Curso de Construção de Sites - HTML/CSS

Felipe Mariani Lopes
Gustavo Toshi Komura
Bolsistas Instrutores
Bolsa Instrutor DINF/PRAE

23 de Maio de 2011

Grupo de Bolsistas Instrutores

do Departamento de Informática

Conteúdo

I HTML **5**

1 História **6**

2 Comentários **7**

3 Separação de Layout e Dados **9**

4 Visão Geral das Tags HTML **10**

5 A tag <body> **11**
5.1 TAGs estruturais de forma . . . 12
5.1.1 A tag . . . 12
5.1.2 id e class . . . 12
5.1.3 A tag . . . 13
5.2 TAGs estruturais de texto . . . 13
5.2.1 A tag <h(n)> . . . 13
5.2.2 A tag <p> . . . 15

6 A tag <head> **17**
6.1 A tag <title> . . . 17
6.2 A tag <meta> . . . 18
6.3 A tag <link> . . . 20
6.4 A tag <script> . . . 20

7 Modelo DOM (Document Object Model) **24**
7.1 DOM aplicado ao HTML . . . 24
7.1.1 O básico sobre árvores . . . 24
7.1.2 O arquivo HTML como uma árvore . . . 25
7.1.3 Fazendo referência a um elemento HTML . . . 26
7.1.4 Aspectos básicos . . . 26
7.1.5 Deixando comentários . . . 26

8 Exercícios **27**

II CSS **28**

9 História **29**

10 Comentários **30**

11 Cascata de Estilos **31**
11.1 In-Line . . . 31
11.2 In-File . . . 32
11.3 Out-File . . . 32

12 Medindo as coisas **34**
12.1 in - Polegada . . . 34
12.2 px - Pixel . . . 34
12.3 pt - Ponto . . . 34
12.4 cm - Centimetro . . . 34
12.5 mm - Milimetro . . . 35

13 PseudoElementos **36**
13.1 Forma sem seletor classe . . . 36
13.2 Estrutura sem seletor classe . . . 36
13.3 Forma com seletor classe . . . 37
13.4 Estrutura com seletor classe . . . 37

14 PseudoClasses **39**
14.1 Forma sem seletor classe . . . 39
14.2 Estrutura sem seletor classe . . . 39
14.3 Forma com seletor classe . . . 40
14.4 Estrutura com seletor classe . . . 40

15 Formatação de Texto **42**
15.1 Forma . . . 42
15.2 Estrutura . . . 43
15.2.1 Exercicio . . . 45

16 Layout **46**
16.1 Posicionamento . . . 46
16.1.1 Absoluto . . . 46
16.1.2 Relativo . . . 47
16.1.3 Exercicio . . . 47
16.2 Espaçamento - margin/padding . . . 48
16.2.1 Exercicio . . . 49
16.3 Dimensão . . . 49
16.3.1 Exercicio . . . 50
16.4 Bordas . . . 50
16.4.1 Exercicio . . . 51
16.5 Exemplo . . . 51
16.5.1 Exercicio . . . 53

III Indicações **54**

17 Sites Importantes, Referências e Ferramentas **55**

IV Projeto Final **56**

Parte I

HTML

Capítulo 1

História

O **HTML** (**HyperText Markup Language/ Linguagem de Marcação de Hipertexto**) surgiu com o intuito de resolver os problemas de **Tim Berners-Lee**. Ele queria disseminar pesquisas e se comunicar com seus colegas, por esse motivo, ele criou ferramentas para solucionar esses problemas. Estas ferramentas combinadas com a internet da época ganhou um destaque mundial e por incrivel que pareça esta é a linguagem mais apropiadas até o momento para ser usada em sites.

Com o passar dos tempos, (**W3C**) **Wide Web Consortium** vem tentando melhorar esta linguagem, retirando as ambiguidades que exitem nela. A versão que está sendo mantida até o momento foi a lançada no final de 1999 com o nome de **HTML 4.01**. Foi lançada uma outra versão em 2001, mas não foi bem aceita.

Atualmente o **W3C**, está trabalhando no **XHTML**, que é baseado em **XML**, sendo considerado como um sucessor do **HTML**, eles estão procurando acertar a sintaxe, para não haver mais problemas com ambiguidades no código.

Capítulo 2

Comentários

Para iniciarmos nossos estudos, devemos inicialmente dar muita importância na clareza de códigos. No inicio vai parecer uma bobagem deixar comentários, porém veremos a seguir que eles são muito importantes por causa:

- Do esquecimento do que foi feito.
- Da grande quantidade de código.
- Da agilidade no entendimento do código.

Por estes motivos é importante ganhar o costume de documentar tudo do que foi feito, para não ter problemas futuramente.

Em **HTML** fazemos comentários da seguinte forma:

```
<!– comentário –>
```

Vamos ao exemplo:

```
==========================================
    <!– HTML 5 –>
    <!DOCTYPE html>
    <!– Inicio do html –>
    <html>
    <!– A partir daqui vai aparecer na pagina –>
        <body>
    <!– Titulo mais importante –>
            <h1>Titulo 1 </h1>
    <!– Titulo menos importante –>
            <h2>Titulo 2.1 </h2>
    <!– Primeiro texto –>
                texto 1
            <h2>Titulo 2 </h2>
    <!– Segundo texto –>
                texto 2.2
        </body>
    <!– A partir daqui fecha a pagina –>
    </html>
    <!– Final do html –>
==========================================
```

É importante destacar que tudo dentro do comentário não vai aparecer na página, não importa o que for colocado dentro, pode ser uma **TAG** ou um texto, mas não vai fazer nenhuma diferença e não se preocupe se você não entendeu nenhuma definição no cóodigo acima, somente se atente que comentar um código é muito importante.

Capítulo 3

Separação de Layout e Dados

Bom mais uma importância nos nossos estudos em **HTML**, é perceber a diferença entre **LAYOUT** e **DADOS**.

LAYOUT vem a ser todo o desenho, tudo que podemos posicionar, colorir, determinar o tamanho, etc. Isto é, faremos todas estas colocações sobre os **DADOS**.

DADOS vem a ser o conteúdo que vai ser inserido na página, como por exemplo, texto e conteúdo de tabelas.

Exemplificando:

Para ficar mais claro, imaginemos que desejamos colocar uma tabela de horário e aulas, mas ao criarmos está tabela aonde vamos colocá-la, qual o espaço que ela vai ocupar na página, que tipo de letra ela vai possuir e qual cor de letra e fundo ela vai ter ?

LAYOUT: Aonde vamos coloca-la, qual o espaço que ela vai ocupar na página, que tipo de letra e qual cor de letra e fundo ela vai ter.

DADOS: O conteúdo da tabela.

Vemos logo a seguir um exemplo de tabela em **HTML**.

O conteúdo da tabela são os números (Horário e Dias das aulas) que são os **DADOS** da página.

Percebe-se que a tabela está no meio da página então foi utilizado um atributo align=“center“ que posiciona a tabela no centro da página que é o **LAYOUT** da página.

Vamos dar uma olhada para a tabela no browser.

Figura 3.1: table

Desta forma fica claro que aonde está escrito ”Dado”é o que vai ser inserido na tabela **(DADO)**, por outro lado toda a formatação da tabela vai ser o **(LAYOUT)**.

Capítulo 4

Visão Geral das Tags HTML

Bom agora vamos dar uma olhada do que é composto um arquivo **HTML**. Todo arquivo **HTML** possui **TAG**s, desta forma é razoável darmos uma olhada na funcionalidade dela.

A primeira perguntas é "O que é uma **TAG** ?"

R= **TAG** é uma espécie de marcação, é com elas que definiremos o que colocar na página (tabela, imagem, texto, etc).

Tudo bem, devo saber mais alguma coisa sobre **TAG**s ?

R= Sim, uma **TAG** é uma marcação, porém se por exemplo desejarmos colocar uma tabela e ainda não existisse o **CSS**, como que iriamos definir numa celula a cor e o tipo da letra ?.

