este é o segundo parágrafo do texto. Este é o segundo parágrafo do texto.
Este é o segundo parágrafo do texto. Este é o segundo parágrafo do texto.</p>
```

Como resultado:

Este é o primeiro parágrafo do texto. Este é o primeiro parágrafo do texto. Este é o primeiro parágrafo do texto. Este é o primeiro parágrafo do texto.

Este é o segundo parágrafo do texto. Este é o segundo parágrafo do texto. Este é o segundo parágrafo do texto. Este é o segundo parágrafo do texto.

Subdividir um conteúdo em parágrafos torna um texto mais acessível. Leitores de tela e outras tecnologias assistidas providenciam atalhos que permitem a navegação entre parágrafos. Possibilitando, então, uma leitura rápida do texto.

Utilizar elementos <p> vazios para adicionar linhas em branco entre parágrafos é uma abordagem problemática para os que dependem das tecnologias leitoras de tela. O leitor anunciará a existência de um parágrafo, mas não lerá nenhum conteúdo, pois não há. Isso confunde e frustra os que dependem dos leitores de tela.

10. Linha horizontal

Há muitas maneiras de se desenhar um separador horizontal em *HTML* e o jeito certo vai depender de cada caso.

Em geral, a *tag HTML* `<hr>` (*horizontal rule*) é uma *tag* semântica que serve para criar uma linha horizontal, separando elementos horizontalmente. Porém a *tag* `<hr>` não é ideal para separar elementos em listas ``, por exemplo. Neste caso, devemos utilizar CSS.

A *tag* `<hr>` (*horizontal rule*) é um elemento *HTML* ideal para separação entre parágrafos e blocos de conteúdo. Você pode até usar outro método para separar o conteúdo visualmente, mas estaria perdendo a função semântica da *tag* `<hr>`.

Na versão *HTML5*, a *tag* passou a não ser apenas uma linha visual, mas passou a ser uma *tag* semântica, ou seja, a função dessa *tag* agora também serve para que os mecanismos de busca entendam que quando a *tag* está presente, ela denota uma separação de conteúdo, e por isso é importante que seja usada nos lugares certos. Veja o exemplo abaixo:

```
<p>Lorem ipsum dolor sit amet.</p>
<hr>
<p>Consectetur adipiscing elit.</p>
```

Antigamente você também podia adicionar atributos para estilizar a *tag*, mas estes atributos não estão mais presentes no *HTML5*, ficando a critério do CSS a função de estilizar a *tag*.

Note que também não precisamos mais fazer `<hr />` para fechar esse tipo de *tag*. Isso é um padrão *XHTML*, ainda válido no *HTML5*, mas que já caiu em desuso.

10.1. Linha horizontal em `` ou `<div>`

Como regra geral, a *tag* `<hr>` deve ser usada como forma semântica de dividir parágrafos ou conteúdos diferentes. Desta forma o Google entende que se trata de informações separadas.

Portanto não é uma boa ideia incluir `<hr>` dentro de elementos ``, em menus, só para obter o efeito visual, pois a *tag* ``, quando fechada, já denota uma separação entre o item anterior com o próximo item.

No caso de listas, divisões de menu, ou qualquer outro tipo de divisão em que você só quer obter o resultado visual, a regra é usarmos CSS. E existem duas maneiras comuns de separarmos algo visualmente: Usando pseudo elementos `::after`, `::before` ou `border`.

10.1.1 Separando com CSS *border*

O exemplo abaixo serve tanto para menus compostos por `` ou `<div>`, como qualquer outra *tag* do tipo `display: block`.

HTML:

```
<ul class="separator">
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
    <li>Item 4</li>
</ul>
```

CSS:

```
ul.separator {
    list-style: none;
    padding: 0;
    width: 100%;
}

ul.separator li {
    padding: .5em 0;
    border-bottom: 1px solid #CCC;
}
```

Se você não quiser mostrar a borda no último item, pode usar o seguinte CSS conforme o exemplo:

```
ul.separator li:last-child {
    border-bottom: 0;
}
```

10.1.2. Separação com pseudo elementos

Neste exemplo, nós utilizamos “`::after`” para criar pseudo elementos dentro da *tag* ``, e estilizamos esses elementos para que visualmente sejam como linhas horizontais. O efeito é bastante parecido ao exemplo anterior.

HTML:

```
<ul class="separator">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
  <li>Item 4</li>
</ul>
```

CSS:

```
ul.separator {
  list-style: none;
  padding: 0;
  width: 100%;
}
```

```
ul.separator li {
  padding-top: .5em;
}
```

```
ul.separator li::after {
  content: "";
  display: block;
  border-bottom: 1px solid #CCC;
  padding-top: .5rem;
}
```

