Trabalhando com arquivos

GitHub

https://github.com/

Trabalhando com arquivos:
Criar, mover, excluir, reverter, log, show, checkout
Revert.

GitHub

Durante o trabalho no repositório local, acontecem muitas modificações, arquivos são criados, movidos, editados, novas subpastas são criadas, e todas essas modificações são rastreadas pelo GIT.

Quando commitadas essas modificações podem ser até mesmo revertidas.

Os commits e as alterações realizadas podem ser visualizados com comandos específicos.

As modificações e a movimentação podem ser realizadas via ambiente gráfico, ou via linha de comandos.

Trabalhando com arquivos:
Criar e excluir.

Criando um arquivo e excluindo ele em seguida sem adicionar.

Se um arquivo é criado e não é adicionado, ele aparecerá no status como untracked.

Caso mude de ideia, e o arquivo não seja mais necessário, é possível excluí-lo.

Para remover, basta simplesmente excluir ele da pasta.

Após a exclusão, basta verificar novamente o status para ver que não existem mais conteúdos como untracked a serem adicionados ou commitados.

Qualquer arquivo ou pasta modificado (criado/editado/movido) mas não adicionado, pode ter essa modificação revertida sem necessidade de commit.

Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Criar e excluir.

GitHub

Criando arquivo SEM ADICIONAR.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git status                                                    #verificar o estado do repositório
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

Fulano@Win11 ~/Documents/projeto1 (main)
$ touch arquivo1.txt                                            #criando arquivo vazio

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status                                                    #verificar o estado do repositório
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        arquivo1.txt                                            #arquivo não rastreado encontrado

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

Trabalhando com arquivos:
Criar e excluir.

Removendo arquivo NÃO ADICIONADO.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git status                                                    #verificar o estado do repositório
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        arquivo1.txt                                            #arquivo não rastreado encontrado

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

Fulano@Win11 ~/Documents/projeto1 (main)
$ rm arquivo1.txt                                               #removendo arquivo não adicionado

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean                           #sem alterações no status
```

Trabalhando com arquivos:
Criar, editar, mover.

GitHub

Criando um arquivo e excluindo ele em seguida após ele ser adicionado.

Quando um arquivo/pasta, que foi adicionado e commitado é editado, excluído ou movido, ele irá aparecer no status como modified, deteted ou renamed.

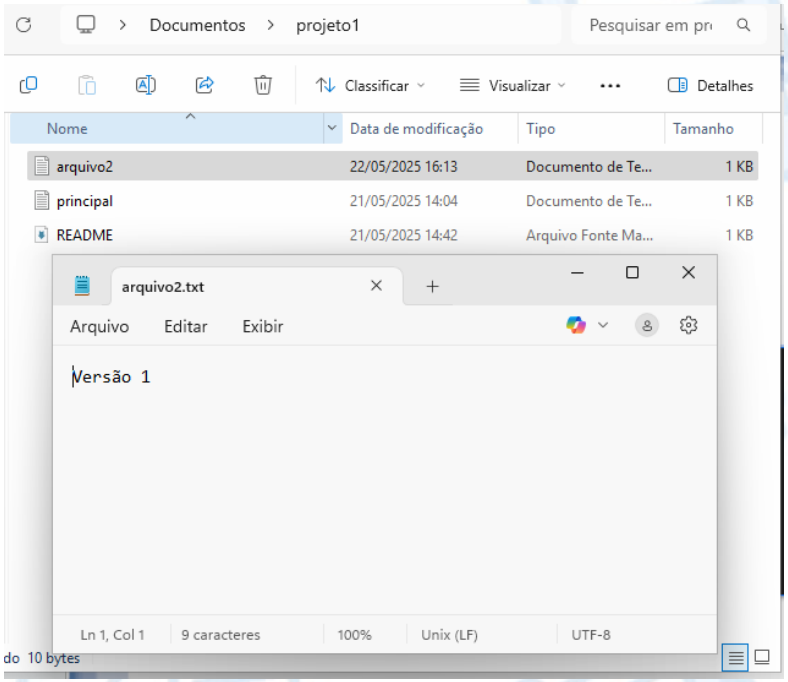
Deve ser adicionado novamente e em seguida commitado.

Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Criar e editar.

Criando

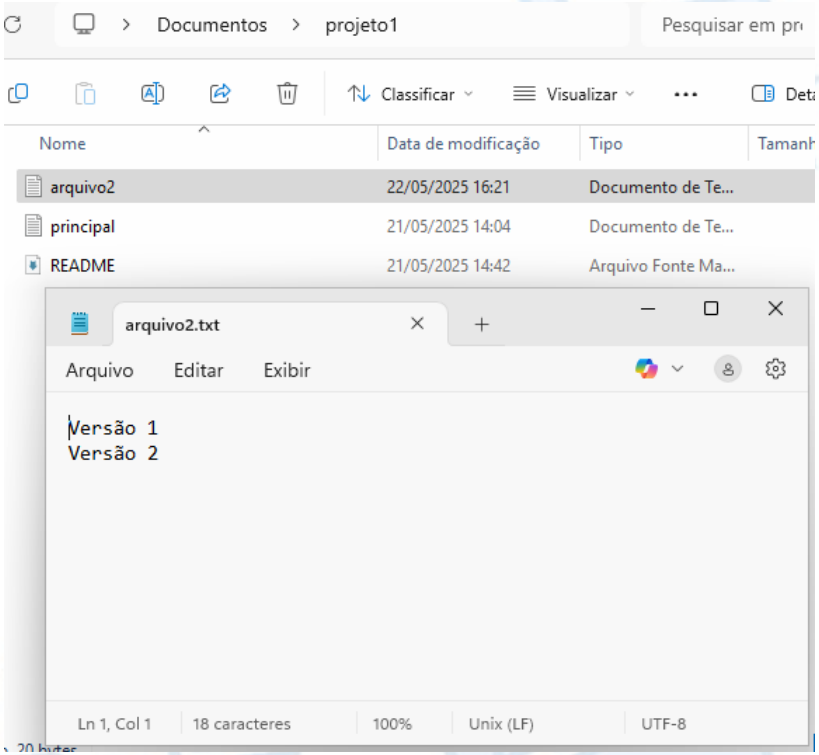
```
Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
README.md  principal.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo "Versão 1" >> arquivo2.txt                    #criando arquivo

Fulano@Win11 ~/Documents/projeto1 (main)
$ git add .

Fulano@Win11 ~/Documents/projeto1 (main)
$ git commit -m "Adicionado arquivo2.txt versão 1"

Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
README.md  arquivo2.txt  principal.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```



Trabalhando com arquivos:
Criar e editar.

