

Git & GitHub

Introdução

Prática de Laboratório

Wesley Dias Maciel

2023/02

Exercício 01

Objetivo:

Instalar o Git e criar conta no GitHub.

Git.

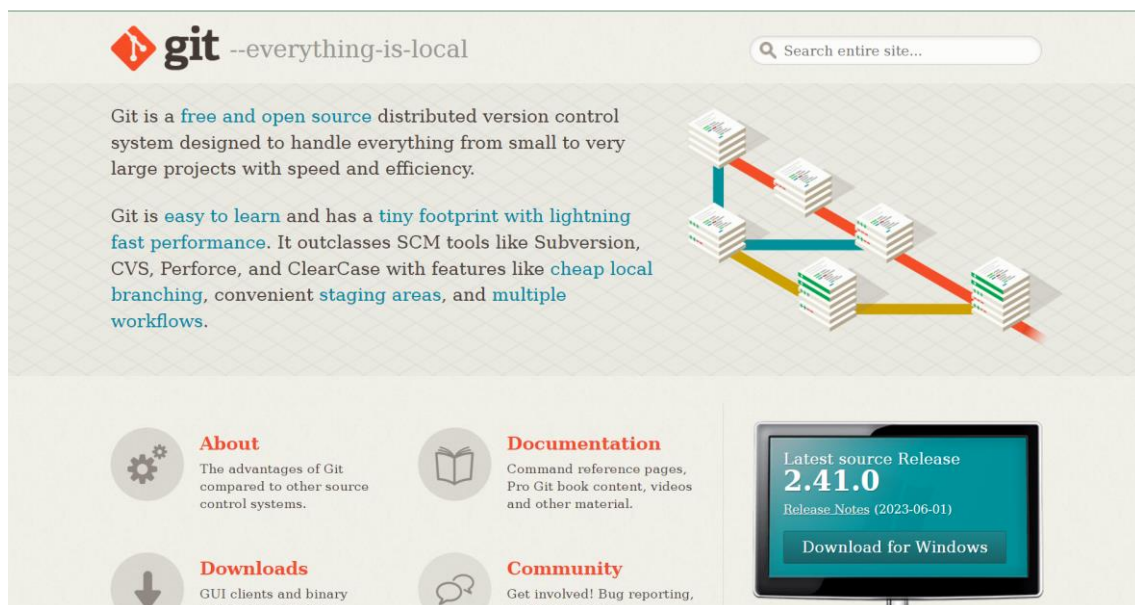
Sistema de controle de versão (Version Control System, VCS). Criado pelo Linus Torvalds.




“O controle de versão é um sistema que registra as mudanças feitas em um arquivo ou em um conjunto de arquivos ao longo do tempo de forma que você possa recuperar versões específicas. Mesmo que os exemplos desse livro mostrem arquivos de código fonte sob controle de versão, você pode usá-lo com praticamente qualquer tipo de arquivo em um computador”.

(Scott Chacon e Ben Strub, 2 Edição, 2014)

1) Página: <https://git-scm.com/>.

A screenshot of the Git website homepage. The header features the Git logo (a red diamond with a white 'G') and the tagline "--everything-is-local". A search bar is located in the top right corner. The main content area describes Git as a "free and open source distributed version control system" and lists its features: "easy to learn", "tiny footprint with lightning fast performance", "cheap local branching", "convenient staging areas", and "multiple workflows". To the right of the text is a diagram showing a branching model with several stacks of code blocks connected by colored lines (red, blue, yellow). Below the main content are four sections: "About" (advantages of Git), "Documentation" (command reference, Pro Git book, videos), "Downloads" (GUI clients, binary releases), and "Community" (bug reporting, mailing list, chat). On the right side of these sections is a monitor displaying the "Latest source Release 2.41.0" and a "Download for Windows" button.

2) Download para Windows: <https://git-scm.com/download/win>.

 **git** --everything-is-local

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git** book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on [Amazon.com](#).

Download for Windows

[Click here to download](#) the latest (2.41.0) 64-bit version of **Git for Windows**. This is the most recent [maintained build](#). It was released **about 1 month ago**, on 2023-07-13.

Other Git for Windows downloads

Standalone Installer
[32-bit Git for Windows Setup](#).
[64-bit Git for Windows Setup](#).

Portable ("thumbdrive edition")
[32-bit Git for Windows Portable](#).
[64-bit Git for Windows Portable](#).

Using winget tool


Install [winget tool](#) if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version **2.41.0**. If you want the newer version, you can build it from [the source code](#).

3) Instalar o Git localmente em sua máquina, seguindo os passos do instalador.

4) Livro: <https://git-scm.com/book/en/v2>

 **git** --fast-version-control


Search entire site...

About
Documentation
Reference
Book
Videos
External Links
Downloads
Community

This book is available in [English](#).
Full translation available in
[azerbaycan dili](#),
[български език](#),
[Deutsch](#),
[Español](#),
[Français](#),
[Ελληνικά](#),
[日本語](#).

