

Aula #15: Comando git remote - Adicionando e Removendo Repositórios Remotos

Comando git remote	1
Como Adicionar um Repositório Remoto	2
Como Remover um Repositório Remoto	8
Próxima Aula	9
Exercícios	9
Fontes e Links Complementares	10

Comando git remote

Conforme já vimos anteriormente, um repositório remoto nada mais é que um repositório que está localizado em alguma máquina ou servidor que não a nossa máquina.

Repositórios remotos são extremamente importantes para o trabalhos em equipes, uma vez que todos os commits e alterações feitas por todos os membros da equipe são enviados para esses repositórios. Ou seja, é impossível trabalhar em equipe utilizando o Git sem ter que lidar com repositórios remotos.

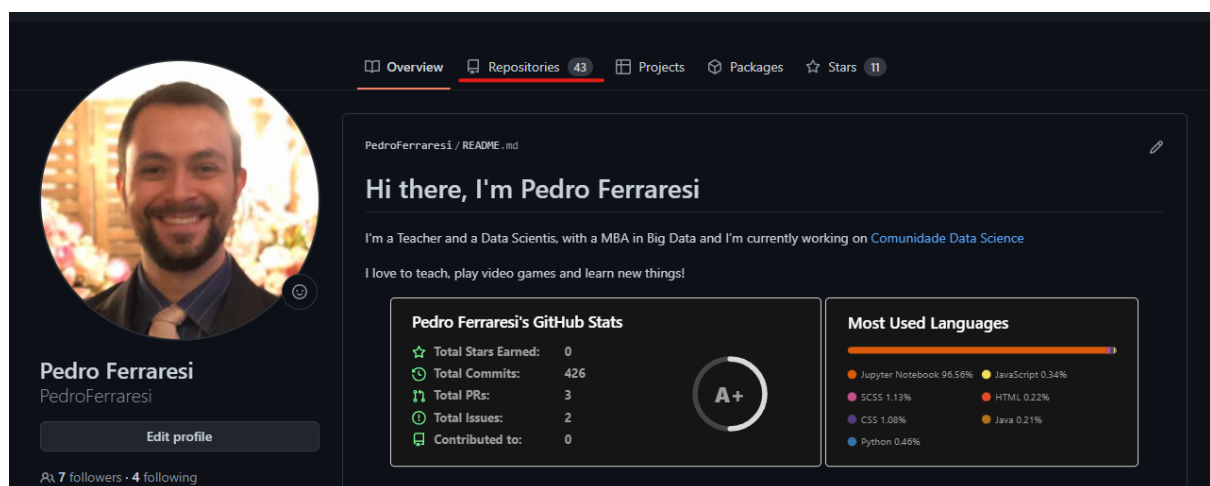
Mas como podemos trabalhar ou configurar repositórios remotos no Git? Fazemos isso através do comando `git remote`. É com esse comando que conseguimos

configurar e indicar para o repositório local quais os repositórios remotos estamos trabalhando. É com ele que inserimos e removemos repositórios remotos.

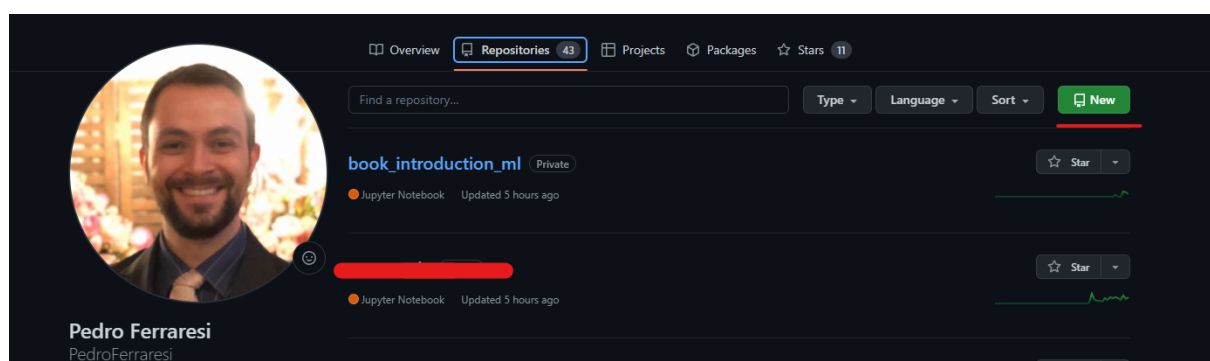
Como Adicionar um Repositório Remoto

Para adicionar um repositório remoto em nosso repositório local é extremamente simples, basta utilizarmos o comando `git remote add <nome_repo> <url_repo>`, sendo que o `<nome_repo>` é o nome que utilizamos para identificar o repositório remoto em nosso repositório local, e a `<url_repo>` é a url do repositório remoto que iremos adicionar.

