

Manual de Sobrevivência do Git

Material de Complementar



Objetivos de Aprendizagem

Ao final deste módulo, esperamos que você seja capaz de:

1. Analisar as diferenças entre GIT e GITHUB.
2. Refletir sobre a importância das ferramentas GIT e GITHUB para o desenvolvedor de software.
3. Fazer download e configurar a ferramenta.
4. Identificar os principais comandos.



Diferenças entre GIT e GITHUB

Olá, alune!

Seja bem-vinde ao módulo de GIT, neste material complementar vamos refletir acerca das principais dúvidas sobre GIT e GITHUB, além de trazer muito material bacana para aprofundar ainda mais o seu conhecimento. Preparede?

Vamos lá!

Git e Github



Fonte: Gabsferreira, 2022.

Antes de mais nada: GIT e GITHUB são coisas diferentes!

- **GIT:** Serviço de controle de versões (com comandos específicos seja por linha de comando ou por interface gráfica).



- **GITHUB:** Repositório de códigos público disponível na nuvem que usa GIT como seu serviço de controle de versões. Existem outros que também usam GIT, tais como Bitbucket, GitLab, entre outros.

Pois bem, de posse disto, bora lá categorizar alguns comandos que você vai utilizar pouco e outros que você vai utilizar muito.



Hora de aprofundar um pouco mais o conhecimento!
[Clique aqui](#) e acesse uma vídeo aula do Prof. Isidro, onde explica um pouco mais sobre essas diferenças.

Como criar uma conta no GITHUB

O GitHub permite você criar uma conta gratuita, para isso, basta acessar o site e seguir as instruções na tela para criar uma conta de usuário.

A conta pessoa serve como sua identidade, ou em uma organização, o que permite que várias contas pessoais colaborem em múltiplos projetos.



Para utilizar o GITHUB você obrigatoriamente precisa criar uma conta [clikando aqui](#).
Depois é só seguir o passo a passo do site.



Como utilizar o GIT

Agora que você já sabe como utilizar o GitHub, vamos falar um pouco sobre o GIT. Lembre-se que são ferramentas diferentes, hein?!

Git é o sistema de controle de versão open source mais usado no mundo atualmente! Ele é usado para controlar o histórico de alterações de arquivos e principalmente de projetos de desenvolvimento de software. Ele permite mais flexibilidade no fluxo de trabalho, segurança e desempenho.



Download

Fazendo o Download das ferramentas do GIT Simples também, [clique aqui](#) e baixe para o seu computador.

Comandos de SETUP do Usuário e Máquina

Para auxiliar sua operação do dia-a-dia, você precisará “identificar” sua máquina junto ao Github e ter uma chave RSA para que isso seja efetivo. Isso você deverá fazer apenas uma vez e só terá que fazer novamente se mudar de máquina.



Configurando seus dados locais:

Definindo seu nome de usuário:

```
git config --global user.name "Seu nome cadastrado no Github"
```

```
git config --global user.email "Seu email do Github"
```

Criando sua chave:

Usando um client SSH (se você tem Windows, um programa bem legal pra fazer isso é o [PUTTY](#)).

```
ssh-keygen -t rsa -b 4096 -C "seu email do github"
```

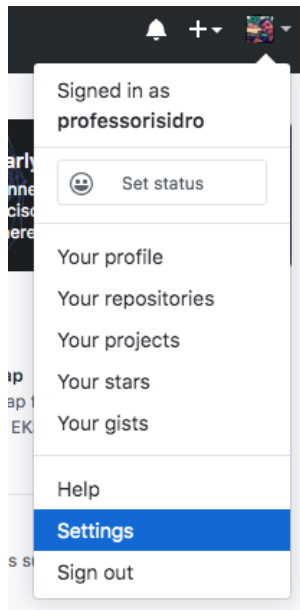
Neste caso, criamos uma chave RSA com 4096 bits a partir do email do github fornecido.