Bom as duas linguagens não surgiram juntas, o que nos faz crer que o **CSS** veio para fazer auxilio ao **HTML**, mas isso não vem ao caso no momento. Vamos analizar que toda **TAG** possuem **atributos**.

Bom voltamos as perguntas "O que é um **atributo** ?"

R= **Atributo** vem a ser o que se deseja fazer com o que está sendo colocado, isto é, digamos por enquanto que é a formatação que vai ser feita no que está sendo colocado.

Exemplo:

A **TAG** <img> define que dejamos colocar uma imagem na página.

Tudo bem, mas qual é a imagem e qual o tamanho que ela vai ter na página ?

R= Ainda não sabemos.

Por isso que fazemos uso dos **atributos**.

Neste caso os "src", "width"e "height", respondem nossa pergunta, com "src"localização da imagem a ser mostrado na página, "width"largura da imagem e "height"altura da imagem.

Vai ficar assim:

<img src="local da imagem"width="comprimento"height="altura">

Resumidamente um atributo, se refere as caracteristicas de cada objeto.

Por exemplo uma mesa, representa a nossa **TAG**, a partir dai, definimos através dos atrbutos sua cor, comprimento, tamanho e altura.

Por motivos didaticos a partir de agora, vamos definir as **TAG**s e atributos com as seguintes cores:

TAGs principais assim.

TAGs do body assim.

TAGs do head assim.

atributos assim.

Capítulo 5

A tag <body>

<**body**> **TAG** que como o própio nome deduz, <**body**> representa o corpo da página. Nesta **TAG** que será inserido todo o conteúdo da página, tudo que nós como usuários vemos no browser.

Inicialmente vamos começar com um exemplo bem simples. Toda vez teremos que usar esta mesma estrutura para criar uma página.

```
================================================
    <!DOCTYPE html>
    <html>
        <body>
        </body>
    </html>
================================================
```

Este código corresponde a imagem 5.1 body simples

Bom as primeiras perguntas seriam:
O que é isso ?
R= Um código **HTML**.
Como assim, isto é um/uma site/página ?
R= Sim, a única diferença é que ele está em forma de texto e ainda não foi inserido nada na página.
Qual a diferenca entre site e página ?
R= Site é o conjunto de páginas, isto é, uma página consiste em somente um conteúdo de todo site, por exemplo, em um site existe a página principal e as subpáginas como noticias, contato, etc. O site assume uma existência quando se intreliga todas estas páginas. Resumidamente um site é um conjunto de páginas.
Como isso acontece ?
R= O browser interpretao o código e tranfsorma em uma página visual.

A seguir veremos no browser que somente com o código acima, aparentemente não vai existir nenhuma página, porém por enquanto somente aprendemos a fazer a base dela e ainda não foi inserido nada que apareça.

Uma observação muito importante, é que todo arquivo **HTML** deve começar e terminar obrigatóriamente com a **TAG** <**html**>, pois é por ela que o browser sabe que está interpretando um arquivo **HTML**.

Também é importante observar que boa parte das **TAG**s, nos iremos abri-las e fecha-las após fazer uso.

Por exemplo, nos abrimos a **TAG** <html> e no final do arquivo somos obrigados a fecha-la assim

</html>. A / indica que fechamos a **TAG**.

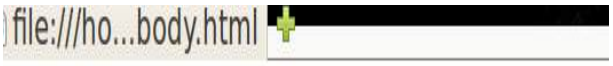


Figura 5.1: body simples

5.1 TAGs estruturais de forma

As tags estruturais são aquelas que nos auxiliam a criar divisões no nosso documento possibilitando a criação de "unidades funcionais", são usadas durante a criação do nosso layout.

5.1.1 A tag

A tag , como o nome sugere, ajuda a definir uma divisão dentro do nosso documento **HTML**, ela é comumente utilizada para agrupar elementos que no seu projeto se relacionam entre si.

Tente imaginar esta **TAG** como se fosse uma caixa ou um bloquinho, aonde tudo que você colocar nela, vai possuir as mesmas caracteristicas.

Geralmente se usa a **TAG** quando se pretende isolar uma parte da página e aplicar um estilo **CSS** ou algum efeito com Javascript que veremos exaustivamente na parte de **CSS**, por estar entre as top top das **TAG**s atualmente.

Vejamos um exemplo do

```
==============================================
    <!DOCTYPE html>
    <html>
        <body>
            <div id="box"> </div>
            <div id="box"> </div>
        </body>
    </html>
==============================================
```

5.1.2 id e class

Aqui vale ressaltar uma pequena diferença entre os atributos id e class. As classs são atributos que indentificam de uma forma geral, já os ids são indentificadores únicos.

Por exemplo fazendo uma comparação entre código de barra e número de série. Um código de barra, por exemplo de um determinado tipo de celular, nos dá o preço, cor e local do produto em estoque, e isso vale para todos os celulares daquele tipo, isto é todos os celulares daquele tipo tem o memso código de

barra, já o número de série, cada um tem o seu, por ser a indentificação dele. Então podemos dizer que o código de barra é o class e o número de série é o id

5.1.3 A tag

A tag é usada quando queremos diferenciar/agrupar um bloco de texto dentro de uma **TAG** <p>. Visualmente, a **TAG** não tem efeito algum.

Geralmente se usa a tag quando se pretende isolar uma parte do texto e aplicar um estilo usando **CSS** ou algum efeito com Javascript.

Vejamos um exemplo do <spam>

```
==============================================
    <!DOCTYPE html>
    <html>
        <body>
            <h1><spam class="tred"> titulo </spam></h1>
            <h2><spam class="tgreen"> titulo2 </spam></h2>
              <p><spam class="pblue"> texto </spam></p>
        </body>
    </html>
==============================================
```

5.2 TAGs estruturais de texto

Agora veremos as tags estruturais que usamos quando queremos inserir texto no nosso documento **HTML**. Usaremos quando precisarmos inserir grandes textos ou criar cabeçalhos.

5.2.1 A tag <h(n)>

<**h**> **TAG** representa titulo em **HTML**.

Esta **TAG** só pode ser usada dentro da **TAG** <body>, pois parece mais interessante mostrar um titulo na página e não fora dela.

```
==============================================
    <!DOCTYPE html>
    <html>
        <body>
            <h1>titulo </h1>
        </body>
    </html>
==============================================
```

Este código corresponde a imagem 5.2 titulo único

Quando estamos produzindo um texto, normalmente ele contém titulo e subtitulos. Bom para fazermos isto em **HTML**, utilizamos a familia da **TAG** <h> com o valor de importância do titulo, sendo

o de maior importância 1 e de menor 6. De <h1> até <h6>.

```
<!DOCTYPE html>
<html>
    <body>
        <h1>titulo </h1>
        <h2>titulo </h2>
        <h3>titulo </h3>
    </body>
</html>
```

Este código corresponde a imagem 5.3 titulo encadeados

Veremos a seguir como esses dois códigos vão ser interpretados pelo browser.

Figura 5.2: titulo único

Figura 5.3: titulo encadeados

5.2.2 A tag <p>

<p> **TAG** que representa textos em **HTML**.

Esta **TAG** só pode ser usada dentro da **TAG** <body> tanbém, pois parece mais interessante mostrar um texto na página e não fora dela.

```
==========================================
    <!DOCTYPE html>
    <html>
        <body>
            <h1>Titulo 1 </h1>
            <h2>Titulo 2.1 </h2>
                <p>texto 1 </p>
            <h2>Titulo 2 </h2>
                <p>texto 2.2 </p>
        </body>
    </html>
==========================================
```

Este código corresponde a imagem 5.4 titulo encadeados e texto

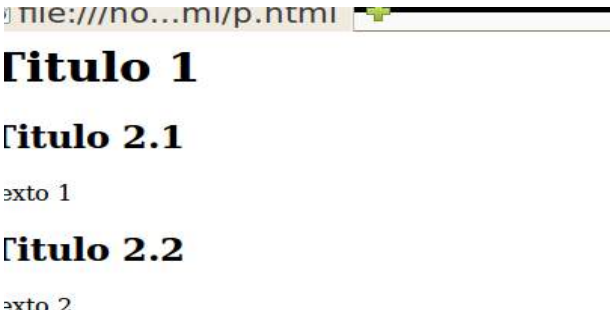


Figura 5.4: titulo encadeados e texto

E o que acontece se não colocarmos a **TAG** <p> e colocar somente o texto ?.

