



# Git Init Módulo I - Introdução ao Git e Github

12/04/2023

---

Alexis Comesana Silvera  
UDESC - Sistemas de Informação  
São Bento do Sul - Santa Catarina

# SUMÁRIO

<b>Módulo I - Introdução ao Git e Github</b>	<b>2</b>
O que é o Git?	2
Por que é importante para programadores?	4
O que é Github?	5
Como instalar e configurar o Git no seu computador?	6
Windows	6
Linux	7
Mac	7
Impedimentos	8
Configuração inicial	8
Criando uma conta no Github	8
Criando um repositório no Github	11
Clonando seu repositório para seu computador	13

## Módulo I - Introdução ao Git e Github

### O que é o Git?

No início dos anos 2000, Linus Torvalds desenvolveu o sistema operacional de código aberto Linux. Enquanto trabalhava no desenvolvimento do Linux, Torvalds começou a ter problemas com as ferramentas de controle de versão de software disponíveis na época. Ele sentiu que elas eram muito lentas e complicadas para trabalhar em um projeto grande como o Linux.

Frustrado com as ferramentas existentes, Torvalds decidiu criar sua própria ferramenta de controle de versão de software que seria rápida e fácil de usar. Ele chamou sua nova ferramenta de "**Git**" e começou a usá-la para gerenciar o código-fonte do Linux.

Ao longo do tempo, o Git se tornou cada vez mais popular entre os programadores e desenvolvedores de software de todo o mundo, graças à sua facilidade de uso e recursos avançados de controle de versão. O Git é agora amplamente considerado como uma das melhores ferramentas de controle de versão de software disponíveis e é amplamente utilizado em projetos de todos os tamanhos e complexidades.

O Git é uma ferramenta de controle de versão de software que permite que programadores gerenciem e rastreiem as alterações feitas em um

código-fonte ao longo do tempo. Em outras palavras, o Git é como um caderno que registra tudo o que foi feito em um projeto, desde as primeiras linhas de código até a versão final do software.

Imagine que você está trabalhando em um projeto de programação com outras pessoas. Cada um faz alterações no código-fonte para adicionar novos recursos, corrigir erros ou melhorar o desempenho do software. O Git permite que todos os envolvidos no projeto trabalhem juntos, sem se preocupar em sobrescrever o trabalho um do outro ou perder alterações importantes.

O Git registra todas as alterações feitas no código-fonte em um repositório, que é como um arquivo grande que contém todas as versões do projeto. O repositório é então hospedado em um servidor online, como o Github, para que todos os envolvidos no projeto possam acessá-lo e trabalhar nele.

Ao trabalhar com o Git, você pode criar diferentes versões do seu código-fonte, cada uma delas com suas próprias alterações e melhorias. Se algo der errado em uma versão do código, você pode facilmente voltar para uma versão anterior que estava funcionando corretamente.

Além disso, o Git permite que você trabalhe em diferentes partes do código-fonte ao mesmo tempo. Você pode criar diferentes "ramos" ou **"branches"** do projeto, cada um com suas próprias alterações, e depois fundi-los de volta na versão principal quando estiverem prontos. Isso é útil quando você precisa trabalhar em diferentes partes do código simultaneamente, sem interferir no trabalho de outras pessoas.

Em resumo, o Git é uma ferramenta poderosa para gerenciar projetos de programação e colaboração em equipe. Ele permite que você rastreie as alterações feitas no código-fonte ao longo do tempo, trabalhe em diferentes partes do código simultaneamente, e reverta facilmente para versões anteriores quando necessário.

## Por que é importante para programadores?

Agora que você já sabe o que é o Git, vamos entender por que essa ferramenta é tão importante para programadores.

O Git permite que você trabalhe em equipe de forma mais eficiente e produtiva. Sem ele, trabalhar em um projeto em equipe pode ser muito difícil e confuso, com várias pessoas trabalhando em diferentes partes do código-fonte, fazendo alterações simultâneas e correndo o risco de sobrescrever o trabalho um do outro.

Com o Git, cada membro da equipe pode fazer alterações no código-fonte em seu próprio **"branch"** (ramificação) sem interferir no trabalho dos outros. Quando estiver pronto para enviar suas alterações, você pode mesclá-las (ou seja, combiná-las) com o branch principal do projeto usando o comando **"merge"**. Isso ajuda a garantir que cada alteração seja cuidadosamente revisada e testada antes de ser incorporada ao projeto.

Além disso, o Git permite que você acompanhe todo o histórico de alterações em um projeto, permitindo que você veja quando e por quem as alterações foram feitas e o que foi alterado em cada commit. Isso pode ajudar a identificar erros e problemas no código-fonte mais rapidamente e a rastrear o progresso do projeto ao longo do tempo.