Inserindo sua chave nas suas configurações do GitHub

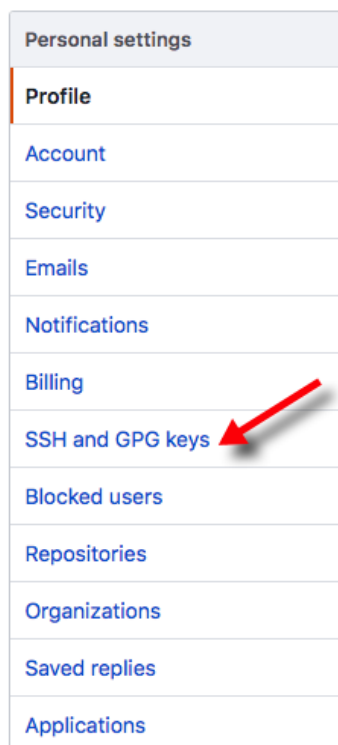
Uma vez feito isso, haverá uma pasta (na sua área de usuário) chamada .ssh. Lá dentro, há um arquivo id_rsa.pub. É neste arquivo que terá sua chave. Ela inicia com "rsa-ssh" e termina com o email que você cadastrou. Copie esse conteúdo.

Agora, vá na sua conta do Github, no link "settings" (como na imagem abaixo)





Em seguida, vá no link "SSH and GPG Keys" assim como na figura





Se você já tiver alguma chave cadastrada, ok, entretanto é necessário criar uma nova chave.

SSH keys

[New SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

 SSH	Machos de Isidro 20:06:12:29:82:77:cd:4c2b:86:fe:1f:8e:78:18:55:89 Added on 16 Aug 2019 Last used within the last week --- Read/write	Delete
 SSH	Machos de Isidro 7c:8b:11:ca:77:ed:15:a1:58:1e:5d:73:ed:17:1f:11:11:32 Added on 6 Sep 2019 Last used within the last week --- Read/write	Delete

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#).

GPG keys

[New GPG key](#)

There are no GPG keys associated with your account.

Learn how to [generate a GPG key and add it to your account](#).

Lá dentro, apenas dê um nome a ela (eu costumo identificar as minhas máquinas) e depois cole o conteúdo que você copiou lá do id_rsa.pub no campo KEY.

SSH keys / Add new

Title

Key

Begins with 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

[Add SSH key](#)

Beleza! Se estiver tudo ok, você só vai precisar testar da seguinte maneira:

```
ssh -T git@github.com
```



Se tudo ocorrer bem, você receberá uma mensagem assim:

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

Comandos de SETUP do seu Repositório

De todo jeito você deverá criar seu repositório lá no GITHUB.


Basicamente o link é este [aqui](#) e vai carregar esta tela aqui:

Create a new repository

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

Owner

Repository name *

 professorisidro ▾

/

Great repository names are short and memorable. Need inspiration? How about **stunning-spork**?

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▾

Add a license: **None** ▾

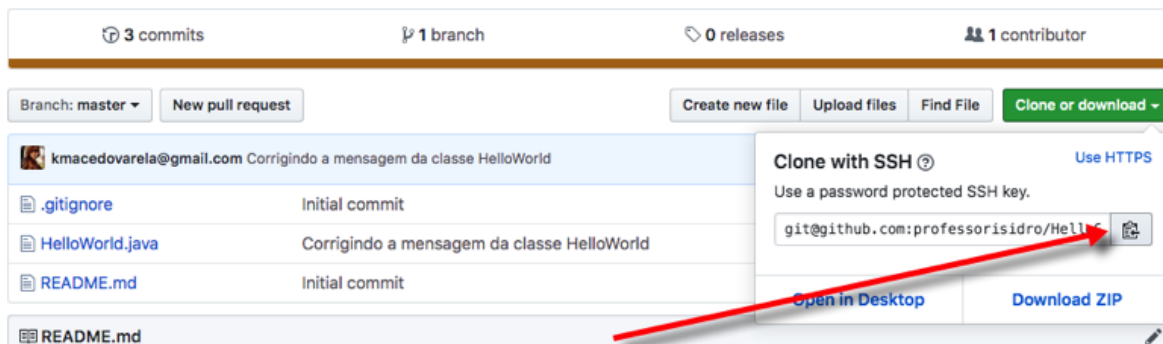


Create repository



Uma vez isso feito, você vai precisar do caminho dele, para que você consiga sincronizar a pasta da sua máquina com o repositório.

Veja a imagem abaixo como obter esse link:



Atenção

Isso é super importante: todo repositório remoto tem que estar vinculado a uma pasta local da sua máquina, pois a partir dela estará toda a hierarquia de pastas e arquivos do seu projeto. Portanto, tenha sempre sua área específica para os seus projetos do Github (eu sugiro uma pasta chamada projetos ou github).