R= Na verdade não acontece nada, porém não vai ser possivel fazer nenhuma formatação neste texto que está sem a **TAG** <p>, e isso pode ou não fazer diferença no resultado final da página, pois vai depender muito do que se deseja fazer, por isso para não haver complicações futuras use a **TAG** e pronto.

```
<!DOCTYPE html>
<html>
    <body>
        <h1>Titulo 1 </h1>
        <h2>Titulo 2.1 </h2>
            texto 1
        <h2>Titulo 2 </h2>
            texto 2.2
    </body>
</html>
```

Este código corresponde a imagem 5.5 titulo encadeados e texto sem p

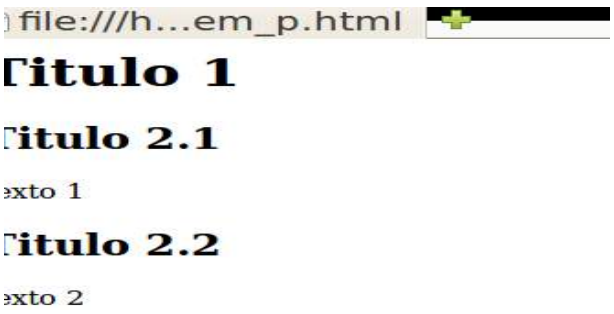


Figura 5.5: titulo encadeados e texto sem p

Capítulo 6

A tag <head>

<head> **TAG** que como o própio nome deduz, <head> representa a cabeça da página. Está parte não é visivel para o usuário, mas é muito importante para indentificação da página.

```
==========================================
    <!DOCTYPE html>
    <html>
        <head>
        </head>
        <body>
            <h1>titulo </h1>
                <p>texto </p>
        </body>
    </html>
==========================================
```

Este código corresponde a imagem 5.6 cabeça

Figura 6.1: cabeça

Olhando para o código e para a imagem percebe-se que ela não fez nehuma diferença na parte visual, mas logo veremos que ela tem uma grande importância para dar um tipo de identidade para a página.

6.1 A tag <title>

<title> **TAG** que representa o titulo da página.

Somente para curiosidade, quando se define o titulo da página, observa-se no topo do browser o titulo que foi dado a ela.

Também é importante destacar que a **TAG** <title> só pode ser usada dentro da **TAG** <head>.

```
==========================================
<!DOCTYPE html>
<html>
    <head>
        <title>titulo da página </title>
    </head>
    <body>
        <h1>titulo </h1>
            <p>texto </p>
    </body>
</html>
==========================================
```

Este código corresponde a imagem 6.7 titulo

Figura 6.2: titulo

Bom a difernça visual não foi tão significativa, mas pelo menos nossa página já tem um nome.

6.2 A tag <meta>

Apesar de não promover nenhuma mudança estrutural ou qualquer tipo de apelo visual, a **TAG** <meta> tem sido uma das mais badaladas do momento. Seu uso tem papel importante no processo de indexação feito pelos sites de busca.

Estar bem colocado nos resultados das buscas passou a ser uma coisa tão importante que surgiu um novo profissional no mercado. Um profissional especializado em otimização dos metadados para uma melhor indexação por parte dos sites de busca, esta especialidade ficou conhecida como **SEO (Search Engine Optimization**, ou **Otimização para Motores de Busca**).

Metadado e informação sobre informação. A **TAG** <meta> proporciona uma maneira de se criar metadados sobre nosso documento **HTML**. O que são, especificamente os metadados ? Com eles podemos especificar uma descrição para a página, **palavras-chave, autor do documento** e mais um monte de informações relevantes para o motor de busca que pretende indexar seu site.

Vou omitir imagens, pois percebe-se que na parte visual não faz difernça, mas se lembre que estamos dando uma indentificação para a página e também contribuindo para buscas na rede.

Com palavra chave: Geralmente vai ser a partir dela que os sistemas de busca na rede, fazem a procura de uma/um página/site.

Para isso vamos precisar de dois **atributos** do <meta>:
O **name**=”tipo”**content**=”contexto”.
Com **name**=”keyword”**content**=”palavra chave”

Veja o código.

```
==========================================
<!DOCTYPE html>
<html>
    <head>
        <title>titulo da página</title>
        <meta name=”keywords”content=”palavra chave”/>
    </head>
    <body>
        <h1>titulo </h1>
            <p>texto </p>
    </body>
</html>
==========================================
```

Com descrição da página.

Com **name**=”description”**content**=”descrição da página”.

Veja o código.

```
==========================================
<!DOCTYPE html>
<html>
    <head>
      <title>titulo da página </title>
      <meta name=”keywords”content=”palavra chave”/>
      <meta name=”description”content=”descrição da página”/>
    </head>
    <body>
      <h1>titulo </h1>
       <p>texto </p>
    </body>
</html>
==========================================
```

Com tipo de contexto.

Vamos precisar dos **atributos** do <meta>:

O **http-equiv**=”tipo”**content**=”contexto”e **charset**=”tipo da codificação”.
Com **http-equiv**=”Content-Type”**content**=”text/tipo do texto”e **charset**=”tipo da codificação”

Veja o código.

```
==========================================
<!DOCTYPE html>
<html>
  <head>
    <title>titulo da página</title>
    <meta name="keywords"content="palavra chave"/>
    <meta name="description"content= "descrição da página"/>
    <meta http-equiv="Content-Type"content="text/tipo do texto; charset=tipo da codificação"/>
  </head>
  <body>
    <h1>titulo </h1>
     <p>texto </p>
  </body>
</html>
==========================================
```

6.3 A tag <link>

<**link**> **TAG** que representa a linkagem de outro arquivo na página. Vai ser a partir desta **TAG** que faremos linkagem com os arquivos **.css** (estilo css), **.js** (arquivos javascript) e **.php** (arquivos php).

```
==========================================
<!DOCTYPE html>
<html>
  <head>
    <title>titulo da página</title>
    <meta name="keywords"content="palavra chave"/>
    <meta name="description"content= "descrição da página"/>
    <meta http-equiv="Content-Type"content= "text/tipo do texto; charset=tipo da codificação"/>
    <link href="caminho do arquivo"rel="stylesheet"type="text/tipo do texto"/>
  </head>
  <body>
    <h1>titulo </h1>
     <p>texto </p>
  </body>
</html>
==========================================
```

6.4 A tag <script>

<**script**> **TAG** que representa a presença de um script que provávelmente será usado na página.

Com esta **TAG** o browser sabe que tudo a partir dela deve ser interpretado como um script, até ser informado que o script terminou, quando isso acontece ele passa novamente a interpretar tudo como **HTML**.

É permitido fazer uso do <script> tanto dentro do <head> como no <body>.

Na <head>.

```
==============================================
<!DOCTYPE html>
<html>
  <head>
    <title>titulo da página</title>
    <meta name="keywords"content="palavra chave"/>
    <meta name="description"content= "descrição da página"/>
    <meta http-equiv="Content-Type"content= "text/tipo do texto; charset=tipo da codificação"/>
    <link href="caminho do arquivo"rel="stylesheet"type="text/tipo do texto"/>
    <script type="text/tipo do texto"script desejado></script>
  </head>
  <body>
    <h1>titulo </h1>
     <p>texto </p>
  </body>
</html>
==============================================
```

No <body>.

```
==============================================
<!DOCTYPE html>
<html>
  <head>
    <title>titulo da página</title>
    <meta name="keywords"content="palavra chave"/>
    <meta name="description"content= "descrição da página"/>
    <meta http-equiv="Content-Type"content= "text/tipo do texto; charset=tipo da codificação"/>
    <link href="caminho do arquivo"rel="stylesheet"type="text/tipo do texto"/>
  </head>
  <body>
    <script type="text/tipo do texto"script desejado></script>
    <h1>titulo </h1>
     <p>texto </p>
  </body>
</html>
==============================================
```

Também se pode fazer uma linkagem, como visto anteriormente, por exemplo com um arquivo **.js** (javascript), isto é, vamos utilizar um script que esteja em outro arquivo.

```
==========================================
<!DOCTYPE html>
<html>
  <head>
    <title>titulo da página</title>
    <meta name="keywords"content="palavra chave"/>
    <meta name="description"content= "descrição da página"/>
    <meta http-equiv="Content-Type"content= "text/tipo do texto; charset=tipo da codificação"/>
    <link href="caminho do arquivo"rel="stylesheet"type="text/tipo do texto"/>
    <script type="text/javascript"src="caminho do arquivo"></script>
  </head>
  <body>
    <h1>titulo </h1>
     <p>texto </p>
  </body>
</html>
==========================================
```

Uma observação que achamos muito relevante neste momento, é destacar que quando fazemos uma linkagem de um arquivo, podemos dizer que estamos inserindo todo o outro arquivo naquele ponto do nosso código. Por isso, tente imaginar que todo o código que está no outro arquivo vai ser inserido no ponto aonde você fez a linkagem.

