



**Comandos Git**

* **Ajuda**
* git help
* **Ajuda em comandos específicos**
* git help add
* git help commit
* git help <qualquer\_comando\_git>
* **Defina o nome de usuário e e-mail**
  + git config --global user.name "nome do usuario"
  + git config --global user.email email@email.com
* **Exclua todos os registros que se referem ao usuário e ao e-mail**
* git config --global --unset user.name "nome do usuario"
* git config --global --unset user.email email@email.com
* **Ver configuração do Git**
* git config --list
* **Crie um novo repositório**
* git init
* **Verifique o status dos arquivos/diretórios**
* git status (mostra o status dos arquivos em seu repositório)
* **Adicione um arquivo**
* git add nome\_arquivo\_diretorio (arquivo específico)
* git add . / git add --all (todos os arquivos)
* **Confirme um arquivo/diretório**
* git commit nome\_arquivo -m "enviar mensagem"
* **Remova um arquivo ou diretório**
* git rm arquivo
* git rm -r diretorio (remove o diretório e os arquivos que ele contém)
* **Veja o histórico de atividades**
* git log (mostra o histórico)
* git log -- <histórico do arquivo> (mostra o histórico de um arquivo específico)
* git log --author=usuario (mostra o histórico de um determinado usuário)

**Desfazer operações**

* **Desfazendo a mudança local em seu diretório de trabalho local**
* git checkout -- arquivo (só deve ser usado enquanto o arquivo ainda não foi adicionado ao espaço de trabalho temporário)
* **Desfazendo mudança local na área de trabalho temporária**
* git reset HEAD arquivo (deve ser usado quando o arquivo já foi adicionado na área temporária)

“*Unstaged changes after reset:M arquivo*” (se a seguinte saída for exibida, o comando de redefinição não alterou o diretório de trabalho)

* git checkout nome\_arquivo (permite que você mude o diretório)

**Repositório Remoto**

* **Veja os repositórios remotos (para saber para onde as alterações são enviadas ou de onde as baixamos)**
* git remote
* git remote -v
* git remote add origin git@github.com:meunome/arquivo-git.git (vincula o repositório local a um repositório remoto)
* git remote show origin (permite que você veja informações de repositórios remotos)
* git remote rename origin nome\_novo (renomeia um repositório remoto)
* git remote rm nome\_git (desvincula um repositório remoto)
* git push -u origin main (o envio para o repositório deve conter seu nome e branch)
* **Atualize o repositório local com base no repositório remoto**
* git pull (atualiza arquivos em relação à branch atual)
* git fetch (obter as alterações, mas não aplicá-las à branch atual)
* **Clone um repositório remoto existente**
* git clone git@github.com:meunome/arquivo-git.git

**Branches**

A main (anteriormente, chamada de master) é a branch principal do Git.

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

* git branch nova\_branch\_nome (cria uma nova branch)
* git checkout nova\_branch\_nome (mudar para uma branch existente) - Neste caso, o ponteiro HEAD principal está apontando para a branch nova\_branch\_nome.
* git checkout -b nova\_branch\_nome (crie uma nova branch e aponte para ela)
* git checkout main (volta para a branch principal main)
* git merge nova\_branch\_nome (resolve a união (*merge*) entre as branches) - Para mesclar, você deve estar na branch que deve receber as alterações.
* git branch -d nova\_branch\_nome (apagando uma branch)
* git branch (lista branches)
* git branch -v (lista branches com informações dos últimos commits)
* git branch --merged (lista branches que já estão unidas (*merged*) com a main)
* git branch --no-merged (lista branches que não estão unidas (*merged*) com a main)
* git pull origin nomeBranch (extrai arquivos de uma branch existente)
* git push origin nova\_branch\_nome (cria uma branch remota com o mesmo nome)
* git merge --abort o git reset --merge (quando temos problema com a união (*merge*) e queremos desfazê-lo)
* git reset HEAD (quando queremos voltar a um commit anterior. Se quisermos voltar a mais de um commit, devemos colocar o número de commits após HEAD. Exemplo: HEAD~2)
* **Reescrevendo o histórico**
* git commit --amend -m "Minha nova mensagem" (alterar mensagens de confirmação)

**Comandos de terminal**

* **crtl+l ou clear**
* Limpe o console
* **mkdir nome\_da\_pasta**
* Crie uma pasta
* **cd**
* Entre em uma pasta
* **cd ..**
* Sair de uma pasta
* **ls**
* Veja o que está dentro da pasta
* **rm nome**
* Apaga o arquivo
* **rm -r nome**
* Exclui o diretório e todos os arquivos que ele contém
* **rm -rf nome**
* Exclui à força o diretório e todos os arquivos que ele contém