Outra vantagem do Git é a facilidade de colaboração em projetos de código aberto. O Github, por exemplo, permite que desenvolvedores de todo o mundo trabalhem juntos em projetos de código aberto, permitindo que qualquer pessoa faça alterações no código-fonte e envie um pull request para ser revisado e incorporado ao projeto.

Em resumo, o Git é importante para programadores porque permite que eles trabalhem juntos de forma mais eficiente e produtiva, acompanhem o histórico de alterações em um projeto e colaborem em projetos de código aberto em todo o mundo.

## O que é Github?

Em 2008, Tom Preston-Werner, Chris Wanstrath e PJ Hyett começaram a trabalhar em uma plataforma que permitisse que programadores hospedassem, colaborassem e contribuíssem com projetos de software de forma pública. Eles queriam criar um espaço onde desenvolvedores de todo o mundo pudessem compartilhar código, encontrar soluções para problemas e colaborar em projetos em equipe.

A plataforma que eles criaram foi chamada de Github e, em pouco tempo, se tornou uma das maiores e mais populares comunidades de desenvolvimento de software do mundo. Com o Github, os programadores podem criar locais onde o código é armazenado (repository), trabalhar em equipe, criar solicitações de alteração (pull requests) e muito mais.

Com sua interface amigável e recursos avançados, o Github se tornou uma ferramenta essencial para desenvolvedores de todo o mundo, desde iniciantes até veteranos da indústria. Além disso, o Github é uma plataforma de código aberto que permite que desenvolvedores colaborem e contribuam para projetos de outros desenvolvedores, ajudando a impulsionar a inovação e o progresso da tecnologia.

Existem outras plataformas de hospedagem de repositórios Git além do Github, como GitLab, Bitbucket e SourceForge. Essas plataformas também oferecem recursos de controle de versão e colaboração em equipe, mas cada uma tem suas próprias características e diferenças em relação ao Github.

No entanto, o curso Git Init irá se concentrar no Github como plataforma de hospedagem de repositórios, pois é uma das mais populares e amplamente utilizadas no mercado atualmente. Isso não significa que as outras plataformas sejam menos importantes, mas simplesmente que o curso não irá abordá-las.

## Como instalar e configurar o Git no seu computador?

Para instalar e configurar o Git no seu computador, você precisará seguir alguns passos simples. O processo é semelhante para sistemas operacionais Windows, Linux e Mac, então abordaremos cada um deles abaixo:

### Windows

1. Acesse o site oficial do Git (<https://git-scm.com/downloads>) e faça o download do instalador para Windows.
2. Após o download, execute o instalador clicando duas vezes no arquivo baixado. O instalador solicitará que você concorde com os termos de serviço e permita que o Git seja instalado no seu computador.
3. Durante a instalação, você será solicitado a escolher as opções de configuração. Certifique-se de que as opções **"Git Bash Here"** e **"Git GUI Here"** estejam marcadas, pois elas são ferramentas úteis para trabalhar com o Git.
4. Depois de concluir a instalação, abra o terminal de comando (pressionando **"Windows + R"** e digitando **"cmd"** no campo de pesquisa) e execute o seguinte comando **"git --version"** para verificar se o Git foi instalado corretamente.

Se o Git foi instalado com sucesso, você verá a versão instalada do Git exibida no terminal.

## Linux

1. Abra o terminal de comando e execute o seguinte comando **"sudo apt-get update"** para atualizar o gerenciador de pacotes do Linux.
2. Após atualizar o gerenciador de pacotes, execute o seguinte comando **"sudo apt-get install git"** para instalar o Git.
3. Após a instalação, execute o seguinte comando **"git --version"** para verificar se o Git foi instalado corretamente.

Se o Git foi instalado com sucesso, você verá a versão instalada do Git exibida no terminal.

## Mac

1. Acesse o site oficial do Git (<https://git-scm.com/downloads>) e faça o download do instalador para Mac.
2. Após o download, execute o instalador clicando duas vezes no arquivo baixado. O instalador solicitará que você concorde com os termos de serviço e permita que o Git seja instalado no seu computador.
3. Durante a instalação, certifique-se de selecionar a opção **"Use Git from the command line and also from 3rd-party software"**, pois ela permitirá que você use o Git a partir do terminal.
4. Depois de concluir a instalação, abra o terminal de comando (pressionando **"Command + Espaço"** e digitando **"Terminal"** no campo de pesquisa) e execute o seguinte comando **"git --version"** para verificar se o Git foi instalado corretamente.

Se o Git foi instalado com sucesso, você verá a versão instalada do Git exibida no terminal.