Book

The entire Pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is available here. All content is licensed under the [Creative Commons Attribution Non Commercial Share Alike 3.0 license](#). Print versions of the book are available on [Amazon.com](#).






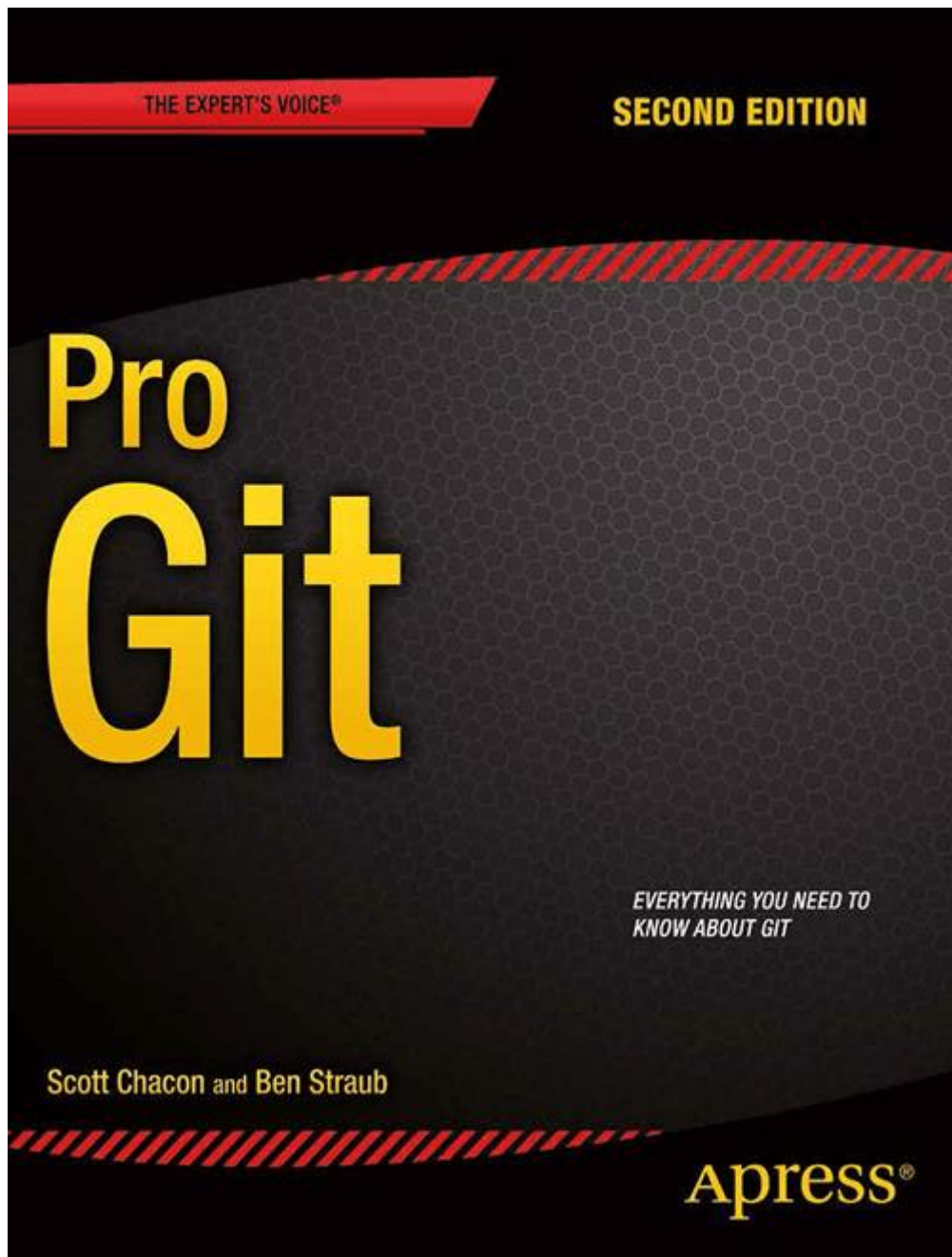
2nd Edition (2014)

1. Getting Started


- 1.1 About Version Control
- 1.2 A Short History of Git
- 1.3 What is Git?
- 1.4 The Command Line
- 1.5 Installing Git
- 1.6 First-Time Git Setup
- 1.7 Getting Help
- 1.8 Summary

2. Git Basics

Download Ebook




- 5) Link para o livro em português: <https://git-scm.com/book/pt-br/v2>


--everything-is-local

About

Documentation

Reference

Book

Videos

External Links

Downloads

Community

This book is available in [English](#).

Full translation available in

azərbaycan dili,

български език,

Deutsch,

Español,


Français,

Ελληνικά,

日本語.



Book

The entire Pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is available here. All content is licensed under the [Creative Commons Attribution Non Commercial Share Alike 3.0](#) license. Print versions of the book are available on [Amazon.com](#).



