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
```

> *Oque começar com "." (.git) é uma pasta oculta.*

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

- **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
```

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

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 ..
```

Para adicionar use:

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

Salvando Alterações no Repositório Local

Para ver a arvore de trabalho:

```
$ git status
```

Para adicionar um read.me

```
$ touch README.md
```

Agora precisamos adicionar o arquivo a area 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 voce 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 voce abriu do arquivo README.md e salvar*

readme.so
Use readme.so's
https://readme.so/pt/editor

> *Use o guia do GitHub para entender o markdown*

Guia de início rápido de
Conheça os recursos avan
LEIAME para seu perfil do
https://docs.github.com
started-with-writing-and-for
on-github

> *Após fazer o push do readme é possível edita-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 gitigonore"
```

Descartar mudanças de um arquivo:

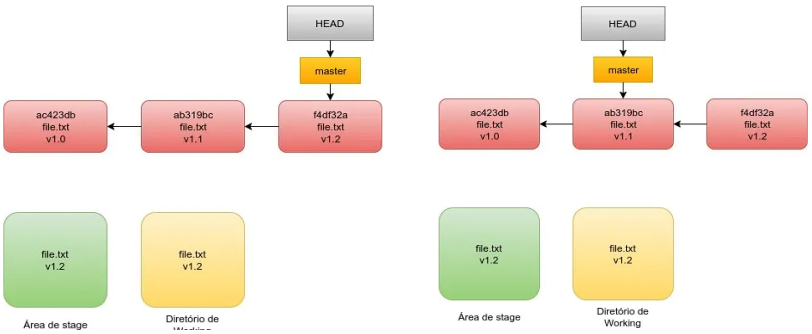
```
$ git restore ARQUIVO
```

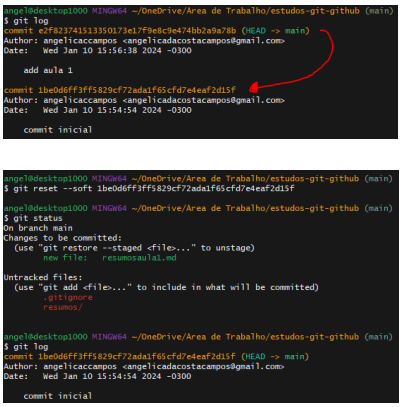
Como “juntar” um commit anterior ao posterior

reset soft

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

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





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
```

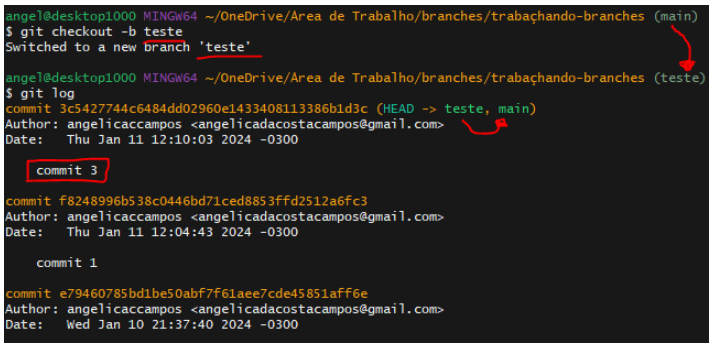
> *As mudanças/commits feitas de forma remota não são salvas localmente*

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

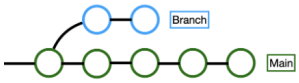
Como criar uma branch de teste, após criar o arquivo, adiciona-lo e fazer o commit dele:

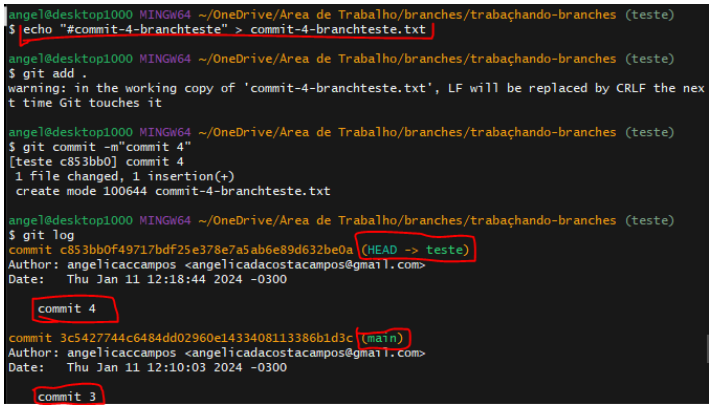
```
$ git checkout -b teste
```

> *Branch = ramo, é uma ramificação do projeto.*

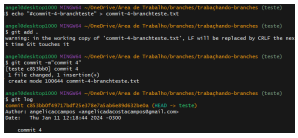


> *Usada para fazer testes sem afetar a branch principal a main*





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
```

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

Como mesclar a branch main e outra branch:

```
$ git merge teste
```

Visualize todas as branches:

> *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 acontece quando temos alterações concorrentes. Quando tem mudanças de varias pessoas na mesma linha do código, o git entende qual a alteração deve ser mantida.

Quando esse tipo de conflito acontece o git faz voce escolher qual mudança voce 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 voce criar um commit remotamente e queira salva-lo localmente sem que seja mesclado com o local

> *git pull = git fetch + git merge*

```
$ git fetch origin main
```

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

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

Para apenas baixar esse commit para o local:

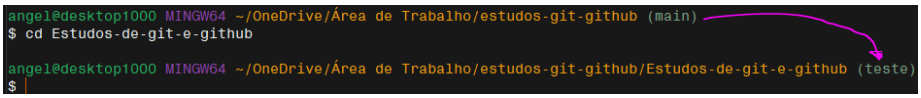
```
$ git merge origin/main
```

Clonar apenas uma branch

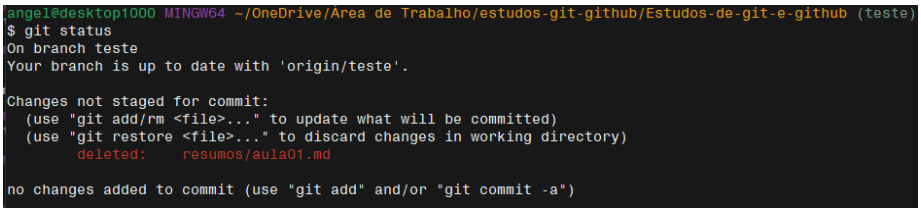
```
$ git clone https://github.com/angelicaccampos/Estudos-de-git-e-github
```

Para entrar na nova branch:

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



Quando deletamos um arquivo essa ação é mostrada no status



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

```
$ git stash
```

Para listar essas modificações arquivadas:

```
$ git stash list
```

```
$ git stash apply
```