

GIT & GITHUB

Do Básico ao Avançado

Domine Git e GitHub com este material completo. Com ele você aprenderá a gerenciar fluxos de trabalho de maneira simples e dinâmica.



Git Init?

O que é Git & GitHub?

Apesar de serem ferramentas diferentes, ou seja, Git serve para uma coisa e GitHub para outra, ambas são ferramentas que trabalham juntas com um ÚNICO propósito: **VERSIONAR O SEU PROJETO!**

O que é Git?

O Git, nada mais é do que um software que consegue gerenciar as versões do seu projeto. Softwares como esses são popularmente conhecidos como VCS (Version Control System).

Antigamente quando você queria manter as versões antigas do seu código ou do seu sistema como um todo, você basicamente, tinha que ir salvando em diversos backups, todas as alterações, correções e implementações que você fazia no seu código/sistema.

Então poderíamos esperar uma pasta com diversos arquivos como esses:



Versões do meu sistema de e-commerce (backups)



Meu-Commerce-v.1.0.0.zip



Meu-Commerce-v.1.0.2.zip



Meu-Commerce-v.1.0.1.zip



Meu-Commerce-v.1.0.3.zip

Mas por que alguém se preocuparia em manter versões antigas dos seus projetos?

Simple, um dos principais benefícios em salvar suas versões antigas é garantir a proteção do código-fonte do sistema, além de permitir a otimização do processo de desenvolvimento como um todo.



Reparação de Bugs

Supondo que a implementação, ou a alteração de uma funcionalidade não seja muito bem vista pelo usuário final (mercado/personas/público-alvo), com o controle de versão, você consegue regredir a versões anteriores.

“Agora imagina, que você alterou todo o sistema, não chegou a salvar a versão anterior, e o usuário final reclamou! PUFT... você terá o retrabalho de codificar tudo de novo...”

Controle na Arquitetura do Sistema

Hoje em dia, muitas alterações precisam ser implementadas mesmo após a entrega do software, o que implica em novos códigos que por sua vez podem provocar grandes alterações na arquitetura do sistema.



Com o versionamento de código, você consegue acompanhar a evolução do sistema como um todo, encontrando pontos nas quais podemos fazer e aplicar futuras melhorias.

Além disso, o versionamento nos ajuda a evitar duplicações de códigos, tanto aqueles que já foram excluídos, como também aqueles que já existem (o que ocasiona uma duplicação).



Ajuda bastante no trabalho colaborativo

Quanto maior e mais complexo for o sistema, maiores são as chances de ter mais de uma pessoa trabalhando nele.

E quanto mais desenvolvedores trabalhando dentro de um único sistema, maiores são as chances de erros provocados por falhas humanas.

Por vezes, encontramos mais de uma pessoa trabalhando no mesmo arquivo de forma simultânea sabia?



Pedro e João, estão fazendo alterações no mesmo arquivo, que pertence a um projeto.

Por meio do versionamento, eles não precisam estar próximos um do outro para dividir suas tarefas.

Por meio do versionamento de código, é possível reconhecer facilmente qualquer alteração nos programas utilizados, o que torna o trabalho mais produtivo e sem riscos ao sistema como um todo.

O que é o GitHub?

Já o GitHub é uma plataforma online, que nos permite hospedar projetos versionados com o Git.

Imagina uma grande biblioteca de código online, aonde você tanto pode hospedar seus códigos (de forma pública), como também encontrar e ajudar a aprimorar o código de outras pessoas... imaginou?

Legal, né? Essa plataforma existe é conhecida como **GITHUB** =)

Pode me dar um exemplo?

Mais é claro que sim ;)



Vamos imaginar que você está desenvolvendo um projeto, deixe a sua imaginação fluir, pode ser um projeto de um TCC, um sistema web, um aplicativo mobile, um livro... enfim imagine qualquer coisa que envolva projetos digitais.

E toda vez que você salva aquele projeto para começar o trabalho no dia seguinte, pode bater aquela dúvida:



“Como posso controlar aquilo que foi feito nos dias anteriores?”

“Uma vez que eu não quero perder o raciocínio de tudo o que eu fiz até agora”



“Já sei... vou criar diversos arquivos e nomeá-los como: projeto-versão1, projeto-versão2, projeto-versão3 e assim por diante...”

