TREINAMENTOS

Desenvolvimento Web com HTML, CSS e Javascript

Desenvolvimento Web com HTML, CSS e Javascript

25 de novembro de 2013
As apostilas atualizadas estão disponíveis em **www.k19.com.br**

Esta apostila contém:

- 289 exercícios de fixação.
- 164 exercícios complementares.
- 56 questões de prova.

Sumário ii

Sobre a K19 1

Seguro Treinamento 2

Termo de Uso 3

Cursos 4

1 Introdução 1
1.1 Sites e Aplicações Web . . . 1
1.2 Navegadores e Dispositivos . . . 3
1.3 Web Servers e HTTP . . . 4
1.4 Domínios e endereços IP . . . 5
1.5 DNS (Domain Name System) . . . 9
1.6 Serviços de Hospedagem . . . 10
1.7 SEO (Search Engine Optimization) . . . 11
1.8 Arquitetura Web . . . 11
1.9 Ativando o IIS e ASP Classic no Windows 7 . . . 12
1.10 Instalação do Apache HTTP Server no Ubuntu 13.04 . . . 16
1.11 Apache HTTP Server no Mac OS X Mountain Lion . . . 16
1.12 Resumo do Capítulo . . . 17
1.13 Prova . . . 18

2 HTML 21
2.1 Introdução . . . 21
2.2 Estrutura Básica . . . 21
2.3 Editores HTML . . . 25
2.4 Ferramentas de Desenvolvimento Web . . . 25
2.5 Exercícios de Fixação . . . 26
2.6 Semântica HTML . . . 32
2.7 Títulos . . . 34
2.8 Exercícios de Fixação . . . 35
2.9 Parágrafos . . . 38
2.10 Character Entities . . . 39
2.11 Exercícios de Fixação . . . 40
2.12 Texto . . . 41
2.13 Exercícios de Fixação . . . 52
2.14 Listas . . . 58
2.15 Exercícios de Fixação . . . 62
2.16 Iframes . . . 68
2.17 Links . . . 69
2.18 Exercícios de Fixação . . . 70
2.19 Âncoras . . . 71
2.20 Exercícios de Fixação . . . 73
2.21 Imagens . . . 75
2.22 URLs absolutas e relativas . . . 76
2.23 Exercícios de Fixação . . . 78
2.24 Tabelas . . . 78

2.25 Exercícios de Fixação . . . 83
2.26 Formulários . . . 85
2.27 Exercícios de Fixação . . . 91
2.28 Caixas de entrada específicas . . . 93
2.29 Exercícios de Fixação . . . 97
2.30 Checkboxes e Radios . . . 99
2.31 Seleção de cores . . . 100
2.32 Exercícios de Fixação . . . 101
2.33 Botões . . . 103
2.34 Exercícios de Fixação . . . 105
2.35 Drop-down list . . . 106
2.36 Exercícios de Fixação . . . 108
2.37 Fieldset . . . 109
2.38 Exercícios de Fixação . . . 110
2.39 Autocomplete . . . 111
2.40 Exercícios de Fixação . . . 112
2.41 Campos ou botões desabilitados . . . 113
2.42 Campos fixos . . . 114
2.43 Exercícios de Fixação . . . 114
2.44 Validação . . . 115
2.45 Exercícios de Fixação . . . 117
2.46 Partes de um documento HTML . . . 118
2.47 Divisão de conteúdo . . . 123
2.48 Exercícios de Fixação . . . 123
2.49 Agrupando elementos . . . 124
2.50 Erro: Fechamento inadequado das tags . . . 124
2.51 Erro: Imagens sem texto alternativo . . . 124
2.52 Erro: Criar listas com o elemento br . . . 125
2.53 Exercícios Complementares . . . 125
2.54 Resumo do Capítulo . . . 133
2.55 Prova . . . 135

3 CSS **141**
3.1 Box Model . . . 141
3.2 Regras . . . 143
3.3 Aplicando CSS ao HTML . . . 144
3.4 Comentários . . . 145
3.5 Chrome DevTools . . . 145
3.6 Exercícios de Fixação . . . 150
3.7 Cores . . . 154
3.8 Unidades de medida . . . 155
3.9 Backgrounds . . . 156
3.10 Exercícios de Fixação . . . 163
3.11 Textos . . . 164
3.12 Exercícios de Fixação . . . 170
3.13 Fontes . . . 171
3.14 Exercícios de Fixação . . . 176
3.15 Listas . . . 177
3.16 Exercícios de Fixação . . . 178

3.17 Bordas . . . 179
3.18 Exercícios de Fixação . . . 184
3.19 Outline . . . 185
3.20 Exercícios de Fixação . . . 187
3.21 Sombras . . . 188
3.22 Exercícios de Fixação . . . 190
3.23 Margens . . . 191
3.24 Exercícios de Fixação . . . 194
3.25 Altura e Largura . . . 195
3.26 Exercícios de Fixação . . . 196
3.27 Display e Visibilidade . . . 197
3.28 Exercícios de Fixação . . . 200
3.29 Opacidade . . . 201
3.30 Exercícios de Fixação . . . 202
3.31 Posicionamento . . . 203
3.32 Exercícios de Fixação . . . 207
3.33 Overflow e clip . . . 209
3.34 Exercícios de Fixação . . . 211
3.35 Transformações . . . 213
3.36 Exercícios de Fixação . . . 215
3.37 Transições . . . 217
3.38 Exercícios de Fixação . . . 221
3.39 Animações . . . 222
3.40 Exercícios de Fixação . . . 225
3.41 Seletores . . . 228
3.42 Exercícios de Fixação . . . 232
3.43 Media Queries . . . 238
3.44 Exercícios de Fixação . . . 241
3.45 Sprites . . . 242
3.46 Exercícios de Fixação . . . 243
3.47 Gradientes (Conteúdo Extra) . . . 244
3.48 Herança (Conteúdo Extra) . . . 248
3.49 box-sizing (Conteúdo Extra) . . . 249
3.50 Design Responsivo (Conteúdo Extra) . . . 249
3.51 Exercícios Complementares . . . 253
3.52 Resumo do Capítulo . . . 295
3.53 Prova . . . 297

4 JavaScript **305**
4.1 Aplicando JavaScript ao HTML . . . 305
4.2 Carregamento . . . 305
4.3 Chrome DevTools . . . 306
4.4 Exercícios de Fixação . . . 308
4.5 Variáveis . . . 312
4.6 Operadores . . . 312
4.7 Controle de fluxo . . . 319
4.8 Exercícios de Fixação . . . 320
4.9 Objetos . . . 324
4.10 Exercícios de Fixação . . . 328

4.11 Funções . . . 334
4.12 Exercícios de Fixação . . . 336
4.13 Arrays . . . 338
4.14 Strings . . . 341
4.15 Exercícios de Fixação . . . 342
4.16 DOM - Document Object Model . . . 350
4.17 Exercícios de Fixação . . . 352
4.18 Eventos . . . 353
4.19 Exercícios de Fixação . . . 358
4.20 Web Storage . . . 360
4.21 Exercícios de Fixação . . . 361
4.22 History . . . 362
4.23 Exercícios de Fixação . . . 364
4.24 Geolocalização . . . 365
4.25 Exercícios de Fixação . . . 365
4.26 Alarmes (Conteúdo Extra) . . . 366
4.27 Exercícios Complementares . . . 367
4.28 Resumo do Capítulo . . . 369
4.29 Prova . . . 370

5 jQuery **375**
5.1 Introdução . . . 375
5.2 Exercícios de Fixação . . . 376
5.3 Eventos . . . 379
5.4 Exercícios de Fixação . . . 382
5.5 Seletores . . . 385
5.6 Exercícios de Fixação . . . 392
5.7 Efeitos e Animações . . . 393
5.8 Exercícios de Fixação . . . 394
5.9 Manipulação . . . 396
5.10 Exercícios de Fixação . . . 403
5.11 Mais métodos . . . 406
5.12 AJAX . . . 408
5.13 Exercícios de Fixação . . . 410
5.14 Exercícios Complementares . . . 411
5.15 Resumo do Capítulo . . . 417
5.16 Prova . . . 418

A Projeto **421**
A.1 Exercícios de Fixação . . . 421

B Quizzes **435**

C Respostas **437**

Sobre a K19

A K19 é uma empresa especializada na capacitação de desenvolvedores de software. Sua equipe é composta por profissionais formados em Ciência da Computação pela Universidade de São Paulo (USP) e que possuem vasta experiência em treinamento de profissionais para área de TI.

O principal objetivo da K19 é oferecer treinamentos de máxima qualidade e relacionados às principais tecnologias utilizadas pelas empresas. Através desses treinamentos, seus alunos tornam-se capacitados para atuar no mercado de trabalho.

Visando a máxima qualidade, a K19 mantém as suas apostilas em constante renovação e melhoria, oferece instalações físicas apropriadas para o ensino e seus instrutores estão sempre atualizados didática e tecnicamente.

Seguro Treinamento

Na K19 o aluno faz o curso quantas vezes quiser!

Comprometida com o aprendizado e com a satisfação dos seus alunos, a K19 é a única que possui o Seguro Treinamento. Ao contratar um curso, o aluno poderá refazê-lo quantas vezes desejar mediante a disponibilidade de vagas e pagamento da franquia do Seguro Treinamento.

As vagas não preenchidas até um dia antes do início de uma turma da K19 serão destinadas ao alunos que desejam utilizar o Seguro Treinamento. O valor da franquia para utilizar o Seguro Treinamento é 10% do valor total do curso.

Termo de Uso

Termo de Uso

Todo o conteúdo desta apostila é propriedade da K19 Treinamentos. A apostila pode ser utilizada livremente para estudo pessoal . Além disso, este material didático pode ser utilizado como material de apoio em cursos de ensino superior desde que a instituição correspondente seja reconhecida pelo MEC (Ministério da Educação) e que a K19 seja citada explicitamente como proprietária do material.

É proibida qualquer utilização desse material que não se enquadre nas condições acima sem o prévio consentimento formal, por escrito, da K19 Treinamentos. O uso indevido está sujeito às medidas legais cabíveis.

Conheça os nossos cursos

K01- Lógica de Programação

K02 - Desenvolvimento Web com HTML, CSS e JavaScript

K03 - SQL e Modelo Relacional

K11 - Orientação a Objetos em Java

K12 - Desenvolvimento Web com JSF2 e JPA2

K21 - Persistência com JPA2 e Hibernate

K22 - Desenvolvimento Web Avançado com JFS2, EJB3.1 e CDI

K23 - Integração de Sistemas com Webservices, JMS e EJB

K41 - Desenvolvimento Mobile com Android

K51 - Design Patterns em Java

K52 - Desenvolvimento Web com Struts

K31 - C# e Orientação a Objetos

K32 - Desenvolvimento Web com ASP.NET MVC

www.k19.com.br/cursos

INTRODUÇÃO

CAPÍTULO 1

Sites e Aplicações Web

Normalmente, as pessoas utilizam o termo **site** quando se referem a blogs, sites de notícias, sites institucionais, portais, lojas virtuais, entre outros. Já a denominação **aplicação web** é muito utilizada para sistemas de gestão empresarial que são acessados através de navegadores (browsers).

Não há uma definição exata que diferencie claramente sites e aplicações web. Alguns defendem que os sites são *read-only* (somente leitura) enquanto as aplicações web são *read-write* (leitura e escrita). Nessa definição, os sites apenas fornecem conteúdo enquanto as aplicações web podem fornecer e/ou receber conteúdo.

Seguindo essa linha de raciocínio, algumas pessoas gostam de utilizar o grau de interatividade com os usuários para classificar como site ou aplicação web. Nessa visão, aplicações web seriam mais interativas e os sites menos interativos. Contudo, não há uma medida definida para calcular esse grau de interatividade e com isso aplicar de forma objetiva essa classificação. Dessa forma, essa diferenciação, na prática, é subjetiva.

Outras pessoas preferem utilizar o critério do *propósito* para classificar como site ou aplicação web. Por exemplo, se o propósito é divulgar as informações de uma empresa, os dados de um produto, as notícias de um determinado assunto, utiliza-se o termo site. Se o propósito é criar uma ferramenta para controlar as atividades administrativas de uma organização, utiliza-se a denominação aplicação web.

Utilizando o critério do *propósito* para classificar como site ou aplicação web, podemos concluir que, geralmente, os sites necessitam de uma interface mais atrativa pois normalmente estão "vendendo" alguma coisa ou alguma ideia. Por outro lado, na maior parte dos casos, o mais importante para as aplicações web é possuir uma interface fácil de utilizar.

Independentemente do critério de classificação utilizado, os tópicos discutidos nesse treinamento são importantes tanto para o desenvolvimento de sites quanto para o desenvolvimento de aplicações web. Portanto, do nosso ponto de vista, essas diferenças não serão tão relevantes.

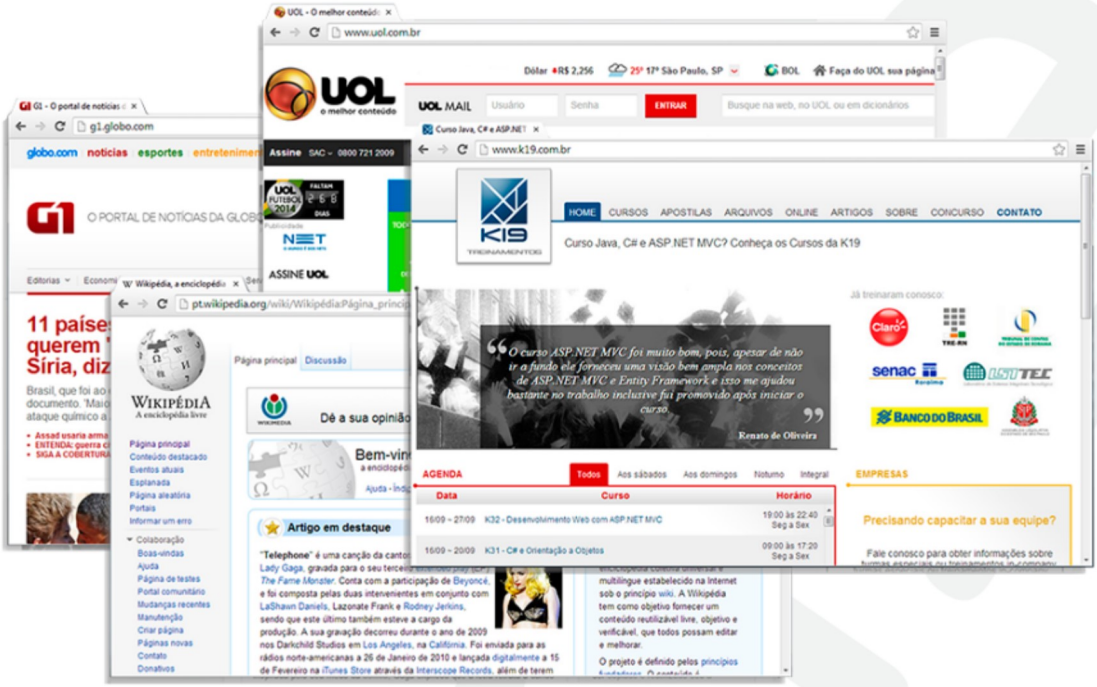


Figura 1.1: Sites

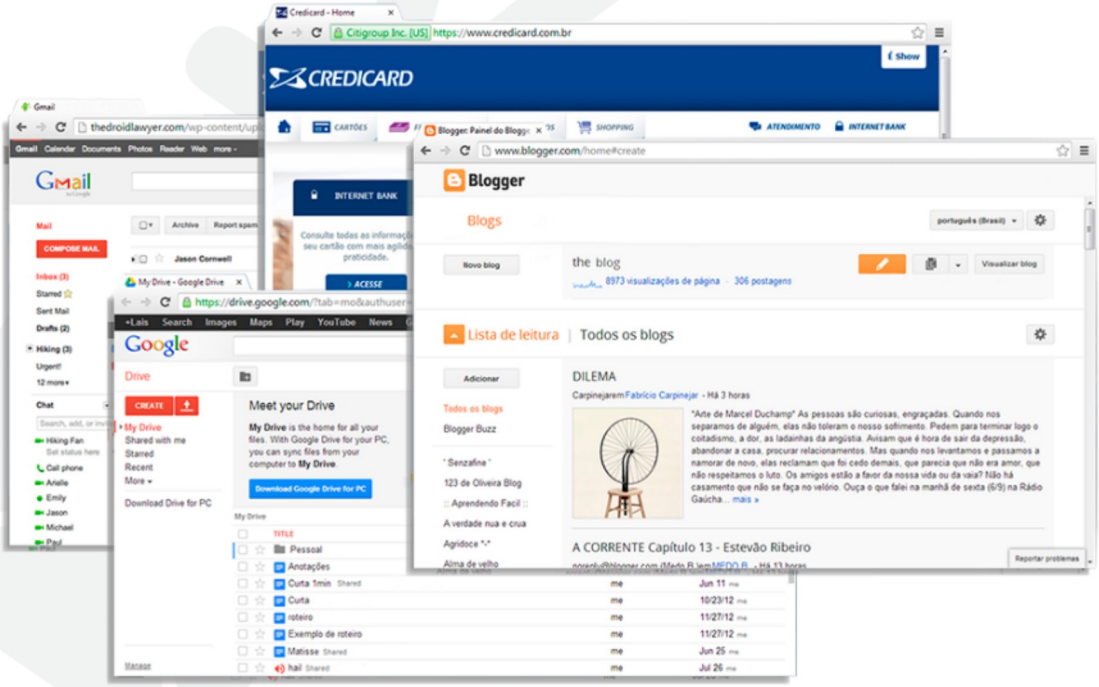


Figura 1.2: Aplicações Web

Navegadores e Dispositivos

As pessoas acessam sites e aplicações web através de navegadores (browsers) como **Chrome**, **Firefox**, **Internet Explorer** e **Safari**. Normalmente, esses navegadores possuem algumas diferenças na forma de exibir as páginas web aos usuários. Antigamente, essas diferenças eram maiores. Com o passar do tempo, os navegadores ficaram cada vez mais parecidos nesse aspecto. Contudo, os desenvolvedores web ainda devem tomar cuidado com essas diferenças.

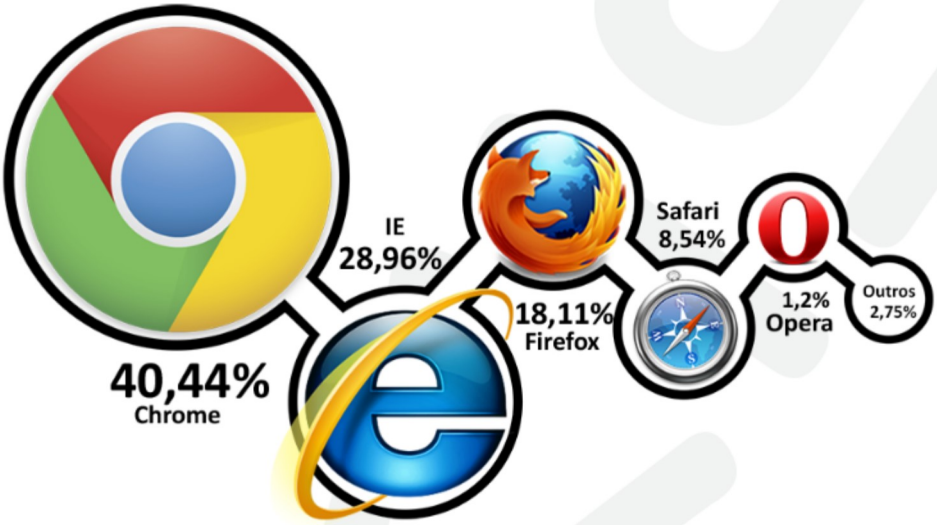


Figura 1.3: Índice de utilização dos navegadores em Outubro de 2013 (fonte `http.//gs.statcounter.com/`*)*

Outro problema de compatibilidade importante é causado pela grande variedade de dispositivos que podem ser utilizados para acessar os sites e as aplicações web. Atualmente, as pessoas acessam os sites e as aplicações web através de computadores tradicionais, tablets, celulares, televisores, entre outros. Esses dispositivos possuem telas de tamanhos diferentes. Dessa forma, os desenvolvedores web devem considerar essas diferenças na criação das páginas web. Hoje em dia, fala-se muito de **design responsivo**. Os sites ou as aplicações web são ditos responsivos se eles estão preparados para diferentes tamanhos de tela.

Figura 1.4: Design Responsivo

Web Servers e HTTP

Os sites e as aplicações web são implantados em computadores conectados à **Internet** ou a uma rede privada qualquer (**Intranet**). Normalmente, os sites são implantados em computadores conectados à Internet pois assim poderão ser acessados praticamente de qualquer lugar do mundo. Por outro lado, as aplicações web, muitas vezes, são implantadas em computadores conectados a uma Intranet pois é comum ser necessário restringir o acesso externo à elas.

Figura 1.5: Internet

Figura 1.6: Intranet

Os computadores nos quais os sites e as aplicações web são implantados são chamados de **Web**

Servers. Quando acessamos uma página web através de um navegador, ele realiza uma requisição ao Web Server onde essa página está armazenada. Ao receber a resposta do Web Server com a página web solicitada, o navegador a exibe para nós. As mensagens de requisição e resposta trocadas entre o navegador e o web server são definidas pelo protocolo **HTTP**.

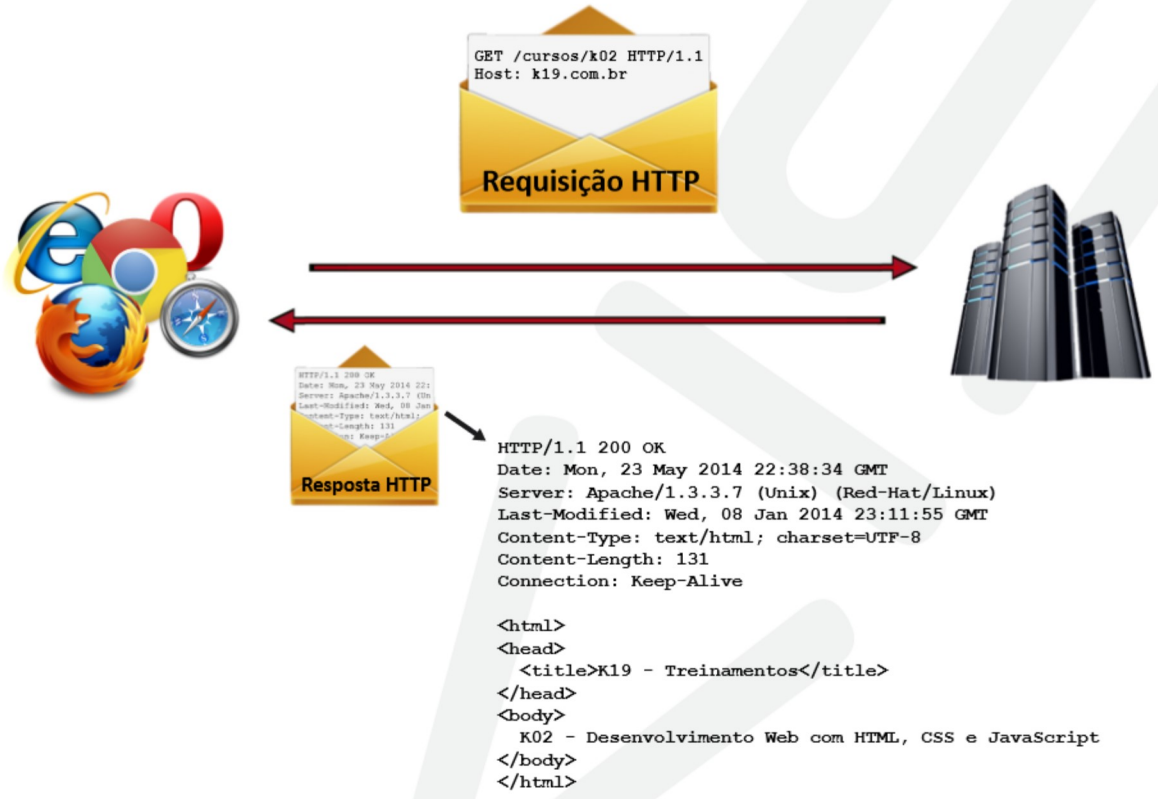


Figura 1.7: Interação entre os navegadores e os Web Servers

Considere um site ou uma aplicação web implantado em um Web Server que está conectado a uma rede. A princípio, qualquer navegador executando em um dispositivo conectado a essa rede pode realizar requisições a esse Web Server.

Alguns softwares especializados são utilizados para administrar os sites e as aplicações web implantadas nos Web Servers. Os principais são o **Apache HTTP Server** da **Apache Software Foundation** e o **Internet Information Services (IIS)** da **Microsoft**.

Domínios e endereços IP

Os dispositivos conectados a uma rede são identificados através de endereços formados por sequências de números. Esses endereços são chamados de **endereços IP**. Em uma Intranet, quem controla os endereços IP dos dispositivos conectados é a própria organização que administra essa Intranet. Por outro lado, os endereços IP dos dispositivos conectados à Internet são gerenciados pelos provedores de acesso (ISP).

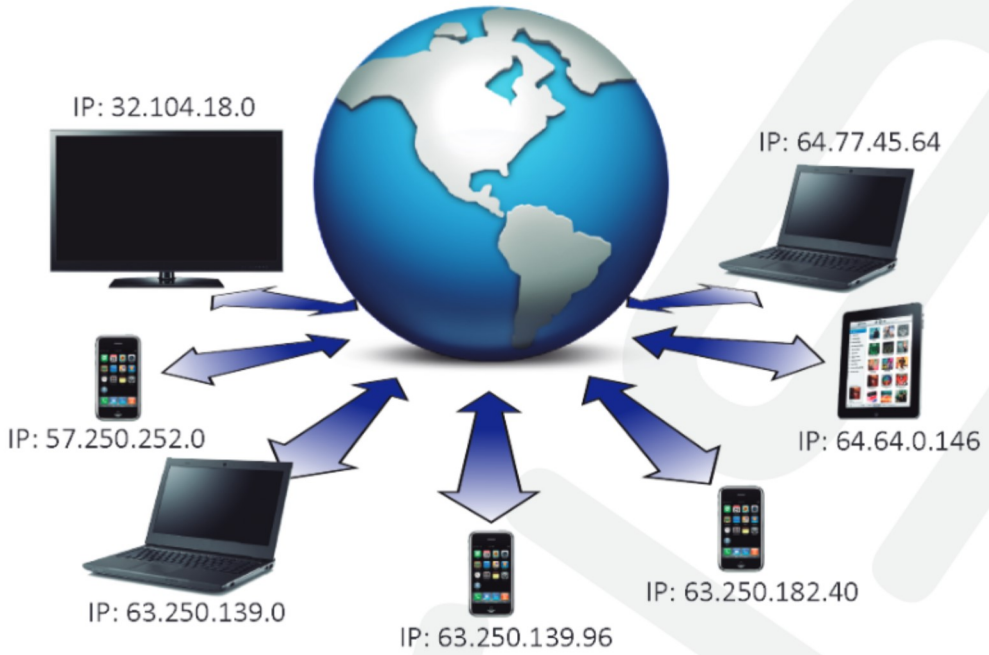


Figura 1.8: Endereços IP

A princípio, para acessar uma página de um site ou de uma aplicação web, devemos conhecer o endereço IP do Web Server que contém esse site ou essa aplicação web. Atualmente, o endereço IP do Web Server onde o site da K19 está implantado é **184.72.247.119**. Podemos utilizar esse endereço IP para acessar as páginas do site da K19. O endereço IP **200.144.183.244** está vinculado a um dos Web Servers onde o site da USP está implantado. Também podemos acessar as páginas do site da USP utilizando diretamente esse endereço IP. Analogamente, podemos acessar as páginas da **Wikipédia** utilizando o endereço IP **208.80.152.130**.

Figura 1.9: Acessando o Web Server da K19 pelo endereço IP

Figura 1.10: Acessando o Web Server da USP pelo endereço IP

Figura 1.11: Acessando o Web Server da Wikipédia pelo endereço IP

Se você tiver uma memória muito boa pode decorar os endereços IP dos sites que acessa com maior frequência. Mas, para maior parte das pessoas, seria impossível decorar tantos números. Para resolver esse problema, os endereços IP são associados a domínios. Veja alguns exemplos de domínios.

- www.k19.com.br
- www.usp.br
- www.wikipedia.org

As pessoas conseguem decorar ou deduzir os domínios bem mais facilmente do que os endereços IP.

Figura 1.12: Acessando o Web Server da K19 pelo domínio

Figura 1.13: Acessando o Web Server da USP pelo domínio

Figura 1.14: Acessando o Web Server da Wikipédia pelo domínio

Há uma outra vantagem importante dos domínios sobre os endereços IP. Em alguns casos, o endereço IP de um Web Server precisa ser alterado. Geralmente, essa modificação ocorre por motivos técnicos. Supondo que essa mudança ocorra, quem estiver acessando esse Web Server através do endereço IP antigo não conseguirá mais acessá-lo dessa forma. Por outro lado, quem estiver acessando esse Web Server através do domínio dele poderá continuar acessando da mesma forma pois esse domínio pode ser facilmente associado ao novo endereço IP.

Figura 1.15: Alteração do Web Server

Os domínios são controlados por organizações geralmente vinculadas ao governo. Por exemplo, os domínios **.br** são controlados e disponibilizados pelo **Registro de Domínios para a Internet no Brasil (registro.br)**.

DNS (Domain Name System)

Como vimos, podemos acessar um Web Server diretamente através do seu endereço IP ou indiretamente através de um domínio. Para utilizar a segunda abordagem, é necessário consultar um servidor **DNS** para "traduzir" o domínio desejado para o endereço IP correspondente. Basicamente, a tarefa dos servidores **DNS** é informar o endereço IP associado a um domínio.

Figura 1.16: Interação entre os navegadores e os servidores DNS

Há diversos servidores DNS que são públicos. Eis uma lista com alguns deles.

- OpenDNS (208.67.222.222 e 208.67.220.220)
- Google Public DNS (8.8.8.8 e 8.8.4.4)
- Level3 (209.244.0.3 e 209.244.0.4)

Serviços de Hospedagem

Uma organização pode possuir computadores atuando como Web Servers em sua própria infraestrutura ou na infraestrutura de empresas especializadas. Em determinadas situações, a primeira possibilidade é mais conveniente.

Por exemplo, normalmente, as instituições bancárias preferem manter os seus sites e as suas aplicações web em Web Servers dentro da sua própria infraestrutura. Essa abordagem permite um controle maior da comunicação entre essas instituições e os seus clientes. Mas, ela exige grandes investimentos. Para garantir que os seus Web Servers estejam sempre funcionando, essas instituições bancárias mantêm profissionais 24 horas por dia 7 dias por semana (24/7). Caso contrário, os sites ou as aplicações web dessas instituições podem ficar fora do ar e gerar grandes prejuízos.

Por outro lado, muitas vezes, é mais conveniente manter os Web Servers de uma organização na infraestrutura de uma empresa especializada. Dessa forma, a responsabilidade de mantê-los funcionando é delegada à essa empresa. Essa abordagem, geralmente, diminui os custos. Contudo, o controle é delegado a uma outra empresa. O serviço oferecido por essas empresas é denominado **serviço de hospedagem**. Eis uma lista de empresas que oferecem esse tipo de serviço.

- Amazon

- Localweb
- UOL

SEO (Search Engine Optimization)

Hoje em dia, a principal forma de encontrar um site é utilizar alguma ferramenta de busca. A mais importante delas, atualmente, é o Google. Essas ferramentas encontram os sites que possuem conteúdo relacionado às palavras chaves utilizadas nas buscas. Os sites mais relevantes são apresentados antes e com maior destaque.

Para responder rapidamente às consultas realizadas, as ferramentas de busca analisam previamente os sites. Esse processo de análise é chamado de **indexação**. A indexação é realizada por programas de computador que interagem com os sites para obter informações sobre o conteúdo de cada um deles. Esses programas são chamados de **robôs**.

Figura 1.17: Robôs

No desenvolvimento de um site, podemos aplicar técnicas que facilitam e melhoram a análise dos robôs das ferramentas de busca. Essas técnicas são desenvolvidas pelos especialistas em **SEO (Search Engine Optimization)**.

Normalmente, as técnicas de SEO são mais importantes para os sites do que para as aplicações web.

Arquitetura Web

Vamos utilizar um diagrama para apresentar uma visão geral da arquitetura web.

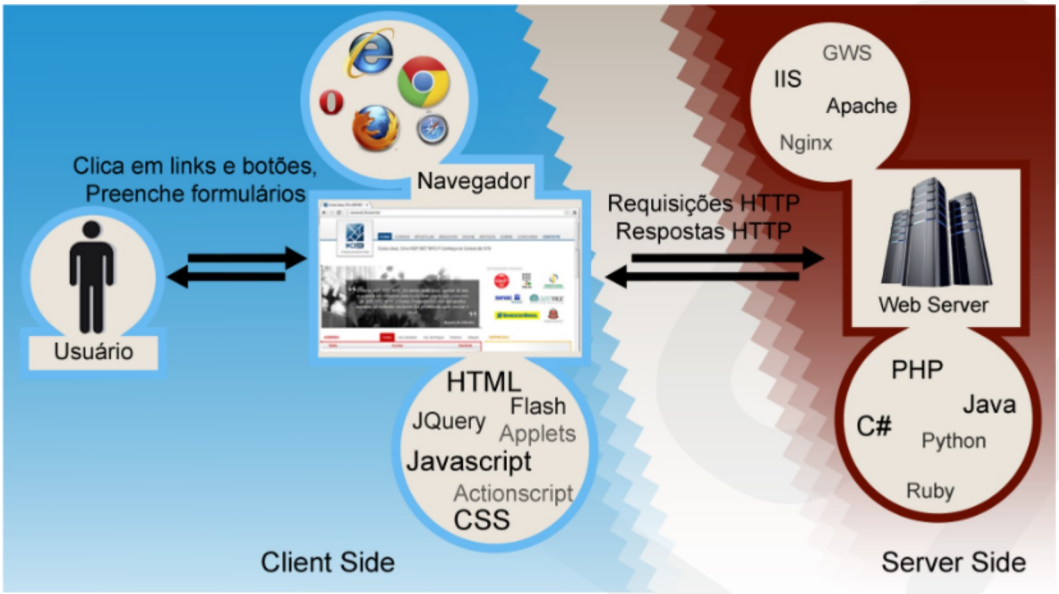


Figura 1.18: Diagrama geral da arquitetura web

Observe que o diagrama foi dividido em duas partes: **client side** e **server side**. Alguns profissionais se especializam no *client side* e outros no *server side*. Outra denominação possível para *client side* é **front-end** e para *server side* é **back-end**.

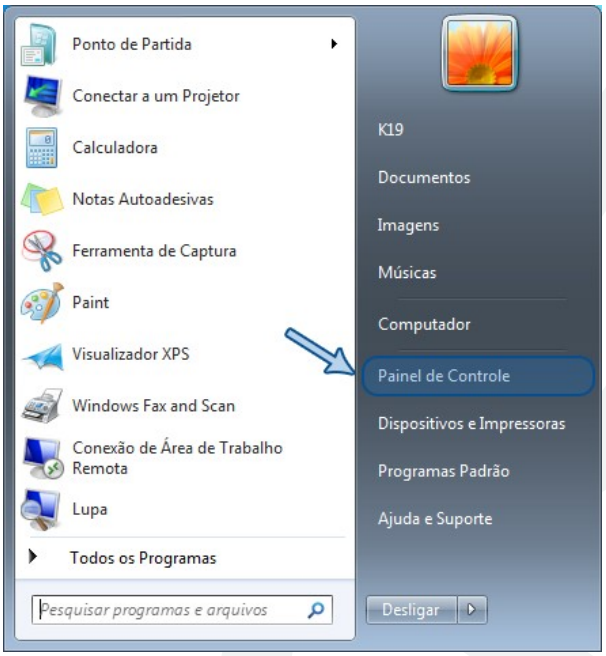
Por exemplo, um **web designer** deve ter bons conhecimentos do *client side*. Por outro lado, um **programador web** deve ter bons conhecimentos do *server side*.

Geralmente, os *web designers* não precisam ter conhecimento do *server side* mas tê-lo pode ser útil. Por outro lado, os *programadores web* precisam conhecer razoavelmente bem o *client side*. Nesse treinamento, estamos interessados especificamente no *client side*.

Ativando o IIS e ASP Classic no Windows 7

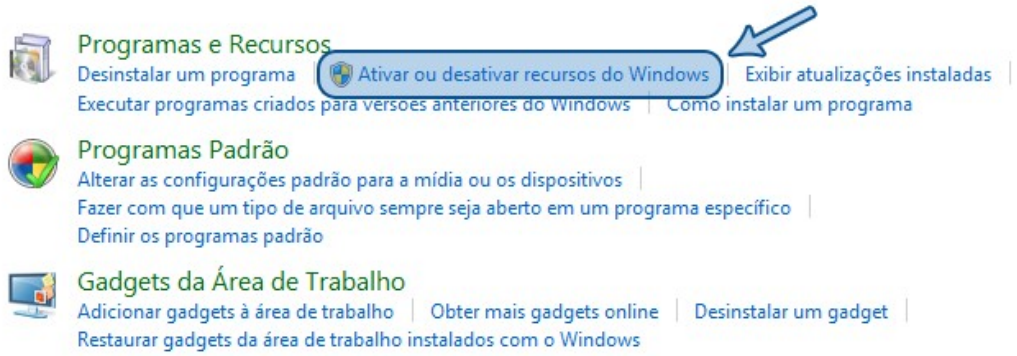
Se você quiser utilizar o Windows 7 para fazer os exercícios dos próximos capítulos, instale o IIS e o ASP Classic de acordo com os passos a seguir.

No menu *Iniciar*, clique em *Painel de Controle*.



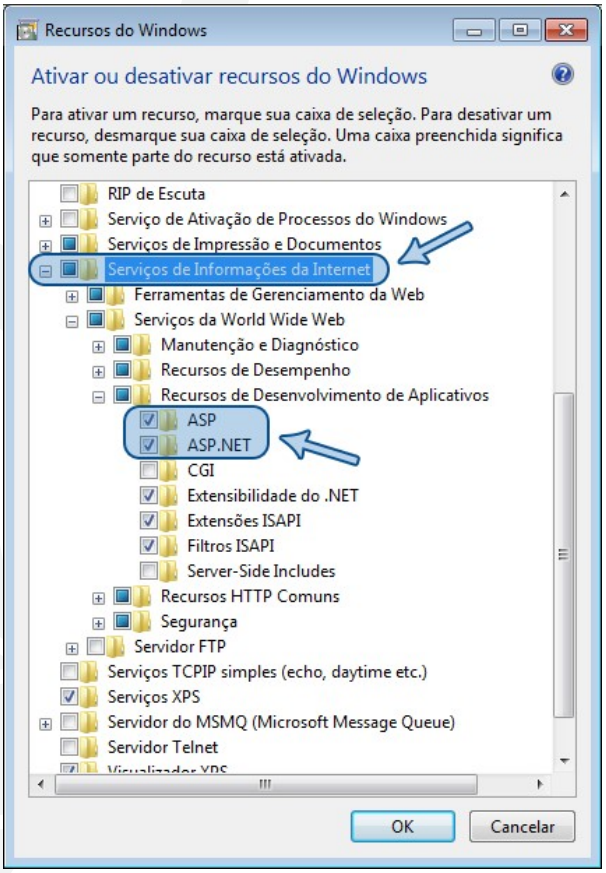
No *Painel de Controle*, clique em *Programas*.

Em *Programas*, clique em *Ativar ou desativar recursos do Windows*.

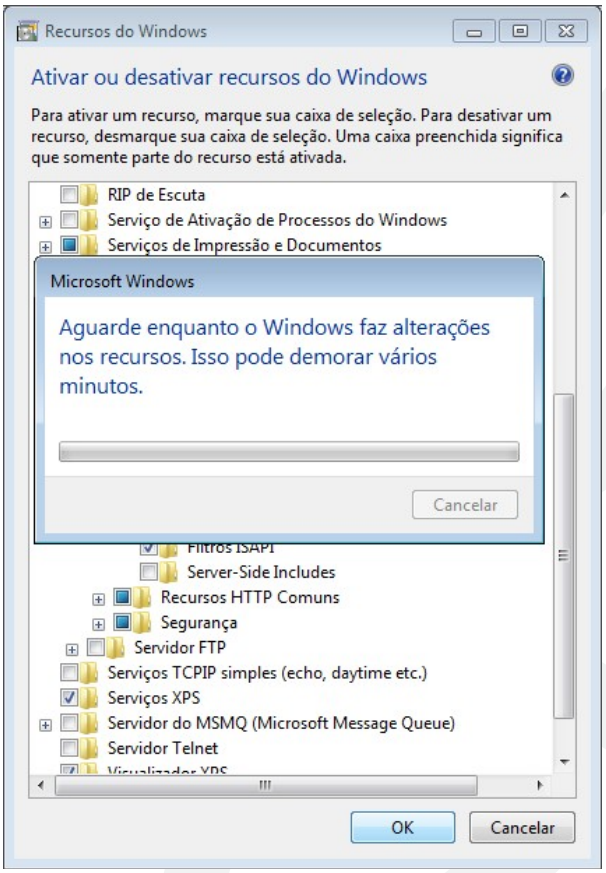


Aguarde o carregamento da lista de recursos que podem ser ativados ou desativados.

Selecione os recursos de acordo com a imagem a seguir e clique no botão *OK*.



Aguarde a ativação dos recursos selecionados.



Depois do término da ativação, abra um navegador e acesse o endereço `http.//localhost`.

Instalação do Apache HTTP Server no Ubuntu 13.04

Se você quiser utilizar o Ubuntu 13.04 para fazer os exercícios dos próximos capítulos, instale o Apache HTTP Server de acordo com os passos a seguir.

Abra um terminal e digite os seguintes comandos.

```
sudo apt-get update
```

```
sudo apt-get install apache2
```

```
sudo apt-get install php5 libapache2-mod-php5
```

```
sudo service apache2 restart
```

Configuração do Userdir

Configure o *userdir* do Apache HTTP Server seguindo passos abaixo.

```
sudo a2enmod userdir
```

```
sudo service apache2 restart
```

```
mkdir ~/public_html && chmod 0755 ~/public_html
```

Apache HTTP Server no Mac OS X Mountain Lion

Se você quiser utilizar o Mac OS X Mountain Lion para fazer os exercícios dos próximos capítulos, configure o Apache HTTP Server de acordo com os passos a seguir.

1. No diretório **/etc/apache2/users/**, crie um arquivo de configuração com o nome do seu usuário (ex: jonas.conf).

```
1 <Directory "/Users/nome_do_usuario/Sites/">
2   Options FollowSymLinks Indexes MultiViews
3   AllowOverride All
4   Order allow,deny
5   Allow from all
6 </Directory>
```

Código XML 1.1: nome_do_usuario.conf

2. Altere as permissões do arquivo criado anteriormente.

```
sudo chmod 644 nome_do_usuario.conf
```

3. Localize, no arquivo **/etc/apache2/httpd.conf**, a linha:

```
1 #LoadModule php5_module libexec/apache2/libphp5.so
```

4. Descomente a linha encontrada anteriormente.

```
LoadModule php5_module libexec/apache2/libphp5.so
```

5. Crie uma pasta chamada **Sites** na pasta do seu usuário.

```
mkdir ~/Sites
```

6. Inicie o Apache HTTP Server.

```
sudo apachectl start
```

7. Configure o Apache HTTP Server para iniciar automaticamente.

```
sudo defaults write /System/Library/LaunchDaemons/org.apache.httpd Disabled -bool false
```

Resumo do Capítulo

1. Não há um critério exato para diferenciar o que é um **site** e o que é uma **aplicação web**.

2. Os desenvolvedores web devem considerar as diferenças entre os navegadores (browsers) e os dispositivos (computadores, tablets, celulares, tvs, entre outros) na criação das páginas web de um site ou de uma aplicação web.

3. A **Internet** é a maior rede pública de computadores.

4. Uma rede privada de computadores é denominada **Intranet**

5. Os sites e as aplicações web são implantados em computadores denominados **Web Server**s.

6. As mensagens trocadas entre os navegadores e os Web Servers seguem o protocolo **HTTP**.

7. Os dispositivos conectados a uma rede de computadores são identificados através de **endereços IP**.

8. Para não ter que decorar endereços IP, podemos utilizar **domínios**.

9. A utilização de **domínios** facilita uma eventual troca do endereço IP de um Web Server.

10 A função principal de um servidor **DNS** é traduzir domínios para endereços IP.

11 Organizações de grande porte costuma implantar os seus sites e as suas aplicações web em Web Servers dentro da sua própria infraestrutura. Por outro lado, organizações menores, normalmente, preferem contratam empresas especializadas no **serviço de hospedagem**.

12 As técnicas de **SEO** são aplicadas com o intuito de melhorar o posicionamento de um site nos resultados das consultas realizadas nas ferramentas de busca.

13 Podemos dividir a arquitetura web em duas partes: **client side** e **server side**.

Prova

1 Qual alternativa está correta?

a) Existe uma divisão clara e exata entre o conceito de site e aplicação web.

b) Geralmente, os sites possuem maior grau de interatividade com o usuário do que as aplicações web.

c) As aplicações web não são utilizadas como ferramentas para controle administrativo de uma empresa.

d) Geralmente, os sites "vendem" algum produto ou alguma ideia.

e) Geralmente, as aplicações web "vendem" algum produto ou alguma ideia.

2 Qual alternativa está correta?

a) A forma de exibição das página web é padrão em todos os navegadores.

b) A forma de exibição das página web é padrão em todos os dispositivos.

c) O Design Responsivo trata de questões relacionadas à performance dos sites.

d) O Design Responsivo trata de questões relacionadas à performance dos navegadores.

e) Os sites ou as aplicações web são ditos responsivos se eles estão preparados para diferentes tamanhos de tela.

3 Qual alternativa está correta?

a) HTTP é protocolo de comunicação.

b) A Internet é uma rede de computadores privada.

c) Os sites e as aplicações web são implantados em Web Computers.

d) Aplicações web só podem ser implantadas em redes privadas.

e) Sites só podem ser implantados na Internet.

4 Qual alternativa está correta?

a) O endereço IP de um dispositivo determina precisamente a localização geográfica desse dispositivo.

b) O conteúdo de um site é determinado pelo domínio associado ao Web Server onde esse site está implantado.

c) Domínios são associados a endereços IP.

d) Para aplicações web devemos utilizar domínios especiais.

e) Todo domínio inicia com **www**.

5 Qual alternativa está correta?

a) A função principal de um servidor DNS (Domain Name System) é informar o nome do proprietário de um site.

b) O objetivo principal das empresas que fornecem serviços de hospedagem é vender domínios.

c) Técnicas de SEO melhoram a velocidade de carregamento de um site.

d) Técnicas de SEO melhoram o posicionamento dos sites nas ferramentas de busca.

e) O objetivo das empresas especializadas em SEO é oferecer serviços de hospedagem.

6 Qual alternativa está correta?

a) É comum dividir a arquitetura web em *left side* e *right side*.

b) É comum dividir a arquitetura web em *down side* e *up side*.

c) É comum dividir a arquitetura web em *outside* e *inside*.

d) É comum dividir a arquitetura web em *client side* e *server side*.

e) É comum dividir a arquitetura web em *front side* e *back side*.

Minha Pontuação ☐ Pontuação Mínima: 4 Pontuação Máxima: 6

HTML

CAPÍTULO 2

Introdução

Quando acessamos uma página web, estamos interessados na informação contida nessa página. Essa informação pode estar na forma de texto, imagem ou vídeo. O conteúdo de uma página web é definido com a linguagem HTML (HyperText Markup Language). HTML é uma linguagem de marcação originalmente proposta por Tim Berners-Lee no final da década de 1980. O objetivo do Tim Barners-Lee era criar um mecanismo simples que pudesse ser utilizado por qualquer pessoa que quisesse disseminar documentos científicos.

Desde sua proposta até os dias de hoje, a linguagem HTML sofreu diversas alterações. A cada versão, novos recursos são adicionados e problemas corrigidos. A versão mais atual da especificação da linguagem HTML é a 5. Essa versão ainda não foi finalizada, a previsão é que ela seja lançada em definitivo em Julho de 2014. Porém, os navegadores já implementam diversos recursos do HTML5. A especificação está disponível no endereço `http.//www.w3.org/TR/html5`.

As especificações da linguagem HTML são publicadas pelo **World Wide Web Consortium** mais conhecido por sua sigla **W3C**. Além do HTML, o W3C também é responsável por linguagens como o XML, o SVG e pela interface DOM (Document Object Model), por exemplo.

Estrutura Básica

Basicamente, um documento HTML é composto por elementos hierarquicamente organizados. Para inserir um elemento em um documento HTML, devemos utilizar as tags correspondentes a esse elemento. As tags são definidas com parênteses angulares (< e >). Os elementos podem possuir atributos e conteúdo. Os atributos são formados por nome e valor. Normalmente, os valores dos atributos são definidos dentro de aspas dupla e o conteúdo dos elementos é um texto ou outros elementos.

Figura 2.1: Estrutura básica de um documento HTML

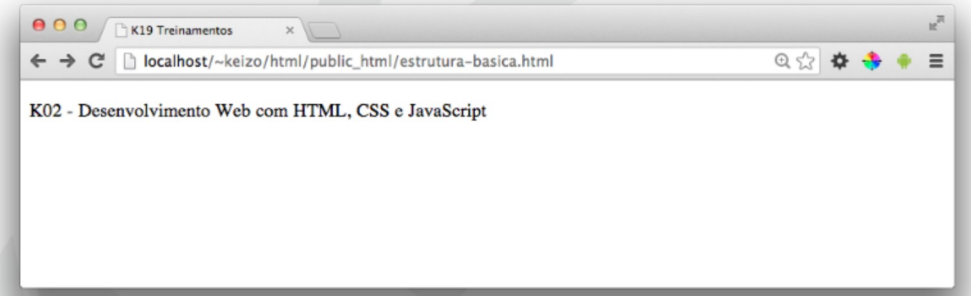


Figura 2.2: Estrutura básica de um documento HTML

Mais Sobre

A especificação do HTML5 define duas sintaxes distintas para produzir documentos HTML. A primeira é denominada **sintaxe HTML** e a segunda é denominada **sintaxe XHTML**. A maior parte dos autores recomendam a utilização da **sintaxe HTML**. Nos exemplos e nos exercícios dessa apostila, a preferência será pela **sintaxe HTML**.

Mais Sobre

De acordo com a especificação da linguagem HTML, alguns elementos são denominados **Normal Elements**. Esses elementos são abertos com uma tag e fechados com outra tag. Por exemplo, o elemento **p** é um **Normal Element**. Observe a utilização da tag de abertura e da tag de fechamento do elemento **p**.

```
1 <p>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</p>
```

Mais Sobre

Há também elementos denominados **Void Elements**. Esses elementos não possuem conteúdo. Na **sintaxe HTML**, esses elementos são abertos e fechados com apenas uma tag com ou sem o caractere "**/**". Um exemplo de **Void Element** é o elemento **br**.

```
1 <p>
2   K01 - Lógica de Programação<br>
3   K02 - Desenvolvimento Web com HTML, CSS e JavaScript<br/>
4   K03 - SQL e Modelo Relacional
5 </p>
```

Os **Void Elements** são: **area**, **base**, **br**, **col**, **embed**, **hr**, **img**, **input**, **keygen**, **link**, **meta**, **param**, **source**, **track**, **wbr**.

Na **sintaxe XHTML**, os **Void Elements** podem ser abertos e fechados na mesma tag ou em tags separadas. Contudo, o caractere "**/**" é obrigatório.

```
1 <p>
2   K01 - Lógica de Programação<br/>
3   K02 - Desenvolvimento Web com HTML, CSS e JavaScript<br></br>
4   K03 - SQL e Modelo Relacional
5 </p>
```

<!DOCTYPE>

Para um navegador exibir corretamente uma página web, devemos informar explicitamente o tipo do documento. O tipo do documento é informado com a declaração **<!DOCTYPE>**. Quando conveniente, discutiremos as principais diferenças entre os tipos de documentos mais importantes. Veja a declaração **<!DOCTYPE>** para os principais tipos de documentos.

```
1 <!DOCTYPE html>
```

Código HTML 2.4: HTML 5

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
2   "http://www.w3.org/TR/html4/strict.dtd">
```

Código HTML 2.5: HTML 4.01 Strict

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
2   "http://www.w3.org/TR/html4/loose.dtd">
```

Código HTML 2.6: HTML 4.01 Transitional

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
  "http.//www.w3.org/TR/html4/frameset.dtd">
```

Código HTML 2.7: HTML 4.01 Frameset

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http.//www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Código HTML 2.8: XHTML 1.0 Transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
  "http.//www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Código HTML 2.9: XHTML 1.0 Frameset

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http.//www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Código HTML 2.10: XHTML 1.1

<html>

Os elementos HTML, com exceção do **DOCTYPE**, devem ser inseridos no conteúdo do elemento **html**. Esse elemento é aberto com a tag **<html>**, fechado com a tag **</html>** e deve conter exatamente um elemento **head** seguido de exatamente um elemento **body**.

Diversos autores recomendam a utilização do atributo **lang**. Esse atributo indica a língua utilizada no documento HTML ou na maior parte dele. Algumas ferramentas de leitura ou de tradução de texto podem utilizar esse atributo para descobrir facilmente em qual língua os textos contidos no documento HTML ou na maior parte dele foram escritos.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

<head>

A principal função do elemento **head** é agrupar informações sobre o documento HTML (metainformação). São exemplos de metainformações: o encoding, a taxa de atualização, o autor, a descrição e as palavras chaves do documento HTML.

```
<head>
  <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
  <meta http-equiv="refresh" content="30">
  <meta name="author" content="Rafael Cosentino">
  <meta name="description" content="Cursos de Java e .NET">
  <meta name="keywords" content="curso java, curso C#">
</head>
```

O elemento **head** é aberto e fechado pelas tags **<head>** e **</head>** respectivamente. Obrigatoriamente, salvo algumas poucas exceções, o corpo do **head** deve conter exatamente uma ocorrência do elemento **title**. Esse elemento define o título do documento HTML.

```
<head>
  ...
  <title>K19 Treinamentos</title>
</head>
```

<body>

O conteúdo de uma página web deve ser definido no corpo do elemento **body**. Por exemplo, podemos inserir no corpo do **body** cabeçalhos, textos, listas, tabelas, entre outros componentes. Esse elemento é aberto pela tag **<body>** e fechado pela tag **</body>**.

```
<body>
  <h1>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</h1>
</body>
```

Comentários

Podemos adicionar comentários em um documento HTML dentro das tags **<!--** e **-->**. Os comentários são ignorados pelos navegadores.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <!-- corpo do head -->
  </head>
  <body>
    <!-- corpo do body -->
  </body>
</html>
```

Editores HTML

Documentos HTML podem ser criados em qualquer editor de texto básico. Contudo, para ganhar produtividade, podemos utilizar ferramentas com mais recursos. Há diversas opções de editores HTML, alguns gratuitos outros pagos. Nesse treinamento, utilizaremos o **Netbeans** como editor HTML. Na verdade, o Netbeans oferece muito mais recursos além do editor HTML. Contudo, agora, não estamos interessados nesses outros recursos. Você pode obter gratuitamente o Netbeans no endereço https.//netbeans.org/.

Ferramentas de Desenvolvimento Web

Hoje em dia, os principais navegadores possuem ferramentas para testar e depurar as páginas web e a interação com os Web Servers. Nos exemplos dessa apostila, utilizaremos **Chrome DevTools** do navegador **Chrome**. Outra ferramenta muito importante é o **Firebug** do navegador **Firefox**.

Exercícios de Fixação

1. Abra o **Netbeans** e crie um projeto chamado **html**.

Importante
No **Windows**, utilizando o IIS (Internet Information Services) como Web Server, você deve salvar o projeto **html** em **C:\inetpub\wwwroot**. Lembre-se que é necessário instalar o IIS conforme vimos anteriormente.

Importante
No **Ubuntu**, utilizando o Apache HTTP Server como Web Server, você deve salvar o projeto **html** em **/home/<USUARIO>/public_html**. Lembre-se que é necessário instalar e configurar o Apache HTTP Server como vimos anteriormente.

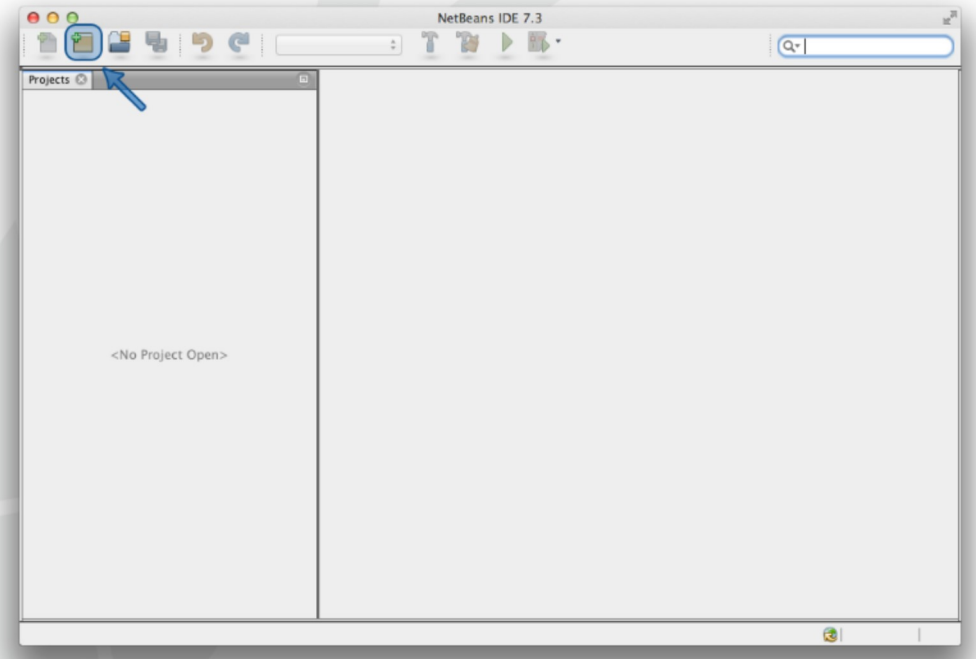


*Figura 2.3: Projeto **html***

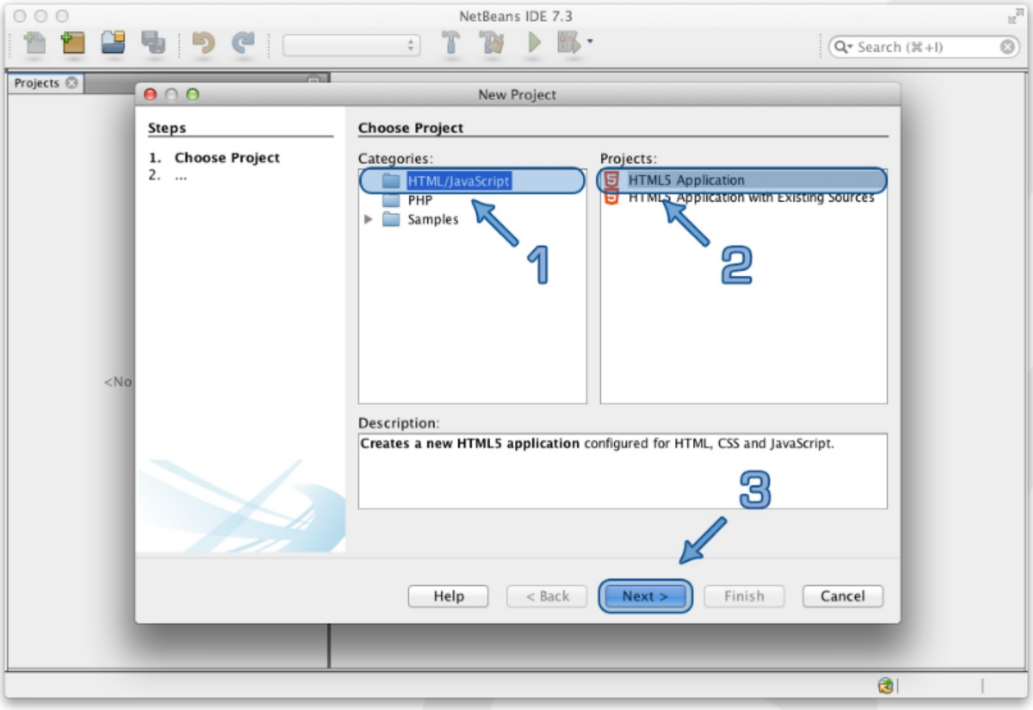


*Figura 2.4: Projeto **html***

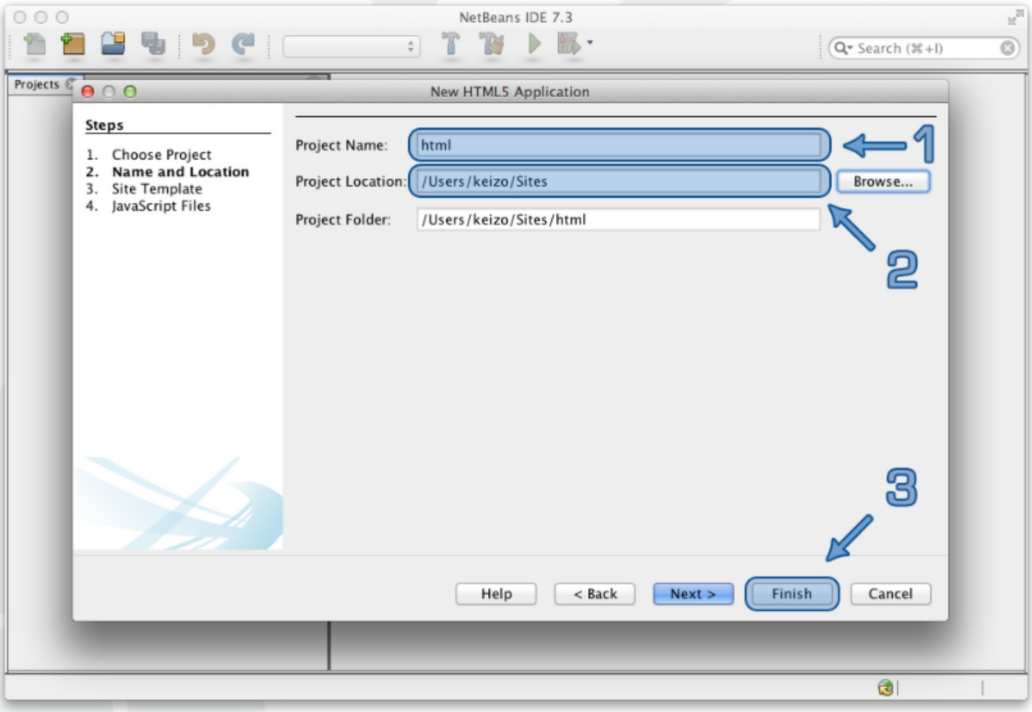


*Figura 2.5: Projeto **html***

Arquivo: `https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao1.zip`

2 Edite o arquivo **index.html** do projeto **html** que está em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <!-- cabeçalho -->
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 Treinamentos</title>
  </head>

  <!-- corpo -->
  <body>
    <p>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</p>
  </body>
</html>
```

Código HTML 2.16: index.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao2.zip

3 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/index.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/index.html.

4 Utilize o **Chrome DevTools** ou o **Firebug** para analisar e modificar os elementos do documento HTML.

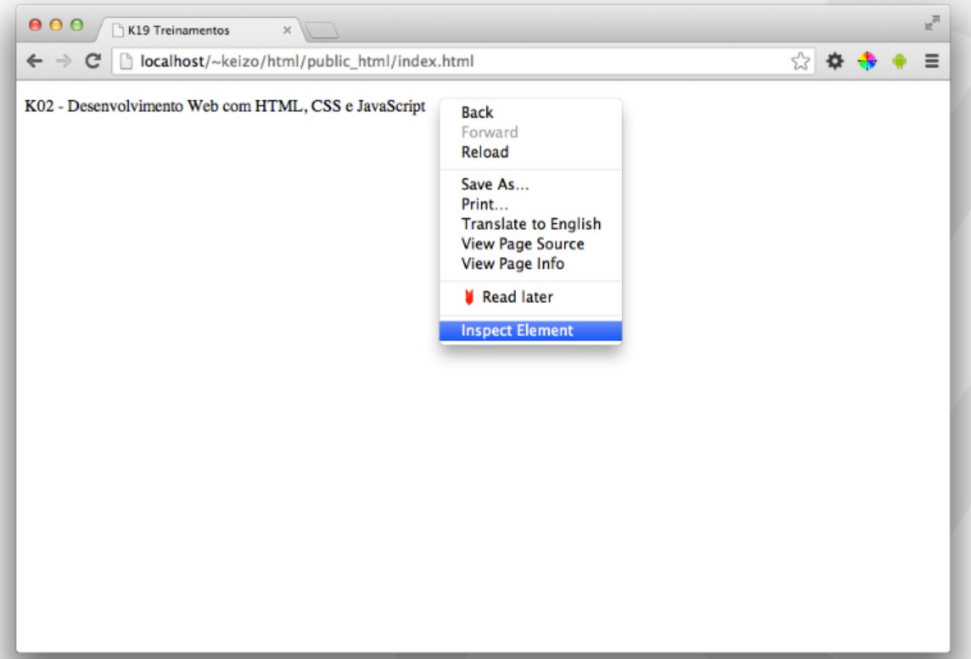


Figura 2.6: Chrome DevTools

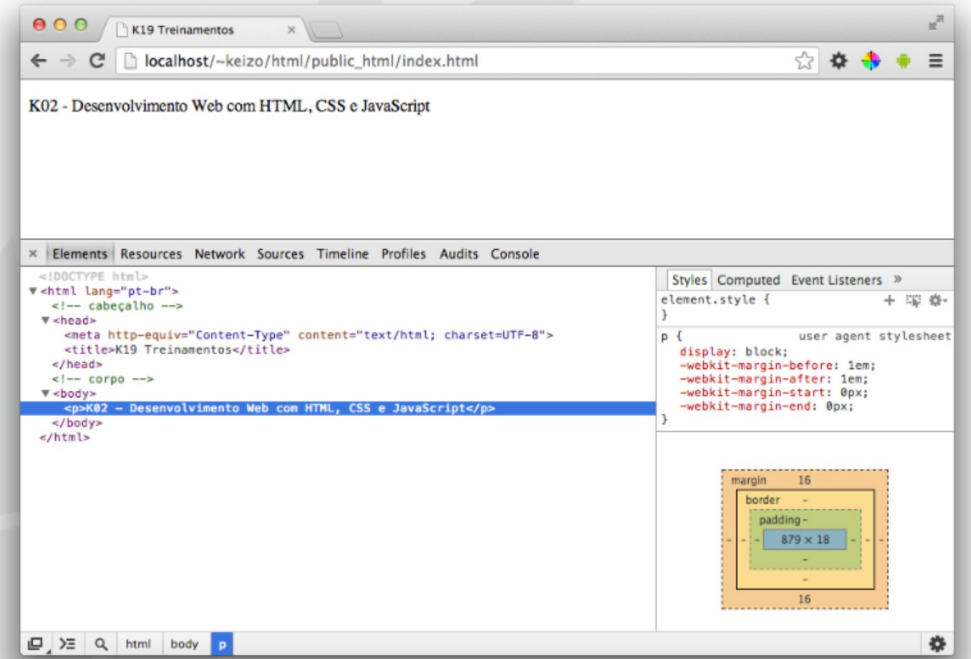


Figura 2.7: Chrome DevTools

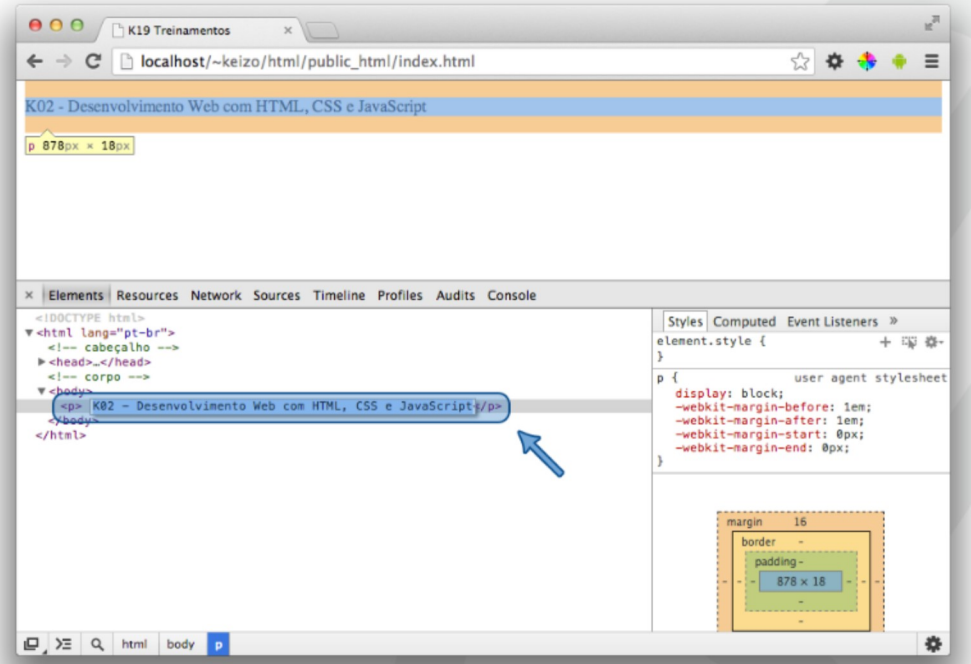


Figura 2.8: Chrome DevTools

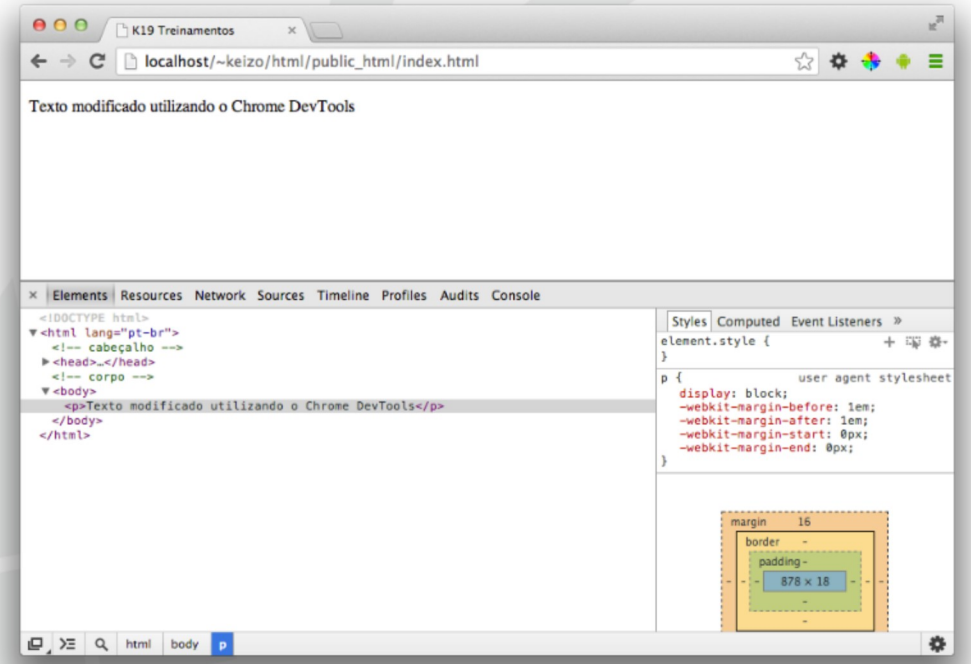


Figura 2.9: Chrome DevTools

5 Utilize o **Chrome DevTools** ou o **Firebug** para analisar a requisição HTTP e a resposta HTTP.

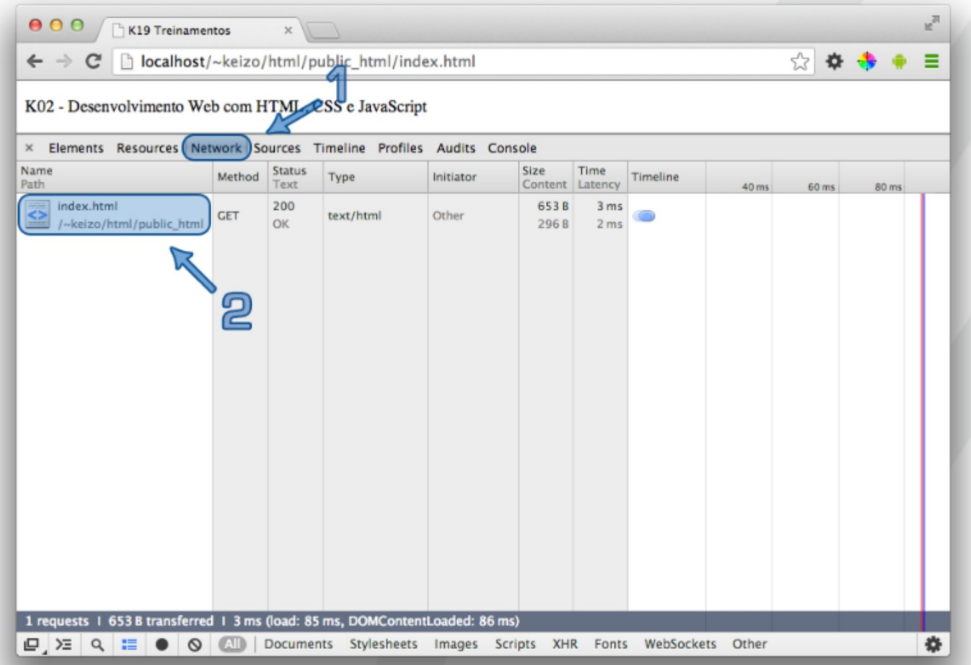


Figura 2.10: Chrome DevTools

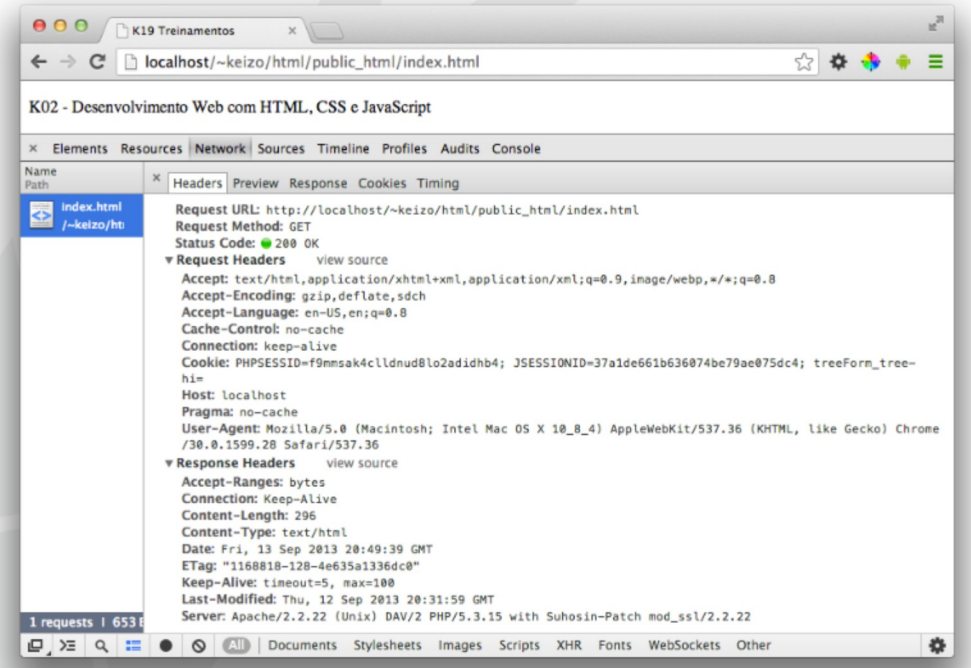


Figura 2.11: Chrome DevTools

Semântica HTML

As placas de sinalização são fundamentais para manter a ordem no trânsito. Cada placa possui um significado específico. A utilização equivocada das placas de trânsito pode causar problemas como colisões de veículos.

Considere um cruzamento no qual é proibido virar a esquerda ou a direita (exemplo: Av. Rebouças com Av Brasil ou Av. Prof. Francisco Morato com Av. Vital Brasil). Nesse cruzamento, o funcionário encarregado de instalar as placas de trânsito decidiu utilizar a sinalização abaixo.

Figura 2.12: Marcador de perigo.

Essa sinalização está totalmente equivocada. Ela indica a existência de obstáculos na pista que obrigam a passagem dos veículos pelo lado direito ou pelo lado esquerdo. O motorista que conhece o significado correto dessa placa pode decidir virar à direita ou à esquerda e sofrer uma colisão. Para não gerar confusão o funcionário poderia utilizar a placa abaixo.

Figura 2.13: Siga em frente.

Para manter a organização no trânsito, a semântica(significado) das sinalizações deve ser respeitada. Analogamente, no desenvolvimento web, é fundamental respeitar o significado dos elementos

HTML.

De acordo com a especificação da linguagem HTML, a maior parte dos elementos possuem um propósito bem definido. Para o funcionamento correto das páginas de uma aplicação web, é fundamental respeitar o propósito de cada elemento e utilizá-lo nas condições corretas. Muitos autores utilizam o termo semântica HTML ao se referirem ao uso correto dos elementos da linguagem HTML.

No exemplo abaixo, utilizamos o elemento **p** para definir um parágrafo. De acordo com a especificação da linguagem HTML, esse elemento deve ser utilizado justamente para definir parágrafos. Portanto, ele foi aplicado corretamente.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de uso correto da semântica HTML</title>
6    </head>
7    <body>
8      <p>Este é um texto para mostrar o significado da tag p.</p>
9    </body>
10 </html>
```

Código HTML 2.17: Exemplo de uso correto da semântica HTML

Agora, vejamos outro exemplo:

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Meus amigos - Site do Jonas - Criado pelo Jonas</title>
6    </head>
7    <body>
8      <address>
9        Rafael Cosentino
10       rafael.cosentino@k19.com.br
11       Sócio fundador da K19 Treinamentos
12       Av. Brigadeiro Faria Lima, 1571 - Jardim Paulistano - São Paulo, SP
13       CEP 01452-001
14     </address>
15
16     <address>
17       Marcelo Martins
18       marcelo.martins@k19.com.br
19       Sócio fundador da K19 Treinamentos
20       Av. Brigadeiro Faria Lima, 1571 - Jardim Paulistano - São Paulo, SP
21       CEP 01452-001
22     </address>
23   </body>
24 </html>
```

Código HTML 2.18: Exemplo de uso incorreto da semântica HTML

Dessa vez, utilizamos o elemento **address**. De acordo com a especificação, esse elemento deve ser utilizado para fornecer informações de contato dos autores ou donos do documento. Normalmente, esse elemento aparece no início ou no final das páginas.

Se observarmos o exemplo mais atentamente, trata-se de uma página do site do Jonas (repare no título da página). O autor da página é o Jonas e não o Rafael ou o Marcelo. Portanto, o elemento **address** foi aplicado incorretamente. Além disso, devemos evitar o uso desse elemento para informar endereços postais a menos que essas informações sejam relacionadas ao autor ou dono do docu-

mento.

Títulos

Assim como em um livro, uma página web pode conter uma hierarquia de títulos para estabelecer uma divisão do seu conteúdo. Para inserir títulos em uma página web, devemos utilizar os elementos **h1**, **h2**, **h3**, **h4**, **h5** e **h6**. Os sufixos numéricos de 1 a 6 indicam o nível do título dentro da hierarquia de títulos do documento. Veja o exemplo:

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de títulos</title>
6    </head>
7    <body>
8      <h1>Título 1</h1>
9      <h2>Título 2</h2>
10     <h3>Título 3</h3>
11     <h4>Título 4</h4>
12     <h5>Título 5</h5>
13     <h6>Título 6</h6>
14   </body>
15 </html>
```

Código HTML 2.19: Exemplo de títulos

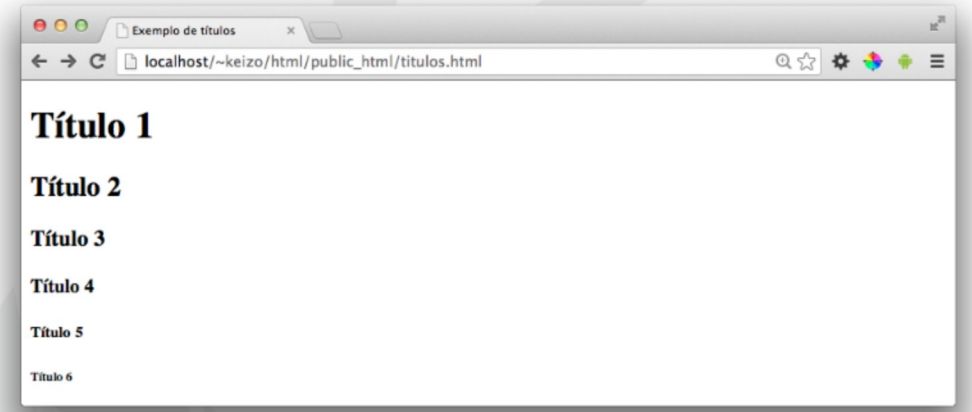


Figura 2.14: Exemplo de títulos

Observe que os títulos não foram exibidos lado a lado e sim um embaixo do outro. Geralmente, os navegadores exibem os títulos como blocos. Por padrão, esses blocos ocupam todo o espaço horizontal do elemento onde os títulos estão contidos. No exemplo, os títulos estão contidos no corpo da página. Portanto, ocupam todo o espaço horizontal da página. Esse é o comportamento padrão dos **block-level elements**. Utilizando as ferramentas de desenvolvimento do **Chrome**, podemos visualizar claramente esses blocos.

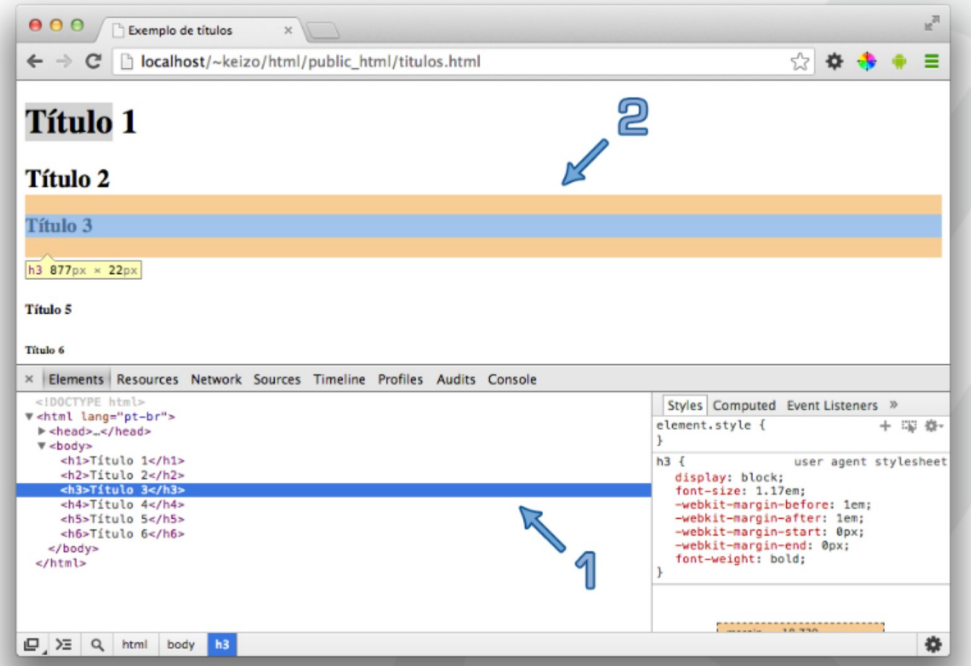


Figura 2.15: Exemplo de títulos

Geralmente, os navegadores utilizam tamanhos diferentes de fonte para cada título. Não existe um padrão de tamanho e de fonte entre os navegadores. Em outras palavras, os títulos de uma página web podem ser exibidos de maneiras diferentes em navegadores distintos. Para padronizar a exibição dos títulos, podemos aplicar as regras CSS.

Devemos utilizar os títulos com cautela, pois eles são usados como critério de ranqueamento por buscadores como Google, Yahoo e Bing. O uso correto dos elementos de título é fortemente recomendado pelos especialistas em SEO (Search Engine Optimization). Para utilizá-los corretamente, devemos respeitar basicamente as seguintes regras.

- Manter a ordem lógica dos títulos. Um elemento **h2** deve ser precedido de um elemento **h1**. Um elemento **h3** deve ser precedido de um elemento **h2**. E assim sucessivamente.
- O título de uma seção deve descrever bem o conteúdo dela.

Exercícios de Fixação

6 Crie um novo documento HTML chamado **titulos.html** com o conteúdo abaixo no projeto **html** em **Site Root**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
```

```
5     <title>K02 - Desenvolvimento Web com HTML, CSS e Javascript</title>
6   </head>
7   <body>
8     <h1>K02 - Desenvolvimento Web com HTML, CSS e Javascript</h1>
9
10    <p>Atualmente, praticamente todos os sistemas corporativos possuem
11    interfaces web. Para quem deseja atuar no mercado de desenvolvimento
12    de software, é obrigatório o conhecimento das linguagens. HTML, CSS
13    e JavaScript.</p>
14
15    <p>Essas linguagens são utilizadas para o desenvolvimento da camada de
16    apresentação das aplicações web.</p>
17
18    <h2>Pré-requisitos</h2>
19
20    <ul>
21      <li>Familiaridade com algum sistema operacional (Windows/Linux/Mac OS X)</li>
22      <li>Lógica de programação</li>
23    </ul>
24
25    <h2>Agenda</h2>
26
27    <h3>Aos domingos</h3>
28
29    <ul>
30      <li>10/11/2013 das 08.00 às 14.00</li>
31      <li>23/02/2014 das 14.00 às 20.00</li>
32    </ul>
33
34    <h3>Aos sábados</h3>
35
36    <ul>
37      <li>15/03/2014 das 08.00 às 14.00</li>
38      <li>26/04/2014 das 14.00 às 20.00</li>
39    </ul>
40  </body>
41 </html>
```

Código HTML 2.20: titulos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao6.zip

7 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/titulos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/titulos.html.

Utilize o **Chrome DevTools** ou o **Firebug** para analisar os elementos que formam o documento HTML, a requisição HTTP e a resposta HTTP.

8 Imagine que você possua um site de culinária no qual você disponibiliza algumas receitas. Crie uma página com uma receita fictícia utilizando títulos para organizar conteúdo. Utilize quantos níveis de título achar necessário. Para isso, adicione um arquivo chamado **receita.html** no projeto **html** em **Site Root**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
```

```
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>Como preparar um delicioso macarrão instantâneo em 6 min.</title>
6   </head>
7   <body>
8     <h1>Como preparar um delicioso macarrão instantâneo em 6 min.</h1>
9
10    <p>Com esta receita você se tornará um profissional na arte de
11    preparar um macarrão instantâneo.</p>
12
13    <h2>Ingredientes</h2>
14
15    <ul>
16      <li>Macarrão instantâneo da sua marca favorita</li>
17      <li>600ml de água</li>
18    </ul>
19
20    <h2>Modo de preparo</h2>
21
22    <h3>No micro-ondas</h3>
23
24    <p>Coloque o macarrão instantâneo em um recipiente com 600ml de água e
25    programe o micro ondas para 6 minutos. Aperte o botão iniciar ou
26    equivalente.</p>
27
28    <h4>Dicas</h4>
29
30    <ul>
31      <li>
32        Utilize um recipiente que permita o macarrão ficar totalmente
33        submerso na água.
34      </li>
35      <li>
36        Quando ouvir o bip, não saia correndo. O micro ondas não explodirá.
37        O bip significa que o macarrão está pronto.
38      </li>
39    </ul>
40
41    <h3>No fogão</h3>
42
43    <p>
44      Coloque o macarrão instantâneo em uma panela com 600ml de água e
45      leve ao fogo por 3 minutos.
46    </p>
47
48    <h4>Dicas</h4>
49
50    <ul>
51      <li>
52        Utilize uma panela que permita o macarrão ficar totalmente
53        submerso na água.
54      </li>
55      <li>
56        Não se distraia com a televisão ou qualquer outra coisa.
57        Você poderá queimar a sua refeição
58      </li>
59    </ul>
60  </body>
61 </html>
```

Código HTML 2.21: receita.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao8.zip

9 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/receita.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/~<USUARIO>/html/public_html/receita.html`.

Utilize o **Chrome DevTools** ou o **Firebug** para analisar os elementos que formam o documento HTML, a requisição HTTP e a resposta HTTP.

Parágrafos

Os parágrafos de uma página web são definidos através do elemento **p**. Assim como os títulos, normalmente, os navegadores definem os parágrafos como **block-level elements**. Dessa forma, por padrão, eles ocupam horizontalmente todo o espaço do elemento pai.

Os navegadores ajustam os textos dos parágrafos à largura do elemento pai. As quebras de linha necessárias são inseridas automaticamente. Os espaços excedentes definidos em um documento HTML entre as palavras de um parágrafo são desconsiderados pelos navegadores na exibição das páginas web. Analogamente, as quebras de linha presentes no documento HTML também são desconsideradas pelos navegadores.

Caso seja necessário forçar uma quebra de linha entre duas palavras contidas em um parágrafo, podemos utilizar o elemento **br**. Quando um parágrafo contém palavras muito longas, os navegadores podem ter dificuldades para ajustar as quebras de linha. Podemos indicar explicitamente com o elemento **wbr**, como as palavras podem ser "quebradas". Veja o exemplo abaixo.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de parágrafos</title>
6    </head>
7    <body>
8      <h1>Exemplo de parágrafos</h1>
9      <p>
10       Este        parágrafo          serve para demonstrar a    inserção
11       automática          das         quebras       de    linha.
12
13       Observe  também que          os espaços       excedentes    contidos
14       no        documento          HTML              entre  as palavras
15       desse  parágrafo    são                           desconsiderados
16
17
18       na exibição da       página  web.
19
20       As            quebras de            linha   também            são
21       desconsideradas.        O navegador            ajusta  o      texto
22       desse          parágrafo     ao espaço              horizontal     do
23       corpo               dá                       página.
24     </p>
25
26     <p>
27       Neste parágrafo mostramos a inserção de uma quebra de linha forçada.
28       <br>Percebeu?<br>Além disso, podemos indicar ao navegador como quebrar
29       palavra otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
30       otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
31       otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
32       otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
33       otorrino<wbr>laringo<wbr>logista.
34     </p>
35   </body>
36 </html>
```

Código HTML 2.22: Exemplo de parágrafos

Figura 2.16: Exemplo de parágrafos

Character Entities

Para inserir determinados caracteres em um documento HTML, devemos utilizar o código do caractere desejado. A utilização desses códigos evita problemas de encoding na exibição das páginas web. Você pode conhecer esses caracteres e os seus respectivos códigos através do seguinte endereço http://dev.w3.org/html5/html-author/charref. Esses caracteres são conhecidos como **Character Entities** ou **HTML Entities**. Veja o exemplo a seguir.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de character entities</title>
6    </head>
7    <body>
8      <h1>Exemplo de character entities</h1>
9      <ul>
10       <li>&starf, &bigstar, &#x02605, &#9733,</li>
11       <li>&phone, &#x0260E, &#9742,</li>
12       <li>&female, &#x02640, &#9792,</li>
13       <li>&sung, &#x0266A, &#9834,</li>
14     </ul>
15   </body>
16 </html>
```

Código HTML 2.23: Exemplo de character entities

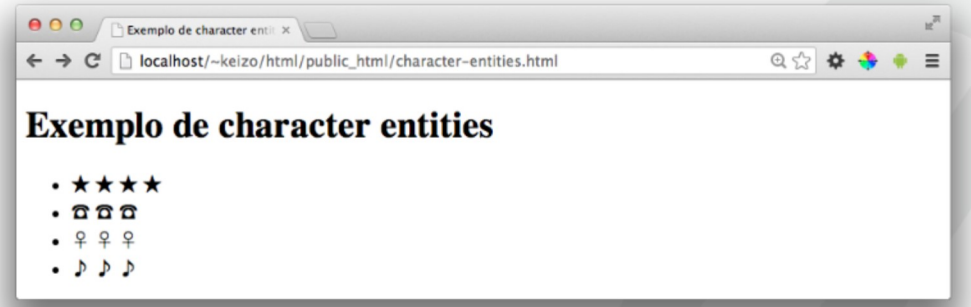


Figura 2.17: Exemplo de character entities

Exercícios de Fixação

10 Crie um novo documento HTML chamado **paragrafos.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de parágrafos</title>
  </head>
  <body>
    <h1>Exemplo de parágrafos</h1>
    <p>
      Este        parágrafo          serve para demonstrar a    inserção
      automática          das         quebras       de    linha.

      Observe também que          os espaços       excedentes    contidos
      no        documento          HTML              entre  as palavras
      desse parágrafo    são                            desconsiderados


      na exibição da       página web.

      As           quebras de            linha   também            são
      desconsideradas.        O navegador            ajusta  o      texto
      desse         parágrafo     ao espaço             horizontal     do
      corpo               dá                      página.
    </p>

    <p>
      Neste parágrafo mostramos a inserção de uma quebra de linha forçada.
      <br>Percebeu?<br>Além disso, podemos indicar ao navegador como quebrar
      palavra otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr/logista
      otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
      otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
      otorrino<wbr>laringo<wbr>logista otorrino<wbr>laringo<wbr>logista
      otorrino<wbr>laringo<wbr>logista.
    </p>
  </body>
</html>
```

Código HTML 2.24: paragrafos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao10.zip

11 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/paragrafos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/paragrafos.html.

12 Crie um novo documento HTML chamado **character-entities.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de character entities</title>
  </head>
  <body>
    <h1>Exemplo de character entities</h1>
    <ul>
      <li>&starf, &bigstar, &#x02605, &#9733,</li>
      <li>&phone, &#x0260E, &#9742,</li>
      <li>&female, &#x02640, &#9792,</li>
      <li>&sung, &#x0266A, &#9834,</li>
    </ul>
  </body>
</html>
```

Código HTML 2.25: character-entities.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao12.zip

13 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/character-entities.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/character-entities.html.

Texto

Texto pré-formatado

Como vimos, os navegadores desconsideram os espaços excedentes entre as palavras contidas em um documento HTML assim como desconsideram as quebras de linha. Contudo, é possível inserir texto formatado com quantos espaços e quebras de linha desejarmos através do elemento **pre**. O texto contido no conteúdo de um elemento **pre** é exibido com todos os espaços e quebras de linha inseridos no documento HTML. Além disso, os navegadores costumam utilizar fonte mono espaçada para mostrar esse texto.

