

FÁBRICA DE DESENVOLVIMENTO

FÁBRICA DE DESENVOLVIMENTO - 2018

Prof. Thiago T. I. Yamamoto

#08 - CSS



thiagoyama



thiagoyama@gmail.com

#08 - CSS

- Cores
- Sintaxe do CSS
- Utilização do CSS
- Cores de fundo
- Fontes
- Textos
- Links
- Listas
- Agrupamentos de elementos
- Box model

- Somente 16 nomes de cores são consideradas validas pelo W3C. Se necessário utilizar outra, não listada abaixo, utilizar código HEXA.

#	Nome da cor	HEXA	Exemplo
1	Aqua	#00FFFF	
2	Black	#000000	
3	Blue	#0000FF	
4	Brown	#A52A2A	
5	Cyan	#00FFFF	
6	Fuchsia	#FF00FF	
7	Gray	#808080	
8	Green	#008000	
9	Lime	#00FF00	
10	Navy	#000080	
11	Olive	#808000	
12	Red	#FF0000	
13	Silver	#C0C0C0	
14	Teal	#008080	
15	White	#FFFFFF	
16	Yellow	#FFFF00	

- Cascading Style Sheets (folha de estilo em cascata)
- Estilo que define como os elementos (fontes, margens, bordas, textos e etc.) HTML serão exibidos.
- Resolvem problemas do HTML
 - Separar estruturação de dados da formatação
- Economizam tempo de trabalho, aumentando a reusabilidade
- Múltiplos estilos podem ser definidos em cascata dentro de um único documento

- Tags HTML foram desenvolvidas originalmente para definir o conteúdo do documento
- O layout do documento era supostamente tratado pelo browser, sem usar qualquer formatação
- O conteúdo do documento HTML foi claramente misturado com o layout de apresentação de documentos.
- Para resolver este problema, o W3C criou o Estilo (STYLE) no HTML 4.0.

- Facilidade de manutenção
- Novas possibilidades de apresentação visual
- Criação de sites com Web Standards
- Diminuição do tempo de download
- Facilidade de expansão de funcionalidades

- Para construir uma CSS basta utilizar corretamente os 5 componentes.
 - Seletor
 - Bloco de declaração
 - Declaração
 - Propriedade
 - Valor

- Seletor
 - É o elemento que diz para o documento em qual TAG será aplicada o CSS.

```
p { color: blue; }
```




Seletor

1

SINTAXE DO CSS

- Bloco de declaração
 - É o escopo de todo o seu CSS.

p { color: blue; }




Bloco de declaração

2

- Declaração

- É a forma que o HTML receberá ao ser renderizado.

p { color: blue; }



Declaração

3

- Propriedade
 - É o aspecto que será alterado na renderização.

```
p { color: blue; }
```

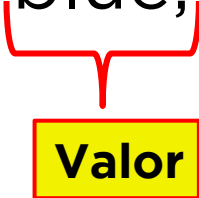


Propriedade

4

- Valor
 - É exatamente o valor que será aplicado na renderização.

```
p { color: blue; }
```



The diagram illustrates the syntax of a CSS rule. The text `p { color: blue; }` is shown. A red bracket is drawn under the word `blue`, and a yellow box with a red border is placed below the bracket, containing the word **Valor**. This indicates that `blue` is the value being assigned to the `color` property.

5

- Se o valor é composto de múltiplas palavras, deve-se utilizar aspas

```
p {font-family: "Times New Roman"}
```

- Para especificar mais de uma propriedade, usa-se ponto e vírgula.

```
p {text-align: center;  
    color: blue;  
    font-family: Arial;  
}
```

- Agrupamento de seletores:

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

- O seletor id
 - Este é usado para criar seu próprio estilo e somente poderá ser especificado em um único atributo id no HTML.

#nome_qualquer { propriedade: valor; }

- O seletor class
 - Este é usado para grupos de elementos que receberão o mesmo estilo e poderá ser especifica em diferentes atributos class no HTML

.nome_qualquer { propriedade: valor; }

I UTILIZAÇÃO DO CSS

- 3 formas de utilização do CSS:
 - Em uma tag HTML (inline)
 - No cabeçalho (<head>) de uma página HTML
 - Em um arquivo CSS externo

- Mistura a estrutura com apresentação
- Não recomendado pelos padrões web
- Exemplo:
 - Alterando a cor de um parágrafo na página:

```
<p style="color: blue">Texto do parágrafo</p>
```

CSS DENTRO DO HTML

- Um único estilo para um documento
- Não recomendado pelos padrões web
- Deve ser inserido no cabeçalho do HTML pelas TAGs <style>
- Exemplo:

```
<html>
<head>
  <style type="text/css">
    p{color: blue;}
  </style>
</head>
...
</html>
```

- Ideal para aplicar em várias páginas
- Recomendado pelos padrões web
- Possibilita mudar o layout mudando apenas o arquivo CSS
- Cada página deve ter um vínculo com a folha de estilo

- Exemplo:

```
<html>
```

```
<head>
```

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

```
</head>
```

```
...
```

```
</html>
```

- CSS externo é apenas um arquivo com extensão .css
 - Todas as paginas que utilizam esta folha de estilo, terão o seu conteúdo alterado.
 - Posso utilizar os três modos? Sim, mas atente-se a ordem de execução:
 1. CSS externo
 2. CSS dentro do HTML (<head>)
 3. Em uma TAG HTML (inline) *
- * Esta última sobre-escreve as outras caso o mesmo elemento tenha sido declarado nos outros modos.

