



Conhecendo outras tags do HTML



**Certified
Developer**
The Ultimate Tech Degree

DigitalHouse >
Coding School



Temas

1

Listas

2

Textos

3

Rotas

4

Acessibilidade

5

GIT

1 | Listas

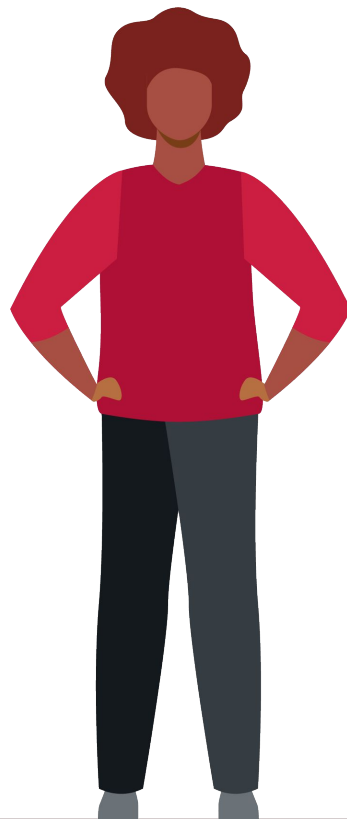




Objetivo

Podemos representar informações de diferentes maneiras em nossa página, uma dessas maneiras são as listagens.

Podemos listar informações utilizando diferentes tipos de listagem que o HTML nos fornece.





Listas Ordenadas

Ela é responsável por justamente criar uma lista com algum tipo de ordenação.

HTML

```
<ol>  
  <li>Banana</li>  
  <li>Laranja</li>  
  <li>Maçã</li>  
</ol>
```

1. Banana
2. Laranja
3. Maçã



Tipos de Ordenação

Podemos **alterar os tipos de ordenação** de uma Lista Ordenada com o atributo '**type**', por padrão ela se inicia com números mas podemos utilizar letras e algarismos romanos.

Vamos então ver alguns exemplos de diferentes tipos de ordenação.





Ordenação letra minúscula

Escrevendo **a** minúsculo temos uma ordenação com identificadores alfabéticos minúsculos:

HTML

```
<ol type="a">  
  <li>Banana</li>  
  <li>Laranja</li>  
  <li>Maçã</li>  
</ol>
```

- a. Banana
- b. Laranja
- c. Maçã



Ordenação letra maiúscula

Escrevendo **A** maiúsculo temos uma ordenação com identificadores alfabéticos maiúsculos:

HTML

```
<ol type="A">  
  <li>Banana</li>  
  <li>Laranja</li>  
  <li>Maçã</li>  
</ol>
```

- A. Banana
- B. Laranja
- C. Maçã

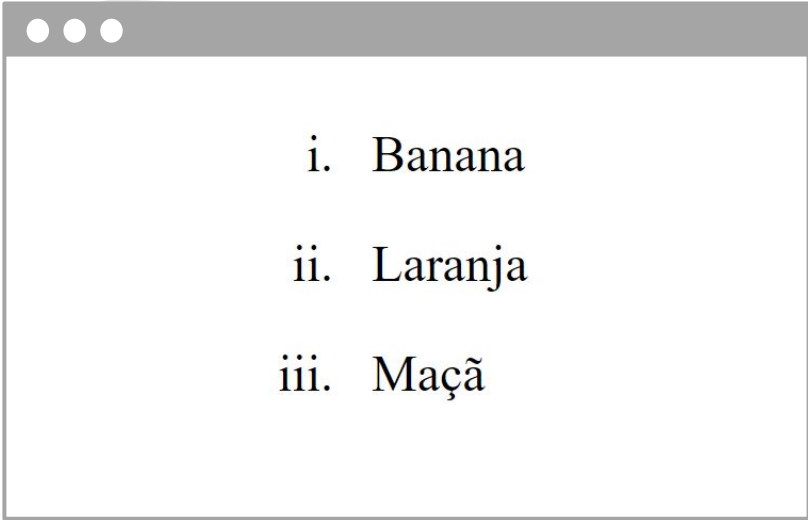


Ordenação romano minúsculo

Escrevendo **i** minúsculo temos uma ordenação com identificadores romanos minúsculos:

HTML

```
<ol type="i">  
  <li>Banana</li>  
  <li>Laranja</li>  
  <li>Maçã</li>  
</ol>
```

- 
- i. Banana
 - ii. Laranja
 - iii. Maçã



Ordenação romano maiúsculo

Escrevendo **I** maiúsculo temos uma ordenação com identificadores romanos maiúsculos:

HTML

```
<ol type="I">  
  <li>Banana</li>  
  <li>Laranja</li>  
  <li>Maçã</li>  
</ol>
```

- I. Banana
- II. Laranja
- III. Maçã



Listas Desordenadas

Ao contrário dos exemplos que vimos anteriormente, as listas desordenadas criam uma lista sem ordenação, onde o identificador foi substituído por um ponto:

HTML

```
<ul>  
  <li>Banana</li>  
  <li>Laranja</li>  
  <li>Maçã</li>  
</ul>
```

- Banana
- Laranja
- Maçã

Hora da revisão



Conclusão

- Vimos que é possível apresentarmos informações em formatos de lista.
- Existem dois tipos de listas, a Ordenada e a Desordenada.
- É possível alterar o tipo de ordenação por alguns padrões que o HTML suporta.

2 | Textos



Títulos

Citações



Objetivo

Parágrafos

**Revisão e
Conclusão**

1 | Títulos



Objetivo

Quando pensamos em representar alguma informação como uma listagem, logo percebemos que precisamos definir um título para dizer ao usuário sobre o que é aquele conteúdo.





Tags de Cabeçalho

As tags de cabeçalho são as responsáveis por possibilitarem a criação de títulos. Elas variam entre h1 a h6, sendo o h1 o mais importante e o h6 o menos relevante.

h1 **h2** **h3** h4 h5 h6



Título Principal - h1

Tem como função ser o título principal daquela seção:

```
HTML <h1>Let's go CTD!</h1>
```

Let's go CTD!



