
GitHub

O GitHub é uma plataforma de hospedagem de código-fonte baseada na web, que usa o sistema de controle de versão Git. É amplamente utilizado por desenvolvedores e equipes para colaborar em projetos de software, gerenciar e revisar código, e monitorar o progresso do trabalho.

Aqui estão algumas das principais funcionalidades do GitHub:

- **Controle de versão:** Mantém o histórico completo de todas as alterações feitas no código.
- **Colaboração:** Permite que várias pessoas trabalhem no mesmo projeto, facilitando a comunicação e a gestão de tarefas.
- **Pull Requests:** Ferramenta para discutir e revisar alterações no código antes de integrá-las ao projeto principal.
- **Issues:** Sistema de rastreamento de bugs e gerenciamento de tarefas.
- **GitHub Actions:** Automação de fluxos de trabalho, como testes e implantação contínua.

Guia de Instalação e Operações no GitHub

Para macOS:

Instalação do Git

1. Verifique se o Git está instalado:

git --version

2. Instale o Git (se necessário):

2.1 /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

2.2 brew install git

Configuração do Git Mac

1. Configure seu nome de usuário e e-mail:

git config --global user.name "Seu Nome"

git config --global user.email "seu.email@example.com"

2. Verifique as configurações:

git config --list

Instalação do Git no Windows

1. **Baixe o instalador do Git:**
- Acesse o site oficial do Git: git-scm.com e clique em "Download for Windows".
2. **Execute o instalador:**
- Após o download, abra o arquivo do instalador. Siga as instruções na tela. Durante a instalação, você terá várias opções de configuração. As opções padrão geralmente são suficientes, mas você pode ajustar conforme suas preferências.
3. **Verifique a instalação:**
- Após a instalação, abra o Git Bash (que é instalado junto com o Git) e verifique se o Git está instalado corretamente digitando:
    - git --version

Configuração do Git

1. **Configurar nome de usuário e e-mail:**
   - Antes de começar a usar o Git, configure seu nome de usuário e e-mail que serão usados nas suas operações de commit.

git config --global user.name "Seu Nome"

git config --global user.email "seu.email@example.com"

Ex:

git config --global user.name "Bruno"

git config --global user.email "goncalves.bruno@gmail.com"

Para que serve: Este comando define o nome e o e-mail que serão associados aos seus commits. --global significa que essas configurações serão aplicadas a todos os seus repositórios no sistema.

Verificar as configurações

Pode verificar suas configurações atuais do Git com:

git config --list

Para que serve: Este comando lista todas as configurações atuais do Git.

Criar uma Conta no GitHub

1. Acesse o Site do GitHub:
   - Abra seu navegador e acesse https://github.com.
2. Clique em "Sign up":
   - Na página inicial, clique em Sign up no canto superior direito.
3. Preencha o Formulário de Cadastro:
   - Username: Escolha um nome de usuário único.
   - Email address: Insira seu e-mail.
   - Password: Crie uma senha segura.
   - Verify your account: Complete o CAPTCHA.

Finalize o Cadastro:

- Siga as instruções na tela para finalizar a criação da sua conta.Pode ser solicitado a verificar seu e-mail.

Gerar e Configurar uma Chave SSH(Secure Shell)

É um protocolo de rede criptográfico usado para operar serviços de rede de forma segura em uma rede não segura. Ele fornece um canal seguro sobre uma rede insegura em uma arquitetura cliente-servidor, conectando um cliente SSH a um servidor SSH.

Principais Funcionalidades do SSH:

Autenticação Criptografada,Transferência Segura de Arquivos,Acesso Remoto Seguro.

Componentes Principais:

1. Cliente SSH:
    - O software usado pelo usuário para fazer a conexão segura ao servidor SSH. No terminal, isso é feito com o comando ssh.
2. Servidor SSH:
    - O software que aceita conexões SSH do cliente. Ele autentica o cliente e estabelece o canal criptografado.

Gerar uma Nova Chave SSH

1- Abra o Git Bash (Windows) ou o Terminal (macOS/Linux).

