**Curso Git e GitHub**

**aulaEAD (Professor José de Assis)**

<https://www.youtube.com/watch?v=FF1f4bKYhoo&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjgIj>

Resumo do curso feito por Roberto Pinheiro

# Aula 01 - O que é git? - O que é GitHub? Porquê aprender git?

Git éum sistema de controle de versões. GitHub é uma espécie de rede social de projetos.

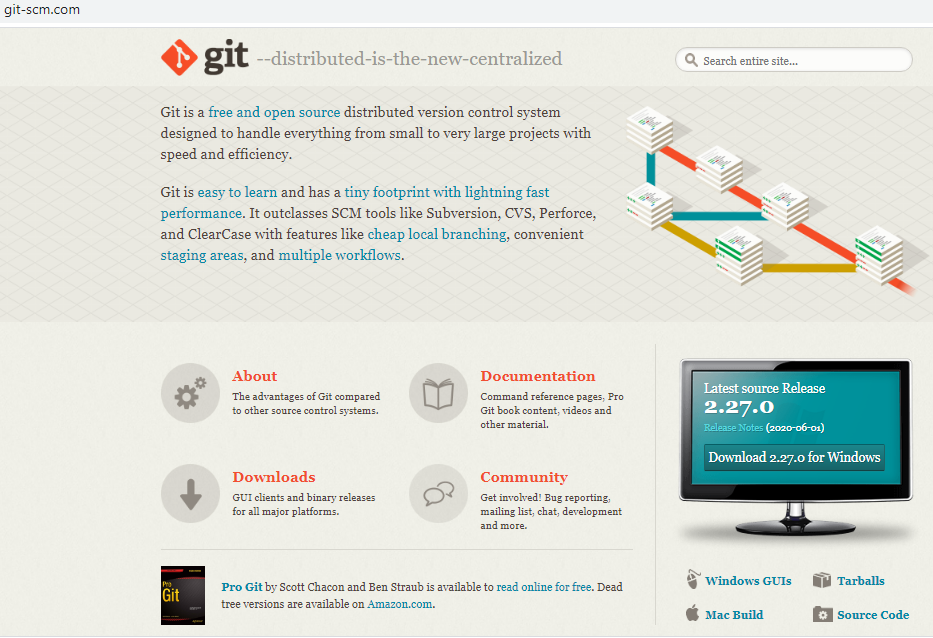
Outras alternativas ao GitHub: Subversion, Git Lab. Porém Git e Github são os mais populares e os mais usados.

Vantagens de usar essa ferramenta:

1. Backup: armazena uma cópia de seus projetos nas nuvens.
2. Controle de versões: o Git documenta e armazena cada modificação que é feita no código, através de commits.
3. Trabalhar em equipe: é uma das maiores vantagens. Um projeto pode ser clonado e trabalhado em equipe.
4. Portfolio: permite criar um portfolio com todos os seus projetos.

# Aula 02 - Instalação e configuração do GIT no Windows

- Acesse o endereço git.scm.com, faça o download do aplicativo e instale em seu computador.



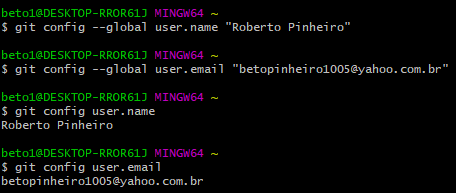
## Verificando a versão do Git instalado



## Configurações iniciais

git config --global user.name "Roberto Pinheiro"

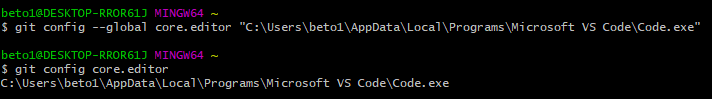
git config --global user.email "betopinheiro1005@yahoo.com.br"



## Configurando o editor padrão

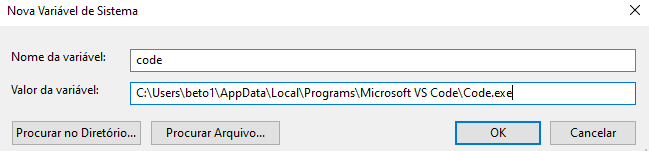
git config --global core.editor "C:\Users\beto1\AppData\Local\Programs\Microsoft VS Code\Code.exe"

git config core.editor

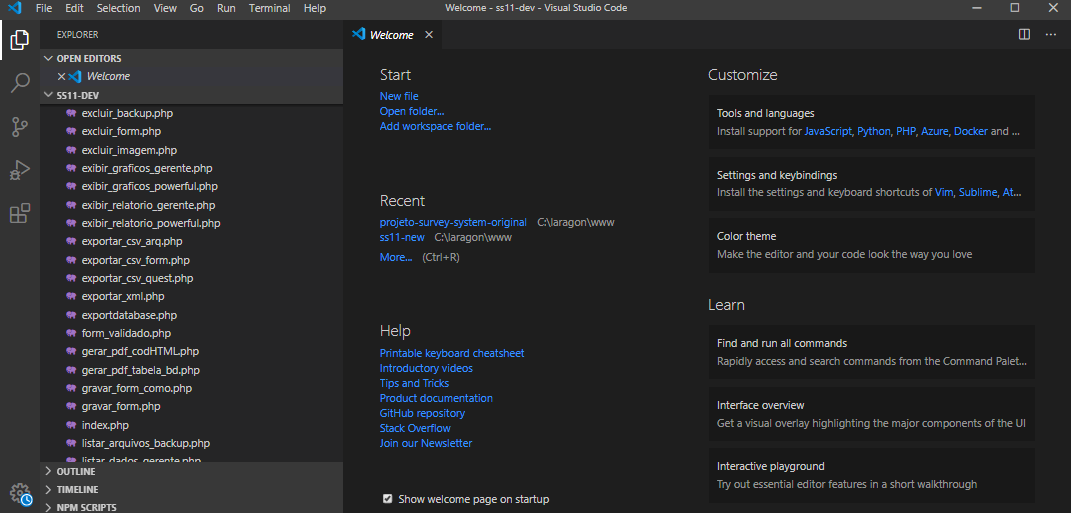


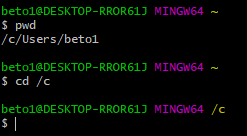
Em “variáveis de ambiente”, criar uma variável de sistema com o nome "code" e com o seguinte caminho:

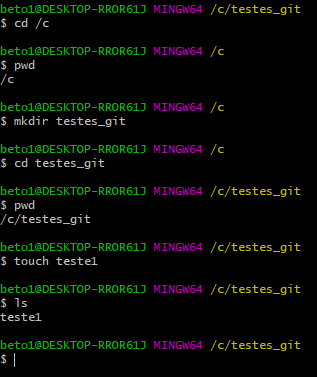
C:\Users\beto1\AppData\Local\Programs\Microsoft VS Code\Code.exe





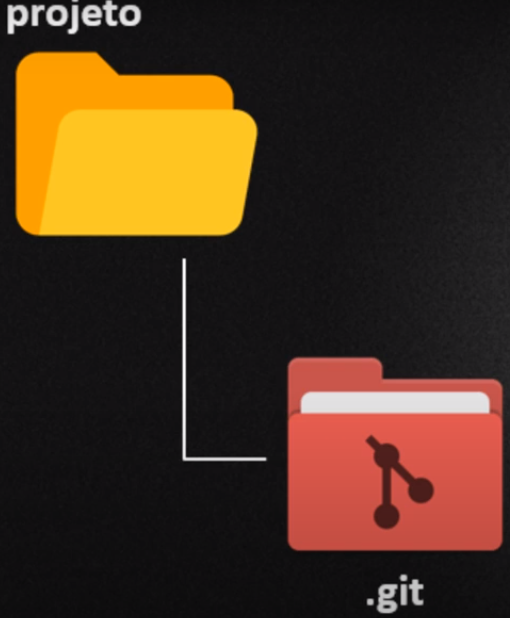






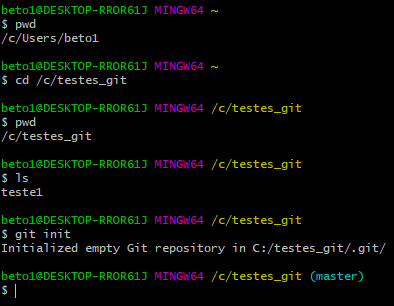
# Aula 03 - Iniciando um repositório local / comandos básicos

O Git cria um repositório local no diretório do projeto. Este repositório local pode ser clonado posteriormente para um repositório remoto. Pode também ser feito o oposto.

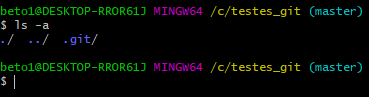


Para criar um repositório local, na pasta do projeto digite:

git init



ls –a



## Grafo de commits

Grafo é uma estrutura que contém objetos relacionados. Commit é um comentário que se escreve sempre que se modifica alguma coisa no projeto.



O Git precisa gerenciar as mudanças do projeto. Para isso existe uma espécie de container onde devemos colocar todos os arquivos e pastas que foram criados ou modificados. Isso é feito com o comando:

git add

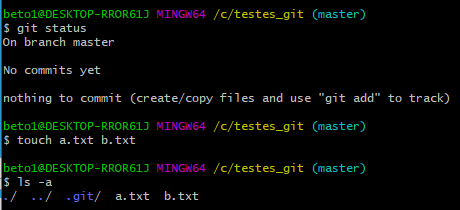
Para verificar o que está armazenado no container é usado o comando:

git status

Para identificar e armazenar esse container no repositório local é usado o comando:

git commit

O .git é o repositório e dentro dele serão armazenados os containers que são identificados. Dentro desses containers há o conteúdo do seu projeto.

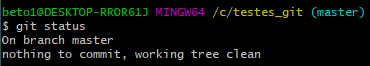


## Adicionando arquivos ao container

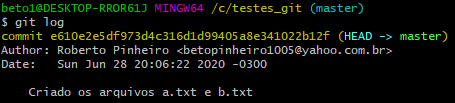


## Identificando e armazenando o container no repositório



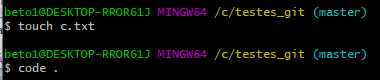


## Gerenciando o repositório

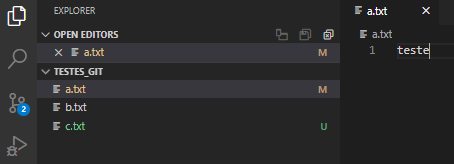


## Fazendo alterações no projeto

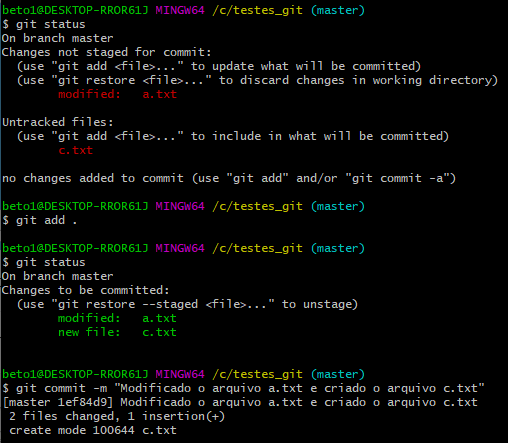
### Criando um novo arquivo



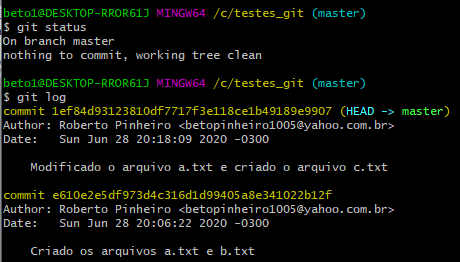
### Alterando o arquivo a.txt



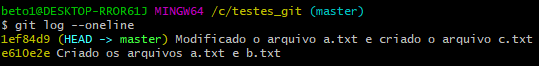
### Criando um novo container e adicionando-o ao repositório



## Gerenciando os commits criados



### Exibindo os logs de uma maneira mais simples



# Aula 04 - Navegando entre as versões do projeto

## Controle de versões

* Rastrear mudanças
* Desfazer alterações
* Recuperar versões anteriores do projeto

## Head → Master

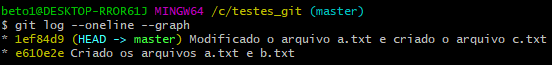
É pelo head que conseguimos rastrear os commits e recuperar uma versão anterior do projeto.