Editando

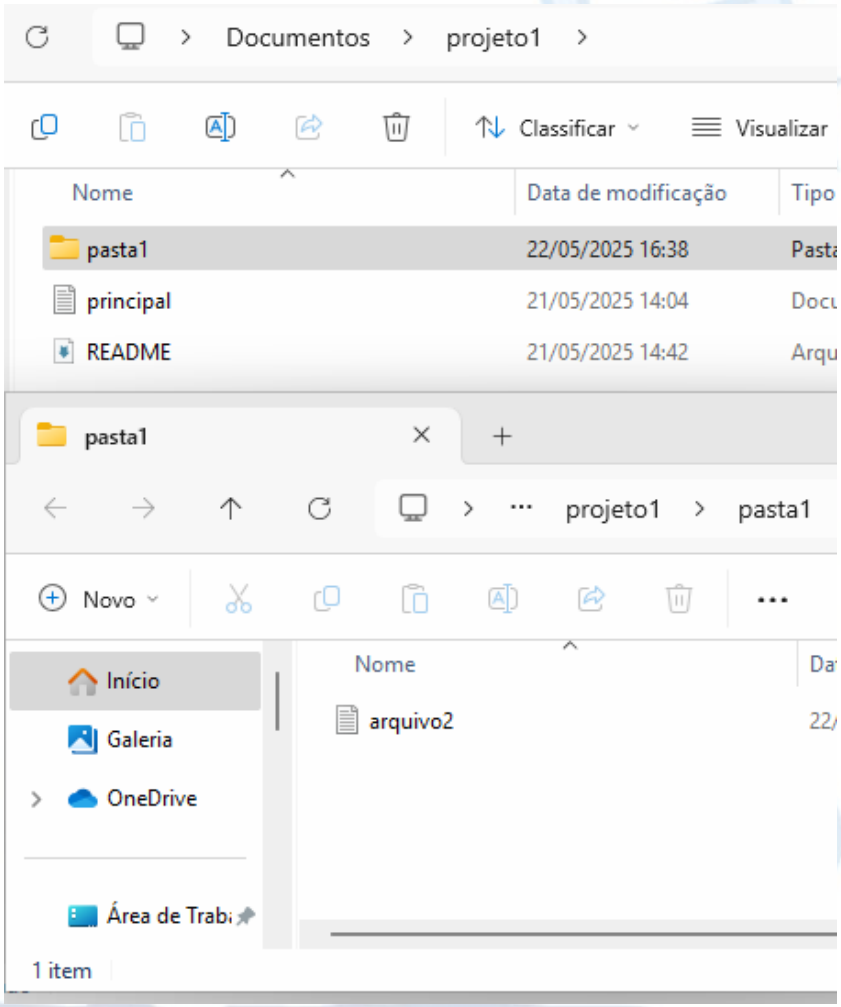
```
Fulano@Win11 ~/Documents/projeto1 (main)
$ echo "Versão 2" >> arquivo2.txt                    #editando  arquivo

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   arquivo2.txt
no changes added to commit (use "git add" and/or "git commit -a")

Fulano@Win11 ~/Documents/projeto1 (main)
$ git add .

Fulano@Win11 ~/Documents/projeto1 (main)
$ git commit -m "editado versão 2"

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```



Trabalhando com arquivos:
Criar e mover.

Criando pastas e movendo arquivos commitados

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ mkdir pasta1                                  #criando pasta

Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
README.md  arquivo2.txt  pasta1/  principal.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ git mv arquivo2.txt pasta1/                   #movendo arquivo

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    arquivo2.txt -> pasta1/arquivo2.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
README.md  pasta1/  principal.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ ls pasta1/
arquivo2.txt
```

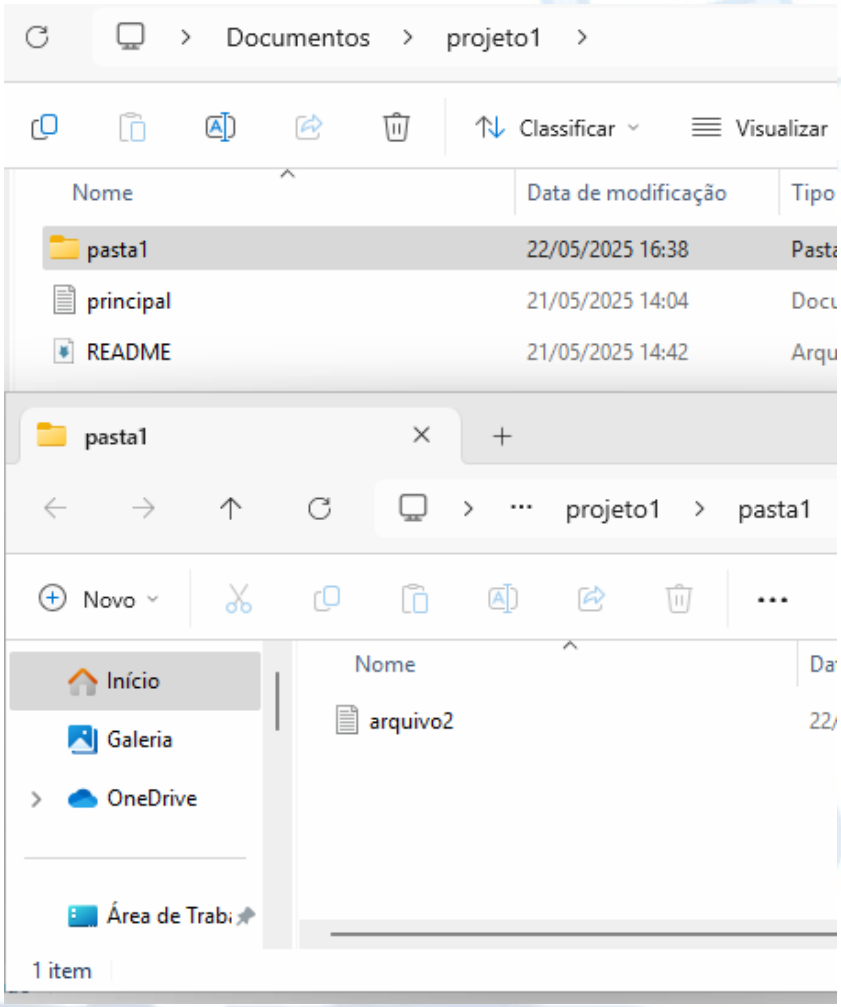


Trabalhando com arquivos:
Criar e mover.

GitHub

Criando pastas e movendo arquivos commitados

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    arquivo2.txt -> pasta1/arquivo2.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ add .

Fulano@Win11 ~/Documents/projeto1 (main)
$ git commit -m "arquivo2 movido para pasta 1"

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```



Trabalhando com arquivos:
Reverter arquivos.

GitHub

Após editar um arquivo adicionado anteriormente, e antes de adicioná-lo novamente, é possível reverter o arquivo para o estado anterior.