Subtítulo - h2

Serve como subtítulo para o h1:

HTML

```
<h1>Let's go CTD!</h1>  
<h2>Subtitulo</h2>
```

Let's go CTD!

Subtitulo



Subtítulo - h3

Serve como subtítulo para o h2:

HTML

```
<h1>Let's go CTD!</h1>  
<h2>Subtitulo</h2>  
<h3>Subtitulo</h3>
```

Let's go CTD!

Subtitulo

Subtitulo



Subtítulo - h4

Serve como subtítulo para o h3:

HTML

```
<h1>Let's go CTD!</h1>  
<h2>Subtitulo</h2>  
<h3>Subtitulo</h3>  
<h4>Subtitulo</h4>
```

Let's go CTD!

Subtitulo

Subtitulo

Subtitulo



Subtítulo - h5

Serve como subtítulo para o h4:

HTML

```
<h1>Let's go CTD!</h1>  
<h2>Subtitulo</h2>  
<h3>Subtitulo</h3>  
<h4>Subtitulo</h4>  
<h5>Subtitulo</h5>
```

Let's go CTD!

Subtitulo

Subtitulo

Subtitulo

Subtitulo



Subtítulo - h6

Serve como subtítulo para o h5:

HTML

```
<h1>Let's go CTD!</h1>  
<h2>Subtitulo</h2>  
<h3>Subtitulo</h3>  
<h4>Subtitulo</h4>  
<h5>Subtitulo</h5>  
<h6>Subtitulo</h6>
```

Let's go CTD!

Subtitulo

Subtitulo

Subtitulo

Subtitulo

Subtitulo



Hierarquia

Apesar dos textos irem diminuindo de tamanho de acordo com a numeração das tags, elas não devem ser usadas considerando o tamanho.

Elas devem ser usadas de acordo com a necessidade de criar mais um título/subtítulo, sendo que o h6 é o limite dessa hierarquia.



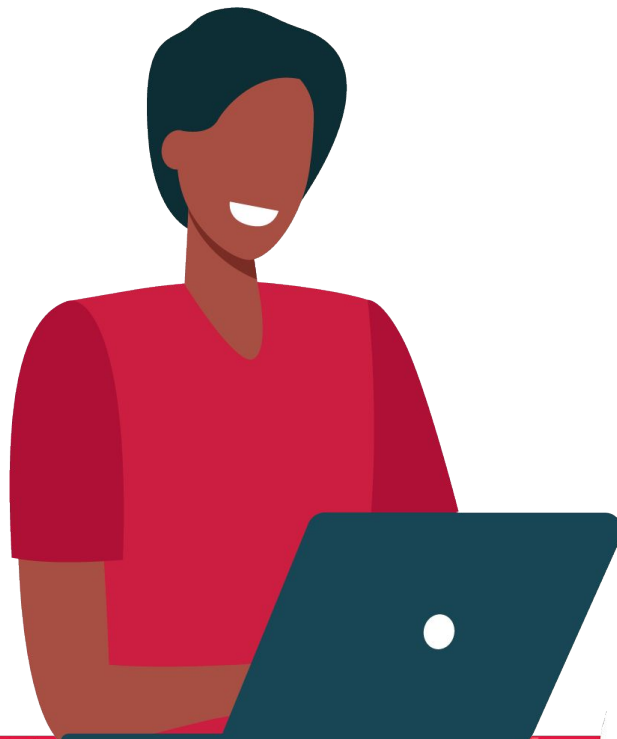
2 | Parágrafos



Objetivo

Agora que já descobrimos como definir títulos vamos aprender a definir parágrafos.

Os parágrafos são bem simples de serem utilizados, são representados pela tag `<p>`.





Definindo um parágrafo

Para criarmos um parágrafo precisamos apenas abrir a tag `<p>` escrever o conteúdo de nosso parágrafo e logo após isso fechar a nossa tag `</p>`.

HTML

```
<p>Olá, eu sou um parágrafo :)</p>
```

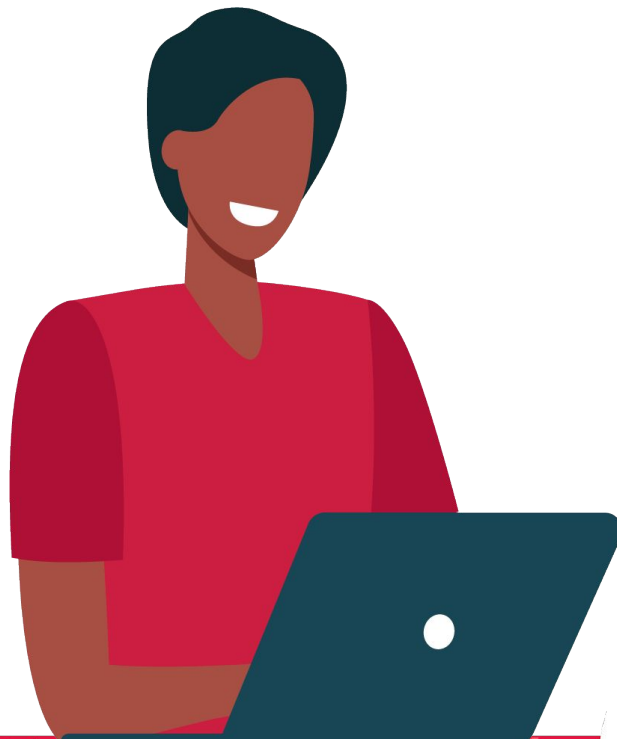
3 | Citações



Objetivo

Pode acontecer de querermos usar uma citação de alguém ao longo do desenvolvimento dos conteúdos da página, para isso temos as tags `<q>` e `<blockquote>`.

Bora dar uma olhada em como isso funciona!





Pequenas Citações - <q>

A tag <q> possibilita criar uma pequena citação de uma pessoa ou artigo de onde retiramos um trecho de nosso conteúdo textual dentro de um parágrafo.

HTML

```
<p>
```

```
<q>Eu não sei de nada cara - Sócrates</q>
```

```
</p>
```




Grandes Citações - `<blockquote>`

Já para as grandes citações pode usar o `<blockquote>` com um parágrafo.

