

Curso de Git e Github Digital Innovation One

Professor Otávio Reis

Primeiros comandos com Git

1. Iniciar o GIT
2. Iniciar o versionamento
3. Criar um commit.

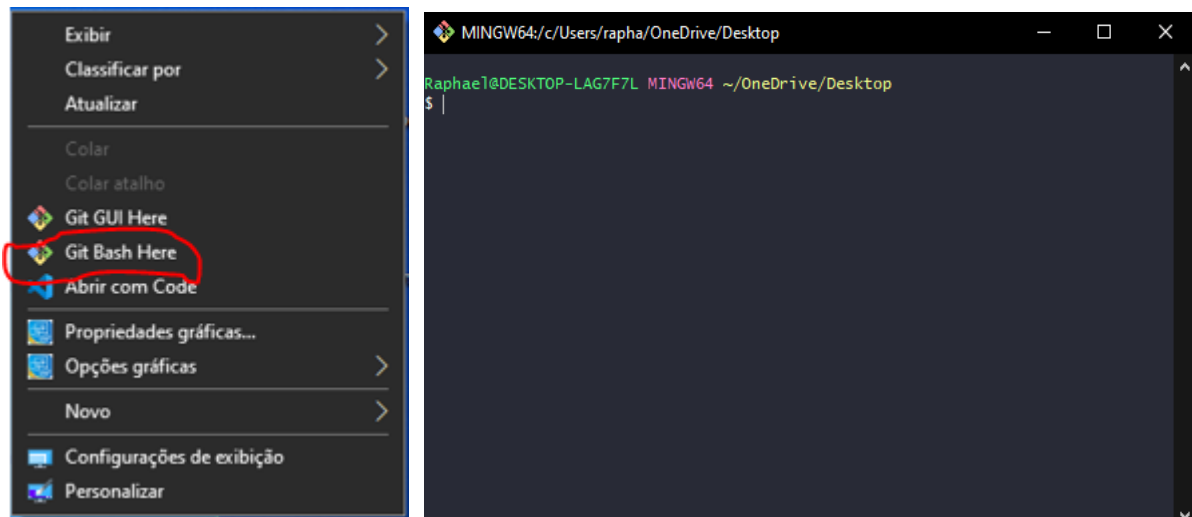
- `git init`
- `git add`
- `git commit`

Pelo terminal sempre o comando se inicia pelo nome do programa, no caso acima estamos chamando pelo **Git** e posteriormente o comando **específico do Git**

Criando um repositório

Criada a pasta padrão em **C:/Workspace**

Se clicarmos com o botão direito do mouse encontraremos o **Git Bash Here**, clicando neste atalho, automaticamente já se abre o **terminal do Git Bash**



Para entrarmos na pasta C:, digitamos no terminal `cd c:`, pronto, já estamos no diretório C:, aí digitamos `cd workspace` (para acessarmos a pasta Workspace)

Para criação de uma pasta dentro do Workspace

`mkdir livro-receitas`

```

MINGW64:/c/workspace
hello.txt
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini

Raphael@DESKTOP-LAG7F7L MINGW64 ~
$ cd c
bash: cd: c: No such file or directory

Raphael@DESKTOP-LAG7F7L MINGW64 ~
$ cd c:

Raphael@DESKTOP-LAG7F7L MINGW64 /c
$ cd workspace

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace
$ mkdir livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace
$ ls
hello.txt  livro-receitas/

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace
$

```

Iniciando o Git dentro da pasta livro-receitas

Comando **git init**

```

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas
$ git init
Initialized empty Git repository in C:/workspace/livro-receitas/.git/

```

Para limpar a tela **Ctrl + I**

Para dentro de pastas ocultas o comando é o seguinte = **ls -a**

```

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ ls -a
./  ../  .git/

```

Entrando na pasta oculta = **cd .git**

```

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas/.git (GIT_DIR!)
$

```

Posteriormente ao entrar na pasta oculta, podemos dar um novo comando **ls**, aí aparece as pastas dentro desta pasta oculta

```

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas/.git (GIT_DIR!)
$ ls
HEAD  config  description  hooks/  info/  objects/  refs/

```

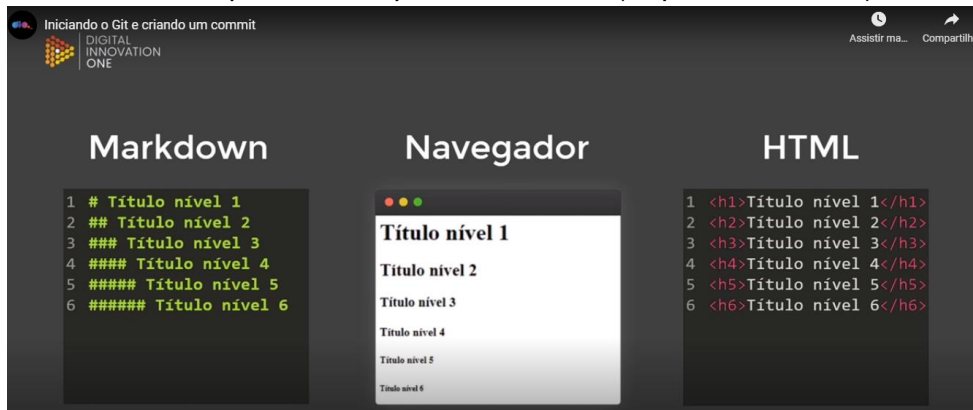
Voltando um nível dentro da pasta = **Cd ..**

Criando o primeiro arquivo dentro da pasta, se é a primeira vez que estamos usando o Git, precisaremos configurar os **nickname/username** e **email**, isso para poder identificar o autor do commit

Comando = email > `git config --global user.email "raprithon@gmail.com"`
Username > `git config --global user.name raprithon`

Adicionando um Arquivo na pasta Workspace

Colocaremos na pasta um arquivo **Markdown** (arquivo Markdown)



Para salvar um arquivo em Markdown

Nomedoarquivo.md

Para ir aumentando o tamanho do texto utilizamos a #

Strogonoff de Frango
Strogonoff de Frango
Strogonoff de Frango
****Negrito****



Agora daremos um commit

No terminal

`git add *` > Enter

`git commit -m "commit inicial"` (aqui dentro colocamos uma mensagem de texto dizendo informações referente ao oq estamos fazendo neste commit)

```
MINGW64:/c/workspace/livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receita
s (master)
$ ls
strogonoff.md

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receita
s (master)
$ git add *

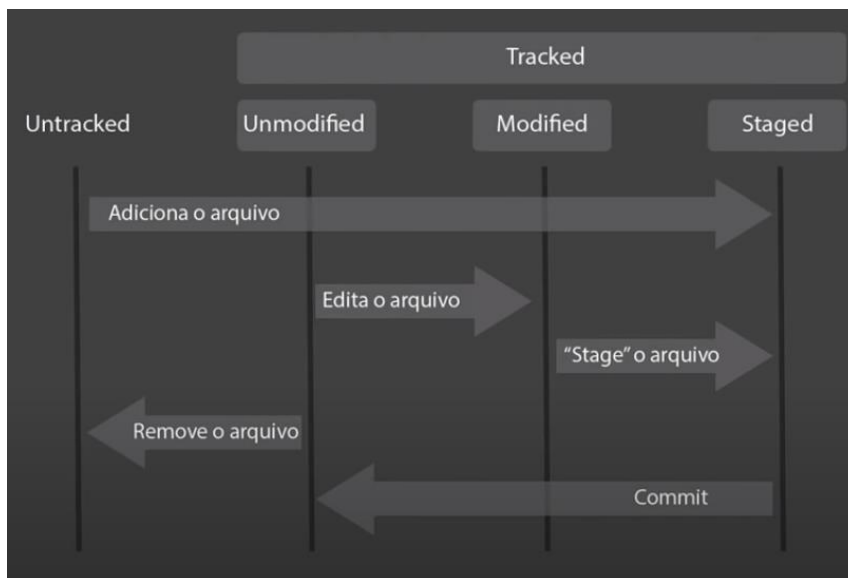
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receita
s (master)
$ git commit -m "commit inicial"
[master (root-commit) 71f673b] commit inicial
1 file changed, 26 insertions(+)
create mode 100644 strogonoff.md

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receita
s (master)
$ |
```

