

• git ignore

Nome	Data de modificação	Tipo	Tamanho
.git	24/06/2022 22:23	Pasta de arquivos	
.gitignore.txt	24/06/2022 22:48	Documento de Te...	1 KB
Texto1.txt	24/06/2022 21:24	Documento de Te...	1 KB
Texto2.txt	24/06/2022 21:54	Documento de Te...	0 KB
Texto3.txt	24/06/2022 22:16	Documento de Te...	0 KB
Texto4.txt	24/06/2022 22:20	Documento de Te...	0 KB
Texto5.txt	24/06/2022 22:48	Documento de Te...	0 KB

Crie um arquivo com o nome **.gitignore**.

Para remover a extensão **.txt** do arquivo **.gitignore**, clique em cima do arquivo aperte a tecla **F2**, retire o **arquivo txt** e pressione **ENTER**. Caso apareça uma janela de confirmação de remoção da extensão pressione **SIM** para aceitar. Vide imagem abaixo.


Disco Local (C:) > Teste2022

Nome

^

Nome	Data de modificação	Tipo	Tamanho
.git	24/06/2022 22:23	Pasta de arquivos	
.gitignore	24/06/2022 22:48	Documento de Te...	1 KB
Texto1.txt			
Texto2.txt			
Texto3.txt			
Texto4.txt			
Texto5.txt			

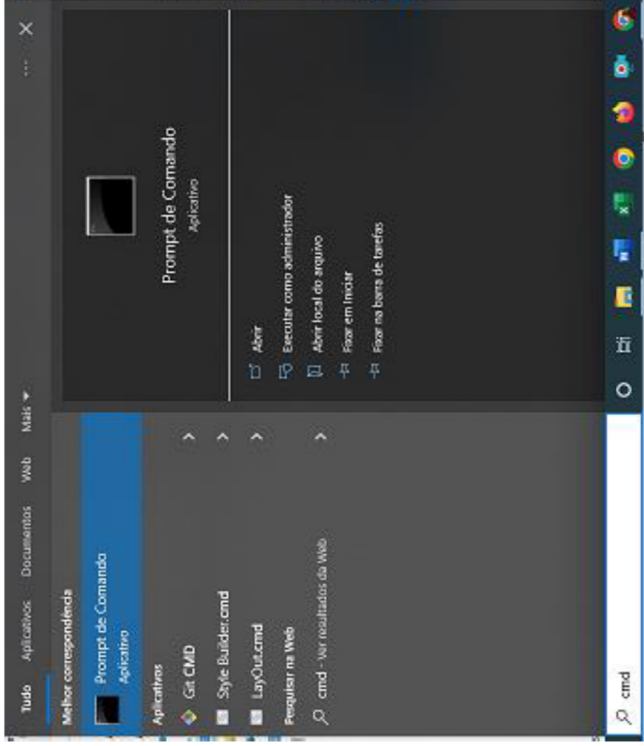
Renomear



Se a extensão de um nome de arquivo for alterada, o arquivo poderá se tornar inutilizável.

Tem certeza de que deseja alterá-la?

- git ignore



Caso não seja possível realizar a alteração da extensão utilizado o recurso acima. Também é possível realizar a remoção da extensão através do prompt de comando.

