

# *Disciplina Desenvolvimento Web I*

Análise e Desenvolvimento de Sistemas

*Material de Apoio*

*Profa. Ana Paula Müller Giancoli*

## Sumário

<b>Sumário . . . . .</b>	<b>1</b>	
<b>1</b>	<b>INTRODUÇÃO . . . . .</b>	<b>5</b>
<b>1.1</b>	<b>Apresentação HTML . . . . .</b>	<b>5</b>
1.1.1	W3C - <i>World Wide Web Consortium</i> . . . . .	5
1.1.2	WHAT Working Group . . . . .	5
1.1.3	W3C - 3 pilares . . . . .	5
1.1.4	Registro de Domínio . . . . .	6
1.1.5	Significado HTML . . . . .	6
1.1.6	História do HTML . . . . .	6
1.1.7	Como surgiu o HTML5 . . . . .	7
<b>1.2</b>	<b>Apresentação CSS . . . . .</b>	<b>8</b>
1.2.1	CSS - Cascading Style Sheets . . . . .	8
1.2.2	Apresentação CSS3 . . . . .	8
<b>1.3</b>	<b>Apresentação Javascript . . . . .</b>	<b>8</b>
1.3.1	Linguagem de Script . . . . .	8
<b>2</b>	<b>LINKS DIVERSOS . . . . .</b>	<b>10</b>
2.1	Links sobre acessibilidade na web . . . . .	10
2.2	Links sobre Evolução da Web . . . . .	10
2.3	Links sobre Internacionalização . . . . .	10
2.4	Links sobre formatação com Cores . . . . .	10
2.5	Links de referências MDN . . . . .	11
2.6	Links para formatação de elementos com CSS . . . . .	11
2.7	Links para formatação de Fontes e ícones . . . . .	11
2.8	Links de Frameworks CSS . . . . .	11
2.9	Links para Pré-Processadores CSS . . . . .	12
2.10	Links para Manuais Resumidos de Tags HTML, GitHub, propriedades CSS, Modal.js . . .	12
2.11	Links para reservar espaços de imagens na página . . . . .	12
2.12	Links diversos . . . . .	12
<b>3</b>	<b>FERRAMENTAS PARA WEB . . . . .</b>	<b>13</b>
3.1	Validadores . . . . .	13
3.2	Editores . . . . .	13
<b>4</b>	<b>ESTRUTURA SEMÂNTICA DO HTML5 . . . . .</b>	<b>14</b>
4.1	Semântica x Estilos . . . . .	14
4.2	HTML Semântico . . . . .	14

<b>4.3</b>	<b>Representação da Estrutura HTML5</b>	<b>15</b>
<b>4.4</b>	<b>Significado de cada item da Estrutura HTML5</b>	<b>15</b>
<b>4.5</b>	<b>Exemplo da nova estrutura</b>	<b>16</b>
<b>4.6</b>	<b>Acessibilidade e tecnologias assistivas</b>	<b>16</b>
<b>4.7</b>	<b>Tags do HTML5</b>	<b>17</b>
<b>4.8</b>	<b>Search Engine Optimization (SEO)</b>	<b>17</b>
<b>5</b>	<b>SELETORES, LINKS DE ESTILO, HIPERLINKS, CORES</b>	<b>18</b>
<b>5.1</b>	<b>Seletores Básicos</b>	<b>18</b>
5.1.1	Seletores HTML	18
5.1.2	Seletores de Classes	18
5.1.3	Seletores Identificadores	19
5.1.4	Seletores Universal	19
<b>5.2</b>	<b>Carregando a formatação no HTML</b>	<b>19</b>
5.2.1	Exemplo1: arquivo de estilo dentro do HTML	19
5.2.2	Exemplo2: arquivo de estilo externo ao HTML	20
<b>5.3</b>	<b>Utilizando hiperlink no HTML</b>	<b>20</b>
<b>5.4</b>	<b>Utilizando cores para formatação com o arquivo CSS</b>	<b>21</b>
5.4.1	Exemplo3: arquivo de estilo com exemplos de formatações de cores	22
<b>6</b>	<b>PROPRIEDADES CSS DE FONTES E TEXTOS</b>	<b>23</b>
<b>6.1</b>	<b>Fontes</b>	<b>23</b>
6.1.1	Fontes genéricas	23
6.1.1.1	Exemplo de fontes genéricas	23
6.1.1.2	Webfonts	23
6.1.1.3	Google Fontes	24
6.1.2	Propriedades de Fontes	24
6.1.3	Exemplos do uso de propriedades de formatação de fontes	25
<b>6.2</b>	<b>Unidades de Medida</b>	<b>26</b>
6.2.1	Relativas	26
6.2.2	Absolutas	26
<b>6.3</b>	<b>Textos</b>	<b>26</b>
6.3.1	Formatação de cores dos textos	26
<b>7</b>	<b>PROPRIEDADES CSS DE LAYOUT</b>	<b>29</b>
<b>7.1</b>	<b>Box Model padrão do CSS</b>	<b>29</b>
7.1.1	Box-sizing	29
<b>7.2</b>	<b>Margin</b>	<b>30</b>
<b>7.3</b>	<b>Padding</b>	<b>31</b>
<b>7.4</b>	<b>Position</b>	<b>32</b>
<b>7.5</b>	<b>Float</b>	<b>32</b>
<b>7.6</b>	<b>Visibility</b>	<b>35</b>
<b>7.7</b>	<b>Outline</b>	<b>36</b>
<b>7.8</b>	<b>Display</b>	<b>37</b>
<b>7.9</b>	<b>Flexbox Layout</b>	<b>37</b>
<b>7.10</b>	<b>Propriedades de formatação de background</b>	<b>51</b>
<b>8</b>	<b>TAGS DE LISTAS, TABELAS</b>	<b>54</b>
<b>8.1</b>	<b>Tags de Listas ordenadas e desordenadas</b>	<b>54</b>
<b>8.2</b>	<b>Tags de Listas de Definição</b>	<b>54</b>

8.2.1	Propriedades de formatação de listas ordenadas e desordenadas . . . . .	54
<b>8.3</b>	<b>Tags de Tabelas . . . . .</b>	<b>54</b>
8.3.1	Propriedades de formatação de tabelas . . . . .	55
<b>9</b>	<b>MANIPULANDO FORMULÁRIOS . . . . .</b>	<b>57</b>
<b>9.1</b>	<b>Tags de Formulários . . . . .</b>	<b>57</b>
9.1.1	Form . . . . .	57
9.1.2	Fieldset e Legend . . . . .	59
9.1.3	Label . . . . .	60
9.1.4	Input . . . . .	61
9.1.4.1	Tipos de input . . . . .	63
9.1.5	Textarea . . . . .	65
9.1.6	Select, option, optgroup . . . . .	65
9.1.7	Datalist e o atributo list . . . . .	65
9.1.8	Button . . . . .	65
9.1.9	Output . . . . .	65
9.1.10	Meter . . . . .	65
9.1.11	Progress . . . . .	66
9.1.12	Enviando dados de um Formulário . . . . .	66
9.1.12.1	Atributos do FORMULÁRIO: . . . . .	67
<b>10</b>	<b>APLICANDO BOOTSTRAP E MEDIA QUERIES . . . . .</b>	<b>68</b>
<b>10.1</b>	<b>De acordo com o site do Bootstrap: . . . . .</b>	<b>68</b>
<b>11</b>	<b>ANIMAÇÕES COM CSS . . . . .</b>	<b>70</b>
<b>11.1</b>	<b>Configurando Animação . . . . .</b>	<b>70</b>
11.1.1	Animation . . . . .	70
11.1.2	Transition . . . . .	71
<b>11.2</b>	<b>Definindo sequencia de animação com Keyframes . . . . .</b>	<b>72</b>
<b>11.3</b>	<b>Efeitos de transições . . . . .</b>	<b>72</b>
<b>11.4</b>	<b>Exemplificando . . . . .</b>	<b>73</b>
11.4.1	Exemplo 1 . . . . .	73
11.4.2	Exemplo 2 . . . . .	75
11.4.3	Exemplo 3 . . . . .	77
11.4.4	Outros Exemplos . . . . .	79
<b>12</b>	<b>LEITURA DE CONTEÚDOS POR ROBÔS . . . . .</b>	<b>80</b>
<b>13</b>	<b>CANVAS . . . . .</b>	<b>83</b>
<b>13.1</b>	<b>O que é canvas ? . . . . .</b>	<b>83</b>
<b>13.2</b>	<b>Estrutura Básica . . . . .</b>	<b>83</b>
13.2.1	Criando nosso primeiro canvas . . . . .	83
13.2.1.1	Desenhando . . . . .	86
<b>14</b>	<b>ATIVIDADES . . . . .</b>	<b>90</b>
<b>14.1</b>	<b>Atividades Capítulo 04 . . . . .</b>	<b>90</b>
14.1.1	Atividade 01 . . . . .	90
14.1.1.1	Imagen . . . . .	90
14.1.1.2	Modelo . . . . .	90
<b>14.2</b>	<b>Atividades Capítulo 05 . . . . .</b>	<b>92</b>
14.2.1	Atividade 01 . . . . .	92

<b>14.3</b>	<b>Atividades Capítulo 06</b>	<b>92</b>
14.3.1	Atividade 01	92
<b>14.4</b>	<b>Atividades Capítulo 07</b>	<b>92</b>
14.4.1	Atividade 01	92
14.4.1.1	Modelo	92
<b>14.5</b>	<b>Atividades Capítulo 08</b>	<b>94</b>
14.5.1	Atividade 01	94
<b>14.6</b>	<b>Atividades Capítulo 09</b>	<b>94</b>
14.6.1	Atividade 01	94
<b>14.7</b>	<b>Atividades Capítulo 10</b>	<b>94</b>
14.7.1	Atividade 01	94
<b>14.8</b>	<b>Atividades Capítulo 11</b>	<b>95</b>
14.8.1	Atividade 01	95
<b>14.9</b>	<b>Atividades Capítulo 13</b>	<b>95</b>
14.9.1	Atividade 01	95
<b>15</b>	<b>REFERÊNCIAS</b>	<b>96</b>

# 1 Introdução

## 1.1 Apresentação HTML

### 1.1.1 W3C - *World Wide Web Consortium*

- Em 1989, Tim Bernes-Lee escreveu um sistema chamado World Wide Web.
- Foi responsável pela criação do primeiro navegador Web, do servidor e das páginas Web.
- Além de ter escrito as primeiras especificações para URLs, HTTP e HTML.
- Em 1994, fundou o consórcio chamado World Wide Web (W3C) no laboratório de ciência da computação do MIT que agrupa empresas, órgãos governamentais e organizações independentes, e com objetivos de desenvolver padrões para a criação e a interpretação de conteúdos para a Web.

*Acesse:* <https://www.w3.org/> e <https://pt.wikipedia.org/wiki/W3C> *para definição.*

- Escritórios no mundo: <https://www.w3.org/Consortium/Offices/staff>.

### 1.1.2 WHAT Working Group

- Enquanto o W3C focava suas atenções para a criação da segunda versão do XHTML, um grupo chamado *Web Hypertext Application Technology Working Group* ou WHATWG trabalhava em uma versão do HTML que trazia mais flexibilidade para a produção de websites e sistemas baseados na web.
- O WHATWG foi fundado por desenvolvedores de empresas como Mozilla, Apple e Opera em 2004.

*Acesse:* <http://www.whatwg.org> *para maiores detalhes.*

- Eles não estavam felizes com o caminho que a Web tomava e nem com o rumo dado ao XHTML.
- Por isso, estas organizações se juntaram para escrever o que seria chamado hoje de HTML5.
- Entre outros assuntos que o WHATWG se focava era Web Forms 2.0 que foi incluído no HTML5 e o Web Controls 1.0 que foi abandonado por enquanto.
- A participação no grupo é livre e você pode se inscrever na lista de email para contribuir.

*Acesse:* <https://whatwg.org/mailing-list>

### 1.1.3 W3C - 3 pilares

- A web é baseada em 3 pilares:
  - Um esquema de nomes para localização de fontes de informação na Web, esse esquema chama-se URI (*Uniform Resource Identifier*).
  - Um protocolo de acesso para acessar estas fontes, o HTTP.
  - Uma linguagem de hipertexto para a fácil navegação entre as fontes de informação: o HTML.
- Browser (navegadores) para navegação. Por exemplo: IE, Netscape, Opera, Mozilla Firefox, Safari, Chrome.

### 1.1.4 Registro de Domínio

*Faça a criação do registro do seu domínio em <http://registro.br/>.*

- Apresentação de 2 vídeos: O que é um domínio e como registrar um domínio .br.

### 1.1.5 Significado HTML

#### Linguagem de Marcação de Hipertexto

- Uma linguagem de marcação é utilizada para definir como o conteúdo de um determinado documento deve ser exibido.
- Autor cria, o browser interpreta e exibe de forma clara para leitura e visualização.
- Uma página HTML é um simples arquivo texto com a extensão (.htm) ou (.html).
- O desenvolvimento de páginas se torna simples, barato, já que você precisará somente de um editor de texto e um navegador para visualização.

**É uma linguagem para publicação de conteúdo (texto, imagem, vídeo, áudio e etc) na Web.**

**Hipertexto** são conjuntos de elementos – ou nós – ligados por conexões.

**Elementos** podem ser palavras, imagens, vídeos, áudio, documentos etc.

**Elementos** conectados formam uma grande rede de informação.

**Distribuir** informação de uma maneira global, é necessário haver uma linguagem que seja entendida universalmente por diversos meios de acesso. O HTML se propõe a ser esta linguagem.

### 1.1.6 História do HTML

**Desenvolvido** originalmente por Tim Berners-Lee.

**HTML** ganhou popularidade quando o Mosaic - browser desenvolvido por Marc Andreessen na década de 1990 - ganhou força.

**Desde** então, desenvolvedores e fabricantes de browsers utilizaram o HTML como base, compartilhando as mesmas convenções.

**Entre** 1993 e 1995, o HTML ganhou as versões HTML+, HTML2.0 e HTML3.0, onde foram propostas diversas mudanças para enriquecer as possibilidades da linguagem. Ainda não era considerado um padrão.

**Apenas** em 1997, o grupo de trabalho do W3C responsável por manter o padrão do código, trabalhou na versão 3.2 da linguagem, fazendo com que ela fosse tratada como prática comum.

**Desde** o começo o HTML foi criado para ser uma linguagem independente de plataformas, browsers e outros meios de acesso.

**Interoperabilidade** significa menos custo.

**Cria-se** apenas um código HTML e este código pode ser lido por diversos meios, ao invés de versões diferentes para diversos dispositivos.

**Dessa** forma, evitou-se que a Web fosse desenvolvida em uma base proprietária, com formatos incompatíveis e limitada.

*Acesse:* <http://www.evolutionoftheweb.com/?hl=pt-br> para acompanhar a evolução.

### 1.1.7 Como surgiu o HTML5

- Quando o HTML4 foi lançado, o W3C alertou os desenvolvedores sobre algumas boas práticas que deveriam ser seguidas ao produzir códigos client-side.
- Problemas que o HTML4 não tratava adequadamente:
  - separação da estrutura do código com a formatação e princípios de acessibilidade.
  - dificuldade de manipulação dos elementos via Javascript ou CSS.
  - para criar um sistema com a possibilidade de Drag'n Drop de elementos, era necessário criar um grande script, com bugs e que muitas vezes não funcionavam de acordo em todos os browsers.
  - ainda não trazia diferencial real para a semântica do código.
  - Nova versão: HTML5.
- Enquanto o WHATWG define:
  - as regras de marcação que usaremos no HTML5 e no XHTML,
  - definem APIs (Application Programming Interface) que formarão a base da arquitetura web.
  - API: é um conjunto de rotinas e padrões de programação para acesso a um aplicativo de software ou plataforma baseado na Web.
- Objetivos do HTML5:
  - facilitar a manipulação do elemento possibilitando o desenvolvedor a modificar as características dos objetos de forma não intrusiva e de maneira que seja transparente para o usuário final.
  - fornecer ferramentas para a CSS e o Javascript fazerem seu trabalho da melhor maneira possível.
  - permitir por meio de suas APIs a manipulação das características destes elementos, de forma que o website ou a aplicação continue leve e funcional.
  - criar novas tags e modificar a função de outras.
  - modifica a forma de como escrevemos código e organizamos a informação na página.
- Versões antigas:
  - não continham um padrão universal para a criação de seções comuns e específicas como rodapé, cabeçalho, sidebar, menus e etc.
  - não havia um padrão de nomenclatura de IDs, Classes ou tags.
  - não havia um método de capturar de maneira automática as informações localizadas nos rodapés dos websites.
  - há outros elementos e atributos que sua função e significado foram modificados na versão atual HTML5 e que agora podem ser reutilizados de forma mais eficaz.
    - \* Por exemplo, elementos como tag `<b>` ou `<i>` que foram descontinuados em versões anteriores do HTML agora assumem funções diferentes e entregam mais significado para os usuários.
- Versão Atual HTML 5.1, lançada em 01/11/2016.

**Acesse** <https://www.w3.org/TR/html/>.

- Mais semântica com menos código.
- Mais interatividade sem a necessidade de instalação de plugins e perda de performance.
- É a criação de código interoperável, pronto para dispositivos atuais e futuros,

- Facilita a reutilização da informação de diversas formas.
- O WHATWG tem mantido o foco para manter a retrocompatibilidade. Nenhum site deverá ter de ser refeito totalmente para se adequar aos novos conceitos e regras.
- O HTML5 está sendo criado para que seja compatível com os browsers recentes, possibilitando a utilização das novas características imediatamente.

**Acesse:** <https://en.wikipedia.org/wiki/HTML5> e

<http://www.frontendbrasil.com.br/artigos/a-historia-do-html/>.

## 1.2 Apresentação CSS

### 1.2.1 CSS - Cascading Style Sheets

- É uma linguagem de estilo utilizada para definir a apresentação de documentos escritos em uma linguagem de marcação, como HTML ou XML.
- Seu principal benefício é prover a separação entre o formato e o conteúdo de um documento.
- Ao invés de colocar a formatação dentro do documento, o desenvolvedor cria um link (ligação) para uma página que contém os estilos, procedendo de forma idêntica para todas as páginas de um portal.
- Quando quiser alterar a aparência do portal basta portanto modificar apenas um arquivo.

### 1.2.2 Apresentação CSS3

- Quando o HTML foi criado, as propriedades para efetuar a formatação dos documentos eram feitas no próprio código HTML.
- Desde então, várias tags HTML foram criadas, e a W3C introduziu o conceito de semântica na estrutura.
- Isto significa que: por exemplo a tag <strong>, que inicialmente só permite que o texto continue em negrito no HTML e nada mais.
- Com o conceito de CSS, podemos redefinir a tag, e formatar o texto com a tag <strong> para ser caixa alta ou um tipo de fonte em específico.
- O HTML e CSS estão envolvidos há algum tempo.
- O poder do CSS vem da habilidade e da combinação de regras para diferentes páginas de acordo com os layouts necessários.

## 1.3 Apresentação Javascript

### 1.3.1 Linguagem de Script

- A linguagem de marcação HTML destina-se a estruturar uma página web.
- O CSS destina-se a formatar o conteúdo das páginas aplicando estilos.
- A versão anterior ao HTML5, não possuía funcionalidades para adicionar interatividade avançada às páginas.
- JavaScript foi criada pela Netscape em parceria com a Sun Microsystems, com finalidade de fornecer esta interatividade.

- Primeira versão foi lançada em 1995, e chamava-se JavaScript1.0 e lançada em 1996 juntamente com o navegador Netscape Navigator 2.0.
- Foi desenvolvida para rodar no lado do cliente, isto é, a interpretação e o funcionamento dependem de funcionalidades hospedadas no navegador do usuário.

## 2 Links Diversos

### 2.1 Links sobre acessibilidade na web

- *Dicas para começar a desenvolver para web com acessibilidade:* <https://www.w3.org/WAI/gettingstarted/tips/>.
- *Passos a serem verificados para validar se sua página está adequada para acessibilidade:* <https://www.w3.org/WAI/eval/preliminary>.
- *Estratégias, diretrizes, recursos para tornar a Web acessível a pessoas com deficiência:* <https://www.w3.org/WAI/>.
- *Guias:* <https://www.w3.org/WAI/intro/wcag.php> e <https://www.w3.org/WAI/tutorials/>.
- *Antes e Depois da aplicação de acessibilidade:* <https://www.w3.org/WAI/demos/bad/>.
- *Cartilha sobre acessibilidade na web:*

<http://www.w3c.br/pub/Materiais/PublicacoesW3C/cartilha-w3cbr-acessibilidade-web-fasciculo-I.html>.

- *Acessibilidade:* <http://webaim.org/>.

### 2.2 Links sobre Evolução da Web

- *Páginas salvas:* <https://archive.org/>.
- *Evolução da Web:* <http://www.evolutionoftheweb.com/?hl=pt-br>.

### 2.3 Links sobre Internacionalização

- *Internacionalização (i18n):* <https://www.w3.org/International/> e <https://www.w3.org/International/questions/qa-choosing-encodings>.
  - *i18n:* <https://www.w3.org/International/quicktips/>.
  - *Encodings:* <https://www.w3.org/International/tutorials/tutorial-char-enc/>.
  - *Idiomas na Web:* <https://www.w3.org/International/getting-started/language>.
  - *Unicode Table:* <https://unicode-table.com/en/>.
  - *Unicode Table Emoji:* <https://unicode-table.com/en/sets/emoji/>.
  - *Tags em HTML e XML:* <https://www.w3.org/International/articles/language-tags/>.
  - *Definindo a linguagem do browser:*
- <https://www.w3.org/International/questions/qa-lang-priorities>.

### 2.4 Links sobre formatação com Cores

- *Cores:* <http://htmlcolorcodes.com/>.
- *Barras:* <http://fontawesome.io/icon/bars/>.

## 2.5 Links de referências MDN

- *Referências MDN para HTML:* <https://developer.mozilla.org/en-US/docs/Web/HTML>.
- *Referências MDN para CSS:* <https://developer.mozilla.org/en-US/docs/Web/CSS>.
- *Referências MDN para HTTP:* <https://developer.mozilla.org/en-US/docs/Web/HTTP>.
- *Referências MDN para Javascript:* <https://developer.mozilla.org/en-US/docs/Web/JavaScript>.
- *Referências MDN para Acessibilidade:*  
<https://developer.mozilla.org/en-US/docs/Web/Accessibility>.

## 2.6 Links para formatação de elementos com CSS

- *Referências CSS Flex-box:* <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>.
- *Referências CSS garden:* <http://www.csszengarden.com/>.
- *Referências CSS display e visibility:* <http://webdesign.about.com/od/css/f/blFAQhidden.htm>.
- *Referências CSS responsive data table:* <https://css-tricks.com/responsive-data-table-roundup/>.
- *Referências CSS Emoji / Unicode:* <https://unicode-table.com/en/sets/emoji/>  
[https://en.wikipedia.org/wiki/List\\_of\\_Unicode\\_characters](https://en.wikipedia.org/wiki/List_of_Unicode_characters).
- *Referências Tabela UNICODE:* <https://unicode-table.com/en/>.

## 2.7 Links para formatação de Fontes e ícones

- <http://zocial.smcllns.com/>.
- <http://fontawesome.io/get-started/>.
- <http://weloveiconfonts.com/>.
- <https://fonts.google.com/>.
- <https://css-tricks.com/examples/IconFont/>.
- <http://www.flaticon.com/authors/vaadin>.
- <http://www.flaticon.com/authors>.

## 2.8 Links de Frameworks CSS

- *Bootstrap:* <http://getbootstrap.com.br/>.
- *Foundation:* <http://foundation.zurb.com/>.
- *960Grid:* <http://960.gs/>.
- *Suzy:* <http://susy.oddbird.net/>.

## 2.9 Links para Pré-Processadores CSS

- *Sass*: <http://sass-lang.com/>.
- *LESS*: <http://lesscss.org/>.
- *Stylus*: <http://stylus-lang.com/>.
- *Comparativo entre*:

<https://tableless.com.br/sass-vs-less-vs-stylus-batalha-dos-pre-processadores/>.

## 2.10 Links para Manuais Resumidos de Tags HTML, GitHub, propriedades CSS, Modal.js

- *Estrutura HTML*: <http://www.uxdesign.blog.br/front-end/html5-estrutura-semantica/>.
  - *Instruções do GitHub*:
- [https://services.github.com/on-demand/downloads/pt\\_BR/github-git-cheat-sheet.pdf](https://services.github.com/on-demand/downloads/pt_BR/github-git-cheat-sheet.pdf).
- *Utilizando Modal.js no Bootstrap*: <http://getbootstrap.com/javascript/modals>.
  - *Propriedades CSS*: <https://makeawebsitehub.com/css3-mega-cheat-sheet/>.
  - *Tags HTML*: <https://makeawebsitehub.com/the-html-5-mega-cheat-sheet/>.

## 2.11 Links para reservar espaços de imagens na página

- *Espaços cinzas*: <http://placehold.it>.
- *Espaços com gatinhos*: [http://placekitten.com/](http://placekitten.com).
- *Animais*: [http://lorempixel.com/](http://lorempixel.com).