Assim como os títulos e os parágrafos, normalmente, os navegadores definem os textos pré-formatados como **block-level element**. Dessa forma, por padrão, eles ocupam todo o espaço horizontal do elemento pai.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de texto pré-formatado</title>
6    </head>
7    <body>
8      <h1>Exemplo de texto pré-formatado</h1>
9      <pre>
10 Os           espaços           excedentes   são considerados     pelos
11 navegadores. Assim como            as
12
13 quebras
14
15 de
16
17 linha. Observe também o tipo de          fonte  utilizada nesse texto.
18     </pre>
19   </body>
20 </html>
```

Código HTML 2.26: Exemplo de texto pré-formatado

Figura 2.18: Exemplo de texto pré-formatado

Código

Para adicionar códigos em um documento HTML, devemos utilizar o elemento **code**. Normalmente, esse elemento é utilizado para definir códigos escritos em alguma linguagem de programação. Normalmente, os navegadores utilizam fonte mono espaçada para exibir o conteúdo desse elemento.

Diferentemente dos títulos, dos parágrafos e dos textos pré-formatados, normalmente, os navegadores definem os códigos como **inline-level elements**. Dessa forma, por padrão, os códigos ocupam horizontalmente somente o espaço necessário.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
```

```
5     <title>Exemplo de códigos</title>
6   </head>
7   <body>
8     <h1>Exemplo de códigos</h1>
9     <p>
10      Em Java, utilizamos o código <code>System.out.println("K19"),</code>.
11      Em C#, utilizamos o código <code>System.Console.WriteLine("K19"),</code>.
12    </p>
13  </body>
14 </html>
```

Código HTML 2.27: Exemplo de código

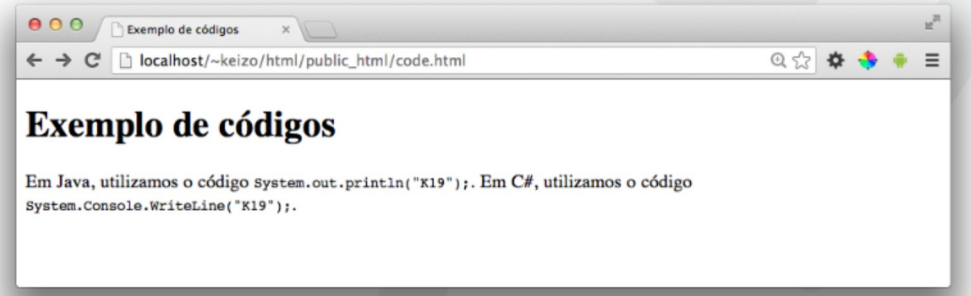


Figura 2.19: Exemplo de código

Elemento i

A especificação da linguagem HTML 5 não é muito precisa na definição da semântica do elemento **i**. Ela indica a utilização desse elemento para definir nomes científicos, termos técnicos, expressões idiomáticas em outras línguas, transliterações, pensamentos. Normalmente, os navegadores definem o **i** como **inline-level element** e exibem o seu conteúdo em itálico.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de utilização do elemento i</title>
6    </head>
7    <body>
8      <h1>Exemplo de utilização do elemento i</h1>
9      <ul>
10       <li>Porquinho-da-índia ou <i>Cavia porcellus</i></li>
11       <li><i>Backup</i>(cópia de segurança)</li>
12       <li><i>shoot the breeze</i>(bater papo ou jogar conversa fora)</li>
13       <li><i>Moskvá</i>(transliteração da palavra Moscou em russo)</li>
14       <li><i>Se não sabes, aprende, se já sabes, ensina.</i> (Confúcio)</li>
15     </ul>
16   </body>
17 </html>
```

*Código HTML 2.28: Exemplo de utilização do elemento **i***

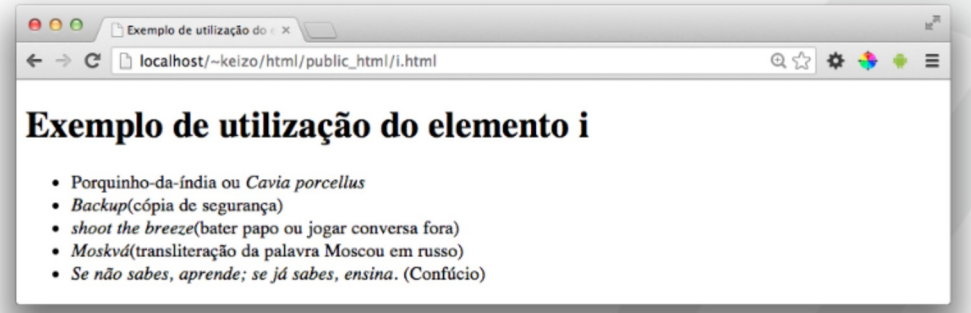


*Figura 2.20: Exemplo de utilização do elemento **i***

Elemento b

A especificação da linguagem HTML 5 não é muito precisa na definição da semântica do elemento **b**. Ela indica, por exemplo, a utilização desse elemento para definir as palavras chave do resumo de um documento e o nome do produto em texto de avaliação. Normalmente, os navegadores definem o **b** como **inline-level element** e exibem o conteúdo em negrito.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de utilização do elemento b</title>
6    </head>
7    <body>
8      <h1>Exemplo de utilização do elemento b</h1>
9      <p>
10       Atualmente, praticamente todos os <b>sistemas corporativos</b>
11       possuem <b>interfaces web</b>. Para quem deseja atuar no mercado
12       de <b>desenvolvimento de software</b>, é obrigatório o conhecimento
13       das linguagens. <b>HTML</b>, <b>CSS</b> e <b>JavaScript</b>.
14     </p>
15   </body>
16 </html>
```

*Código HTML 2.29: Exemplo de utilização do elemento **b***

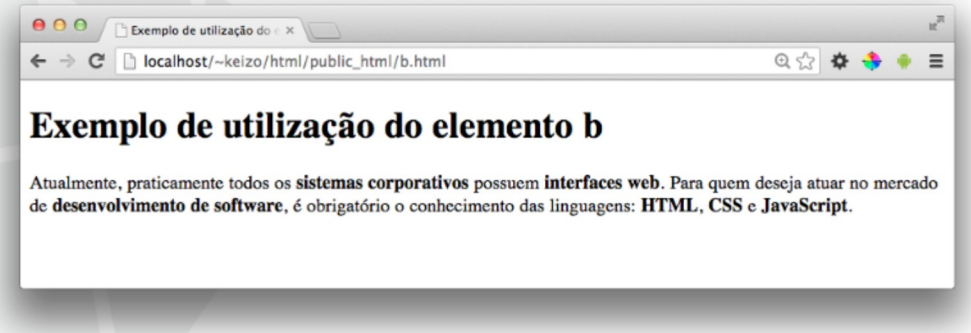


*Figura 2.21: Exemplo de utilização do elemento **b***

Texto subscrito ou sobrescrito

Podemos definir textos subscrito ou sobrescrito com os elementos **sub** e **sup** respectivamente. Normalmente, os navegadores definem esses elementos como **inline-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de texto subscrito ou sobrescrito</title>
6    </head>
7    <body>
8      <h1>Exemplo de texto subscrito ou sobrescrito</h1>
9      <p>
10       A cidade de São Paulo possui uma área de 1.523 km<sup>2</sup>.
11       Em 2011, São Paulo emitiu 16,430 milhões de toneladas de
12       CO<sub>2</sub>.
13     </p>
14   </body>
15 </html>
```

Código HTML 2.30: Exemplo de texto subscrito ou sobrescrito

Figura 2.22: Exemplo de texto subscrito ou sobrescrito

Texto importante ou enfatizado

Podemos definir textos importantes ou enfatizados com os elementos **strong** e **em** respectivamente. Por padrão, nos navegadores, o conteúdo de um elemento **strong** é exibido em negrito e o conteúdo de um elemento **em** é exibido em itálico. Normalmente, os navegadores definem esses elementos como **inline-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de texto importante ou enfatizado</title>
6    </head>
7    <body>
8      <h1>Exemplo de texto importante ou enfatizado</h1>
9      <p>
10       O <strong>Brasil</strong> é o único
11       país que ganhou <em>cinco vezes</em>
12       a <strong>copa do mundo de futebol</strong>.
13     </p>
14   </body>
15 </html>
```

Código HTML 2.31: Exemplo de texto importante ou enfatizado

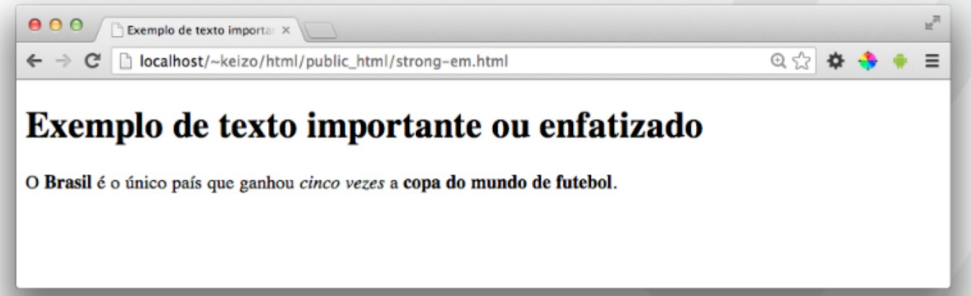


Figura 2.23: Exemplo de texto importante ou enfatizado

Citações

Podemos definir citações longas ou curtas com os elementos **blockquote** e **q** respectivamente. Por padrão, o elemento **blockquote** é exibido como elemento de bloco e o elemento **q** é exibido como elemento de linha. Esses dois elementos possuem o atributo **cite** que deve ser utilizado para indicar a fonte da citação.

Também podemos citar, com o elemento **cite**, títulos de trabalhos, livros, músicas, filmes, programas de TV, peças de teatro, entre outros. Por padrão, esse elemento é exibido como elemento de linha.

Normalmente, os navegadores definem o **blockquote** como **block-level element**. Por outro lado, o **q** e o **cite**, normalmente, são definidos como **inline-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de citações</title>
6    </head>
7    <body>
8      <h1>Exemplo de citações</h1>
9      <blockquote cite="http.//en.wikipedia.org/wiki/Computer">
10       <p>
11         A computer is a general purpose device that can be programmed
12         to carry out a set of arithmetic or logical operations. Since a
13         sequence of operations can be readily changed, the computer
14         can solve more than one kind of problem.
15       </p>
16
17       <p>
18         Conventionally, a computer consists of at least one processing
19         element, typically a central processing unit (CPU) and some form
20         of memory. The processing element carries out arithmetic and
21         logic operations, and a sequencing and control unit that can
22         change the order of operations based on stored information.
23         Peripheral devices allow information to be retrieved from an
24         external source, and the result of operations saved and retrieved.
25       </p>
26     </blockquote>
27
```

```
28      <p>
29        Nelson Rodrigues disse.
30        <q cite="http.//pt.wikipedia.org/wiki/Nelson_Rodrigues">
31          O adulto não existe. O homem é um menino perene
32        </q>.
33      </p>
34
35      <p>
36        A peça <cite>A mulher sem pecado</cite> de Nelson Rodrigues estreou
37        em 1941 no Rio de Janeiro.
38      </p>
39    </body>
40  </html>
```

Código HTML 2.32: Exemplo de citações

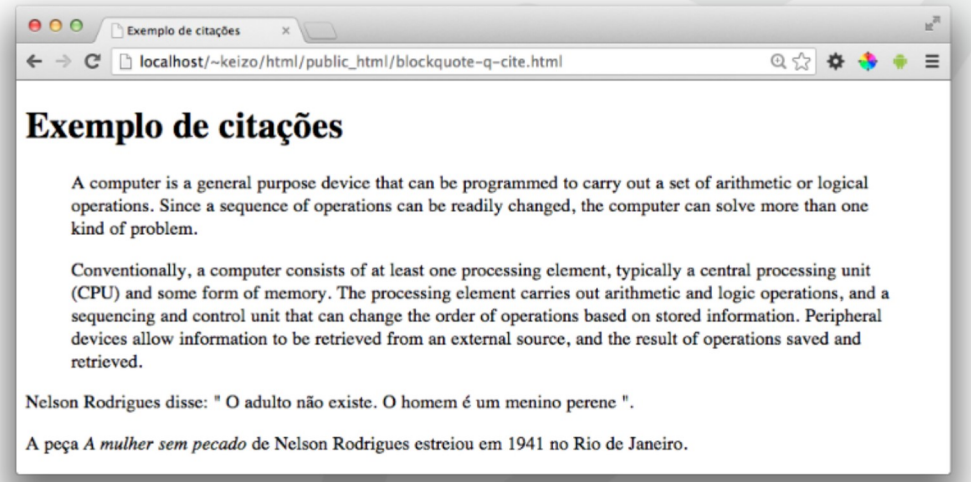


Figura 2.24: Exemplo de citações

Abreviações

Podemos definir abreviações com o elemento **abbr**. Por padrão, esse elemento é exibido como elemento de linha. O atributo **title** desse elemento é utilizado para definir um *tootip*. Normalmente, os navegadores definem esse elemento como **inline-level element**

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de abreviações</title>
6    </head>
7    <body>
8      <h1>Exemplo de abreviações</h1>
9      <p>
10       O <abbr title="Instituto de Matemática e Estatística">IME</abbr> é
11       uma das unidades da <abbr title="Universidade de São Paulo">USP</abbr>.
12     </p>
13   </body>
14 </html>
```

Código HTML 2.33: Exemplo de abreviações

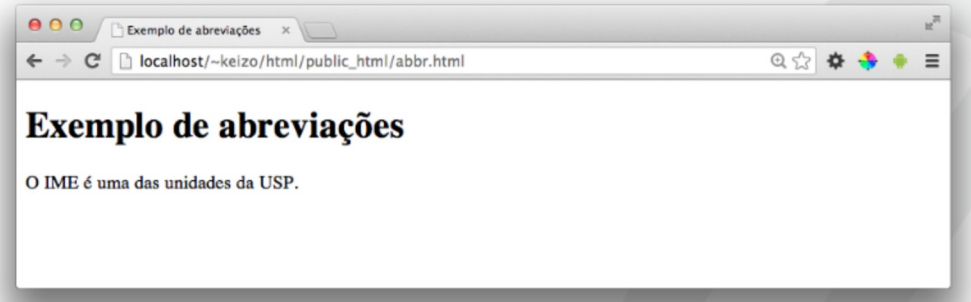


Figura 2.25: Exemplo de abreviações

Definições

Podemos inserir definições em um documento HTML com o elemento **dfn**. Por padrão, esse elemento é exibido como elemento de linha. Normalmente, os navegadores definem o **dfn** como **inline-level element**

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de definições</title>
6    </head>
7    <body>
8      <h1>Exemplo de definições</h1>
9      <p>
10       Um <dfn>pingado</dfn> é um copo de café com um pouco de leite.<br>
11       Uma <dfn>bomba de chocolate</dfn> é um doce de padaria feito com
12       massa de farinha de trigo com recheio cremoso e cobertura de chocolate.
13     </p>
14   </body>
15 </html>
```

Código HTML 2.34: Exemplo de definições

Figura 2.26: Exemplo de definições

Alterações

Em alguns casos, é importante informar que o texto de um documento HTML sofreu alterações. Um texto que foi acrescentado em um documento HTML deve ser definido com o elemento **ins**. Um texto que não faz mais parte do documento deve ser definido com o elemento **del**. Um texto que deixou de ser correto, preciso ou relevante deve ser definido com o elemento **s**. Por padrão, esses elementos são exibidos como elementos de linha. Normalmente, os navegadores definem esses elementos como **inline-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de alterações</title>
6    </head>
7    <body>
8      <h1>Exemplo de alterações</h1>
9      <p>
10       Natal é a capital do Rio Grande do Norte. <ins>Sua fundação foi
11       em 25 de dezembro de 1599.</ins> <del>A área total dessa cidade é
12       167,263 km<sup>2</sup></del>. <s>De acordo com o IBGE, a população
13       dessa cidade é de 774.230 habitantes.</s> De acordo com o IBGE a
14       população dessa cidade é de 803.739 habitantes.
15     </p>
16   </body>
17 </html>
```

Código HTML 2.35: Exemplo de alterações

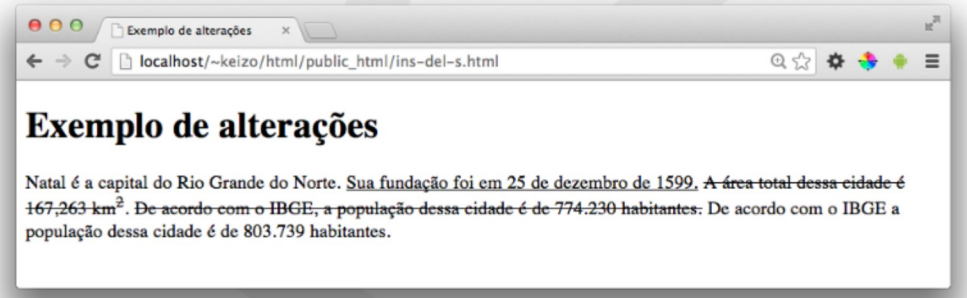


Figura 2.27: Exemplo de alterações

Os elementos **ins** e **del** possuem os atributos **cite** e **datetime**. O atributo **cite** deve ser utilizado para indicar a URL de um documento com as justificativas da alteração. O atributo **datetime** deve ser utilizado para informar quando ocorreu a alteração.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de alterações</title>
6    </head>
7    <body>
8      <h1>Exemplo de alterações</h1>
9      <p>
10       Natal é a capital do Rio Grande do Norte. <ins cite="ins-explicacao.html"
11       datetime="2013-10-15T22.55.03Z">Sua fundação foi em 25 de dezembro de
```

```
12        1599.</ins> <del cite="del-explicacao.html" datetime="2013-10-10T17.43.07Z">A
13        área total dessa cidade é 167,263 km<sup>2</sup></del>. <s>De acordo com o
14        IBGE, a população dessa cidade é de 774.230 habitantes.</s> De acordo com o
15        IBGE a população dessa cidade é de 803.739 habitantes.
16      </p>
17    </body>
18  </html>
```

Código HTML 2.36: Exemplo de alterações

Data e hora

O elemento **time** permite que datas e horários sejam definidos em um documento HTML. Há duas formas de utilizar esse elemento. A primeira é definir a data e o horário desejado no conteúdo do elemento **time**. A segunda é definir a data e o horário desejado no valor do atributo **datetime**. Nessas duas opções, as datas e os horários devem ser escritos seguindo o formato definido na especificação da linguagem HTML (http.//www.w3.org/TR/html5/text-level-semantics.html#the-time-element). Quando utilizamos o atributo **datetime**, o conteúdo do elemento **time** não precisa seguir um formato específico. Normalmente, os navegadores definem o **time** como **inline-level element**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de datas e horas</title>
6    </head>
7    <body>
8      <h1>Exemplo de datas e horas</h1>
9      <ul>
10       <li><time>2010-08</time></li>
11       <li><time>1984-10-30</time></li>
12       <li><time>12-25</time></li>
13       <li><time>09.00</time></li>
14       <li><time>2013-12-25 23.59</time></li>
15       <li><time datetime="2010-08">Fundação da K19</time></li>
16       <li><time datetime="1984-10-30">Aniversário do Rafael</time></li>
17       <li><time datetime="12-25">Natal</time></li>
18       <li><time datetime="09.00">Horário de Entrada</time></li>
19       <li><time datetime="2013-12-31 23.59.59">Ano Novo 2014</time></li>
20     </ul>
21   </body>
22 </html>
```

Código HTML 2.37: Exemplo de datas e horas

Figura 2.28: Exemplo de datas e horas

Texto marcado

O elemento **mark** permite que determinados trechos de um texto sejam marcados. Normalmente, os navegadores definem o **mark** como **inline-level element**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de texto marcado</title>
  </head>
  <body>
    <h1>Exemplo de texto marcado</h1>
    <p>
      <mark>Java</mark> e <mark>C#</mark> são linguagens de programação
      <mark>orientadas a objeto</mark>.
    </p>
  </body>
</html>
```

Código HTML 2.38: Exemplo de texto marcado

Figura 2.29: Exemplo de texto marcado

Exercícios de Fixação

14 Crie um novo documento HTML chamado **texto-pre-formatado.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de texto pré-formatado</title>
  </head>
  <body>
    <h1>Exemplo de texto pré-formatado</h1>
    <pre>
Os            espaços           excedentes   são considerados     pelos
navegadores. Assim como            as

quebras

de

linha. Observe também o tipo de          fonte  utilizada nesse texto.
    </pre>
  </body>
</html>
```

Código HTML 2.39: texto-pre-formatado.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao14.zip

15 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/texto-pre-formatado.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/texto-pre-formatado.html.

16 Crie um novo documento HTML chamado **codigo.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de códigos</title>
  </head>
  <body>
    <h1>Exemplo de códigos</h1>
    <p>
      Em Java, utilizamos o código <code>System.out.println("K19"),</code>
      Em C#, utilizamos o código <code>System.Console.WriteLine("K19"),</code>
    </p>
  </body>
</html>
```

Código HTML 2.40: codigo.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao16.zip

17 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/codigo.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/codigo.html.

18 Crie um novo documento HTML chamado **i.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de utilização do elemento i</title>
  </head>
  <body>
    <h1>Exemplo de utilização do elemento i</h1>
    <ul>
      <li>Porquinho-da-índia ou <i>Cavia porcellus</i></li>
      <li><i>Backup</i>(cópia de segurança)</li>
      <li><i>shoot the breeze</i>(bater papo ou jogar conversa fora)</li>
      <li><i>Moskvá</i>(transliteração da palavra Moscou em russo)</li>
      <li><i>Se não sabes, aprende, se já sabes, ensina.</i> (Confúcio)</li>
    </ul>
  </body>
</html>
```

Código HTML 2.41: i.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao18.zip

19 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/i.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/i.html.

20 Crie um novo documento HTML chamado **b.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de utilização do elemento b</title>
  </head>
  <body>
    <h1>Exemplo de utilização do elemento b</h1>
    <p>
      Atualmente, praticamente todos os <b>sistemas corporativos</b>
      possuem <b>interfaces web</b>. Para quem deseja atuar no mercado
      de <b>desenvolvimento de software</b>, é obrigatório o conhecimento
      das linguagens. <b>HTML</b>, <b>CSS</b> e <b>JavaScript</b>.
    </p>
```

```
  </body>
</html>
```

Código HTML 2.42: b.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao20.zip

21 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/b.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/b.html.

22 Crie um novo documento HTML chamado **texto-subscrito-sobrescrito.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de texto subscrito ou sobrescrito</title>
  </head>
  <body>
    <h1>Exemplo de texto subscrito ou sobrescrito</h1>
    <p>
      A cidade de São Paulo possui uma área de 1.523 km<sup>2</sup>.
      Em 2011, São Paulo emitiu 16,430 milhões de toneladas de
      CO<sub>2</sub>.
    </p>
  </body>
</html>
```

Código HTML 2.43: texto-subscrito-sobrescrito.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao22.zip

23 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/texto-subscrito-sobrescrito.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/texto-subscrito-sobrescrito.html.

24 Crie um novo documento HTML chamado **texto-importante-enfatizado.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de texto importante ou enfatizado</title>
  </head>
  <body>
    <h1>Exemplo de texto importante ou enfatizado</h1>
```

```
    <p>
      O <strong>Brasil</strong> é o único
      país que ganhou <em>cinco vezes</em>
      a <strong>copa do mundo de futebol</strong>.
    </p>
  </body>
</html>
```

Código HTML 2.44: texto-importante-enfatizado.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao24.zip

25 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/texto-importante-enfatizado.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/texto-importante-enfatizado.html.

26 Crie um novo documento HTML chamado **citacoes.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de citações</title>
  </head>
  <body>
    <h1>Exemplo de citações</h1>
    <blockquote cite="http://en.wikipedia.org/wiki/Computer">
      <p>
        A computer is a general purpose device that can be programmed
        to carry out a set of arithmetic or logical operations. Since a
        sequence of operations can be readily changed, the computer
        can solve more than one kind of problem.
      </p>

      <p>
        Conventionally, a computer consists of at least one processing
        element, typically a central processing unit (CPU) and some form
        of memory. The processing element carries out arithmetic and
        logic operations, and a sequencing and control unit that can
        change the order of operations based on stored information.
        Peripheral devices allow information to be retrieved from an
        external source, and the result of operations saved and retrieved.
      </p>
    </blockquote>

    <p>
      Nelson Rodrigues disse.
      <q cite="http://pt.wikipedia.org/wiki/Nelson_Rodrigues">
        O adulto não existe. O homem é um menino perene
      </q>.
    </p>

    <p>
      A peça <cite>A mulher sem pecado</cite> de Nelson Rodrigues estreou
      em 1941 no Rio de Janeiro.
    </p>
  </body>
</html>
```

Código HTML 2.45: citacoes.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao26.zip

27 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/citacoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/citacoes.html.

28 Crie um novo documento HTML chamado **abreviacoes.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de abreviações</title>
6    </head>
7    <body>
8      <h1>Exemplo de abreviações</h1>
9      <p>
10       O <abbr title="Instituto de Matemática e Estatística">IME</abbr> é
11       uma das unidades da <abbr title="Universidade de São Paulo">USP</abbr>.
12     </p>
13   </body>
14 </html>
```

Código HTML 2.46: abreviacoes.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao28.zip

29 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/abreviacoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/abreviacoes.html.

30 Crie um novo documento HTML chamado **definicoes.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de definições</title>
6    </head>
7    <body>
8      <h1>Exemplo de definições</h1>
9      <p>
10       Um <dfn>pingado</dfn> é um copo de café com um pouco de leite.<br>
11       Uma <dfn>bomba de chocolate</dfn> é um doce de padaria feito com
12       massa de farinha de trigo com recheio cremoso e cobertura de chocolate.
13     </p>
14   </body>
15 </html>
```

Código HTML 2.47: definicoes.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao30.zip

31 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/definicoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/definicoes.html.

32 Crie um novo documento HTML chamado **alteracoes.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de alterações</title>
6    </head>
7    <body>
8      <h1>Exemplo de alterações</h1>
9      <p>
10       Natal é a capital do Rio Grande do Norte. <ins>Sua fundação foi
11       em 25 de dezembro de 1599.</ins> <del>A área total dessa cidade é
12       167,263 km<sup>2</sup></del>. <s>De acordo com o IBGE, a população
13       dessa cidade é de 774.230 habitantes.</s> De acordo com o IBGE a
14       população dessa cidade é de 803.739 habitantes.
15     </p>
16   </body>
17 </html>
```

Código HTML 2.48: alteracoes.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao32.zip

33 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/alteracoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/alteracoes.html.

34 Crie um novo documento HTML chamado **data-hora.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de datas e horas</title>
6    </head>
7    <body>
8      <h1>Exemplo de datas e horas</h1>
9      <ul>
10       <li><time>2010-08</time></li>
11       <li><time>1984-10-30</time></li>
12       <li><time>12-25</time></li>
13       <li><time>09.00</time></li>
14       <li><time>2013-12-25 23.59</time></li>
```

```
      <li><time datetime="2010-08">Fundação da K19</time></li>
      <li><time datetime="1984-10-30">Aniversário do Rafael</time></li>
      <li><time datetime="12-25">Natal</time></li>
      <li><time datetime="09.00">Horário de Entrada</time></li>
      <li><time datetime="2013-12-31 23.59.59">Ano Novo 2014</time></li>
    </ul>
  </body>
</html>
```

Código HTML 2.49: data-hora.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao34.zip

35 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/data-hora.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/data-hora.html.

36 Crie um novo documento HTML chamado **mark.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de texto marcado</title>
  </head>
  <body>
    <h1>Exemplo de texto marcado</h1>
    <p>
      <mark>Java</mark> e <mark>C#</mark> são linguagens de programação
      <mark>orientadas a objeto</mark>.
    </p>
  </body>
</html>
```

Código HTML 2.50: mark.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao36.zip

37 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/mark.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/mark.html.

Listas

A linguagem HTML define três tipos distintos de listas.

- Lista de descrições

- Lista com ordem
- Lista sem ordem

Lista de descrições

Para criar uma lista de descrições, devemos utilizar o elemento **dl**. Essas listas são formadas por termos ou nomes e as suas respectivas descrições. Os termos ou nomes são definidos com o elemento **dt**. As descrições são definidas com o elemento **dd**. Normalmente, os navegadores definem todos esses elementos como **block-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de lista de descrições</title>
6    </head>
7    <body>
8      <h1>Exemplo de lista de descrições</h1>
9      <dl>
10       <dt>K01 - Lógica de Programação</dt>
11       <dd>
12         Conhecimentos em Lógica de Programação é o pré-requisito fundamental
13         para que uma pessoa consiga aprender qualquer Linguagem de Programação...
14       </dd>
15       <dt>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</dt>
16       <dd>
17         Atualmente, praticamente todos os sistemas corporativos possuem
18         interfaces web. Para quem deseja atuar no mercado de desenvolvimento...
19       </dd>
20       <dt>K03 - SQL e Modelo Relacional</dt>
21       <dd>
22         Como as aplicações corporativas necessitam armazenar dados é fundamental
23         que os desenvolvedores possuam conhecimentos sobre persistência de dados.
24       </dd>
25     </dl>
26   </body>
27 </html>
```

Código HTML 2.51: Exemplo de lista de descrições

Figura 2.30: Exemplo de lista de descrições

Lista com ordem

Para criar uma lista com ordem, devemos utilizar o elemento **ol**. Os itens de uma lista com ordem são definidos com o elemento **li**. Normalmente, os navegadores definem todos esses elementos como **block-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de lista com ordem</title>
6    </head>
7    <body>
8      <h1>Macarrão instantâneo - K19 Receitas</h1>
9      <h2>Modo de preparo</h2>
10
11     <ol>
12       <li>Ferver 600ml de água em uma panela.</li>
13       <li>Retirar o macarrão do pacote.</li>
14       <li>Colocar o macarrão na panela no fogo baixo.</li>
15       <li>Cozinhar o macarrão por 3min.</li>
16       <li>Temperar a gosto.</li>
17     </ol>
18   </body>
19 </html>
```

Código HTML 2.52: Exemplo de lista com ordem

Figura 2.31: Exemplo de lista com ordem

Lista sem ordem

Para criar uma lista sem ordem, devemos utilizar o elemento **ul**. Os itens de uma lista sem ordem são definidos com o elemento **li**. Normalmente, os navegadores definem todos esses elementos como **block-level elements**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de lista sem ordem</title>
6    </head>
7    <body>
8      <h1>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</h1>
9      <h2>Pré-requisitos</h2>
```

```
10
11      <ul>
12        <li>Conhecimento de algum sistema operacional (Windows/Linux/MacOS X)</li>
13        <li>Lógica de programação</li>
14      </ul>
15    </body>
16  </html>
```

Código HTML 2.53: Exemplo de lista sem ordem

Figura 2.32: Exemplo de lista sem ordem

Listas aninhadas

Uma lista pode ser definida dentro de outra lista. Quando listas sem ordem são aninhadas, normalmente, os navegadores alternam o marcadores dos itens.

```
1   <!DOCTYPE html>
2   <html lang="pt-br">
3     <head>
4       <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5       <title>Exemplo de listas aninhadas</title>
6     </head>
7     <body>
8       <h1>Exemplo de listas aninhadas</h1>
9       <h2>Continentes</h2>
10      <ul>
11        <li>
12          Europa
13          <ul>
14            <li>Portugual</li>
15            <li>França</li>
16            <li>Alemanha</li>
17          </ul>
18        </li>
19        <li>
20          Ásia
21          <ul>
22            <li>Japão</li>
23            <li>China</li>
24            <li>Índia</li>
25          </ul>
26        </li>
27      </ul>
28      <h2>Cronograma da minha viagem</h2>
29      <ol>
30        <li>
31          Europa
```

```
32          <ol>
33            <li>Portugual</li>
34            <li>França</li>
35            <li>Alemanha</li>
36          </ol>
37        </li>
38        <li>
39          Ásia
40          <ol>
41            <li>Japão</li>
42            <li>China</li>
43            <li>Índia</li>
44          </ol>
45        </li>
46      </ol>
47    </body>
48  </html>
```

Código HTML 2.54: Exemplo de listas aninhadas

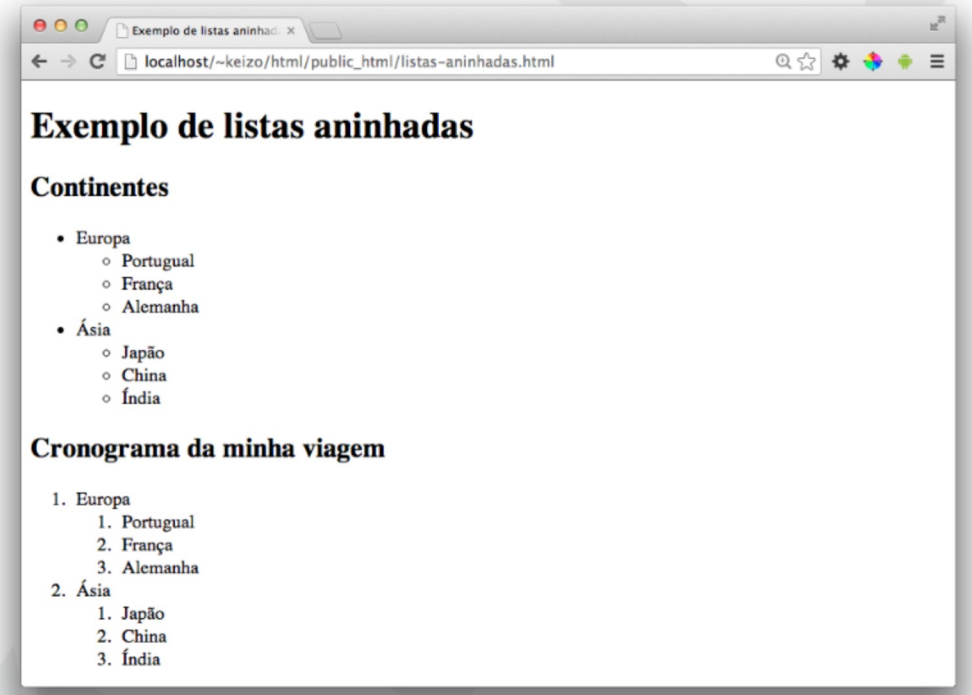


Figura 2.33: Exemplo de lista aninhadas

Exercícios de Fixação

38 Crie um novo documento HTML chamado **lista-descricoes.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
```

```
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>Exemplo de lista de descrições</title>
6   </head>
7   <body>
8     <h1>Exemplo de lista de descrições</h1>
9     <dl>
10      <dt>K01 - Lógica de Programação</dt>
11      <dd>
12        Conhecimentos em Lógica de Programação é o pré-requisito fundamental
13        para que uma pessoa consiga aprender qualquer Linguagem de Programação...
14      </dd>
15      <dt>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</dt>
16      <dd>
17        Atualmente, praticamente todos os sistemas corporativos possuem
18        interfaces web. Para quem deseja atuar no mercado de desenvolvimento...
19      </dd>
20      <dt>K03 - SQL e Modelo Relacional</dt>
21      <dd>
22        Como as aplicações corporativas necessitam armazenar dados é fundamental
23        que os desenvolvedores possuam conhecimentos sobre persistência de dados.
24      </dd>
25    </dl>
26  </body>
27 </html>
```

Código HTML 2.55: lista-descricoes.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao38.zip

39 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/lista-descricoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/lista-descricoes.html.

40 Crie um novo documento HTML chamado **lista-com-ordem.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de lista com ordem</title>
6    </head>
7    <body>
8      <h1>Macarrão instantâneo - K19 Receitas</h1>
9      <h2>Modo de preparo</h2>
10
11     <ol>
12       <li>Ferver 600ml de água em uma panela.</li>
13       <li>Retirar o macarrão do pacote.</li>
14       <li>Colocar o macarrão na panela no fogo baixo.</li>
15       <li>Cozinhar o macarrão por 3min.</li>
16       <li>Temperar a gosto.</li>
17     </ol>
18   </body>
19 </html>
```

Código HTML 2.56: lista-com-ordem.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao40.zip

41 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/lista-com-ordem.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/lista-com-ordem.html.

42 Crie um novo documento HTML chamado **lista-sem-ordem.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de lista sem ordem</title>
  </head>
  <body>
    <h1>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</h1>
    <h2>Pré-requisitos</h2>

    <ul>
      <li>Conhecimento de algum sistema operacional (Windows/Linux/MacOS X)</li>
      <li>Lógica de programação</li>
    </ul>
  </body>
</html>
```

Código HTML 2.57: lista-sem-ordem.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao42.zip

43 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/lista-sem-ordem.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/lista-sem-ordem.html.

44 Crie um novo documento HTML chamado **listas-aninhadas.html** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de listas aninhadas</title>
  </head>
  <body>
    <h1>Exemplo de listas aninhadas</h1>
    <h2>Continentes</h2>
    <ul>
      <li>
        Europa
        <ul>
          <li>Portugal</li>
          <li>França</li>
          <li>Alemanha</li>
        </ul>
      </li>
      <li>
```

```
20          Ásia
21          <ul>
22            <li>Japão</li>
23            <li>China</li>
24            <li>Índia</li>
25          </ul>
26        </li>
27      </ul>
28      <h2>Cronograma da minha viagem</h2>
29      <ol>
30        <li>
31          Europa
32          <ol>
33            <li>Portugual</li>
34            <li>França</li>
35            <li>Alemanha</li>
36          </ol>
37        </li>
38        <li>
39          Ásia
40          <ol>
41            <li>Japão</li>
42            <li>China</li>
43            <li>Índia</li>
44          </ol>
45        </li>
46      </ol>
47    </body>
48  </html>
```

Código HTML 2.58: listas-aninhadas.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao44.zip

45 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/listas-aninhadas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/listas-aninhadas.html.

46 Crie um novo documento HTML chamado **restaurante.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Menu - K19 Pizzaria</title>
6    </head>
7    <body>
8      <h1>Menu - K19 Pizzaria</h1>
9
10     <dl>
11       <dt>À moda da casa</dt>
12       <dd>
13         Presunto coberto com mussarela, ovos e palmito.
14       </dd>
15       <dt>À moda do pizzaiolo</dt>
16       <dd>
17         Mussarela, presunto, ovos e bacon.
18       </dd>
19       <dt>Aliche</dt>
20       <dd>
```

```
21          Aliche, parmesão e rodelas de tomate.
22        </dd>
23      </dl>
24    </body>
25  </html>
```

Código HTML 2.59: restaurante.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao46.zip

47 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/restaurante.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/restaurante.html.

48 Crie um novo documento HTML chamado **manual.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Operação de saque - Manual do caixa eletrônico - K19 Bank</title>
6    </head>
7    <body>
8      <h1>Operação de saque - Manual do caixa eletrônico - K19 Bank</h1>
9
10     <ol>
11       <li>Insira o cartão</li>
12       <li>Digite a senha</li>
13       <li>Escolha a opção de saque</li>
14       <li>Informe o valor que deseja sacar</li>
15       <li>Insira o cartão novamente</li>
16       <li>Aguarde até a liberação do dinheiro</li>
17     </ol>
18   </body>
19 </html>
```

Código HTML 2.60: manual.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao48.zip

49 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/manual.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/manual.html.

50 Crie um novo documento HTML chamado **cursos.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
```

```
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>K00 - Formação Básica - K19 Treinamentos</title>
6   </head>
7   <body>
8     <h1>K00 - Formação Básica</h1>
9
10    <ul>
11      <li>K01 - Lógica de Programação</li>
12      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
13      <li>K03 - SQL e Modelo Relacional</li>
14    </ul>
15  </body>
16 </html>
```

Código HTML 2.61: cursos.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao50.zip

51 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/cursos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/cursos.html.

52 Crie um novo documento HTML chamado **formacoes.html** no projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Formações - K19 Treinamentos</title>
6    </head>
7    <body>
8      <h1>Formações - K19 Treinamentos</h1>
9      <ul>
10       <li>
11         K00 - Formação Básica
12         <ul>
13           <li>K01 - Lógica de Programação</li>
14           <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
15           <li>K03 - SQL e Modelo Relacional</li>
16         </ul>
17       </li>
18       <li>
19         K10 - Formação Desenvolvedor Java
20         <ul>
21           <li>K11 - Orientação a Objetos em Java</li>
22           <li>K12 - Desenvolvimento Web com JSF2 e JPA2</li>
23         </ul>
24       </li>
25       <li>
26         K30 - Formação Desenvolvedor .NET
27         <ul>
28           <li>K31 - C# e Orientação a Objetos</li>
29           <li>K32 - Desenvolvimento Web com ASP.NET MVC</li>
30         </ul>
31       </li>
32     </ul>
33   </body>
34 </html>
```

Código HTML 2.62: formacoes.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao52.zip

53 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/formacoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/formacoes.html.

Iframes

Um documento HTML pode conter outros documentos HTML. Para adicionar um documento HTML dentro de outro, devemos utilizar o elemento **iframe**. Esse elemento possui o atributo **src**. Esse atributo indica o caminho absoluto ou relativo de um documento HTML. O conteúdo do elemento **iframe** é utilizado pelos navegadores que não oferecem suporte a esse elemento. Normalmente, o **iframe** é definido pelos navegadores como **inline-level element**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de frames</title>
6    </head>
7    <body>
8      <h1>Exemplo de frames</h1>
9      <iframe src="http.//www.k19.com.br">
10       Esse navegador não suporta o elemento <i>iframe</i>.
11     </iframe>
12     <iframe src="http.//www.usp.br">
13       Esse navegador não suporta o elemento <i>iframe</i>.
14     </iframe>
15   </body>
16 </html>
```

Código HTML 2.63: Exemplo de frames

Figura 2.34: Exemplo de frames

Importante
No início do crescimento da Internet, um grande número de sites utilizavam iframes. Hoje em dia, os especialistas em design web não recomendam mais utilização desse recurso. Então, evite a utilização de iframes.

Links

Normalmente, um site é formado por um conjunto de páginas que estão interligadas de alguma forma. Para permitir que um usuário navegue de uma página para outra, devemos utilizar os links. Basicamente, um link faz a ligação de uma página para outra do mesmo site (link interno) ou para uma página de outro site (link externo).

Para criarmos um link, devemos utilizar o elemento **a**. Esse elemento possui um atributo chamado **href**. O valor desse atributo indica o caminho relativo ou absoluto de uma outra página. Normalmente, o **a** é definido pelos navegadores como **inline-level element**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de links</title>
6    </head>
7    <body>
8      <ul>
9        <li>
10         <a href="pagina2.html">Exemplo de link com caminho relativo</a>
11       </li>
12       <li>
13         <a href="outros/pagina3.html">Outro exemplo de link com caminho relativo</a>
14       </li>
15       <li>
16         <a href="http.//www.k19.com.br">Exemplo de link com caminho absoluto</a>
17       </li>
18     </ul>
19   </body>
20 </html>
```

Código HTML 2.64: Exemplo de links

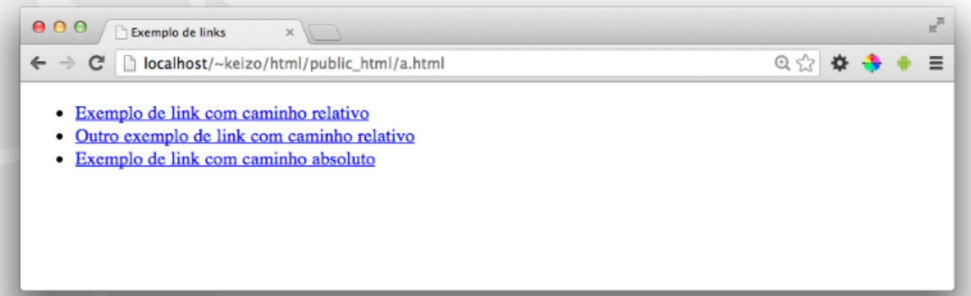


Figura 2.35: Exemplo de links

Além do atributo **href**, podemos utilizar o atributo **target** para informar onde o destino de um link deve ser aberto.

- **_blank**: Abre o destino do link em uma nova janela ou aba.
- **_self**: Abre o destino do link na mesma janela ou no mesmo frame que exibe o documento que contém o link.
- **_parent**: Abre o destino do link na janela ou no frame onde está contido o frame que exibe o documento que contém o link.
- **_top**: Abre o destino do link na janela que é "ancestral" do frame que exibe o documento que contém o link.

Um link com **target _self** possui o mesmo comportamento de um link com **target _top** se esses links estiverem em um documento HTML que não esteja dentro de outro documento HTML.

Dentro de um documento HTML com diversos frames, podemos definir que o destino de um determinado link deve ser aberto em um determinado frame. Para realizarmos tal tarefa, o valor do atributo **target** deve ser o nome do frame onde o destino do link deve ser aberto.

Por padrão, o destino de um link é aberto na mesma página ou frame onde ele está contido. Em outras palavras, se o atributo `target` não for definido explicitamente, o padrão é o comportamento do **_self**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de uso da tag a com o atributo target</title>
6    </head>
7    <body>
8      <ul>
9        <li>
10         <a href="pagina1.html" target="_blank">Abre em outra janela ou aba</a>
11       </li>
12       <li>
13         <a href="pagina2.html" target="_self">Abre na mesma janela ou aba</a>
14       </li>
15       <li>
16         <a href="pagina3.html">Abre na mesma janela ou aba</a>
17       </li>
18     </ul>
19   </body>
20 </html>
```

Código HTML 2.65: Exemplo de links com target

Exercícios de Fixação

54 Crie dois arquivos chamados **links-origem.html** e **links-destino.html** dentro do projeto **html** em **Site Root**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
```

```
5     <title>Exemplo de links - origem</title>
6   </head>
7   <body>
8     <h1>Exemplo de links - origem</h1>
9     <ul>
10      <li><a href="http.//www.k19.com.br" target="_blank">Link externo</a></li>
11      <li><a href="links-destino.html" target="_self">Link interno</a></li>
12      <li><a href="links-destino.html" target="_top">Link interno</a></p>
13      <li><a href="links-destino.html">Link interno</a></li>
14    </ul>
15  </body>
16 </html>
```

Código HTML 2.66: links-origem.html

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de links - destino</title>
6    </head>
7    <body>
8      <h1>Exemplo de links - destino</h1>
9    </body>
10 </html>
```

Código HTML 2.67: links-destino.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao54.zip

55 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/links-origem.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/links-origem.html.

Âncoras

Além de criar links para outras páginas, podemos criar um link para uma determinada seção de um documento HTML. Esse recurso é chamado de **ancoragem**. O primeiro passo para utilizar esse recurso é identificar a seção que será o destino desse link. Essa identificação é realizada com o atributo **id**. O **id** é um **atributo global**, ou seja, todos os elementos possuem esse atributo. O segundo passo é criar os links utilizando os identificadores das seções de acordo com a sintaxe do exemplo a seguir. Observe a utilização do caractere **#**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de âncoras</title>
6    </head>
7    <body>
8      <h1>Exemplo de âncoras</h1>
9      <ul>
10       <li>
11         <a href="#brasil">Brasil</a>
```

```
12        </li>
13        <li>
14          <a href="#historia">História</a>
15        </li>
16        <li>
17          <a href="#geografia">Geografia</a>
18        </li>
19        <li>
20          <a href="#demografia">Demografia</a>
21        </li>
22        <li>
23          <a href="http.//pt.wikipedia.org/wiki/Brasil#Economia">Economia</a>
24        </li>
25      </ul>
26
27      <h2 id="brasil">Brasil</h2>
28      ...
29
30      <h3 id="historia">História</h3>
31      ...
32
33      <h3 id="geografia">Geografia</h3>
34      ...
35
36      <h3 id="demografia">Demografia</h3>
37      ...
38    </body>
39  </html>
```

Código HTML 2.68: Exemplo de âncoras

Figura 2.36: Exemplo de âncoras

Exercícios de Fixação

56 Crie um documento HTML chamado **ancoras1.html** no projeto **html** em **Site Root**. Adicione um link para uma seção dentro desse documento. Dica: insira conteúdo suficiente para que a barra de rolagem vertical do navegador apareça.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de âncoras</title>
  </head>
  <body>
    <h1>Exemplo de âncoras</h2>
    <ul>
      <li>
        <a href="#sobre">Sobre o texto dessa página</a>
      </li>
    </ul>

    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec justo
    massa, sodales sit amet eleifend a, elementum eu nibh. Donec egestas dolor
    quis turpis dictum tincidunt. Donec blandit tempus velit, sit amet
    adipiscing velit consequat placerat. Curabitur id mauris.</p>

    <p> ... </p>

    <p> ... </p>

    <p> ... </p>

    <p>At nisi imperdiet lacinia. Ut quis arcu at nisl ornare viverra.
    Duis vel tristique tellus. Maecenas ultrices placerat tortor. Pellentesque
    feugiat accumsan commodo. Proin non urna justo, id pulvinar lacus.</p>

    <h2 id="sobre">Sobre o texto dessa página</h2>
    <p>
      O texto dessa página foi gerado através do site.
      <a href="http.//www.lipsum.com/">http.//www.lipsum.com/</a>
    </p>
  </body>
</html>
```

Código HTML 2.69: ancoras1.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao56.zip

57 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/ancoras1.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/ancoras1.html.

58 Crie um novo arquivo chamado **ancoras2.html** no projeto **html** em **Site Root**. Identifique uma seção com o nome **ok**. Dica: insira conteúdo suficiente para que a barra de rolagem vertical do

navegador apareça.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício sobre âncoras</title>
  </head>
  <body>
    <h1>Ancora página 2</h1>

    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec justo
    massa, sodales sit amet eleifend a, elementum eu nibh. Donec egestas dolor
    quis turpis dictum tincidunt. Donec blandit tempus velit, sit amet
    adipiscing velit consequat placerat. Curabitur id mauris </p>
    <p> ... </p>
    <p> ... </p>
    <p> ... </p>

    <p> at nisi imperdiet lacinia. Ut quis arcu at nisl ornare viverra.
    Duis vel tristique tellus. Maecenas ultrices placerat tortor. Pellentesque
    feugiat accumsan commodo. Proin non urna justo, id pulvinar lacus.</p>

    <h2 id="ok">OK</h2>

    <p>Se você chegou aqui deu tudo certo! .)</p>
  </body>
</html>
```

Código HTML 2.70: ancoras2.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao58.zip

59 Crie um novo link no arquivo **ancora1.html** que aponte para âncora **ok** do arquivo **ancoras2.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de âncoras</title>
  </head>
  <body>
    <h1>Exemplo de âncoras</h2>
    <ul>
      <li>
        <a href="#sobre">Sobre o texto dessa página</a>
      </li>
      <li>
        <a href="ancoras2.html#ok">Seção OK</a>
      </li>
    </ul>

    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec justo
    massa, sodales sit amet eleifend a, elementum eu nibh. Donec egestas dolor
    quis turpis dictum tincidunt. Donec blandit tempus velit, sit amet
    adipiscing velit consequat placerat. Curabitur id mauris.</p>

    <p> ... </p>

    <p> ... </p>

    <p> ... </p>

    <p>At nisi imperdiet lacinia. Ut quis arcu at nisl ornare viverra.
    Duis vel tristique tellus. Maecenas ultrices placerat tortor. Pellentesque
    feugiat accumsan commodo. Proin non urna justo, id pulvinar lacus.</p>

```

```
33      <h2 id="sobre">Sobre o texto dessa página</h2>
34      <p>
35        O texto dessa página foi gerado através do site.
36        <a href="http.//www.lipsum.com/">http.//www.lipsum.com/</a>
37      </p>
38    </body>
39  </html>
```

Código HTML 2.71: ancoras1.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao59.zip

60 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/ancoras1.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/ancoras1.html.

Imagens

Certamente, os sites seriam muito entediantes se não fosse possível adicionar imagens ao conteúdo das páginas. Podemos adicionar imagens em documento HTML com o elemento **img**. Esse elemento possui um atributo chamado **src**. Esse atributo indica o caminho absoluto ou relativo da imagem que queremos adicionar. Normalmente, o **img** é definido pelos navegadores como **inline-level element**.

O elemento **img** possui um atributo obrigatório chamado **alt**. Esse atributo define um texto alternativo que pode ser utilizado, por exemplo, se houver um problema ao carregar a imagem ou por softwares de leitura de tela.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de imagens</title>
6    </head>
7    <body>
8      <h1>K19 Treinamentos</h1>
9      <img src="http.//www.k19.com.br/css/img/main-header-logo.png" alt="K19"/>
10
11     <h2>Cursos</h2>
12     <ul>
13       <li>
14         <img src="http.//www.k19.com.br/css/img/k01-logo-large.png" alt="K01"/>
15         K01 - Lógica de Programação
16       </li>
17       <li>
18         <img src="http.//www.k19.com.br/css/img/k02-logo-large.png" alt="K02"/>
19         K02 - Desenvolvimento Web com HTML, CSS e JavaScript
20       </li>
21       <li>
22         <img src="http.//www.k19.com.br/css/img/k03-logo-large.png" alt="K03"/>
23         K03 - SQL e Modelo Relacional
24       </li>
25       <li>
26         <img src="http.//www.k19.com.br/css/img/k11-logo-large.png" alt="K11"/>
27         K11 - Orientação a Objetos em Java
```

```
28        </li>
29        <li>
30          <img src="http.//www.k19.com.br/css/img/k12-logo-large.png" alt="K12"/>
31          K12 - Desenvolvimento Web com JSF2 e JPA2
32        </li>
33      </ul>
34    </body>
35  </html>
```

Código HTML 2.72: Exemplo de imagens

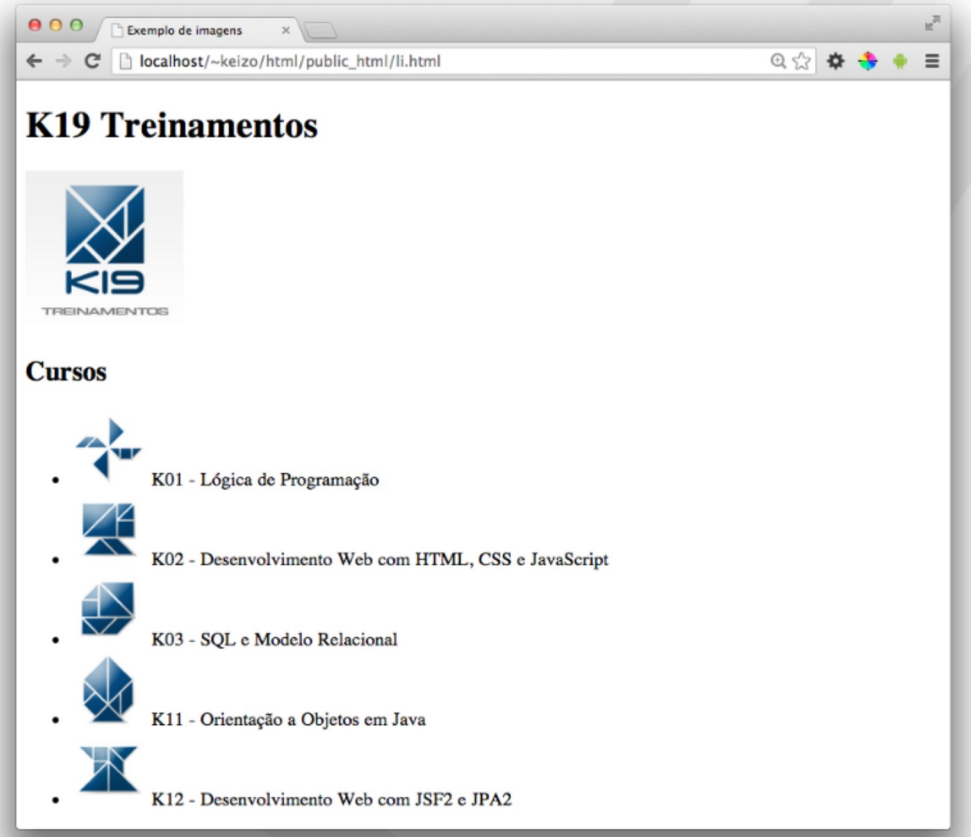


Figura 2.37: Exemplo de imagens

Também podemos definir um **tooltip** para uma imagem com o atributo **title**.

```
1 <img src="http.//www.k19.com.br/css/img/main-header-logo.png" alt="K19" title="K19"/>
```

Código HTML 2.73: Exemplo de imagens

URLs absolutas e relativas

Os links e as imagens podem ser adicionados em um documento HTML com URLs absolutas ou relativas.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>URLs absolutas e relativas</title>
  </head>
  <body>
    <img src="imagens/k19-logo.png" alt="K19">
    <a href="paginas/cursos.html">Cursos</a>
  </body>
</html>
```

Código HTML 2.74: http://www.k19.com.br/index.html

No exemplo acima, a URL do documento HTML é **http://www.k19.com.br/index.html**. Nesse documento, uma imagem e um link foram adicionados com URLs relativas (**imagens/k19-logo.png** e **paginas/cursos.html**). Por padrão, essas URLs são relativas a URL do documento HTML que contém a imagem e o link. Dessa forma, a URL da imagem é **http://www.k19.com.br/imagens/k19-logo.png** e a do link é **http://www.k19.com.br/paginas/cursos.html**.

A mesma imagem e o mesmo link podem ser adicionados com URL absoluta. Veja o exemplo a seguir.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>URLs absolutas e relativas</title>
  </head>
  <body>
    <img src="http.//www.k19.com.br/imagens/k19-logo.png" alt="K19">
    <a href="http.//www.k19.com.br/paginas/cursos.html">Cursos</a>
  </body>
</html>
```

Código HTML 2.75: http://www.k19.com.br/index.html

Podemos definir uma URL base para as imagens e os links adicionados em um documento HTML com URLs relativas. A URL base é definida no valor do atributo **href** do elemento **base**. Esse elemento deve ser adicionado no conteúdo do elemento **head**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <base href="http.//www.k19.com.br/site">
    <title>URLs absolutas e relativas</title>
  </head>
  <body>
    <img src="imagens/k19-logo.png" alt="K19">
    <a href="paginas/cursos.html">Cursos</a>
  </body>
</html>
```

Código HTML 2.76: www.k19.com.br/index.html

No exemplo acima, a URL da imagem é **http://www.k19.com.br/site/imagens/k19-logo.png** e a do link é **http://www.k19.com.br/site/paginas/cursos.html**.

Exercícios de Fixação

61 Crie um documento HTML em um arquivo chamado **imagens.html** no projeto **html** em **Site Root** com algumas imagens.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de imagens</title>
  </head>
  <body>
    <h1>K19 Treinamentos</h1>
    <img src="http.//www.k19.com.br/css/img/main-header-logo.png" alt="K19"/>

    <h2>Cursos</h2>
    <ul>
      <li>
        <img src="http.//www.k19.com.br/figs/k01-logo-large.png" alt="K01"/>
        K01 - Lógica de Programação
      </li>
      <li>
        <img src="http.//www.k19.com.br/figs/k02-logo-large.png" alt="K02"/>
        K02 - Desenvolvimento Web com HTML, CSS e JavaScript
      </li>
      <li>
        <img src="http.//www.k19.com.br/figs/k03-logo-large.png" alt="K03"/>
        K03 - SQL e Modelo Relacional
      </li>
      <li>
        <img src="http.//www.k19.com.br/figs/k11-logo-large.png" alt="K11"/>
        K11 - Orientação a Objetos em Java
      </li>
      <li>
        <img src="http.//www.k19.com.br/figs/k12-logo-large.png" alt="K12"/>
        K12 - Desenvolvimento Web com JSF2 e JPA2
      </li>
    </ul>
  </body>
</html>
```

Código HTML 2.77: imagens.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao61.zip

62 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/imagens.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/imagens.html.

Tabelas

Suponha que você esteja desenvolvendo o site de uma empresa e precisa apresentar alguns relatórios em documentos HTML. Considere que os dados desses relatórios são obtidos a partir de planilhas. Daí surge a necessidade de exibir dados de forma tabular em páginas web.

Para resolver esse problema, podemos utilizar o elemento **table**. Esse elemento permite apresentar dados de forma tabular. As linhas de uma tabela são definidas com o elemento **tr**, as células de títulos com o elemento **th** e as células de dados com o elemento **td**. Os elementos **th** e **td** possuem um atributo chamado **colspan** e outro chamado **rowspan**. O **colspan** define quantas colunas uma célula deve ocupar e o **rowspan** define quantas linhas uma célula deve ocupar. Normalmente, o **table** é definido pelos navegadores como **block-level element**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de tabela</title>
6    </head>
7    <body>
8      <h1>Carros</h1>
9      <table>
10       <tr>
11         <th>Marca</th>
12         <th>Modelo</th>
13         <th>Ano</th>
14       </tr>
15       <tr>
16         <td>Toyota</td>
17         <td>Corolla</td>
18         <td>2010</td>
19       </tr>
20       <tr>
21         <td>Honda</td>
22         <td>Civic</td>
23         <td>2011</td>
24       </tr>
25       <tr>
26         <td>Mitsubishi</td>
27         <td>Lancer</td>
28         <td>2012</td>
29       </tr>
30       <tr>
31         <td colspan="3">Última atualização. 06/2012</td>
32       </tr>
33     </table>
34   </body>
35 </html>
```

Código HTML 2.78: Exemplo de tabela

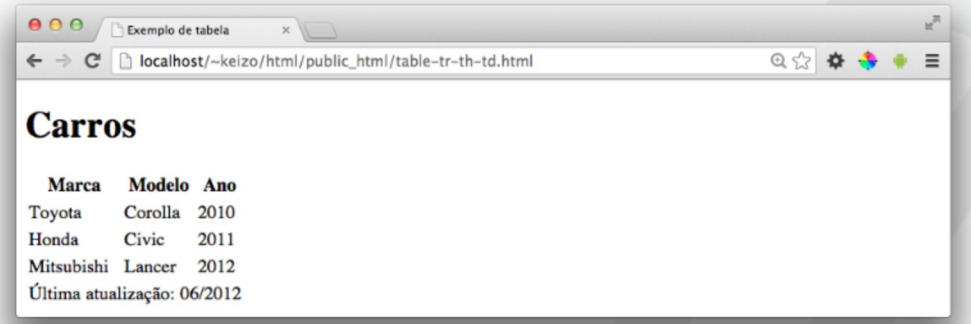


Figura 2.38: Exemplo de tabela

Cabeçalho, corpo e rodapé

Para melhorar a semântica das tabelas, podemos definir explicitamente o cabeçalho, o corpo e o rodapé de uma tabela através dos elementos **thead**, **tbody** e **tfoot** respectivamente.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de tabela</title>
6    </head>
7    <body>
8      <h1>Carros</h1>
9
10     <table>
11       <thead>
12         <tr>
13           <th>Marca</th>
14           <th>Modelo</th>
15           <th>Ano</th>
16         </tr>
17       </thead>
18       <tfoot>
19         <tr>
20           <td colspan="3">Última atualização. 06/2012</td>
21         </tr>
22       </tfoot>
23       <tbody>
24         <tr>
25           <td>Toyota</td>
26           <td>Corolla</td>
27           <td>2010</td>
28         </tr>
29         <tr>
30           <td>Honda</td>
31           <td>Civic</td>
32           <td>2011</td>
33         </tr>
34         <tr>
35           <td>Mitsubishi</td>
36           <td>Lancer</td>
37           <td>2012</td>
38         </tr>
39       </tbody>
40     </table>
41   </body>
42 </html>
```

Código HTML 2.79: Exemplo de tabela

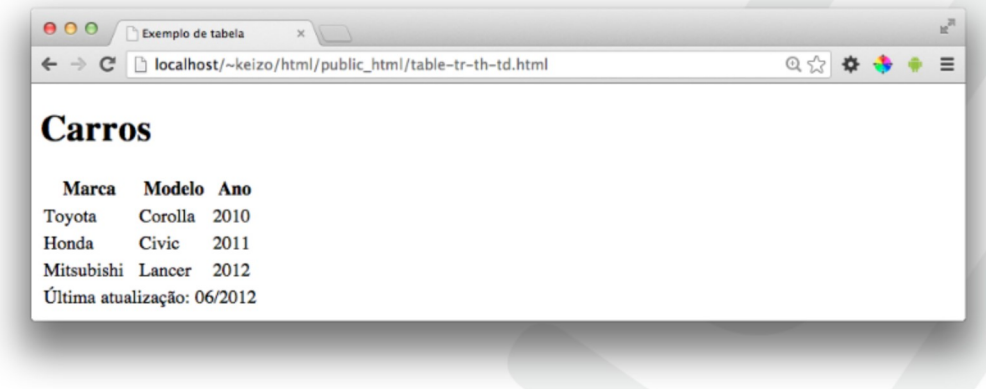


Figura 2.39: Exemplo de tabela

Repare que visualmente não mudou absolutamente nada. Além disso, apareceram mais alguns elementos: **thead**, **tfoot** e **tbody**.

Qual é a semântica desses elementos?

- **thead**: define o cabeçalho da tabela
- **tfoot**: define o rodapé da tabela
- **tbody**: define o corpo da tabela

Por que complicar? Qual o motivo da existência desses elementos?

- O elemento **thead**, assim como o **tfoot**, servem para agrupar as linhas de cabeçalho e de rodapé, respectivamente.
- O código fica mais claro.
- Facilita a aplicação de estilos CSS (através do seletor de elemento)
- Os navegadores podem utilizar barras de rolagem para exibir o conteúdo de uma tabela longa. Essa característica não é obrigatória.
- Na impressão de uma tabela longa, a ferramenta utilizada pode replicar o cabeçalho e o rodapé em todas as páginas. Essa característica não é obrigatória.

Títulos

Também podemos definir um título para uma tabela através do elemento **caption**. Esse elemento deve ser o primeiro filho do elemento **table**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>Exemplo de título de tabela</title>
6   </head>
```

```
7   <body>
8     <table>
9       <caption>Carros</caption>
10      ...
11    </table>
12  </body>
13 </html>
```

Código HTML 2.80: Exemplo de título de tabela

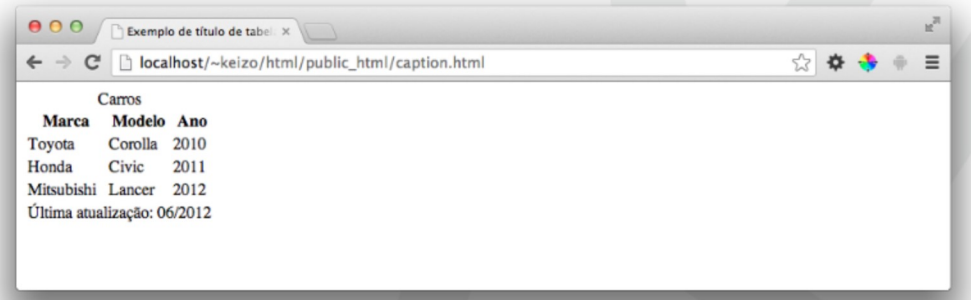


Figura 2.40: Exemplo de título de tabela

Agrupando colunas

Podemos dividir as colunas de uma tabela em grupos. Normalmente, o principal motivo para estabelecer essa divisão é poder depois definir formatações específicas para cada grupo de colunas. Para dividir em grupos as colunas de uma tabela, devemos utilizar os elementos **colgroup** e **col**. O atributo **span** do elemento **col** define a quantidade de colunas de um determinado grupo.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exemplo de agrupamento de colunas</title>
6    </head>
7    <body>
8      <h1>Carros</h1>
9
10     <table>
11       <colgroup>
12         <!-- Agrupando a primeira e a segunda coluna -->
13         <col span="2" style="background-color.gray">
14         <!-- Agrupando apenas a terceira coluna -->
15         <col style="background-color.yellow">
16       </colgroup>
17       ...
18     </table>
19   </body>
20 </html>
```

Código HTML 2.81: Exemplo de agrupamento de colunas

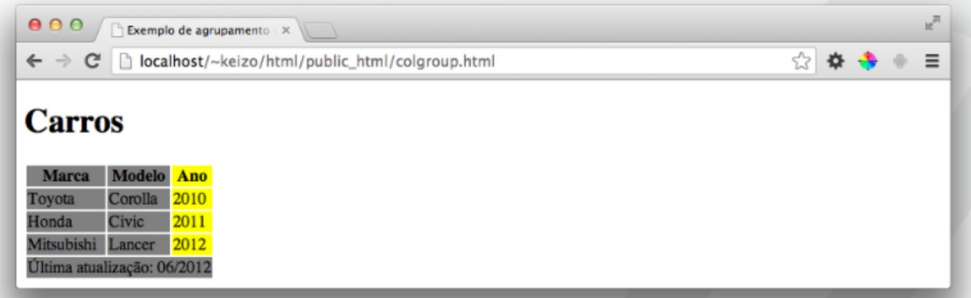


Figura 2.41: Exemplo de agrupamento de colunas

Exercícios de Fixação

63 Crie uma página HTML em um arquivo chamado **tabela.html** no projeto **html** em **Site Root** que contenha uma tabela de acordo com a imagem abaixo:

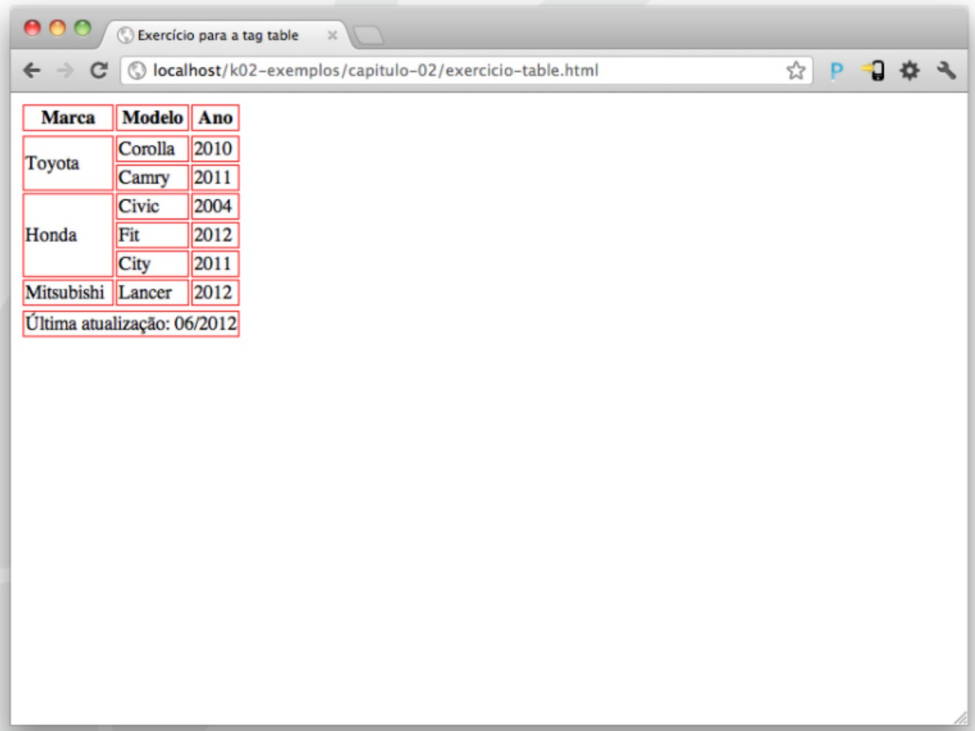


Figura 2.42: Exercício para a tag `table`

As linhas vermelhas foram colocadas na imagem apenas para facilitar a visualização do problema.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de tabela</title>
  </head>
  <body>
    <table>
      <thead>
        <tr>
          <th>Marca</th>
          <th>Modelo</th>
          <th>Ano</th>
        </tr>
      </thead>
      <tfoot>
        <tr>
          <td colspan="3">Última atualização. 06/2012</td>
        </tr>
      </tfoot>
      <tbody>
        <tr>
          <td rowspan="2">Toyota</td>
          <td>Corolla</td>
          <td>2010</td>
        </tr>
        <tr>
          <td>Camry</td>
          <td>2011</td>
        </tr>

        <tr>
          <td rowspan="3">Honda</td>
          <td>Civic</td>
          <td>2004</td>
        </tr>
        <tr>
          <td>Fit</td>
          <td>2012</td>
        </tr>
        <tr>
          <td>City</td>
          <td>2011</td>
        </tr>

        <tr>
          <td>Mitsubishi</td>
          <td>Lancer</td>
          <td>2012</td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```

Código HTML 2.82: tabela.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao63.zip

64 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/tabela.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/tabela.html.

65 Altere o arquivo **tabela.html** do projeto **html** que está em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de tabela</title>
  </head>
  <body>
    <table>
      <caption>Carros</caption>
      <colgroup>
        <col span="2" style="background-color.gray">
        <col style="background-color.yellow">
      </colgroup>
      ...
    </table>
  </body>
</html>
```

Código HTML 2.83: tabela.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao65.zip

66 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/tabela.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/tabela.html.

Formulários

Alguns sites e praticamente todas as aplicações web necessitam obter informações enviadas pelos usuários. Por exemplo, considere uma empresa que deseja receber os pedidos dos seus clientes através do seu site. O site dessa empresa precisa oferecer alguma forma de interação que possibilite o recebimento de dados fornecidos pelos usuários.

Para tornar os sites e as aplicações web mais interativos, podemos utilizar formulários. Através dos formulários, os usuários podem enviar informações aos Web Servers.

Para criar um formulário, devemos utilizar o elemento **form**. Esse elemento possui um atributo chamado **action**. O valor desse atributo indica para qual endereço os dados do formulário serão enviados.

```
<form action="pagina.html">
...
</form>
```

Os formulários são compostos por caixas de texto, checkboxes, radios, caixas de seleção, botões, entre outros componentes. Provavelmente, você reconhecerá diversos desses componentes na imagem abaixo.

Figura 2.43: Componentes de formulário

Caixas de texto

Geralmente, os formulários possuem uma ou mais caixas de texto. As caixas de texto são adicionadas nos documentos HTML através do elemento **input**. Esse elemento possui um atributo chamado **type**. Para definir uma caixa de texto, o valor do atributo **type** deve ser **text**. Normalmente, o **input** é definido pelos navegadores como **inline-level element**.

```
1 <input type="text">
```

Considere um formulário com diversas caixas de texto. Quando as informações preenchidas nesse formulário chegarem ao Web Server, certamente, ele precisará saber o que foi preenchido em cada caixa. Por isso, é necessário identificar esses dados. O atributo **name** do elemento **input** é utilizado para resolver esse problema.

```
1 <input type="text" name="endereco">
2 <input type="text" name="cidade">
```

Parâmetros, GET e POST

Vamos analisar como os valores preenchidos nos formulários são enviados aos Web Servers. Considere o seguinte exemplo.

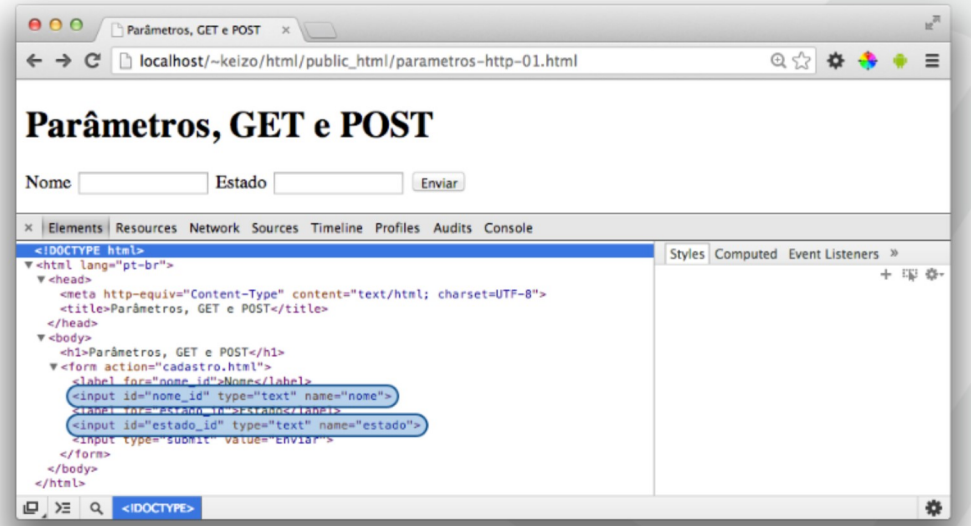


Figura 2.44: Parâmetros HTTP

O formulário acima possui duas caixas de texto. Nas duas, o atributo **name** foi definido. Quando as caixas forem preenchidas e o formulário enviado, os dados desse formulário são adicionados na requisição HTTP.

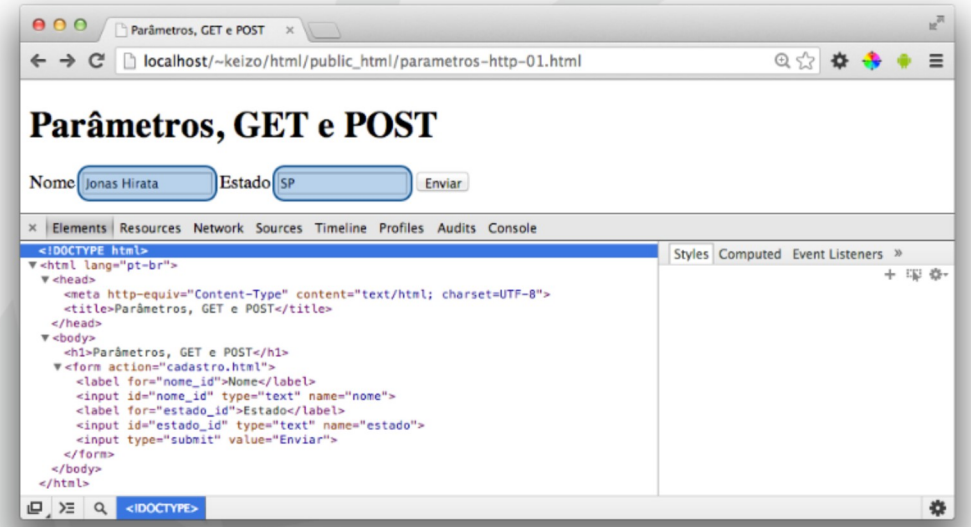


Figura 2.45: Parâmetros HTTP

Observe na imagem abaixo que os valores preenchidos nas caixas de texto são enviados como parâmetros na URL da requisição HTTP. Os nomes dos parâmetros são exatamente os nomes definidos nas caixas de texto com o valor do atributo **name**. Note também que os valores dos parâmetros ficam expostos na barra de endereço do navegador.

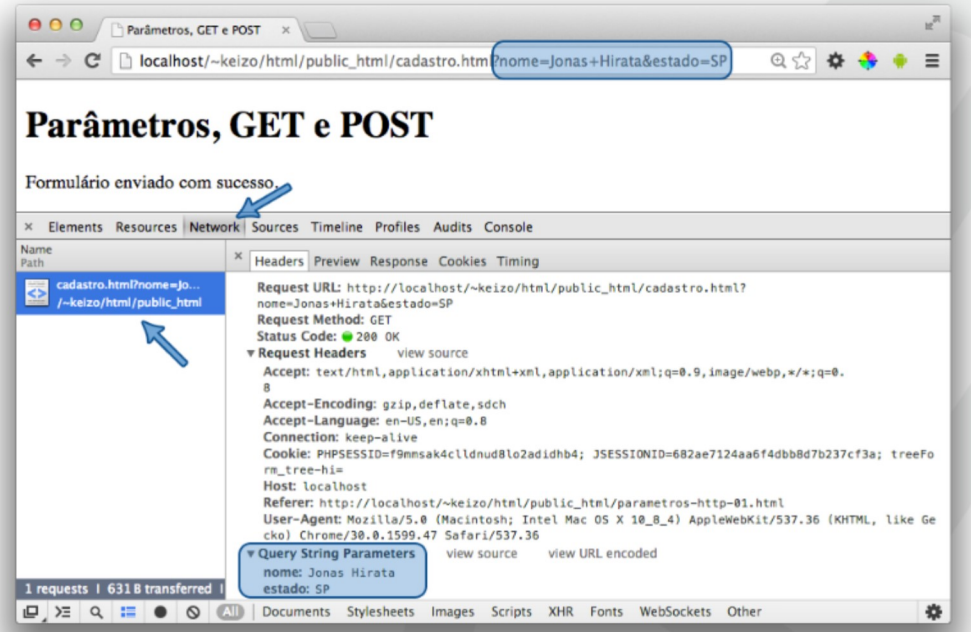


Figura 2.46: Parâmetros HTTP

Muitas vezes, não é adequado exibir os valores dos atributos na barra de endereço dos navegadores. Podemos ocultar esses valores adicionando o atributo **method** com o valor **post** no elemento **form**. Esse atributo aceita apenas dois valores: **get** e **post**. Ele define o tipo de requisição HTTP que o navegador deve realizar para enviar o formulário. Nas requisições do tipo GET, os parâmetros são adicionados na URL da requisição. Nas requisições do tipo POST, os parâmetros são adicionados no conteúdo da requisição.

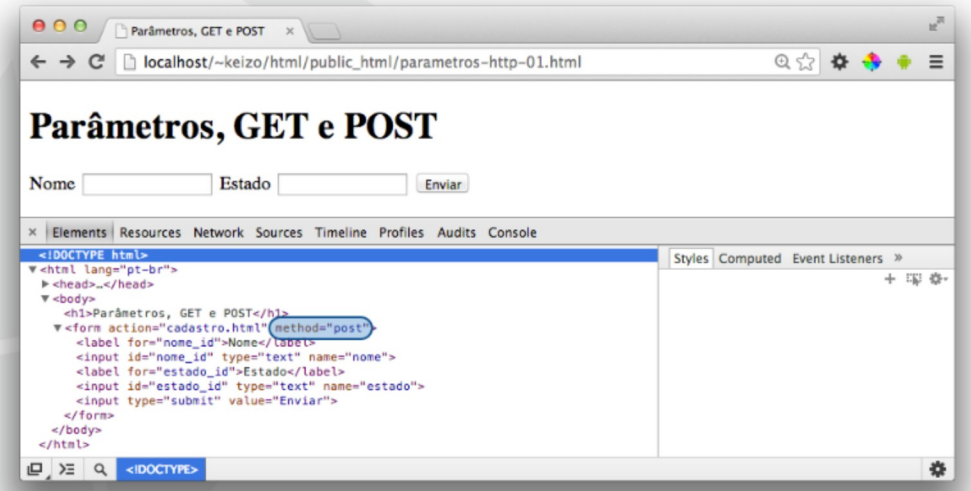


Figura 2.47: Parâmetros HTTP

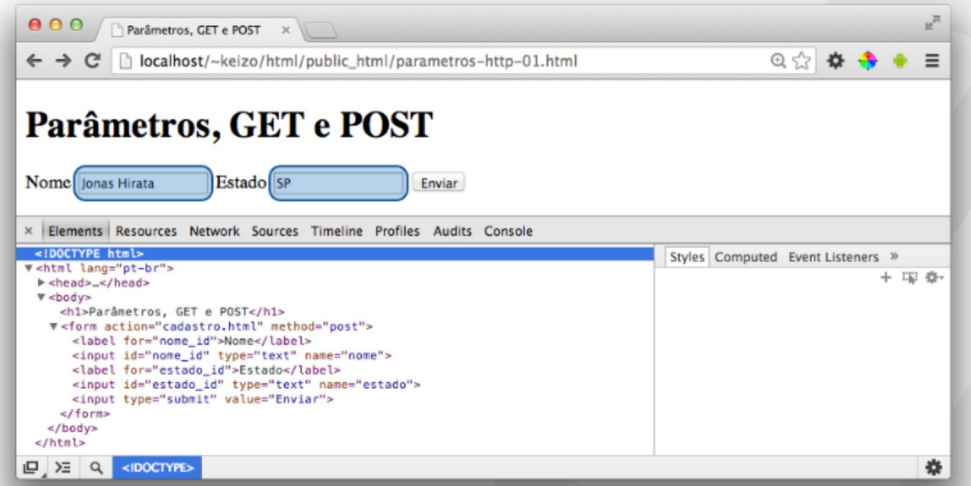


Figura 2.48: Parâmetros HTTP

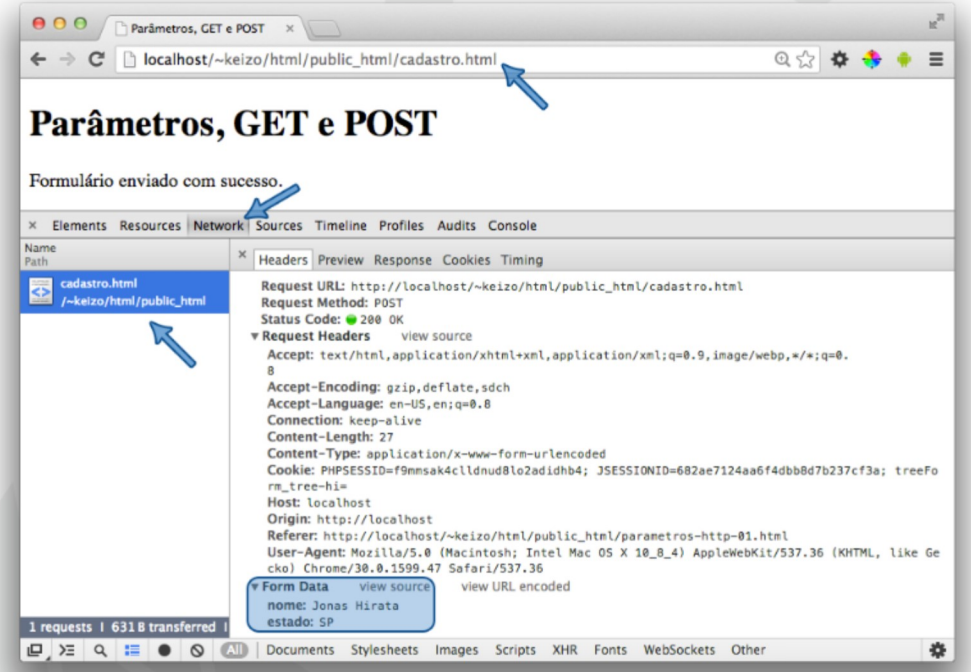


Figura 2.49: Parâmetros HTTP

Rótulos

Nos formulários, os rótulos são fundamentais para informar aos usuários quais dados devem ser preenchidos. Para adicionar um rótulo, devemos utilizar o elemento **label**. Os textos dos rótulos são definidos no conteúdo desse elemento.

```
<label>Nome. </label>
<input type="text" name="nome">
```

Para melhorar a acessibilidade dos documentos HTML, os rótulos devem ser explicitamente associados aos campos dos formulários. Para estabelecer esse vínculo, o primeiro passo é identificar os campos através do atributo **id**. O segundo passo é definir o atributo **for** do elemento **label** com o identificador do campo correspondente ao rótulo.

```
<label for="nome_id">Nome. </label>
<input type="text" name="nome" id="nome_id">
```

Figura 2.50: Rótulos

Placeholders

Como vimos, os rótulos são utilizados para informar aos usuários quais dados devem ser preenchidos nos formulários. Além dos rótulos, podemos utilizar placeholders para dar dicas ou exemplos do conteúdo que desejamos em cada caixa de entrada. Um placeholder é criado através do atributo **placeholder** do elemento **input**.

```
<label for="nome_id">Nome. </label>
<input id="nome_id" type="text" name="nome" placeholder="Digite o seu nome aqui">
```

Figura 2.51: Placeholder

```
<label for="nome_id">Nome. </label>
<input id="nome_id" type="text" name="nome" placeholder="Rafael Cosentino">
```

Figura 2.52: Placeholder

Botões de submit

Para adicionar um botão de submit em um formulário, podemos utilizar o elemento **input** com **type** igual a **submit**. Esse tipo de botão envia os dados do formulário para o Web Server. Os textos desses botões são definidos com o atributo **value**.

```
<input id="botao_id" type="submit" value="enviar">
```

Figura 2.53: Botões de submit

Outra forma de adicionar um botão de submit em um documento HTML é utilizar o elemento **button** com **type** igual a **submit**. Diferentemente do elemento **input**, o elemento **button** permite a criação de botões com imagens além de texto.

```
<button id="botao_id" type="submit">
  Enviar
  <img src="submit.png" alt="Enviar">
</button>
```

Figura 2.54: Botões de submit

Exercícios de Fixação

67 Se você estiver utilizando o **Ubuntu**, crie um arquivo chamado **parametros.php** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Parâmetros</title>
  </head>
  <body>
    <h1>Parâmetros</h1>
    <?php
    $params = file_get_contents('php.//input') . '&' . $_SERVER['QUERY_STRING'],
    $params = explode('&', $params),

    echo '<ul>',
    foreach ($params as $param) {
      if (!empty($param)) {
        echo '<li>',
        echo urldecode($param),
        echo '</li>',
      }
    }
    echo '</ul>',
    ?>
  </body>
</html>
```

Código HTML 2.93: parametros.php

Se você estiver utilizando o **Windows**, crie um arquivo chamado **parametros.asp** no projeto **html** em **Site Root**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Parâmetros</title>
  </head>
  <body>
    <h1>Parâmetros</h1>
    <ul>
    <%
      dim Params, ArrayOfParams
      If Request.Form <> "" And Request.QueryString <> "" Then
        Params = Request.Form & "&" & Request.QueryString
      ElseIf Request.Form <> "" Then
        Params = Request.Form
      Else
        Params = Request.QueryString
      End If

      ArrayOfParams = Split(Params, "&")

      For Each Param In ArrayOfParams
        Response.Write("<li>")
        Response.Write(Param)
        Response.Write("</li>")
      Next
    %>
    </ul>
  </body>
</html>
```

Código HTML 2.94: parametros.asp

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao67.zip

68 Crie um arquivo chamado **formulario-basico.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de formulário básico</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <label for="nome_id">Nome. </label>
      <input id="nome_id" type="text" name="nome" placeholder="Seu nome">

      <label for="estado_id">Estado. </label>
      <input id="estado_id" type="text" name="estado" placeholder="Seu estado">

      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

Código HTML 2.95: formulario-basico.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao68.zip

69 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/formulario-basico.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/formulario-basico.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Caixas de entrada específicas

As caixas de texto são componentes muito genéricos. Antes do HTML5, informações de natureza bem distintas eram obtidas através desses componentes. Para melhorar a semântica dos documentos HTML, tipos específicos de caixas foram adicionados no HTML5.

Caixas de busca

Assim como as caixas de texto, as caixas de busca são adicionadas nos formulário com o elemento **input**. A diferença é que o valor do atributo **type** deve ser **search** ao invés de **text**.

```
1 <input id="keywords_id" name="keywords" type="search">
```

As caixas de busca devem ser utilizadas para coletar palavras chave que serão utilizadas em algum tipo de pesquisa. A princípio não há nenhuma diferença prática entre as caixas de texto e as caixas de busca. Contudo, essa diferenciação adiciona valor semântico aos documentos HTML e possibilita, por exemplo, que os navegadores diferenciem visualmente esses dois tipos de caixas.

Figura 2.55: Caixas de busca

Caixas de números

Para coletar dados numéricos, podemos utilizar caixas específicas para números. No HTML5, há dois tipos de caixas para esse propósito. Os dois são definidos com o elemento **input**. O valor do atributo **type** é **number** para o primeiro tipo e **range** para o segundo tipo.

```
1 <input id="numero1_id" name="numero1" type="number">
2 <input id="numero2_id" name="numero2" type="range">
```

Esses dois tipos de componentes devem ser utilizados para coletar valores de sequências numéricas pré-definidas. A principal diferença entre eles é que o primeiro (**type=number**) deve oferecer um mecanismo preciso para os usuários selecionarem o valor desejado enquanto o segundo

(**type=range**) não possui essa obrigação. A imagem a seguir mostra uma possível forma dos navegadores exibirem essas caixas.

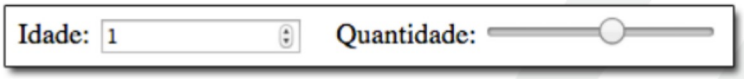


Figura 2.56: Caixas de números

Para definir a sequência dos números que podem ser selecionados pelos usuários, podemos utilizar os atributos **min**, **max** e **step**. Por exemplo, para coletar um número da sequência {0; 0,2; 0,4; 0,6; 0,8; 1}, os valores dos atributos **min**, **max** e **step** devem ser **0**, **1** e **0.2** respectivamente.

```
1 <input id="numero1_id" type="number" name="numero1" min="0" max="1" step="0.2">
2 <input id="numero2_id" type="range" name="numero2" min="0" max="1" step="0.2">
```

Caixas de email, telefone e url

No HTML5 foram definidas caixas de entradas específicas para emails, telefones e urls. Essas caixas são adicionadas com o elemento **input**. O valor do atributo **type** deve ser **email**, **tel** e **url** para emails, telefones e urls respectivamente.

```
1 <input id="email_id" name="email" type="email">
2 <input id="telefone_id" name="telefone" type="tel">
3 <input id="url_id" name="url" type="url">
```

A usabilidade das páginas web melhora com a utilização dessas caixas. Por exemplo, a configuração do teclado dos celulares ou tablets pode ser alterada de acordo com o tipo de caixa de entrada. Nas caixas de email, o caractere "**@**" pode ser adicionado ao teclado. Nas caixas de telefone, o teclado não precisa conter as letras do alfabeto. Nas caixas de url, teclas especiais como "**.com**" ou "**www**" podem ser adicionadas ao teclado.

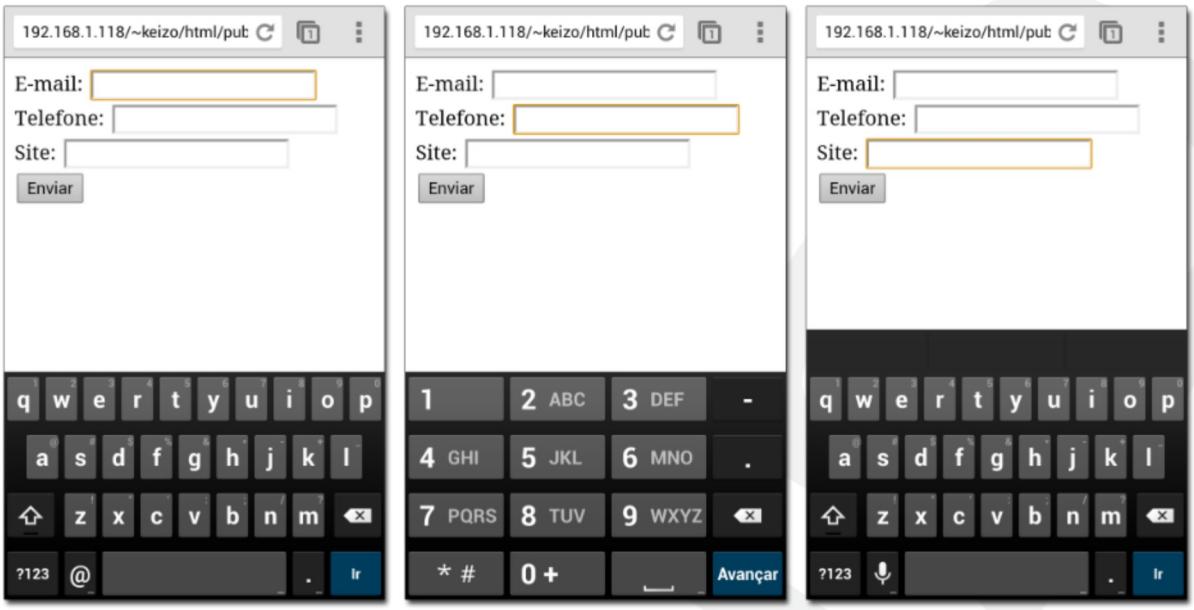


Figura 2.57: Caixas de email, telefone e url

Caixas de datas e horas

Diversos tipos de caixas de entrada para coletar datas e horas foram adicionados no HTML5. Todas essas caixas são adicionadas com o elemento **input** e o valor do atributo **type** desse elemento assumirá um dos valores listados a seguir.

- **date**: Utilizado para coletar data (dia, mês e ano) sem fuso horário.
- **datetime**: Utilizado para coletar data (dia, mês e ano) e hora (hora, minuto, segundo e fração de segundo) com fuso horário em UTC.
- **datetime-local**: Utilizado para coletar data (dia, mês e ano) e hora (hora, minuto, segundo e fração de segundo) sem fuso horário.
- **month**: Utilizado para coletar data composta por mês e ano sem fuso horário.
- **time**: Utilizado para coletar hora (hora, minuto, segundo e fração de segundo) sem fuso horário.
- **week**: Utilizado para coletar data composta por semana e ano sem fuso horário.

A criação desses componentes permite que os navegadores melhorem a usabilidade das páginas web. A forma de exibição das caixas de datas e horas pode facilitar o processo de preenchimento dos formulários. Inclusive, os navegadores podem exibir esses componentes de formas diferentes de acordo com o dispositivo (computador, celular, tablet, entre outros).