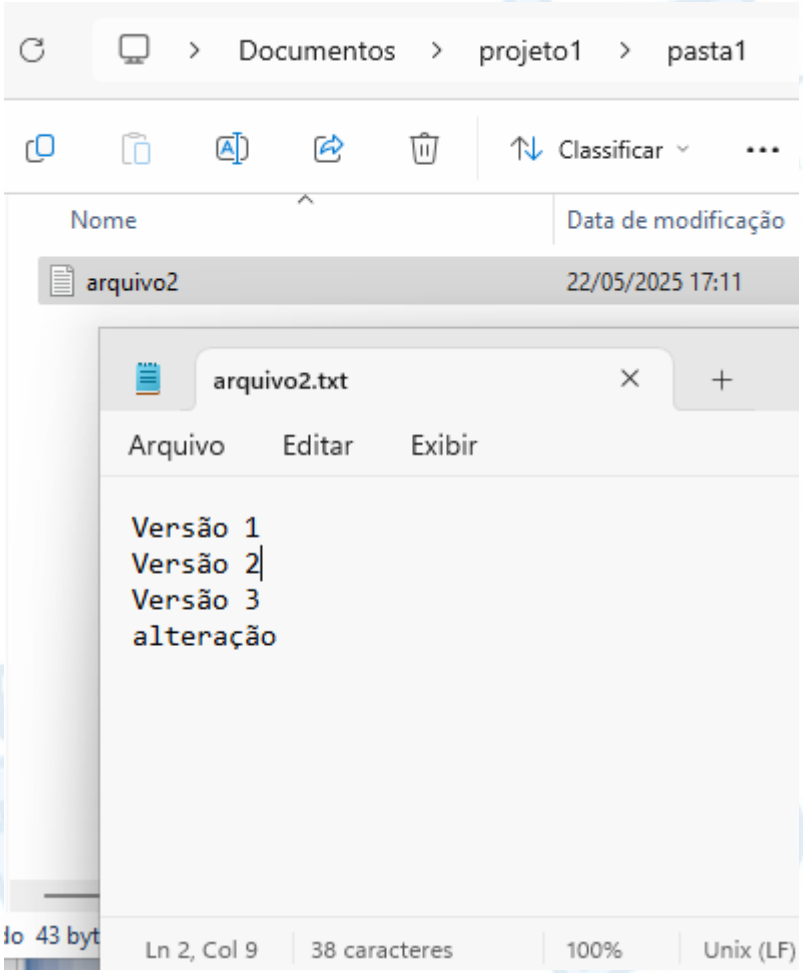
A operação pode ser realizada com o comando `checkout`.

Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Reverter arquivos.

Revertendo arquivo adicionado e alterado.

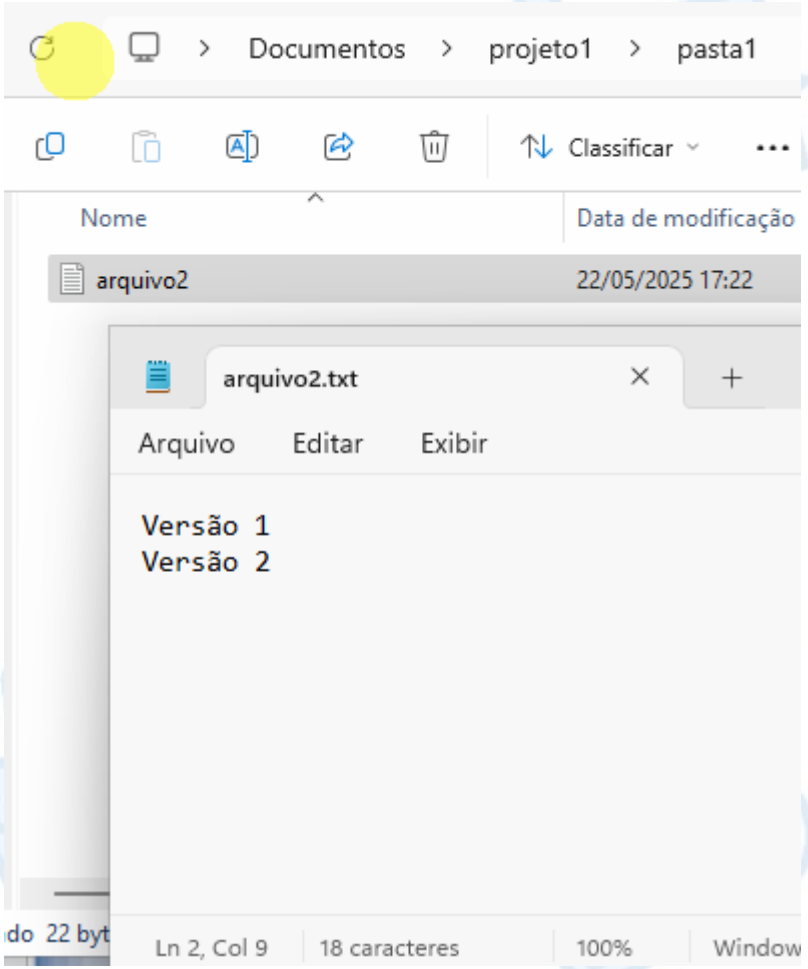
```
Fulano@Win11  ~/Documents/projeto1/pasta1  (main)
$ ls
arquivo2.txt

Fulano@Win11 ~/Documents/projeto1/pasta1 (main)
$ echo -e "Versão 3 \nalteração" >> arquivo2.txt            #editando
arquivo

Fulano@Win11 ~/Documents/projeto1/pasta1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   arquivo2.txt
no changes added to commit (use "git add" and/or "git commit -a")
```



Trabalhando com arquivos:
Reverter arquivos.

Revertendo arquivo adicionado e alterado.

```
Fulano@Win11 ~/Documents/projeto1/pasta1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   arquivo2.txt
no changes added to commit (use "git add" and/or "git commit -a")

Fulano@Win11 ~/Documents/projeto1/pasta1 (main)
$ git checkout arquivo2.txt
Updated 1 path from the index

Fulano@Win11 ~/Documents/projeto1/pasta1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```



Trabalhando com arquivos:
Log visualizar.

GitHub

Visualizando logs dos commits:

Com o comando `git log` é possível visualizar o log de modificações dos commits.

`git log` - Últimos commits feitos, do mais novo para o mais antigo.

`git log <aquivo>` - Exibe commits feitos de um arquivo específico.

`git log --oneline` - Exibe commits em somente uma linha cada, resumido.

Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Log

GitHub

Visualizando log dos commits, comando: `gti log`

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log
commit a923ed29c75963811d2994f6529c5d552b2d5132 (HEAD -> main)
Author: fulano<fulano@example.com>
Date:   Thu May 22 16:48:46 2025 -0300

    arquivo2 movido para pasta 1

commit 0214413933be075a06423e354238a9673274dadd
Author: fulano<fulano@example.com>
Date:   Thu May 22 16:26:11 2025 -0300

    editado versão 2

commit f450b4ce93d256395a21bd99b33154332772bfca
Author: fulano<fulano@example.com>
Date:   Thu May 22 16:16:13 2025 -0300

    Adicionado arquivo2.txt versão 1

commit c6569ffc24dee51eac51c9d540207af98b12e024 (origin/main)
Author: fulano<fulano@example.com>
Date:   Wed May 21 14:43:12 2025 -0300

    Add arquivo README

commit 2175e23cb8841468bba5551f31c29644a4d8c802
Author: fulano<fulano@example.com>
Date:   Wed May 21 14:18:52 2025 -0300

    Commit inicial do projeto1
```

O log visualizado é um arquivo, para sair dele após a leitura, basta clicar na tela **“Q”** do teclado.