Revendo

GIT INIT = Para criar a pasta oculta .git, com ele é criado um repositório dentro da pasta, quando utilizamos ele de fato estamos iniciando o Git na pasta específica

Conceito de Tracked ou Untracked



3 estágios

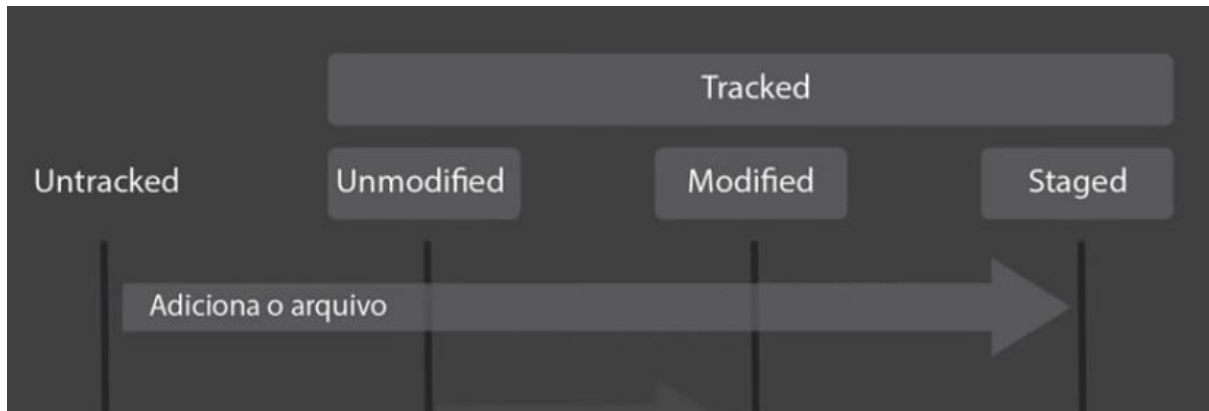
- **Unmodified** - Ainda não foi modificado
- **Modified** - Já foi modificado
- **Staged** - Conceito chave, onde ficam os arquivos que estão preparados para um outro tipo de agrupamento (Atrás do palco)

Untracked = Arquivos que o git ainda não tem ciência

Tracked = Arquivos que a gente tem ciência deles

Momento em que o Git ainda não conhece o arquivo (Untracked)

Com o comando **git add** o arquivo vai de **Untracked** direto para o **Stage** (área em que está aguardando para ação)



Unmodified

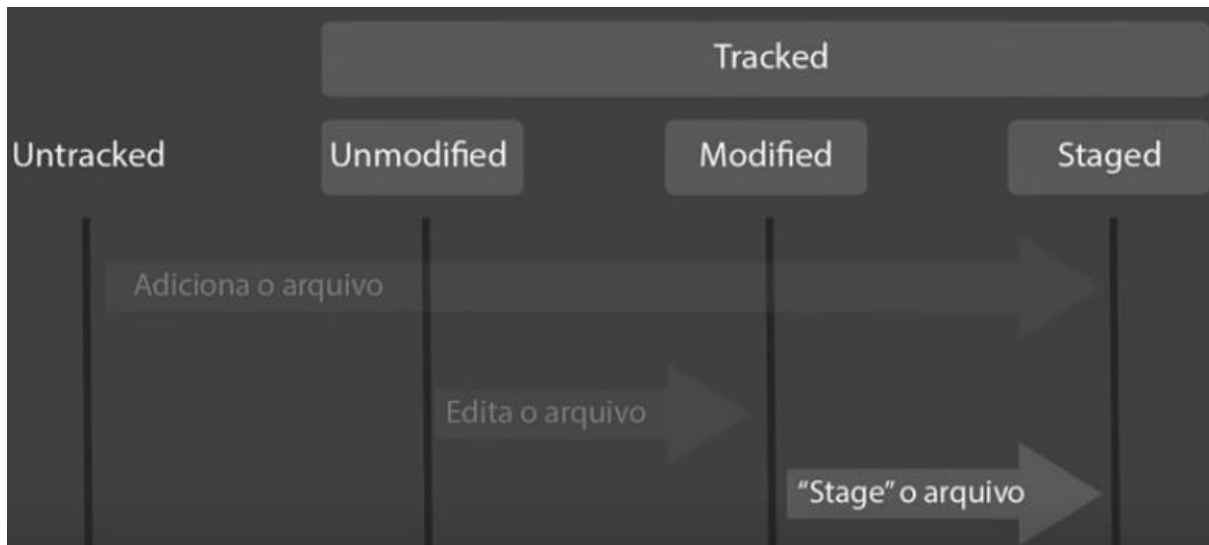
É o arquivo que ainda não sofreu **modificação**

Quando abrimos os arquivos e fazemos uma modificação, automaticamente o Git passará de **Unmodified** para **Modified** (Fazendo uma comparação entre os **SHA1**)



Staged

Se novamente rodarmos o **git add** neste arquivo **modified** ele vai para o **Staged** (entra em uma área especial para entrar em ação)



Se removermos um arquivo sem ter sofrido nenhuma modificação ele volta para o Untracked (pois o Git ainda não tinha ciência deste arquivo, não havia registrado em nenhum momento este arquivo, não tinha conhecimento dele)

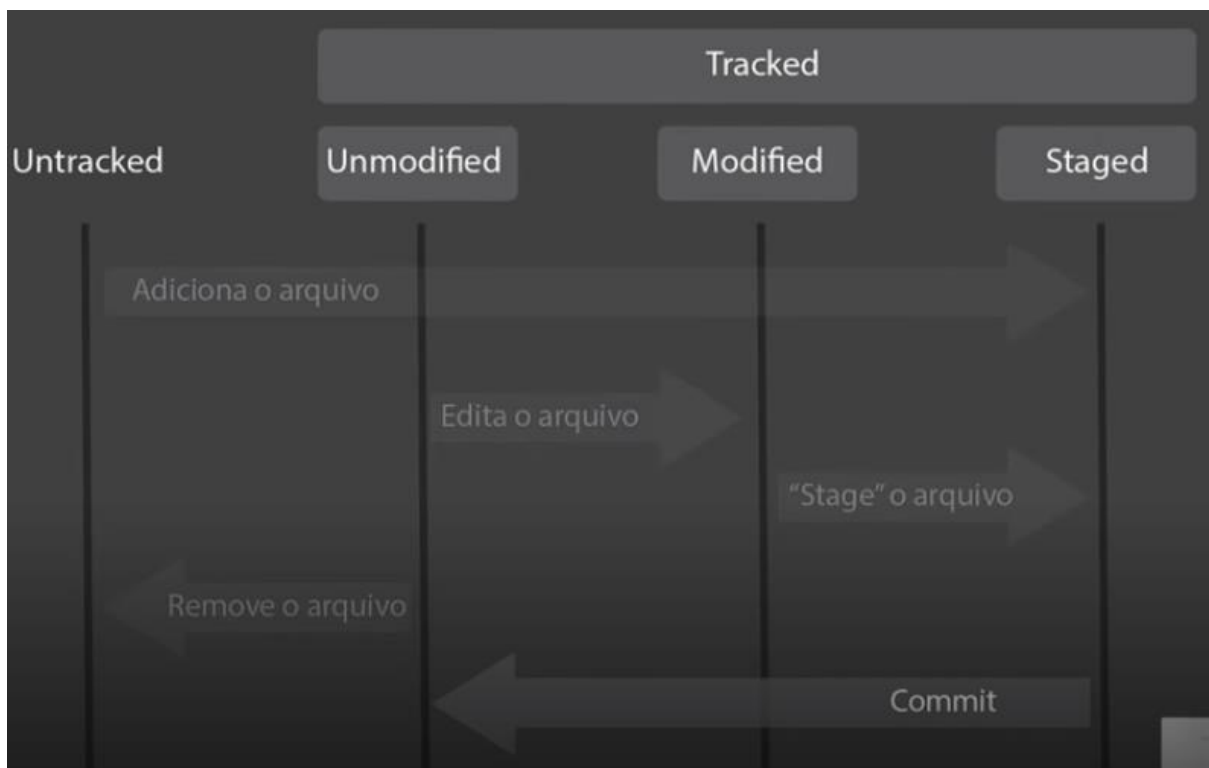
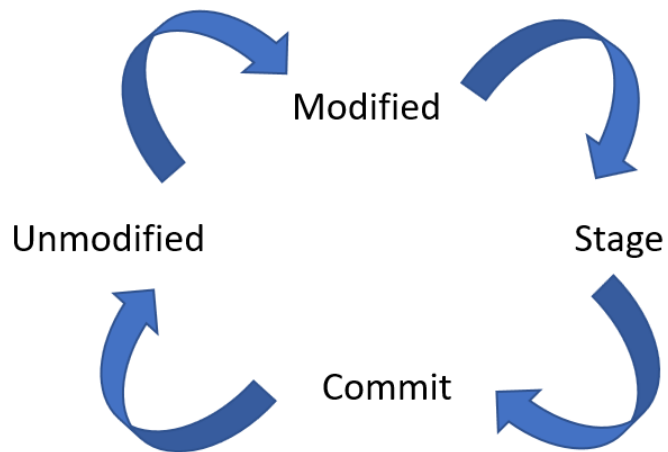