```
1 <input id="data_id" name="data" type="date">
2 <input id="data_hora_fuso_id" name="data-hora-fuso" type="datetime">
3 <input id="data_hora_id" name="data-hora" type="datetime-local">
4 <input id="mes_id" name="mes" type="month">
5 <input id="hora_id" name="hora" type="time">
6 <input id="semana_id" name="semana" type="week">
```

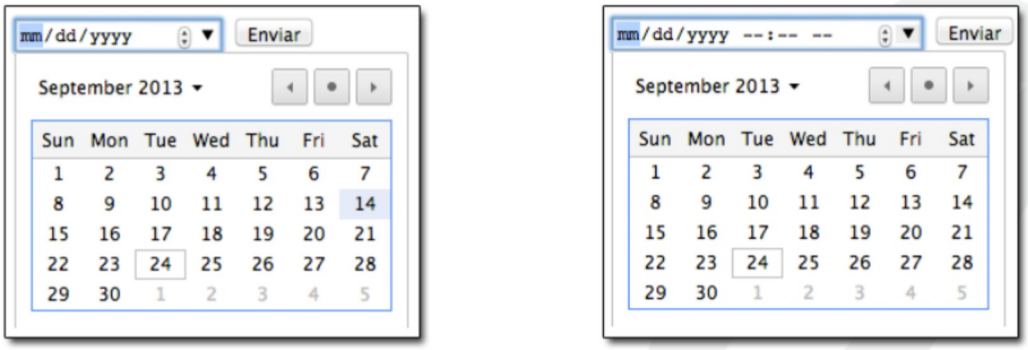


Figura 2.58: date e datetime-local

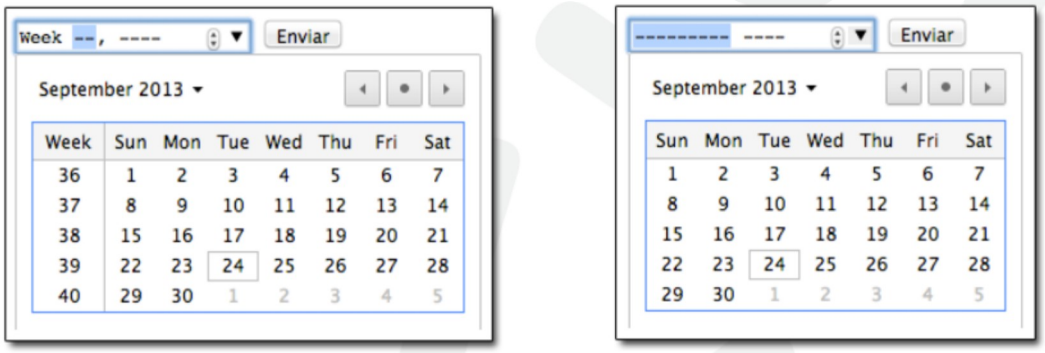


Figura 2.59: week e month

Figura 2.60: time

Caixas de senha

As senhas devem ser coletadas com caixas específicas para esse tipo de informação. Para adicionar uma caixa de senha em um formulário, devemos utilizar o elemento **input** com o valor **password** para o atributo **type**. Normalmente, os navegadores utilizam símbolos como o asterisco ou o círculo para omitir o conteúdo das caixas de senha.

```
1 <input id="senha_id" name="senha" type="password">
```

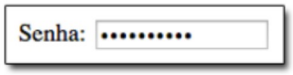


Figura 2.61: Caixa de senha

Caixas de texto longo

Para coletar um texto com várias linhas, podemos utilizar o elemento **textarea**. A quantidade de linhas de um **textarea** é definida com o atributo **rows** e a quantidade de colunas com o atributo **cols**.

Esse elemento também possui o atributo **name** que funciona como no elemento **input**.

```
<textarea id="mensagem_id" name="mensagem" rows="5" cols="50">

</textarea>
```

Podemos definir o limite de caracteres que podem ser inseridos no conteúdo do elemento **textarea** através do atributo **maxlength**.

```
<textarea id="mensagem_id" name="mensagem" maxlength="140">

</textarea>
```

Figura 2.62: Caixa de texto longo

Exercícios de Fixação

70 Crie um arquivo chamado **formulario-caixas-especificas.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de formulário com  caixas específicas</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <label for="keywords_id">Busca. </label>
      <input
        id="keywords_id"
        type="search"
        name="keywords"
        placeholder="Keywords">

      <br>

      <label for="idade_id">Idade. </label>
      <input
        id="idade_id"
        type="number"
        name="idade"
        min="18"
        max="60">

      <br>

      <label for="curtiu_id">Curtiu. </label>
      <input
        id="curtiu_id"
        type="range"
```

```
32          name="curtiu"
33          min="0"
34          max="10">
35
36        <br>
37
38        <label for="email_id">Email. </label>
39        <input
40          id="email_id"
41          type="email"
42          name="email"
43          placeholder="Seu email">
44
45        <br>
46
47        <label for="telefone_id">Telefone. </label>
48        <input
49          id="telefone_id"
50          type="tel"
51          name="telefone"
52          placeholder="Seu telefone">
53
54        <br>
55
56        <label for="site_id">Site. </label>
57        <input
58          id="site_id"
59          type="url"
60          name="site"
61          placeholder="Seu site">
62
63        <br>
64
65        <label for="nascimento_id">Nascimento. </label>
66        <input
67          id="nascimento_id"
68          type="date"
69          name="nascimento">
70
71        <br>
72
73        <label for="festa_aniversario_id">Festa de Aniversário. </label>
74        <input
75          id="festa_aniversario_id"
76          type="datetime-local"
77          name="festa_aniversario">
78
79        <br>
80
81        <label for="proximo_carnaval_mes_id">Mês do próximo Carnaval. </label>
82        <input
83          id="proximo_carnaval_mes_id"
84          type="month"
85          name="proximo_carnaval_mes">
86
87        <br>
88
89        <label for="semana_rock_in_rio_id">Semana do Rock in Rio. </label>
90        <input
91          id="semana_rock_in_rio_id"
92          type="week"
93          name="semana_rock_in_rio">
94
95        <br>
96
97        <label for="horario_de_entrada_id">Horário de entrada. </label>
98        <input
99          id="horario_de_entrada_id"
100         type="time"
101         name="horario_de_entrada">
```

```
102
103       <br>
104
105       <label for="senha_id">Senha. </label>
106       <input
107         id="senha_id"
108         type="password"
109         name="senha"
110         placeholder="Sua senha">
111
112       <br>
113
114       <label for="mensagem_id">Mensagem. </label>
115       <textarea
116         id="mensagem_id"
117         name="mensagem"
118         rows="5"
119         cols="50"
120         maxlength="150"></textarea>
121
122       <br>
123
124       <input type="submit" value="Enviar">
125     </form>
126   </body>
127 </html>
```

Código HTML 2.104: formulario-caixas-especificas.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao70.zip

71 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/formulario-caixas-especificas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/formulario-caixas-especificas.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Checkboxes e Radios

Para adicionar um checkbox em um formulário, devemos utilizar o elemento **input** com **type** igual a **checkbox**. Ao utilizar esse componente, é importante definir um valor para o atributo **value**. No envio do formulário, esse valor é transmitido ao Web Server se o checkbox correspondente estiver marcado.

```
1 <input id="sim_id" name="newsletter" type="checkbox" value="sim">
```

Eventualmente é interessante agrupar um determinado conjunto de checkboxes. Por exemplo, considere um formulário que coleta as linguagens de programação que os usuários conhecem. Para cada linguagem, podemos definir um checkbox. Para agrupar esses checkboxes, basta definir o atributo **name** com o mesmo valor para eles.

```
1 <input id="java_id" type="checkbox" value="java" name="linguagens">
2 <input id="csharp_id" type="checkbox" value="csharp" name="linguagens">
3 <input id="php_id" type="checkbox" value="php" name="linguagens">
4 <input id="ruby_id" type="checkbox" value="ruby" name="linguagens">
5 <input id="perl_id" type="checkbox" value="perl" name="linguagens">
```

Figura 2.63: Checkboxes

Para adicionar um radio em um formulário, devemos utilizar o elemento **input** com **type** igual a **radio**. Ao utilizar esse componente, é importante definir um valor para o atributo **value**. No envio do formulário, esse valor é transmitido ao Web Server se o radio correspondente estiver marcado.

```
1 <input id="masculino_id" name="sexo" type="radio" value="masculino">
```

Eventualmente é interessante agrupar um determinado conjunto de radios. Por exemplo, considere um formulário que coleta o time preferido dos usuários. Para cada time, podemos definir um radio. Para agrupar esses radios, basta definir o atributo **name** com o mesmo valor para eles.

```
1 <input id="sp_id" type="radio" value="sao-paulo" name="time-preferido">
2 <input id="barcelona_id" type="radio" value="barcelona" name="time-preferido">
3 <input id="milan_id" type="radio" value="milan" name="time-preferido">
4 <input id="mu_id" type="radio" value="manchester-united" name="time-preferido">
5 <input id="bdm_id" type="radio" value="bayern-de-munique" name="time-preferido">
```

Figura 2.64: Radios

Por padrão, quando um formulário é exibido para os usuários, os checkboxes e os radios não estão marcados. Algumas vezes, desejamos que determinados checkboxes e radios estejam marcados quando os formulários são apresentados aos usuários. Para resolver esse problema, podemos utilizar o atributo **checked** do elemento **input**. Esse atributo não precisa de valor.

```
1 <input id="java_id" type="checkbox" value="java" name="linguagens" checked>
```

```
1 <input id="sp_id" type="radio" name="time-preferido" value="sao-paulo" checked>
```

Seleção de cores

No HTML5, para coletar uma cor, podemos utilizar o elemento **input** com **type** igual a **color**.

```
<input id="cor_id" name="cor" type="color">
```

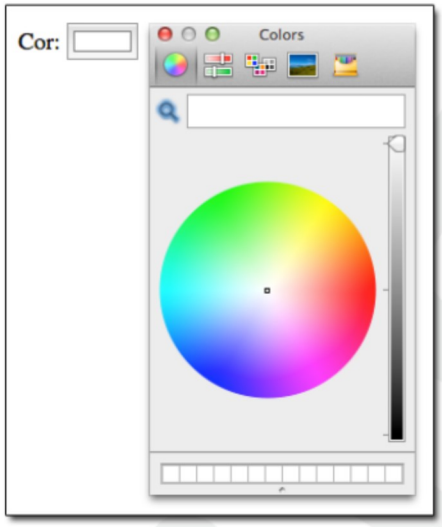


Figura 2.65: Seleção de cores

Exercícios de Fixação

72 Crie um arquivo chamado **checkboxes.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de checkboxes</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <label for="java_id">Java</label>
      <input id="java_id" type="checkbox" value="java" name="linguagens" checked>

      <label for="csharp_id">C#</label>
      <input id="csharp_id" type="checkbox" value="csharp" name="linguagens">

      <label for="php_id">PHP</label>
      <input id="php_id" type="checkbox" value="php" name="linguagens">

      <label for="ruby_id">Ruby</label>
      <input id="ruby_id" type="checkbox" value="ruby" name="linguagens" checked>

      <label for="perl_id">Perl</label>
      <input id="perl_id" type="checkbox" value="perl" name="linguagens">

      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

Código HTML 2.112: checkboxes.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao72.zip

73 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/checkboxes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/checkboxes.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

74 Crie um arquivo chamado **radios.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de radios</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <label for="sp_id">São Paulo</label>
      <input id="sp_id" type="radio" value="sao-paulo" name="time-preferido" checked>

      <label for="barcelona_id">Barcelona</label>
      <input id="barcelona_id" type="radio" value="barcelona" name="time-preferido">

      <label for="milan_id">Milan</label>
      <input id="milan_id" type="radio" value="milan" name="time-preferido">

      <label for="mu_id">Manchester United</label>
      <input id="mu_id" type="radio" value="manchester-united" name="time-preferido">

      <label for="bdm_id">Bayern de Munique</label>
      <input id="bdm_id" type="radio" value="bayern-munique" name="time-preferido">

      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

Código HTML 2.113: radios.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao74.zip

75 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/radios.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/radios.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

76 Crie um arquivo chamado **selecao-cores.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Exercício de seleção de cores</title>
6    </head>
7    <body>
8      <form action="parametros.php" method="post">
9        <label for="cor_id">Cor. </label>
10       <input id="cor_id" name="cor" type="color">
11
12       <input type="submit" value="Enviar">
13     </form>
14   </body>
15 </html>
```

Código HTML 2.114: selecao-cores.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao76.zip

77 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/selecao-cores.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/selecao-cores.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Botões

Botões genéricos

Para adicionar um botão genérico em um formulário, podemos utilizar o elemento **input** com **type** igual a **button**. As ações desse tipo de componente são definidas com JavaScript. Os textos desses botões são definidos com o atributo **value**.

```
1 <input id="botao_id" type="button" value="botão">
```

Figura 2.66: Botões genéricos

Outra forma de adicionar um botão genérico em um documento HTML é utilizar o elemento **button** com **type** igual a **button**. Diferentemente do elemento **input**, o elemento **button** permite a criação de botões com imagens além de texto.

```
1 <button id="botao_id" type="button">
2   Botão genérico
3   <img src="botao.png" alt="Botão">
4 </button>
```

Figura 2.67: Botões genéricos

Botões de reset

Para adicionar um botão de reset em um formulário, podemos utilizar o elemento **input** com **type** igual a **reset**. Esse tipo de botão reinicia os dados do formulário. Os textos desses botões são definidos com o atributo **value**.

```
1 <input id="botao_id" type="reset" value="reiniciar">
```

Figura 2.68: Botões de reset

Outra forma de adicionar um botão de reset em um documento HTML é utilizar o elemento **button** com **type** igual a **reset**. Diferentemente do elemento **input**, o elemento **button** permite a criação de botões com imagens além de texto.

```
1 <button id="botao_id" type="reset">
2   Botão de reset
3   <img src="reset.png" alt="Reiniciar">
4 </button>
```

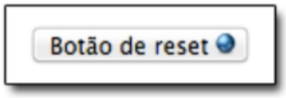


Figura 2.69: Botões de reset

Botões de upload

Para adicionar um botão de upload em um formulário, podemos utilizar o elemento **input** com **type** igual a **file**. Esse tipo de botão permite selecionar um arquivo para um eventual upload. O formulário que contém esse botão deve possuir o atributo **enctype** com o valor **multipart/form-data**.

```
<input id="botao_id" name="file" type="file">
```

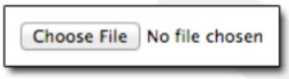


Figura 2.70: Botões de upload

Imagem como botão de submit

Uma imagem pode funcionar como um botão de submit. Para isso, devemos utilizar o elemento **input** com **type** igual a **image**. O caminho absoluto ou relativo da imagem que será utilizada deve ser definida com o atributo **src**. Um texto alternativo deve ser definido com o atributo **alt**. Esse texto pode ser utilizado caso ocorra algum problema no carregamento da imagem.

```
<form action="parametros.php" method="post" enctype="multipart/form-data">
  <input id="botao_id" alt="enviar" type="image" src="submit.png">
</form>
```

Figura 2.71: Imagem como botão de submit

Exercícios de Fixação

78 Crie um arquivo chamado **botoes.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de checkboxes</title>
  </head>
  <body>
    <form action="parametros.php" method="post" enctype="multipart/form-data">
```

```
9       <input id="file_id" type="file" name="file">
10
11      <input id="b1" type="button" value="Botão genérico">
12
13      <button id="b2" type="button">
14        Botão genérico
15        <img src="http.//www.k19.com.br/figs/botao-generico.png" alt="Botão">
16      </button>
17
18      <input id="b3" type="reset" value="Reiniciar">
19
20      <button id="b4" type="reset">
21        Botão de reset
22        <img src="http.//www.k19.com.br/figs/botao-reset.png" alt="Reiniciar">
23      </button>
24
25      <input id="b5"
26        type="image"
27        alt="enviar"
28        src="http.//www.k19.com.br/figs/submit.png">
29    </form>
30  </body>
31 </html>
```

Código HTML 2.121: botoes.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao78.zip

79 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/botoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/botoes.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Drop-down list

Muitos formulários permitem que os usuários selecionem um ou mais itens de uma lista de opções. Essa seleção pode ser realizada através de um **drop-down list**. Para adicionar esse tipo de componente, devemos utilizar o elemento **select**. Normalmente, o **select** é definido pelos navegadores como **inline-level element**.

```
1 <select id="estados_id" name="estado">
2   ...
3 </select>
```

As opções devem ser definidas no conteúdo do elemento **select** e elas são adicionadas com o elemento **option**. O conteúdo do elemento **option** é exibido para os usuários. Esse elemento possui um atributo chamado **value**. Quando o formulário for enviado, o valor do atributo **value** é transmitido ao Web Server se a opção correspondente foi selecionada pelo usuário. Se esse atributo não estiver definido, o conteúdo do elemento **option** é transmitido ao Web Server se a opção correspondente foi selecionada pelo usuário.

```
1 <select id="estados_id" name="estado">
2   <option value="SP">São Paulo</option>
3   <option value="RJ">Rio de Janeiro</option>
4   <option value="RS">Rio Grande do Sul</option>
5   <option value="RN">Rio Grande do Norte</option>
6 </select>
```

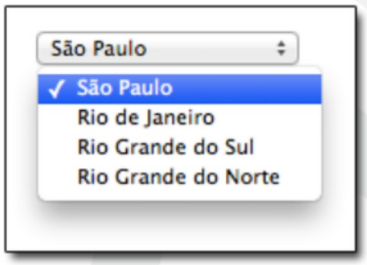


Figura 2.72: Drop-down list

Por padrão, apenas um item de um drop-down list pode ser selecionado pelos usuários. Mas, utilizando o atributo **multiple**, um ou mais itens podem ser selecionados. Geralmente, mantendo a tecla CTRL pressionada, os usuários poderão escolher dois ou mais itens.

```
1 <select id="estados_id" name="estado" multiple="multiple">
2   <option value="SP">São Paulo</option>
3   <option value="RJ">Rio de Janeiro</option>
4   <option value="RS">Rio Grande do Sul</option>
5   <option value="PR">Paraná</option>
6   <option value="RN">Rio Grande do Norte</option>
7   <option value="BA">Bahia</option>
8 </select>
```

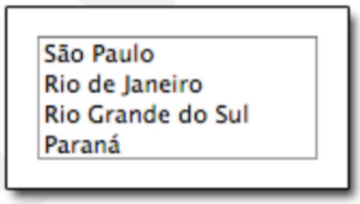


Figura 2.73: Drop-down list

Nos drop-down lists com muitas opções, é interessante agrupar as opções em categorias. Esse agrupamento pode ser realizado com o elemento **optgroup**. Esse elemento possui um atributo chamado **label**. O valor desse atributo é exibido no drop-down list.

```
1 <select id="estados_id" name="estado">
2   <optgroup label="Região Sudeste">
```

```
3      <option value="SP">São Paulo</option>
4      <option value="RJ">Rio de Janeiro</option>
5    </optgroup>
6    <optgroup label="Região Sul">
7      <option value="RS">Rio Grande do Sul</option>
8      <option value="PR">Paraná</option>
9    </optgroup>
10   <optgroup label="Região Nordeste">
11     <option value="RN">Rio Grande do Norte</option>
12     <option value="BA">Bahia</option>
13   </optgroup>
14 </select>
```

Figura 2.74: Drop-down list

Drop-down list com itens pré-selecionados

Quando um drop-down list é exibido para os usuários, opções podem estar por padrão selecionadas. O atributo **selected** do elemento **option** define as opções que devem estar selecionadas quando um drop-down list é exibido para os usuários. Esse atributo não precisa de valor.

```
1  <select id="estados_id" name="estado" multiple="multiple">
2    <optgroup label="Região Sudeste">
3      <option value="SP" selected>São Paulo</option>
4      <option value="RJ">Rio de Janeiro</option>
5    </optgroup>
6    <optgroup label="Região Sul">
7      <option value="RS" selected>Rio Grande do Sul</option>
8      <option value="PR">Paraná</option>
9    </optgroup>
10   <optgroup label="Região Nordeste">
11     <option value="RN">Rio Grande do Norte</option>
12     <option value="BA">Bahia</option>
13   </optgroup>
14 </select>
```

Exercícios de Fixação

80 Crie um arquivo chamado **drop-down-list.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
```

```
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>Exercício de checkboxes</title>
6   </head>
7   <body>
8     <form action="parametros.php" method="post">
9       <select id="estados_id" name="estados" multiple="multiple">
10        <option value="SP">São Paulo</option>
11        <option value="RJ">Rio de Janeiro</option>
12        <option value="RS">Rio Grande do Sul</option>
13        <option value="PR">Paraná</option>
14        <option value="RN">Rio Grande do Norte</option>
15        <option value="BA">Bahia</option>
16      </select>
17
18      <select id="estado_id" name="estado">
19        <optgroup label="Região Sudeste">
20          <option value="SP">São Paulo</option>
21          <option value="RJ">Rio de Janeiro</option>
22        </optgroup>
23        <optgroup label="Região Sul">
24          <option value="RS">Rio Grande do Sul</option>
25          <option value="PR">Paraná</option>
26        </optgroup>
27        <optgroup label="Região Nordeste">
28          <option value="RN">Rio Grande do Norte</option>
29          <option value="BA">Bahia</option>
30        </optgroup>
31      </select>
32
33      <input type="submit" value="Enviar">
34    </form>
35  </body>
36 </html>
```

Código HTML 2.127: drop-down-list.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao80.zip

81 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/drop-down-list.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/drop-down-list.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Fieldset

Os campos de um formulário muito longo podem ser agrupados logicamente com o elemento **fieldset**. No conteúdo desse elemento, podemos utilizar o elemento **legend** para definir a legenda do fieldset. Normalmente, o **fieldset** é definido pelos navegadores como **block-level element**.

```
<fieldset>
  <legend>Dados Pessoais</legend>
  ...
</fieldset>
<fieldset>
  <legend>Formação</legend>
  ...
</fieldset>
<fieldset>
  <legend>Experiência</legend>
  ...
</fieldset>
```

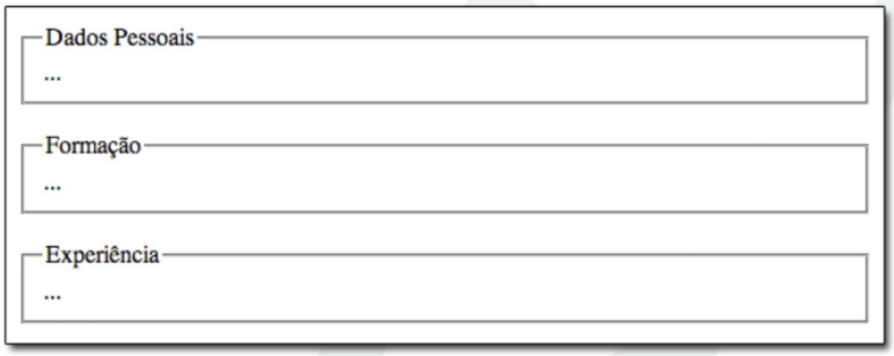


Figura 2.75: Fieldset

Exercícios de Fixação

82 Crie um arquivo chamado **fieldset.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de fieldset</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <fieldset>
        <legend>Dados Pessoais</legend>
        <label for="nome_id">Nome. </label>
        <input id="nome_id" type="text" name="nome">
      </fieldset>
      <fieldset>
        <legend>Formação</legend>
        <label for="curso_id">Curso. </label>
        <input id="curso_id" type="text" name="curso">
      </fieldset>
      <fieldset>
        <legend>Experiência</legend>
        <label for="empresa_id">Empresa. </label>
        <input id="empresa_id" type="text" name="empresa">
```

```
23        </fieldset>
24
25        <input type="submit" value="Enviar">
26      </form>
27    </body>
28  </html>
```

Código HTML 2.129: fieldset.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao82.zip

83 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/fieldset.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/fieldset.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Autocomplete

Para melhorar a usabilidade, podemos utilizar o recurso do autocomplete nas caixas de entrada. Para utilizar esse recurso, devemos criar uma lista de sugestões com o elemento **datalist**. É fundamental identificar as listas de sugestões com o atributo **id**.

```
1 <datalist id="comidas_id">
2   ...
3 </datalist>
```

As opções devem ser definidas no conteúdo do elemento **datalist** e elas são adicionadas com o elemento **option**. O atributo **value** de um elemento **option** define uma sugestão.

```
1 <datalist id="comidas_id">
2   <option value="Lasanha">
3   <option value="Pizza">
4   <option value="Sopa">
5   <option value="Salada">
6 </datalist>
```

Com a lista de sugestões criada, podemos associá-la a uma caixa de entrada através do atributo **list** do elemento **input**.

```
1 <input id="comida_id" name="comida" type="text" list="comidas_id">
2
```

```
<datalist id="comidas_id">
  <option value="Lasanha">
  <option value="Pizza">
  <option value="Sopa">
  <option value="Salada">
</datalist>
```

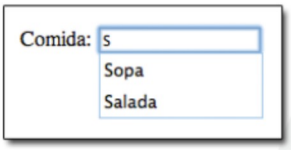


Figura 2.76: Autocomplete

Exercícios de Fixação

84 Crie um arquivo chamado **autocomplete.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de autocomplete</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <input id="comida_id" name="comida" type="text" list="comidas_id">

      <datalist id="comidas_id">
        <option value="Lasanha">
        <option value="Pizza">
        <option value="Sopa">
        <option value="Salada">
      </datalist>

      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

Código HTML 2.133: autocomplete.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao84.zip

85 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/autocomplete.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/~<USUARIO>/html/public_html/autocomplete.html`.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Campos ou botões desabilitados

Eventualmente, determinados campos ou botões de um formulário devem ser desabilitados. Para desabilitar um campo ou botão, podemos utilizar o atributo **disabled** dos elementos **input**, **select**, **textarea** e **button**. Esse atributo não precisa de valor.

```
1 <input id="nome_id" name="nome" type="text" disabled>
```

Figura 2.77: Campos desabilitados

```
1 <select id="nomes_id" disabled>
2   ...
3 </select>
```

Figura 2.78: Campos desabilitados

```
1 <textarea disabled>
2 </textarea>
```

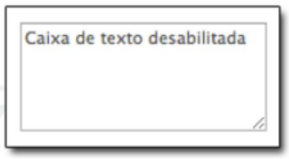


Figura 2.79: Campos desabilitados

```
1 <button type="button" disabled>
2   Botão desabilitado
3   <img src="botao.png" alt="Botão desabilitado">
4 </button>
```

Figura 2.80: Botões desabilitados

Campos fixos

Eventualmente, determinados campos de um formulário devem ser fixo, ou seja, os usuários não podem alterar o conteúdo. Para fixar o conteúdo de um campo, podemos utilizar o atributo **readonly** dos elementos **input** e **textarea**. Esse atributo não precisa de valor.

```
<input id="nome_id" name="nome" type="text" value="Marcelo Martins" readonly>
```

Figura 2.81: Campos fixos

```
<textarea readonly>
Gostaria de efetuar a matrícula no K10 - Formação Desenvolvedor Java.
</textarea>
```

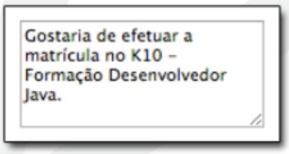


Figura 2.82: Campos fixos

Exercícios de Fixação

86 Crie um arquivo chamado **disabled-readonly.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de autocomplete</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <label for="nome_id">Nome. </label>
      <input id="nome_id" name="nome" type="text" value="Marcelo Martins" readonly>

      <label for="email_id">Email. </label>
      <input id="email_id" name="email" type="email" disabled>
```

```
14
15        <input type="submit" value="Enviar">
16      </form>
17    </body>
18  </html>
```

Código HTML 2.140: disabled-readonly.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-fixacao86.zip

87 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/disabled-readonly.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/disabled-readonly.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Validação

Alguns recursos para realizar a validação dos campos de um formulário foram adicionados no HTML5. Essas validações ocorrem antes do envio dos formulários. Por padrão, algumas validações são realizadas automaticamente de acordo com o tipo do campo. Por exemplo, os navegadores verificam se o conteúdo de uma caixa de email é um email válido.

```
1 <input id="email_id" name="email" type="email">
```

Figura 2.83: Validação de email

Os navegadores também verificam se o conteúdo de uma caixa de URL é uma url válida.

```
1 <input id="url_id" name="url" type="url">
```

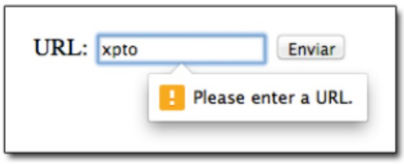


Figura 2.84: Validação de URL

Outras validações devem ser definidas explicitamente. Por exemplo, se um determinado campo é obrigatório, devemos utilizar o atributo **required**. Esse atributo não precisa de valor.

```
<input id="nome_id" name="nome" type="text" required>
```

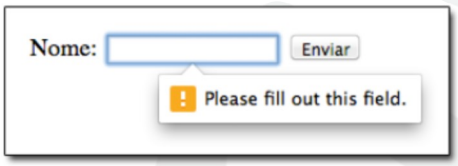


Figura 2.85: Validação de campo obrigatório

Se um determinado campo deve respeitar uma expressão regular, devemo utilizar o atributo **pattern**.

```
<input
  id="placa_id"
  name="placa"
  type="text"
  pattern="[A-Z]{3} [0-9]{4}"
  title="As placas são formadas por 3 letras ou 4 números">
```

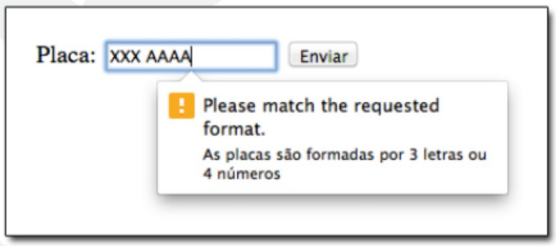


Figura 2.86: Validação com expressão regular

Se o conteúdo de um campo numérico deve estar em um determinado intervalo, podemo utilizar os atributos **min** e **max**.

```
<input
  id="idade_id"
  name="idade"
  type="number"
  min="18"
  max="60"
  title="A idade miníma é 18 e a máxima é 60">
```

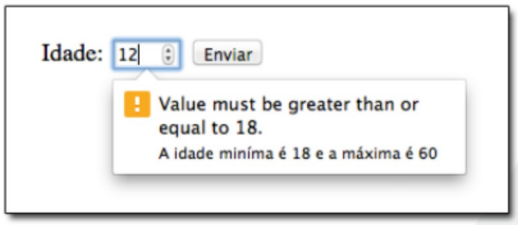


Figura 2.87: Validação de numérica de mínimo e máximo

Exercícios de Fixação

88 Crie um arquivo chamado **validacao.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de validação</title>
  </head>
  <body>
    <form action="parametros.php" method="post">
      <label for="nome_id">Nome. </label>
      <input id="nome_id" name="nome" type="text" required>

      <label for="email_id">Email. </label>
      <input id="email_id" name="email" type="email">

      <label for="url_id">Site. </label>
      <input id="url_id" name="url" type="url">

      <label for="placa_id">Placa. </label>
      <input
        id="placa_id"
        name="placa"
        type="text"
        pattern="[A-Z]{3} [0-9]{4}"
        title="As placas são formadas por 3 letras ou 4 números">

      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

Código HTML 2.146: validacao.html

Importante
No **Windows**, altere o código acima substituindo o trecho "**parametros.php**" pelo trecho "**parametros.asp**".

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao88.zip

89 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/validacao.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/validacao.html.

Utilize o **Chrome DevTools** ou o **Firebug** para verificar os parâmetros enviados através do formulário. Chame o instrutor caso tenha alguma dúvida.

Partes de um documento HTML

Normalmente, um documento HTML pode ser dividido em "pedaços". Por exemplo, é comum existir cabeçalhos, rodapés e menus de navegação na maior parte das páginas web. No HTML5, diversos elementos foram adicionados para definir especificamente os principais "pedaços" que geralmente um documento HTML possui.

header

O elemento **header** é utilizado para definir um cabeçalho de um documento HTML ou de uma seção de um documento HTML. O objetivo desse elemento é definir um conteúdo de introdução ou de navegação. Normalmente, no **header**, encontramos os elementos **h1**, **h2**, **h3**, **h4**, **h5** e **h6**. No conteúdo do **header**, também é comum encontrar sumários, formulários de busca, logos, entre outros.

```
<body>
  <header>
    <h1>Blog da K19</h1>
    <img src="k19-logo.png" alt="K19">
  </header>

  <article>
    <header>
      <h1>Novos elementos do HTML5</h1>
      <img src="html5-logo.png" alt="HTML5">
    </header>

    ...
  </article>

  ...
</body>
```

Código HTML 2.147: Exemplo de cabeçalho

footer

O elemento **footer** é utilizado para definir um rodapé para o documento HTML ou para uma seção do documento HTML. É muito comum encontrarmos em seu conteúdo informações sobre a seção a qual ele pertence como quem a escreveu, links relacionados ao conteúdo da seção e informações legais, por exemplo.

```
<body>
  ...
  <article>
    <p>Loren ipsum...</p>

```

```
6     <footer>Postado por. Jonas Hirata</footer>
7   </article>
8
9   <footer>&copy,2013 K19 Treinamentos.</footer>
10 </body>
```

Código HTML 2.148: Exemplo de rodapé

nav

O elemento **nav** é utilizado para definir um bloco com os principais links de um documento HTML. É comum, definir os principais links de um documento HTML nos rodapés. Dessa forma, o elemento **nav** aparecerá com frequência no conteúdo do elemento **footer**.

```
1  <body>
2    <header>
3      <h1>Blog da K19</h1>
4
5      <nav>
6        <a href="#">Home</a>
7        <a href="#">Últimos posts</a>
8        <a href="#">Arquivo</a>
9      </nav>
10   </header>
11
12   <section>
13     <h1>Meus Posts</h1>
14
15     ...
16   </section>
17
18   <footer>
19     <p>&copy,2013 K19 Treinamentos.</p>
20     <nav>
21       <a href="#">Home</a>
22       <a href="#">Últimos posts</a>
23       <a href="#">Arquivo</a>
24     </nav>
25   </footer>
26 </body>
```

Código HTML 2.149: Exemplo de nav

article

O elemento **article** é utilizado para definir uma composição independente em um documento HTML. A princípio, um **article** pode ser distribuído independentemente ou facilmente reutilizado. É apropriado utilizar esse elemento para definir um post de um fórum, um artigo de uma revista ou jornal, um post de um blog, um comentário enviado por um usuário, um widget interativo ou qualquer item independente de conteúdo.

Considere um blog no qual os usuários podem enviar comentários sobre os posts publicados. Os posts desse blog podem ser definidos com o elemento **article**. Os comentários podem ser definidos com elementos **article** internos aos que definem os posts. Um elemento **article** dentro de outro elemento **article** deve definir algo relacionado ao conteúdo do **article** que o contém.

```
1  <body>
2    <article>
3      <h1>Primeiro post do blog</h1>
4      <p>Loren ipsum...</p>
5
```

```
6     <h2>Comentários</h2>
7     <article>Legal este post!</article>
8     <article>Bacana este post!</article>
9     <article>Da hora este post!</article>
10  </article>
11 </body>
```

Código HTML 2.150: Exemplo de artigo

aside

O elemento **aside** é utilizado para definir algum conteúdo que esteja relacionado ao conteúdo principal, mas não é o foco do documento HTML. Normalmente, esse elemento é exibido em uma coluna lateral em relação ao conteúdo principal.

```
1  <body>
2    <article>
3      <h1>Novidades do HTML5</h1>
4      <p>Loren ipsum...</p>
5
6      <h2>Comentários</h2>
7      <article>Legal este post!</article>
8      <article>Bacana este post!</article>
9      <article>Da hora este post!</article>
10   </article>
11   <aside>
12     <h1>Posts relacionados</h1>
13     ...
14   </aside>
15 </body>
```

Código HTML 2.151: Exemplo de aside

section

O elemento **section** é utilizado para definir uma seção genérica de um documento. Uma seção é o agrupamento de um conteúdo dentro de um tema. Normalmente, as seções possuem cabeçalho e rodapé.

```
1  <body>
2    <section>
3      <h1>Últimos Posts</h1>
4      <article>
5        <h1>Primeiro post do blog</h1>
6        <p>Loren ipsum...</p>
7      </article>
8
9      <article>
10       <h1>Segundo post do blog</h1>
11       <p>Loren ipsum...</p>
12     </article>
13
14     <article>
15       <h1>Terceiro post do blog</h1>
16       <p>Loren ipsum...</p>
17     </article>
18   </section>
19 </body>
```

Código HTML 2.152: Exemplo de seção

figure

O elemento **figure** deve ser utilizado para definir um conteúdo que é auto-suficiente e tipicamente referenciado como uma unidade singular do fluxo principal do documento. Opcionalmente o conteúdo pode possuir uma legenda.

Por exemplo, esse elemento pode ser utilizado para definir ilustrações, diagramas, fotos, vídeos, códigos fonte em um documento HTML. Normalmente, esse itens são referenciados no conteúdo principal mas podem ser removidos sem afetar o fluxo do documento.

figcaption

O elemento **figcaption** deve ser filho de um elemento **figure**. Ele é utilizado para definir a legenda do conteúdo do elemento pai. Além disso, o elemento **figcaption** deve ser o primeiro ou último filho de um elemento **figure**.

```
<body>
  <h1>Novidades do HTML5</h1>

  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum
    enim est, ultrices ac vehicula vitae, pharetra id mauris. Quisque sit
    amet nisl mollis sem fermentum accumsan. Sed sed quam nisi, cursus
    sodales metus. Curabitur dapibus, massa sed sollicitudin viverra, odio
    justo dignissim metus, vel tempor turpis neque id erat. Aenean fermentum
    ultricies ante a luctus.
  </p>

  <figure>
    <img src="#" title="Aenean fermentum" alt="Mauris convallis, leo sollicitudin">
    <figcaption>Figura 3b. Curabitur dapibus, massa sed sollicitudin.</figcaption>
  </figure>

  <p>
    Mauris fermentum lorem nec nisi euismod elementum. Aenean nec magna
    dolor, vel fermentum turpis. Mauris convallis, leo sollicitudin
    egestas malesuada, nunc est ultrices enim, eget varius odio felis et
    velit. Sed ac lorem nibh, ut convallis ante.
  </p>
</body>
```

Código HTML 2.153: Exemplo de figure e figurecaption

header

Bologué

nav

Home Destaque Arquivo

section

Últimos Posts

article

Neque porro quisquam est qui dolorem

São Paulo, SP - 12/11/2013 - 03:45 PM

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec faucibus vulputate elit, sit amet venenatis ante lacinia eget. Ut id urna vel quam dictum elementum. Mauris rutrum magna nec erat ullamcorper scelerisque. Aenean non est libero. Mauris venenatis, velit quis mattis tempus, neque eros tristique purus, non pretium nisi urna at nunc. Morbi nec commodo dui, quis vulputate ligula. Maecenas vitae libero a turpis posuere fringilla. Pellentesque risus neque, pretium vitae elementum ac, ultrices et eros.

figure

Nullam dictum in nisl mollis ullamcorper.

figcaption

Sed sit amet enim sapien. Nulla quis lectus non orci pellentesque tristique. Duis eget ullamcorper turpis, facilisis commodo nisl. Aliquam rhoncus ut metus a vehicula. Suspendisse pellentesque ligula arcu, in viverra massa rhoncus a.

Comentários

article

Mauris Vulputate
13/11/2013 - 10:27 AM

Sed sit amet enim sapien. Nulla quis lectus non orci pellentesque tristique. Duis eget ullamcorper turpis, facilisis commodo nisl. Aliquam rhoncus ut metus a vehicula. Suspendisse pellentesque ligula arcu, in viverra massa rhoncus a.

article

Donec Tristique
13/11/2013 - 11:42 AM

Sed sit amet enim sapien. Nulla quis lectus non orci pellentesque tristique. Duis eget ullamcorper turpis, facilisis commodo nisl. Aliquam rhoncus ut metus a vehicula. Suspendisse pellentesque ligula arcu, in viverra massa rhoncus a.

aside

PLEASE, JOIN US FOR CHRISTMAS PARTY ON 25 12·2013 at 6 pm AT THE WELLINGTON HOTEL, NEW YORK

Links

- K19 Treinamentos
- Nulla tincidunt tincidunt
- Phasellus ullamcorper nibh

section

Posts em Destaque

article

Donec quis suscipit odio, k19 treinamentos

São Paulo, SP - 12/10/2013 - 06:19 PM

Sed sit amet enim sapien. Nulla quis lectus non orci pellentesque tristique. Duis eget ullamcorper turpis, facilisis commodo nisl. Aliquam rhoncus ut metus a vehicula. Suspendisse pellentesque ligula arcu, in viverra massa rhoncus a. Nulla at quam arcu. Quisque sed erat ac ligula auctor porttitor. Interdum et malesuada fames ac ante ipsum primis in faucibus. Vestibulum eleifend augue at ligula porta consectetur. Mauris a luctus leo. Curabitur auctor aliquet aliquam.

footer

Donec varius a nibh

- Sed sit amet enim sapien
- Nulla quis lectus non orci pellentesque tristique
- Duis eget ullamcorper turpis, facilisis commodo
- Suspendisse pellentesque ligula arcu, in viverra
- Nulla at quam arcu
- Quisque sed erat ac ligula auctor porttitor

Donec varius a nibh

- Maecenas non enim vel est volutpat
- Vivamus tristique tempor est, interdum
- Proin id pretium felis
- Nulla tincidunt tincidunt est, ac porta quam
- Sed ultricies justo vitae ullamcorper pulvinar
- Nullam dictum in nisl mollis ullamcorper

Like us on Bookface

Figura 2.88: Partes de um documento HTML

Normalmente, os elementos **header**, **footer**, **nav**, **header**, **article**, **aside**, **section**, **figure** e **figcaption** são definidos pelos navegadores como **block-level elements**.

Divisão de conteúdo

Muitas vezes, desejamos separar o conteúdo de um documento HTML. Geralmente, essas divisões ocorrem quando há uma mudança de tópico no conteúdo. Essas divisões podem ser realizadas com o elemento **hr**. Normalmente, o **hr** é definido pelos navegadores como **block-level element**.

```
<body>
  <h1>Novidades do HTML5</h1>

  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum
    enim est, ultrices ac vehicula vitae, pharetra id mauris. Quisque sit
    amet nisl mollis sem fermentum accumsan. Sed sed quam nisi, cursus
    sodales metus. Curabitur dapibus, massa sed sollicitudin viverra, odio
    justo dignissim metus, vel tempor turpis neque id erat. Aenean fermentum
    ultricies ante a luctus.
  </p>

  <hr>

  <p>
    Mauris fermentum lorem nec nisi euismod elementum. Aenean nec magna
    dolor, vel fermentum turpis. Mauris convallis, leo sollicitudin
    egestas malesuada, nunc est ultrices enim, eget varius odio felis et
    velit. Sed ac lorem nibh, ut convallis ante.
  </p>
</body>
```

Código HTML 2.154: Exemplo de separação de conteúdo

Exercícios de Fixação

90 Crie um arquivo chamado **hr.html** no projeto **html** em **Site Root** com o seguinte conteúdo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exercício de separação de conteúdo</title>
  </head>
  <body>
    <h1>Desenvolvimento Web</h1>

    <h2>HTML</h2>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum
      enim est, ultrices ac vehicula vitae, pharetra id mauris. Quisque sit
      amet nisl mollis sem fermentum accumsan. Sed sed quam nisi, cursus
      sodales metus. Curabitur dapibus, massa sed sollicitudin viverra, odio
      justo dignissim metus, vel tempor turpis neque id erat. Aenean fermentum
      ultricies ante a luctus.
    </p>

    <hr>

```

```
    <h2>CSS</h2>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum
      enim est, ultrices ac vehicula vitae, pharetra id mauris. Quisque sit
      amet nisl mollis sem fermentum accumsan. Sed sed quam nisi, cursus
      sodales metus. Curabitur dapibus, massa sed sollicitudin viverra, odio
      justo dignissim metus, vel tempor turpis neque id erat. Aenean fermentum
      ultricies ante a luctus.
    </p>
  </body>
</html>
```

Código HTML 2.155: hr.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-fixacao90.zip

91 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/hr.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/hr.html.

Agrupando elementos

Podemos agrupar elementos através dos elementos **div** ou **span**. Normalmente, os navegadores definem o elemento **div** como um **block-level element** e o **span** como um **inline-level element**. O principal objetivo em agrupar elementos com **div** ou **span** é aplicar formatações específicas para cada grupo. Por isso, a utilidade desses elementos ficará mais clara quando falarmos sobre CSS.

Erro: Fechamento inadequado das tags

Um erro comum, ao definir um documento HTML, é o fechamento inadequado das tags.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Fechamento inadequado das tags</title>
  </head>
  <body>
    <strong>Fechamento <em>inadequado</strong> das tags</em>
  </body>
</html>
```

Erro: Imagens sem texto alternativo

Um erro comum, ao definir um documento HTML, é não definir um texto alternativos para as imagens.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Imagens sem texto alternativo</title>
  </head>
  <body>
    <img src="imagem.png">
  </body>
</html>
```

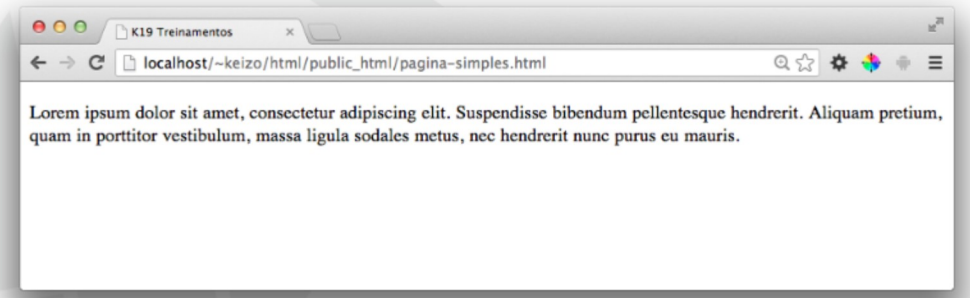
Erro: Criar listas com o elemento br

Um erro comum, ao definir um documento HTML, é criar listas com o elemento **br**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Criar listas com o elemento br</title>
  </head>
  <body>
    1. HTML<br>
    2. CSS<br>
    3. JavaScript
  </body>
</html>
```

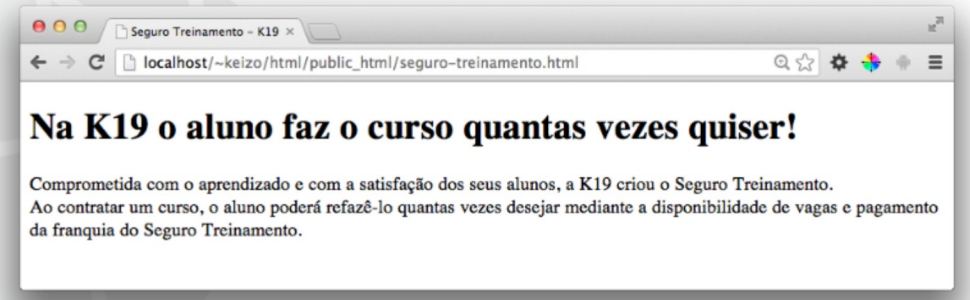
Exercícios Complementares

1 Crie um arquivo chamado **pagina-simples.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

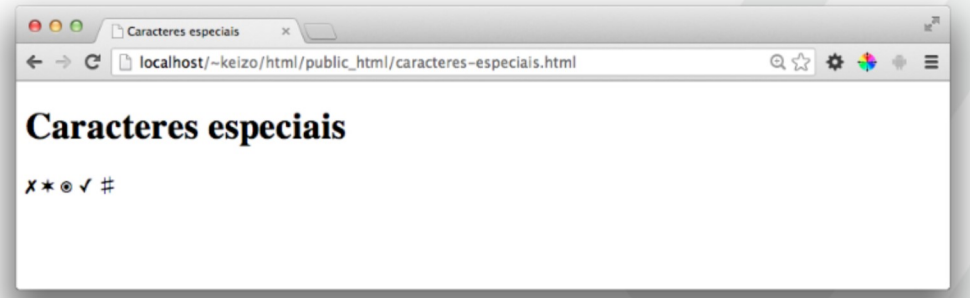


2 Crie um arquivo chamado **geografia.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

3 Crie um arquivo chamado **seguro-treinamento.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

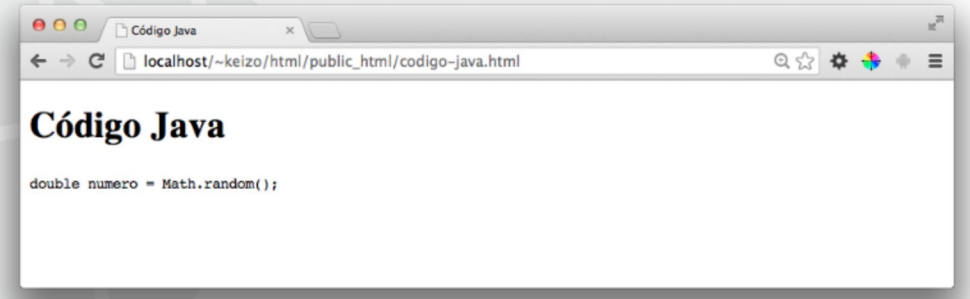


4 Crie um arquivo chamado **caracteres-especiais.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

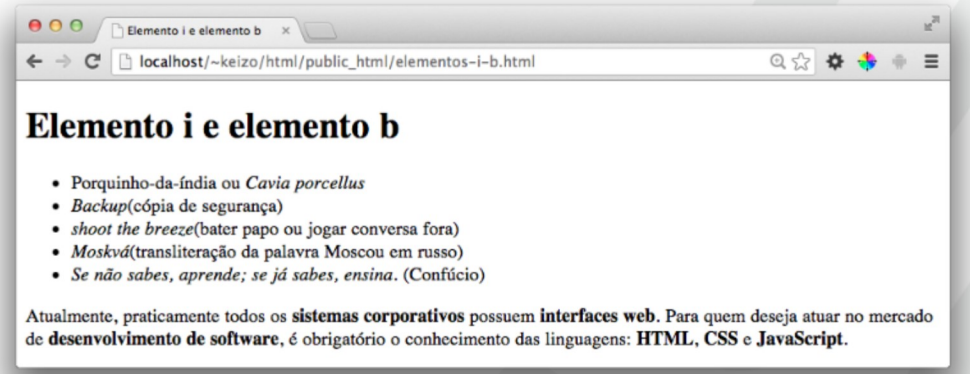


5 Crie um arquivo chamado **espacos-e-quebras-de-linha.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

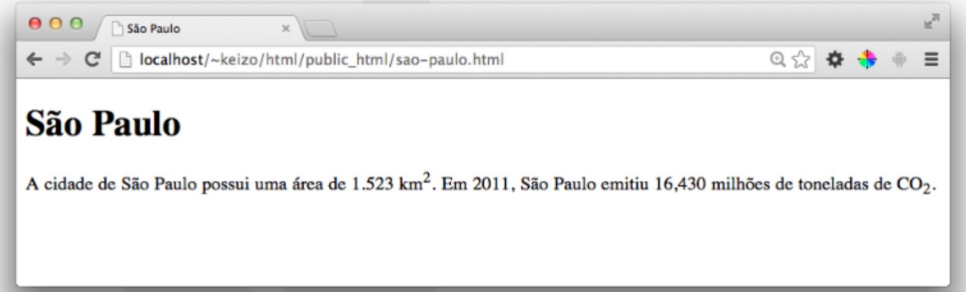
6 Crie um arquivo chamado **codigo-java.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



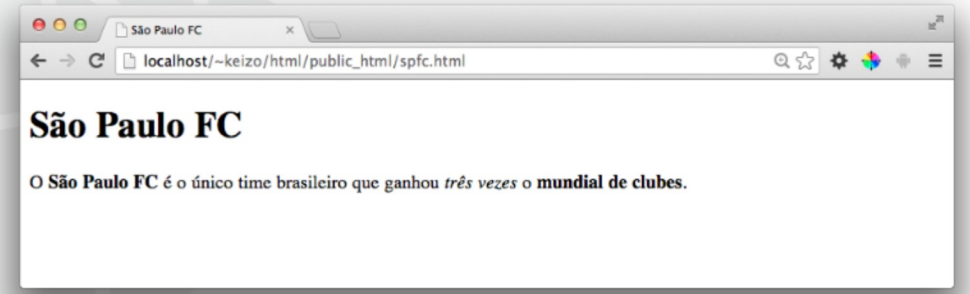
7 Crie um arquivo chamado **elementos-i-b.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



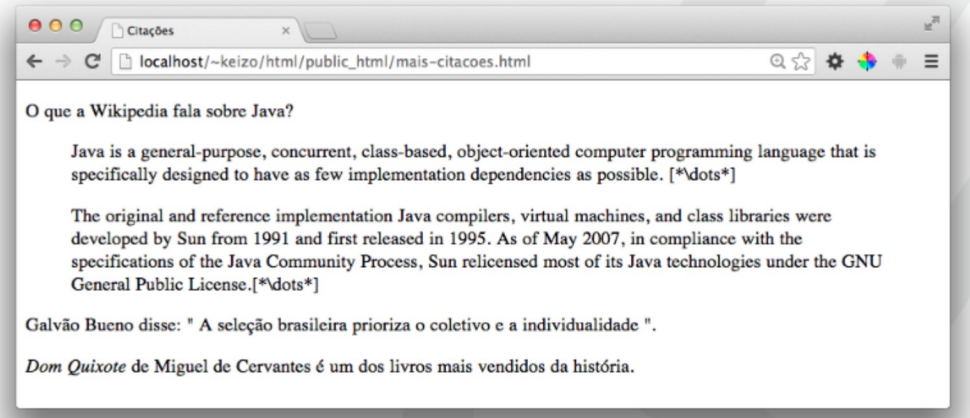
8 Crie um arquivo chamado **sao-paulo.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



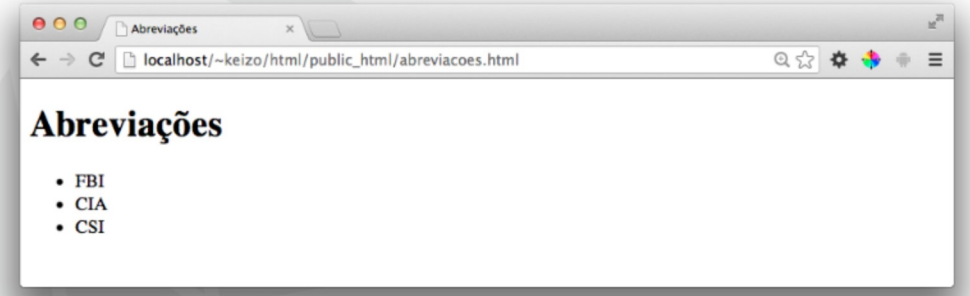
9 Crie um arquivo chamado **spfc.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



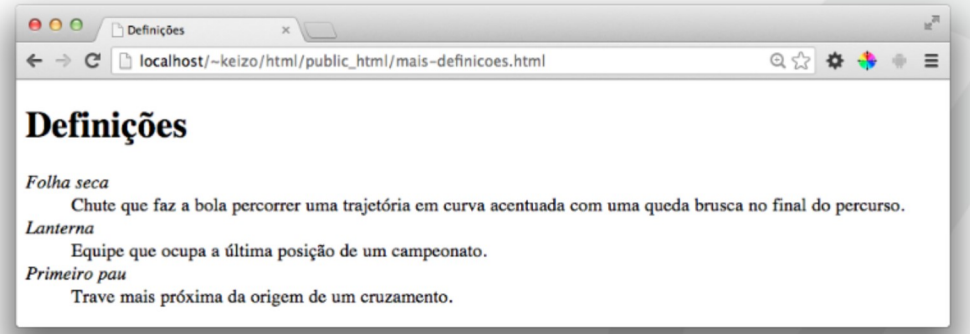
10 Crie um arquivo chamado **mais-citacoes.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



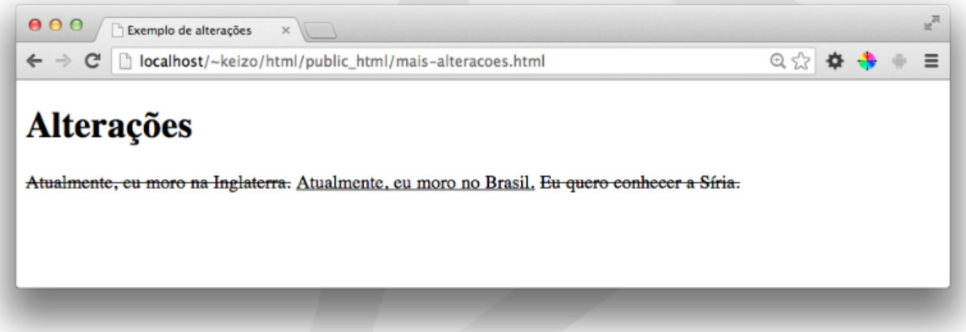
11 Crie um arquivo chamado **abbr.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



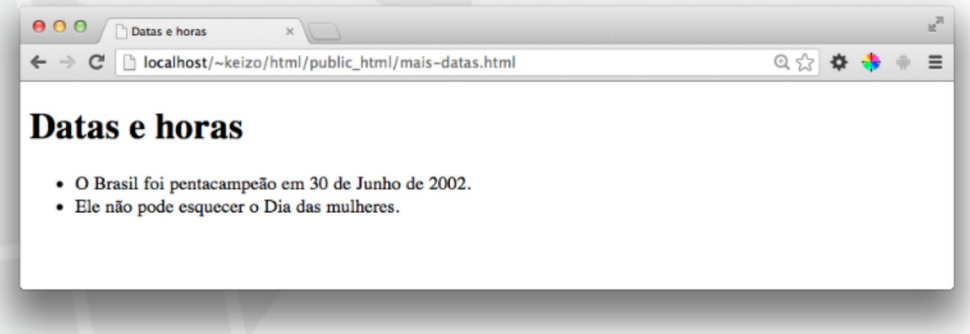
12 Crie um arquivo chamado **mais-definicoes.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



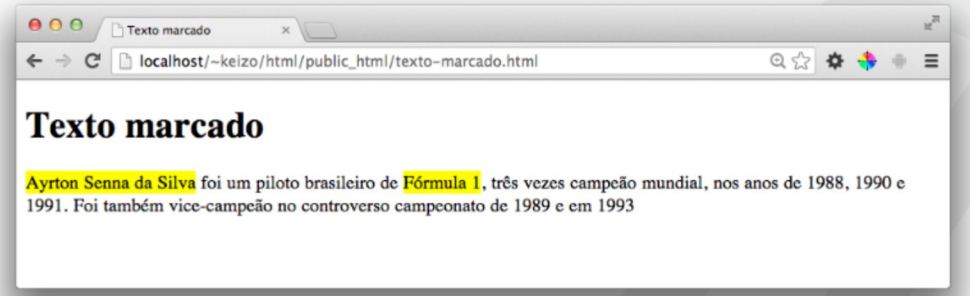
13 Crie um arquivo chamado **mais-alteracoes.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



14 Crie um arquivo chamado **mais-datas.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

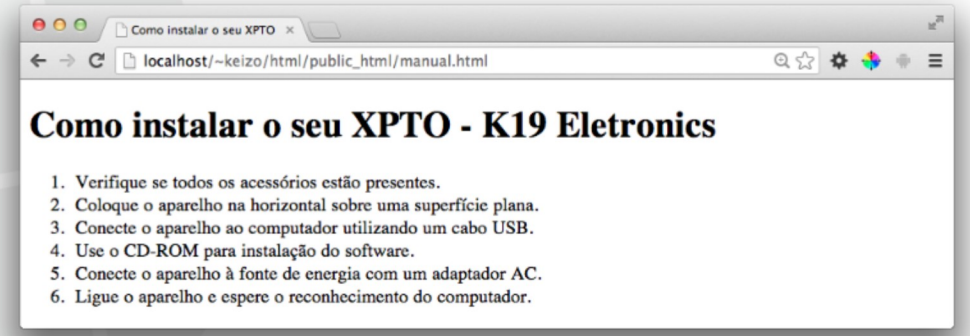


15 Crie um arquivo chamado **texto-marcado.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

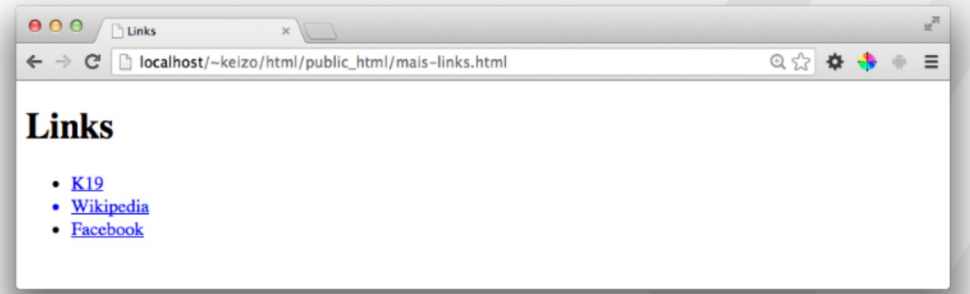


16 Crie um arquivo chamado **pontos-turisticos.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

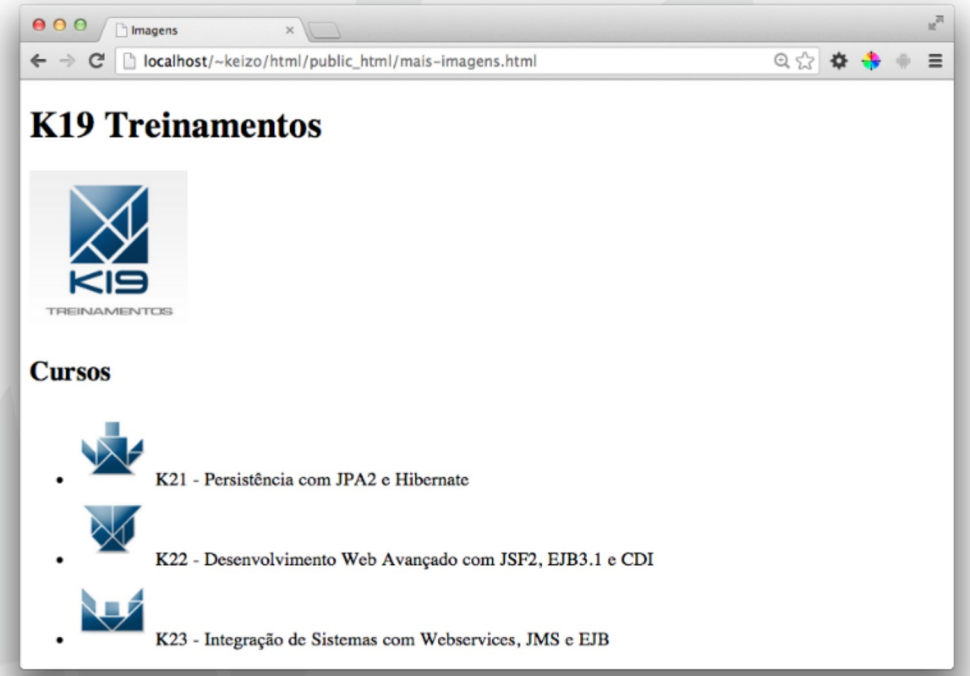
17 Crie um arquivo chamado **manual-k19.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



18 Crie um arquivo chamado **mais-links.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

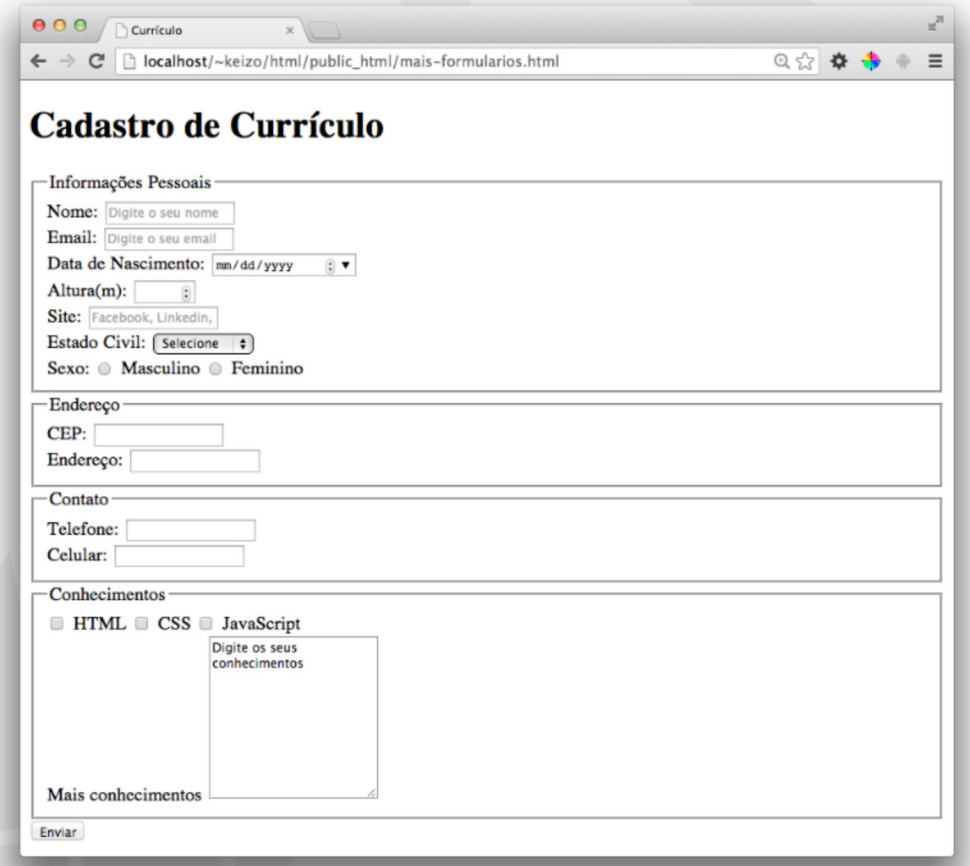


19 Crie um arquivo chamado **mais-imagens.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



20 Crie um arquivo chamado **mais-tabelas.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.

21 Crie um arquivo chamado **mais-formularios.html** no projeto **html** em **Site Root**. Utilize a imagem abaixo como base.



Resumo do Capítulo

1 A linguagem HTML é utilizada na criação das páginas web e é especificada pelo W3C.

2 Um documento HTML é composto por elementos hierarquicamente organizados.

3 Os elementos são inseridos em um documento HTML com o uso de tags.

4 O elemento **DOCTYPE** é utilizado para declarar o tipo do documento.

5 A raiz de um documento HTML é definida pelo elemento **html**.

6 Informações sobre um documento HTML devem ser definidas no corpo do elemento **head**.

7 No corpo do elemento **body**, é definido o conteúdo do documento HTML.

8 A semântica dos elementos HTML deve ser respeitada para a correta interpretação do conteúdo por buscadores e softwares de leitura (acessibilidade), por exemplo.

9 Os elementos **h1**, **h2**, **h3**, **h4**, **h5** e **h6** são utilizados para definir títulos em um documento HTML.

10 Parágrafos são inseridos em um documento HTML com o elemento **p**.

11 Determinados caracteres devem ser inseridos em um documento HTML através do seu respectivo código.

12 Os elementos **pre**, **code**, **i**, **b**, **sub**, **sup**, **strong**, **em**, **blockquote**, **q**, **cite**, **abbr**, **dfn**, **ins**, **del**, **s**, **time** e **mark** são utilizados para definir os textos de um documento HTML.

13 As listas sem ordem são definidas com o elemento **ul**. As listas com ordem são definidas com o elemento **ol**. As listas de descrições são definidas com o elemento **dl**.

14 Os itens das listas com ordem e sem ordem são definidos com o elemento **li**.

15 Os termos de uma lista de descrições são definidos com o elemento **dt**. As descrições desses termos são definidas com o elemento **dd**.

16 Podemos adicionar um documento HTML dentro de outro documento HTML com o elemento **iframe**.

17 Os links de um documento HTML são criados com o elemento **a**.

18 O elemento **img** é utilizado para adicionar imagens em um documento HTML.

19 As tabelas são criadas com os elementos **table**, **tr**, **th**, **td**, **thead**, **tbody**, **caption**, **colgroup** e **col**.

20 Para criar um formulário, devemos utilizar o elemento **form**.

21 Basicamente, os campos de um formulário são criados com os elementos **input**, **textarea** e **select**.

22 As partes de um documento HTML podem ser definidas com os elementos **header**, **footer**, **article**, **aside**, **section**, **figure**, **figcaption**, **div** e **span**.

23 As partes de um documento HTML podem ser separadas com o elemento **hr**.

Prova

1 Quem define a linguagem HTML?

a) Microsoft.

b) Anatel.

c) ABNT.

d) W3C.

e) NASA.

2 O que o elemento **DOCTYPE** determina?

a) O tipo do navegador.

b) O tipo do sistema operacional.

c) O tipo do documento.

d) O tipo do dispositivo.

e) Nenhuma das anteriores.

3 Qual alternativa está correta?

a) A raiz de um documento HTML é o elemento **body**.

b) O elemento **head** é colocado no corpo do elemento **body**.

c) O elemento **body** é colocado no corpo do elemento **head**.

d) A raiz de um documento HTML 5 é o elemento **html5**.

e) Os elementos **head** e **body** são colocados no corpo do elemento **html**.

4 Qual alternativa apresenta um comentário HTML válido?

a) <!– comentário –>

b) <* comentário *>

c) // comentário

d) /* comentário */

e) # comentário

5 Qual alternativa está incorreta?

a) Respeitar a semântica dos elementos HTML facilita o trabalho das ferramentas de busca.

b) Respeitar a semântica dos elementos HTML facilita o trabalho das ferramentas de leitura de tela para deficientes visuais.

c) Respeitar a semântica dos elementos HTML diminui o tamanho das páginas web.

d) A semântica dos elementos HTML é definida na especificação da linguagem HTML.

6 Qual alternativa está correta?

a) Títulos com **h2** são mais importantes do que títulos com **h1**.

b) Os títulos são definidos com os elementos **h1**, **h2**, **h3**, **h4**, **h5** e **h6**.

c) Os títulos são definidos com os elementos **h1**, **h2** e **h3**.

d) Os títulos são definidos com os elementos **header1**, **header2**, **header3**, **header4**, **header5** e **header6**.

e) O elemento **header1** equivale ao elemento **h1**.

7 Qual alternativa está correta?

a) Texto pré-formatado pode ser definido com o elemento **text-pre**.

b) O elemento **code** é utilizado para associar código JavaScript a um documento HTML.

c) O elemento **stronger** define textos importantes.

d) Citações longas são definidas com o elemento **blockquote**.

e) Podemos marcar texto com o elemento **text-mark**.

8 Qual alternativa está correta?

a) As listas com ordem são definidas através do elemento **dl**.

b) Os termos de uma lista de descrições são definidos com o elemento **li**.

c) As listas com ordem e sem ordem são definidas através dos elementos **ul** e **ol** respectivamente.

d) Os ítens de uma lista com ordem são definidos através do elemento **item**.

e) Uma lista pode ser definida dentro de outra lista.

9 Qual alternativa está incorreta?

a) Os links são definidos com o elemento **a**.

b) O destino de um link é determinado pelo atributo **href**.

c) O destino de um link com **target="_blank"** é aberto em uma nova aba ou janela.

d) O destino de um link com **target="_self"** é aberto na mesma aba ou janela.

e) O atributo **to** define o destino dos links.

10 Para exibir um texto quando o mouse passa sobre uma imagem, devemos utilizar qual atributo do elemento **img**?

a) alt.

b) title.

c) tooltip.

d) text.

e) description.

11 Qual alternativa contém apenas elementos relacionados às tabelas?

a) table, td e column.

b) table, row e col.

c) table, table-head e table-body.

d) th, tr e td.

e) tabular, tr e td.

12 Qual alternativa está correta?

a) As caixas de texto são adicionadas com o elemento **input-text**.

b) Os rótulos são definidos com o elemento **input-label**.

c) Podemos criar um botão de submit com o elemento **input**.

d) As caixas de senha são definidas com o elemento **password**.

e) Os checkboxes e os radios são criados com os elementos **checkbox** e **radio** respectivamente.

13 Qual alternativa está correta?

a) As opções de um drop-down list são definidas com o elemento **input**.

b) As opções de um drop-down list podem ser agrupadas com o elemento **option-group**.

c) O atributo **selected** do elemento **alternative** define as opções selecionadas.

d) Podemos criar um drop-down lista com o elemento **drop-down**.

e) O elemento **select** é utilizado para criar drop-down lists.

14 Qual alternativa está correta?

a) Os campos de um formulário podem ser agrupados com o elemento **inputset**.

b) Os campos de um formulário podem ser agrupados com o elemento **fieldset**.

c) O elemento **label** deve ser utilizado em conjunto com o elemento **fieldset**.

d) O elemento **legend** deve ser utilizado em conjunto com o elemento **inputset**.

e) A legenda de um **fieldset** é definida através do atributo **legend**.

15 Qual alternativa está correta?

a) As sugestões para o autocomplete de um campo são definidas pelos elementos **datalist** e **option**.

b) As sugestões para o autocomplete de um campo são definidas através do atributo **autocomplete**.

c) O elemento **disabled** é utilizado para desabilitar campos ou botões.

d) O atributo **off** é utilizado para desabilitar campos ou botões.

e) Campos fixos podem ser definidos com o atributo **no-write**.

Minha Pontuação [ ] Pontuação Mínima: [12] Pontuação Máxima: [15]

CSS

CAPÍTULO 3

Quando não definimos uma formatação específica para os elementos HTML de uma página web, eles são exibidos com o estilo determinado pelo navegador utilizado. Há dois problemas principais em deixar os navegadores decidirem qual formatação deve ser adotada. O primeiro é que cada navegador pode adotar uma formatação diferente. Consequentemente, uma mesma página web não será exibida exatamente da mesma forma em todos os navegadores. O segundo problema é que as formatações adotadas pelos principais navegadores não são "bonitas".

Podemos padronizar a forma que as páginas web são exibidas nos diferentes navegadores e obter um visual agradável definindo a nossa própria formatação. O estilo das páginas deve ser definido com a linguagem **CSS (Cascading Style Sheets)**.

Mais Sobre
Assim como a linguagem HTML, a linguagem CSS é definida pelo W3C. Você pode consultar a especificação do CSS no endereço http.//www.w3.org/Style/CSS/specs. O W3C faz diversas recomendações sobre como implementar as funcionalidades do CSS. Contudo, nem sempre, os navegadores respeitam essas recomendações e eventualmente cada navegador adota uma abordagem própria para uma determinada funcionalidade. Essas diferenças podem gerar problemas nas páginas web.

Mais Sobre
Nas primeiras versões da linguagem HTML, a formatação era realizada através de atributos específicos dos elementos. Por exemplo, a cor do fundo de uma página poderia ser definida da seguinte forma:

```
1 <body bgcolor="red">
```

Hoje em dia, essa abordagem é totalmente não recomendada. Qualquer alteração visual dos elementos HTML deve ser definida através da linguagem CSS.

Box Model

Todo elemento HTML, no corpo de uma página web, está contido em um **box**. Considere o documento HTML abaixo.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
```

```
5       <title>Desenvolvimento Web</title>
6     </head>
7     <body>
8       <h1>Desenvolvimento Web</h1>
9
10      <h2>HTML</h2>
11      <p>
12        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum
13        enim est, ultrices ac vehicula vitae, pharetra id mauris. Quisque sit
14        amet nisl mollis sem fermentum <strong>accumsan</strong>.
15      </p>
16      <a href="#">Mais sobre</a>
17
18      <h2>CSS</h2>
19      <p>
20        Lorem ipsum dolor sit amet, <em>consectetur</em> adipiscing elit. Vestibulum
21        enim est, ultrices ac vehicula vitae, pharetra id mauris. Quisque sit
22        amet nisl mollis sem fermentum accumsan.
23      </p>
24      <a href="#">Mais sobre</a>
25    </body>
26  </html>
```

Agora, observe em destaque na imagem a seguir, os boxes correspondentes aos elementos HTML do documento anterior.

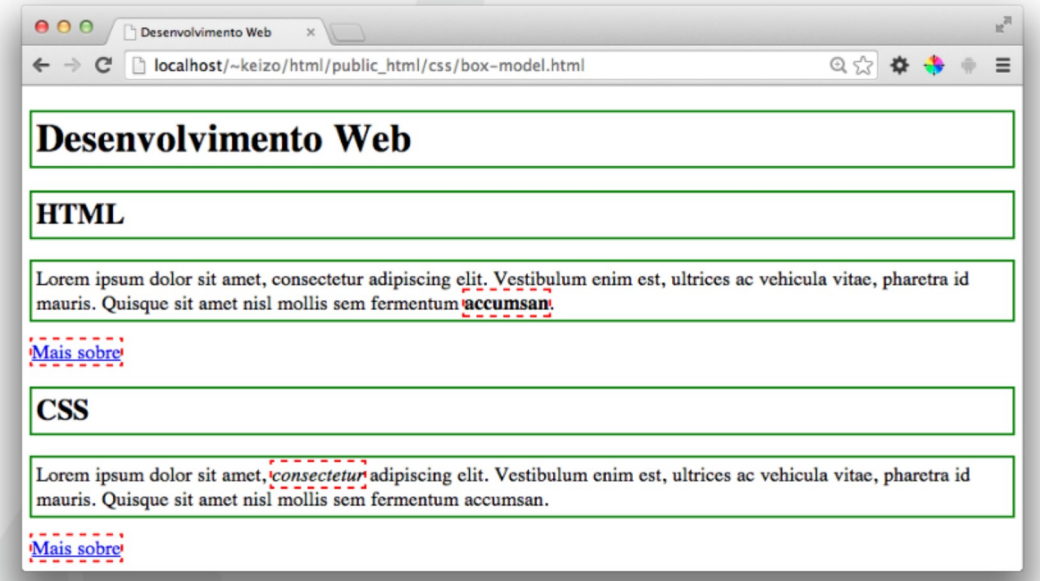


Figura 3.1: Boxes

Na imagem, os boxes dos **block-level elements** foram exibidos com linha contínua enquanto os boxes dos **inline-level elements** foram exibidos com linha tracejada. Os boxes dos **block-level elements** ocupam todo o espaço horizontal e provocam quebras de linha. Por outro lado, os boxes dos **inline-level elements** ocupam horizontalmente somente o espaço necessário para o seu conteúdo e não provocam quebras de linha.

O box de um elemento HTML é composto por conteúdo (content), margem interna (padding),

borda (border) e margem externa (margin).

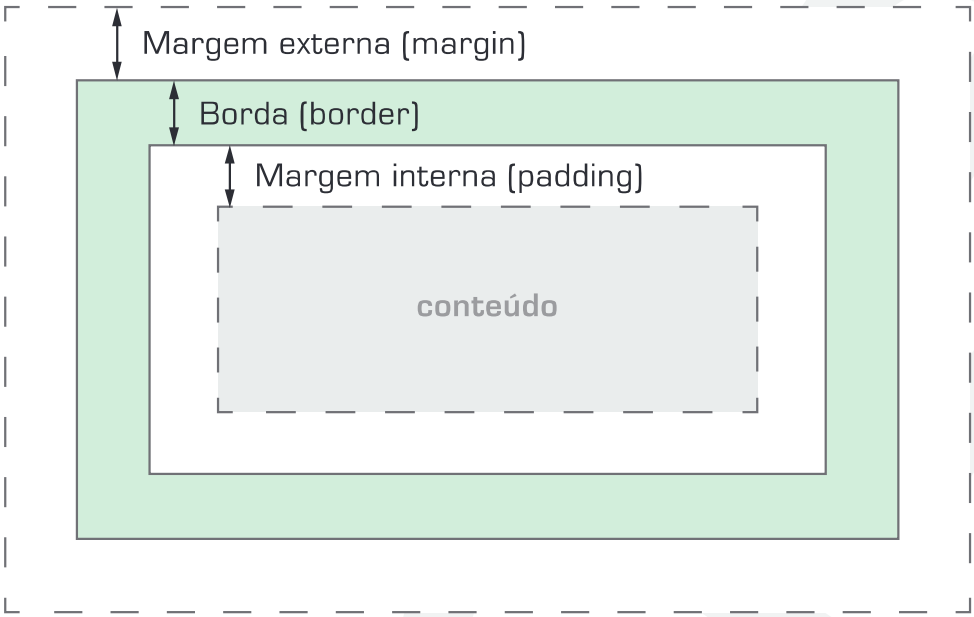


Figura 3.2: Box model

Observe que a largura de um box corresponde à soma dos seguintes comprimentos.

- Margem externa da esquerda (margin-left)
- Borda da esquerda (border-left)
- Margem interna da esquerda (padding-left)
- Largura do conteúdo (width)
- Margem interna da direita (padding-right)
- Borda da direita (border-right)
- Margem externa da direita (margin-right)

Analogamente, a altura de um box corresponde à soma dos seguintes comprimentos.

- Margem externa superior (margin-top)
- Borda superior (border-top)
- Margem interna superior (padding-top)
- Altura do conteúdo (height)
- Margem interna inferior (padding-bottom)
- Borda inferior (border-bottom)
- Margem externa inferior (margin-bottom)

Regras

Podemos aplicar regras de formatação para cada elemento HTML. Observe a estrutura de uma regra CSS.

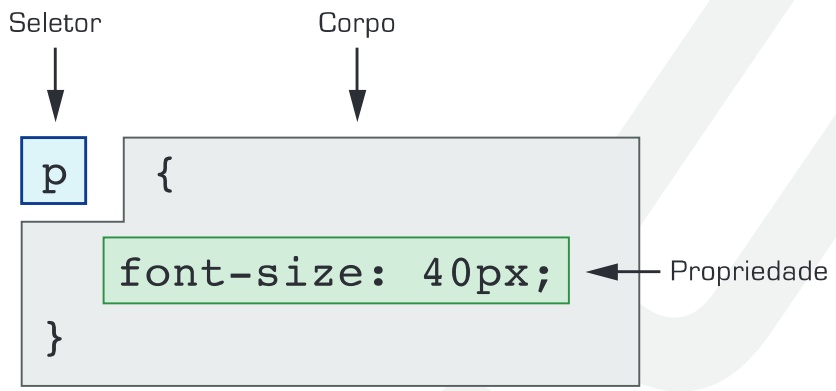


Figura 3.3: Estrutura de uma regra CSS

O seletor determina para quais elementos a regra de formatação deve ser aplicada. No corpo de uma regra CSS, podemos definir diversas propriedades. Cada propriedade define uma característica visual. Por exemplo, a propriedade **font-size** é utilizada para definir o tamanho da fonte utilizada nos textos.

Aplicando CSS ao HTML

Podemos aplicar propriedades CSS a elementos HTML de três maneiras diferentes.

Inline style

Podemos aplicar propriedades CSS através do atributo **style** presente em diversos elementos HTML. Essa abordagem não é recomendada pois o acoplamento entre os elementos HTML e as propriedades CSS é muito alto dificultando a manutenção das páginas web.

```
<p style="color. red">
  ...
</p>
```

Internal style sheet

As regras CSS podem ser definidas no conteúdo do elemento **style**. Esse elemento deve ser definido dentro do **head**.

```
<head>
  <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
  <title>Desenvolvimento Web</title>
  <style>
    p {
      color. red,
    }
  </style>
</head>
```

External style sheet

Podemos separar o código HTML do código CSS definido as regras de formatação em um arquivo independente.

```
p {
  color. red,
}
```

Código CSS 3.1: estilo.css

Depois de definir as regras CSS em um arquivo separado, ele deve ser associado ao documento HTML através do elemento **link**.

```
<head>
  <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
  <title>Desenvolvimento Web</title>
  <link rel="stylesheet" type="text/css" href="estilo.css">
</head>
```

Comentários

Podemos adicionar comentários no código CSS através dos marcadores **/*** e ***/**.

```
/* uma regra CSS para formatar os parágrafos */
p {
  color. red,
}
```

Código CSS 3.2: estilo.css

Chrome DevTools

Como vimos no capítulo anterior, podemos utilizar ferramentas como o Chrome DevTools para analisar e modificar os documentos HTML. Essas ferramentas, também podem ser utilizadas para editar as propriedades CSS dos elementos de uma página web. Considere o código HTML e o CSS abaixo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>K19 - Chrome DevTools</title>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <link rel="stylesheet" type="text/css" href="chrome-devtools.css">
  </head>
  <body>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <p class="class1">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
  </body>
</html>
```

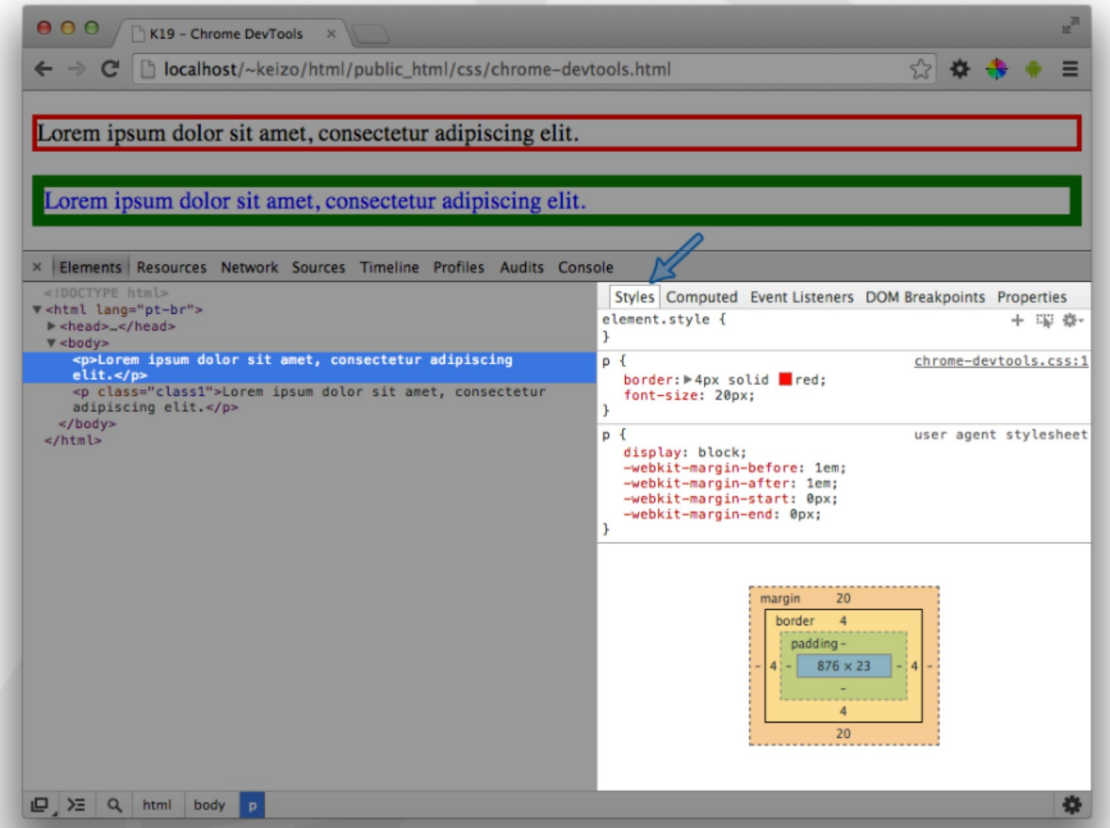
Código HTML 3.6: chrome-devtools.html

```
p {
    border. 4px solid red,
```

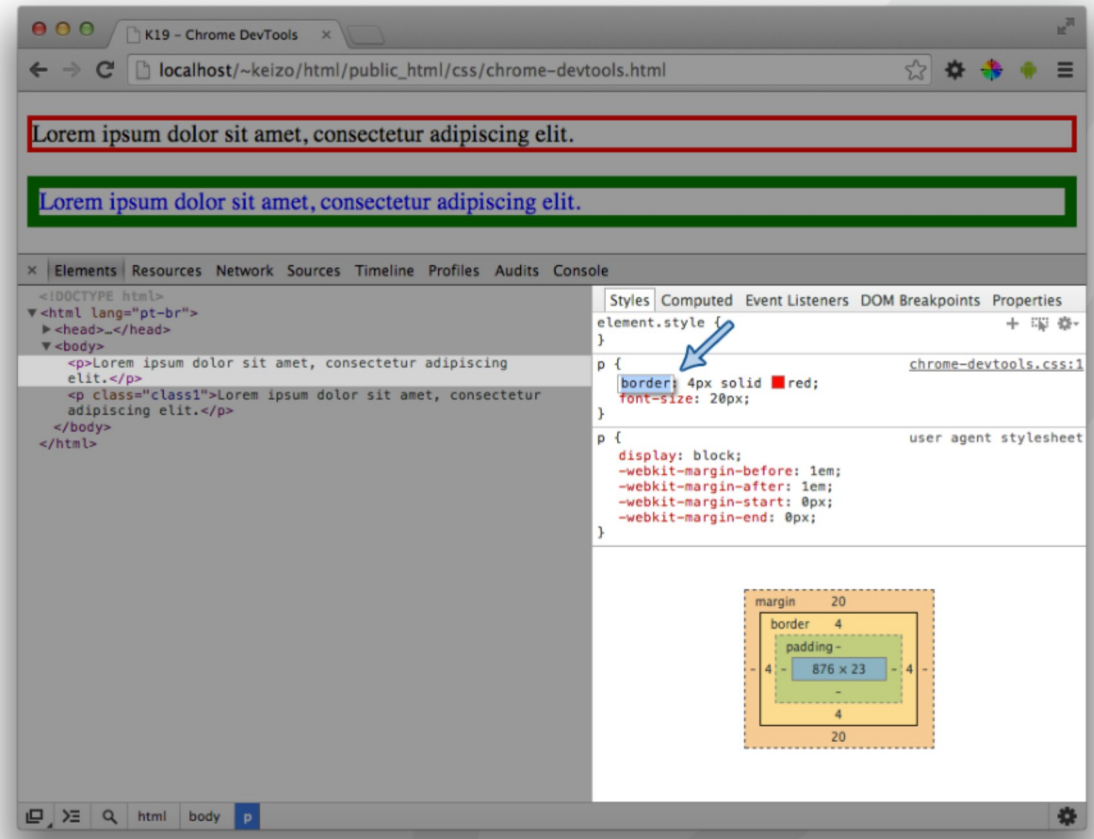
```
3     font-size. 20px,
4 }
5
6 .class1 {
7     color. blue,
8     border. 10px solid green,
9 }
```

Código CSS 3.3: chrome-devtools.css

Observe a figura abaixo. Através do painel **Styles** podemos visualizar as regras CSS que foram aplicadas a um determinado elemento HTML. Esse painel também mostra a origem dessas regras e as dimensões do box do elemento.



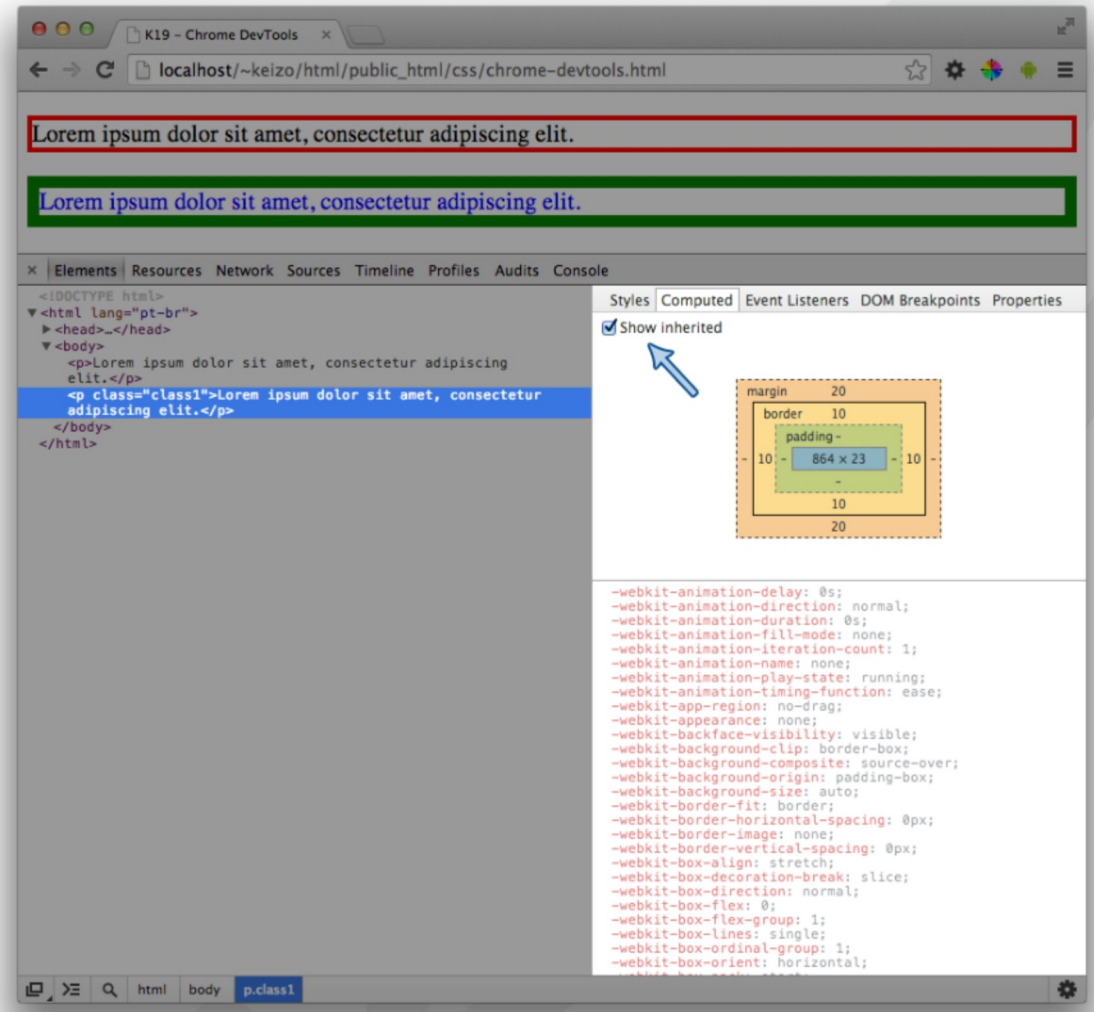
Podemos incluir, remover, desabilitar ou alterar as propriedades de um elemento clicando sobre elas no painel **Styles**.



Observe na imagem abaixo que o elemento HTML selecionado no painel **Styles** é afetado por duas regras CSS oriundas do arquivo **chrome-devtools.css**. Repare que a propriedade **border** é definida nessas duas regras. O Chrome DevTools indica através de um traço horizontal qual ocorrência dessa propriedade está sendo desconsiderada. Mais adiante, discutiremos como determinar a prioridade das regras CSS em situações como essa.

No painel **Computed** podemos conferir todas as propriedades CSS que foram aplicadas em um elemento.

Se desejarmos visualizar as propriedades que foram herdadas dos elementos ancestrais, podemos marcar a opção **Show inherited**.



Exercícios de Fixação

1. Abra o **Netbeans** e crie um projeto chamado **css**.

Importante
No **Windows**, utilizando o IIS (Internet Information Services) como Web Server, você deve salvar o projeto **css** em **C:\inetpub\wwwroot**. Lembre-se que é necessário instalar o IIS conforme vimos anteriormente.

Importante
No **Ubuntu**, utilizando o Apache HTTP Server como Web Server, você deve salvar o projeto **css** em **/home/<USUARIO>/public_html**. Lembre-se que é necessário instalar e configurar o Apache HTTP Server como vimos anteriormente.

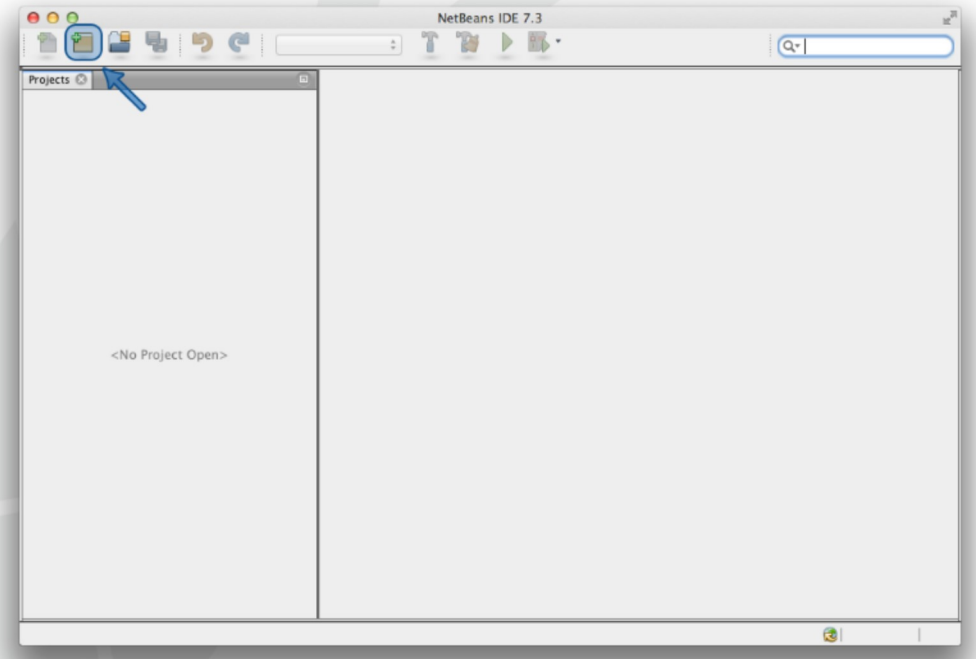


*Figura 3.4: Projeto **css***

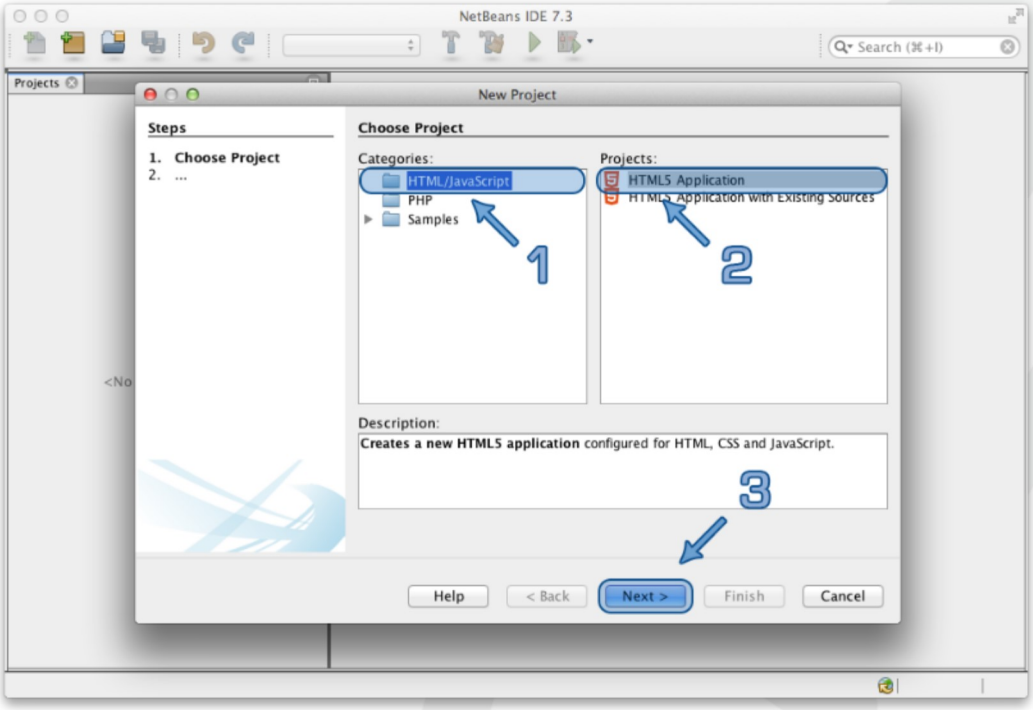


*Figura 3.5: Projeto **css***

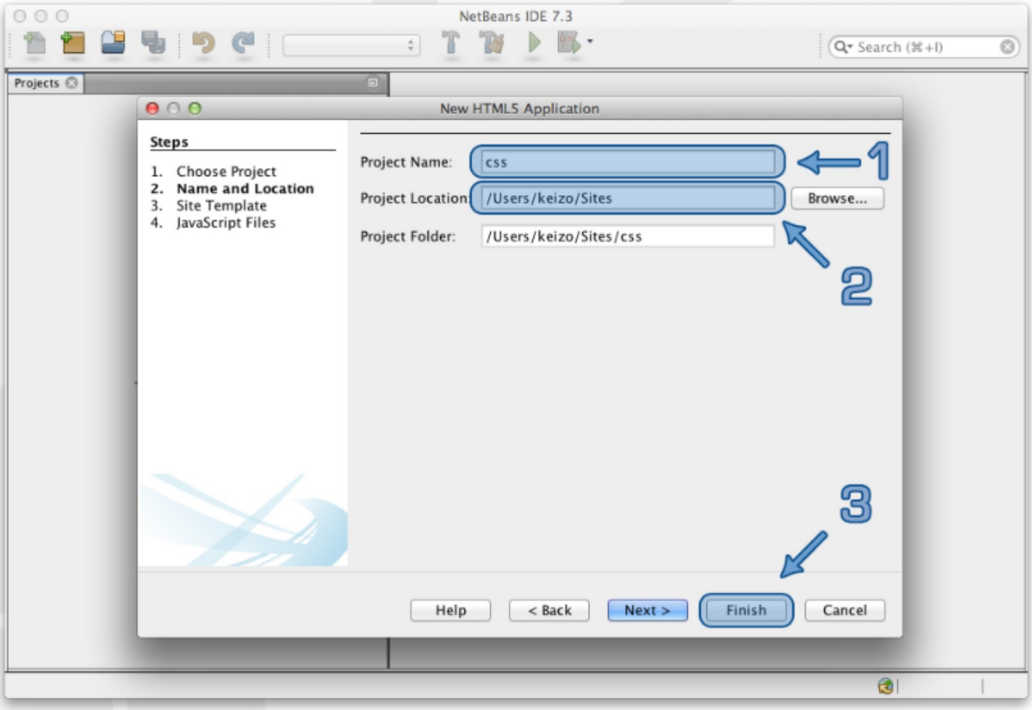


*Figura 3.6: Projeto **css***

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao1.zip

2 No projeto **css**, crie um arquivo chamado **css.html**. Todos os parágrafos desse documento devem ser exibidos em negrito, com cor azul e com fonte 20px. Defina uma regra CSS para formatar os parágrafos.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>CSS</title>
    <style type="text/css">
      p {
        font-weight. bold,
        font-size. 20px,
        color. #0000FF,
      }
    </style>
  </head>
  <body>
    <p>Um parágrafo formatado</p>
    <p>Outro parágrafo formatado</p>
  </body>
</html>
```

Código HTML 3.7: css.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao2.zip

3 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/css.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/css.html.