## 2.12 Links diversos

- *Acesse*: <http://standardista.com/mobile/>.
- *Acesse*: <https://diveintohtml5.com.br/>.
- *Acesse - Torne suas páginas da Web rápidas em todos os dispositivos*: <https://developers.google.com/speed/pagespeed/insights/>.

### 3 Ferramentas para Web

#### 3.1 Validadores

- *Validador de i18n:* <http://validator.w3.org/i18n-checker/>.
- *Validador de marcação HTML:* <https://validator.w3.org/>.
- *Validador de CSS:* <https://jigsaw.w3.org/css-validator/>.
- *Validador Unificado:* <https://validator.w3.org/unicorn/>.
- *Validador de links:* <https://validator.w3.org/checklink>.
- *Acesso rápido a diversas especificações HTML, Javascript, CSS, SVG, XPath:*  
<https://www.w3.org/2009/cheatsheet/>.
- *Validador de acessibilidade de sites:* <http://wave.webaim.org/>.

#### 3.2 Editores

- *Windows:*
  - Notepad++, Visual Studio. <https://notepad-plus-plus.org/>.
- *Mac:*
  - Xcode, TexMate, SublimeText. <https://macromates.com/>.
- *Atom criado pelo GitHub:* <http://atom.io>.
- *XDK da Intel:* <https://software.intel.com/en-us/intel-xdk>.
- *Jsbin.com, Codepen.io, C9.io*
- *Sublime Text:* <https://www.sublimetext.com/>.
- *Documentação Atalhos Sublime Text:*  
[http://docs.sublimetext.info/en/latest/reference/keyboard\\_shortcuts\\_osx.html](http://docs.sublimetext.info/en/latest/reference/keyboard_shortcuts_osx.html)

## 4 Estrutura Semântica do HTML5

### 4.1 Semântica x Estilos

- A semântica refere-se ao significado da palavra na linguagem:
  - A tag `<b>` é negrito e define um estilo, sem significado semântico.
  - Já a tag `<strong>` tem estilo e significado semântico, indica urgência e seriedade quando o texto é lido por um programa de leitura.
  - O `<i>` insere o estilo itálico.
  - Já o `<em>` tem o significado semântico de enfatizar o conteúdo.

*Acesse:* [https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5\\_element\\_list](https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5_element_list)

Figura 1 – Tags Estilos x Tags Semântica

Tag	Type	Description
<code>&lt;b&gt;</code>	Style	Makes text bold
<code>&lt;i&gt;</code>	Style	Makes text italics
<code>&lt;em&gt;</code>	Semantic	Emphasizes text Text is italics by default in a browser
<code>&lt;strong&gt;</code>	Semantic	Important text Text is bold by default in a browser

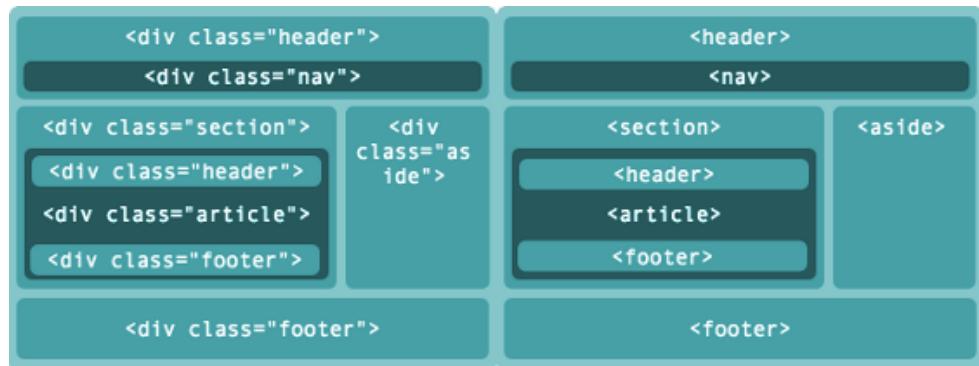
Fonte: <http://www.edx.org>

### 4.2 HTML Semântico

- **Significado:** É o HTML que concentra no significado da informação nas páginas web ao invés de uma simples apresentação visual.
- Nas versões anteriores do HTML não haviam tags com uma semântica apropriada para cada uma dessas divisões.
- Dessa forma, os desenvolvedores acabavam usando a tag `<div>` para todas as situações, e criando seus próprios padrões de nomeclaturas através dos atributos `id` ou `class`.

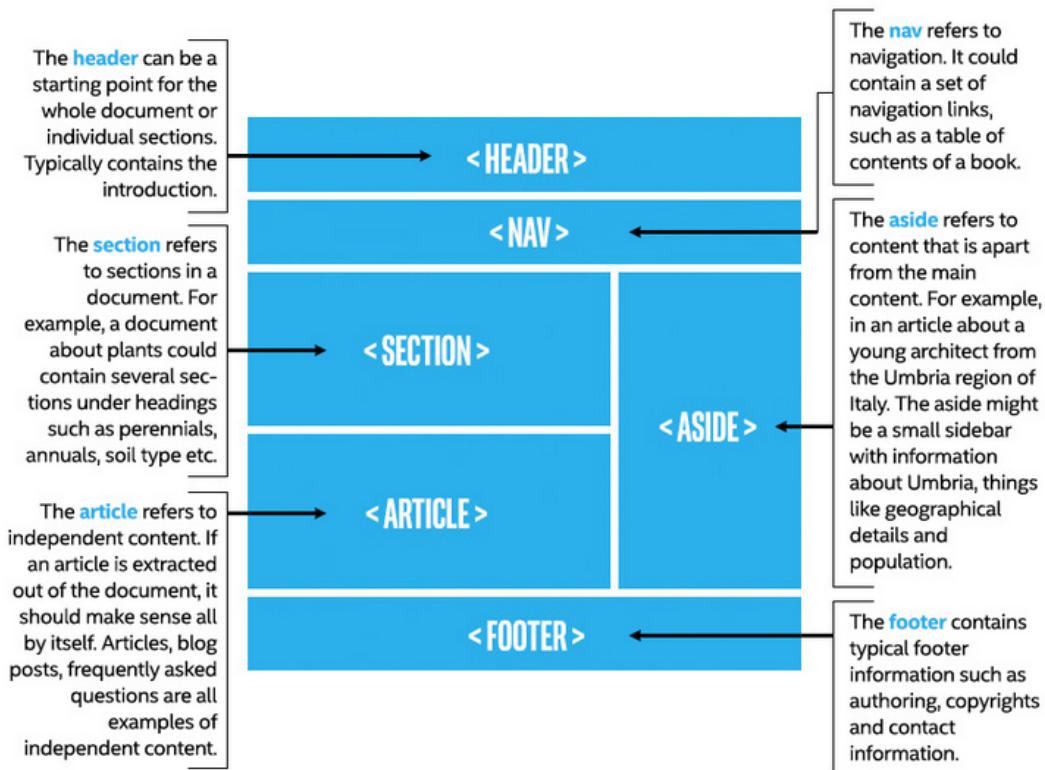
### 4.3 Representação da Estrutura HTML5

Figura 2 – Nova estrutura do HTML5

Fonte: <http://www.uxdesign.blog.br/front-end/html5-estrutura-semantica/>

### 4.4 Significado de cada item da Estrutura HTML5

Figura 3 – Significado nova estrutura do HTML5

Fonte: <http://www.edx.org>

- **header**: Contém a introdução, informações iniciais. É usado para definir o cabeçalho de uma página ou sessão, e pode conter logo, títulos, menu de navegação, campo de busca, etc.
- **section**: É a sessão do documento.
- **nav**: Links para navegação.
- **article**: Conteúdo independente.

- **aside:** Conteúdo à parte do conteúdo principal. Footer: Conteúdo referente a informações relacionadas ao autor e ao copyright, blocos de navegação ou links relacionados.

*Acesse:* <http://colinaut.com/2009/10/15/the-semantics-of-html5-structural-elements/>

<http://www.uxdesign.blog.br/front-end/html5-estrutura-semantica/>

<https://www.caelum.com.br/apostila-html-css-javascript/html-semantico-e-posicionamento-no-css/>

## 4.5 Exemplo da nova estrutura

Figura 4 – Estrutura HTML5 Semântico

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3      <head>
4          <meta charset="utf-8">
5          <title>Página Exemplo</title>
6      </head>
7      <body>
8          <h1>Título Principal</h1>
9          <header>
10             <h2>Título do Header</h2>
11         </header>
12         <article>
13             <h2>Título do Article</h2>
14             <section id="id1part1">
15                 <h3>Título Seção I</h3>
16             </section>
17             <section id="id2part2">
18                 <h3>Título Seção II</h3>
19             </section>
20         </article>
21         <nav>
22             <h2>Título do Nav</h2>
23             <ul>
24                 <li><a href="#">Next post</a></li>
25                 <li><a href="#">Previous post</a></li>
26                 <li><a href="#">Contact author</a></li>
27             </ul>
28         </nav>
29         <aside>
30             <h2>Título do Aside</h2>
31         </aside>
32         <footer>
33             <h2>Título do Footer</h2>
34         </footer>
35     </body>
36 </html>
```

Fonte: autoria própria

## 4.6 Acessibilidade e tecnologias assistivas

- Propõe acessibilidade das páginas Web.
- Tecnologias assistivas dependem do documento para apresentarem as informações aos usuários.

*Acesse:* <https://www.w3.org/standards/webdesign/accessibility>

<https://www.w3.org/WAI/gettingstarted/tips/>

<https://www.w3.org/WAI/eval/preliminary.>

## 4.7 Tags do HTML5

- São diversas as tags para se utilizar no código HTML. Consulte.

*Acesse:* <https://developer.mozilla.org/en-US/docs/Web/HTML/Element>.

## 4.8 Search Engine Optimization (SEO)

- De acordo com <https://moz.com/beginners-guide-to-seo>:
  - A maioria do tráfego web é conduzido pelos principais motores de busca comerciais: Google, Bing e Yahoo!
  - Se os motores de busca não conseguirem encontrar o seu site ou adicionar o seu conteúdo aos seus bancos de dados, perderá oportunidades incríveis de gerar tráfego para o seu Web site.

*Acesse:* <http://www.seemarketing.com.br/meta-tags-google.php> *e*

<https://moz.com/beginners-guide-to-seo> *e*

<https://moz.com/beginners-guide-to-seo/how-search-engines-operate>.

# 5 Seletores, links de estilo, hiperlinks, cores

## 5.1 Seletores Básicos

Os seletores são classificados de 3 tipos:

- Seletores HTML
- Seletores de Classes
- Seletores Identificadores

Figura 5 – Seletores

**TABLE 3.1 Basic Selectors**

Format	Selector Name	What Elements Are Styled	Compatibility
a	HTML	All specified HTML tags	IE4, FF1, S1, C1, O3.5, CSS1
.myClass	Class	Any HTML tag where class is applied	IE4, FF1, S1, C1, O3.5, CSS1
a.myClass	Dependent Class	Specified HTML tag where class is applied	IE4, FF1, S1, C1, O3.5, CSS1
#myID	ID	Any HTML tag where ID applied	IE4, FF1, S1, C1, O3.5, CSS1
a#myID	Dependent ID	Specified HTML tag where ID is applied	IE4, FF1, S1, C1, O3.5, CSS1
*	Universal	All HTML tags	IE7, FF1, S1, C1, O3.5, CSS2

Fonte: TEAGUE J. C.

### 5.1.1 Seletores HTML

- Utiliza-se as tags HTML para definir as formatações no arquivo de estilo CSS.

Figura 6 – Arquivo <estilo.css> com Seletores HTML

```

1 p {
2   color: #ff1493;
3 }
4
5 a {
6   background-color: #ff1493;
7 }
```

Fonte: autoria própria

### 5.1.2 Seletores de Classes

- Utiliza-se as classes nomeadas das tags HTML para definir as formatações no arquivo de estilo CSS.

Figura 7 – Arquivo &lt;estilo.css&gt; com Seletores de Classes

```

1 .rodape {
2     color: #ff1493;
3 }
4
5 p.rodape {
6     color: #ff8765;
7 }
8
9 .cabecalho {
10    background-color: #ff1493;
11 }

```

Fonte: autoria própria

### 5.1.3 Seletores Identificadores

- Utiliza-se os identificadores (id) das tags HTML para definir as formatações no arquivo de estilo CSS.

Figura 8 – Arquivo &lt;estilo.css&gt; com Seletores Identificadores

```

1 #texto {
2     color: #ff1493;
3 }
4
5 #imagem {
6     background-color: #ff1493;
7 }
8
9 p#texto {
10    color: #ff1823;
11 }

```

Fonte: autoria própria

### 5.1.4 Seletores Universal

- Aplicado em todas as tags do documento HTML.

Figura 9 – Arquivo &lt;estilo.css&gt; com Seletores Universal

```

1 * {
2     color: #ff1493;
3 }

```

Fonte: autoria própria

## 5.2 Carregando a formatação no HTML

- Há duas formas de se utilizar o arquivo CSS no HTML:
  - Inserindo o código do arquivo de estilo dentro do HTML.
  - Inserindo um link para um arquivo de estilo externo ao HTML.

### 5.2.1 Exemplo1: arquivo de estilo dentro do HTML

Arquivo <index.html>

Figura 10 – Arquivo de estilo dentro do HTML

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>Aula02</title>
6     <style>
7       #texto {
8         color: #ff1493;
9       }
10
11      #imagem {
12        background-color: #ff1493;
13      }
14
15      p#texto {
16        color: #ff1823;
17      }
18      @import url("reveal.css");
19    </style>
20  </head>
21  <body>
22    <!-- inserir seu código html aqui -->
23  </body>
24 </html>

```

Fonte: autoria própria

### 5.2.2 Exemplo2: arquivo de estilo externo ao HTML

Arquivo &lt;index.html&gt;

Figura 11 – Arquivo de estilo externo ao HTML

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>Aula02</title>
6     <link rel="stylesheet" href="css/estilo.css">
7   </head>
8   <body>
9     <!-- inserir seu código html aqui -->
10  </body>
11 </html>

```

Fonte: autoria própria

## 5.3 Utilizando hiperlink no HTML

- É um texto ou imagem que ao ser selecionado, direciona o usuário para:
  - Outra página.
  - Uma parte do texto da própria página (#book).
  - Ou um link local, email.
- Link para uma página externa: <a href="http://google.com">Site</a>
- Link para um arquivo/página local: <a href="contatos.html">Página</a>
- Link para um elemento da própria página: <a href="#detalhes">Elemento</a>

- Link para um email (será utilizado o software de email do computador):
  - `<a href="mailto:paulagiancoli@ifsp.edu.br">Email</a>`
- *Estados do hiperlink*:
  - unvisited (azul e subscrito),
  - visited (roxo e subscrito),
  - active (vermelho e subscrito).

## 5.4 Utilizando cores para formatação com o arquivo CSS

- A formatação de estilo utilizando as cores pode ser feita de diversas formas:
- Utilizando-se:
  - a palavra chave da cor.
  - a palavra transparent.
  - a cor corrente.
  - RGB (Red, Green, Blue) em hexadecimal.
  - RGB em decimal.
  - RGB em percentagem.
  - RGBA com opacidade.
  - HSL (Hue-matiz, Saturation-saturação, Lightness-brilho)
  - HSLA com opacidade.

Figura 12 – Formatação utilizando os padrões de cores

**TABLE 7.1 Color Values**

Name	Form	Example	Compatibility
Keyword	<keyword>	coral	IE4, FF1, S1, C1, O3.5, CSS1
Transparent	transparent	–	IE4, FF1, S1, C1, O3.5, CSS1
Current color	currentcolor	–	FF3.6, S2, C2, O10, CSS3
RGB hex	#rrggbb, #rgb	#ff7f50, #f90	IE4, FF1, S1, C1, O3.5, CSS1
RGB decimal	rgb(rrr,ggg,bbb)	255,127,80	IE4, FF1, S1, C1, O3.5, CSS1
RGB percentage	rgb(rrr%,ggg%,bbb%)	rgb(100%, 50%,31%)	IE4, FF1, S1, C1, O3.5, CSS1
HSL	hsl(hhh,sss%,lll%)	hsl(16,65%,100%)	FF3, S2, C2, O9.5, CSS3
RGBA	rgba(rrr,ggg,bbb,d.d)	rgba(255,127,80,.86)	FF3.6, S3, C3, O9.5, CSS3
HSLA	hsla(hhh,sss%,lll%,d.d)	hsl(16,65%,100%,.23)	FF3.6, S3, C3, O9.5, CSS3

Fonte: TEAGUE J. C.

- *Hue* é um grau na roda de cores; 0 (ou 360) é vermelho, 120 é verde, 240 é azul. As variações entre estes valores refletem as diferentes tonalidades.
- *Saturation* é um valor percentual; 100% é a cor completa.
- *Lightness* é também uma porcentagem; 0% é escuro (preto), 100% é claro (branco) e 50% é a média.

Acesse: <https://www.w3.org/TR/2003/CR-css3-color-20030514/>.

#### 5.4.1 Exemplo3: arquivo de estilo com exemplos de formatações de cores

Arquivo <estilo.css>

Figura 13 – Arquivo de estilo com formatações de cores

```
1  p { color: blue; }
2  a { color: transparent; }
3  .roteiro { color: currentColor; }
4  /* usar somente 3 posições #f00; */
5  .classe { color: #ff0000; }
6  #roteiro { color: rgb(255, 104, 154); }
7  /* o valor de a, varia de 0 (transparente) a 1 (opaco)*/
8  .rodape { color: rgba(255, 104, 154, 0.7); }
9  #rodape { color: hsl(16, 65%, 100%); }
10 /* o valor de a, varia de 0 (transparente) a 1 (opaco)*/
11 #cabecalho { color: hsla(16, 65%, 100%, .23); }
```

Fonte: autoria própria

# 6 Propriedades CSS de Fontes e Textos

## 6.1 Fontes

- Pesquisas indicam que para uma boa leitura dos sites, cada linha de texto deve conter entre 45 a 90 caracteres. A quantidade de caracteres depende da família de fonte utilizada.
- O padrão é 65 caracteres.
- Sugestões:
  - para dispositivos pequenos, utilizar o tamanho da fonte: 16px, altura mínima linha: 1.2em. Ou,
  - utilizar o tamanho da fonte: 18px, altura mínima linha: 1.2em.

### 6.1.1 Fontes genéricas

- Permite formatar a família de fontes. Valores:
  - Serif
  - Sans-serif
  - Monospace
  - Cursive
  - Fantasy
- Caso o navegador não encontre a fonte desejada, utiliza o default do micro.
- Se o nome for composto, deve ser declarado entre aspas duplas.

Figura 14 – Exemplo de formatação font-family

**A fonte utilizada é Serif**  
**A fonte utilizada é Sans-Serif**  
**A fonte utilizada é Monospace**  
**A fonte utilizada é Cursíve**  
**A fonte utilizada é Fantasy**

Fonte: autoria própria

#### 6.1.1.1 Exemplo de fontes genéricas

Figura 15 – Parte código formatação font-family

```
14 .class {
15   font-family: serif, sans-serif, "Times New Roman";
16 }
```

Fonte: autoria própria

#### 6.1.1.2 Webfonts

- **font-family:** indica o nome da fonte a ser utilizada.

- **src: url(' ')**: indica o local da fonte com a extensão.
- IE aceita somente extensão EOT. Demais aceitam TTF e OTF. Fontes com direitos autorais, pagas.

Figura 16 – Regra do CSS @font-face para font-family com webfonts

```

18 @font-face {
19   font-family: helvetica neue;
20   src: url('Helvetica Neue LT Std-UltLt.otf');
21 }

```

Fonte: autoria própria

**Acesse:** <https://www.fontsquirrel.com/fonts/list/popular> e<https://tableless.com.br/font-face-fonts-externas-na-web/>.

#### 6.1.1.3 Google Fontes

- Para utilizar as fontes do google, acesse o link, escolha a fonte a utilizar e inserir no seu arquivo html o link para a fonte, e no seu arquivo css, a regra de formatação da fonte.

Figura 17 – Arquivo html com link para o Google Fontes com a fonte selecionada

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>Aula02</title>
6     <link rel="stylesheet" href="css/estilo.css">
7     <link href="https://fonts.googleapis.com/css?family=Roboto" rel="stylesheet">
8   </head>
9   <body>
10    <!-- inserir seu código html aqui -->
11  </body>
12 </html>

```

Fonte: autoria própria

Figura 18 – Arquivo css com a formatação da fonte selecionada no Goolge Fontes

```

23 .class {
24   font-family: 'Roboto', sans-serif;
25 }

```

Fonte: autoria própria

**Acesse:** <https://fonts.google.com>.

#### 6.1.2 Propriedades de Fontes

- **font-size:** permite formtar o tamanho da fonte.
  - valores possíveis: pixel, percentage, xx-small, x-small, small, smaller medium, large, x-large, xx-large, larger.
  - **Exemplo:** font-size: 16px.
- **font-size-adjust:** permite formatar a altura da fonte lowercase em relação a uppercase.
  - valores possíveis: valor númerico múltiplo do tamanho da fonte.

- utilize o valor `none` para não formatar este item.
- **Exemplo:** `font-size: 16px;` Se `font-size-adjust: .5;`, as letras menores serão exibidas na altura de `8px`.
- ***font-style*:** permite formatar o estilo da fonte.
  - valores possíveis: `normal`, `italic`, `oblique`.
  - **Exemplo:** `font-style: italic;`
- ***font-weight*:** permite formatar o negrito da fonte.
  - valores possíveis: `normal`, `bold`.
  - **Exemplo:** `font-weight: bold;`
- ***font-variant*:** permite formatar a fonte `lowercase` em `uppercase`, em tamanho menor.
  - valores possíveis: `normal`, `small-caps`.
  - **Exemplo:** `font-variant: small-caps;`
- ***line-height*:** permite formatar a altura de linha.
  - utilizar um múltiplo do tamanho da fonte.
  - Se usar `2`, equivale a espaçamento duplo.
  - valores possíveis: `px`, `em`, `%`.
  - dê preferência para `em`.
  - **Exemplo:** `line-height: 2em;`
- ***font*:** permite formatar a fonte de uma única vez.
  - Verificar cada uma das propriedades que são opcionais.
  - O valor da altura de linha: `line-height` poderá ser `em` `em`, `%`, `px`.
  - **Exemplo:** `font: italic bold small-caps 3em/.9 'Title Font', Constantia, Georgia, Times, "Times New Roman";`

### 6.1.3 Exemplos do uso de propriedades de formatação de fontes

Figura 19 – Arquivo css com as propriedades de formatação da fontes

```
.class {
    font-size: valor;
    font-size-adjust: valor;
    font-style: valor;
    font-weight: valor;
    font-variant: valor;
    font: font-style font-variant font-weight font-size(obrigatório)/line-height font-family(obrigatório);
}
```

Fonte: autoria própria

obs.: usar as propriedades individualmente ou somente `font`.

**Acesse:** <https://developer.mozilla.org/pt-BR/docs/Web/CSS/font>.

## 6.2 Unidades de Medida

### 6.2.1 Relativas

Unidade	Descrição
em*	relativo ao tamanho da fonte do elemento
ex	relativo ao x-height da fonte corrente (raro usar)
ch	relativo a width de 0
rem*	relativo ao tamanho da fonte do elemento root
vw	relativo a 1% do width do viewport
vh	relativo a 1% da height do viewport
vmin	relativo a 1% da dimensão menor do viewport
vmax	relativo a 1% da dimensão maior do viewport
%	
<i>Viewport = tamanho da janela do navegador</i>	
Se <i>viewport = 50cm wide, 1vw = 0.5cm.</i>	

### 6.2.2 Absolutas

Unidade	Descrição	Conversão
cm	centímetros	
mm	milímetros	
in	polegadas	1in = 96px = 2.54cm
px*	pixels	1px = 1/96 in
pt	pontos	1pt = 1/72 in
<i>Valor default dos navegadores: 16px</i>		
<i>1em = 100% = 16px</i>		

## 6.3 Textos

### 6.3.1 Formatação de cores dos textos

- **color:** permite formatar a cor da texto.
  - valores possíveis: rgb, rgba, hexadecimal, hsl.
  - Verificar aula anterior sobre cores.
  - **Exemplo:** color: rgb(245, 0, 35);
- **text-align:** permite formatar o alinhamento horizontal do texto.
  - valores possíveis: right, left, center, justify.
  - **Exemplo:** text-align: right;
- **text-transform:** permite formatar como o texto será visualizado.
  - valores possíveis: none, capitalize, uppercase, lowercase.
  - **Exemplo:** text-transform: capitalize;
- **text-decoration:** permite formatar o subescrito do texto.
  - valores possíveis: none, overline, line-through, underline.
  - **Exemplo:** text-decoration: none;

- **text-indent:** permite formatar a indentação da primeira linha do texto
  - valores possíveis: px, em, %.
  - **Exemplo:** text-indent: 10px;
- **text-direction:** permite formatar a direção do texto.
  - valores possíveis: ltr (valor default), rtl, initial, inherit.
  - **Exemplo:** text-direction: rtl;
- **text-shadow:** permite formatar a sombra do texto. Blur = borrão. Se for 0px, a sombra será sólida.
  - valores possíveis: none, initial, inherit.
  - **sintaxe:** text-shadow: h-shadow v-shadow blur color;
    - \* Valores h-shadow e v-shadow positivos (direita, abaixo)
    - \* Valores h-shadow positivo e v-shadow negativo (direita, acima)
    - \* Valores h-shadow e v-shadow negativos (esquerda, acima)
    - \* Valores h-shadow negativo e v-shadow positivo (esquerda, abaixo)
  - **Exemplo:** text-shadow: 2px 2px 3px #fff;
- **letter-spacing:** permite formatar o espaçamento entre letras.
  - valores possíveis: normal, px, em, %.
  - o tamanho pode ser em pixel, em, e se negativo, aproxima as letras.
  - dê preferência para em.
  - **Exemplo:** letter-spacing: 2em;
- **word-spacing:** permite formatar o espaçamento entre palavras.
  - valores possíveis: normal, px, em, %.
  - o tamanho pode ser em pixel, em, e se negativo, aproxima as palavras.
  - dê preferência para em.
  - **Exemplo:** word-spacing: 2em;
- **vertical-align:** permite formatar o alinhamento vertical do texto. Funciona somente com elementos inline, inline-block, table-cells.
  - valores possíveis: baseline (valor default), px, em, %, initial, inherit, sub, super, top, text-top, middle, bottom, text-bottom.
  - **Exemplo:** vertical-align: middle;
- **white-spacing:** permite formatar o alinhamento vertical do texto.
  - valores possíveis:
    - \* normal (valor default),
    - \* nowrap (mais de um espaço é compactado em um só, não quebra linha),
    - \* pre (espaço é preservado pelo navegador, quebra se tiver br),
    - \* pre-line (mais de um espaço é compactado em um só),
    - \* pre-wrap (espaço é preservado pelo navegador),
    - \* initial,
    - \* inherit.