2nd Edition (2014)

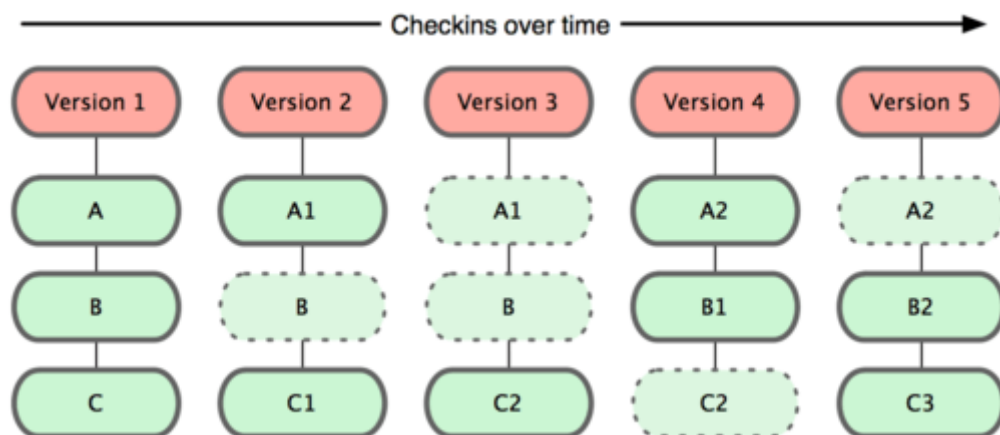
Download Ebook

- Começando**
 - Sobre Controle de Versão
 - Uma Breve História do Git
 - O Básico do Git
 - A Linha de Comando
 - Instalando o Git
 - Configuração Inicial do Git
 - Pedindo Ajuda
 - Sumário
- Fundamentos de Git**

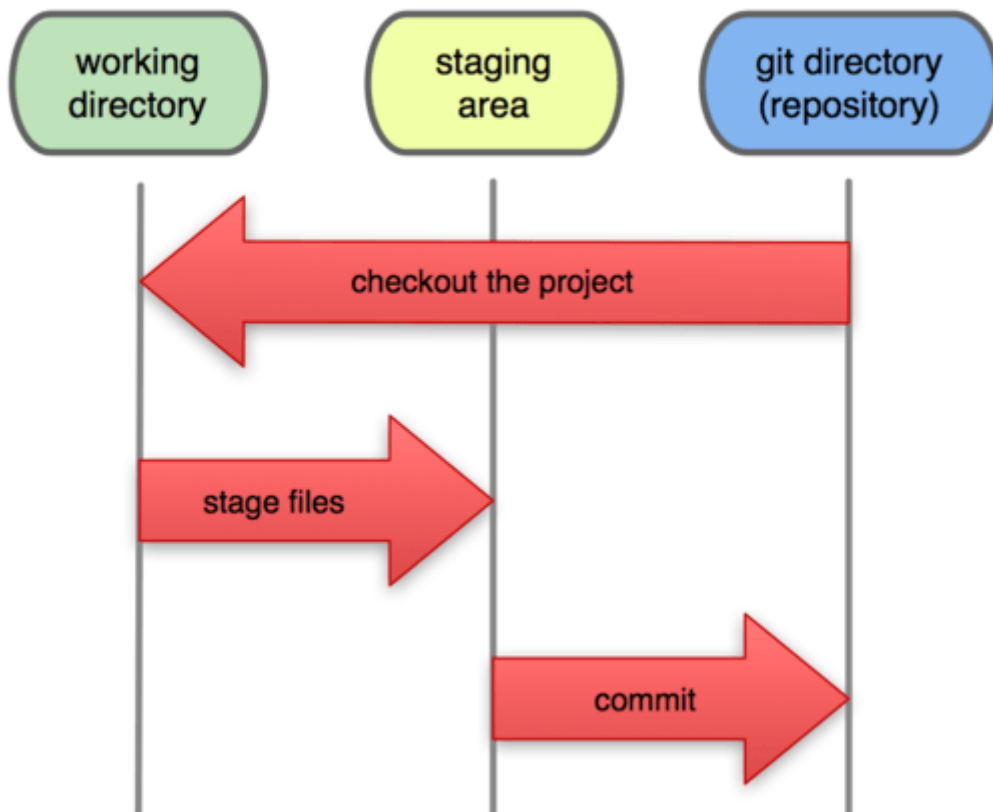
6) Curso: <https://www.udemy.com/course/git-e-github-para-iniciantes/>

7) Git armazena dados como “snapshots” do projeto ao longo do tempo.



8) Os três estados: diretório de trabalho, área de preparação (“staging area”) e o diretório do Git.

Local Operations



Diretório do Git:

- 1) Local onde o Git armazena os metadados e o banco de objetos de seu projeto.
- 2) Essa é a parte mais importante do Git.
- 3) É a parte copiada quando você clona um repositório de outro computador.

Diretório de trabalho:

- 1) Uma versão do projeto que está sendo trabalhada.
- 2) Esses arquivos são obtidos a partir da base de dados comprimida no diretório do Git e colocados em disco para que você possa utilizar ou modificar.