Para ficar bem claro, segue um exemplo:

Arquivo **HTML**

```
==========================================
<!DOCTYPE html>
<html>
   <head>
    <link href="caminho do arquivo css"rel="stylesheet"type="text/tipo do texto"/>
   </head>
   <body>
      <div id="box"> </div>
      <div id="box"> </div>
   </body>
</html>
==========================================
```

Arquivo **CSS**

```
==========================================
#box { width: 100cm; height: 100cm; }
==========================================
```

É equivalente ao arquivo **HTML**

```
<!DOCTYPE html>
<html>
    <head>
        <style type="text/css">
            #box
            {
            width: 100cm;
            height: 100cm;
            }
        </style>
    </head>
    <body>
        <div id="box"> </div>
        <div id="box"> </div>
    </body>
</html>
```

Capítulo 7

Modelo DOM (Document Object Model)

O "**Documento Object Model**"e uma plataforma e interface livre de linguagem que permitira programas e **scripts** a **acessar** e **fazer update dinamicamente** de conteudo, estrutura e estilo de documentos. O documento pode ser ainda processado e o resultado daquele processamento pode ser incorporado a pagina atual. **http://www.w3.org/DOM/**

7.1 DOM aplicado ao HTML

7.1.1 O básico sobre árvores

Esqueçam as árvores dos bosques ou mesmo as do gramadao do campus, estas são bonitas e "bacanas" pela sombra e pela beleza de suas folhas. As árvores que precisamos conhecer agora são bonitas e interessantes por outro motivo, nós podemos utiliza-las como formato para nosso arquivo **HTML**. Antes de prosseguir, vamos a uma definição de árvore:

"Um grafo aciclico conectado onde cada nó tem zero ou mais filhos e no máximo um nó pai. Além disso, os filhos de cada nó tem uma ordem especifica."**http://en.wikipedia.org/wiki/Tree**

Esta definição não irá nos ajudar muito, pois, para entende-la, teriamos que explicar o que é um grafo. Depois teriamos mais conceitos para introduzir e nosso foco se perderia. Leia somente como curiosidade.

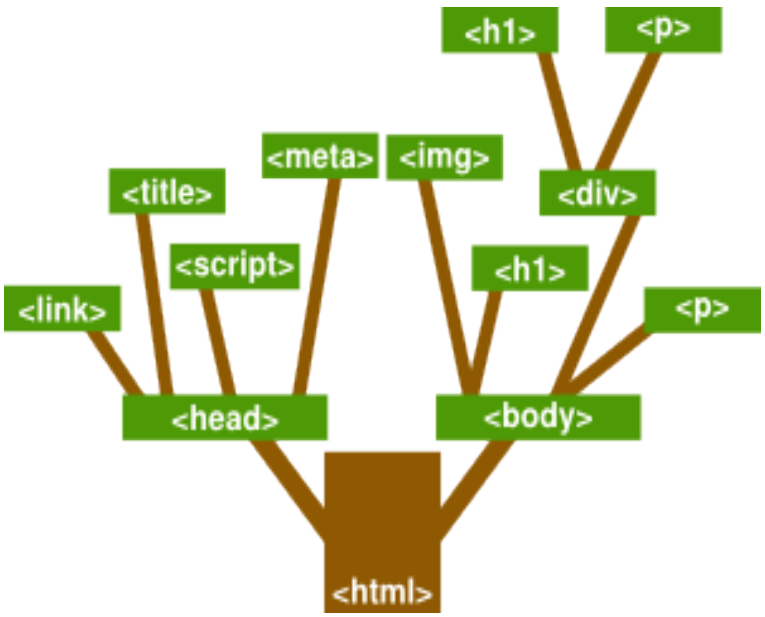
7.1.2 O arquivo HTML como uma árvore

Para entender melhor este conceito, vamos dar uma olhada no seguinte trecho de código **HTML** que possui algum texto e umas imagens:

```
==========================================
<!DOCTYPE html>
<html>
  <head>
    <title>titulo da página</title>
    <meta name="keywords"content="palavra chave"/>
    <meta name="description"content= "descrição da página"/>
    <meta http-equiv="Content-Type"content= "text/tipo do texto; charset=tipo da codificação"/>
    <link href="caminho do arquivo"rel="stylesheet"type="text/tipo do texto"/>
    <script type="text/javascript"src="caminho do arquivo"></script>
  </head>
  <body>
    <h1>titulo </h1>
     <p>texto </p>
    <img src="imagem"/>
    <div id="texto-modificado">
     <h1>titulo </p>
      <p>texto </h1>
    </div>
  </body>
</html>
==========================================
```

Vamos transformar este nosso arquivo em uma árvore:

Perceba a "real"semelhança entre o nosso documento com uma árvore. Para começar, a raiz da nossa estrutura é a **TAG** <html>, ou seja, tudo que esta dentro desta **TAG** é "filho"dela. Esta definição podemos repetir para todas as **TAG**s da nossa estrutura, assim formamos a nossa árvore.



7.1.3 Fazendo referência a um elemento HTML

Agora que a noção de árvore já não assusta mais, ou melhor, estamos quase entendendo a coisa toda, vamos pegar o mesmo exemplo e tentar tracaar o caminho até alguns dos elementos presentes no código:

SEQUÊNCIA DE IMAGENS COM O CAMINHO ATÉ ALGUNS DOS ELEMENTOS REALCADOS

7.1.4 Aspectos básicos

Agora que conseguimos olhar para um documento **HTML** e não mais ver um monte de letras jogadas, mas sim uma grande e frondosa árvore, vamos partir para os primeiros passos no mundo do **CSS**! Novamente, quando os conceitos são simples, ou quando um excesso de aprofundamento for desnecessário, vamos apenas descrever sucintamente o conceito ou elemento. Como sempre, não se esqueça de voltar a esta apostila quando tiver alguma dúvida.

7.1.5 Deixando comentários

Muitas vezes precisamos fazer anotações dentro dos nossos documentos **CSS**. Quando você quiser anotar coisas do tipo "estilo do menu principal"ou "estilo da caixa magica", basta colocar seu comentário desta forma:

```
/* barras e asteriscos */!
```

Vamos ao exemplo:

```
=============================================
/* estilo black box*/
#box
{
width: 200px;
height: 200px;
border: none;
background-color: black;
}
=============================================
```

Não se preocupe se não entender nenhuma das linhas acima, você só precisa entender o conceito de comentário que é o de deixar recados ou fazer anotações dentro do próprio código que não faz nenhuma diferença no resultado final.

Capítulo 8

Exercícios

1 - Crie uma página e de um nome pra ela

2 - Faça uma linkagem imaginária com um arquivo **.css**, **.js** e **.php**. Para isso por exemplo crie os arquivo meu.css, ... eles podem estar vazios, depois faça a linkagem.

3 - Crie dois titulos e insira texto com cores.

4 - Agora crie subtitulos para dividir o seu texto.

5 - Tente colocar os titulos principais no centro.