HTML

```
<blockquote>
```

```
<p>
```

```
    Lorem ipsum dolor sit amet, consectetur adipiscing elit,  
    sed do eiusmod tempor incididunt ut labore et dolore  
    magna aliqua. Ut enim ad minim veniam, quis nostrud  
    exercitation.
```

```
</p>
```

```
</blockquote>
```



Direcionando a citação

Com o atributo **cite** nós podemos vincular aquela **citação** com um link de onde foi retirada, ela serve tanto para o **q** quanto para o **blockquote**.

HTML

```
<q cite="www.site.com"> citação... </q>
```

HTML

```
<blockquote cite="www.site.com"> citação...  
</blockquote>
```

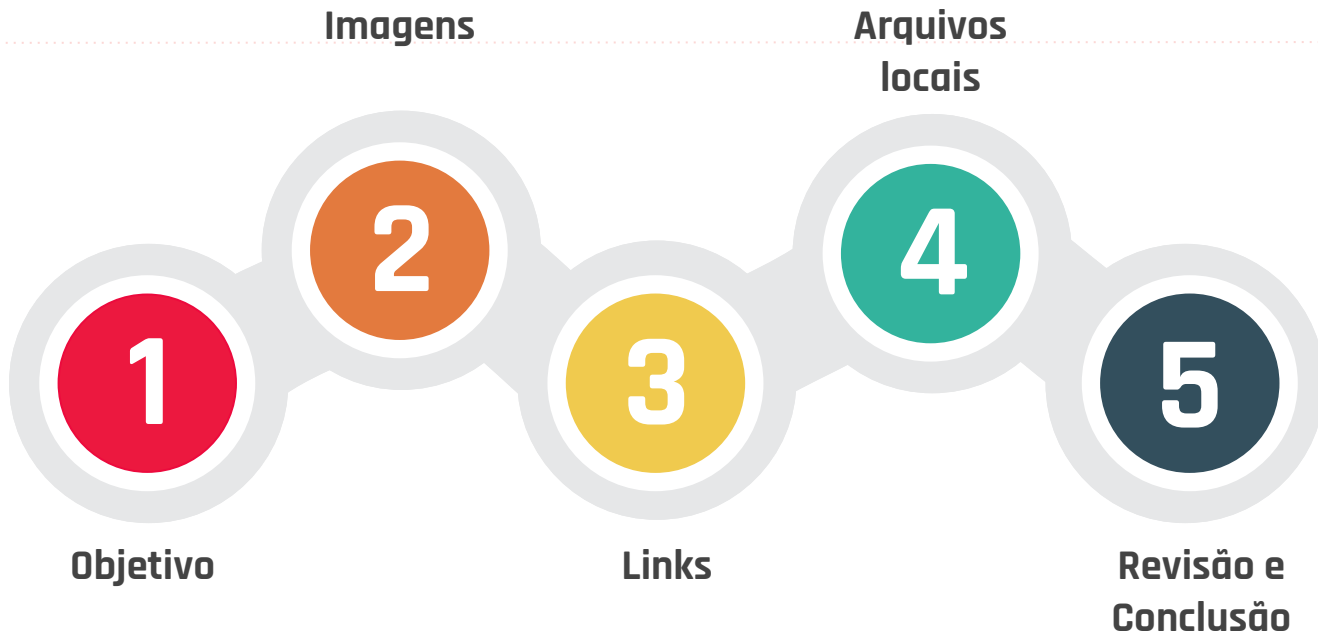
Hora da revisão



Conclusão

- Vimos que é possível definir Títulos e Subtítulos para os nossos conteúdos.
- Definição de parágrafos para representarmos textos.
- Citações curtas e longas de trechos ou frases de outras pessoas.

3 | Rotas

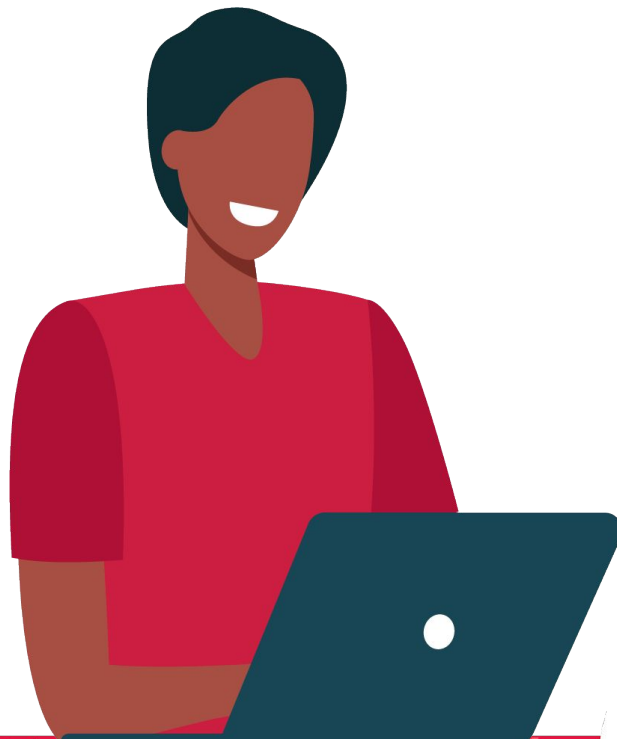




Objetivo

Agora aprenderemos um pouco sobre como funciona a importação de outros arquivos dentro do HTML

Para isso vamos utilizar o link `<a>` e imagem `` para conseguirmos referenciar arquivos locais e na internet!



1 | **Imagens**



Utilizando Imagens

Conseguimos inserir imagens em nossa página através da tag única ``.

Ela recebe um atributo **src** que nos permite especificar onde está nossa imagem!

HTML

```

```

2 | Links



Utilizando Links

Com os links o processo é o mesmo, porém agora ao invés de **src** temos o **href** e podemos também **definir um conteúdo para o nosso link**.