Para exemplificar o funcionamento deste comando, vamos criar um repositório remoto no Github. A partir da sua página de perfil, clique na aba de repositórios



E depois clique no botão `new`, para criar um novo repositório.



Dê o nome de `cds_curso_git` ao repositório, deixe-o `público` e `não` `adicione` arquivos de `README.md`, `license` e nem `.gitignore`.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * **Repository name ***

PedroFerraresi / ✓

Great repository names are short and memorable. Need inspiration? How about [supreme-octo-spork?](#)

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.](#)

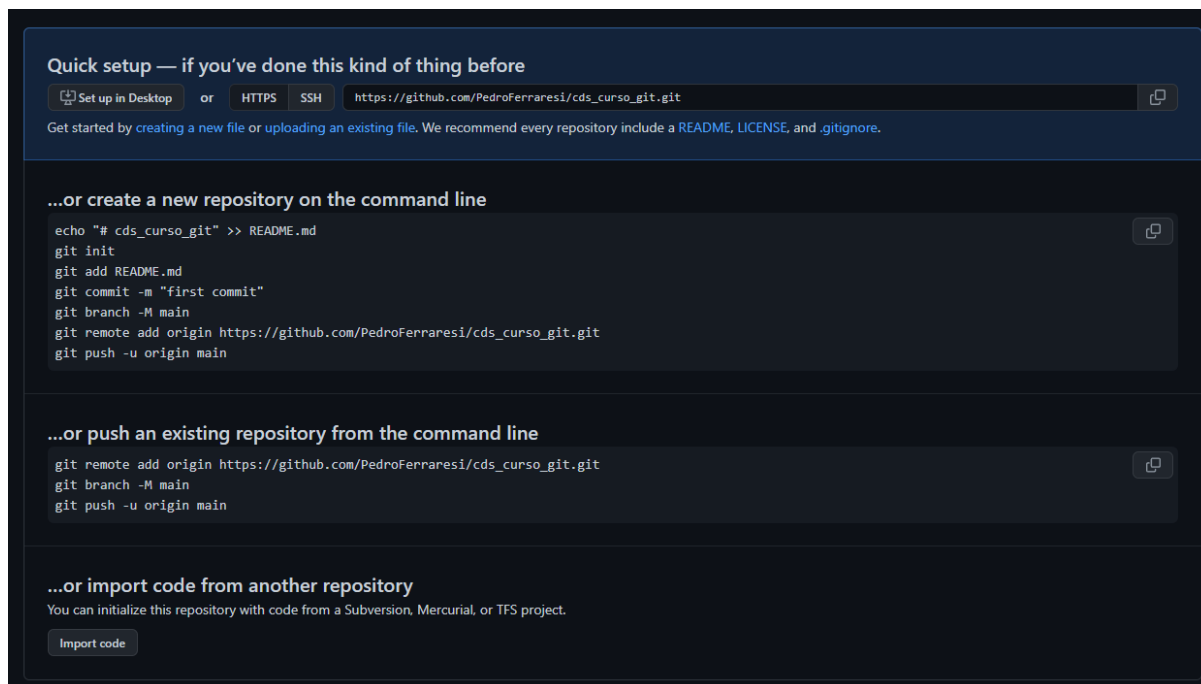
Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