Vinculando sua pasta local ao repositório

Alternativa 1: Clonando o link do seu repositório à sua pasta local

```
git clone "URL obtida no github"
```

Isso irá criar uma pasta na sua máquina de mesmo nome do repositório do Github (isso é bacana pq facilita muito) e, a partir daí você já poderá trabalhar.

Além disso, há uma pasta `.git`, dentro da sua pasta, responsável por armazenar todas as informações do seu projeto.

Alternativa 2: Criando sua pasta local e associando-a ao repositório. Após criar a sua pasta (com o nome que você quiser), você precisa dos seguintes comandos:

```
git init
```

```
git remote add origin "URL obtida no github"
```



Atenção

Se você criou seu projeto no GITHUB e ele já está com um README na sua estrutura, é necessário, antes de mais nada, você obter na sua máquina local, tudo o que tiver no seu repositório (antes de colocar seus próprios arquivos).

Faça um PULL do seu repositório na sua máquina (veremos esse comando mais a frente)

```
git pull origin master
```

Pronto, já está associada a pasta local ao repositório remoto. Agora bora para as atividades do dia-a-dia.

Comandos do seu dia-a-dia



Vamos entender o que significa cada comando e, a partir daí, traçar um pequeno guia de “boas práticas” de versionamento.

```
git status
```

Verifique quais são os arquivos que foram adicionados ao projeto, quem está atualizado, quem está efetivado.

```
git add "arquivo"/"diretório"
```

Insere os arquivos ou diretórios (pode ser o diretório atual, representado pelo ponto .) para serem rastreados pelo controle de versão. Sempre que você alterar ou inserir um novo arquivo, adicione-o ao seu controle de versões.

```
git pull origin "nome da branch"
```

Obtém do repositório a última atualização da ramificação do projeto. Em geral o nome padrão da Branch é master. E uma “Branch” é uma nova ramificação da estrutura do projeto. Usar Branches é muito útil para você preservar versões estáveis do seu software.



Atenção

A branch master é sempre aquela versão mais atual do sistema (a de produção), entretanto a equipe de desenvolvimento pode (e deve) trabalhar em inclusões de novos módulos do sistema em uma branch separada para não “contaminar” uma versão estável.



```
git commit -m "Mensagem do Commit"
```

Efetiva as alterações locais feitas no seu repositório como sendo "última versão". Cuidado: em GIT, o Commit efetiva a versão local como definitiva. Ainda falta fazer o upload para o repositório.

```
git push origin "nome da branch"
```

Efetiva o Upload das suas alterações para o Repositório remoto. Lembre-se que só sofrerão upload os arquivos que forem efetivados ("commitados").

Boas Práticas para seu dia-a-dia

Apenas para que você tenha uma facilidade de operação e também garantir um pequeno procedimento, vamos lá com a "receita de bolo" para você atualizar seu repositório.

Lembre-se: cada commit é uma efetivação de elementos relevantes, ou seja, mudanças que podem afetar seu software de forma efetiva.

1. `git pull origin master` // obtenho as últimas alterações feitas pelos outros
2. `git add .` // adiciono meus novos arquivos
3. `git commit -m "mensagem"` // efetivo as alterações



4. `git push origin master` // faço o upload para o repositório

Claro que, sempre entre um comando e outro, vale um git status para acompanhar o que acontece.



Podcast

Você está curtindo o material?

O que acha de ouvir [esse podcast](#) sobre GIT e aprofundar ainda mais o conhecimento?

Espero que você tenha curtido esse material complementar!

Abraços, bons estudos e nos vemos em breve!



Referências Bibliográficas

FERREIRA, Gabriel. Instalando e Configurando o GIT. 2022. Elaborado pelo Prof. Isidro. Disponível em:

<http://gabsferreira.com/instalando-o-git-e-configurando-github/>.

Acesso em: 19 jan. 2022.

MASSETTO, Francisco Isidro. Manual de Sobrevivência do GitHub. 2022.

Elaborado pelo Prof. Isidro. Disponível em:

<https://www.professorisidro.com.br/manual-de-sobrevivencia-do-github/>.

Acesso em: 19 jan. 2022.