HTML

```
<a href="www.digitalhouse.com">Acessar a DH</a>
```

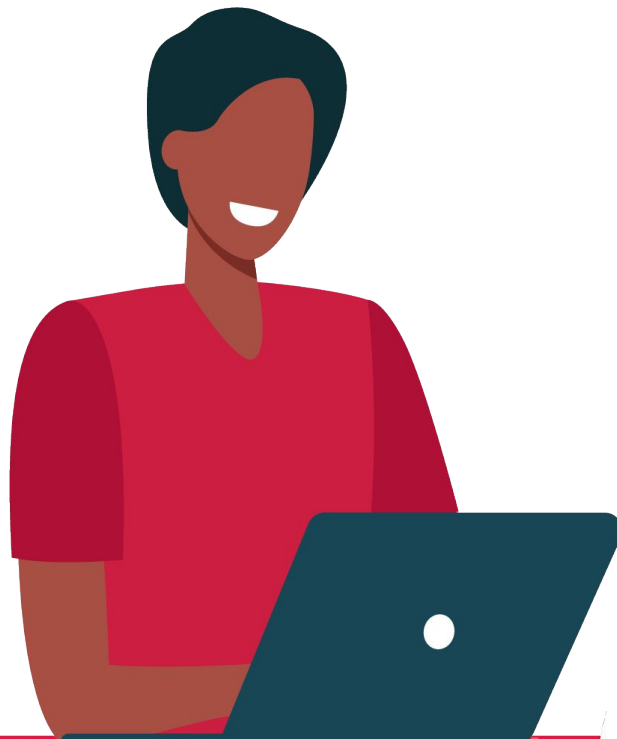
3 | Arquivos locais



Referenciando arquivos locais

Aprendemos a referenciar arquivos que estão na internet, mas e se quisermos utilizar uma foto que estivesse no computador ou fazer um link para um arquivo html local? Isso é possível?

Mas é claro, vamos ver isso **agora!**

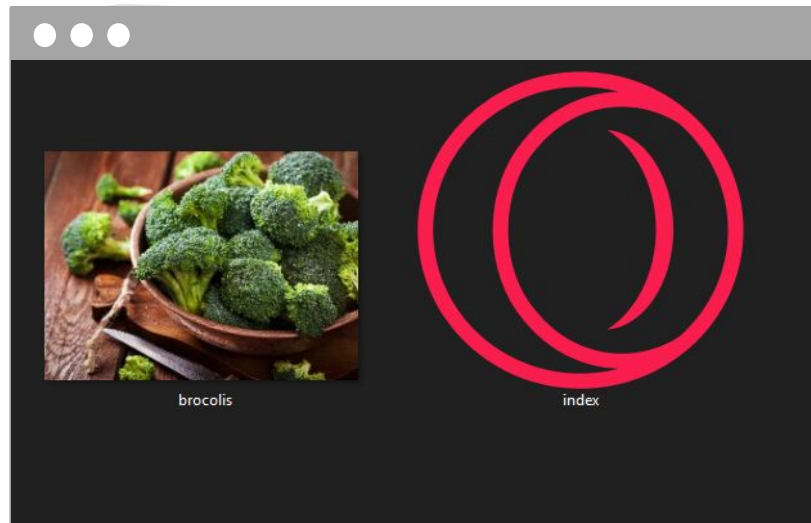




Referenciando arquivos locais

Para isso vamos imaginar o seguinte cenário:

Você tem uma pasta para o seu site onde estão os seus arquivos HTML e imagens, sendo eles, brócolis(sua imagem) e index(seu arquivo html).





Referenciando arquivos locais

Para referenciarmos a nossa imagem, precisamos dizer onde ela está partindo como ponto de referência o arquivo HTML em que estamos digitando.

Nesse caso como a imagem está na mesma pasta que o nosso arquivo a única coisa que precisamos fazer é digitar o nome do arquivo juntamente com seu formato.

HTML

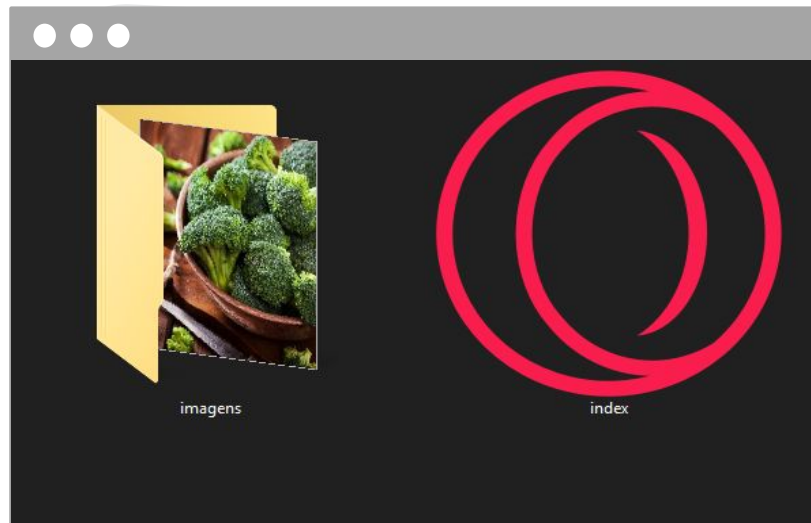
```
<img href="brocolis.jpeg">
```



Referenciando arquivos locais

Agora vamos imaginar um segundo cenário:

Você tem uma pasta para o seu site onde estão os seus arquivos HTML(index) e uma pasta(imagens) aonde seu arquivo de imagem(brócolis) está, o que fazer nessa situação?





Referenciando arquivos locais

Para referenciarmos a nossa imagem que está em outra pasta, precisamos referenciar a pasta em que ela está e logo em seguida dizer o nome e formato dela.