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

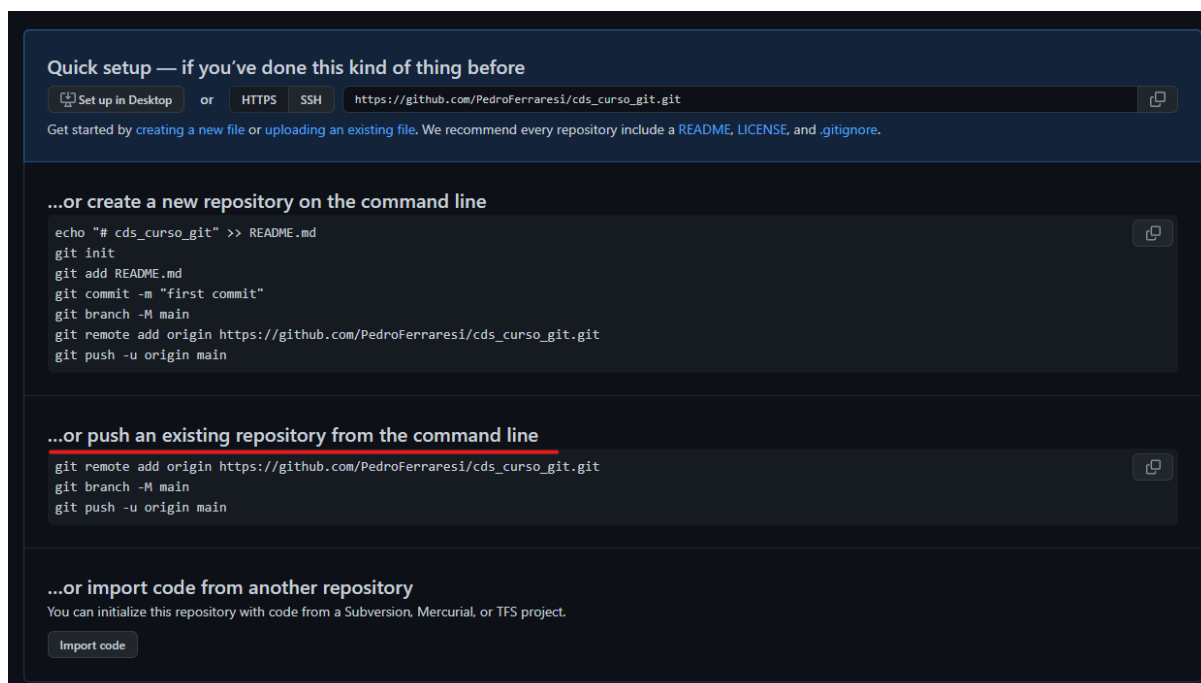
You are creating a public repository in your personal account.

[Create repository](#)

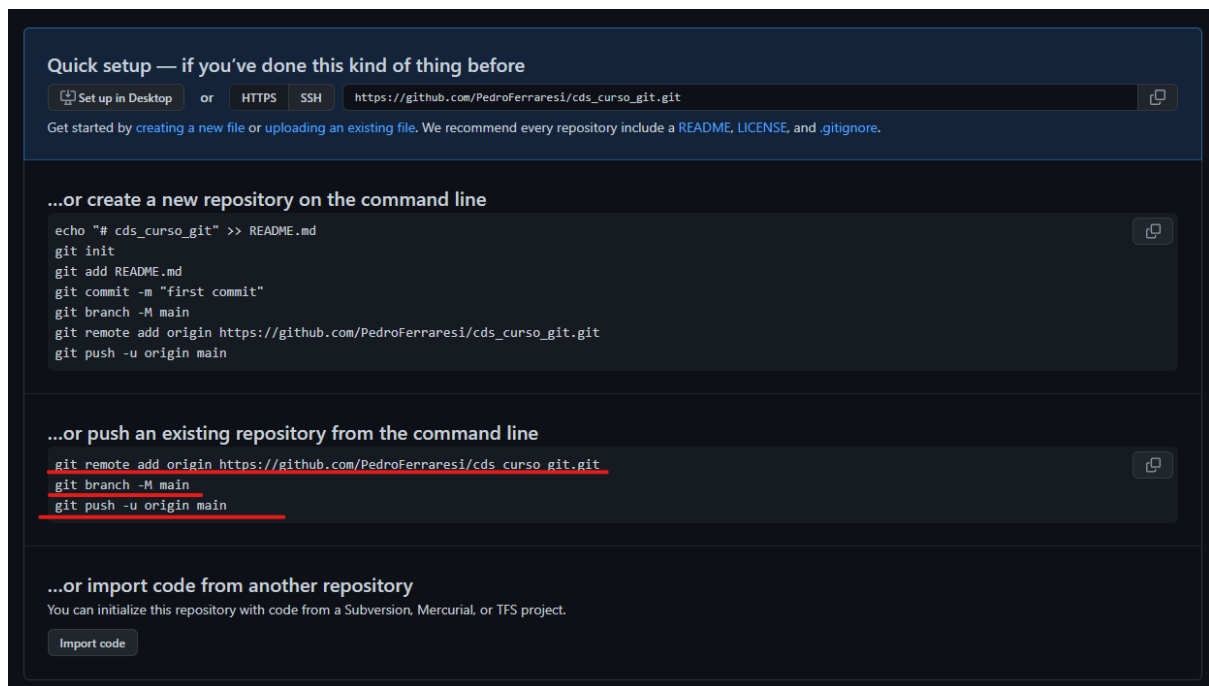
Clique no botão Create Repository para criar o repositório. Com o repositório criado, teremos a seguinte tela:



Nela, o Github já nos diz o que fazer para configurarmos o repositório remoto recém criado em nosso repositório local, para que possamos enviar todo o nosso trabalho já realizado para o repositório remoto recém criado:



Dessa forma, vamos abrir o nosso projeto, e no terminal do Git, vamos digitar os comandos que o Github está nos informando:



Mas antes, vamos entender comando a comando o que está sendo feito. O primeiro comando, é a utilização do `git remote`. Nele, estamos utilizando o parâmetro `add`, para adicionar um repositório remoto, e após esse parâmetro, o nome desse repositório e a sua URL:

```
git remote add <nome_repo_remoto> <url_repo_remoto>
```

O segundo comando, o `git branch -M main`, serve para renomear a branch em que estamos. Caso o nome da branch do seu repositório local `master`, podemos utilizar esse comando para renomear a nossa branch local, isso porque no Github, a branch principal possui o nome de `main`, e não de `master`. No nosso caso, como o nome da branch principal do repositório local é `main`, não precisamos executar esse comando.

Já o último comando, `git push`, ele serve para fazer o upload de todos os commits do nosso repositório para o repositório remoto.

Um ponto importante a respeito da aula de hoje: Tanto os comandos `git push` quanto `git branch` serão abordados em aulas futuras. Portanto, por enquanto vamos somente entender o que eles estão fazendo nessa sequência de comandos!

Dessa forma, vamos então executar os dois comandos que precisamos. Primeiro, vamos adicionar o novo repositório com o comando `git remote`:

```
Pedro@Ryzen MINGW64 ~/Documents/reposcurso_git/projeto (main)
$ git remote add origin https://github.com/PedroFerraesi/cds_curso_git.git
```

O parâmetro `origin` do comando acima, representa o nome do repositório remoto. Portanto, quando trabalhamos com mais de um repositório remoto em nosso projeto, podemos nomear os repositórios remotos conforme a nossa necessidade ou a localização deles.

Como a nossa branch local principal já é a `main`, não precisamos executar o comando `git branch -M main`, para renomear a branch em que estamos. Por fim, vamos subir os nossos commits para o repositório remoto utilizando o comando `git push`. A anatomia desse comando é:

```
git push <nome_repos_remoto> <branch_local>
```