Staged - Aguardando para o Commit

De Staged > Para Commit

Após dado o Commit o arquivo volta para o estado de **Unmodified**, isto é o **Commit** dizendo que **acabou de dar como se fosse um print do arquivo naquele momento**

Ciclo



O que são repositórios

As ações feitas no repositório local não são refletidas automaticamente no repositório do servidor (em nuvem).

Os arquivos tratados no terminal Local (repositório de trabalho), ficam **transitando entre o repositório de trabalho e stage** conforme vamos fazendo ações nestes arquivos, após feito o **commit**, o arquivo passa a integrar o repositório local, que consequentemente podemos mandar para o repositório remoto (servidor)



Quando adicionamos um arquivo que era **untracked** e damos um **git add**, este arquivo é movimentado para o **Staged**

Quando temos um arquivo **modified** e damos um **git add**, ele também vai para **Staged**

Os arquivos ficam transitando entre **Working Directory** e **Staging Area**

O processo após o **commit** é saindo de **Staged** e indo para **Unmodified**, com isso neste momento tirou uma foto do arquivo e consequentemente entra para o diretório local, tudo que está no repositório local obrigatoriamente está com **Commit**



Entendendo na prática o ciclo acima

git status = traz o status do repositório local

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Dentro da pasta livro-receitas, criaremos uma pasta receitas

Comando mkdir receitas

Movendo um arquivo entre pastas

Comando **mv** = Movimentar

Estrutura do comando

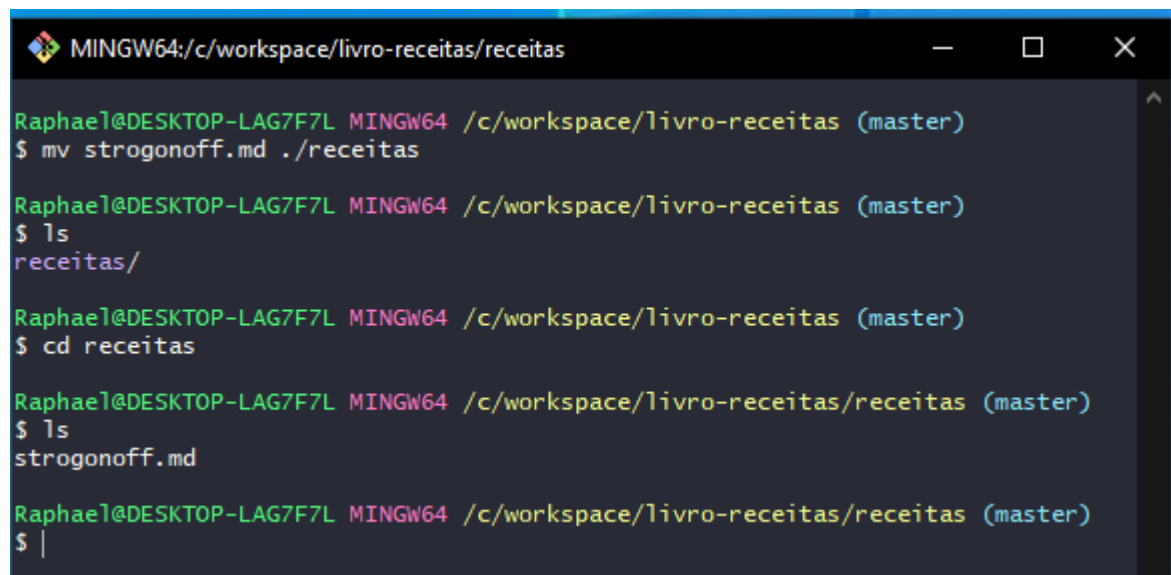
mv strogonoff.md ./receitas

mv = comando de movimento

strogonoff.md = Arquivo

./ = De Para

receitas = Pasta de destino



```
MINGW64:/c/workspace/livro-receitas/receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ mv strogonoff.md ./receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ ls
receitas/

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ cd receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas/receitas (master)
$ ls
strogonoff.md

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas/receitas (master)
$ |
```

Após feito o procedimento acima damos novamente um comando git status

```
MINGW64:/c/workspace/livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    deleted:   strogonoff.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    receitas/

no changes added to commit (use "git add" and/or "git commit -a")

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$
```

Como a pasta criada recentemente, a Receitas, ainda não passou pelo git add (commit), o sistema não a reconhece, por isso a informação de que o arquivo Strogonoff aparece, por isso é importante que façamos um Commit na pasta receitas

Arquivo que aparece deletado

```
deleted:   strogonoff.md
```

O Git encontrou receitas e sugeriu para darmos um comando git add

```
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    receitas/
```

Seguindo a orientação do Git faremos o seguinte comando

git add - Adicionar

git rm - Excluir

Comando para reconhecimento do arquivo e da pasta

git add strogonoff.md receitas

O comando acima é como se agora o git soubesse da existência novamente do Strogonoff e que agora existe a pasta receitas

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    renamed:   strogonoff.md -> receitas/strogonoff.md
```

Após feito o procedimento de reconhecimento da pasta receitas e movimentação do arquivo strogonoff.md para dentro dela, daremos um **Commit** que é o **snapshot** da pasta atual

git commit -m "cria pasta receitas, move arquivo para receitas"

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git commit -m "cria pasta receitas, move arquivo para receitas"
[master aaccf57] cria pasta receitas, move arquivo para receitas
1 file changed, 0 insertions(+), 0 deletions(-)
rename strogonoff.md => receitas/strogonoff.md (100%)
```

Readme = Arquivo que no caso fará o papel de indexador

Comando para criação do README

echo > README.md

```
MINGW64:/c/workspace/livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ echo > README.md

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ ls
README.md  receitas/

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ |
```

Preenchemos o arquivo README.md e posteriormente salvamos

No terminal digitamos git status, para ver como está o nosso Git e aparece a seguinte informação:

Abaixo ele nos informa que devemos adicionar o README.md (pois ele ainda, não está trackeado pelo Git)

Por isso ele nos diz use **git add <file>**

```
MINGW64:/c/workspace/livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md

nothing added to commit but untracked files present (use "git add" to track)

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ |
```