4 Para organizar melhor o conteúdo e a formatação da página criada anteriormente, crie um arquivo CSS chamado **style.css** no projeto **css**.

```
p {
  font-weight. bold,
  font-size. 20px,
  color. #0000FF,
}
```

Código CSS 3.4: style.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao4.zip

5 Modifique o arquivo **css.html** para aplicar as regras de formatação criadas no exercício anterior.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>CSS</title>
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <p>Um parágrafo formatado</p>
```

```
10      <p>Outro parágrafo formatado</p>
11    </body>
12  </html>
```

Código HTML 3.8: css.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao5.zip

Cores

Há diversas formas de escolher uma cor na linguagem CSS.

Nome

A especificação da linguagem CSS nomeou 147 cores. Você pode verificar a lista desses nomes no endereço http.//www.w3.org/TR/css3-color/#svg-color. As cores básicas do CSS são: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white e yellow.

Hexadecimal

A quantidade de cores suportada pela linguagem CSS é 16.777.216. Como vimos apenas 147 cores foram nomeadas. Para escolher uma cor que não possui um nome, podemos utilizar o código hexadecimal da cor desejada. Todas as cores possuem um código hexadecimal. Veja o código hexadecimal das cores básicas.

código hexadecimal	nome da cor	cor
#00FFFF	aqua	♠
#000000	black	♠
#0000FF	blue	♠
#FF00FF	fuchsia	♠
#808080	gray	♠
#008000	green	♠
#00FF00	lime	♠
#800000	marron	♠

código hexadecimal	nome da cor	cor
#000080	navy	♠
#808000	olive	♠
#800080	purple	♠
#FF0000	red	♠
#C0C0C0	silver	♠
#008080	teal	♠
#FFFFFF	white	
#FFFF00	yellow	♠

O código hexadecimal de uma cor começa com o caractere **#**. Os dois primeiros dígitos definem a quantidade de vermelho, os dois dígitos do meio definem a quantidade de verde e os dois últimos dígitos definem a quantidade de azul.

Função rgb()

A função **rgb()** devolve uma cor a partir de uma quantidade de vermelho, de verde e de azul. Essas quantidades podem ser definidas em porcentagem ou em valores inteiros de 0 a 255. Veja como podemos obter as cores básicas a partir da função **rgb()**.

rgb()	nome da cor	cor
rgb(0, 255, 255)	aqua	♠
rgb(0, 0, 0)	black	♠
rgb(0, 0, 255)	blue	♠
rgb(255, 0, 255)	fuchsia	♠
rgb(128, 128, 128)	gray	♠
rgb(0, 128, 0)	green	♠
rgb(0, 255, 0)	lime	♠
rgb(128, 0, 0)	marron	♠

rgb()	nome da cor	cor
rgb(0, 0, 128)	navy	♠
rgb(128, 128, 0)	olive	♠
rgb(128, 0, 128)	purple	♠
rgb(255, 0, 0)	red	♠
rgb(192, 192, 192)	silver	♠
rgb(0, 128, 128)	teal	♠
rgb(255, 255, 255)	white	
rgb(255, 255, 0)	yellow	♠

Função rgba()

A função **rgba()** é muito semelhante à função **rgb()**. A diferença é que a função **rgba()** possui um quarto parâmetro que define a opacidade da cor. O valor desse parâmetro deve ser maior ou igual a 0 e menor ou igual a 1. Por exemplo, para definir a cor **blue** com 50% de opacidade, podemos utilizar a função **rgba()** com os seguintes parâmetros.

```
1 rgba(0, 0, 255, 0.5)
```

Função hsl()

Podemos usar a escala **HSL** (Hue, Saturation and Lightness ou Matiz, Saturação e Brilho) para selecionar uma cor através da função **hsl()**. Essa função recebe 3 parâmetros. O primeiro deve ser um valor inteiro de 0 até 360. O segundo e o terceiro são porcetagens.

```
1 hsl(180, 50%, 60%)
```

Função hsla()

Assim como a função **hsl()**, a função **hsla()** também permite a seleção das cores através da escala **HSL**. A diferença é que a última permite que a opacidade da cor seja definida.

```
1 hsla(180, 50%, 60%, 0.5)
```

Unidades de medida

Para definir um tamanho ou uma distância, devemos utilizar as unidades da linguagem CSS. Podemos classificar essas unidades em absolutas e relativas.

Unidades absolutas

Unidade	Descrição
cm	centímetro
mm	milímetro
in	polegada
pt	ponto (1 pt é igual a 1/72 in)
pc	pica (1 pc é igual a 12 pt)
px	pixel

Apesar de ser considerada absoluta, a unidade **px** depende da resolução e do tamanho da tela. Por exemplo, considere uma tela de 15 polegadas com resolução 1920x1080 e outra de 14 polegadas com resolução 1024x768. Na primeira tela, 1 px é igual a 0,172 mm. Na segunda tela, 1 px equivale a 0,277 mm.

Unidades relativas

Unidade	Descrição
em	1 em é igual ao tamanho da fonte do elemento onde essa unidade for utilizada.
ex	Geralmente, 1 ex é igual a altura da letra "x" da fonte atual.
ch	Geralmente, 1 ch é igual a largura do dígito "0" da fonte atual.
rem	1 rem é igual ao tamanho da fonte do elemento raiz do documento HTML.
vw	1 vw é igual a 1% da largura da área de visualização do navegador (viewport).
vh	1 vh é igual a 1% da altura da área de visualização do navegador (viewport).
vmin	1 vmim é igual ao valor mínimo entre vw e vh.
vmax	1 vmim é igual ao valor máximo entre vw e vh.

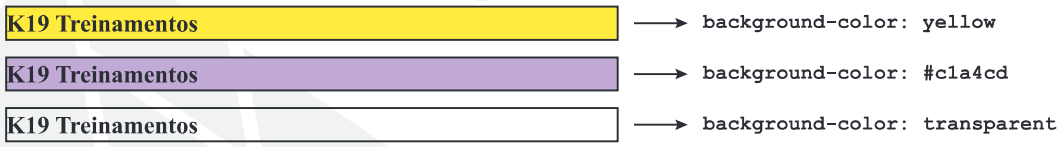
Backgrounds

Podemos definir a formatação do background de um elemento HTML com as seguintes propriedades CSS:

background-color

A propriedade **background-color** é utilizada para definir a cor do background dos elementos HTML.

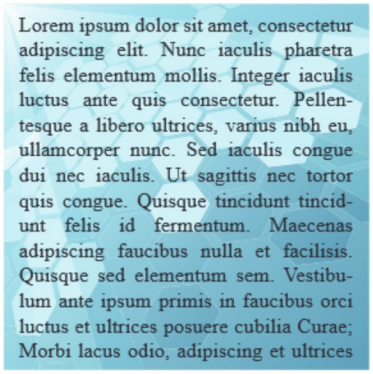
Valor	Descrição
color	Uma cor CSS.
transparent	Faz o background ser transparente (padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



background-image

A propriedade **background-image** é utilizada para definir a imagem que deve ser utilizada como background de um elemento HTML.

Valor	Descrição
url('*url*')	URL da imagem de background.
none	Sem imagem de background (padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



→ `background-image: url('bg-image.png')`

Mais Sobre

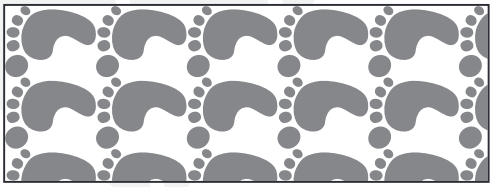
Podemos definir múltiplas imagens de background. As URLs dessas imagens devem ser separadas por vírgula. As imagens são sobrepostas sendo que a primeira ficará no primeiro plano, a segunda no segundo plano e assim por diante até a última que ficará no último plano.

```
1 background-image. url("imagem1.png"),url("imagem2.png"),url("imagem3.png"),
```

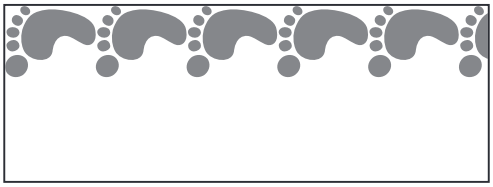
background-repeat

Por padrão, se uma imagem utilizada como background for menor do que o box do elemento HTML correspondente, ela será replicada na horizontal e na vertical. A propriedade **background-repeat** permite controlar como essa replicação deve ser realizada. Essa propriedade aceita os seguintes valores:

Valor	Descrição
repeat	A imagem de background é replicada na horizontal e na vertical (padrão).
repeat-x	A imagem de background é replicada apenas na horizontal.
repeat-y	A imagem de background é replicada apenas na vertical.
no-repeat	A imagem não é replicada.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



`background-repeat: repeat`



`background-repeat: repeat-x`

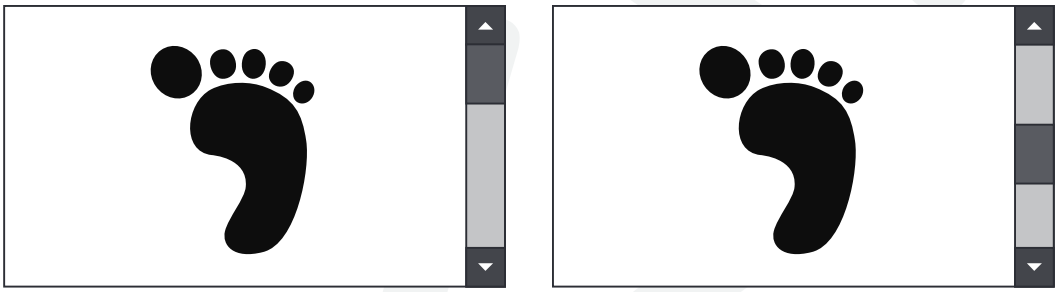
background-repeat: repeat-y

background-repeat: no-repeat

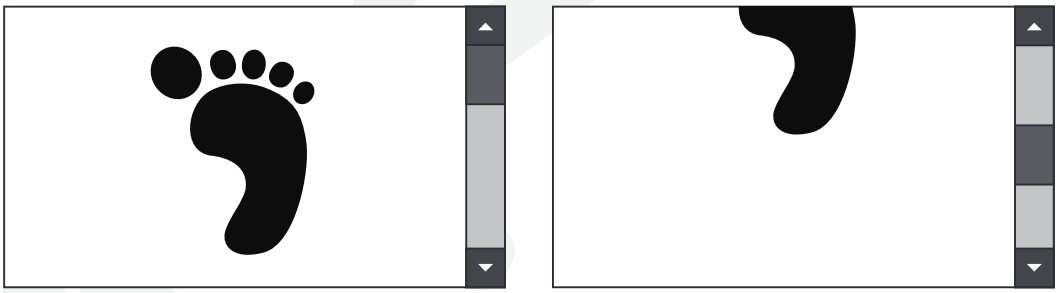
background-attachment

A propriedade **background-attachment** permite definir qual é comportamento dos backgrounds em relação a rolagem do conteúdo.

Valor	Descrição
scroll	O background acompanha a rolagem do conteúdo (padrão).
fixed	O background não acompanha a rolagem do conteúdo.



background-attachment: fixed



background-attachment: scroll

background-position

Para derterminar a posição das imagens utilizadas como background dos elementos HTML, devemos utilizar a propriedade **background-position**.

Valor	Descrição
left top	Canto superior esquerdo (padrão).
center top	Centralizado horizontalmente no topo.
right top	Canto superior direito.
left center	Do lado esquerdo centralizado verticalmente.
center center	Centralizado horizontalmente e verticalmente.

Valor	Descrição
right center	Do lado direito centralizado verticalmente.
left bottom	Canto inferior esquerdo.
center bottom	Centralizado horizontalmente na base.
right bottom	Canto inferior direito.
left	O mesmo que "left center".
center	O mesmo que "center center".
right	O mesmo que "right center".
top	O mesmo que "center top".
bottom	O mesmo que "center bottom".
x% *y%*	O valor *x* corresponde à posição horizontal e o *y* à posição vertical. Eles são relativos ao canto superior esquerdo.
x%	O mesmo que "*x*% 50%".
medida1 *medida2*	A *medida1* corresponde à posição horizontal e a *medida2* à posição vertical. Elas são relativas ao canto superior esquerdo.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

left top

center top

right top

left center

center center

right center

left bottom

center bottom

right bottom

background-clip

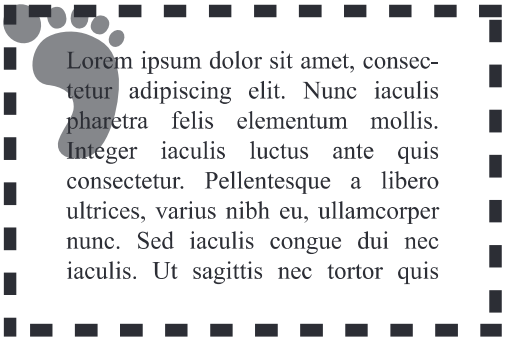
A propriedade **background-clip** determina qual área do box de um elemento HTML deve ser afetada pela propriedade **background-color**.

Valor	Descrição
border-box	Área delimitada pelas bordas (padrão).
padding-box	Área delimitada pelas margens internas.
content-box	Área delimitada pelo conteúdo.

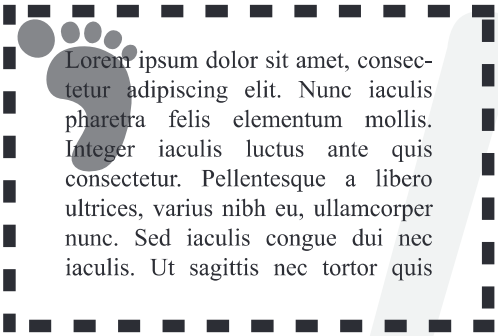
background-origin

A propriedade **background-origin** determina onde ficará o ponto de origem para o posicionamento das imagens utilizadas como background.

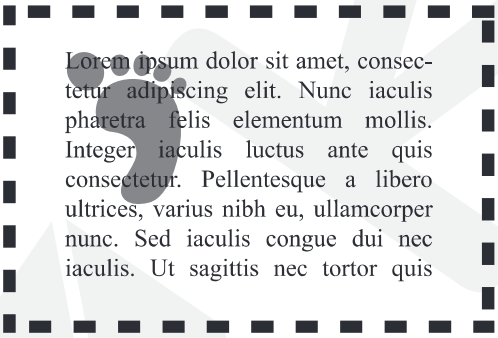
Valor	Descrição
border-box	Canto superior esquerdo da área delimitada pelas bordas.
padding-box	Canto superior esquerdo da área delimitada pelas margens internas (padrão).
content-box	Canto superior esquerdo da área delimitada pelo conteúdo.




```
background-origin: border-box;
background-repeat: no-repeat;
```




```
background-origin: padding-box;
background-repeat: no-repeat;
```



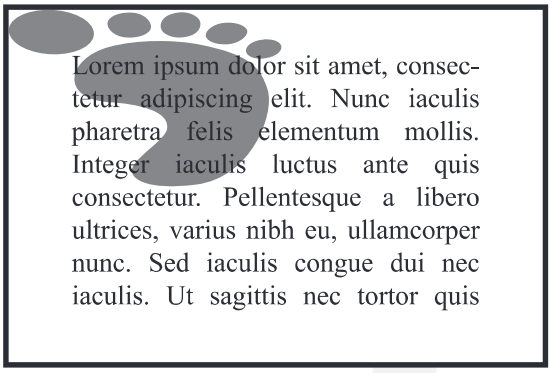

```
background-origin: content-box;
background-repeat: no-repeat;
```

background-size

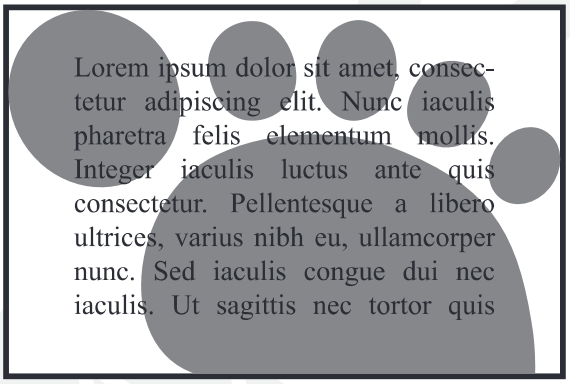
Quando uma imagem é utilizada como background de um elemento HTML, podemos definir o tamanho dessa imagem com a propriedade **background-size**.

Valor	Descrição
auto auto	Tamanho original da imagem (padrão).
auto	O mesmo que "auto auto".
medida auto	A *medida* corresponde à largura. A altura é calculada de forma proporcional à largura.
auto *medida*	A *medida* corresponde à altura. A largura é calculada de forma proporcional à altura.

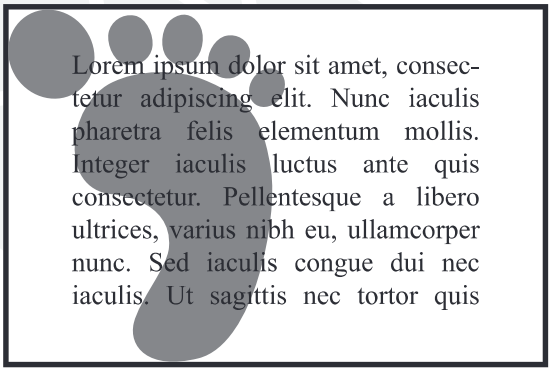
Valor	Descrição
medida1 *medida2*	A *medida1* corresponde à largura e a *medida2* à altura.
x% *y%*	O valor *x* corresponde à largura e o *y* à altura. Esses valores são porcetagens relativas à dimensão do elemento HTML.
x% auto	O valor *x* corresponde à largura. Esse valor é uma porcentagem relativa à largura do elemento HTML. A altura é calculada automaticamente de forma proporcional a largura.
auto *y%*	O valor *y* corresponde à altura. Esse valor é uma porcentagem relativa à altura do elemento HTML. A largura é calculada automaticamente de forma proporcional a altura.
cover	A imagem de background é redimensionada proporcionalmente de tal forma a cobrir toda a área do elemento HTML.
contain	A imagem de background é redimensionada proporcionalmente de tal forma a ter o maior tamanho possível sem extrapolar a área do elemento HTML.



→ `background-size: 50% 50%`



→ `background-size: cover`



→ `background-size: contain`

background

A propriedade **background** é um atalho para as outras propriedades de background. A sintaxe dessa propriedade é:

```
background. color position size repeat origin clip attachment image,
```

Exercícios de Fixação

6 No projeto **css**, crie um arquivo chamado **backgrounds.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Backgrounds</title>
    <link rel="stylesheet" type="text/css" href="backgrounds.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
    <div id="div5"></div>
  </body>
</html>
```

Código HTML 3.9: backgrounds.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao6.zip

7 No projeto **css**, crie um arquivo CSS chamado **backgrounds.css**.

```
div {
  width. 200px,
  height. 100px,
  margin. 10px,
  border. 5px black dashed,
}

#div1 {
  background-color. yellow,
}

#div2 {
  background-color. #c1a4cd,
  background-clip. padding-box,
}

#div3 {
  background-image.url("http.//k19.com.br/figs/seguro-treinamento.png"),
  background-repeat. no-repeat,
}

#div4 {
  background-image.url("http.//k19.com.br/figs/seguro-treinamento.png"),
  background-repeat. no-repeat,
  background-origin. border-box,
}
```

```
27
28 #div5 {
29   background-image.url("http.//k19.com.br/figs/seguro-treinamento.png"),
30   background-repeat. no-repeat,
31   background-size. contain,
32 }
33
34 body {
35   background-image. url("http.//k19.com.br/figs/main-header-logo.png"),
36   background-repeat. no-repeat,
37   background-attachment. fixed,
38   background-position. top center,
39 }
```

Código CSS 3.10: backgrounds.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao7.zip

8 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/backgrounds.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/backgrounds.html.

Textos

Podemos controlar a formatação dos textos contidos nos documentos HTML com as seguintes propriedades CSS.

color

Podemos definir a cor do texto contido em um elemento HTML através da propriedade **color**.

```
1 p {
2   color. blue,
3 }
```

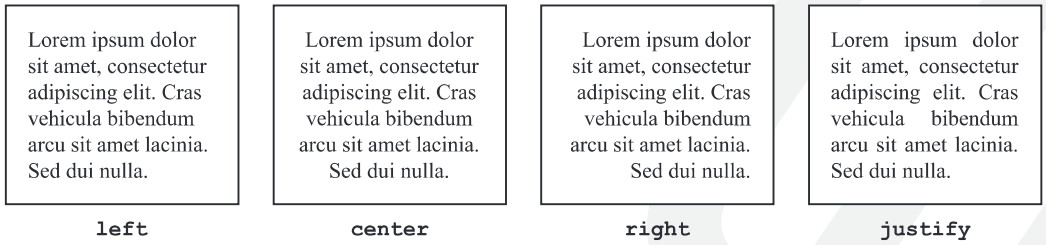
Valor	Descrição
color	Uma cor CSS.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

text-align

O alinhamento do texto contido em um elemento HTML pode ser determinado com a propriedade **text-align**.

Valor	Descrição
left	Alinhado à esquerda (padrão).
right	Alinhado à direita.
center	Centralizado.
justify	Justificado.

Valor	Descrição
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



text-decoration

Podemos definir a decoração dos textos contidos em um elemento HTML com a propriedade **text-decoration**.

Valor	Descrição
none	Texto sem decoração (padrão).
underline	Texto com traço inferior.
overline	Texto com traço superior.
line-through	Texto riscado.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `text-decoration: none`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `text-decoration: underline`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `text-decoration: overline`

~~Lorem ipsum dolor sit amet, consectetur adipiscing elit.~~ ⟶ `text-decoration: line-through`

text-transform

A capitalização dos textos pode ser determinada através da propriedade **text-transform**.

Valor	Descrição
none	Texto normal (padrão).
capitalize	A primeira letra de cada palavra será transformada em maiúscula.
uppercase	Todas as letras do texto serão transformadas em maiúsculas.
lowercase	Todas as letras do texto serão transformadas em minúsculas.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet. Consectetur adipiscing elit. ⟶ `text-transform: none`

Lorem Ipsum Dolor Sit Amet. Consectetur Adipiscing. ⟶ `text-transform: capitalize`

LOREM IPSUM DOLOR SIT AMET. CONSECTUR. ⟶ `text-transform: uppercase`

lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `text-transform: lowercase`

text-indent

Normalmente, indentamos a primeira linha de um bloco de texto. Podemos definir o comprimento dessa indentação através da propriedade **text-indent**.

Valor	Descrição
medida	A *medida* corresponde ao tamanho da indentação.
x%	O valor *x* é o tamanho da indentação em porcentagem relativa à largura do elemento HTML pai.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu.
Sed dui nulla, cursus et lacinia eu, vulputate ac dolor.
Quisque faucibus congue congue.

`text-indent: 0`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu.
Sed dui nulla, cursus et lacinia eu, vulputate ac dolor.
Quisque faucibus congue congue.

`text-indent: 20px`

letter-spacing

O espaçamento entre as letras pode ser determinado através da propriedade **letter-spacing**.

Valor	Descrição
normal	Espaçamento padrão definido para a fonte utilizada (padrão).
medida	A *medida* corresponde ao espaçamento extra desejado. Esse valor pode ser negativo.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet. Consectetur adipiscing elit... ⟶ `letter-spacing: normal`

L o r e m i p s u m d o l o r s i t a m e t . . . ⟶ `letter-spacing: 10px`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean mats nam ... ⟶ `letter-spacing: -2px`

word-spacing

O espaçamento entre as palavras pode ser determinado com a propriedade **word-spacing**.

Valor	Descrição
normal	Espaçamento padrão definido para a fonte utilizada (padrão).
medida	A *medida* corresponde ao espaçamento extra desejado. Esse valor pode ser negativo.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet. Consectetur adipiscing elit... ⟶ `word-spacing: 0`

Lorem ipsum dolor sit amet. Consectetur adipisc... ⟶ `word-spacing: 4px`

Lorem ipsum dolor sit amet. Consectetur adipiscing elit. Pellentesque dolor tincid... ⟶ `word-spacing: -4px`

word-wrap

Podemos controlar, através da propriedade **word-wrap**, como as palavras são divididas quando ocorre uma quebra de linha automática.

Valor	Descrição
normal	As palavras são divididas apenas nos "pontos de quebras" definidos com o elemento **wbr** (padrão).
break-word	As palavras podem ser divididas em qualquer ponto.

Otorrinolaringologista.

`word-wrap: normal`

Otorrinolaringo
logista.

`word-wrap: break-word`

line-height

A altura das linhas de um texto pode ser determinada com a propriedade **line-height**.

Valor	Descrição
normal	Altura padrão de linha (padrão).
x	A altura das linhas é o produto do valor *x* pelo tamanho da fonte atual.
medida	A *medida* corresponde à altura das linhas.
x%	O valor *x* é a altura das linhas em porcentagem relativa ao tamanho da fonte atual.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu.
Sed dui nulla, cursus et lacinia eu, vulputate ac dolor.
Quisque faucibus congue congue.

`line-height: normal`

Lorem ipsum dolor sit amet, consectetur adipiscing

elit. Cras vehicula bibendum arcu.

Sed dui nulla, cursus et lacinia eu, vulputate ac

`line-height: 20px`

white-space

É possível controlar a forma de tratamento dos espaços em branco e das quebras de linha através da propriedade **white-space**. Os valores possíveis para essa propriedade são:

Valor	Descrição
normal	Sequências de espaços em branco contidas no código HTML são tratadas como um único espaço. As quebras de linha contidas no código HTML **não** geram quebras de linha na página renderizada. Quebras de linha automáticas são inseridas quando necessário (padrão).

Valor	Descrição
nowrap	Sequências de espaços em branco contidas no código HTML são tratadas como um único espaço. As quebras de linha contidas no código HTML **não** geram quebras de linha na página renderizada. Nenhuma quebra de linha é inserida automaticamente.
pre	Sequências de espaços em branco são completamente consideradas. As quebras de linha contidas no código HTML geram quebras de linha na página renderizada. Nenhuma quebra de linha é inserida automaticamente.
pre-line	Sequências de espaços em branco contidas no código HTML são tratados como um único espaço. As quebras de linha contidas no código HTML geram quebras de linha na página renderizada. Quebras de linha automáticas são inseridas quando necessário.
pre-wrap	Sequências de espaços em branco são completamente consideradas. As quebras de linha contidas no código HTML geram quebras de linha na página renderizada. Quebras de linha automáticas são inseridas quando necessário.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

HTML

```
<div>Lorem    ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula
bibendum arcu.

          Sed    dui nulla, cursus et lacinia eu, vulputate ac dolor.
          Quisque faucibus congue congue.</div>
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu. Sed dui nulla, cursus et lacinia eu, vulputate ac dolor. Quisque faucibus congue congue.

white-space: normal

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

white-space: nowrap

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu.

Sed dui nulla, cursus et lacinia eu, vulputate ac dolor.
Quisque faucibus congue congue.

white-space: pre

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu.

Sed dui nulla, cursus et lacinia eu, vulputate ac dolor.
Quisque faucibus congue congue.

white-space: pre-line

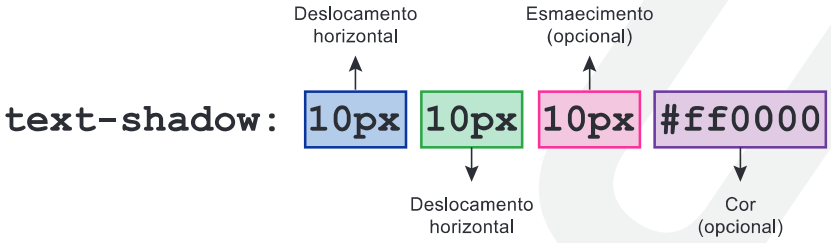
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula bibendum arcu.

Sed dui nulla, cursus et lacinia eu, vulputate ac dolor.
Quisque faucibus congue congue.

white-space: pre-wrap

text-shadow

Para adicionar uma sombra no texto de um elemento HTML, devemos utilizar a propriedade **text-shadow**. Para usar essa propriedade, devemos definir as seguintes informações sobre a sobra desejada: a posição horizontal, a posição vertical, o raio do blur e a cor.



Várias sombras podem ser aplicadas ao texto de um elemento HTML. As informações que definem essas sombras devem ser separadas por vírgula.

```
text-shadow: 10px 10px 10px #ff0000, -10px -10px 10px #0000ff
```

Outras propriedades

A linguagem CSS define mais algumas propriedades de formatação de texto.

- unicode-bdi
- hanging-punctuation
- punctuation-trim
- text-align-last
- text-justify
- text-outline
- text-overflow
- text-wrap
- direction

- word-break

Exercícios de Fixação

9 No projeto **css**, crie um arquivo chamado **textos.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Textos</title>
    <link rel="stylesheet" type="text/css" href="textos.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
      vehicula bibendum arcu sit amet lacinia. Sed dui nulla.
    </p>
    <p id="p2">
      Otorrinolaringologista.
    </p>
    <p id="p3">
      Lorem      ipsum dolor sit amet,
      consectetur adipiscing elit. Cras vehicula
          bibendum arcu.
          Sed
      dui nulla, cursus et lacinia eu, vulputate ac dolor.

      Quisque faucibus congue congue.
    </p>
    <p id="p4">
      Jonas Hirata.
    </p>
  </body>
</html>
```

Código HTML 3.10: textos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao9.zip

10 No projeto **css**, crie um arquivo CSS chamado **textos.css**.

```
p {
  border. 2px black solid,
}

#p1 {
  width. 400px,
  color. blue,
  text-align. justify,
  text-decoration. line-through,
  text-transform. uppercase,
  text-indent. 20px,
  letter-spacing. 2px,
  line-height. 25px,
  word-spacing. 11px,
}

#p2 {
  width. 100px,
  word-wrap. break-word,
```

```
}

#p3 {
  width. 400px,
  white-space. pre-line,
}

#p4 {
  width. 100px,
  text-transform. uppercase,
  letter-spacing. 5px,
  text-shadow. 2px 2px #ff0000,
}
```

Código CSS 3.12: textos.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao10.zip

11 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/textos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/textos.html.

Fontes

Podemos controlar as fontes utilizadas nos textos de uma página web com as seguintes propriedades.

font-family

Uma lista com as fontes desejadas ordenadas por prioridade pode ser determinada através da propriedade **font-family**.

```
font-family. "Arial Black", "Georgia", "Courier New",
```

Eventualmente, uma determinada fonte pode não estar disponível. Os navegadores devem utilizar a primeira fonte disponível da lista definida com a propriedade **font-family**. Por isso, essa lista deve ser ordenada da fonte com maior prioridade para a fonte com menor prioridade.

Algumas fontes são consideradas seguras (Web Safe Fonts). Essa qualidade é atribuída às fontes que são suportadas em praticamente todos os sistemas operacionais e navegadores. Uma lista com essas fontes pode ser consultada no endereço http://cssfontstack.com/.

A linguagem CSS classifica as fontes em 5 grupos genéricos:

- **serif**: Fontes serifadas. Os acabamentos nas extremidades das hastes das letras são denominados serifas.
- **sans-serif**: Fontes não serifadas.
- **cursive**: Fontes com estilo semelhante à escrita manual.

- **fantasy**: Fontes que possuem caracteres com decorações.
- **monospace**: Fontes que possuem caracteres com largura exatamente igual.

Se qualquer fonte de um determinado grupo genérico pode ser utilizada, devemos adicionar o nome desse grupo na lista definida com a propriedade **font-family**. Nesse caso, os próprios navegadores escolhem uma fonte do grupo genérico escolhido.

```
1 font-family. "Arial Black", "Georgia", "Courier New", "sans-serif",
```

No exemplo acima, se as fontes "Arial Black", "Georgia" e "Courier New" não estiverem disponíveis, os navegadores utilizarão alguma fonte do grupo **sans-serif**.

font-size

O tamanho da fonte é determinado através da propriedade **font-size**.

Valor	Descrição
xx-small	Tamanho de fonte extra extra pequeno.
x-small	Tamanho de fonte extra pequeno.
small	Tamanho de fonte pequeno.
medium	Tamanho de fonte médio (padrão).
large	Tamanho de fonte grande.
x-large	Tamanho de fonte extra grande.
xx-large	Tamanho de fonte extra extra grande.
smaller	Um tamanho de fonte menor do que o do elemento HTML pai.
larger	Um tamanho de fonte maior do que o do elemento HTML pai.
medida	A *medida* corresponde ao tamanho da fonte.
x%	O valor *x* é o tamanho da fonte. Esse valor é uma porcentagem do tamanho da fonte do elemento HTML pai.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

9px ⟶ K19 Treinamentos

xx-small ⟶ K19 Treinamentos

x-small ⟶ K19 Treinamentos

small ⟶ K19 Treinamentos

medium ⟶ K19 Treinamentos

large ⟶ K19 Treinamentos

x-large ⟶ K19 Treinamentos

xx-large ⟶ K19 Treinamentos

font-style

Podemos determinar o estilo da fonte através da propriedade **font-style**.

Valor	Descrição
normal	Fonte normal (padrão).
italic	Fonte itálica.
oblique	Fonte oblíqua.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `font-style: normal`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `font-style: italic`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `font-style: oblique`

font-variant

Para determinar que o texto de um elemento HTML deve ser exibido em **small caps**, devemos utilizar a propriedade **font-vaiant**. Todas as letras de um texto em **small caps** são transformadas em maiúsculas e o tamanho da fonte da primeira letra de cada palavra é maior do que o tamanho da fonte das demais letras.

Valor	Descrição
normal	Fonte normal (padrão).
small-caps	Fonte em small caps.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. ⟶ `font-variant: normal`

LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING E... ⟶ `font-variant: small-caps`

font-weight

A grossura dos caracteres pode ser definida com a propriedade **font-weight**.

Valor	Descrição
normal	Grossura normal (padrão).
bold	Negrito.
100	Grossura 100.
200	Grossura 200.
300	Grossura 300.
400	Grossura 400. Equivale ao valor “normal”
500	Grossura 500.
600	Grossura 600.
700	Grossura 700. Equivale ao valor “bold”
800	Grossura 800.
900	Grossura 900.
bolder	Uma grossura maior do que a do elemento HTML pai.
lighter	Uma grossura menor do que a do elemento HTML pai.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Importante
Normalmente, as fontes não definem uma grossura diferente para cada um dos possíveis valores da propriedade **font-weight**. Por isso, geralmente, dois ou mais valores são mapeados para a mesma grossura de fonte.

font-weight

Fonte utilizada:
Roboto
http://google.com/fonts/specimen/Roboto

Curiosidade:
A fonte Roboto é a fonte padrão utilizada nos sistemas Android a partir da versão 4.0.

K19 Treinamentos → `font-weight: normal`
K19 Treinamentos → `font-weight: 100`
K19 Treinamentos → `font-weight: 200`
K19 Treinamentos → `font-weight: 300`
K19 Treinamentos → `font-weight: 400`
K19 Treinamentos → `font-weight: 500`
K19 Treinamentos → `font-weight: 600`
K19 Treinamentos → `font-weight: 700`
K19 Treinamentos → `font-weight: 800`
K19 Treinamentos → `font-weight: 900`
K19 Treinamentos → `font-weight: bold`

font

A propriedade **font** é um atalho para as outras propriedades de fonte.

```
font. font-style font-variant font-weight font-size/line-height font-family,
```

@font-face

A regra **@font-face** permite utilizar uma fonte mesmo que ela não esteja disponível no sistema operacional do usuário. A fonte pode estar em um arquivo local (máquina do usuário) ou remoto (servidor). Para cada fonte que deve ser adicionada, devemos definir uma regra **font-face**. Os navegadores carregam as fontes sob demanda, ou seja, somente se elas forem necessárias. Veja um exemplo de utilização dessa regra.

```
@font-face {
  font-family. 'K19',
  src. url('http.//www.k19.com.br/fonts/K19.woff'),
}
```

A função **url()** deve ser utilizada quando queremos uma fonte que está armazenada remotamente. Para carregar uma fonte disponível na máquina do usuário, devemos utilizar a função **local()**, passando como parâmetro o nome da fonte desejada.

```
@font-face {
  font-family. 'NomeDaFonte',
  src. local('nome-da-fonte'),
}
```

Para utilizar as fontes definidas na regra **@font-face**, basta definir a a propriedade **font-family**.

```
font-family. 'NomeDaFonte',
```

Os formatos de fonte suportados pelos navegadores variam de navegador para navegador. Em uma regra **font-face**, podemos indicar a mesma fonte em vários formatos para suportar diversos navegadores.

```
@font-face {
  font-family. 'NomeDaFonte',
  src. url('nome-da-fonte.eot'), /* IE9 */
  src. url('nome-da-fonte.eot?#iefix') format('embedded-opentype'), /* IE6-IE8 */
     url('nome-da-fonte.woff') format('woff'), /* Navegadores Modernos */
     url('nome-da-fonte.ttf') format('truetype'), /* Safari, Android, iOS */
     url('nome-da-fonte.svg#webfontregular') format('svg'), /* iOS antigos */
}
```

O suporte dos navegadores aos diversos formatos de fonte está constantemente evoluindo. Há diversos sites que informam de maneira atualizada quais formatos são suportados por cada navegador. Recomendamos o site **Can I use** (http.//caniuse.com/).

Existem diversos repositórios de fontes pagas e/ou gratuitas. Podemos utilizar as fontes desses repositórios através da regra **font-face**. Um dos repositórios mais importantes é o **Google Fonts** (http.//www.google.com/fonts).

Outras propriedades

A linguagem CSS define mais algumas propriedades de formatação de fonte.

- font-size-adjust

- font-stretch

Exercícios de Fixação

12 No projeto **css**, crie um arquivo chamado **fontes.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Fontes</title>
    <link rel='stylesheet' type='text/css'
      href='http.//fonts.googleapis.com/css?family=Butcherman'>
    <link rel="stylesheet" type="text/css" href="fontes.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
      vehicula bibendum arcu sit amet lacinia. Sed dui nulla.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
      vehicula bibendum arcu sit amet lacinia. Sed dui nulla.
    </p>
  </body>
</html>
```

Código HTML 3.11: fontes.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao12.zip

13 No projeto **css**, crie um arquivo CSS chamado **fontes.css**.

```
p {
  border. 2px black solid,
}

#p1 {
  width. 400px,
  font-family. "Arial Black", "Georgia", "Courier New", "sans-serif",
  font-size. xx-large,
  font-style. italic,
  font-variant. small-caps,
  font-weight. bold,
}

#p2 {
  width. 400px,
  font-family. 'Butcherman',
}
```

Código CSS 3.20: fontes.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao13.zip

14 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/fontes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/~<USUARIO>/css/public_html/fontes.html`.

Listas

Podemos controlar a formatação das listas com as seguintes propriedades.

list-style-type

Podemos definir o símbolo utilizado nos itens de uma determinada lista com a propriedade **list-style-type**. A listagem completa desses símbolos pode ser consultada no endereço `http.//www.w3.org/TR/css3-lists/#ua-stylesheet`. Veja alguns possíveis valores da propriedade **list-style-type**.

Listas sem ordem

• K19 Treinamentos	◦ K19 Treinamentos	▪ K19 Treinamentos
• K19 Treinamentos	◦ K19 Treinamentos	▪ K19 Treinamentos
• K19 Treinamentos	◦ K19 Treinamentos	▪ K19 Treinamentos
`disc`	`circle`	`square`

Listas com ordem

1. K19 Treinamentos	I. K19 Treinamentos	i. K19 Treinamentos
2. K19 Treinamentos	II. K19 Treinamentos	ii. K19 Treinamentos
3. K19 Treinamentos	III. K19 Treinamentos	iii. K19 Treinamentos
`decimal`	`upper-roman`	`lower-roman`
A. K19 Treinamentos	a. K19 Treinamentos	א. K19 Treinamentos
B. K19 Treinamentos	b. K19 Treinamentos	ב. K19 Treinamentos
C. K19 Treinamentos	c. K19 Treinamentos	ג. K19 Treinamentos
`upper-alpha`	`lower-alpha`	`hebrew`
α. K19 Treinamentos	ア. K19 Treinamentos	一. K19 Treinamentos
β. K19 Treinamentos	イ. K19 Treinamentos	二. K19 Treinamentos
γ. K19 Treinamentos	ウ. K19 Treinamentos	三. K19 Treinamentos
`lower-greek`	`katakana`	`cjk-ideographic`

list-style-image

É possível definir uma imagem para ser utilizada nos itens de uma determinada lista com a propriedade **list-style-image**.

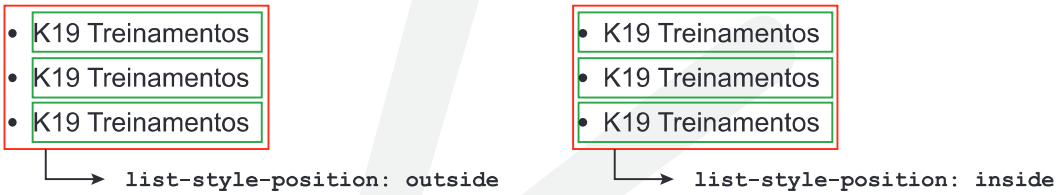
Valor	Descrição
url(*"url"*)	URL da imagem.

Valor	Descrição
none	Sem imagem (padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

list-style-position

Por padrão, os bullets de uma lista não são exibidos dentro dos boxes dos itens. Podemos alterar esse comportamento através da propriedade **list-style-position**.

Valor	Descrição
inside	Os bullets são exibidos dentro dos boxes dos itens.
outside	Os bullets não são exibidos dentro dos boxes dos itens. (padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



list-style

A propriedade **list-style** é um atalho para as outras propriedades de lista.

```
list-style. list-style-type, list-style-position, list-style-image
```

Exercícios de Fixação

15 No projeto **css**, crie um arquivo chamado **listas.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Listas</title>
    <link rel="stylesheet" type="text/css" href="listas.css">
  </head>
  <body>
    <ul id="ul1">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
```

```
13      </ul>
14
15      <ul id="ul2">
16        <li>K01 - Lógica de Programação</li>
17        <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
18        <li>K03 - Modelo Relacional e SQL</li>
19      </ul>
20
21      <ol>
22        <li>K01 - Lógica de Programação</li>
23        <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
24        <li>K03 - Modelo Relacional e SQL</li>
25      </ol>
26    </body>
27  </html>
```

Código HTML 3.12: listas.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao15.zip

16 No projeto **css**, crie um arquivo CSS chamado **listas.css**.

```
1  li {
2    border. 2px black solid,
3    width. 400px,
4  }
5
6  #ul1 {
7    list-style-type. square,
8    list-style-position. inside,
9  }
10
11 #ul2 {
12   list-style-image. url('http.//k19.com.br/figs/k02-logo-small.png'),
13 }
14
15 ol {
16   list-style-type. upper-roman,
17 }
```

Código CSS 3.22: listas.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao16.zip

17 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/listas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

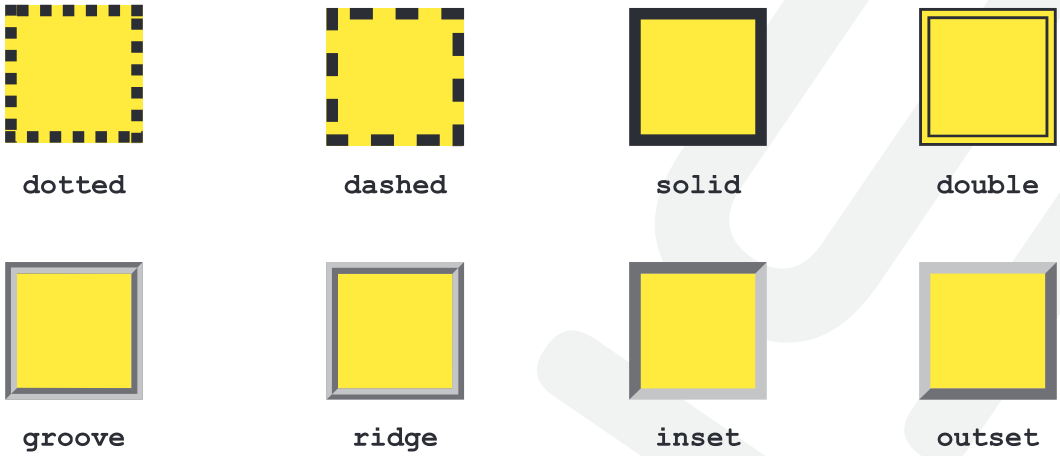
http://localhost/~<USUARIO>/css/public_html/listas.html.

Bordas

A linguagem CSS possui diversas propriedades para definir a formatação das bordas dos elementos HTML.

border-style

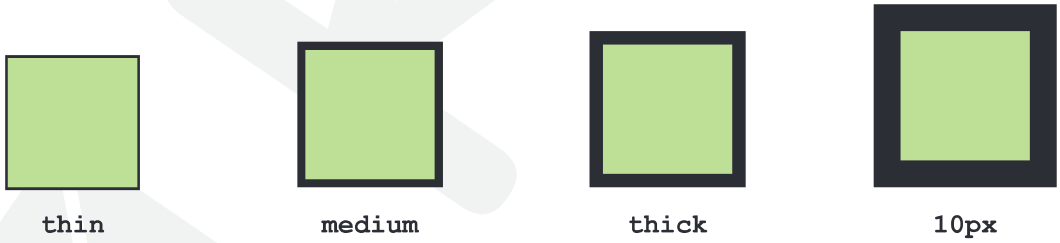
Podemos definir o estilo das bordas de um elemento HTML com a propriedade **border-style**. Veja os principais valores dessa propriedade.



border-width

A grossura das bordas pode ser determinada com a propriedade **border-width**.

Valor	Descrição
thin	Borda fina.
medium	Borda média (padrão).
thick	Borda grossa.
medida	A *medida* corresponde à grossura da borda.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



border-color

A propriedade **border-color** define a cor das bordas de um elemento HTML. Essa propriedade aceita o valor **transparent**(padrão) ou qualquer cor CSS.

Valor	Descrição
color	Uma cor CSS.
transparent	Faz a borda ser transparente.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

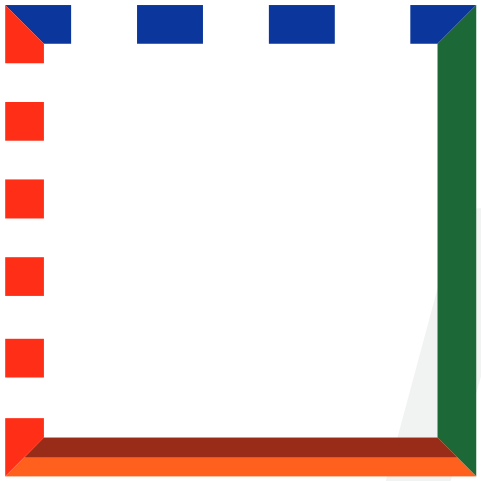
border

A propriedade **border** é um atalho para as propriedades que definem o estilo, a grossura e a cor das bordas dos elementos HTML.

```
1 border. border-width border-style border-color,
```

border-left-* border-top-* border-right-* border-bottom-*

O estilo, a grossura e a cor podem ser definidos individualmente para a borda da esquerda, borda superior, borda da direita e borda inferior de um elemento HTML. Veja o exemplo a seguir.



```
border-width: 10px;

border-top-style: dashed;
border-top-color: blue;

border-right-style: solid;
border-right-color: green;

border-bottom-style: groove;
border-bottom-color: orange;

border-left-style: dotted;
border-left-color: red;
```

border-left border-top border-right border-bottom

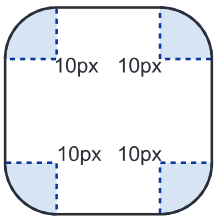
As propriedades **border-left**, **border-top**, **border-right** e **border-bottom** são atalhos para as propriedades que definem individualmente o estilo, a grossura e a cor das bordas da esquerda, superiores, da direita e inferiores dos elementos HTML. Veja alguns exemplos:

```
border-top: 10px dashed blue;
border-right: 10px solid green;
border-bottom: 10px groove orange;
border-left: 10px dotted red;
```

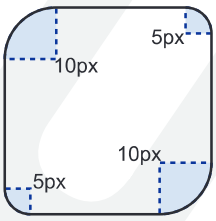
border-radius

Bordas com cantos arredondados podem ser definidas com a propriedade **border-radius**.

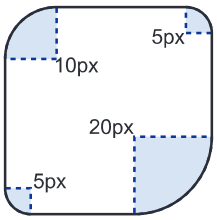
Valor	Descrição
medida	A *medida* corresponde ao raio.
x%	O valor *x* é o raio em porcentagem.



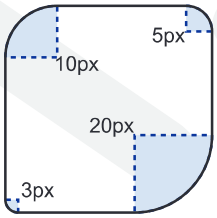
`border-radius: 10px`



`border-radius: 10px 5px`



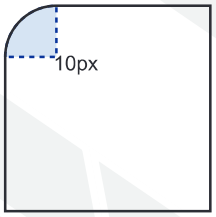
`border-radius: 10px 5px 20px`



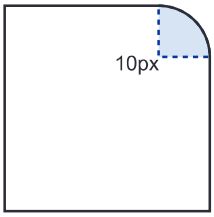
`border-radius: 10px 5px 20px 3px`

border-*-*-radius

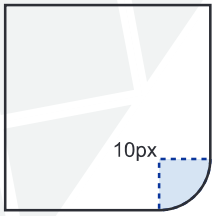
As propriedades **border-top-left-radius**, **border-top-right-radius**, **border-bottom-right-radius** e **border-bottom-left-radius** são utilizadas para definir o arredondamento das bordas dos cantos superior esquerdo, superior direito, inferior direito e inferior esquerdo respectivamente. Veja alguns exemplos:



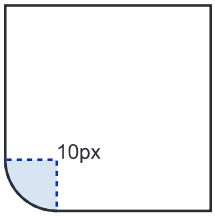
`border-top-left-radius: 10px`



`border-top-right-radius: 10px`



`border-bottom-right-radius: 10px`



`border-bottom-left-radius: 10px`

border-collapse

Podemos definir bordas para cada célula de uma tabela e para a própria tabela. Por padrão, essas bordas são exibidas separadamente. Mas, é possível determinar que duas bordas "vizinhas" sejam exibidas como se fossem uma só com a propriedade **border-collapse**.

Valor	Descrição
separate	Bordas separadas (padrão).
collapse	Bordas unidades.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Célula 1	Célula 2
Célula 3	Célula 4

`border-collapse: separate`

Célula 1	Célula 2
Célula 3	Célula 4

`border-collapse: collapse`

border-spacing

A propriedade **border-spacing** permite definir o espaçamento entre as células de uma tabela. Essa propriedade afeta o espaçamento entre as bordas somente se o valor da propriedade **border-collapse** for **separate**.

Valor	Descrição
medida1 medida2	A *medida1* corresponde ao espaçamento horizontal e a *medida2* corresponde ao espaçamento vertical.
medida	A *medida* corresponde ao espaçamento horizontal e vertical.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Célula 1	Célula 2
Célula 3	Célula 4

`border-spacing: 5px`

Célula 1	Célula 2
Célula 3	Célula 4

`border-spacing: 10px 5px`

Outras propriedades

A linguagem CSS define mais algumas propriedades de formatação de bordas.

- border-image
- border-image-outset
- border-image-repeat
- border-image-slice
- border-image-source
- border-image-width

Exercícios de Fixação

18 No projeto **css**, crie um arquivo chamado **bordas.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Bordas</title>
    <link rel="stylesheet" type="text/css" href="bordas.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
    <div id="div5"></div>

    <table id="tabela1">
      <tr>
          <td>K01</td>
          <td>K02</td>
          <td>K03</td>
      </tr>
    </table>

    <table id="tabela2">
      <tr>
          <td>K01</td>
          <td>K02</td>
          <td>K03</td>
      </tr>
    </table>
  </body>
</html>
```

Código HTML 3.13: bordas.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao18.zip

19 No projeto **css**, crie um arquivo CSS chamado **bordas.css**.

```
div {
  width. 200px,
  height. 100px,
  margin. 10px,
}

#div1 {
  border-style. solid,
  border-width. thin,
  border-color. red,
  border-radius. 10px,
}

#div2 {
  border-style. dotted,
  border-color. yellow,
  border-radius. 10px 30px,
}

#div3 {
```

```
21   border-style. dashed,
22   border-width. 4px,
23   border-color. blue,
24   border-radius. 10px 20px 30px 40px,
25 }
26
27 #div4 {
28   border-style. double,
29   border-width. 8px,
30   border-color. green,
31   border-top-left-radius. 10px,
32   border-top-right-radius. 20px,
33   border-bottom-right-radius. 30px,
34   border-bottom-left-radius. 40px,
35 }
36
37 #div5 {
38   border-left-style. solid,
39   border-left-width. thin,
40   border-left-color. red,
41
42   border-top-style. dotted,
43   border-top-color. yellow,
44
45   border-right-style. dashed,
46   border-right-width. 4px,
47   border-right-color. blue,
48
49   border-bottom-style. double,
50   border-bottom-width. 8px,
51   border-bottom-color. green,
52 }
53
54 table, td {
55   font-size. xx-large,
56   border. 1px solid black,
57   margin. 10px,
58 }
59
60 #tabela1 {
61   border-collapse. collapse,
62 }
63
64 #tabela2 {
65   border-spacing. 20px,
66 }
```

Código CSS 3.24: bordas.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao19.zip

20 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/bordas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

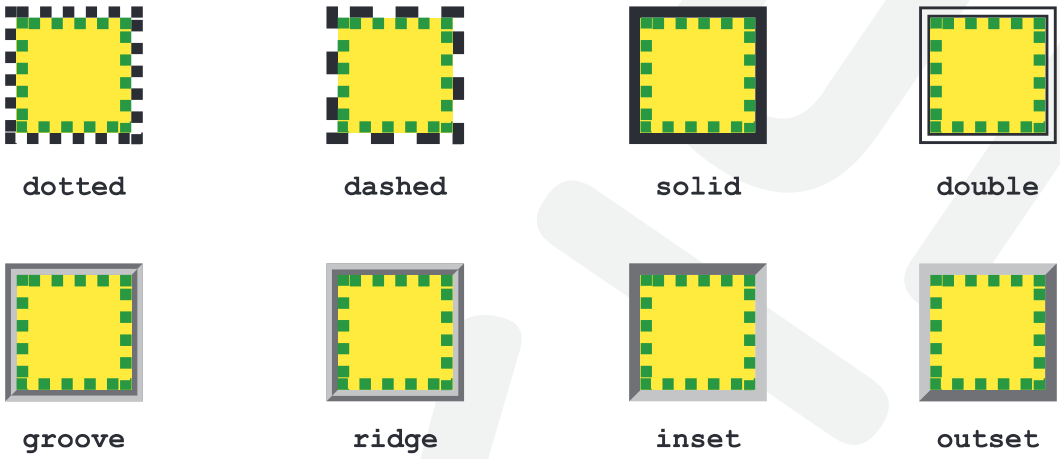
http.//localhost/~<USUARIO>/css/public_html/bordas.html.

Outline

O box de um elemento HTML pode ser contornado por uma linha denominada **outline**. Essa linha não afeta as dimensões dos boxes. A linguagem CSS possui diversas propriedades para definir a formatação do outline dos elementos HTML.

outline-style

Podemos definir o estilo do outline de um elemento HTML com a propriedade **outline-style**. Veja os principais valores dessa propriedade.



outline-color

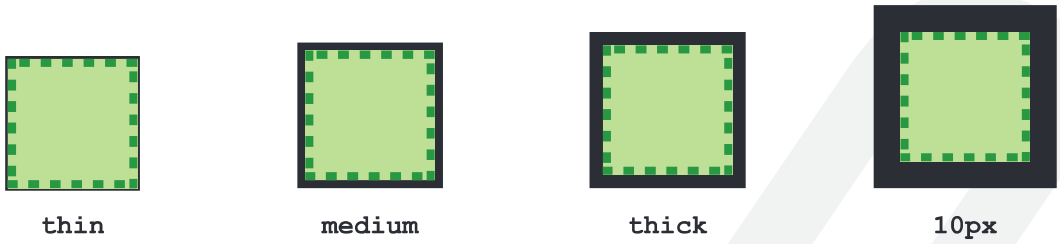
A propriedade **outline-color** define a cor do outline de um elemento HTML.

Valor	Descrição
color	Uma cor CSS.
invert	A cor inversa à cor do background (padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

outline-width

A grossura dos outlines pode ser determinada com a propriedade **outline-width**.

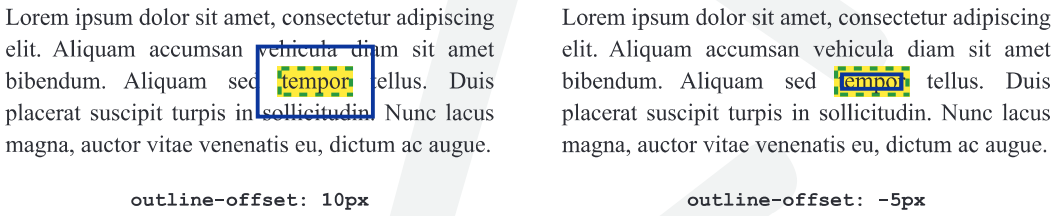
Valor	Descrição
thin	Outline fino.
medium	Outline médio (padrão).
thick	Outline grosso.
medida	A *medida* corresponde à grossura do outline.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



outline-offset

Podemos definir a distância do outline de um elemento HTML em relação às bordas do mesmo com a propriedade **outline-offset**.

Valor	Descrição
medida	A *medida* corresponde ao tamanho do outline-offset (0px é o valor padrão). Esse valor pode ser negativo.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



outline

A propriedade **outline** é um atalho para as propriedades **outline-color**, **outline-style** e **outline-width**. Veja a sintaxe dessa propriedade.

```
outline. outline-color outline-style outline-width,
```

Exercícios de Fixação

21 No projeto **css**, crie um arquivo chamado **outlines.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Outlines</title>
    <link rel="stylesheet" type="text/css" href="outlines.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.14: outlines.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao21.zip

22 No projeto **css**, crie um arquivo CSS chamado **outlines.css**.

```
div {
  width. 200px,
  height. 100px,
  border. 4px solid black,
  margin. 50px,
}

#div1 {
  outline-style. solid,
  outline-color. red,
  outline-width. 4px,
}

#div2 {
  outline-style. dashed,
  outline-color. blue,
  outline-width. 2px,
  outline-offset. -20px,
}

#div3 {
  outline-style. solid,
  outline-color. green,
  outline-width. 4px,
  outline-offset. 20px,
}
```

Código CSS 3.26: outlines.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao22.zip

23 No **Windows**, utilize o **Chrome** para acessar o endereço:

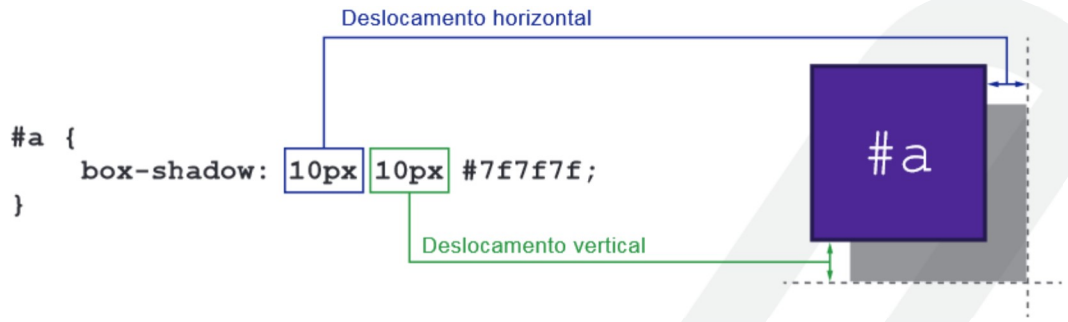
http://localhost/css/public_html/outlines.html.

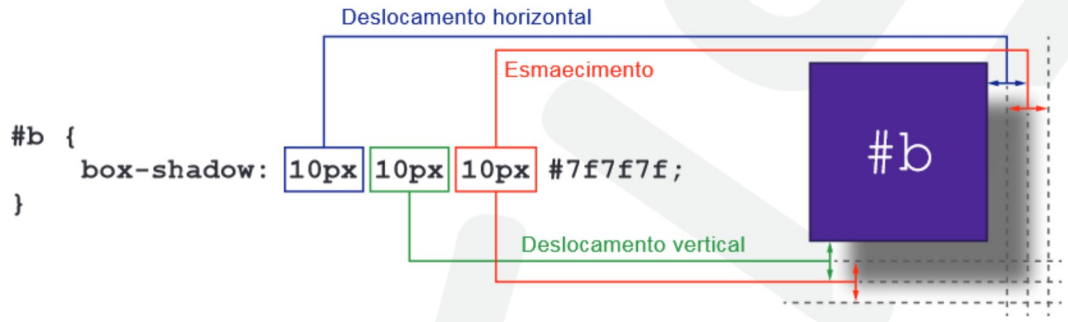
No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

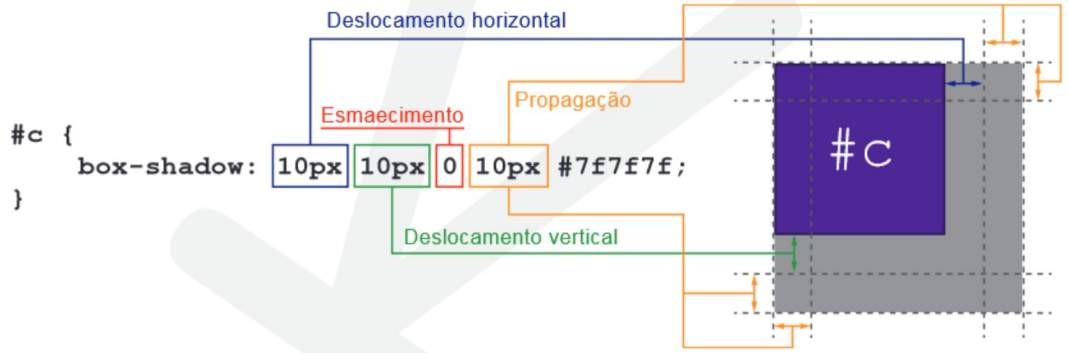
http://localhost/~<USUARIO>/css/public_html/outlines.html.

Sombras

A propriedade **box-shadow** permite adicionar sombras externas e internas nos boxes dos elementos HTML. O valor dessa propriedade pode ser definido de diversas maneiras. Confira, nas imagens abaixo, algumas formas de utilização da propriedade **box-shadow** para definir sombras externas.

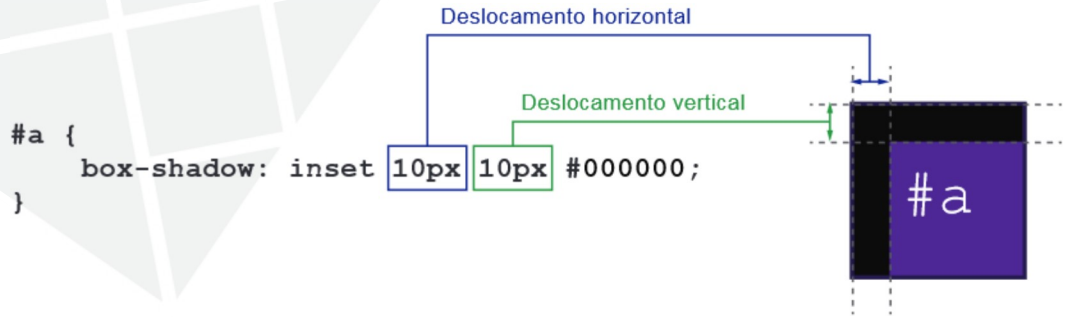


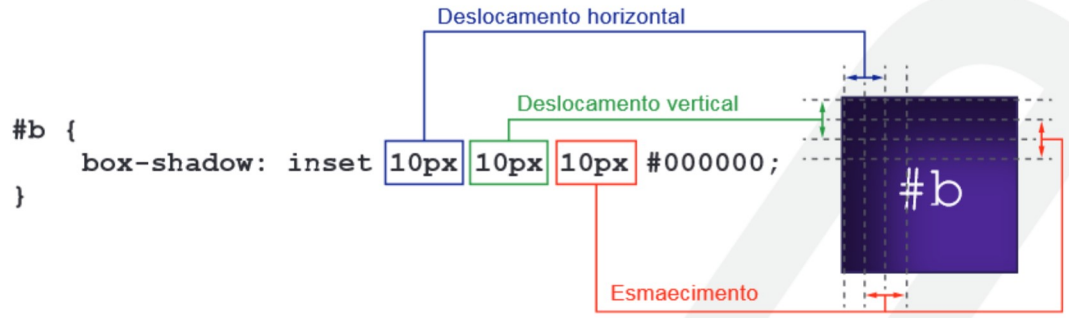


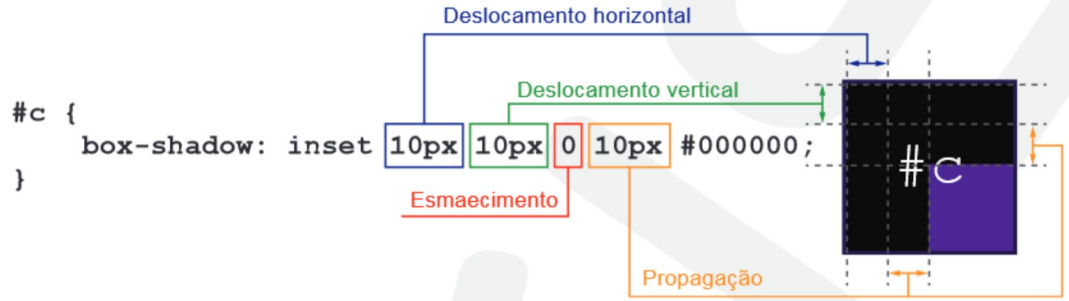


Os valores dos deslocamentos vertical e horizontal podem ser negativos fazendo com que a sombra externa seja projetada para cima e para a esquerda.

As regras para definir uma da sombra interna são praticamente as mesmas da sombra externa. A única diferença é que devemos começar a atribuição do valor com a palavra **inset**.







Exercícios de Fixação

24 No projeto **css**, crie um arquivo chamado **sombras.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Sombras</title>
    <link rel="stylesheet" type="text/css" href="sombras.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.15: sombras.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao24.zip

25 No projeto **css**, crie um arquivo CSS chamado **sombras.css**.

```
div {
  width. 200px,
  height. 100px,
  border. 4px solid black,
  margin. 50px,
}

#div1 {
```

```
  box-shadow. 10px 5px gray,
}

#div2 {
  box-shadow. 10px 5px 5px gray,
}

#div3 {
  box-shadow. inset 10px 10px gray,
}

#div4 {
  box-shadow. inset 10px 10px 5px gray,
}
```

Código CSS 3.27: sombras.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao25.zip

26 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/sombras.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/sombras.html.

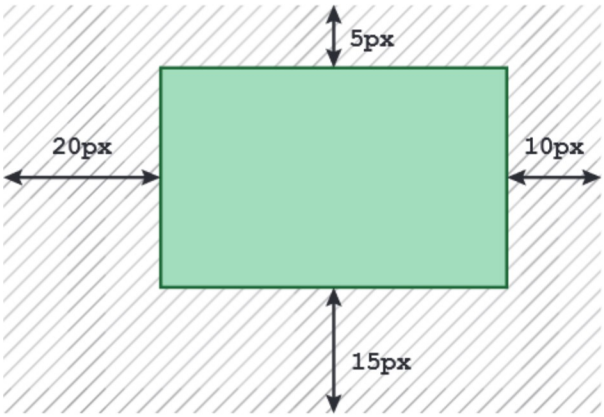
Margens

A linguagem CSS possui diversas propriedades para definir a formatação das margens internas e externas dos elementos HTML.

margin-*

Os tamanhos das margens externas do box de elemento HTML podem ser definidos individualmente com as propriedades **margin-left**, **margin-top**, **margin-right** e **margin-bottom**.

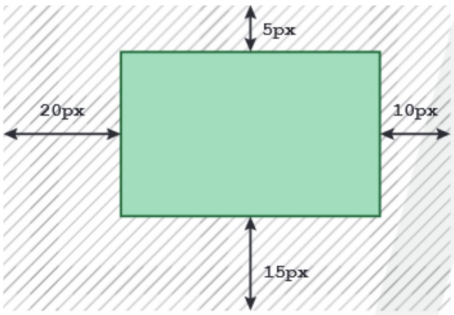
Valor	Descrição
auto	O tamanho da margem é calculado automaticamente pelos navegadores.
medida	A *medida* corresponde ao tamanho da margem (0px é o padrão).
x%	O valor *x* corresponde ao tamanho da margem. Esse valor é uma porcentagem da largura do elemento HTML pai.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



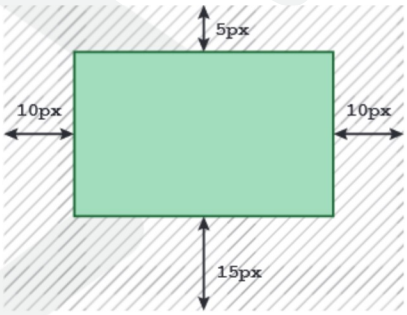

```
margin-top: 5px
margin-right: 10px
margin-bottom: 15px
margin-left: 20px
```

margin

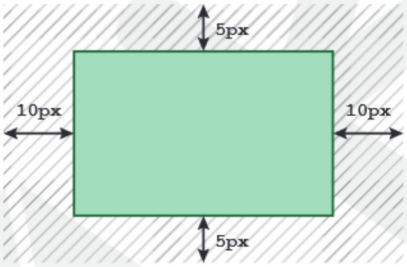
A propriedade **margin** é um atalho para as propriedades **margin-left**, **margin-top**, **margin-right** e **margin-bottom**. Podemos utilizar a propriedade **margin** de 4 formas diferentes:



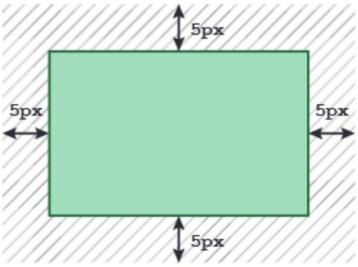

```
margin: 5px 10px 15px 20px
```




```
margin: 5px 10px 15px
```




```
margin: 5px 10px
```

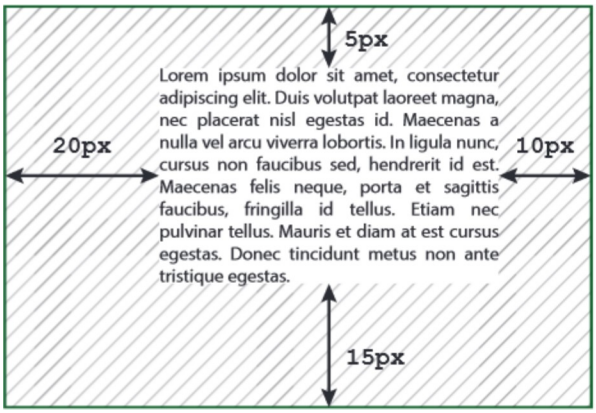



```
margin: 5px
```

padding-*

Os tamanhos das margens internas do box de elemento HTML podem ser definidos individualmente com as propriedades **padding-left**, **padding-top**, **padding-right** e **padding-bottom**.

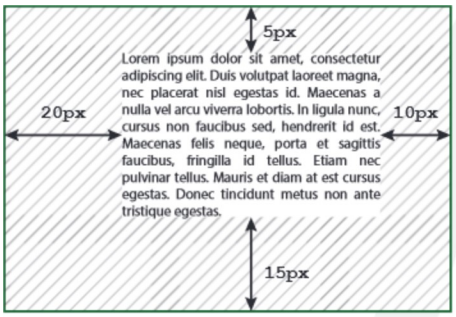
Valor	Descrição
medida	A *medida* corresponde ao tamanho da margem (0px é o padrão).
x%	O valor *x* corresponde ao tamanho da margem. Esse valor é uma porcentagem da largura do elemento HTML pai.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



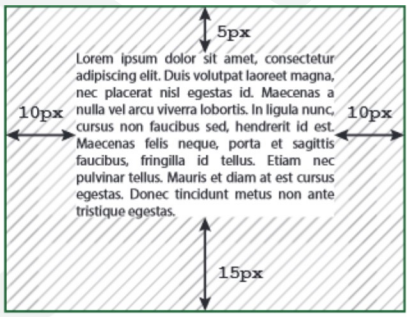

```
padding-top: 5px
padding-right: 10px
padding-bottom: 15px
padding-left: 20px
```

padding

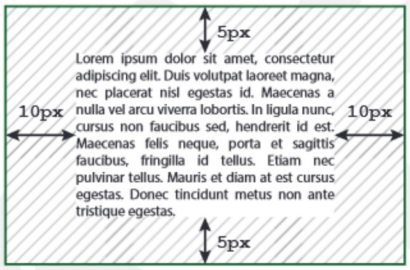
A propriedade **padding** é um atalho para as propriedades **padding-left**, **padding-top**, **padding-right** e **padding-bottom**. Podemos utilizar a propriedade **padding** de 4 formas diferentes:



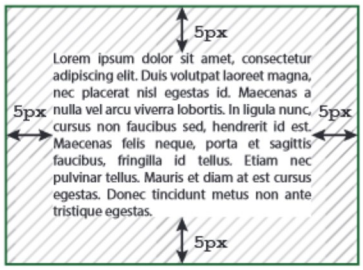

```
padding: 5px 10px 15px 20px
```




```
padding: 5px 10px 15px
```




```
padding: 5px 10px
```




```
padding: 5px
```

Mais Sobre

Considere o seguinte exemplo.

```
1 <div id="div1">
2   <div id="div2">...</div>
3 <div>
```

```
1 #div2 {
```

```
  width. 50%,
}
```

Podemos utilizar as propriedades de margem para alinhar de forma centralizada na horizontal o **div2**.

```
#div2 {
  margin-left. auto,
  margin-right. auto,
  width. 50%,
}
```

Exercícios de Fixação

27 No projeto **css**, crie um arquivo chamado **margens.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Margens</title>
    <link rel="stylesheet" type="text/css" href="margens.css">
  </head>
  <body>
    <div>
      <p id="p1">K19</p>
      <a id="a1" href="#">k19</a>
      <a id="a2" href="#">k19</a>
      <p id="p2">K19</p>
    </div>
  </body>
</html>
```

Código HTML 3.17: margens.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao27.zip

28 No projeto **css**, crie um arquivo CSS chamado **margens.css**.

```
div {
  width. 600px,
  border. 1px solid black,
}

p, a {
  border. 1px solid black,
}

#p1 {
  margin. 5px 10px 20px 30px,
  padding. 5px 0px 0px 20px,
}

#a1 {
  margin-left. 10px,
```

```
17   padding. 5px,
18 }
19
20 #a2 {
21   margin-left. 20px,
22   padding. 10px,
23 }
24
25 #p2 {
26   margin. 20px,
27   padding. 10px,
28 }
```

Código CSS 3.30: margens.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao28.zip

29 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/margens.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/margens.html.

Altura e Largura

A linguagem CSS possui algumas propriedades para controlar a largura e a altura dos elementos HTML.

height e width

A altura e a largura dos boxes dos elementos HTML podem ser definidas com as propriedades **height** e **width** respectivamente.

Valor	Descrição
auto	A altura ou largura são calculadas automaticamente pelos navegadores.
medida	A *medida* corresponde à largura ou a altura.
x%	A *medida* corresponde à largura ou a altura como porcentagem das dimensões do elemento HTML pai
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

min-height e max-height

A propriedade **height** é utilizada para determinar a altura exata do box de um elemento HTML. Uma outra abordagem é definir um limite inferior e/ou um limite superior para essa dimensão. Esses limites podem ser definidos com as propriedades **min-height** e **max-height**.

Valor	Descrição
none	Sem valor (padrão para **max-height**).
medida	A *medida* corresponde à altura máxima ou mínima desejada (0px é padrão para **min-height**).

Valor	Descrição
x%	O valor *x* corresponde à altura máxima ou mínima. Esse valor é uma porcentagem da altura do elemento HTML pai
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

min-width e max-width

A propriedade **width** é utilizada para determinar a largura exata do box de um elemento HTML. Uma outra abordagem é definir um limite inferior e/ou um limite superior para essa dimensão. Esses limites podem ser definidos com as propriedades **min-width** e **max-width**.

Valor	Descrição
none	Sem valor (padrão para **max-width**).
medida	A *medida* corresponde à largura máxima ou mínima desejada (0px é padrão para **min-width**).
x%	O valor *x* corresponde à largura máxima ou mínima. Esse valor é uma porcentagem da largura do elemento HTML pai
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Exercícios de Fixação

30 No projeto **css**, crie um arquivo chamado **altura-largura.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Altura e Largura</title>
    <link rel="stylesheet" type="text/css" href="altura-largura.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.18: altura-largura.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao30.zip

31 No projeto **css**, crie um arquivo CSS chamado **altura-largura.css**.

```
p {
    border. 1px solid black,
    max-width. 600px,
    max-height. 250px,
    min-height. 100px,
}

#p1 {
    width. 200px,
}

#p2 {
    width. 400px,
}
```

Código CSS 3.31: altura-largura.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao31.zip

32 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/altura-largura.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/altura-largura.html.

Display e Visibilidade

A linguagem CSS possui algumas propriedades para controlar a forma de exibição e a visibilidade dos elementos HTML.

display

A propriedade **display** define a forma de exibição dos elementos HTML. Os 5 principais valores dessa propriedade são: **inline**(padrão), **block**, **inline-block**, **list-item** e **none**.

Valor	Descrição
inline	Considere o box de um elemento HTML com **display: inline**. Esse box não gera quebras de linha e as suas dimensões são calculadas com base no conteúdo do elemento HTML. As propriedades **width** e **height** não modificam a largura e a altura do box. As margens e as bordas do box de um elemento HTML com **display: inline** "empurram" os boxes vizinhos horizontalmente. Verticalmente, os boxes vizinhos não são "empurrados".
block	Considere o box de um elemento HTML com **display: block**. Esse box gera quebras de linha e, por padrão, ocupa todo o espaço horizontal do elemento pai. Podemos controlar a largura e a altura desse box com as propriedades **width** e **height**.

Valor	Descrição
inline-block	O comportamento dos boxes dos elementos HTML com **display: inline-block** é similar ao dos boxes dos elementos HTML com **display: inline**. Basicamente, a diferença entre eles é que podemos modificar a largura e altura dos boxes dos elementos HTML com **display: inline-block**.
list-item	O comportamento dos boxes dos elementos HTML com **display: list-item** é similar ao dos boxes dos elementos HTML com **display: block**. Basicamente, a diferença entre eles é que um bullet é exibido no canto esquerdo de box de um elemento HTML com **display: list-item**.
none	Os boxes dos elementos HTML com **display: none** não são exibidos e não ocupam espaço na página.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis placerat suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`display: inline`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis

placerat

suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`display: block`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis placerat suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`display: inline-block`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis

- placerat

suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`display: list-item`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`display: none`

visibility

Podemos determinar se um elemento HTML deve ser exibido ou não através da propriedade **visibility**. O valor **visible**(padrão) indica que o elemento deve ser exibido e o valor **hidden** indica que o elemento não deve ser exibido.

Valor	Descrição
visible	Visível (padrão).
hidden	Invisível.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis placerat suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`visibility: visible`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`visibility: hidden`

Importante

Na renderização, os elementos HTML com **display: none** são totalmente desconsiderados. Em outras palavras, é como se eles simplesmente não existissem. Dessa forma, eles não afetam o posicionamento dos demais elementos. Por outro lado, os elementos HTML com **visibility: hidden** apesar de não serem exibidos fazem parte da página renderizada e podem afetar o posicionamento dos demais elementos pois o espaço ocupado por eles pode não ser nulo.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

`display: none`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed tempor tellus. Duis suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae venenatis eu, dictum ac augue. Nunc eleifend dignissim felis.

visibility: hidden

Exercícios de Fixação

33 No projeto **css**, crie um arquivo chamado **display-visibilidade.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Display e Visibilidade</title>
    <link rel="stylesheet" type="text/css" href="display-visibilidade.css">
  </head>
  <body>
    <p>
      Lorem ipsum dolor sit amet, <span id="span1">inline</span>
      adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum.
    </p>

    <p>
      Lorem ipsum dolor sit amet, <span id="span2">block</span>
      adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum.
    </p>

    <p>
      Lorem ipsum dolor sit amet, <span id="span3">inline-block</span>
      adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum.
    </p>

    <p>
      Lorem ipsum dolor sit amet, <span id="span4">list-item</span>
      adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum.
    </p>

    <p>
      Lorem ipsum dolor sit amet, <span id="span5">none</span>
      adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum.
    </p>

    <p>
      Lorem ipsum dolor sit amet, <span id="span6">hidden</span>
      adipiscing elit. Aliquam accumsan vehicula diam sit amet bibendum.
    </p>
  </body>
</html>
```

Código HTML 3.19: display-visibilidade.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao33.zip

34 No projeto **css**, crie um arquivo CSS chamado **display-visibilidade.css**.

```
p {
  border. 2px solid black,
```

```
3   width. 600px,
4 }
5
6 span {
7   padding. 15px,
8   margin. 40px,
9   width. 200px,
10  height. 100px,
11  border. 2px solid black,
12  background-color. yellow,
13 }
14
15 #span1 {
16  display. inline,
17 }
18
19 #span2 {
20  display. block,
21 }
22
23 #span3 {
24  display. inline-block,
25 }
26
27 #span4 {
28  display. list-item,
29 }
30
31 #span5 {
32  display. none,
33 }
34
35 #span6 {
36  visibility. hidden,
37 }
```

Código CSS 3.32: display-visibilidade.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao34.zip

35 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/display-visibilidade.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/display-visibilidade.html.

Opacidade

Podemos definir a opacidade dos elementos HTML com a propriedade **opacity**.

Valor	Descrição
x	O valor x é um número entre 0(totalmente transparente) e 1(sem transparência, padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

Exercícios de Fixação

36 No projeto **css**, crie um arquivo chamado **opacidade.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Opacidade</title>
    <link rel="stylesheet" type="text/css" href="opacidade.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
    <div id="div5"></div>
  </body>
</html>
```

Código HTML 3.20: opacidade.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao36.zip

37 No projeto **css**, crie um arquivo CSS chamado **opacidade.css**.

```
div {
  width. 100px,
  height. 100px,
  background-color. red,
}

#div1 {
  opacity. 0,
}

#div2 {
  opacity. 0.25,
}

#div3 {
  opacity. 0.5,
}

#div4 {
  opacity. 0.75,
}

#div5 {
  opacity. 1,
}
```

Código CSS 3.33: opacidade.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao37.zip

38 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/opacidade.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

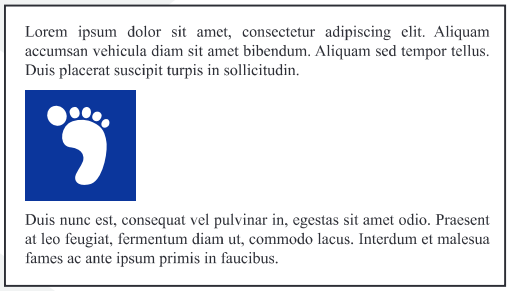
http.//localhost/~<USUARIO>/css/public_html/opacidade.html.

Posicionamento

A linguagem CSS define quatro formas para posicionar os boxes dos elementos HTML: **static**, **relative**, **fixed** e **absolute**. Mostraremos o funcionamento de cada uma dessas formas. A propriedade **position** determina a forma de posicionamento desejada.

static

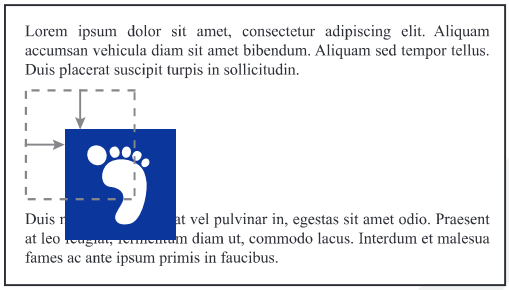
O valor **static** é o valor padrão da propriedade **position**. O box de um elemento com **position: static** é colocado na sua posição natural. Não podemos alterar a posição dos boxes dos elementos HTML com **position: static**.



`position: static`

relative

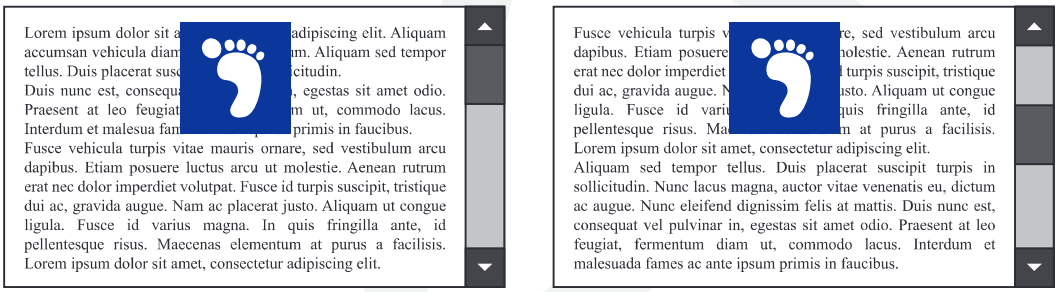
O box de um elemento com **position: relative** é colocado em uma posição relativa a sua posição natural. As propriedades **left**, **top**, **right** e **bottom** são utilizadas para determinar o deslocamento entre a posição desejada e a posição natural. O espaço que seria ocupado por esse box se ele estivesse em sua posição natural não é ocupado por nenhum outro elemento.



`position: relative`

fixed

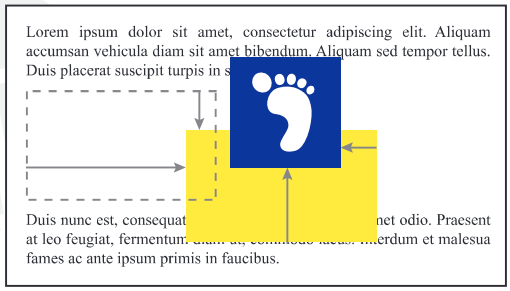
O box de um elemento com **position: fixed** é colocado em uma posição relativa à página e não acompanha a rolagem do conteúdo. As propriedades **left**, **top**, **right** e **bottom** são utilizadas para determinar o deslocamento entre a posição desejada e à pagina.



`position: fixed`

absolute

O box de um elemento com **position: absolute** é colocado em uma posição relativa à posição do box do elemento HTML ancestral mais próximo com **position** diferente de **static**. Se não existir um ancestral nessas condições a posição será relativa à página. As propriedades **left**, **top**, **right** e **bottom** são utilizadas para determinar o deslocamento entre a posição desejada e o ponto de referência.



