# Criando e Sincronizando um Repositório Git no Github.

**Autor: Enzo Lima (Site Pessoal, E-mail, Instagram)**



---

Neste texto, irei abordar o uso do Git e do Github, ferramentas essenciais para o controle de versionamento de código e colaboração em projetos que estão sendo desenvolvidos - independente da área.

## Primeiros Passos

O ponto de partida será a instalação do git. Para saber como fazer isso no seu sistema operacional, consulte o site oficial do projeto.

Para ter uma experiência mais intuitiva no aprendizado, recomendo ativar as seguintes funções do seu explorador de arquivos:

- Mostrar a extensão dos arquivos
- Exibir arquivos ocultos

Onde ativar essas funções varia de sistema operacional. Então, pesquise.

## Criando um repositório local

Um repositório local nada mais é do que uma pasta em seu computador que será sincronizada com o repositório da nuvem. Então o primeiro passo é criar uma pasta vazia em um local apropriado do seu desktop.

Com o git instalado, abra-o dentro da pasta em que você criou e digite o seguinte comando:

```
git init
```

Esse comando tornará sua pasta vazia em um diretório git! Caso tenha mostrado isso em seu terminal, quer dizer que deu certo:

```
Initialized empty Git repository in [DIRETÓRIO DO REPOSITÓRIO]
```

## Adicionando os arquivos no repositório

Um bom repositório não existe sem um `README.md`, nele é colocado a descrição do seu repositório. Então, crie um arquivo de texto com esse nome no diretório.

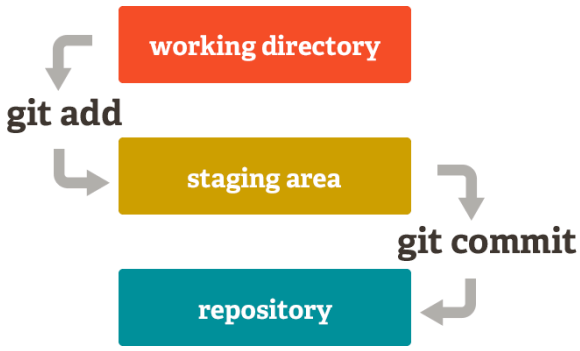
Temos o arquivo criado, mas ele ainda não foi adicionado no nosso repositório. Qualquer alteração antes de ser "commitada", deve ser adicionada. Então para adicionar de fato o arquivo ao repositório, digite:

```
git add README.md
```

Caso tenha mais de um arquivo novo ou modificado, use:

```
git add .
```

Explicando um pouco mais sobre o que aconteceu, o `git add` manda os arquivos para uma área de teste que oferece a oportunidade de organizar e planejar as alterações que serão incluídas no próximo commit.




Reprodução: Thaddeus Resource Center

Para consultar essa área de teste, digite `git status`. Ele mostrará o seguinte:

```
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   README.md
```

## Primeiro commit

Está tudo pronto para o primeiro commit! Mas o que é isso?

O commit é um registro de alterações em um repositório de código-fonte. Ou seja, qualquer alteração aplicada do seu repositório é chamada de commit.

Lembrando que todo commit precisa de um título, então para "commitar" e intitular o que foi adicionado lá, digite:

```
git commit -m "primeiro commit"
```

Use o `git status` novamente. Se mostrar a seguinte mensagem, podemos prosseguir:

```
On branch master
nothing to commit, working tree clean
```

## Alterando o nome da branch principal

Não sendo muito técnico, branch é um caminho independente do desenvolvimento dentro de um repositório. Inicialmente, o projeto tem apenas um branch, mas pode ser criado outros com o intuito de trabalhar em mudanças no código, sem afetar a linha principal do projeto.

Quando você cria um novo repositório, a primeira branch se chama `master`. Entretanto, é recomendado alterar o nome dela para `main`, por ser uma nomenclatura mais atualizada.

Essa alteração pode ser feita digitando em seu terminal:

```
git branch -M "main"
```

## Publicando nosso repositório local no Github

Para fazer a hospedagem do projeto, crie um repositório no Github clicando no botão `new` na aba de repositórios do seu perfil. Como já temos um `README.md`, não é preciso ativar a opção de adicionar esse arquivo.

Quando o repositório for criado, será mostrado um guia do próprio Github de como fazer a publicação.

É preciso conectar os dois repositórios (local e remoto) usando o seguinte comando:

```
git remote add origin https://github.com/usuario/nomedorepositorio.git
```

Finalizamos o processo com o comando `git push -u origin main` para "empurrar" os commits para o repositório remoto - será solicitado seu login no Github.

> OBS: No ambiente linux, ao invés da senha, você deve colocar um token que pertence a sua conta. Saiba mais aqui.

Com o login feito, ele começará a encaminhar seus arquivos locais para o repositório da plataforma! Estando concluída nossa primeira publicação.

## Criando, alterando e mesclando as branches

Criamos uma nova branch usando:

```
git checkout -b "segunda-branch"
```

A partir de agora, qualquer comando que você fizer, será destinada a nova branch. Portanto, o `git push` não terá `main` no final, e sim o nome da `branch` que você acabou de criar:

```
git push origin segunda-branch
```

Use `git checkout main` para voltar a `branch` principal.

Podemos mesclar uma `branch` na outra com:

```
git merge nome-da-branch-que-você-quer-mesclar
```

Lembrando que após mesclar, deve-se dar um `git push` para "subir" a mesclagem.

## Outros comandos

`git clone link-de-outro-repositorio` : Clona um repositório no Github para o seu computador.

`git pull` : Traz as alterações feitas no repositório remoto para o repositório local.

## Bônus

Vamos supor que você esteja trabalhando em um projeto mais simples no Github, talvez fazer todo esse processo pode ser um pouco cansativo. Por isso, recomendo o Github Desktop.

Com ele é possível fazer grande parte das funções do git de maneira gráfica com uma curva de aprendizado bastante simples.

---

## Conclusão

Espero que você tenha terminado esse material sabendo o básico desses recursos e com vontade de aprender mais sobre. Tenha a documentação do git como referência.

Se gostou, não deixe de compartilhar o material com aquele amigo dev. que com certeza irá se beneficiar do conteúdo.

Acha que alguma coisa ficou faltando aqui? Me mande um email.