Porém desta vez usaremos o comando `git add *`, pois neste caso queremos pegar tudo que foi modificado no diretório de trabalho

```
MINGW64:/c/workspace/livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md

nothing added to commit but untracked files present (use "git add" to track)

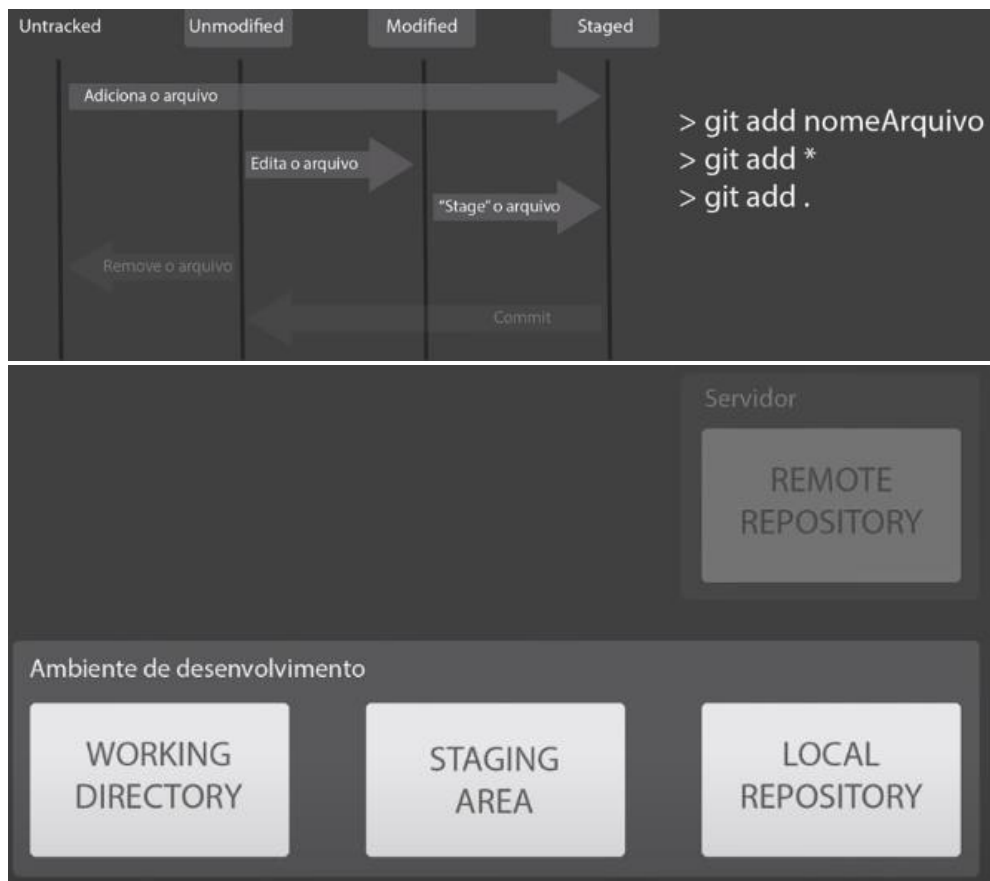
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git add *
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ .....
```

Após feito isso, finalmente daremos o famoso Commit (Snapshot do arquivo), para que enfim seja criado mais um versionamento da nossa pasta.

git commit -m "adiciona index"

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git commit -m "adiciona index"
[master 01977ae] adiciona index
1 file changed, 5 insertions(+)
create mode 100644 README.md
```



GitHub

Passagem de arquivos do repositório local para o repositório remoto

Para sabermos a lista de configurações do nosso repositório local o comando é:
git config --list

Aparecerão todas as informações como por exemplo email cadastrado entre outros

Para sair do config list, basta apertarmos a letra Q

Para modificarmos as informações do git config, ou seja, para reescrevermos

`git config --global --unset user.email`

`git config --global --unset user.name`

Estes dois comando acima tiram das configurações os itens descritos

Para retornar as informações acima basta fazer os seguintes comandos


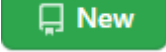
```
git config --global user.email "raprithon@gmail.com"
```

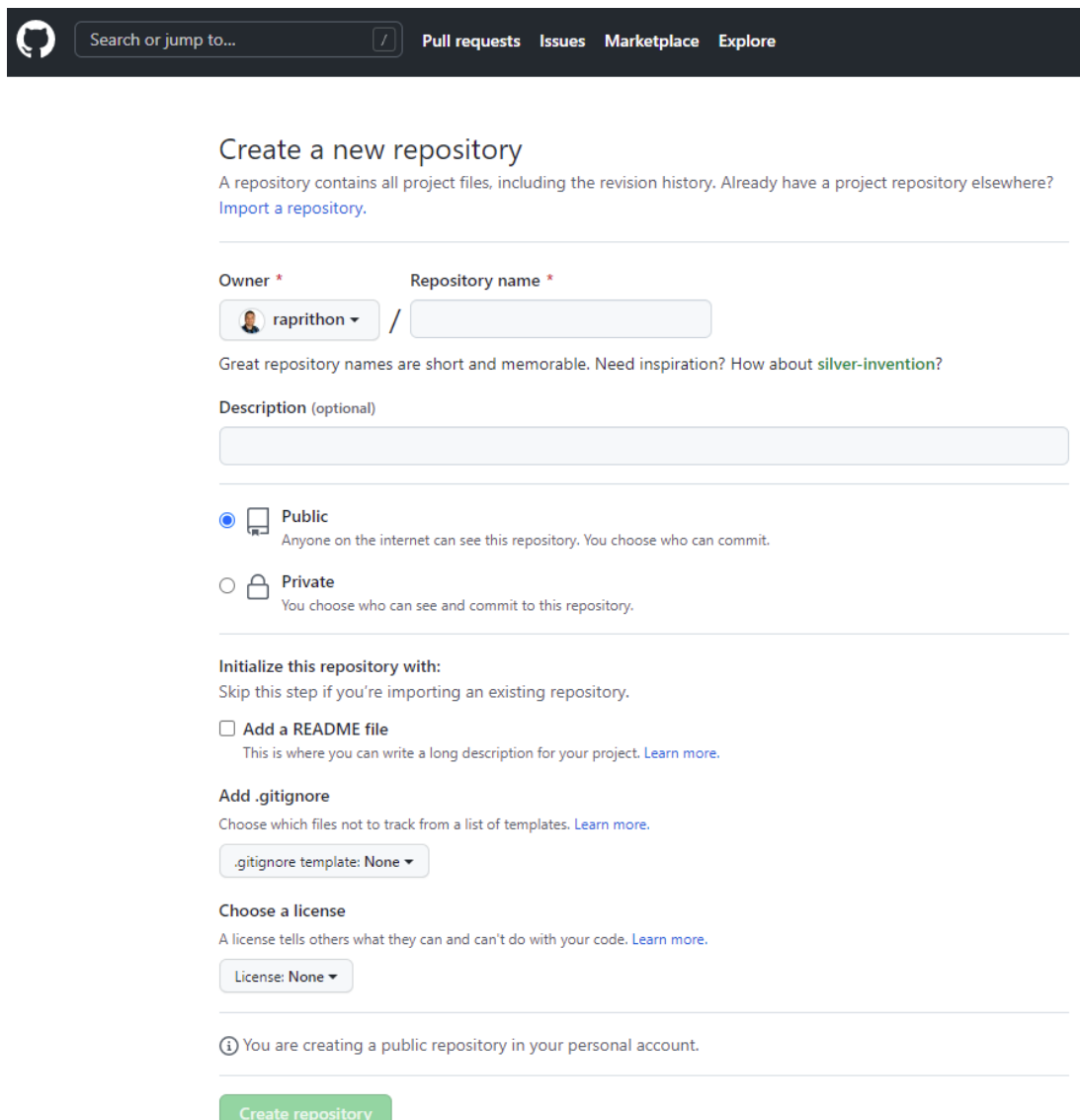
```
git config --global user.username "raprithon"
```

Entrando no GitHub

Além de ser um repositório remoto, ele também é uma rede social.

Criando o repositório


Quando clicamos na aba  **Repositories** e depois clicamos em , seremos direcionados para a tela abaixo:



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 ***

 raprithon /

Great repository names are short and memorable. Need inspiration? How about [silver-invention?](#)

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.


☐  **Private**
You choose who can see and commit to this repository.

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

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

 You are creating a public repository in your personal account.

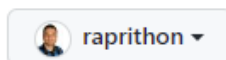
[Create repository](#)

Aqui colocamos o nome do nosso repositório

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 *



Great repository names are short and memorable. Need inspiration? How about [silver-invention](#)?

Colocamos uma descrição abaixo

Description (optional)

meu livro de receitas

Definimos se é público ou privado



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Aqui ele quer saber se inicializa um README, porém quando criamos o README no repositório local, não é necessário deixar aqui marcado

Initialize this repository with:

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



Add a README file

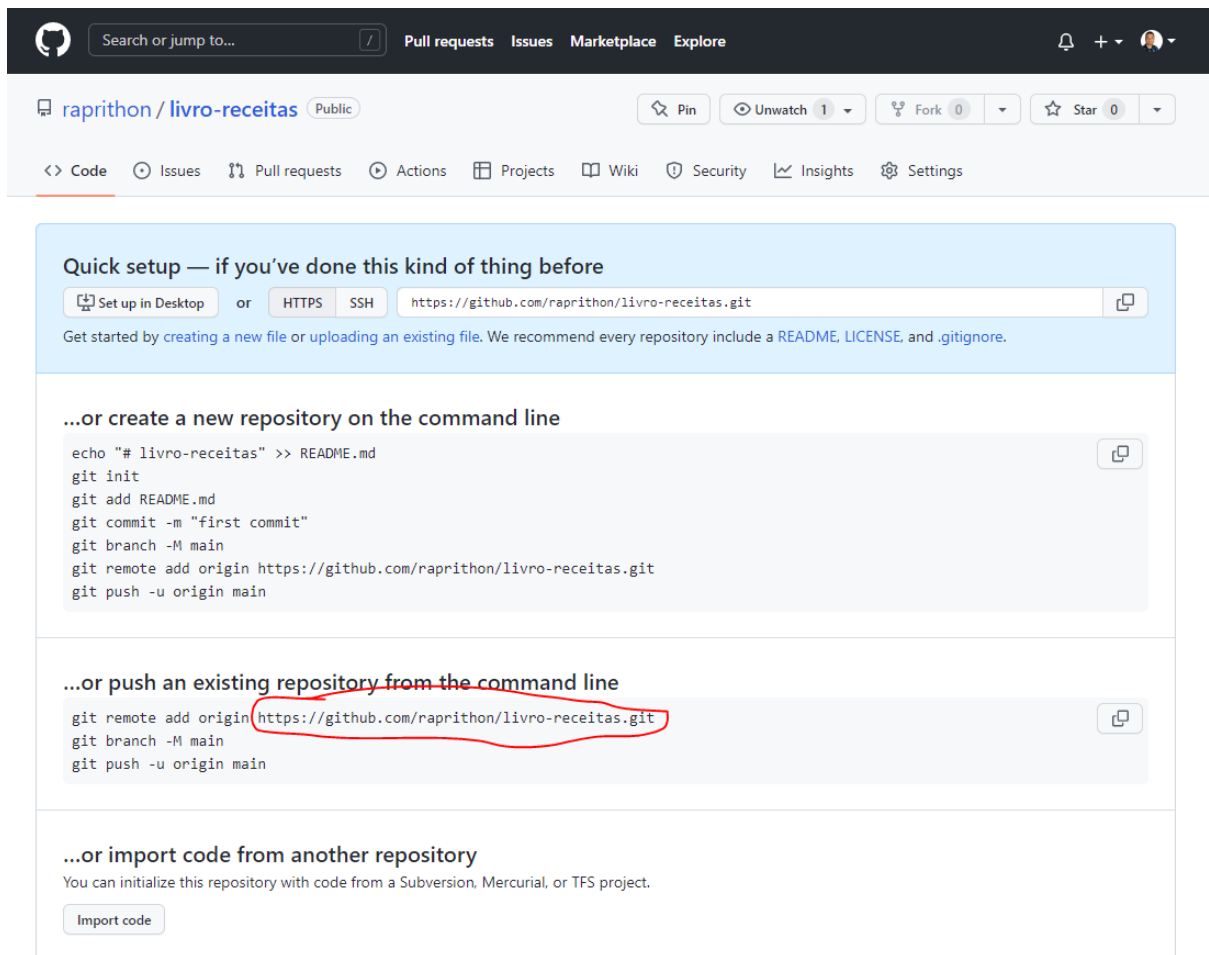
This is where you can write a long description for your project. [Learn more.](#)

Aí no final clicamos no botão abaixo para criação do repositório

Create repository

Para que haja o link entre o nosso repositório local e o nosso repositório remoto, precisamos fazer o seguinte:

copiamos a URL que é fornecida no campo **...or push an existing repository from the command line**



Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH <https://github.com/raprithon/livro-receitas.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# livro-receitas" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/raprithon/livro-receitas.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/raprithon/livro-receitas.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

No caso a url aqui é `https://github.com/raprithon/livro-receitas.git`

Apontando o nosso repositório local para o repositório remoto

Adicionando primeiro o link do nosso servidor remoto

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

Por convenção sempre usamos o apelido origin

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git remote add origin https://github.com/raprithon/livro-receitas.git
```