Trabalhando com arquivos:
Log

GitHub

Visualizando log dos commits, comando: `gti log <arquivo>`

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log arquivo2.txt

commit a923ed29c75963811d2994f6529c5d552b2d5132
(HEAD -> main)
Author: fulano<fulano@example.com>
Date:   Thu May 22 16:48:46 2025 -0300

    arquivo2 movido para pasta 1



Fulano@Win11 ~/Documents/projeto1 (main)
$ git log ../README.md
commit c6569ffc24dee51eac51c9d540207af98b12e024
(origin/main)
Author: fulano<fulano@example.com>
Date:   Wed May 21 14:43:12 2025 -0300

    Add arquivo README
```

O log de arquivos específicos.

Trabalhando com arquivos:
Log

GitHub

Visualizando log dos commits, comando: `git log --oneline <arquivo>`

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline
a923ed2 (HEAD -> main) arquivo2 movido para pasta 1
0214413 editado versão 2
f450b4c Adicionado arquivo2.txt versão 1
c6569ff (origin/main) Add arquivo README
2175e23 Commit inicial do projeto1
```

O log dos commits simplificado.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo2.txt
a923ed2 (HEAD -> main) arquivo2 movido para pasta 1
```

O log de arquivo específico simplificado.

É possível notar que tanto na versão completa como na simplificada, cada commit tem um identificador único.

Trabalhando com arquivos:
Show visualizar.

GitHub

Visualizando modificações nos arquivos entre os commits:

Com o comando `git show` é possível visualizar as modificações dos arquivos em relação aos seus diferentes commits.

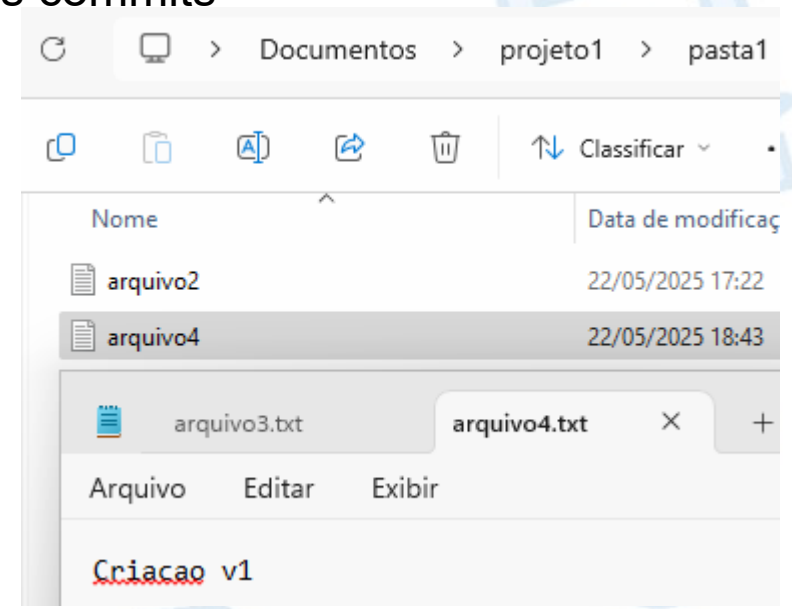
`git show <id_commit com 4 dígitos>` - Mostra as modificações do arquivo no commit indicado.

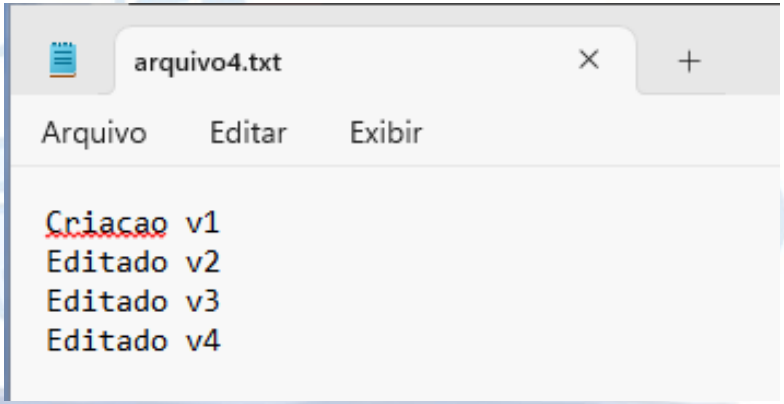
Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Show

Gerando arquivo novo, algumas modificações e seus commits

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ echo "Criacao v1" >> arquivo4.txt
$ git add .
$ git commit -m "Criacao v1"

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo "Editado v2" >> arquivo4.txt
$ git add .
$ git commit -m "Editado v2"

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo "Editado v3" >> arquivo4.txt
$ git add .
$ git commit -m "Editado v3"

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo "Editado v4" >> arquivo4.txt
$ git add .
$ git commit -m "Editado v4"
```

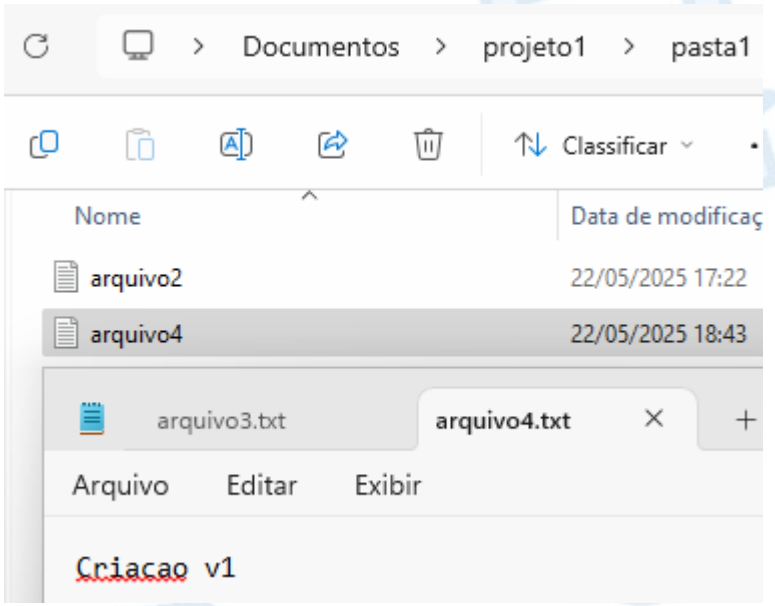


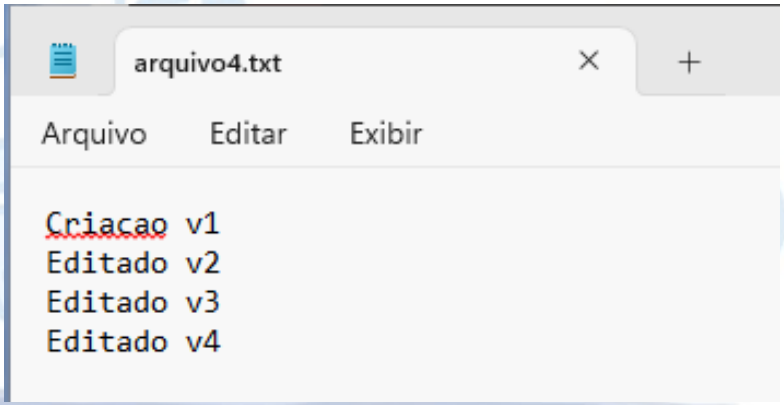


Trabalhando com arquivos:
Show

Show do último (mais recente) commit do arquivo.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo4.txt
1dc6224 (HEAD -> main) Editado v4
ff740ce Editado v3
2b202fd Editado v2
2f8fa3a Criacao v1