6 - De palavras-chaves para a sua página.

7 - De uma descrição para sua página.

8 - Insira uma imagem em sua página e determine o seu tamanho.

9 - Qual a importância dos atributos ?

10 - Qual a diferença entre id e class ?

11 - Desenhe a árvore da sua página.

12 - Quais são as **TAG**s que estão na folha da sua árvore e qual á a **TAG** que sempre vaiser a raiz da árvore ?

Parte II

CSS

Capítulo 9

História

Com falado anteriormente, existe uma grande dificuldade de se manter um código totalmente legivel, sabendo que o **HTML** vinha se popularizando, foram surgindo formas de se formatar a página. Isto não significa que não foi importante, porém foi inevitável incluir mais **TAG**s na linguagem. Com isso o código se tornava cada vez mais confuso e menos compativel com os browsers da época, bom mas o que poderia ser feito com relação a isto?

Para a nossa salvação **Håkon Wium Lie**, teve a idéia de tentar separar informação da formatação, isto é, a partir da página criar um arquvio separadamente, que seja somente para a formatação (cor/tamanho do texto por exemplo).

Citado anteriormente também, o **W3C** (**World Wide Web Consortium**) aceitou a idéia de **Håkon**, que chamou **Bert Bos** para ajudar no projeto, **Bert** estava trabalhando em um projeto de um browser (**Argo**) isto tudo em 1995, onde a internet alavancou de vez.

A primeira versão do **CSS** foi lançada em 1996 (**CSS 1**), consequentemente visando o melhoramento da liguagem, foram lançados o **CSS 2** e o **CSS 3**.

Capítulo 10

Comentários

Para iniciarmos nossos estudos com o **CSS**, devemos inicialmente dar muita importância na clareza de códigos. No inicio vai parecer uma bobagem deixar comentários, porém veremos a seguir que eles são muito importantes por causa:

- Do esquecimento do que foi feito.
- Da grande quantidade de código.
- Da agilidade no entendimento do código.

Por estes motivos é importante ganhar o costume de documentar tudo do que foi feito, para não ter problemas futuramente.

Em **CSS** fazemos comentários da seguinte forma:

```
/* comentário */
```

É importante destacar que tudo dentro do comentário não vai poder ser usado como formatação, não importa o que for colocado dentro, pode ser uma **TAG** ou um texto, mas não vai fazer nenhuma diferença.

Capítulo 11

Cascata de Estilos

Existem muitas maneiras de se utilizar o **CSS**, para determinar o estilo de um elemento de uma página, ou mesmo de um site inteiro. Para isto temos três técnicas que veremos logo a seguir:

11.1 In-Line

Dentro da página **HTML** e a formatação **CSS** definindo no local que ele vai ser aplicado, isto é na **TAG** que a formatação vai ser utilizada.

Geralmente vamos recorrer a este modo somente quando necessitarmos de formatações **simples, locais** e **únicas**.

Vejamos um exemplo:

```
==============================================
    <!DOCTYPE html>
    <html>
        <head>
        </head>
        <body>
            <h1>titulo 1</h1>
              <p style="color: sienna ; margin-left: 20px"> texto </p>
            <h1>titulo 2</h1>
               <p>texto</p>
            <h1>titulo 3</h1>
               <p>texto</p>
        </body>
    </html>
==============================================
```

É importante destacar que neste caso o **style**, se comporta como um atributo de <**p**> que faz uso do **CSS**.

Por enquanto não se preocupe com a sintaxe do **CSS**, mas somente para um bom entendimento color indica que se vai mexer na cor e sienna representa a cor que vai ser inserida, margin-left inidica que se vai determinar um espaçamento entre a lateral e a borda esquerda e 20px consiste no espaçamento que vai ser dado.

11.2 In-File

Dentro da página **HTML** e a formatação **CSS** definindo na **TAG** <head>.

Geralmente vamos recorrer a este modo, somente quando necessitarmos de **muitas** formatações **locais** e **únicas** para aquela página, isto é, as outras páginas que vão compor nosso site, não vão necessitar das formatações que estamos aplicando na página atual.

Vejamos um exemplo:

```
==========================================
<!DOCTYPE html>
<html>
    <head>
        <style type="text/css">
          h1 { color: sienna; }
          p { margin-left: 20px; }
          body { background-image: url("images/back40.gif"); }
        </style>
    </head>
    <body>
        <h1>titulo 1</h1>
          <p>texto</p>
        <h1>titulo 2</h1>
          <p>texto</p>
        <h1>titulo 3</h1>
          <p>texto</p>
    </body>
</html>
==========================================
```

Novamente aviso para você não se preocupar com a sintaxe do **CSS**, mas somente para um bom entendimento.

h1 representa o título de maior importância em **HTML**, color indica que se vai mexer na cor e sienna representa a cor que vai ser inserida, então neste caso, toda vez que aparecer **h1** ele vai ter a cor determinada.

E **p** representa texto em **HTML**, margin-left inidica que se vai determinar um espaçamento entre a lateral e a borda esquerda e 20px consiste no espaçamento que vai ser dado, toda vez que aparecer **p** ele vai ter o espaçamento determinada.

E **body** representa o fundo da página em **HTML**, bakground-image inidica que vai ser colocado um imagem como fundo e url indica aonde está localizado a imagem, neste caso em especial, o **body** aparece somente uma vez, então a imagem colocada será o fundo da página.

11.3 Out-File

Está que vamos usar.

Ela é definida a partir de um arquivo **.css**, que vai conter todas as formatações para a página e na página teremos que fazer uma linkagem do arquivo **.css** para fazermos uso do mesmo.

Vejamos um exemplo:

```
<!DOCTYPE html>
<html>
    <head>
        < link rel="stylesheet"type="text/css"href="mystyle.css" />
    </head>
    <body>
        <h1>titulo 1</h1>
          <p>texto</p>
        <h1>titulo 2</h1>
          <p>texto</p>
        <h1>titulo 3</h1>
          <p>texto</p>
    </body>
</html>
```

Neste caso o código acima tem a extensão **.html**, percebe-se em negrito que foi feito uma linkagem com um arquivo com a extensão **.css**, a seguir um exemplo de um arquivo **CSS**.

```
/* arquivo mystyle.css*/
h1
{
color: red;
margin: 5px;
}

p
{
color: blue;
text-align: center;
margin-top: 10px;
}
```

Perceba que não existe nenhuma diferença do arquivo acima com o de baixo

```
/* arquivo mystyle.css*/
h1 { color: red; margin: 5px; }
p { color: blue; text-align: center; margin-top: 10px; }
```

As vezes farei uso do segundo tipo de arquivo por simplesmente haver economia de espaço, mas em casos especiais em que se fazer necessário o uso do primeiro, eu o farei para uma melhor visualização e entendimento do código.

Capítulo 12

Medindo as coisas

Logo mais, necessitaremos fazer uso de medidas para determinar o tamanho dos nossos caixas ou bloquinhos que vão compor nossa página e nos ajudar a organizar-la, para isso determinaremos a seguir algumas das medidas que podemos utilizar.

12.1 in - Polegada

```
================================================
#box
{
width: 100in;
height: 100in;
}
================================================
```

12.2 px - Pixel

```
================================================
#box { width: 100px; height: 100px; }
================================================
```

12.3 pt - Ponto

```
================================================
#box { width: 100pt; height: 100pt; }
================================================
```

12.4 cm - Centimetro

```
================================================
#box { width: 100cm; height: 100cm; }
================================================
```

12.5 mm - Milimetro

```
================================================
#box { width: 100mm; height: 100mm; }
================================================
```

É importante mostar que **#box**, no arquivo **HTML**, vai estar represetado da seguinte forma:

```
================================================
    <!DOCTYPE html>
    <html>
        <head>
            < link rel=”stylesheet”type=”text/css”href=”medindo.css”/>
        <head>
        <body>
            <div id=“box“> </div>
            <div id=“box“> </div>
        </body>
    </html>
================================================
```

Na página aonde encontramos   vai aparecer o bloquinho que definimos no arquivo **CSS**.