- Dividiremos as propriedades em grupos
 - Cores de fundo
 - Fontes
 - Textos
 - Links
 - Listas
 - Agrupamentos de elementos
 - Box Model
 - Outras propriedades

CORES DE FUNDO

- **background-color:** É a propriedade que define a cor de fundo de um elemento.
- **background-image:** É a propriedade que define uma imagem como fundo.
- **background-repeat:** É a propriedade que controla a repetição de uma imagem de fundo.
 - repeat-x: imagem repete na horizontal
 - repeat-y: imagem repete na vertical
 - repeat: imagem repete em ambas as posições
 - no-repeat: a imagem não se repete

- **background-attachment:** É a propriedade que define se a imagem será fixa na página ou se vai rolar com o conteúdo.
 - scroll: imagem rola com o conteúdo.
 - fixed: imagem fica fixa na página.
- **background-position:** É a propriedade que define o posicionamento da imagem na página.

- **Exemplo**

```
<style>
  body {
    background-color: #FFFF00;
    background-image: url(DEFINA AQUI A SUA IMAGEM);
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-position: top center;
  }
</style>
```

FONTES

- **font-family:** É a propriedade que define a lista de fontes para ser utilizada.
 - serif: caracterizada pelas fontes que possuem “pé”
 - » Ex. Times, Georgia,
 - san-serif: caracterizada pelas fontes que não possuem “pé”
 - » Ex. Arial, Verdana
 - monospace: caracterizada pelas fontes que possuem uma largura fixa.
 - » Ex. Corrier New
 - Ao especificar a fonte, sempre informar qual a sua característica, caso nenhuma da lista for encontrada.

- **font-style:** É a propriedade que define o estilo da fonte.
 - normal: não possui alteração no estilo da fonte. (default)
 - italic: altera o estilo da fonte para itálico
 - oblique: similar ao italic, porem menos suportado..
- **font-variant:** É a propriedade que define a variação em tamanho da fonte
 - normal: não possui alteração no estilo da fonte. (default)
 - small-caps: define um “caps” um pouco abaixo do padrão caixa alta.

- **font-weight:** É a propriedade que define o “peso” da fonte
 - normal: não possui alteração no estilo da fonte. (default)
 - bold: altera o estilo da fonte para negrito
- **font-size:** É a propriedade que define o tamanho da fonte.

- Exemplo

```
<style>  
  body {  
    font-family: arial, verdana, sans-serif;  
    font-style: italic;  
    font-variant: small-caps;  
    font-weight: bold;  
    font-size: 15px;  
  }  
</style>
```

TEXTOS

- **color:** É a propriedade que define a cor de um texto
- **text-align:** É a propriedade que define o alinhamento do texto
 - left: alinha o texto a esquerda
 - right: alinha o texto a direita
 - center: alinha o texto ao centro
 - justify: alinha o texto a pagina
- **text-decoration:** É a propriedade que define “decoração” no texto
 - underline e overline: define uma linha abaixo e/ou acima do texto
 - line-through: define uma linha cortando o texto

- **text-indent:** É a propriedade que define um espaçamento da primeira linha do texto
- **text-transform:** É a propriedade que controla as letras de um texto
 - capitalize: transforma a primeira letra do texto em uppercase
 - uppercase: transforma o texto em caixa alta
 - lowercase: transforma o texto em caixa baixa
- **line-height:** É a propriedade que define a distancia entre as linhas do texto
- **word-spacing:** É a propriedade que define o espaçamento entre as palavras
- **letter-spacing:** É a propriedade que define o espaçamento entre os caracteres do texto

- Exemplo

```
<style>
  body {
    color:#FF0000;
    text-align:justify;
    text-decoration:line-through;
    text-indent: 10px;
    text-transform:capitalize;
    line-height: 5px;
    word-spacing: 10px;
    letter-spacing: 15px;
  }
</style>
```

LINKS

- Para usarmos CSS em links, precisamos conhecer um pouco mais sobre Pseudo-Classes. CSS Pseudo-Classes são usadas para adicionar efeitos especiais em alguns seletores.

seletor:pseudo-classes {propriedade: valor;}

- CSS class também é possível utilizarmos com pseudo-classes

seletor.class:pseudo-classes {propriedade: valor;}

- Os CSS pseudos disponíveis são:
 - :active – Adiciona um estilo no elemento ativo
 - :focus – Adiciona um estilo que tem foco ativo
 - :hover – Adiciona um estilo que tem o ponteiro do mouse sobre o elemento
 - :link – Adiciona um estilo no link que não foi visitado
 - :visited – Adiciona um estilo no link visitado

- **Exemplo**

```
<style>
```

```
p:hover { text-decoration:underline; }
```

```
p.titulo_principal:hover { background-color:#999999; }
```

```
</style>
```

LISTAS

- **list-style-image:** É a propriedade que define uma imagem como marcador da lista
- **list-style-position:** É a propriedade que define a posição do marcador da lista
- **list-style-type:** É a propriedade que define o estilo de marcador da lista

- **Exemplo**

```
<style>
```

```
li {
```

```
    list-style-type:circle;
```

```
    list-style-image:url(DEFINA AQUI A SUA IMAGEM);
```

```
    list-style-position:inside;
```