“Staging área”:

- 1) É um arquivo, geralmente armazenado no diretório do Git.
- 2) Armazena informação sobre o que entrará no próximo “commit”.
- 3) Algumas vezes, é chamado de índice, “index”, “stage” ou “staging area”.

Fluxo básico do Git:

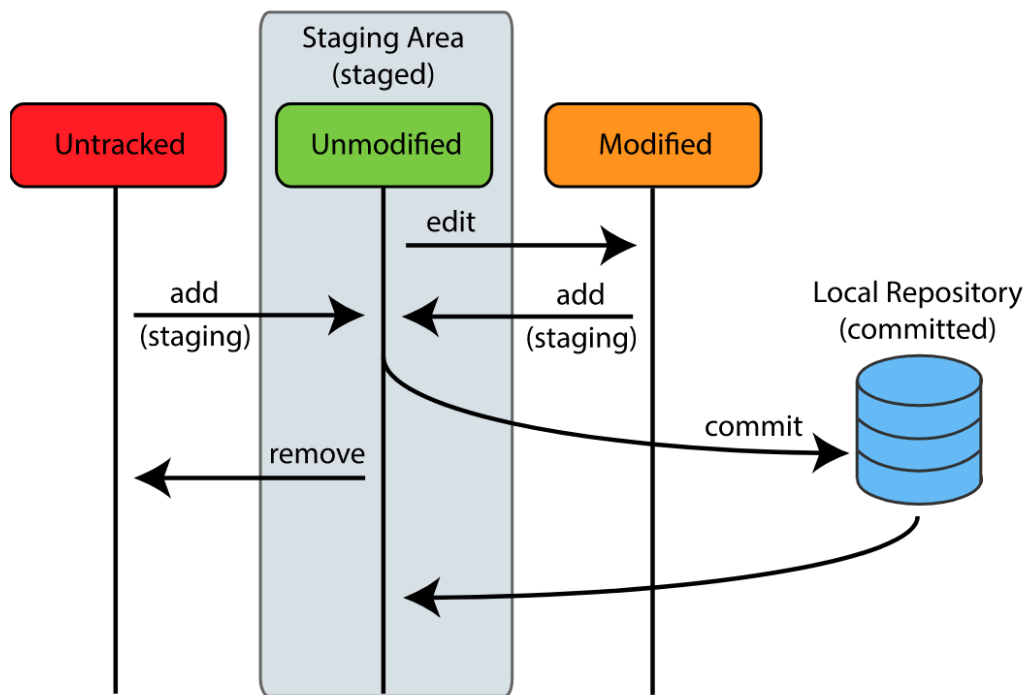
- 1) Você modifica arquivos no seu diretório de trabalho.

- 2) Você seleciona os arquivos, adicionando “snapshots” deles na “staging area”.
- 3) Você faz um “commit”, que leva os arquivos como eles estão na “staging area” e os armazena permanentemente no seu diretório do Git.

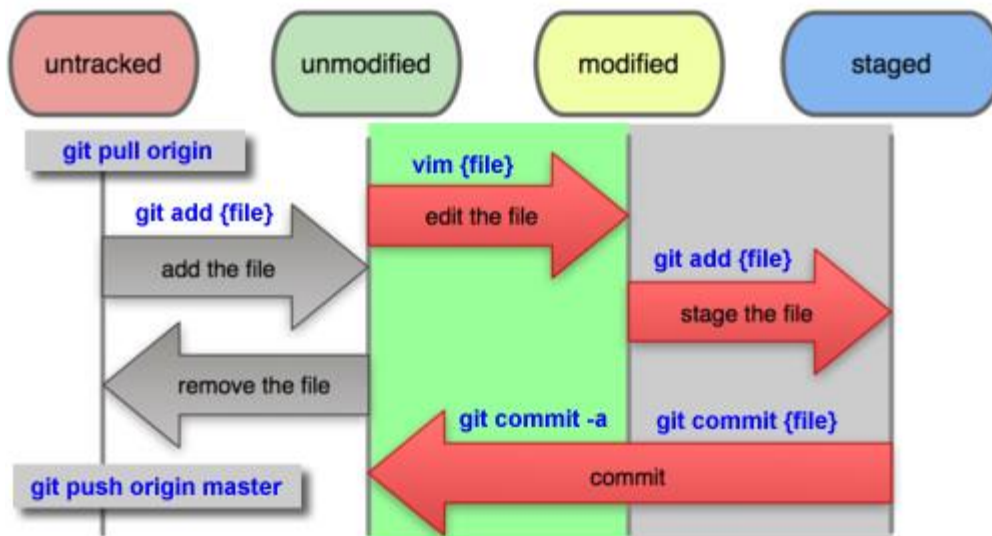
OBS:

- 1) Se uma versão de um arquivo está no diretório do Git, ela é considerada “committed”.
- 2) Se uma versão de um arquivo foi adicionada à “staging area”, ela é considerada “staged”.
- 3) Se uma versão de um arquivo foi alterada, mas não foi adicionada à “staging area”, ela é considerada modificada.