- **Exemplo:** white-spacing: pre-line;

Figura 20 – Arquivo css com as propriedades de formatação de textos

```
.class {  
    color: valor;  
    text-align: valor;  
    text-transform: valor;  
    text-decoration: valor;  
    text-indent: valor;  
    text-direction: valor;  
    text-shadow: h-shadow v-shadow blur color;  
    letter-spacing: valor;  
    word-spacing: valor;  
    line-height: valor;  
    vertical-align: valor;  
    white-space: valor;  
}
```

Fonte: autoria própria

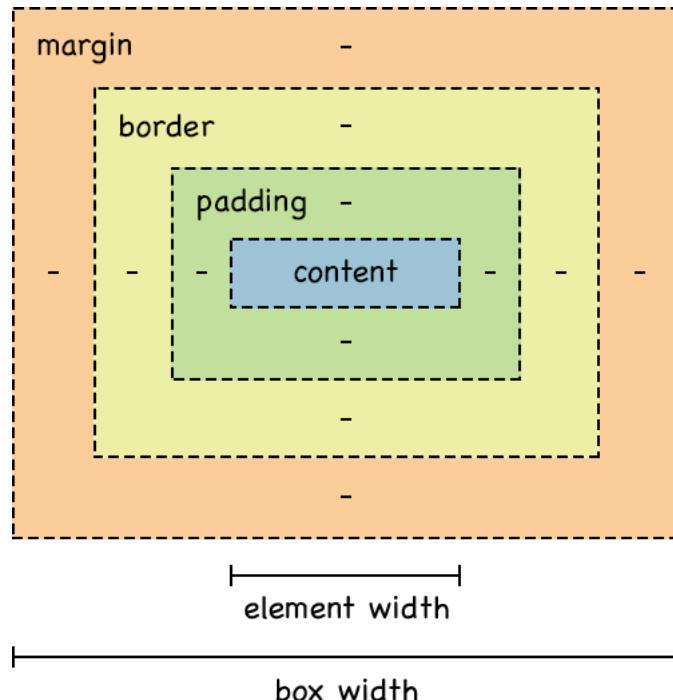
Acesse: [https://developer.mozilla.org/en-US/docs/Learn/CSS/Styling\\_text/Fundamentals](https://developer.mozilla.org/en-US/docs/Learn/CSS/Styling_text/Fundamentals).

# 7 Propriedades CSS de Layout

## 7.1 Box Model padrão do CSS

- O box model (modelo das caixas) em CSS, descreve os boxes (as caixas) geradas pelos elementos HTML.
- O box model, detalha ainda, as opções de ajuste de margens, bordas, padding e conteúdo para cada elemento.
- É uma propriedade que determina o tamanho final (altura e largura) dos elementos.
- E o box model tradicional da especificação tem uma regra bastante confusa:
  - a propriedade ***element width*** trata do tamanho do conteúdo do elemento, não considerando seu ***padding*** e ***border*** (é a mesma coisa com a altura).
  - Se você quiser que um elemento ocupe metade da página mas com uma borda de 10px, não funciona utilizar:
    - \* border : 10px solid #555;
    - \* width : 50%;
    - \* O tamanho final do elemento terá 20px mais metade do tamanho do pai.
- Surge então a propriedade ***box-sizing*** que considera o tamanho final do elemento, considerando o ***padding*** e ***border***.

Figura 21 – Diagrama representativo do Box Model



Fonte: udacity.com

### 7.1.1 Box-sizing

- Por padrão todos os elementos possuem o valor do ***box-sizing: content-box***.
  - valores possíveis:

- \* border-box: indica que o tamanho considerará até a borda do elemento (**width = border + padding**).
- \* content-box: indica que o tamanho dele é definido pelo seu conteúdo apenas (valor default). É o box model tradicional (**width = element width**).
- Os elementos em blocos, ocupam o máximo da largura e o mínimo de altura.
- Os elementos em linha, ocupam o espaço do conteúdo. Quando se usa valores em %, os cálculos das larguras e alturas dos elementos são feitos em relação aos elementos pais.
- As versões mais antigas do Firefox precisa do prefixo **-moz-** e, Chrome, Safari e Android, do **-webkit-**.

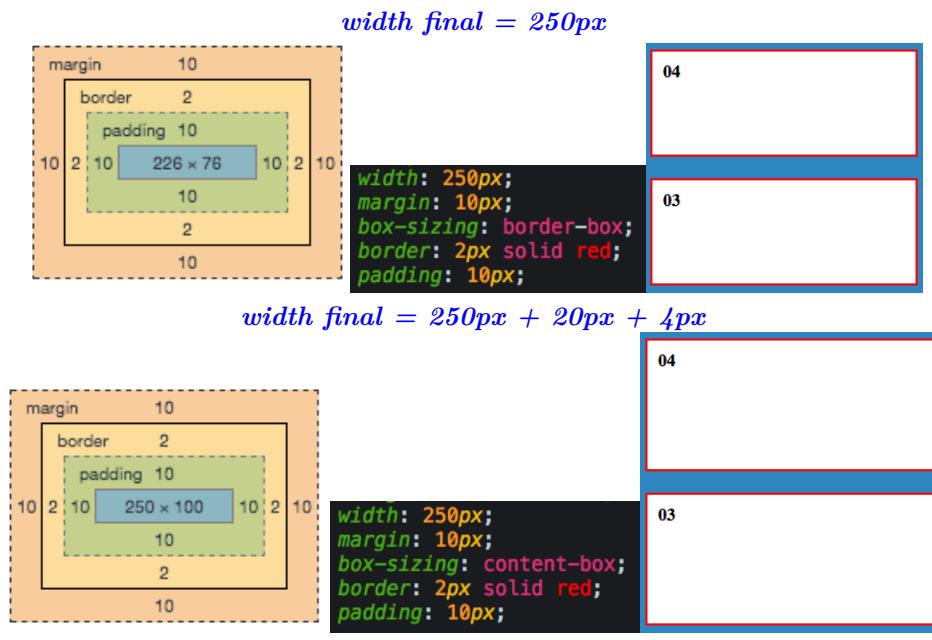
Figura 22 – Uso dos prefixos

```
-webkit-box-sizing: border-box;
-moz-box-sizing: border-box;
box-sizing: border-box;
```

Fonte: autoria própria

- **Exemplo:**

Figura 23 – Comparaçāo box-sizing: border-box e box-sizing: content-box



Fonte: autoria própria

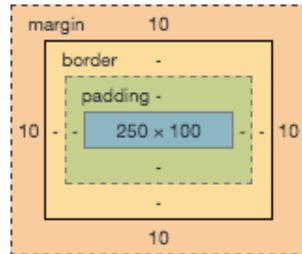
Acesse: <https://developer.mozilla.org/en-US/docs/Web/CSS/box-sizing>.

## 7.2 Margin

- É a distância a partir de cada um dos lados e de seus elementos vizinhos (ou às bordas do documento).
- **margin:** aplica formatação do espaçamento do elemento em relação a seus adjacentes.
  - valores possíveis: em, pixels, %.
  - **Sintaxe:**
    - \* **margin:** top/bottom/right/left;

- \* **margin:** top/bottom right/left;
- \* **margin:** top right/left bottom;
- \* **margin:** top right bottom left;
- **Exemplo:** margin: 10px 10px 5px 5px; (formata o espaço entre o elemento e os vizinhos em relação ao topo em 10px, direita em 10px, abaixo em 5px, esquerda em 5px).

Figura 24 – Margin



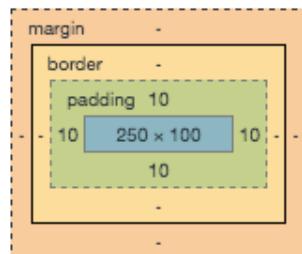
Fonte: autoria própria

**Acesse:** <https://developer.mozilla.org/en-US/docs/Web/CSS/margin>.

### 7.3 Padding

- É a distância do elemento entre a borda e o seu conteúdo.
- **padding:** aplica formatação do espaçamento ao redor do elemento em relação a sua borda.
  - valores possíveis: em, pixels, %.
  - **Sintaxe:**
    - \* **padding:** top/bottom/right/left;
    - \* **padding:** top/bottom right/left;
    - \* **padding:** top right/left bottom;
    - \* **padding:** top right bottom left;
  - **Exemplo:** padding: 10px 10px 5px 5px; (formata o espaço entre o conteúdo e a borda do topo em 10px, direita em 10px, abaixo em 5px, esquerda em 5px).

Figura 25 – Padding



Fonte: autoria própria

**Acesse:** <https://developer.mozilla.org/en-US/docs/Web/CSS/padding>.

## 7.4 Position

- Indica ao browser como ele tratará aquele elemento.
- **position:** Aplica formatação à posição dos elementos.
  - valores possíveis:
    - \* **static:** por default, os elementos são static.
      - Não pode definir posição com left, right, top e bottom.
      - Onde o elemento estiver ele ficará.
    - \* **relative:** mantém na posição do fluxo normal do documento.
      - Permitido definir, top, left, right e bottom.
      - O elemento será posicionado aonde for designado, caso esteja sozinho. Caso contrário, será movido conforme valores atribuídos e demais elementos da página.
      - O posicionamento é sempre feito em relação os demais elementos da página.
      - Valores negativos, movem os elementos para sentidos inversos.
    - \* **absolute:** leva o elemento para fora do fluxo normal do documento, deixando um espaço pra trás.
      - Em um elemento definido desta forma, as coordenadas x, y são relativas a seu pai, posicionando em relação as bordas do elemento ou corpo, se não tiver elemento pai.
    - \* **fixed:** similar ao absoluto, porém, está travado na posição relativa ao viewport.
      - Quando usa scroll, os elementos fixos não serão movidos com o resto do documento.

**Exemplos de posicionamento - Acesse:** <https://scotch.io/bar-talk/5-things-you-might-not-know-about-the-css-positioning-types>.

**Acesse:** <https://developer.mozilla.org/en-US/docs/Web/CSS/position>.

**Acesse:** <https://css-tricks.com/css-grid-one-layout-multiple-ways/>.

## 7.5 Float

- O Float é uma propriedade que faz o objeto flutuar à esquerda ou à direita do restante do conteúdo. Mediante o arquivo html abaixo, aplicaremos algumas modificações para exemplificar o uso do float.

Figura 26 – Arquivo html com as propriedades CSS com a tag &lt;style&gt;

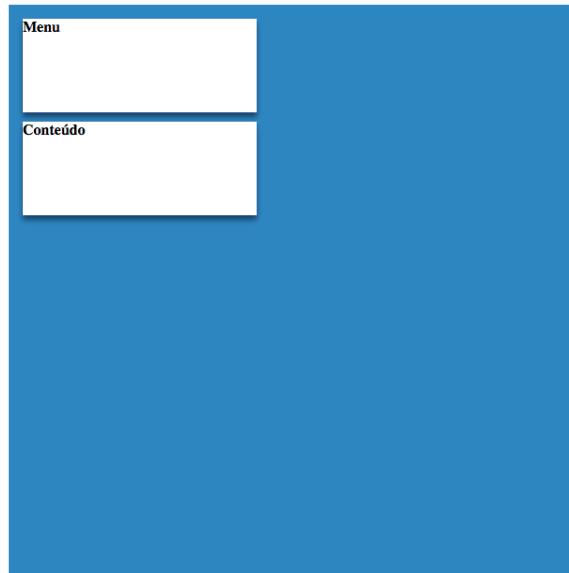
```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title></title>
6      <style type="text/css">
7          .tudo {
8              background-color: #2E86C1;
9              height: 600px;
10             width: 600px;
11             padding: 5px;
12         }
13
14         .menu, .conteudo {
15             height: 100px;
16             width: 250px;
17             color: #000;
18             font-weight: bold;
19             box-shadow: 0px 5px 8px rgba(0, 0, 25, 0.5);
20             background-color: #fff;
21             margin: 10px;
22         }
23     </style>
24 </head>
25 <body>
26     <div class="tudo">
27         <div class="menu">Menu</div>
28         <div class="conteudo">Conteúdo</div>
29     </div>
30 </body>
31 </html>

```

Fonte: autoria própria

Figura 27 – Visualização no navegador com arquivo html e as propriedades CSS com a tag &lt;style&gt;



Fonte: autoria própria

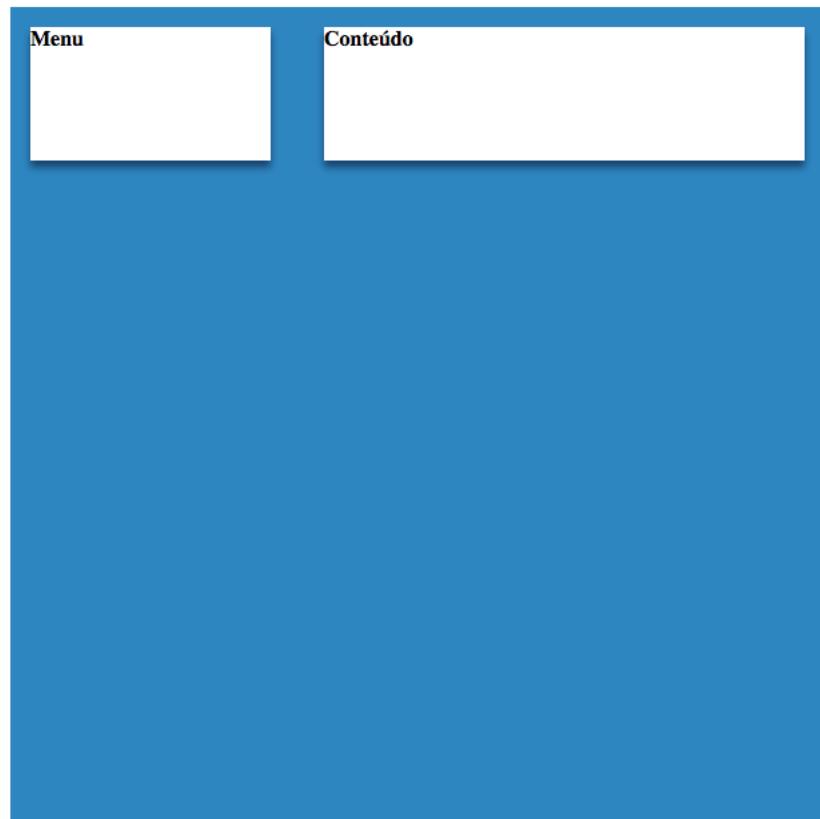
- **float:** Aplica a formatação no elemento de forma a mostrar como ele interage com os demais.
  - valores possíveis:
    - \* right, left ou none.

Figura 28 – Aplicando a propriedade float nos elementos menu e conteúdo

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title></title>
6      <style type="text/css">
7          .tudo {
8              background-color: #2E86C1;
9              height: 600px;
10             width: 600px;
11             padding: 5px;
12         }
13         .menu, .conteudo {
14             height: 100px;
15             width: 250px;
16             color: #000;
17             font-weight: bold;
18             box-shadow: 0px 5px 8px rgba(0, 0, 25, 0.5);
19             background-color: #fff;
20             margin: 10px;
21         }
22         .menu {
23             float: left;
24             width: 30%;
25         }
26         .conteudo {
27             float: right;
28             width: 60%;
29         }
30     </style>
31 </head>
32 <body>
33     <div class="tudo">
34         <div class="menu">Menu</div>
35         <div class="conteudo">Conteúdo</div>
36     </div>
37 </body>
38 </html>
```

Fonte: autoria própria

Figura 29 – Visualização no navegador com a propriedade float aplicada



Fonte: autoria própria

- **clear:** Limpa a formatação dos elementos que usaram float.

- valores possíveis:
    - \* right, left, both ou none.

**Float, Clear, Clearfix - Acesse:** <https://edsonjunior.com/entendendo-float-clear-clearfix/>.

**Exemplos de Float Acesse:**

<http://www.richardbarros.com.br/blog/css/css-truques-para-dominar-a-propriedade-float>.

**Acesse:** <https://developer.mozilla.org/en-US/docs/Web/CSS/float>.

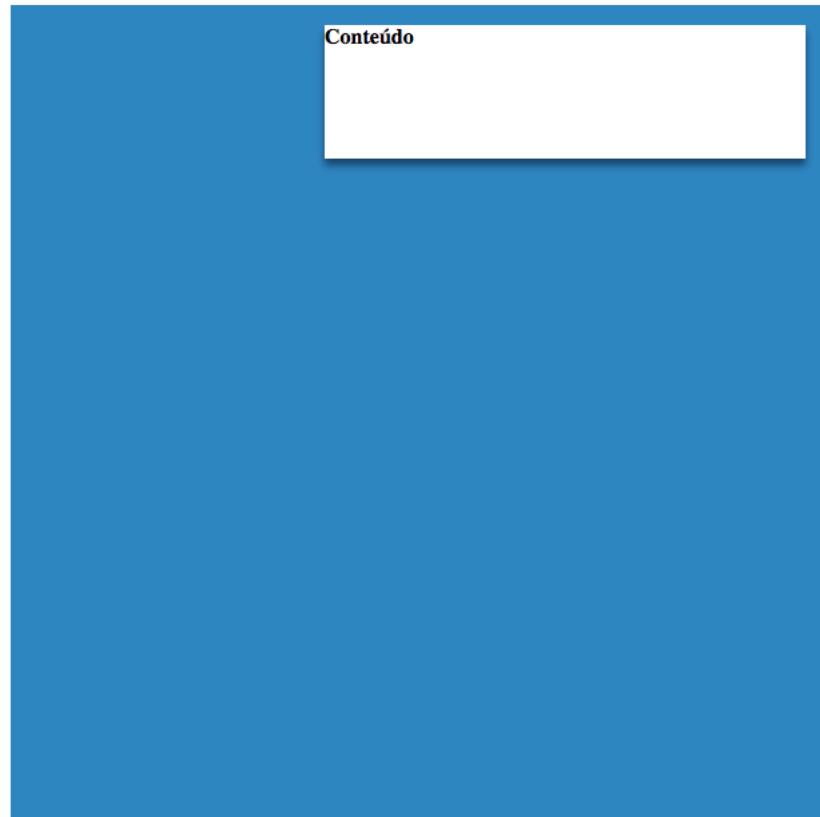
## 7.6 Visibility

- Pode ser usada para ocultar um elemento enquanto deixa o espaço onde ele teria sido criado. Também utilizado para ocultar linhas e colunas de uma tabela.

- **visibility:**

- valores possíveis:
    - \* visible: valor default, o elemento está sempre visível.
    - \* hidden: esconde o elemento não modifica o layout. Os filhos serão visíveis se estiverem com o valor visible.
    - \* collapse: as linhas ou colunas estão ocultas e o espaço que ocuparam é removido.

Figura 30 – Visualização no navegador com a propriedade CSS visibility: hidden



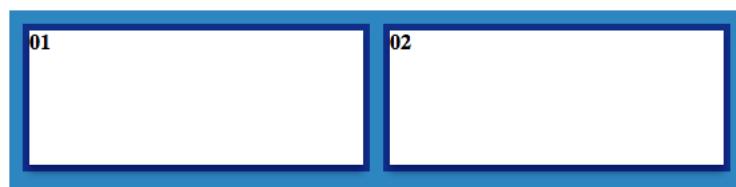
Fonte: autoria própria

Acesse: <https://developer.mozilla.org/en-US/docs/Web/CSS/visibility>.

## 7.7 Outline

- **outline**: Aplica a formatação ao redor do elemento, externo às bordas.
  - valores possíveis: de cor, de estilo de linha e espessura.
    - \* **outline-color: valor;**
    - \* **outline-style: valor;**
    - \* **outline-width: valor; ou**
    - \* **outline: outline-color outline-style outline-width;**
    - \* **Exemplo: outline: rgba(0, 0, 107, .65) solid 5px;**

Figura 31 – Exemplo - Visualização no navegador com a propriedade CSS



Fonte: autoria própria

Acesse: <https://developer.mozilla.org/en-US/docs/Web/CSS/outline>.

Acesse: <https://css-tricks.com/almanac/properties/o/outline/>.

## 7.8 Display

- **display:** Aplica a formatação de quebra antes ou depois, conforme os valores da propriedade.
  - valores possíveis:
    - \* **inline** : elementos exibidos em linha (valor default). Por exemplo: as tags strong, a, em.
    - \* **block** : é inserida uma quebra de linha antes e depois do elemento. Por exemplo: as tags h1, h2, p. São exibidos em blocos.
    - \* **none** : não exibe o elemento com efeitos avançados. Se utilizar a propriedade visibility: hidden; você está indicando ao browser para esconder o elemento.
    - \* **inline-block** : área como bloco, e itens dentro como linha.
    - \* **list-item** : igual a li.
    - \* **flex** : cada conteúdo filho se comporta como se fosse um bloco.
  - \* Outros valores, **acesse:** <https://developer.mozilla.org/en-US/docs/Web/CSS/display>.

## 7.9 Flexbox Layout

- Iniciamos a aplicação deste conceito criando um arquivo com extensão html.

**Acesse:** [https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\\_Flexible\\_Box\\_Layout](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout).

Figura 32 – Arquivo html com o link para o arquivo estilos.css

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>FlexBox</title>
6     <link rel="stylesheet" type="text/css" href="css/estilo.css">
7   </head>
8   <body>
9     <section id="content">
10       <article class="item">01</article>
11       <article class="item">02</article>
12       <article class="item">03</article>
13       <article class="item">04</article>
14     </section>
15   </body>
16 </html>
```

Fonte: autoria própria

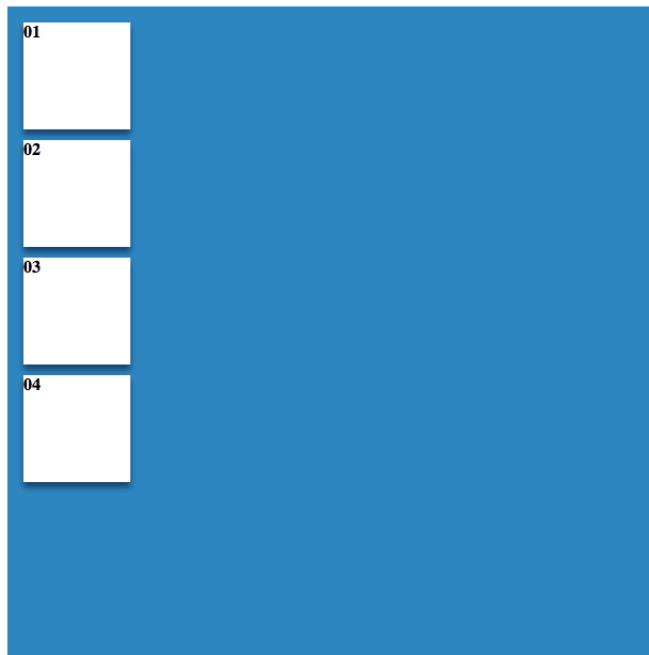
Figura 33 – Arquivo css com formatação inicial

```

1 #content {
2   background-color: #2E86C1; /* propriedade de cor de fundo */
3   height: 600px;           /* propriedade de altura do conteúdo */
4   width: 600px;            /* propriedade de largura do conteúdo */
5   padding: 5px;             /* propriedade de espaçamento ao redor do conteúdo */
6 }
7
8 /* propriedades para formatação da classe item */
9 .item {
10   height: 100px;           /* propriedade de altura dos filhos article */
11   width: 100px;            /* propriedade de largura dos filhos article */
12   color: #000;              /* propriedade de cor de texto */
13   font-weight: bold;        /* propriedade de estilo negrito para o texto */
14   box-shadow: 0px 5px 8px rgba(0, 0, 25, 0.5); /* propriedade de sombra para os filhos article */
15   background-color: #fff;    /* propriedade de cor de fundo */
16   margin: 10px;             /* propriedade de margem aplicada a todos os lados do elemento */
17 }
18 }
```

Fonte: autoria própria

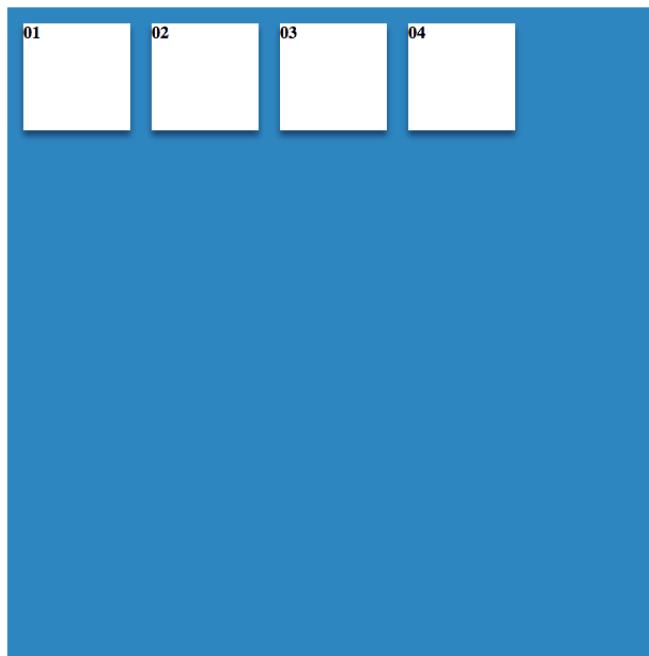
Figura 34 – Visualização no navegador



Fonte: autoria própria

- **display:** flex;
  - Indica que o conteúdo dentro de um **container** se comportará como um bloco, um container.
  - **Exemplo:** No arquivo css, inserir a propriedade **display: flex** no identificador **#content**. Na figura a seguir, é exibido como é visualizado no navegador.