Fulano@Win11 ~/Documents/projeto1 (main)
$ git show 1dc6
commit 1dc6224dccc42ccf7dc1efe5bfe42f332cba4587 (HEAD -> main)
Author: fulano<fulano@example.com>
Date:   Thu May 22 18:45:48 2025 -0300

    Editado v4

diff --git a/pasta1/arquivo4.txt b/pasta1/arquivo4.txt
index f822db1..aac0123 100644
--- a/pasta1/arquivo4.txt
+++ b/pasta1/arquivo4.txt
@@ -1,3 +1,4 @@
 Criacao v1
 Editado v2
 Editado v3
+Editado v4
```

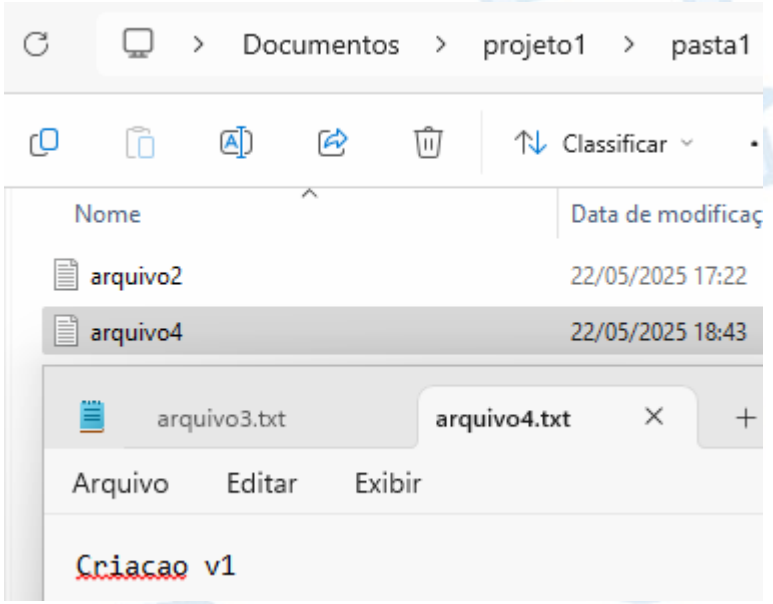


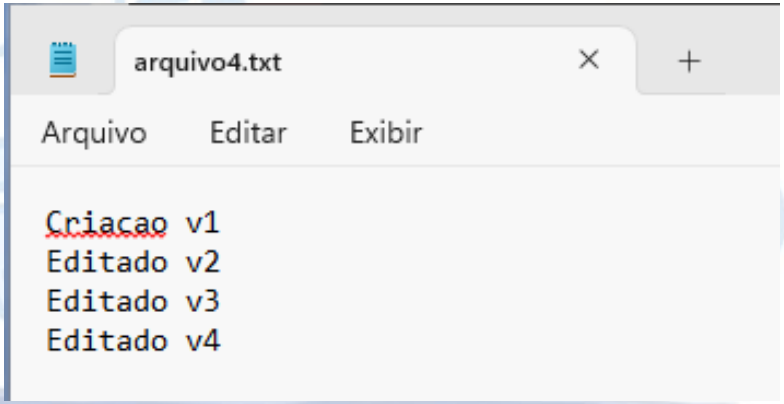


Trabalhando com arquivos:
Show

Show do segundo commit do arquivo.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo4.txt
1dc6224 (HEAD -> main) Editado v4
ff740ce Editado v3
2b202fd Editado v2
2f8fa3a Criacao v1

Fulano@Win11 ~/Documents/projeto1 (main)
$ git show 2b20
commit 2b202fdff4ea9fbe8e21066a1dbc9cfa7d75217a
Author: fulano<fulano@example.com>
Date:   Thu May 22 18:45:04 2025 -0300

    Editado v2

diff --git a/pasta1/arquivo4.txt b/pasta1/arquivo4.txt
index 82d69df..67c6373 100644
--- a/pasta1/arquivo4.txt
+++ b/pasta1/arquivo4.txt
@@ -1 +1,2 @@
 Criacao v1
+Editado v2
```





Trabalhando com arquivos:
Deletar

GitHub

Deletar arquivo commitado.

O arquivo pode ser removido via ambiente gráfico ou cli.

Após a operação é necessário fazer o commit novamente.

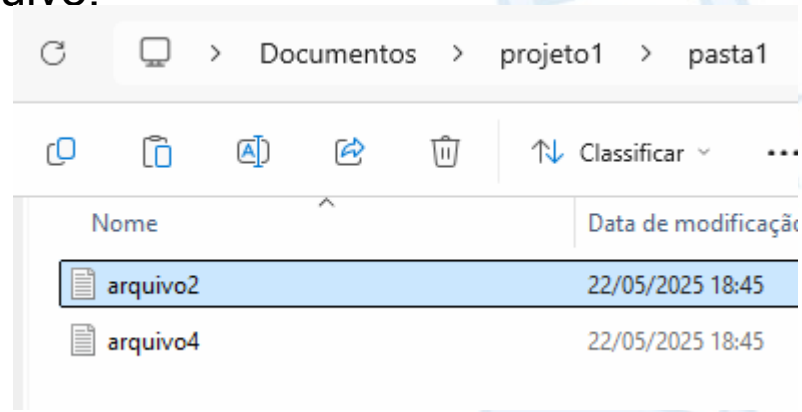
Veja as saídas dos comandos a seguir.