9) Ciclo de vida dos arquivos.



File Status Lifecycle



- 1) **untracked:** arquivo adicionado ao projeto, mas ainda não rastreado pelo Git. O arquivo ainda não é conhecido pelo Git, não faz parte do diretório do Git. O arquivo não é monitorado pelo Git.
- 2) **unmodified:** arquivo adicionado ao Git e ainda não modificado. O arquivo faz parte do diretório e não foi alterado.
- 3) **modified:** arquivo do diretório e que foi modificado.
- 4) **staged:** arquivos que farão parte de uma nova versão, quando o usuário executar o próximo comando “commit”. Esses arquivos são considerados “unmodified”.
- 5) **committed:** arquivos que fazem parte de uma versão.

Exercício 02

Git: configurações iniciais.

- 1) Num terminal de sua máquina local, verificar a instalação local, reportando a versão do Git instalada.

```
$ git --version
```

```
git version 2.41.0.windows.3
```

ou

```
$ git -v
```

```
git version 2.41.0.windows.3
```

git config

Configurações para todo o sistema, todos os usuários: informar o parâmetro “**--system**”.

Configurações para o usuário corrente: informar o parâmetro “**--global**”.

Configurações locais do projeto: não informar parâmetro, deixar em branco.

- 2) Definir nome, e-mail e editor de texto padrão do usuário.

```
$ git config --global user.name "Wesley Dias Maciel"
```

```
$ git config --global user.email "wesleydiasmaciел@gmail.com"
```

```
$ git config --global core.editor notepad
```

OBS: VS Code:

```
git config --global core.editor "code --wait"
```

ou

```
git config --global core.editor "code"
```

- 3) Exibir as configurações fornecidas.

```
$ git config user.name
```

```
Wesley Dias Maciel
```

```
$ git config user.email
```

wesleydiasmaci@gmail.com

```
$ git config core.editor
```

notepad

```
$ git config --list
```

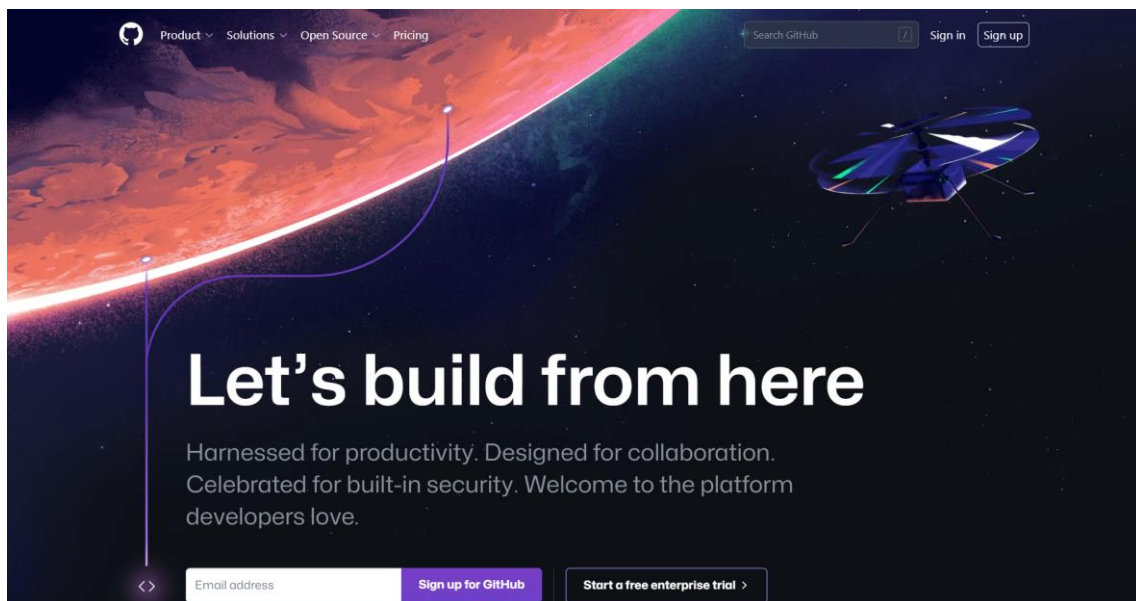
```
core.symlinks=false
core.autocrlf=true
core.fscache=true
color.diff=auto
color.status=auto
color.branch=auto
color.interactive=true
help.format=html
rebase.autosquash=true
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
http.sslbackend=openssl
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.required=true
filter.lfs.process=git-lfs filter-process
credential.helper=manager
user.name=Wesley Dias Maciel
user.email=wesleydiasmaci@gmail.com
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
core.editor=notepad
```

Exercício 03

GitHub.

Repositório remoto, público ou privado. Plataforma de hospedagem (de código-fonte) que usa o Git como sistema de controle de versão. Local na Web para armazenamento de projetos. Também usado como rede social.