`position: absolute`

*Figura 3.7: Elemento com **position** diferente de **static** utilizado como referência*

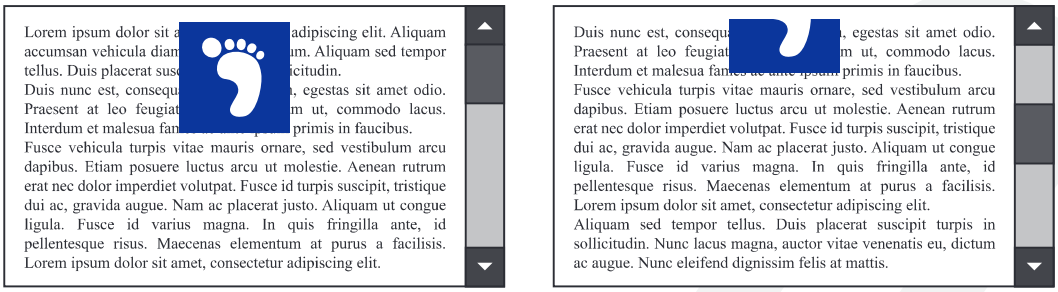
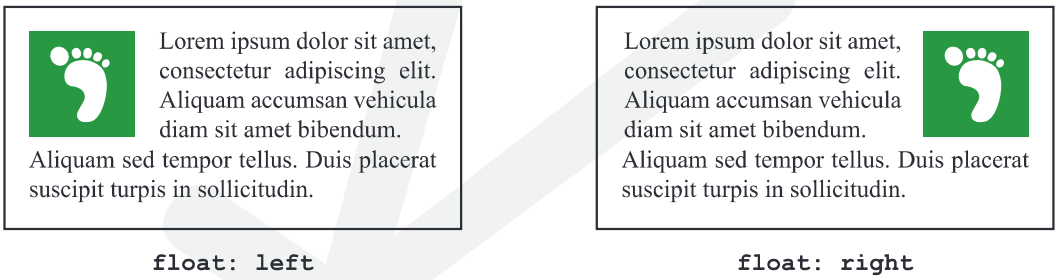


Figura 3.8: Página utilizada como referência

float

Podemos fazer o box de um elemento HTML flutuar ao redor dos demais elementos através da propriedade **float**.

Valor	Descrição
left	Flutuar à esquerda.
right	Flitar à direita.
none	Sem flutuação (padrão).
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



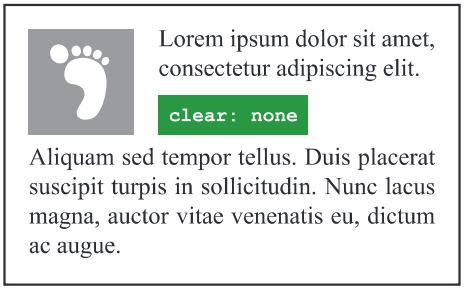
Importante
Os elementos HTML com **float** diferente de **none** podem ser reposicionados se a janela for redimensionada.

Importante
Os elementos HTML com **float** diferente de **none** não afetam a altura ou a largura do elemento HTML onde eles estão contidos.

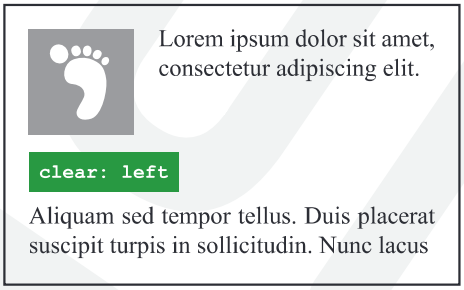
clear

Os box dos elementos HTML próximos do box de um elemento HTML om **float** diferente de **none** são "empurrados" horizontalmente à esquerda ou à direita. Podemos determinar que, ao invés de serem "empurrados" horizontalmente, eles sejam exibidos em uma nova linha através da propriedade **clear**.

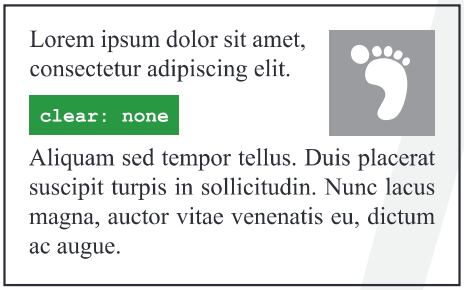
Valor	Descrição
left	Não permite boxes flutuando à esquerda dele.
right	Não permite boxes flutuando à direita dele.
both	Não permite boxes flutuando nem à esquerda nem à direita dele.
none	Permite boxes flutuando à esquerda e à direita (padrão) dele.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



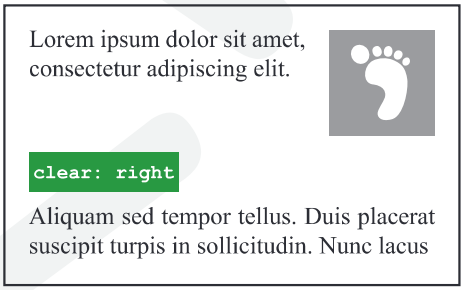
`clear: none`



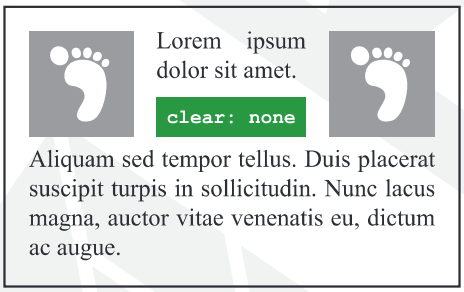
`clear: left`



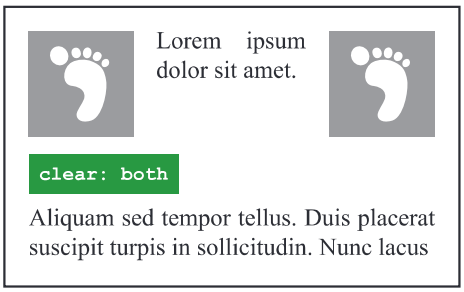
`clear: none`



`clear: rigth`



`clear: none`

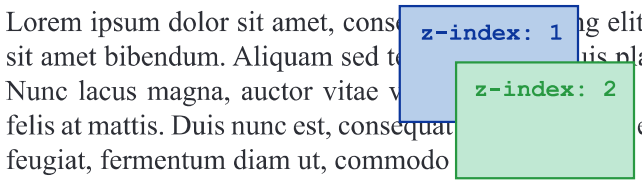


`clear: both`

z-index

A sobreposição dos boxes dos elementos HTML pode ser controlada através da propriedade **z-index**. Basicamente, essa propriedade permite determinar em qual plano um box deve ser exibido.

Valor	Descrição
auto	Mesmo plano do box do elemento HTML pai (padrão).
x	O valor *x* é o plano desejado. Esse valor pode ser negativo.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.



Lorem ipsum dolor sit amet, cons ng elit. Aliquam accumsan vehicula diam sit amet bibendum. Aliquam sed t uis placerat suscipit turpis in sollicitudin. Nunc lacus magna, auctor vitae v ac augue. Nunc eleifend dignissim felis at mattis. Duis nunc est, consequat egestas sit amet odio. Praesent at leo feugiat, fermentum diam ut, commodo

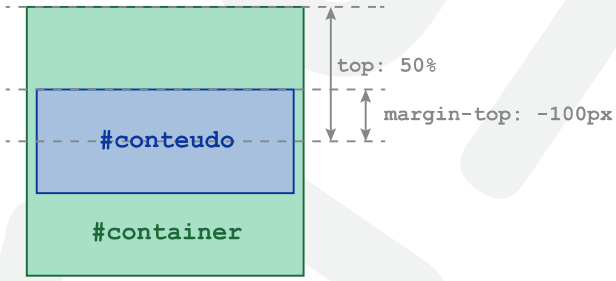
Alinhamento vertical

Utilizando as propriedades de margem e posicionamento podemos alinhar um elemento de forma centralizada na vertical.

Qualquer valor diferente de **static**

```
#container {
   height: 600px;
   position: relative;
}

#conteudo {
   height: 200px;
   position: absolute;
   top: 50%;
   margin-top: -100px;
}
```



Exercícios de Fixação

39 No projeto **css**, crie um arquivo chamado **posicionamento.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Posicionamento</title>
    <link rel="stylesheet" type="text/css" href="posicionamento.css">
  </head>
  <body>
    <img src="http.//www.k19.com.br/app/webroot/figs/main-header-logo.png">
    <p id="p1">
        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
        accumsan vehicula diam sit amet bibendum.
    </p>
    <p id="p2">
        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
        accumsan vehicula diam sit amet bibendum.
    </p>

    <p id="p3">
        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
        accumsan vehicula diam sit amet bibendum.
    </p>

    <p id="p4">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
      accumsan vehicula diam sit amet bibendum.
    </p>
    <p id="p5">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
      accumsan vehicula diam sit amet bibendum.
    </p>
```

```
    <p id="p6">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
      accumsan vehicula diam sit amet bibendum.
    </p>
    <p id="p7">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
      accumsan vehicula diam sit amet bibendum.
    </p>
    <p id="p8">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam
      accumsan vehicula diam sit amet bibendum.
    </p>

    <div id="div1">
      Posicionamento fixo
    </div>

    <div id="div2">
      Posicionamento relativo
      <div id="div3">
        Posicionamento absoluto
      </div>
    </div>
  </body>
</html>
```

Código HTML 3.21: posicionamento.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao39.zip

40 No projeto **css**, crie um arquivo CSS chamado **posicionamento.css**.

```
img {
  float. left,
}

#p3 {
  clear. left,
}

div {
  width. 200px,
  height. 100px,
  color. white,
}

#div1 {
  position. fixed,
  right. 20px,
  top. 40px,
  background-color. green,
}

#div2 {
  position. relative,
  top. 30px,
  left. 60px,
  background-color. blue,
}

#div3 {
  position. absolute,
  bottom. -40px,
  right. -100px,
  background-color. red,
  z-index. -1,
}
```

Código CSS 3.34: posicionamento.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao39.zip

41 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/posicionamento.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/posicionamento.html.

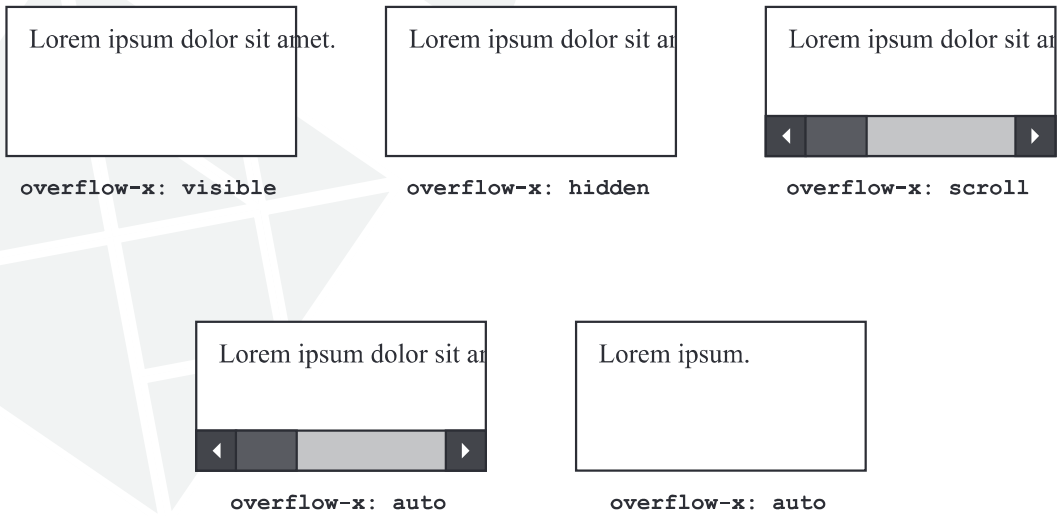
Overflow e clip

Eventualmente, o conteúdo de um elemento HTML extrapola os limites do mesmo. A linguagem CSS define algumas propriedades para determinar como essa situação deve ser tratada.

overflow-x

A propriedade **overflow-x** determina o que os navegadores devem fazer quando o conteúdo de um elemento HTML extrapola horizontalmente os limites do mesmo.

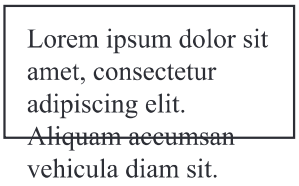
Valor	Descrição
visible	O conteúdo excedente na horizontal será exibido (padrão).
hidden	O conteúdo excedente na horizontal não será exibido.
scroll	O conteúdo excedente na horizontal não será exibido e um mecanismo rolagem horizontal será disponibilizado.
auto	O conteúdo excedente na horizontal não será exibido e um mecanismo rolagem horizontal será disponibilizado somente se necessário.



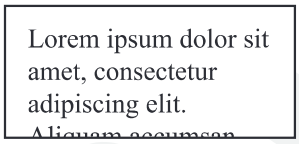
overflow-y

A propriedade **overflow-y** determina o que os navegadores devem fazer quando o conteúdo de um elemento HTML extrapola verticalmente os limites do mesmo.

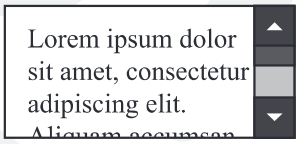
Valor	Descrição
visible	O conteúdo excedente na vertical será exibido (padrão).
hidden	O conteúdo excedente na vertical não será exibido.
scroll	O conteúdo excedente na vertical não será exibido e um mecanismo rolagem vertical será disponibilizado.
auto	O conteúdo excedente na vertical não será exibido e um mecanismo rolagem vertical será disponibilizado somente se necessário.



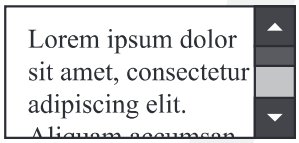
`overflow-y: visible`



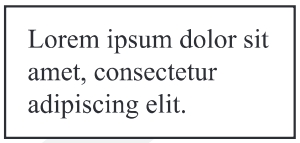
`overflow-y: hidden`



`overflow-y: scroll`



`overflow-y: auto`

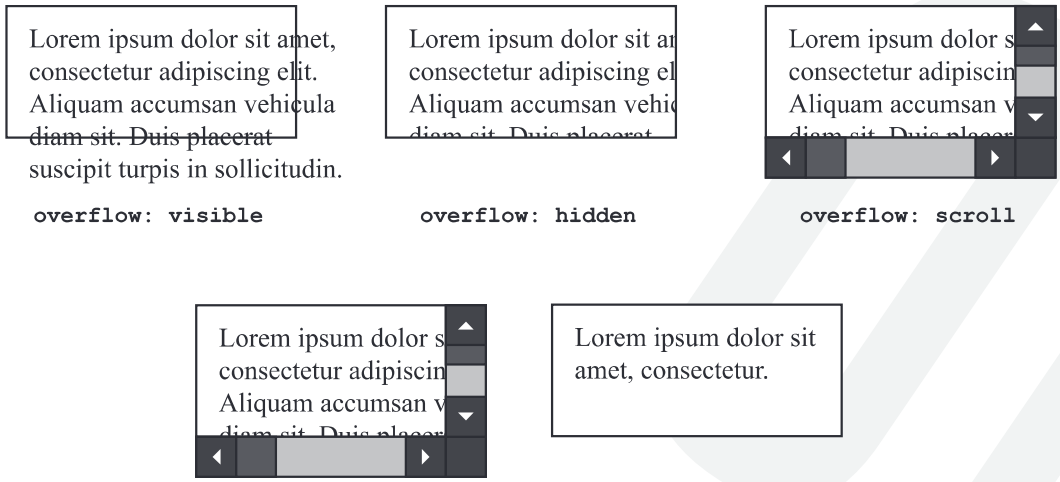


`overflow-y: auto`

overflow

A propriedade **overflow** determina o que os navegadores devem fazer quando o conteúdo de um elemento HTML extrapola os limites do mesmo.

Valor	Descrição
visible	O conteúdo excedente será exibido (padrão).
hidden	O conteúdo excedente não será exibido.
scroll	O conteúdo excedente não será exibido e um mecanismo rolagem será disponibilizado.
auto	O conteúdo excedente não será exibido e um mecanismo rolagem será disponibilizado somente se necessário.
no-display	O elemento HTML não será exibido se o seu conteúdo extrapolar os limites do seu box.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

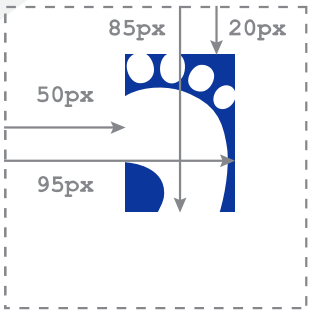


clip

Podemos determinar a região do box de um elemento HTML que deve ser exibida com a propriedade **clip**. Essa propriedade só tem efeito sobre elementos com **position: absolute**.

Valor	Descrição
auto	O box inteiro será exibido (padrão).
rect(*top, right, bottom, left*)	Retângulo que define a região que será exibida.
inherit	Assume o valor da mesma propriedade no elemento HTML pai.

clip: auto



clip: rect(20px, 95px, 85px, 50px)

Exercícios de Fixação

42 No projeto **css**, crie um arquivo chamado **overflow-clip.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Overflow e clip</title>
    <link rel="stylesheet" type="text/css" href="overflow-clip.css">
  </head>
```

```
8   <body>
9     <div id="div1">
10      <p>Lorem ipsum dolor sit amet, consectetur.</p>
11      <p>Lorem ipsum dolor sit amet, consectetur.</p>
12      <p>Lorem ipsum dolor sit amet, consectetur.</p>
13      <p>Lorem ipsum dolor sit amet, consectetur.</p>
14    </div>
15
16    <div id="div2">
17      <p>Lorem ipsum dolor sit amet, consectetur.</p>
18      <p>Lorem ipsum dolor sit amet, consectetur.</p>
19      <p>Lorem ipsum dolor sit amet, consectetur.</p>
20      <p>Lorem ipsum dolor sit amet, consectetur.</p>
21    </div>
22
23    <div id="div3">
24      <p>Lorem ipsum dolor sit amet, consectetur.</p>
25      <p>Lorem ipsum dolor sit amet, consectetur.</p>
26      <p>Lorem ipsum dolor sit amet, consectetur.</p>
27      <p>Lorem ipsum dolor sit amet, consectetur.</p>
28    </div>
29
30    <div id="div4">
31      <p>Lorem ipsum dolor sit amet, consectetur.</p>
32      <p>Lorem ipsum dolor sit amet, consectetur.</p>
33      <p>Lorem ipsum dolor sit amet, consectetur.</p>
34      <p>Lorem ipsum dolor sit amet, consectetur.</p>
35    </div>
36
37    <div id="div5">
38      Olá.
39    </div>
40
41    <img id="img1" src="http.//www.k19.com.br/figs/main-header-logo.png">
42    <img id="img2" src="http.//www.k19.com.br/figs/main-header-logo.png">
43  </body>
44 </html>
```

Código HTML 3.22: overflow-clip.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao42.zip

43 No projeto **css**, crie um arquivo CSS chamado **overflow-clip.css**.

```
1  div {
2    margin. 70px,
3    width. 200px,
4    height. 100px,
5    border. 1px solid red,
6  }
7
8  p {
9    width. 270px,
10 }
11
12 #div1 {
13   overflow. visible,
14 }
15
16 #div2 {
17   overflow. hidden,
18 }
19
20 #div3 {
21   overflow. scroll,
22 }
23
```

```
24 #div4 {
25   overflow. auto,
26 }
27
28 #div5 {
29   overflow. auto,
30 }
31
32 #img1 {
33   position. absolute,
34   top. 10px,
35   left. 400px,
36   clip. auto,
37 }
38
39 #img2 {
40   position. absolute,
41   top. 250px,
42   left. 400px,
43   clip.rect(10px, 118px, 108px, 40px),
44 }
```

Código CSS 3.35: overflow-clip.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao43.zip

44 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/overflow-clip.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/overflow-clip.html.

Transformações

No CSS3 foram introduzidas algumas funções para realizar as transformações de translado, escalonamento, distorção e rotação nos elementos de uma página HTML. Essas funções são utilizadas em conjunto com a propriedade **transform** do CSS3.

translate()

O resultado da aplicação da função **translate(m, n)** é semelhante ao resultado obtido ao mover um elemento através do atributo **position** com o valor **relative**. Ao utilizar a função **translate(m, n)** um elemento é transladado a **m** unidades de medida da esquerda e **n** unidades de medida do topo.

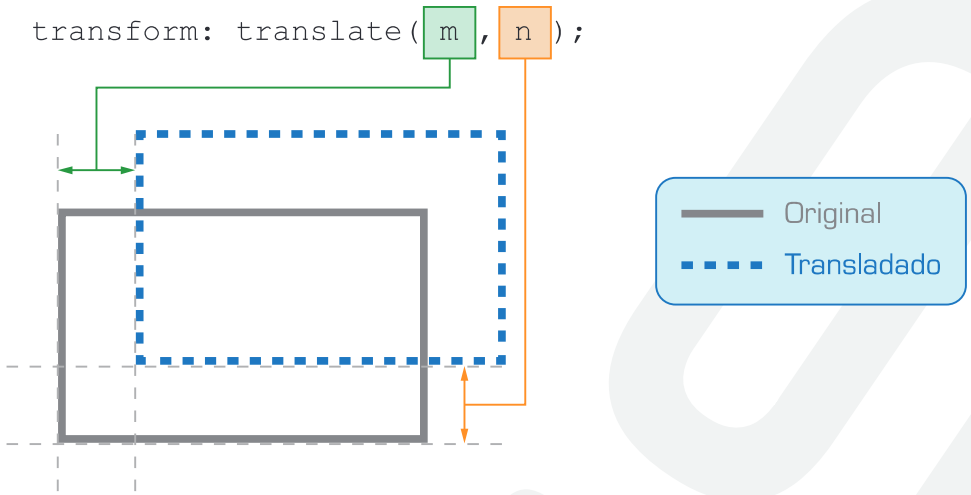


Figura 3.9: Função translate()

scale()

A função **scale(m, n)** escalonará as dimensões de um elemento. O escalonamento será aplicado sobre a largura a uma taxa definida por **m** e sobre a altura a uma taxa definida por **n**. Caso a função seja chamada com apenas um parâmetro, a mesma taxa será aplicada na altura e largura do elemento.

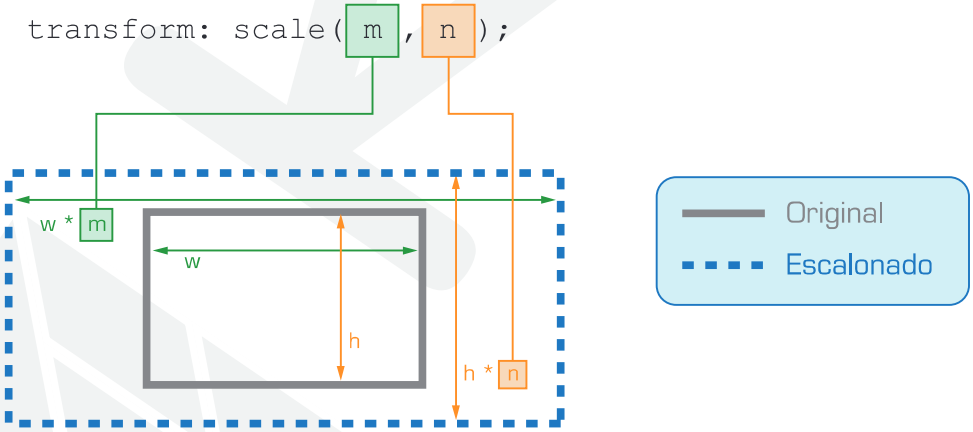


Figura 3.10: Função scale()

rotate()

A função **rotate(m)** rotacionará um elemento em torno do seu ponto de origem. O valor de **m** deve ser dado em graus, voltas ou grado.

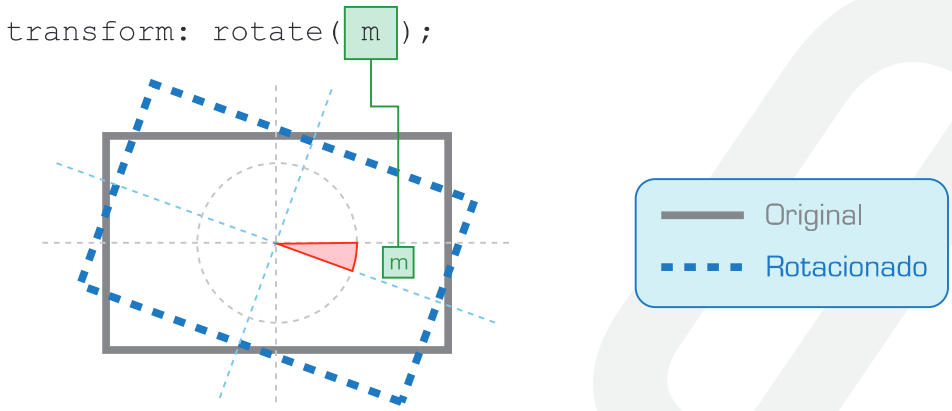


Figura 3.11: Função rotate()

skew()

A função **skew(m, n)** distorcerá um elemento em relação aos eixos x e y. **m** e **n** definem as distorções aplicadas nos eixos x e y respectivamente. Os valores de **m** e **n** devem ser dados em graus, voltas ou grado.

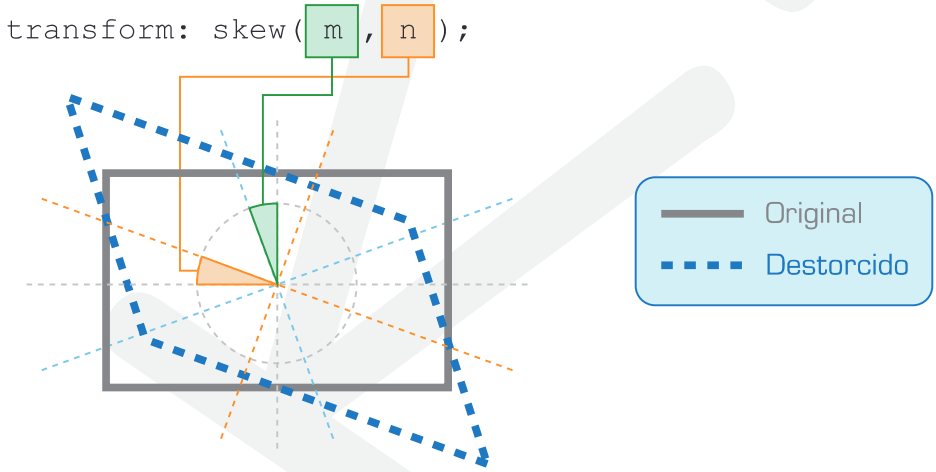


Figura 3.12: Função skew()

Exercícios de Fixação

45 No projeto **css**, crie um arquivo chamado **transformacoes.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Transformações</title>
    <link rel="stylesheet" type="text/css" href="transformacoes.css">
    </head>
    <body>
```

```
    <div id="div1">
      translate()
    </div>

    <div id="div2">
      scale()
    </div>

    <div id="div3">
      rotate()
    </div>

    <div id="div4">
      skew()
    </div>
    </body>
</html>
```

Código HTML 3.23: transformacoes.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao45.zip

46 No projeto **css**, crie um arquivo CSS chamado **transformacoes.css**.

```
div {
  margin. 70px,
  border. 1px solid black,
  background-color. yellow,
  width. 100px,
  height. 100px,
}

#div1 {
  transform. translate(50px, -20px),

  /* Webkit. Chrome, Safari */
  -webkit-transform. translate(50px, -20px),
}

#div2 {
  transform. scale(0.5, 2),

  /* Webkit. Chrome, Safari */
  -webkit-transform. scale(0.5, 2),
}

#div3 {
  transform. rotate(15deg),

  /* Webkit. Chrome, Safari */
  -webkit-transform. rotate(15deg),
}

#div4 {
  transform. skew(10deg, -45deg),

  /* Webkit. Chrome, Safari */
  -webkit-transform. skew(10deg, -45deg),
}

/* Moz. Firefox */
/* substituir "webkit" por "moz"

/* O. Opera */
/* substituir "webkit" por "o"

/* Ms. Internet Explorer */
```

```
/* substituir "webkit" por "ms" */
```

Código CSS 3.36: transformacoes.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao46.zip

47 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/transformacoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/transformacoes.html.

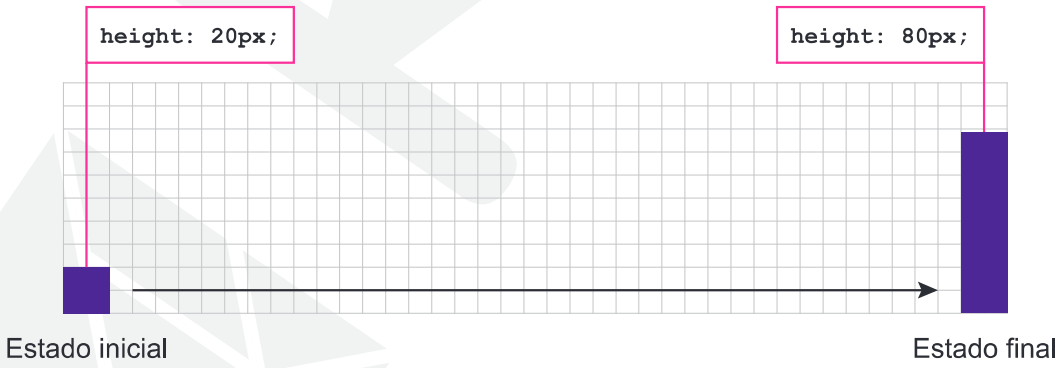
Transições

Como vimos anteriormente, as regras CSS podem ser aplicadas aos elementos HTML de acordo com o estado atual do mesmo. Por exemplo, considere as duas regras CSS a seguir.

```
div {
  width. 50px,
  height. 50px,
}

div.hover {
  width. 100px,
  height. 100px,
}
```

De acordo com as duas regras CSS acima, quando o ponteiro do mouse for colocado por cima de um **div**, a largura e a altura desse elemento HTML aumentará instantaneamente de 50px para 100px.



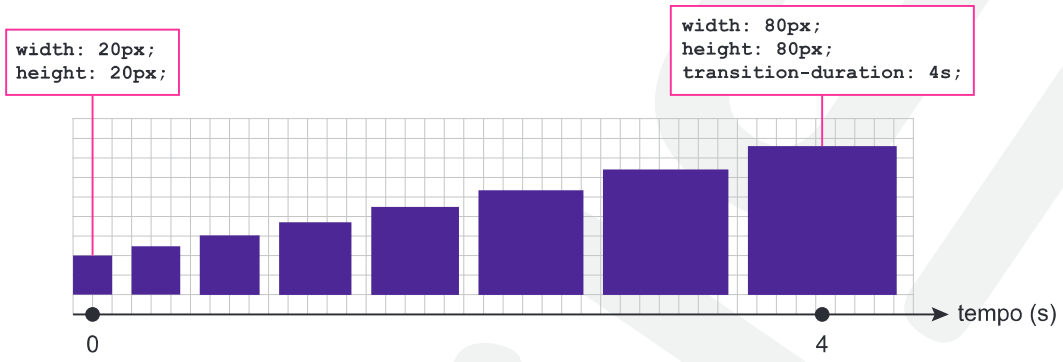
Quando a mudança visual de um elemento HTML é muito significativa, o resultado pode não ser agradável aos usuário. A linguagem CSS possui propriedades para melhorar essas transições.

transition-duration

A duração das transições pode ser controlada com a propriedade **transition-duration**.

Valor	Descrição
tempo	A duração em segundos(s) ou milissegundos(ms). Os é o valor padrão.

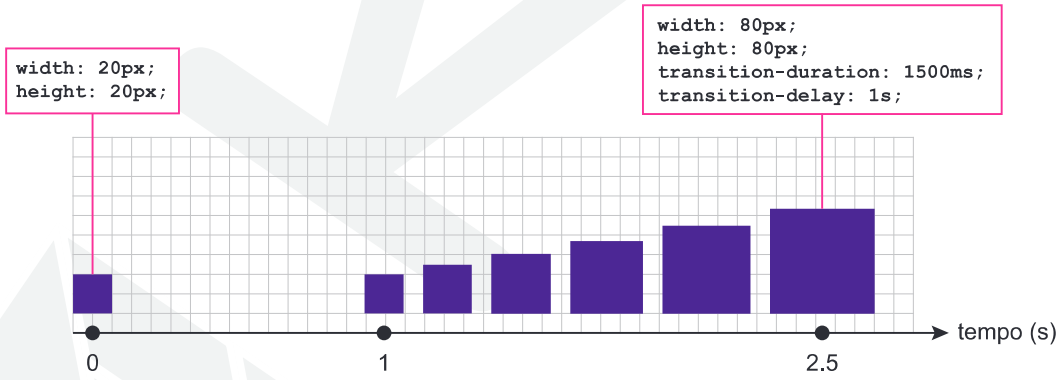
Os navegadores devem gerar, automaticamente, frames intermediários entre visual inicial e o final. Esses frames devem ser exibidos sequencialmente durante o tempo determinado com a propriedade **transition-duration**.



transition-delay

Podemos estabelecer um atraso para o início de uma transição com a propriedade **transition-delay**.

Valor	Descrição
tempo	A duração em segundos(s) ou milissegundos(ms). Os é o valor padrão.

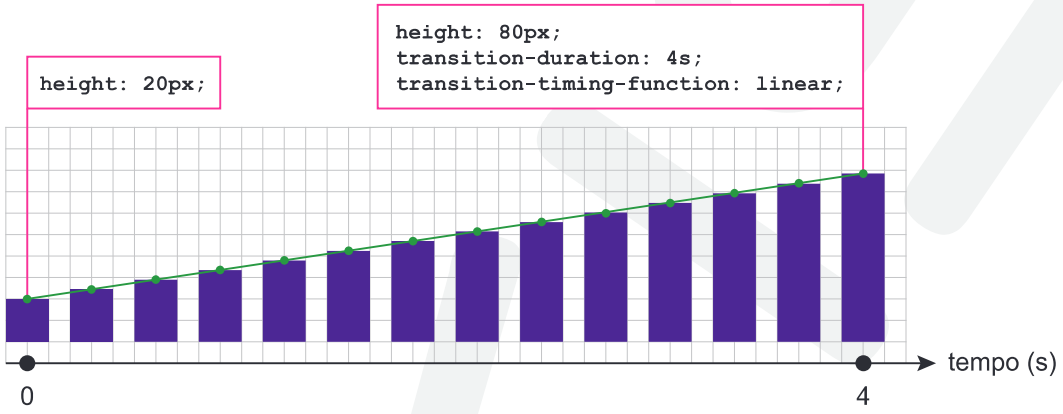


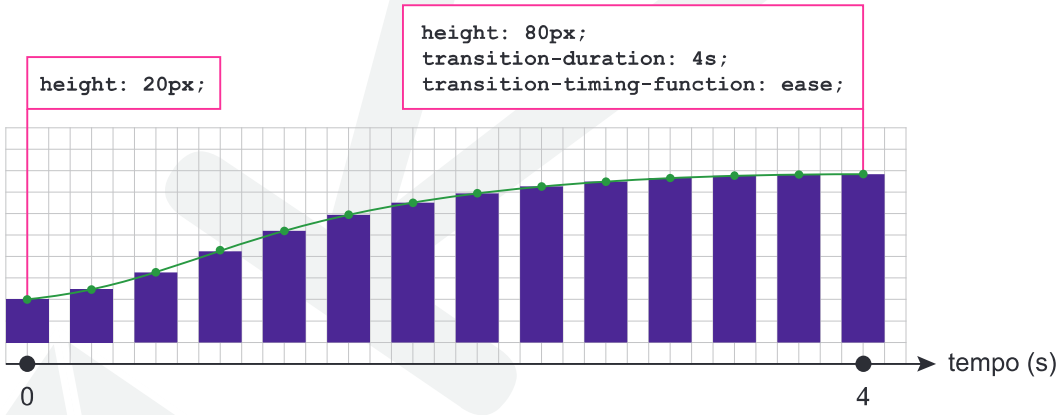
transition-timing-function

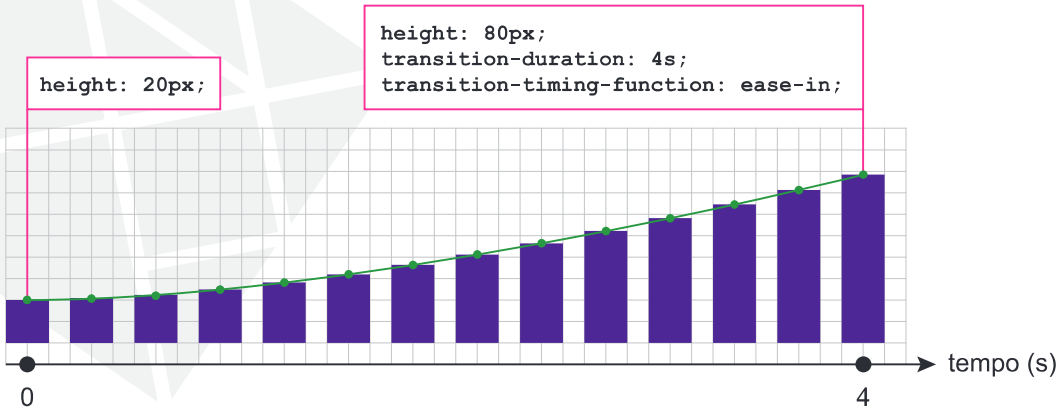
Os frames que foram uma transição podem ser exibidos com padrões diferentes. Por exemplo, uma transição pode iniciar devagar, no meio acelerar e terminar devagar. O padrão desejado pode ser determinado através da propriedade **transition-timing-function**.

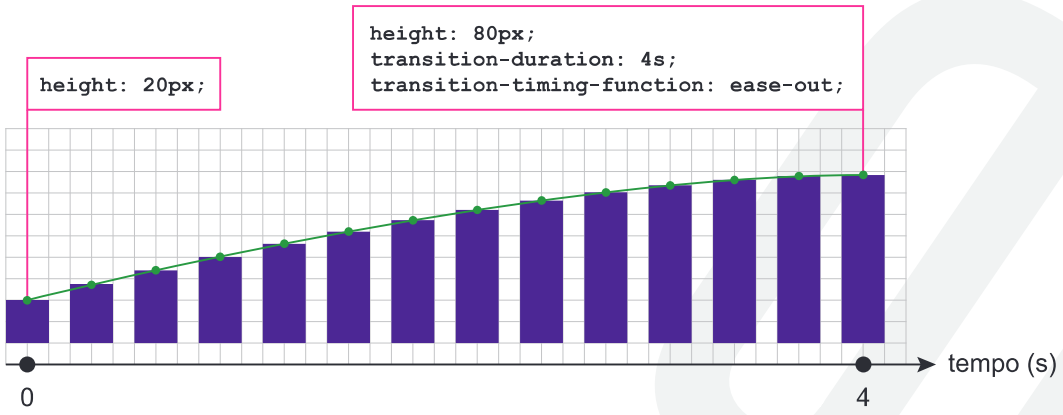
Valor	Descrição
linear	Mesma velocidade do início até o final da transição. Equivale a cubic-bezier(0, 0, 1, 1).
ease	O início da transição é lento, o meio é rápido e o final volta a ser lento (padrão). Equivale a cubic-bezier(0.25, 0.1, 0.25, 1).

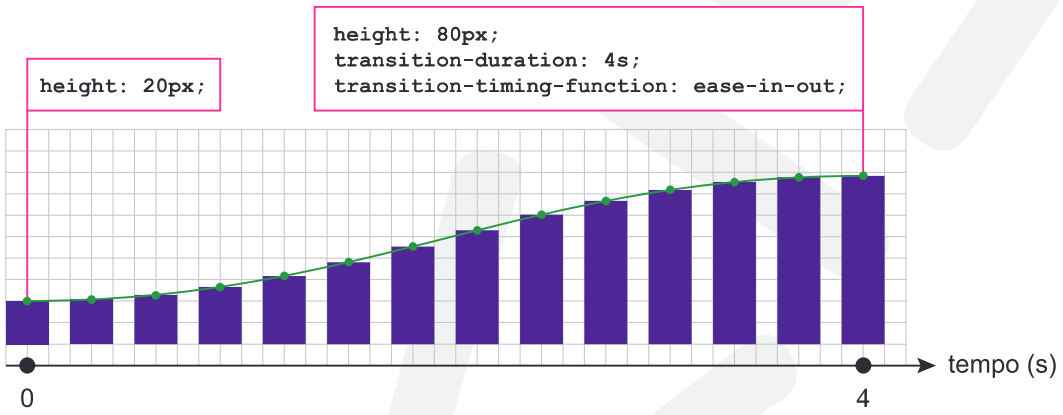
Valor	Descrição
ease-in	O início da transição é lento. O meio e o final são rápidos. Equivale a cubic-bezier(0.42, 0, 1, 1)
ease-out	O início e o meio da transição são rápidos e o final é lento. Equivale a cubic-bezier(0, 0, 0.58, 1)
ease-in-out	Semelhante ao valor **ease** com início e final mais longos. Equivale a cubic-bezier(0.42, 0, 0.58, 1).
cubic-bezier(n,n,n,n)	A transição seguirá o padrão definido pela função cúbica de bezier.







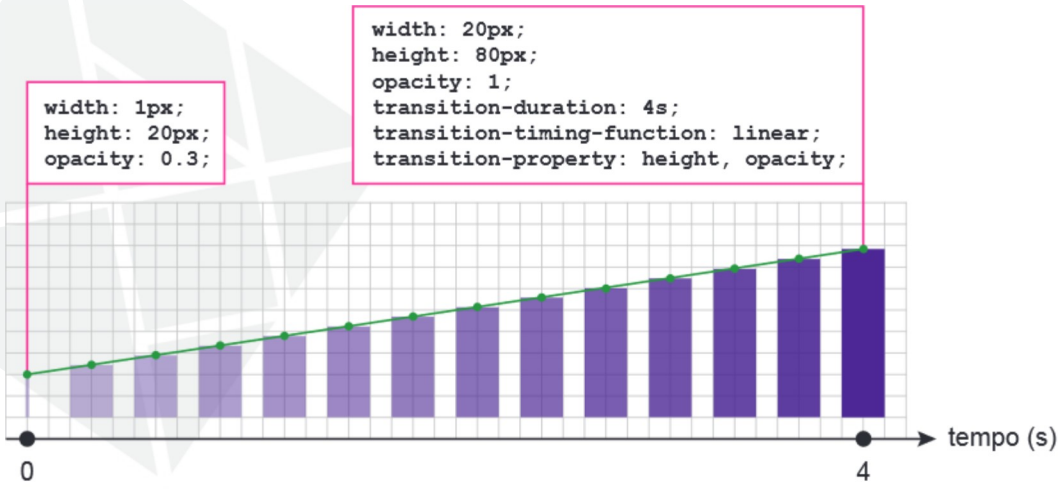




transition-property

Podemos escolher quais propriedades CSS devem ser consideradas nas transições através da propriedade **transition-property**.

Valor	Descrição
none	Nenhuma propriedade será considerada na transição.
all	Todas as propriedades serão consideradas na transição (padrão).
lista-de-propriedades	Lista das propriedades que devem ser consideradas na transição.



Exercícios de Fixação

48 No projeto **css**, crie um arquivo chamado **transicoes.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Transições</title>
    <link rel="stylesheet" type="text/css" href="transicoes.css">
  </head>
  <body>
    <div id="div1">
      K19 Treinamentos
    </div>
  </body>
</html>
```

Código HTML 3.24: transicoes.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao48.zip

49 No projeto **css**, crie um arquivo CSS chamado **transicoes.css**.

```css
#div1 {
  margin. 300px,
  padding. 10px,
  width. 200px,
  height. 200px,
  background-color. #064e83,
  color. white,
  transition. 1s ease-in-out,
}

#div1.hover {
  transform. scale(4, 4) rotate(360deg),

  /* Webkit. Chrome, Safari */
  -webkit-transform. scale(4, 4) rotate(360deg),
}

/* Moz. Firefox */
/* substituir "webkit" por "moz"

/* O. Opera */
/* substituir "webkit" por "o"

/* Ms. Internet Explorer */
/* substituir "webkit" por "ms" */
```

Código CSS 3.38: transicoes.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao49.zip

50 No **Windows**, utilize o **Chrome** para acessar o endereço:

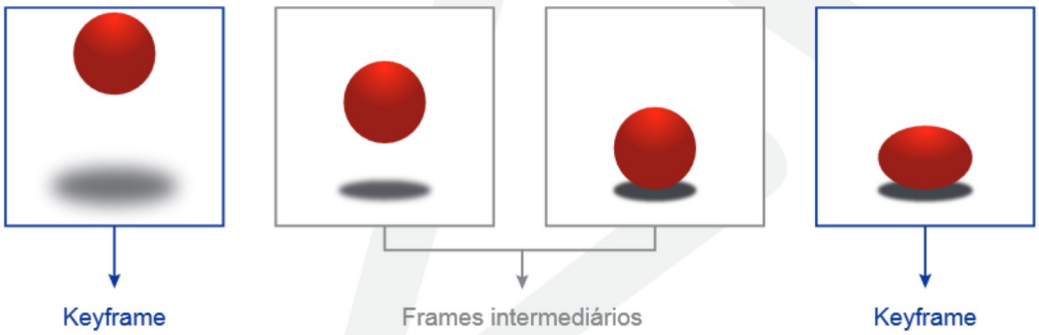
http://localhost/css/public_html/transicoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/~<USUARIO>/css/public_html/transicoes.html`.

Animações

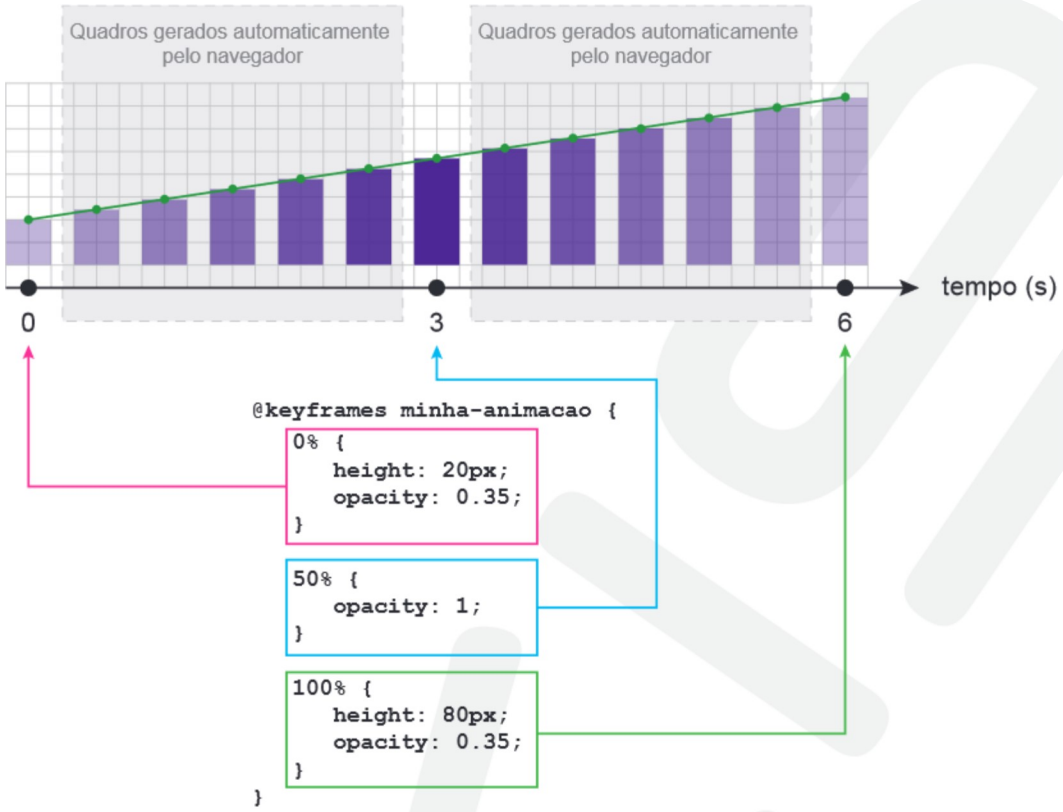
A partir da sua terceira versão, a linguagem CSS adicionou algumas propriedades para criação de animações. Essas animações são criadas através do conceito de *keyframes*. Nessa abordagem, devemos definir os quadros principais e os navegadores completam a animação inserindo os quadros intermerdiários.



Os quadros principais são definidos com a propriedade **keyframes**. A forma mais simples de utilização dessa propriedade é definir o primeiro e o último quadro da animação. Veja, no exemplo abaixo, a criação de uma animação chamada **k19-animacao**.

```
@keyframes k19-animacao {
  from {
    width. 10px,
  }
  to {
    width. 100px,
  }
}
```

Os quadros principais são definidos de acordo com o momento no qual eles serão exibidos. Esse instante é descrito como uma porcentagem do tempo total da animação.



Mais Sobre

Essencialmente, as duas animações abaixo são exatamente iguais.

```
1 @keyframes animacao1 {
2   from {
3     width. 10px,
4   }
5   to {
6     width. 70px,
7   }
8 }
```

```
1 @keyframes animacao2 {
2   0% {
3     width. 10px,
4   }
5   100% {
6     width. 70px,
7   }
8 }
```

Depois de definir as animações com a propriedade **keyframes**, devemos associá-la aos elementos HTML através da propriedade **animation-name** e determinar o tempo de duração com a propriedade **animation-duration**.

```
1 div {
```

```
2   animation-name. k19-animacao,
3   animation-duration. 3s,
4 }
5
6 @keyframes k19-animacao {
7   from {
8     width. 10px,
9   }
10  to {
11    width. 100px,
12  }
13 }
```

animation-timing-function

Podemos determinar o padrão de exibição dos quadros de uma animação com a propriedade **animation-timing-function**.

Valor	Descrição
linear	Mesma velocidade do início até o final da animação. Equivale a cubic-bezier(0, 0, 1, 1).
ease	O início da animação é lento, o meio é rápido e o final volta a ser lento (padrão). Equivale a cubic-bezier(0.25, 0.1, 0.25, 1).
ease-in	O início da animação é lento. O meio e o final são rápidos. Equivale a cubic-bezier(0.42, 0, 1, 1)
ease-out	O início e o meio da animação são rápidos e o final é lento. Equivale a cubic-bezier(0, 0, 0.58, 1)
ease-in-out	Semelhante ao valor **ease** com início e final mais longos. Equivale a cubic-bezier(0.42, 0, 0.58, 1).
cubic-bezier(n,n,n,n)	A animação seguirá o padrão definido pela função cúbica de Bézier.

animation-delay

Podemos adicionar um atraso para o início de uma animação com a propriedade **animation-delay**.

Valor	Descrição
tempo	A duração em segundos(s) ou milissegundos(ms). 0s é o valor padrão.

animation-iteration-count

A quantidade de repetições de uma animação pode ser determinada com a propriedade **animation-iteration-count**.

Valor	Descrição
x	O valor x é a quantidade de repetições da animação. 1 é o valor padrão.
infinite	A animação se repetirá indefinidamente.

animation-direction

A direção adotada na exibição de uma animação pode ser determinada através da propriedade **animation-direction**.

Valor	Descrição
normal	A animação será exibida na direção normal (padrão).
reverse	A animação será exibida na direção inversa.
alternate	A animação será exibida nas direções normal e inversa de forma alternada, iniciando com a direção normal
alternate-reverse	A animação será exibida nas direções normal e inversa de forma alternada, iniciando com a direção inversa.

animation-play-state

As animações podem ser pausadas ou reiniciadas através da propriedade **animation-play-state**.

Valor	Descrição
paused	A animação parada.
running	A animação em execução.

Exercícios de Fixação

51 No projeto **css**, crie um arquivo chamado **animacoes.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Animações</title>
    <link rel="stylesheet" type="text/css" href="animacoes.css">
  </head>
  <body>
    <img id="img1" src="http.//www.k19.com.br/figs/bolinha-vermelha.png">
    <img id="img2" src="http.//www.k19.com.br/figs/bolinha-vermelha-sombra.png">
  </body>
</html>
```

Código HTML 3.25: animacoes.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao51.zip

52 No projeto **css**, crie um arquivo CSS chamado **animacoes.css**.

```
#img1 {
  position. absolute,
  top. 0px,
  left. 100px,
  z-index. 2,
  animation. animacao1 1s linear, animacao2 0.5s linear,
        animacao3 500ms linear, animacao4 500ms linear,
  animation-delay. 2s, 3s, 3.5s, 4s,

  /* Webkit. Chrome, Safari */
  -webkit-animation. animacao1 1s linear, animacao2 0.5s linear,
            animacao3 500ms linear, animacao4 500ms linear,
  -webkit-animation-delay. 2s, 3s, 3.5s, 4s,
}

```

```
16 #img2 {
17   position. absolute,
18   top. 400px,
19   left. 110px,
20   z-index. 1,
21   transform. scale(2.5, 2.5),
22   animation. animacao5 1s linear, animacao6 0.5s linear,
23         animacao7 500ms linear, animacao8 500ms linear,
24   animation-delay. 2s, 3s, 3.5s, 4s,
25
26   /* Webkit. Chrome, Safari */
27   -webkit-transform. scale(2.5, 2.5),
28   -webkit-animation. animacao5 1s linear, animacao6 0.5s linear,
29             animacao7 500ms linear, animacao8 500ms linear,
30   -webkit-animation-delay. 2s, 3s, 3.5s, 4s,
31 }
32
33 @keyframes animacao1 {
34   to {
35     top. 310px,
36   }
37 }
38
39 @keyframes animacao2 {
40   from {
41     top. 310px,
42   }
43   to {
44     top. 335px,
45     transform. scale(1.25, 0.75),
46   }
47 }
48
49 @keyframes animacao3 {
50   from {
51     top. 335px,
52     transform. scale(1.25, 0.75),
53   }
54   to {
55     top. 285px,
56     transform. scale(0.75, 1.25),
57   }
58 }
59
60 @keyframes animacao4 {
61   from {
62     top. 285px,
63     transform. scale(0.75, 1.25),
64   }
65   to {
66     top. 310px,
67     transform. scale(1, 1),
68   }
69 }
70
71 @keyframes animacao5 {
72   to {
73     transform. scale(1.5, 1.5),
74   }
75 }
76
77 @keyframes animacao6 {
78   from {
79     transform. scale(1.5, 1.5),
80   }
81   to {
82     transform. scale(1.75, 1.75),
83   }
84 }
85
```

```
86 @keyframes animacao7 {
87   from {
88     transform. scale(1.75, 1.75),
89   }
90   to {
91     transform. scale(1.25, 1.25),
92   }
93 }
94
95 @keyframes animacao8 {
96   from {
97     transform. scale(1.25, 1.25),
98   }
99   to {
100     transform. scale(1.5, 1.5),
101   }
102 }
103
104 /* Webkit. Chrome, Safari */
105
106 @-webkit-keyframes animacao1 {
107   to {
108     top. 310px,
109   }
110 }
111
112 @-webkit-keyframes animacao2 {
113   from {
114     top. 310px,
115   }
116   to {
117     top. 335px,
118     -webkit-transform. scale(1.25, 0.75),
119   }
120 }
121
122 @-webkit-keyframes animacao3 {
123   from {
124     top. 335px,
125     -webkit-transform. scale(1.25, 0.75),
126   }
127   to {
128     top. 285px,
129     -webkit-transform. scale(0.75, 1.25),
130   }
131 }
132
133 @-webkit-keyframes animacao4 {
134   from {
135     top. 285px,
136     -webkit-transform. scale(0.75, 1.25),
137   }
138   to {
139     top. 310px,
140     -webkit-transform. scale(1, 1),
141   }
142 }
143
144 @-webkit-keyframes animacao5 {
145   to {
146     -webkit-transform. scale(1.5, 1.5),
147   }
148 }
149
150 @-webkit-keyframes animacao6 {
151   from {
152     -webkit-transform. scale(1.5, 1.5),
153   }
154   to {
155     -webkit-transform. scale(1.75, 1.75),
```

```
156   }
157 }
158
159 @-webkit-keyframes animacao7 {
160   from {
161     -webkit-transform. scale(1.75, 1.75),
162   }
163   to {
164     -webkit-transform. scale(1.25, 1.25),
165   }
166 }
167
168 @-webkit-keyframes animacao8 {
169   from {
170     -webkit-transform. scale(1.25, 1.25),
171   }
172   to {
173     -webkit-transform. scale(1.5, 1.5),
174   }
175 }
176
177 /* Moz. Firefox */
178 /* substituir "webkit" por "moz"
179
180 /* O. Opera */
181 /* substituir "webkit" por "o"
182
183 /* Ms. Internet Explorer */
184 /* substituir "webkit" por "ms" */
```

Código CSS 3.43: animacoes.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao52.zip

53 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/animacoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/animacoes.html.

Seletores

Como vimos, os seletores são utilizados para determinar quais elementos HTML devem ser afetados por uma regra CSS. A linguagem CSS oferece uma grande variedade de seletores.

Seletores básicos

Seletor	Padrão	Exemplo	Descrição
Universal	*	*	Seleciona todos os elementos
Tipo	e	p	Seleciona todos os elementos **p**
Classe	.class	.erro	Seleciona todo elemento da classe **erro**
ID	#id	#conteudo	Seleciona o elemento de id **conteudo**
Descendente	e1 e2	p a	Seleciona todo elemento **a** descendente de um elemento **p**
Filho	e1 > e2	p > a	Seleciona todo elemento **a** filho de um elemento **p**

Seletor	Padrão	Exemplo	Descrição
Irmão Adjacente	e1 + e2	h1 + p	Seleciona todo elemento **p** imediatamente precedido de um elemento **h1**
Irmão	e1 ˜ e2	h1 ˜ p	Seleciona todo elemento **p** precedido de um elemento **h1**

Seletores de atributos

Padrão	Exemplo	Descrição
[a]	[min]	Seleciona todos os elementos que tenham o atributo **min** definido
[a="v"]	[min="10"]	Seleciona todos os elementos que tenham o atributo **min** igual a 10
[a˜="v"]	[title˜="k19"]	Seleciona todos os elementos que possuam a palavra **k19** no valor do atributo **title**
[aˆ="v"]	[hrefˆ="https"]	Seleciona todos os elementos que tenham o valor do atributo **href** iniciando com **https**
[a$="v"]	[href$=".css"]	Seleciona todos os elementos que tenham o valor do atributo **href** terminando com **.css**
[a*="v"]	[title*="k19"]	Seleciona todos os elementos que possuam a string **k19** no valor do atributo **title**
[a|="v"]	[href|="https"]	Seleciona todos os elementos que tenham o atributo **href** iniciando com **https**

Pseudo-classes

Padrão	Exemplo	Descrição
:root	:root	Seleciona o elemento raiz do documento
:nth-child(n)	:nth-child(3)	Seleciona todo terceiro elemento contido em algum outro elemento
:nth-last-child(n)	:nth-last-child(3)	Seleciona todo terceiro elemento de trás para frente contido em algum outro elemento
:nth-of-type(n)	a:nth-of-type(3)	Seleciona todo elemento **a** que é o terceiro elemento contido em algum outro elemento
:nth-last-of-type(n)	a:nth-last-of-type(3)	Seleciona todo elemento **a** que é o terceiro elemento de trás para frente contido em algum outro elemento
:first-child	:first-child	Seleciona todo primeiro elemento contido em algum outro elemento
:last-child	:last-child	Seleciona todo último elemento contido em algum outro elemento
:first-of-type	a:first-of-type	Seleciona todo elemento **a** que é o primeiro elemento contido em algum outro elemento
:last-of-type	a:last-of-type	Seleciona todo elemento **a** que é o último elemento contido em algum outro elemento
:only-child	:only-child	Seleciona todo elemento que é filho único
:only-of-type	a:only-of-type	Seleciona todo elemento que é o único filho do tipo **a** de algum outro elemento

Padrão	Exemplo	Descrição
:empty	:empty	Seleciona todo elemento que não tem conteúdo
:link	a:link	Seleciona todo link não visitado
:visited	a:visited	Seleciona todo link visitado
:active	a:active	Seleciona todo link ativo
:hover	a:hover	Seleciona todo link sob o ponteiro do mouse
:focus	input:focus	Seleciona o **input** que está no foco
:target	:target	Seleciona a âncora atual
:lang(lang)	:lang(pt)	Seleciona todos os elementos com a linguagem **pt**
:enabled	input:enabled	Seleciona todos os **input**s habilitados
:disabled	input:disabled	Seleciona todos os **input**s desabilitados
:checked	input:checked	Seleciona todos os **input**s marcados
:not(s)	:not(.erro)	Seleciona todo elemento que não é da classe **erro**

Pseudo-elementos

Padrão	Exemplo	Descrição
:first-line	p:first-line	Seleciona a primeira linha de todo parágrafo
:first-letter	p:first-letter	Seleciona a primeira letra de todo parágrafo
:before	p:before	Utilizado para adicionar conteúdo antes dos parágrafos
:after	p:after	Utilizado para adicionar conteúdo depois dos parágrafos

Prioridade de seletores

Eventualmente, um elemento HTML é afetado por duas ou mais regras CSS. Além disso, uma determinada propriedade CSS pode ser definida com valores diferentes em duas ou mais dessas regras. Nesse caso, essa propriedade terá o valor definido na regra CSS de maior prioridade. Essas prioridades são calculadas de acordo com os seletores utilizados nas regras CSS e na ordem em que elas foram definidas.

Considere o código HTML e o código CSS a seguir.

```
1 <p id="p1" class="class1">
2   Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3 </p>
```

```
1 p {
2   color. red,
3   font-size. 14px,
4 }
5
6 p {
7   color. blue,
8 }
```

Perceba que todo elemento **p** é afetado pelas duas regras CSS acima. Nessas duas regras, a propriedade **color** foi definida com valores diferentes. Os seletores dessas duas regras possuem a mesma importância. Nesse caso, a última regra tem maior prioridade. Portanto, o texto do parágrafo será exibido em azul.

A propriedade **font-size** não entrou em conflito, pois foi definida em apenas uma das regras CSS. Portanto, o tamanho da fonte do texto do parágrafo será 14px.

Agora, suponha que o código CSS seja o seguinte:

```
.class1 {
  color. red,
  font-size. 14px,
}

p {
  color. blue,
}
```

Observe que o parágrafo do exemplo é afetado pelas duas regras CSS acima. Nesse caso, qual será a cor do texto do parágrafo? A resposta correta é vermelho. Isso ocorre, pois os seletores de classe possuem importância maior que os seletores de tipo.

Conforme a complexidade dos seletores utilizados em nossas regras CSS aumenta, a dificuldade em determinar qual deles possui maior importância também aumenta. Por isso, devemos seguir o algoritmo definido na especificação da linguagem CSS para determinar a importância dos seletores. Nesse algoritmo, os seletores obterão uma pontuação em quatro critérios diferentes.

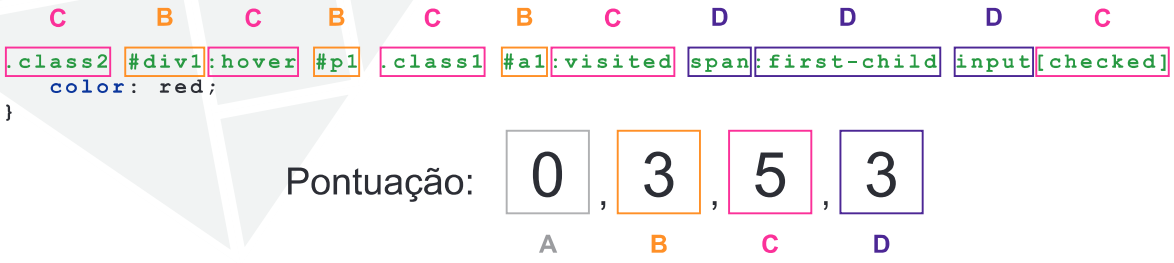
A: Caso as propriedades sejam definidas através do atributo **style**, a pontuação nesse critério será 1. Caso contrário será 0.

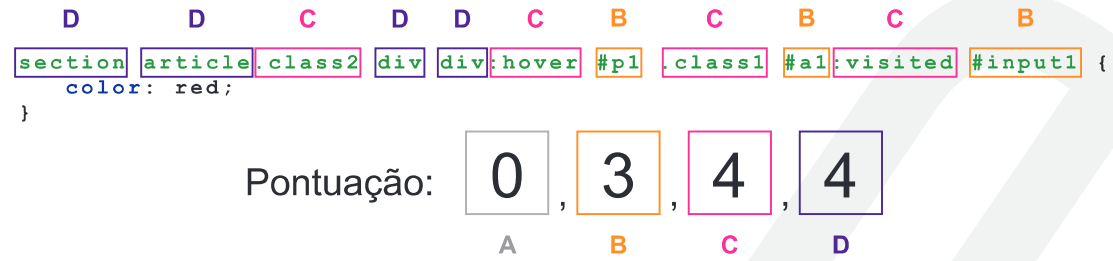
B: A pontuação nesse critério será a quantidade de seletores de ID que formam o seletor da regra CSS.

C: A pontuação nesse critério será a soma das quantidades de seletores de classe, de atributos e de pseudo-classes que formam o seletor da regra CSS.

D: A pontuação nesse critério será a soma das quantidades de seletores de tipo e de pseudo-elementos que formam o seletor da regra CSS.

O critério A possui prioridade sobre o critério B, que por sua vez possui prioridade sobre o critério C, que por sua vez possui prioridade sobre o critério D. Veja nas imagens abaixo, um exemplo de como calcular a pontuação dos seletores.





No critério A, as duas regras obtiveram a mesma pontuação. Como ocorreu um empate, devemos analisar o critério seguinte. No critério B, houve outro empate. Portanto, devemos analisar o próximo critério. No critério C, a primeira regra possui uma pontuação maior. Dessa forma, o seletor da primeira regra possui maior importância fazendo com que essa regra tenha maior prioridade.

Exercícios de Fixação

54 No projeto **css**, crie um arquivo chamado **seletores-basicos.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Seletores - 1</title>
    <link rel="stylesheet" type="text/css" href="seletores-basicos.css">
  </head>
  <body>
    <div id="conteudo">
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>

      <ul>
        <li>Item 1</li>
        <li class="cancelado">Item 2</li>
        <li>
          Item 3
          <ul>
            <li>Subitem 1</li>
            <li>Subitem 2</li>
          </ul>
        </li>
        <li class="cancelado">Item 4</li>
      </ul>

      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    </div>
  </body>
</html>
```

Código HTML 3.27: seletores-basicos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao54.zip

55 No projeto **css**, crie um arquivo CSS chamado **seletores-basicos.css**.

```css
* {
```

```
  margin. 0px,
  padding. 0px,
}

body {
  background-color. #ddd,
}

#conteudo {
  width. 800px,
  margin. 0 auto,
  background-color. white,
  border. 1px solid #333,
  padding. 10px,
}

.cancelado {
  text-decoration. line-through,
}

ul {
  padding. 0 0 0 40px,
}

ul li {
  color. blue,
}

#conteudo > ul > li {
  margin. 0 0 20px 0,
}

ul + p {
  color. red,
}

ul ~ p {
  font-style. italic,
}
```

Código CSS 3.46: seletores-basicos.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao55.zip

56 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/seletores-basicos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/seletores-basicos.html.

57 No projeto **css**, crie um arquivo chamado **seletores-de-atributos.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Seletores - 2</title>
    <link rel="stylesheet" type="text/css" href="seletores-de-atributos.css">
  </head>
  <body>
    <a href="#a-link">Link 1</a>
    <a href="#b-link" title="Link para a página 2">Link 2</a>
```

```
    <a href="#link-c" title="Página 3">Link 3</a>
    <a href="#link-d" title="Link para a página 4">Link 4</a>
    <a href="#link-e" title="Link para a página da K19 Treinamentos">Link 5</a>
    <a href="#links-f" title="Link para a página 6">Link 6</a>
    <a href="#link" title="Link para a página 7">Link 7</a>

    <hr>

    <img src="http.//www.k19.com.br/figs/main-header-logo.png"
      title="Logo da K19">
    <img src="http.//www.k19.com.br/figs/k01-logo-large.png">
    <img src="http.//www.k19.com.br/figs/k02-logo-large.png">
  </body>
</html>
```

Código HTML 3.28: seletores-de-atributos.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao57.zip

58 No projeto **css**, crie um arquivo CSS chamado **seletores-de-atributos.css**.

```
[title] {
  border. 4px solid red,
}

[title='Página 3'] {
  color. green,
}

[title~='K19'] {
  padding. 10px,
  background. yellow,
}

[title^='Link'] {
  margin-right. 30px,
}

[href$='link'] {
  outline. 4px solid blue,
}

[src*='large'] {
  border. 4px solid green,
}

[href|='#link'] {
  font-size. 30px,
}
```

Código CSS 3.47: seletores-de-atributos.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao58.zip

59 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/seletores-de-atributos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/seletores-de-atributos.html.

60 No projeto **css**, crie um arquivo chamado **seletores-de-pseudo-classes.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Seletores - 3</title>
    <link rel="stylesheet" type="text/css" href="seletores-de-pseudo-classes.css">
  </head>
  <body>
    <h1 id="seletores-3-ancora">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </h1>

    <ul id="lista1">
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
    </ul>

    <ul id="lista2">
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
      <li>Item 4</li>
      <li>Item 5</li>
    </ul>

    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>

    <div>
      <h1>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h1>
      <h2>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h2>
      <h2>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h2>
    </div>

    <div>
      <h1>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h1>
    </div>

    <div>
      <h2>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h2>
      <h3>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h3>
      <h3>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h3>
    </div>

    <div></div>

    <div>
      <a href="#seletores-3-ancora">Link 1</a>
      <a href="http.//www.k19.com.br">Link 2</a>
    </div>

    <div>
      <input type="text">
      <input type="text" disabled>
      <input type="text">
    </div>

    <div>
      <input type="checkbox">
      <input type="checkbox" checked>
      <input type="checkbox">
    </div>

    <ul lang="es">
```

```
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
    </ul>

    <ul lang="en">
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
    </ul>

    <ol lang="en">
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
    </ol>
  </body>
</html>
```

Código HTML 3.29: seletores-de-pseudo-classes.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao60.zip

61 No projeto **css**, crie um arquivo CSS chamado **seletores-de-pseudo-classes.css**.

```
.root {
    font-size. 30px,
}

#lista1 li.first-child {
    color. red,
}

#lista1 li.last-child {
    color. blue,
}

#lista2 li.nth-child(2) {
    color. magenta,
}

#lista2 li.nth-last-child(2) {
    color. orange,
}

p.first-of-type {
    color. yellow,
}

p.last-of-type {
    color. aqua,
}

p.nth-of-type(2) {
    color. firebrick,
}

p.nth-last-of-type(2) {
    color. lawngreen,
}

.only-child {
    border. 4px solid red,
}

h2.only-of-type {
    color. plum,
```

```
43 }
44
45 .empty {
46     height. 20px,
47     border. 4px dashed black,
48 }
49
50 a.link {
51     color. springgreen,
52     font-weight. bold,
53 }
54
55 a.visited {
56     color. #ccc,
57 }
58
59 a.active {
60     color. orange,
61 }
62
63 a.hover {
64     color. fuchsia,
65 }
66
67 input.focus {
68     background-color. red,
69     color. white,
70 }
71
72 .target {
73     border. 4px solid magenta,
74     color. purple,
75 }
76
77 .lang(en) {
78     color. red,
79 }
80
81 .enabled {
82     outline. 4px solid red,
83 }
84
85 .disabled {
86     outline. 4px solid blue,
87 }
88
89 .checked {
90     outline. 4px solid green,
91 }
92
93 .not(ul) > li {
94     padding-left. 100px,
95 }
```

Código CSS 3.48: seletores-de-pseudo-classes.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao61.zip

62 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/seletores-de-pseudo-classes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/seletores-de-pseudo-classes.html.

63 No projeto **css**, crie um arquivo chamado **seletores-de-pseudo-elementos.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Seletores - 4</title>
    <link rel="stylesheet" type="text/css" href="seletores-de-pseudo-elementos.css">
  </head>
  <body>
    <h1>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</h1>

    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.30: seletores-de-pseudo-elementos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao63.zip

64 No projeto **css**, crie um arquivo CSS chamado **seletores-de-pseudo-elementos.css**.

```css
p.first-line {
  background-color. yellow,
}

p.first-letter {
  font-size. 50px,
}

h1.before {
  content. '~ ',
  color. red,
}

h1.after {
  content. ' ~',
  color. red,
}
```

Código CSS 3.49: seletores-de-pseudo-elementos.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao64.zip

65 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/seletores-de-pseudo-elementos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/seletores-de-pseudo-elementos.html.

Media Queries

Media Types

No momento em que definimos as regras CSS dos nossos documentos, também podemos determinar para quais tipos de mídia essas regras serão válidas, ou seja, podemos decidir quais regras serão aplicadas quando o documento for apresentado em uma tela, no papel ou narrado por um sintetizador de voz, por exemplo. Podemos informar o tipo de mídia de duas formas: com o atributo **media** do elemento **link** ou com a regra **media** do CSS.

```
1 <link rel="stylesheet" type="text/css" href="aquivo.css" media="screen">
```

```
1 @media screen {
2   body {
3     background-color. black,
4   }
5
6   ...
7 }
```

A linguagem CSS define os seguintes tipos de mídia.

Media Type	Descrição
all	Todos os tipos de mídia.
braille	Dispositivos em braille com resposta a toques.
embossed	Impressoras em braille.
handheld	Dispositivos portáteis (exceto smartphones mais modernos).
print	Impressoras convencionais.
projection	Projetores.
screen	Tela de computadores, smartphones ou algum dispositivo do gênero.
speech	Sintetizadores de voz.
tty	Dispositivos de grades com caracteres de tamanho fixo (teletypes, terminal ou dispositivos com limitações de exibição).
tv	Televisores.

Podemos definir as mesmas regras CSS para diversos tipos de mídia. Para isso devemos separar esses tipos por vírgula.

```
1 <link rel="stylesheet" type="text/css" href="aquivo.css" media="screen, print">
```

```
1 @media screen, print {
2   body {
3     background-color. black,
4   }
5
6   ...
7 }
```

Media Groups

Algumas propriedades não são aplicáveis a determinados tipos de mídia. Por exemplo, a propriedade **width** não pode ser aplicada à mídia **speech**. Por outro lado, algumas propriedades podem ser aplicadas à diversas mídias. Por exemplo, a propriedade **position** pode ser aplicada às mídias **screen**, **projection**, **handheld**, **print**, **tty** e **tv**.

Para identificar, mais facilmente, quais propriedades podem ser aplicadas a um determinado tipo de mídia, a especificação da linguagem CSS divide em grupos os diversos tipos de mídia.

Confira na tabela a seguir a relação entre os tipos de mídia e os grupos de mídia:

Media Type	Media Groups			
	continuous e **paged**	**visual**, **audio**, **speech** e **tactile**	**grid** e **bitmap**	**interactive** e **static**
braille	continuous	tactile	grid	ambos
embossed	paged	tactile	grid	static
handheld	ambos	visual, audio, speech	ambos	ambos
print	paged	visual	bitmap	static
projection	paged	visual	bitmap	interactive
screen	continuous	visual, audio	bitmap	ambos
speech	continuous	speech	nenhum	ambos
tty	continuous	visual	grid	ambos
tv	ambos	visual, audio	bitmap	ambos

Propriedades específicas

Os tipos de mídia do CSS permitem definir diferentes regras para cada tipo de mídia. Entretanto, eventualmente é necessário definir regras CSS de acordo com características mais específicas dos dispositivo de saída. Para isso, devemos utilizar as chamadas **media queries**. Veja o seguinte exemplo.

```
1 @media screen and (min-width. 500px) {
2   ...
3 }
```

No exemplo acima, as regras CSS serão aplicadas caso o tipo de mídia utilizado seja **screen** e a área de renderização do documento (viewport) tenha, no mínimo, 500 pixels de largura. Como vimos as **media queries** também podem ser definidas através do atributo **media** do elemento **link**.

```
1 <link
2   rel="stylesheet"
3   type="text/css"
4   href="arquivo.css"
5   media="screen and (min-width. 500px)"/>
```

No exemplo abaixo, o código dentro da regra **media** será aplicado quando a largura do viewport for pelo menos 500 pixels ou quando a orientação for **portrait**.

```
1 @media (min-width. 500px), (orientation. portrait) {
2   ...
3 }
```

Eventualmente, uma media query pode não fazer sentido. Considere o seguinte exemplo.

```
1 @media speech and (min-width. 500px) {
2   ...
3 }
```

No exemplo acima, a media query é contraditória, pois o tipo de mídia **speech** não é compatível com a propriedade **min-width**. Dessa forma, o código CSS definido dentro da regra **media** nunca será processado.

Veja a seguir algumas propriedades que podemos utilizar nas media queries

Propriedade	Descrição	Valor	Media Type	min e max?
width	Determina qual deve ser a largura do viewport no dispositivo de saída.	Medidas	visual e tactile	sim
height	Determina qual deve ser a altura do viewport no dispositivo de saída.	Medidas	visual e tactile	sim
device-width	Determina qual deve ser a largura da tela do dispositivo de saída.	Medidas	visual e tactile	sim
device-height	Determina qual deve ser a altura da tela do dispositivo de saída.	Medidas	visual e tactile	sim
orientation	Verifica se o viewport está na orientação portrait (retrato) ou landscape (paisagem).	portrait ou landscape	bitmap	não
aspect-ratio	Determina a razão entre as propriedades width e height.	inteiro/inteiro	bitmap	sim
color-index	Determina o número de cores do dispositivo de saída.	inteiro	visual	sim
resolution	Determina a resolução do dispositivo de saída.	dpi	bitmap	sim

Exercícios de Fixação

66 No projeto **css**, crie um arquivo chamado **media-types.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Media Types</title>
    <link rel="stylesheet" type="text/css" href="media-types.css">
  </head>
  <body>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
  </body>
</html>
```

Código HTML 3.34: media-types.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao66.zip

67 No projeto **css**, crie um arquivo CSS chamado **media-types.css**.

```
@media all {
  p {
    font-size. 50px,
```

```
  }
}

@media screen {
  p {
    color. red,
  }
}

@media print {
  p {
    color. blue,
  }
}

@media handheld, screen and (max-device-width.480px) {
  p {
    color. green,
  }
}
```

Código CSS 3.55: media-types.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-fixacao67.zip

68 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/media-types.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

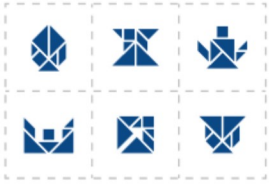
http.//localhost/~<USUARIO>/css/public_html/media-types.html.

Sprites

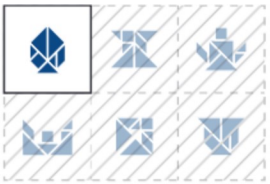
Normalmente, uma página web é formada por diversas imagens pequenas ou médias. Se essas imagens forem adicionadas com o elemento **img**, o navegador realizará uma requisição HTTP para cada imagem. Consequentemente, o tempo de carregamento da página será alto devido a quantidade de requisições e ao *overhead* dos pacotes HTTP.

Analogia
Adicionar individualmente imagens pequenas ou médias em uma página web é como ter quer comprar diversos itens pequenos ou médios no supermercado e fazer uma viagem para trazer cada um deles.

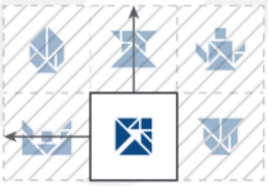
Uma técnica amplamente utilizada para melhorar o tempo de carregamento das páginas web é denominada **sprite**. Essa técnica consiste em agrupar todas as imagens pequenas e médias em uma única imagem grande e depois "recortá-la" com as propriedades CSS.



cursos-icones.png



background-image: url('cursos-icones.png');
background-position: 0px 0px;



background-image: url('cursos-icones.png');
background-position: -100px -100px;

Exercícios de Fixação

69 No projeto **css**, crie um arquivo chamado **sprites.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Sprites</title>
    <link rel="stylesheet" type="text/css" href="sprites.css">
  </head>
  <body>
    <ul>
      <li>K11 - Orientação a Objetos em Java</li>
      <li class="k12">K12 - Desenvolvimento Web com JSF2 e JPA2</li>
      <li class="k21">K21 - Persistência com JPA2 e Hibernate</li>
      <li class="k22">K22 - Desenvolvimento Web Avançado com JSF2, EJB3.1 e CDI</li>
      <li class="k23">K23 - Integração de Sistemas com Webservices, JMS e EJB</li>
      <li class="k20">K20 - Formação Desenvolvedor Java Avançado</li>
    </ul>
  </body>
</html>
```

Código HTML 3.35: sprites.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao69.zip

70 No projeto **css**, crie um arquivo CSS chamado **sprites.css**.

```
li.before {
  content. "",
  display. inline-block,
  width. 44px,
  height. 44px,
  vertical-align. middle,
  background-image. url("http.//www.k19.com.br/figs/k19-logos-sprite.png"),
  background-repeat. no-repeat,
```

```
}

li.k12.before {
  background-position. -44px 0px,
}

li.k21.before {
  background-position. -88px 0px,
}

li.k22.before {
  background-position. 0px -44px,
}

li.k23.before {
  background-position. -44px -44px,
}

li.k20.before {
  background-position. -88px -44px,
}
```

Código CSS 3.56: media-types.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-fixacao70.zip

71 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/sprites.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/sprites.html.

Gradientes (Conteúdo Extra)

Até a versão 2 do CSS, não era possível definir um gradiente de cores como background de um elemento a não ser através de imagens estáticas (jpg, gif ou png). A partir da versão 3, podemos definir um gradiente diretamente no código CSS através das funções **linear-gradient**, **repeating-linear-gradient**, **radial-gradient** e **repeating-radial-gradient**.

De acordo com a especificação CSS, os gradientes são semelhantes às imagens. Eles podem ser definidos em qualquer propriedade que aceite uma imagem como valor.

Mais Sobre
Até o momento do fechamento da versão atual desta apostila, o navegador Firefox só suporta as funções de gradiente nas propriedades **background** ou **background-image**.

Gradiente linear

Para definir um gradiente linear, é necessário utilizar a função **linear-gradient**.

```
                    Definição da                    Lista de
                  linha do gradiente              paradas de cores
linear-gradient( [ <angulo> | to <lado-ou-canto>,]? <parada-de-cor> [, <parada-de-cor>]+ )
```

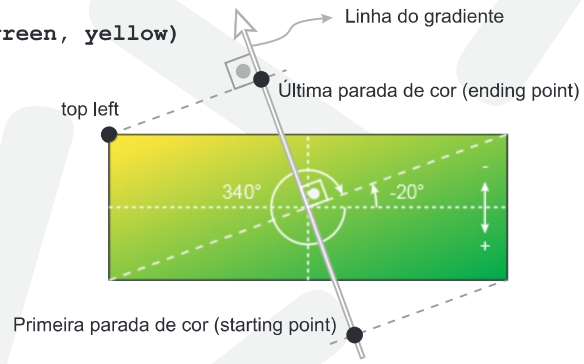
<angulo>
Representa o ângulo de inclinação do gradiente.

<parada-de-cor>
Representa uma cor e a posição em que ela deve aparecer sobre a linha do gradiente.

<lado-ou-canto>
Representa a região do box do gradiente na qual a última <parada-de-cor> será posicionada. É formado pelos pares de lados left | right e/ou top | bottom, não importando a ordem em que aparecem. A primeira <parada-de-cor> será posicionada na região simetricamente oposta a da última <parada-de-cor>.

No exemplo abaixo, podemos entender melhor cada um dos parâmetros:

```
linear-gradient(to top left, green, yellow)
```



Veja mais alguns exemplos:

```
linear-gradient(to top right, green 50%, yellow)
```

```
linear-gradient(135deg, green, white, blue)
```

Repare que quando a primeira parada de cor começa em uma posição diferente de 0%, o gradiente é preenchido com a cor da primeira parada de cor do 0% até essa posição.

Gradiente linear com repetição

Para definir um gradiente linear com repetição, devemos utilizar a função **repeating-linear-gradient** que recebe exatamente os mesmos parâmetros da função **linear-gradient**.

A princípio o gradiente linear com repetição funcionará da mesma forma que o gradiente linear. Seu comportamento só será alterado nas seguintes situações:

- Se a primeira parada de cor não começar em 0%.

- Se a última parada de cor não terminar em 100%.

Se uma ou as duas situações ocorrerem, o gradiente se repetirá para completar as regiões da linha do gradiente sem cores definidas.

Veja alguns exemplos:

```
linear-gradient(to top left, green, yellow)
```

```
repeating-linear-gradient(to top left, green, yellow)
```

```
linear-gradient(to top left, green 50%, yellow)
```

```
repeating-linear-gradient(to top left, green 50%, yellow)
```

```
linear-gradient(to top left, green, yellow 50%)
```

```
repeating-linear-gradient(to top left, green, yellow 50%)
```

Gradiente radial

Para definir um gradiente radial, devemos utilizar a função **radial-gradient**.

```
                    Definição da forma, dimensão e posicão do gradiente
radial-gradient( [
                  [circle  || <tamanho>]                      [at <posição>]?, |
                  [ellipse || [<tamanho> | <porcentagem>]{2}] [at <posição>]?, |
                  [[circle | ellipse] || <alcance> ]          [at <posição>]?, |
                                                               at <posição>,
                 ]?
                 <parada-de-cor>, [, <parada-de-cor>]+)
                                Lista de paradas de cores
```

<tamanho> | <porcentagem>
Representa o tamanho do raio da forma do gradiente. No caso da elipse, o tamanho pode ser dado através da porcentagem em relação ao tamanho do box do gradiente.

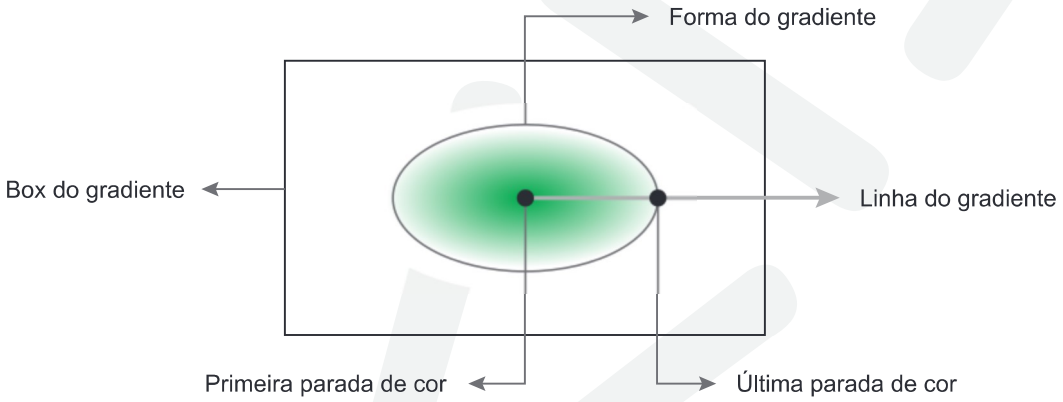
<parada-de-cor>
Representa uma cor e a posição em que ela deve aparecer sobre a linha do gradiente.

<posição>
Representa a posição do box do gradiente na qual a primeira <parada-de-cor> será posicionada. Pode ser informada através dos pares de lados left | right e/ou top | bottom, não importando a ordem em que aparecem. Também pode receber o valor center, que é o padrão. A posição também pode ser informada através de coordenadas em relação ao canto superior esquerdo do box do gradiente.

<alcance>
Representa a forma como o gradiente será distribuído dentro do box do gradiente.

No exemplo abaixo, podemos entender melhor cada um dos parâmetros:

```
radial-gradient(50% 50% at center, green, white)
```



Gradiente radial com repetição

Para definir um gradiente radial com repetição, devemos utilizar a função **repeating-radial-gradient** que recebe exatamente os mesmos parâmetros da função **radial-gradient**.

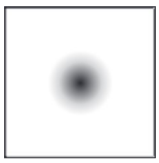
A princípio o gradiente radial com repetição funcionará da mesma forma que o gradiente radial. Seu comportamento só será alterado nas seguintes situações:

- Se a primeira parada de cor não começar em 0%.
- Se a última parada de cor não terminar em 100%.

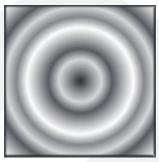
Se uma ou as duas situações ocorrerem, o gradiente se repetirá para completar as regiões da linha do gradiente sem cores definidas.

Confira o exemplo:

```
radial-gradient(black 0%, white 25%)
```



```
repeating-radial-gradient(black 0%, white 25%)
```



Herança (Conteúdo Extra)

Herança em CSS é o mecanismo pelo qual os valores de algumas propriedades são passadas de um elemento pai para um elemento filho. Os valores de algumas propriedades são herdados automaticamente enquanto outros não. Além disso, existem propriedades que não podem assumir o valor proveniente da mesma propriedade do elemento pai.

Uma propriedade herda o valor do elemento pai quando atribuímos o valor **inherit** à ela.

Considere os seguintes códigos:

```
<div>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
</div>
```

```
div {
  color. red,
}
```

No exemplo acima, a propriedade color foi definida apenas para o elemento **div**. No elemento **p**, essa propriedade assumirá o valor padrão que é **inherit**, ou seja, será herdado automaticamente do elemento pai. Como consequência a cor do texto do parágrafo será vermelha.

Agora, considere este outro exemplo:

```
<div>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
</div>
```

```
div {
  border. 1px solid red,
}
```

O elemento **div** terá borda de 1 pixel com cor vermelha. Diferentemente do exemplo anterior, o parágrafo não herdará o valor para a propriedade **border**, pois ela não possui o valor **inherit** como padrão. Se quiséssemos que o elemento **p** herdasse o valor da propriedade border, deveríamos definir as seguintes regras CSS:

```
div {
  border. 1px solid red,
}

p {
  border. inherit,
}
```

box-sizing (Conteúdo Extra)

De acordo com o box model, devemos levar em consideração as margens internas, bordas e as propriedades **width** e **height** de um elemento para determinarmos as suas dimensões. No começo, isso pode não parecer muito natural, pois tendemos a imaginar a largura e altura final do elemento como sendo as medidas atribuídas somente às propriedades **width** e **height**, respectivamente. Isso acaba gerando uma grande confusão, principalmente para quem está começando a aprender CSS.

Através da propriedade **box-sizing** podemos decidir como as propriedades **width** e **height** se comportarão. Confira na tabela abaixo os possíveis valores da propriedade:

content-box

Comportamento padrão. As propriedades **width** e **height** definem as dimensões da área do conteúdo do elemento.

padding-box

As propriedades **width** e **height** definem as dimensões da área do conteúdo do elemento em conjunto com as margens internas.

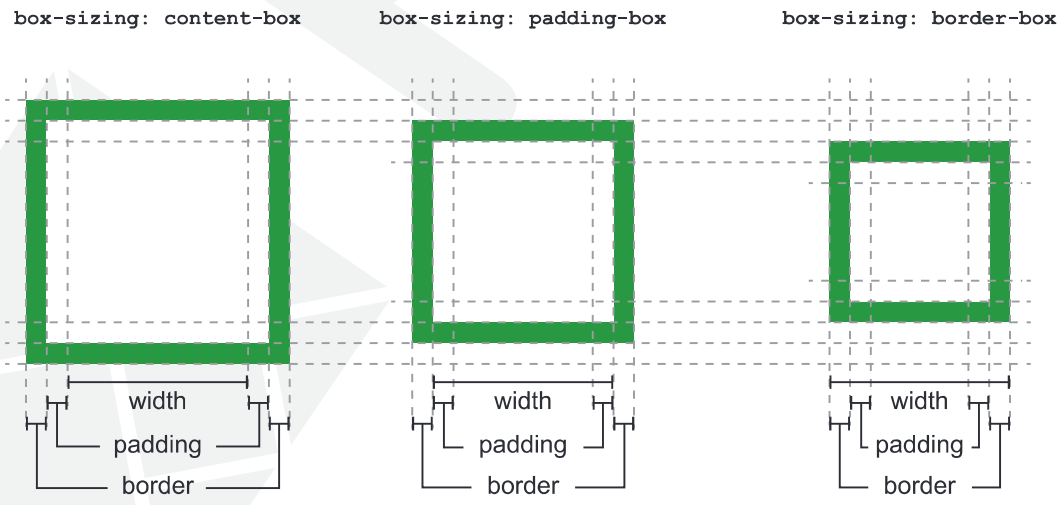
border-box

As propriedades **width** e **height** definem as dimensões da área do conteúdo do elemento em conjunto com as margens internas e bordas.

inherit

Assume o valor da mesma propriedade no elemento HTML pai.

Veja a figura abaixo para compreender melhor cada um dos valores:



Design Responsivo (Conteúdo Extra)

Durante muito tempo, os computadores tradicionais (desktops e laptops) foram os principais dispositivos de acesso à Internet. Nos últimos anos, uma grande variedade de novos equipamentos capazes de acessar a Internet foram introduzidos no mercado. Desses dispositivos podemos destacar os smartphones e tablets (dispositivos móveis).

O número de usuários que acessam a Internet através de dispositivos móveis está cada vez maior e só tende a crescer. Portanto, os desenvolvedores web devem ficar cada vez mais preocupados em atender de maneira satisfatória o público mobile.

Atender o público mobile de maneira satisfatória significa apresentar o conteúdo de uma página utilizando da melhor forma possível os recursos do dispositivo.

Por exemplo, uma página pode ter uma usabilidade muito boa quando acessada através de um desktop. Porém, ao acessá-la através de um smartphone, as letras podem ficar muito pequenas, o conteúdo pode não se encaixar no tamanho da tela ou as interações através do toque podem não funcionar tão bem se comparadas às interações realizadas com o mouse.

Uma solução para esse problema seria a criação de uma versão do site para cada tipo de dispositivo. Geralmente, a versão para desktop é a mais completa e as demais possuem recursos reduzidos. Essa é uma solução adotada por diversos sites. Normalmente, as versões mobile são acessadas através de URLs específicas. Em geral, essas URLs começam com mobile, mob ou até mesmo m (ex: `http.//m.kekanto.com`). Entretanto, antes de adotar essa solução devemos levantar algumas questões:

Como fica a manutenção do site?

Muito mais trabalhosa. Basicamente, para cada tipo de dispositivo, teríamos um site diferente, ou seja, teríamos que gerenciar dois ou mais sites ao invés de apenas um.

O que fazer com as URLs do site?

Para cada tipo de dispositivo uma URL diferente. Por exemplo, uma para mobile (`http.//mob.dominio.com`), outra para televisores (`http.//tv.dominio.com`) e outra para desktops (`http.//www.dominio.com`). Além disso, deveríamos nos preocupar com o redirecionamento das URLs, pois seria mais apropriado que um usuário acessando o endereço `www.dominio.com` através de sua televisão fosse redirecionado para `tv.dominio.com`.

O usuário vai ficar insatisfeito se ele não puder fazer tudo que ele fazia no desktop?

Sim! Se a versão diferenciada for uma versão reduzida da versão desktop, ele ficará extremamente decepcionado. Os usuários tendem a querer ter todo o poder do desktop nos smartphones e afins.

Para evitar esses e outros problemas, uma outra solução que está ganhando cada vez mais força parte do princípio de que a página deve ser a mesma. O que deve mudar é a forma de apresentá-la ao usuário. Essa é a ideia do Design Responsivo.

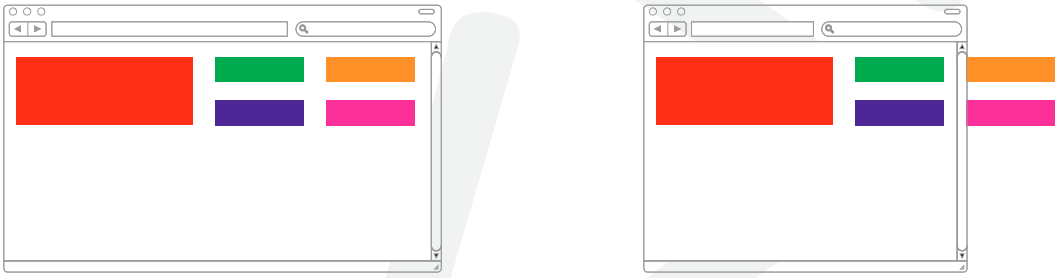
O design responsivo pode ser dividido em três partes principais: layouts flexíveis, media queries e mídias flexíveis. A seguir iremos discutir sobre cada parte.

Layouts Flexíveis

A ideia por trás dos layouts flexíveis é fazer com que os elementos de uma página sejam capazes de terem as suas dimensões adaptadas dinamicamente. Isso é bem interessante quando queremos que o layout da página se adapte a diferentes tamanhos de tela ou alguns elementos sejam redimensionados após uma interação do usuário, por exemplo.