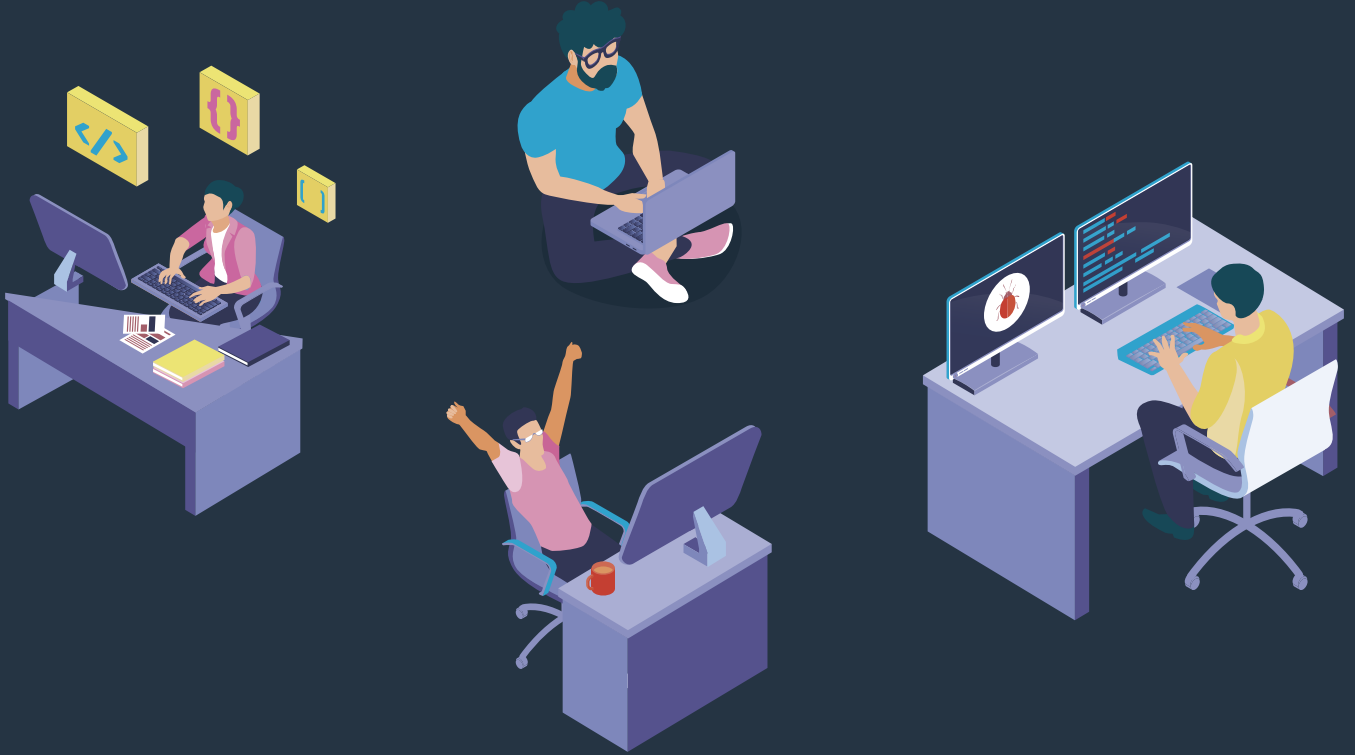


NÃO FAÇA ISSO!

Toda vez que você se encontrar nessa situação, lembre-se do GIT e também do GITHUB!

Com o Git você consegue fazer a versão de cada parte do seu projeto, rastrear todo o histórico de alterações, além de receber uma série de ferramentas que te permitem trabalhar em colaboração com mais pessoas desenvolvendo o mesmo projeto.

Imagina que você e mais alguns amigos
estão desenvolvendo o mesmo projeto
*e que nem sempre terão condições de estarem
no mesmo lugar físico*



O que você pode fazer é...

- ✓ Pegar o código-fonte do seu projeto
- ✓ Hospeda-lo no GitHub (Tornando um projeto compartilhado)
- ✓ Usar as ferramentas do Git e também do GitHub
- ✓ E controlar a versão entre todos os participantes, de modo que não haja duplicações, mantenha a integridade do código e que nenhuma parte dele seja sobrescrita e perdida para sempre.

END (y/n)



Instalação Inicial

Para começarmos a utilizar o Git e o GitHub, primeiro nós precisamos instalar o software do GIT por meio da página oficial deles:

<https://git-scm.com>



Geralmente a opção de Download fica localizada dentro de um ícone de um monitor.