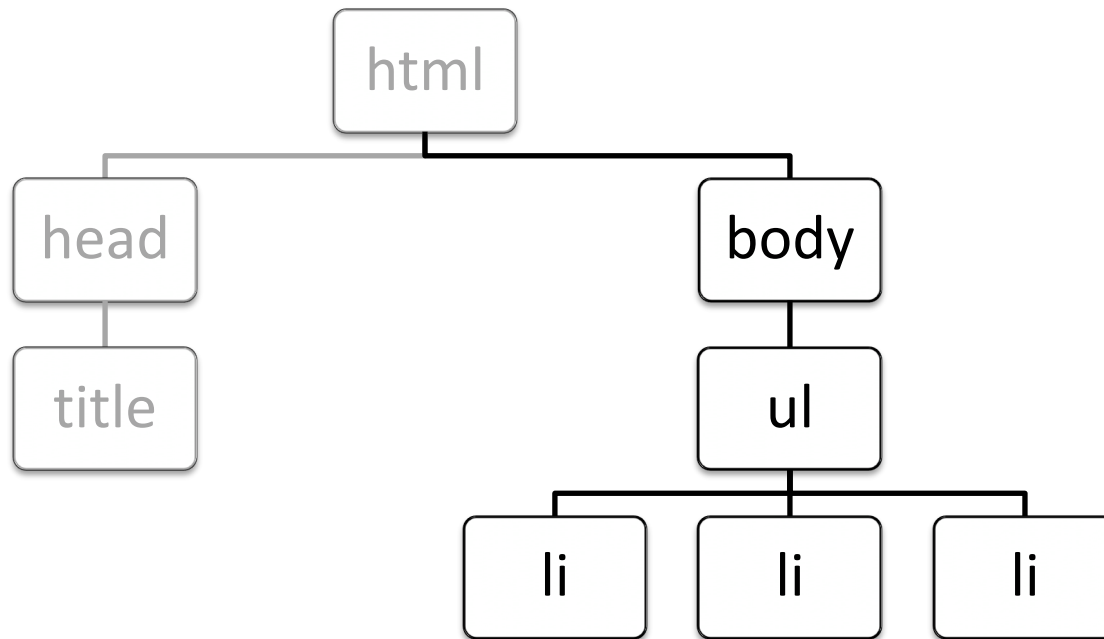
```
}
```

```
</style>
```

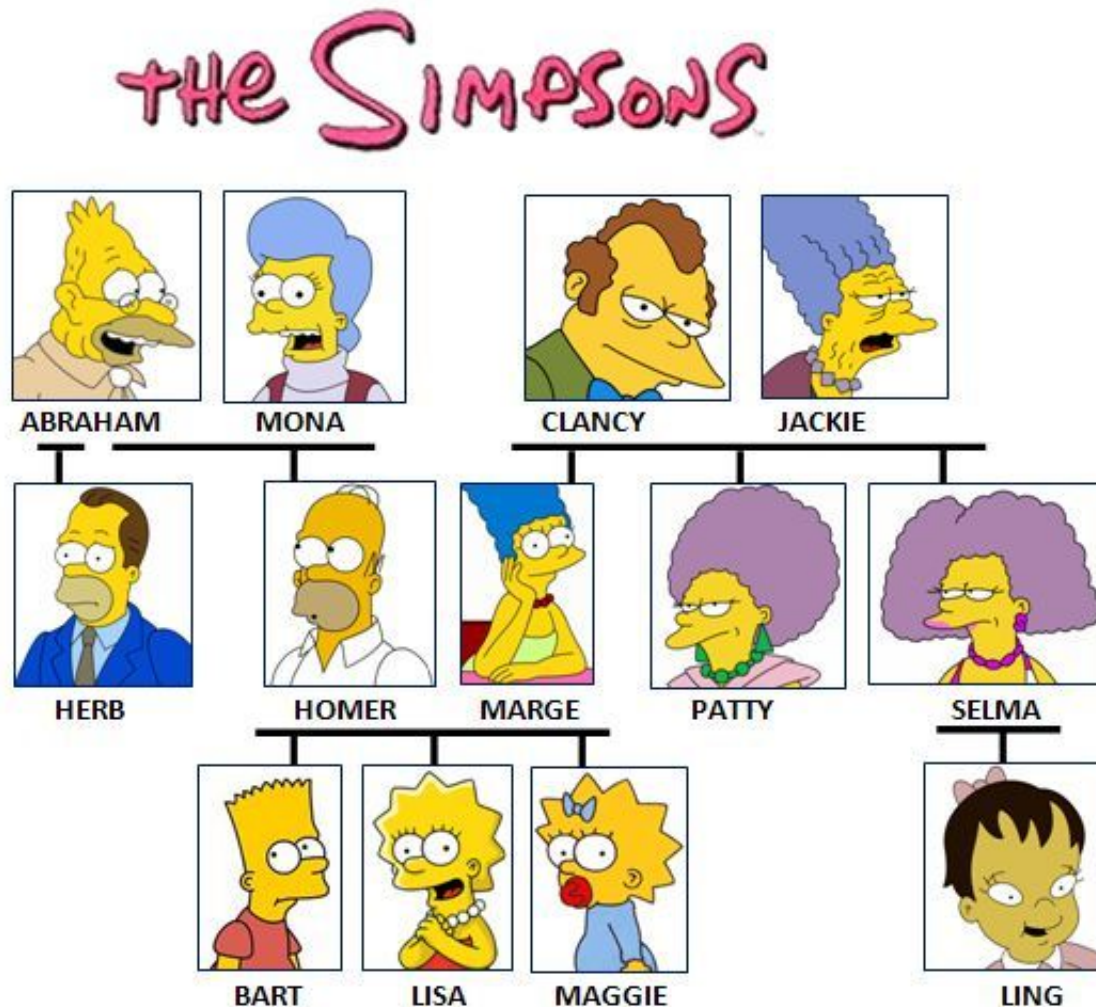
AGRUPAMENTO DE ELEMENTOS

- Não podemos falar em agrupamento sem citar as TAGs `<div>` e `` então vamos relembrar:
 - **`<div></div>`**: tag que define um bloco de elementos na pagina HTML
 - **``**: tag que em conjunto do estilo é possível formatar um texto. Ela por si só não faz alteração alguma no código.
- Mas para entendermos um pouco mais sobre agrupamentos de elementos, vamos falar sobre hierarquia de elementos.