Master é o ramo principal.

git log --graph

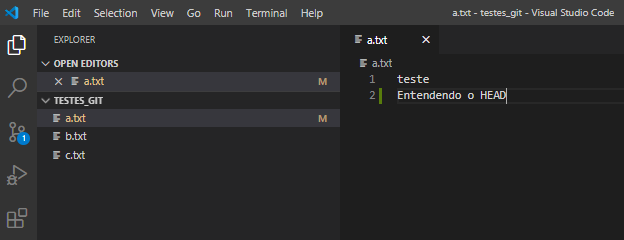
|  |  |
| --- | --- |
| e610e2e  1ef84d9 |  |





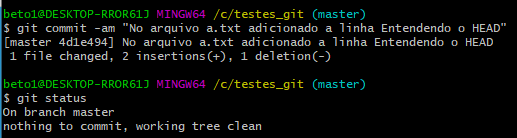
Head é o ponto onde o seu projeto se encontra no grafo de commits. Sempre vai ser o último commit.

.code

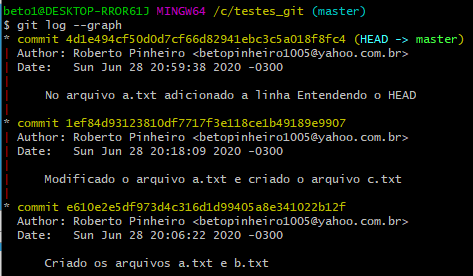


Para adicionar ao container e simultaneamente criar um novo commit

git commit –am “No arquivo a.txt adicionado a linha Entendendo o HEAD”



### Gerenciando os commits criados



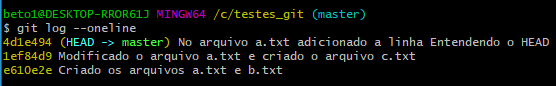
O Git não gera nova cópia do projeto a cada novo commit. O Git rastreia as alterações do projeto, armazenando de forma inteligente somente as mudanças que foram feitas.

### Identificando o ramo em que o projeto se encontra

Para identificar o ramo em que o projeto se encontra entre com o seguinte comando:

git branch





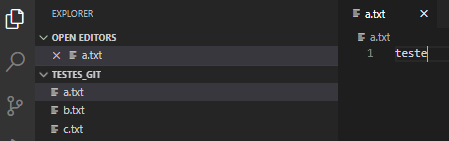
|  |  |
| --- | --- |
| e610e2e  1ef84d9  4d1e494 |  |

## Rastreando as mudanças e recuperando uma versão anterior

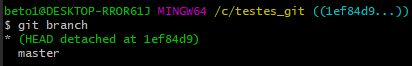
Devemos apontar o HEAD para o respectivo commit.

git checkout 1ef84d9



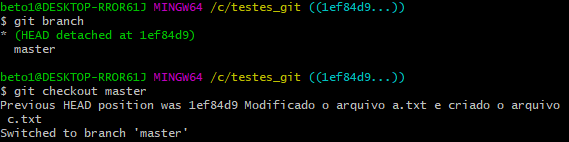


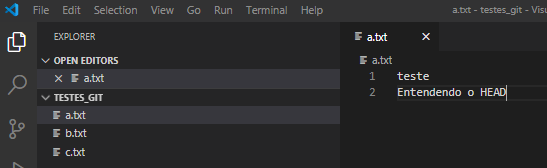


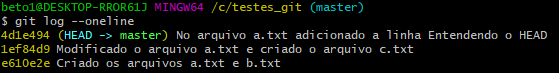


## Retornando à última versão

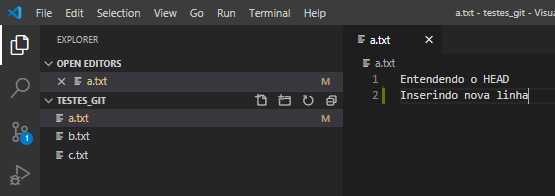
git checkout master





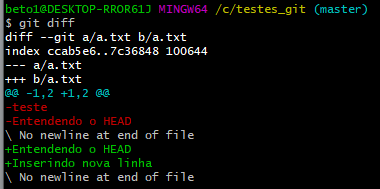


## Desfazendo alterações



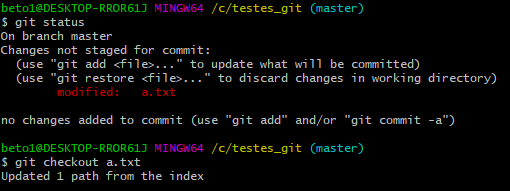
### Visualizando as últimas mudanças antes do commit

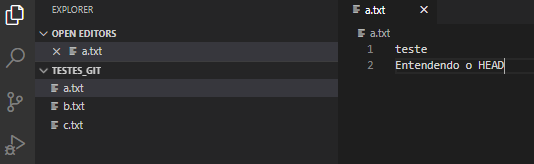
git diff



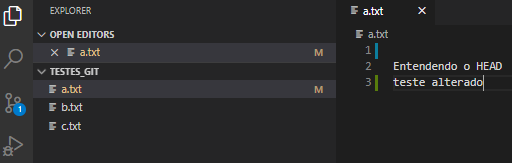
Para descartar as mudanças feitas use o comando:

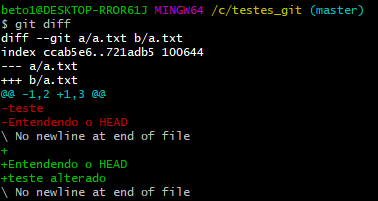
git checkout <file>





### Modificando novamente o arquivo a.txt

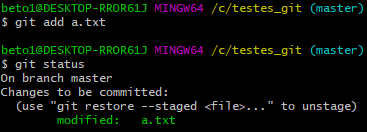




### Adicionando o arquivo a.txt alterado ao container



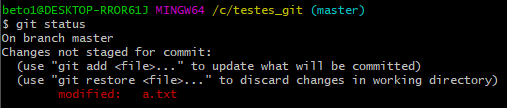
### Adicionando o conteudo ao container

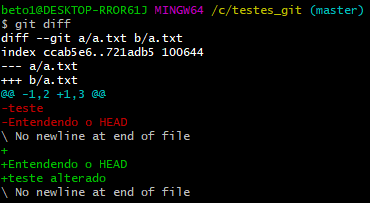


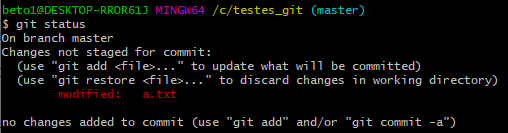
### Removendo o conteúdo do container

git reset HEAD









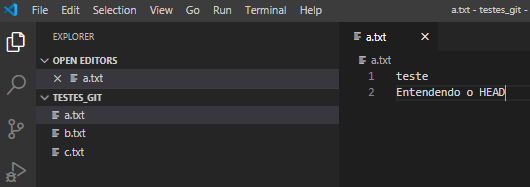
### Criando um novo commit

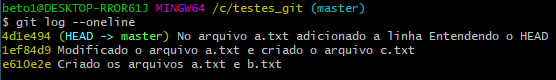


## Removendo um commit

git reset --hard 4d1e494







# Aula 05 - Criando ramificações do projeto

Criar uma ramificação permite que testes e mudanças no projeto sejam feitas sem interferir no projeto original.

É possível fazer a fusão das ramificações implantando as mudanças em uma única versão do projeto.

Para verificar o ramo em que se encontra digite:

git branch

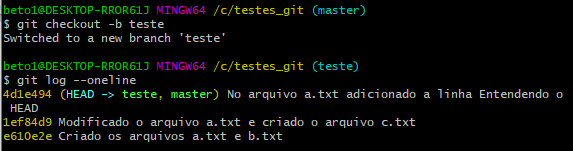


Por padrão, quando se cria um projeto, o Git automaticamente cria o ramo “master”.

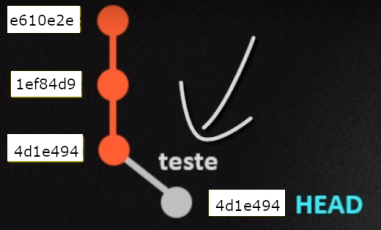
## Criando uma nova ramificação

Para criar uma nova ramificação do projeto, entre com o comando:

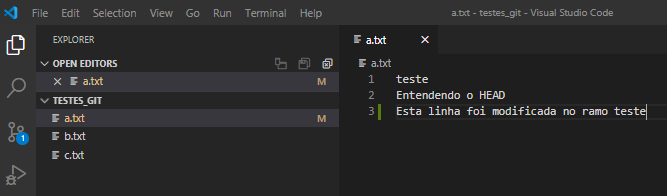
git checkout –b <nome\_ramo>

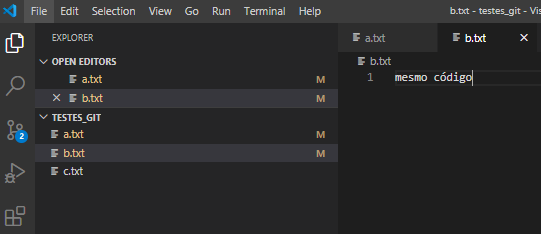


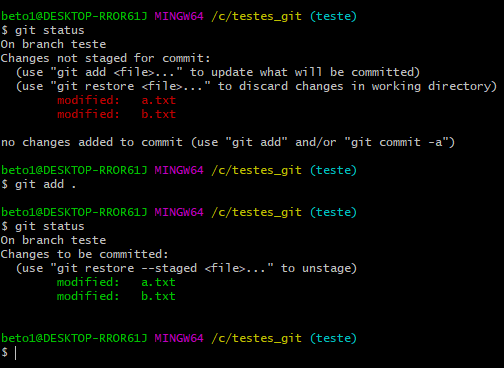


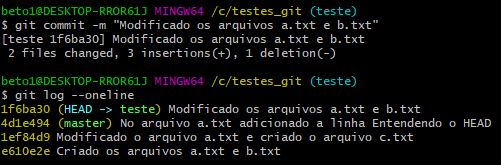


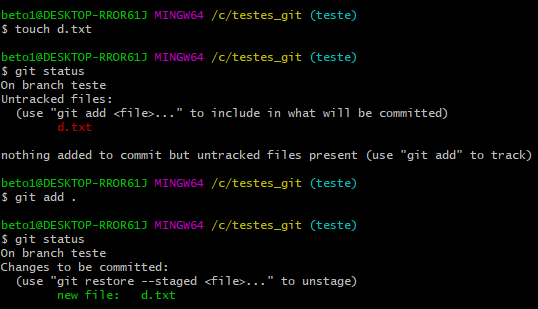
code .

