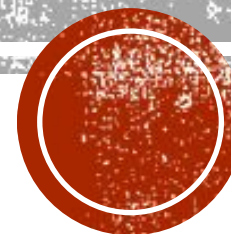


# GITHUB



Na história, era uma vez ...



Foi criado em 2005 por Linus Torvalds para o versionamento do kernel Linux após desentendimento com a empresa responsável pelo DVCS. Foi rapidamente adotado por outros projetos.



## Diferença entre Git X Github

Git é uma implementação de versionamento, enquanto GitHub é um servidor onde as informações versionadas são armazenadas. Consequentemente, as informações estão em um local mais seguro que seu próprio computador, que é sujeito a muito mais problemas.



## Por que aprender git?

- Ferramenta de controle de versão.
- Ótimo para trabalhar em equipe.
- Uma ótima forma de backup.
- Mantém o histórico de tudo.



# VAMOS PARA PRÁTICA

GitHub

Criando primeiro  
repositório



# Instalação

## **Linux**

sudo apt install git # Debian based

## **Mac**

brew install git

## **Windows**

Baixe o executável no site oficial e instale.



# CONFIGURAÇÕES

Antes de começarmos é necessário configurar algumas informações no git, como por exemplo o nome e o e-mail do usuário já que esses dados que estarão atrelados ao commit.

```
git config --global user.name  
git config --global user.email
```



# CRIANDO UM REPOSITÓRIO

Repositório é o local onde você armazenará e versionará o seu código.

Navegue até o diretório do seu projeto e digite:

```
# Inicia um repositório vazio git init
```

Quando você executar esse comando uma pasta `.git` será criada no diretório e essa pasta conterá todos os meta dados para gerenciar o seu repositório.





# COMANDOS PARA SE VIRAR NOS 30

git status  
git add  
git commit  
git push  
git pull  
git diff  
git log  
git clone  
git remote



# PRATICANDO AGORA

Passo a passo agora:

1. Escolha uma linguagem
2. Escolha sua dupla para essa aula prática.
3. Crie uma pasta local no seu pc.
4. Entre na sua conta github e crie o repositório com o nome dado igual a pasta do repositório local da sua máquina.



# Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

## Repository template

Start your repository with a template repository's contents.

No template ▾

Owner \*



Repository name \*

meu-primeiro-repositorio ✓

Great repository names **meu-primeiro-repositorio** is available. Inspiration? How about **musical-doodle**?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

## Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)


☐ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository



## Quick setup — if you've done this kind of thing before

 Set up in Desktop

or

HTTPS

SSH

`https://github.com/rafalup/meu-primeiro-repositorio.git`



Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

## ...or create a new repository on the command line

```
echo "# meu-primeiro-repositorio" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/rafalup/meu-primeiro-repositorio.git
git push -u origin main
```



## ...or push an existing repository from the command line

```
git remote add origin https://github.com/rafalup/meu-primeiro-repositorio.git
git branch -M main
git push -u origin main
```



## ...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code



# DOMINANDO OS PASSOS BÁSICOS

`git init`

`git remote add origin <url>`

`git add .`

`git commit -m "....."`

`git push --set-upstream origin master`



# FAZENDO PULL REQUEST

1 - Fork um repositório (de outro usuário) que você deseja contribuir:

[https://github.com/\[OUTRO-USUÁRIO\]/\[REPOSITÓRIO\]](https://github.com/[OUTRO-USUÁRIO]/[REPOSITÓRIO])

2 - Clone seu repositório 'forkado' com seu nome de usuário:

`git clone https://github.com/[SEU-USUÁRIO]/[REPOSITÓRIO]`

3 - Crie uma nova branch

`Cd [REPOSITORIO]`

`git branch nova-branch`



4 - Mude para a nova-branch usando o comando checkout:

```
git checkout nova-branch  
#Switched to branch 'nova-branch'
```

5 - Após fazer qualquer alteração no *seu* [REPOSITÓRIO] que você 'forkou'

```
git add -A  
git commit -m "alterei o arquivo x"
```

6 - Use o comando git push pra *empurrar* as alterações para seu repositório, a opção --set-upstream é porque utilizamos uma nova branch nova-branch.

```
git push --set-upstream origin nova-branch
```



7 - Vá no Github e clique em **Compare e Pull Request** , se quiser adicione um *comentário* e clique em **Create Pull Request**.

**git remote -v**

Para sincronize com o 'fork' use o comando abaixo para adicionar outros repositórios.

**git remote add upstream https://github.com/original-[USUÁRIO]/original-[REPOSITÓRIO].git**

8 - Para sincronizar com o 'fork' usaremos o comando git fetch para buscar as *branches* junto com seus respectivos *commits*, de novo usaremos a opção *upstream*.

**git fetch upstream**

**#From https://github.com/original-[USUÁRIOS]/original-[REPOSITÓRIO]**

**# \* [new branch] master -> upstream/master**





# REFERÊNCIAS

- <https://git-scm.com/>
- <https://www.digitalocean.com/community/tutorials/how-to-install-git-on-ubuntu-20-04-pt>
- <https://www.hostinger.com.br/tutoriais/tutorial-do-git-basics-introducao>
- <https://abracd.org/porque-e-quando-usar-git-e-github/>
- <https://github.com/mateusKoppe/git-guia-basico>
- <https://terminalroot.com.br/2017/12/como-criar-um-pull-request-no-github.html>