Quando criamos layouts estáticos, em geral, trabalhamos com unidades de medida estáticas como **px**, **pt**, **in**, **cm** e etc. Ao criarmos layouts flexíveis passamos a trabalhar muito mais com as unidades relativas como **%**, **em**, **ex** e etc. Fazemos isso para evitar que ocorram deformações ou overflow na apresentação do conteúdo.

Para facilitar o trabalho de quem desenhará o layout de uma página e também de quem produzirá o código HTML e CSS, uma técnica muito comum é a de tentar visualizar a tela como um sistema de grades. Nesse sistema, cada grade deve ter as suas dimensões definidas com unidades de medida relativas. Dessa forma, se a tela do usuário for redimensionada, todo o sistema de grade será reajustado.



Layout estático

Layout flexível

Media Queries

Como o objetivo do design responsivo é apresentar o conteúdo de uma página de maneiras diferentes em dispositivos com tamanhos de telas diferentes, em algum momento sentiremos a necessidade de criar regras CSS para cada tamanho de tela ou para cada conjunto de tamanho de telas. Em uma situação como essa, as media queries se encaixam perfeitamente e devemos utilizá-las.

Breakpoints

O ponto principal da relação das media queries com o design responsivo é a definição daquilo que chamamos de breakpoints (pontos de quebra). Muitos autores costumam definir esses pontos

em torno dos tamanhos dos viewports mais comuns como 320px, 480px, 768px, 1024px e assim por diante. Essa é uma estratégia normalmente chamada de **device-driven breakpoints**, **device-based breakpoints** ou pontos de quebra orientados à dispositivos. Evite essa técnica, pois, como o próprio nome diz, ela faz com que as regras CSS sejam aplicadas quando uma página for aberta em dispositivos com tamanhos de tela esperados, ou seja, a página poderá estar despreparada quando for aberta por um dispositivo com tela de 400px, por exemplo.

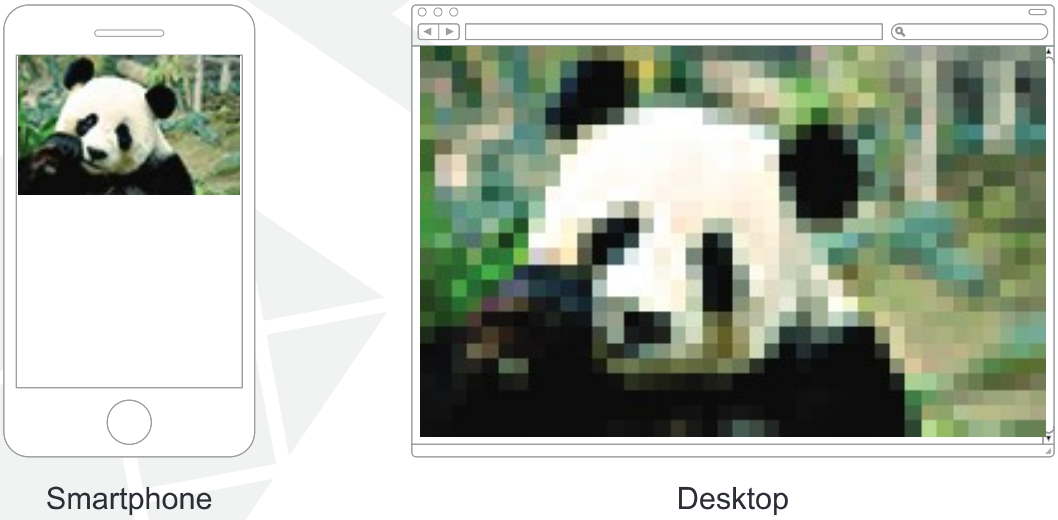
Uma estratégia muito mais interessante é a que chamamos de **content-driven breakpoints**, **content-based breakpoints** ou pontos de quebra orientados ao conteúdo. Nessa estratégia definimos os pontos de quebra observando se o conteúdo da página se acomoda corretamente ao tamanho do viewport. Uma técnica muito eficiente é começar do menor para o maior tamanho de viewport. Em geral, o menor tamanho refere-se aos dispositivos mobile e, por isso, a técnica recebe o nome de **mobile first**.

Aplicar a estratégia do **content-drive breakpoints** juntamente com a técnica **mobile first** é relativamente simples: redimensione a janela do navegador para que o viewport fique com o menor tamanho esperado (mobile). Esse será o primeiro breakpoint. Aplique as regras CSS na página e em seguida redimensione a janela do navegador até que o conteúdo não seja apresentado corretamente ou a página tenha o layout "quebrado". Quando isso ocorrer significa que encontramos o próximo breakpoint. Nesse breakpoint, aplique as regras CSS para que a página volte a ser apresentada corretamente. Repita esse processo até considerar que tenha definido todos os breakpoints necessários.

Mídias Flexíveis

Para criarmos uma página que seja confortável para a leitura, devemos ajustar o posicionamento e a dimensão dos elementos conforme vimos anteriormente. As imagens, players de vídeo e áudio não são exceções. Entretanto, quando trabalhamos com certos tipos de mídias devemos ter alguns cuidados especiais.

Suponha uma imagem de 320px de largura que gostaríamos de exibir em uma página de maneira que ela se adapte à largura da tela. Conseguiríamos fazer isso utilizando a propriedade width: 100%, por exemplo. Veja abaixo como ficaria essa página exibida em um smartphone e em um desktop.



Na imagem acima podemos perceber que a qualidade da imagem diminuiu ao esticá-la. Para evitar ou amenizar esse tipo de problema poderíamos criar diversas versões da imagem, uma para

cada breakpoint. Poderíamos também obter uma imagem original de tamanho maior, pois, em geral, diminuir uma imagem perde menos qualidade do que esticar. Ou ainda, dependendo da imagem, poderíamos utilizar uma versão vetorial da mesma. Como podemos observar, existem diversas técnicas para resolver esse tipo de problema e, caso tenha interesse, procure por **Icon Fonts**, formato de imagem **SVG** e **Compressive Images**. Esses são termos muito utilizados por técnicas de otimização de imagens na web.

No caso dos players de vídeo e áudio recomendamos a leitura do artigo neste endereço: http://alistapart.com/article/creating-intrinsic-ratios-for-video

Exercícios Complementares

1. No projeto **css**, crie um arquivo chamado **background-color.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-color</title>
    <link rel="stylesheet" type="text/css" href="background-color.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.38: background-color.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar1.zip

2. No projeto **css**, crie um arquivo CSS chamado **background-color.css**.

```css
div {
  height. 100px,
  width. 400px,
}

#div1 {
  background-color. red,
}

#div2 {
  background-color. #00ff00,
}

#div3 {
  background-color. rgb(0, 0, 255),
}
```

Código CSS 3.60: background-color.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar2.zip

3. No **Windows**, utilize o **Chrome** para acessar o endereço:

   http://localhost/css/public_html/background-color.html.

   No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

   http://localhost/~<USUARIO>/css/public_html/background-color.html.

4. No projeto **css**, crie um arquivo chamado **background-image.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-image</title>
    <link rel="stylesheet" type="text/css" href="background-image.css">
  </head>
  <body>

  </body>
</html>
```

Código HTML 3.39: background-image.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar4.zip

5. No projeto **css**, crie um arquivo CSS chamado **background-image.css**.

```
body {
  background-image.
    url("http://www.k19.com.br/figs/planeta.png"),
    url("http://www.k19.com.br/figs/fundo.jpg"),
}
```

Código CSS 3.61: background-image.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar5.zip

6. No **Windows**, utilize o **Chrome** para acessar o endereço:

   http://localhost/css/public_html/background-image.html.

   No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

   http://localhost/~<USUARIO>/css/public_html/background-image.html.

7. No projeto **css**, crie um arquivo chamado **background-repeat.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-repeat</title>
    <link rel="stylesheet" type="text/css" href="background-repeat.css">
  </head>
  <body>
    <div id="div1"></div>
```

```
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.40: background-repeat.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar7.zip

8 No projeto **css**, crie um arquivo CSS chamado **background-repeat.css**.

```
div {
  height. 300px,
  width. 400px,
  border. 1px solid black,
  margin. 10px,
}

#div1 {
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
}

#div2 {
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
  background-repeat. no-repeat,
}

#div3 {
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
  background-repeat. repeat-x,
}

#div4 {
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
  background-repeat. repeat-y,
}
```

Código CSS 3.62: background-repeat.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar8.zip

9 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/background-repeat.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/background-repeat.html.

10 No projeto **css**, crie um arquivo chamado **background-attachment.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-attachment</title>
    <link rel="stylesheet" type="text/css" href="background-attachment.css">
  </head>
  <body>

```

```
10   </body>
11 </html>
```

Código HTML 3.41: background-attachment.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar10.zip

11 No projeto **css**, crie um arquivo CSS chamado **background-attachment.css**.

```
1 body {
2   height. 2000px,
3   background-image. url("http://www.k19.com.br/figs/fundo.jpg"),
4 }
```

Código CSS 3.63: background-attachment.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar11.zip

12 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/background-attachment.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/background-attachment.html.

Observe o comportamento da imagem de background quando você movimenta a barra de rolagem.

13 No projeto **css**, altere o arquivo CSS **background-attachment.css**.

```
1 body {
2   height. 2000px,
3   background-image. url("http://www.k19.com.br/figs/fundo.jpg"),
4   background-attachment. fixed,
5 }
```

Código CSS 3.64: background-attachment.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar13.zip

14 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/background-attachment.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/background-attachment.html.

Observe o comportamento da imagem de background quando você movimenta a barra de rolagem.

15 No projeto **css**, crie um arquivo chamado **background-position.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-position</title>
    <link rel="stylesheet" type="text/css" href="background-position.css">
  </head>
  <body>

  </body>
</html>
```

Código HTML 3.42: background-position.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar15.zip

16 No projeto **css**, crie um arquivo CSS chamado **background-position.css**.

```css
body {
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
  background-repeat. no-repeat,
  background-position. center top,
}
```

Código CSS 3.65: background-position.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar16.zip

17 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/background-position.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/background-position.html.

18 No projeto **css**, crie um arquivo chamado **background-clip.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-clip</title>
    <link rel="stylesheet" type="text/css" href="background-clip.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.43: background-clip.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar18.zip

19 No projeto **css**, crie um arquivo CSS chamado **background-clip.css**.

```
div {
  width. 400px,
  height. 50px,
  margin. 10px,
  padding. 50px,
  border. 10px dotted black,
  background-color. yellow,
}

#div1 {
  background-clip. border-box,
}

#div2 {
  background-clip. padding-box,
}

#div3 {
  background-clip. content-box,
}
```

Código CSS 3.66: background-clip.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar19.zip

20 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/background-clip.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/background-clip.html.

21 No projeto **css**, crie um arquivo chamado **background-origin.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-origin</title>
    <link rel="stylesheet" type="text/css" href="background-origin.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.44: background-origin.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar21.zip

22 No projeto **css**, crie um arquivo CSS chamado **background-origin.css**.

```
div {
  width. 400px,
  height. 100px,
  margin. 10px,
  padding. 25px,
```

```
  border. 10px dotted black,
  background-repeat. no-repeat,
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
}

#div1 {
  background-origin. border-box,
}

#div2 {
  background-origin. padding-box,
}

#div3 {
  background-origin. content-box,
}
```

Código CSS 3.67: background-origin.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar22.zip

23 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/background-origin.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/background-origin.html.

24 No projeto **css**, crie um arquivo chamado **background-size.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - background-size</title>
    <link rel="stylesheet" type="text/css" href="background-size.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.45: background-size.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar24.zip

25 No projeto **css**, crie um arquivo CSS chamado **background-size.css**.

```
div {
  width. 400px,
  height. 100px,
  margin. 10px,
  padding. 25px,
  border. 10px dotted black,
  background-repeat. no-repeat,
  background-image. url("http.//www.k19.com.br/figs/planeta-small.png"),
}

```

```
#div1 {
  background-size. 50px auto,
}

#div2 {
  background-size. cover,
}

#div3 {
  background-size. contain,
}
```

Código CSS 3.68: background-size.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar25.zip

26 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/background-size.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/background-size.html.

27 No projeto **css**, crie um arquivo chamado **color.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - color</title>
    <link rel="stylesheet" type="text/css" href="color.css">
  </head>
  <body>
    <p id="p1">K19 Treinamentos</p>
    <p id="p2">K19 Treinamentos</p>
    <p id="p3">K19 Treinamentos</p>
  </body>
</html>
```

Código HTML 3.46: color.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar27.zip

28 No projeto **css**, crie um arquivo CSS chamado **color.css**.

```
#p1 {
  color. red,
}

#p2 {
  color. #00ff00,
}

#p3 {
  color. rgb(0, 0, 255),
}
```

Código CSS 3.69: color.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar28.zip

29 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/color.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/color.html.

30 No projeto **css**, crie um arquivo chamado **text-align.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - text-align</title>
    <link rel="stylesheet" type="text/css" href="text-align.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p4">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.47: text-align.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar30.zip

31 No projeto **css**, crie um arquivo CSS chamado **text-align.css**.

```
p {
  border. 1px solid black,
  width. 400px,
}
```

```

#p1 {
  text-align. left,
}

#p2 {
  text-align. center,
}

#p3 {
  text-align. right,
}

#p4 {
  text-align. justify,
}
```

Código CSS 3.70: text-align.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar31.zip

32 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/text-align.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/text-align.html.

33 No projeto **css**, crie um arquivo chamado **text-decoration.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - text-decoration</title>
    <link rel="stylesheet" type="text/css" href="text-decoration.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.48: text-decoration.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar33.zip

34 No projeto **css**, crie um arquivo CSS chamado **text-decoration.css**.

```
p {
  width. 400px,
}

#p1 {
  text-decoration. underline,
}

#p2 {
  text-decoration. overline,
}

#p3 {
  text-decoration. line-through,
}
```

Código CSS 3.71: text-decoration.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar34.zip

35 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/text-decoration.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/text-decoration.html.

36 No projeto **css**, crie um arquivo chamado **text-transform.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - text-transform</title>
    <link rel="stylesheet" type="text/css" href="text-transform.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
```

```
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.49: text-transform.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar36.zip

37 No projeto **css**, crie um arquivo CSS chamado **text-transform.css**.

```
p {
  width. 400px,
}

#p1 {
  text-transform. capitalize,
}

#p2 {
  text-transform. uppercase,
}

#p3 {
  text-transform. lowercase,
}
```

Código CSS 3.72: text-transform.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar37.zip

38 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/text-transform.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/text-transform.html.

39 No projeto **css**, crie um arquivo chamado **text-indent.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - text-indent</title>
    <link rel="stylesheet" type="text/css" href="text-indent.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
```

```
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.50: text-indent.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar39.zip

40 No projeto **css**, crie um arquivo CSS chamado **text-indent.css**.

```
p {
  width. 400px,
}

#p1 {
  text-indent. 0px,
}

#p2 {
  text-indent. 30px,
}
```

Código CSS 3.73: text-indent.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar40.zip

41 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/text-indent.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/text-indent.html.

42 No projeto **css**, crie um arquivo chamado **letter-spacing.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - letter-spacing</title>
    <link rel="stylesheet" type="text/css" href="letter-spacing.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p4">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.51: letter-spacing.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar42.zip

43 No projeto **css**, crie um arquivo CSS chamado **letter-spacing.css**.

```
#p1 {
  letter-spacing. 2px,
}

#p2 {
  letter-spacing. -2px,
}

#p3 {
  letter-spacing. 4px,
}

#p4 {
  letter-spacing. 6px,
}
```

Código CSS 3.74: letter-spacing.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar43.zip

44 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/letter-spacing.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/letter-spacing.html.

45 No projeto **css**, crie um arquivo chamado **word-spacing.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - word-spacing</title>
    <link rel="stylesheet" type="text/css" href="word-spacing.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p4">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.52: word-spacing.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar45.zip

46 No projeto **css**, crie um arquivo CSS chamado **word-spacing.css**.

```
#p1 {
    word-spacing. 10px,
}

#p2 {
    word-spacing. -5px,
}

#p3 {
    word-spacing. 30px,
}

#p4 {
    word-spacing. 50px,
}
```

Código CSS 3.75: word-spacing.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar46.zip

47 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/word-spacing.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/word-spacing.html.

48 No projeto **css**, crie um arquivo chamado **word-wrap.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - word-wrap</title>
    <link rel="stylesheet" type="text/css" href="word-wrap.css">
  </head>
  <body>
    <p id="p1">
      Otorrinolaringologista.
    </p>
    <p>
      Otorrinolaringologista.
    </p>
  </body>
</html>
```

Código HTML 3.53: word-wrap.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar48.zip

49 No projeto **css**, crie um arquivo CSS chamado **word-wrap.css**.

```
p {
    width. 100px,
    border. 1px solid black,
}

```

```
#p1 {
    word-wrap. break-word,
}
```

Código CSS 3.76: word-wrap.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar49.zip

50 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/word-wrap.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/word-wrap.html.

51 No projeto **css**, crie um arquivo chamado **line-height.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - line-height</title>
    <link rel="stylesheet" type="text/css" href="line-height.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.54: line-height.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar51.zip

52 No projeto **css**, crie um arquivo CSS chamado **line-height.css**.

```
p {
    width. 400px,
    border. 1px solid black,
}

#p1 {
    line-height. 20px,
}

#p2 {
    line-height. 40px,
}
```

Código CSS 3.77: line-height.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar52.zip

53 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/line-height.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/line-height.html.

54 No projeto **css**, crie um arquivo chamado **white-space.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - white-space</title>
    <link rel="stylesheet" type="text/css" href="white-space.css">
  </head>
  <body>
    <p id="p1">
      Lorem
      ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula
      bibendum arcu.

      Sed
        dui nulla, cursus et lacinia eu, vulputate ac dolor.
      Quisque faucibus congue congue.
    </p>
    <p id="p2">
      Lorem
      ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula
      bibendum arcu.

      Sed
        dui nulla, cursus et lacinia eu, vulputate ac dolor.
      Quisque faucibus congue congue.
    </p>
    <p id="p3">
      Lorem
      ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula
      bibendum arcu.

      Sed
        dui nulla, cursus et lacinia eu, vulputate ac dolor.
      Quisque faucibus congue congue.
    </p>
    <p id="p4">
      Lorem
      ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula
      bibendum arcu.

      Sed
        dui nulla, cursus et lacinia eu, vulputate ac dolor.
      Quisque faucibus congue congue.
    </p>
    <p id="p5">
      Lorem
      ipsum dolor sit amet, consectetur adipiscing elit. Cras vehicula
      bibendum arcu.
```

```

      Sed
        dui nulla, cursus et lacinia eu, vulputate ac dolor.
      Quisque faucibus congue congue.
    </p>
  </body>
</html>
```

Código HTML 3.55: white-space.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar54.zip

55 No projeto **css**, crie um arquivo CSS chamado **white-space.css**.

```
p {
    width. 400px,
    border. 1px solid black,
}

#p1 {
    white-space. nowrap,
}

#p2 {
    white-space. pre,
}

#p3 {
    white-space. pre-line,
}

#p4 {
    white-space. pre-wrap,
}
```

Código CSS 3.78: white-space.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar55.zip

56 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/white-space.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/white-space.html.

57 No projeto **css**, crie um arquivo chamado **text-shadow.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - text-shadow</title>
    <link rel="stylesheet" type="text/css" href="text-shadow.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
```

```
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p4">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p5">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.56: text-shadow.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar57.zip

58 No projeto **css**, crie um arquivo CSS chamado **text-shadow.css**.

```
p {
  font-size. xx-large,
}

#p1 {
  text-shadow. 3px 3px #ff0000,
}

#p2 {
  text-shadow. -3px -3px #ff0000,
}

#p3 {
  text-shadow. 0 0 10px #ff0000,
}

#p4 {
  text-shadow. 10px 10px 10px #ff0000,
}

#p5 {
  text-shadow. 5px 5px 5px #ff0000, -5px -5px 5px #0000ff,
}
```

Código CSS 3.79: text-shadow.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar58.zip

59 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/text-shadow.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/text-shadow.html.

60 No projeto **css**, crie um arquivo chamado **font-family.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
```

```
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>K19 - font-family</title>
6     <link rel="stylesheet" type="text/css" href="font-family.css">
7   </head>
8   <body>
9     <p id="p1">
10      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
11    </p>
12    <p id="p2">
13      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
14    </p>
15    <p id="p3">
16      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
17    </p>
18    <p id="p4">
19      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
20    </p>
21    <p id="p5">
22      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
23    </p>
24  </body>
25 </html>
```

Código HTML 3.57: font-family.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar60.zip

61 No projeto **css**, crie um arquivo CSS chamado **font-family.css**.

```
1  p {
2    font-size. xx-large,
3  }
4
5  #p1 {
6    font-family. "Arial",
7  }
8
9  #p2 {
10   font-family. "Courier",
11 }
12
13 #p3 {
14   font-family. "Verdana",
15 }
16
17 #p4 {
18   font-family. "serif",
19 }
20
21 #p5 {
22   font-family. "cursive",
23 }
```

Código CSS 3.80: font-family.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar61.zip

62 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/font-family.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/font-family.html.

63 No projeto **css**, crie um arquivo chamado **font-size.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - font-size</title>
    <link rel="stylesheet" type="text/css" href="font-size.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p4">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p5">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.58: font-size.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar63.zip

64 No projeto **css**, crie um arquivo CSS chamado **font-size.css**.

```
#p1 {
    font-size. x-small,
}

#p2 {
    font-size. x-large,
}

#p3 {
    font-size. xx-large,
}

#p4 {
    font-size. 18px,
}

#p5 {
    font-size. 30px,
}
```

Código CSS 3.81: font-size.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar64.zip

65 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/font-size.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/font-size.html.

66 No projeto **css**, crie um arquivo chamado **font-style.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - font-style</title>
    <link rel="stylesheet" type="text/css" href="font-style.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.59: font-style.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar66.zip

67 No projeto **css**, crie um arquivo CSS chamado **font-style.css**.

```
p {
  font-size. xx-large,
}

#p1 {
  font-style. italic,
}

#p2 {
  font-style. oblique,
}
```

Código CSS 3.82: font-style.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar67.zip

68 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/font-style.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/font-style.html.

69 No projeto **css**, crie um arquivo chamado **font-variant.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - font-variant</title>
    <link rel="stylesheet" type="text/css" href="font-variant.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.60: font-variant.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar69.zip

70 No projeto **css**, crie um arquivo CSS chamado **font-variant.css**.

```
p {
  font-size. xx-large,
}

#p1 {
  font-variant. small-caps,
}
```

Código CSS 3.83: font-variant.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar70.zip

71 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/font-variant.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/font-variant.html.

72 No projeto **css**, crie um arquivo chamado **font-weight.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - font-weight</title>
    <link rel="stylesheet" type="text/css" href="font-weight.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
```

```
14      </p>
15      <p id="p3">
16        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
17      </p>
18    </body>
19  </html>
```

Código HTML 3.61: font-weight.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar72.zip

73 No projeto **css**, crie um arquivo CSS chamado **font-weight.css**.

```
1  p {
2    font-size. xx-large,
3  }
4
5  #p1 {
6    font-weight. 400,
7  }
8
9  #p2 {
10   font-weight. 700,
11 }
12
13 #p3 {
14   font-weight. bold,
15 }
```

Código CSS 3.84: font-weight.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar73.zip

74 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/font-weight.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/font-weight.html.

75 No projeto **css**, crie um arquivo chamado **font-face.html**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>K19 - font-face</title>
6      <link rel="stylesheet" type="text/css" href="font-face.css">
7      <link rel="stylesheet" type="text/css"
8        href="http.//fonts.googleapis.com/css?family=Norican" >
9      <link rel="stylesheet" type="text/css"
10       href="http.//fonts.googleapis.com/css?family=Bad+Script" >
11     <link rel="stylesheet" type="text/css"
12       href="http.//fonts.googleapis.com/css?family=Ceviche+One">
13   </head>
14   <body>
15     <p id="p1">
16       Lorem ipsum dolor sit amet, consectetur adipiscing elit.
17     </p>
18     <p id="p2">
```

```
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </p>
  </body>
</html>
```

Código HTML 3.62: font-face.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar75.zip

76 No projeto **css**, crie um arquivo CSS chamado **font-face.css**.

```
p {
  font-size. xx-large,
}

#p1 {
  font-family. "Norican",
}

#p2 {
  font-family. "Bad Script",
}

#p3 {
  font-family. "Ceviche One",
}
```

Código CSS 3.85: font-face.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar76.zip

77 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/font-face.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/font-face.html.

78 No projeto **css**, crie um arquivo chamado **list-style-type.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - list-style-type</title>
    <link rel="stylesheet" type="text/css" href="list-style-type.css">
  </head>
  <body>
    <ul id="ul1">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
    </ul>
    <ul id="ul2">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
```

```
18      </ul>
19      <ul id="ul3">
20        <li>K01 - Lógica de Programação</li>
21        <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
22        <li>K03 - Modelo Relacional e SQL</li>
23      </ul>
24      <ol id="ol1">
25        <li>K01 - Lógica de Programação</li>
26        <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
27        <li>K03 - Modelo Relacional e SQL</li>
28      </ol>
29      <ol id="ol2">
30        <li>K01 - Lógica de Programação</li>
31        <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
32        <li>K03 - Modelo Relacional e SQL</li>
33      </ol>
34      <ol id="ol3">
35        <li>K01 - Lógica de Programação</li>
36        <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
37        <li>K03 - Modelo Relacional e SQL</li>
38      </ol>
39    </body>
40  </html>
```

Código HTML 3.63: list-style-type.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar78.zip

79 No projeto **css**, crie um arquivo CSS chamado **list-style-type.css**.

```
1  #ul1 {
2    list-style-type. disc,
3  }
4
5  #ul2 {
6    list-style-type. circle,
7  }
8
9  #ul3 {
10   list-style-type. square,
11 }
12
13 #ol1 {
14   list-style-type. decimal,
15 }
16
17 #ol2 {
18   list-style-type. lower-latin,
19 }
20
21 #ol3 {
22   list-style-type. georgian,
23 }
```

Código CSS 3.86: list-style-type.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar79.zip

80 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/list-style-type.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/list-style-type.html.

81 No projeto **css**, crie um arquivo chamado **list-style-image.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - list-style-image</title>
    <link rel="stylesheet" type="text/css" href="list-style-image.css">
  </head>
  <body>
    <ul id="ul1">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
    </ul>
    <ul id="ul2">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
    </ul>
    <ul id="ul3">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
    </ul>
  </body>
</html>
```

Código HTML 3.64: list-style-image.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar81.zip

82 No projeto **css**, crie um arquivo CSS chamado **list-style-image.css**.

```
#ul1 {
  list-style-image. url("http://www.k19.com.br/figs/star.png"),
}

#ul2 {
  list-style-image. url("http://www.k19.com.br/figs/cake.png"),
}

#ul3 {
  list-style-image. url("http://www.k19.com.br/figs/flying_heart.png"),
}
```

Código CSS 3.87: list-style-image.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar82.zip

83 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/list-style-image.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/list-style-image.html.

84 No projeto **css**, crie um arquivo chamado **list-style-position.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - list-style-position</title>
    <link rel="stylesheet" type="text/css" href="list-style-position.css">
  </head>
  <body>
    <ul id="ul1">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
    </ul>
    <ul id="ul2">
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Modelo Relacional e SQL</li>
    </ul>
  </body>
</html>
```

Código HTML 3.65: list-style-position.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar84.zip

85 No projeto **css**, crie um arquivo CSS chamado **list-style-position.css**.

```
li {
  border. 1px solid black,
}

#ul1 {
  list-style-position. inside,
}
```

Código CSS 3.88: list-style-position.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar85.zip

86 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/list-style-position.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/list-style-position.html.

87 No projeto **css**, crie um arquivo chamado **border-style.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - border-style</title>
    <link rel="stylesheet" type="text/css" href="border-style.css">
  </head>
  <body>
    <div id="div1"></div>
```

```
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.66: border-style.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar87.zip

88 No projeto **css**, crie um arquivo CSS chamado **border-style.css**.

```
div {
  width. 400px,
  height. 200px,
  margin. 10px,
  border-width. 6px,
}

#div1 {
  border-style. dotted,
}

#div2 {
  border-style. dashed,
}

#div3 {
  border-style. solid,
}

#div4 {
  border-style. double,
}
```

Código CSS 3.89: border-style.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar88.zip

89 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/border-style.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/border-style.html.

90 No projeto **css**, crie um arquivo chamado **border-width.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - border-width</title>
    <link rel="stylesheet" type="text/css" href="border-width.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
```

```
  </body>
</html>
```

Código HTML 3.67: border-width.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar90.zip

91 No projeto **css**, crie um arquivo CSS chamado **border-width.css**.

```
div {
  width. 400px,
  height. 200px,
  margin. 10px,
  border-style. solid,
}

#div1 {
  border-width. thin,
}

#div2 {
  border-width. thick,
}

#div3 {
  border-width. 2px,
}

#div4 {
  border-width. 10px,
}
```

Código CSS 3.90: border-width.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar91.zip

92 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/border-width.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/border-width.html.

93 No projeto **css**, crie um arquivo chamado **border-color.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - border-color</title>
    <link rel="stylesheet" type="text/css" href="border-color.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.68: border-color.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar93.zip

94 No projeto **css**, crie um arquivo CSS chamado **border-color.css**.

```
div {
  width. 400px,
  height. 200px,
  border-style. solid,
}

#div1 {
  border-color. red,
}

#div2 {
  border-color. #00ff00,
}

#div3 {
  border-color. rgb(0, 0, 255),
}
```

Código CSS 3.91: border-color.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar94.zip

95 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/border-color.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/border-color.html.

96 No projeto **css**, crie um arquivo chamado **bordas-individuais.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - bordas-individuais</title>
    <link rel="stylesheet" type="text/css" href="bordas-individuais.css">
  </head>
  <body>
    <div></div>
  </body>
</html>
```

Código HTML 3.69: bordas-individuais.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar96.zip

97 No projeto **css**, crie um arquivo CSS chamado **bordas-individuais.css**.

```
div {
  width. 400px,
  height. 200px,

```

```
  border-top-style. dashed,
  border-top-color. red,
  border-top-width. 2px,

  border-right-style. dotted,
  border-right-color. #00ff00,
  border-right-width. 8px,

  border-bottom-style. double,
  border-bottom-color. rgb(0, 0, 255),
  border-bottom-width. 14px,

  border-left-style. solid,
  border-left-color. black,
  border-left-width. 20px,
}
```

Código CSS 3.92: bordas-individuais.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar97.zip

98 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/bordas-individuais.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/bordas-individuais.html.

99 No projeto **css**, crie um arquivo chamado **border-radius.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - border-radius</title>
    <link rel="stylesheet" type="text/css" href="border-radius.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.70: border-radius.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar99.zip

100 No projeto **css**, crie um arquivo CSS chamado **border-radius.css**.

```
div {
  width. 400px,
  height. 100px,
  margin. 10px,
  border. 2px solid black,
}

#div1 {
  border-radius. 20px,
```

```
}

#div2 {
  border-radius. 20px 60px,
}

#div3 {
  border-radius. 20px 60px 100px,
}

#div4 {
  border-radius. 20px 60px 100px 140px,
}
```

Código CSS 3.93: border-radius.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar100.zip

101 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/border-radius.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/border-radius.html.

102 No projeto **css**, crie um arquivo chamado **border-collapse.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - border-collapse</title>
    <link rel="stylesheet" type="text/css" href="border-collapse.css">
  </head>
  <body>
    <table id="tabela1">
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
    </table>
    <table id="tabela2">
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
    </table>
  </body>
</html>
```

Código HTML 3.71: border-collapse.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar102.zip

103 No projeto **css**, crie um arquivo CSS chamado **border-collapse.css**.

```
table {
  border. 2px solid black,
  margin. 10px,
}

td {
  border. 2px solid black,
}

#tabela1 {
  border-collapse. collapse,
}
```

Código CSS 3.94: border-collapse.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar103.zip

104 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/border-collapse.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/border-collapse.html.

105 No projeto **css**, crie um arquivo chamado **border-spacing.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - border-spacing</title>
    <link rel="stylesheet" type="text/css" href="border-spacing.css">
  </head>
  <body>
    <table id="tabela1">
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
    </table>
    <table id="tabela2">
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
      <tr>
        <td>K19</td>
        <td>K19</td>
      </tr>
    </table>
  </body>
</html>
```

Código HTML 3.72: border-spacing.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar105.zip

106 No projeto **css**, crie um arquivo CSS chamado **border-spacing.css**.

```
table {
  border. 2px solid black,
  margin. 10px,
}

td {
  border. 2px solid black,
}

#tabela1 {
  border-spacing. 10px 5px,
}
```

Código CSS 3.95: border-spacing.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar106.zip

107 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/border-spacing.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/border-spacing.html.

108 No projeto **css**, crie um arquivo chamado **outline-style.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - outline-style</title>
    <link rel="stylesheet" type="text/css" href="outline-style.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.73: outline-style.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar108.zip

109 No projeto **css**, crie um arquivo CSS chamado **outline-style.css**.

```
div {
  width. 400px,
  height. 100px,
  margin. 10px,
}

#div1 {
```

```
  outline-style. dashed,
}

#div2 {
  outline-style. dotted,
}

#div3 {
  outline-style. solid,
}
```

Código CSS 3.96: outline-style.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar109.zip

110 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/outline-style.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/outline-style.html.

111 No projeto **css**, crie um arquivo chamado **outline-color.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - outline-color</title>
    <link rel="stylesheet" type="text/css" href="outline-color.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.74: outline-color.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar111.zip

112 No projeto **css**, crie um arquivo CSS chamado **outline-color.css**.

```
div {
  width. 400px,
  height. 100px,
  margin. 10px,
  outline-style. solid,
}

#div1 {
  outline-color. red,
}

#div2 {
  outline-color. #00ff00,
}

#div3 {
```

```
  outline-color. rgb(0, 0, 255),
}
```

Código CSS 3.97: outline-color.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar112.zip

113 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/outline-color.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/outline-color.html.

114 No projeto **css**, crie um arquivo chamado **outline-width.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - outline-width</title>
    <link rel="stylesheet" type="text/css" href="outline-width.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.75: outline-width.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar114.zip

115 No projeto **css**, crie um arquivo CSS chamado **outline-width.css**.

```
div {
  width. 400px,
  height. 100px,
  margin. 20px,
  outline-style. solid,
}

#div1 {
  outline-width. thin,
}

#div2 {
  outline-width. thick,
}

#div3 {
  outline-width. 2px,
}
```

Código CSS 3.98: outline-width.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar115.zip

116 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/outline-width.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/outline-width.html.

117 No projeto **css**, crie um arquivo chamado **outline-offset.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - outline-offset</title>
    <link rel="stylesheet" type="text/css" href="outline-offset.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
  </body>
</html>
```

Código HTML 3.76: outline-offset.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar117.zip

118 No projeto **css**, crie um arquivo CSS chamado **outline-offset.css**.

```
div {
  width. 400px,
  height. 50px,
  margin. 50px,
  outline. 5px solid red,
  border. 5px solid black,
}

#div1 {
  outline-offset. 20px,
}

#div2 {
  outline-offset. -20px,
}
```

Código CSS 3.99: outline-offset.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar118.zip

119 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/outline-offset.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/outline-offset.html.

120 No projeto **css**, crie um arquivo chamado **box-shadow.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - box-shadow</title>
    <link rel="stylesheet" type="text/css" href="box-shadow.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.77: box-shadow.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar120.zip

121 No projeto **css**, crie um arquivo CSS chamado **box-shadow.css**.

```
div {
  width. 200px,
  height. 100px,
  border. 4px solid black,
  margin. 50px,
}

#div1 {
  box-shadow . 10px 5px gray,
}

#div2 {
  box-shadow . 10px 5px 5px gray,
}

#div3 {
  box-shadow . inset 10px 10px gray,
}

#div4 {
  box-shadow . inset 10px 10px 5px gray,
}
```

Código CSS 3.100: box-shadow.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar121.zip

122 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/box-shadow.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/box-shadow.html.

123 No projeto **css**, crie um arquivo chamado **margens-externas.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - margens-externas</title>
    <link rel="stylesheet" type="text/css" href="margens-externas.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.78: margens-externas.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar123.zip

124 No projeto **css**, crie um arquivo CSS chamado **margens-externas.css**.

```css
div {
  display. inline-block,
  border. 1px solid black,
  width. 100px,
  height. 100px,
}

#div1 {
  margin-top. 50px,
  margin-right. 50px,
}

#div2 {
  margin-bottom. 50px,
}

#div3 {
  margin-left. 50px,
}

#div4 {
  margin-left. 50px,
  margin-bottom. 50px,
}
```

Código CSS 3.101: margens-externas.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar124.zip

125 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/margens-externas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/margens-externas.html.

126 No projeto **css**, crie um arquivo chamado **margens-internas.html**.

```html
<!DOCTYPE html>
```

```html
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - margens-internas</title>
    <link rel="stylesheet" type="text/css" href="margens-internas.css">
  </head>
  <body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
    <div id="div4"></div>
  </body>
</html>
```

Código HTML 3.79: margens-internas.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar126.zip

127 No projeto **css**, crie um arquivo CSS chamado **margens-internas.css**.

```css
div {
  display. inline-block,
  border. 2px solid black,
  width. 100px,
  height. 100px,
  background-color. yellow,
  background-clip. content-box,
}

#div1 {
  padding-top. 50px,
  padding-right. 50px,
}

#div2 {
  padding-bottom. 50px,
}

#div3 {
  padding-left. 50px,
}

#div4 {
  padding-left. 50px,
  padding-bottom. 50px,
}
```

Código CSS 3.102: margens-internas.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-css-complementar127.zip

128 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/css/public_html/margens-internas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/css/public_html/margens-internas.html.

129 No projeto **css**, crie um arquivo chamado **conteudo-dimensao.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - conteudo-dimensao</title>
    <link rel="stylesheet" type="text/css" href="conteudo-dimensao.css">
  </head>
  <body>
    <p id="p1">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p2">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
    <p id="p3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. In
      mollis, nibh et tempor sollicitudin, orci lorem iaculis sem,
      sit amet sodales quam massa sit amet risus. Fusce malesuada
      eleifend massa, ac mollis mi fermentum vitae. Nullam consequat
      leo nunc, in ornare turpis aliquam quis.
    </p>
  </body>
</html>
```

Código HTML 3.80: conteudo-dimensao.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar129.zip

130 No projeto **css**, crie um arquivo CSS chamado **conteudo-dimensao.css**.

```css
p {
  border. 1px solid black,
  max-height. 200px,
  max-width. 400px,
}

#p1 {
  width. 200px,
}

#p2 {
  height. 130px,
}
```

Código CSS 3.103: conteudo-dimensao.css

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-css-complementar130.zip

131 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/css/public_html/conteudo-dimensao.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/css/public_html/conteudo-dimensao.html.

Resumo do Capítulo

1. A linguagem **CSS** é utilizada para formatar visualmente as páginas web.

2. Podemos considerar cada elemento HTML como um **box**.

3. A estrutura dos boxes é definida pelo **Box Model**.

4. De acordo com o **Box Model**, cada box possui **conteúdo**, **margens internas**, **bordas** e **margens externas**.

5. Por padrão, os boxes dos **block-level elements** ocupam todo o espaço horizontal do elemento pai e provocam quebras de linha.

6. Por padrão, os boxes dos **inline-level elements** ocupam apenas o espaço necessário e não provocam quebras de linha.

7. A formatação dos elementos HTML é definida através das **regras CSS**

8. Uma regra CSS é composta por **seletor** e **corpo**. No **corpo**, são definidas as **propriedades CSS**.

9. Os seletores definem quais elementos HTML serão afetados pelas regras CSS.

10. As propriedades CSS definem as características visuais dos elementos HTML.

11. As regras CSS podem ser aplicadas aos documentos HTML de três formas diferentes: **inline style**, **internal style sheet** e **external style sheet**.

12. Os comentários CSS são definidos com os marcadores **/*** e ***/**.

13. Na linguagem CSS, toda cor está associada a um código numérico expresso em hexadecimal. Algumas cores possuem nome.

14 Podemos escolher uma cor através do código dela ou através das funções **rgb()**, **rgba()**, **hsl()** e **hsla()**. As cores que possuem nome podem ser escolhidas através do seu nome.

15 As unidades absolutas da linguagem CSS são: **cm**, **mm**, **in**, **pt**, **pc** e **px**.

16 As unidades relativas da linguagem CSS são: **em**, **ex**, **ch**, **rem**, **vw**, **vh**, **vmin** e **vmax**.

17 As principais propriedades de background são: **background-color**, **background-image**, **background-repeat**, **background-attachment**, **background-position**, **background-clip**, **background-origin**, **background-size** e **background**.

18 As principais propriedades de texto são: **color**, **text-align**, **text-decoration**, **text-transform**, **text-indent**, **letter-spacing**, **word-spacing**, **word-wrap**, **line-height**, **white-space** e **text-shadown**.

19 As principais propriedades de fonte são: **font-family**, **font-size**, **font-style**, **font-variant**, **font-weight**, **font** e **@font-face**.

20 As principais propriedades de lista são: **list-style-type**, **list-style-image**, **list-style-position** e **list-style**.

21 As principais propriedades de borda são: **border-style**, **border-width**, **border-color**, **border**, **border-left-***, **border-top-***, **border-right-***, **border-bottom-***, **border-radius**, **border-*-*-radius**, **border-collapse** e **border-spacing**.

22 As principais propriedades de outline são: **outline-style**, **outline-color**, **outline-width**, **outline-offset** e **outline**.

23 A principal propriedade de sombra é **box-shadow**.

24 As principais propriedades de margens são: **margin-***, **margin**, **padding-*** e **padding**.

25 As principais propriedades de altura e largura são: **height**, **width**, **min-height**, **max-height**, **min-width** e **max-width**.

26 A propriedade **display** define a forma de exibição dos boxes.

27 A propriedade **display** define a forma de exibição dos boxes.

28 A propriedade **visibility** define a visibilidade dos boxes.

29 Podemos controlar a opacidade dos boxes com a propriedade **opacity**.

30 Os quatro tipos de posicionamento do CSS são: **static**, **relative**, **fixed** e **absolute**.

31 As principais propriedades de posicionamento são: **position**, **float**, **clear** e **z-index**.

32 As principais propriedades de overflow e clips são: **overflow**, **overflow-x**, **overflow-y** e **clip**.

33 As principais funções de transformação são: **translate()**, **scale()**, **rotate()** e **skew()**.

34 As principais propriedades de transição são: **transition-duration**, **transition-delay**, **transition-timing-function** e **transition-property**.

35 As principais propriedades de animação são: **animation-timing-function**, **animation-delay**, **animation-iteration-count**, **animation-direction** e **animation-play-state**.

36 Podemos dividir os seletores do CSS em **básicos**, **seletores de atributo**, **pseudo-classes** e **pseudo-elementos**.

37 Quando ocorre conflito entre duas ou mais regras CSS, devemos calcular a prioridade dos seletores dessas regras para saber qual delas será aplicada.

38 Com as media queries podemos definir formatações específicas para os diversos tipos de dispositivos que desejamos suportar.

39 A técnica sprites melhora o tempo de carregamento das páginas web.

Prova

1 Considere a formatação padrão dos navegadores. O que podemos dizer sobre um **block-level element**?

a) Ocupa metade do espaço horizontal e não provoca quebras de linha.

b) Ocupa horizontalmente somente o espaço necessário para o seu conteúdo e não provocam

quebras de linha.

c) Ocupa todo o espaço vertical e provoca quebras de linha.

d) Ocupa horizontalmente somente o espaço necessário para o seu conteúdo e provocam quebras de linha.

e) Ocupa todo o espaço horizontal e provoca quebras de linha.

2 Considere a formatação padrão dos navegadores. O que podemos dizer sobre um **inline-level element**?

a) Ocupa metade do espaço horizontal e não provoca quebras de linha.

b) Ocupa horizontalmente somente o espaço necessário para o seu conteúdo e não provocam quebras de linha.

c) Ocupa todo o espaço vertical e provoca quebras de linha.

d) Ocupa horizontalmente somente o espaço necessário para o seu conteúdo e provocam quebras de linha.

e) Ocupa todo o espaço horizontal e provoca quebras de linha.

3 Como podemos aplicar um código CSS a um documento HTML?

a) Através do elemento **link** da seguinte forma:

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

b) Através do elemento **css** da seguinte forma:

```
<css>p {font-size. 12px}</css>
```

c) Através do elemento **style** da seguinte forma:

```
<style rel="stylesheet" type="text/css" src="estilo.css"></style>
```

d) Através do elemento **link** da seguinte forma:

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

e) Através do elemento **stylesheet** da seguinte forma:

```
<stylesheet src="estilo.css"></stylesheet>
```

4 Qual das alternativas é um valor inválido de cor CSS?

a) red

b) #00ff00

c) preto

d) rgba(0, 100, 30, 0.1)

e) hsl(100, 20%, 10%)

5 Qual das alternativas é uma unidade inválida de medida CSS?

a) px

b) em

c) rem

d) qm

e) pc

6 Qual das propriedades abaixo pode ser utilizada para definir uma imagem de fundo de um elemento?

a) background-attachment

b) background

c) bg-color

d) background-size

e) bg-image

7 Para definirmos a cor da fonte de um texto, qual propriedade CSS devemos utilizar?

a) color

b) font-color

c) text-color

d) foreground-color

e) colour

8 Considere o seguinte código HTML:

```
<div>Lorem          ipsum dolor sit amet, consectetur adipiscing elit.

Cras vehicula bibendum arcu.
Sed dui        nulla, cursus et lacinia eu,
vulputate ac dolor.
Quisque faucibus congue congue.</div>
```

O que acontecerá quando esse código for renderizado em um navegador?

a) Todas as sequências de espaços serão tratadas como um único espaço e as quebras de linha serão respeitadas.

b) Todas as sequências de espaços serão respeitadas e as quebras de linha serão ignoradas.

c) Todas as sequências de espaços serão respeitadas e as quebras de linha serão tratadas como um espaço.

d) Todas as sequências de espaços serão tratadas como um único espaço e as quebras de linha serão ignoradas.

e) Todas as sequências de espaços serão tratadas como um único espaço e as quebras de linha serão tratadas como um espaço.

9 Para alterar o comportamento padrão com relação ao tratamento dos espaços e quebras de linha, devemos utilizar qual propriedade CSS?

a) white-space

b) word-spacing

c) word-wrap

d) line-height

e) white-spacing

10 Podemos alterar a fonte de um texto através da propriedade CSS **font-family**. Quais valores podem ser atribuídos a essa propriedade?

a) O nome de uma fonte.

b) O nome de um grupo de fonte classificado pela espessura das fontes.

c) O nome de uma família de fonte.

d) O nome de uma fonte ou o nome de uma família de fonte.

e) O nome de um grupo de fonte classificado pelo tamanho das fontes.

11 Considere um elemento com as seguintes propriedades CSS:

```
width. 100px,
border. 10px solid black,
margin. 10px,
padding. 10px,
```

Ao ser renderizado em um navegador, o elemento terá:

a) 120px de largura.

b) 120px de altura.

c) 130px de largura.

d) 140px de altura.

e) 160px de largura.

12 Qual a diferença entre uma "borda" produzida pela propriedade **outline** e outra produzida pela propriedade **border**?

a) Nenhuma diferença.

b) A borda produzida pela propriedade border é mais grossa.

c) A borda produzida pela propriedade outline não pode ter sua cor alterada.

d) A borda produzida pela propriedade outline não afeta as dimensões do box do elemento.

e) A borda produzida pela propriedade border não funciona em todos os navegadores.

13 Podemos utilizar a propriedade **display** com valor **none** ou a propriedade **visibility** com valor **hidden** para fazer com que um elemento não apareça em uma página HTML. Qual a diferença entre as duas abordagens?

a) Nenhuma diferença.

b) O espaço do box de um elemento com propriedade **display** igual a **none** permanece inalterado.

c) O espaço do box de um elemento com propriedade **visibility** igual a **hidden** deixa de existir.

d) O espaço do box de um elemento com propriedade **display** igual a **none** deixa de existir.

e) Se o valor **none** for atribuído a propriedade **display**, ela não poderá mudar mais de valor.

14 Considere um elemento com as seguintes propriedades CSS:

```
position. absolute,
top. 0px,
left. 0px,
```

O que podemos dizer sobre esse elemento?

a) Será posicionado a 0px do topo e 0px da esquerda da página.

b) Será posicionado a 0px do topo e 0px da esquerda do elemento pai.

c) Será posicionado a 0px do topo e 0px da esquerda do primeiro elemento ancestral.

d) Será posicionado a 0px do topo e 0px da esquerda do primeiro elemento ancestral que tenha a propriedade **position** diferente de **static**.

e) Será posicionado a 0px do topo e 0px da esquerda do primeiro elemento ancestral que tenha a propriedade **position** diferente de **static**. Caso nenhum ancestral se encaixe nessa condição, ele será posicionado a 0px do topo e 0px da esquerda da página.

15 Considere o seguinte código:

```
<div style="width. 50px, height. 50px, font-size. 16px,
      overflow. hidden, border. 1px solid red">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed eget arcu neque.
  Cras at varius libero, vitae varius mauris. Nullam posuere eget nisi at
  posuere. Cras eu laoreet ipsum.
</div>
```

O que podemos dizer sobre o conteúdo do elemento **div**?

a) O trecho do conteúdo que ultrapassar na vertical e na horizontal os limites do elemento **div** será exibido.

b) O trecho do conteúdo que ultrapassar na vertical os limites do elemento **div** será exibido.

c) O trecho do conteúdo que ultrapassar na vertical e na horizontal os limites do elemento **div** não será exibido.

d) O trecho do conteúdo que ultrapassar na horizontal os limites do elemento **div** será exibido.

16 Considere o seguinte código:

```
<div id="div1" class="class1">
  <p id="p1">Lorem ipsum dolor sit amet.</p>
  <p id="p2" class="class1">Lorem ipsum dolor sit amet.</p>
  <div id="div2">
    Lorem ipsum dolor sit amet.
    <p id="p3" class="class1">Lorem ipsum dolor sit amet.</p>
    <div id="div3">
      Lorem ipsum dolor sit amet.
      <p id="p4" class="class1">Lorem ipsum dolor sit amet.</p>
    </div>
```

```
11   </div>
12   <div id="div4">
13     Lorem ipsum dolor sit amet.
14     <p id="p5" class="class1">Lorem ipsum dolor sit amet.</p>
15     <div id="div5">
16       Lorem ipsum dolor sit amet.
17       <p id="p6" class="class1">Lorem ipsum dolor sit amet.</p>
18     </div>
19   </div>
20   <p id="p7" class="class1">Lorem ipsum dolor sit amet.</p>
21   <a href="#" class="class1">Lorem ipsum dolor sit amet.</a>
22   <div id="div6">Lorem ipsum dolor sit amet.</div>
23 </div>
```

Considere também que o seguinte código CSS foi aplicado ao documento:

```
1 #div1 > p + div p.class1 {
2   color. red,
3 }
```

Quem ficará com texto vermelho?

a) Apenas o elemento com atributo **id** igual a **p5**.

b) Os elementos com atributo **id** igual a **p5** e **p6**.

c) Os elementos com atributo **id** igual a **p2**, **p5** e **p7**.

d) Apenas o elemento com atributo **id** igual a **p3**.

e) Os elementos com atributo **id** igual a **p3** e **p4**.

17 Considere o seguinte código:

```
1 @media all and (width. 480px), (orientation. portrait) {
2   ...
3 }
```

O que podemos dizer sobre as regras CSS definidas dentro da media query acima?

a) Serão aplicadas quando o viewport tiver 480px e a altura do viewport for maior que a sua largura.

b) Serão aplicadas apenas quando a largura do viewport for maior que a sua altura.

c) Serão aplicadas apenas quando o viewport tiver 480px de largura.

d) Serão aplicadas quando o viewport tiver 480px ou a altura do viewport for maior que a sua largura.

e) Serão aplicadas apenas quando o viewport tiver 480px de altura. **div**.

Minha Pontuação ☐ Pontuação Mínima: 13 Pontuação Máxima: 17

JAVASCRIPT

CAPÍTULO 4

Nos capítulos anteriores, vimos que as linguagens HTML e a CSS são fundamentais para a criação de páginas web. O foco do HTML é o conteúdo enquanto o foco do CSS é a formatação dessas páginas.

As linguagens HTML e CSS não são linguagens de programação. Para resolver determinados problemas, é necessário utilizar uma linguagem de programação. Por isso, veremos nesse capítulo, a linguagem de programação JavaScript.

Com a linguagem JavaScript podemos construir páginas extremamente dinâmicas e interativas. O foco do JavaScript é implementar o comportamento ou a inteligência das páginas web.

Aplicando JavaScript ao HTML

Há duas formas de associar código JavaScript e documentos HTML.

JavaScript interno

O código JavaScript pode ser definido dentro de um documento HTML no corpo do elemento **script**.

```
1 <script>
2   alert("K19"),
3 </script>
```

Nessa abordagem, o código JavaScript fica "amarrado" a um único documento HTML.

JavaScript externo

O código JavaScript pode ser definido em arquivos separados e depois associados aos documentos HTML através do elemento **script**.

```
1 alert("K19"),
```

Código Javascript 4.1: script.js

```
1 <script src="script.js"></script>
```

Nessa outra abordagem, podemos reutilizar o mesmo código JavaScript em vários documentos HTML.

Carregamento

Como vimos, o elemento **script** permite associar código JavaScript aos documentos HTML. Esse elemento pode ser adicionado dentro do corpo dos elementos **head** e **body**. A localização do elemento **script** afeta o momento no qual o código JavaScript será carregado pelos navegadores.

No exemplo a seguir, o elemento **script** foi colocado dentro do corpo do **head**. Dessa forma, o código JavaScript será carregado antes do **body** ser processado. Consequentemente, a página só será exibida depois do carregamento do JavaScript.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Desenvolvimento Web</title>
6      <script src="script.js"></script>
7    </head>
8    <body>
9      ...
10   </body>
11 </html>
```

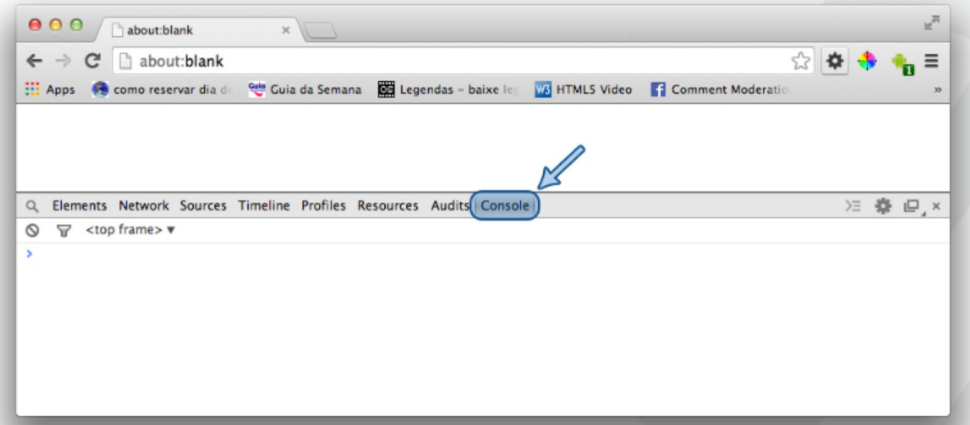
No próximo exemplo, o elemento **script** foi colocado no final do **body**. Consequentemente, o JavaScript só será carregado depois de todos os outros elementos do **body**.

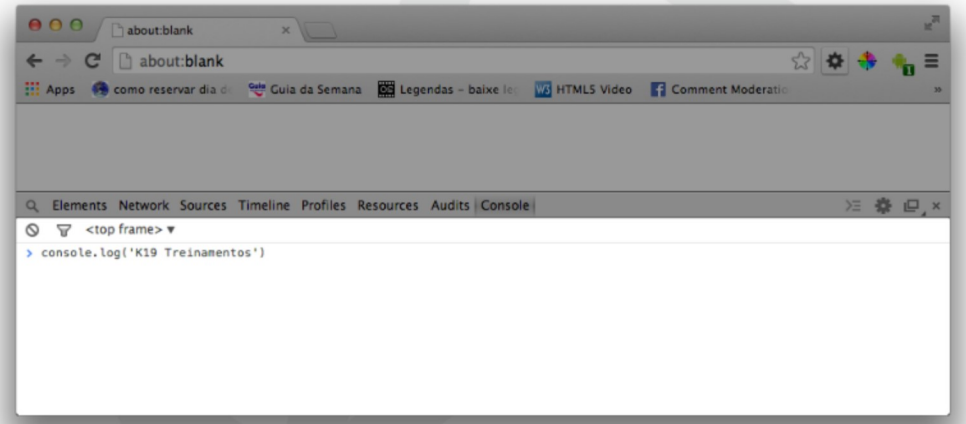
```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Desenvolvimento Web</title>
6    </head>
7    <body>
8      ...
9      <script src="script.js"></script>
10   </body>
11 </html>
```

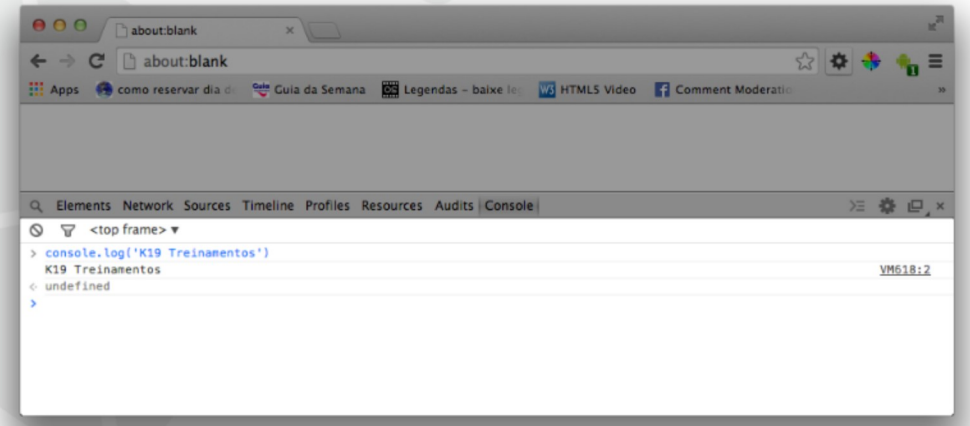
Normalmente, a segunda abordagem é mais recomendada pois as páginas web são exibidas mais rapidamente aos usuários. Contudo, eventualmente, o código JavaScript deve ser utilizado antes do processamento dos elementos do **body**. Nesses casos, é necessário utilizar a primeira abordagem.

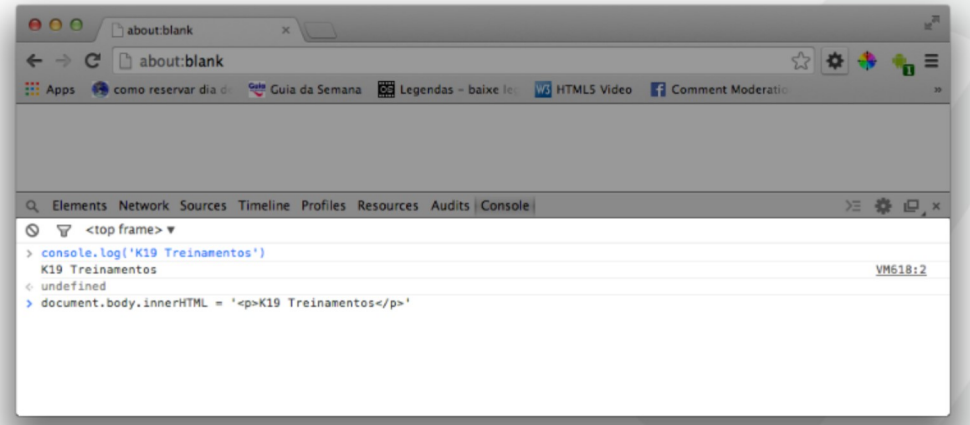
Chrome DevTools

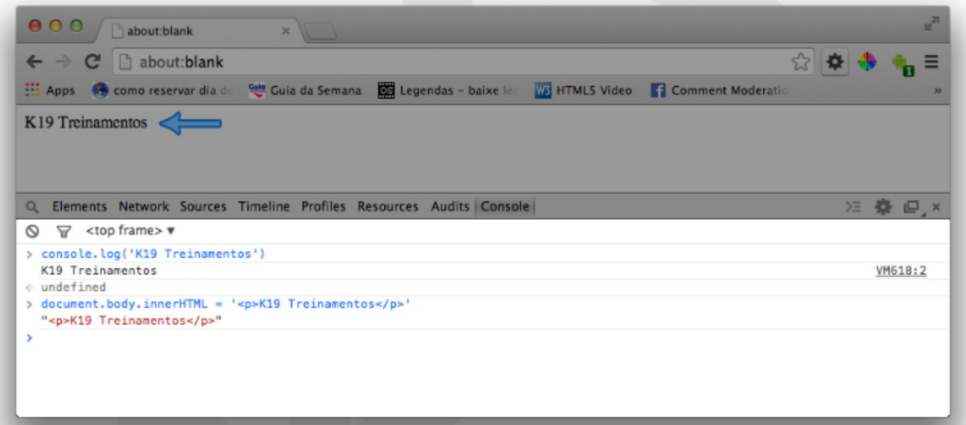
Os navegadores oferecem alguns recursos para testar ou depurar código JavaScript. Nas imagens abaixo, mostramos a utilização do painel **Console** do Chrome DevTools.


about:blank
about:blank
Apps
como reservar dia d
Guia da Semana
Legendas - baixe le
HTML5 Video
Comment Moderati
Elements Network Sources Timeline Profiles Resources Audits Console
<top frame>


about:blank
about:blank
Apps
como reservar dia d
Guia da Semana
Legendas - baixe le
HTML5 Video
Comment Moderati
Elements Network Sources Timeline Profiles Resources Audits Console
<top frame>
console.log('K19 Treinamentos')


about:blank
about:blank
Apps
como reservar dia d
Guia da Semana
Legendas - baixe le
HTML5 Video
Comment Moderati
Elements Network Sources Timeline Profiles Resources Audits Console
<top frame>
console.log('K19 Treinamentos')
K19 Treinamentos
VM618:2
undefined





Exercícios de Fixação

1. Abra o **Netbeans** e crie um projeto chamado **javascript**.

Importante
No **Windows**, utilizando o IIS (Internet Information Services) como Web Server, você deve salvar o projeto **javascript** em **C:\inetpub\wwwroot**. Lembre-se que é necessário instalar o IIS conforme vimos anteriormente.

Importante
No **Ubuntu**, utilizando o Apache HTTP Server como Web Server, você deve salvar o projeto **javascript** em **/home/<USUARIO>/public_html**. Lembre-se que é necessário instalar e configurar o Apache HTTP Server como vimos anteriormente.

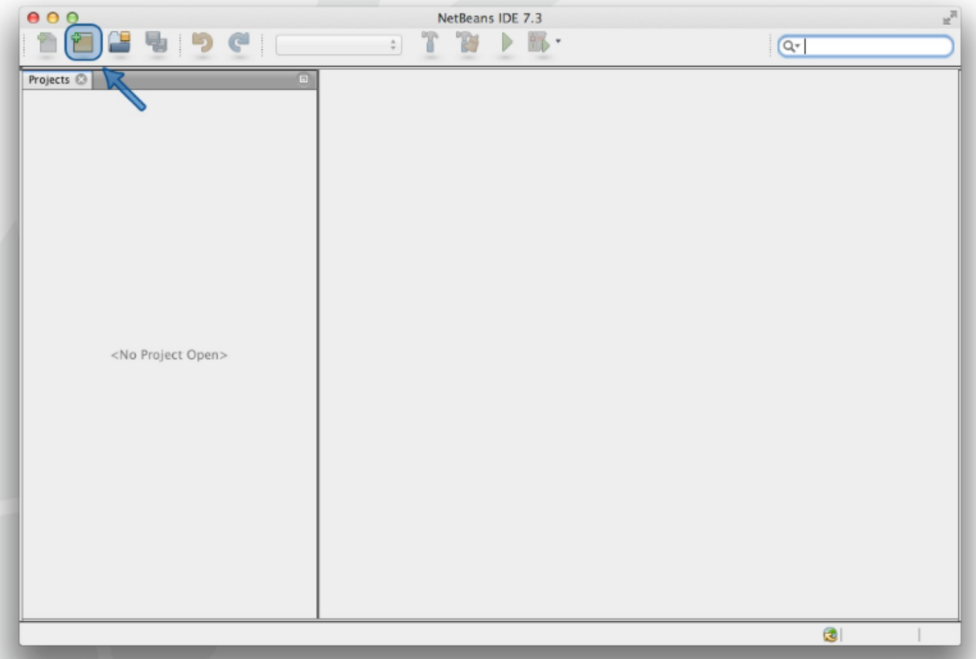


*Figura 4.1: Projeto **javascript***

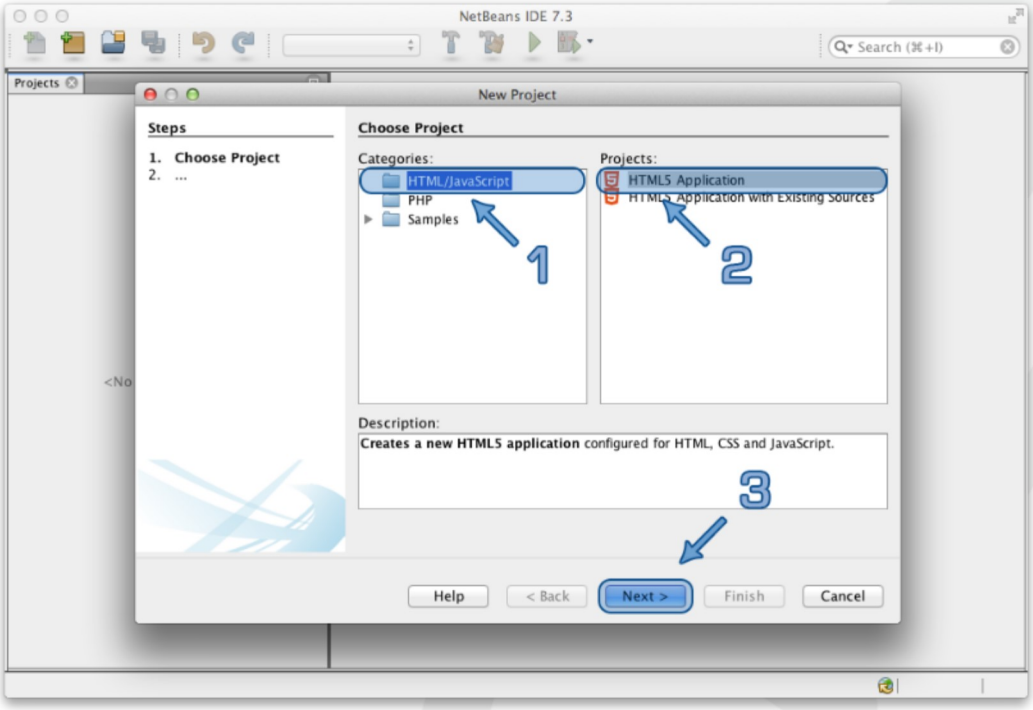


*Figura 4.2: Projeto **javascript***

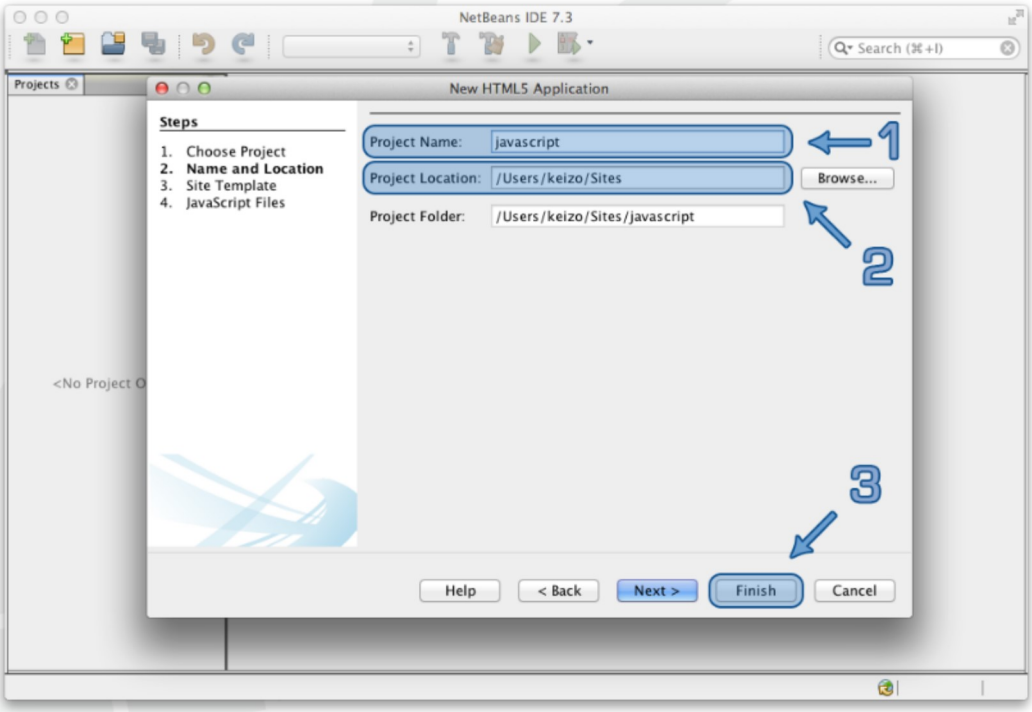


*Figura 4.3: Projeto **javascript***

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao1.zip

2 No projeto **javascript**, crie um arquivo chamado **javascript.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>JavaScript</title>
    <script>
      alert("K19"),
    </script>
  </head>
  <body>
    <h1>K19 - JavaScript</h1>
  </body>
</html>
```

Código HTML 4.5: javascript.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao2.zip

3 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/javascript.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/javascript.html.

4 Altere o arquivo **javascript.html** do projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>JavaScript</title>
  </head>
  <body>
    <h1>K19 - JavaScript</h1>
    <script>
      alert("K19"),
    </script>
  </body>
</html>
```

Código HTML 4.6: javascript.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao4.zip

5 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/javascript.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/javascript.html.

6 No projeto **javascript**, crie um arquivo chamado **script.js**.

```
alert("K19"),
```

Código Javascript 4.2: script.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao6.zip

7 Altere novamente o arquivo **javascript.html** do projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>JavaScript</title>
    <script src="script.js"></script>
  </head>
  <body>
    <h1>K19 - JavaScript</h1>
  </body>
</html>
```

Código HTML 4.7: javascript.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao7.zip

8 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/javascript.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/javascript.html.

Variáveis

Assim como qualquer linguagem de programação, JavaScript permite a criação de variáveis através da palavra chave **var**. Toda variável deve ter um nome (identificador).

```
var idadeDoJonas = 30,
var precoDoProduto = 28.75,
var nomeDoInstrutor = "Marcelo Martins",
var acessoLiberado = true,
```

No exemplo anterior, todas as variáveis foram inicializadas. Antes da inicialização, as variáveis possuem o valor especial **undefined**. No próximo exemplo, a variável **altura** não é inicializada. Portanto, ela possuíra o valor **undefined**.

```
var altura,
```

Operadores

Para manipular os valores das variáveis de um programa, devemos utilizar os operadores oferecidos pela linguagem de programação adotada. A linguagem JavaScript possui diversos operadores e os principais são categorizados da seguinte forma:

- Aritmético: **+ - * / %**
- Atribuição: **= += -= *= /= %= ++ --**
- Relacional: **== != < <= > >=**
- Lógico: **&& ||**

Aritméticos

Os operadores aritméticos funcionam de forma muito semelhante aos operadores da matemática. Os operadores aritméticos são:

- Adição **+**
- Subtração **-**
- Multiplicação *****
- Divisão **/**
- Módulo **%**

```
var umMaisUm = 1 + 1,
// umMaisUm = 2

var tresVezesDois = 3 * 2,
// tresVezesDois = 6

var quatroDivididoPorDois = 4 / 2,
// quatroDivididoPor2 = 2

var seisModuloCinco = 6 % 5,
// seisModuloCinco = 1

var x = 7,

x = x + 1 * 2,
// x = 9

x = x - 3,
// x = 6

x = x / (6 - 2 - (3 * 5) / (16 - 1)),
// x = 2
```

Código Javascript 4.5: Exemplo de uso dos operadores aritméticos

Importante
O módulo de um número x, na matemática, é o valor numérico de x desconsiderando o seu sinal (valor absoluto). Na matemática expressamos o módulo da seguinte forma: $|-2| = 2$.

Em linguagens de programação, o módulo de um número é o resto da divisão desse número por outro. No exemplo acima, o resto da divisão de 6 por 5 é igual a 1. Além disso, lemos a expressão 6%5 da seguinte forma: seis módulo cinco.

Importante
As operações aritméticas em JavaScript obedecem as mesmas regras da matemática com relação à precedência dos operadores e parênteses. Portanto, as operações são resolvidas a partir dos parênteses mais internos até os mais externos, primeiro resolvemos as multiplicações, divisões e os módulos. Em seguida, resolvemos as adições e subtrações.

Mais Sobre
As operações de potenciação e raiz quadrada podem ser realizadas através dos métodos **Math.pow** e **Math.sqrt**. Veja alguns exemplos.

```
1 var a = Math.pow(3, 5),
2 // a = 243
3
4 var b = Math.sqrt(9),
5 // b = 3
```

Código Javascript 4.6: Potenciação e raiz quadrada

Concatenação de Strings

Como vimos anteriormente, o operador + é utilizado para realizar soma aritmética. Mas, ele também pode ser utilizado para concatenar strings. Veja um exemplo.

```
1 var s1 = "Marcelo",
2 var s2 = " ",
3 var s3 = "Martins",
4
5 // "Marcelo Martins"
6 var s4 = s1 + s2 + s3,
```

Considere o exemplo a seguir.

```
1 var s1 = "Idade. ",
2 var idade = 30,
3
4 // "Idade. 30"
5 var s2 = s1 + idade,
```

Observe que o operador + foi aplicado a um valor numérico e a um texto. Nesses casos, o valor numérico é, automaticamente, transformado em texto e a concatenação é realizada.

Pare para pensar...
As expressões são avaliadas da esquerda para a direita. Dessa forma, considere o seguinte exemplo:

```
1 alert(1 + 2 + 3 + " testando"),
2 alert("testando" + 1 + 2 + 3),
```

O que seria exibido nesse exemplo?

Atribuição

Nas seções anteriores, já vimos um dos operadores de atribuição, o operador = (igual). Os operadores de atribuição são:

- Simples =
- Incremental +=
- Decremental -=
- Multiplicativa *=
- Divisória /=
- Modular %=
- Incremento ++
- Decremento --

```
var valor = 1,
// valor = 1

valor += 2,
// valor = 3

valor -= 1,
// valor = 2

valor *= 6,
// valor = 12

valor /= 3,
// valor = 4

valor %= 3,
// valor = 1

valor++,
// valor = 2

valor--,
// valor = 1
```

Código Javascript 4.10: Exemplo de uso dos operadores de atribuição.

As instruções acima poderiam ser escritas de outra forma:

```
var valor = 1,
// valor = 1

valor = valor + 2,
// valor = 3

valor = valor - 1,
// valor = 2

valor = valor * 6,
// valor = 12

valor = valor / 3,
// valor = 4

valor = valor % 3,
// valor = 1

```

```
19 valor = valor + 1,
20 // valor = 2
21
22 valor = valor - 1,
23 // valor = 1
```

Código Javascript 4.11: O mesmo exemplo anterior, usando os operadores aritméticos.

Como podemos observar, os operadores de atribuição, exceto o simples (=), reduzem a quantidade de código escrito. Podemos dizer que esses operadores funcionam como "atalhos" para as operações que utilizam os operadores aritméticos.

Relacionais

Muitas vezes precisamos determinar a relação entre uma variável ou valor e outra outra variável ou valor. Nessas situações, utilizamos os operadores relacionais. As operações realizadas com os operadores relacionais devolvem valores booleanos. Os operadores relacionais são:

- Igualdade ==
- Diferença !=
- Menor <
- Menor ou igual <=
- Maior >
- Maior ou igual >=

```
1 var valor = 2,
2 var t = false,
3 t = (valor == 2),   // t = true
4 t = (valor != 2),   // t = false
5 t = (valor < 2),    // t = false
6 t = (valor <= 2),   // t = true
7 t = (valor > 1),    // t = true
8 t = (valor >= 1),   // t = true
```

Código Javascript 4.12: Exemplo de uso dos operadores relacionais em JavaScript.

Lógicos

A linguagem JavaScript permite verificar duas ou mais condições através de operadores lógicos. Os operadores lógicos devolvem valores booleanos. A seguir descreveremos o funcionamento desses operadores.

- **&&** ("E")

```
1 var a = Math.random(),
2 var b = Math.random(),
3
4 alert(a > 0.2 && b < 0.8),
```

Código Javascript 4.13: Exemplo de uso do operador &

A **tabela verdade** é uma forma prática de visualizar o resultado dos operadores lógicos. Veja a seguir a tabela verdade do operador **&&**.

a > 0.2	b < 0.8	a > 0.2 && b < 0.8
V	V	V
V	F	F
F	V	F
F	F	F

Figura 4.4: Tabela verdade do operador **&&**

- || ("OU")

```
var a = Math.random(),
var b = Math.random(),

alert(a > 0.2 || b < 0.8),
```

Código Javascript 4.14: Exemplo de uso do operador ||

A **tabela verdade** é uma forma prática de visualizar o resultado dos operadores lógicos. Veja a seguir a tabela verdade do operador ||.

a > 0.2	b < 0.8	a > 0.2 || b < 0.8
V	V	V
V	F	V
F	V	V
F	F	F

Figura 4.5: Tabela verdade do operador ||

Operador "!"

Valores booleanos podem ser invertidos com o operador de "**!**" (negação). Por exemplo, podemos verificar se uma variável numérica armazena um valor maior do que 0.5 de duas formas diferentes.

```
d > 0.5
```

```
!(d <= 0.5)
```

Pré e Pós Incremento ou Pré e Pós Decremento

Os operadores "++" e "--" podem ser utilizados de duas formas diferentes, antes ou depois de uma variável numérica.

```
var i = 10,
i++,
```

```
3 i--,
```

```
1 var i = 10,
2 ++i,
3 --i,
```

No primeiro exemplo, o operador "++" foi utilizado depois da variável **i**. Já no segundo exemplo, ele foi utilizado antes da variável **i**. A primeira forma de utilizar o operador "++" é chamada de **pós incremento**. A segunda é chamada de **pré incremento**. Analogamente, o operador "--" foi utilizado na forma de **pós decremento** no primeiro exemplo e **pré decremento** no segundo exemplo.

Mas, qual é a diferença entre **pré incremento** e **pós incremento** ou entre **pré decremento** e **pós decremento**? Vamos apresentar a diferença com alguns exemplos.

```
1 var i = 10,
2
3 // true
4 alert(i++ == 10),
```

Observe que o operador "++" foi utilizado nas expressões do exemplo acima em conjunto com o operador "==". Como dois operadores foram utilizados na mesma expressão, você pode ter dúvida em relação a ordem de execução desses operadores. O incremento com o operador "++" será realizado antes ou depois da comparação com o operador "==”?

Como o operador "++" foi utilizado na forma de **pós incremento**, a comparação ocorrerá antes do incremento. Analogamente, a comparação ocorreria antes do decremento se o operador "--" fosse utilizado na forma de **pós decremento**.

Agora, considere a utilização do operador "++" na forma de **pré incremento**.

```
1 var i = 10,
2
3 // false
4 alert(++i == 10),
```

Nesse último exemplo, a comparação com o operador "==" é realizada depois do incremento do operador "++". Analogamente, a comparação ocorreria depois do decremento se o operador "--" fosse utilizado na forma de **pré decremento**.

Pare para pensar...

Considere o comportamento do **pré incremento**, **pós incremento**, **pré decremento** e **pós decremento**. O que seria exibido nos exemplos abaixo?

```
1 var i = 10,
2
3 var j = ++i + i--,
4
5 alert(j),
```

Operador ternário "?:"

Considere um programa que controla as notas dos alunos de uma escola. Para exemplificar, vamos gerar a nota de um aluno aleatoriamente.

```
1 var nota = Math.random(),
```

O programa deve exibir a mensagem "aprovado" se nota de um aluno for maior ou igual a 0.5 e "reprovado" se a nota for menor do que 0.5. Esse problema pode ser resolvido com o operador ternário.

Figura 4.6: Operador ternário

Quando a **condição**(**nota >= 0.5**) é verdadeira, o operador ternário devolve o **primeiro resultado** (**"aprovado"**). Caso contrário, devolve o **segundo resultado**(**"reprovado"**). Podemos guardar o resultado do operador ternário em uma variável ou simplesmente exibi-lo.

```
1 var resultado = nota >= 0.5 ? "aprovado" . "reprovado",
2 alert(nota >= 0.5 ? "aprovado" . "reprovado"),
```

Nos exemplos anteriores, o operador ternário foi utilizado com valores do tipo **string**. Contudo, podemos utilizá-lo com qualquer tipo de valor. Veja o exemplo a seguir.

```
1 var i = nota >= 0.5 ? 1 . 2,
2 var d = nota >= 0.5 ? 0.1 . 0.2,
```

Controle de fluxo

if e else

O comportamento de uma aplicação pode ser influenciado por valores definidos pelos usuários. Por exemplo, considere um sistema de cadastro de produtos. Se um usuário tenta adicionar um produto com preço negativo, a aplicação não deve cadastrar esse produto. Caso contrário, se o preço não for negativo, o cadastro pode ser realizado normalmente.

Outro exemplo, quando o pagamento de um boleto é realizado em uma agência bancária, o sistema do banco deve verificar a data de vencimento do boleto para aplicar ou não uma multa por atraso.

Para verificar uma determinada condição e decidir qual bloco de instruções deve ser executado, devemos aplicar o comando **if**.

```
1 if (preco < 0) {
2   alert('O preço do produto não pode ser negativo'),
```

```
3 } else {
4   alert('Produto cadastrado com sucesso'),
5 }
```

O comando **if** permite que valores booleanos sejam testados. Se o valor passado como parâmetro para o comando **if** for **true**, o bloco do **if** é executado. Caso contrário, o bloco do **else** é executado. O comando **else** assim como o seu bloco são opcionais.

while

Em alguns casos, é necessário repetir um determinado trecho de código várias vezes. Por exemplo, suponha que seja necessário exibir 5 vezes a mensagem: "Bom Dia". Podemos resolver essa tarefa com o seguinte código JavaScript.

```
1 console.log("Bom Dia"),
2 console.log("Bom Dia"),
3 console.log("Bom Dia"),
4 console.log("Bom Dia"),
5 console.log("Bom Dia"),
```

Se ao invés de 5 vezes fosse necessário exibir 100 vezes a mensagem "Bom Dia", já seriam 100 linhas de código JavaScript. É muito trabalhoso utilizar essa abordagem para solucionar esse problema.

Através do comando **while**, é possível executar várias vezes um determinado trecho de código.

```
1 var contador = 0,
2
3 while(contador < 100) {
4   console.log("Bom Dia"),
5   contador++,
6 }
```

Código Javascript 4.27: while

A variável **contador** indica o número de vezes que a mensagem "Bom Dia" foi exibida. O operador **++** incrementa a variável **contador** a cada iteração. O parâmetro do comando **while** deve ser um valor booleano.

for

O comando **for** é análogo ao **while**. A principal diferença entre esses dois comandos é que o **for** recebe três argumentos.

```
1 for(var contador = 0, contador < 100, contador++) {
2   console.log("Bom Dia"),
3 }
```

Exercícios de Fixação

9 Adicione um arquivo chamado **exibe-nome.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="exibe-nome.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.8: exibe-nome.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao9.zip

10 Adicione um arquivo chamado **exibe-nome.js** no projeto **javascript**.

```
for(var contador = 0, contador < 100, contador++) {
  console.log("Rafael Cosentino"),
}
```

Código Javascript 4.29: exibe-nome.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao10.zip

11 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/exibe-nome.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/exibe-nome.html.

Verifique as mensagens exibidas no console do navegador.

12 Adicione um arquivo chamado **exibe-1-ate-100.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="exibe-1-ate-100.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.9: exibe-1-ate-100.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao12.zip

13 Adicione um arquivo chamado **exibe-1-ate-100.js** no projeto **javascript**.

```
for(var contador = 0, contador < 100, contador++) {
  console.log(contador),
```

```
}
```

Código Javascript 4.30: exibe-1-ate-100.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao13.zip

14 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/exibe-1-ate-100.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/exibe-1-ate-100.html.

Verifique as mensagens exibidas no console do navegador.

15 Percorra todos os números de 1 até 100. Para os números ímpares, exiba no console um "*", e para os números pares, dois "**". Veja o exemplo abaixo:

```
*
**
*
**
*
**
```

Adicione um arquivo chamado **exibe-padrao-1.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="exibe-padrao-1.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.10: exibe-padrao-1.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao15.zip

16 Adicione um arquivo chamado **exibe-padrao-1.js** no projeto **javascript**.

```
for(var contador = 1, contador <= 100, contador++) {
  var resto = contador % 2,
  if(resto == 1) {
    console.log("*"),
  } else {
    console.log("**"),
  }
}
```

Código Javascript 4.31: exibe-padrao-1.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao16.zip

17 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/exibe-padrao-1.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/exibe-padrao-1.html.

Verifique as mensagens exibidas no console do navegador.

18 Percorra todos os números de 1 até 100. Para os números múltiplos de 4, exiba a palavra "PIN", e para os outros, exiba o próprio número. Veja o exemplo abaixo:

```
1
2
3
PIN
5
6
7
PIN
```

Adicione um arquivo chamado **exibe-padrao-2.html** no projeto **javascript**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>K19 - JavaScript</title>
6      <script type="text/javascript" src="exibe-padrao-2.js"></script>
7    </head>
8    <body>
9    </body>
10 </html>
```

Código HTML 4.11: exibe-padrao-2.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao18.zip

19 Adicione um arquivo chamado **exibe-padrao-2.js** no projeto **javascript**.

```
1 for(var contador = 1, contador <= 100, contador++) {
2   var resto = contador % 4,
3   if(resto == 0) {
4     console.log("PI"),
5   } else {
6     console.log(contador),
```

```
  }
}
```

Código Javascript 4.32: exibe-padrao-2.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao19.zip

20 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/exibe-padrao-2.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/exibe-padrao-2.html.

Verifique as mensagens exibidas no console do navegador.

Objetos

Um objeto é um conjunto de propriedades. Toda propriedade possui nome e valor. O nome de uma propriedade pode ser qualquer sequência de caracteres. O valor de uma propriedade pode ser qualquer valor exceto **undefined**. Podemos adicionar uma nova propriedade a um objeto que já existe. Um objeto pode herdar propriedades de outro objeto utilizando a ideia de **prototype**.

Criando objetos

Um objeto pode ser criado de forma literal com a sintaxe **JSON**. Veja o exemplo a seguir.

```
var objetoVazio = {},
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
},
```

Um objeto pode se relacionar com outros objetos através de propriedades. Observe o código abaixo.

```
var formacaoJava = {
  sigla. "K10", nome. "Formação Desenvolvedor Java",
  cursos. [
    {
      sigla. "K11",
      nome. "Orientação a Objetos em Java"
    },
    {
      sigla. "K12",
      nome. "Desenvolvimento Web com JSF2 e JPA2"
    },
  ]
},
```

Recuperando o valor de uma propriedade

Para recuperar os valores das propriedades de um objeto, podemos utilizar o operador "." ou "[]". Veja o exemplo a seguir.

```
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
},

console.log(curso.sigla),
console.log(curso["sigla"]),

var sigla = "sigla",
console.log(curso[sigla]),
```

Alterando o valor de uma propriedade

Para alterar o valor de uma propriedade, basta atribuir um novo valor à propriedade do objeto.

```
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
},

curso.sigla = "K12",
curso.nome = "Desenvolvimento Web com JSF2 e JPA2",

console.log(curso.sigla),
console.log(curso.nome),
```

Referências

Os objetos são acessados através de referências.

```
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
},

// copiando uma referência
var x = curso,

x.sigla = "K12",
x.nome = "Desenvolvimento Web com JSF2 e JPA2",

// imprime K12
console.log(curso.sigla),

// imprime Desenvolvimento Web com JSF2 e JPA2
console.log(curso.nome),
```

Protótipos

Podemos criar um objeto baseado em outro objeto existente (protótipo). Para isso, podemos utilizar a propriedade especial **__proto__**. Observe o código abaixo.

```
// criando um objeto com duas propriedades
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
```

```
5  },
6
7  // criando um objeto sem propriedades
8  var novo_curso = {},
9
10 // definindo o primeiro objeto como protótipo do segundo
11 novo_curso.__proto__ = curso,
12
13 // imprime K11
14 console.log(novo_curso.sigla),
15
16 // imprime Orientação a Objetos em Java
17 console.log(novo_curso.nome),
```

Também podemos utilizar o método **create** de **Object** para criar objetos baseados em objetos existentes. Veja o exemplo abaixo.

```
1  // criando um objeto com duas propriedades
2  var curso = {
3    sigla. "K11",
4    nome. "Orientação a Objetos em Java"
5  },
6
7  // criando um objeto sem propriedades
8  var novo_curso = {},
9
10 // definindo o primeiro objeto como protótipo do segundo
11 novo_curso = Object.create(curso),
12
13 // imprime K11
14 console.log(novo_curso.sigla),
15
16 // imprime Orientação a Objetos em Java
17 console.log(novo_curso.nome),
```

Se uma propriedade for adicionada a um objeto, ela também será adicionada a todos os objetos que o utilizam como protótipo.

```
1  var curso = {
2    sigla. "K11",
3    nome. "Orientação a Objetos em Java"
4  },
5
6  var novo_curso = Object.create(curso),
7
8  curso.carga_horaria = 36,
9
10 // imprime K11
11 console.log(novo_curso.sigla),
12
13 // imprime Orientação a Objetos em Java
14 console.log(novo_curso.nome),
15
16 // imprime 36
17 console.log(novo_curso.carga_horaria),
```

Por outro lado, se uma propriedade for adicionada a um objeto, ela não será adicionada no protótipo desse objeto.

```
1  var curso = {
2    sigla. "K11",
3    nome. "Orientação a Objetos em Java"
4  },
5
```

```
6  var novo_curso = Object.create(curso),
7
8  novo_curso.carga_horaria = 36,
9
10 // imprime K11
11 console.log(curso.sigla),
12
13 // imprime Orientação a Objetos em Java
14 console.log(curso.nome),
15
16 // imprime undefined
17 console.log(curso.carga_horaria),
```

Se o valor de uma propriedade de um objeto for modificado, os objetos que o utilizam como protótipo podem ser afetados.

```
1  var curso = {
2    sigla. "K11",
3    nome. "Orientação a Objetos em Java"
4  },
5
6  var novo_curso = Object.create(curso),
7
8  curso.sigla = "K12",
9  curso.nome = "Desenvolvimento Web com JSF2 e JPA2",
10
11 // imprime K12
12 console.log(novo_curso.sigla),
13
14 // imprime Desenvolvimento Web com JSF2 e JPA2
15 console.log(novo_curso.nome),
```

Por outro lado, alterações nos valores das propriedades de um objeto não afetam o protótipo desse objeto.

```
1  var curso = {
2    sigla. "K11",
3    nome. "Orientação a Objetos em Java"
4  },
5
6  var novo_curso = Object.create(curso),
7
8  novo_curso.sigla = "K12",
9  novo_curso.nome = "Desenvolvimento Web com JSF2 e JPA2",
10
11 // imprime K11
12 console.log(curso.sigla),
13
14 // imprime Orientação a Objetos em Java
15 console.log(curso.nome),
```

Considere um objeto que foi construído a partir de um protótipo. Se o valor de uma propriedade herdada do protótipo for alterada nesse objeto, ela se torna independente da propriedade no protótipo. Dessa forma, alterações no valor dessa propriedade no protótipo não afetam mais o valor dela no objeto gerado a partir do protótipo.

```
1  var curso = {
2    sigla. "K11",
3    nome. "Orientação a Objetos em Java"
4  },
5
6  var novo_curso = Object.create(curso),
7
```

```
novo_curso.sigla = "K12",
novo_curso.nome = "Desenvolvimento Web com JSF2 e JPA2",

curso.sigla = "K21",
curso.nome = "Persistência com JPA2 e Hibernate",

// imprime K12
console.log(novo_curso.sigla),

// imprime Desenvolvimento Web com JSF2 e JPA2
console.log(novo_curso.nome),
```

Removendo uma Propriedade

Podemos remover uma propriedade de um objeto com a função **delete**.

```
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
},

// imprime K11
console.log(curso.sigla),

delete curso.sigla,

// imprime undefined
console.log(curso.sigla),
```

Verificando a Existência de uma Propriedade

Podemos verificar se uma propriedade existe, podemos utilizar a função **in**.

```
var curso = {
  sigla. "K11",
  nome. "Orientação a Objetos em Java"
},

// imprime true
console.log("sigla" in curso),

// imprime false
console.log("carga_horaria" in curso),
```

Exercícios de Fixação

21 Adicione um arquivo chamado **objetos.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="objetos.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.12: objetos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao21.zip

22 Adicione um arquivo chamado **objetos.js** no projeto **javascript**. Crie objetos com propriedades chamadas **sigla** e **nome**. Exiba o valor dessas propriedades no console do navegador.

```
var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},
console.log(curso.sigla),
console.log(curso.nome),

var curso2 = {sigla. "K12", nome. "Desenvolvimento Web com JSF2 e JPA2"},
console.log(curso2.sigla),
console.log(curso2.nome),
```

Código Javascript 4.47: objetos.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao22.zip

23 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/objetos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/objetos.html.

Verifique as mensagens exibidas no console do navegador.

24 Adicione um arquivo chamado **referencias.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="referencias.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.13: referencias.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao24.zip

25 Adicione um arquivo chamado **referencias.js** no projeto **javascript**.

```
var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},

// exibe K11
console.log(curso.sigla),

// exibe Orientação a Objetos em Java
console.log(curso.nome),
```

```
8
9  var x = curso,
10
11 x.sigla = "K12",
12 x.nome = "Desenvolvimento Web com JSF2 e JPA2",
13
14 // exibe K12
15 console.log(curso.sigla),
16
17 // exibe Desenvolvimento Web com JSF2 e JPA2
18 console.log(curso.nome),
```

Código Javascript 4.48: referencias.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao25.zip

26 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/referencias.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/referencias.html.

Verifique as mensagens exibidas no console do navegador.

27 Adicione um arquivo chamado **prototype.html** no projeto **javascript**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>K19 - JavaScript</title>
6      <script type="text/javascript" src="prototype.js"></script>
7    </head>
8    <body>
9    </body>
10 </html>
```

Código HTML 4.14: prototype.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao27.zip

28 Adicione um arquivo chamado **prototype.js** no projeto **javascript**. Crie um objeto a partir de outro objeto existente.

```
1  var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},
2
3  var novo_curso = Object.create(curso),
4
5  // exibe K11
6  console.log(novo_curso.sigla),
7
8  // exibe Orientação a Objetos em Java
9  console.log(novo_curso.nome),
```

Código Javascript 4.49: prototype.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao28.zip

29 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/prototype.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/prototype.html.