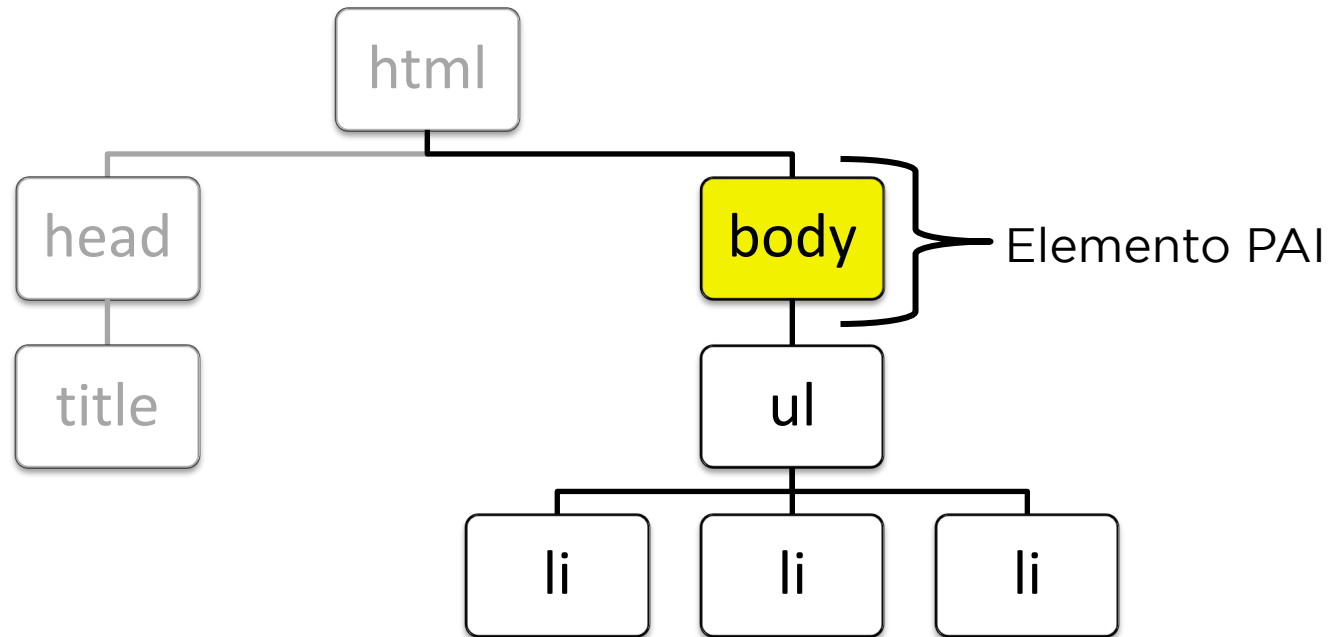
- Todo documento HTML possui estrutura em árvores.
- Estas árvores são compostas por elementos do HTML



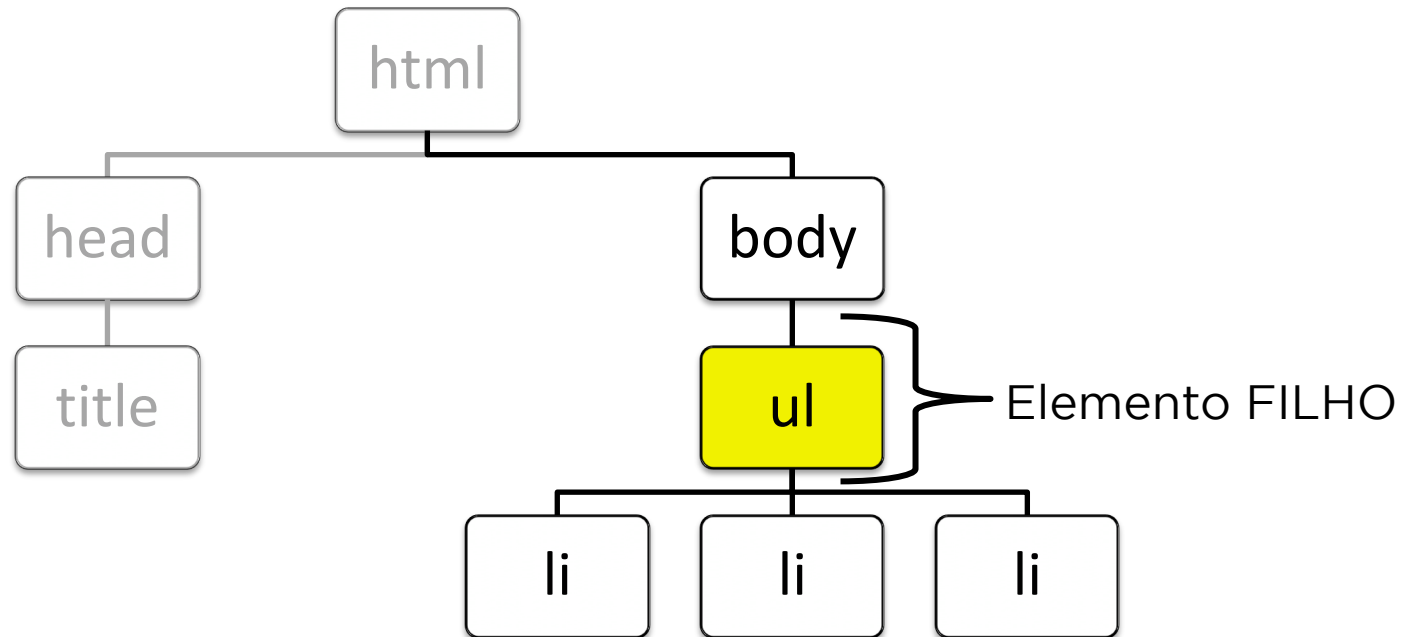
- A hierarquia de documentos é a mesma que existe em famílias.