```
ul.separator li:last-child::after {
  display: none;
}
```

11. Comentários

O comentário em HTML é um trecho de código ou texto que não é executado ou exibido pelo navegador. Ele é utilizado para diferentes finalidades, como para deixar de executar uma parte do código, adicionar alguma informação sobre um comando específico ou, ainda, para deixar alguma informação sobre a página ou a aplicação.

Para adicionar um comentário em um código HTML, é preciso utilizar as tags próprias para esse propósito. Portanto, todo o texto que estiver delimitado pelas tags “<!--”, que indica o início do trecho, e “-->” que significa o fechamento, é considerado comentário, mesmo que o conteúdo se estenda para outras linhas.

Se o comentário for um texto informativo qualquer e não estiver delimitado pelas tags, o conteúdo será exibido pelo navegador sem nenhum tipo de formatação. Portanto, é preciso se certificar de utilizar as tags da forma correta. O mesmo vale para um trecho de código, ou seja, ele será executado pelo navegador.

12. Listas em HTML

Não são raros os casos em que queremos exibir uma listagem em nossas páginas. O HTML tem algumas tags definidas para que possamos fazer isso de maneira correta. Estas listas podem ser ordenadas ou não ordenadas. Vejamos alguns exemplos.

12.1. Listas com marcadores

A lista mais comum é a lista não-ordenada definida pela tag `` (unordered list).

Para alterar os pontos da lista não ordenada, basta inserir o atributo 'style', agora na tag ``, podemos escolher outros formatos para os bullets. Escolhemos através do valor `list-style-type`: `<ul style="list-style-type: TIPO">`. Onde o tipo pode ser, por exemplo: `circle`, `square`, `disc` e `none`. Ou seja, um círculo (sem ser preenchida), um quadrado, um disco (um círculo preenchido) ou nenhum marcador.

```
<ul style = "list-style-type: circle">
  <li>Primeiro item da lista</li>
  <li>
    Segundo item da lista:
    <ul>
      <li>Primeiro item da lista aninhada</li>
      <li>Segundo item da lista aninhada</li>
    </ul>
  </li>
  <li>Terceiro item da lista</li>
</ul>
```

Note que, para cada item da lista não-ordenada, utilizamos uma marcação de item de lista `` (*list item*). No exemplo acima, utilizamos uma estrutura composta na qual o segundo item da lista contém uma nova lista.

12.2. Listas numeradas

A mesma tag de item de lista `` é utilizada quando demarcamos uma lista ordenada. Alguns artigos científicos e outros tipos de documento exigem que listemos informações organizadas numa ordem alfabética (A, B, C, D etc). Isso pode ser feito através do atributo `style`, da tag ``, que como o próprio nome pode sugerir, define o estilo de ordenamento. O valor do atributo `style` é o "list-style: estilo", onde 'estilo' é o estilo de nossa lista. Por exemplo, para que ele fique em ordem alfabética, com letras maiúsculas é: `upper-alpha` Caso desejemos a ordenação

com letras minúsculas: lower-alpha. Há uma série de valores para o list-style, pois há diversos tipos de alfabetos e sistemas numéricos ao redor do mundo.

```
<ol>
  <li>Primeiro item da lista</li>
  <li>Segundo item da lista</li>
  <li>Terceiro item da lista</li>
  <li>Quarto item da lista</li>
  <li>Quinto item da lista</li>
</ol>
```

As listas ordenadas (- ordered list) também podem ter sua estrutura composta por outras listas ordenadas como no exemplo que temos para as listas não-ordenadas. Também é possível ter listas ordenadas aninhadas em um item de uma lista não-ordenada e vice-versa.

12.3. Listas e sublistas

Algumas vezes será necessário criar listas dentro de itens de uma lista, como se aninhássemos uma lista em outra. Isso é possível de ser feito em HTML, pois o navegador vai saber interpretar de maneira correta esse aninhamento.

A parte de HTML, e sua lógica, é bem parecida com o que havíamos feito com as *tags*. Na verdade, vamos aninhar *tags* novamente, as *tags* de lista: e . Vamos enumerar essas seções, com a *tag* . Vamos fazer isso com o seguinte código:

```
<ol>
  <li>Introducao</li>
  <li>Basico</li>
  <li>Textos</li>
  <li>SEO</li>
  <li>Monetizacao</li>
</ol>
```

Agora vamos criar outra lista, que são as páginas que estão dentro da seção "Introdução". Vamos chamar elas de: "O que é HTML", "O que é XHTML" e "O que é CSS". Essa lista seria:

