# Git para iniciantes Desvendando a ferramenta

SERGIO CABRAL



# Índice

1.	Perguntas e respostas rápidas	. 2
2.	Git na prática	. 3
	2.1. Como criar ou baixar um repositório Git	. 3
	2.2. O básico do básico	. 5
	2.3. Branches e Tags	. 7

Este conteúdo foi preparado para ser apresentado ao vivo na programação da EBAC sobre o tema Primeiros passos para se tornar um desenvolvedor.

Se tudo estiver certo, você consegue a versão atualizada desse **e-book** no link para download em git.sergiocabral.com/ebook.pdf

# 1. Perguntas e respostas rápidas

### O que é o Git?

Um software de linha de comando conhecido como VCS ou SCM.



(D)VCS = (Distributed) Version Control System SCM = Source Control Management

### O que o Git faz?

Uma ferramenta que possibilita fazer alterações em um conjunto de arquivos e registrar cada alteração numa linha do tempo.

### Por que preciso de um Git?

Além de manter um histórico de versões, torna possível trabalhar com equipes distribuídas.



Lembre-se que o seu **eu** de amanhã será diferente do seu **eu** de hoje e vocês precisam trabalhar em equipe.

### Como começo a usar o Git?

Você instala no modo *avançar-avançar* e transforma qualquer pasta em um repositório Git com o comando git init. Ou você pode usar o comando git clone para clonar um repositório existente.

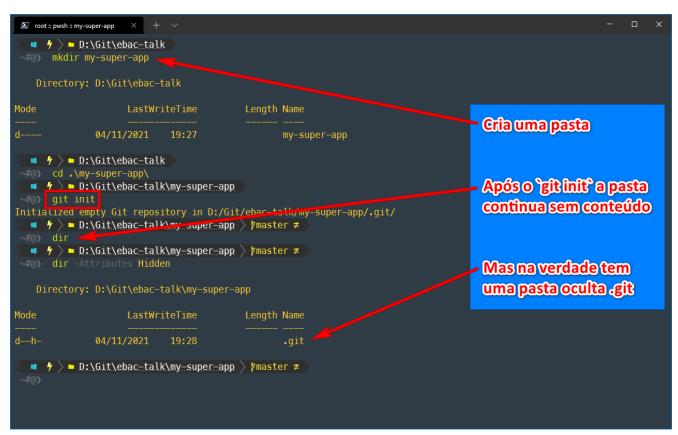
### Saiba também

- · Git não tem interface gráfica; é linha de comando.
- Git não é a única ferramenta do tipo, mas é a mais utilizada.
- Git não é GitHub.

# 2. Git na prática

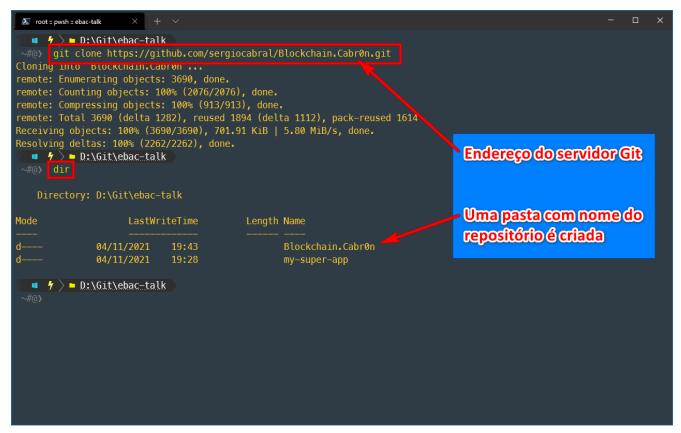
## 2.1. Como criar ou baixar um repositório Git

Resultado do git init.



Exemplo 1.

### Resultado do git clone.



Exemplo 2.

**(1)** 

O Git grava seu banco de dados na pasta .git.

### 2.2. O básico do básico

### git status

Estou num repositório Git? Quais arquivos ainda não estão no repositório?

### git add <nome-do-arquivo>

Marca arquivos para entrarem no repositório.

### git commit -m "<mensagem>"

Grava arquivos no repositório.

### git log

Mostra o histórico de alterações.

```
∑ root :: pwsh :: my-super-app × +
 echo "conteúdo qualquer" > my-file
 On branch master
No commits yet
Untracked files:
nothing added to commit but untracked files present (use "git add" to track)
 On branch master
                                               O que está prestes a entrar
No commits yet
                                               no repositório é chamado de
Changes to be committed:
                                               "stage", "staging area" ou "index"
 # / > ■ D:\Git\ebac-talk\my-super-app > /master # = +1
~#@) git commit -m "uma mensagem descritiva aqui"
[master (root-commit) f019806] uma mensagem descritiva aqui
1 file changed, 1 insertion(+)
create mode 100644 my-file
  Author: Sergio Cabral <git@sergiocabral.com>
Date: Tue Nov 16 09:22:52 2021 -0300
  uma mensagem descritiva aqui
```

Exemplo 3.

Há três lugares onde seus arquivos podem estar **TTT**, *The Three Trees* (as três árvores)

# 1

O que está na pasta do seu projeto		Gerenciado pelo sistema operacional.
O que vai para o repositório	Staging Area	Próximo <i>commit</i> ainda <u>não feito</u> .
O repositório		Último <i>commit</i> <u>feito</u> e Pai do próximo <i>commit</i> .

# 2.3. Branches e Tags

Commits sempre são sequências de 40 caracteres (um hash gerado com SHA1).

Por exemplo: f019806e7923e3477da92378afb0175f7f0aefed

Branches e tags são apenas nomes para commits.

Ao apagar branches ou tags nenhum dado é perdido.

### **Criar branches**

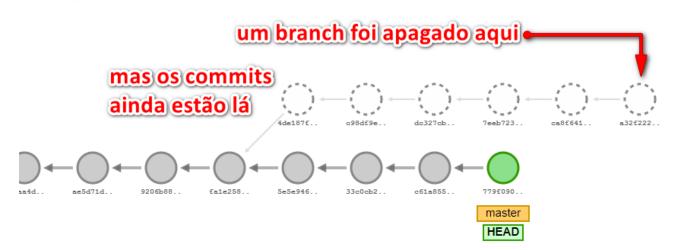
- git branch <nome-do-branch>
- git checkout -b <nome-do-branch>

### **Excluir branches**

• git branch -D <nome-do-branch>



Quando se usa -d (minúsculo) o Git não vai excluir branches que apontam para commits que nunca sofreram *merge*.



Só vai ser possível acessar pelo hash

### Exemplo 4.

### **Criar tags**

• git tag <nome-da-tag> -m "<descrição-opcional>"

### **Excluir tags**

• git tag -d <nome-da-tag>

Branches e tags são semelhantes, mas a diferença está em que:



- Um **branch** acompanha cada novo *commit* feito a partir dele.
- · Uma tag é permanente para um determinado commit na corrente do tempo.