Para 'entrarmos' em uma pasta, basta colocar: **/nomeDaPagina**

HTML

```

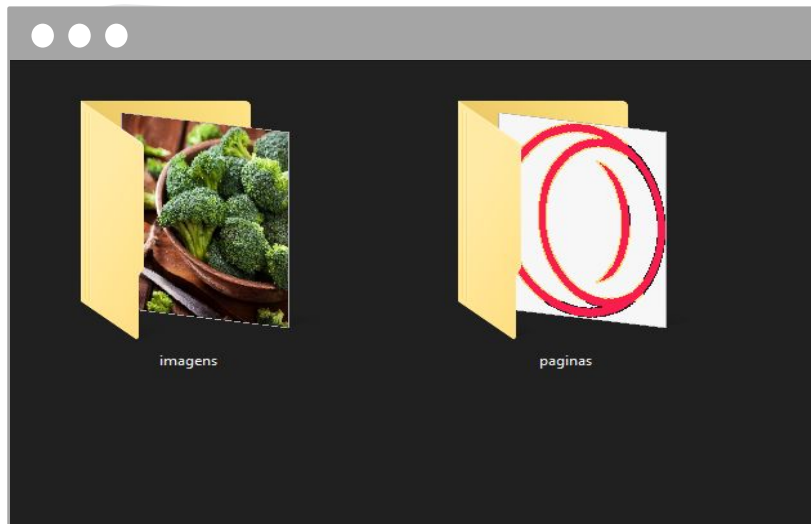
```



Referenciando arquivos locais

Agora vamos imaginar um terceiro cenário:

Você tem uma pasta para o seu site onde estão os seus arquivos, uma pasta para suas páginas que contém o seu arquivo HTML(index) e uma pasta(imagens) onde seu arquivo de imagem(brócolis) está, o que fazer nessa situação?





Referenciando arquivos locais

Como já havíamos comentado anteriormente, o nosso ponto de referência é o arquivo HTML que estamos digitando, nesse caso: index.html

Para referenciarmos a nossa imagens teremos que 'sair' da pasta atual em que o index.html está (páginas) para 'entrarmos' na pasta de imagens.

É muito simples 'sair' de uma pasta, basta acrescentar `../` que o HTML irá entender que você está querendo voltar uma pasta.

HTML

```

```

Hora da revisão



Conclusão

- Vimos que é possível referenciar imagens/sites e arquivos tanto por links quanto por arquivos locais.
- É possível acessar arquivos locais navegando entre os diretórios onde **/nomeDaPagina** serve para 'entrar' em uma pasta e **../** serve para 'sair' de uma página.

3 | Acessibilidade



Semântica



Objetivo

**Revisão e
Conclusão**



Objetivo

Acessibilidade nas nossas páginas é basicamente aplicar algumas técnicas para que as pessoas com algum tipo de dificuldade consigam navegar tranquilamente em nosso site sem estragar a sua experiência.

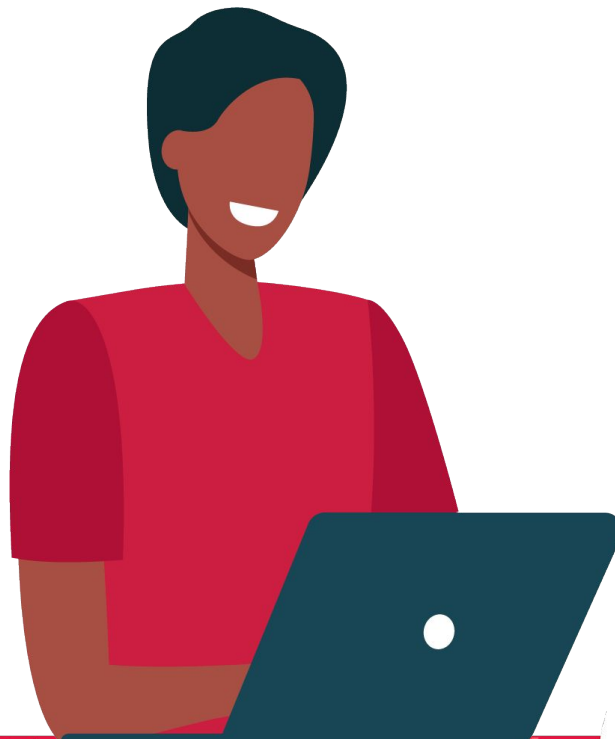




Leitores de Tela

Pessoas com baixa visão utilizam Leitores de Tela para poderem navegar melhor.

Para facilitarmos o trabalho dos leitores podemos utilizar diversas estratégias, mas iremos abordar duas, que são: HTML Semântico e Descrição de Imagens.





HTML Semântico

Quando falamos sobre Semântica queremos dizer **utilizar as tags corretas** para as **informações corretas**.



HTML

```
<div>
```

Olá eu sou um parágrafo

```
</div>
```

Texto inserido diretamente em uma tag div



HTML

```
<p>
```

Olá eu sou um parágrafo

```
</p>
```

Texto inserido em uma tag que representa um parágrafo



Descrição de Imagens

A tag `` nos fornece um atributo chamado **alt**, aonde podemos definir uma descrição da imagem.

Isso ajuda e muito os leitores de tela a descreverem o que está acontecendo na imagem para uma pessoa com baixa visão, por exemplo.

HTML

```
<img href="image.png" alt="Carros em alta velocidade">
```

Hora da revisão



Conclusão

- Podemos facilitar a navegação de pessoas com dificuldades através da Acessibilidade.
- Descrição de imagens e especialmente HTML Semântico são indispensáveis para proporcionar uma boa experiência para pessoas com dificuldade.

4 | GIT



GIT

**Atualizando nosso
repositório**



Objetivo

GITHub

**Revisão e
Conclusão**

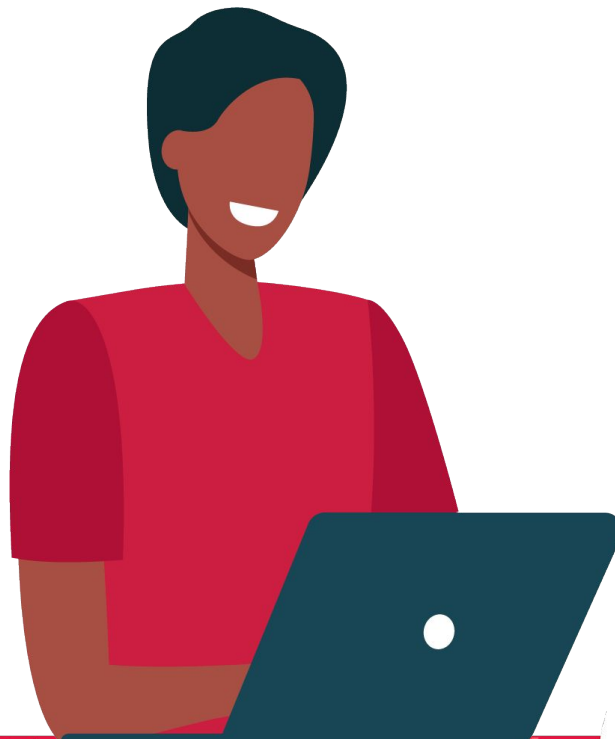
1 | GIT



Objetivo

GIT é o melhor sistema de versionamento e o seu melhor amigo na hora de separar arquivos para seu projeto.