Figura 35 – Visualização no navegador com a propriedade display:flex

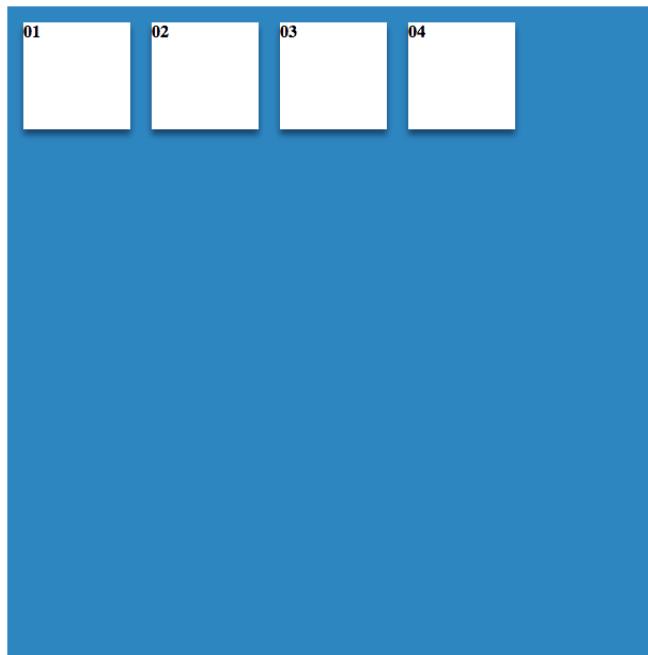


Fonte: autoria própria

- **flex-direction:** formata a direção do posicionamento dos elementos.
  - valores possíveis:

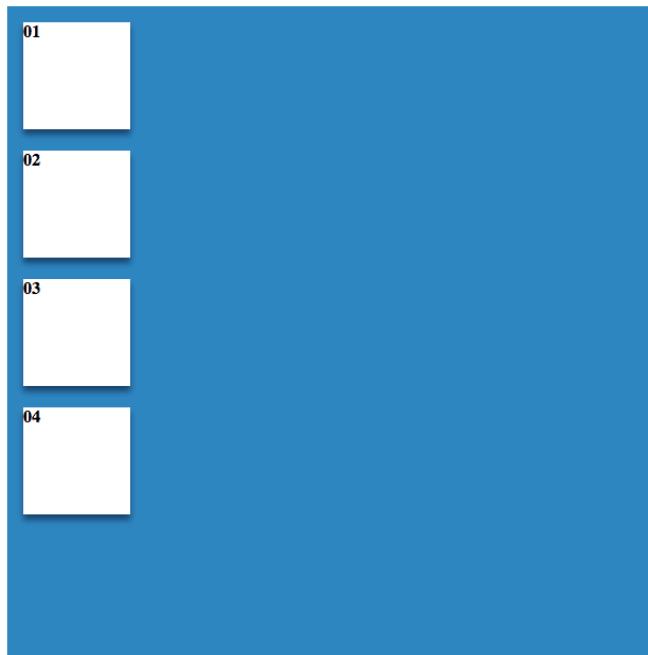
- \* **row**: exibe os itens na direção de linha, da esquerda pra direita. (valor default).
  - \* **column**: exibe os itens na direção de coluna de cima pra baixo.
  - \* **row-reverse**: exibe os itens na direção de linha, da direita pra esquerda.
  - \* **column-reverse**: exibe os itens na direção de coluna de baixo pra cima.
- **Exemplo:** No arquivo css, inserir a propriedade **flex-direction: row** no identificador **#content**. E em seguida, altere para os demais valores.

Figura 36 – Visualização no navegador com a propriedade flex-direction: row



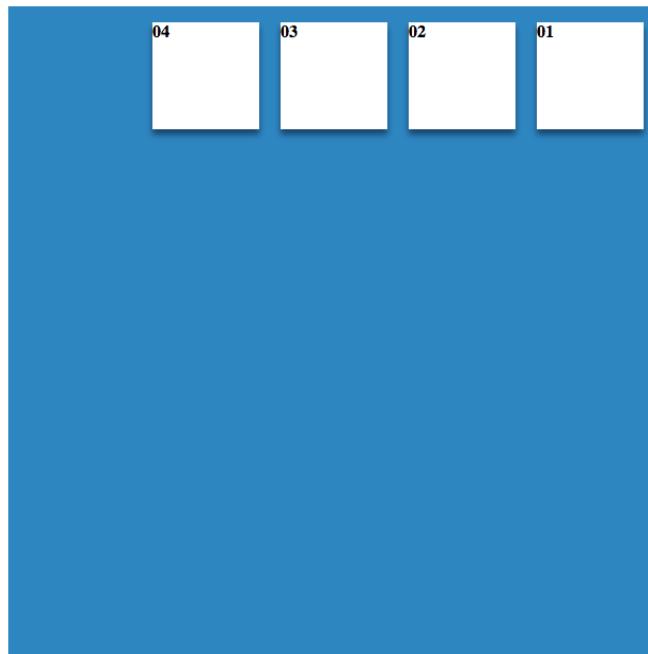
Fonte: autoria própria

Figura 37 – Visualização no navegador com a propriedade flex-direction: column



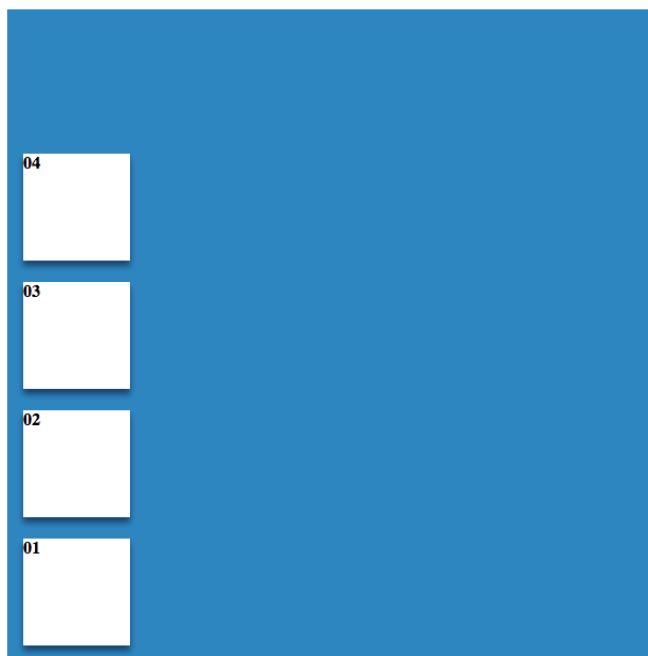
Fonte: autoria própria

Figura 38 – Visualização no navegador com a propriedade flex-direction: row-reverse



Fonte: autoria própria

Figura 39 – Visualização no navegador com a propriedade flex-direction: column-reverse



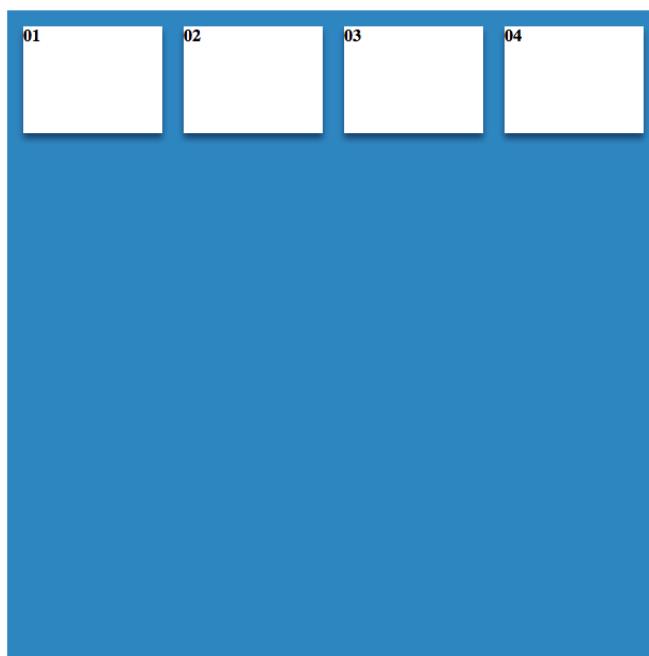
Fonte: autoria própria

- **flex-wrap:** determina se os elementos filhos aceitarão ou não uma quebra de conteúdo.
  - valores possíveis:
    - \* wrap: permitem a quebra de conteúdo filhos.
    - \* nowrap: não permitem a quebra de conteúdo filhos (valor default).
    - \* wrap-reverse: efetuam a quebra de conteúdo filhos e são exibidos de forma inversa (de baixo para cima).
  - Ou usar: **flex-flow: <'flex-direction'> <'flex-wrap'>**

- **Exemplo1:** No arquivo css, inserir a propriedade `flex-wrap: nowrap` no identificador `#content`. Nesta situação, a visualização será idêntica ao que foi apresentado com a propriedade `flex-direction: row`. Pois os itens filhos são ajustados dentro do espaço para não quebrarem.
- **Exemplo2:** No arquivo css, altere a propriedade de largura (`width: 100px`) na classe `.item`, para `width: 250px`. Os itens filhos são redimensionados para caberem dentro do espaço definido. Agora, altere a propriedade `flex-wrap: nowrap` no identificador `#content` para `flex-wrap: wrap`.
- **Exemplo3:** E em seguida, altere a propriedade `flex-wrap` para os demais valores.
- **Exemplo4:** Altere `flex-direction: row` para a propriedade `flex-direction: column-reverse`. E a propriedade `flex-wrap: wrap-reverse`.

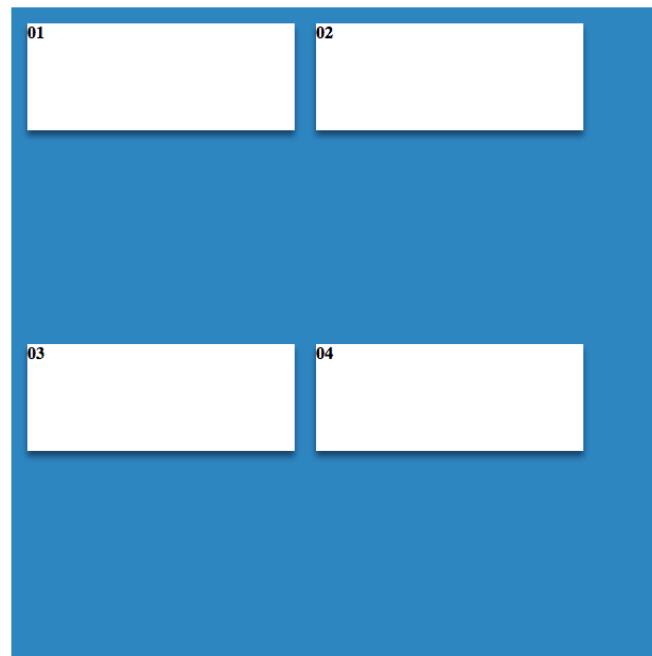
**Acesse:** <https://css-tricks.com/almanac/properties/f/flex-flow/>.

Figura 40 – Exemplo2 - Visualização no navegador com a propriedade `flex-wrap: nowrap`



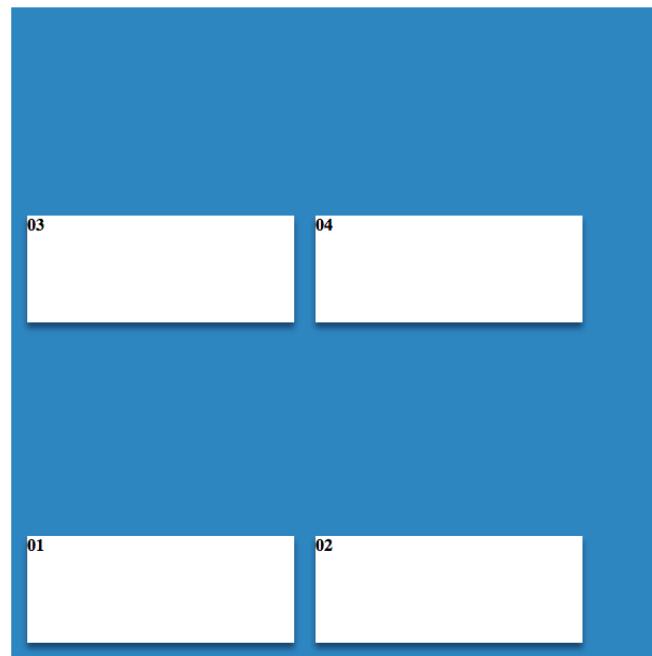
Fonte: autoria própria

Figura 41 – Exemplo3 - Visualização no navegador com a propriedade flex-wrap: wrap



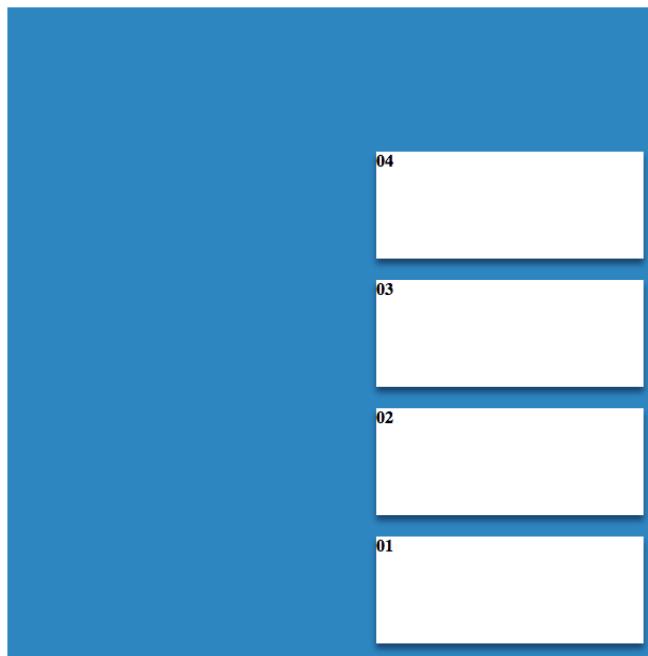
Fonte: autoria própria

Figura 42 – Exemplo4 - Visualização no navegador com a propriedade flex-wrap: wrap-reverse



Fonte: autoria própria

Figura 43 – Exemplo4 - Visualização no navegador com a propriedade flex-direction: column-reverse e flex-wrap: wrap-reverse



Fonte: autoria própria

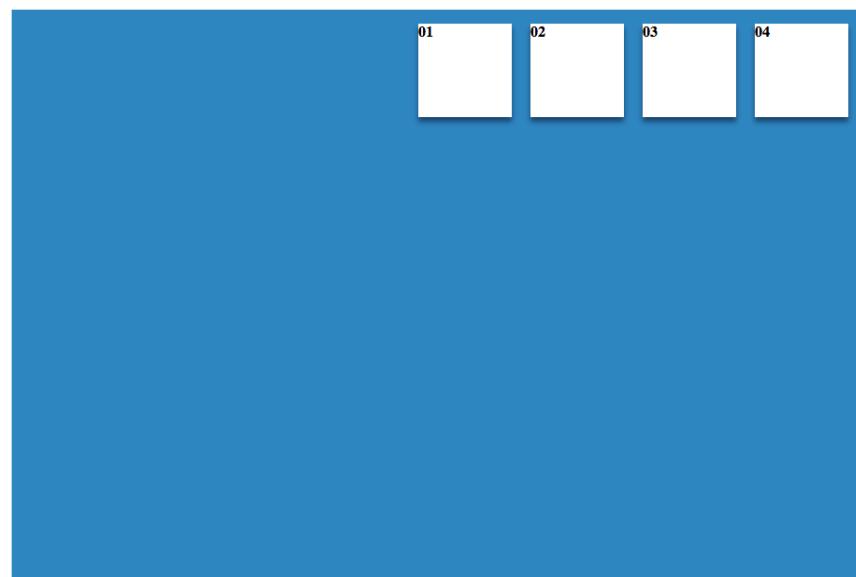
- **justify-content:** determina o posicionamento dos elementos na página.
  - valores possíveis:
    - \* flex-start: todos os elementos filhos são alinhados à esquerda, no topo (valor default).
    - \* flex-end: todos os elementos filhos são alinhados à direita, sem inversão de ordem.
    - \* space-around: os elementos filhos são distribuídos ao redor do elemento e de maneira uniforme dentro do espaço do container.
    - \* space-between: os elementos filhos são distribuídos a partir do centro do container para as extremidades. Sem inserir margem na lateral.
  - **Exemplo1:** No arquivo css:
    - \* Altere a propriedade de largura (`width: 250px`) na classe `.item`, para `width: 100px;`
    - \* E a propriedade de largura (`width: 600px`) no identificador `#content`, para `width: 900px;`
    - \* E a propriedade `flex-direction` para `row`;
    - \* E a propriedade `flex-wrap` para `wrap`;
    - \* E acrescente `justify-content: flex-start`;
  - **Exemplo2:** No arquivo css, altere a propriedade `justify-content: flex-start` para `justify-content: flex-end`;
  - **Exemplo3:** No arquivo css, altere a propriedade `justify-content: flex-end` para `justify-content: space-around`;
  - **Exemplo4:** No arquivo css, altere a propriedade `justify-content: space-around` para `justify-content: space-between`;

Figura 44 – Exemplo1 - Visualização no navegador com a propriedade justify-content: flex-start



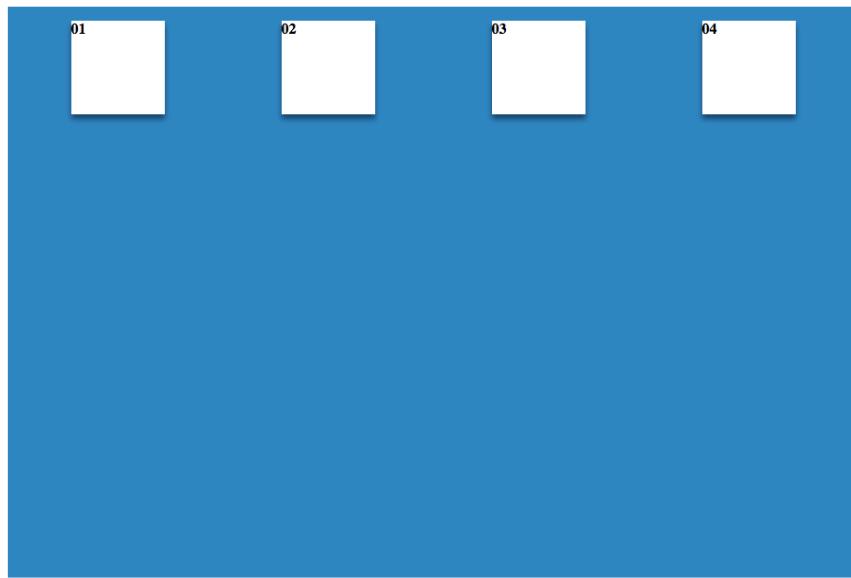
Fonte: autoria própria

Figura 45 – Exemplo2 - Visualização no navegador com a propriedade justify-content: flex-end



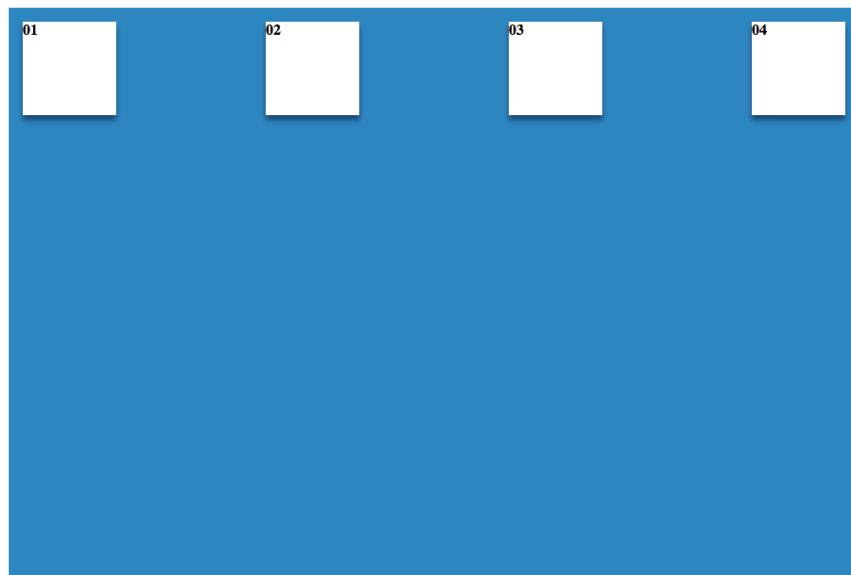
Fonte: autoria própria

Figura 46 – Exemplo3 - Visualização no navegador com a propriedade justify-content: space-around



Fonte: autoria própria

Figura 47 – Exemplo4 - Visualização no navegador com a propriedade justify-content: space-between



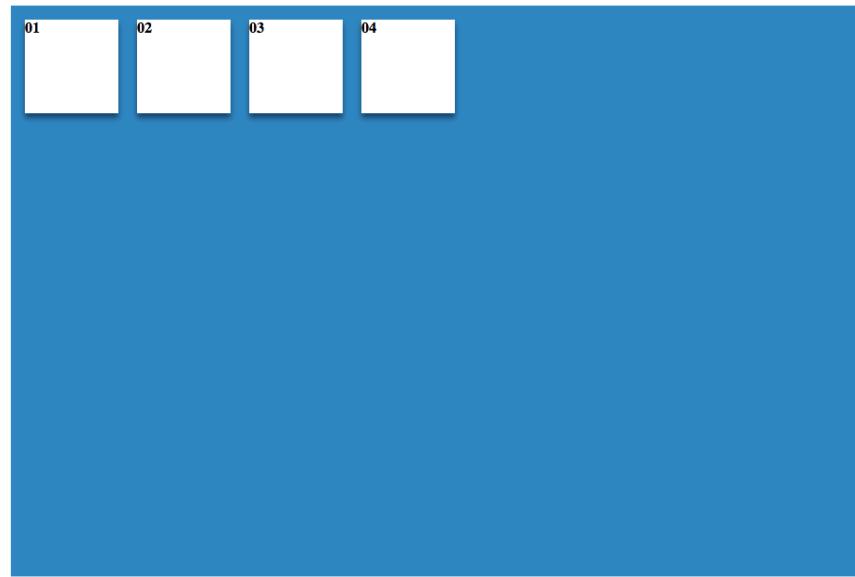
Fonte: autoria própria

**Acesse:** <https://css-tricks.com/almanac/properties/j/justify-content/>.

- **align-items:** determina o posicionamento vertical dos elementos na página.
  - valores possíveis:
    - \* **flex-start:** exibe os elementos do container do topo para baixo, da esquerda para direita (valor default).
    - \* **flex-end:** exibe os elementos do container de baixo pra cima, da esquerda para direita.
    - \* **center:** centraliza todos os elementos.
    - \* **stretch:** exibe os elementos do topo para baixo, caso não tenha sido definido a altura dos elementos filhos do container, os mesmos são esticados até abaixo. Para visualizar, remova a propriedade **height: 100px** da classe **.item**.