Capítulo 13

PseudoElementos

Quando queremos mudar o estilo da primeira letra ou do primeiro caractere de um texto, utilizamos os **pseudo-elementos**. **Pseudo-elementos** sao propriedades dos elementos que não pertencem diretamente aquela árvore de elementos que vimos anteriormente. Vamos aos **pseudo-elementos** que o **CSS** nos fornece:

13.1 Forma sem seletor classe

```
================================================
/* aplica formatação na primeira letra*/
p:first-letter
{
color: #ff0000 ;
font-size: xx-large ;
}
================================================

================================================
/* aplica formatação na primeira linha*/
p:first-line { color: #0000ff ; font-variant: small-caps ; }
================================================
```

13.2 Estrutura sem seletor classe

Neste caso, fizemos uso da seguinte estrutura:

```
================================================
/* estrutura da pseudo-elemento*/
seletor:pseudo-elemento
{
propiedade: valor ;
}
================================================
```

13.3 Forma com seletor classe

Anteriorme foi usado somente o seletor, no entanto também podemos fazer uso do seletor com uma classe, da seguinte forma:

```
================================================
/* aplica formatação na primeira letra*/
p.first:first-letter
{
color: #0000ff ;
font-size: xx-large ;
}
================================================
```

```
================================================
/* aplica formatação na primeira linha*/
p.first:first-line { color: #ff0000 ; font-variant: small-caps ; }
================================================
```

13.4 Estrutura com seletor classe

Neste caso, fizemos uso da seguinte estrutura:

```
================================================
/* estrutura da pseudo-elemento seletor com classe*/
seletor.classe:pseudo-elemento
{
propiedade: valor ;
}
================================================
```

Se aplicarmos desta forma no **HTML**

```
================================================
    <!DOCTYPE html>
    <html>
        <head>
            < link rel="stylesheet"type="text/css"href="pseudo-elemento.css"/>
        </head>
        <body>
            <h1>Titulo 1 </h1>
            <h2>Titulo 2.1 </h2>
                <p>texto 1 </p>
            <h2>Titulo 2.2 </h2>
                <p id="first">texto 2 </p>
        </body>
    </html>
================================================
```

E o seguinte arquivo **CSS**

```
=========================================
p:first-letter { color: #ff0000 ; font-size: xx-large ; }
p:first-line { color: #0000ff ; font-variant: small-caps ; }
p.first:first-letter { color: #0000ff ; font-size: xx-large ; }
p.first:first-line { color: #ff0000 ; font-variant: small-caps ; }
=========================================
```

Os códigos acima correspondem a imagem 13.1 pseudo-elemento

Neste momento, algumas observações se fazem necessárias.
As cores em **CSS**, podem ser definidas de várias formas assim como no **HTML**.
Podem ser definidas em hexadecimal como acima da seguinte forma:

#o valor hexadecimal

O valor hexadecimal pode ter de 4 a 6 digitos com cada digito podendo variar de 0 a f. Para obter os valores das cores, procure no "Google"pela tabela hexadecimal de cores ou você pode encontar esta tabela no "W3Schools".
Vamos obter como resultado no browser do código acima:

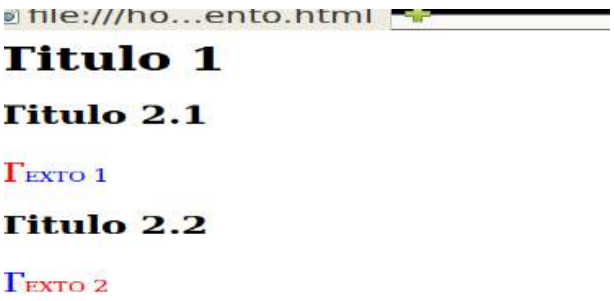


Figura 13.1: pseudo-elemento

Também podemos definir as cores pelo valor rgb que elas possuem. Podemos fazer da seguinte forma:

rgb(valor, valor, valor)

Estes valores podem variar de 0 a 255.
E por último também podemos definir as cores pelo própio nome delas em inglês.
Existem muitas outras formas de se definir cores em **CSS** e **HTML**, porém não vou citar com detalhes neste material, mas posso dizer que temos o rgba, hsl e hsla.

Capítulo 14

PseudoClasses

Pseudoclasses são muito simples, elas servem para adicionar "efeitos especiais"a alguns elementos, vamos tentar entende-los dando uma olhada em alguns exemplos.

14.1 Forma sem seletor classe

Para exemplificação do código abaixo a **TAG** <a> define um **link** em **HTML** e #FF0000 define a cor veremlha em hexadecimal.

```
==============================================
/* link com cor veremlha ao ser visitado*/
a:visited
{
color: #FF0000 ;
}
==============================================

==============================================
/* link com cor veremlha quando ainda não foi visitado*/
a:link { color: #FF0000 ; }
==============================================

==============================================
/* link com cor veremlha quando o mouse passa por cima*/
a:hover { color: #FF0000 ; }
==============================================

==============================================
/* link com cor veremlha quando é ativado*/
a:active { color: #FF0000 ; }
==============================================
```

14.2 Estrutura sem seletor classe

Neste caso, fizemos uso da seguinte estrutura:

```
==========================================
/* estrutura da pseudo-classe*/
seletor:pseudo-classe
{
propiedade: valor ;
}
==========================================
```

14.3 Forma com seletor classe

Anteriorme foi usado somente o seletor, no entanto também podemos fazer uso do seletor com uma classe, da seguinte forma:

```
==========================================
/* link com cor veremlha ao ser visitado*/
a.amarelo:visited
{
color: #FF0000 ;
}
==========================================

==========================================
/* link com cor veremlha quando ainda não foi visitado*/
a.amarelo:link { color: #FFFF00 ; }
==========================================

==========================================
/* link com cor veremlha quando o mouse passa por cima*/
a.amarelo:hover { color: #FF0000 ; }
==========================================

==========================================
/* link com cor veremlha quando é ativado*/
a.amarelo:active { color: #FF0000 ; }
==========================================
```

14.4 Estrutura com seletor classe

Neste caso, fizemos uso da seguinte estrutura:

```
==========================================
/* estrutura da pseudo-classe seletor com classe*/
seletor.classe:pseudo-classe
{
propiedade: valor ;
}
==========================================
```

Se aplicarmos desta forma no **HTML**

```
<!DOCTYPE html>
<html>
    <head>
        < link rel="stylesheet" type="text/css" href="pseudo-classe.css" />
    </head>
    <body>
        <h1>Titulo 1 </h1>
        <h2>Titulo 2.1 </h2>
            <p>texto 1 </p>
        <h2>Titulo 2.2 </h2>
            <p>texto 2 </p>
        <a href="www.inf.ufpr.br/instrutores"> Instrutores</a>
        <a class="amarelo" href="www.inf.ufpr.br/instrutores"> Instrutores</a>
            <p>texto 3 </p>
    </body>
</html>
```

E o seguinte arquivo **CSS**

```
a:link { color: #ff0000 ; }
a:visited { color: #00ff00 ; }
a:hover { color: #ff00ff ; }
a:active { color: #0000ff ; }
a.amarelo:link { color: #fff00 ; }
a.amarelo:visited { color: #00ff00 ; }
a.amarelo:hover { color: #ff00ff ; }
a.amarelo:active { color: #0000ff ; }
```

Os códigos acima correspondem a imagem 13.2 pseudo-classe

Vamos obter como resultado no browser do código acima:

Figura 14.1: pseudo-classe

Capítulo 15

Formatação de Texto

Nem só de formato, coloração e alguns efeitos diferentes vive o mundo do diagramador **CSS**, na maioria das vezes temos que formatar textos, deixando-os agradáveis ao olhar e fáceis de ler, de uma maneira que não canse a visão e não mande nosso usuário pra longe do nosso site.

Para efeitos didáticos, dividimos o conteúdo em dois tópicos, **forma** e **estrutura**. No primeiro, veremos como determinar a familia da letra, tamanho e coloração da fonte. Quando formos tratar de **estrutura**, vamos dar uma olhada em técnicas na hora de estruturar seu texto, para criar um layout "bacana".