Ele possibilita várias pessoas trabalharem no mesmo projeto de maneira ágil!





Instalando

Para instalarmos o git basta irmos até o [site oficial](#) e realizarmos o download.

Após isso é só instalar o programa normalmente como qualquer outro!

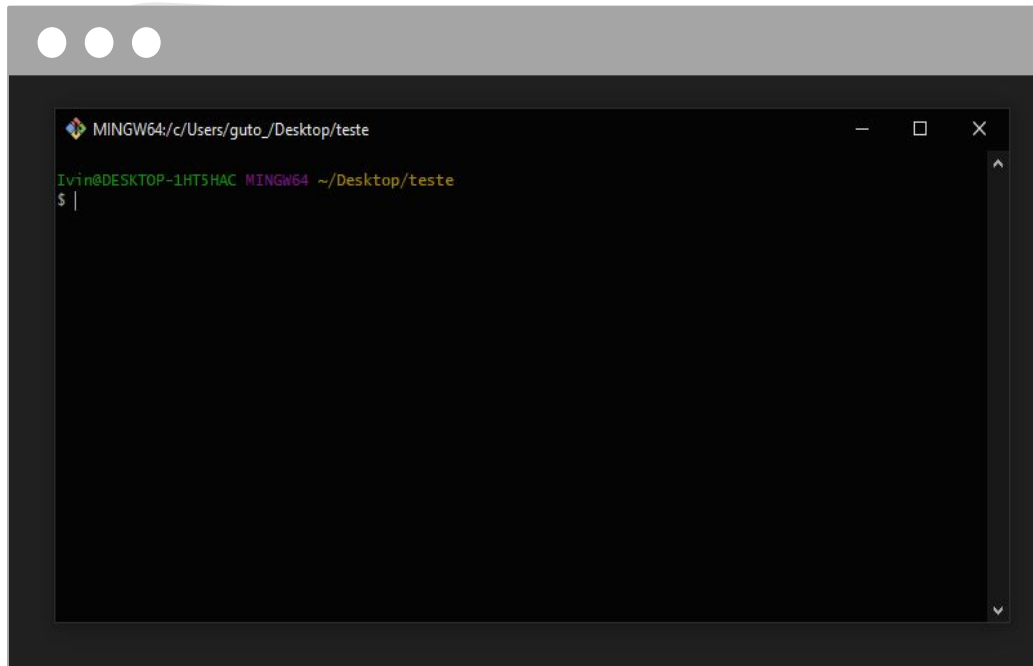
Após instalado clique com o botão direito em alguma pasta e veja se a opção **git bash here** foi adicionada.





Sobre o GIT

GIT é um terminal onde podemos digitar comandos e controlar o versionamento de nosso projeto de maneira fácil e simples.



2 | GITHub



Objetivo

GITHUB é um dos lugares que nos permitem alocar os nossos projetos para que outras pessoas possam baixar, contribuir e até mesmo avaliar.

Faça um cadastro no GITHUB, é bem fácil e intuitivo.

GitHub



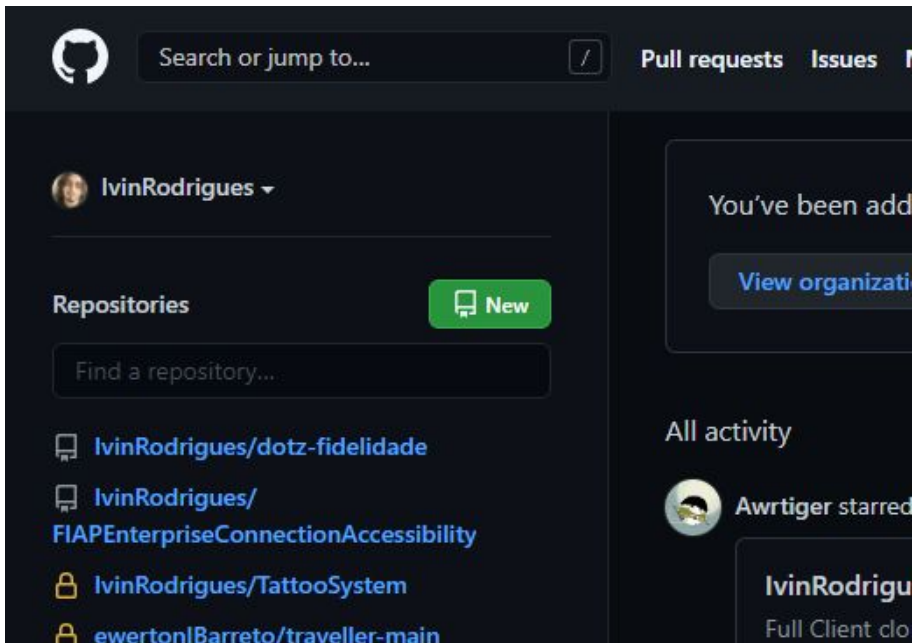


Como as coisas funcionam

Agora que você já criou uma conta, vamos criar um repositório.

Repositório é onde o seu projeto irá ser armazenado.

Já logado no git, clique no botão verde **new** que está à esquerda da tela.





Certified Tech
Developer

The Ultimate Degree

Como as coisas funcionam

Após clicar em new, você será redirecionado a essa tela.