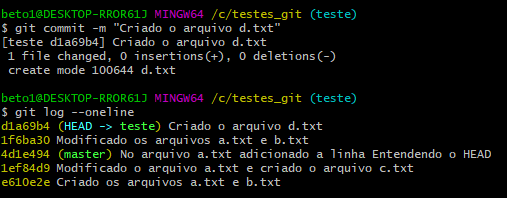








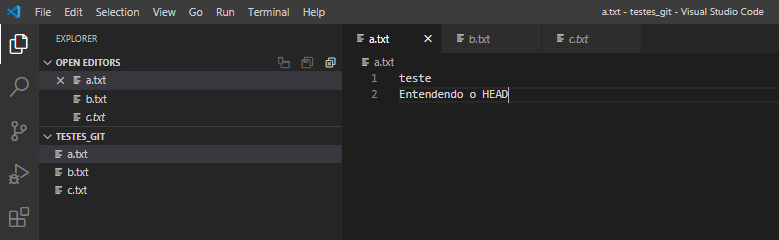




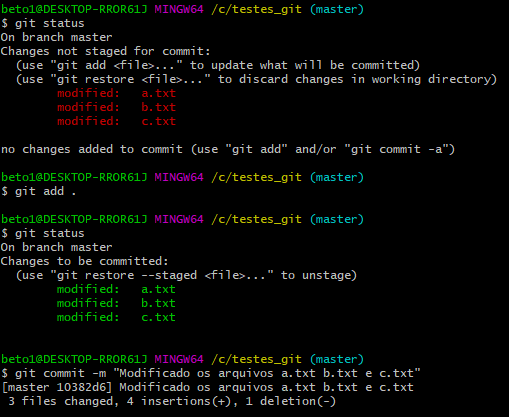
## Retornando ao ramo master

git checkout master

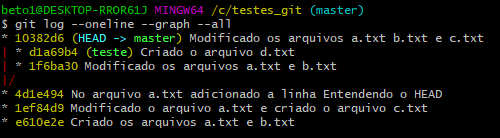




### Fazendo alterações nos arquivos do ramo master



git log --oneline --graph --all

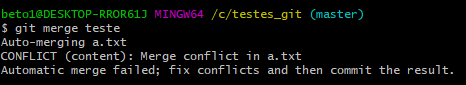


## Unindo o ramo teste ao ramo master (fusão dos dois ramos)

É necessário estar no ramo master.

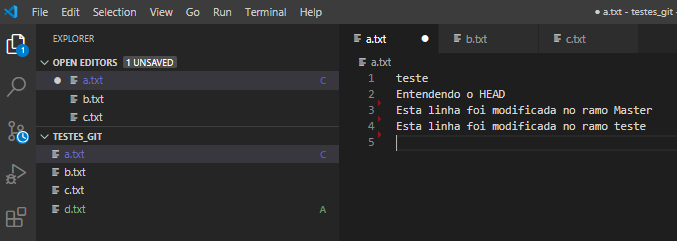
git merge <ramo>

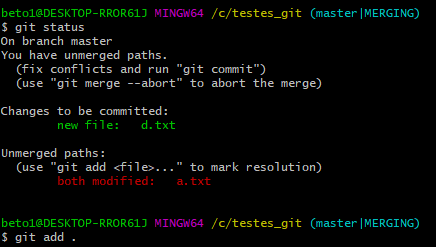
git merge teste

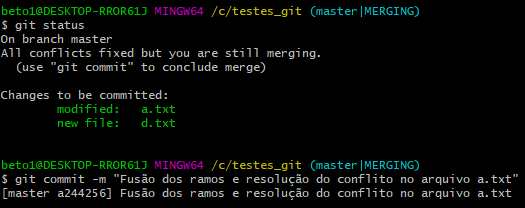


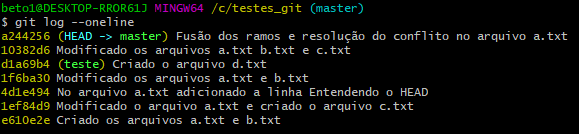
### Resolvendo o conflito

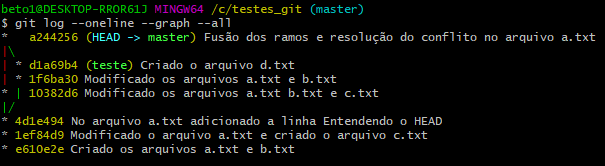






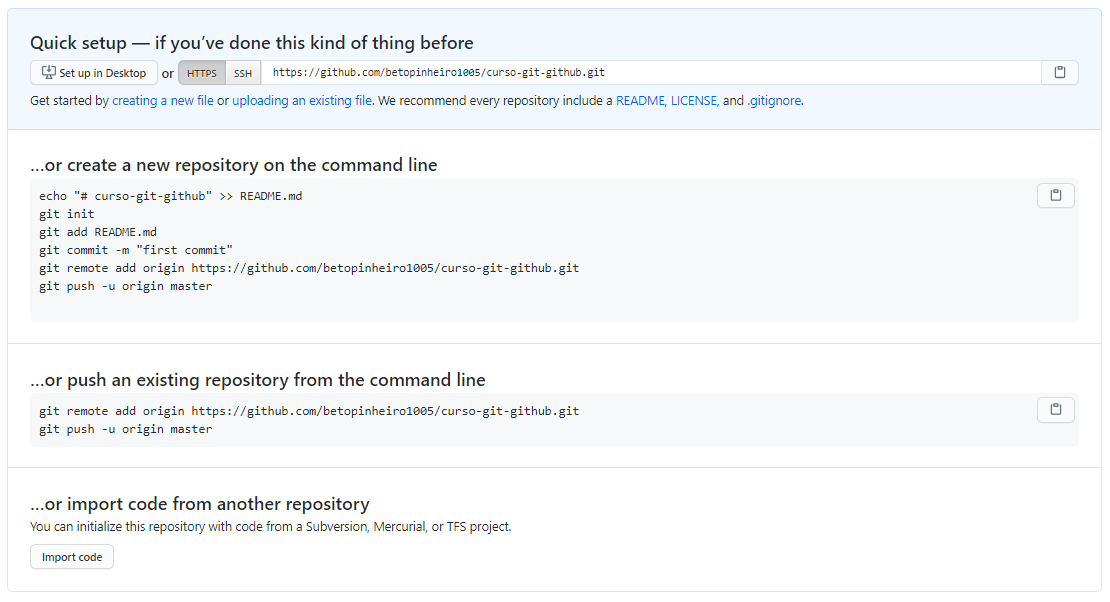






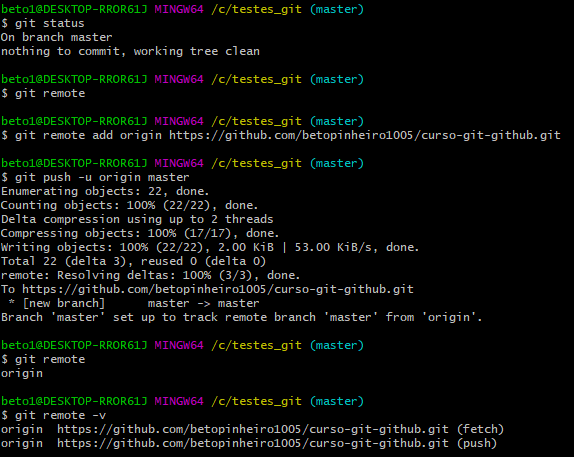
# Aula 06 - Enviando um projeto local para o GitHub

## Ligando o projeto ao repositório remoto

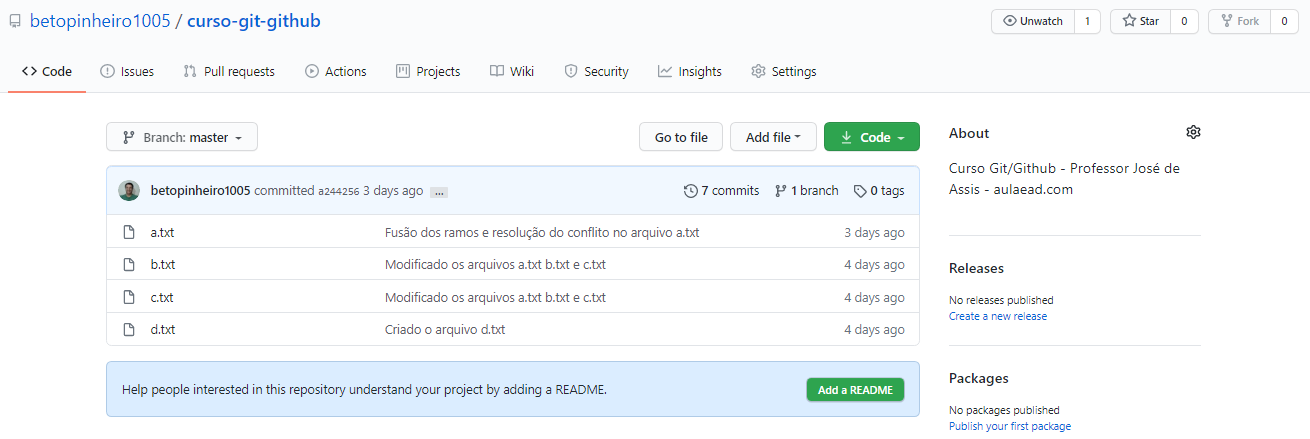


git remote add origin https://github.com/betopinheiro1005/curso-git-github.git

git push -u origin master

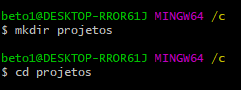


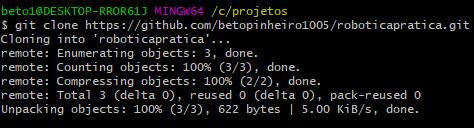
Atualizando a página no Github:

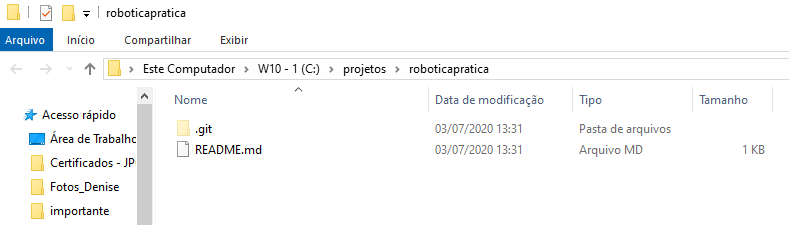


# Aula 07 - Criando um repositório no Github e clonando no Windows

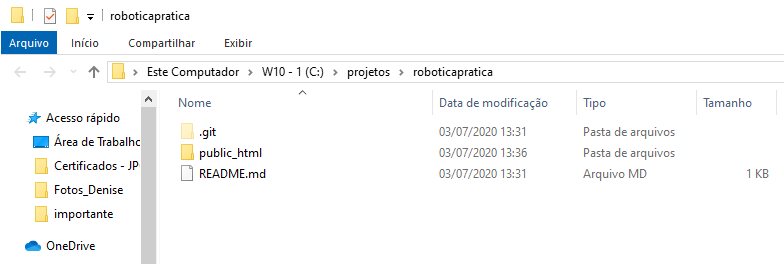




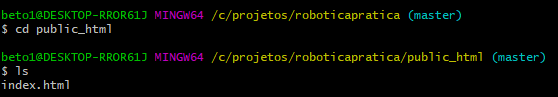


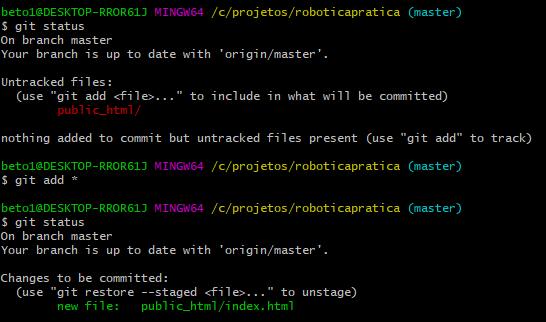


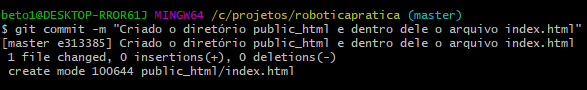


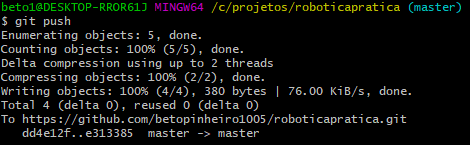




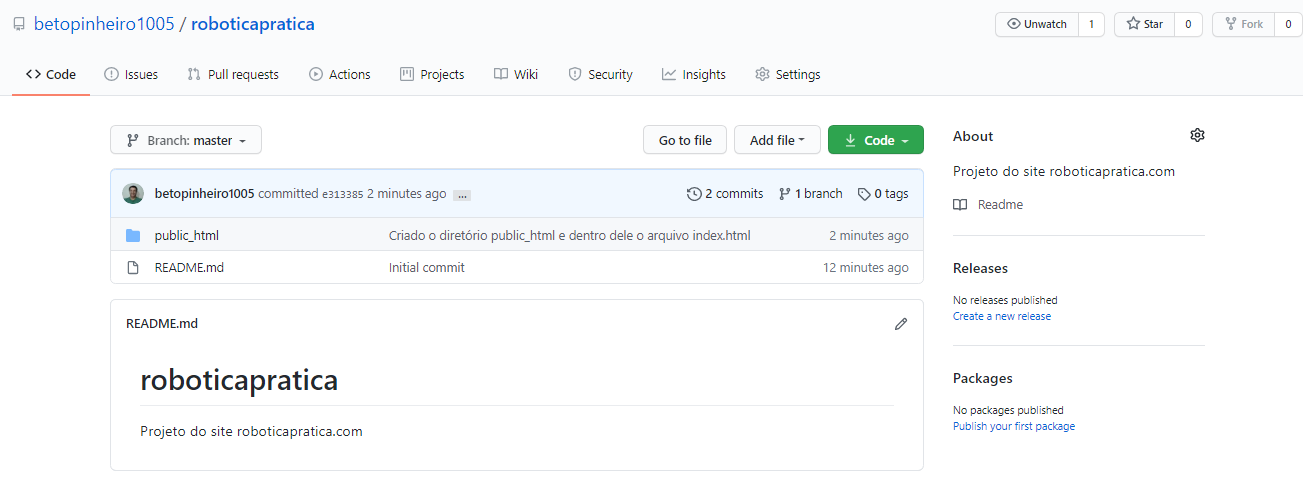


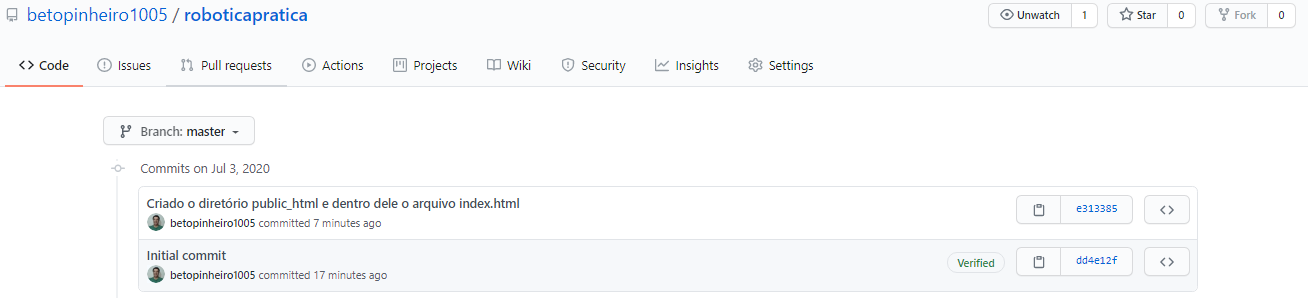






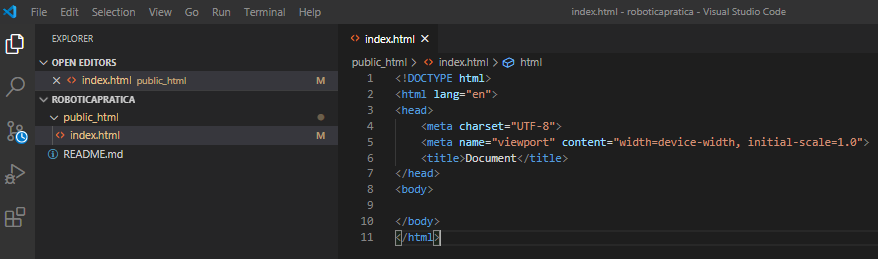
Recarregando a página no Github:

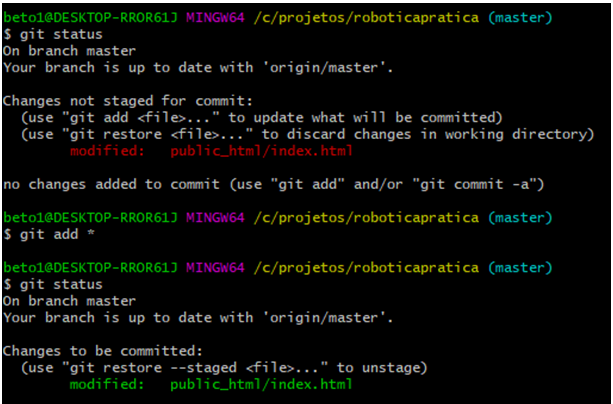


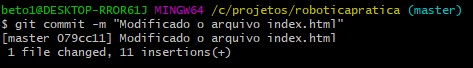


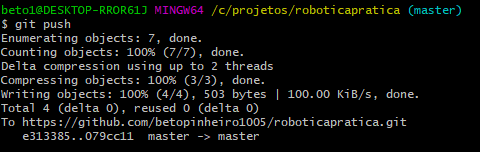
code .

html:5

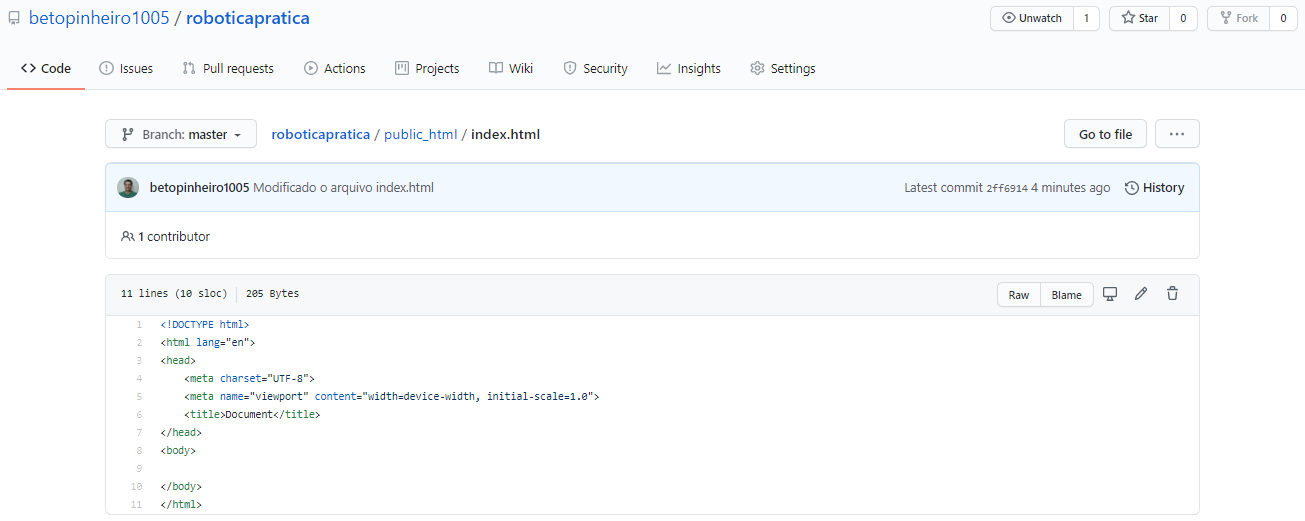








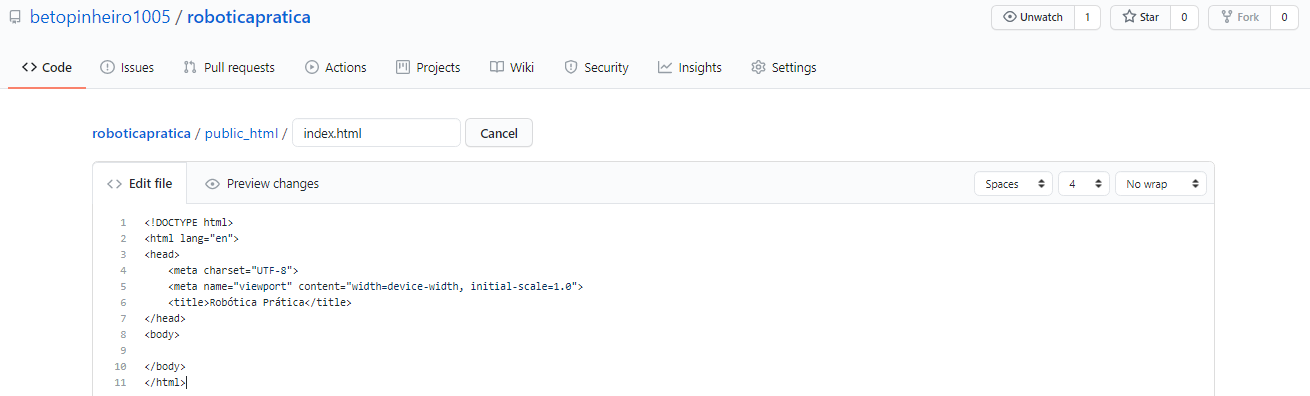
Recarregue a página no Github:

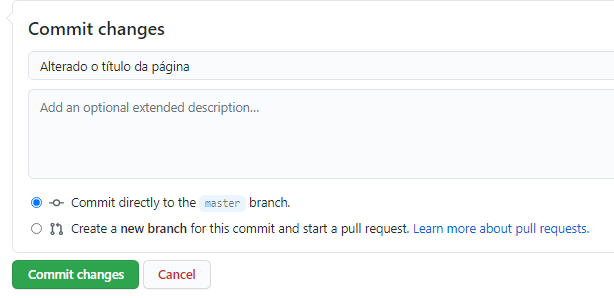


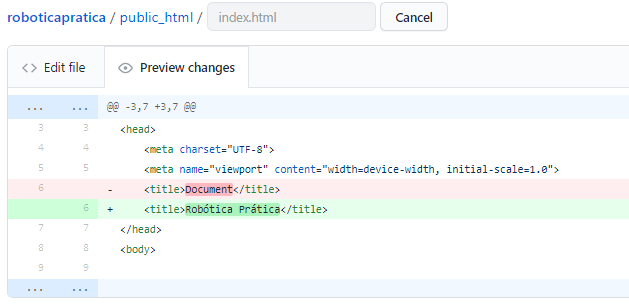
# Aula 09 - Criando e editando arquivos e pastas no GITHUB - sincronizando alterações