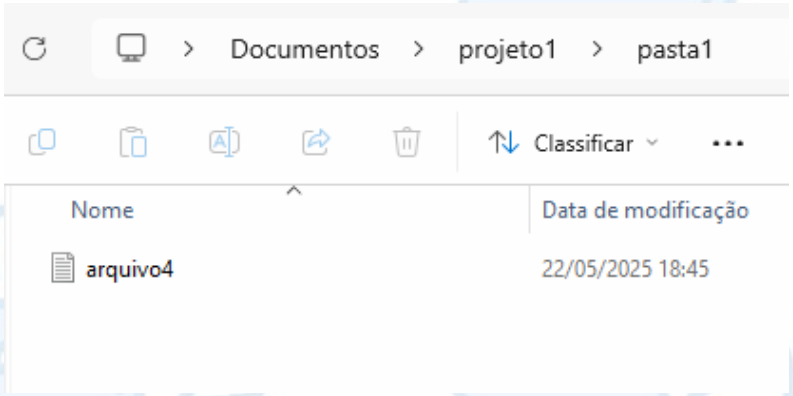
Trabalhando com arquivos:
Deletar

GitHub

Comando `git rm <arquivo>` para remover o arquivo.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
arquivo2.txt  arquivo4.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ git rm arquivo2.txt
rm 'pasta1/arquivo2.txt'

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 11 commits.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    arquivo2.txt

Fulano@Win11 ~/Documents/projeto1 (main)
$ git commit -m "Deletado arquivo2.txt"

Fulano@Win11 ~/Documents/projeto1 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 12 commits.
  (use "git push" to publish your local commits)
nothing to commit, working tree clean

Fulano@Win11 ~/Documents/projeto1 (main)
$ ls
arquivo2.txt
```





Trabalhando com arquivos:
Checkout

GitHub

Com o comando `git checkout` é possível, voltar a um commit específico para verificar os arquivos.

O comando também pode ser utilizado para navegar para uma branch específica.

`git checkout <commit>` - Volta ao estado anterior de um commit específico.

`git checkout <branch>` - Volta ao estado atual de uma branch.

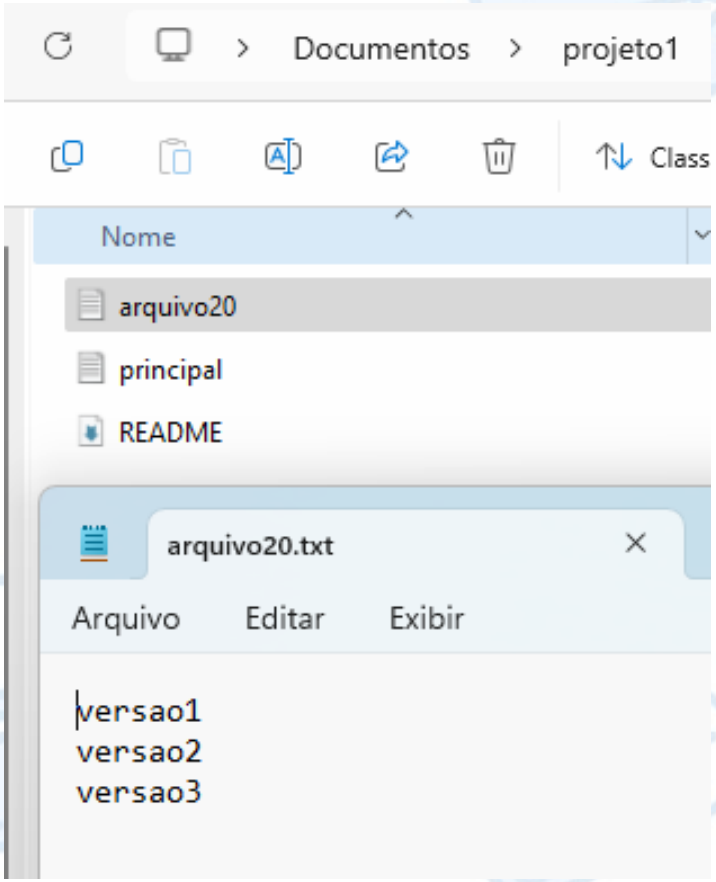
Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Checkout

Criando arquivo com 3 modificações e com um commit para cada modificação.

As modificações podem se via ambiente gráfico ou CLI.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ echo “versao1” >> aquivo20.txt
$ git add .
$ git commit -m “arquivo20 v1”

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo “versao2” >> aquivo20.txt
$ git add .
$ git commit -m “arquivo20 v2”

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo “versao3” >> aquivo20.txt
$ git add .
$ git commit -m “arquivo20 v3”

Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo20.txt
3e143dc (HEAD -> main) arquivo20 v3
7ba0ea7 arquivo20 v2
fbc676f arquivo20 v1
```



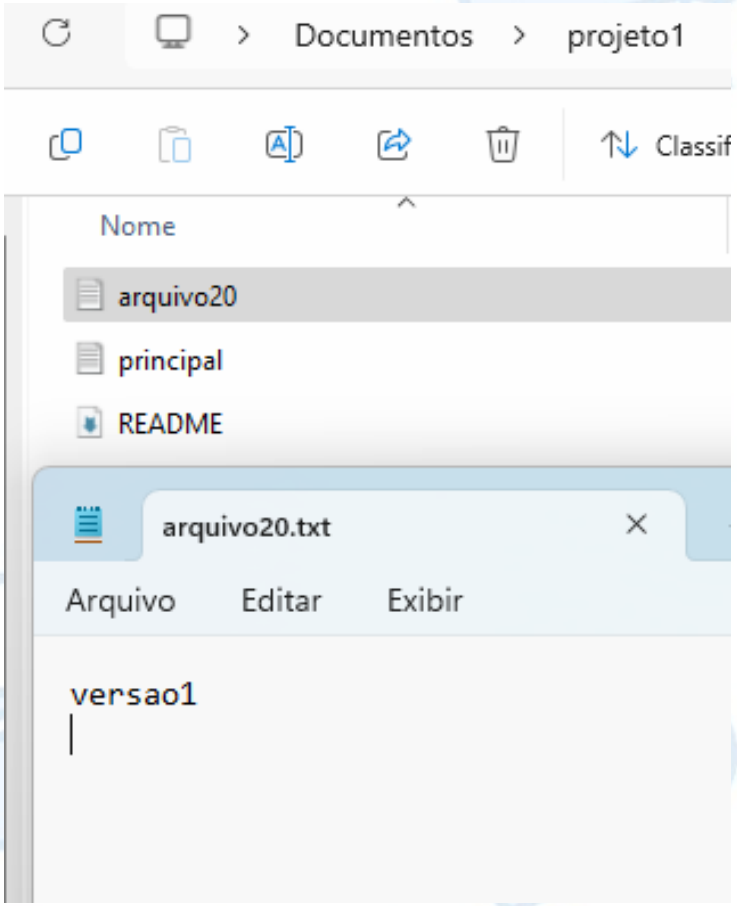
Trabalhando com arquivos:
Checkout

GitHub

Agora é possível voltar ao estado de um commit anterior.

