



Primeiros Passos com Git e GitHub

Criando e Clonando Repositórios

- Clonando um Repositório Existente

Para obter uma copia de um diretório:

```
$ git clone URL
```

Para mudar o nome da pasta do diretório que voce quer clonar use (opcional):

```
$ git clone url clonado
```

- Transformar um diretório local que não está sob controle de versão, num repositório GIT.

Primeiro criamos uma pasta pelo terminal dentro de uma pasta, usamos o comando:

```
$ mkdir repo-local
```

Para entrar dentro dessa pasta usamos o comando:

```
$ cd repo-local/
```

Para transformar essa pasta em um repositório usamos o comando:

```
$ git init
```

Sempre fazer a inicialização do git

Para saber as configurações de um repositório é necessário abrir o bash na pasta do repositório e usar os comandos:

```
$ cd .git
```

```
$ cat config
```

Quando criamos um repositório localmente ele não está vinculado a um repositório remoto .Para saber se o repositório está vinculado remotamente

```
$ git remot -v
```

Para adicionar um repositório local a um repo remoto se necessário volte para main usando o comando

```
$ cd ..
```

O que começar com
"." (.git) é uma pasta
oculta.

O Git não reconhece
pasta ou diretórios
vazios. Use .gitkeep
para reconhecer
arquivos vazios

O diretório .git é
responsável pelo
controle de
versionamento do
código.

Para adicionar use:

```
$ git remote add origin https://github.com/angelicaccampos/remoto.git
```

Salvando Alterações no Repositório Local

Para ver a árvore de trabalho:

```
$ git status
```

Para adicionar um read.me

```
$ touch README.md
```

Agora precisamos adicionar o arquivo a área de preparação

```
$ git add README.md
```

Criei o commit usando

```
$ git commit -m"commit inicial"
```

Para saber todas as informações dos commit

```
$ git log
```

Alguns arquivos não devem receber commit, para ignorados:

```
$ echo resumos/ > .gitignore
```

Para remover a pasta do git ignore:

```
$ echo > .gitignore
```


Se você quiser adicionar vários arquivos:

```
$ git add .
```

Este site eu usei para escrever o README.md do meu repositório, para inserir o markdown escrito basta copiar o código fonte e colar no bloco de notas que você abriu do arquivo README.md e salvar

readme.so

Use
readme.so'
s


 <https://readme.so/p/t/editor>



Use o guia do GitHub para entender o markdown

Guia de início rápido de

Conheça os recursos avançados do README para seu perfil do

 <https://docs.github.com/en/get-started/writing-on-github>

Após fazer o push do readme é possível editá-lo no repositório remoto

Os arquivos podem ser editados com md no editor do próprio repositório no GitHub, essas edições serão salvas como commit

O GitHub tem o vscode online, basta clicar "." para abrir e editar os arquivos do repositório, é preciso estar logado na sua conta. Faça o commit direto do vscode

Desfazendo Alterações no Repositório Local

Como remover o versionamento

```
$ rm -rf .git
```

Como alterar a ultima mensagem do ultimo commit

```
$ git commit --amend -m"add gitignore"
```

Descartar mudanças de um arquivo:

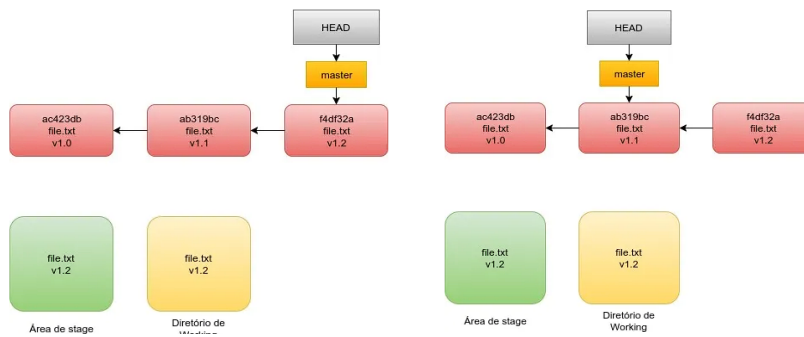
```
$ git restore ARQUIVO
```

Como "juntar" um commit anterior ao posterior

reset soft

```
$ git reset --soft HEAD~
```

As alterações devem ser feitas antes de serem enviadas para o repositório remoto para evitar conflitos



```
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github (main)
$ git top
commit e2f823741513350173c17f98c9e474bb2a9a76b (HEAD -> main)
Author: angelicacampas <angelicadacostacampas@gmail.com>
Date: Wed Jan 10 15:56:38 2024 -0300

    add aula 1

commit 1be0d6ff3ff5829cf72ada1f65cf47e4ef2d15f
Author: angelicacampas <angelicadacostacampas@gmail.com>
Date: Wed Jan 10 15:54:54 2024 -0300

    commit inicial
```

```
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github (main)
$ git reset --soft 1be0d6ff3ff5829cf72ada1f65cf47e4ef2d15f
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   resumos/aula.md

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

angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github (main)
$ git top
commit 1be0d6ff3ff5829cf72ada1f65cf47e4ef2d15f (HEAD -> main)
Author: angelicacampas <angelicadacostacampas@gmail.com>
Date: Wed Jan 10 15:54:54 2024 -0300

    commit inicial
```

O ultimo commit some

reset mixed

```
$ git reset --mixed HEAD~
```

reset hard

```
$ git reset --hard HEAD~
```

Como ter um histórico detalhado das alterações:

```
$ git reflog
```

Criar vários arquivos de uma vez:

```
$ touch resumos/aula01.md resumos/aula02.md
```