Para sabermos quais repositórios temos cadastrado digitamos o seguinte código **git remote -v**

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git remote -v
origin https://github.com/raprithon/livro-receitas.git (fetch)
origin https://github.com/raprithon/livro-receitas.git (push)
```

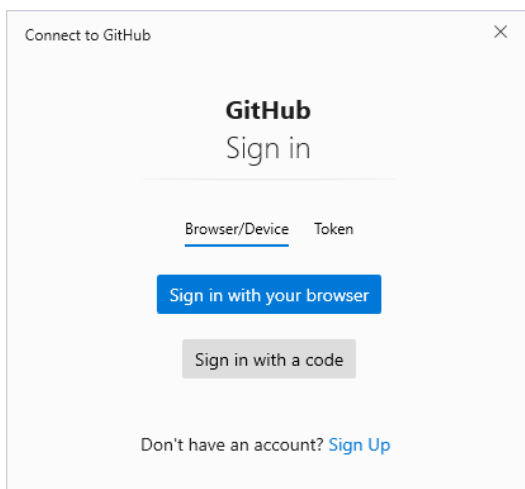
É sempre bom darmos um comando `git status` para certificarmos de que não exista nenhum pendência

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
nothing to commit, working tree clean
```

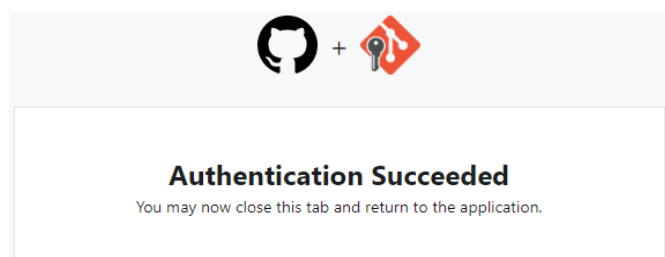
Quando falamos de empurrar é porque o comando justamente para subir nosso arquivo do repositório local para o remoto é **PUSH**

`git push origin master`

Após colocar o comando de push, o GitHub vai nos pedir uma autenticação



Neste caso clicamos em **Sign in with your browser** aparecerá a tela a seguir se estiver tudo certo



Consequentemente no terminal aparecerá a seguinte mensagem, dizendo que enviou do repositório local para o repositório remoto