Basta clicar no botão de **Download** para baixar o executável.

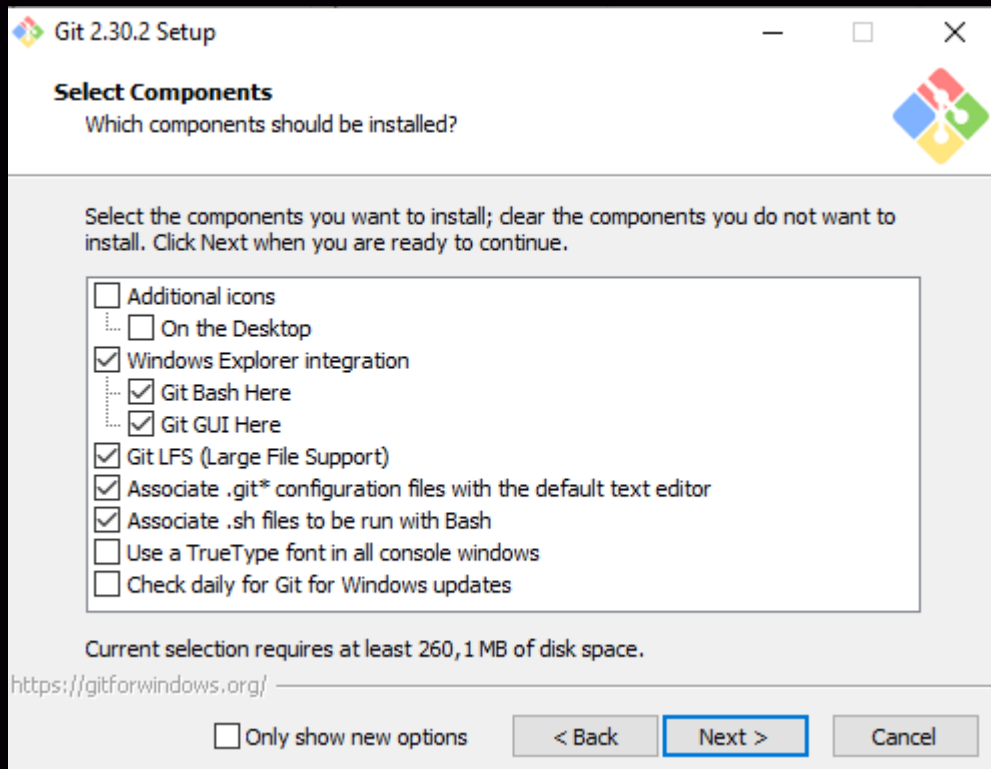
Não se preocupe, o site oficial do GIT reconhece a versão do sistema operacional que você está utilizando.

Após fazer a instalação, execute o arquivo que foi baixado, e já com o instalador, aberto comece aceitando os termos (*Clique em Next*)



1

Clique em **Next** para aceitar os termos de uso.



Mantenha as configurações padrão, e clique em **Next**.

A seguir certifique-se de que essas opções estão devidamente selecionadas (Windows)

Use Vim (the ubiquitous text editor) as Git's default editor

The [Vim editor](#), while powerful, [can be hard to use](#). Its user interface is unintuitive and its key bindings are awkward.

● Let Git decide

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project [intends](#) to change this default to a more inclusive name in the near future.

● Git from the command line and also from 3rd-party software

(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

● Use the OpenSSL library

Server certificates will be validated using the ca-bundle.crt file.

● Checkout Windows-style, commit Unix-style line endings

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

● Use MinTTY (the default terminal of MSYS2)

Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via ``winpty`` to work in MinTTY.

● Default (fast-forward or merge)

This is the standard behavior of ``git pull``: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