Como apagar um arquivo da area de preparação

```
$ git restore --staged resumos/aula02.md
```

Enviando e Baixando Alterações com o Repositório Remoto

Para enviar os arquivos do repositório local para o remoto após fazer a conexão:

```
$ git push -u origin main
```

Para salvar as alterações localmente feitas no repositório remoto ou pelo vscode online:

```
$ git pull
```

As
mudanças/commits
feitas de forma
remota não são
salvas localmente

Trabalhando com Branches - Criando, Mesclando, Deletando e Tratando Conflitos

Como criar uma branch de teste, após criar o arquivo, adiciona-lo e fazer o commit dele:

```
$ git checkout -b teste
```

```
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/branches/trabachando-branches (main)
$ git checkout -b teste
Switched to a new branch 'teste'

angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/branches/trabachando-branches (teste)
$ git log
commit 3c5427744c6484dd02960e1433408113386b1d3c (HEAD -> teste, main)
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Thu Jan 11 12:10:03 2024 -0300

    commit 3

commit f8248996b538c0446bd71ced8853fffd2512a6fc3
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Thu Jan 11 12:04:43 2024 -0300

    commit 1

commit e79460785bd1be50abf7f61aee7cde45851aff6e
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Wed Jan 10 21:37:40 2024 -0300
```

```
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/branches/trabachando-branches (teste)
$ echo "#commit-4-branchteste" > commit-4-branchteste.txt
$ git add .
warning: in the working copy of 'commit-4-branchteste.txt', LF will be replaced by CRLF the next time Git touches it
$ git commit -m "commit 4"
[teste c853bb0] commit 4
1 file changed, 1 insertion(+)
create mode 100644 commit-4-branchteste.txt

angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/branches/trabachando-branches (teste)
$ git log
commit c853bb0f49717bdf25e378e7a5ab6e89d632be0a (HEAD -> teste)
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Thu Jan 11 12:18:44 2024 -0300

    commit 4

commit 3c5427744c6484dd02960e1433408113386b1d3c (main)
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Thu Jan 11 12:10:03 2024 -0300

    commit 3
```

Agora a branch teste está apontando para o commit 4 e a main está apontando para o commit 3

Como mudar de branch:

```
$ git checkout main
```

Para ver os últimos commits de cada branch

```
$ git branch -v
```

Como mesclar a branch main e outra branch:

```
$ git merge teste
```

Visualize todas as branches:

Branch = ramo, é uma ramificação do projeto.

Usada para fazer testes sem afetar a branch principal a main



```
$ echo "#commit-4-branchteste" > commit-4-branchteste.txt
$ git add .
warning: in the working copy of 'commit-4-branchteste.txt', LF will be replaced by CRLF the next time Git touches it
$ git commit -m "commit 4"
[teste c853bb0] commit 4
1 file changed, 1 insertion(+)
create mode 100644 commit-4-branchteste.txt
$ git log
commit c853bb0f49717bdf25e378e7a5ab6e89d632be0a (HEAD -> teste)
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Thu Jan 11 12:18:44 2024 -0300

    commit 4

commit 3c5427744c6484dd02960e1433408113386b1d3c (main)
Author: angelicacampes <angelicadacostacampes@gmail.com>
Date: Thu Jan 11 12:10:03 2024 -0300

    commit 3
```

```
$ git checkout main
Switched to branch 'main'
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/branches/trabachando-branches (main)
```

As branches funcionam de forma independente, então as alterações se mantêm dentro da branch que foram criadas

Pesquisar as convenções para nomear as branches

```
$ git branch
```

Excluindo branches:

```
$ git branch -d teste
```

- **Possíveis conflitos de merge (comuns quando se trabalha em equipe)**

Esses conflitos acontecem quando temos alterações concorrentes. Quando tem mudanças de várias pessoas na mesma linha do código, o git entende qual a alteração deve ser mantida.

Quando esse tipo de conflito acontece o git faz você escolher qual mudança você quer que seja salva e manda uma mensagem indicando um conflito de merging

e as mensagens dos commits

As branches criadas localmente também devem ser enviadas para o repositório remoto

Conflito de merge: ocorrem quando alterações concorrentes são feitas na mesma linha de um arquivo ou quando uma pessoa edita um arquivo e outra pessoa exclui o mesmo arquivo.

Trabalhando com Branches - Comandos Úteis no Dia a Dia

Se você criar um commit remotamente e queira salvá-lo localmente sem que seja mesclado com o local

```
$ git fetch origin main
```

Para ver a diferença de conteúdo você pode usar esse comando para checar o commit feito remotamente

```
$ git diff main origin/main
```

Para apenas baixar esse commit para o local:

```
$ git merge origin/main
```

Clonar apenas uma branch

```
$ git clone https://github.com/angelicaccampos/Estudos-de-git-e-github
```

Para entrar na nova branch:

git pull = git fetch + git merge

```
$ cd Estudos-de-git-e-github
```

```
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github (main)
$ cd Estudos-de-git-e-github
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github/Estudos-de-git-e-github (teste)
$
```

Quando deletamos um arquivo essa ação é mostrada no status

```
angel@desktop1000 MINGW64 ~/OneDrive/Área de Trabalho/estudos-git-github/Estudos-de-git-e-github (teste)
$ git status
On branch teste
Your branch is up to date with 'origin/teste'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    resumos/aula01.md

no changes added to commit (use "git add" and/or "git commit -a")
```

Para que essa modificação não seja levada para uma branch criada posteriormente usamos esse código

```
$ git stash
```

Para listar essas modificações arquivadas:

```
$ git stash list
```

```
$ git stash apply
```