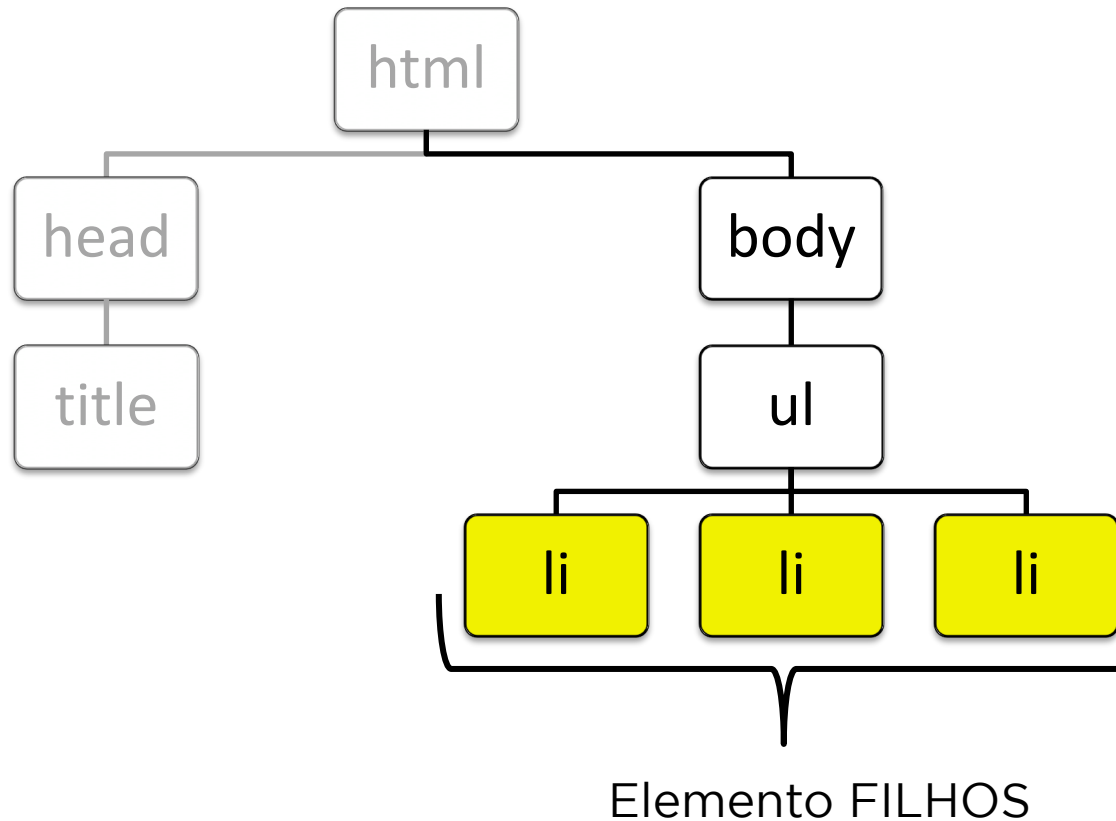
- O elemento PAI esta conectado diretamente abaixo a um elemento da árvore.



- O elemento FILHO está conectado diretamente abaixo a um elemento da árvore.



- O elemento FILHOS que compartilham as mesmas características.



- Conhecendo então um pouco mais sobre hierarquia de elementos podemos concluir que:
 - Agrupar elementos facilita a vida do desenvolvedor
 - Aplicar os estilos específicos nos elementos necessários, mesmo que ele faça parte de outras estruturas
 - E etc.
- As DIVs e SPAN facilitam o agrupamento, por suas características nativas.

- Exemplo 1

```
<style>
```

```
  p { color:#009900 }
```

```
  p em { color:#FF0000;}
```

```
</style>
```

```
<body>
```

```
  <p> parágrafo qualquer com <em>ênfase em vermelho</em> neste  
  texto </p>
```

```
</body>
```

- Exemplo 2

```
<style>
```

```
  p { color:#009900 }
```

```
  div.materia_1 p em { color:#FF0000; }
```

```
  div.materia_2 p span { font-weight:bold; }
```

```
</style>
```

```
<body>
```