## Impedimentos

Durante a instalação do Git, podem surgir erros que impedem o processo de ser concluído com sucesso. Felizmente, muitos desses problemas já foram enfrentados por outros usuários e existem diversas soluções disponíveis online. Para encontrar soluções para um erro específico, basta pesquisar na internet com palavras-chave relacionadas ao erro, geralmente encontrado no próprio terminal e adicionar a palavra **"Git"** ao final da pesquisa. Isso pode ajudar a encontrar fóruns, tutoriais ou vídeos que ofereçam soluções e dicas úteis para resolver o problema.

## Configuração inicial

Após instalar o Git, é importante configurá-lo com as suas informações pessoais, como nome de usuário e endereço de e-mail. Isso é necessário para que o Git possa identificar quem está fazendo as alterações no código. Para fazer isso, abra o terminal de comando e execute os seguintes comandos **"git config --global user.name seu\_nome"** e **"git config --global user.email seu\_email"**, substituindo **"seu\_nome"** e **"seu\_email"** pelos seus próprios dados.

Para conferir se seu usuário foi configurado corretamente, basta executar o seguinte comando **"git config --list"** e conferir as linhas que começam com **"user.name=NOME"** e **"user.email=EMAIL"**.

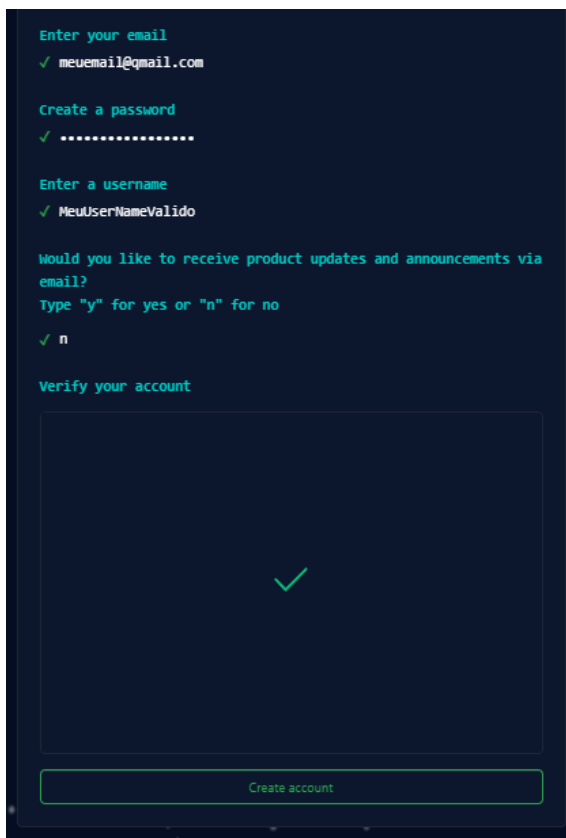
## Criando uma conta no Github

Para começar a usar o Github, você precisa criar uma conta, para isso você pode seguir passo a passo abaixo:

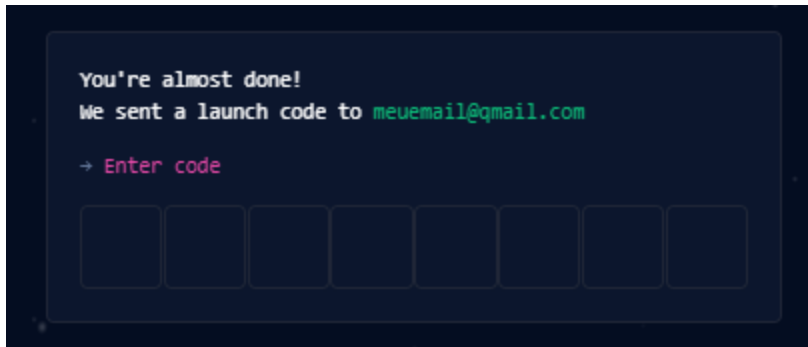
1. Acesse o site do Github em [www.github.com](https://www.github.com).
2. Clique no botão **"Sign up"** no canto superior direito da página inicial.



3. Preencha seus dados de registro, incluindo seu nome de usuário, endereço de e-mail e uma senha segura. Certifique-se de usar um endereço de e-mail válido e uma senha forte para proteger sua conta.

A screenshot of the GitHub account creation form. The form is dark-themed and shows several sections with green checkmarks indicating successful validation: 'Enter your email' with 'meuemail@gmail.com', 'Create a password' with a masked password, 'Enter a username' with 'MeuUserNameValido', and a confirmation step asking 'Would you like to receive product updates and announcements via email?' with a response of 'n'. At the bottom, there is a 'Verify your account' section with a large green checkmark and a 'Create account' button.