Verifique as mensagens exibidas no console do navegador.

30 Adicione um arquivo chamado **propriedade-1.html** no projeto **javascript**. Defina uma propriedade em um objeto utilizado como protótipo e verifique que essa propriedade será adicionada nos objetos criados a partir desse protótipo.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="propriedade-1.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.15: propriedade-1.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao30.zip

31 Adicione um arquivo chamado **propriedade-1.js** no projeto **javascript**.

```
var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},

var novo_curso = Object.create(curso),

curso.carga_horaria = 36,

// exibe K11
console.log(novo_curso.sigla),

// exibe Orientação a Objetos em Java
console.log(novo_curso.nome),

// exibe 36
console.log(novo_curso.carga_horaria),
```

Código Javascript 4.50: propriedade-1.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao31.zip

32 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/propriedade-1.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/propriedade-1.html.

Verifique as mensagens exibidas no console do navegador.

33 Adicione um arquivo chamado **propriedade-2.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="propriedade-2.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.16: propriedade-2.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao33.zip

34 Adicione um arquivo chamado **propriedade-2.js** no projeto **javascript**. Defina uma propriedade em um objeto e verifique que o protótipo desse objeto não é afetado.

```
var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},

var novo_curso = Object.create(curso),

novo_curso.carga_horaria = 36,

// exibe K11
console.log(curso.sigla),

// exibe Orientação a Objetos em Java
console.log(curso.nome),

// exibe undefined
console.log(curso.carga_horaria),
```

Código Javascript 4.51: propriedade-2.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao34.zip

35 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/propriedade-2.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/propriedade-2.html.

Verifique as mensagens exibidas no console do navegador.

36 Adicione um arquivo chamado **propriedade-3.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
```

```
5     <title>K19 - JavaScript</title>
6     <script type="text/javascript" src="propriedade-3.js"></script>
7   </head>
8   <body>
9   </body>
10 </html>
```

Código HTML 4.17: propriedade-3.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao36.zip

37 Adicione um arquivo chamado **propriedade-3.js** no projeto **javascript**. Altere o valor de uma propriedade de um objeto utilizado como protótipo e verifique que essa alteração afetará os objetos criados a partir desse protótipo.

```
1  var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},
2
3  var novo_curso = Object.create(curso),
4
5  // exibe K11
6  console.log(novo_curso.sigla),
7
8  // exibe Orientação a Objetos em Java
9  console.log(novo_curso.nome),
10
11 curso.sigla = "K12",
12 curso.nome = "Desenvolvimento Web com JSF2 e JPA2",
13
14 // exibe K12
15 console.log(novo_curso.sigla),
16
17 // exibe Desenvolvimento Web com JSF2 e JPA2
18 console.log(novo_curso.nome),
```

Código Javascript 4.52: propriedade-3.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao37.zip

38 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/propriedade-3.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/propriedade-3.html.

Verifique as mensagens exibidas no console do navegador.

39 Adicione um arquivo chamado **propriedade-4.html** no projeto **javascript**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>K19 - JavaScript</title>
6     <script type="text/javascript" src="propriedade-4.js"></script>
7   </head>
8   <body>
9   </body>
```

```
10 </html>
```

Código HTML 4.18: propriedade-4.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao39.zip

40 Adicione um arquivo chamado **propriedade-4.js** no projeto **javascript**. Reescreva em um objeto as propriedades herdadas de um protótipo e verifique que alterações nos valores dessas propriedades no protótipo não afetam mais os valores delas nesse objeto.

```
1  var curso = {sigla. "K11", nome. "Orientação a Objetos em Java"},
2
3  var novo_curso = Object.create(curso),
4
5  novo_curso.sigla = "K12",
6  novo_curso.nome = "Desenvolvimento Web com JSF2 e JPA2",
7
8  // exibe K12
9  console.log(novo_curso.sigla),
10
11 // exibe Desenvolvimento Web com JSF2 e JPA2
12 console.log(novo_curso.nome),
13
14 curso.sigla = "K21",
15 curso.nome = "Persistência com JPA2 e Hibernate",
16
17 // exibe K12
18 console.log(novo_curso.sigla),
19
20 // exibe Desenvolvimento Web com JSF2 e JPA2
21 console.log(novo_curso.nome),
```

Código Javascript 4.53: propriedade-4.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao40.zip

41 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/propriedade-4.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/propriedade-4.html.

Verifique as mensagens exibidas no console do navegador.

Funções

As funções em JavaScript são objetos. Você pode armazená-las em variáveis, arrays e outros objetos. Elas podem ser passadas como argumento ou devolvidas por outra função. Veja o exemplo abaixo.

```
1 var multiplicacao = function(x, y) {
2   return x * y,
3 }
```

Código Javascript 4.54: Uma forma de criar uma função

```
function multiplicacao(x, y) {
  return x * y,
}
```

Código Javascript 4.55: Outra forma de criar uma função

Utilizando uma Função

Para utilizar a função **multiplicacao**, podemos chamá-la da seguinte forma.

```
var resultado = multiplicação(3,2),
```

Código Javascript 4.56: Utilizando a função

Método

Quando uma função faz parte de um objeto, ela é chamada de método. Para executar um método, devemos utilizar a referência de um objeto e passar os parâmetros necessários. Observe o código abaixo.

```
var conta = {
  saldo. 0,
  deposita. function(valor) {
    this.saldo += valor,
  }
}

conta.deposita(100),
console.log(conta.saldo),
```

Código Javascript 4.57: Método

Apply

Uma função pode ser associada momentaneamente a um objeto e executada através do método **apply**.

```
var deposita = function(valor) {
  this.saldo += valor,
}

var conta = {
  saldo. 0
}

deposita.apply(conta, [200]),
console.log(conta.saldo),
```

Código Javascript 4.58: Método apply

Arguments

Os argumentos passados na chamada de uma função podem ser recuperados através do array **Arguments**. Inclusive, esse array permite que os argumentos excedentes sejam acessados.

```
var soma = function() {
  var soma = 0,

  for(var i = 0, i < arguments.length, i ++) {
    soma += arguments[i],
  }

  return soma,
}

var resultado = soma(2,4,5,6,1),

console.log(resultado),
```

Código Javascript 4.59: Arguments

Exercícios de Fixação

42 Adicione um arquivo chamado **multiplicacao.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="multiplicacao.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.19: multiplicacao.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao42.zip

43 Adicione um arquivo chamado **multiplicacao.js** no projeto **javascript**.

```
var multiplicacao = function(x, y) {
  return x * y,
},

var resultado = multiplicacao(5, 3),

console.log(resultado),
```

Código Javascript 4.60: multiplicacao.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao43.zip

44 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/multiplicacao.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/multiplicacao.html.

Verifique as mensagens exibidas no console do navegador.

45 Adicione um arquivo chamado **metodos.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="metodos.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.20: metodos.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao45.zip

46 Adicione um arquivo chamado **metodos.js** no projeto **javascript**.

```
var conta = {
  saldo. 0,
  deposita. function(valor) {
    this.saldo += valor,
  }
},

conta.deposita(500),

console.log(conta.saldo),
```

Código Javascript 4.61: metodos.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao46.zip

47 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/metodos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/metodos.html.

Verifique as mensagens exibidas no console do navegador.

48 Adicione um arquivo chamado **soma.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="soma.js"></script>
  </head>
```

```
  <body>
  </body>
</html>
```

Código HTML 4.21: soma.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao48.zip

49 Adicione um arquivo chamado **soma.js** no projeto **javascript**.

```
var soma = function() {
  var soma = 0,

  for(var i = 0, i < arguments.length, i ++) {
    soma += arguments[i],
  }

  return soma,
},

var resultado = soma(2,4,5,6,1),

console.log(resultado),
```

Código Javascript 4.62: soma.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao49.zip

50 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/soma.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/soma.html.

Verifique as mensagens exibidas no console do navegador.

Arrays

A linguagem Javascript provê um objeto com características semelhantes a um array. Esse tipo de objeto pode ser criado de forma literal com a sintaxe JSON.

```
var vazio = [],
var cursos = ["K01","K02","K03"],

console.log(vazio[0]),
// undefined

console.log(cursos[0]),
// K01

console.log(vazio.length),
// 0

console.log(cursos.length),
// 3
```

Percorrendo um Array

Para percorrer um array, podemos utilizar o comando **for**.

```
var cursos = ["K01","K02","K03"],

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Adicionando Elementos

Para adicionar um elemento no final de um array, podemos utilizar a propriedade **length**.

```
var cursos = ["K01","K02"],

cursos[cursos.length] = "K03",
// ["K01", "K02", "K03"]

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Também podemos utilizar o método **push()**.

```
var cursos = ["K01","K02"],

cursos.push("K03"),
// ["K01", "K02", "K03"]

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Removendo Elementos

O método **delete()** permite remover elementos de um array.

```
var cursos = ["K01", "K02", "K03"],

delete cursos[0],
// [undefined, "K02", "K03"]

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Concatenando Arrays

O método **concat()** permite concatenar dois arrays.

```
var formacaoBasica = ["K01", "K02", "K03"],
var formacaoJava = ["K11", "K12"],

var formacaoCompleta = formacaoBasica.concat(formacaoJava),
// ["K01", "K02", "K03", "K11", "K12"]

for(var i = 0, i < formacaoCompleta.length, i++) {
  console.log(formacaoCompleta[i]),
}
```

Gerando uma String com os Elementos de um Array

O método **join()** cria uma string a partir de um array.

```
var formacaoJava = ["K11", "K12"],

var resultado = formacaoJava.join(","),
// K11,K12

console.log(resultado),
```

Removendo o Último Elemento

O método **pop()** remove e retorna o último elemento.

```
var cursos = ["K01", "K02", "K03"],

var curso = cursos.pop(),
// ["K01", "K02"]

console.log("Elemento removido. " + curso),

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Invertendo os Elementos de um Array

O método **reverse()** inverte a ordem dos elementos de um array.

```
var cursos = ["K01", "K02", "K03"],

cursos.reverse(),
// ["K03", "K02", "K01"]

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Removendo o Primeiro Elemento

O método **shift()** remove e retorna o primeiro elemento de um array.

```
var cursos = ["K01", "K02", "K03"],

var curso = cursos.shift(),
// ["K02", "K03"]

console.log("Elemento removido. " + curso),

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Copiando um Trecho de um Array

O método **slice()** cria uma cópia de uma porção de um array.

```
var cursos = ["K01", "K02", "K03", "K11", "K12"],

```

```
var formacaoBasica = cursos.slice(0, 3),
// ["K01", "K02", "K03"]

for(var i = 0, i < formacaoBasica.length, i++) {
  console.log(formacaoBasica[i]),
}
```

Removendo e Adicionando Elementos em um Array

O método **splice()** permite remover elementos do array e adicionar novos elementos.

```
var cursos = ["K11","K12","K21","K22","K23"],

cursos.splice(2, 3, "K31", "K32"),
// ["K11", "K12", "K31", "K32"]

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Adicionando um Elemento na Primeira Posição

O método **unshift()** adiciona elementos na primeira posição de um array.

```
var cursos = ["K12","K21","K22","K23"],

cursos.unshift("K11"),
// ["K11", "K12", "K21", "K22", "K23"]

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Strings

Acessando os Caracteres de uma String por Posição

O método **charAt()** retorna o caractere na posição especificada.

```
var curso = "K12",

console.log(curso.charAt(0)),
```

Recuperando um Trecho de uma String

O método **slice()** retorna uma porção de uma string.

```
var curso = "K12 - Desenvolvimento Web com JSF2 e JPA2",

console.log(curso.slice(0,3)),
```

Dividindo uma String

O método **split()** cria uma array de strings a partir de um separador.

```
var curso = "K12-Desenvolvimento Web com JSF2 e JPA2",
var aux = curso.split("-"),

console.log(aux[0]),
console.log(aux[1]),
```

Exercícios de Fixação

51 Adicione um arquivo chamado **array-length.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-length.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.22: array-length.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao51.zip

52 Adicione um arquivo chamado **array-length.js** no projeto **javascript**.

```
var vazio = [],
var cursos = ["K11","K12","K21","K22","K23","K31","K32"],

console.log(vazio[0]),
console.log(cursos[0]),

console.log(vazio.length),
console.log(cursos.length),
```

Código Javascript 4.79: array-length.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao52.zip

53 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/array-length.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/array-length.html.

Verifique as mensagens exibidas no console do navegador.

54 Adicione um arquivo chamado **array-elementos.html** no projeto **javascript**.

```
<!DOCTYPE html>
```

```
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>K19 - JavaScript</title>
6     <script type="text/javascript" src="array-elementos.js"></script>
7   </head>
8   <body>
9   </body>
10 </html>
```

Código HTML 4.23: array-elementos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao54.zip

55 Adicione um arquivo chamado **array-elementos.js** no projeto **javascript**.

```
1 var cursos = ["K11","K12","K21","K22","K23","K31","K32"],
2 for(var i = 0, i < cursos.length, i++) {
3   console.log(cursos[i]),
4 }
```

Código Javascript 4.80: array-elementos.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao55.zip

56 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-elementos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/array-elementos.html.

Verifique as mensagens exibidas no console do navegador.

57 Adicione um arquivo chamado **array-add.html** no projeto **javascript**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>K19 - JavaScript</title>
6     <script type="text/javascript" src="array-add.js"></script>
7   </head>
8   <body>
9   </body>
10 </html>
```

Código HTML 4.24: array-add.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao57.zip

58 Adicione um arquivo chamado **array-add.js** no projeto **javascript**.

```
1 var cursos = ["K11","K12","K21","K22","K23","K31","K32"],
2
3 cursos[cursos.length] = "K01",
```

```
4
5 for(var i = 0, i < cursos.length, i++) {
6   console.log(cursos[i]),
7 }
```

Código Javascript 4.81: array-add.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao58.zip

59 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-add.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/array-add.html.

Verifique as mensagens exibidas no console do navegador.

60 Adicione um arquivo chamado **array-push.html** no projeto **javascript**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>K19 - JavaScript</title>
6      <script type="text/javascript" src="array-push.js"></script>
7    </head>
8    <body>
9    </body>
10 </html>
```

Código HTML 4.25: array-push.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao60.zip

61 Adicione um arquivo chamado **array-push.js** no projeto **javascript**.

```
1 var cursos = ["K11","K12","K21","K22","K23","K31","K32"],
2
3 cursos.push("K01"),
4
5 for(var i = 0, i < cursos.length, i++) {
6   console.log(cursos[i]),
7 }
```

Código Javascript 4.82: array-push.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao61.zip

62 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-push.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/array-push.html.

Verifique as mensagens exibidas no console do navegador.

63 Adicione um arquivo chamado **array-concat.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-concat.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.26: array-concat.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao63.zip

64 Adicione um arquivo chamado **array-concat.js** no projeto **javascript**.

```
var formacaoJava = ["K11","K12"],
var formacaoJavaAvancado = ["K21","K22","K23"],

var formacaoCompleta = formacaoJava.concat(formacaoJavaAvancado),

for(var i = 0, i < formacaoCompleta.length, i++) {
  console.log(formacaoCompleta[i]),
}
```

Código Javascript 4.83: array-concat.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao64.zip

65 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-concat.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/array-concat.html.

Verifique as mensagens exibidas no console do navegador.

66 Adicione um arquivo chamado **array-pop.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-pop.js"></script>
  </head>
  <body>
  </body>
```

```
10 </html>
```

Código HTML 4.27: array-pop.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao66.zip

67 Adicione um arquivo chamado **array-pop.js** no projeto **javascript**.

```
1 var cursos = ["K11","K12","K21","K22","K23"],
2
3 var curso = cursos.pop(),
4
5 console.log(curso),
```

Código Javascript 4.84: array-pop.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao67.zip

68 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-pop.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/array-pop.html.

Verifique as mensagens exibidas no console do navegador.

69 Adicione um arquivo chamado **array-reverse.html** no projeto **javascript**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>K19 - JavaScript</title>
6     <script type="text/javascript" src="array-reverse.js"></script>
7   </head>
8   <body>
9   </body>
10 </html>
```

Código HTML 4.28: array-reverse.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao69.zip

70 Adicione um arquivo chamado **array-reverse.js** no projeto **javascript**.

```
1 var cursos = ["K11","K12","K21","K22","K23"],
2
3 cursos.reverse(),
4
5 for(var i = 0, i < cursos.length, i++) {
6   console.log(cursos[i]),
7 }
```

Código Javascript 4.85: array-reverse.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao70.zip

71 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/array-reverse.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/array-reverse.html.

Verifique as mensagens exibidas no console do navegador.

72 Adicione um arquivo chamado **array-shift.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-shift.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.29: array-shift.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao72.zip

73 Adicione um arquivo chamado **array-shift.js** no projeto **javascript**.

```
var cursos = ["K11","K12","K21","K22","K23"],

var curso = cursos.shift(),

console.log("Elemento removido. " + curso),

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Código Javascript 4.86: array-shift.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao73.zip

74 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/array-shift.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/array-shift.html.

Verifique as mensagens exibidas no console do navegador.

75 Adicione um arquivo chamado **array-slice.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-slice.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.30: array-slice.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao75.zip

76 Adicione um arquivo chamado **array-slice.js** no projeto **javascript**.

```
var cursos = ["K11","K12","K21","K22","K23"],

var formacaoJava = cursos.slice(0,2),

for(var i = 0, i < formacaoJava.length, i++) {
  console.log(formacaoJava[i]),
}
```

Código Javascript 4.87: array-slice.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao76.zip

77 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/array-slice.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/array-slice.html.

Verifique as mensagens exibidas no console do navegador.

78 Adicione um arquivo chamado **array-unshift.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-unshift.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.31: array-unshift.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao78.zip

79 Adicione um arquivo chamado **array-unshift.js** no projeto **javascript**.

```
var cursos = ["K12","K21","K22","K23"],

cursos.unshift("K11"),

for(var i = 0, i < cursos.length, i++) {
  console.log(cursos[i]),
}
```

Código Javascript 4.88: array-unshift.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao79.zip

80 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-unshift.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/array-unshift.html.

Verifique as mensagens exibidas no console do navegador.

81 Adicione um arquivo chamado **array-split.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="array-split.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.32: array-split.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao81.zip

82 Adicione um arquivo chamado **array-split.js** no projeto **javascript**.

```
var curso = "K12-Desenvolvimento Web com JSF2 e JPA2",
var aux = curso.split("-"),

console.log(aux[0]),
console.log(aux[1]),
```

Código Javascript 4.89: array-split.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao82.zip

83 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/array-split.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/array-split.html.

Verifique as mensagens exibidas no console do navegador.

DOM - Document Object Model

Os documentos HTML assim como os seus elementos são representados por objetos JavaScript. Podemos acessar o objeto que representa o documento HTML atual através da variável **document**. Os objetos que representam os elementos do documento HTML atual podem ser acessados através dessa variável.

Recuperando elementos

- Podemos buscar elementos HTML por ID através do método **getElementById**. Esse método devolve **null** se não existir elemento HTML com o identificador desejado.

```
var elemento = document.getElementById("conteudo"),
```

- Podemos buscar elementos HTML pelo valor do atributo **name** através do método **getElementsByName**. Esse método devolve um array com os elementos recuperados.

```
var array = document.getElementsByName("categoria"),
```

- Podemos buscar elementos HTML de um determinado tipo através do método **getElementsByTagName**. Esse método devolve um array com os elementos recuperados.

```
var array = document.getElementsByTagName("p"),
```

- Podemos buscar elementos HTML pelo valor do atributo **class** através do método **getElementsByClassName**. Esse método devolve um array com os elementos recuperados.

```
var array = document.getElementsByClassName("confirmado"),
```

- Podemos buscar elementos HTML utilizando seletores CSS através dos métodos **querySelector** e **querySelectorAll**. O primeiro método devolve o primeiro elemento compatível com o seletor CSS passado como parâmetro. O segundo método devolve um array com todos os elementos compatíveis com o seletor CSS passado como parâmetro.

```
var elemento = document.querySelector("div.aprovado"),
var array = document.querySelectorAll("div.aprovado"),
```

- Podemos acessar os atributos de um elemento HTML através do método **getAttribute**. Esse método recebe como parâmetro o nome do atributo desejado e devolve **null** ou **“”**(string vazia) se o atributo não existir.

```
var valor = elemento.getAttribute("type"),
```

- Também podemos acessar os atributos de um elemento HTML através da propriedade **attributes**.

```
for(var i = 0, i < elemento.attributes.length - 1, i++) {
  console.log(elemento.attributes[i].name + " = " + elemento.attributes[i].value),
}
```

- Podemos acessar o conteúdo de um elemento HTML através da propriedade **innerHTML**. Essa propriedade armazena o conteúdo dos elementos HTML em forma de string.

```
var conteudo = elemento.innerHTML,
```

Alterando elementos

- Podemos modificar os atributos de um elemento HTML através do método **setAttribute**.

```
elemento.setAttribute("type", "text"),
```

- Podemos modificar o estilo de um elemento HTML através da propriedade **style**.

```
elemento.style.color = "red",
```

As propriedades com nome composto devem seguir a convenção **Camel Case**.

```
elemento.style.borderColor = "red",
```

- Podemos modificar o conteúdo de um elemento HTML através da propriedade **innerHTML**.

```
elemento.innerHTML = "Acesse <a href='http.//www.k19.com.br'>K19</a>",
```

Removendo elementos

- Podemos remover um elemento HTML através do método **remove**.

```
var elemento = document.getElementById("conteudo"),
elemento.remove(),
```

- Podemos remover um elemento HTML através do seu pai.

```
var elemento = document.getElementById("conteudo"),
elemento.parentNode.removeChild(elemento),
```

Adicionando elementos

- Podemos criar um elemento HTML através do método **createElement** e adicioná-lo em um documento HTML através do método **appendChild**.

```
var titulo = document.createElement("h1"),
titulo.innerHTML = "K19 Treinamentos",
var elemento = document.getElementById("conteudo"),
elemento.appendChild(titulo),
```

- Também podemos adicionar um elemento HTML em um documento HTML através do método **insertBefore**.

```
var titulo = document.createElement("h1"),
titulo.innerHTML = "Cursos K19",

var conteudo = document.getElementById("conteudo"),
var tabela = conteudo.getElementById("tabela-de-cursos"),

conteudo.insertBefore(titulo, tabela),
```

Mais Sobre

Os objetos que representam os documentos e os elementos HTML possuem uma grande quantidade de propriedades e métodos. Para saber mais sobre essas propriedade e esses métodos, recomendamos a consulta aos seguintes endereços:

- https.//developer.mozilla.org/en-US/docs/Web/API/Document
- https.//developer.mozilla.org/en-US/docs/Web/API/element

Exercícios de Fixação

84 Adicione um arquivo chamado **DOM.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
  </head>
  <body>
    <h1 id="titulo" class="centralizado">K19 - JavaScript</h1>

    <form id="formulario" action="#">
      <label for="esporte">Esporte. </label>
      <input id="esporte" type="checkbox" name="categoria">

      <label for="politica">Política. </label>
      <input id="politica" type="checkbox" name="categoria">

      <label for="economia">Economia. </label>
      <input id="economia" type="checkbox" name="categoria">

      <label for="entretenimento">Entretenimento. </label>
      <input id="entretenimento" type="checkbox" name="categoria">
    </form>

    <script type="text/javascript" src="DOM.js"></script>
  </body>
</html>
```

Código HTML 4.33: DOM.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao84.zip

85 Adicione um arquivo chamado **DOM.js** no projeto **javascript**.

```
var titulo = document.getElementById("titulo"),
console.log(titulo.tagName + ". " + titulo.innerHTML),

var array = document.getElementsByName("categoria"),
for(var i = 0, i < array.length, i++) {
  console.log(array[i].tagName + ". " + array[i].id),
}

array = document.getElementsByTagName("label"),
for(var i = 0, i < array.length, i++) {
  console.log(array[i].tagName + ". " + array[i].getAttribute("for")),
}

array = document.getElementsByClassName("centralizado"),
for(var i = 0, i < array.length, i++) {
  console.log(array[i].tagName + ". " + array[i].id),
}

var formulario = document.querySelector("body > form"),
console.log(formulario.tagName + ". " + formulario.getAttribute("action")),

array = document.querySelectorAll("label"),
for(var i = 0, i < array.length, i++) {
  console.log(array[i].tagName + ". " + array[i].getAttribute("for")),
}
```

Código Javascript 4.106: DOM.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao85.zip

86 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/DOM.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/DOM.html.

Verifique as mensagens exibidas no console do navegador.

87 No console do navegador, execute os seguintes comandos um de cada vez e observe o resultado.

```
var titulo = document.getElementById("titulo"),

titulo.style.color = "red",

titulo.innerHTML = "K19 - JavaScript - DOM",

titulo.remove(),

var body = document.querySelector("body"),

var form = document.getElementById("formulario"),

body.insertBefore(titulo, form),
```

Eventos

A interatividade entres as páginas web e os usuários depende fortemente dos eventos que podem ser reconhecidos pelos navegadores. Por exemplo, atualmente, os navegadores são capazes de identificar o movimento do mouse, a digitação do usuário, a finalização do carregamento de uma página web, o redimensionamento da janela, entre outros eventos.

Há uma quantidade muito grande de eventos que podem ser reconhecidos pelos navegadores. Recomendamos que você consulte os seguintes endereços para obter maiores informações:

- http://www.w3.org/TR/DOM-Level-3-Events/
- http://en.wikipedia.org/wiki/DOM_events

Podemos definir o que deve ocorrer quando um determinado evento é detectado. No exemplo abaixo, toda vez que o usuário clicar no parágrafo, a mensagem "K19" será exibida no console do navegador.

```
1 <p onclick="console.log('K19')">
2   Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3 </p>
```

load

O evento **load** é disparado no término do carregamento de um recurso (imagem, script, folhas de estilos, entre outros) contido em um documento HTML ou do próprio documento. Podemos definir o tratamento desse evento utilizando o atributo **onload**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **body**, **img**, **script** e **link**.

```
1 <body onload="console.log('K19')">
2 </body>
```

resize

O evento **resize** é disparado quando ocorre uma alteração no tamanho de determinados elementos. Podemos definir o tratamento desse evento utilizando o atributo **onresize**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **body**, **img**, **form** e **textarea**.

```
1 <body onresize="console.log('K19')">
2 </body>
```

scroll

O evento **scroll** é disparado quando a barra de rolagem de um determinado elemento HTML é movida. Podemos definir o tratamento desse evento utilizando o atributo **onscroll**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **body**, **section**, **div** e **textarea**.

```
1 <body onscroll="console.log('K19')">
2 </body>
```

focus

O evento **focus** é disparado quando um determinado elemento ganha o foco. Podemos definir o tratamento desse evento utilizando o atributo **onfocus**. Esse atributo pode ser utilizado em diversos

elementos HTML. Por exemplo: **input**, **select** e **textarea**.

```
1 <input onfocus="console.log('K19')" ...>
```

change

O evento **change** é disparado quando um determinado elemento HTML perde o foco após ter o seu conteúdo alterado. Podemos definir o tratamento desse evento utilizando o atributo **onchange**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **input**, **select** e **textarea**.

```
1 <input onchange="console.log('K19')" ...>
```

blur

O evento **blur** é disparado quando um determinado elemento perde o foco. Podemos definir o tratamento desse evento utilizando o atributo **onblur**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **input**, **select** e **textarea**.

```
1 <input onblur="console.log('K19')" ...>
```

select

O evento **select** é disparado quando o texto contido em um **input** ou **textarea** é selecionado. Podemos definir o tratamento desse evento utilizando o atributo **onselect**.

```
1 <input onselect="console.log('K19')" ...>
```

submit

O evento **submit** é disparado imediatamente antes do envio de um formulário. Podemos definir o tratamento desse evento utilizando o atributo **onsubmit**.

```
1 <form onsubmit="console.log('K19')" ...>
```

reset

O evento **reset** é disparado quando um formulário é reiniciado. Podemos definir o tratamento desse evento utilizando o atributo **onreset**.

```
1 <form onreset="console.log('K19')" ...>
```

mousedown, mouseup e click

Os eventos **mousedown**, **mouseup** e **click** estão relacionados ao ato de clicar nos elementos HTML com o botão esquerdo ou com o botão do meio do mouse. Podemos definir o tratamento desses eventos utilizando os atributos **onmousedown**, **onmouseup** e **onclick**. Esses atributos podem ser utilizados em diversos elementos HTML. Por exemplo: **p**, **div**, **table** e **img**. Esses eventos ocorrem na seguinte ordem:

1. mousedown
2. mouseup
3. click

```
<p onmousedown="console.log('1')"
  onmouseup="console.log('2')"
  onclick="console.log('3')">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
</p>
```

dblclick

O evento **dblclick** é disparado quando um determinado elemento recebe um duplo clique. Podemos definir o tratamento desse evento utilizando o atributo **ondblclick**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **p**, **div**, **table** e **img**.

```
<p ondblclick="console.log('K19')">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
</p>
```

mousemove

O evento **mousemove** é disparado quando o usuário movimenta o mouse sobre um determinado elemento. Podemos definir o tratamento desse evento utilizando o atributo **onmousemove**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **p**, **div**, **table** e **img**.

```
<p onmousemove="console.log('K19')">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
</p>
```

mouveover

O evento **mouveover** é disparado quando o ponteiro do mouse passa a estar sobre um determinado elemento. Podemos definir o tratamento desse evento utilizando o atributo **onmouveover**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **p**, **div**, **table** e **img**.

```
<p onmouveover="console.log('K19')">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
</p>
```

mouseout

O evento **mouseout** é disparado quando o ponteiro do mouse deixa de estar sobre um determinado elemento. Podemos definir o tratamento desse evento utilizando o atributo **onmouseout**. Esse atributo pode ser utilizado em diversos elementos HTML. Por exemplo: **p**, **div**, **table** e **img**.

```
<p onmouseout="console.log('K19')">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit.
</p>
```

keydown, keypress e keyup

Os eventos **keydown**, **keypress** e **keyup** estão relacionados ao ato de pressionar uma tecla quando o foco está em um determinado elemento HTML. Podemos definir o tratamento desses eventos utilizando os atributos **onkeydown**, **onkeypress** e **onkeyup**. Esses atributos podem ser utilizados em diversos elementos HTML. Por exemplo: **input** e **textarea**. Esses eventos ocorrem na seguinte ordem:

1. keydown
2. keypress
3. keyup

```
<input
  onkeydown="console.log('1')"
  onkeypress="console.log('2')"
  onkeyup="console.log('3')" ...>
```

Mais Sobre
O evento **keypress** não é disparado para teclas que não representam caracteres. Por exemplo, *SHIFT*, *CTRL*, *ALT*, entre outros.

Métodos: addEventListener e removeEventListener

Podemos definir o tratamento dos eventos programaticamente através do método **addEventListener**. O primeiro parâmetro desse método é o nome do evento que desejamos tratar. O segundo parâmetro é a função que tratará o evento.

```
function click() {
  console.log("click"),
}

var elemento = document.getElementById("conteudo"),
elemento.addEventListener("click", click),
```

Nessa abordagem, podemos associar diversas funções para tratar um determinado evento para um determinado elemento HTML.

Também podemos utilizar funções anônimas como mostra o exemplo a seguir.

```
var elemento = document.getElementById("conteudo"),
elemento.addEventListener(
  "click",
  function() {
    console.log("click")
  }
),
```

Podemos remover um **listener** com o método **removeEventListener**.

```
elemento.removeEventListener("click", click),
```

O método **removeEventListener** não pode ser utilizado para funções anônimas.

Propriedades dos objetos do DOM

Outra forma de definir programaticamente o tratamento dos eventos é utilizar as propriedades de evento dos objetos que representam os elementos HTML. Veja o exemplo a seguir.

```
var elemento = document.getElementById("conteudo"),
elemento.onclick = function() {
  console.log("click")
},
```

Nessa abordagem, apenas uma função pode ser associada a um determinado evento para um determinado elemento HTML.

Exercícios de Fixação

88 Adicione um arquivo chamado **eventos.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - JavaScript</title>
    <script type="text/javascript" src="eventos.js"></script>
    <style>
      body {
        width. 2000px,
        height. 2000px,
      }

      div {
        width. 200px,
        height. 200px,
        background-color. yellow,
      }
    </style>
  </head>
  <body onresize="resize()" onscroll="scroll()">
    <form>
      <label for="campo-nome">Nome.</label>
      <input id="campo-nome" type="text">
    </form>

    <div></div>
  </body>
</html>
```

Código HTML 4.50: eventos.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao88.zip

89 Adicione um arquivo chamado **eventos.js** no projeto **javascript**.

```
window.onload = function(event) {
  console.log("documento carregado"),

  var input = document.getElementById("campo-nome"),

  input.onfocus = function() {
    console.log("focus"),
```

```
  },

  input.addEventListener("change", function() {
    console.log("change"),
  }),

  input.onblur = function() {
    console.log("blur"),
  },

  input.onselect = function() {
    console.log("select"),
  },

  input.onkeydown = function() {
    console.log("keydown"),
  },

  input.onkeypress = function() {
    console.log("keypress"),
  },

  input.onkeyup = function() {
    console.log("keyup"),
  },

  var div = document.querySelector("div"),

  div.onmousedown = function() {
    console.log("mousedown"),
  },

  div.onmouseup = function() {
    console.log("mouseup"),
  },

  div.onclick = function() {
    console.log("click"),
  },

  div.ondblclick = function() {
    console.log("dblclick"),
  },

  div.onmousemove = function(event) {
    var x = event.clientX,
    var y = event.clientY,
    console.log("mousemove. (" + x + ", " + y +")"),
  },

  div.onmouseover = function() {
    console.log("mouseover"),
  },

  div.onmouseout = function() {
    console.log("mouseout"),
  },
},

function resize() {
  var w = window.outerWidth,
  var h = window.outerHeight,

  console.log("resize(" + w + ", " + h + ")"),
}

function scroll() {
  var x = window.pageXOffset,
  var y = window.pageYOffset,

```

```
  console.log("scroll(" + x + ", " + y + ")"),
}
```

Código Javascript 4.112: eventos.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao89.zip

90 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/eventos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/eventos.html.

Verifique as mensagens exibidas no console do navegador.

Web Storage

Podemos armazenar dados nos navegadores dos usuários através do **Web Storage**. Há duas formas de armazenamento.

Local storage: Os dados são armazenados nos navegadores indefinidamente. Podemos recuperar esses dados mesmo após o fechamento de uma aba ou da janela do navegador.

Session storage: Os dados armazenados nos navegadores são descartados após o fechamento de uma aba ou do navegador.

Os dados armazenados com a API do Web Storage são separados por domínio e não são compartilhados entre navegadores diferentes. Os navegadores podem determinar a quantidade de espaço de armazenamento disponível. Por padrão, esse espaço deve ser de pelo menos 5MB por domínio.

Armazenando dados

O método **setItem** dos objetos **localStorage** e **sessionStorage** é utilizado para inserir dados no Web Storage. Esse método recebe dois parâmetros, uma chave e o um valor.

```
localStorage.setItem("usuario", "Rafael Cosentino"),
sessionStorage.setItem("usuario", "Rafael Cosentino"),
```

No exemplo abaixo, mostramos outra forma de armazenar dados no Web Storage.

```
localStorage.usuario = "Rafael Cosentino",
sessionStorage.usuario = "Rafael Cosentino",
```

Recuperando dados

O método **getItem** dos objetos **localStorage** e **sessionStorage** é utilizado para recuperar dados do Web Storage. Esse método recebe uma chave como parâmetro.

```
usuario = localStorage.getItem("usuario"),
usuario = sessionStorage.getItem("usuario"),
```

No exemplo abaixo, mostramos outra forma de recuperar dados do Web Storage.

```
usuario = localStorage.usuario,
usuario = sessionStorage.usuario,
```

Podemos percorrer todas as entradas do **localStorage** ou do **sessionStorage** da seguinte forma:

```
for(var i = 0, i < localStorage.length, i++) {
  var key = localStorage.key(i),
  var value = localStorage.getItem(key),
}

for(var i = 0, i < sessionStorage.length, i++) {
  var key = sessionStorage.key(i),
  var value = sessionStorage.getItem(key),
}
```

Removendo dados

O método **removeItem** dos objetos **localStorage** e **sessionStorage** é utilizado para remover dados do Web Storage. Esse método recebe uma chave como parâmetro.

```
localStorage.removeItem("usuario"),
sessionStorage.removeItem("usuario"),
```

Para remover todas as entradas do **localStorage** ou do **sessionStorage**, podemos utilizar o método **clear**.

```
localStorage.clear(),
sessionStorage.clear(),
```

Exercícios de Fixação

91 Adicione um arquivo chamado **web-storage.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Web Storage</title>
    <script src="web-storage.js"></script>
  </head>
  <body>
    <h1>K19 - Web Storage</h1>
    <p id="saudacao"></p>
    <label for="campoNome">Nome.</label>
    <input id="campoNome" type="text">
    <button id="botaoEnviar">Enviar</button>
    <button id="botaoLimpar">Limpar</button>
  </body>
</html>
```

Código HTML 4.51: web-storage.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao91.zip

92 Adicione um arquivo chamado **web-storage.js** no projeto **javascript**.

```
function atualizaSaudacao() {
  var saudacao = document.getElementById("saudacao"),
  if(localStorage.nome) {
    saudacao.innerHTML = "Olá " + localStorage.nome,
  } else {
    saudacao.innerHTML = null,
  }
},

window.onload = function() {
  var botaoEnviar = document.getElementById("botaoEnviar"),
  botaoEnviar.onclick = function() {
    var campoNome = document.getElementById("campoNome"),
    localStorage.nome = campoNome.value,
    atualizaSaudacao(),
  },

  var botaoLimpar = document.getElementById("botaoLimpar"),
  botaoLimpar.onclick = function() {
    localStorage.removeItem("nome"),
    atualizaSaudacao(),
  },

  atualizaSaudacao(),
},
```

Código Javascript 4.120: web-storage.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao92.zip

93 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/web-storage.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/web-storage.html.

Preencha o formulário. Abra e feche a janela do navegador.

History

Podemos avançar ou retroceder no histórico dos navegadores através do objeto **history**. Esse objeto pode ser acessado através do objeto **window**.

Avançando ou retrocedendo

Podemos avançar ou retroceder uma página no histórico através dos métodos **forward** e **back** respectivamente.

```
window.history.forward(),
```

```
window.history.back(),
```

Também podemos utilizar o método **go**. Esse método recebe um número inteiro como parâmetro.

```
// avança uma página
window.history.go(1),

// retrocede uma página
window.history.go(-1),

// avança três páginas
window.history.go(3),

// retrocede três páginas
window.history.go(-3),
```

A quantidade de páginas no histórico pode ser obtida através da propriedade **length**.

```
var numeroDePaginasDoHistorico = window.history.length,
```

Adicionando ou modificando entradas do histórico

Através do método **pushState**, podemos adicionar uma entrada na posição atual do histórico. Com o método **replaceState**, podemos modificar a entrada atual do histórico. Esses dois métodos recebem os mesmos parâmetros.

```
pushState(stateObject, title, url),
```

```
replaceState(stateObject, title, url),
```

stateObject: Um objeto que pode ser utilizado para armazenar informações sobre a nova entrada.

title: O título da nova entrada.

url: A URL da nova entrada.

Como exemplo, considere que a página correspondente à URL `www.k19.com.br` esteja sendo exibida no navegador.

```
var state = { info. "info" },
window.history.pushState(state, "K19 - Cursos", "cursos"),
```

O código acima adicionaria uma nova entrada com título "K19 - Cursos" e URL `www.k19.com.br/cursos`. Já o código abaixo, substituiria a entrada atual do histórico por uma nova com título "K19 - Apostilas" e URL **www.k19.com.br/apostilas**.

```
var state = { info. "info" },
window.history.replaceState(state, "K19 - Apostilas", "apostilas"),
```

Exercícios de Fixação

94 Adicione um arquivo chamado **historico-pagina1.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Histórico Página 1</title>
  </head>
  <body>
    <h1>K19 - Histórico Página 1</h1>
    <button onclick="window.history.forward(),">Avançar 1 página</button>
    <button onclick="window.history.go(2),">Avançar 2 página</button>
  </body>
</html>
```

Código HTML 4.52: historico-pagina1.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao94.zip

95 Adicione um arquivo chamado **historico-pagina2.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Histórico Página 2</title>
  </head>
  <body>
    <h1>K19 - Histórico Página 2</h1>
    <button onclick="window.history.back(),">Voltar 1 página</button>
    <button onclick="window.history.forward(),">Avançar 1 página</button>
  </body>
</html>
```

Código HTML 4.53: historico-pagina2.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao95.zip

96 Adicione um arquivo chamado **historico-pagina3.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Histórico Página 3</title>
  </head>
  <body>
    <h1>K19 - Histórico Página 3</h1>
    <button onclick="window.history.go(-2),">Voltar 2 página</button>
    <button onclick="window.history.back(),">Voltar 1 página</button>
  </body>
</html>
```

Código HTML 4.54: historico-pagina3.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao96.zip

97 No **Windows**, utilize o **Chrome** para acessar na sequência os endereços:

http.//localhost/javascript/public_html/historico-pagina1.html.
http.//localhost/javascript/public_html/historico-pagina2.html.
http.//localhost/javascript/public_html/historico-pagina3.html.

No **Ubuntu**, utilize o **Chrome** para acessar na sequência os endereços:

http.//localhost/~<USUARIO>/javascript/public_html/historico-pagina1.html.
http.//localhost/~<USUARIO>/javascript/public_html/historico-pagina2.html.
http.//localhost/~<USUARIO>/javascript/public_html/historico-pagina3.html.

Navegue através dos botões.

Geolocalização

Se o usuário permitir, podemos obter a localização dele através da API de Geolocalização do HTML5. Essa localização pode ser utilizada para diversos propósitos. Por exemplo, podemos exibir publicidade relacionada à região onde o usuário está.

O primeiro passo para utilizar essa API, é definir uma função JavaScript para receber a localização quando ela for obtida.

```
function positionCallback(position) {
  console.log("Latitude. " + position.coords.latitude),
  console.log("Longitude. " + position.coords.longitude),
}
```

Depois, devemos utilizar a função **getCurrentPosition** para solicitar a localização do usuário.

```
navigator.geolocation.getCurrentPosition(positionCallback),
```

Exercícios de Fixação

98 Adicione um arquivo chamado **geolocalizacao.html** no projeto **javascript**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - Geolocalizacao</title>
    <style type="text/css">
      #mapa {
        width. 800px,
        height. 600px,
      }
    </style>

    <!-- google maps api -->
```

```
    <script src="https.//maps.googleapis.com/maps/api/js?v=3.exp&sensor=false">
    </script>

    <script src="geolocalizacao.js"></script>
  </head>
  <body>
    <h1>K19 - Geolocalizacao</h1>
    <div id="mapa"></div>
  </body>
</html>
```

Código HTML 4.55: geolocalizacao.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao98.zip

99 Adicione um arquivo chamado **geolocalizacao.js** no projeto **javascript**.

```
function positionCallback(position) {
  var latitude = position.coords.latitude,
  var longitude = position.coords.longitude,
  var geolocation = new google.maps.LatLng(latitude, longitude),

  var mapOptions = {
    zoom. 18,
    center. geolocation,
    mapTypeId. google.maps.MapTypeId.ROADMAP
  },

  var div = document.getElementById("mapa"),
  var map = new google.maps.Map(div, mapOptions),

  var marker = new google.maps.Marker({
    position. geolocation,
    map. map,
    title. "Você está aqui!"
  }),
}

window.onload = function() {
  navigator.geolocation.getCurrentPosition(positionCallback),
},
```

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-fixacao99.zip

100 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/geolocalizacao.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/geolocalizacao.html.

Alarmes (Conteúdo Extra)

Podemos agendar tarefas para serem executadas no futuro. Para isso, podemos utilizar os métodos **setTimeout** e **setInterval**. A primeira permite definir uma função que deve ser executada apenas uma vez depois de um determinado tempo. A segunda permite definir uma função que deve ser executada de forma periódica.

No exemplo a seguir, definimos uma função que será executada uma vez depois de 1600 milisegundos.

```
window.setTimeou(function(){
  console.log("timeout"),
}, 1600),
```

Por outro lado, nesse outro exemplo, definimos uma função que será executada a cada 2500 milisegundos.

```
window.setInterval(function(){
  console.log("interval"),
}, 2500),
```

Podemos cancelar os alarmes com os métodos **clearTimeout** e **clearInterval**. Esses métodos necessitam da referencia da função associada ao alarme.

```
function alarmeTimeout() {
  console.log("timeout"),
}

window.setTimeout(alarmeTimeout, 1600),

window.clearTimeout(alarmeTimeout),
```

```
function alarmeInterval() {
  console.log("interval"),
}

window.setInterval(alarmeInterval, 2500),

window.clearInterval(alarmeInterval),
```

Exercícios Complementares

1. Crie um documento HTML vinculado a um documento JavaScript que exiba no console do navegador os números de 1 até 50 duas vezes.

2. Crie um documento HTML vinculado a um documento JavaScript que exiba no console do navegador o seu nome cinco vezes. A cada exibição do seu nome, exiba três vezes a palavra “K19”.

3. Crie um documento HTML vinculado a um documento JavaScript que percorra todos os números de 1 até 60. Para os números múltiplos de 3, exiba “***”. Para os números que não são múltiplos de 3, exiba “*”.

4. Crie um documento HTML vinculado a um documento JavaScript que exiba no console do navegador todos os números de 1 até 80 exceto os múltiplos de 4 e 7. Para esses exiba “*”.

5 Crie um documento HTML vinculado a um documento JavaScript que exiba no console do navegador um triângulo de "*". Veja o exemplo abaixo:

```
*
**
***
****
*****
```

6 Crie um documento HTML vinculado a um documento JavaScript que exiba no console do navegador vários triângulos de "*". Observe o padrão abaixo.

```
*
**
***
****
*
**
***
****
```

7 Os números de Fibonacci são uma sequência de números definida recursivamente. O primeiro elemento da sequência é 0 e o segundo é 1. Os outros elementos são calculados somando os dois antecessores.

```
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233...
```

Crie um documento HTML vinculado a um documento JavaScript para exibir os 30 primeiros números da sequência de Fibonacci.

8 Crie um documento HTML vinculado a um documento JavaScript que armazene 10 números em um array. Preencha todas as posições do array com valores sequenciais e em seguida exiba-os na tela. Depois, escolha duas posições aleatoriamente e troque os valores contidos nelas. Repita essa operação 10 vezes. Ao final, exiba o array novamente.

Dica: O método **Math.random()** gera números aleatórios maiores ou iguais a 0 e menores do que 1. O método **Math.floor()** recebe um número real como parâmetro e devolve o maior inteiro menor ou igual a esse número real.

9 Crie um documento HTML vinculado a um documento JavaScript que armazene 10 números em um array. Preencha todas as posições do array com valores aleatórios e em seguida exiba-os na tela. Após exibir o array, ordene o array do menor valor para o maior. Ao final, exiba o array ordenado.

Dica: Os arrays possuem um método chamada **sort**. Esse método ordena os elementos dos arrays.

Resumo do Capítulo

1 A linguagem de programação JavaScript é utilizada principalmente para aumentar a interatividade entre as páginas web e os usuários.

2 Código JavaScript pode ser aplicado aos documentos HTML de duas formas: **JavaScript interno** e **JavaScript externo**.

3 O carregamento do código JavaScript depende do posicionamento do elemento **script**.

4 Comentários JavaScript de bloco podem ser definidos com os marcadores **/*** e ***/**. **script**.

5 Comentários JavaScript de linha podem ser definidos com o marcador **//**. **script**.

6 Em JavaScript, as variáveis são criadas através da palavra especial **var**.

7 Variáveis não inicializadas possuem o valor especial **undefined**.

8 Os operadores do JavaScript são:

- Aritmético: **+ - * / %**
- Atribuição: **= += -= *= /= %= ++ --**
- Relacional: **== != < <= > >=**
- Lógico: **&& ||**

9 O operador + é utilizado para concatenar strings.

10 A linguagem JavaScript permite a definição de objetos com a sintaxe do **JSON**.

11 Funções JavaScript são criadas com a palavra **function**.

12 Os métodos **getElementById**(), **getElementsByName**(), **getElementsByTagName**(), **getElementsByClassName**(), **querySelector**() e **querySelectorAll**() são utilizados para recuperar os ele-

mentos de um documento HTML.

13 Os métodos **getAttribute()** e **setAttribute()** são utilizados, respectivamente, para recuperar e alterar os valores dos atributos dos elementos HTML.

14 O conteúdo de um elemento HTML pode ser acessado através da propriedade **innerHTML**.

15 A propriedade **style** pode ser utilizada para alterar as propriedades CSS dos elementos HTML.

16 Os métodos **remove()** e **removeChild()** são utilizados para remover elementos de um documento HTML.

17 Os métodos **appendChild()** e **insertBefore()** são utilizados para adicionar elementos em um documento HTML.

18 A interatividade entre as páginas web e os usuários depende dos eventos JavaScript.

19 Podemos armazenar dados na máquina dos usuários com a API do **Web Storage**.

20 Podemos controlar o histórico do navegador com a API **History**.

21 Podemos obter a localização dos usuários através da API de geolocalização.

Prova

1 Qual alternativa está correta?

a) Podemos associar código JavaScript aos documentos HTML através dos elementos **script** e **js**.

b) O código JavaScript sempre é carregado antes de todos os elementos HTML.

c) O código JavaScript sempre é colocado dentro do documento HTML.

d) O elemento **script** só pode ser colocada no corpo do elemento **head**.

e) Podemos associar código JavaScript aos documentos HTML através do elemento **script**.

2 Qual alternativa está correta?

a) A palavra especial **var** é utilizada para criar variáveis.

b) As variáveis JavaScript armazenam apenas valores do tipo string.

c) As variáveis JavaScript não inicializadas possuem o valor **empty**.

d) Na linguagem JavaScript, duas variáveis não podem armazenar o mesmo valor.

e) Os nomes das variáveis JavaScript devem iniciar obrigatoriamente com a letra **v**.

3 Considere o seguinte trecho de código JavaScript.

```
var i = 10,

var j = ++i + i--,
```

Qual é o valor da variável **j**?

a) 18.

b) 19.

c) 20.

d) 21.

e) 22.

4 Considere o seguinte trecho de código JavaScript.

```
var i = 10,
var j = 15,

console.log(!(i < j) ? (j % 2 == i % 3 ? "K01" . "K02") . (j % 4 > i % 4 ? "K03" . "↩
    K11")),
```

O que será exibido no console do navegador?

a) K01.

b) K02.

c) K03.

d) K11.

e) Nenhuma das anteriores.

5 Considere o seguinte trecho de código JavaScript.

```
var i = 10,

while(i++ < 15) {
  console.log(++i),
}
```

Quais números serão exibidos no console do navegador?

a) 12, 14 e 16.

b) 12 e 14.

c) 11, 13 e 15.

d) 11 e 13.

e) 11, 12, 13 e 14.

6 Considere o seguinte trecho de código JavaScript.

```
var objeto = {a. "b", b. "a"},
console.log(objeto["a"]),
console.log(objeto["b"]),
delete objeto.a,
console.log(objeto["a"]),
console.log(objeto["b"]),
```

O que será exibido no console do navegador?

a) b, a, b, undefined.

b) a, b, undefined, a.

c) b, a, undefined, a.

d) b, a, undefined, b.

e) a, b, a, undefined.

7 Qual alternativa está correta?

a) As funções JavaScript são criadas com as palavras **func** e **function**.

b) As funções JavaScript são criadas com a palavra **function**.

c) As funções JavaScript são criadas com a palavra **func**.

d) Toda função JavaScript deve receber parâmetros.

e) A linguagem JavaScript não suporta funções.

8 Qual alternativa está correta?

a) O método **getElementsById()** recupera elementos HTML por ID.

b) O método **getElementByTagName()** recupera elementos HTML de um determinado tipo.

c) A propriedade **css** dos elementos HTML permite que as propriedades CSS sejam acessadas e/ou modificadas.

d) O método **remove()** remove elementos de um documento HTML.

e) Podemos acessar o conteúdo de um elemento HTML através da propriedade **html**.

9 Qual alternativa está correta?

a) O evento **blur** é disparado quando um determinado elemento ganha o foco.

b) O evento **keyup** é sempre disparado antes do evento **keypress**.

c) A linguagem JavaScript não suporta eventos.

d) A única forma de associar os eventos aos elementos HTML é o método **addEventListener**().

e) Todas as alternativas anteriores estão incorretas.

10 Qual alternativa está correta?

a) A API Web Storage permite que dados sejam armazenados nos Web Servers.

b) Os dados armazenados em Session Storage são descartados após o fechamento da aba ou do navegador correspondente.

c) Para armazenadas dados com a API Web Storage, devemos utilizar o método **put**().

d) Para armazenadas dados com a API Web Storage, devemos utilizar o método **add**().

e) Podemos apagar todos os dados armazenados com a API Web através do método **reset**().

11 Qual alternativa está correta?

a) A API History permite descobrir as senhas dos usuários.

b) Através da API de geolocalização podemos descobrir a localização dos Web Servers.

c) A localização dos usuários pode ser obtida através da API de geolocalização somente com a autorização dos usuários.

d) A localização dos usuários pode ser obtida através da API de geolocalização mesmo sem a autorização dos usuários.

e) Todas as alternativas anteriores estão incorretas.

Minha Pontuação ☐ Pontuação Mínima: 8 Pontuação Máxima: 11

jQuery

CAPÍTULO 5

Introdução

Basicamente, **jQuery** é uma biblioteca JavaScript. Ela foi desenvolvida para simplificar e diminuir a quantidade de código JavaScript. Os principais recursos oferecidos por essa essa biblioteca são:

- Seletores
- Manipulação de elementos HTML
- Manipulação de propriedades CSS
- Eventos
- Efeitos e Animações
- AJAX

Para utilizar o **jQuery**, basta adicionar o arquivo JavaScript que contém o código dessa biblioteca em um documento HTML. Esse arquivo JavaScript pode ser obtido no endereço http.//docs.jquery.com/Downloading_jQuery. Ele está disponível em duas versões: **Compressed** e **Uncompressed**. Em produção, a primeira versão deve ser utilizada para não sobrecarregar a transferência de dados entre o Web Server e os navegadores. Em desenvolvimento, podemos utilizar a segunda versão pois ela possui um código bem mais legível o que facilita a depuração.

```
1 <head>
2   <script type="text/javascript" src="jquery.js"></script>
3 </head>
```

Nessa abordagem, a quantidade de dados transferidos entre o Web Server e os navegadores é muito alta. Além disso, dependendo da localização dos usuários, a latência para o download do arquivo JavaScript do **jQuery** pode ser alta também. Para diminuir essa quantidade de dados e essa latência, podemos utilizar um **CDN (Content Delivery Network)**. Basicamente, CDN é uma rede de computadores conectados à Internet focada na distribuição de conteúdo. Eis algumas opções de CDN:

- CDN do jQuery (MaxCDN)

```
1 <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
```

- Google CDN

```
<script
  src="http.//ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>
```

- Microsoft CDN

```
<script
  src="http.//ajax.aspnetcdn.com/ajax/jQuery/jquery-1.10.2.min.js"></script>
```

- CDNJS

```
<script
  src="http.//cdnjs.cloudflare.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>
```

A documentação completa do **jQuery** está no endereço http.//docs.jquery.com/.

Exercícios de Fixação

1. Abra o **Netbeans** e crie um projeto chamado **jQuery**.

Importante
No **Windows**, utilizando o IIS (Internet Information Services) como Web Server, você deve salvar o projeto **jQuery** em **C:\inetpub\wwwroot**. Lembre-se que é necessário instalar o IIS conforme vimos anteriormente.

Importante
No **Ubuntu**, utilizando o Apache HTTP Server como Web Server, você deve salvar o projeto **jQuery** em **/home/<USUARIO>/public_html**. Lembre-se que é necessário instalar e configurar o Apache HTTP Server como vimos anteriormente.

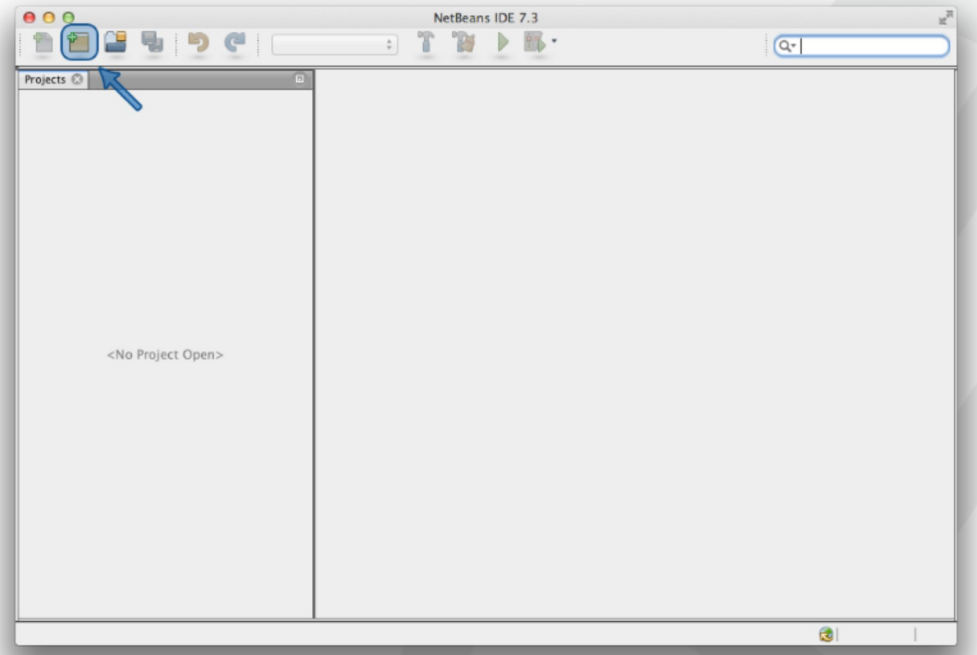


*Figura 5.1: Projeto **jQuery***

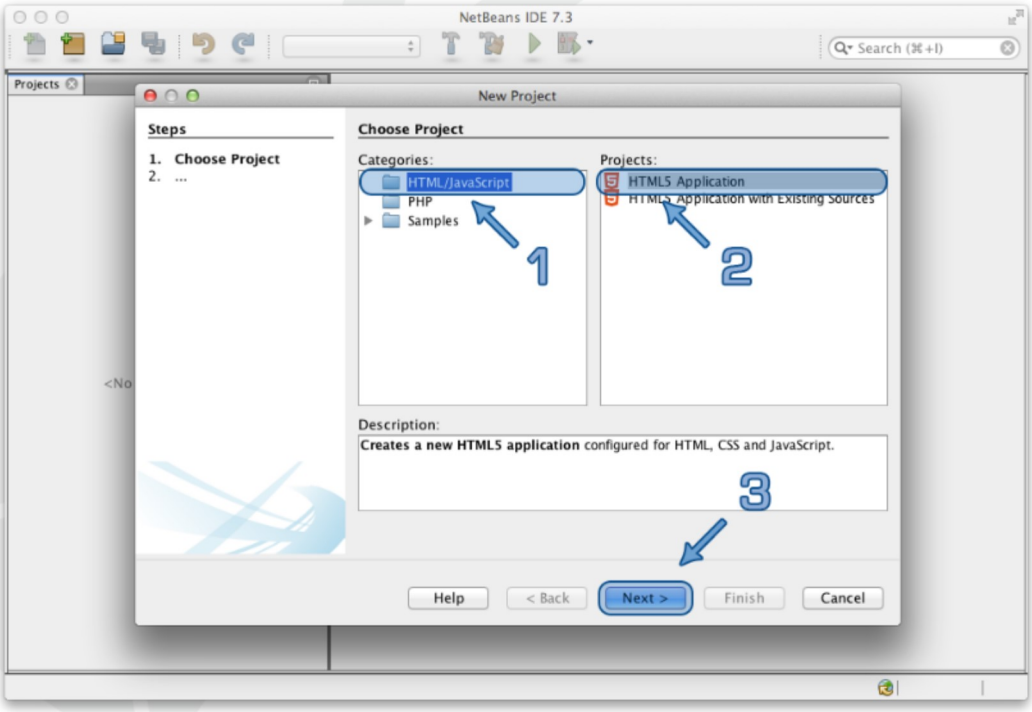


*Figura 5.2: Projeto **jQuery***

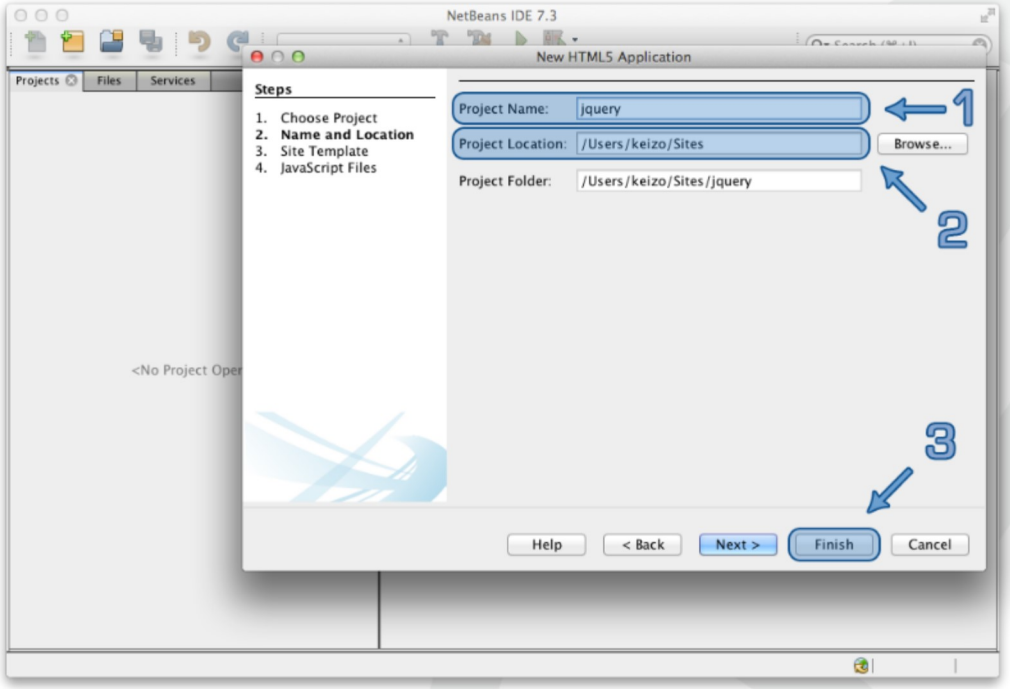


*Figura 5.3: Projeto **jQuery***

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao1.zip

2 No projeto **jQuery**, crie um arquivo chamado **jQuery.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - jQuery</title>
    <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
    <style type="text/css">
      div {
        height. 200px,
        width. 400px,
        background-color. yellow,
      }
    </style>
  </head>
  <body>
    <button id="esconder">Esconder</button>
    <button id="mostrar">Mostrar</button>
    <div></div>

    <script src="jQuery.js"></script>
  </body>
</html>
```

Código HTML 5.6: jQuery.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao2.zip

3 No projeto **jQuery**, crie um arquivo chamado **jQuery.js**.

```
$("#esconder").click(function() {
  $("div").slideUp(),
}),

$("#mostrar").click(function() {
  $("div").slideDown(),
}),
```

Código Javascript 5.1: jQuery.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao3.zip

4 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/jQuery/public_html/jQuery.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/jQuery/public_html/jQuery.html.

Eventos

Evento	Descrição
ready	Esse evento é disparado após o carregamento árvore de elementos do documento HTML. http://api.jquery.com/ready
resize	Esse evento é disparado quando a janela do navegador é redimensionada. http://api.jquery.com/resize
scroll	Esse evento é disparado quando a barra de rolagem de um elemento é movimentada. http://api.jquery.com/scroll
focus	Esse evento é disparado quando um elemento ganha o foco. http://api.jquery.com/focus
focusin	Esse evento é disparado quando um elemento ou um dos seus descendentes ganha o foco. http://api.jquery.com/focusin
blur	Esse evento é disparado quando um elemento perde o foco. http://api.jquery.com/blur
focusout	Esse evento é disparado quando um elemento ou um dos seus descendentes perde o foco. http://api.jquery.com/focusout
select	Esse evento é disparado quando o valor de um elemento é selecionado. http://api.jquery.com/select
change	Em determinados casos, esse evento é disparado depois de um elemento ter o seu valor modificado e em seguida perder o foco. Em outros casos, esse evento é disparado depois de um elemento ter o seu valor modificado e selecionado nessa ordem. http://api.jquery.com/change
keydown	Esse evento é disparado quando uma tecla é pressionada. http://api.jquery.com/keydown

Evento	Descrição
keypress	Esse evento é semelhante ao evento **keydown**. Contudo, ele não é disparado para teclas que não representam caracteres (exemplo: *SHIFT*, *CTRL*, *ALT*, entre outros). O **keypress** é disparado depois do **keydown**. http://api.jquery.com/keypress
keyup	Esse evento é disparado quando uma tecla é solta. http://api.jquery.com/keyup
click	Esse evento é disparado quando o usuário clica sobre um elemento com o botão esquerdo ou com o botão do meio do mouse. http://api.jquery.com/click
dblclick	Esse evento é disparado quando o usuário realiza um duplo clique sobre um elemento com o botão esquerdo ou com o botão do meio do mouse. http://api.jquery.com/dblclick
mousedown	Esse evento é disparado quando o usuário pressiona um botão do mouse sobre um determinado elemento. http://api.jquery.com/mousedown
mouseup	Esse evento é disparado quando o usuário solta um botão do mouse sobre um determinado elemento. http://api.jquery.com/mouseup
mouseenter	Esse evento é disparado no momento em que o ponteiro do mouse passa a estar sobre um determinado elemento. http://api.jquery.com/mouseenter
mouseleave	Esse evento é disparado no momento em que o ponteiro do mouse passa a não estar sobre um determinado elemento. http://api.jquery.com/mouseleave
hover	Esse evento é disparado no **mouseenter** e **mouseleave**. http://api.jquery.com/hover
mouveover	Esse evento é disparado no momento em que o ponteiro do mouse passa a estar sobre um determinado elemento ou sobre os seus descendentes. http://api.jquery.com/mouveover
mouseout	Esse evento é disparado no momento em que o ponteiro do mouse passa a não estar sobre um determinado elemento ou sobre os seus descendentes. http://api.jquery.com/mouseout
mousemove	Esse evento é disparado quando o ponteiro do mouse se move sobre um determinado elemento ou sobre os seus descendentes. http://api.jquery.com/mousemove
submit	Esse evento é disparado quando um formulário é enviado. http://api.jquery.com/submit

on

Podemos definir o tratamento dos eventos com o método **on**. No exemplo abaixo, o primeiro parâmetro é o nome do evento que será tratado e o segundo parâmetro a função que tratará o evento.

```
$("body").on("click", function(){
  console.log("click"),
}),
```

O tratamento de diversos eventos pode ser definido com uma única chamada do método **on**.

```
$("body").on("click mouseenter mouseleave", function(){
```

```
  console.log("click mouseenter mouseleave"),
}),
```

off

Podemos eliminar o tratamento de um evento com o método **off**. A seguir, mostramos algumas formas de utilização desse método.

```
/* Removendo todos os tratamentos associados aos eventos dos parágrafos */
$("p").off(),

/* Removendo todos os tratamentos associados ao evento de clique dos parágrafos */
$("p").off("click"),

/* Removendo um tratamento específico associado ao evento de clique dos parágrafos */
$("p").off("click", tratamento),
```

Atalhos

O tratamento de um evento pode ser definido através de métodos que funcionam como atalhos para o método **on**. No exemplo abaixo, utilizamos o método **click** que é um atalho para **on("click", função)**.

```
$("body").click(function(){
  console.log("click"),
}),
```

Analogamente, para cada evento, há um método de atalho com o mesmo nome.

Propriedades

Os eventos são associados à informações especificas. Por exemplo, quando o evento **click** é disparado, ele é associado a uma coordenada horizontal e uma vertical. Essas coordenadas definem a posição do ponteiro do mouse quando o clique ocorreu. Para recuperar essas informações, basta adicionar um parâmetro nas funções de tratamento de eventos.

No exemplo abaixo, utilizamos as propriedades **pageX** e **pageY** para recuperar as coordenadas correspondentes à posição do ponteiro do mouse quando um clique ocorrer no **body**.

```
$("body").click(function(event){
  console.log("click. (" + event.pageX + ", " + event.pageY + ")"),
}),
```

Propriedade	Descrição
pageX	Coordenada horizontal da posição do ponteiro do mouse. http://api.jquery.com/event.pageX
pageY	Coordenada vertical da posição do ponteiro do mouse. http://api.jquery.com/event.pageY
event.timeStam	A diferença em milissegundos entre 01/01/1970 e o momento no qual o evento foi criado. http://api.jquery.com/event.timeStamp
event.type	O tipo de evento. http://api.jquery.com/event.type

Propriedade	Descrição
event.which	A tecla ou o botão do mouse que foi pressionado. http.//api.jquery.com/event.which

Mais Sobre

Podemos utilizar a propriedade **event.which** para recuperar o **keyCode** das teclas pressionadas nos eventos **keydown**, **keypress** e **keyup**. Contudo, é importante saber que o mapeamento de teclas para os eventos **keydown** e **keyup** é diferente do mapeamento de teclas para o evento **keypress**.

Exercícios de Fixação

5 No projeto **jQuery**, crie um arquivo chamado **eventos.html**.

```
<!DOCTYPE html>
<html lang="pt-br" id="html">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - jQuery</title>
    <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
    <script src="eventos.js"></script>
    <style type="text/css">
      body {
        width. 1000px,
        height. 800px,
      }
      #div1 {
        width. 400px,
        height. 300px,
        background-color. yellow,
      }
      #div2 {
        width. 400px,
        height. 300px,
        background-color. blue,
      }
      #div-interno {
        width. 400px,
        height. 300px,
        margin. auto,
        background-color. green,
      }
    </style>
  </head>
  <body id="body">
    <form id="form1">
      <input type="text" id="input1">
    </form>
    <form id="form2">
      <input type="text" id="input2">
    </form>
    <div id="div1">
      <div id="div-interno"></div>
    </div>
    <div id="div2"></div>
  </body>
</html>
```

Código HTML 5.7: eventos.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao5.zip

6 No projeto **jQuery**, crie um arquivo chamado **eventos.js**.

```
/* tratando o evento ready*/
$(document).ready(function(){
  console.log("ready"),

  /* tratando o evento resize */
  $(window).resize(function() {
    var largura = $(window).width(),
    var altura = $(window).height(),
    console.log("resize. (" + largura + ", " + altura + ")"),
  }),

  /* tratando o evento scroll */
  $(window).scroll(function() {
    var x = $(window).scrollLeft(),
    var y = $(window).scrollTop(),
    console.log("scroll. (" + x + ", " + y + ")"),
  }),

  /* tratando o evento focus */
  $("*").focus(function() {
    var tagName = this.tagName,
    var id = this.id,
    console.log("focus. (" + tagName + ", #" + id + ")"),
  }),

  /* tratando o evento focusin */
  $("*").focusin(function() {
    var tagName = this.tagName,
    var id = this.id,
    console.log("focusin. (" + tagName + ", #" + id + ")"),
  }),

  /* tratando o evento blur */
  $("*").blur(function() {
    var tagName = this.tagName,
    var id = this.id,
    console.log("blur. (" + tagName + ", #" + id + ")"),
  }),

  /* tratando o evento focusout */
  $("*").focusout(function() {
    var tagName = this.tagName,
    var id = this.id,
    console.log("focusout. (" + tagName + ", #" + id + ")"),
  }),

  /* tratando o evento select */
  $("input").select(function() {
    console.log("select. (" + window.getSelection() + ")"),
  }),

  /* tratando o evento change */
  $("input").change(function() {
    console.log("change. (" + $(this).val() + ")"),
  }),

  /* tratando o evento keydown */
  $("input").keydown(function(event) {
    console.log("keydown. (" + event.which + ")"),
```

```
60    }),
61
62    /* tratando o evento keypress */
63    $("input").keypress(function(event) {
64      console.log("keypress. (" + event.which + ")"),
65    }),
66
67    /* tratando o evento keyup */
68    $("input").keyup(function(event) {
69      console.log("keyup. (" + event.which + ")"),
70    }),
71
72    /* tratando o evento click */
73    $("#div1").click(function(event) {
74      console.log("click. (" + event.pageX + ", " + event.pageY + ")"),
75    }),
76
77    /* tratando o evento dblclick */
78    $("#div1").dblclick(function(event) {
79      console.log("dblclick. (" + event.pageX + ", " + event.pageY + ")"),
80    }),
81
82    /* tratando o evento mousedown */
83    $("#div1").mousedown(function(event) {
84      console.log("mousedown. (" + event.pageX + ", " + event.pageY + ")"),
85    }),
86
87    /* tratando o evento mouseup */
88    $("#div1").mouseup(function(event) {
89      console.log("mouseup. (" + event.pageX + ", " + event.pageY + ")"),
90    }),
91
92    /* tratando o evento mouseenter */
93    $("#div1").mouseenter(function(event) {
94      console.log("mouseenter. (" + event.pageX + ", " + event.pageY + ")"),
95    }),
96
97    /* tratando o evento mouseleave */
98    $("#div1").mouseleave(function(event) {
99      console.log("mouseleave. (" + event.pageX + ", " + event.pageY + ")"),
100   }),
101
102   /* tratando o evento hover */
103   $("#div1").hover(function(event) {
104     console.log("hover. (" + event.pageX + ", " + event.pageY + ")"),
105   }),
106
107   /* tratando o evento mouseover */
108   $("#div1").mouseover(function(event) {
109     console.log("mouseover. (" + event.pageX + ", " + event.pageY + ")"),
110   }),
111
112   /* tratando o evento mouseout */
113   $("#div1").mouseout(function(event) {
114     console.log("mouseout. (" + event.pageX + ", " + event.pageY + ")"),
115   }),
116
117   /* tratando o evento mousemove */
118   $("#div2").mousemove(function(event) {
119     console.log("mousemove. (" + event.pageX + ", " + event.pageY + ")"),
120   }),
121 }),
```

Código Javascript 5.7: eventos.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao6.zip

7 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/jQuery/public_html/eventos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/jQuery/public_html/eventos.html.

Utilize o console do navegador para observar as mensagens exibidas.

Seletores

Seletor	Descrição
"*"	Seleciona todos os elementos. Exemplo: selecionando todos os elementos. `$("*"),` http://api.jquery.com/all-selector
[atributo]	Seleciona os elementos que possuem o atributo especificado. Exemplo: selecionando os divs que possuem id. `$("div[id]"),` http://api.jquery.com/has-attribute-selector
[atributo="valor"]	Seleciona os elementos tal que o valor de um determinado atributo é igual ao valor especificado. Exemplo: selecionando os imgs com title igual a logo. `$("img[title='logo']"),` http://api.jquery.com/attribute-equals-selector
[atributo!="valor"]	Seleciona os elementos tal que o valor de um determinado atributo não é igual ao valor especificado. Exemplo: selecionando os imgs com title diferente de logo. `$("img[title!='logo']"),` http://api.jquery.com/attribute-not-equal-selector
[atributo^="valor"]	Seleciona os elementos tal que o valor de um determinado atributo começa com o valor especificado. Exemplo: selecionando os imgs com title iniciando com a string logo. `$("img[title^='logo']"),` http://api.jquery.com/attribute-starts-with-selector
[atributo$="valor"]	Seleciona os elementos tal que o valor de um determinado atributo termina com o valor especificado. Exemplo: selecionando os imgs com title terminando com a string logo. `$("img[title$='logo']")` http://api.jquery.com/attribute-ends-with-selector

Seletor	Descrição
[atributo*="valor"]	Seleciona os elementos tal que o valor de um determinado atributo contém o valor especificado como substring. Exemplo: selecionando os imgs com title contendo a string logo. `$("img[title*='logo']")` http://api.jquery.com/attribute-contains-selector
[atributo~="valor"]	Seleciona os elementos tal que o valor de um determinado atributo contém o valor especificado como palavra. Exemplo: selecionando os imgs com title contendo a palavra logo. `$("img[title~='logo']")` http://api.jquery.com/attribute-contains-word-selector
[atributo|="valor"]	Seleciona os elementos tal que o valor de um determinado atributo é igual ao valor especificado ou inicia com esse valor seguido do caractere "**-**". Exemplo: selecionando os imgs com title iniciando com o prefixo logo. `$("img[title|='logo']")` http://api.jquery.com/attribute-contains-prefix-selector
[f1][f2]...[fn]	Seleciona os elementos de acordo com os filtros de atributo especificados. Exemplo: selecionando os inputs com id e type igual a text. `$("input[id][type='text']")` http://api.jquery.com/multiple-attribute-selector
"#id"	Seleciona o elemento com o identificador especificado. Exemplo: selecionando o elemento com id igual a conteudo . `$("#conteudo")` http://api.jquery.com/id-selector
"tipo"	Seleciona todos os elementos do tipo especificado. Exemplo: selecionando os divs. `$("div")` http://api.jquery.com/element-selector
"s1 s2"	Seleciona todos os elementos que combinam com o seletor **s2** e são descendentes de elementos que combinam com o seletor **s1**. Exemplo: selecionando os spans descendentes de um div. `$("div span")` http://api.jquery.com/descendant-selector
"s1 > s2"	Seleciona todos os elementos que combinam com o seletor **s2** e são filhos de elementos que combinam com o seletor **s1**. Exemplo: selecionando os spans filhos de um div. `$("div > span")` http://api.jquery.com/child-selector

Seletor	Descrição
"s1 + s2"	Seleciona todos os elementos que combinam com o seletor **s2** e são irmãos sucessores adjacentes de elementos que combinam com o seletor **s1**. Exemplo: selecionando os ps sucessores adjacentes de um h1. `$("h1 + p")` http://api.jquery.com/next-adjacent-Selector
"s1 ~ s2"	Seleciona todos os elementos que combinam com o seletor **s2** e são irmãos sucessores de elementos que combinam com o seletor **s1**. Exemplo: selecionando os ps sucessores de um h1. `$("h1 ~ p")` http://api.jquery.com/next-siblings-selector
".classe"	Seleciona todos os elementos da classe especificada. Exemplo: selecionando os elementos da classe titulo. `$(".titulo")` http://api.jquery.com/class-selector
"s1,s2,s3..."	Seleciona todos os elementos que combinam com pelo menos um dos seletores especificados. Exemplo: selecionando os divs ou os elementos da classe titulo. `$("div, .titulo")` http://api.jquery.com/multiple-selector
:first	Seleciona o primeiro elemento da lista dos elementos que combinam com o seletor utilizado. Exemplo: selecionando o primeiro div. `$("div.first")` http://api.jquery.com/first-selector
:last	Seleciona o último elemento da lista dos elementos que combinam com o seletor utilizado. Exemplo: selecionando o último div. `$("div.last")` http://api.jquery.com/last-selector
:eq(n)	Seleciona o n-ésimo elemento da lista dos elementos que combinam com o seletor utilizado. Exemplo: selecionando o quarto div. `$("div.eq(3)")` http://api.jquery.com/eq-selector
:gt(n)	Da lista de elementos que combinam com o seletor utilizado, seleciona do (n+1)-ésimo elemento até o último. Exemplo: selecionando do quarto ao último div. `$("div.gt(2)")` http://api.jquery.com/gt-selector

Seletor	Descrição
:lt(n)	Da lista de elementos que combinam com o seletor utilizado, seleciona do primeiro elemento até o (n-1)-ésimo. Exemplo: selecionando do primeiro ao décimo div. `$("div.lt(10)")` http://api.jquery.com/lt-selector
:even	Da lista de elementos que combinam com o seletor utilizado, seleciona todos os elementos de índice par. Exemplo: selecionando os divs de índice 0, 2, 4, 6 . . . `$("div.even")` http://api.jquery.com/even-selector
:odd	Da lista de elementos que combinam com o seletor utilizado, seleciona todos os elementos de índice ímpar. Exemplo: selecionando os divs de índice 1, 3, 5, 7 . . . . `$("div.odd")` http://api.jquery.com/odd-selector
:first-child	Seleciona os elementos que não possuem um irmão antecessor. Exemplo: selecionando os divs que não possuem um irmão antecessor. `$("div.first-child")` http://api.jquery.com/first-child-selector
:first-of-type	Seleciona os elementos que não possuem um irmão antecessor do mesmo tipo. Exemplo: selecionando os divs que não possuem um irmão div antecessor. `$("div.first-of-type")` http://api.jquery.com/first-of-type-selector
:last-child	Seleciona os elementos que não possuem um irmão sucessor. Exemplo: selecionando os divs que não possuem um irmão sucessor. `$("div.last-child")` http://api.jquery.com/last-child-selector
:last-of-type	Seleciona os elementos que não possuem um irmão sucessor do mesmo tipo. Exemplo: selecionando os divs que não possuem um irmão div sucessor. `$("div.last-of-type")` http://api.jquery.com/last-of-type-selector
:nth-child(n)	Seleciona os elementos que possuem (n - 1) irmãos antecessores. Exemplo: selecionando os divs que possuem exatamente 3 irmãos antecessores. `$("div.nth-child(4)")` http://api.jquery.com/nth-child-selector

Seletor	Descrição
:nth-last-child(n)	Seleciona os elementos que possuem (n - 1) irmãos sucessores. Exemplo: selecionando os divs que possuem exatamente 3 irmãos sucessores. `$("div.nth-child(4)")` http://api.jquery.com/nth-last-child-selector
:nth-of-type(n)	Seleciona os elementos que possuem (n - 1) irmãos antecessores do mesmo tipo. Exemplo: selecionando os divs que possuem exatamente 3 irmãos div antecessores. `$("div.nth-of-type(4)")` http://api.jquery.com/nth-of-type-selector
:nth-last-of-type(n)	Seleciona os elementos que possuem (n - 1) irmãos sucessores do mesmo tipo. Exemplo: selecionando os divs que possuem exatamente 3 irmãos div sucessores. `$("div.nth-last-of-type(4)")` http://api.jquery.com/nth-last-of-type-selector
:only-child	Seleciona todos os elementos que não possuem irmãos. Exemplo: selecionando os divs que não possuem irmãos. `$("div.only-child")` http://api.jquery.com/only-child-selector
:only-of-type	Seleciona todos os elementos que não possuem irmãos do mesmo tipo. Exemplo: selecionando os divs que não possuem irmãos div. `$("div.only-of-type")` http://api.jquery.com/only-of-type-selector
:parent	Seleciona todos os elementos que possuem filhos. Exemplo: selecionando os divs que possuem filho. `$("div.parent")` http://api.jquery.com/parent-selector
:hidden	Seleciona os elementos considerados **hidden**. Os elementos que não ocupam espaço na página são considerados **hidden**. Exemplo: selecionando os divs hidden. `$("div.hidden")` http://api.jquery.com/hidden-selector
:visible	Seleciona os elementos considerados **visible**. Os elementos que ocupam espaço na página são considerados **visible**. Exemplo: selecionando os divs visible. `$("div.visible")` http://api.jquery.com/visible-selector

Seletor	Descrição
:disabled	Seleciona todos os elementos desabilitados. Exemplo: selecionando os inputs desabilitados. `$("input.disabled")` http://api.jquery.com/disabled-selector
:enabled	Seleciona todos os elementos habilitados. Exemplo: selecionando os inputs habilitados. `$("input.enabled")` http://api.jquery.com/enabled-selector
:contains("texto")	Seleciona todos os elementos que possuem o texto especificado em seu conteúdo. Exemplo: selecionando os divs que contém o texto k19. `$("div.contains('k19')")` http://api.jquery.com/contains-selector
:empty	Seleciona todos os elementos sem conteúdo. Exemplo: selecionando os divs que não possuem conteúdo. `$("div.empty")` http://api.jquery.com/empty-selector
:has(seletor)	Seleciona os elementos que possuem pelo menos um descendente que combina com o seletor passado como parâmetro. Exemplo: selecionando os divs que possuem pelo menos um img. `$("div.has(img))` http://api.jquery.com/has-selector
:not(seletor)	Seleciona os elementos que não combinam com o seletor especificado. Exemplo: selecionando os elementos que não são div. `$(".not(div)")` http://api.jquery.com/not-selector
:button	Seleciona todos os elementos **button** ou com **type="button"**. Exemplo: selecionando os inputs com type='button'. `$("input.button")` http://api.jquery.com/button-selector
:file	Seleciona todos os elementos com **type="file"**. Exemplo: selecionando os inputs com type='file'. `$("input.file")` http://api.jquery.com/file-selector

Seletor	Descrição
:image	Seleciona todos os elementos com **type="image"**. Exemplo: selecionando os inputs com type='image'. `$("input.image")` http://api.jquery.com/image-selector
:password	Seleciona todos os elementos com **type="password"**. Exemplo: selecionando os inputs com type='password'. `$("input.password")` http://api.jquery.com/password-selector
:radio	Seleciona todos os elementos com **type="radio"**. Exemplo: selecionando os inputs com type='radio'. `$("input.radio")` http://api.jquery.com/radio-selector
:checkbox	Seleciona todos os elementos com **type="checkbox"**. Exemplo: selecionando os inputs com type='checkbox'. `$("input.checkbox")` http://api.jquery.com/checkbox-selector
:reset	Seleciona todos os elementos com **type="reset"**. Exemplo: selecionando os inputs com type='reset'. `$("input.reset")` http://api.jquery.com/reset-selector
:text	Seleciona os elementos com **type="text"**. Exemplo: selecionando os inputs com type='text'. `$("input.text")` http://api.jquery.com/text-selector
:submit	Seleciona os elementos com **type="submit"**. Exemplo: selecionando os inputs com type='submit'. `$("input.submit")` http://api.jquery.com/submit-selector
:input	Seleciona todos os elementos **input** e **textarea**. http://api.jquery.com/input-selector
:header	Seleciona os títulos (**h1**,**h2**,**h3**,**h4**,**h5** e **h6**). http://api.jquery.com/header-selector
:lang(linguagem)	Seleciona todos os elementos com **lang** igual à linguagem especificada. Exemplo: selecionando os divs com lang='pt-br'. `$("div.lang('pt-br')` http://api.jquery.com/lang-selector

Seletor	Descrição
:animated	Seleciona todos os elementos que estão com alguma animação em andamento. Exemplo: selecionando os divs com animação em andamento. `$("div.animated")` http://api.jquery.com/animated-selector
:root	Seleciona o elemento raiz do documento. http://api.jquery.com/root-selector
:selected	Seleciona os elementos **option** marcados. http://api.jquery.com/selected-selector
:checked	Seleciona todos os checkboxes, radios e options marcados. http://api.jquery.com/checked-selector
:focus	Seleciona o elemento que tem o foco. http://api.jquery.com/focus-selector
:target	Seleciona o elemento com **id** igual ao fragmento da URL. http://api.jquery.com/target-selector

Exercícios de Fixação

8 No projeto **jQuery**, crie um arquivo chamado **seletores.html**.

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - jQuery</title>
    <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
    <script src="seletores.js"></script>
  </head>
  <body>
    <h1 id="titulo">K19 - jQuery</h1>

    <form>
      <label for="nome">Nome. </label>
      <input id="nome" type="text" placeHolder="Jonas Hirata" required>

      <label for="email">Email. </label>
      <input id="email" type="text" placeHolder="jonas.hirata@email.com">

      <input type="submit" value="Enviar">
    </form>

    <ul>
      <li>K01 - Lógica de Programação</li>
      <li>K02 - Desenvolvimento Web com HTML, CSS e JavaScript</li>
      <li>K03 - Lógica de Programação</li>
    </ul>
  </body>
</html>
```

Código HTML 5.8: seletores.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao8.zip

9 No projeto **jQuery**, crie um arquivo chamado **seletores.js**.

```
$(document).ready(function() {
  $("input[type='text'][required]").css({
    "background-color". "yellow"
  }),

  $("label.first").css({
    "color". "red"
  }),

  $("input.last").css({
    "color". "blue"
  }),

  $("li.even").css({
    "background-color". "gray"
  }),

  $("input.eq(1)").css({
    "background-color". "green"
  }),

  $(".header").css({
    "color". "darkgreen"
  }),
}),
```

Código Javascript 5.63: seletores.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao9.zip

10 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/jQuery/public_html/seletores.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/jQuery/public_html/seletores.html.

Efeitos e Animações

Efeito	Descrição
fadeIn	Faz os elementos se tornarem opacos gradativamente. http.//api.jquery.com/fadeIn
fadeOut	Faz os elementos se tornarem transparentes gradativamente. http.//api.jquery.com/fadeOut
fadeToggle	Faz os elementos opacos se tornarem transparentes gradativamente e os elementos transparentes se tornarem opacos gradativamente. http.//api.jquery.com/fadeToggle
fadeTo(opacidade)	Modifica a opacidade dos elementos gradativamente até atingir a opacidade especificada. http.//api.jquery.com/fadeTo
hide	Esconde os elementos. http.//api.jquery.com/hide

Efeito	Descrição
show	Exibe os elementos. http://api.jquery.com/show
toggle	Esconde os elementos que estão sendo exibidos e exibe os elementos que estão escondidos. http://api.jquery.com/toggle
slideDown	Exibe os elementos com efeito de deslizamento. http://api.jquery.com/slideDown
slideUp	Esconde os elementos com efeito de deslizamento. http://api.jquery.com/slideUp
slideToggle	Exibe os elementos que estão escondidos com efeito de deslizamento. Esconde os elementos que estão sendo exibidos com efeito de deslizamento. http://api.jquery.com/slideToggle
animate	Modifica gradativamente determinadas propriedades CSS até atingir os valores especificados. http://api.jquery.com/animate

Duração

Podemos definir a duração dos efeitos ou animações em milissegundos. No exemplo abaixo, o tempo de execução do efeito **fadeOut** foi definido como 1000 milissegundos.

```
$("a").click(function(){
  $(this).fadeOut(1000),
}),
```

Callback

Podemos definir uma função para ser executada ao término da execução dos efeitos ou das animações. No exemplo abaixo, uma função que exibe a mensagem "terminou o fadeOut" foi associada ao término do efeito **fadeOut**.

```
$("a").click(function(){
  $(this).fadeOut(1000, function() {
    console.log("terminou o fadeOut"),
  }),
}),
```

Exercícios de Fixação

11 No projeto **jQuery**, crie um arquivo chamado **efeitos.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - jQuery</title>
    <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
    <script src="efeitos.js"></script>
    <style type="text/css">
      div {
        width. 1000px,
```

```
        height. 100px,
        background-color. yellow,
        border. 1px solid black,
      }
    </style>
  </head>
  <body>
    <button id="fadeIn">fadeIn</button>
    <button id="fadeOut">fadeOut</button>
    <button id="fadeToggle">fadeToggle</button>
    <button id="fadeTo1">fadeTo 0.5</button>
    <button id="fadeTo2">fadeTo 1.0</button>
    <button id="hide">hide</button>
    <button id="show">show</button>
    <button id="toggle">toggle</button>
    <button id="slideDown">slideDown</button>
    <button id="slideUp">slideUp</button>
    <button id="slideToggle">slideToggle</button>
    <button id="animate1">animate1</button>
    <button id="animate2">animate2</button>
    <div id="div"></div>
  </body>
</html>
```

Código HTML 5.9: efeitos.html

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao11.zip

12 No projeto **jQuery**, crie um arquivo chamado **efeitos.js**.