Preencha o nome do seu repositório e clique no outro botão verde **Create repository** na parte inferior da tela.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *



IvinRodrigues ▾

Repository name *

/ aula05teste



Great repository names are short and memorable. Need inspiration? How about [furry-invention?](#)

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☒ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

☒ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

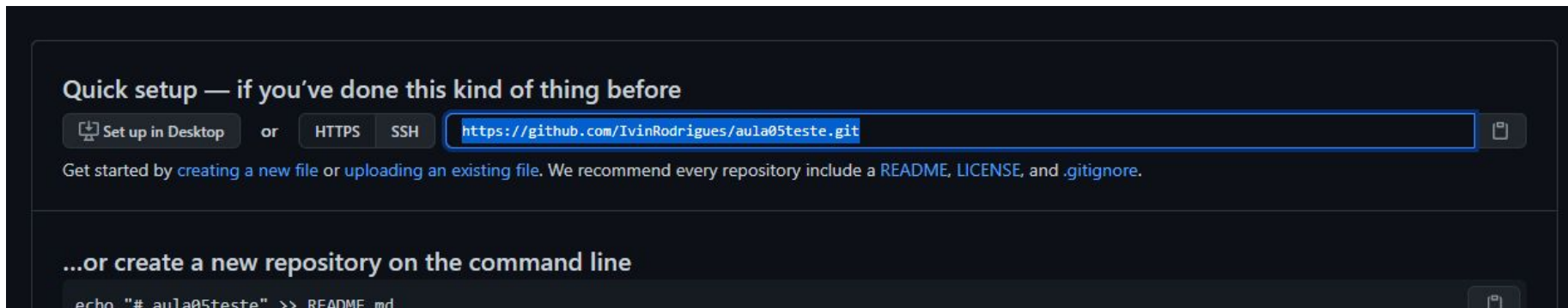
Create repository



Como as coisas funcionam

Pronto, agora seu repositório está criado! O próximo passo agora é baixá-lo em seu computador.

Para isso, copie o link que ele te fornece no campo superior direito na primeira seção:





Como as coisas funcionam

Com o link do nosso repositório criado, vamos cloná-lo em nossa máquina.

Para isso, abra o terminal do git através do botão direito do mouse, depois clique em **Git Bash Here**.

O terminal do GIT irá aparecer e o que precisamos fazer é digitar o comando **git clone**, colar o nosso link e pressionar Enter.

GIT

```
git clone LinkQueCopiamos
```



Como as coisas funcionam

Após a clonagem ser concluída, uma pasta com o nome do repositório vai ser criada. No terminal do git, digite **cd nomeDaPastaGerada** e o git irá acessar a pasta que acabamos de clonar.

Agora, qualquer arquivo que criarmos dentro dessa pasta nós poderemos enviar de maneira fácil para o nosso repositório do GitHub.



aula05teste

MINGW64:/c/Users/guto_/Desktop/teste

```
Ivin@DESKTOP-1HT5HAC MINGW64 ~/Desktop/teste
$ git clone https://github.com/IvinRodrigues/aula05
Cloning into 'aula05teste'...
warning: You appear to have cloned an empty repository.

Ivin@DESKTOP-1HT5HAC MINGW64 ~/Desktop/teste
$ |
```

3

Atualizando o nosso Repositório



Objetivo

Vamos criar um arquivo qualquer em nossa pasta clonada, apenas para podermos atualizar nosso repositório.

Crie um novo arquivo de texto dentro da pasta clonada chamado **teste**.

The screenshot shows a terminal window on the right and a file explorer window on the left. The terminal window displays the following commands and output:

```
lvin@DESKTOP-LHT1MAC: ~/Desktop/teste
$ git clone https://github.com/IvinRodrigues
Cloning into 'aula01teste'...
warning: You appear to have cloned an empty repository.
$
```

The file explorer window on the left shows the path 'teste > aula01teste' and indicates that the folder is empty ('Esta pasta está vazia.').



Atualizando o nosso Repositório

Após criar o arquivo, vá para o terminal do git e digite **git status**, ele é o comando responsável por nos dizer o status da nossa branch atual.

Após fazer isso, ele irá dizer que o arquivo teste.txt foi criado, porém ele ainda não foi commitado!



teste

```
MINGW64/c/Users/guto/Desktop/teste/aula05teste

Ivin@DESKTOP-1HT5HAC MINGW64 ~/Desktop/teste/aula05teste (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
       teste.txt

nothing added to commit but untracked files present (use "git add" to track)

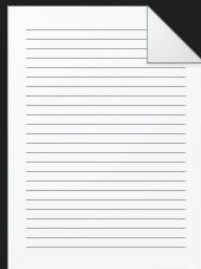
Ivin@DESKTOP-1HT5HAC MINGW64 ~/Desktop/teste/aula05teste (master)
$ |
```



Atualizando o nosso Repositório

Para commitarmos os nossos arquivos, é necessário adicioná-los como arquivos que queremos salvar.

Para isso, temos os comandos: **git add**, que nos permite justamente adicionar os arquivos, e **user git add**, para adicionar todos os arquivos de uma vez!



teste

```
MINGW64/c/Users/guto/Desktop/teste/aula05teste

Ivin@DESKTOP-IHTSHAC MINGW64 ~/Desktop/teste/aula05teste (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
       teste.txt

nothing added to commit but untracked files present (use "git add" to track)
Ivin@DESKTOP-IHTSHAC MINGW64 ~/Desktop/teste/aula05teste (master)
$ git add .

Ivin@DESKTOP-IHTSHAC MINGW64 ~/Desktop/teste/aula05teste (master)
$ |
```