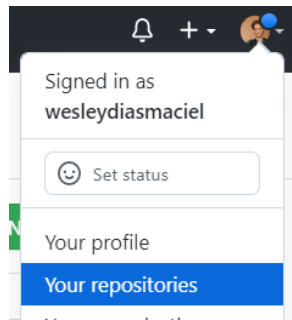
- 1) Página: <https://github.com/>.



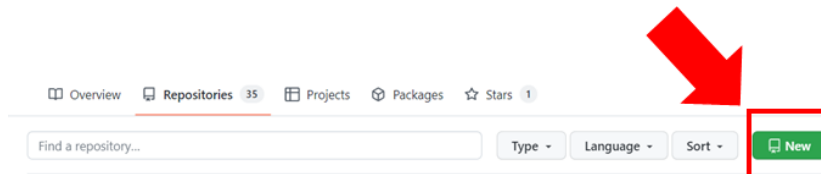
- 2) Criar uma conta e entrar: <https://github.com/login>.

A screenshot of the GitHub login page. It features the GitHub logo at the top, followed by the text "Sign in to GitHub". Below this is a form with two input fields: "Username or email address" and "Password". A "Forgot password?" link is positioned next to the password field. A green "Sign in" button is located below the password field. At the bottom of the form, there is a link that says "New to GitHub? Create an account."

- 3) No canto superior direito, clique em "seus repositórios":



4) Na nova página, clique em “Novo”:



5) Na nova página, informe:

a) Um nome para seu repositório:

Create a new repository

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

Owner *

wesleydiasmaciell

Repository name *

exemplo

Great repository names are short exemplo is available. ed inspiration? How about [congenial-octo-garbanzo?](#)

Description (optional)

b) Escolha a forma de visualização de seu repositório: público ou privado.



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.

c) Não altere as demais opções e clique no botão “criar repositório”:

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.](#)

.gitignore template: None ▾

Choose a license

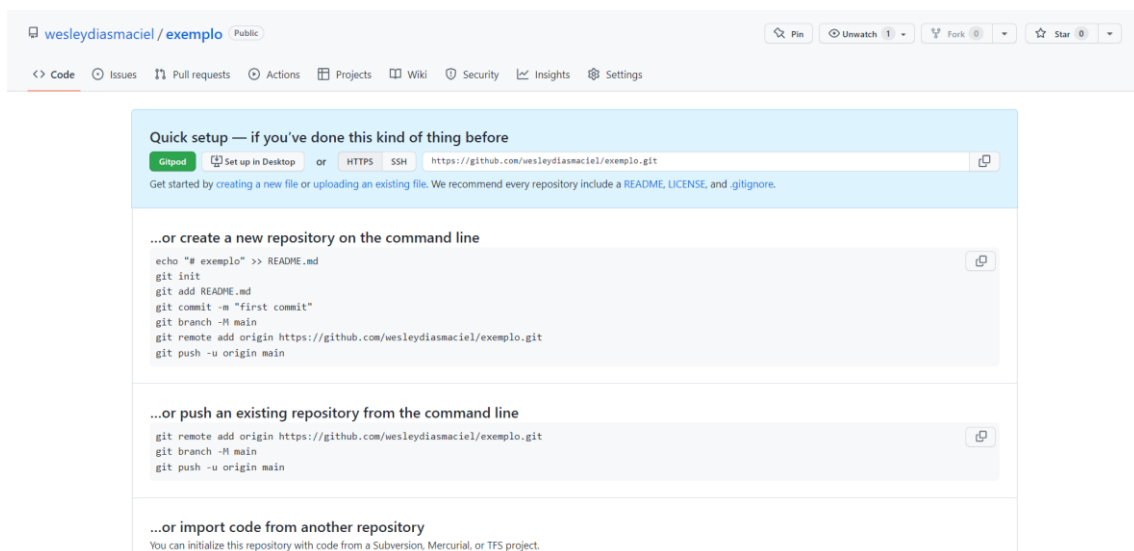
A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

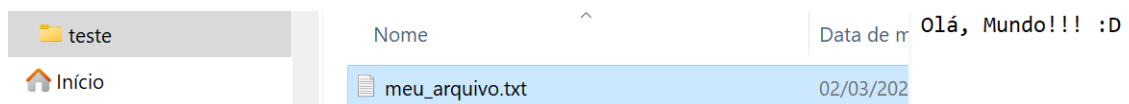
 You are creating a public repository in your personal account.

Create repository

6) Tela do seu novo repositório:



7) Crie uma pasta “teste” com um arquivo “meu_arquivo.txt” em seu computador. No arquivo, escreva: “Olá, Mundo!!! :D”.