De volta ao commit da versão 1.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo20.txt
3e143dc (HEAD -> main) arquivo20 v3
7ba0ea7 arquivo20 v2
fbc676f arquivo20 v1

Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout fbc676f

Note: switching to 'fbc676f'.
You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.
If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:
  git switch -c <new-branch-name>
Or undo this operation with:
  git switch -
Turn off this advice by setting config variable advice.detachedHead to false
HEAD is now at fbc676f arquivo20 v1
```



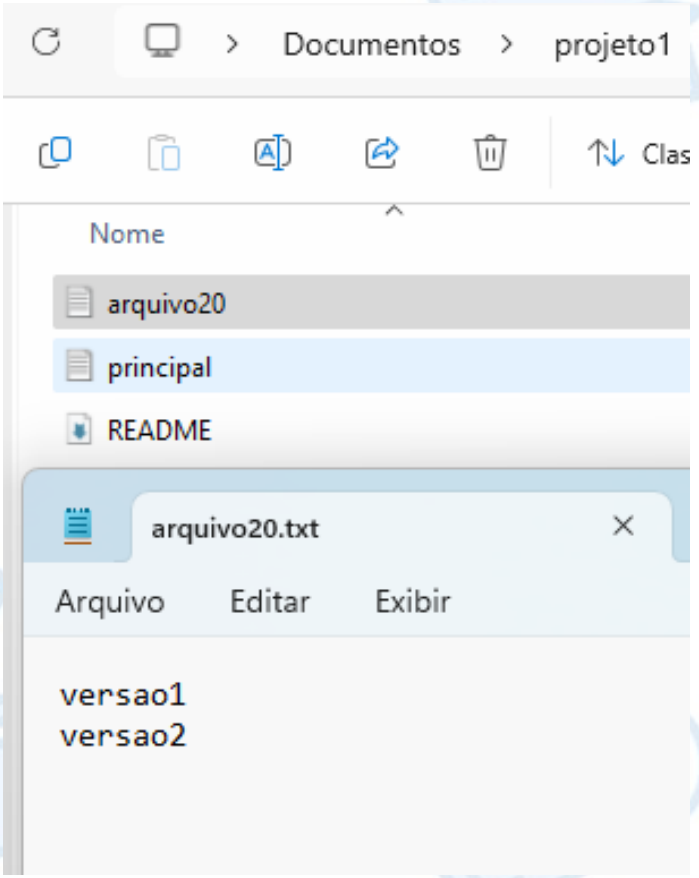
Trabalhando com arquivos:
Checkout

GitHub

Agora é possível voltar ao estado de um commit anterior.

De volta ao commit da versão 2.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo20.txt
3e143dc (HEAD -> main) arquivo20 v3
7ba0ea7 arquivo20 v2
fbc676f arquivo20 v1

Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout 7ba0ea7

Previous HEAD position was fbc676f arquivo20 v1
HEAD is now at 7ba0ea7 arquivo20 v2
```

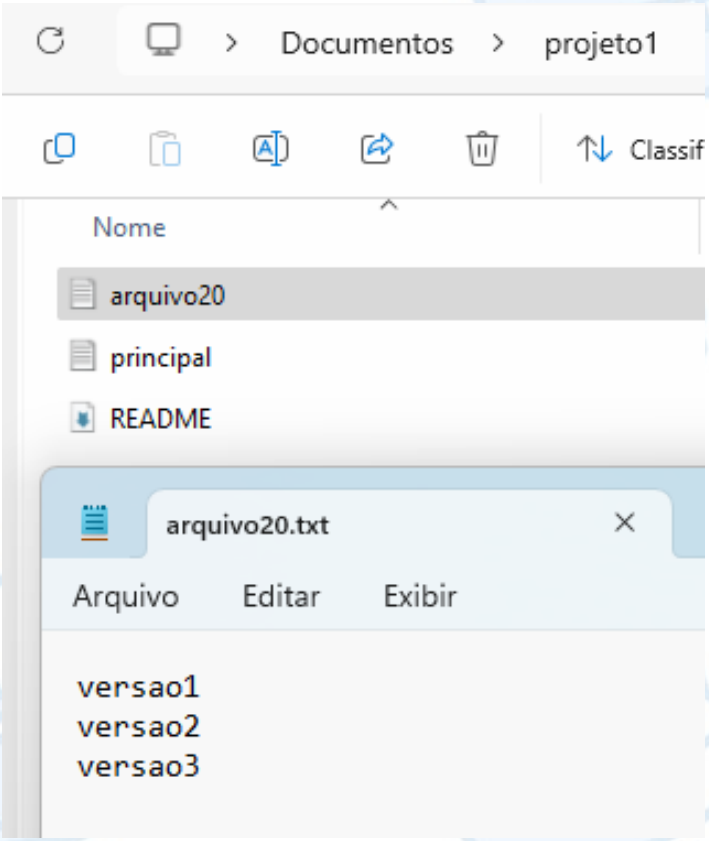


Trabalhando com arquivos:
Checkout

Voltar ao estado atual.

De volta ao estado mais recente da branch.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git checkout main

Previous HEAD position was 7ba0ea7 arquivo20 v2
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 6 commits.
  (use "git push" to publish your local commits)
```



Trabalhando com arquivos:
Revert

GitHub

Com o comando `git revert` é possível, reverter as alterações de um commit específico.

`git revert HEAD`	- Reverte o commit mais recente.
`git revert <commit>`	- Reverte o commit especificado.
`git revert --abort`	- Aborta uma reversão.
`git revert --continue`	- Continua uma reversão.

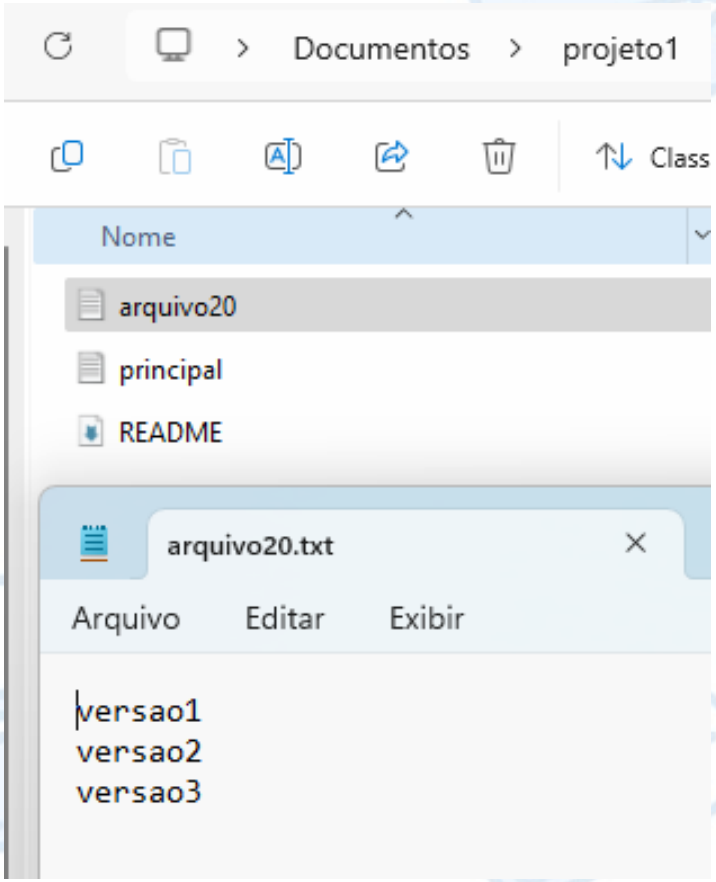
Veja as saídas dos comandos a seguir.

Trabalhando com arquivos:
Revert

Criando arquivo com 3 modificações e com um commit para cada modificação.

As modificações podem se via ambiente gráfico ou CLI.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ echo “versao1” >> aquivo20.txt
$ git add .
$ git commit -m “arquivo20 v1”

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo “versao2” >> aquivo20.txt
$ git add .
$ git commit -m “arquivo20 v2”

Fulano@Win11 ~/Documents/projeto1 (main)
$ echo “versao3” >> aquivo20.txt
$ git add .
$ git commit -m “arquivo20 v3”

Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo20.txt
3e143dc (HEAD -> main) arquivo20 v3
7ba0ea7 arquivo20 v2
fbc676f arquivo20 v1
```