Digite **CMD** na barra de pesquisa, clique em Prompt de Comando

 Prompt de Comando

```
C:\Users\User>cd\teste2022
```

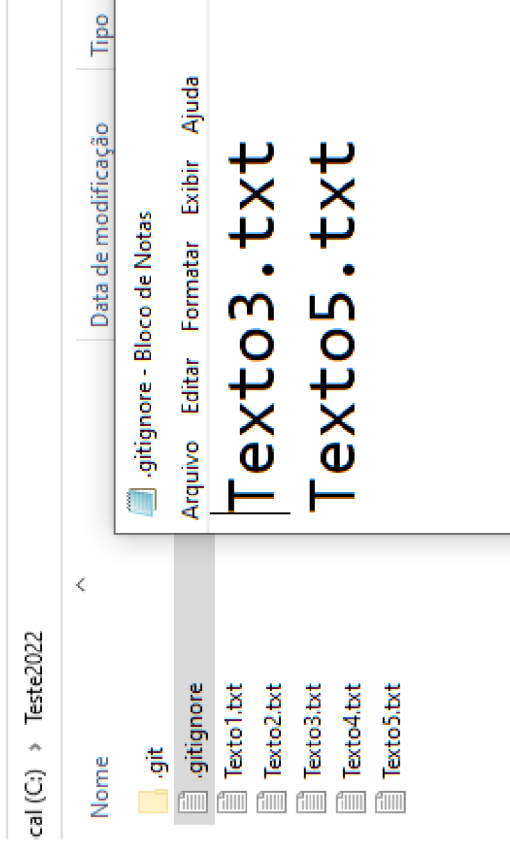
```
C:\Teste2022>ren .gitignore.txt .gitignore
```

Caso a sua pasta onde está sendo realizado o versionamento seja igual ao exemplo → **teste2022**.

Digite **cd\teste2022** para acessar a pasta.

Ao acessar a pasta teste2022 digite o comando ren **.gitignore.txt .gitignore**

- git ignore



Acesse o arquivo **.gitignore** e digite o nome dos arquivos que você não queira que seja rastreado pelo git.

No exemplo ao lado os arquivos **Texto3.txt** e **Texto5.txt** não serão rastreado portando não apareceram no rastreamento do git.

Antes dos arquivos **Texto3.txt** e **Texto5.txt** serem adicionados ao arquivo **.gitignore** eles eram rastreados pelo git como mostra a janela ao lado. Após a inclusão dos arquivos no **.gitignore** eles não serão mais listados como mostra a imagem abaixo.

```
User@DESKTOP-8PNLGLK MINGW64 /c/Teste2022
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in commit)
        .gitignore
        Texto3.txt
        Texto4.txt
        Texto5.txt
```

```
User@DESKTOP-8PNLGLK MINGW64 /c/Teste2022
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what
  (use "git restore <file>..." to discard
    modified:   .gitignore
```

- branch

```
User@DESKTOP-8PNLG1K MINGW64 /c/Teste2022 (master)
$ git branch
* master
```

Para consultar quantas branch's o git possui na pasta de controle de versão digite **git branch**.

```
User@DESKTOP-8PNLG1K MINGW64 /c/Teste2022 (master)
$ git branch teste
```

Para criar uma nova branch ou ramificação digite **git branch** e o nome no exemplo ao lado o nome da branch é teste.

```
User@DESKTOP-8PNLG1K MINGW64 /c/Teste2022
$ git branch
* master
  teste
```

Ao consultar as branch's na pasta onde está sendo realizado o versionamento agora aparecem duas branch's.

```
User@DESKTOP-8PNLG1K MINGW64 /c/Teste2022 (master)
$ git checkout teste
Switched to branch 'teste'

User@DESKTOP-8PNLG1K MINGW64 /c/Teste2022 (teste)
$ |
```

Para acessar a nova branch criada basta digitar o comando **git checkout** e o nome da branch criada.

- reset

```
User@DESKTOP-8PNLGL1K MINGW64 /c/TESTE2022 (master)
$ git log
commit ee551d66380fb2ac6fa952c939c4d48cc12db313 (HEAD -> master)
Author: reinaldo.santos <09105590@senaimgdcente.com>
Date:   Sun Jun 26 22:42:13 2022 -0300

    .gitignore

commit 507785f8ef3f6a3889f82545cd510f5aa74d6b26
Author: reinaldo.santos <09105590@senaimgdcente.com>
Date:   Sun Jun 26 22:33:34 2022 -0300

    Texto 4.txt
```

Caso seja necessário estornar/desfazer um commit realizado basta utilizar o comando git reset que pode ser utilizado com as variações:

- git reset --soft **id do commit**
- git reset --mixed **id do commit**
- git reset --hard **id do commit**

- **--soft** retorna o commit na área de staging pronto para ser commitado.
- **--mixed** retorna o commit em ponto de ser adicionado na área de staging para ser commitado posteriormente.
- **--hard** retorna o commit e desfaz todo processo realizado incluindo a eliminação dos conteúdos(arquivos) criados.