8) Num terminal, entre na pasta “teste”:

```
PS C:\Users\wesle\Documents\teste> dir

Diretório: C:\Users\wesle\Documents\teste

Mode                LastWriteTime         Length Name
----                -
-a-----         02/03/2023   18:09             17 meu_arquivo.txt

PS C:\Users\wesle\Documents\teste> |
```

9) Dentro da pasta “teste”, execute os comandos:

a) git init

```
PS C:\Users\wesle\Documents\teste> git init
Initialized empty Git repository in C:/Users/wesle/Documents/teste/.git/
PS C:\Users\wesle\Documents\teste> |
```

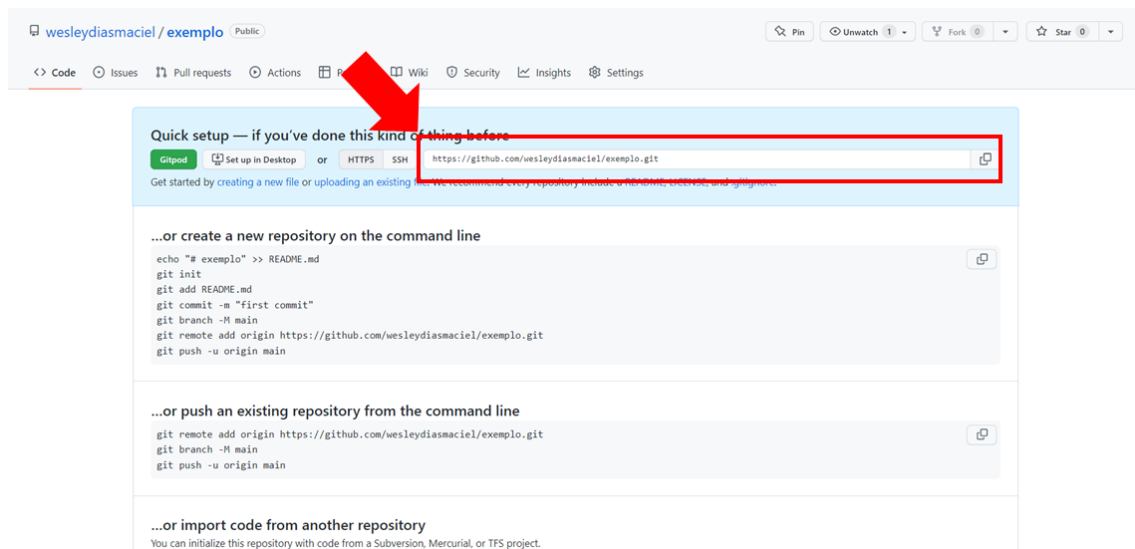
b) git add .

```
PS C:\Users\wesle\Documents\teste> git add .
PS C:\Users\wesle\Documents\teste> |
```

c) git commit -m “Meu primeiro commit.”

```
PS C:\Users\wesle\Documents\teste> git commit -m 'Meu primeiro commit.'
[main (root-commit) 96122be] Meu primeiro commit.
1 file changed, 1 insertion(+)
create mode 100644 meu_arquivo.txt
PS C:\Users\wesle\Documents\teste> |
```

10) Copie a URL de seu repositório no GitHub:



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

Gitpod Set up in Desktop or HTTPS SSH <https://github.com/wesleydiasmaciell/exemplo.git>

Get started by creating a new file or uploading an existing one. We recommend every repository include a README, at least one file, and a .gitignore.

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

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

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

```
git remote add origin https://github.com/wesleydiasmaciell/exemplo.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.

11) Dentro da pasta “teste”, execute os comandos:

a) git remote add origin <url do repositório remoto>

```
PS C:\Users\wesle\Documents\teste> git remote add origin https://github.com/wesleydiasmaciell/exemplo.git
PS C:\Users\wesle\Documents\teste> |
```

OBS: caso você precise alterar (substituir) a URL do repositório remoto, execute:

```
$ git remote set-url origin <nova_url>
```

b) git branch

```
PS C:\Users\wesle\Documents\teste> git branch
* main
PS C:\Users\wesle\Documents\teste> |
```

c) Se o nome do branch não for **main**, altere o nome do branch com o comando:
git branch -m <nome-antigo> <novo-nome>

Exemplo:

```
PS C:\Users\wesle\Documents\teste> git branch
* master
PS C:\Users\wesle\Documents\teste> git branch -m master main
PS C:\Users\wesle\Documents\teste> git branch
* main
PS C:\Users\wesle\Documents\teste> |
```

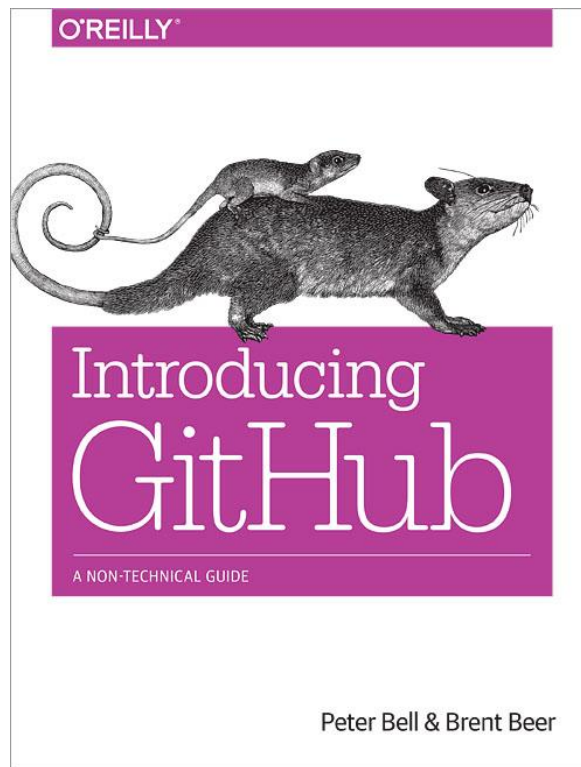
d) git push -u origin main