```
<ol>
  <li>O que e HTML</li>
  <li>O que e XHTML</li>
  <li>O que e CSS</li>
</ol>
```

Porém, precisamos aninhar essa lista dentro do item "Introdução", da lista passada. Para isto, basta colocar o código desta segunda lista abaixo do item "Introdução". Use a tabulação (identação) para ficar mais fácil de visualizar. Veja só como ficou nosso código:

```
<ol>
  <li>Introducao</li>
    <ol>
      <li>O que e HTML</li>
      <li>O que e XHTML</li>
      <li>O que e CSS</li>
    </ol>

  <li>Basico</li>
  <li>Textos</li>
  <li>SEO</li>
  <li>Monetizacao</li>
</ol>
```

O resultado é:

1. Introducao
 1. O que e HTML
 2. O que e XHTML
 3. O que e CSS
2. Basico
3. Textos
4. SEO
5. Monetizacao

13. FORMATAÇÃO DE FRASES

O HTML permite dois tipos de formatação: lógico e físico. Aqui veremos as formatações mais utilizadas:

13.1. Estilos Lógicos

- `<cite>`: para títulos de livros, filmes, e citações curtas;
- `<code>`: para indicar trechos de código de programas;
- `<dfn>`: indica definição de uma palavra, em geral apresenta o texto em itálico;
- ``: ênfase, também normalmente apresentado em itálico;
- `<kdb>`: indica uma entrada via teclado;
- `<samp>`: indica uma sequência de caracteres, por exemplo uma mensagem de erro ou um resultado;
- ``: forte ênfase, mostrado normalmente em negrito;
- `<var>`: indica variáveis, ou valores que o usuário deverá escrever; geralmente mostrado em itálico.

13.2. Estilos Físicos

- ``: quando disponível no browser, é mostrado em negrito (em alguns browsers, pode aparecer sublinhado);
- `<i>`: itálico (em alguns casos, caracteres apenas inclinados);
- `<tt>`: tipo *teletype* - fonte de espaçamento fixo;
- `<u>`: sublinhado; deve ser usado com cuidado, pois confunde-se com a apresentação de links;
- `<strike>` ou `<s>`: frase riscada;
- `<big>`: fonte um pouco maior;
- `<small>`: fonte um pouco menor;
- `<sub>`: frase em estilo índice, como em H₂O;
- `<sup>`: frase em estilo expoente, como em km².

14 LIGAÇÕES (USO DE LINK)

O elemento HTML link faz parte do conjunto de tags que formam os metadados do cabeçalho do documento HTML e é utilizado para acessar recursos externos da página. Na prática, a tag <link> diz ao navegador que existem recursos externos vinculados à página que podem ser utilizados no código fonte, como os estilos CSS.

Ao adicionar um arquivo de estilos à página, por exemplo, temos mais organização no código fonte HTML. E não é só isso, a tag também é usada para indicar a pessoa autora da página, versões do conteúdo em outra língua e muito mais.

14.1 Atributos

O elemento <link> contém uma série de atributos que conferem características e funcionalidades a ele. Com o elemento <link>, podemos vincular um arquivo externo que esteja no mesmo domínio da página, ou até mesmo utilizar recursos de outros domínios. Neste cenário, entretanto, é necessário informar ao navegador que foi feita uma requisição que não pertence à mesma origem, ou seja, que pertence a outro domínio.

Isso porque existe uma regra de segurança utilizada pelos navegadores chamada CORS — Cross-Origin Resource Sharing — que bloqueia as requisições de domínios cruzados para oferecer mais segurança à página. Portanto, utilizamos o atributo crossorigin para que essas solicitações externas funcionem. Ele pode conter dois valores específicos:

- anonymous: utilizado quando a requisição é solicitada sem o envio de credenciais de acesso, como certificados digitais, senhas etc;
- use-credentials: utilizado quando a requisição é feita com o uso de credencial de acesso.

Href

Um dos principais atributos do elemento <link> é o href, que é usado para determinar a localização do arquivo acessado. Se ele estiver na mesma pasta que o documento HTML, basta informar o nome correspondente ou o diretório em que ele está. Já se estiver em outro domínio, é preciso informar a URL completa. Veja no exemplo:

```
<link rel="stylesheet" href="estilo.css">  
<link rel="stylesheet" href="url-do-arquivo">
```

Hreflang

O atributo hreflang é utilizado para indicar qual a linguagem do documento linkado. Seu uso é importante quando o site contém versões em várias línguas e utiliza arquivos específicos em cada uma delas.

Media

O atributo media é usado quando queremos indicar em quais dispositivos o arquivo referenciado pode ser utilizado. Algumas das possibilidades de valores são:

- print: o arquivo será utilizado em caso de impressão da página ou na visualização de impressão.
- screen: usado em telas de diferentes dispositivos, como notebooks, desktop, smartphones e tablets.
- speech: utilizado em leitores de páginas.
- all: em todos os tipos de dispositivos.

Link rel

O atributo rel é obrigatório ao utilizar o elemento <link>, pois determina qual a relação entre o documento linkado e o atual. Esse atributo pode conter diferentes valores, por isso, confira os principais a seguir.

Alternate

Esse valor indica que o conteúdo referenciado corresponde a um documento alternativo. Um exemplo para a utilização desse valor é quando utilizamos conteúdos em linguagens diferentes, por exemplo, e queremos indicar qual a versão indicada para cada uma. Nesse cenário, ele deve ser utilizado em conjunto com o atributo hreflang para mostrar a qual linguagem o documento relacionado pertence. Veja um exemplo de código:

```
<link rel = "alternate" href = "http://seusite.com.br/pagina.html"
hreflang = "pt-br" />
<link rel = "alternate" href = "http://seusite.com.br/es/pagina.html"
hreflang = "es" />
<link rel = "alternate" href = "http://seusite.com.br/en/pagina.html"
hreflang = "en" />
```

Author

Quando o atributo rel é indicado como author significa que o endereço relacionado é uma referência à pessoa autora do documento. Conforme o exemplo:

```
<link rel="author" href="http://endereco_do_author">
dns-prefetch
```

Ao fazer uma requisição HTTP, o navegador precisa resolver o DNS solicitado, ou seja, deve traduzir o endereço informado no atributo href para o endereço IP correspondente. Quando essas requisições são realizadas entre domínios cruzados – ou seja, feitas para origens diferentes – isso pode levar um certo tempo e, dessa forma, prejudicar o desempenho da página.

Uma forma de resolver esse problema é com a atribuição do valor dns-prefetch no atributo rel. Na prática, ele resolve o DNS antes do recurso ser solicitado. Assim, quando for necessário utilizá-lo, o acesso será mais rápido, pois a etapa de resolução de DNS já foi realizada.

Help

Quando atribuímos o valor help ao atributo rel, indicamos que a URL informada corresponde a um documento de ajuda referente à página corrente. O conteúdo pode ser a indicação de outra página ou um arquivo de texto, por exemplo.

```
<link rel="help" href="endereço-documento-de-ajuda" >
```

Icon

O atributo rel deve ser preenchido com o valor icon quando queremos indicar qual é o favicon da página. Falaremos sobre os tamanhos dos ícones mais adiante.

O favicon pode ser configurado para ser exibido em diferentes tipos de dispositivos, pois ele pode conter diferentes formatos e tamanhos, conforme a resolução da tela e o modelo do aparelho utilizado pela pessoa usuária.

Para que o navegador decida qual arquivo deve utilizar de acordo com o dispositivo, devemos usar o atributo sizes e informar o tamanho correspondente a cada arquivo. Dessa forma, a pessoa usuária terá em seu dispositivo a imagem mais indicada.

É importante dizer que o atributo sizes só é utilizado para indicar o tamanho dos ícones. Além disso, ele pode conter dois tipos de conteúdo, são eles:

- Height x Width: que indica a altura e largura do ícone
- any: utilizado para imagens escaláveis, como o formato SVG.

Veja um código de exemplo:

```
<link rel="icon" href="favicon.png" sizes="16x16 32x32" type =  
"image/png">  
<link rel="icon" href="favicon-96.png" sizes="96x96" type =  
"image/png">
```

```
<link rel="icon" href="favicon.svg" sizes = "any" type =  
"image/svg+xml">
```

Next e prev

Os valores next e prev referentes ao atributo rel devem ser utilizados quando queremos indicar páginas sequenciais, pois isso facilita o entendimento sobre qual será a próxima página a ser carregada.

O valor next deve ser utilizado sempre na primeira página da sequência, enquanto o prev deve ser indicado na última. Entretanto, se houver páginas intermediárias, tanto o next quanto o prev devem ser informados. Veja o código de exemplo:

```
<!-- na primeira página -->  
<link rel="next" href="pagina2.html">  
<!-- na segunda página -->  
<link rel="next" href="pagina3.html">  
<link rel="prev" href="pagina1.html">  
<!-- na última página -->  
<link rel="prev" href="pagina2.html">
```

Stylesheet

O stylesheet é um dos valores mais comumente utilizados com o elemento <link>, pois ele tem a função de indicar que o documento relacionado corresponde a um arquivo de estilos CSS.

14.2. Caminhos

14.2.1. Caminho relativo

O caminho relativo pode ser usado sempre que queremos fazer referência a um documento que esteja no mesmo servidor do documento atual.

Através do campo “location do browser”, vemos que este documento está localizado em um diretório “/manuals/HTML/” do servidor “www.icmsc.sc.usp.br”. Para escrevermos um link deste documento para o documento “doc2.html” no diretório “/manuals/HTML/exemplos”, tudo que precisamos fazer é escrever:

Veja o exemplo de caminho relativo.

que é apresentado como:

Veja o [exemplo de caminho relativo](#).

Da mesma forma, se quisermos um link deste documento para um outro que esteja em diretório diferente neste mesmo servidor, escrevemos, por exemplo:

`Instituto de Ciências Matemáticas de São Carlos.`

que produz o link: [Instituto de Ciências Matemáticas de São Carlos.](#)

Para usar links com caminhos relativos é preciso, portanto, conhecer a estrutura do diretório do servidor no qual estamos trabalhando.

O esquema do diretório de nosso servidor está disponível no Relatório no. 35 e no relatório do servidor Web (em final de preparação).

14.2.2. Caminho absoluto

Utilizamos caminho absoluto quando desejamos referenciar um documento que esteja em outro servidor, por exemplo:

`Grupo Intermídia.`

que oferece um link para um documento no servidor WWW do Grupo de Pesquisa Intermídia: [Grupo Intermídia.](#)

Com a mesma sintaxe, é possível escrever links para qualquer servidor de informações da Internet.

14.3 Inserindo imagens

Logo no início da web, as páginas continham apenas conteúdos de textos e links. Isso tornava os conteúdos um tanto quanto limitados e monótonos. Não demorou muito para serem criados recursos de inclusão de imagens nas páginas. No HTML, a tag responsável pela inserção de imagens é a tag ``. Porém, o HTML tecnicamente não faz a inclusão da imagem em si, o que ele faz é um link a imagem para a página, de forma que ela seja aberta como se estivesse inserida na mesma, parecido com o processo da tag link `<a>`. Dessa forma, podemos adicionar imagens tanto localmente quanto de forma global, ou seja, através de uma URL externa ao domínio principal. Além disso, também é importante reforçar que a tag `` não possui uma tag de fechamento e traz consigo o padrão de display inline-block do CSS além de ajuste de largura e altura no padrão automático.

Para inserir uma imagem no HTML basta utilizar a tag `` com o atributo `src`. Ou seja: o atributo `src`, ou `source`, vai conter a url da imagem que será inserida.

Diante disso, a sintaxe final será:

``

Para inserir uma imagem local, podemos apenas incluir o nome da imagem com sua extensão, como no exemplo abaixo:

``

Se ela estiver contida dentro de uma pasta local, nós devemos referenciar a pasta também. Por exemplo, se minha página principal está contida na pasta `public_html` e dentro dela temos uma pasta chamada `imagens`, onde está a nossa imagem, utilizaremos a seguinte url:

```

```

Agora digamos que temos uma pasta `public_html` onde dentro dela temos as pastas `imagens` e `pages`, onde a nossa página esta dentro da pasta `pages` e a imagem está dentro da pasta `imagens`. Dessa forma, utilizamos `../` para poder voltar uma pasta e assim entrar na pasta que queremos.

```

```

Por outro lado, podemos ainda incluir a url completa da nossa imagem. Digamos que nosso domínio é `https://meudominio.com` e utilizando o exemplo anterior, dessa forma, utilizaremos o seguinte código:

```

```

Com isso, já estamos prontos para incluir qualquer imagem localmente a nossa página, utilizando o recurso HTML `img`.

Agora que já aprendemos a sintaxe básica do HTML `img` e como inserir imagens locais, vamos estudar como inserir uma imagem global, ou seja, que se encontra fora do nosso servidor, fora do nosso domínio. Para isso, basta adicionarmos a URL absoluta da imagem no atributo `src`.

Digamos que você queira uma imagem que se encontra na URL absoluta `https://www.outrodominio.com/galeria_de_fotos/foto_01.png` . Dessa forma, basta utilizar o seguinte código:

```

```

Porém, encontramos um problema ao utilizar imagens externas: ficamos dependendo que o outro domínio esteja sempre funcionando, pois se em algum momento ele ficar fora do ar, a imagem não será carregada. Para evitarmos problemas com a sua HTML `img`, recomendamos que evite o uso de imagens externas. Além disso, outro grande problema pode ser a utilização de imagens que contêm direitos autorais. Portanto, esteja sempre atento a essas questões e, quando necessário, lembre-se de referenciá-las devidamente.

14.3.1. Título e texto alternativo

Além do atributo `src`, a tag `` possui outros atributos que complementam a sua estrutura: podemos, por exemplo, incluir um atributo de título para a imagem.

Esse título será mostrado apenas quando o usuário passar o mouse sobre a imagem. Portanto, para incluir o título a imagem, basta chamar pelo atributo title = "...". Vejamos então o exemplo abaixo:

```

```

Da mesma forma, também podemos incluir a nossa imagem, um texto alternativo. Esse texto será mostrado no local da imagem caso a URL da imagem esteja errada, a imagem não esteja em um formato suportado ou até que a imagem seja baixada. Para incluir o texto alternativo basta chamar pelo atributo alt = "...". Também é uma prática muito bem vista pelos mecanismos de buscas, portanto muito utilizado para otimização de SEO. Vejamos abaixo um exemplo de como utilizar o texto alternativo na HTML img:

```

```

14.3.2. Largura e altura do HTML img

A tag traz consigo o padrão de altura e largura automática, de forma que ela irá incluir o tamanho original da imagem, sem distorções, ou irá ajustar dentro do container a qual ela for inserido. Porém, podemos manipular esse tamanho diretamente através do CSS ou ainda através do atributos height (altura) e width (largura). Se utilizarmos apenas um deles, o outro irá se ajustar automaticamente, proporcionalmente, sem distorcer a imagem. Utilizando os dois, ele irá conter exatamente a altura e a largura definidas, portanto poderá distorcer a imagem. Além disso, se você utilizar imagens com baixa resolução em tamanhos acima da original, muito provavelmente a mesma também ficará distorcida, por isso é muito importante usar imagens com uma resolução adequada para suas páginas.

Inicialmente, iremos definir apenas uma largura de 100px para uma imagem.

```

```

Desta forma, mesmo com o ajuste apenas a largura, a altura se ajustou automaticamente, sem distorcer a imagem. Vejamos agora com a aplicação tanto de uma largura quanto de uma altura, ambas de 200px.

```

```

Perceba então que, com isso, a imagem se ajustou à largura e à altura de 200px, porém dependendo da imagem ficará distorcida.

14.4. Utilizando imagens para outra página

Para utilizar uma imagem como um link ou âncora para outra página, basta você criar à âncora através da tag <a> e inserir uma imagem dentro. O exemplo abaixo irá mostrar como criar uma imagem clicável para a página do Google.

```
<a href=http://google.com.br>  </a>
```

Frames (Datagrama) são usadas para mostrar mais de um documento HTML em uma janela. Isso significa que você terá menos conteúdo, o que mostra o papel de dizer ao navegador que páginas devem ser exibidas. Desde a introdução de PHP e CSS essa tecnica tem sido cada vez menos utilizada.

Geralmente frames são usadas para exibir o menu em um parate e o conteúdo em outra. Quando alguém clicar em um link do menu, outra página irá abrir a parte do conteúdo.

Exemplicaremos isso com o seguinte código:

```
<html>
<head></head>
<frameset cols="30%,*">
<frame src="menu.html">
<frame src="content.html">
</frameset>
</html>
```

- frameset - a tag que estabelece as características das frames, as frames individuais serão definidas entre elas.
- frameset cols="#%,*" - "Cols" estabelece o peso que cada frame tem. No exemplo anterior estabelecemos que a primeira frame (menu) ocuparia 30% da área usada, e nós usamos o sinal " *" para indicar ao navegador que o resto, no resto da página mostrará o conteúdo.
- frame src="" - o endereço dos arquivos que serão mostrados como menu e conteúdo

Para adicionar um título ou banner, use o código abaixo.

```
<html>
<head></head>
<frameset rows="20%,*">
<frame src="title.html">
<frameset cols="30%,*">
<frame src="menu.html">
<frame src="content.html">
</frameset>
</frameset>
```

</html>

- `frameset rows="#%, *"` - "rows" estabelece o peso de cada frame que será exibida. No exemplo anterior, escolhemos que a primeira frame teria 20% e o resto do espaço seria dividido entre `menu.html` e `content.html`

Você deve ter percebido que entre as frames há algumas linhas cinzas, que na maioria das vezes é indesejada. Apagá-las é possível usando as tags `frameborder` e `framespacing`. Esses atributos serão introduzidos sob a tag `frameset`.

****Nota:** `Frameset` e `frameborder` são os mesmos atributos. Alguns navegadores não reconhecem os dois, apenas um. Dizendo isso, recomendamos usar os dois, por maior segurança.

- `frameborder="#"` - O valor 0 significa que não haverá bordas
- `border="#"` - modifica a grossura da borda (usado por Netscape)
- `framespacing="#"` - modifica a grossura da borda (usado por Internet Explorer)

Aqui um exemplo prático:

```
<html>
<head></head>
<frameset border="0" frameborder="0" framespacing="0" rows="20%,*">
<frame src="title.html">
  <frameset border="0" frameborder="0" framespacing="0" cols="30%,*">
    <frame src="menu.html">
    <frame src="content.html">
  </frameset>
</frameset>
</html>
```

Para manter o menu em sua posição atual, e então quando clicarmos para abrir a página de contatos, por exemplo, ao invés da página de conteúdo, iremos nomear cada frame e especificaremos onde ela abrirá usando a tag '`base target`'.

Aqui o nosso código para a página:

```
<html>
<head>
<base target="content">
</head>
<frameset rows="20%,*">
  <frame name="title" src="title.html">
  <frameset cols="30%,*">
```

```
<frame name="menu" src="menu.html">
<name="content" src="content.html">
</frameset>
</frameset>
</html>
```

Você pode personalizar sua frame ainda mais usando noresize e scrolling.

```
<html>
<head></head>
<frameset border="2" frameborder="1" framespacing="2" rows="20%,*">
<frame src="title.html" noresize scrolling="no">
<frameset border="4" frameborder="1" framespacing="4" cols="30%,*">
<frame src="menu.html" scrolling="auto" noresize>
<frame src="content.html" scrolling="yes" noresize>
</frameset>
</frameset>
</html>
```

- No resize - não deixa a frame mudar suas diensões baseadas na tela do usuário.
- Scrolling = "(yes/no)" - permite ou não usar a função rolagem.

15. FORMULÁRIOS ELETRÔNICOS

Através de um simples formulário de contato, nós veremos os requisitos básicos para construir formulários HTML. Formulários HTML são um dos principais pontos de interação entre um usuário e um web site ou aplicativo. Eles permitem que os usuários enviem dados para o web site. Na maior parte do tempo, os dados são enviados para o servidor da web, mas a página da web também pode interceptar para usá-los por conta própria.

Um formulário HTML é feito de um ou mais widgets. Esses widgets podem ser campos de texto (linha única ou de várias linhas), caixas de seleção, botões, checkboxes ou radio buttons. A maior parte do tempo, estes elementos são emparelhados com uma legenda que descreve o seu objetivo.

15.1 Atributos para o elemento <form>

Todos formulários HTML começam com um elemento <form>.

Este elemento define um formulário. É um elemento de container como um elemento <div> ou <p> , mas ele também suporta alguns atributos específicos para configurar a forma como o formulário se comporta. Todos os seus atributos são opcionais, mas é considerada a melhor prática sempre definir pelo menos o atributo action e o atributo method.

- Action define o local (uma URL) em que os dados recolhidos do formulário devem ser enviados.
- Method define qual o método HTTP para enviar os dados (ele pode ser "GET" ou "POST")

15.1.3. Input

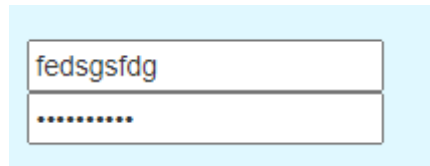
A tag input não necessita de tag final e pode ter vários atributos:

- texto
- senha
- radio
- checkbox
- reset
- enviar

15.1.4. Type

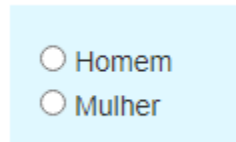
```
<input type="text" /><br />
```

```
<input type="password" />
```



15.1.5. Radio


```
<input type="radio" /> Homem<br />
<input type="radio" /> Mulher
```



15.1.6. Password

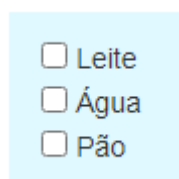
O campo de senha é uma categoria especial de inclusão de tags. Ainda sim, escreve-las é tão simples como escrever um campo de texto normal.