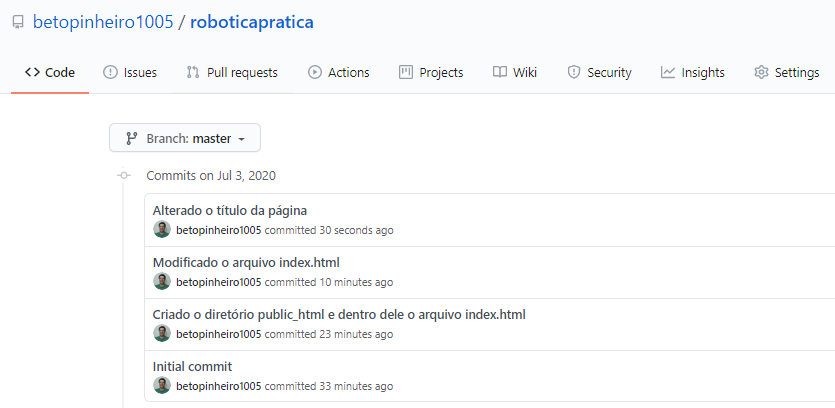
## Criando e editando arquivos e pastas no repositório remoto

### Alterando o título da página index.html

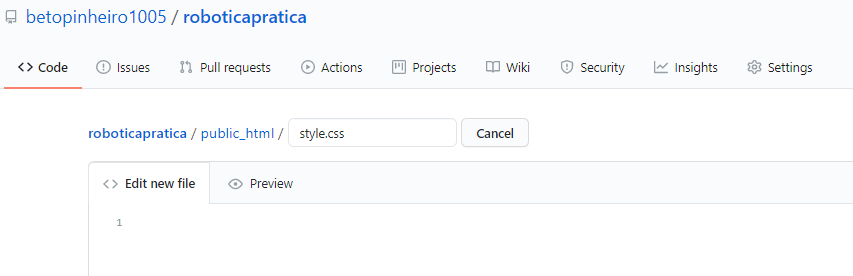
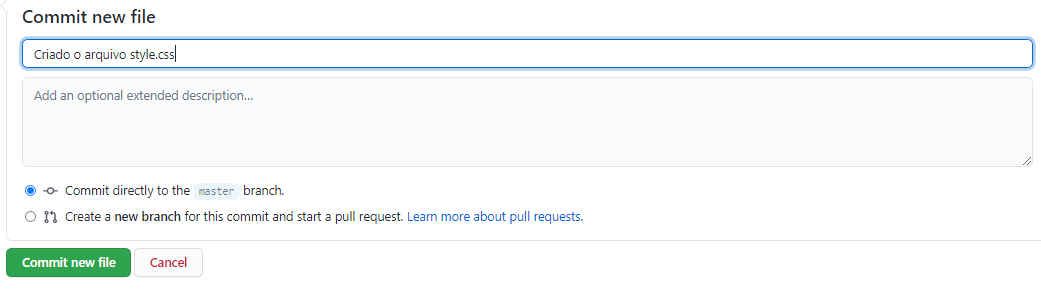


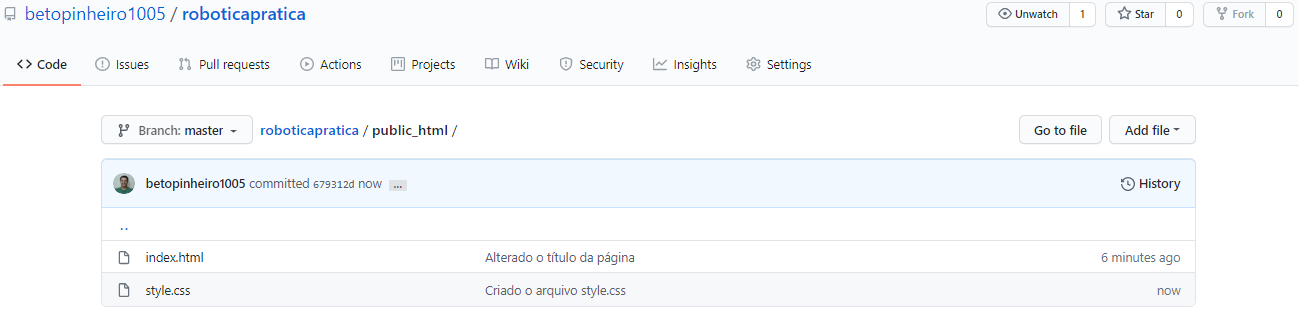




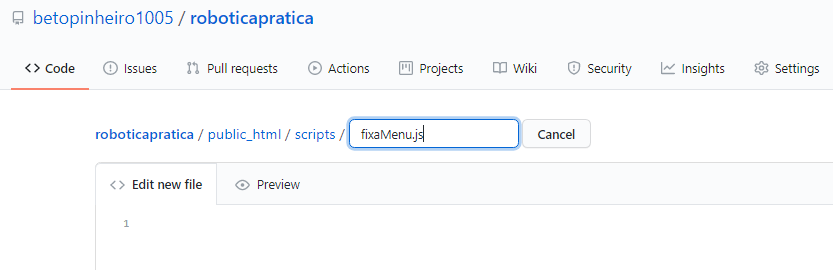


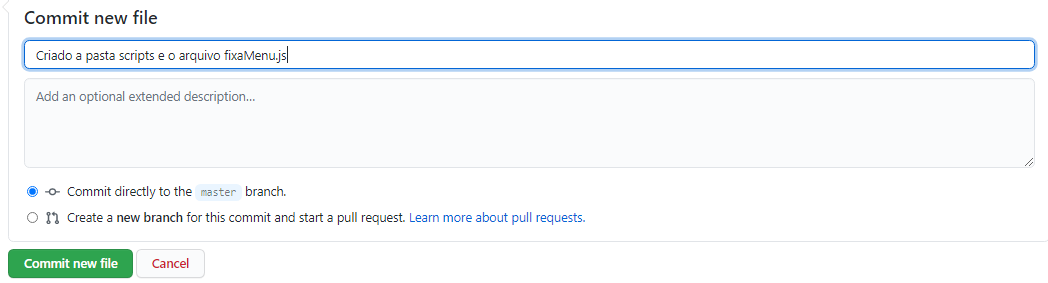
### Criando um novo arquivo

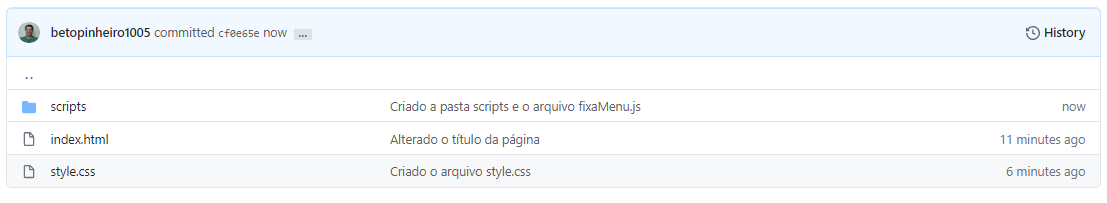


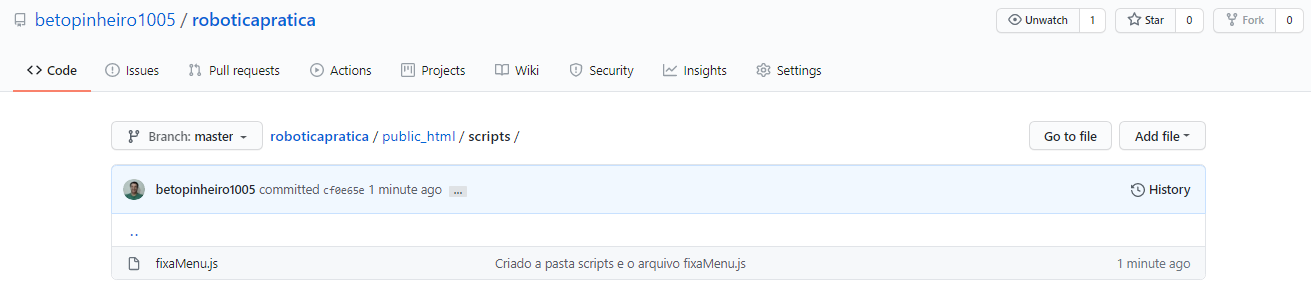


### Criando uma nova pasta





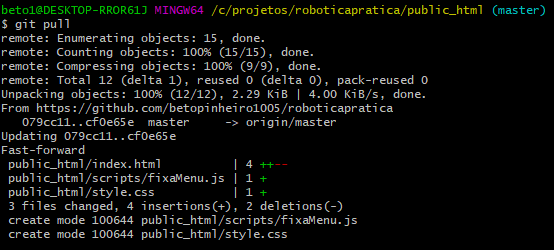


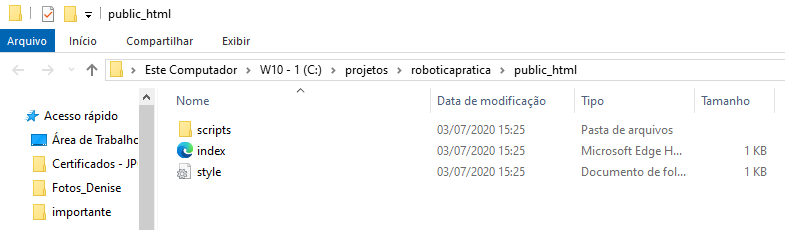


## Sincronizando as mudanças feitas no Github com o repositório local

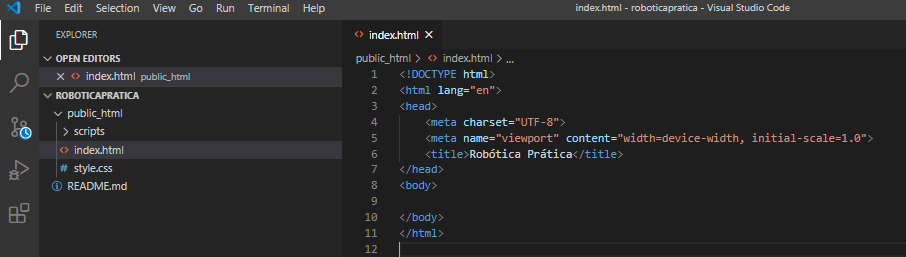
No terminal bash (do Git), entre com o seguinte comando:

git pull





No Visual Studio Code, o título da página já aparece alterado:



# Aula 10 - Trabalhando com vários repositórios locais - Resolvendo conflitos

Vamos ver como resolver problemas de conflitos quando trabalhamos com a sincronização de um repositório remoto com vários repositórios locais.