Além disso, o Github nos diz para usar o parâmetro `-u` junto ao comando. Isso fará com que configuremos a branch remota de forma que ela seja a representação da nossa branch local!

```
Pedro@Ryzen MINGW64 ~/Documents/repos/curso_git/projeto (main)
$ git push -u origin main
Enumerating objects: 48, done.
Counting objects: 100% (48/48), done.
Delta compression using up to 16 threads
Compressing objects: 100% (42/42), done.
Writing objects: 100% (48/48), 723.93 KiB | 11.31 MiB/s, done.
Total 48 (delta 15), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (15/15), done.
To https://github.com/PedroFerraesi/cds_curso_git.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Feito isso, podemos verificar em nosso repositório remoto que todos os nossos commits foram enviados para eles!

The screenshot shows the GitHub interface for the repository 'PedroFerraesi/cds_curso_git'. At the top, there are navigation buttons: 'Go to file', 'Add file', and 'Code'. Below this, the repository name and owner are displayed, along with statistics: '83160eb 3 days ago' and '14 commits'. A list of files and folders is shown, including 'data', 'notebooks', 'src', '.gitignore', 'README.md', 'app.py', 'license', and 'requirements.txt', each with a description and the commit date. Below the file list, the 'README.md' content is visible, featuring the title 'Curso de Git' and a description: 'Este repositório foi criado para hospedar o esqueleto do projeto que será utilizado para explicar e ensinar o uso básico do Git dentro da Comunidade DS'.

File/Folder	Description	Commit Date
data	Ajustado estura de patas	3 days ago
notebooks	Rodado arquivo de notebook	27 days ago
src	Criado arquivo de carregamento de dados e ajustado arquivo principal ...	3 days ago
.gitignore	Primeiro Commit	last month
README.md	Primeiro Commit	last month
app.py	Criado arquivo de carregamento de dados e ajustado arquivo principal ...	3 days ago
license	Criado arquivo de licenca	last month
requirements.txt	Primeiro Commit	last month

Curso de Git

Este repositório foi criado para hospedar o esqueleto do projeto que será utilizado para explicar e ensinar o uso básico do Git dentro da Comunidade DS

Dessa forma, adicionamos e configuramos um repositório remoto em nosso repositório local. Agora, todos os novos commits que fizermos em nosso repositório local ou remoto, podem ser enviados de um para o outro!

Como Remover um Repositório Remoto

Da mesma forma que o comando que utilizamos para adicionarmos um repositório remoto em nosso repositório local, o comando para remover é simples: `git remote remove <nome_repo_remoto>`.

Antes de removermos o repositório remoto que acabamos de adicionar, vamos primeiro utilizar o comando `git remote -v` para verificar quais repositórios remotos temos cadastrados em nosso repositório local.

```
Pedro@Ryzen MINGW64 ~/Documents/repos/curso_git/projeto (main)
$ git remote -v
origin https://github.com/PedroFerraresi/cds_curso_git.git (fetch)
origin https://github.com/PedroFerraresi/cds_curso_git.git (push)
```

Observe que temos duas entradas: uma para o `fetch` e outra para o `push`. O `fetch`, representa o endereço que iremos utilizar para baixar os commits para o nosso repositório local, enquanto `push` representa o endereço que iremos enviar os commits do nosso repositório local. Ambos são o mesmo endereço: O nosso repositório remoto.

Para remover o repositório remoto, basta utilizarmos o comando `git remote remove origin`:

```
Pedro@Ryzen MINGW64 ~/Documents/repos/curso_git/projeto (main)
$ git remote remove origin
```


Como podemos observar através do comando `git remote -v`, o repositório remoto foi removido corretamente!

```
Pedro@Ryzen MINGW64 ~/Documents/repos/curso_git/projeto (main)
$ git remote -v
```

Para finalizar a aula, vamos inseri-lo novamente com o comando:

```
git remote add origin
https://github.com/PedroFerrearesi/cds_curso_git.git
```

```
Pedro@Ryzen MINGW64 ~/Documents/repos/curso_git/projeto (main)
$ git remote add origin https://github.com/PedroFerrearesi/cds_curso_git.git

Pedro@Ryzen MINGW64 ~/Documents/repos/curso_git/projeto (main)
$ git remote -v
origin https://github.com/PedroFerrearesi/cds_curso_git.git (fetch)
origin https://github.com/PedroFerrearesi/cds_curso_git.git (push)
```

Isso concluí o uso básico do comando `git remote`, que utilizamos tanto para inserir quanto para remover repositórios remotos no nosso repositório local.

Próxima Aula

Na próxima aula, iremos iniciar um tour pela ferramenta online Github e conhecer, de forma básica, as suas principais ferramentas!

Exercícios

Link para o formulário de exercícios de fixação de conteúdo: [Exercícios](#)

Fontes e Links Complementares

[Documentação - Comando git remote](#)

[Tutorial - Github: Gerenciando Repositórios Remotos](#)

[Tutorial - Resolvendo problemas de download de commits](#)

[Artigo Medium - Fundamentos do Github - Diferença entre fetch, push, pull e fork](#)

[Livro - Pro Git - 2.5 - Trabalhando com Repositórios Remotos](#)