15.1 Forma

Para se fazer a modificação de um texto, precisamos ter conhecimento sobre algumas formas de estruturação do **CSS**.

```
==========================================
/* texto no centro e vermelho*/
h1
{
text-align: center;
color: red;
}
==========================================
```

text indica que estou fazendo uma manipulação em texto, align indica que estou alinhando o texto e center indica que o texto será alinhado no centro.

color indica que estou fazendo uma manipulação na cor e red indica que a cor será vermelho.

```
==========================================
/* texto no centro e vermelho*/
h1.centerred
{
text-align: left;
color: blue;
}
==========================================
```

text indica que estou fazendo uma manipulação em texto, align indica que estou alinhando o texto e left indica que o texto será alinhado no lado esquerdo.

color indica que estou fazendo uma manipulação na cor e blue indica que a cor será azul.

15.2 Estrutura

Fizemos uso da seguinte estrutura:

```
==========================================
/* estrutura*/
seletor
{
propiedade: valor;
}
==========================================
```

Também podemos fazer uso com uma classe, podemos ver a estrutura logo a seguir:

```
==========================================
/* estrutura*/
seletor.classe
{
propiedade: valor;
}
==========================================
```

Se aplicarmos desta forma no **HTML**

```
<!DOCTYPE html>
<html>
    <head>
        < link rel="stylesheet" type="text/css" href="forma-texto.css" />
    </head>
    <body>
        <h1 class="cent">Titulo 1 </h1>
        <h2>Titulo 2.1 </h2>
            <p class="cent">texto 1 </p>
        <h2 class="cent">Titulo 2.2 </h2>
            <p class="first">texto 2 </p>
        <a href="www.inf.ufpr.br/instrutores"> Instrutores</a>
        <a class="amarelo" href="www.inf.ufpr.br/instrutores"> Instrutores</a>
        <h1 class="dir">Titulo 2 </h1>
            <p class="dir">texto 3 </p>
    </body>
</html>
```

E o seguinte arquivo **CSS**

```
h1.cent { text-align: center ; }
h1.dir { text-align: right ; }
h1:first-letter { color: #ff0000 ; font-size: xx-large ; }
h1:first-line { color: #0000ff ; font-variant: small-caps ; }
h1.cent { text-align: center ; }
p.cent { text-align: center ; }
p.dir { text-align: right ; }
p:first-letter { color: #0000ff ; font-size: xx-large ; }
p:first-line { color: #ff0000 ; font-variant: small-caps ; }
a:link { color: #ff0000 ; }
a.amarelo:link { color: #ffff00 ; }
```

Os códigos acima correspondem a imagem 13.2 pseudo-classe

Vamos obter como resultado no browser do código acima:

Figura 15.1: form-texto

15.2.1 Exercicio

1 - A imagem realmente corresponde ao código ?

2 - Se não o por que ?

3 - Faça está mesma página somente no arquivo **HTML** e verifique qual a diferença entre fazer uso do **CSS** dentro e fora do mesmo arquivo.

4 - Agora tente fazer está página sem o uso do **CSS**, isso não é obrigatório, mas só imagine como iria ficar o seu arquivo.

Capítulo 16

Layout

Vimos como cuidar do nosso texto, criando uma boa composição e tornando-o agradável ao leitor, sem ser cansativo nem estranho. Neste momento vamos aprender como criar os templates mais comumente utilizados em desenvolvimento web, duas colunas, tres colunas e alguns outros que podemos inventar agora. Para isso vamos ter que aprender como fazer o posicionamento e aprender algumas definições dos bloquinhos que vamos criar.

16.1 Posicionamento

Para darmos uma boa impressão em nossa página, somos obrigados a posicionar imagens, tabelas e textos, por isso faremos uso da propiedade position. Para isso consideramos três tipos de valores que veremos logo a seguir:

16.1.1 Absoluto

Consiste em posicionar algo no lugar **absoluto**, isto é, fazendo uso do valor fixed, ao se fazer uma movimentação na página, através do scrool, o conteúdo com este valor permanece fixo no lugar colocado.

Vejamos alguns exemplos com relação a este tipo de valor:

```
==========================================
/* posicionamento absoluto no lado esquerdo*/
h2.posleft
{
position: fixed;
left: -20px;
}
/* posicionamento absoluto no lado direito*/
h2.posright
{
position: fixed;
left: 20px;
}
==========================================
```

16.1.2 Relativo

Consiste em posicionar algo no lugar **relativamente**, isto é, fazendo uso do valor relative, ao se fazer uma movimentação na página, através do scrool, o conteúdo com este valor se movimenta com a página.

Vejamos alguns exemplos com relação a este tipo de valor:

```
==========================================
/* posicionamento relativo no lado esquerdo*/
h2.posleft
{
position: relative;
left: -20px;
}
/* posicionamento relativo no lado direito*/
h2.posright
{
position: relative;
left: 20px;
}
==========================================
```

Ainda existe mais um caso, vejamos o exemplo:

```
==========================================
/* posicionamento absoluto no lado esquerdo*/
h2.posleft1
{
position: absolute;
left: -20px;
}
/* posicionamento absoulto no lado esquerdo*/
h2.posleft2
{
position: absolute;
left: -20px;
}
==========================================
```

Pode se perceber que posleft1 e posleft2, estão colocando alguma coisa na mesma posição, porém fazendo uso do valor absolute, o que for colocado, não importa o que seja, vai ficar um sobrepondo o outro.

16.1.3 Exercicio

1 - Verifique como os códigos se comprotam no browser.
2 - Qual a diferença entre fixed e do relative ?
3 - Para que fins usariamos o absolute ?

16.2 Espaçamento - margin/padding

Bom agora já sabemos como posicionar os conteúdos na página, porém ainda não sabemos como criar bloquinhos. Vermos logo a seguir como se faz isso:

Um bloquinho é predefinida por quatro intens que determinam o seu contexto, que são:

- **Margem** o espaço entre a borda e as laterais.
- **Borda** separação da margem e o conteúdo.
- **Padding** o espaço entre a borda e o contexto.
- **Context** o espaço determinado para ser utilizado.

Margin define o espaço que vamos deixar entre a borda e as laterais.

```
==============================================
/* espaço da margem*/
#margem
{
margin-top: 100px;
margin-bottom: 100px;
margin-right: 50px;
margin-left: 50px;
}
==============================================
```

Também se pode fazer da seguinte forma

```
==============================================
/* espaço da margem 2*/
#margem2
{
margin: 100px;
}
==============================================
```

Ao colocar desta forma, arbitrariamente o valor margin-top, margin-botton, margin-right e margin-left é 200px.

O padding define o espaço que vamos deixar entre a borda e o contexto.

```
==========================================
/* espaço do padding*/
#padding
{
padding-top: 50px;
padding-bottom: 50px;
padding-right: 100px;
padding-left: 100px;
}
==========================================
```

Também se pode fazer da seguinte forma

```
==========================================
/* espaço do padding 2*/
#padding2
{
padding: 50px;
}
==========================================
```

Ao colocar desta forma, arbitrariamente o valor padding-top, padding-botton, padding-right e padding-left é 50px.

16.2.1 Exercicio

1 - Veja como todos os códigos acima se comportam no browser.
2 - É possivel visualizar algo diferene na sua página ?
3 - Qual seria o grande propósito de se utilizar o margin e padding ?

16.3 Dimensão

Num exemplo, fazendo uso da **TAG** , podemos pré definir o tamanho da caixa que desejamos mexer.

```
==========================================
/* determinando tamanho da caixa a ser trabalhada*/
#caixa
{
width: 200px;
height: 200px;
}
==========================================
```

Também se pode fazer da seguinte forma.

```
==========================================
/* determinando tamanho da caixa a ser trabalhada 2*/
#caixa2
{
width: 200px;
}
==========================================
```

Ao colocar desta forma, arbitrariamente o valor height é 200px

16.3.1 Exercicio

1 - Veja como todos os códigos acima se comportam no browser.
2 - É possivel visualizar algo diferene na sua página ?
3 - É possivel criar um bloco de 1000 de altura e 1000 de comprimento ?
4 - Com relação a questão anterior, se for possivel criar está enorme caixa, para que propósito iriamos utiliza-la ?