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git push origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (8/8), 1.27 KiB | 433.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/raprithon/livro-receitas.git
* [new branch]      master -> master
```

Resultado da ação de subir o repositório

raprithon / livro-receitasPublic

Pin

Unwatch1

Fork0

Star0

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

master1 branch0 tags

Go to file

Add file

Code

About

raprithon adiciona index01977ae1 hour ago3 commits

receitascria pasta receitas, move arquivo para receitas4 hours ago

README.mdadiciona index1 hour ago

README.md

Livro de receitas👨‍🍳

Olá! Bem vindo ao meu livro de receitas👉

- Strogonoff de frango

meu livro de receitas

Readme

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

raprithon / livro-receitasPublic

Pin

Unwatch1

Fork0

Star0

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

masterlivro-receitas / receitas /

Go to file

Add file

...

raprithon cria pasta receitas, move arquivo para receitasaaccf574 hours agoHistory

..

strogonoff.mdcria pasta receitas, move arquivo para receitas4 hours ago

26 lines (20 sloc) | 1021 Bytes

<>RawBlame

Strogonoff de Frango 🐔

Ingredientes

- 3 [peitos de frango](#) cortados em cubos
- 1 dente de alho picado
- sal e pimenta a gosto
- 1 cebola picada
- 2 colheres (sopa) de maionese
- 1 colher de manteiga
- 1/2 [copo de ketchup](#)
- 1/3 copo de mostarda
- 1 copo de cogumelos
- 1 copo de creme de leite
- [batata palha](#) a gosto

Mode de Preparo

- Em uma panela, misture o frango, o alho, a maionese, o sal e a pimenta.
- Em uma frigideira grande, derreta a manteiga e doure a cebola.
- Junte o frango temperado até que esteja dourado.
- Adicione os cogumelos, o ketchup e a mostarda.
- Incorpore o [creme de leite](#) e retire do fogo antes de ferver.
- Sirva com arroz branco e batata palha.

Neste campo circulado de vermelho é o início do código SHA1

raprithon / livro-receitas Public

PinUnwatch 1Fork 0Star 0

<>CodeIssuesPull requestsActionsProjectsWikiSecurityInsightsSettings

master1 branch0 tagsGo to fileAdd fileCode

raprithon adiciona index	01977ae · 1 hour ago · 3 commits
receitas	cria pasta receitas. move arquivo para receitas · 4 hours ago
README.md	adiciona index · 1 hour ago

README.md

Livro de receitas 🍳

Olá! Bem vindo ao meu livro de receitas 🍷

- Strogonoff de frango

About

meu livro de receitas

Readme

0 stars

1 watching

0 forks

Releases

No releases published

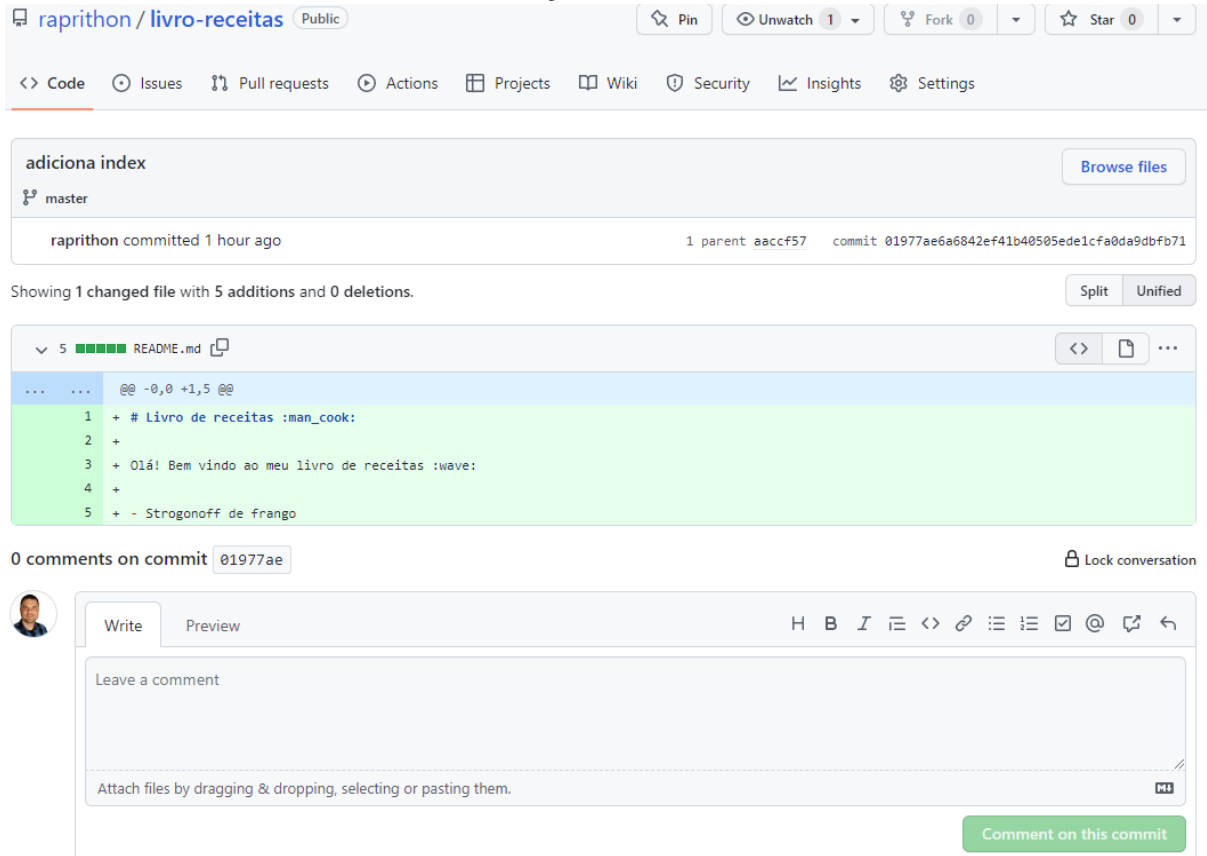
Create a new release

Packages

No packages published

Publish your first package

Clicando no SHA1 ele nos traz informações dos commits realizados



raprithon / livro-receitas Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

adiciona index

master

raprithon committed 1 hour ago

1 parent aaccf57 commit 01977ae6a6842ef41b40505ede1cfa0da9dbfb71

Showing 1 changed file with 5 additions and 0 deletions.

Split Unified

5 README.md

```
@@ -0,0 +1,5 @@
1 + # Livro de receitas :man_cook:
2 +
3 + Olá! Bem vindo ao meu livro de receitas :wave:
4 +
5 + - Strogonoff de frango
```

0 comments on commit 01977ae

Lock conversation

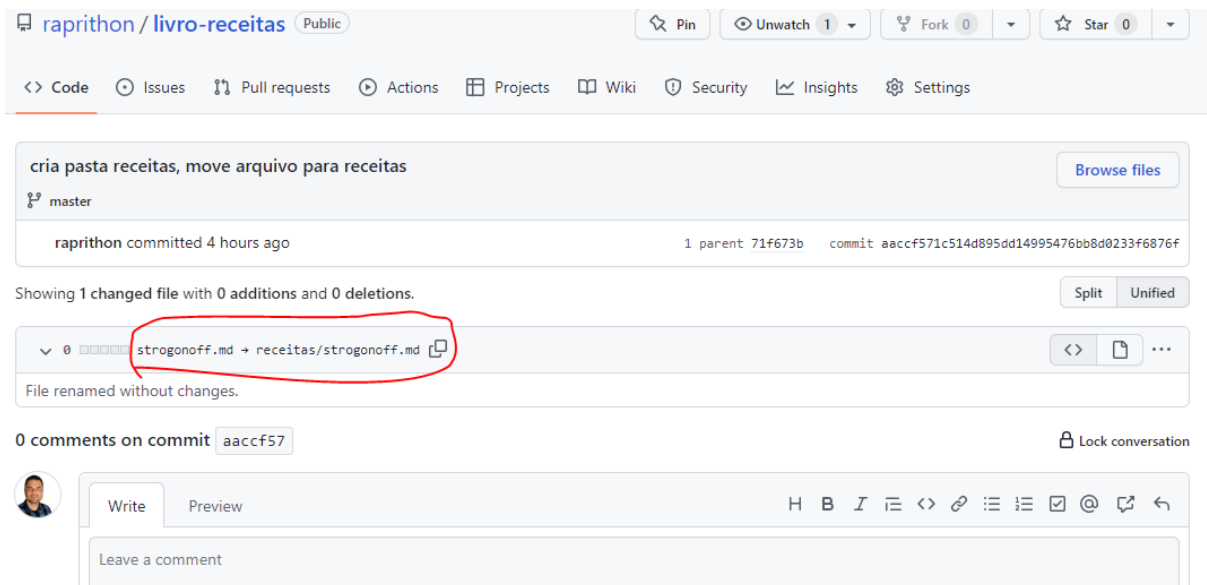
Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Comment on this commit

Abaixo um exemplo do commit onde transferimos os arquivos entre pastas



raprithon / livro-receitas Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

cria pasta receitas, move arquivo para receitas

master

raprithon committed 4 hours ago

1 parent 71f673b commit aaccf571c514d895dd14995476bb8d0233f6876f

Showing 1 changed file with 0 additions and 0 deletions.

Split Unified

0 strogonoff.md → receitas/strogonoff.md

File renamed without changes.

0 comments on commit aaccf57

Lock conversation

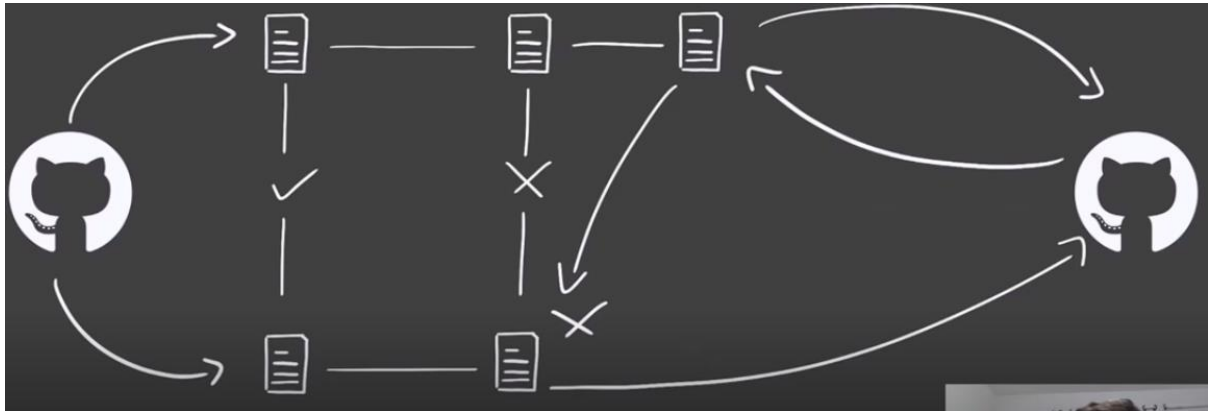
Write Preview

Leave a comment

É sempre ideal que o email do repositório seja o mesmo do commit, para que não haja divergência de informação

Resolvendo conflitos

Imagine que você tem um arquivo no GitHub, uma pessoa vem e clona ele para a máquina dela, você baixa o mesmo arquivo do GitHub e coloca na sua máquina, ambos executam uma alteração que por coincidência seja na mesma linha, para resolver este conflito o GitHub pedirá para que você defina qual atualização será a oficial.



Acrescentando mais informação no arquivo README.md

Abri o arquivo, acrescentei o item Pavê, salvei e fechei

Dei o comando **git status** e aparece a seguinte informação.

Como o arquivo sofreu uma edição ele está no status de **modified**.