2- Digite o seguinte comando para gerar uma nova chave SSH:

ssh-keygen -t ed25519 -C "seu.email@example.com"

3- Se você estiver usando uma versão mais antiga do SSH, pode usar:

ssh-keygen -t rsa -b 4096 -C "seu.email@example.com"

Substitua **seu.email@example.com** pelo seu e-mail. **Quando solicitado a especificar um local para salvar a chave, pressione Enter** para aceitar o local **padrão (/home/usuario/.ssh/id_ed25519 ou /home/usuario/.ssh/id_rsa).**

Adicionar a Chave SSH ao Agente SSH

1- Inicie o agente SSH:

```
eval "$(ssh-agent -s)"
```

2- Adicione a chave SSH ao agente:

```
ssh-add ~/.ssh/id_ed25519
```

3- Se você usou RSA:

```
ssh-add ~/.ssh/id_rsa
```

Copiar a Chave SSH para o GitHub:

1-Copie a chave pública SSH para a área de transferência:

windows:

```
cat ~/.ssh/id_ed25519.pub | clip
```

Mac:

```
cat ~/.ssh/id_ed25519.pub | pbcopy
```

2-Ou, se você usou RSA:

```
cat ~/.ssh/id_rsa.pub | clip
```

Adicionar a Chave SSH ao seu Perfil do GitHub

1. Acesse GitHub e faça login.
2. No canto superior direito, clique na sua **foto de perfil** e vá em Settings.
3. No menu à esquerda, clique em **SSH and GPG keys.**
4. Clique em **New SSH key.**
5. Cole a chave pública copiada e dê um título para identificar a chave.
6. Clique em **Add SSH key.**

Testar a Conexão SSH

1- Para verificar se a configuração está correta, você pode testar a conexão com o GitHub:

ssh -T git@github.com

2- Deve ver uma mensagem de sucesso:

Hi seu-usuario! You've successfully authenticated, but GitHub does not provide shell access.

MacOS:Encontrar a chave SSH existente

- Abra o Terminal.
- Navegue até o diretório .ssh:

cd ~/.ssh

- Liste os arquivos:

ls -la

Encontre os arquivos id_ed25519 (chave privada) e id_ed25519.pub (chave pública).

MacOS:Copiar as chaves para um dispositivo ou email

- **Exiba o conteúdo da chave pública:**

cat id_ed25519.pub

- **Exiba o conteúdo da chave privada:**

cat id_ed25519

- Copie os conteúdos exibidos.
- Abra seu cliente de email.
- Crie um novo email e cole os conteúdos das chaves pública e privada no corpo da mensagem.
- Envie o email para o destino desejado.

Windows:Encontrar a chave SSH existente

- Abra o PowerShell ou Command Prompt (cmd).
- Navegue até o diretório .ssh:

cd ~/.ssh

- Liste os arquivos:

ls -la

Encontre os arquivos id_ed25519 (chave privada) e id_ed25519.pub (chave pública).

Windows:Copiar as chaves para um dispositivo ou email:

- **Exiba o conteúdo da chave pública:**

cat id_ed25519.pub

- **Exiba o conteúdo da chave privada:**

cat id_ed25519

- Copie os conteúdos exibidos.
- Abra seu cliente de email.
- Crie um novo email e cole os conteúdos das chaves pública e privada no corpo da mensagem.
- Envie o email para o destino desejado.

Transferir para outro dispositivo (macOS ou Windows)

Criar a pasta e mover-se para dentro dela:

- Abra o Terminal (macOS) ou PowerShell (Windows).
- Crie o diretório .ssh e navegue até ele:

  mkdir -p ~/.ssh

  cd ~/.ssh

Copiar as chaves para o novo dispositivo:

- Cole a chave privada no arquivo id_ed25519:

  echo "sua-chave-privada-copiada" > id_ed25519

  chmod 600 id_ed25519

Transferir para outro dispositivo (macOS ou Windows)

Cole a chave pública no arquivo id_ed25519.pub:

```
echo "sua-chave-publica-copiada" > id_ed25519.pub
chmod 644 id_ed25519.pub
```

Ajustar as permissões do diretório .ssh:

```
chmod 700 ~/.ssh
```

Verificar se a transferência deu certo:

Testar a autenticação no GitHub:

- Abra o Terminal (macOS) ou PowerShell (Windows).
- Teste a conexão SSH com o GitHub:

  ssh -T git@github.com

- Se a autenticação for bem-sucedida, você verá uma mensagem de boas-vindas do GitHub, algo como:

Hi username! You've successfully authenticated, but GitHub does not provide shell access.

