**Git e GitHub**

Sumário

[Sumário 2](#_Toc91842638)

[Comandos básicos 3](#_Toc91842639)

[git 3](#_Toc91842640)

[git config –list 3](#_Toc91842641)

[git init 3](#_Toc91842642)

[git remote -v 3](#_Toc91842643)

[git push 3](#_Toc91842644)

[git clone 3](#_Toc91842645)

[git config --global 3](#_Toc91842646)

[git config --global --unset 3](#_Toc91842647)

[git --version 3](#_Toc91842648)

[git status 3](#_Toc91842649)

[git pull 4](#_Toc91842650)

[git add 4](#_Toc91842651)

[git commit -m "" 4](#_Toc91842652)

[git push 4](#_Toc91842653)

[git remote add 4](#_Toc91842654)

[ssh-keygen -t 5](#_Toc91842655)

[eval $(ssh-agent -s) 5](#_Toc91842656)

[ssh-add 5](#_Toc91842657)

# Comandos básicos

### git

Exibe a lista de comandos disponíveis.

### git config –list

Retorna a lista de todas as configurações do git local.

### git init

Inicia o git e cria a pasta e repositório com os arquivos do projeto.

### git remote -v

Exibe a lista de repositório do GitHub cadastrados (são os repositórios remotos).

### git push

"Empurra"/envia os arquivos para o GitHub. Obs.: origin é o alias do URL. Ex.:

git push origin master

### git clone

Clona um projeto. Ex.:

git clone https://github.com/asantospe/livro-receitas.git

### git config --global

#### git config --global user.email “you@example.com”

Configura o e-mail do usuário padrão. Se desejar que seja apenas no diretório atual, remover o --global. Ex.:

git config --global user.email "email@outlook.com"

#### git config --global user.name "Your Name"

Configura o nome do usuário padrão. Ex.:

git config --global user.name "asantospe"

### git config --global --unset

Deleta as configurações feitas (serve para refazer as configurações - deletar e adicionar, manualmente, novamente).

Ex.: **Remover o e-mail padrão do usuário:**

git config --global --unset email@outlook.com

### git --version

Exibe a versão do sistema.

### git status

Exibe o status dos arquivos no diretório atual (se está sincronizado ou não) com o GitHub.

git pull

Atualiza o projeto de acordo com as informações que estão no GitHub. Muito utilizado para corrigir conflito de merge. Quando há um conflito, o Git irá modificar o arquivo e irá marcar o conteúdo que difere com o que está no Git.

Graphical user interface, application

Description automatically generated

- Pavê (que está acima de ===) é o que está no repositório local e – Bolo de cenoura (que está abaixo de ===) é o que está no Git. Após fazer a correção, é necessário colocar o arquivo na área de Stage (*git add* *\** e *git commit -m “”*). Agora, é possível enviar um push.

### git add

Sincroniza um ou mais arquivos do diretório para o Git (\* para todos os arquivos), altera o status dos arquivos de Untracked (arquivos com modificações que o Git não sabe) para Tracked (arquivos com modificações que o Git sabe) - o Git tem ciência dos arquivos novos e/ou alterados. Se executar esse comando pela segunda vez, ele vai para Staged (status antes de enviar para o GitHub).

Ex.: **Sincronizar** todos **os arquivos do repositório:**

git add \*

### git commit -m ""

Confirma as alterações, seja add ou remove, no repositório local (Git, não GitHub). Finaliza o estado dos arquivos, tirando um snapshot, e volta os arquivos para o estado Unmodified. Ex.:

git commit -m "Alterado componente x, recurso y: 30122021\_11h15\_Alexander."

### git push

Confirma as alterações de commit do servidor local (Git) para nuvem (GitHub).

### git remote add

Após criar um repositório no GitHub é necessário enviar o repositório do Git, usando o seguinte comando. Ex.:

git remote add origin https://github.com/asantospe/livro-receitas.git

Onde origin é o “alias” do URL.

### ssh-keygen -t

Gera a chave pública.

### eval $(ssh-agent -s)

Inicia o SSH agente do git.

### ssh-add

Adiciona a chave privada no repositório local (não enviar essa chave para o GitHub).