- \* **baseline**: é um alinhamento sensível ao conteúdo.

Figura 48 – Visualização no navegador com a propriedade align-items: flex-start



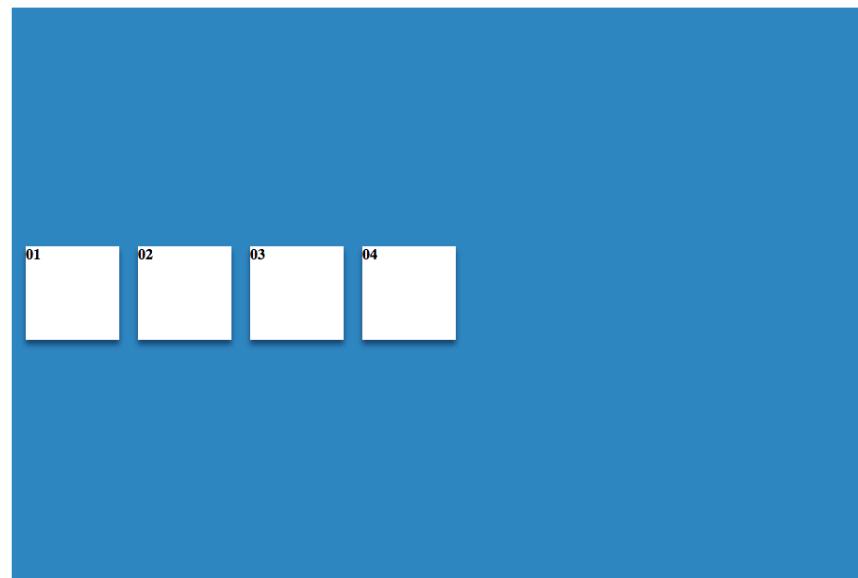
Fonte: autoria própria

Figura 49 – Visualização no navegador com a propriedade align-items: flex-end



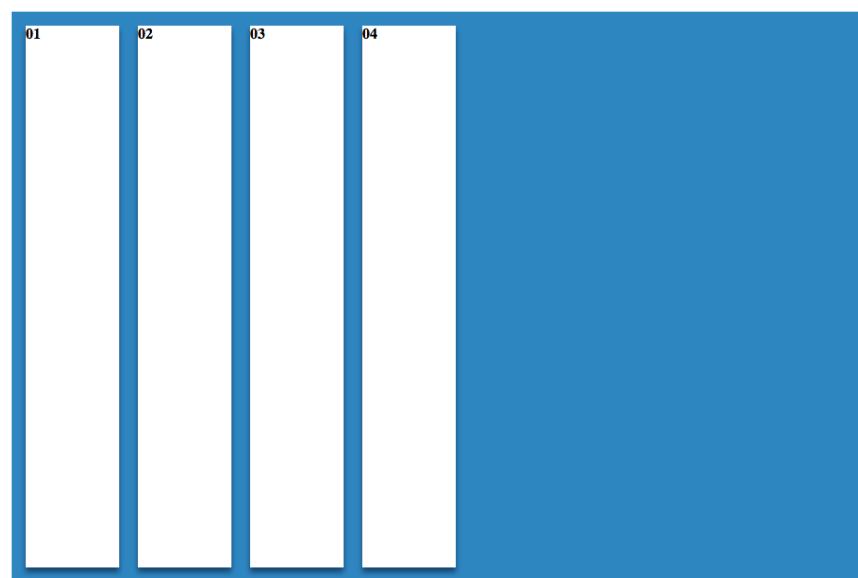
Fonte: autoria própria

Figura 50 – Visualização no navegador com a propriedade align-items: center



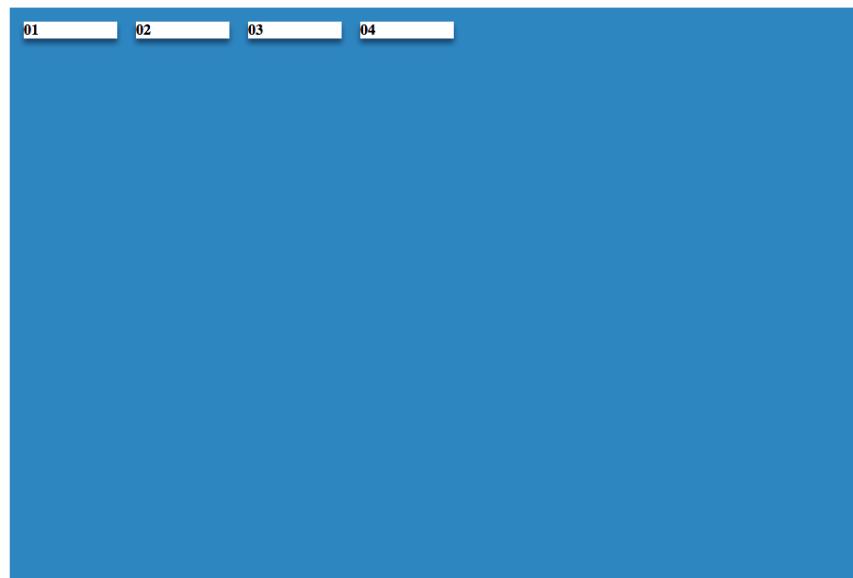
Fonte: autoria própria

Figura 51 – Visualização no navegador com a propriedade align-items: stretch



Fonte: autoria própria

Figura 52 – Visualização no navegador com a propriedade align-items: baseline



Fonte: autoria própria

**Acesse:** <https://css-tricks.com/almanac/properties/a/align-items/>.

- Existem outras propriedades: align-content.

**Acesse:** <https://css-tricks.com/almanac/properties/a/align-content/>.

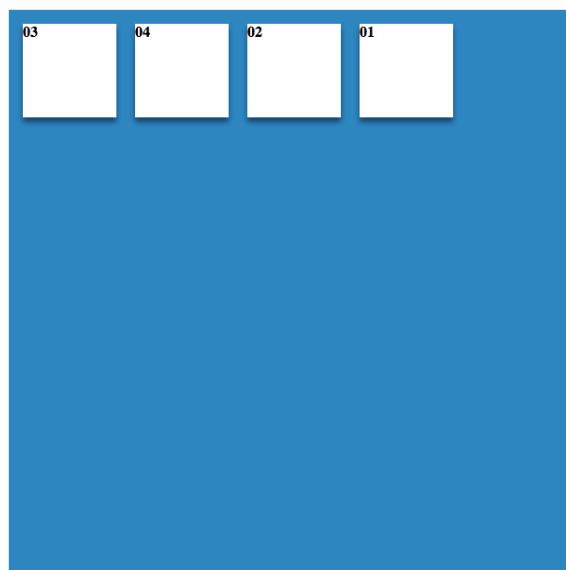
- **order:** determina a ordem em que os elementos serão exibidos na página.

– valores possíveis: valor inteiro.

– **Exemplo** : definindo uma pseudo-classe para os itens filhos. Teremos:

```
* .item:nth-child(1) { order: 4; }
* .item:nth-child(2) { order: 3; }
* .item:nth-child(3) { order: 1; }
* .item:nth-child(4) { order: 2; }
```

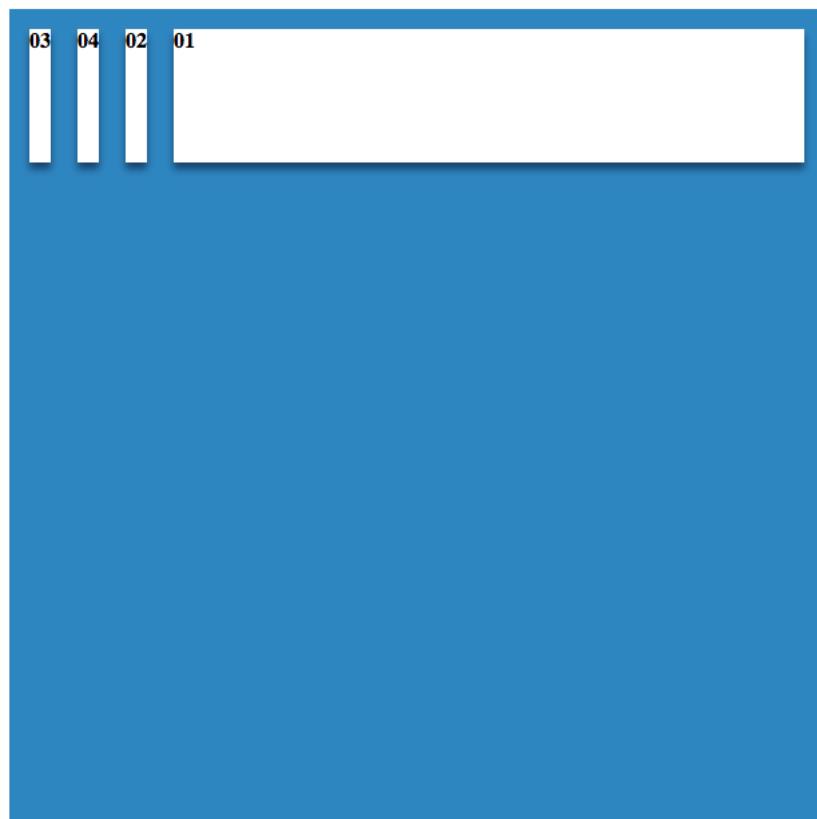
Figura 53 – Visualização no navegador com a pseudo-classe :nth-child e a propriedade order



Fonte: autoria própria

- **flex-grow:** define a propriedade de crescer o elemento na página.
  - valores possíveis: valor inteiro.
  - O valor default é 0.
  - Não é possível valores negativos.
  - Se um item filho tiver o flex-grow: 2, ele ocupará duas vezes mais o espaço previsto para os outros elementos. E se o width dos elementos estiver definido, ele ocupará o espaço restante.
  - **Exemplo :** na classe `.item`, definindo os valores default para as 3 propriedades e, removendo `width` da classe `.item` e
    - \* `flex-grow: 0; flex-shrink: 1; flex-basis: auto;`
    - \* Definindo as pseudo-classes:
      - `.item:nth-child(1) { order: 4; flex-grow: 2; }`
      - `.item:nth-child(2) { order: 3; }`
      - `.item:nth-child(3) { order: 1; }`
      - `.item:nth-child(4) { order: 2; }`

Figura 54 – Visualização no navegador com a propriedade flex-grow: 2; e width: 0px no filho

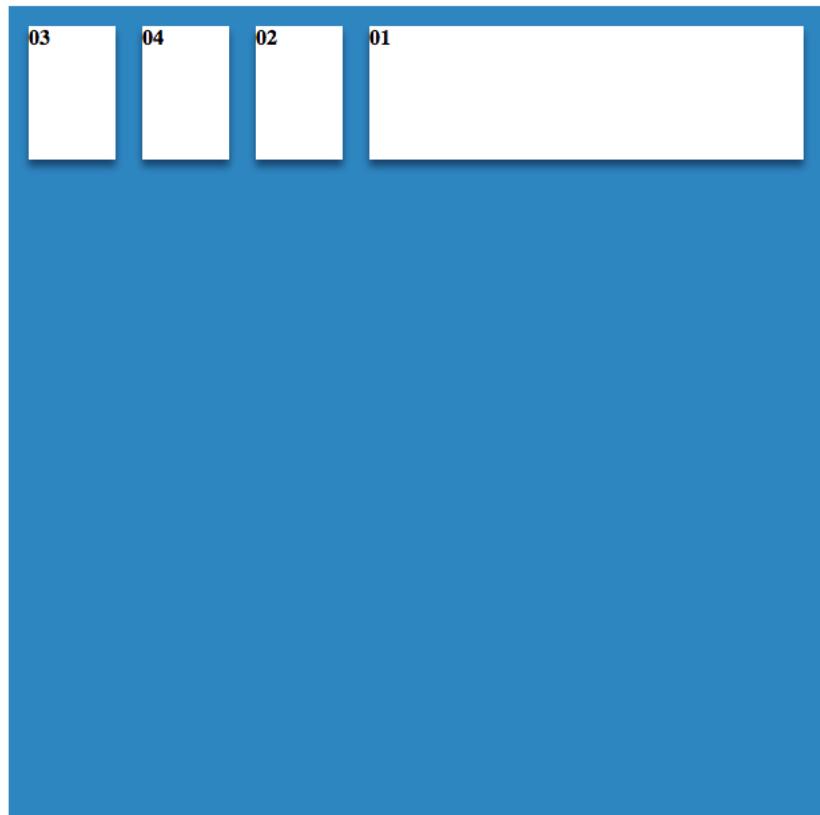


Fonte: autoria própria

- **flex-shrink:** define a propriedade de encolher o elemento na página. Somente como informativo.
  - valores possíveis: valor inteiro.
  - O valor default é 1.
  - Não é possível valores negativos.
- **flex-basis:** define o width dos elementos antes de distribuir o restante para o flex-grow.

- valores possíveis: valor em pixel.
- O valor default é auto.
- Se definir 0, os elementos não serão redimensionados.
- **Exemplo:** `flex-basis: 65px;` os elementos serão definidos nesta largura e depois o que restar é usado nos elementos que foram configurados com flex-grow.

Figura 55 – Visualização no navegador com a propriedade flex-basis: 65px; na classe .item



Fonte: autoria própria

- **flex:** define os três valores ao mesmo tempo.
  - `flex: flex-grow flex-shrink flex-basis;`
  - **Exemplo:** `flex: 0 1 auto;` (valores default de cada propriedade)

*Especificação da W3C (atualizada) - Acesse:*

<https://www.w3.org/TR/2017/WD-css-align-3-20170215/>.

*Especificação da W3C - Acesse:* <https://www.w3.org/TR/css-flexbox-1/>.

*Demos online para praticar Flexbox - Acesse:*

<https://demos.scotch.io/visual-guide-to-css3-flexbox-fbox-playground/demos/>.

*Tutorial da propriedade Flexbox - Acesse:*

<https://scotch.io/tutorials/a-visual-guide-to-css3-flexbox-properties>.

*Compatibilidade dos navegadores - Acesse:* <http://caniuse.com/#search=flex-box>.

*Guia para usar Flexbox - Acesse:* <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>.

*Polyfill (compatibilidade para todos os navegadores) - Acesse:* <https://autoprefixer.github.io>.

## 7.10 Propriedades de formatação de background

- Tags para inserir imagens em seu código HTML:

- <figure> </figure> O elemento figura representa um conteúdo de fluxo, opcionalmente com um subtítulo.
- <figcaption> </figcaption> Representa a legenda do Figure.
- <img> Representa uma imagem.

**Acesse:** [https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5\\_element\\_list](https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5_element_list).

**Acesse exemplos de padrões:** <http://lea.verou.me/css3patterns/>.

- Permite formatar o fundo dos elementos, das caixas:

- *background-color*.....define a cor do fundo;
- *background-image*.....define uma imagem de fundo;
- *background-repeat*.....define a maneira como a imagem de fundo é posicionada;
- *background-attachment*....define se a imagem de fundo "rola" ou não com a tela;
- *background-position*.....define como e onde a imagem de fundo é posicionada;
- *background-clip*.....define a área do box onde a imagem de fundo é aplicada;
- *background-origin*.....define a posição de origem da imagem no box;
- *background-size*.....define as dimensões da imagem no box; ou
- *background*.....maneira abreviada para declarar todas as propriedades anteriores
- valores possíveis:

\* ***background-color*:**

- código hexadecimal: #ffc6d9.
- código rgb: rgb(255,235,0).
- código rgba: rgb(255,235,0, 0.7).
- código hsl: hsl(210,100%,40%).
- código hsla: hsla(155,80%,35%,0.4).
- palavra-chave: red, blue, green...etc.
- transparente: transparent.

\* ***background-image*:**

- URL: url(caminho/imagem.gif).
- URL: url('caminho/imagem.gif').
- URL: url("caminho/imagem.gif").
- Gradiente linear: linear-gradient(45deg, blue, red).
- Gradiente radial: radial-gradient(20px 50px, green, blue, red).

\* ***background-repeat*:**

- imagem não repete: no-repeat.
- imagem repete na vertical e horizontal: repeat.
- imagem repete na vertical: repeat-y.
- imagem repete na horizontal: repeat-x.
- imagem toca as quatro bordas internas do box e são espaçadas de modo a se distribuirem igualmente: space.

- imagem toca as quatro bordas internas do box e são redimensionadas de modo a preencherem o fundo tocando umas nas outras: round.

\* **background-attachment:**

- fixed: imagem permanece fixa em relação à viewport (janela do navegador) e não rola juntamente com o conteúdo.
- scroll: imagem não permanece fixa em relação à viewport (janela do navegador) e rola juntamente com o conteúdo - este é o valor padrão.
- local: imagem permanece fixa em relação à viewport (janela do navegador) mas quando aplicada ao box de um elemento rola juntamente com o conteúdo se o elemento tiver um mecanismo de rolagem definido por overflow: scroll.

\* **background-position:**

- Xpx Ypx
- X% Y%
- top left
- top center
- top right
- center left
- center center
- center right
- bottom left
- bottom center
- bottom right
- top
- right
- bottom
- left

\* **background-clip:**

- border-box: a imagem ocupa até a área das bordas do box, inclusive, se houver uma - este é o comportamento padrão.
- padding-box: a imagem ocupa até a área de padding do box, inclusive, se houver uma e não ocupa a área de bordas.
- content-box: a imagem ocupa até a área de conteúdo do boxe não ocupa as áreas de padding e bordas se houver.

\* **background-origin:**

- border-box: a imagem tem por origem o canto superior esquerdo do box considerada das bordas do box, se houver uma - este é o comportamento padrão.
- padding-box: a imagem tem por origem o canto superior esquerdo do box sem considerar as bordas do box, se houver uma.
- content-box: a imagem tem por origem o canto superior esquerdo do box sem considerar o padding e as bordas do box, se houver.

\* **background-size:**

- 150px 70px: a imagem terá as dimensões definidas por unidade de medida CSS.
- auto: a imagem terá suas dimensões originais - este é o valor padrão.
- contain: a imagem terá as dimensões de modo a que sua maior dimensão ocupe toda a extensão do box e a menor dimensão seja tal que o aspect ratio (relação entre largura e altura) seja preservado.

- cover: a imagem terá as dimensões de modo a que sua menor dimensão ocupe toda a extensão do box e a maior dimensão seja tal que o aspect ratio (relação entre largura e altura) seja preservado.

*Acesse:* <https://css-tricks.com/almanac/properties/b/background/>.

# 8 Tags de Listas, Tabelas

- Para criarmos listas de itens no HTML, utilizamos as seguintes tags:

## 8.1 Tags de Listas ordenadas e desordenadas

- <ol> </ol> Define uma lista ordenada de itens, que é uma lista que mudam o seu significado, se alterar a ordem dos seus elementos.
- <ul> </ul> Define uma lista não ordenada.
- <li> </li> Define um item de uma lista ol ou de uma ul.

## 8.2 Tags de Listas de Definição

- <dl> Representa uma Lista de Definição.
- <dt> Representa um item da Lista de Definição <dd>.
- <dd> Representa a definição dos termos listados imediatamente antes dele.

*Acesse:* [https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5\\_element\\_list](https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5_element_list).

*Especificação W3C em andamento - Acesse:*  
<https://www.w3.org/TR/2017/WD-css-tables-3-20170307/>.

### 8.2.1 Propriedades de formatação de listas ordenadas e desordenadas

- Propriedades a serem aplicadas em tags de listas:
  - *list-style-type: valor;*
  - *list-style-image: valor;*
  - *list-style-position: valor; ou*
  - *list-style: list-style-type list-style-image list-style-position;*
  - valores possíveis:
    - \* *list-style-type:* circle, square, decimal, decimal-leading-zero, upper-roman, lower-roman, upper-alpha, lower-alpha, lower-greek ou none.
    - \* *list-style-image:* url(caminho da imagem).
    - \* *list-style-position:* inside (alinhado a esquerda), outside (indentado).

## 8.3 Tags de Tabelas

- Para criarmos uma tabela no HTML, utilizamos as seguintes tags:
  - <**thead**> </**thead**> Representa o cabeçalho da tabela. Neste item, inserimos as tags <th>, <tr> combinadas.
  - <**tbody**> </**tbody**> Representa o corpo da tabela. Neste item, inserimos as tags pertinentes para criar o corpo da tabela <tr>, <td>.
  - <**tfoot**> </**tfoot**> Representa o rodapé da tabela. Neste item, inserimos as tags <tr> e <td> combinadas para indicar informações de rodapé.

- **<tr> </tr>** Representa as linhas de uma tabela.
- **<th> </th>** Representa os títulos das colunas.
- **<td> </td>** Representa os detalhes de cada linha, inseridos dentro de cada **<tr>**.
- O elemento HTML **<col> </col>** define uma coluna dentro de uma tabela e é usado para definir a semântica comum em todas as células comuns. Geralmente é encontrado dentro de um elemento **<colgroup>**.
- O elemento HTML **<colgroup> </colgroup>** define um grupo de colunas dentro de uma tabela.
- Quando o elemento **<table>** que é o pai deste **<caption>** é o único descendente de um elemento **<figure>**, use o elemento **<figcaption>** em vez disso.

### 8.3.1 Propriedades de formatação de tabelas

- Para alinhar a legenda utilize `text-align`, `caption-side`.
- Propriedades CSS:
  - **`width:`** controla a largura das colunas.
  - pseudoclasse **:nth-child** configura o alinhamento das células de cada coluna.
  - **`text-align:`** alinha o conteúdo horizontalmente nas células.
  - **`vertical-align:`** alinha o conteúdo verticalmente nas células.
  - **`border-collapse:`**
    - \* A propriedade CSS `border-collapse` determina se as bordas de uma tabela são separadas ou recolhidas.
    - \* No modelo `separate`, as células adjacentes possuem suas próprias bordas distintas. No modelo `collapse`, as células de tabela adjacentes partilham fronteiras. O modelo `separate` é o tradicional modelo de borda de tabela HTML. Células adjacentes cada um tem suas próprias fronteiras distintas. A distância entre eles dada pela propriedade `border-spacing`.
    - \* No modelo de borda `collapse`, as células de tabela adjacentes partilham fronteiras.
    - \* Valores:
      - **`separate`:** É uma palavra-chave solicitando o uso do modelo de renderização de tabela de borda separada. É o valor padrão.
      - **`collapse`:** É uma palavra-chave que solicita a utilização do modelo de renderização de tabela de borda recolhida.
  - **`border-spacing:`**
    - \* A propriedade CSS de espaçamento de bordas especifica a distância entre as bordas de células de tabela adjacentes (apenas para o modelo de bordas separadas). Isso é equivalente ao atributo `cellspacing` no HTML de apresentação, mas um segundo valor opcional pode ser usado para definir diferentes espaçamentos horizontal e vertical.
    - \* O valor de espaçamento entre bordas também é usado ao longo da borda externa da tabela, onde a distância entre a borda da tabela e as células na primeira / última coluna ou linha é a soma do espaçamento entre bordas relevante (horizontal ou vertical) e Relevante (superior, direita, inferior ou esquerda) na tabela.
    - \* Esta propriedade aplica-se somente quando **`border-collapse` é `separate`**.
  - valores possíveis para **`border-spacing:`**
    - \* **`border-spacing:`** `length`.
      - *Exemplo:* `border-spacing: 2px;`

- \* **border-spacing:** horizontal vertical.
  - *Exemplo:* border-spacing: 1cm 2em;
  - \* **border-spacing:** inherit;
- **caption-side:**
  - \* A propriedade CSS do lado da legenda posiciona o conteúdo do <caption> de uma tabela no lado especificado.
  - \* valores possíveis para **caption-side:**
    - Directional values. *Exemplo:* caption-side: top; caption-side: bottom;
    - Warning: non-standard values. *Exemplo:* caption-side: left; caption-side: right; caption-side: top-outside; caption-side: bottom-outside;
    - Global values. *Exemplo:* caption-side: inherit; caption-side: initial; caption-side: unset;
- **empty-cells:**
  - \* A propriedade CSS especifica como os agentes de usuário devem renderizar bordas e fundos em torno de células sem conteúdo visível.
  - \* valores possíveis para **empty-cells:**
    - **show:** É uma palavra-chave que indica que as bordas e fundos devem ser desenhados como em células normais.
    - **hide:** É uma palavra-chave que indica que não há bordas ou fundos a serem desenhados.
- **table-layout:**
  - \* valores possíveis para **table-layout:**
    - **auto:** um algoritmo de layout de tabela automático é comumente usado pela maioria dos navegadores para layout de tabela. As larguras da tabela e suas células dependem do seu conteúdo.
    - **fixed:** as larguras da tabela e da coluna são definidas pelas larguras dos elementos <table> e <col> ou pela largura da primeira linha de células. Células em linhas subsequentes não afetam larguras de coluna.
- **vertical-align:**
  - \* A propriedade CSS de alinhamento vertical especifica o alinhamento vertical de uma linha.
  - \* valores possíveis para **vertical-align:**
    - **baseline:** Alinha a linha de base do elemento com a linha de base de seu pai.
    - **sub:** Alinha a linha de base do elemento com a linha de base do subscrito de seu pai.
    - **super:** Alinha a linha de base do elemento com a linha de base do superíndice de seu pai.
    - **text-top:** Alinha a parte superior do elemento com a parte superior da fonte do elemento pai.
    - **text-bottom:** Alinha a parte inferior do elemento com a parte inferior da fonte do elemento pai.
    - **middle:** Alinha o meio do elemento com a linha de base mais metade da altura x do pai.