Criar um Repositório Direto no GitHub

1. **Faça Login no GitHub:**
   - Após criar sua conta, faça login usando seu nome de usuário e senha.
2. **Criar um Novo Repositório:**
   - No canto superior direito, clique no ícone de "+" e depois em New repository.
3. **Preencha os Detalhes do Repositório:**
   - **Repository name:** Dê um nome ao seu repositório (por exemplo, meu-projeto).
   - **Description:** Adicione uma descrição opcional.
   - **Public/Private:** Escolha se o repositório será público ou privado.
   - **Initialize this repository with a README:** Marque essa opção se quiser que um arquivo README seja criado automaticamente.
   - Clique em **Create repository.**

4. **Adicionar Arquivos ou Pastas:**

- No repositório recém-criado, clique em **Add file > Upload files.**
- **Arraste e solte arquivos ou clique em choose your files** para selecionar os arquivos que deseja enviar.
- Adicione **uma mensagem de commit** descrevendo os arquivos enviados e clique em **Commit changes.**

Criar um Diretório pelo Terminal e Colocar no GitHub Remoto

1. **Abrar o Git Bash ou Terminal no projeto, e garanta que o branch será main:**

   git config --global init.defaultBranch main

2. **Criar um Novo Diretório e Inicializar um Repositório Local:**

- Abra o Git Bash ou Terminal.
- Crie um novo diretório e inicialize um repositório Git:

mkdir meu-projeto

cd meu-projeto

git init

CASO PRECISE REINICIAR CONFIGURAÇÃO GIT: rm -rf .git

3. Verifique a branch atual:

git branch

4. Crie todo o seu projeto, após finalizar

5. Adicione os arquivos ao índice

git add .

6. Fazer um Commit das Mudanças:

git commit -m "projeto criado a partir do terminal"

7. Criar um Repositório Remoto no GitHub:

- Acesse o GitHub e crie um novo repositório (sem inicializar com README).

8. Conectar o Repositório Local ao Repositório Remoto:

- No Terminal, conecte o repositório local ao remoto:

git remote add origin git@github.com:seu-usuario/meu-projeto.git

9. Envie o Repositório Local para o GitHub:

git push -u origin main

Operações Básicas com Git

1. **Clonar um repositório:**

- Para clonar um repositório do GitHub, use:

git clone git@github.com:usuario/repo.git

Exemplo:Bruno quer clonar um repositório chamado "meu-projeto" do GitHub.

git clone git@github.com:bruno/meu-projeto.git

Para que serve: Este comando cria uma cópia do repositório remoto no seu sistema local.

2. **Adicionar mudanças:**

- Para adicionar mudanças ao índice (staging area), use:

git add .

Exemplo: Bruno fez alterações no projeto e quer adicionar todos os arquivos modificados para serem incluídos no próximo commit.

git add .

Para que serve: Este comando adiciona todas as mudanças (arquivos novos, modificados e deletados) ao índice para serem incluídas no próximo commit.

3. Commitar mudanças:

- Para gravar as mudanças adicionadas no repositório local, use:

git commit -m "Mensagem do commit"

Exemplo: Bruno adicionou as mudanças ao índice e agora quer gravá-las no repositório local com uma mensagem de commit.

git commit -m "Adicionar nova funcionalidade"

Para que serve: Este comando grava as mudanças no repositório local. A mensagem de commit deve descrever as mudanças realizadas.

4. **Enviar mudanças (push):**

- Para enviar os commits do repositório local para o repositório remoto, use:

git push origin branch

Exemplo: Bruno quer enviar seus commits do repositório local para o repositório remoto na branch principal (main).

git push origin main

Para que serve: Este comando envia suas mudanças para o repositório remoto. origin é o nome do repositório remoto e branch é o nome da branch para onde você está enviando as mudanças.

5. **Atualizar o repositório local (pull):**

- Para atualizar seu repositório local com mudanças do repositório remoto, use:

git pull origin branch

Exemplo: Bruno quer atualizar seu repositório local com as últimas mudanças do repositório remoto na branch principal (main).

git pull origin main

Para que serve: Este comando busca e integra mudanças do repositório remoto na sua branch local

Exemplo de Uso Completo

Imagine que Bruno está trabalhando em um projeto chamado meu-projeto. Aqui está um exemplo de todas as operações básicas juntas:

1. **Clonar o repositório:**

git clone git@github.com:bruno/meu-projeto.git

2. **Fazer algumas mudanças no projeto e adicionar arquivos modificados:**

cd meu-projeto

Fazendo algumas mudanças nos arquivos do projeto

git add .

3. **Commitar as mudanças com uma mensagem:**

git commit -m "Corrigir bug na funcionalidade de login"

4. **Enviar as mudanças para o repositório remoto:**

git push origin main

5. **Atualizar o repositório local com as últimas mudanças do remoto:**

git pull origin main