4. Clique em "**Create account**".
5. O Github irá solicitar que você escolha um plano de assinatura. Para a maioria dos usuários, o plano gratuito é suficiente. Selecione o plano gratuito e clique em "**Continue**".
6. O Github irá solicitar que você verifique seu endereço de e-mail. Verifique sua caixa de entrada e clique no link de verificação enviado pelo Github.



7. Após verificar seu endereço de e-mail, você pode personalizar seu perfil do Github adicionando uma foto e uma descrição.

Pronto! Agora você tem uma conta no Github e pode começar a explorar os recursos e projetos disponíveis na plataforma. Você pode seguir outros usuários, contribuir para projetos, criar seus próprios projetos e muito mais.

## Criando um repositório no Github

Criar um repositório no Github é uma das primeiras coisas que você precisa fazer para começar a usar esta plataforma. Um repositório é um espaço onde você pode armazenar seus projetos e compartilhá-los com outras pessoas.

Para criar um repositório no Github, siga estes passos:

1. Faça login na sua conta do Github.
2. No canto superior direito da página, clique no botão "+", e depois selecione **"New Repository"** (ou **"Novo repositório"**, se estiver em português).
3. Na página **"Create a new repository"** (ou **"Criar um novo repositório"**), você precisa escolher um nome para o seu repositório. Escolha um nome que seja descritivo e fácil de lembrar. Você também pode adicionar uma descrição do seu projeto, se quiser.
4. Se você quer que outras pessoas possam ver e contribuir para o seu projeto, marque a caixa **"Public"** (ou **"Público"**). Se quiser que o projeto seja privado, escolha **"Private"** (ou **"Privado"**). Note que para ter repositórios privados no Github, você precisa ter uma conta paga.
5. Adicione um arquivo README ao seu repositório, marque a caixa **"Add a README file"** (ou **"Adicionar um arquivo README"**). O README é um arquivo que descreve o seu projeto e pode ajudar outras pessoas a entender do que se trata.
6. Escolha a opção de licença que deseja usar no seu projeto. A licença determina como outras pessoas podem usar o seu código-fonte. Se você não tiver certeza de qual licença usar, deixe a opção padrão.
7. Clique no botão **"Create Repository"** (ou **"Criar repositório"**).

Seu repositório foi criado no Github com sucesso! Agora você pode começar a adicionar arquivos ao seu projeto e compartilhá-lo com outras pessoas.

## Clonando seu repositório para seu computador

Agora que você já criou um repositório no Github, é hora de fazer seu primeiro commit. Mas antes disso, é necessário clonar o repositório em seu computador. Clonar um repositório significa criar uma cópia local dele em seu computador. Isso permite que você faça alterações em seus arquivos e os envie de volta para o repositório no Github.

Para clonar o repositório que você acabou de criar, siga estes passos:

1. Abra o terminal (ou prompt de comando) no seu computador. No Windows, você pode fazer isso pressionando a tecla **"Windows" + "R"** e digitando **"cmd"** na caixa de diálogo que aparece. No Linux e Mac, você pode pressionar **"Ctrl" + "Alt" + "T"** para abrir o terminal.
2. Navegue até o diretório onde deseja clonar o repositório usando o comando **"cd"**. Por exemplo, se você quiser clonar o repositório em sua pasta de documentos, digite **"cd Documents"** e pressione Enter.
3. Agora, vamos clonar o repositório. Para fazer isso, vá para a página do repositório no Github e clique no botão verde **"Code"** (ou **"Código"**, se estiver em português).
4. Na janela que aparece, selecione o HTTPS e copie o link fornecido.
5. Volte para o terminal e digite o comando **"git clone"** seguido do link do repositório que você copiou anteriormente. Por exemplo, digite **"git clone https://github.com/seu-usuario/nome-do-repositorio.git"**.
6. Pressione Enter e o Git começará a baixar o repositório para o diretório que você navegou anteriormente.

Agora você tem uma cópia local do seu repositório do Github no seu computador. Você pode acessar e editar os arquivos do repositório usando o seu editor de código ou qualquer outra ferramenta que você preferir.

Caso ja tenha feito tudo isso e ainda quer mais e não consegue esperar pelo segundo modulo, aqui vai uma palavra magica que serve pra quase tudo, a palavra é "Cheat Sheet", com essa palavra você pode encontrar todos os comandos que precisa, basta colocar o nome do programa antes ou depois da palavra, aqui vai um pouco do que pode encontrar se digitar no google "Git Cheat Sheet"

[https://training.github.com/downloads/pt\\_BR/github-git-cheat-sheet/](https://training.github.com/downloads/pt_BR/github-git-cheat-sheet/)