**Acesse:** <https://developer.mozilla.org/pt-BR/docs/Web/CSS/table-layout>.

**Artigo - O Futuro do Estilo:** <https://www.w3.org/Style/CSS/Planet/>.

# 9 Manipulando Formulários

## 9.1 Tags de Formulários

- Na elaboração de um formulário podemos utilizar diversas tags do HTML para estruturar o formulário. Dentre elas estão, as tags de títulos `<h1>`, `<h2>`, ..., `<h6>`, `<div>`, `<section>`, `<p>`

### 9.1.1 Form

- Cada vez que você deseja criar um formulário HTML, você deve iniciá-lo usando esse elemento, colocando todo o conteúdo dentro.
- Muitas tecnologias assistivas ou plugins do navegador podem usar o `<form>` e implementar atalhos especiais para torná-los mais fáceis de usar.

- *Sintaxe:*

- `<form> ... </form>`

- *Exemplo:*

Figura 56 – Exemplo de um formulário - Parte I

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Formulários</title>
6  </head>
7  <body>
8      <form>
9          <h1>Payment form</h1>
10         <p>Required fields are followed by <strong><abbr title="required">*</abbr></strong>. </p>
11         <section>
12             <h2>Contact information</h2>
13             <fieldset>
14                 <legend>Title</legend>
15                 <ul>
16                     <li>
17                         <label for="title_1">
18                             <input type="radio" id="title_1" name="title" value="M." >
19                             Mister
20                         </label>
21                     </li>
22                     <li>
23                         <label for="title_2">
24                             <input type="radio" id="title_2" name="title" value="Ms.">
25                             Miss
26                         </label>
27                     </li>
28                 </ul>
29             </fieldset>
30             <p>
31                 <label for="name">
32                     <span>Name: </span>
33                     <strong><abbr title="required">*</abbr></strong>
34                 </label>
35                 <input type="text" id="name" name="username">
36             </p>

```

Fonte:

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

Figura 57 – Exemplo de um formulário - Parte II

```

37      <p>
38          <label for="mail">
39              <span>E-mail: </span>
40              <strong><abbr title="required">*</abbr></strong>
41          </label>
42          <input type="email" id="mail" name="usermail">
43      </p>
44      <p>
45          <label for="password">
46              <span>Password: </span>
47              <strong><abbr title="required">*</abbr></strong>
48          </label>
49          <input type="password" id="pwd" name="password">
50      </p>
51  </section>
52  <section>
53      <h2>Payment information</h2>
54      <p>
55          <label for="card">
56              <span>Card type:</span>
57          </label>
58          <select id="card" name="usercard">
59              <option value="visa">Visa</option>
60              <option value="mc">Mastercard</option>
61              <option value="amex">American Express</option>
62          </select>
63      </p>
64      <p>
65          <label for="number">
66              <span>Card number:</span>
67              <strong><abbr title="required">*</abbr></strong>
68          </label>
69          <input type="text" id="number" name="cardnumber">
70      </p>

```

Fonte:

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

Figura 58 – Exemplo de um formulário - Parte III

```

71      <p>
72          <label for="date">
73              <span>Expiration date:</span>
74              <strong><abbr title="required">*</abbr></strong>
75              <em>formatted as mm/yy</em>
76          </label>
77          <input type="text" id="date" name="expiration">
78      </p>
79  </section>
80  <section>
81      <p> <button type="submit">Validate the payment</button> </p>
82  </section>
83 </form>
84 </body>
85 </html>

```

Fonte:

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

Figura 59 – Visualização do Exemplo de um formulário

## Payment form

Required fields are followed by \*.

### Contact information

Title

Mister  
 Miss

Name: \*

E-mail: \*

Password: \*

### Payment information

Card type:  

Card number: \*

Expiration date: \* *formatted as mm/yy*

Fonte:

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

**Acesse:**

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form).

#### 9.1.2 Fieldset e Legend

- É uma maneira de criar grupos que compartilham o mesmo propósito, para fins de estilo e semântica.
- Você pode rotular um `<fieldset>`, incluindo um `<legend>` logo abaixo da tag de abertura `<fieldset>`.
- O conteúdo textual da `<legend>` formalmente descreve a finalidade da `<fieldset>`. É incluído dentro.
- **Sintaxe:**
  - `<fieldset> <legend> Legenda do Fieldset </legend> ....demais conteúdos ... </fieldset>`

Figura 60 – Exemplo do uso de &lt;form&gt;, &lt;fieldset&gt;, &lt;legend&gt;

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Formulários</title>
</head>
<body>
    <form>
        <fieldset>
            <legend>Legenda do Fieldset</legend>
            <p>
                <input type="radio" name="size" id="size_1" value="small">
                <label for="size_1">Item1</label>
            </p>
            <p>
                <input type="radio" name="size" id="size_2" value="medium">
                <label for="size_2">Item2</label>
            </p>
            <p>
                <input type="radio" name="size" id="size_3" value="large">
                <label for="size_3">Item3</label>
            </p>
        </fieldset>
    </form>
</body>
</html>
```

Fonte: adaptado

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

Figura 61 – Visualização do Exemplo do uso de &lt;form&gt;, &lt;fieldset&gt;, &lt;legend&gt;



Fonte: adaptado

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

- **Sugestão:** sempre que tiver botões de radio, descreva-os dentro de um fieldset.

**Acesse:** <https://developer.mozilla.org/pt-BR/docs/Web/Guide/HTML/Forms>.

**Acesse:**

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form).

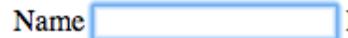
### 9.1.3 Label

- É uma maneira de definirmos um rótulo para os elementos de um formulário.
- Elemento importante para criar formulários acessíveis (quando implementados adequadamente, os leitores de tela falam o rótulo de um elemento do formulário juntamente com quaisquer instruções relacionadas).
- O atributo for da tag <label> serve para relacionar uma tag <label> com uma tag <input> pelo seu respectivo identificador ID, assim o texto do label clicável, dá foco no input correspondente.
- **Sintaxe:**

- <label for="xxx"> Rótulo </label>

Figura 62 – Exemplo do uso de <label> e visualização no navegador

```
<label for="name">Name:</label>
<input type="text" id="name" name="user_name">
```



Fonte: adaptado

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How\\_to\\_structure\\_an\\_HTML\\_form](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/How_to_structure_an_HTML_form)

- Formatando o outline com CSS

Figura 63 – Exemplo do uso de <label> com a propriedade outline formatada no CSS

```
input {
    outline: #00FF00 solid 1px;
```



Fonte: autoria própria

#### 9.1.4 Input

- Os campos de textos de um formulário são inseridos por meio da tag <input>. Eles são uma maneira muito conveniente para permitir que o usuário insira qualquer tipo de dados.
- Atributos de formulários:
  - **type** (é um atributo obrigatório para a tag <input> e define o tipo de entrada de texto que está sendo usada). O valor default é type="text".

Figura 64 – Exemplo de input do tipo text e dos atributos required, max, min, step, pattern, placeholder

```
<body>
  <h1>Criando Formulários</h1>
  <form>
    <p> <!-- atributo required -->
      <label for="nome">Name</label>
      <input type="text" name="nome" id="nome" required placeholder="Nome Completo">
      <label for="idade">Idade</label>
      <input type="number" name="idade" id="idade">
    </p>
    <p> <!-- atributos max e min / required -->
      <label for="dinner">Jantar</label>
      <input type="time" min="17:00" max="22:00" name="dinner" id="dinner" required>
    </p>
    <p> <!-- atributo step max, min -->
      <label for="price">Preço</label>
      <input type="number" min="5" max="100" step="5" name="cost" id="price" required>
    </p>
    <p> <!-- atributos pattern -->
      <label for="credit">Cartão Crédito</label>
      <input type="text" id="credit" name="card" pattern="[0-9]{13-16}" title="de 13 a 16 dígitos do cartão" placeholder="Cartão Crédito">
    </p>
    <p>
      <label for="phonenum">Phone Number:</label>
      <input type="tel" pattern="^\d{2}-\d{4}-\d{4}$" placeholder="99-9999-9999">
    </p>
    <input type="submit" name="submit">
  </form>
</body>
```

Fonte: autoria própria

- **required** (usado para marcar um campo do formulário como obrigatório, ao tentar enviar o formulário, os navegadores farão a verificação e exibirão uma mensagem e posicionarão o cursor no primeiro campo inválido).

Figura 65 – Exemplo do atributo required

Formulário com campos Name (com valor 'Name') e Idade (vazio). Um tooltip amarelo aparece sobre o campo Idade com o texto 'Preencha este campo.'

Fonte: autoria própria

- **min e max** (ao necessitar definir valores mínimos e máximos para campos de valores. Por exemplo, para os tipos number, range).

Figura 66 – Exemplo do atributo required, max e min

Formulário com campos Name (com valor 'ana'), Idade (vazio), Jantar (com valor '23:00') e botão Enviar. Um tooltip amarelo aparece sobre o campo Idade com o texto 'O valor deve ser 22:00 ou anterior.'

Fonte: autoria própria

- **step** (incluído nos tipos de dados de entrada de data/hora, range, number).

Figura 67 – Exemplo do atributo required, max, min, step

Formulário com campos Name (com valor 'Ana'), Idade (com valor '10'), Jantar (com valor '18:02') e Preço (com valor '15'). O campo Preço está destacado com um recorte azul.

Fonte: autoria própria

- **placeholder** (este é o texto que aparece dentro da caixa de entrada de texto que descreve a finalidade da caixa brevemente).

Figura 68 – Exemplo do atributo placeholder

Formulário com campo Name com placeholder 'Nome Completo'.

Fonte: autoria própria

- **pattern** (é suportado onde quer que o atributo placeholder seja permitido. Contém uma expressão regular de estilo javascript em que o valor de <input> deve satisfazer. Ao inserir este atributo, a boa prática sugere a inserção do atributo title indicando a descrição do padrão). O atributo title é utilizado para indicar ao usuário o que deve ser inserido.

Figura 69 – Exemplo do atributo pattern e title

O formulário mostra dois campos:

- Campo "Credit Card" com placeholder "cartão crédito".
- Campo "Phone Number" com placeholder "99-9999-9999".

O campo "Phone Number" contém o valor "11". A interface mostra uma validação de formato com uma caixa de diálogo amarela que diz: "É preciso que o formato corresponda ao exigido."

Fonte: autoria própria

[Acesse: http://www.html5pattern.com/.](http://www.html5pattern.com/)

- **readonly** (o usuário não pode modificar o valor de entrada) ou,
- **disabled** (o valor de entrada nunca é enviado com o resto dos dados do formulário).
- **size** (o tamanho físico da caixa) e,
- **maxlength** (aplicável nos tipos de dados text, password, url, search, tel, email. O número máximo de caracteres que podem ser inseridos na caixa).
- **multiple** (permite a escolha de mais de uma opção no campo ao ser usado com a tag <select>).
- **form** (utilizado para indicar qual formulário aquele elemento está associado e ao ser enviado o formulário, o elemento com esse atributo é enviado junto com o formulário). Se usar um atributo form igual a vazio, ele não será considerado em nenhum formulário da página.
- **autocomplete** (recurso nativo dos navegadores. Armazena o valor informado pelo usuário, e ao voltar a página, ele é recarregado). Exemplo: autocomplete = "on" ou autocomplete = "off".
- **autofocus** (especifica qual elemento terá o foco direcionado. Somente um único elemento por página poderá ter este atributo).

Figura 70 – Exemplos do uso de alguns atributos listados acima

```
input {
    outline: #00FF00 solid 1px;
}
```



Fonte: autoria própria

#### 9.1.4.1 Tipos de input

- Atributos já existentes de versões anteriores do HTML:
  - **Text:** Exibido como uma caixa de entrada de dados, permite a inserção de uma linha de texto.
  - **Password:** Campo de senha. Exibido uma caixa com símbolos para ocultar a senha digitada.
  - **Radio:** Botão de radio, permite escolher uma única opção entre os itens disponíveis.
  - **Checkbox:** Caixa de seleção, permite escolher mais de uma opção.
  - **Submit:** Botão de envio, remete os dados do formulário. Exibido o conteúdo de value. Se o atributo name for inserido, ao enviar o formulário será enviado o nome e o valor.

- **Reset:** Botão de redefinição. Restaura o formulário com os valores padrão. Não é muito usado devido a experiências desagradáveis.
  - **File:** Tem por finalidade permitir que o usuário carregue, anexe ou interaja com um arquivo local a partir de seu computador ou rede. Atributos permitidos: name, disabled, accept, autofocus, multiple, required, capture.
    - \* O atributo accept de alguns navegadores móveis permite acesso a câmera, microfone de alguns dispositivos.
  - **Hidden:** permite os atributos name, value. O campo com este tipo de dados não é exibido no formulário, utilizado para passar valores para os servidores.
  - **Image:** semelhante ao comportamento do tipo submit e utiliza todos os atributos da tag <img>, especificamente <src>, <alt>.
  - **Button:** Botão, e é usado para manipular eventos.
- Atributos inseridos com o HTML5:
    - **Number:** Campo de texto para inserir um número. Permitido a combinação dos atributos min, max e step. Em alguns dispositivos móveis permite exibir um teclado numérico.
    - **Email:** Campo destinado a inserção do email. O navegador fará a validação básica. Nos dispositivos móveis, o teclado é exibido A-Z, ponto, botão \_123(leva ao teclado modificado) incluindo o caracter @.
    - **Search:** Campo de busca, exibido como uma caixa com cantos arredondados. Em alguns navegadores aparece à direita um botão que pode limpá-lo quando selecionado. Já nos dispositivos móveis, exibe a lupa.
    - **Tel:** Campo utilizado para inserir um número de telefone. Não exige sintaxe ou padrão específico. Poderá inserir um atributo placeholder para indicar o formato de inserção, pattern para exigir o formato e codificar em javascript a validação deste número.
    - **Url:** Campo utilizado para inserir um endereço web. O endereço deve ser inserido completo incluindo o tipo de protocolo. Nos dispositivos móveis, o teclado é exibido A-Z, ponto, barra inclinada e .com.
    - **Range:** É exibido na forma de um controle deslizante. Atributos permitidos min, max, step. Em value, defina o ponto inicial a posicionar a marcação do deslizante.
    - **Color:** permite a seleção de uma cor. Utiliza a paleta de cores do sistema operacional. O valor padrão é #000000.
  - Atributos inseridos com o HTML5 para data e hora:
    - **Date:** Fornece uma data com ano, mês e dia. Alguns navegadores fornecem o suporte ao calendário.
    - **Datetime:** Fornece dois campos: um para data (ano, mês, dia) e outro para hora (hora, minuto, segundo, fração de segundo). Permite usar os atributos min, max, step.
    - **Month:** Fornece a inclusão do mês e do ano, sem o dia do mês e o fuso horário.
    - **Time:** Fornece um mecanismo que coloca a hora em formato de 24h. As horas podem ser maiores ou iguais a zero e menores do que 24h.
    - **Week:** Permite o uso de uma data composta com o número da semana dentro de um ano, sem o mês, hora e dia.

*Acesse:* <http://standardista.com/mobile/>.

*Acesse - Propriedade CSS appearance:*  
<https://css-tricks.com/almanac/properties/a/appearance/>.

### 9.1.5 Textarea

- É um campo de texto de forma livre, sem restrições de quebra de linha. Bloco de texto rolável e de várias linhas.
- O atributo wrap surgiu com o HTML5, especificando os valores soft (default - o texto será enviado sem quebra de linha, caso não tenha sido inseridas) e hard (inclui quebras de linhas explícitas).
- Ao utilizar o atributo wrap como sendo hard, especifique também o atributo cols, obrigatoriamente.
- Utilize o CSS para definir largura e altura. E não mais o row e cols como era nas versões anteriores.

### 9.1.6 Select, option, optgroup

- A tag <select> especifica um menu de opções (seleção).
- Um <select> deve conter um ou mais <option> ou, um ou mais <optgroup> contendo elementos <option>.

### 9.1.7 Datalist e o atributo list

- Para os tipos text, email, url, tipos relacionadas a data, hora e tipos numéricos do elemento <input>, o atributo list aponta para uma lista de valores que o navegador deve oferecer para o usuário, além de permitir a seleção.
- O elemento list recebe como valor o identificador do elemento <datalist> que contém a lista.

### 9.1.8 Button

- O elemento <button> pode ser de 3 tipos: submit, reset, button.
- O valor default é submit.

### 9.1.9 Output

- É um elemento similar ao elemento <span>, porém pode ser usado para controle de formulário. Como novidade no HTML5, ele pode ter os atributos form, name, for e os eventos onchange, onforminput, onformchange além dos manipuladores de eventos universais.
- Não tem o atributo value, sendo o seu valor definido pelo conteúdo inserido entre as tags <output> e </output>.
- Já o atributo for, tem um funcionamento diferente do for para o label. O for quando utilizado com output, recebe como valor uma lista separada de espaços com os identificadores dos outros elementos associados à saída.

### 9.1.10 Meter

- É utilizado como aferidor de alguma medida dentro de um intervalo conhecido.
- É usado para indicar o valor atual em relação aos valores mínimo e máximo.
- Os atributos permitidos com <meter> são min, max, high, low, optimum.
  - O atributo optimum fornece o valor do ponto que marca a posição ótima do aferidor.
  - Os atributos min e max, os respectivos valores mínimos e máximos.

- Os atributos high e low, devem ser o menor valor que pode ser considerado um valor alto e o maior valor que pode ser considerado baixo.
- Sugestão: utilize <meter> quando souber os valores mínimo e máximo e os valores puderem variar.

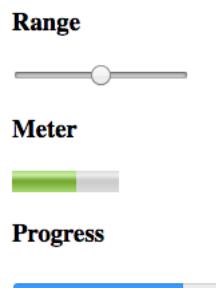
### 9.1.11 Progress

- Semelhante ao <meter>, porém é utilizado para indicar progresso. Indica até que ponto evolui algo.
- Os atributos permitidos são o value, max onde ambos são valores de ponto flutuante positivos. E value menor do que max.

Figura 71 – Exemplos do uso de alguns atributos listados acima

```
<body>
  <h1>Exemplos Range, Meter, Progress</h1>
  <p>
    <h3>Range</h3>
    <label for="seletor"></label>
    <input type="range" id="seletor" name="range" min="0" max="100" step="1">
  </p>
  <p>
    <h3>Meter</h3>
    <meter value="60" min="0" max="100" low="73" high="87">D</meter>
  </p>
  <p>
    <h3>Progress</h3>
    <progress id="p" max="100" value="80"><span id="completed">0</span>%</progress>
  </p>
</body>
```

## Exemplos Range, Meter, Progress

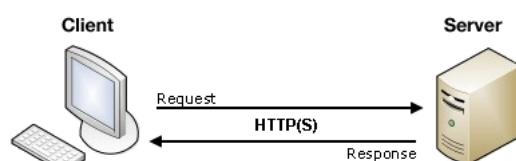


Fonte: autoria própria

### 9.1.12 Enviando dados de um Formulário

- De acordo com Mozilla Foundation, a web é basicamente estruturada em uma arquitetura cliente / servidor, onde um cliente (navegador) envia um pedido a um servidor (servidor web tal como Apache, Nginx, IIS, Tomcat), utilizando o protocolo HTTP. E o servidor responde à solicitação usando o mesmo protocolo.

Figura 72 – Exemplos arquitetura web



Fonte:

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Sending\\_and\\_retrieving\\_form\\_data](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Sending_and_retrieving_form_data)

### 9.1.12.1 Atributos do FORMULÁRIO:

- **action:** define onde os dados são enviados. Seu valor deve ser um URL válido. Se esse atributo não for fornecido, os dados serão enviados para o URL da página que contém o formulário.
  - Exemplo: <form action="http://teste.com"> .... </form>
- **method:** define como os dados são enviados.
  - O protocolo HTTP fornece várias maneiras de executar uma solicitação.
  - Dados de formulários HTML podem ser transmitidos através de métodos GET ou POST.
    - \* **GET:** se um formulário é enviado usando esse método, os dados enviados ao servidor são anexados ao URL. Os dados são anexados ao URL como uma série de pares nome/valor. Após a conclusão do endereço da Web de URL, incluímos um ponto de interrogação (?) seguido dos pares nome/valor, cada um separado por um e comercial (&).
    - \* **POST:** se um formulário é enviado usando esse método, os dados são anexados ao corpo da solicitação HTTP. As solicitações HTTP nunca são exibidas para o usuário.
- Do lado do sevidor, para acessar estes dados, depende da plataforma de desenvolvimento que você usa e de quaisquer frameworks específicos que você possa usar com ele.

**Acesse:**

[https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Sending\\_and\\_retrieving\\_form\\_data](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Sending_and_retrieving_form_data).

**Acesse:** [https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Form\\_validation](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Form_validation).

# 10 Aplicando Bootstrap e Media Queries

## 10.1 De acordo com o site do Bootstrap:

- **Bootstrap** é o mais popular framework HTML, CSS, e JS para desenvolvimento de projetos responsivo e focado para dispositivos móveis na web.
- Para começar a utilizar, acesse a página <http://getbootstrap.com.br/getting-started/> e baixe a versão minificada que contém as pastas do CSS, JS e Fontes.
- Em seguida, ao lado direito, selecione **BASIC TEMPLATE** e crie um arquivo html no seu editor de texto preferido.

