## Iniciando um Repositório

Todo repositørio Git armazena as informações dentro de uma pasta oculta chamada “/.git”. Para que os arquivos de uma pasta possam ser versionados pelo Git, é preciso iniciar o repositório. Basta executar o comando abaixo:

Copiar

$ git init

## 2 - Apagando um repositório

Há momentos em que não queremos apagar nossos arquivos, mas queremos remover as informações sobre aquele repositório criado com o $ git init (talvez criar um repositório novo com os mesmos arquivos). Para isso não usamos o Git. Lembre-se que um repositório Git armazena as informações dentro de uma pasta oculta chamada /.git. Então basta apagar esta pasta oculta que o seu atual diretório deixará de ser um repositório.

Copiar

$ rm -rf .git

## 3 -Listando Arquivos Modificados

Esse comando indica o estado do seu repositório. Em outras palavras, ele vai listar todos os arquivos que foram modificados desde o seu último commit.

Copiar

$ git status

## 4 - Desfazendo Alterações

Poderíamos ter um post inteiro apenas falando sobre como desfazer alterações no Git, já que há vários cenários possíveis. Mas vamos resumir os principais:

### 4.1 - Arquivos não monitorados

Se o arquivo foi modificado mas você ainda não executou git add, um simples git checkout removerá as alterações, deixando o arquivo como ele estava no último commit. Passe o nome do arquivo a ter as alterações desfeitas ou . para desfazer as alterações em todos os arquivos modificados. Muito útil se você está apenas experimentando um código mas não quer que ele seja salvo.

Copiar

git checkout .

Esse comando não apaga novos arquivos. Para apagar novos arquivos que ainda não foram adicionados ao Stage, execute:

Copiar

git clean -df

### 4.2 - Removendo arquivos do Stage

Se você executou git add e quer desfazer, use o reset.

Copiar

git reset

Para desfazer as modificações, após o reset use o checkout ou clean mostrados anteriormente.

### 4.3 - Desfazendo o último commit

Caso você tenha feito alterações e já tenha chegado a realizar um *commit*, para desfazer é necessário usar o revert.

Copiar

git revert HEAD

Será criado um novo commit indicando que o último commit foi desfeito.

Esse comando apaga novos arquivos.

## 5 - Renomear Commit

Escreveu algo errado no último commit? Esse comando te permite renomear a mensagem do último commit feito:

Copiar

$ git commit --amend

## 6 - Branches

Sempre é bom não trabalhar apenas na main para evitar problemas e ter um projeto mais flexível.

## 6.1 - Listando Branches

Esse comando lista todas as branches presentes no repositório do seu computador.

Copiar

$ git branch

Caso você queira que ele liste também as branches que estão no repositório remoto, adicione -a:

Copiar

$ git branch -a

## 6.2 - Indo para outra branch

Para mudar para uma outra *branch* basta usar o comando checkout, passando o nome da *branch*.

Copiar

$ git checkout minha-branch

Se você adicionar -b, uma nova *branch* será criada.

Copiar

$ git checkout -b minha-nova-branch

## 6.3 - Excluindo branches

Para excluir uma branch local basta executar o comando branch com -d ou -D, passando o nome da branch que você quer apagar.

Copiar

git branch -d nome-da-branch

git branch -D nome-da-branch

A opção -d é mais segura, pois ela só apaga a branch se você já tiver feito merge ou enviado as alterações para seu repositório remoto, evitando perda de código.

A opção -D ignora o estado da sua branch, forçando a sua remoção.

Esse comando apaga apenas a branch local, não removendo branches remotas.

## 6.4 - Renomeando branches

Para renomear a sua atual branch local, execute o comando branch com a opção -m, passando o novo nome.

Copiar

git branch -m novo-nome-da-branch

Se você estiver em uma branch e quiser renomear outra, você deve passar primeiro o nome atual da branch que quer renomear:

Copiar

git branch -m nome-atual novo-nome