



Comandos Git

→ Ajuda

- ☐ git help -> comando geral

→ Comando específico

- ☐ git help add
- ☐ git help commit
- ☐ git help <qualquer_comando_git>

→ Setar usuário e email

- ☐ git config --global user.name "nome do usuário"
- ☐ git config --global user.email email@email.com

→ Remover todas as linhas que referenciam o usuário e email

- ☐ git config --global --unset user.name "nome do usuário"
- ☐ git config --global --unset user.email email@email.com

→ Lista configurações

- ☐ git config --list

→ Criando novo repositório

- ☐ git init

→ Verificar o estado dos arquivos/diretórios

- ☐ git status (mostra qual a situação do arquivos no seu repositório)

→ Adicionando arquivo

- ☐ git add meu_diretorio (arquivo específico)
- ☐ git add . / git add --all (todos os arquivos)

→ Comitar arquivo/diretório



- ❑ `git commit arquivo -m "mensagem de commit"`

→ **Remover arquivo/diretório**

- ❑ `git rm arquivo` (remove arquivo)
- ❑ `git rm -r diretório` (remove diretório/pasta)

→ **Visualizar histórico**

- ❑ `git log` (exibir histórico)
- ❑ `git log -- <caminho_do_arquivo>` (exibir histórico de um arquivo específico)
- ❑ `git log --author=usuário` (exibir histórico de um determinado)

Desfazendo operações

→ **Desfazendo alteração local no seu diretório de trabalho local**

- ❑ `git checkout -- arquivo` (Este comando só deve ser utilizado enquanto o arquivo ainda não foi adicionado na área de trabalho temporária)

→ **Desfazendo alteração local na área de trabalho temporária(staged área)**

- ❑ `git reset HEAD arquivo` (Este comando deve ser utilizado quando o arquivo já foi adicionado na área temporária)
- ❑ `Unstaged changes after reset:M` arquivo (se o resultado abaixo for exibido, o comando `reset` não alterou o diretório de trabalho)
- ❑ `git checkout arquivo` (a alteração do diretório pode ser realizada através deste comando)

Repositório Remoto

→ **Exibir os repositórios remotos** (Para sabermos para onde estão sendo enviadas nossas alterações ou de onde estamos baixando as coisas)

- ❑ `git remote`
- ❑ `git remote -v`
- ❑ `git remote add origin git@github.com:meunome/arquivo-git.git` (vincular repositório local com um repositório remoto)



- ❑ `git remote show origin` (exibir informações dos repositórios remotos)
- ❑ `git remote rename origin arquivo-git` (renomear um repositório remoto)
- ❑ `git remote rm arquivo-git` (desvincular um repositório remoto)
- ❑ `git push -u origin master` (o primeiro push no repositório deve conter o nome do mesmo e a sua branch)
- ❑ `git push` (os demais pushes não precisam de outras informações)

→ **Atualizar repositório local de acordo com o repositório remoto**

- ❑ `git pull` (atualizar os arquivos em relação a branch atual)
- ❑ `git fetch` (buscar as alterações, mas não aplicá-las na branch atual)

→ **Clonar um repositório remoto já existente**

- ❑ `git clone git@github.com:meunome/arquivo-git.git`

Branches

A master é a branch principal do GIT

O HEAD é um ponteiro especial que indica qual é o branch atual. Por padrão, o HEAD aponta para o branch principal, o master.

- ❑ `git branch novaBranch_nome` (criando uma nova branch)
- ❑ `git checkout novaBranch_nome` (trocando para uma branch existente). Neste caso, o ponteiro principal HEAD está apontando para o branch chamada novaBranch_nome.
- ❑ `git checkout -b novaBranch_nome` (cria uma nova branch e troca pra ela)
- ❑ `git checkout master` (voltar para a branch principal(master))
- ❑ `git merge novaBranch_nome` (resolve o merge entre duas branches) - para realizar o merge, é necessário estar no branch que deverá receber as alterações.
- ❑ `git branch -d novaBranch_nome` (apagando uma branch)
- ❑ `git branch` (listar branches)
- ❑ `git branch -v` (listar branches com informações dos últimos commits)
- ❑ `git branch --merged` (listar branches já foram fundidos(merged) com a master)
- ❑ `git branch --no-merged` (listar branches que não foram fundidos(merged) com a master)
- ❑ `git pull origin nomeBranch` (puxando arquivos de uma branch já existente)
- ❑ `git push origin novaBranch_nome` (criando um branch remoto com o mesmo nome)
- ❑ `git merge --abort` ou `git reset --merge` (quando você tem problemas com merge e deseja desfazer o merge)



- ❑ git reset HEAD (quando você deseja voltar commits, caso você queira voltar mais de um commit, coloque o número dos commits após o HEAD -> exemplo HEAD~2)

➔ **Reescrevendo o histórico**

- ❑ git commit --amend -m "Minha nova mensagem" (alterando mensagens do commit)

Comandos para o Terminal

➔ ctrl + l ou clear

- ❑ limpa tela

➔ mkdir nome_da_pasta

- ❑ cria uma pasta

➔ cd

- ❑ entrar na pasta

➔ cd ..

- ❑ sair da pasta

➔ ls

- ❑ ver o que tem dentro da pasta

➔ rm

- ❑ deleta arquivo

➔ rm -r nome

- ❑ remover

➔ rm -rf nome

- ❑ remove tudo