Figura 73 – Template Básico

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3      <head>
4          <meta http-equiv="X-UA-Compatible" content="IE=edge">
5          <!-- X-UA-Compatible significa: Modo de compatibilidade com versões mais antigas do IE.
6              E IE=edge significa que o IE deve renderizar aquela página com a versão mais recente que ele tiver.
7              IMPORTANTE: DEVE ser a primeira tag após a tag <head>, pois caso contrário o IE ignora. -->
8          <meta charset="utf-8">
9          <!-- indicar a codificação do próprio documento.
10         O unicode UTF-8 é um formato de codificação que tem um tamanho variável de caractere, podendo ter de 1 até 4 bytes.
11         Os caracteres mais comuns são mapeados para códigos de 1 byte, outros menos comuns, como a maioria dos caracteres
12         acentuados, possuem 2 bytes.
13     -->
14     <meta name="viewport" content="width=device-width, initial-scale=1">
15     <!-- width=device-width significa que a escala inicial do nosso layout é equivalente ao tamanho do dispositivo. Se não
16         fizermos isto, alguns aparelhos móveis vão redimensionar o layout por conta própria e o design responsivo só vai
17         funcionar no desktop! Para isto vamos manipular a metatag viewport. -->
18     <!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
19     <title>Bootstrap 101 Template</title>
20     <!-- Bootstrap -->
21     <link href="css/bootstrap.min.css" rel="stylesheet">
22     <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
23     <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
24     <!--[if lt IE 9]>
25         <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>
26         <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
27     </if>
28     </head>
29     <body>
30         <h1>Aplicando Bootstrap</h1>
31         <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
32         <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
33         <!-- Include all compiled plugins (below), or include individual files as needed -->
34         <script src="js/bootstrap.min.js"></script>
35     </body>
36 </html>
```

Fonte: <http://getbootstrap.com.br/getting-started/#template>

- Bootstrap define estilos globais básicos de exibição, tipografia, e link.
- Bootstrap adota um sistema de grid. Pode escolher entre os dois tipos de "containers" para usar nos projetos. Lembre-se que isso cria um padding a mais, então não deve-se incluir nenhum "container" dentro de outro. Utilize:
  - **.container**: para conteúdo responsivo com largura limitada.
  - **.container-fluid**: para conteúdo responsivo com largura fluída, isto é, toda a largura da janela.
- Sistema de grid é usado para criação de layout de páginas pelo uso de linhas e colunas com conteúdo respectivamente.
- Bootstrap funciona da seguinte forma:
  - As linhas (row) devem estar localizadas dentro de um .container (largura fixa) ou .container-fluid (largura cheia) para alinhamento e padding adequado.

- Use linhas para criar grupos horizontais de colunas.
  - O conteúdo deve ser inserido dentro das colunas, e apenas colunas podem ser filhas imediatas de linhas.
  - Consulte o site para maiores detalhes e mais informações: <http://getbootstrap.com/css/#grid-intro>.
- Media Queries:

```
/* Dispositivos extra pequenos (extra small) (celulares , menores que 768px) */
/* No media query since this is the default in Bootstrap */

/* Dispositivos pequenos (tablets , 768px e superiores) */
@media (min-width: @screen-sm-min) and (max-width: @screen-sm-max) { ... }
/* Dispositivos médios (desktops , 992px e superiores) */
@media (min-width: @screen-md-min) and (max-width: @screen-md-max) { ... }
/* Dispositivos grandes (desktops grandes , 1200px e superiores) */
@media (min-width: @screen-lg-min) { ... }
```

Figura 74 – Opções de GRID

	Dispositivos extra pequenos Telefones (<768px)	Dispositivos pequenos Tablets (≥768px)	Dispositivos médios Desktops (≥992px)	Dispositivos grandes Desktops (≥1200px)
Comportamento do grid	Horizontal em todos os momentos	Colapsa ao princípio, horizontal acima de breakpoints		
Largura do container	nenhum (auto)	750px	970px	1170px
Prefixo de classe	.col-xs-	.col-sm-	.col-md-	.col-lg-
número de colunas	12			
Largura da coluna	Auto	~62px	~81px	~97px
Largura da calha	30px (15px em cada lado da coluna)			
Encaixável	Sim			
Distância	Sim			
Ordenação de colunas	Sim			

Fonte: <http://getbootstrap.com.br/css/#overview>

Acesse: <http://getbootstrap.com.br/css/#overview>.

*Artigo - Desenvolvimento responsivo e viewport (by Leandro Lima):*  
<http://blog.popupdesign.com.br/desenvolvimento-responsivo-e-viewport/>.

*Artigo - Logic in Media Queries:* <https://css-tricks.com/logic-in-media-queries/>.

*Artigo - Design Responsivo III – Media Queries e Compatibilidade (by Dani Guerrato):*  
<http://blog.popupdesign.com.br/design-responsivo-iii-media-queries-e-compatibilidade/>.

*Artigo - Design Responsivo na prática 2: do layout ao HTML (by Dani Guerrato):*  
<https://tableless.com.br/design-responsivo-na-pratica-2-layout-ao-html/>.

# 11 Animações com CSS

## 11.1 Configurando Animação

- De acordo com Mozilla Foundation, as animações CSS3 podem ocorrer a partir do momento em que usamos transições de um estilo.
- As animações são compostas de dois itens:
  - um estilo descrevendo a animação e,
  - um set de keyframes que indicam o estado final e inicial do estilo CSS da animação, bem como possíveis pontos de quebra intermediários ao longo do caminho.
- São três vantagens para animações CSS:
  - São de fácil utilização para animações simples;
  - Você pode criá-las sem mesmo ter que conhecer JavaScript.
  - Executam bem, mesmo sobre moderada carga do sistema.
- Para criar uma sequência de animação CSS, utilizaremos a propriedade `animation` e outras, o que permitirá a configuração da sincronização da animação, a sequencia de como progredir.
- Para definir a aparênciça da animação, utiliza-se a regra `@keyframes`. Veremos mais à frente.

### 11.1.1 Animation

- Valores iniciais para as propriedades de `animation`:

- `animation-name:` none
- `animation-duration:` 0s
- `animation-timing-function:` ease
- `animation-delay:` 0s
- `animation-iteration-count:` 1
- `animation-direction:` normal
- `animation-fill-mode:` none
- `animation-play-state:` running

- Sintaxe:

```
/* @keyframes duration | timing-function | delay | iteration-count | direction |
   fill-mode | play-state | name */
   animation: 3s ease-in 1s 2 reverse both paused slidein;
/* @keyframes duration | timing-function | delay | name */
   animation: 3s linear 1s slidein;
/* @keyframes duration | name */
   animation: 3s slidein;
```

- Propriedades do `animation`:

- ***animation-name:*** Especifica o nome da regra @keyframes <nomeRegra> descrevendo os keyframes da animação.
  - \* Você quem escolhe o nome.
- ***animation-duration:*** Configura o tempo que uma animação deveria levar para completar um ciclo.
  - \* Valores permitidos: tempo em segundos ou milisegundos.
- ***animation-timing-function:*** Configura a sincronização da animação; que é, como a animação transita por keyframes, por estabilizar curvas de aceleração.
  - \* Valores permitidos: ease, ease-in, ease-out, ease-in-out, linear, step-start, step-end, etc.
- ***animation-delay:*** Configura o delay entre o tempo em que o elemento é carregado e o inicio da sequência de animação.
  - \* Valores permitidos: tempo em segundos ou milisegundos.
- ***animation-iteration-count:*** Configura o número de vezes que uma animação deveria se repetir; você pode especificar infinito para repetir a animação indefinidamente.
  - \* Valores permitidos: infinite, ou um número inteiro.
- ***animation-direction:*** Configura se a animação deve ou não alternar a direção em cada execução durante a sequência ou voltar ao ponto inicial e se repetir.
  - \* Valores permitidos: normal, reverse, alternate, alternate-reverse, inherit, initial, unset.
- ***animation-fill-mode:*** Configura que valores são aplicados pela animação antes e depois de se executar.
  - \* Valores permitidos: none, forwards, backwards, both, etc.
- ***animation-play-state:*** Permite você pausar e resumir a sequência da animação.
  - \* Valores permitidos: running, paused, inherit, initial, unset.

**Acesse:** [https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS\\_Animations/Usando\\_anima%C3%A7%C3%A3o\\_no\\_CSS](https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Animations/Usando_anima%C3%A7%C3%A3o_no_CSS).

**Acesse:** <https://css-tricks.com/almanac/properties/a/animation/>.

### 11.1.2 Transition

- Transition: permite definir a transição entre dois estados de um elemento. Estados diferentes podem ser definidos usando pseudo-classes tais como :hover ou :active; ou dinamicamente, usando javascript.
- Valores iniciais para a propriedade de transition:
  - ***transition-delay:*** 0s
  - ***transition-duration:*** 0s
  - ***transition-property:*** all
  - ***transition-timing-function:*** ease
- Sintaxe:

```
/* property name | duration */
    transition: margin-left 4s;
/* property name | duration | delay */
    transition: margin-left 4s 1s;
/* property name | duration | timing function | delay */
```

```

        transition: margin-left 4s ease-in-out 1s;
/* Apply to 2 properties */
        transition: margin-left 4s, color 1s;
/* Apply to all changed properties */
        transition: all 0.5s ease-out;
/* Global values */ transition: inherit; transition: initial; transition: unset;

```

- Propriedades do transition:

- ***transition-delay***: Configura o delay entre o tempo em que o elemento é carregado e o inicio da sequência de animação.  
\* Valores permitidos: tempo em segundos ou milisegundos.
- ***transition-duration***: Configura o tempo que uma animação deveria levar para completar um ciclo.  
\* Valores permitidos: tempo em segundos ou milisegundos.
- ***transition-property***: É usada para especificar os nomes das propriedades CSS às quais um efeito de transição deve ser aplicado.  
\* Valores permitidos: none, all.
- ***transition-timing-function***: Permite que você estabeleça uma curva de aceleração, de modo que a velocidade da transição pode variar ao longo de sua duração.  
\* Valores permitidos: ease, ease-in, ease-ot, ease-in-out, linear, step-start, step-end.

**Acesse:** <https://css-tricks.com/almanac/properties/t/transition/>.

## 11.2 Definindo sequencia de animação com Keyframes

- O @keyframes permite que os autores controlem os passos intermediários em uma seqüência de animação CSS, estabelecendo pontos chave ao longo da seqüência de animação que deve ser alcançada por certos pontos durante a animação.
- Isso lhe dá um controle mais específico sobre os passos intermediários da seqüência de animação do que você obteria ao deixar o navegador lidar com tudo automaticamente.

**Acesse:** <https://developer.mozilla.org/pt-BR/docs/Web/CSS/@keyframes>.

**Acesse:** <https://css-tricks.com/snippets/css/keyframe-animation-syntax/>.

## 11.3 Efeitos de transições

- Aparecimento gradual (fade in)
- Trocar de cor (change color)
- Valores da propriedade Transform:
  - Crescer (scale): O valor scale modificará a dimensão do elemento. Ele aumentará proporcionalmente o tamanho do elemento levando em consideração o tamanho original do elemento.
  - Encolher (scale): O valor scale modificará a dimensão do elemento. Ele diminuirá proporcionalmente o tamanho do elemento levando em consideração o tamanho original do elemento.
  - Rotacionar (rotate): O rotate rotaciona o elemento levando em consideração seu ângulo, especialmente quando o ângulo é personalizado com o transform-origin.

- Translação (translate): O translation moverá o elemento no eixo X e Y. O interessante é que você não precisa se preocupar com floats, positions, margins e etc. Se você aplica o translation, ele moverá o objeto e pronto.
- Arredondar (border-radius)
- Sombras 3D (box-shadow)
- Swing (Mover o retângulo) utilizando animation.
- Inserir borda (box-shadow)
- Skew: modificará os ângulos dos elementos.

*Acesse:* <http://wime.com.br/2015/08/21/8-efeitos-de-transicao-em-css3-css3-transitions/>.

*Acesse:* <https://css-tricks.com/almanac/properties/t/transform/>.

## 11.4 Exemplificando

### 11.4.1 Exemplo 1

- Vejamos:

Figura 75 – Exemplo html

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <title>Girar, Subir, Descer, Pendurar</title>
5      <meta charset="utf-8">
6      <link rel="stylesheet" type="text/css" href="estiloGirarSubirPendurar.css">
7  </head>
8  <body>
9      <section>
10         <h1>Exemplo de Animação - Efeito Girar</h1>
11         <div class="giragira"></div>
12     </section>
13     <section>
14         <h1>Exemplo de Animação - Efeito Subir e Descer</h1>
15         <div class="vaivolta"></div>
16     </section>
17     <section>
18         <h1>Exemplo de Animação - Efeito Pêndulo</h1>
19         <div class="pendura"></div>
20     </section>
21 </body>
22 </html>
```

Fonte: autoria própria

Figura 76 – Exemplo Formatação do html no CSS

```

114  nav,
115  section
116  {
117      margin: 10px auto;
118      padding: 2em;
119      border-radius: 0.8em;
120      width: 80%;
121      border: 2px solid green;
122      height: 300px;
123  }
124  h1 {
125      color: green;
126  }
127
```

Fonte: autoria própria

Figura 77 – Exemplo Formatação Efeito Girar

```

1  /* Girar */
2  /* Para o Mozilla: @-moz-keyframes
3   * Para Safari, Chrome: @-webkit-keyframes */
4
5  @keyframes rodaroda {
6      0% {
7          -webkit-transform: rotate(0deg);
8      }
9      50% {
10         background:red;
11         -webkit-transform: rotate(180deg);
12     }
13     100% {
14         background:blue;
15         -webkit-transform: rotate(360deg);
16     }
17 }
18
19 .giragira {
20     width:150px;
21     height:150px;
22     margin:10px auto 45px;
23     background: black;
24
25     -webkit-animation: rodaroda 0.8s linear infinite;
26     -moz-animation: rodaroda 3s linear infinite;
27     -o-animation: rodaroda 0.5s linear infinite;
28     animation: rodaroda 0.8s linear infinite;
29 }
30

```

Fonte: autoria própria

Figura 78 – Exemplo Formatação Efeito Subir e Descer

```

32  /* Subir e Descer */
33  @keyframes vaivolta {
34      0% {
35         top:320px;
36     }
37     50% {
38         top:360px;
39     }
40     100% {
41         top:380px;
42         border-radius: 40px;
43         background: yellow;
44     }
45 }
46
47 .vaivolta {
48     width:150px;
49     height:150px;
50     margin:170px auto 0;
51     background: blue;
52
53     position:absolute;
54     left: 50%;
55     -webkit-animation: vaivolta 1s alternate linear infinite;
56     -moz-animation: vaivolta 1s alternate linear infinite;
57     -o-animation: vaivolta 1s alternate linear infinite;
58     animation: vaivolta 1s alternate linear infinite;
59 }
60

```

Fonte: autoria própria

Figura 79 – Exemplo Formatação Efeito Pêndulo

```

62 /* Pêndulo */
63 @keyframes pendura {
64   from {
65     -webkit-transform: rotate(0deg);
66   }
67   to {
68     -webkit-transform: rotate(95deg);
69     background: yellow;
70   }
71 }
72
73 .pendura {
74   width:150px;
75   height:150px;
76   margin:35px auto 0;
77   background: blue;
78
79   -webkit-transform: rotate(0deg);
80   -webkit-transform-origin: 10px 10px;
81
82   -moz-transform: rotate(0deg);
83   -moz-transform-origin: 10px 10px;
84
85   -o-transform: rotate(0deg);
86   -o-transform-origin: 10px 10px;
87
88   -webkit-animation: pendura 1s alternate ease infinite;
89   -moz-animation: pendura 1s alternate ease infinite;
90   -o-animation: pendura 1s alternate ease infinite;
91
92 }
93

```

Fonte: autoria própria

#### 11.4.2 Exemplo 2

- Vejamos:

Figura 80 – Exemplo html

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <title>Transições</title>
5   <meta charset="utf-8">
6   <link rel="stylesheet" type="text/css" href="estiloTransition.css">
7 </head>
8 <body>
9   <div class="fade"><h1>FADE IN</h1></div>
10  <div class="color"><h1>CHANGE COLOR</h1></div>
11  <div class="grow"><h1>GROW</h1></div>
12  <div class="shrink"><h1>SHRINK</h1></div>
13  <div class="rotate"><h1>ROTATE</h1></div>
14  <div class="circle"><h1>CIRCLE</h1></div>
15  <div class="threed"><h1>SOMBRA 3D</h1></div>
16  <div class="swing"><h1>SWING</h1></div>
17  <div class="border"><h1>BORDER</h1></div>
18  <div class="skew"><h1>SKEW</h1></div>
19  <div class="box"></div>
20 </body>
21 </html>
22

```

Fonte: adaptado de

<http://wime.com.br/2015/08/21/8-efeitos-de-transicao-em-css3-css3-transitions/>

Figura 81 – Exemplo Formatação CSS

```

1  body {
2      display: flex;
3      flex-direction: row;
4      flex-wrap: wrap;
5  }
6
7  body > div {
8      width: 200px;
9      height: 200px;
10     background: #676470;
11     transition: all 0.3s ease;
12     margin: 10px;
13     padding: 10px;
14 }
15
16 .fade {
17     opacity: 0.5;
18 }
19
20 .fade:hover {
21     opacity: 1;
22 }
23
24 .color:hover {
25     background: #53a7ea;
26 }
27
28 .grow:hover {
29     -webkit-transform: scale(1.3);
30     -ms-transform: scale(1.3);
31     transform: scale(1.3);
32 }
33
34 .shrink:hover {
35     -webkit-transform: scale(0.8);
36     -ms-transform: scale(0.8);
37     transform: scale(0.8);
38 }

```

Fonte: adaptado de

<http://wime.com.br/2015/08/21/8-efeitos-de-transicao-em-css3-css3-transitions/>

Figura 82 – Exemplo Formatação CSS

```

39 |
40 .rotate:hover {
41     -webkit-transform: rotateZ(-30deg);
42     -ms-transform: rotateZ(-30deg);
43     transform: rotateZ(-30deg);
44 }
45
46 .skew:hover {
47     -webkit-transform: skew(35deg);
48     -ms-transform: skew(35deg);
49     transform: skew(35deg);
50 }
51
52 .circle:hover {
53     border-radius: 50%;
54 }
55
56 .threed:hover {
57     box-shadow:
58         8px 8px #53a7ea,
59         9px 9px #53a7ea,
60         10px 10px #53a7ea;
61     -webkit-transform: translateX(-10px);
62     transform: translateX(-10px);
63 }
64
65 .swing:hover{
66     -webkit-animation: swing 1s ease;
67     animation: swing 1s ease;
68     -webkit-animation-iteration-count: 1;
69     animation-iteration-count: 1;
70 }
71

```

Fonte: adaptado de

<http://wime.com.br/2015/08/21/8-efeitos-de-transicao-em-css3-css3-transitions/>

Figura 83 – Exemplo Formatação CSS

```

71
72 @keyframes swing
73 {
74   15%
75   {
76     -webkit-transform: translateX(5px);
77     transform: translateX(5px);
78   }
79   30%
80   {
81     -webkit-transform: translateX(-5px);
82     transform: translateX(-5px);
83   }
84   50%
85   {
86     -webkit-transform: translateX(3px);
87     transform: translateX(3px);
88   }
89   65%
90   {
91     -webkit-transform: translateX(-3px);
92     transform: translateX(-3px);
93   }
94   80%
95   {
96     -webkit-transform: translateX(2px);
97     transform: translateX(2px);
98   }
99   100%
100  {
101    -webkit-transform: translateX(0);
102    transform: translateX(0);
103  }
104}
105

```

Fonte: adaptado de

<http://wime.com.br/2015/08/21/8-efeitos-de-transicao-em-css3-css3-transitions/>

Figura 84 – Exemplo Formatação CSS

```

105 .border:hover {
106   box-shadow: inset 0 0 0 25px #53a7ea;
107 }
108
109 .box {
110   border-style: solid;
111   border-width: 1px;
112   display: block;
113   width: 100px;
114   height: 100px;
115   background-color: #0000FF;
116   -webkit-transition: width 2s, height 2s, background-color 2s, -webkit-transform 2s;
117   transition: width 2s, height 2s, background-color 2s, transform 2s;
118 }
119
120
121 .box:hover {
122   background-color: #FFCCCC;
123   width: 200px;
124   height: 200px;
125   -webkit-transform: rotate(180deg);
126   transform: rotate(180deg);
127 }
128

```

Fonte: adaptado de

<http://wime.com.br/2015/08/21/8-efeitos-de-transicao-em-css3-css3-transitions/>

#### 11.4.3 Exemplo 3

- Vejamos:

Figura 85 – Exemplo html

```
1 <html>
2   <head>
3     <title>Simple CSS3: Animations and Transforms</title>
4     <meta charset="utf-8">
5     <link rel="stylesheet" type="text/css" href="estiloBoxAnimacao.css">
6   </head>
7   <body>
8     <h2 id="animation">Simple CSS3: Animations and Transforms</h2>
9     <div class="box slideright">Slider box</div>
10    <div class="box rotate">Rotate box</div>
11    <div class="box rounding">Round box</div>
12    <div class="box scale">Scaling box</div>
13    <div class="box color">Color box</div>
14    <div class="box skew">skew box</div>
15  </body>
16 </html>
```

Fonte: autoria própria

Figura 86 – Exemplo Formatação CSS

```
1 .box {
2   float: left;
3   margin: 1em;
4   width: 100px;
5   height: 60px;
6   border: 2px solid purple;
7   background-color: white;
8   border-radius: 5px;
9   line-height: 60px;
10  text-align: center;
11  -webkit-transition:all 1s ease-in-out;
12  -moz-transition:all 1s ease-in-out;
13  -o-transition:all 1s ease-in-out;
14  -ms-transition:all 1s ease-in-out;
15 }
16 .slideright:hover {
17  -webkit-transform:translate(3em,0);
18  -moz-transform:translate(3em,0);
19  -o-transform:translate(3em,0);
20  -ms-transform:translate(3em,0);
21 }
22 .rotate:hover {
23  -webkit-transform:rotate(45deg);
24  -moz-transform:rotate(45deg);
25  -o-transform:rotate(45deg);
26  -ms-transform:rotate(45deg);
27 }
28 .
29 .rounding:hover {
30  border-radius: 100px;
31 }
32 .
33 .scale:hover {
34  -webkit-transform:scale(2);
35  -moz-transform:scale(2);
36  -o-transform:scale(2);
37  -ms-transform:scale(2);
38 }
39 }
```

Fonte: autoria própria

Figura 87 – Exemplo Formatação CSS

```
40
41 .color:hover {
42   background: red;
43 }
44
45 .skew:hover {
46   -webkit-transform: scale(1.0) skewX(10deg) skewY(10deg);
47   -webkit-transform-origin: 0% 0%;
48   -moz-transform: scale(1.0) skewX(10deg) skewY(10deg);
49   -moz-transform-origin: 0% 0%;
50   -o-transform: scale(1.0) skewX(10deg) skewY(10deg);
51   -o-transform-origin: 0% 0%;
52   -ms-transform: scale(1.0) skewX(10deg) skewY(10deg);
53   -ms-transform-origin: 0% 0%;
54   transform: scale(1.0) skewX(10deg) skewY(10deg);
55   transform-origin: 0% 0%;
56 }
57 }
```

Fonte: autoria própria

#### 11.4.4 Outros Exemplos

*Acesse:* <https://paulrhayes.com/experiments/cube/multiCubes.html>.

*Acesse:* <https://paulrhayes.com/2009-04/3d-cube-using-css-transformations/>.

## 12 Leitura de Conteúdos por robôs

- Leitura de conteúdo por máquinas?
  - Existe 3 formas de permitir a leitura de um conteúdo incorporado em um documento Web:
    - \* HTML+RDFa -> <https://www.w3.org/TR/html-rdfa/>.
    - \* Microformats -> <http://microformats.org>.
    - \* Microdata: Adicionando microdados às páginas da Web ajuda a motores de busca a entenderem melhor o conteúdo das páginas, seus temas, etc.
- Microdatas:
  - Objetivo: o principal objetivo do microdados é Search Engine Optimization (SEO). Esta informação não é visível para os seres humanos: é informação semântica pura.
  - Tipos populares de microdados são eventos, perfil de uma pessoa, a descrição de uma organização, os detalhes de uma receita, uma descrição do produto, a localização geográfica, etc.
- Exemplos:

Figura 88 – Exemplo de Microdatas

```

1. <section itemscope itemtype="http://schema.org/Person">
2.   <h1>Contact Information</h1>
3.   <dl>
4.     <dt>Name</dt>
5.     <dd itemprop="name">Michel Buffa</dd>
6.     <dt>Position</dt>
7.     <dd><span itemprop="jobTitle">
8.       Professor/Researcher/Scientist</span> for
9.       <span itemprop="affiliation">
10.         University of Côte d'Azur, France
11.       </span>
12.     </dd>
13.   </dl>
14.   <!-- SURFACE ADDRESS GOES HERE -->
15.   <h1>My different online public accounts</h1>
16.   <ul>
17.     <li><a href="http://www.twitter.com/micbuffa"
18.       itemprop="url">Twitter profile</a></li>
19.     <li><a href="http://www.blogger.com/micbuffa"
20.       itemprop="url">Michel Buffa's blog</a></li>
21.   </ul>
22. </section>
```

Fonte: curso EDX

- Analisando o itemprop="url" -> <http://schema.org/url>.

Figura 89 – Exemplo de Microdados