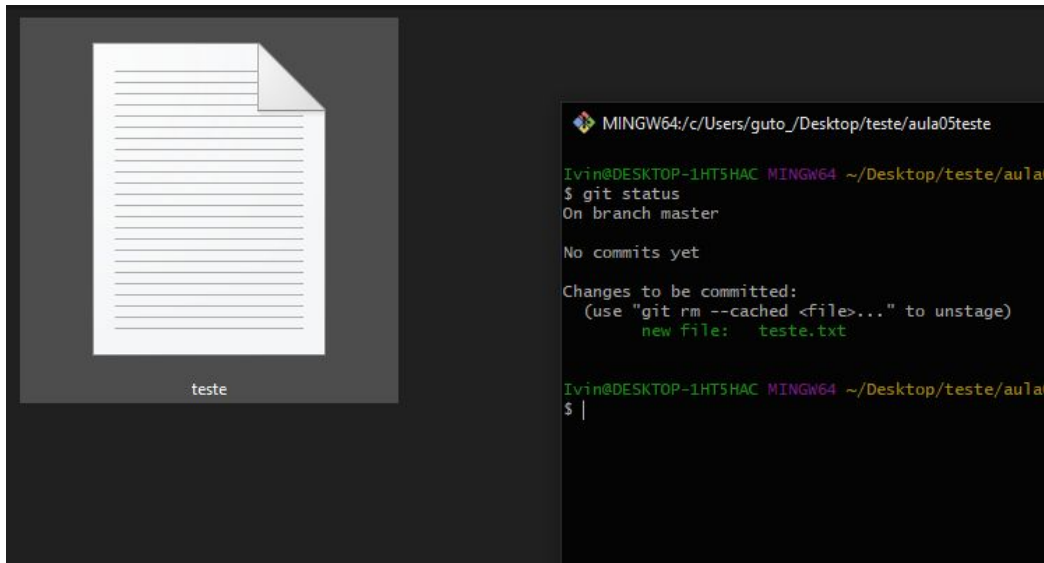


Atualizando o nosso Repositório

Confira se o arquivo foi de fato adicionado, utilizando **git status** novamente.

Caso o nome do arquivo apareça em verde, significa que tudo correu bem.

Caso apareça em vermelho, certifique-se de que o git está na pasta do repositório.





Atualizando o nosso Repositório

Após a checagem, podemos de fato commitar as mudanças utilizando o **git commit**.

É importante fornecer uma mensagem que explique brevemente qual alteração foi feita.

GIT

```
git commit -m 'sua mensagem aqui'
```




Atualizando o nosso Repositório

Com o commit realizado estamos prontos para subir as alterações para nosso repositório no GitHub.

Para isso, precisamos digitar o comando **git push origin**, que irá justamente fazer o upload das nossas mudanças para o nosso repositório online.

GIT

```
git push origin
```



Atualizando o nosso Repositório

Após a execução do comando **git push origin**, vamos ter uma confirmação de que o upload foi realizado com sucesso.

Agora, se voltarmos a página do nosso repositório do GITHub na primeira aba chamada **code**, veremos que o arquivo de texto que criamos está lá!

```
MINGW64:/c/Users/guto_/Desktop/teste/aula05teste

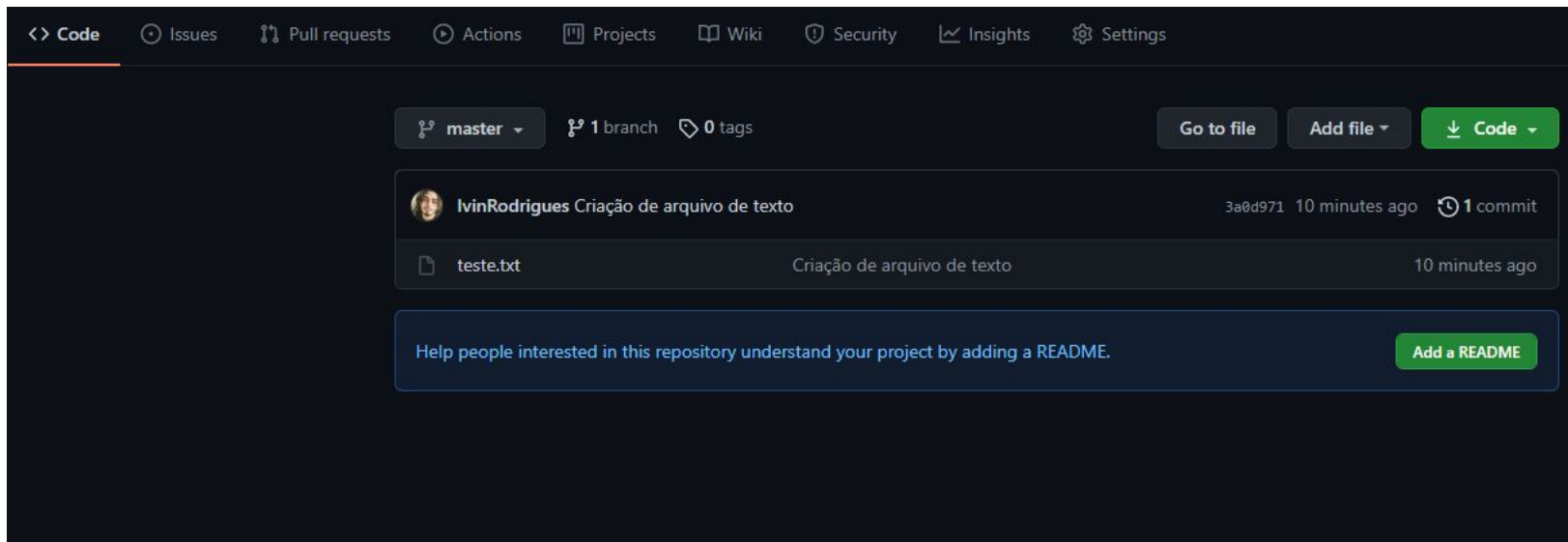
Ivin@DESKTOP-1HT5HAC MINGW64 ~/Desktop/teste/aula05teste (mas
$ git push origin
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 235 bytes | 235.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/IvinRodrigues/aula05teste.git
 * [new branch]      master -> master

Ivin@DESKTOP-1HT5HAC MINGW64 ~/Desktop/teste/aula05teste (mas
$
```



Atualizando o nosso Repositório

Arquivo de texto que criamos, commitados e subimos utilizando apenas o GIT.



Hora da revisão



Conclusão

- Vimos que é possível criarmos repositórios no GitHub para salvarmos nossos projetos.
- É possível criar arquivos locais e depois fazer o upload deles em nosso repositório utilizando apenas 3 comandos.

DigitalHouse>
Coding School