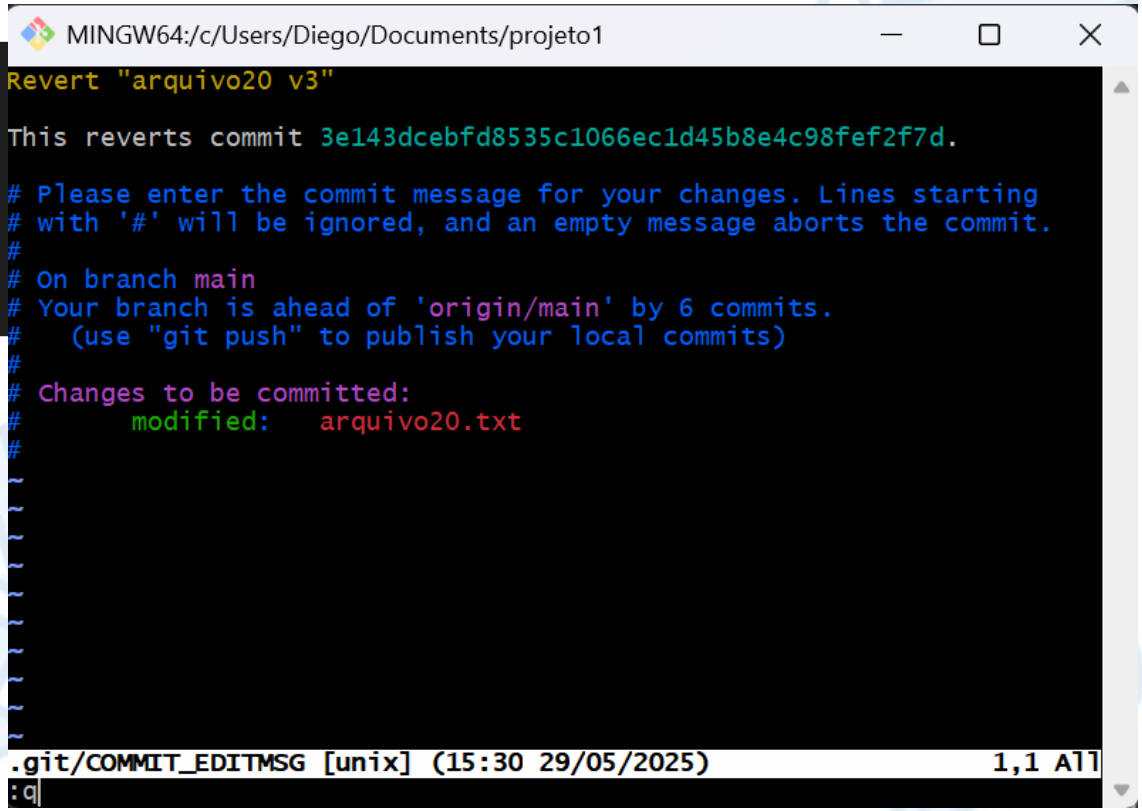
Trabalhando com arquivos:
Revert

Revertendo o último commit.

A reversão do último commit pode ser realizada com facilidade e normalmente não gera conflitos.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo20.txt
3e143dc (HEAD -> main) arquivo20 v3
7ba0ea7 arquivo20 v2
fbc676f arquivo20 v1

Fulano@Win11 ~/Documents/projeto1 (main)
git revert head
```

Será aberto um arquivo.
Basta digitar **:q**
para sair.

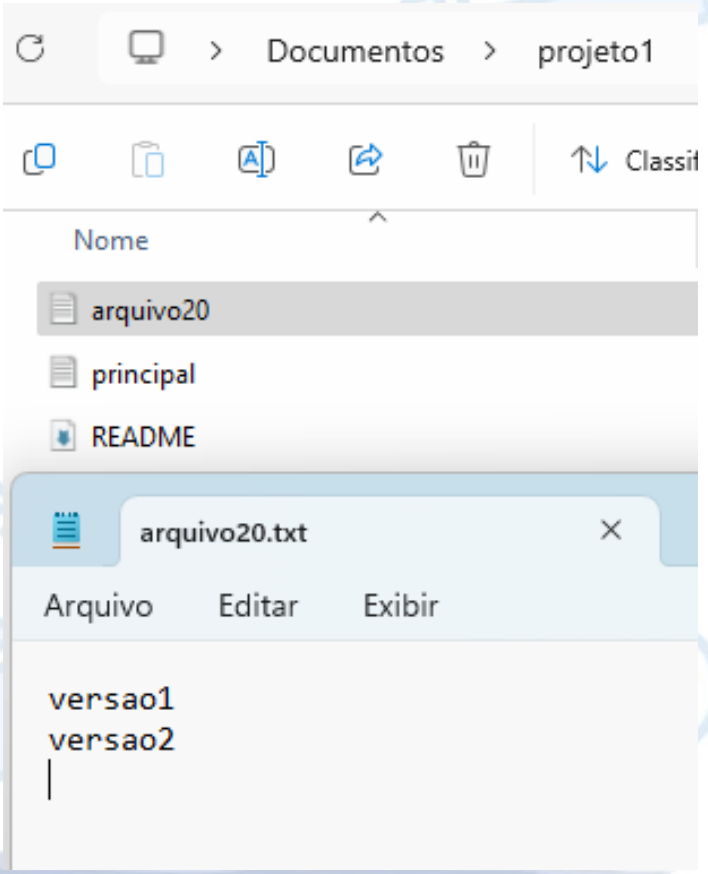


Trabalhando com arquivos:
Revert

A alteração feita no último commit foi desfeita.

Um novo commit foi criado com a observação de reversão.

```
Fulano@Win11 ~/Documents/projeto1 (main)
$ git log --oneline arquivo20.txt
45e6914 (HEAD -> main) Revert "arquivo20 v3"
3e143dc arquivo20 v3
7ba0ea7 arquivo20 v2
fbc676f arquivo20 v1
```



Trabalhando com arquivos:
Revert

GitHub

Com o comando `git revert <commit>` é possível desfazer as alterações de um commit específico mais antigo. Porém, isso deve ser evitado, por gerar conflitos. Neste caso, quando o processo de reversão é iniciado e um conflito é encontrado, o desenvolvedor tem a opção de:

Abortar a reversão: `git revert --abort`

Resolver o conflito e prosseguir com a reversão: `git revert --continue`

GitHub

Pronto já podemos

trabalhar com os arquivos.