```

1. ...
2. </dl>
3.
4. <!-- SURFACE ADDRESS GOES HERE -->
5.
6. <dd itemprop="address" itemscope
7.   itemtype="http://schema.org/PostalAddress">
8.   <span itemprop="streetAddress">10 promenade des anglais</span><br>
9.   <span itemprop="addressLocality">Nice</span>,
10.  <span itemprop="addressRegion">Alpes maritimes, France</span>
11.  <span itemprop="postalCode">06410</span><br>
12.  <span itemprop="addressCountry" itemscope
13.    itemtype="http://schema.org/Country">
14.    <span itemprop="name">France</span>
15.  </span>
16. </dd>
17.
18. <h1>My different online public accounts</h1>
19.
20. ...

```

Fonte: curso EDX

- Diferentes usos:
  - O navegador, ou uma extensão do navegador, pode interpretar o último exemplo de um endereço e pode propor para enviá-lo a um aplicativo de mapa.
  - Um rastreador da Web pode interpretar isso como um endereço e exibi-lo em suas respostas usando um layout de apresentação dedicada.
  - Algum código JavaScript na página pode acessar esses dados.
  - Com outros tipos de microdados, para eventos, por exemplo, o navegador pode aparecer uma aplicação de calendário, etc.
  - Nota: Para usuários avançados, Microdados é muito semelhante ao microformatos, que usam classes de HTML ou para RDFa → <a href="http://www.w3.org/TR/xhtml-rdfa-primer/">, que não valida em HTML4 ou HTML5. Porque RDFa foi considerada muito difícil para os autores a escrever (Google realizou uma pesquisa que descobre que autores fazem 30% mais erros com RDFa do que com outros formatos), microdados é a resposta de HTML5 para a necessidade de incorporar semântica em documentos HTML.
- Links Interessantes microdados:
  - HTML5 Microdata and Schema.org: <http://journal.code4lib.org/articles/6400>.
  - Referência de vocabulários mais comuns com Microdados: <http://schema.org>.
  - Capítulo 10. Diving in Microdata: <http://diveinto.html5doctor.com/extensibility.html>.
  - Ferramenta de Teste microdata: <https://search.google.com/structured-data/testing-tool>. Página a testar: <http://jsbin.com/bamahon>.
  - Ferramenta de visualização de Microdata: <http://tools.seomoves.org/microdata>. Página a testar: <http://output.jsbin.com/wozoye>.
- Geradores de microdata:
  - Schema Structured Data:
    - \* <https://raventools.com/site-auditor/seo-guide/schema-structured-data/>.

- Microdata Generator using Schema.org Vocabulary:
  - \* <http://www.microdatagenerator.com>.
- HTML5 Microdata Format Generator:
  - \* <http://www.barryko.com/seo/html5-microdata-schema-generator/>.

# 13 Canvas

## 13.1 O que é canvas ?

- O canvas é um elemento novo em HTML5, que permite que se desenhem elementos gráficos usando JavaScript. Utilizado para renderizar texto, imagens, gráficos, linhas gradientes e outros efeitos dinamicamente.
- Desenhar na tela se dá através da API de tela 2D. Essa API contém uma variedade de funções que fornecem o poder de se desenhar praticamente o que se queira na tela. canvas.html

*Acesse - API Canvas:* <https://msdn.microsoft.com/en-us/library/ff975057.aspx>.

*Código no Github do Livro Core HTML5 Canvas:* <https://github.com/corehtml5canvas/code>.

*Código online Livro Core HTML5 Canvas:* <http://corehtml5canvas.com/code-live/>.

*Tutorial Canvas:* <http://www.html5canvastutorials.com/>.

*Tutorial Mozilla:* [https://developer.mozilla.org/pt-BR/docs/Web/Guide/HTML/Canvas\\_tutorial](https://developer.mozilla.org/pt-BR/docs/Web/Guide/HTML/Canvas_tutorial).

## 13.2 Estrutura Básica

- Caso não seja definido a largura ou altura do elemento canvas, o atributo de largura padrão é 300 e o atributo de altura padrão é 150.
- O elemento <canvas> pode ser estilizado como qualquer imagem.
- Porque usar as definições de largura e altura na própria tag do canvas ? E não no CSS ?
  - <http://stackoverflow.com/questions/5034529/size-of-html5-canvas-via-css-versus-element-a>
  - Se utilizarmos o CSS, o elemento pode ser redimensionado, mas durante a renderização a imagem é escalonada para caber no tamanho do layout.
  - Nota: Se as suas renderizações parecerem distorcidas, tente especificar os atributos width e height no <canvas> e não usando CSS.

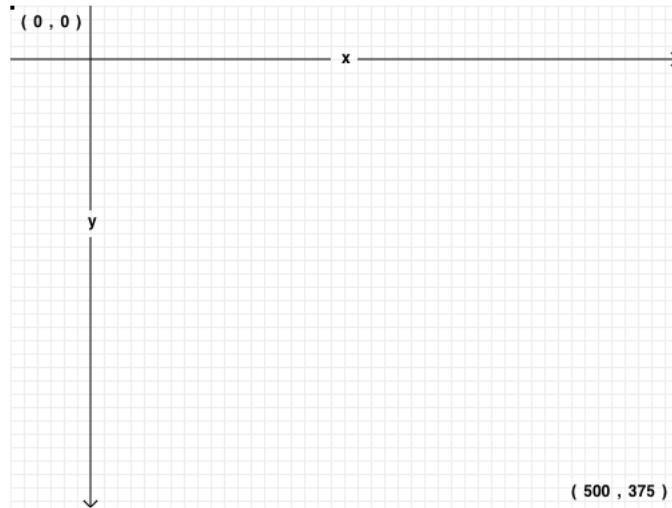
*HTML5 Canvas:* <https://diveintohtml5.com.br/>.

*Atualização recente da Especificação do HTML5 Canvas (09/maio/2017):*  
<https://html.spec.whatwg.org/multipage/scripting.html#the-canvas-element>.

### 13.2.1 Criando nosso primeiro canvas

- Estrutura de coordenadas:

Figura 90 – Coordenadas x, y



Fonte: <https://diveintohtml5.com.br/canvas.html>

- Defina a tag <canvas> </canvas> no seu html e com suas respectivas dimensões.
- Em seguida, crie um arquivo javascript e comece seu código.

Figura 91 – Código HTML

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Desenhando com Canvas</title>
6      <style>
7          #myCanvas {
8              border : 10px solid #555
9          }
10     </style>
11 </head>
12 <body>
13     <canvas id="myCanvas" width="500" height="600">Sem suporte a canvas</canvas>
14 </body>
15 </html>
```

Fonte: autoria própria

Figura 92 – Código Javascript

```

1 // Definindo a área de desenho
2 var c_canvas = document.getElementById("myCanvas");
3 var ctx = c_canvas.getContext("2d");
4
5 // Inserir seu código aqui
6
```

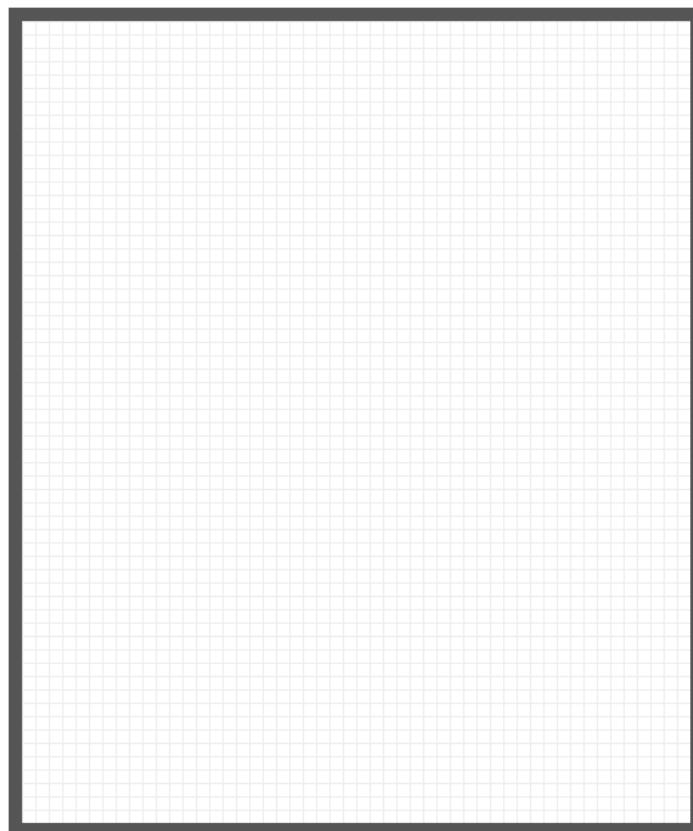
Fonte: adaptado de <https://diveintohtml5.com.br/canvas.html>

Figura 93 – Código Javascript

```
1 // Definindo a área de desenho
2 var c_canvas = document.getElementById("myCanvas");
3 var ctx = c_canvas.getContext("2d");
4 //Desenhando linhas verticais
5 for (var x = 0.5; x < 500; x += 10)
6 {
7     ctx.moveTo(x, 0);
8     ctx.lineTo(x, 600);
9 }
10 //Desenhando linhas horizontais
11 for (var y = 0.5; y < 600; y += 10)
12 {
13     ctx.moveTo(0, y);
14     ctx.lineTo(600, y);
15 }
16
17 ctx.strokeStyle = "#eee";
18 ctx.stroke();
```

Fonte: <https://diveintohtml5.com.br/canvas.html>

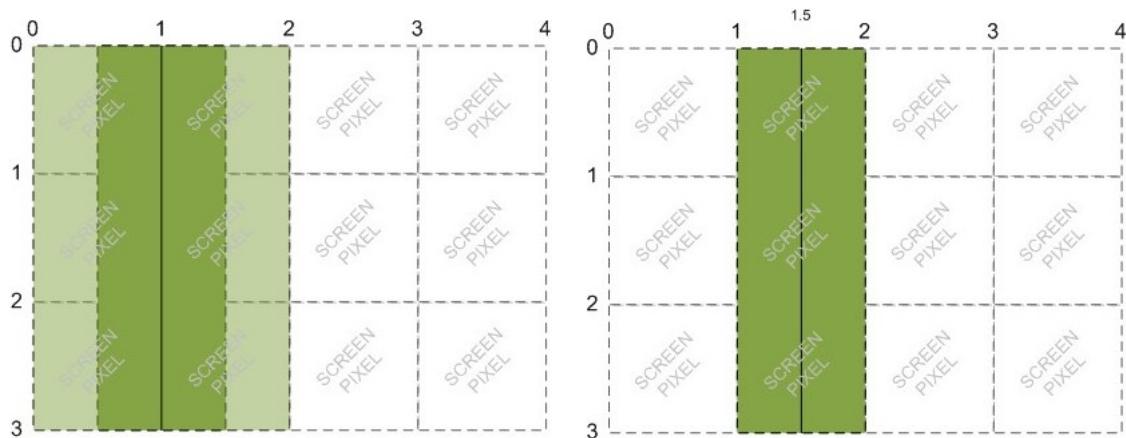
Figura 94 – Exibição no navegador



Fonte: autoria própria

- A tela não consegue exibir meio pixel, então irá expandir a linha para cobrir um total de dois pixels. Veja a primeira Figura abaixo. Na segunda Figura é exibido o desenho da linha com 1 pixel quando utilizamos 0.5 pixel para desenhar.

Figura 95 – Exemplo de desenho do Pixel



Fonte: <https://diveintohtml5.com.br/canvas.html>

### 13.2.1.1 Desenhando

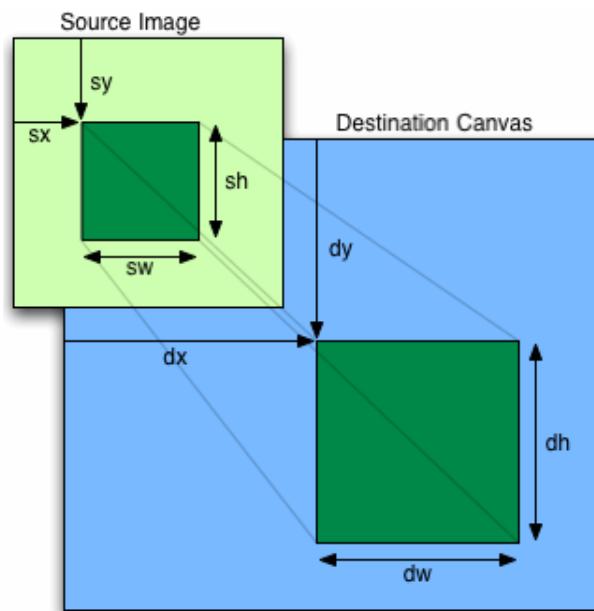
- `.moveTo(x, y)`: move o cursor para o ponto inicial especificado.
- `.lineTo(x, y)`: desenha a linha até o ponto final especificado.
- `.strokeStyle = "#fff"`: definindo a cor da linha.
- `.stroke()`: desenhando a linha no canvas.
- `.fillStyle = "rgb(0, 162, 232)"`: o padrão é preto sólido, mas você pode definir o que quiser.
- `.fillRect(x, y, largura, altura)` desenha um retângulo preenchido com o `fillStyle` atual.
- `.strokeRect(x, y, largura, altura)` desenha um retângulo com o `strokeStyle` atual. Desenha as bordas.
- `.clearRect(x, y, largura, altura)` limpa os pixels no retângulo especificado.
- `.font = "bold 12px sans-serif"`: pode ser qualquer coisa que você colocaria na regra font do CSS. Incluindo font style, font variant, font weight, font size, line height, e font family.
- `.textAlign = "right"`: Controla o alinhamento do texto. É parecido (mas não idêntico) a regra text-align do CSS. Os possíveis valores são start, end, left, right, e center.
- `.textBaseline = "bottom"`: Controla onde o texto é desenhado relativo ao ponto de início. Os possíveis valores são top, hanging, middle, alphabetic, ideographic, ou bottom.
- `.fillText(conteudo, posicao x, posicao y)`: desenha o texto de fato.
- Dois tipos de gradientes:
  - `.createLinearGradient(x0, y0, x1, y1)`: pinta através de uma linha de  $(x_0, y_0)$  até  $(x_1, y_1)$ .
  - `.createRadialGradient(x0, y0, r0, x1, y1, r1)`: pinta através de um cone entre dois círculos. Os primeiros três parâmetros representam o início do círculo, com origem  $(x_0, y_0)$  e raio  $r_0$ . Os últimos três parâmetros representam o fim do círculo, com origem  $(x_1, y_1)$  e raio  $r_1$ .
- Desenhando imagens:
  - `.drawImage(image, dx, dy)` pega uma imagem e a desenha no canvas. A coordenada  $(dx, dy)$  será o canto superior esquerdo da imagem. Coordenadas  $(0, 0)$  devem desenhar a imagem no canto superior esquerdo do canvas.

- `.drawImage(image, dx, dy, dw, dh)` pega uma imagem, escala para a largura de dw e altura de dh, e a desenha no canvas nas coordenadas (dx, dy).
- `.drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh)` pega uma imagem, ajusta ela para o retângulo (sx, sy, sw, sh), escala para as dimensões (dw, dh), e a desenha no canvas nas coordenadas (dx, dy).
  - \* O retângulo de origem é o retângulo [no interior da imagem fonte] cujos cantos são os quatro pontos (sx, sy), (sx+sw, sy), (sx+sw, sy+sh), (sx, sy+sh).
  - \* O retângulo de destino é o retângulo [no interior do canvas] cujos cantos são os quatro pontos (dx, dy), (dx+dw, dy), (dx+dw, dy+dh), (dx, dy+dh).

*Desenhando imagens:*

<https://html.spec.whatwg.org/multipage/embedded-content.html#images>.

Figura 96 – Utilizando `drawImage()` - coordenadas



Fonte: <https://diveintohtml5.com.br/canvas.html>

- Criando como objeto: `new Image()`

```
var cat = new Image();
cat.src = "images/cat.png";
cat.onload = function() {
  context.drawImage(cat, 0, 0);
};
```

- O 3º e 4º parâmetros opcionais do método `drawImage()` controlam a escala da imagem.

Figura 97 – Código Javascript

```
20 // Desenhando uma seta na Horizontal
21 ctx.beginPath();
22 ctx.moveTo(0, 40);
23 ctx.lineTo(240, 40);
24 ctx.moveTo(260, 40);
25 ctx.lineTo(500, 40);
26 ctx.moveTo(495, 35);
27 ctx.lineTo(500, 40);
28 ctx.lineTo(495, 45);
29 // Desenhando uma seta na Vertical
30 ctx.moveTo(60, 0);
31 ctx.lineTo(60, 153);
32 ctx.moveTo(60, 173);
33 ctx.lineTo(60, 375);
34 ctx.moveTo(65, 370);
35 ctx.lineTo(60, 375);
36 ctx.lineTo(55, 370);
37
38 ctx.strokeStyle = "#000";
39 ctx.stroke();
40
```

Fonte: <https://diveintohtml5.com.br/canvas.html>

Figura 98 – Código Javascript

```
41 //Desenhando Retângulo
42 ctx.fillRect(160, 125, 150, 100);
43 ctx.strokeStyle = "#000";
44 ctx.stroke();
45
46 //Desenhando x, y, e as coordenadas
47 ctx.font = "bold 12px sans-serif";
48 ctx.fillText("x", 248, 43);
49 ctx.fillText("y", 58, 165);
50
51 ctx.textBaseline = "top";
52 ctx.fillText("( 0 , 0 )", 8, 5);
53
54 ctx.textAlign = "right";
55 ctx.textBaseline = "bottom";
56 ctx.fillText("( 500 , 375 )", 492, 370);
57
```

Fonte: <https://diveintohtml5.com.br/canvas.html>

Figura 99 – Código Javascript

```
58 // Criando gradiente
59 var my_gradient = ctx.createLinearGradient(0, 0, 300, 0);
60 my_gradient.addColorStop(0, "black");
61 my_gradient.addColorStop(1, "white");
62 ctx.fillStyle = my_gradient;
63 ctx.fillRect(100, 350, 150, 225);
64
65 // Invertendo
66 /*
67 var my_gradient = ctx.createLinearGradient(0, 0, 0, 225);
68 my_gradient.addColorStop(0, "black");
69 my_gradient.addColorStop(1, "white");
70 ctx.fillStyle = my_gradient;
71 ctx.fillRect(0, 0, 300, 225);
72 */
73
74 //Utilizando drawImage()
75 var cat = document.getElementById("cat");
76 ctx.drawImage(cat, 0, 0);
77
```

Fonte: <https://diveintohtml5.com.br/canvas.html>

*Demo canvas:* <http://www.canvasdemos.com/>.

# 14 Atividades

## 14.1 Atividades Capítulo 04

### 14.1.1 Atividade 01

- Elaborar um arquivo com extensão .html de acordo com o modelo abaixo, e imagem fornecidos.

#### 14.1.1.1 Imagem

Figura 100 – Imagem para desenvolver Atividade01



Fonte: <https://developers.google.com/web/tools/lighthouse/>

#### 14.1.1.2 Modelo

# Lighthouse

By *Developers Google*

Fev 08, 2017

\*\*\***Lighthouse is an open-source!**\*\*\* Automated tool for improving the quality of your web apps. Run it as a Chrome Extension or from the command line.



Lighthouse can be run as a Chrome Extension, from the command line, or used programmatically as a Node module. You give Lighthouse a URL that you want to audit, it runs a barrage of tests against the page, and then it generates a report on how well the page did. “From here you can use the failing tests” as indicators on what you can do to improve your app.

## Setting up Lighthouse

There are two primary ways to run Lighthouse, as a Chrome Extension, or as a command line tool. The Chrome Extension provides a more user-friendly interface for reading reports. The command line tool enables you to integrate Lighthouse into continuous integration systems.<sup>1</sup> Install the Chrome Extension [To install the extension, do the following](#) Or Chrome Extension.

## Chrome Extension

Go to the page that you want to audit “Click on the Lighthouse icon (Lighthouse icon)in the Chrome toolbar. You should see a ” menu. If you want to only run a subset of the audits, click the Options button and disable the audits that you don't care about. Scroll down and press OK to confirm your changes [Extension](#) Click the Generate report button to run Lighthouse's tests against the currently-open page.

The Lighthouse Viewer is a useful way for viewing and sharing reports, online. [There are several ways to view a report:](#)

*The game of Go has long been viewed as the most challenging of classic games for artificial intelligence due to its enormous search space and the difficulty of evaluating board positions and moves.*

The Viewer also lets you share reports with others. Reports can be shared by clicking the share icon in the top right corner and signing in to Github. “Reports are stashed as secret gists in your account so you can easily delete or update the report later on.” Using Github for data storage also means free version control.

---

<sup>1</sup>Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 3.0 License, and code samples are licensed under the Apache 2.0 License. For details, see our Site Policies. Java is a registered trademark of Oracle and/or its affiliates

## 14.2 Atividades Capítulo 05

### 14.2.1 Atividade 01

- Elaborar um arquivo com extensão .html de acordo com o solicitado abaixo. Criar um arquivo com extensão .css e inserir o link apropriado no html para utilizá-lo.
- Aplicar os conceitos de HTML semântico.
- Aplicar os conceitos de seletores de html, classes, identificadores.
- Aplicar a propriedade de formatação de cores.
- Aplicar os conceitos de hiperlinks.
- Utilize imagens, reservas de espaços de imagens.

## 14.3 Atividades Capítulo 06

### 14.3.1 Atividade 01

- Elaborar um arquivo com extensão .html de acordo com o solicitado abaixo. Criar um arquivo com extensão .css e inserir o link apropriado no html para utilizá-lo.
- Aplicar os conceitos adquiridos no capítulo anterior.
- Aplicar os conceitos à respeito de formatação de propriedades de fontes e textos.

## 14.4 Atividades Capítulo 07

### 14.4.1 Atividade 01

- Elaborar um arquivo com extensão .html de acordo com o solicitado abaixo. Criar um arquivo com extensão .css e inserir o link apropriado no html para utilizá-lo.
- Aplicar os conceitos adquiridos no capítulo anterior.
- Aplicar os conceitos à respeito de formatação de propriedades de CSS de layout, utilizar flexbox, position, etc.

#### 14.4.1.1 Modelo

# Lighthouse



## \*\*\*Lighthouse is an open-source!\*\*\*

*Automated tool for improving the quality of your web apps. Run it as a Chrome Extension or from the command line.*

Lighthouse can be run as a Chrome Extension, from the command line, or used programmatically as a Node module. You give Lighthouse a URL that you want to audit, it runs a barrage of tests against the page, and then it generates a report on how well the page did. "From here you can use the failing tests" as indicators on what you can do to improve your app.

### Chrome Extension

Go to the page that you want to audit "Click on the Lighthouse icon (Lighthouse icon) in the Chrome toolbar. You should see a " menu. If you want to only run a subset of the audits, click the Options button and disable the audits that you don't care about. Scroll down and press OK to confirm your changes [Extension](#) Click the Generate report button to run Lighthouse's tests against the currently-open page.

### Setting up Lighthouse

There are two primary ways to run Lighthouse, as a Chrome Extension, or as a command line tool. The Chrome Extension provides a more user-friendly interface for reading reports. The command line tool enables you to integrate Lighthouse into continuous integration systems.<sup>1</sup> [Install the Chrome Extension](#) To install the extension, do the following. Or [Chrome Extension](#).

The Lighthouse Viewer is a useful way for viewing and sharing reports, online. [There are several ways to view a report](#):

*The game of Go has long been viewed as the most challenging of classic games for artificial intelligence due to its enormous search space and the difficulty of evaluating board positions and moves.*

The Viewer also lets you share reports with others. Reports can be shared by clicking the share icon in the top right corner and signing in to Github. "Reports are stashed as secret gists in your account so you can easily delete or update the report later on." Using Github for data storage also means free version control.

By *Developers Google*

Fev 08, 2017

<sup>1</sup>Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 3.0 License, and code samples are licensed under the Apache 2.0 License. For details, see our Site Policies. Java is a registered trademark of Oracle and/or its affiliates

## 14.5 Atividades Capítulo 08

### 14.5.1 Atividade 01

- Desenvolver uma página que contenha:
  - Uma receita completa que contenha, imagem, lista de ingredientes, passo a passo, rendimento, tempo de elaboração, serve quantas porções.
  - Uma lista de compras que contenha os itens a serem comprados, preços estimados, e locais possíveis para compra com os respectivos links.
  - Duas tabelas, sendo que uma delas contenha informações nutricionais dos alimentos (similar aos rótulos dos produtos), e a outra, contendo foto de pratos prontos e breves descrições sobre cada um.
- Lembre-se que a apresentação destes itens devem ser bem dispostos, com margens, padding, cores, textos, fontes, ícones.
- Utilize os recursos de formatação CSS para que fique bonito, apresentável.

## 14.6 Atividades Capítulo 09

### 14.6.1 Atividade 01

- Desenvolver uma página que contenha:
  - Um formulário de registro de usuário, contendo dados pessoais, tal como nome completo, pai, mãe, telefones, rg, cpf etc. Identifique como form1.
  - Um formulário de registro de usuário, contendo dados físicos, tal como altura, peso, tipo sanguíneo etc. Identifique como form2.
  - Um formulário de pesquisa de satisfação, contendo uma pesquisa de satisfação. Identifique como form3. Pesquisa na web, algumas perguntas que poderiam ser feitas.
- Lembre-se de aplicar a formatação CSS, criando áreas delimitadas para cada formulário, utilize os recursos de CSS, seja criativo.

## 14.7 Atividades Capítulo 10

### 14.7.1 Atividade 01

- Desenvolver seu currículum online, que contenha:
  - Experiência Profissional, Formação Acadêmica, Cursos, Interesses, Habilidades Profissionais, Informações de Contato, Breve descrição do seu perfil, Links para redes sociais e outros, Áreas de Atuação.
  - Quem é você, seu perfil contendo uma foto, seus links de facebook, twitter, gmail, linkedin, icq, skype e outros. Utilize os ícones dos sites mencionados abaixo.
  - Um dropdown de botões com as opções de acesso da própria página e para links externos.
  - Ícones, desenhos, imagens, textos.
  - Aplique Bootstrap, e se desejar personalize seu CSS, criando um novo arquivo .css e linkando no html logo após o link do bootstrap.
  - Utilize os sites de apoio:
    - \* Ícones: <http://fontawesome.io/icons/>
    - \* Ideias de Layout de CV: <https://www.freshdesignweb.com/vcard-resume-templates/>

- Use um template básico para começar:

Figura 101 – Template Básico sem comentários

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3      <meta http-equiv="X-UA-Compatible" content="IE=edge">
4      <meta charset="utf-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1">
6      <title>Bootstrap 101 Template</title>
7      <link href="css/bootstrap.min.css" rel="stylesheet">
8      <link rel="stylesheet" type="text/css" href="css/style.css">
9  </head>
10 <body>
11     <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
12     <script src="js/bootstrap.min.js"></script>
13 </body>
14 </html>
```

Fonte: <http://getbootstrap.com/getting-started/#template>

## 14.8 Atividades Capítulo 11

### 14.8.1 Atividade 01

- Desenvolver uma página que contenha elementos animados por CSS. Seja criativo!

## 14.9 Atividades Capítulo 13

### 14.9.1 Atividade 01

- Desenvolver uma página que contenha elementos canvas. Faça uma pesquisa em sites com uso de canvas e crie sua aplicação. Utilize suas horas em sala de aula para desenvolver uma página bem divertida.

## 15 Referências

- TEAGUE J. C. - Visual Quick Star Guide CSS3