```
PS C:\Users\wesle\Documents\teste> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 250 bytes | 250.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/wesleydiasmaciels/exemplo.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\wesle\Documents\teste> |
```

12) Veja a atualização de seu repositório no GitHub:

The screenshot shows the GitHub interface for the repository 'wesleydiasmaciels/exemplo'. A red arrow points to the commit history table. The table lists a single commit by 'wesleydiasmaciels' titled 'Meu primeiro commit.' with a file named 'meu_arquivo.txt' added. The commit was made 29 minutes ago. The repository has 0 stars, 1 watching, and 0 forks. The page also includes sections for About, Releases, and Packages.

13) Livro:



Resumo

Comandos Básicos

git init //Inicia um repositório e começa a observar as alterações do projeto.
git add . //Realiza adições na staging area.
git commit -m 'mensagem' //Cria uma versão.
git remote add origin <url do repositório remoto> //Associar o repositório local ao repositório remoto.
git push -u origin main //Realizar envio do repositório local para o repositório remoto.
git branch //Lista o nome do branch.
git branch -m <nome-antigo> <novo-nome> //Renomeia o branch.
git checkout -b <nova branch> //Sai da branch corrente e entra na nova branch. Se a nova branch não existir, ela é criada.

Comandos Básicos de Configuração

git --version //Versão instalada do Git.
git -v //Versão instalada do Git.

git config --system user.name "nome-do-usuário" //Nome do usuário (nível: sistema, todos os usuários).
git config --global user.name "nome-do-usuário" //Nome do usuário (nível: usuário corrente)
git config user.name "nome-do-usuário" //Nome do usuário (nível: projeto corrente)

git config --system user.email "email-do-usuário" // E-mail do usuário (nível: sistema, todos os usuários).
git config --global user.email "email-do-usuário" // E-mail do usuário (nível: usuário corrente)
git config user.email "email-do-usuário" //E-mail do usuário (nível: projeto corrente)

git config --system core.editor "editor" // E-mail do usuário (nível: sistema, todos os usuários).
git config --global core.editor "editor" // E-mail do usuário (nível: usuário corrente)
git config core.editor "editor" //E-mail do usuário (nível: projeto corrente)

git config user.name //Lista o nome do usuário.
git config user.email //Lista o e-mail do usuário.
git config core.editor //Lista o editor.
git config --list //Lista as configurações.

Comandos Principais

git --version //Reportar a versão do sistema.
git config //Configurar o sistema.
git init //Inicia um repositório e começa a observar as alterações do projeto.

git status //Reportar o estado do repositório.
git add . //Realiza adições na staging area.
git commit -m 'mensagem' //Cria uma versão.
git log //Histórico dos commits realizados.
git shortlog //Histórico resumido dos commits realizados.
git show <id do commit> //Apresenta informação sobre um commit.
git diff //Exibe diferenças entre alterações.
git remote add origin <url do repositório remoto> //Associar o repositório local ao repositório remoto.
git remote //Listar a associação criada.
git remote -v //Listar detalhes sobre a associação criada.
git push origin main //Realizar envio do repositório local para o repositório remoto.

git push -u origin main //Realizar envio do repositório local para o repositório remoto.
-u: para não ter que informar “origin” e “main” nas próximas vezes que o repositório local tiver que ser enviado para o repositório remoto.
origin: repositório remoto.
main: para enviar do “branch” “main” local, para o “branch” “main” remoto. Se o “branch” “main” remoto não existir, ele é criado. Equivale a:
git push -u origin main:main
Que significa:
git push -u origin <branch local>:<branch remoto>
git push //Enviar do repositório local para o repositório remoto.

git branch //Lista o nome do branch.
git branch -m <nome-antigo> <novo-nome> //Renomeia o branch.
git checkout -b <nova branch> //Sai da branch corrente e entra na nova branch. Se a nova branch não existir, ela é criada.

git fetch origin //Buscar o repositório remoto para o repositório local, sem realizar o “merge” dos “branches”.

git pull //Realiza o “merge” dos “branches”.
git pull origin //Buscar o repositório remoto para o repositório local, realizando o “merge” dos “branches”.
git remote set-url origin <nova_url> //Substitui a URL do repositório remoto.