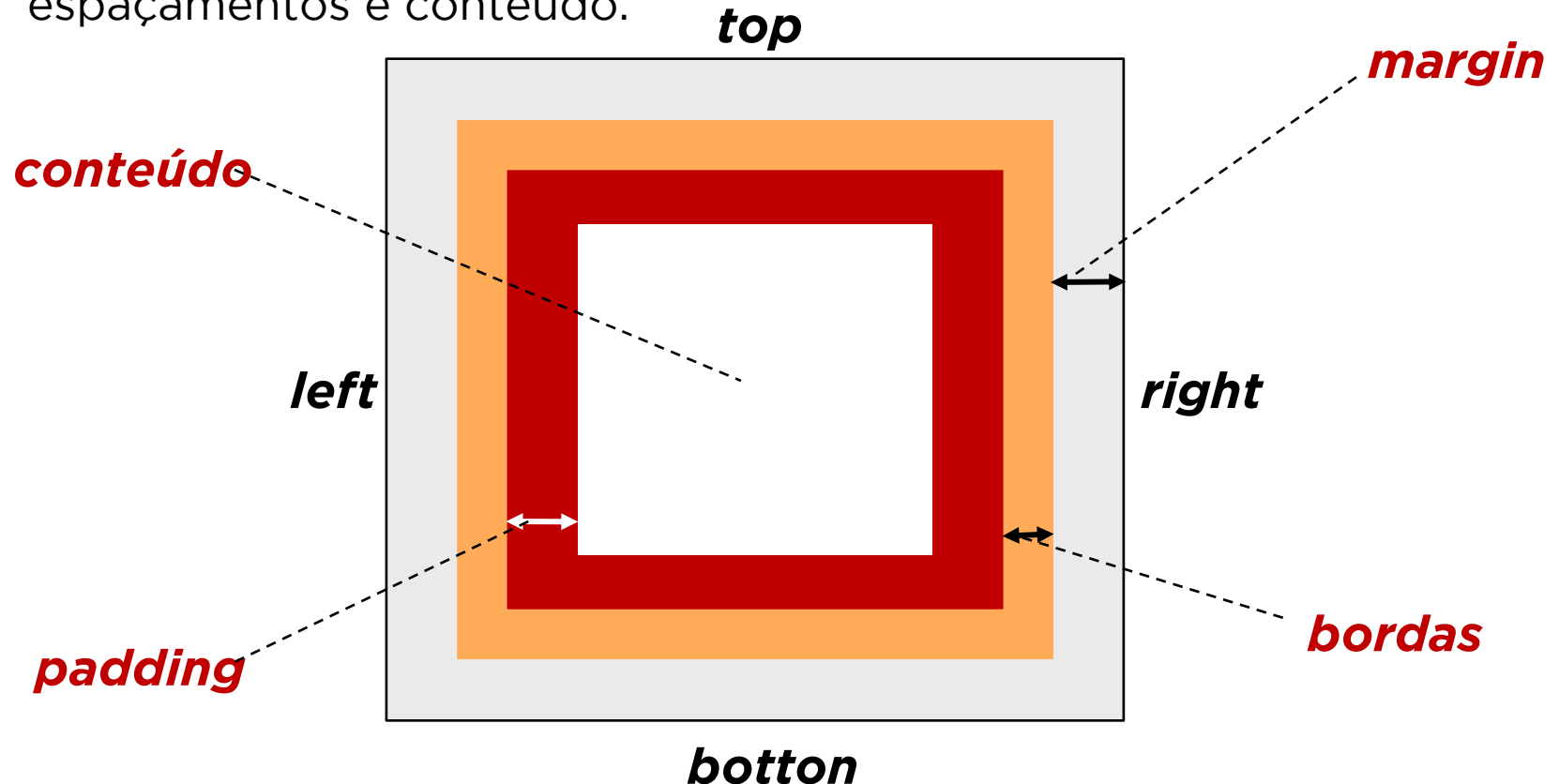
```
  <div class="materia_1"><p> DIV da <em>matéria 1 com ênfase</em>  
  no texto </p></div>
```

```
  <div class="materia_2"><p> DIV da <span>matéria 2 com a tag  
  span</span> no texto </p></div>
```

```
</body>
```

BOX MODEL

- Um CSS box model, compreende todos os elementos do documento HTML e esta relacionado ao layout da sua pagina.
- Essencialmente um box model consiste de margens, bordas, espaçamentos e conteúdo.



- A leitura do tamanho deve ser realizada em sentido horário
- O tamanho final de um box model é definido pela soma de:
 - **Largura total** = width + left padding + right padding + left border + right border + left margin + right margin.
 - **Altura total** = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin.
- O atributo **width** e **height** define somente o tamanho do conteúdo de um elemento.
- Logo, sempre considerar esta regra no tamanho final dos elementos.

- Qual será o tamanho final deste elemento?

```
<style>
```

```
div.ex1{
```

```
    width:220px;
```

```
    height:500px;
```

```
    padding:10px;
```

```
    border:5px solid gray;
```

```
    margin:10px;
```

```
}
```

```
</style>
```

- Largura final: $220 + 10 + 5 + 10 = 245\text{px}$
- Altura final: $500 + 10 + 5 + 10 = 525\text{px}$

- Margem, espaçamento e bordas podem ser declarados da seguinte forma:

1. utilizando seus atributos específicos

- » `propriedade-top: valor;`
- » `propriedade-left: valor;`
- » `propriedade-bottom: valor;`
- » `propriedade-right: valor;`

2. utilizar a forma compilada (shorthand)

- » `propriedade: top right bottom left;`
- » `propriedade: valor;`

- Especifico

```
<style>
  div.ex1{
    margin-top:10px;
    margin-left:10px;
    margin-bottom:10px;
    margin-right:10px;
  }
</style>
```

- Compilado (shorthand)

```
<style>
  div.ex1{ margin:10px 10px 10px 10px; }
  div.ex2{ margin:10px 20px 10px; }
  div.ex3{ margin:10px; }
</style>
```