```
MINGW64:/c/workspace/livro-receitas

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

Recapitulando

Para que um arquivo esteja pronto para o **commit**, precisamos que ele esteja no **Staged (atrás do palco)**, para que ele chegue a Stage é importante que façamos primeiro o comando **git add** (existe uma variação aqui entre **git add** e o nome do arquivo, como por exemplo **git add README.md** e o **git add ***, no caso do **git add ***, o **"*"** tem função adicionar tudo).

Pronto, agora o arquivo está pronto para o **commit**, dando o seguinte comando **git commit -m "Adiciona receita pave"**

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git commit -m "Adiciona receita pave"
[master 593e776] Adiciona receita pave
1 file changed, 1 insertion(+)
```

Para enviar o código da nossa máquina para o repositório remoto

git push origin master

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/livro-receitas (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 306 bytes | 306.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/raprithon/livro-receitas.git
01977ae..593e776 master -> master
```

Comando para puxar do repositório remoto para o local **git pull**

git pull origin master

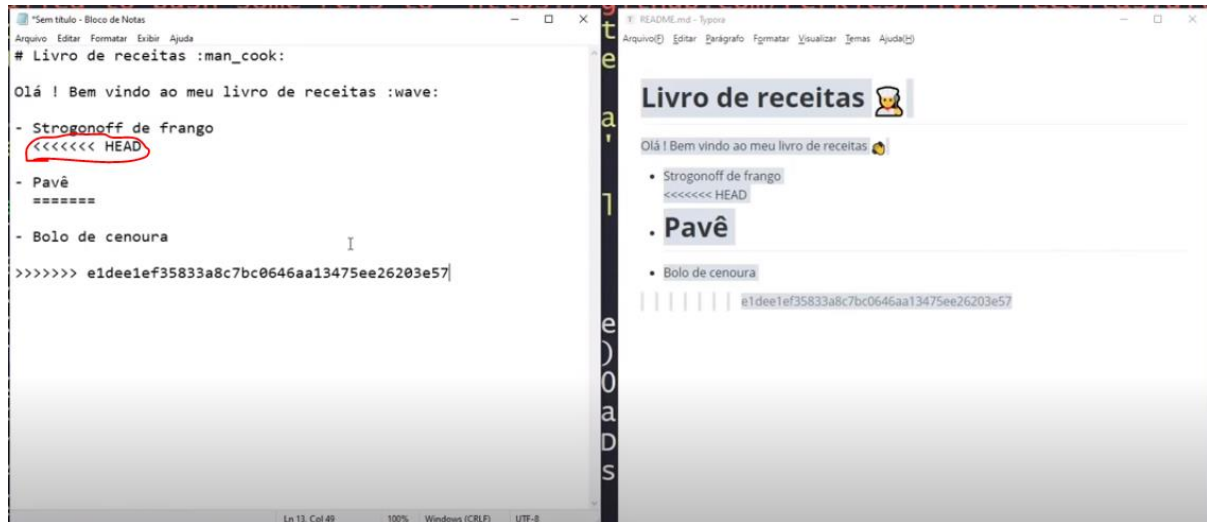
Quando existe conflito entre as versões do repositório remoto e repositório local, ao darmos o git pull origin master aparece a msg abaixo

```
Otavio@perkles-desktop MINGW64 /c/workspace/livro-receitas (master)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 701 bytes | 50.00 KiB/s, done.
From https://github.com/Perkles/livro-receitas
* branch                master      -> FETCH_HEAD
  54c6046..e1dee1e master      -> origin/master
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the resu
```

Onde aparecer Merge conflict, quer nos dizer que existe um conflito ao fazer a junção das informações do Readme.md, neste caso o Readme.md do repositório local está diferente do remoto

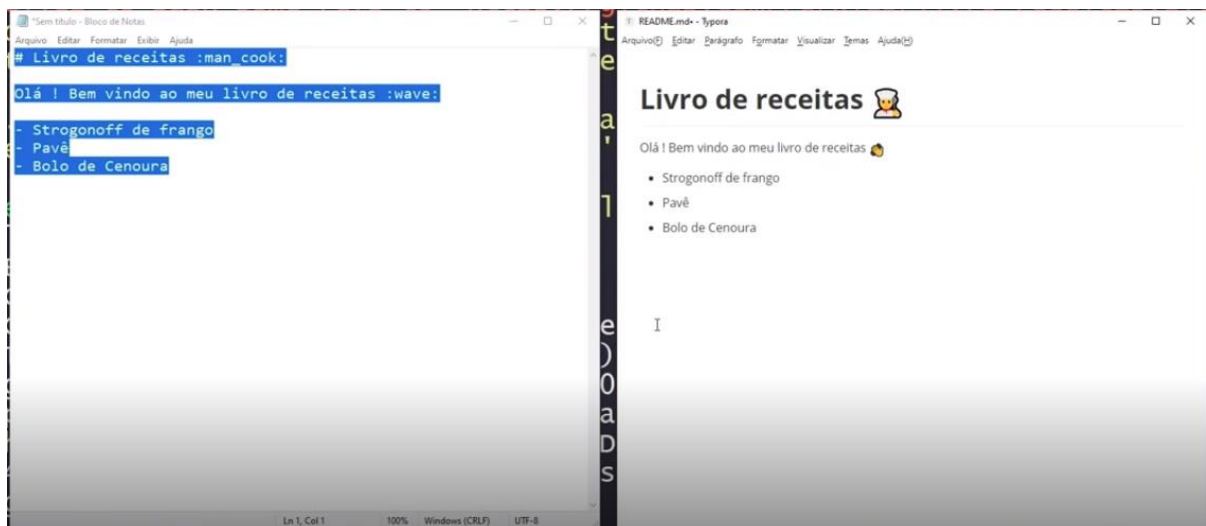
<<<<<<< HEAD - significa que algo que está abaixo disso é a alteração mais recente

===== Já desta marcação para baixo é a alteração encontrada no outro repositório



O GitHub pede para definir qual é a versão que deve ser utilizada.