```
<input type="password" size="5" maxlength="5" />
<input type="password" size="10" maxlength="10" />
```



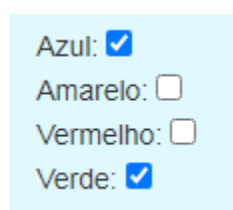
15.1.7. Checkbox

```
<input type="checkbox" /> Leite<br />
<input type="checkbox" /> Água<br />
<input type="checkbox" /> Pão
```



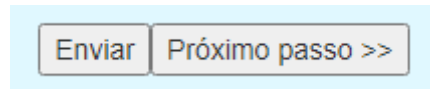
Boxes por default

```
Azul: <input type="checkbox" name="colors" value="blue" checked="yes" /><br />
Amarelo: <input type="checkbox" name="colors" value="yellow" /><br />
Vermelho: <input type="checkbox" name="colors" value="red" /><br />
Verde: <input type="checkbox" name="colors" value="green" checked="yes" />
```



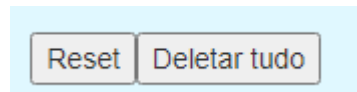
15.1.8. Submit

```
<input type="submit" value="Enviar" />  
<input type="submit" value="Próximo passo >>" />
```



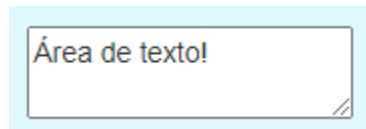
15.1.9. Reset

```
<input type="reset" value="Reset" />  
<input type="reset" value="Deletar tudo" />
```



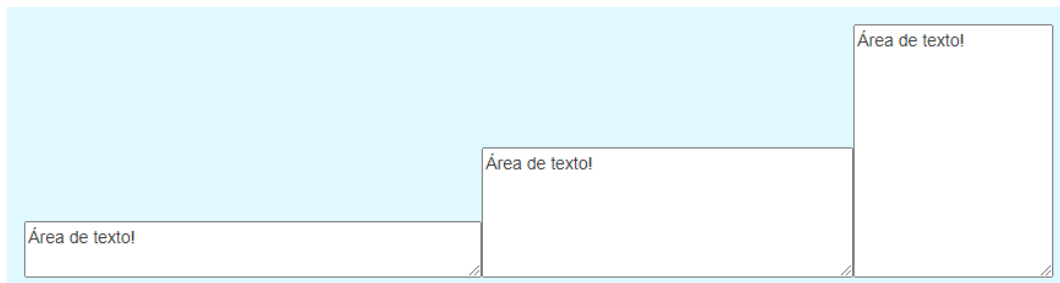
15.1.10. Textarea

```
<textarea>Área de texto!</textarea>
```



Área de texto com o SIZE

```
<textarea cols="50" rows="2">Área de texto!</textarea>  
<textarea cols="40" rows="5">Área de texto!</textarea>  
<textarea cols="20" rows="10">Área de texto!</textarea>
```



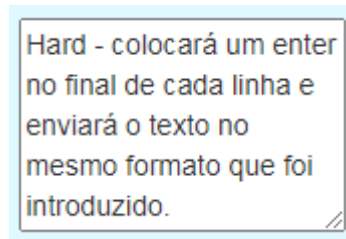
O atributo WRAP: o atributo da tag `<textarea>` estabelecerá o jeito que o texto reagirá quando encontrar o fim da linha.

Wrap terá um dos três valores: hard, soft, off.

- Hard wrap - colocará um enter no final de cada linha e enviará o texto no mesmo formato que foi introduzido.
- Soft wrap - colocará um enter no final de cada linha, mas diferente de Hard, enviará o texto em um formato livre.
- Off wrap - não colocará o texto em nenhum fomato, permitindo o texto em uma única e continua linha.

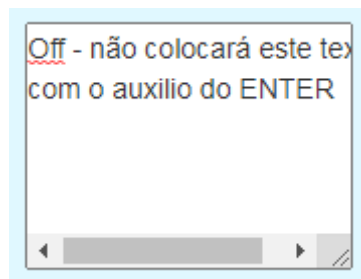
Hard e Soft


```
<textarea cols="20" rows="5" wrap="hard">Hard - colocará um enter no final de cada linha e enviará o texto no mesmo formato que foi introduzido.</textarea>
```



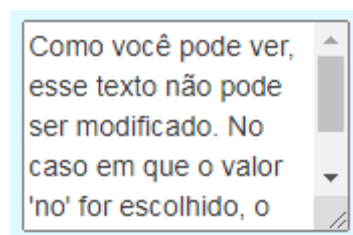
Off

```
<textarea cols="20" rows="5" wrap="off">Off - não colocará o texto com o auxilio do ENTER</textarea>
```



Readonly

```
<textarea cols="20" rows="5" wrap="hard" readonly="yes">Como você pode ver, esse texto não pode ser modificado. No caso em que o valor 'no' for escolhido, o resultado oposto será obtido.</textarea>
```



Disable

```
<textarea cols="20" rows="5" wrap="hard" disabled="yes">O atributo disabled não é muito diferente de readonly. O texto será mostrado em cinza, desabilitando na mesma hora a possibilidade de modificação do texto contido na área de texto.</textarea>
```