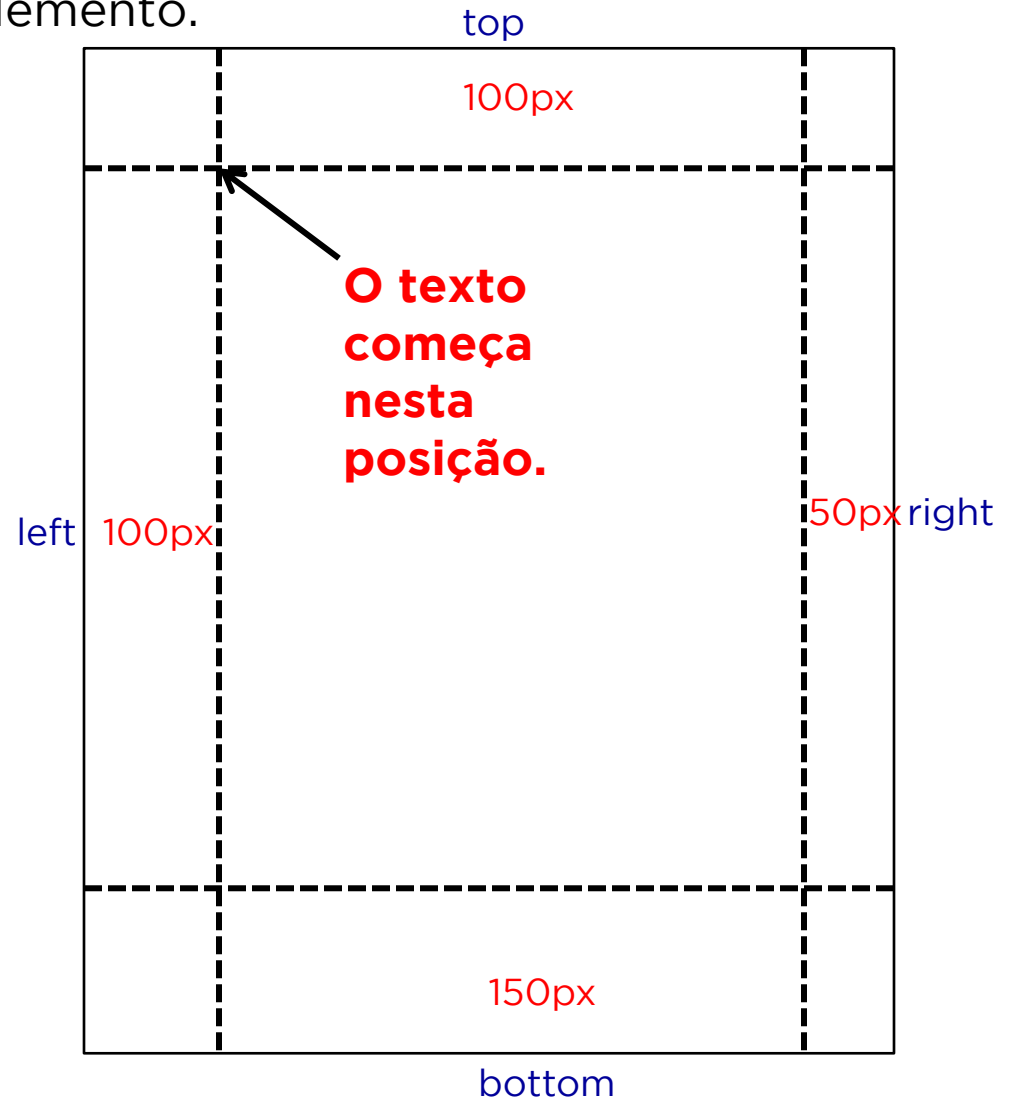
- Tamanhos:
 - px: É a unidade para pixel.
 - %: É a unidade para porcentagem.
 - pt: É a unidade para pontos.
 - em: É a unidade do tamanho de um caracter.
- Cores:
 - Nomes em inglês
 - RGB em valores inteiros ou porcentagem
 - Código hexadecimal

- **Exemplo**

```
<style>  
  div.ex3{  
    margin:10px;  
    color: #FFFF00;  
    font-size: 2em;  
  }  
</style>
```

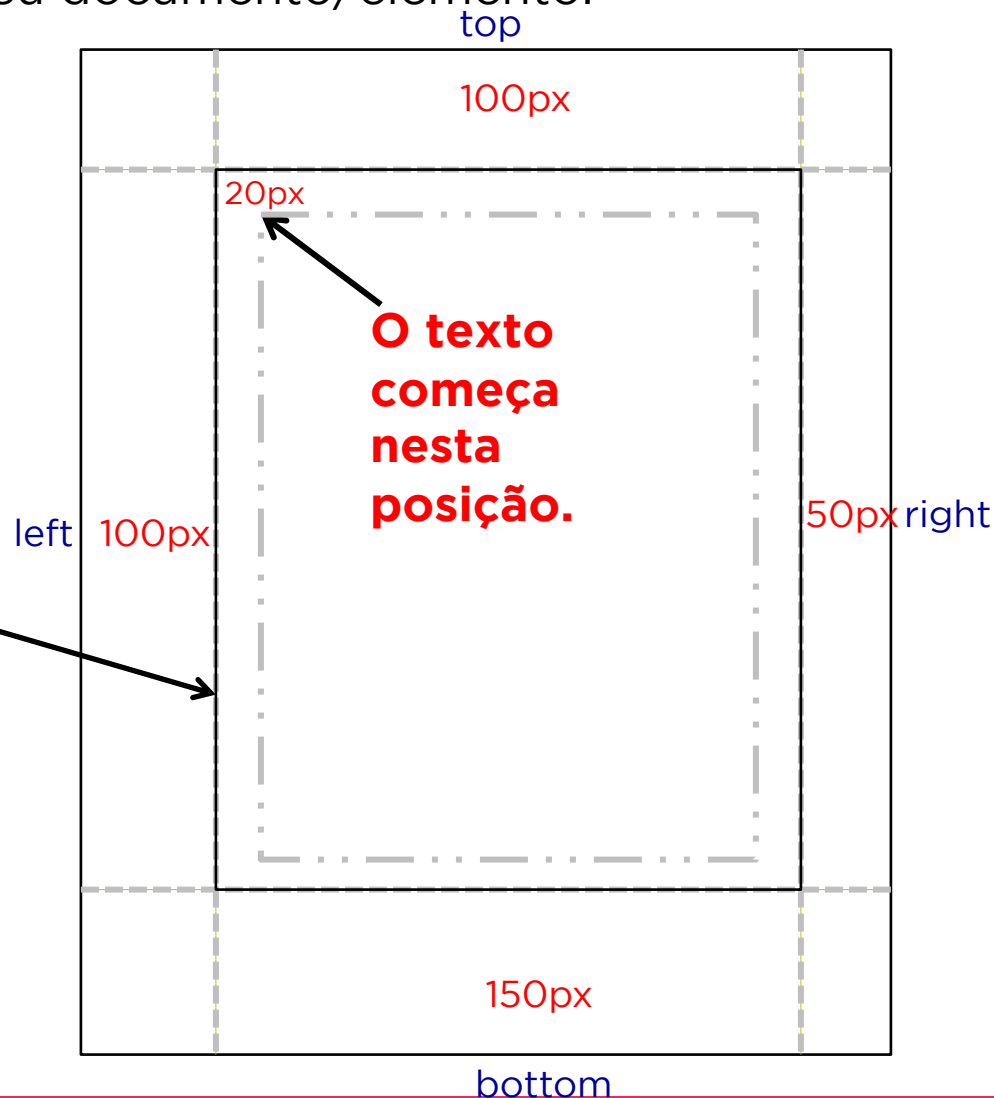
- **Margens** são as distancias definidas entre os lados que determinam o limite do seu documento/elemento.

```
<style>  
  body{  
    margin-top:100px;  
    margin-right:50px;  
    margin-bottom:150px;  
    margin-left:100px;  
  }  
</style>
```