No caso abaixo já é o arquivo corrigido e a versão oficial que deverá ser considerada



Quando voltamos ao terminal local e digitamos **git status** aparece a mensagem abaixo nos dizendo que ainda existe um conflito, porém como já temos a versão corrigida e oficial, adicionamos o arquivo com **git add *** e depois damos um **commit**

Sequência

git add * e depois **git commit -m "resolve conflitos"**

```
otavio@perkles-desktop MINGW64 /c/workspace/livro-receitas (master|MERGING)
$ git add *

otavio@perkles-desktop MINGW64 /c/workspace/livro-receitas (master|MERGING)
$ git commit -m "resolve conflitos"
[master b0d5f85] resolve conflitos

otavio@perkles-desktop MINGW64 /c/workspace/livro-receitas (master)
$ git push origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 610 bytes | 610.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/Perkles/livro-receitas.git
   e1dee1e..b0d5f85  master -> master
```

Sequência de comandos enviar alteração de arquivo do repositório local para o repositório remoto

Após ter feito uma alteração em um arquivo

git add "nome do arquivo" ou **git add ***

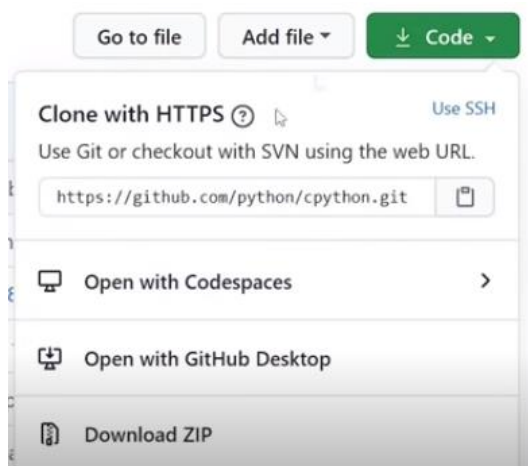
Para que o arquivo seja reconhecido pelo Git (para que Git tenha ciência)

git commit -m "Texto explicativo"

Para empurrar o arquivo do repositório local para o repositório remoto

git push origin master

Como baixar um repositório remoto para um repositório local



Open with Codespaces - Funcionalidade que permite abrir o código no navegador

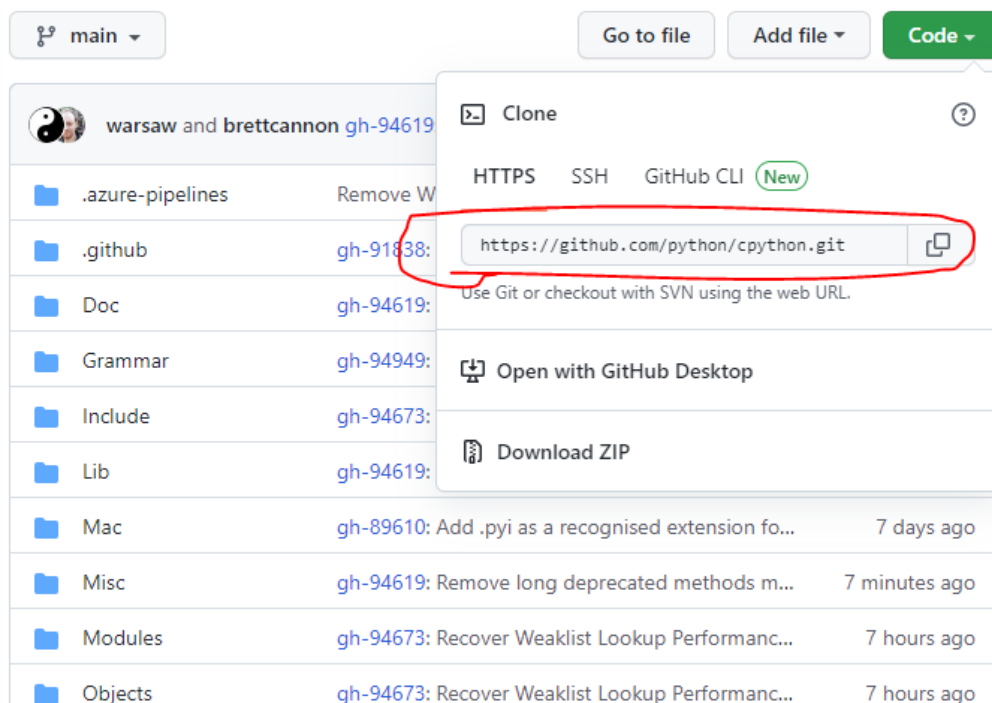
Open with GitHub Desktop - Um programa que é instalado diretamente no Desktop do GitHub

Download ZIP - Permite baixar o arquivo diretamente no computador

Clonando um repositório diretamente no terminal local

Devemos copiar o código indicado na figura abaixo, no caso este do Python é o seguinte:

<https://github.com/python/cpython.git>



Comando dentro do terminal local para clonar o repositório remoto

Basta definirmos em qual pasta que queremos colocar o repositório, no nosso caso aqui colocaremos o repositório na pasta C:\workspace

Definida a pasta, basta digitarmos o código **git clone** e colocarmos a **url** correspondente ao repositório remoto

git clone https://github.com/python/cpython.git

```
MINGW64:/c/workspace

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace
$ git clone https://github.com/python/cpython.git
Cloning into 'cpython'...
remote: Enumerating objects: 907560, done.
remote: Counting objects: 100% (21/21), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 907560 (delta 3), reused 7 (delta 0), pack-reused 907539
Receiving objects: 100% (907560/907560), 491.94 MiB | 2.70 MiB/s, done.
Resolving deltas: 100% (719498/719498), done.
Updating files: 100% (4926/4926), done.
```

Pronto, agora temos o repositório do Python dentro do nosso repositório local

```
MINGW64:/c/workspace

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace
$ ls
cpython/  hello.txt  livro-receitas/
```

Diferença entre baixar uma pasta comum e baixar o repositório

Quando baixamos o repositório ele vem completo como está no GitHub, dentro deste repositório temos a pasta **.Git**, dentro desta pasta contém todo o **versionamento** e contém também **para onde este repositório está apontado**.

Para navegar dentro das pastas basta utilizarmos o comando **cd**

```
MINGW64:/c/workspace/cpython

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace
$ cd cpython

Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/cpython (main)
$ ls
Doc/      Lib/      Modules/  Parser/   Tools/    configure*
Grammar/  Mac/      Objects/  Programs/ aclocal.m4 configure.ac
Include/  Makefile.pre.in PC/       Python/   config.guess* install-sh*
LICENSE  Misc/     PCbuild/  README.rst config.sub* pyconfig.h.in
```

Para apresentar repositórios ocultos

ls -a

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/cpython (main)
$ ls -a
./          .gitignore  Makefile.pre.in  Programs/      configure*
../         Doc/        Misc/            Python/        configure.ac
.azure-pipelines/ Grammar/    Modules/        README.rst     install-sh*
.editorconfig Include/   Objects/        Tools/         pyconfig.h.in
.git/       LICENSE   PC/             aclocal.m4
.gitattributes Lib/      PCbuild/        config.guess*
.github/    Mac/      Parser/         config.sub*
```

Na imagem acima temos a pasta `.git` (a pasta `.git` não é uma pasta comum, ela é um repositório)

Se dermos o comando `git remote -v`, o comando vai trazer para gente os repositórios remotos na qual o repositório que acabamos de baixar (que está no nosso local) está apontado.

```
Raphael@DESKTOP-LAG7F7L MINGW64 /c/workspace/cpython (main)
$ git remote -v
origin  https://github.com/python/cpython.git (fetch)
origin  https://github.com/python/cpython.git (push)
```

Sendo assim, qualquer mudança que fizemos localmente, o servidor remoto saberá onde colocar.