```
$(document).ready(function(){
  /* fadeIn */
  $("#fadeIn").click(function(){
    $("#div").fadeIn(),
  }),

  /* fadeOut */
  $("#fadeOut").click(function(){
    $("#div").fadeOut(),
  }),

  /* fadeToggle */
  $("#fadeToggle").click(function(){
    $("#div").fadeToggle(),
  }),

  /* fadeTo1 */
  $("#fadeTo1").click(function(){
    $("#div").fadeTo("fast", 0.5),
  }),

  /* fadeTo2 */
  $("#fadeTo2").click(function(){
    $("#div").fadeTo("fast", 1.0),
  }),

  /* hide */
  $("#hide").click(function(){
    $("#div").hide(),
  }),

  /* show */
  $("#show").click(function(){
    $("#div").show(),
  }),

  /* toggle */
```

```
38   $("#toggle").click(function(){
39     $("#div").toggle(),
40   }),
41
42   /* slideDown */
43   $("#slideDown").click(function(){
44     $("#div").slideDown(),
45   }),
46
47   /* slideUp */
48   $("#slideUp").click(function(){
49     $("#div").slideUp(),
50   }),
51
52   /* slideToggle */
53   $("#slideToggle").click(function(){
54     $("#div").slideToggle(),
55   }),
56
57   /* animate1 */
58   $("#animate1").click(function(){
59     $("#div").animate({
60       "border-width". "5px",
61       "margin-top". "100px"
62     }),
63   }),
64
65   /* animate2 */
66   $("#animate2").click(function(){
67     $("#div").animate({
68       "border-width". "1px",
69       "margin-top". "0"
70     }),
71   }),
72 }),
```

Código Javascript 5.66: efeitos.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao12.zip

13 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/jQuery/public_html/efeitos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/jQuery/public_html/efeitos.html.

Manipulação

Método	Descrição
addClass	Adiciona uma ou mais classes aos elementos selecionados. Exemplo: adicionando as classes destaque e grid nos divs. `$("div").addClass("destaque grid")` http.//api.jquery.com/addClass

Método	Descrição

removeClass	Remove uma, várias ou todas as classes dos elementos selecionados.
	
	Exemplo: removendo a classe grid dos divs.
	
	`$("div").removeClass("grid"),`
	
	Exemplo: removendo as classes grid e destaque dos divs.
	
	`$("div").removeClass("grid destaque"),`
	
	Exemplo: removendo todas as classes dos divs.
	
	`$("div").removeClass(),`
	
	http://api.jquery.com/removeClass

toggleClass	Inverte a presença de uma ou mais classes nos elementos selecionados. Exemplo: invertendo a presença das classes grid e destaque nos divs. `$("div").toggleClass("grid destaque"),` http://api.jquery.com/toggleClass
hasClass	Verifica se algum dos elementos selecionados é da classe especificada. Exemplo: verificando se algum parágrafo é da classe destaque. `var resposta= $("p").hasClass("destaque")` http://api.jquery.com/hasClass
after	Adiciona um determinado conteúdo imediatamente depois de cada um dos elementos selecionados. Exemplo: adicionando um parágrafo imediatamente depois dos h1. `$("h1").after("<p>subtítulo</p>")` http://api.jquery.com/after
before	Adiciona um determinado conteúdo imediatamente antes de cada um dos elementos selecionados. Exemplo: adicionando um hr imediatamente antes dos tables. `$("table").before("<hr>")` http://api.jquery.com/before
insertAfter	Semelhante ao **after** com sintaxe levemente diferente. Exemplo: adicionando um parágrafo imediatamente depois dos h1. `$(<p>subtítulo</p>).after("h1")` http://api.jquery.com/insertAfter
insertBefore	Semelhante ao **before** com sintaxe levemente diferente. Exemplo: adicionando um hr imediatamente antes dos tables. `$("<hr>").before("table")` http://api.jquery.com/insertBefore

Método	Descrição
append	Adiciona um determinado conteúdo no final do corpo de cada um dos elementos selecionados. Exemplo: adicionando o texto (fonte: K19) no final de cada p. `$("p").append("<em>(fonte K19)</em>")` http://api.jquery.com/append
prepend	Adiciona um determinado conteúdo no começo do corpo de cada um dos elementos selecionados. Exemplo: adicionando o texto (fonte: K19) no começo de cada p. `$("p").prepend("<em>(fonte K19)</em>")` http://api.jquery.com/prepend
appendTo	Semelhante ao **append** com sintaxe levemente diferente. Exemplo: adicionando o texto (fonte: K19) no final de cada p. `$("<em>(fonte K19)</em>").appendTo("p")` http://api.jquery.com/appendTo
prependTo	Semelhante ao **prepend** com sintaxe levemente diferente. Exemplo: adicionando o texto (fonte: K19) no começo de cada p. `$("<em>(fonte K19)</em>").prependTo("p")` http://api.jquery.com/prependTo

attr	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar os valores dos atributos do primeiro elemento selecionado. O segundo é alterar os valores dos atributos de todos os elementos selecionados.
	
	Exemplo: recuperando o id do primeiro div.
	
	`var id = $("div").attr("id"),`
	
	Exemplo: alterando os atributos src, title e alt dos imgs.
	
	`$("img").attr({ src. "http://www.k19.com.br/figs/main-header-logo.png", title. "K19", alt. "K19 Logo" }),`
	
	http://api.jquery.com/attr

prop	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar os valores das propriedades do primeiro elemento selecionado. O segundo é alterar os valores das propriedades de todos os elementos selecionados.
	
	Exemplo: recuperando o valor da propriedade checked do primeiro input com type='checkbox'.
	
	`var checked = $("input['checkbox']").prop("checked"),`
	
	Exemplo: alterando o valor da propriedade checked do primeiro input com type='checkbox'.
	
	`$("input['checkbox']").prop("checked", false)`
	
	http://api.jquery.com/prop

Método	Descrição

css	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar os valores das propriedades CSS do primeiro elemento selecionado. O segundo é alterar os valores das propriedades CSS de todos os elementos selecionados.
	
	Exemplo: recuperando a propriedade CSS color do primeiro div.
	
	`var color = $("div").css("color"),`
	
	Exemplo: alterando as propriedades color, width e height dos divs.
	
	`$("div").css({ color. "red", width. "300px", height. "150px" }),`
	
	http.//api.jquery.com/css

removeAttr	Remove o atributo especificado dos elementos selecionados. Exemplo: removendo o atributo title dos imgs. `$("img").removeAttr("title"),` http.//api.jquery.com/removeAttr
removeProp	Remove a propriedade especificada dos elementos selecionados. Exemplo: removendo a propriedade title dos imgs. `$("img").removeProp("title"),` http.//api.jquery.com/removeProp
clone	Cria uma cópia dos elementos selecionados. Exemplo: criando uma cópia de todos os parágrafos. `var paragrafos = $("p").clone(),` http.//api.jquery.com/clone
detach	Remove os elementos selecionados da árvore de elementos e os devolve. Exemplo: retirando os parágrafos. `var paragrafos = $("p").detach(),` http.//api.jquery.com/detach
empty	Remove o conteúdo dos elementos selecionados. Exemplo: removendo o conteúdo dos parágrafos. `$("p").empty(),` http.//api.jquery.com/empty
remove	Remove os elementos selecionados da árvore de elementos. Exemplo: removendo os parágrafos. `$("p").remove(),` http.//api.jquery.com/remove

Método	Descrição
replaceAll	Substitui todos os elementos selecionados. Exemplo: substituindo todos os parágrafos por K19 . `$("<p>K19</p>").replaceAll("p"),` http://api.jquery.com/replaceAll
replaceWith	Semelhante ao **replaceAll** com sintaxe levemente diferente. Exemplo: substituindo todos os parágrafos por K19 . `$("p").replaceWith("<p>K19</p>"),` http://api.jquery.com/replaceWith

height	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar a altura do primeiro elemento selecionado. O segundo é alterar a altura de todos os elementos selecionados.
	
	Exemplo: recuperando a altura do primeiro div.
	
	`var altura = $("div").height(),`
	
	Exemplo: alterando altura dos divs.
	
	`$("div").height(100),`
	
	http://api.jquery.com/height

width	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar a largura do primeiro elemento selecionado. O segundo é alterar a largura de todos os elementos selecionados.
	
	Exemplo: recuperando a largura do primeiro div.
	
	`var largura = $("div").width(),`
	
	Exemplo: alterando largura dos divs.
	
	`$("div").width(100),`
	
	http://api.jquery.com/width

innerHeight	Recupera o **innerHeight** do primeiro elemento selecionado. O **innerHeight** é a soma da altura, margem interna inferior e margem interna superior. Exemplo: recuperando o innerHeight do primeiro div. `var innerHeight = $("div").innerHeight(),` http://api.jquery.com/innerHeight
innerWidth	Recupera o **innerWidth** do primeiro elemento selecionado. O **innerWidth** é a soma da largura, margem interna da esquerda e margem interna da direita. Exemplo: recuperando o innerWidth do primeiro div. `var innerWidth = $("div").innerWidth(),` http://api.jquery.com/innerWidth

Método	Descrição

outerHeight	Recupera o **outerHeight** do primeiro elemento selecionado. Se esse método for chamado sem parâmetros, o **outerHeight** é a soma da altura, margem interna inferior, margem interna superior, borda inferior e borda superior. Se ele for chamado com o parâmetro **true**, o **outerHeight** é a soma da altura, margem interna inferior, margem interna superior, borda inferior, borda superior, margem externa inferior e margem externa superior.
	
	Exemplo: recuperando o outerHeight do primeiro div.
	
	`var outerHeight1 = $("div").outerHeight(),`
	`var outerHeight2 = $("div").outerHeight(true),`
	
	http://api.jquery.com/outerHeight

outerWidth	Recupera o **outerWidth** do primeiro elemento selecionado. Se esse método for chamado sem parâmetros, o **outerWidth** é a soma da largura, margem interna da esquerda, margem interna da direita, borda da esquerda e borda da direita. Se ele for chamado com o parâmetro **true**, o **outerWidth** é a soma da largura, margem interna da esquerda, margem interna da direita, borda da esquerda, borda da direita, margem externa da esquerda e margem externa da direita.
	
	Exemplo: recuperando o outerWidth do primeiro div.
	
	`var outerWidth1 = $("div").outerWidth(),`
	`var outerWidth2 = $("div").outerWidth(true),`
	
	http://api.jquery.com/outerWidth

html	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar o conteúdo HTML do primeiro elemento selecionado. O segundo é alterar o conteúdo HTML de todos os elementos selecionados.
	
	Exemplo: recuperando o conteúdo HTML do primeiro div.
	
	`var conteudo = $("div").html(),`
	
	Exemplo: alterando o conteúdo HTML dos divs.
	
	`$("div").html("<h1>K19<h/1><p>Cursos da K19.</p>"),`
	
	http://api.jquery.com/html

text	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar o o texto contido no corpo do primeiro elemento selecionado. O segundo é alterar o texto contido no corpo de todos os elementos selecionados.
	
	Exemplo: recuperando o texto contido no corpo do primeiro div.
	
	`var texto = $("div").text(),`
	
	Exemplo: alterando o texto contido no corpo dos divs.
	
	`$("div").text("K19 Cursos"),`
	
	http://api.jquery.com/text

val	Esse método é utilizado para recuperar o valor de elementos como **input**, **textarea** e **select**. Ele considera o primeiro elemento selecionado. Exemplo: recuperador o valor do primeiro input. `var valor = $("input").val(),` http://api.jquery.com/val

Método	Descrição
wrap	Adiciona na árvore de elementos uma estrutura envolvendo cada um dos elementos selecionados. Exemplo: Envolvendo cada parágrafo com a estrutura <section> </section>. `$("p").wrap("<section><div></div></section>"),` http.//api.jquery.com/wrap
wrapAll	Adiciona na árvore de elementos uma estrutura envolvendo o conjunto dos elementos selecionados. Exemplo: Envolvendo os parágrafos com a estrutura <section> </section>. `$("p").wrapAll("<section><div></div></section>"),` http.//api.jquery.com/wrapAll
wrapInner	Adiciona na árvore de elementos uma estrutura envolvendo o conteúdo de cada um dos elementos selecionados. Exemplo: Envolvendo o conteúdo dos parágrafos com a estrutura . `$("p").wrapInner("<em></em>"),` http.//api.jquery.com/wrapInner
unwrap	Remove o elemento pai dos elementos selecionados. Exemplo: removendo os elementos pai dos parágrafos. `$("p").unwrap(),` http.//api.jquery.com/unwrap

offset	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar a posição do primeiro elemento selecionado. O segundo é alterar a posição de todos os elementos selecionados. Nessa alteração, os elementos com **position: static** passam para **position: relative**.
	
	Exemplo: recuperando o offset do primeiro div.
	
	`var offset = $("div").offset(),`
	`var left = offset.left,`
	`var top = offset.top,`
	
	Exemplo: alterando o offset dos divs.
	
	`$("div").offset({`
	`  left. 100,`
	`  top. 20,`
	`}),`
	
	http.//api.jquery.com/offset

position	Recupera a posição do primeiro elemento selecionado em relação ao elemento pai.
	
	Exemplo: recuperando a posição do primeiro div.
	
	`var position = $("div").position(),`
	`var left = position.left,`
	`var top = position.top,`
	
	http.//api.jquery.com/position

Método	Descrição

scrollLeft	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar a posição da barra de rolagem horizontal do primeiro elemento selecionado. O segundo é alterar a posição da barra de rolagem horizontal de todos os elementos selecionados.
	
	Exemplo: recuperando a posição da barra de rolagem horizontal do primeiro div.
	
	`var scrollLeft = $("div").scrollLeft();`
	
	Exemplo: alterando a posição da barra de rolagem horizontal dos divs.
	
	`$("div").scrollLeft(300);`
	
	http.//api.jquery.com/scrollLeft

scrollTop	Esse método pode ser utilizado para dois propósitos. O primeiro é recuperar a posição da barra de rolagem vertical do primeiro elemento selecionado. O segundo é alterar a posição da barra de rolagem vertical de todos os elementos selecionados.
	
	Exemplo: recuperando a posição da barra de rolagem vertical do primeiro div.
	
	`var scrollTop = $("div").scrollTop();`
	
	Exemplo: alterando a posição da barra de rolagem vertical dos divs.
	
	`$("div").scrollTop(300);`
	
	http.//api.jquery.com/scrollTop

Exercícios de Fixação

14 No projeto **jQuery**, crie um arquivo chamado **manipulacao.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - jQuery</title>
    <style>
      .destaque {
        background-color. yellow,
      }
    </style>
    <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
    <script src="manipulacao.js"></script>
  </head>
  <body>
    <button id="addClass">addClass</button>
    <button id="removeClass">removeClass</button>
    <button id="toggleClass">toggleClass</button>
    <button id="hasClass">hasClass</button>

    <button id="after">after</button>
    <button id="before">before</button>
    <button id="insertAfter">insertAfter</button>
    <button id="insertBefore">insertBefore</button>
    <button id="append">append</button>
    <button id="prepend">prepend</button>
    <button id="appendTo">appendTo</button>
    <button id="prependTo">prependTo</button>

```

```
    <button id="height">height</button>
    <button id="width">width</button>
    <button id="innerHeight">innerHeight</button>
    <button id="innerWidth">innerWidth</button>
    <button id="outerHeight">outerHeight</button>
    <button id="outerWidth">outerWidth</button>

    <div id="div1">
      <ul>
        <li>Jonas Hirata</li>
        <li>Marcelo Martins</li>
        <li>Rafael Cosentino</li>
      </ul>
    </div>
  </body>
</html>
```

Código HTML 5.10: manipulacao.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao14.zip

15 No projeto **jQuery**, crie um arquivo chamado **manipulacao.js**.

```
$(document).ready(function() {
  $("#div1").css({
    "border". "2px solid black",
    "width". "800px",
    "height". "200px",
    "margin". "15px",
    "padding". "15px"
  }),

  $("#addClass").click(function(){
    $("#div1").addClass("destaque"),
  }),

  $("#removeClass").click(function(){
    $("#div1").removeClass("destaque"),
  }),

  $("#toggleClass").click(function(){
    $("#div1").toggleClass("destaque"),
  }),

  $("#hasClass").click(function(){
    var destaque = $("#div1").hasClass("destaque"),
    alert(destaque ? "Com destaque" . "Sem destaque"),
  }),

  $("#after").click(function(){
    $("#p1").remove(),
    $("#div1").after("<p id='p1'>after<p>"),
  }),

  $("#before").click(function(){
    $("#p1").remove(),
    $("#div1").before("<p id='p1'>before<p>"),
  }),

  $("#insertAfter").click(function(){
    $("#p1").remove(),
    $("<p id='p1'>insertAfter<p>").insertAfter("#div1"),
  }),

  $("#insertBefore").click(function(){
    $("#p1").remove(),
    $("<p id='p1'>insertBefore<p>").insertBefore("#div1"),
```

```
  }),

  $("#append").click(function(){
    $("#p1").remove(),
    $("#div1").append("<p id='p1'>append<p>"),
  }),

  $("#prepend").click(function(){
    $("#p1").remove(),
    $("#div1").prepend("<p id='p1'>prepend<p>"),
  }),

  $("#appendTo").click(function(){
    $("#p1").remove(),
    $("<p id='p1'>appendTo<p>").appendTo("#div1"),
  }),

  $("#prependTo").click(function(){
    $("#p1").remove(),
    $("<p id='p1'>prependTo<p>").prependTo("#div1"),
  }),

  $("#height").click(function(){
    $("#p1").remove(),
    var height = $("#div1").height(),
    $("#div1").after("<p id='p1'>height. " + height +"<p>"),
  }),

  $("#width").click(function(){
    $("#p1").remove(),
    var width = $("#div1").width(),
    $("#div1").after("<p id='p1'>width. " + width +"<p>"),
  }),

  $("#innerHeight").click(function(){
    $("#p1").remove(),
    var innerHeight = $("#div1").innerHeight(),
    $("#div1").after("<p id='p1'>innerHeight. " + innerHeight +"<p>"),
  }),

  $("#innerWidth").click(function(){
    $("#p1").remove(),
    var innerWidth = $("#div1").innerWidth(),
    $("#div1").after("<p id='p1'>innerWidth. " + innerWidth +"<p>"),
  }),

  $("#outerHeight").click(function(){
    $("#p1").remove(),
    var outerHeight = $("#div1").outerHeight(),
    $("#div1").after("<p id='p1'>outerHeight. " + outerHeight +"<p>"),
  }),

  $("#outerWidth").click(function(){
    $("#p1").remove(),
    var outerWidth = $("#div1").outerWidth(),
    $("#div1").after("<p id='p1'>outerWidth. " + outerWidth +"<p>"),
  }),
}),
```

Código Javascript 5.119: manipulacao.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao15.zip

16 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/jQuery/public_html/manipulacao.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/jQuery/public_html/manipulacao.html.

Mais métodos

Método	Descrição
add	Adiciona elementos a um conjunto. Exemplo: adicionando os spans ao conjunto de divs. `$("div").add("span"),` http.//api.jquery.com/add
parent	Recupera os pais de cada elemento de um conjunto. Exemplo: recuperando os pais dos divs e dos spans. `$("div, span").parent(),` http.//api.jquery.com/parent
parents	Recupera os ancestrais de cada elemento de um conjunto. Exemplo: recuperando os ancestrais dos divs e dos spans. `$("div, span").parents(),` http.//api.jquery.com/parents
siblings	Recupera os irmãos de cada elemento de um conjunto. Exemplo: recupera os irmãos dos divs e dos spans. `$("div, span").siblings()` http.//api.jquery.com/siblings

children	Recupera os filhos de cada elemento de um conjunto. Opcionalmente, pode receber um seletor como parâmetro para filtrar o resultado.
	
	Exemplo: recuperando os filhos dos divs e dos spans.
	
	`$("div, span").children(),`
	
	Exemplo: recuperando os filhos dos divs e dos spans que possuam a classe **destaque**.
	
	`$("div, span").children(".destaque"),`
	
	http.//api.jquery.com/children

find	Recupera os descendentes de cada elemento de um conjunto. Opcionalmente, pode receber um seletor como parâmetro para filtrar o resultado.
	
	Exemplo: recuperando os descendentes dos divs e dos spans.
	
	`$("div, span").find(),`
	
	Exemplo: recuperando os descendentes dos divs e dos spans que possuam a classe **destaque**.
	
	`$("div, span").find(".destaque"),`
	
	http.//api.jquery.com/find

Método	Descrição

each	Permite executar uma função para cada elemento de um conjunto.
	
	Exemplo: exibe no console a largura dos divs.
	
	`$("div").each(function(indice, elemento) {`
	`  console.log($(elemento).width()),`
	`}),`
	
	http://api.jquery.com/each

map	Permite executar uma função para cada elemento de um conjunto. Essa função pode devolver um valor. Esses valores serão devolvidos em um objeto jQuery.
	
	Exemplo: recupera a largura dos divs.
	
	`function pegaAltura(indice, elemento) {`
	`  return $(elemento).width(),`
	`}`
	
	`var alturas = $("div").map(pegaAltura),`
	
	http://api.jquery.com/map

filter	Filtra os elementos de conjunto. Exemplo: recuperando os divs da classe grid. `$("div").filter(".grid"),` http://api.jquery.com/filter
has	Devolve os elementos de um conjunto que possuem um descendente que combina com o seletor especificado. Exemplo: recupera os divs e os spans que possuem um elemento da classe destaque como descendente. `$("div, span").has(".destaque")` http://api.jquery.com/has
not	Remove elementos de um conjunto. Exemplo: removendo os elementos da classe destaque do conjunto de divs e spans. `$("div, span").not(".destaque")` http://api.jquery.com/not
first	Devolve o primeiro elemento de uma lista. Exemplo: devolve o primeiro elemento da lista de divs e spans. `$("div, span").first(),` http://api.jquery.com/first
last	Devolve o último elemento de uma lista. Exemplo: devolve o último elemento da lista de divs e spans. `$("div, span").last()` http://api.jquery.com/last
eq	Recupera o elemento do índice especificado. Exemplo: recupera o sexto elemento da lista de divs e spans. `$("div, span").eq(5),` http://api.jquery.com/eq

Método	Descrição
slice	Recupera um determinado trecho de uma lista de elementos. Exemplo: recuperando do terceiro ao décimo elementos da lista de divs e spans. `$("div, span").slice(2, 9),` http://api.jquery.com/slice
next	Devolve os irmãos adjacentes sucessores dos elementos de um conjunto. Exemplo: recupera os irmãos adjacentes sucessores dos divs e spans. `$("div, span").next()` http://api.jquery.com/next
nextAll	Devolve os irmãos sucessores dos elementos de um conjunto. Exemplo: recupera os irmãos sucessores dos divs e spans. `$("div, span").nextAll()` http://api.jquery.com/nextAll
prev	Devolve os irmãos adjacentes antecessores dos elementos de um conjunto. Exemplo: recupera os irmãos adjacentes antecessores dos divs e spans. `$("div, span").prev()` http://api.jquery.com/prev
prevAll	Devolve os irmãos antecessores dos elementos de um conjunto. Exemplo: recupera os irmãos antecessores dos divs e spans. `$("div, span").prevAll()` http://api.jquery.com/prevAll

AJAX

A forma básica de interação entre os usuários e as páginas web é limitada. Quando o usuário clica em um link ou em um botão de uma página web, uma requisição HTTP é enviada ao servidor correspondente. Quando chegar no servidor, essa requisição será processada. No término desse processamento, o servidor enviará uma resposta HTTP contendo uma página web para a máquina do usuário. Ao receber essa resposta, o navegador do usuário carregará a página inteira.

Nessa abordagem, muitas vezes, ocorre um desperdício de tempo, pois, na maior parte dos casos, não seria necessário carregar a página inteira e sim pequenos "pedaços" dela. Mesmo assim, o navegador sempre carregará todo o conteúdo das páginas web.

Além disso, há outro problema nessa abordagem, o usuário não pode interagir com a página web durante o envio da requisição HTTP; processamento no servidor; envio da resposta HTTP e carregamento do página.

Para aumentar a interatividade entre os usuários e as páginas web, podemos utilizar a forma de interação conhecida como **AJAX (Asynchronous Javascript and XML)**. Nessa outra abordagem, os navegadores podem atualizar "pedaços" de uma página web sem ter carregá-la completamente.

Além disso, com AJAX, os usuários podem interagir com as páginas web durante o envio da requisição HTTP; processamento no servidor; envio da resposta HTTP e carregamento parcial das páginas web.

Utilizaremos os recursos da biblioteca **jQuery** para implementar a interação entre os usuários e as páginas web com AJAX.

load

Podemos obter conteúdo de Web Server através do método **load**. Esse método realiza requisições HTTP do tipo GET com AJAX. No exemplo abaixo, o conteúdo do documento HTML **k19.html** obtido do servidor e inserido no corpo do elemento com **id="conteudo"**.

```
$("#conteudo").load("k19.html"),
```

get

Podemos realizar requisições HTTP do tipo GET com AJAX através do método **get**. Esse método recebe como parâmetro a URL correspondente à requisição que desejamos realizar. No exemplo abaixo, a requisição foi realizada para URL **k19.php** mas o resultado foi ignorado.

```
$.get("k19.php"),
```

Para recuperar o resultado da requisição devemos utilizar o método **done**. Devemos passar como argumento para esse método a função que tratará o resultado. Essa função receberá o resultado como parâmetro. Observe, no exemplo abaixo, que o resultado é exibido no console do navegador.

```
var get = $.get("k19.php"),

get.done(function(resultado) {
  console.log(resultado),
}),
```

Também podemos enviar dados para o servidor. Veja o exemplo a seguir.

```
var dados = {
  nome. "Rafael",
  empresa. "K19"
},

var get = $.get("k19.php", dados),

get.done(function(resultado) {
  console.log(resultado),
}),
```

Esses dados são enviados como parâmetro na URL da requisição.

```
k19.php?nome=Rafael&empresa=K19
```

post

Podemos realizar requisições HTTP do tipo POST com AJAX através do método **post**. Esse método funciona de forma semelhante ao método **get**. Veja alguns exemplos.

```
$.post("k19.php"),
```

```
var post = $.post("k19.php"),

post.done(function(resultado) {
  console.log(resultado),
}),
```

```
var dados = {
  nome. "Rafael",
  empresa. "K19"
},

var post = $.post("k19.php", dados),

post.done(function(resultado) {
  console.log(resultado),
}),
```

Esses dados são enviados corpo da requisição HTTP.

`k19.php?nome=Rafael&empresa=K19`

Exercícios de Fixação

17 No projeto **jQuery**, crie um arquivo chamado **ajax.php**.

```
<?php
$soma = $_REQUEST["x"] + $_REQUEST["y"],

echo $soma,
?>
```

Código HTML 5.11: ajax.php

Arquivo: `https.//github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao17.zip`

18 No projeto **jQuery**, crie um arquivo chamado **ajax.html**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 - jQuery</title>
    <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
    <script src="ajax.js"></script>
  </head>
  <body>
    <form action="ajax.php" method="post">
      <label for="x">X.</label>
      <input id="x" name="x" type="text">
      <label for="y">Y.</label>
      <input id="y" name="y" type="text">

      <input type="submit" value="Enviar">
    </form>

    <p id="resultado"></p>
  </body>
```

```
21 </html>
```

Código HTML 5.12: ajax.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao18.zip

19 No projeto **jQuery**, crie um arquivo chamado **ajax.js**.

```
1  $(document).ready(function() {
2    $("input[type='submit']").click(function() {
3      var x = $("#x").val(),
4      var y = $("#y").val(),
5
6      $.post("ajax.php", {"x". x, "y". y})
7        .done(function(soma) {
8          $("#resultado").html(soma),
9        }),
10
11     return false,
12
13   }),
14 }),
```

Código Javascript 5.148: ajax.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao19.zip

20 No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/jQuery/public_html/ajax.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/jQuery/public_html/ajax.html.

Exercícios Complementares

1 No projeto **jQuery**, crie um arquivo chamado **jquery-complementar.html**.

```
1  <!DOCTYPE html>
2  <html lang="pt-br" id="html">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>K19 - Eventos - jQuery</title>
6      <script src="http.//code.jquery.com/jquery-1.10.1.min.js"></script>
7      <script src="jquery-complementar.js"></script>
8      <style type="text/css">
9        #header {
10         top. 0px,
11         left. 0px,
12         width. 100%,
13         padding. 10px,
14         position. fixed,
15         background-color. white,
16         border-bottom. 10px black solid,
17       }
18
```

```
19        #saida {
20          margin. 0px auto,
21          width. 800px,
22          height. 200px,
23          overflow. auto,
24        }
25
26        #content {
27          width. 800px,
28          margin. 250px auto 0px,
29        }
30
31        pre {
32          padding. 10px,
33          border-radius. 15px,
34          background-color. #eeeeee,
35        }
36
37        #div1, #div2 {
38          background-color. yellow,
39          width. 200px,
40          height. 200px,
41        }
42
43        #div3 {
44          background-color. blue,
45          width. 50px,
46          height. 50px,
47          margin. auto,
48        }
49      </style>
50    </head>
51    <body id="body">
52      <div id="header">
53        <div id="saida"></div>
54      </div>
55      <div id="content">
56
57      <h1>ready</h1>
58  <pre id="ready-on">
59  $(document).ready(function() {
60    var saida = $("#saida"),
61    saida.append("evento ready disparado&lt,br&gt,"),
62  }),
63  </pre>
64
65      <h1>resize</h1>
66      <button id="resize">ON</button>
67  <pre>
68  /* ON */
69  $(window).resize(function() {
70    var largura = $(window).width(),
71    var altura = $(window).height(),
72    saida.append("resize. (" + largura + ", " + altura + ")"),
73    saida.append("&lt,br&gt,"),
74    saida.scrollTop(saida.prop("scrollHeight")),
75  }),
76
77  /* OFF */
78  $(window).off("resize"),
79  </pre>
80
81      <h1>scroll</h1>
82      <button id="scroll">ON</button>
83  <pre>
84  /* ON */
85  $(window).scroll(function() {
86    var x = $(window).scrollLeft(),
87    var y = $(window).scrollTop(),
88    saida.append("scroll. (" + x + ", " + y + ")"),
```

```
  saida.append("&lt,br&gt,"),
  saida.scrollTop(saida.prop("scrollHeight")),
}),

/* OFF */
$(window).off("scroll"),
</pre>

    <h1>focus, focusin, blur e focusout</h1>
    <button id="foco">ON</button>
<pre>
/* ON */
$("*").on("focus focusin blur focusout",
 function(event) {
    var tag = this.tagName,
    var id = this.id,
    var type = event.type,

    saida.append(type + "(" + tag + ", #" + id + ")"),
    saida.append("&lt,br&gt,"),
    saida.scrollTop(saida.prop("scrollHeight")),
  }
),

/* OFF */
$("*").off("focus focusin blur focusout"),
</pre>
<label>Teste.</label>
<input id="input1">

    <h1>select</h1>
    <button id="select">ON</button>
<pre>
/* ON */
$("#input2").select(function() {
  saida.append("select. (" + window.getSelection() + ")"),
  saida.append("&lt,br&gt,"),
  saida.scrollTop(saida.prop("scrollHeight")),
}),

/* OFF */
$("#input2").off("select"),
</pre>
<label>Teste.</label>
<input id="input2">

    <h1>change</h1>
    <button id="change">ON</button>
<pre>
/* ON */
$("input3").change(function() {
  saida.append("change. (" + $(this).val() + ")"),
  saida.append("&lt,br&gt,"),
  saida.scrollTop(saida.prop("scrollHeight")),
}),

/* OFF */
$("#input3").off("change"),
</pre>
<label>Teste.</label>
<input id="input3">

    <h1>keydown, keypress e keyup</h1>
    <button id="key">ON</button>
<pre>
/* ON */
$("#input4").on("keydown keypress keyup",
  function(event) {
    var type = event.type,

```

```
    saida.append(type + ". (" + event.which + ")"),
    saida.append("&lt,br&gt,"),
    saida.scrollTop(saida.prop("scrollHeight")),
  }
),

/* OFF */
$("#input4").off("keydown keypress keyup"),
</pre>
<label>Teste.</label>
<input id="input4">

    <h1>click, dblclick, mousedown, mouseup, mouseenter,
      mouseleave, hover, mouseover e mouseout</h1>
    <button id="mouse">ON</button>
<pre>
/* ON */
$("#div1").on("click dblclick mousedown mouseup " +
  "mouseenter mouseleave hover mouseover mouseout",
  function(event) {
    var type = event.type,

    saida.append(type + ". (" + event.which + ")"),
    saida.append("&lt,br&gt,"),
    saida.scrollTop(saida.prop("scrollHeight")),
  }
),

/* OFF */
$("#div1").off("click dblclick mousedown mouseup " +
  "mouseenter mouseleave hover mouseover mouseout"),
</pre>
<div id="div1"><div id="div3"></div></div>

    <h1>mousemove</h1>
    <button id="mousemove">ON</button>
<pre>
/* ON */
$("#div2").mousemove(function(event) {
  var x = event.pageX,
  var y = event.pageY,
  var type = event.type,

  saida.append(type + ". (" + x + ", " + y + ")"),
  saida.append("&lt,br&gt,"),
  saida.scrollTop(saida.prop("scrollHeight")),
}),

/* OFF */
$("#div2").off("mousemove"),
</pre>
<div id="div2"></div>
    </div>
  </body>
</html>
```

Código HTML 5.13: jquery-complementar.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-complementar1.zip

2 No projeto **jQuery**, crie um arquivo chamado **jquery-complementar.js**.

```
/* ready */
$(document).ready(function(){
  var saida = $("#saida"),
  saida.append("evento ready disparado<br>"),

```

```
6   /* resize */
7   var resize = false,
8   $("#resize").click(function() {
9     if(resize) {
10      $(window).off("resize"),
11      $("#resize").html("ON"),
12      resize = false,
13    } else {
14      $(window).resize(function() {
15        var largura = $(window).width(),
16        var altura = $(window).height(),
17        saida.append("resize. (" + largura + ", " + altura + ")"),
18        saida.append("<br>"),
19        saida.scrollTop(saida.prop("scrollHeight")),
20      }),
21      $("#resize").html("OFF"),
22      resize = true,
23    }
24  }),
25
26  /* scroll */
27  var scroll = false,
28  $("#scroll").click(function() {
29    if(scroll) {
30      $(window).off("scroll"),
31      $("#scroll").html("ON"),
32      scroll = false,
33    } else {
34      $(window).scroll(function() {
35        var x = $(window).scrollLeft(),
36        var y = $(window).scrollTop(),
37        saida.append("scroll. (" + x + ", " + y + ")"),
38        saida.append("<br>"),
39        saida.scrollTop(saida.prop("scrollHeight")),
40      }),
41      $("#scroll").html("OFF"),
42      scroll = true,
43    }
44  }),
45
46  /* focus focusin blur focusout */
47  var foco = false,
48  $("#foco").click(function() {
49    if(foco) {
50      $("*").off("focus focusin blur focusout"),
51      $("#foco").html("ON"),
52      foco = false,
53    } else {
54      $("*").on("focus focusin blur focusout",
55        function(event) {
56          var tag = this.tagName,
57          var id = this.id,
58          var type = event.type,
59
60          saida.append(type + "(" + tag + ", #" + id + ")"),
61          saida.append("<br>"),
62          saida.scrollTop(saida.prop("scrollHeight")),
63        }
64      ),
65      $("#foco").html("OFF"),
66      foco = true,
67    }
68  }),
69
70  /* select */
71  var select = false,
72  $("#select").click(function() {
73    if(select) {
74      $("#input2").off("select"),
75      $("#select").html("ON"),
```

```
      select = false,
    } else {
      $("#input2").select(function() {
        saida.append("select. (" + window.getSelection() + ")"),
        saida.append("<br>"),
        saida.scrollTop(saida.prop("scrollHeight")),
      }),
      $("#select").html("OFF"),
      select = true,
    }
  }),

  /* change */
  var change = false,
  $("#change").click(function() {
    if(change) {
      $("#input3").off("change"),
      $("#change").html("ON"),
      change = false,
    } else {
      $("#input3").change(function() {
        saida.append("change. (" + $(this).val() + ")"),
        saida.append("<br>"),
        saida.scrollTop(saida.prop("scrollHeight")),
      }),
      $("#change").html("OFF"),
      change = true,
    }
  }),

  /* key */
  var key = false,
  $("#key").click(function() {
    if(key) {
      $("#input4").off("keydown keypress keyup"),
      $("#key").html("ON"),
      key = false,
    } else {
      $("#input4").on("keydown keypress keyup",
        function(event) {
          var type = event.type,

          saida.append(type + ". (" + event.which + ")"),
          saida.append("<br>"),
          saida.scrollTop(saida.prop("scrollHeight")),
        }
      ),
      $("#key").html("OFF"),
      key = true,
    }
  }),

  /* mouse */
  var mouse = false,
  $("#mouse").click(function() {
    if(mouse) {
      $("#div1").off("click dblclick mousedown mouseup " +
          "mouseenter mouseleave hover mouseover mouseout"),
      $("#mouse").html("ON"),
      mouse = false,
    } else {
      $("#div1").on("click dblclick mousedown mouseup " +
        "mouseenter mouseleave hover mouseover mouseout",
        function(event) {
          var x = event.pageX,
          var y = event.pageY,
          var type = event.type,

          saida.append(type + ". (" + x + ", " + y + ")"),
          saida.append("<br>"),
```

```
146            saida.scrollTop(saida.prop("scrollHeight")),
147          }
148        ),
149        $("#mouse").html("OFF"),
150        mouse = true,
151      }
152    }),
153
154    /* mousemove */
155    var mousemove = false,
156    $("#mousemove").click(function() {
157      if(mousemove) {
158        $("#div2").off("mousemove"),
159        $("#mousemove").html("ON"),
160        mousemove = false,
161      } else {
162        $("#div2").mousemove(function(event) {
163          var x = event.pageX,
164          var y = event.pageY,
165          var type = event.type,
166
167          saida.append(type + ". (" + x + ", " + y + ")"),
168          saida.append("<br>"),
169          saida.scrollTop(saida.prop("scrollHeight")),
170        }),
171        $("#mousemove").html("OFF"),
172        mousemove = true,
173      }
174    }),
175  }),
```

Código Javascript 5.149: jquery-complementar.js

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-jquery-complementar2.zip

3 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/jQuery/public_html/jquery-complementar.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/jQuery/public_html/jquery-complementar.html.

Resumo do Capítulo

1 Resumidamente, jQuery é uma biblioteca JavaScript.

2 Podemos obter a biblioteca jQuery na versão **compressed** ou **uncompressed**.

3 A utilização de um **CDN (Content Delivery Network)**, melhora o tempo de carregamento e a disponibilidade da biblioteca jQuery.

4 O método **on()** permite associar eventos jQuery aos elementos HTML.

5 O método **off()** permite dissociar eventos jQuery dos elementos HTML.

6 A biblioteca jQuery suporta todos os seletores do CSS. Além disso, ela oferece alguns seletores que não existem no CSS.

7 Podemos definir o tempo de duração dos efeitos e animações do jQuery.

8 Podemos determinar uma função que deve ser executada ao término de um efeito ou animação.

9 A biblioteca jQuery oferece muitos recursos para a manipulação dos elementos HTML.

Prova

1 Qual alternativa está correta?

a) jQuery é uma biblioteca JavaScript.

b) jQuery é uma biblioteca HTML.

c) jQuery é uma biblioteca CSS.

d) jQuery é uma linguagem de programação.

e) Todas as alternativas anteriores estão incorretas.

2 Qual alternativa está correta?

a) Geralmente, a utilização de CDNs aumenta a quantidade dados transmitidos entre os navegadores e os Web Servers.

b) Geralmente, a utilização de CDNs diminui a disponibilidade do conteúdo.

c) Geralmente, a utilização de CDNs aumenta a latência na transmissão de dados para os navegadores.

d) Geralmente, a utilização de CDNs aumenta a segurança dos sites e das aplicação web.

e) Geralmente, a utilização de CDNs diminui o tempo de carregamento das páginas web.

3 Quais alternativas estão corretas?

a) Podemos utilizar o método **on()** para adicionar os tratamentos dos eventos.

b) Podemos utilizar o método **enable()** para determinar os tratamentos dos eventos.

c) Podemos utilizar o método **event()** para adicionar ou eliminar os tratamentos dos eventos.

d) Podemos utilizar o método **off()** para eliminar os tratamentos dos eventos.

e) Podemos utilizar o método **disable()** para eliminar os tratamentos dos eventos.

4 Qual alternativa está correta?

a) jQuery suporta apenas os seletores do CSS.

b) jQuery não suporta todos os seletores do CSS.

c) jQuery não suporta nenhum seletor CSS.

d) jQuery suporta todos os seletores do CSS.

e) Todas as alternativas anteriores estão incorretas.

5 Qual alternativa está correta?

a) fadeOn, fadeOff são efeitos do jQuery.

b) slideLeft e slideRight são efeitos do jQuery.

c) hide e show são efeitos do jQuery.

d) jQuery não possui efeitos.

e) Todas as alternativas anteriores estão incorretas.

6 Quais alternativas estão corretas?

a) O método **empty()** remove o conteúdo dos elementos selecionados.

b) O **after** adiciona conteúdo imediatamente depois dos elementos selecionados.

c) O **before** adiciona conteúdo imediatamente antes dos elementos selecionados.

d) O **addClass** adiciona uma ou mais classes aos elementos selecionados.

e) O **clone** cria uma cópia dos elementos selecionados.

7 Qual alternativa está correta?

a) A utilização do AJAX elimina o código JavaScript.

b) jQuery oferece diversos recursos para a utilização do AJAX.

c) O principal objetivo do AJAX é facilitar a utilização do CSS.

d) AJAX é uma biblioteca JavaScript.

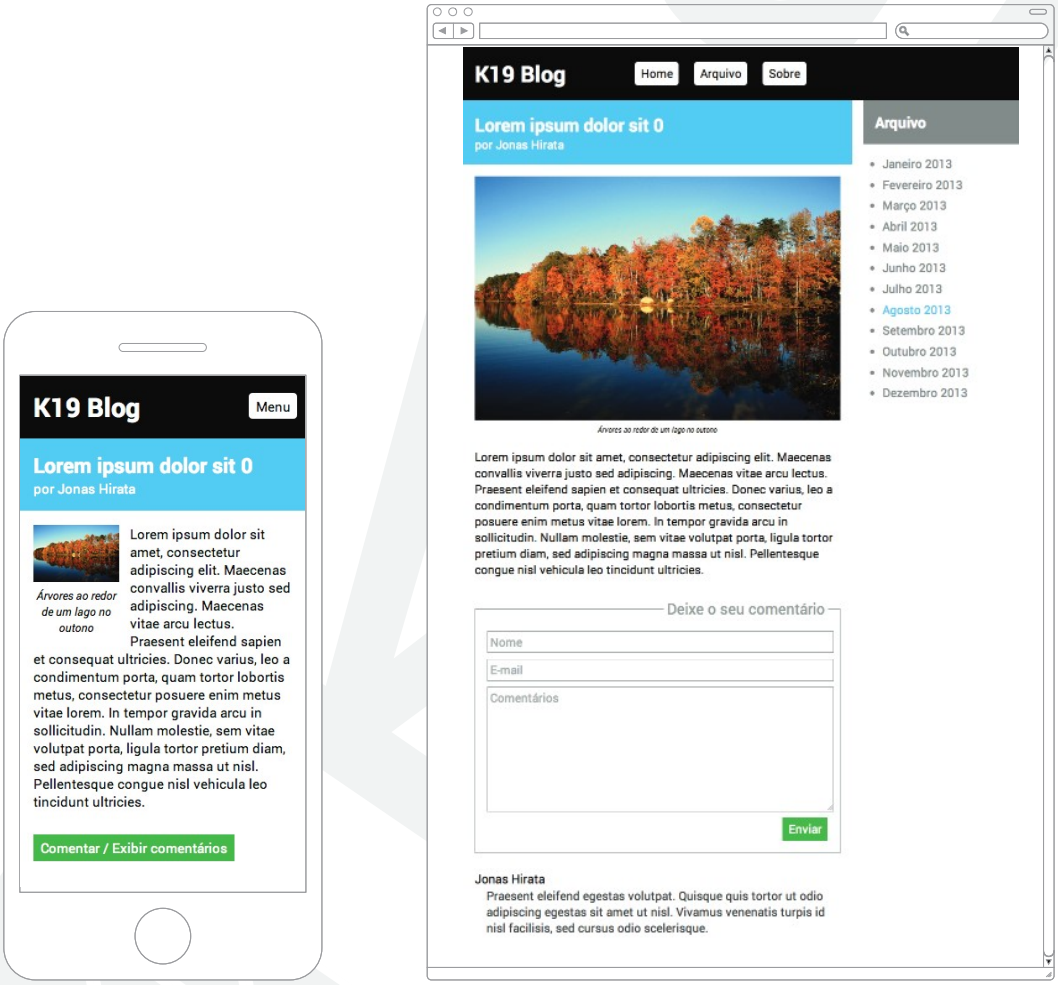
e) O nome AJAX foi inspirado no clube de futebol holandês.

Minha Pontuação [ ] Pontuação Mínima: [5] Pontuação Máxima: [7]

Projeto

Apêndice A

Para exercitar o conteúdo visto nesta apostila, implemente uma página web semelhante à imagem a seguir.



Exercícios de Fixação

1. Abra o **Netbeans** e crie um projeto chamado **blog**.

Importante
No **Windows**, utilizando o IIS (Internet Information Services) como Web Server, você deve salvar o projeto **blog** em **C:\inetpub\wwwroot**. Lembre-se que é necessário instalar o IIS conforme vimos anteriormente.

Importante
No **Ubuntu**, utilizando o Apache HTTP Server como Web Server, você deve salvar o projeto **blog** em **/home/<USUARIO>/public_html**. Lembre-se que é necessário instalar e configurar o Apache HTTP Server como vimos anteriormente.

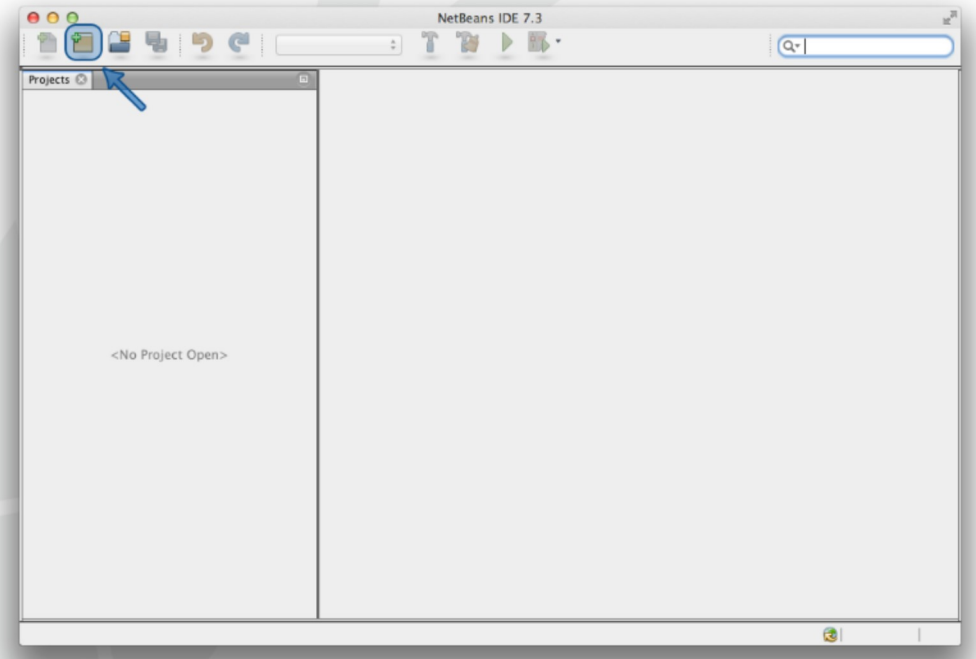


*Figura A.1: Projeto **blog***

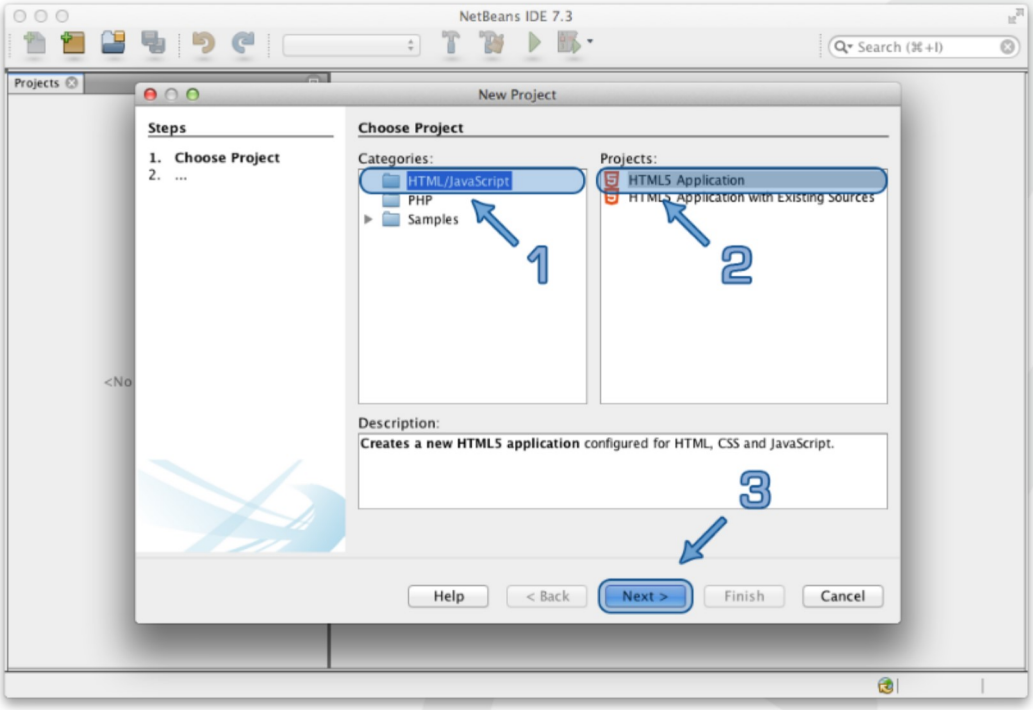


*Figura A.2: Projeto **blog***

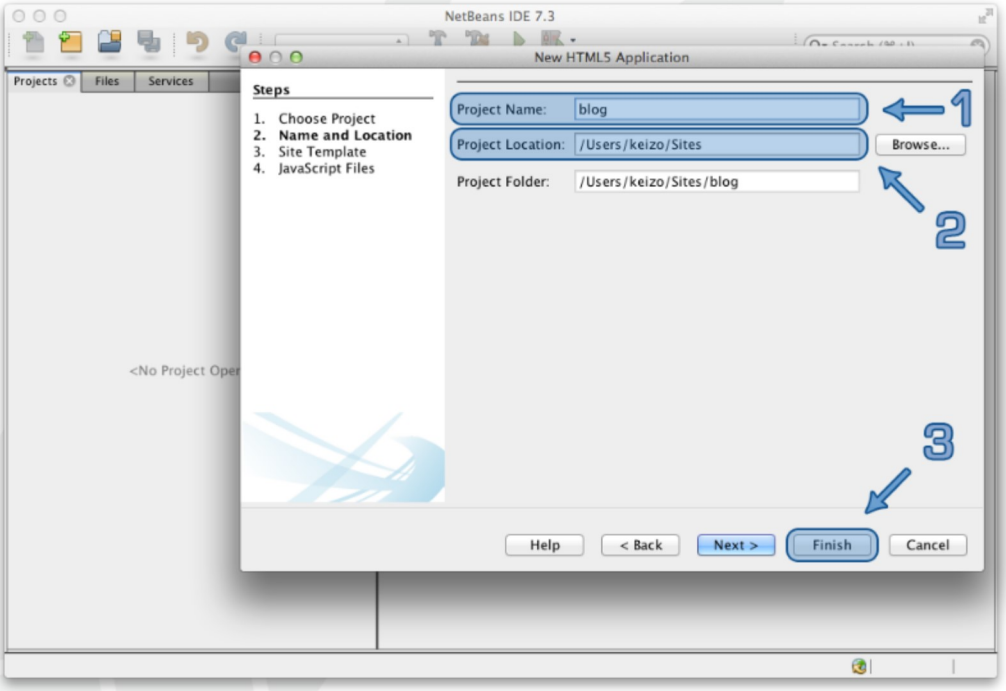


*Figura A.3: Projeto **blog***

Arquivo: `https://github.com/K19/K19-Exercicios/archive/k02-projeto-fixacao1.zip`

2 No projeto **blog**, altere o arquivo **index.html**.

```html
<!DOCTYPE html>
<html>
  <head>
    <title>K19 Blog</title>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <meta name="viewport" content="width=device-width">
    <link rel="stylesheet" type="text/css" href="css/principal.css">
    <script type="text/javascript" src="http.//code.jquery.com/jquery-2.0.3.min.js">
    </script>
    <script type="text/javascript" src="js/principal.js"></script>
  </head>
  <body>
    <div id="main-wrapper">
      <header id="main-header">
        <h1>K19 Blog</h1>

        <nav>
          <input id="main-menu-btn" type="button" value="Menu">

          <ul>
            <li><a href="#">Home</a></li>
            <li><a href="#">Arquivo</a></li>
            <li><a href="#">Sobre</a></li>
          </ul>
        </nav>
      </header>

      <section id="articles-container">
        <!--
          Quando o DIV abaixo estiver na área visível da página
          um conjunto de posts será carregado via AJAX.
        -->
        <div id="lazy-load-point"></div>
      </section>

      <aside>
        <section id="archive-links">
          <header>
            <h1>Arquivo</h1>
          </header>

          <ul>
            <li><a href="#">Janeiro 2013</a></li>
            <li><a href="#">Fevereiro 2013</a></li>
            <li><a href="#">Março 2013</a></li>
            <li><a href="#">Abril 2013</a></li>
            <li><a href="#">Maio 2013</a></li>
            <li><a href="#">Junho 2013</a></li>
            <li><a href="#">Julho 2013</a></li>
            <li><a href="#">Agosto 2013</a></li>
            <li><a href="#">Setembro 2013</a></li>
            <li><a href="#">Outubro 2013</a></li>
            <li><a href="#">Novembro 2013</a></li>
            <li><a href="#">Dezembro 2013</a></li>
          </ul>
        </section>
      </aside>

      <footer>
        <small>&copy, K19 Treinamentos 2013.</small>
      </footer>
    </div>
  </body>
</html>
```

Código HTML A.1: index.html

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-projeto-fixacao2.zip

3 No projeto **blog**, crie um arquivo chamado **principal.css** em uma pasta chamada **css**.

```
/* Declarando as fontes que serão utilizadas na página. */
@font-face {
  font-family. 'Roboto',
  src. url('Roboto-Regular-webfont.eot'),
  src. url('Roboto-Regular-webfont.eot?#iefix') format('embedded-opentype'),
  url('Roboto-Regular-webfont.woff') format('woff'),
  url('Roboto-Regular-webfont.ttf') format('truetype'),
  url('Roboto-Regular-webfont.svg#RobotoRegular') format('svg'),
  font-weight. normal,
  font-style. normal,
}

@font-face {
  font-family. 'Roboto',
  src. url('Roboto-Italic-webfont.eot'),
  src. url('Roboto-Italic-webfont.eot?#iefix') format('embedded-opentype'),
  url('Roboto-Italic-webfont.woff') format('woff'),
  url('Roboto-Italic-webfont.ttf') format('truetype'),
  url('Roboto-Italic-webfont.svg#RobotoItalic') format('svg'),
  font-weight. normal,
  font-style. italic,
}

@font-face {
  font-family. 'Roboto',
  src. url('Roboto-Bold-webfont.eot'),
  src. url('Roboto-Bold-webfont.eot?#iefix') format('embedded-opentype'),
  url('Roboto-Bold-webfont.woff') format('woff'),
  url('Roboto-Bold-webfont.ttf') format('truetype'),
  url('Roboto-Bold-webfont.svg#RobotoBold') format('svg'),
  font-weight. bold,
  font-style. normal,
}

@font-face {
  font-family. 'Roboto',
  src. url('Roboto-BoldItalic-webfont.eot'),
  src. url('Roboto-BoldItalic-webfont.eot?#iefix') format('embedded-opentype'),
  url('Roboto-BoldItalic-webfont.woff') format('woff'),
  url('Roboto-BoldItalic-webfont.ttf') format('truetype'),
  url('Roboto-BoldItalic-webfont.svg#RobotoBoldItalic') format('svg'),
  font-weight. bold,
  font-style. italic,
}

@font-face {
  font-family. 'Roboto',
  src. url('Roboto-Thin-webfont.eot'),
  src. url('Roboto-Thin-webfont.eot?#iefix') format('embedded-opentype'),
  url('Roboto-Thin-webfont.woff') format('woff'),
  url('Roboto-Thin-webfont.ttf') format('truetype'),
  url('Roboto-Thin-webfont.svg#RobotoThin') format('svg'),
  font-weight. 200,
  font-style. normal,
}

@font-face {
  font-family. 'Roboto',
  src. url('Roboto-ThinItalic-webfont.eot'),
  src. url('Roboto-ThinItalic-webfont.eot?#iefix') format('embedded-opentype'),
  url('Roboto-ThinItalic-webfont.woff') format('woff'),
  url('Roboto-ThinItalic-webfont.ttf') format('truetype'),
  url('Roboto-ThinItalic-webfont.svg#RobotoThinItalic') format('svg'),
  font-weight. 200,
  font-style. italic,
}

@font-face {
```

```
69    font-family. 'Roboto',
70    src. url('Roboto-Light-webfont.eot'),
71    src. url('Roboto-Light-webfont.eot?#iefix') format('embedded-opentype'),
72    url('Roboto-Light-webfont.woff') format('woff'),
73    url('Roboto-Light-webfont.ttf') format('truetype'),
74    url('Roboto-Light-webfont.svg#RobotoLight') format('svg'),
75    font-weight. 100,
76    font-style. normal,
77  }
78
79  @font-face {
80    font-family. 'Roboto',
81    src. url('Roboto-LightItalic-webfont.eot'),
82    src. url('Roboto-LightItalic-webfont.eot?#iefix') format('embedded-opentype'),
83    url('Roboto-LightItalic-webfont.woff') format('woff'),
84    url('Roboto-LightItalic-webfont.ttf') format('truetype'),
85    url('Roboto-LightItalic-webfont.svg#RobotoLightItalic') format('svg'),
86    font-weight. 100,
87    font-style. italic,
88  }
89
90  @font-face {
91    font-family. 'Roboto',
92    src. url('Roboto-Medium-webfont.eot'),
93    src. url('Roboto-Medium-webfont.eot?#iefix') format('embedded-opentype'),
94    url('Roboto-Medium-webfont.woff') format('woff'),
95    url('Roboto-Medium-webfont.ttf') format('truetype'),
96    url('Roboto-Medium-webfont.svg#RobotoMedium') format('svg'),
97    font-weight. 300,
98    font-style. normal,
99  }
100
101 @font-face {
102   font-family. 'Roboto',
103   src. url('Roboto-MediumItalic-webfont.eot'),
104   src. url('Roboto-MediumItalic-webfont.eot?#iefix') format('embedded-opentype'),
105   url('Roboto-MediumItalic-webfont.woff') format('woff'),
106   url('Roboto-MediumItalic-webfont.ttf') format('truetype'),
107   url('Roboto-MediumItalic-webfont.svg#RobotoMediumItalic') format('svg'),
108   font-weight. 300,
109   font-style. italic,
110 }
111
112 /*
113 Removendo as margens internas e externas de todos os elementos para
114 evitar diferenças entre os navegadores.
115 */
116 * {
117   margin. 0,
118   padding. 0,
119   font-family. 'Roboto', 'helvetica', 'arial', 'sans-serif',
120 }
121
122 @media (max-width. 480px) {
123   body {
124     font-size.1.2rem,
125   }
126
127   .show-comments {
128     display. inline,
129   }
130
131   .article-comments form,
132   .article-comments article {
133     display. none,
134   }
135 }
136
137 @media (min-width. 640px) {
138   #articles-container {
```

```
    width. 70%,
    float. left,
  }

  aside {
    width. 28%,
    float. left,
    margin. 0 0 0 2%,
  }

  .article-comments form,
  .article-comments article {
    display. block,
  }

  #main-header nav {
    display. inline-block,
    vertical-align. middle,
  }

  #main-header nav input.first-child {
    display. none,
  }

  #main-header nav ul,
  #main-header nav li {
    display. inherit,
    vertical-align. inherit,
  }

  #main-header nav ul {
    margin. 0 0 0 4em,
  }

  #main-header nav li {
    background. white,
    border-radius. 0.3em,
    padding. 0.3em 0.5em,
    font-size. 0.8em,
    margin. 0 0.5em,
    color. black,
  }

  #main-header nav li a {
    color. inherit,
    text-decoration. none,
  }

  #main-header nav li.hover {
    background. #38c3f2,
    color. white,
  }

  .article-text figure {
    margin-bottom. 1em,
  }

  .article-text figure img {
    width. 100%,
  }

  .article-text figure figcaption {
    font-size. 0.5em,
    font-style. italic,
    text-align. center,
  }
}

@media (min-width. 480px) {
  body {
```

```
209     font-size.1.5rem,
210   }
211
212   .show-comments {
213     display. none,
214   }
215 }
216
217 @media (max-width. 640px) {
218   #main-header nav {
219     position. absolute,
220     top. 1em,
221     right. 0.5em,
222     text-align. right,
223   }
224
225   #main-header nav input.first-child {
226     background. white,
227     border-radius. 0.3em,
228     border.none,
229     padding. 0.3em 0.5em,
230     font-size. 0.8em,
231     margin. 0 0 0.1em 0,
232     cursor. pointer,
233   }
234
235   #main-header nav input.first-child.hover {
236     color. #38c3f2,
237   }
238
239   #main-header nav ul {
240     display. none,
241     border-radius. 0.3em,
242     background. white,
243     list-style-type. none,
244     padding. 0.6em,
245     box-shadow. 3px 3px 10px rgba(0, 0, 0, 0.8),
246   }
247
248   #main-header nav li {
249     text-align. left,
250     padding. 0.3em,
251     color. black,
252   }
253
254   #main-header nav li a {
255     color. inherit,
256     text-decoration. none,
257   }
258
259   #main-header nav li.hover {
260     color. #38c3f2,
261   }
262
263   .article-text figure {
264     float. left,
265     margin. 0 0.6em 0.6em 0,
266     width. 5em,
267   }
268
269   .article-text figure img {
270     width. 100%,
271   }
272
273   .article-text figure figcaption {
274     font-size. 0.7em,
275     font-style. italic,
276     text-align. center,
277   }
278 }
```

```
#main-wrapper {
  max-width. 980px,
  width. 100%,
  margin. 0 auto,
}

#main-header {
  position. relative,
  background. black,
  padding. 0.8em,
}

#main-header h1 {
  color. white,
  font-size. 1.5em,
  display. inline-block,
  vertical-align. middle,
}

article header {
  background. #38c3f2,
  color. white,
  padding. 0.8em,
}

article header h1 {
  font-size. 1.2em,
}

article header h2 {
  font-size. 0.8em,
  font-weight. normal,
}

.article-text {
  padding. 0.8em,
}

.article-text p {
  font-size. 0.8em,
  margin-bottom. 0.8em,
}

#archive-links header {
  background. #888,
  color. white,
  padding. 0.8em,
}

#archive-links header h1 {
  font-size. 1em,
}

#archive-links ul {
  padding. 0.8em 0 0.8em 1.6em,
  color. #888,
  font-size. 0.8em,
}

#archive-links li {
  padding. 0.2em 0,
}

#archive-links li a {
  color. inherit,
  text-decoration. none,
}

#archive-links li a.hover {
```

```
349   color. #38c3f2,
350 }
351
352 .article-comments {
353   padding. 0 0.8em,
354 }
355
356 .article-comments > input.first-child {
357   border. none,
358   background. #00c425,
359   color. white,
360   font-size. 0.8em,
361   padding. 0.3em 0.5em,
362   cursor. pointer,
363   margin. 0 0 2em 0,
364 }
365
366 .article-comments fieldset {
367   border. 1px solid #999,
368   padding. 0.8em,
369   text-align. right,
370   margin. 0 0 1em 0,
371 }
372
373 .article-comments fieldset legend {
374   padding. 0 0.2em,
375   color. #999,
376 }
377
378 .article-comments fieldset input.not([type='submit']),
379 .article-comments fieldset textarea{
380   display. block,
381   font-size. 0.8em,
382   margin. 0 0 0.5em 0,
383   width. 100%,
384   border. 1px solid #999,
385   padding. 0.2em,
386 }
387
388 .article-comments fieldset textarea {
389   height. 10em,
390 }
391
392 .article-comments fieldset input[type='submit'] {
393   border. none,
394   background. #00c425,
395   color. white,
396   font-size. 0.8em,
397   padding. 0.3em 0.5em,
398   cursor. pointer,
399 }
400
401 .article-comments article {
402   margin. 0 0 1em 0,
403 }
404
405 .article-comments article p {
406   font-size. 0.8em,
407   color. #444,
408   margin. 0 0 0 1em,
409 }
410
411 #lazy-load-point {
412   height. 24px,
413 }
414
415 #lazy-load-point.loading {
416   background. url('ajax-loader') no-repeat center,
417 }
418
```

```
footer {
  text-align. center,
  clear. left,
}

footer small {
  font-size. 0.5em,
}
```

Código CSS A.1: principal.css

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-projeto-fixacao3.zip

4 No projeto **blog**, crie um arquivo chamado **servico.php**.

```
<?php for ($i = 0, $i < 3, $i++). ?>
  <article>
    <header>
      <h1>Lorem ipsum dolor sit <?php echo $i ?></h1>
      <h2>por Jonas Hirata</h2>
    </header>

    <div class="article-text">
      <figure>
        <img
          src="img/paisagem.jpg"
          title="Paisagem"
          alt="Árvores ao redor de um lago no outono">
        <figcaption>Árvores ao redor de um lago no outono</figcaption>
      </figure>
      <p>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
        Maecenas convallis viverra justo sed adipiscing.
        Maecenas vitae arcu lectus. Praesent eleifend sapien et
        consequat ultricies. Donec varius, leo a condimentum
        porta, quam tortor lobortis metus, consectetur posuere
        enim metus vitae lorem. In tempor gravida arcu in
        sollicitudin. Nullam molestie, sem vitae volutpat
        porta, ligula tortor pretium diam, sed adipiscing magna
        massa ut nisl. Pellentesque congue nisl vehicula leo
        tincidunt ultricies.
      </p>
    </div>

    <div class="article-comments">
      <input
        class="show-comments"
        type="button"
        value="Comentar/Exibir comentários">

      <form>
        <fieldset>
          <legend>Deixe o seu comentário</legend>
          <input type="text" placeholder="Nome">
          <input type="email" placeholder="E-mail">
          <textarea placeholder="Comentários"></textarea>
          <input type="submit" value="Enviar">
        </fieldset>
      </form>

      <article>
        <small>Jonas Hirata</small>
        <p>
          Praesent eleifend egestas volutpat. Quisque quis
          tortor ut odio adipiscing egestas sit amet ut
          nisl. Vivamus venenatis turpis id nisl
          facilisis, sed cursus odio scelerisque.
```

```
53          </p>
54        </article>
55      </div>
56    </article>
57  <?php endfor, ?>
```

Código HTML A.2: servico.php

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-jquery-fixacao4.zip

5 No projeto **blog**, crie um arquivo chamado **principal.js** em uma pasta chamada **js**.

```
1  var scrollTimeout = 0,
2  var isLoadingArticles = false,
3
4  $(document).ready(function() {
5    $('#articles-container').on('click', '.show-comments', function(e) {
6      $(this).parent().find('form, article').slideDown(),
7      $(this).hide(),
8    }),
9
10   // Trata o evento de clique no botão de menu adicionando uma classe
11   // ao mesmo para indicar que o mesmo está expandido.
12   $('#main-menu-btn').click(function(e) {
13     e.stopPropagation(),
14     $(this).parent().find('ul').addClass('expanded').slideDown(),
15   }),
16
17   // Trata o evento de clique em qualquer área da página para contrair e
18   // remover a classe expanded do menu principal.
19   $('html').click(function(e) {
20     $('#main-header ul.expanded').removeClass('expanded').slideUp(400, function() {
21         $(this).removeAttr('style'),
22     }),
23   }),
24
25   $(window).scroll(function() {
26     // Utilizando um timeout para evitar chamadas excessivas
27     // à função afterScroll().
28     clearTimeout(scrollTimeout),
29     scrollTimeout = setTimeout(afterScroll, 500),
30   }),
31
32   // Carregando os artigos assim que a página for carregada.
33   loadArticles(),
34 }),
35
36 function afterScroll() {
37   if (!isLoadingArticles && $('#lazy-load-point').isOnScreen()) {
38     loadArticles(),
39   }
40 }
41
42 function loadArticles() {
43   $.ajax({
44     url. 'servico.php',
45     dataType. 'html',
46     beforeSend. function(jqXHR, settings) {
47       isLoadingArticles = true,
48
49       // Exibe o GIF animado que indica o carregamento do conteúdo.
50       $('#lazy-load-point').addClass('loading'),
51     },
52     complete. function() {
53       // O uso do timeout neste ponto é desnecessário. Foi utilizado aqui
54       // para dar tempo do GIF animado ser visualizado em um servidor
55       // local.
```

```
56        setTimeout(function() {
57          isLoadingArticles = false,
58          // Oculta o GIF animado que indica o carregamento do conteúdo.
59          $('#lazy-load-point').removeClass('loading'),
60        }, 1000),
61      },
62      success. function(data, textStatus, jqXHR) {
63        // O uso do timeout neste ponto é desnecessário. Foi utilizado aqui
64        // para dar tempo do GIF animado ser visualizado em um servidor
65        // local.
66        setTimeout(function() {
67          $('#articles-container').prepend(data),
68        }, 1000),
69      },
70      error. function(jqXHR, textStatus, errorThrown) {
71          alert(errorThrown),
72      }
73    }),
74  }
75
76  // Extendendo o jQuery com o método isOnScreen que verifica se um
77  // elemento está na região visível da página.
78  $.fn.isOnScreen = function() {
79    var win = $(window),
80
81    var viewport = {
82      top. win.scrollTop(),
83      left. win.scrollLeft()
84    },
85    viewport.right = viewport.left + win.width(),
86    viewport.bottom = viewport.top + win.height(),
87
88    var bounds = this.offset(),
89    bounds.right = bounds.left + this.outerWidth(),
90    bounds.bottom = bounds.top + this.outerHeight(),
91
92    return (!(viewport.right < bounds.left ||
93          viewport.left > bounds.right ||
94          viewport.bottom < bounds.top ||
95          viewport.top > bounds.bottom)),
96  },
```

Código Javascript A.1: principal.js

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-projeto-fixacao5.zip

6 Copie o arquivo **k02-projeto.zip** da pasta **K19-Arquivos** para a sua Área de trabalho. Descompacte esse arquivo e copie o conteúdo das pastas **css** e **img** para as pastas **css** e **img** do projeto **blog** respectivamente.

O arquivo k02-projeto.zip também está disponível em http.//k19.com.br/arquivos.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-projeto-fixacao6.zip

7 No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/blog/public_html/index.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/blog/public_html/index.html.

QUIZZES

APÊNDICE B

Quiz 1

Considere uma página HTML contendo um com largura (width) 200px, margem interna (padding) 10px e borda de 3px. Visualmente, qual é o espaço horizontal ocupado por esse elemento?

a) 200px

b) 203px

c) 210px

d) 213px

e) 226px

De acordo com o Box Model do CSS visto no Capítulo 3, ao atribuirmos margens internas e bordas em um elemento com largura definida fazemos com que, visualmente, a largura ocupada por esse elemento seja a soma das propriedades **width**, **padding-left**, **padding-right**, **border-left** e **border-right**. Como utilizamos a propriedade **padding** com o valor 10px para definir as margens internas, podemos considerar que temos 10px nas propriedades **padding-left** e **padding-right**. A mesma idéia se aplica à propriedade **border**, portanto temos **border-left** e **border-right** com 3px cada.

Fazendo a soma temos: 200px (width) + 10px (padding-left) + 10px (padding-right) + 3px (border-left) + 3px (border-right) = **226px**

Portanto o espaço horizontal visualmente ocupado pelo elemento é de **226px**.

Respostas

Apêndice C

Questão 1.1

d

Questão 1.2

e

Questão 1.3

a

Questão 1.4

c

Questão 1.5

d

Questão 1.6

d

Exercício Complementar 2.1

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>K19 Treinamentos</title>
  </head>

  <body>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse
      bibendum pellentesque hendrerit. Aliquam pretium, quam in porttitor
```

```
      vestibulum, massa ligula sodales metus, nec hendrerit nunc purus eu
      mauris.
    </p>
  </body>
</html>
```

Código HTML 2.159: pagina-simples.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/pagina-simples.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/pagina-simples.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar1.zip

Exercício Complementar 2.2

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Curiosidades do Mundo</title>
  </head>

  <body>
    <h1>Curiosidades do Mundo</h1>

    <h2>Europa</h2>
    <p>A Europa é o segundo menor continente em superfície do mundo, cobrindo
    cerca de 10 180 000 quilômetros quadrados ou 2% da superfície da Terra
    e cerca de 6,8% da área acima do nível do mar.</p>

    <h3>Alemanha</h3>
    <p>Com 81,8 milhões de habitantes em janeiro de 2010, o país tem a maior
    população entre os Estados membros da União Europeia e é também o lar da
    terceira maior população de migrantes internacionais em todo o mundo.</p>

    <h4>Hesse</h4>
    <p>A capital é Wiesbaden e a maior cidade Francoforte do Meno (Frankfurt
    am Main), onde está localizado um dos maiores aeroportos do mundo e um
    centro financeiro de grande importância.</p>

    <h5>Frankfurt</h5>
    <p>Frankfurt am Main ou Francoforte do Meno, mais conhecida simplesmente como
    Frankfurt, é a maior cidade do estado alemão de Hesse e a quinta maior cidade
    da Alemanha, com uma população 700.000 habitantes em 2012.</p>

    <h2>Ásia</h2>
    <p>A Ásia é o maior dos continentes, tanto em área como em população.</p>

    <h3>Japão</h3>
    <p>O país é um arquipélago de 6 852 ilhas, cujas quatro maiores são Honshu,
    Hokkaido, Kyushu e Shikoku, representando em conjunto 97% da área
    terrestre nacional.</p>

    <h4>Okinawa</h4>
    <p>Antigamente, Okinawa fazia parte de um reino independente, o reino Ryukyu,
    o que foi decisivo para o desenvolvimento de uma cultura própria do desenrolar
    de uma história particular e significativamente diferenciada do resto do
```

```
    Japão.</p>

    <h5>Nago</h5>
    <p>De 21 de julho até 23 de julho de 2000, foi sede do encontro anual do G8.</p>

    <p>Fonte. wikipedia.org</p>
  </body>
</html>
```

Código HTML 2.160: geografia.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/geografia.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/geografia.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar2.zip

Exercício Complementar 2.3

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Seguro Treinamento - K19</title>
  </head>

  <body>
    <h1>Na K19 o aluno faz o curso quantas vezes quiser!</h1>
    <p>
      Comprometida com o aprendizado e com a satisfação dos seus alunos, a K19
      criou o Seguro Treinamento. <br> Ao contratar um curso, o aluno poderá
      refazê-lo quantas vezes desejar mediante a disponibilidade de vagas e
      pagamento da franquia do Seguro Treinamento.
    </p>
  </body>
</html>
```

Código HTML 2.161: seguro-treinamento.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/seguro-treinamento.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/seguro-treinamento.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar3.zip

Exercício Complementar 2.4

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Caracteres especiais</title>
  </head>

  <body>
    <h1>Caracteres especiais</h1>
    <p>
      &cross, &sext, &ofcir, &check, &sharp,
    </p>
  </body>
</html>
```

Código HTML 2.162: caracteres-especiais.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/caracteres-especiais.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/caracteres-especiais.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar4.zip

Exercício Complementar 2.5

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Texto pré-formatado</title>
  </head>

  <body>
    <h1>Texto pré-formatado</h1>
    <pre>
A B  C   D    E     F      G

1 2  3   4    5     6      7

a b  c   d    e     f      g
    </pre>
  </body>
</html>
```

Código HTML 2.163: espacos-e-quebras-de-linha.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/espacos-e-quebras-de-linha.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/espacos-e-quebras-de-linha.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar5.zip

Exercício Complementar 2.6

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Código Java</title>
  </head>
  <body>
    <h1>Código Java</h1>
    <code>
      double numero = Math.random(),
    </code>
  </body>
</html>
```

Código HTML 2.164: codigo-java.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/codigo-java.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/codigo-java.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar6.zip

Exercício Complementar 2.7

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Elemento i e elemento b</title>
  </head>
  <body>
    <h1>Elemento i e elemento b</h1>
    <ul>
      <li>Porquinho-da-índia ou <i>Cavia porcellus</i></li>
      <li><i>Backup</i>(cópia de segurança)</li>
      <li><i>shoot the breeze</i>(bater papo ou jogar conversa fora)</li>
      <li><i>Moskvá</i>(transliteração da palavra Moscou em russo)</li>
      <li><i>Se não sabes, aprende, se já sabes, ensina.</i> (Confúcio)</li>
    </ul>

    <p>
      Atualmente, praticamente todos os <b>sistemas corporativos</b>
      possuem <b>interfaces web</b>. Para quem deseja atuar no mercado
      de <b>desenvolvimento de software</b>, é obrigatório o conhecimento
      das linguagens. <b>HTML</b>, <b>CSS</b> e <b>JavaScript</b>.
    </p>
  </body>
</html>
```

Código HTML 2.165: elementos-i-b.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/elementos-i-b.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/elementos-i-b.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar7.zip

Exercício Complementar 2.8

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>São Paulo</title>
  </head>
  <body>
    <h1>São Paulo</h1>
    <p>
      A cidade de São Paulo possui uma área de 1.523 km<sup>2</sup>.
      Em 2011, São Paulo emitiu 16,430 milhões de toneladas de
      CO<sub>2</sub>.
    </p>
  </body>
</html>
```

Código HTML 2.166: sao-paulo.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/sao-paulo.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/sao-paulo.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar8.zip

Exercício Complementar 2.9

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>São Paulo FC</title>
  </head>
  <body>
    <h1>São Paulo FC</h1>
    <p>
      O <strong>São Paulo FC</strong> é o único
      time brasileiro que ganhou <em>três vezes</em>
      o <strong>mundial de clubes</strong>.
    </p>
```

```
  </body>
</html>
```

Código HTML 2.167: spfc.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/spfc.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/spfc.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar9.zip

Exercício Complementar 2.10

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Citações</title>
  </head>
  <body>
    <p>O que a Wikipédia fala sobre Java?</p>
    <blockquote cite="http://en.wikipedia.org/wiki/Java_(programming_language)">
      <p>
        Java is a general-purpose, concurrent, class-based, object-oriented
        computer programming language that is specifically designed to have
        as few implementation dependencies as possible. ...
      </p>
      <p>
        The original and reference implementation Java compilers, virtual
        machines, and class libraries were developed by Sun from 1991 and
        first released in 1995. As of May 2007, in compliance with the
        specifications of the Java Community Process, Sun relicensed most
        of its Java technologies under the GNU General Public License....
      </p>
    </blockquote>

    <p>
      Galvão Bueno disse.
      <q cite="http://www.naosalvo.com.br/as-melhorespiores-frases-de-galvao-bueno-em↩
          -um-so-lugar">
        A seleção brasileira prioriza o coletivo e a individualidade
      </q>.
    </p>

    <p>
      <cite> Dom Quixote</cite> de Miguel de Cervantes é um dos livros mais
      vendidos da história.
    </p>
  </body>
</html>
```

Código HTML 2.168: mais-citacoes.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/mais-citacoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/mais-citacoes.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar10.zip

Exercício Complementar 2.11

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Abreviações</title>
  </head>
  <body>
    <h1>Abreviações</h1>
    <ul>
      <li><abbr title="Federal Bureau Investigation">FBI</abbr></li>
      <li><abbr title="Central Intelligence Agency">CIA</abbr></li>
      <li><abbr title="Crime Scene Investigation">CSI</abbr></li>
    </ul>
  </body>
</html>
```

Código HTML 2.169: abbr.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/abbr.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/abbr.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar11.zip

Exercício Complementar 2.12

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Definições</title>
  </head>
  <body>
    <h1>Definições</h1>
    <dl>
      <dt><dfn>Folha seca</dfn></dt>
      <dd>
        Chute que faz a bola percorrer uma trajetória em curva acentuada com
        uma queda brusca no final do percurso.
      </dd>
      <dt><dfn>Lanterna</dfn></dt>
      <dd>
        Equipe que ocupa a última posição de um campeonato.
      </dd>
```

```
      <dt><dfn>Primeiro pau</dfn></dt>
      <dd>
        Trave mais próxima da origem de um cruzamento.
      </dd>
    </dl>
  </body>
</html>
```

Código HTML 2.170: mais-definicoes.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/mais-definicoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/mais-definicoes.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar12.zip

Exercício Complementar 2.13

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exemplo de alterações</title>
  </head>
  <body>
    <h1>Alterações</h1>
    <p>
      <s>Atualmente, eu moro na Inglaterra.</s>
      <ins>Atualmente, eu moro no Brasil.</ins>
      <del>Eu quero conhecer a Síria.</del>
    </p>
  </body>
</html>
```

Código HTML 2.171: mais-alteracoes.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/mais-alteracoes.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/mais-alteracoes.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar13.zip

Exercício Complementar 2.14

```
<!DOCTYPE html>
<html lang="pt-br">
```

```
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>Datas e horas</title>
6   </head>
7   <body>
8     <h1>Datas e horas</h1>
9     <ul>
10      <li>
11        O Brasil foi pentacampeão em <time datetime="2002-06-30">30 de Junho
12        de 2002</time>.
13      </li>
14      <li>
15        Ele não pode esquecer o <time datetime="03-08">Dia das mulheres</time>.
16      </li>
17    </ul>
18  </body>
19 </html>
```

Código HTML 2.172: mais-datas.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/mais-datas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/mais-datas.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar14.zip

Exercício Complementar 2.15

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Texto marcado</title>
6    </head>
7    <body>
8      <h1>Texto marcado</h1>
9      <p>
10       <mark>Ayrton Senna da Silva</mark> foi um piloto brasileiro de
11       <mark>Fórmula 1</mark>, três vezes campeão mundial, nos anos de 1988,
12       1990 e 1991. Foi também vice-campeão no controverso campeonato de 1989
13       e em 1993
14     </p>
15   </body>
16 </html>
```

Código HTML 2.173: texto-marcado.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/texto-marcado.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/texto-marcado.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar15.zip

Exercício Complementar 2.16

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Pontos Turísticos</title>
  </head>
  <body>
    <h1>Pontos Turísticos do Brasil</h1>

    <dl>
      <dt>Ilha Bela - SP</dt>
      <dd>
        Praias, Trilhas e Mergulho em Náufrago.
      </dd>
      <dt>Bonito - MS</dt>
      <dd>
        Mergulho em rios de águas transparentes, cachoeiras, grutas e cavernas.
      </dd>
      <dt>Museu de Arte de São Paulo - SP</dt>
      <dd>
        Grande acervo com diversas obras de artistas consagrados.
      </dd>
    </dl>
  </body>
</html>
```

Código HTML 2.174: pontos-turisticos.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/pontos-turisticos.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/pontos-turisticos.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar16.zip

Exercício Complementar 2.17

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Como instalar o seu XPTO</title>
  </head>
  <body>
    <h1>Como instalar o seu XPTO - K19 Eletronics</h1>

    <ol>
      <li>
        Verifique se todos os acessórios estão presentes.
      </li>
```

```
14        <li>
15          Coloque o aparelho na horizontal sobre uma superfície plana.
16        </li>
17        <li>
18          Conecte o aparelho ao computador utilizando um cabo USB.
19        </li>
20        <li>
21          Use o CD-ROM para instalação do software.
22        </li>
23        <li>
24          Conecte o aparelho à fonte de energia com um adaptador AC.
25        </li>
26        <li>
27          Ligue o aparelho e espere o reconhecimento do computador.
28        </li>
29      </ol>
30    </body>
31  </html>
```

Código HTML 2.175: manual-k19.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/manual-k19.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/manual-k19.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar17.zip

Exercício Complementar 2.18

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5      <title>Links</title>
6    </head>
7    <body>
8      <h1>Links</h1>
9      <ul>
10       <li><a href="http://www.k19.com.br">K19</a></li>
11       <li><a href="http://www.wikipedia.org/">Wikipédia</a></li>
12       <li><a href="http://facebook.com">Facebook</a></li>
13     </ul>
14   </body>
15 </html>
```

Código HTML 2.176: mais-links.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/mais-links.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/mais-links.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar18.zip

Exercício Complementar 2.19

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Imagens</title>
  </head>
  <body>
    <h1>K19 Treinamentos</h1>
    <img src="http.//www.k19.com.br/css/img/main-header-logo.png" />

    <h2>Cursos</h2>
    <ul>
      <li>
        <img src="http.//www.k19.com.br/figs/k21-logo-large.png" />
        K21 - Persistência com JPA2 e Hibernate
      </li>
      <li>
        <img src="http.//www.k19.com.br/figs/k22-logo-large.png" />
        K22 - Desenvolvimento Web Avançado com JSF2, EJB3.1 e CDI
      </li>
      <li>
        <img src="http.//www.k19.com.br/figs/k23-logo-large.png" />
        K23 - Integração de Sistemas com Webservices, JMS e EJB
      </li>
    </ul>
  </body>
</html>
```

Código HTML 2.177: mais-imagens.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/mais-imagens.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/mais-imagens.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar19.zip

Exercício Complementar 2.20

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html,charset=UTF-8">
    <title>Tabelas</title>
  </head>
  <body>
    <table>
      <thead>
        <tr>
          <th>Continente/Subcontinente</th>
```

```
12            <th>Cidade</th>
13            <th>Idioma</th>
14          </tr>
15        </thead>
16        <tfoot>
17          <tr>
18            <td colspan="3">Última atualização. 11/2012</td>
19          </tr>
20        </tfoot>
21        <tbody>
22          <tr>
23            <td rowspan="2">América do Sul</td>
24            <td>São Paulo</td>
25            <td>Português</td>
26          </tr>
27          <tr>
28            <td>Cidade do México</td>
29            <td>Espanhol</td>
30          </tr>
31          <tr>
32            <td rowspan="3">Ásia</td>
33            <td>Tóquio</td>
34            <td>Japonês</td>
35          </tr>
36          <tr>
37            <td>Xangai</td>
38            <td>Mandarim</td>
39          </tr>
40          <tr>
41            <td>Nova Délhi</td>
42            <td>Hindi</td>
43          </tr>
44          <tr>
45            <td>América do Norte</td>
46            <td>Nova Iorque</td>
47            <td>Inglês</td>
48          </tr>
49        </tbody>
50      </table>
51    </body>
52  </html>
```

Código HTML 2.178: mais-tabelas.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/html/public_html/mais-tabelas.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/html/public_html/mais-tabelas.html.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-html-complementar20.zip

Exercício Complementar 2.21

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3      <head>
4          <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5          <title>Currículo</title>
6      </head>
```

```
7     <body>
8         <h1>Cadastro de Currículo</h1>
9         <form action="parametros.php" method="post">
10
11            <fieldset>
12                <legend>Informações Pessoais</legend>
13
14                <label for="nome_id">Nome. </label>
15                <input
16                  id="nome_id"
17                  type="text"
18                  name="nome"
19                  placeholder="Digite o seu nome"
20                  required>
21                <br>
22
23                <label for="email_id">Email. </label>
24                <input
25                  id="email_id"
26                  type="email"
27                  name="email"
28                  placeholder="Digite o seu email"
29                  required>
30                <br>
31
32                <label for="nascimento_id">Data de Nascimento. </label>
33                <input
34                  id="nascimento_id"
35                  type="date"
36                  name="nascimento"
37                  required>
38                <br>
39
40                <label for="altura_id">Altura(m). </label>
41                <input
42                  id="altura_id"
43                  type="number"
44                  name="altura"
45                  step="0.01"
46                  min="0"
47                  max="2">
48                <br>
49
50                <label for="site_id">Site. </label>
51                <input
52                  id="site_id"
53                  type="url"
54                  name="site"
55                  placeholder="Facebook, Linkedin, Twitter">
56                <br>
57
58                <label for="estado_civil_id">Estado Civil. </label>
59                <select id="estado_civil_id" name="estado-civil">
60                    <option value="-1">Selecione</option>
61                    <option value="S">Solteiro</option>
62                    <option value="C">Casado</option>
63                    <option value="D">Divorciado</option>
64                    <option value="V">Viúvo</option>
65                </select>
66                <br>
67
68                <label>Sexo. </label>
69                <input
70                  id="masculino_id"
71                  type="radio"
72                  name="sexo"
73                  value="masculino">
74                <label for="masculino_id">Masculino</label>
75                <input
76                  id="feminino_id"
```

```
77                  type="radio"
78                  name="sexo"
79                  value="feminino">
80              <label for="feminino_id">Feminino</label>
81              <br>
82          </fieldset>
83
84          <fieldset>
85              <legend>Endereço</legend>
86
87              <label for="cep_id">CEP. </label>
88              <input
89                  id="cep_id"
90                  type="text"
91                  name="cep"
92                  required>
93              <br>
94
95              <label for="endereco_id">Endereço. </label>
96              <input
97                  id="endereco_id"
98                  type="text"
99                  name="endereco"
100                 required>
101             <br>
102         </fieldset>
103
104         <fieldset>
105             <legend>Contato</legend>
106
107             <label for="telefone_id">Telefone. </label>
108             <input
109                 id="telefone_id"
110                 type="tel"
111                 name="telefone">
112             <br>
113
114             <label for="celular_id">Celular. </label>
115             <input
116                 id="celular_id"
117                 type="tel"
118                 name="celular">
119             <br>
120         </fieldset>
121
122         <fieldset>
123             <legend>Conhecimentos</legend>
124
125             <input
126                 id="html_id"
127                 type="checkbox"
128                 name="conhecimentos"
129                 value="HTML">
130             <label for="html_id">HTML</label>
131
132             <input
133                 id="css_id"
134                 type="checkbox"
135                 name="conhecimentos"
136                 value="CSS">
137             <label for="css_id">CSS</label>
138
139             <input
140                 id="js_id"
141                 type="checkbox"
142                 name="conhecimentos"
143                 value="JS">
144             <label for="js_id">JavaScript</label>
145             <br>
146
```

```
                <label for="mais_conhecimentos_id">Mais conhecimentos</label>
                <textarea
                  id="mais_conhecimentos_id"
                  rows="10"
                  cols="20"
                  maxlength="500">Digite os seus conhecimentos</textarea>
            </fieldset>

            <input type="submit" value="Enviar">
        </form>
    </body>
</html>
```

Código HTML 2.179: mais-formularios.html

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/html/public_html/mais-formularios.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/html/public_html/mais-formularios.html.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-html-complementar21.zip

Questão 2.1

d

Questão 2.2

c

Questão 2.3

e

Questão 2.4

a

Questão 2.5

c

Questão 2.6

b

Questão 2.7

d

Questão 2.8

e

Questão 2.9

e

Questão 2.10

a

Questão 2.11

d

Questão 2.12

c

Questão 2.13

e

Questão 2.14

b

Questão 2.15

a

Questão 3.1

e

Questão 3.2

b

Questão 3.3

d

Questão 3.4

c

Questão 3.5

d

Questão 3.6

b

Questão 3.7

a

Questão 3.8

d

Questão 3.9

a

Questão 3.10

d

Questão 3.11

e

Questão 3.12

d

Questão 3.13

d

Questão 3.14

e

Questão 3.15

c

Questão 3.16

e

Questão 3.17

d

Exercício Complementar 4.1

No projeto **javascript**, adicione um arquivo chamado **exibe-numeros-1-50-2x.html** e outro chamado **exibe-numeros-1-50-2x.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exibe os números de 1 até 50 duas vezes</title>
    <script type="text/javascript" src="exibe-numeros-1-50-2x.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.56: exibe-numeros-1-50-2x.html

```
for (var x = 0, x < 2, x++) {
  for (var y = 1, y <= 50, y++) {
    console.log(y),
  }
}
```

Código Javascript 4.136: exibe-numeros-1-50-2x.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/javascript/public_html/exibe-numeros-1-50-2x.html`.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/exibe-numeros-1-50-2x.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-complementar1.zip

Exercício Complementar 4.2

No projeto **javascript**, adicione um arquivo chamado **exibe-nome-k19.html** e outro chamado **exibe-nome-k19.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exibe nome e K19</title>
    <script type="text/javascript" src="exibe-nome-k19.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.57: exibe-nome-k19.html

```
for (var x = 0, x < 5, x++) {
  console.log("Rafael Cosentino"),

  for (var y = 0, y < 3, y++) {
    console.log("K19"),
  }
}
```

Código Javascript 4.137: exibe-nome-k19.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/exibe-nome-k19.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/exibe-nome-k19.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-complementar2.zip

Exercício Complementar 4.3

No projeto **javascript**, adicione um arquivo chamado **multiplos-de-tres.html** e outro chamado **multiplos-de-tres.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Múltiplos de três</title>
    <script type="text/javascript" src="multiplos-de-tres.js"></script>
```

```
7   </head>
8   <body>
9   </body>
10 </html>
```

Código HTML 4.58: multiplos-de-tres.html

```
1 for (var x = 1, x <= 60, x++) {
2   if (x % 3 != 0) {
3     console.log("*"),
4   } else {
5     console.log("***")
6   }
7 }
```

Código Javascript 4.138: multiplos-de-tres.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/javascript/public_html/multiplos-de-tres.html`.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/~<USUARIO>/javascript/public_html/multiplos-de-tres.html`.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: `https.//github.com/K19/K19-Exercicios/archive/k02-javascript-complementar3.zip`

Exercício Complementar 4.4

No projeto **javascript**, adicione um arquivo chamado **multiplos-de-quatro-e-sete.html** e outro chamado **multiplos-de-quatro-e-sete.js**.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
5     <title>Múltiplos de quatro e sete</title>
6     <script type="text/javascript" src="multiplos-de-quatro-e-sete.js"></script>
7   </head>
8   <body>
9   </body>
10 </html>
```

Código HTML 4.59: multiplos-de-quatro-e-sete.html

```
1 for (var x = 1, x <= 80, x++) {
2   var resto4 = x%4,
3   var resto7 = x%7,
4
5   if (resto4 == 0 || resto7 == 0) {
6     console.log("*"),
7   } else {
8     console.log(x),
9   }
10 }
```

Código Javascript 4.139: multiplos-de-quatro-e-sete.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/multiplos-de-quatro-e-sete.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/multiplos-de-quatro-e-sete.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-complementar4.zip

Exercício Complementar 4.5

No projeto **javascript**, adicione um arquivo chamado **exibe-triangulo.html** e outro chamado **exibe-triangulo.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exibe triângulo</title>
    <script type="text/javascript" src="exibe-triangulo.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.60: exibe-triangulo.html

```
var linha = "*",
for(var contador = 1, contador <= 10, contador++) {
  console.log(linha),
  linha += "*",
}
```

Código Javascript 4.140: exibe-triangulo.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/exibe-triangulo.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/exibe-triangulo.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-complementar5.zip

Exercício Complementar 4.6

No projeto **javascript**, adicione um arquivo chamado **exibe-triangulos.html** e outro chamado **exibe-triangulos.js**.

```
<!DOCTYPE html>
```

```
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Exibe triângulos</title>
    <script type="text/javascript" src="exibe-triangulos.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.61: exibe-triangulos.html

```
var linha = "*",
for(var contador = 1, contador <= 8, contador++) {
  console.log(linha),
  var resto = contador % 4,
  if(resto == 0) {
    linha = "*",
  } else {
    linha += "*",
  }
}
```

Código Javascript 4.141: exibe-triangulos.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/javascript/public_html/exibe-triangulos.html`.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

`http.//localhost/~<USUARIO>/javascript/public_html/exibe-triangulos.html`.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: `https.//github.com/K19/K19-Exercicios/archive/k02-javascript-complementar6.zip`

Exercício Complementar 4.7

No projeto **javascript**, adicione um arquivo chamado **fibonnaci.html** e outro chamado **fibonnaci.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Fibonacci</title>
    <script type="text/javascript" src="fibonnaci.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.62: fibonnaci.html

```
var penultimo = 0,
var ultimo = 1,

console.log(penultimo),
console.log(ultimo),

for(var contador = 0, contador < 28, contador++) {
```

```
  var proximo = penultimo + ultimo,
  console.log(proximo),

  penultimo = ultimo,
  ultimo = proximo,
}
```

Código Javascript 4.142: fibonnaci.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

http.//localhost/javascript/public_html/fibonnaci.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http.//localhost/~<USUARIO>/javascript/public_html/fibonnaci.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-complementar7.zip

Exercício Complementar 4.8

No projeto **javascript**, adicione um arquivo chamado **embaralha.html** e outro chamado **embaralha.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Embaralha</title>
    <script type="text/javascript" src="embaralha.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.63: embaralha.html

```
var array = new Array(10),

for(var i = 0, i < array.length, i++){
  array[i] = i,
}

for(var i = 0, i < array.length, i++){
  console.log(array[i]),
}

for(var i = 0, i < 10, i++){
  var posicao1 = Math.floor(Math.random() * 10),
  var posicao2 = Math.floor(Math.random() * 10),
  var auxiliar = array[posicao1],

  array[posicao1] = array[posicao2],
  array[posicao2] = auxiliar,
}

console.log("--------------------"),

for(var i = 0, i < array.length, i++){
  console.log(array[i]),
```

```
}
```

Código Javascript 4.143: embaralha.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/embaralha.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/embaralha.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https://github.com/K19/K19-Exercicios/archive/k02-javascript-complementar8.zip

Exercício Complementar 4.9

No projeto **javascript**, adicione um arquivo chamado **ordena.html** e outro chamado **ordena.js**.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta http-equiv="Content-Type" content="text/html, charset=UTF-8">
    <title>Ordena</title>
    <script type="text/javascript" src="ordena.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código HTML 4.64: ordena.html

```
var array = new Array(10),

for(var i = 0, i < array.length, i++){
  array[i] = Math.floor(Math.random() * 10),
}

for(var i = 0, i < array.length, i++){
  console.log(array[i]),
}

array.sort(),

console.log("--------------------"),

for(var i = 0, i < array.length, i++){
  console.log(array[i]),
}
```

Código Javascript 4.144: ordena.js

No **Windows**, utilize o **Chrome** para acessar o endereço:

http://localhost/javascript/public_html/ordena.html.

No **Ubuntu**, utilize o **Chrome** para acessar o endereço:

http://localhost/~<USUARIO>/javascript/public_html/ordena.html.

Utilize o console do navegador para observar as mensagens exibidas.

Arquivo: https.//github.com/K19/K19-Exercicios/archive/k02-javascript-complementar9.zip

Questão 4.1

e

Questão 4.2

a

Questão 4.3

e

Questão 4.4

c

Questão 4.5

a

Questão 4.6

c

Questão 4.7

b

Questão 4.8

d

Questão 4.9

e

Questão 4.10

b

Questão 4.11

c

Questão 5.1

a

Questão 5.2

e

Questão 5.3

a, d

Questão 5.4

d

Questão 5.5

c

Questão 5.6

todas as alternativas estão corretas

Questão 5.7

b