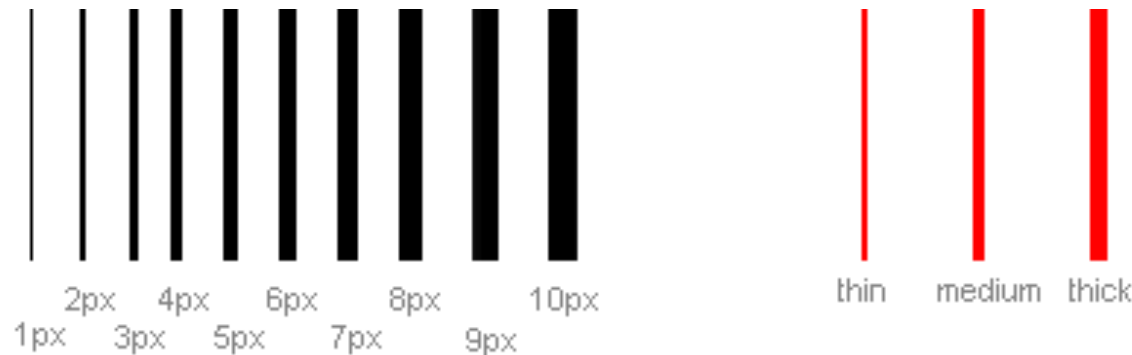
- **Bordas** são as distancias definidas entre as margens e o espaçamento/conteúdo do seu documento/elemento.

```
<style>  
  body{  
    margin-top:100px;  
    margin-right:50px;  
    margin-bottom:150px;  
    margin-left:100px;  
    padding: 20px;  
    border-width:1px;  
  }  
</style>
```



- **border-width:** É a propriedade que define a espessura de borda que será utilizado

– Tamanhos disponíveis



- **border-color:** É a propriedade que define a cor da borda.

- **border-style:** É a propriedade que define o estilo da borda



- **Exemplo**

```
<style>
  h2{
    background-color:#CCFF00;
    padding: 10px;
    border-width: medium;
    border-style: dashed;
  }
</style>
```


OUTRAS PROPRIEDADES

- **Dimensão**

- É possível controlar a largura e altura dos elementos.
 - » **width**: determina a largura de um elemento
 - » **height**: determina a altura de um elemento
 - » **max-width** e **min-width**: determina a largura máxima e mínima de um elemento
 - » **max-height** e **min-height**: determina a altura máxima e mínima de um elemento

- **Exemplo**

```
<style>  
  .d1{  
    width: 200px;  
    height: 300px;  
    max-width: 220px;  
    max-height: 320px;  
  }  
</style>
```

- **Display** (mostra) e **Visibility** (visibilidade)
 - A propriedade `display` determina como será apresentado o elemento
 - A propriedade `visibility` determina se o elemento será visível ou não
- As diferenças entre `display:none` e `visibility:hidden`
 - **visibility:hidden** ocultará um elemento mas continuará ocupando o espaço no layout
 - **display:none** ocultará um elemento e não ocupará o espaço no layout

- Os valores possíveis para **visibility**:
 - **none**: não faz alteração alguma no elemento ou apresenta o elemento
 - **hidden**: oculta o elemento e ocupa o espaço no layout
- Os valores possíveis para **display**:
 - **none**: oculta o elemento e não ocupa o espaço no layout
 - **block**: apresenta o elemento e insere quebra de linha antes e depois do elemento.
 - **inline**: apresenta o elemento em linha sem a quebra de linha

- **Exemplo**

```
<style>
  span{
    display:block;
  }

  li{
    display:inline;
  }
</style>
```

- **Position** (posicionamento): esta propriedade habilita posicionarmos os elementos na tela. Este posicionamento pode ser realizado pelos atributos top, right, bottom e left.
- Para que estes atributos funcionem corretamente, devemos primeiramente definir qual o método do posicionamento , que são:
 - **static**: método default. Segue o fluxo do layout da pagina.
 - **fixed**: método que fixará o elemento mesmo que a pagina possua um scroll.
 - **relative**: método que em conjunto com os atributos top, right, bottom e left, movimentam o elemento de acordo com o valor declarado. Este elemento pode sobrepor outros elementos, mas o seu espaço de origem é preservado.

- **absolute:** método que em conjunto com os atributos top, right, bottom e left, movimentam o elemento de acordo com o valor declarado. Este elemento não preserva o espaço de origem.
- **z-index:** método que possibilita trabalharmos os elementos em camadas (layers) . O principal comportamento do método é que um índice com número maior, sobrepõe o menor.

- **Float** (flutuando): esta propriedade possibilita movimentar um elemento para esquerda ou direita e ainda podemos preencher o espaço vago ao lado com outro elemento.
 - Um elemento somente pode ser movimentado para esquerda e direita. Não é possível movimentar um elemento para cima ou baixo utilizando a propriedade float.
 - Um elemento flutua para esquerda ou direita até quando possuir espaços. Caso não possua mais espaço o elemento flutuara para baixo.
 - » **left:** flutua o elemento para a esquerda
 - » **right:** flutua o elemento para a direita
 - » **clear:** cria um espaço entre os floats

Copyright © 2018 - Prof. Me. Thiago T. I. Yamamoto

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).