● Git Credential Manager Core

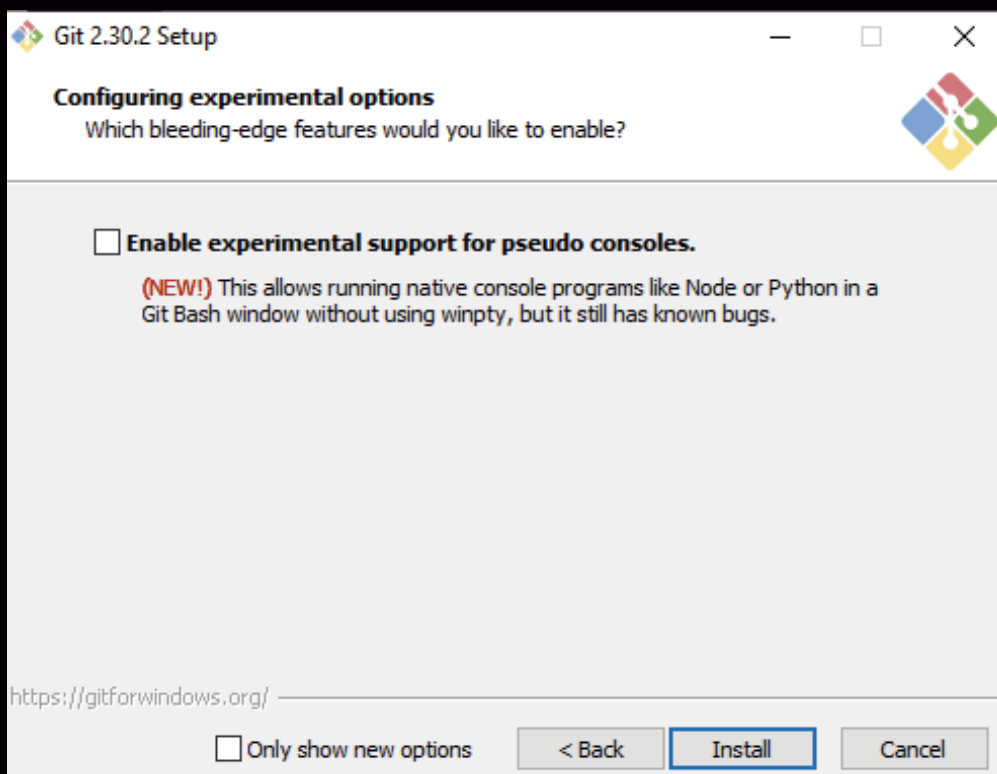
(NEW!) Use the new, [cross-platform version of the Git Credential Manager](#). See more information about the future of Git Credential Manager [here](#).

☒ Enable file system caching

File system data will be read in bulk and cached in memory for certain operations (`"core.fscache"` is set to `"true"`). This provides a significant performance boost.

☐ Enable symbolic links

Enable [symbolic links](#) (requires the `SeCreateSymbolicLink` permission). Please note that existing repositories are unaffected by this setting.



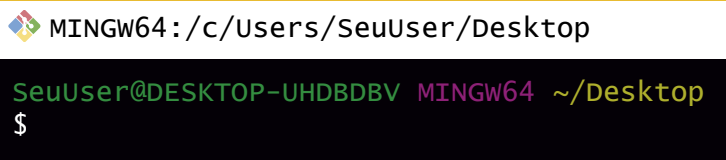
3

Por fim, clique em **Install** para iniciar o processo de instalação do GIT.

Após a Instalação do Git

O Git é um software de linha de comando, e isso significa dizer que para acessarmos ele, precisamos fazer isso por um terminal de comando.

Existem 3 formas básicas de acessarmos o Git via terminal, são elas:

A screenshot of a MINGW64 terminal window. The title bar shows the MINGW64 logo and the path "/c/Users/SeuUser/Desktop". The terminal content shows the prompt "SeuUser@DESKTOP-UHDBDBV MINGW64 ~/Desktop" followed by a dollar sign "\$".

```
MINGW64: /c/Users/SeuUser/Desktop
```

#FORMA 1

Pelo Próprio

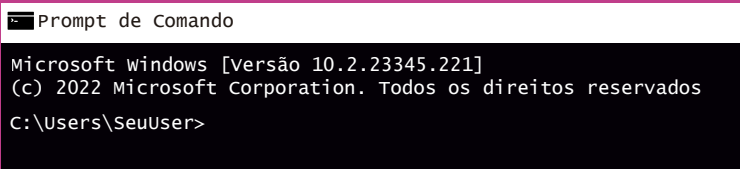
Após a instalação do Git, uma nova funcionalidade será adicionada no menu do mouse, chamada de **Git Bash Here**, ao clicar nesta opção um terminal de código será aberto, e você estará pronto para executar os scripts do GIT.

MINGW64 é um ambiente de desenvolvimento de código que é usado pelo GIT, basicamente é um terminal de comando (shell) simples, que suporta o padrão POSIX, o que o possibilita a execução de scripts autoconf.

