

## Parte 4 - Do git init ao git push

 [app.betrybe.com/course/fundamentals/git/exercicios/parte-4-do-git-init-ao-git-push](https://app.betrybe.com/course/fundamentals/git/exercicios/parte-4-do-git-init-ao-git-push)

Agora você vai aprender a iniciar um repositório **Git**, fazer seu primeiro *commit* e subi-lo para um repositório no **GitHub**.

Esse é um momento apenas de leitura, não é necessário replicar os passos que você verá agora. Foque em entender todo o fluxo, pois na próxima parte você irá precisar dele!

### Criando um repositório local

Antes de se criar um repositório é preciso criar uma pasta para ele. Para isso você deve utilizar o comando `mkdir`, como vimos anteriormente, e então navegar para a pasta criada com o comando `cd`.

Para criar um repositório você deve digitar o comando abaixo. É muito importante que esteja dentro da pasta criada para o repositório.

```
git init
```

Como seu nome bem diz, esse comando é responsável por iniciar um repositório **Git** dentro da pasta em que foi executado.

Para verificar se um repositório **Git** foi de fato iniciado, você pode executar o comando `git status`, que retorna o status do repositório. No contexto de um repositório recém criado onde nenhuma modificação foi feita você receberia a seguinte resposta:

```
No ramo master
```

```
No commits yet
```

```
nada para enviar (crie/copie arquivos e use "git add" para registrar)
```

### Adicionando e comitando

Considere que você já criou alguns arquivos e fez algumas modificações. Para que possamos versionar alterações feitas no código é preciso sempre seguir a seguinte sequência:

- Adicionar ( `add` )
- Comitar ( `commit -m "mensagem"` )

Os comandos ficam da seguinte forma:

```
# Adicionar todos os arquivos modificados
git add .

# Você também pode adicionar arquivos específicos
git add meu_arquivo.js

# Então você comita a alteração
# Ao comitar você deve adicionar também uma mensagem que descreve o que aquela
alteração faz

git commit -m "Mensagem sobre a alteração feita"
```

## Criando um repositório no GitHub

---

Agora é o momento de criar um repositório remoto. Para isso acesse o **GitHub** e procure o ícone com um sinal **+** na barra superior e ao clicar nele busca pela opção **New repository** . Você será redirecionado para o página semelhante a essa:

## 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 \*

 gomesanac ▾

Repository name \*

/

Great repository names are short and memorable. Need inspiration? How about **redesigned-goggles**?

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

### Página de criação de repositório

Você deve então adicionar um nome ao seu repositório, como por exemplo **meu-super-repo**. Após fazer isso o botão **Create repository** será habilitado e ao clicar nele seu repositório será criado.

Por enquanto ignore as outras opções, a medida que for avançando no curso você aprenderá mais sobre elas! Como seu repositório estará vazio, o **GitHub** lhe dará algumas dicas, você verá uma página parecida com essa:

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

or

HTTPS

SSH

git@github.com:gomesanac/repo-teste.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 "# repo-teste" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M master
git remote add origin git@github.com:gomesanac/repo-teste.git
git push -u origin master
```

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

```
git remote add origin git@github.com:gomesanac/repo-teste.git
git branch -M master
git push -u origin master
```

**...or import code from another repository**

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

Import code

Repositório vazio criado

Você deverá clicar no botão **SSH** e então copiar a URL gerada. Você irá precisar dela para realizar a conexão entre seu repositório local e seu repositório remoto.

## Conectando o repositório local com o remoto

Para conectar os dois repositórios você deverá abrir o seu terminal, acessar o diretório do seu repositório e então executar o seguinte comando:

```
git remote add origin git@github.com:user-github/repo-name.git
```

Sendo **origin** um apelido para o seu repositório, poderia ser qualquer outro. E no lugar da URL **git@github.com:user-github/repo-name.git** deve ir a gerada pelo seu repositório.

Para verificar que tudo funcionou corretamente, execute o comando **git remote -v**, você obterá um resultado semelhante a esse:

```
origin  git@github.com:user-github/repo-name.git (fetch)
origin  git@github.com:user-github/repo-name.git (push)
```

## Sincronizando os repositórios

Os repositórios já estão criados e também já estão conectados, agora é a hora de enviar as alterações feitas no repositório local para o repositório remoto.

Para isso, certifique-se que as alterações já foram adicionadas e comitadas e então execute o seguinte comando:

```
git push origin master
```

Com isso, você está enviando as alterações feitas localmente para o a branch principal, **master** , do seu repositório remoto, **origin** .

Se tudo ocorreu conforme o esperado, acesse novamente, ou atualize, a página do seu repositório no **GitHub** , você então verá que os arquivos e alterações que comitou agora se encontram lá!

---