16.4 Bordas

Separa a margem do padding.

```
==========================================
/* expessura e cor da borda cinza*/
#bordacinza
{
border: 10px solid gray;
}
==========================================
```

Neste caso a borda possui 10px de expessura, solid uma cor sólida e por fim gray a cor cinza.

```
==========================================
/* expessura e cor da borda verde*/
#bordaverde
{
border: 5px solid green;
}
==========================================
```

Neste caso a borda possui 5px de expessura, solid uma cor sólida e por fim green a cor verde.

Se por algum engano ou esquecimento, não inserirmos a palavra solid automáticamente a borda não vai possuir cor.

```
==========================================
/* expessura e cor da borda sem cor*/
#bordasemcor
{
border: 5px green;
}
==========================================
```

16.4.1 Exercicio

1 - Como exercicio verifique o que acontece no browser com cada um dos códigos acima.
2 - É possivel visualizar algo diferene na sua página ?
3 - Posso fazer uso de todos os tipo de cores como hexadecimal, nome da cor e rgb, com as cores das bordas ?
4 - Posso representar todas as cores que desejo ? Se não, por que isso acontece ?
5 - Baseando-se em todo seu conhecimento e levando em conta todo o conteúdo obtido do material, qual a importância de um bom **LAYOUT** para uma página ?
6 - O que é mais relevante no **LAYOUT** o posicionamento dos objetos ou o tamanho ?
7 - Percebe-se que as dimensões, também tem como companheiros a margin, padding e border, você julga importante fazer uso de todos ou eu posso omitir por exemplo o padding quando desejar criar um bloquinho ?
8 - Crie vários blocos e posicione-os em diferentes pontos da página, faça bastatne o uso do border para você poder visualizar aonde está a caixa que foi criada.
9 - O que acontece quando eu não coloco a margin, padding, dimensão e border ?

16.5 Exemplo

Bom finalizamos por aqui o conteúdo de **CSS**, isto é, agora podemos criar nosso site.

Para isso, aconselhamos que faça uso de tudo que foi visto, e se for possível for além. Para isso deixamos algumas dicas de ferramentas e sites nos capitulos que se seguem.

Uma pequena observação, para se fazer a tabela de links de navegação da página, aconselhamos fazer uso de uma classe do **CSS**, para todas as páginas que compõem o seu site , pois assim não haverá nenhum problema com relação a formatações diferentes em cada página, o que possívelmente deixaria a pessoa que estiver navegando em seu site, um pouco perdida.

Vejamos um exemplo de site:

Vamos ver o código **HTML**, o **CSS** e por fim o resultado no browser.

Este exemplo, vai ser bem parecido com um código que vimos anteriormente. Porém foi incluido o atributo id e a **TAG** <table>.

Um pequena observação que se faz necessária aqui, é que se analizarmos o atributo class da primeira **TAG** <img> do códgio **HTML**, ao invés de ter usado o atributo class poderia ter sido usado o atributo id, pois verifica-se que no arquivo **CSS** cada imagem tem uma caracteristica diferente.

```
<!DOCTYPE html>
<html>
    <head>
        < link rel="stylesheet" type="text/css" href="forma-texto.css" />
    </head>
    <body>
        <img class="cent" src="../imagens/smily.jpeg" width="100px" height="100px" />
        <img id="cent" src="../imagens/smily.jpeg" width="100px" height="100px" />
        <table id="dir" border="3">
          <tr>
            <td>11</td>
            <td>12</td>
          </tr>
          <tr>
            <td>11</td>
            <td>12</td>
          </tr>
        </table>
        <h1 class="cent">Titulo 1 </h1>
        <h2>Titulo 2.1 </h2>
            <p class="cent">texto 1 </p>
        <h2 class="cent">Titulo 2.2 </h2>
            <p class="first">texto 2 </p>
        <a href="www.inf.ufpr.br/instrutores"> Instrutores</a>
        <a class="amarelo" href="www.inf.ufpr.br/instrutores"> Instrutores</a>
        <h1 class="dir">Titulo 2 </h1>
            <p class="dir">texto 3 </p>
    </body>
</html>
```

```
h1.cent { text-align: center ; }
h1.dir { text-align: right ; }
h1:first-letter { color: #ff0000 ; font-size: xx-large ; }
h1:first-line { color: #0000ff ; font-variant: small-caps ; }
h1.cent { text-align: center ; }
p.cent { text-align: center ; }
p.dir { text-align: right ; }
p:first-letter { color: #0000ff ; font-size: xx-large ; }
p:first-line { color: #ff0000 ; font-variant: small-caps ; }
a:link { color: #ff0000 ; }
a.amarelo:link { color: #ffff00 ; }
img.cent { position: absolute ; width: 200px ; height: 200px ; left: 110px ; top: 30px ; }
#cent { position: absolute ; width: 60px ; height: 60px ; left: 710px ; top: 30px ; }
#dir { position: absolute ; width: 100px ; height: 100px ; left: 410px ; top: 30px ; }
```

Figura 16.1: form-texto

16.5.1 Exercicio

1 - Tendo dominio dos bloquinhos ou caixas, como preferir, crie dois blocos, um deve ter dois titulos, com as cores, azul e vermelho com a primeira letra amarela, tente posicinar está caixa no topo direito da página. O outro bloquinho deve possuir um texto com uma imagem de fundo, o texto deve estar em itálico e com a cor em hexadecimal cinza, posicione esta caixa no centro da página, este dois blocos não podem ultrapassar 200 de altura e 300 de comprimento.

2 - Como poço fazer para criar um tabela de navegação entre as páginas do nosso site. Procure fazer uso da **TAG** e estilos no arquivo **CSS**.

Parte III

Indicações

Capítulo 17

Sites Importantes, Referências e Ferramentas

- **W3Schools**: http://www.w3schools.com
  Consideramos uns dos melhores sites que contêm os melhores tutoriais relacionados a criação de sites, com tutoriais de **HTML, CSS, JavaScript, PHP, SQL e outros**, eles possuem exemplos que podem ser modificados e visualizados na mesmaa página. Porém um dos problemas nestes tutoriais para alguns consultores, é que eles estão totalmente em inglês e isso se torna uma dificuldade para alguns.

- **Free CSS**: http://www.free-css.com/
  Para aquele que não tem muita paciência de fazer completamente seu site, Existe a possibilidade de se fazer uso de templates prontos. Neste site você encontrará váriados tipos, porém vale ressaltar que ao fazer uso destes templates prontos, deve-se se citar no rodapé da página que o **CSS** que está sendo usado na página é do **Free CSS** ou de qualquer outro sites de sua escolha.

- **PHP (Hypertext Preprocessor)**: http://www.php.net/
  Para quem quiser se aventurar em outros mundos e tentar aprender mais sobre a construção de sites, pode fazer uso deste site que possui tutoriais de **PHP**. Uma linguaguem que tem uma grande importância na rede.

- **Markup Validation Service**: http://validator.w3.org/
  O validador de código da W3C. tem como objetivo validar um códgio **HTML**. O uso dele se torna muito interessante, para encontar erros, pois ele te informa o local aonde ele considera que o códgio está errado.

Parte IV

Projeto Final

Para ter uma boa conclusão no curso, sugerimos que seja criado um site de sua preferência, mas que aborde todos os assuntos citados neste material. De preferência em utilizar tanto, as formatações de **HTML** como também as formatações em **CSS**.

Procure utilizar linkagens para arquivos **.css** quando as formatações contidas nele forem muito utilizadas no **HTML**, se não for o caso procure utilizar a formatação **CSS** dentro da página. Se a formatação for muito grande utilize a **TAG <style>**.

É importante destacar que num site, normalmente as mesmas dimensões, separação entre conteúdo e navegação e formatações de navegação entre as páginas.

Esperamos que você tenha tido um bom aproveitamente do conteúdo deste material, contudo para qualquer dúvida nos envie um e-mail com o **subject: duvida construcao de sites**.