É recomendável que você sempre acione esta funcionalidade, dentro da pasta raiz do seu projeto, ou seja, o local aonde estão os arquivos que você quer fazer upload para o GitHub, ok?

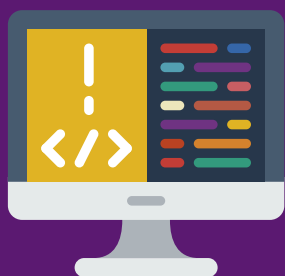
#FORMA 2

Pelo Prompt

A screenshot of a Windows Command Prompt window. The title bar says "Prompt de Comando". The content shows "Microsoft Windows [Versão 10.2.23345.221]" and "(c) 2022 Microsoft Corporation. Todos os direitos reservados". The prompt is "C:\Users\SeuUser>".

```
Prompt de Comando
```

Sim, apesar do Git vir instalado com seu próprio terminal, nós também podemos utilizar a linha de comando padrão do Windows, chamada "**Prompt de Comando / Cmd**" para utilizar o Git. O processo é bem simples, só digitar **git** e ser feliz =)



#FORMA 3

Por meio de Editores de

Famosas IDEs, ou ambientes de desenvolvimento integrados, são na verdade editores de código fonte que auxiliam na

criação e edição de códigos, por meio de uma interface capaz de destacar sintaxes com indicadores visuais, preenchimento automático, verificação de bugs e muito mais!

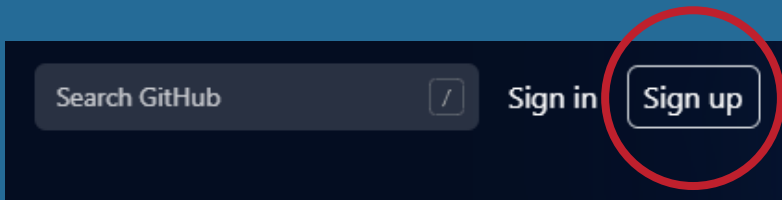


Alguns editores de código contam com terminais integrados, como é o caso do Visual Studio Code, clique aqui para saber [como usar a integração do Git no Visual Studio Code](#)



Criando uma conta
no GitHub

Acesse: <https://github.com>



#1: Clique no botão
Sign Up.

#2: Preencha alguns
de seus dados, como:

- Username (nome de usuário)
- Email Address (seu e-mail de acesso)
- Password (sua senha de acesso)

Por fim, clique em
Create Account para
criar uma nova
conta!

Join GitHub

Create your account

Username *

Email address *

Password *

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Email preferences

☐ Send me occasional product updates, announcements, and offers.

Verify your account



Create account

By creating an account, you agree to the [Terms of Service](#). For more information about GitHub's privacy practices, see the [GitHub Privacy Statement](#). We'll occasionally send you account-related emails.



Create a website with
GitHub Pages



Collaborating with my team



Find and contribute to
open source



School work and student
projects



Use the GitHub API



Other

I am interested in:

languages, frameworks, industries

We'll connect you with communities and projects that fit your interests.

For example: `auth0` `llvm` `cli`

[Complete setup](#)

Antes do GitHub te dar acesso a sua conta você precisa fazer duas coisas...

- 1° Preencher seus interesses e clicar em Complete Setup.
- 2° Confirmar seu endereço de e-mail.

What do you want to do first?

Every developer needs to configure their environment, so let's get your GitHub experience optimized for you.



Start a new project

Start a new repository or bring over an existing repository to keep contributing to it.

Create a repository



Collaborate with your team

Improve the way your team works together and get access to more features with an organization.

Create an organization



Learn how to use GitHub

Get started with an "Introduction to GitHub" course in our Learning Lab.

Start Learning

[Skip this for now >](#)

O GitHub te dará algumas opções sobre o que você pode fazer agora, mas por hora, vamos clicar em **Skip this for now >**

E pronto, seja bem-vindo ao GitHub =)

E agora, o que fazer?

Parabéns por ter chegado até aqui, foi uma jornada leve, simples e bem prática não acha? Neste módulo do e-book você aprendeu um pouco sobre:

- * O que é Git e GitHub.
- * Benefícios do versionamento de Código.
- * Como configurar e instalar o Git.
- * Como criar uma conta no GitHub.

No próximo módulo vamos colocar a mão na massa, aprendendo a criar repositórios, enviar e fazer download de códigos e algumas outras coisas a mais, vamos nessa?

END (y/n)