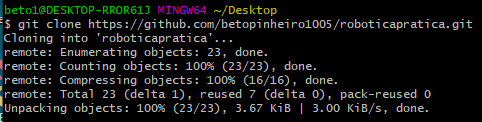
Como exemplo prático vamos simular alguns conflitos do projeto do site roboticapratica.com.br

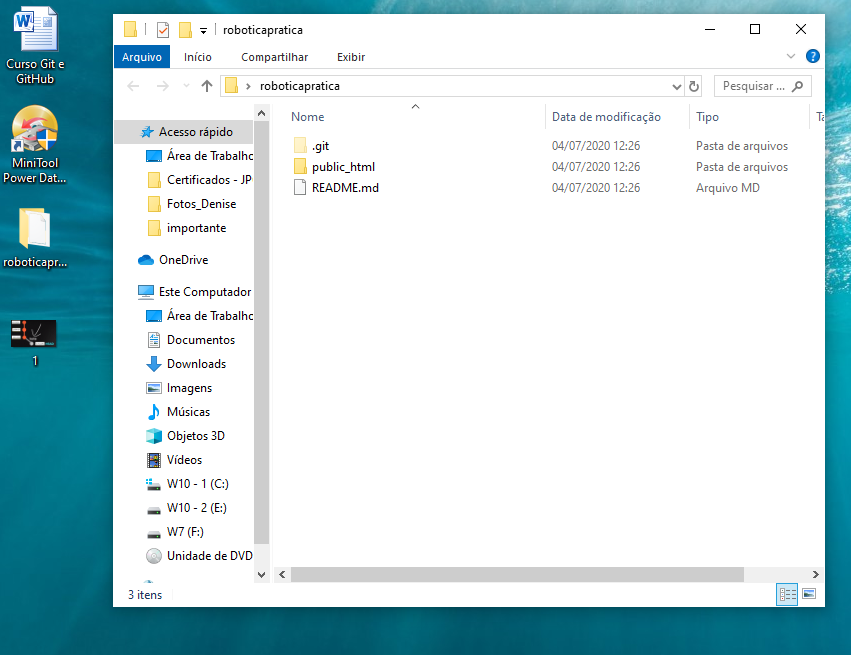
A ideia é trabalhar este mesmo projeto no seu computador pessoal e também no computador de informática da sua faculdade ou do seu escritório.

## Clonando o projeto na área de trabalho

Simulando o trabalho em seu PC, clique com o botão direito do mouse na área de trabalho e selecione a opção “Git Bash Here”. Ao abrir o terminal, entre com o comando:

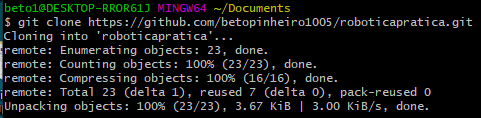
git clone https://github.com/betopinheiro1005/roboticapratica.git

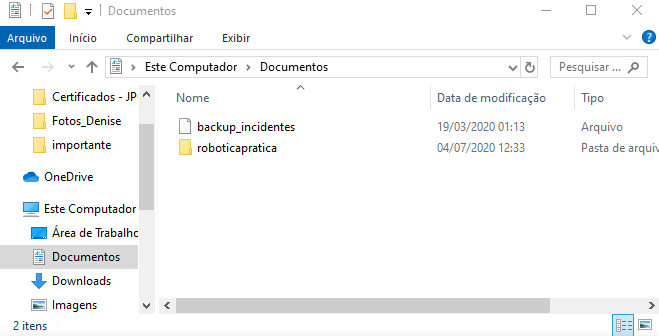




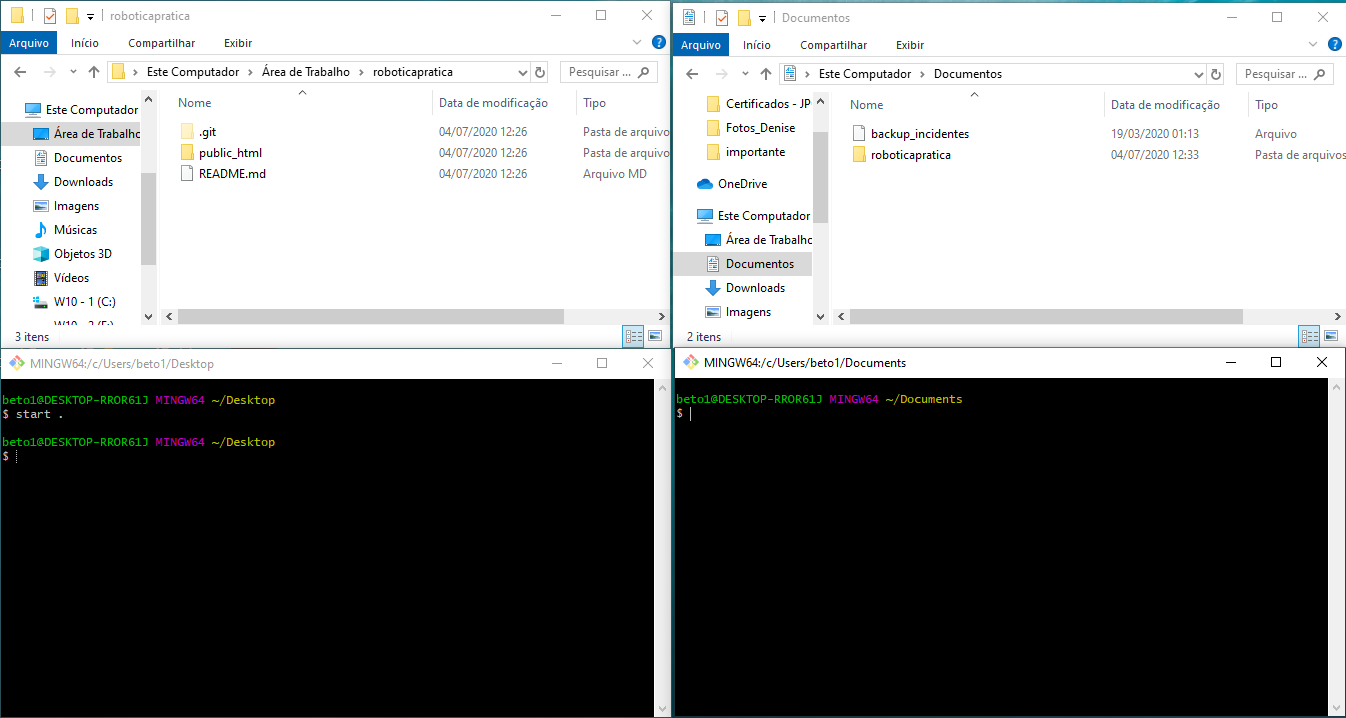
## Clonando o projeto na pasta Documents

Simulando o trabalho no laboratório de informática de sua faculdade, repita a operação na pasta “Documents”:





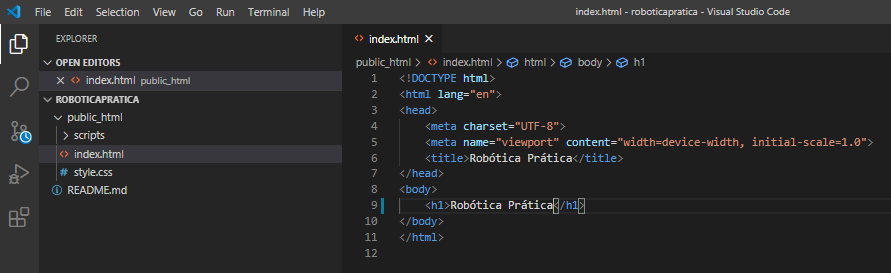
### Visualizando os dois ambientes simultaneamente



No Desktop:

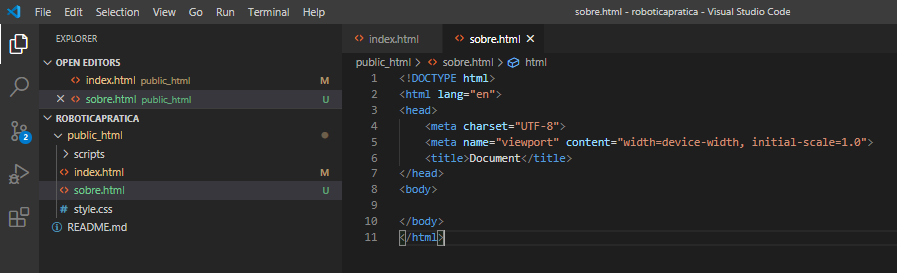


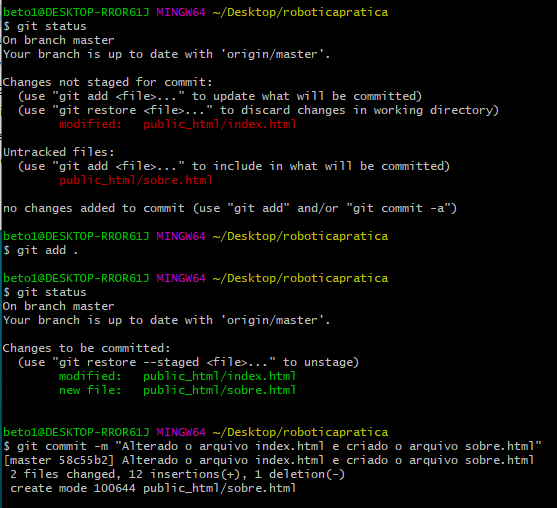
Alterar o arquivo index.html:

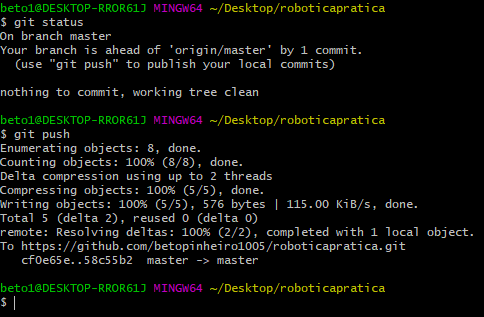


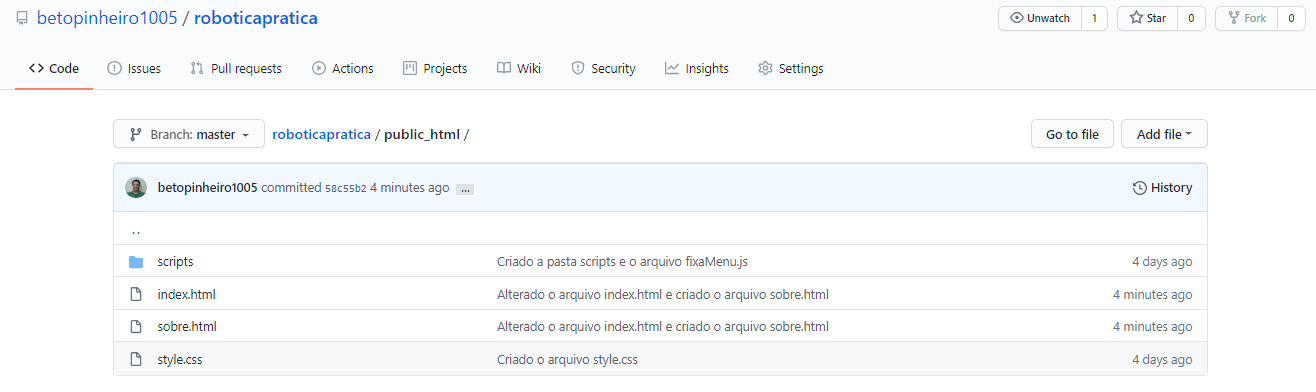
Criar o arquivo sobre.html:

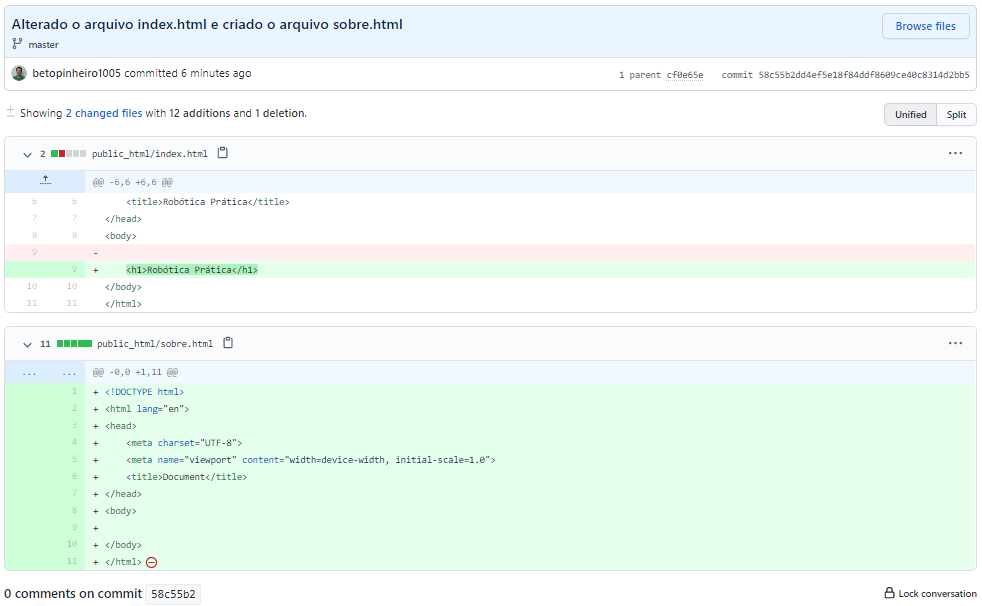
html:5











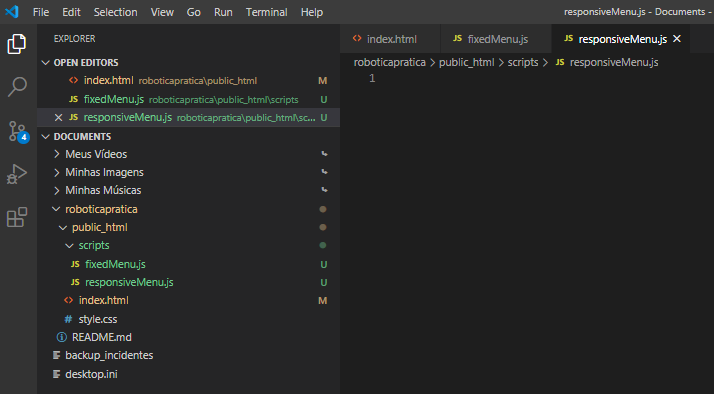
Na pasta “Documents”:

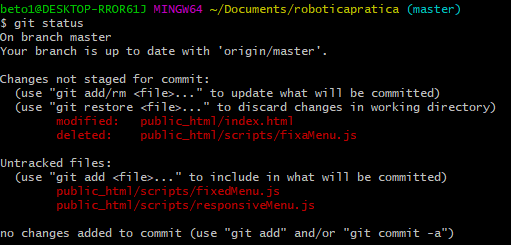
Alterar o arquivo index.html:

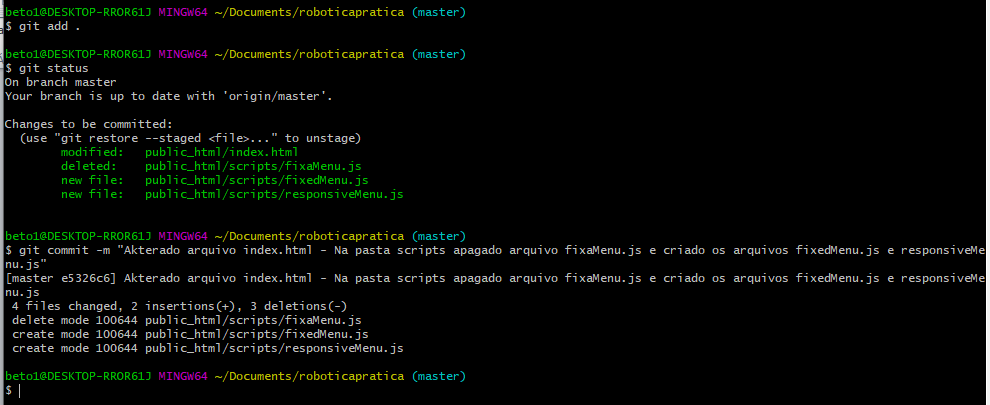




Dentro da subpasta “scripts” apagar o arquivo “fixaMenu.js” e criar o arquivo “fixedMenu.js” e o arquivo “responsiveMenu.js”

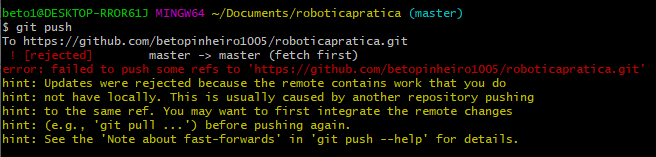






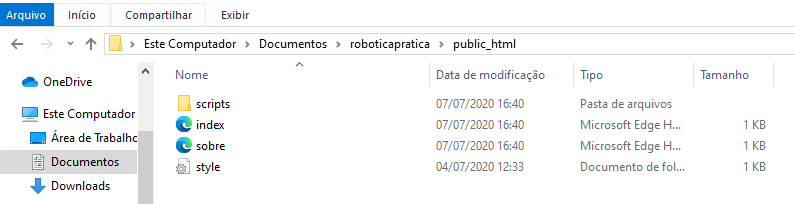
### Sincronizando essa alterações com o repositório remoto

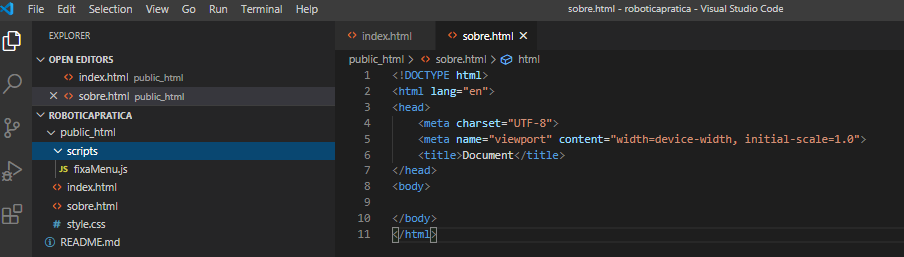
Como há conflitos entre o conteúdo local e o do repositório remoto:



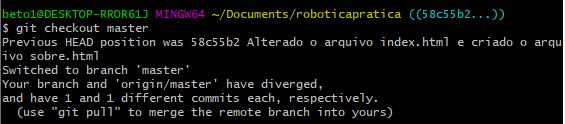
git fetch

Este comando é usado para fazer o download das alterações que estão no repositório remoto para poder fazer uma análise do que foi modificado e resolver os problemas de conflito. Equivale ao git pull porém ele não faz o merge (fusão dos projetos).

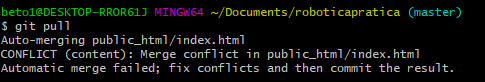


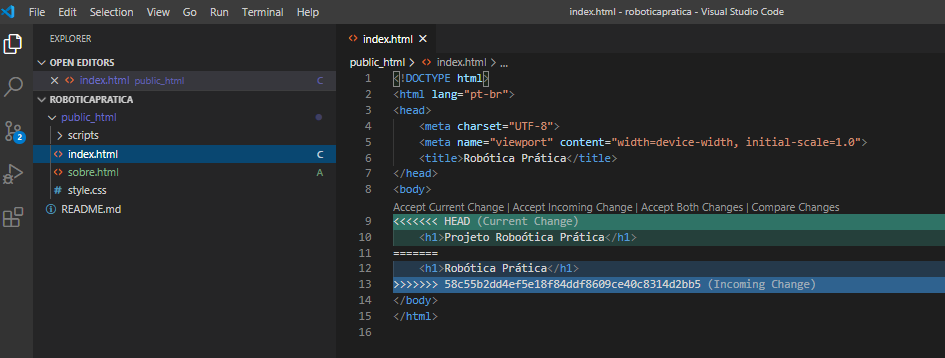


git checkout master



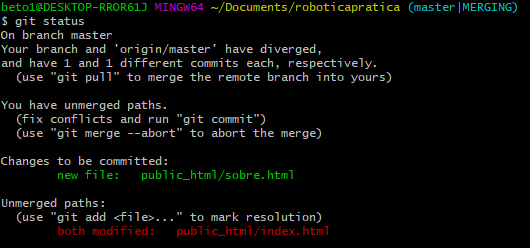
git pull

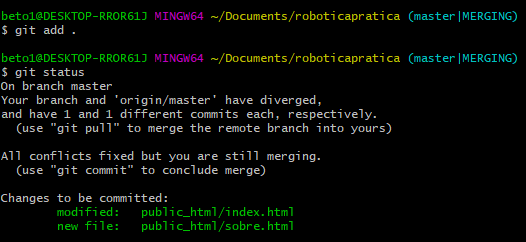


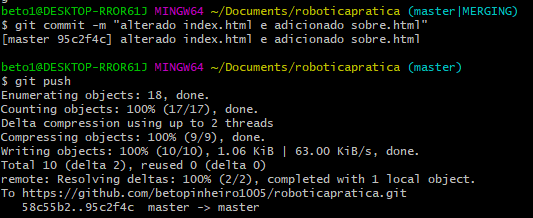


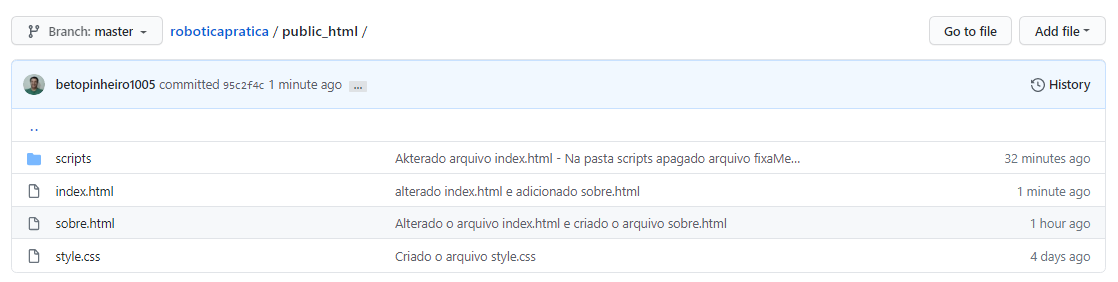
Clicar em “Accept incoming Change”:





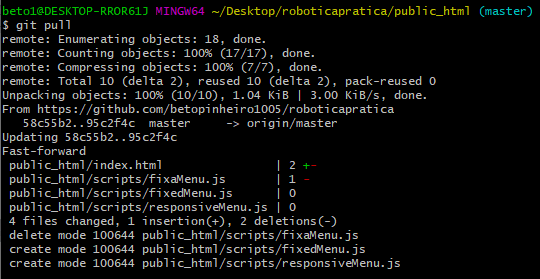


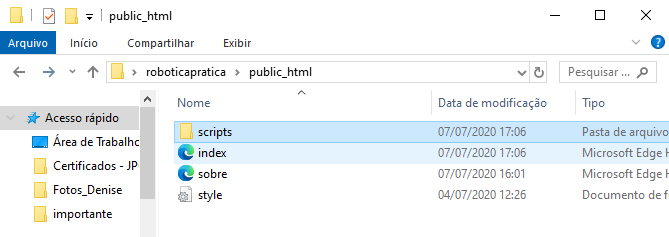


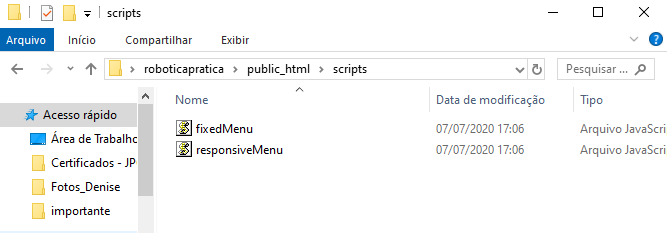


### Sincronizando o repositório remoto com o repositório local (Desktop)

git pull





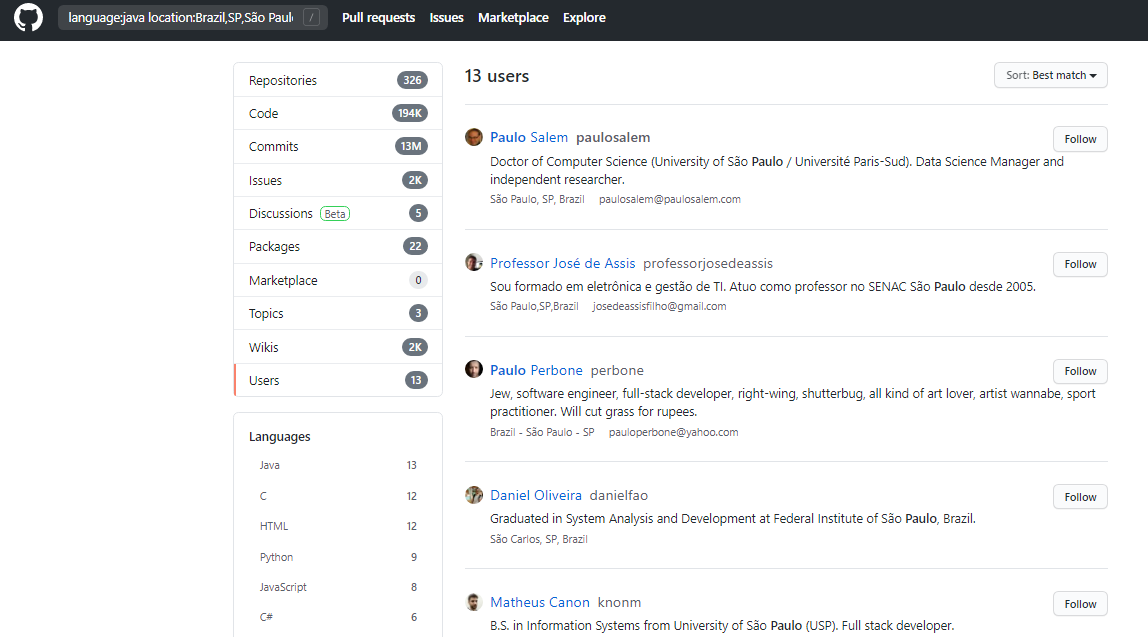


Agora o repositório remoto está sincronizado corretamente com todos os repositórios locais (Desktop e pasta Documents).

# Aula 11 - Usando o GITHUB como currículo e portfólio

## Github como currículo

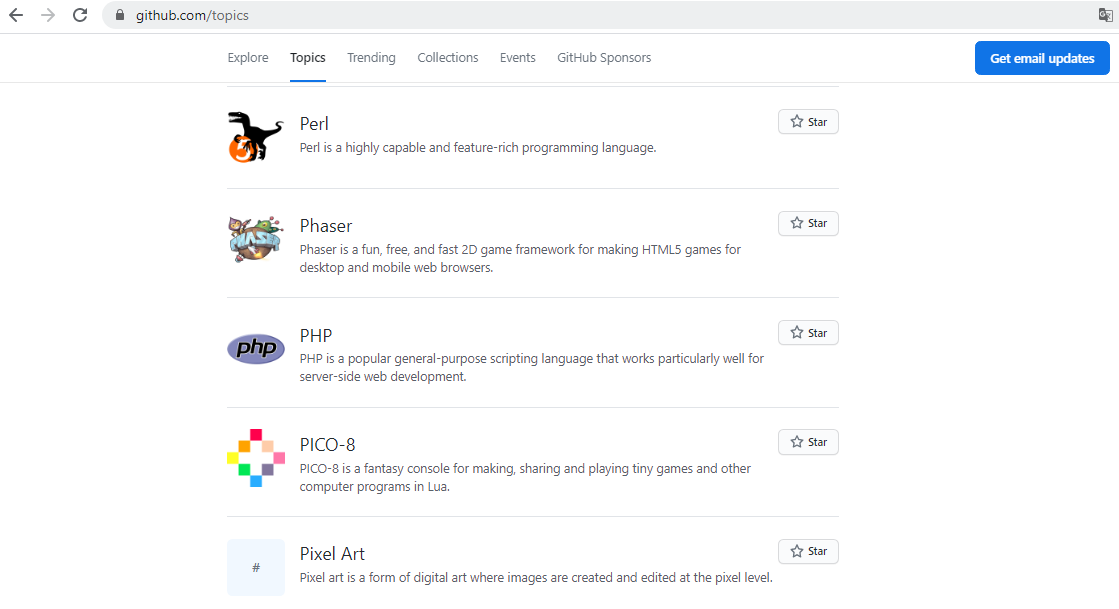
### Como as empresas encontram os profissionais

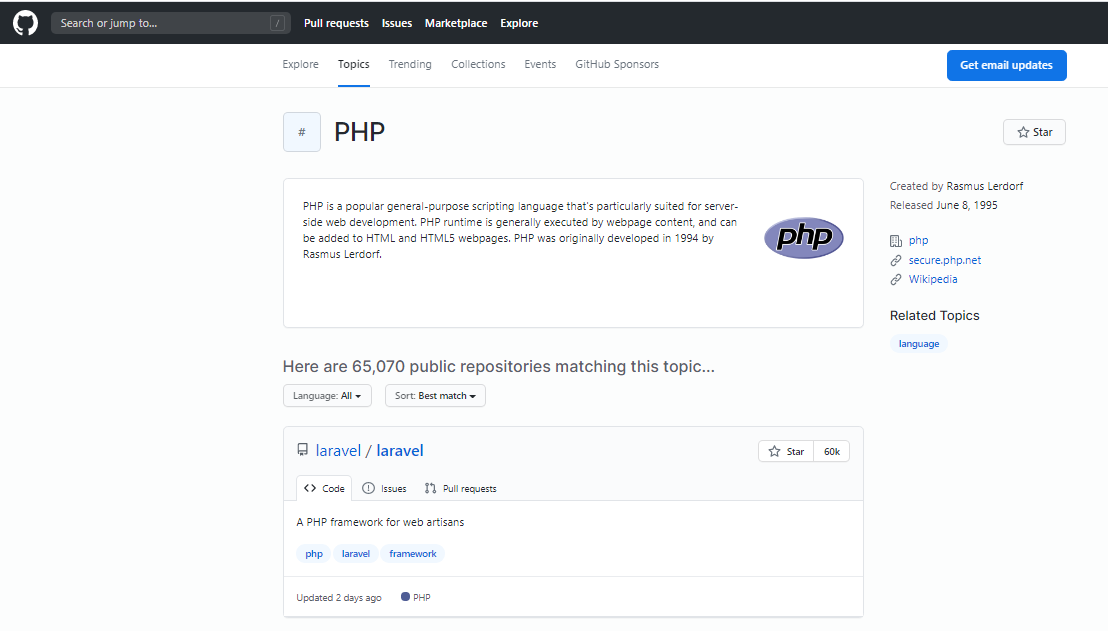


### Pesquisa usando Topics

Clique no link “Explore” e posteriormente no link “Topics”.

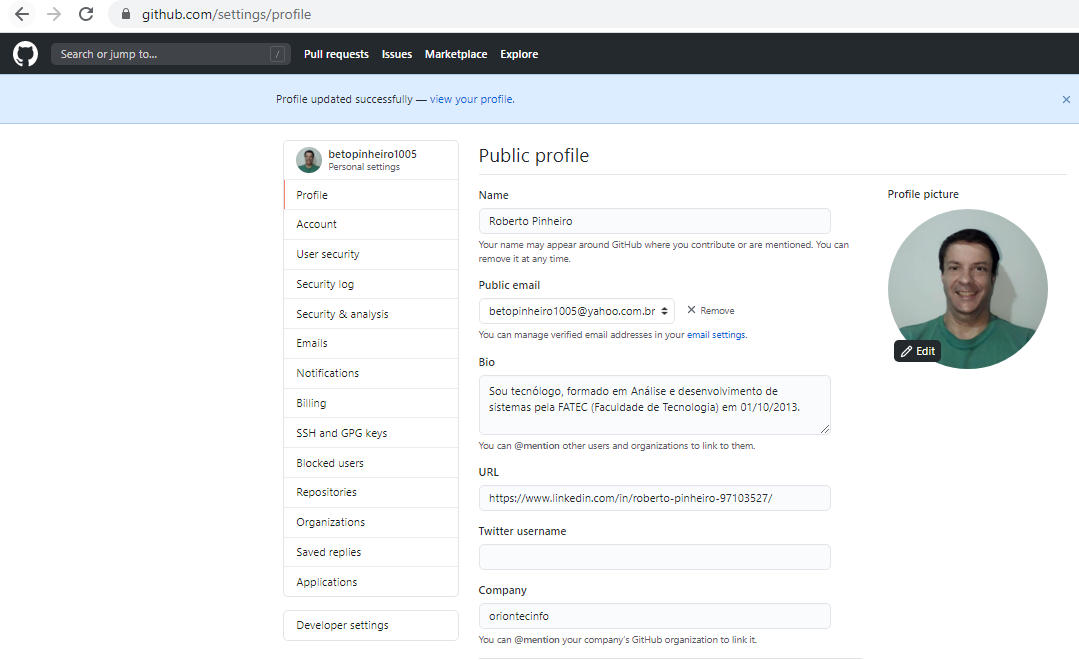
Ex: Filtrando por PHP

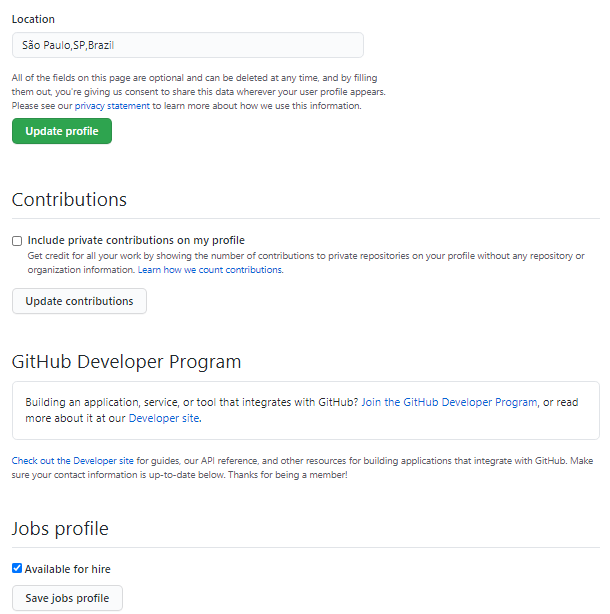


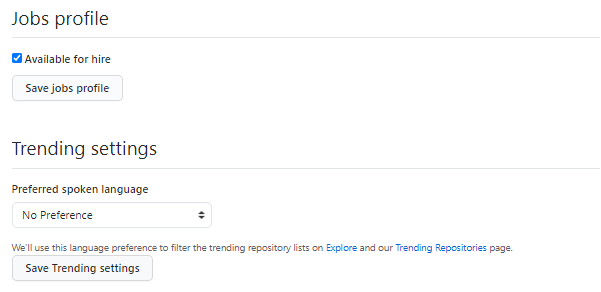


## Configurando o seu perfil

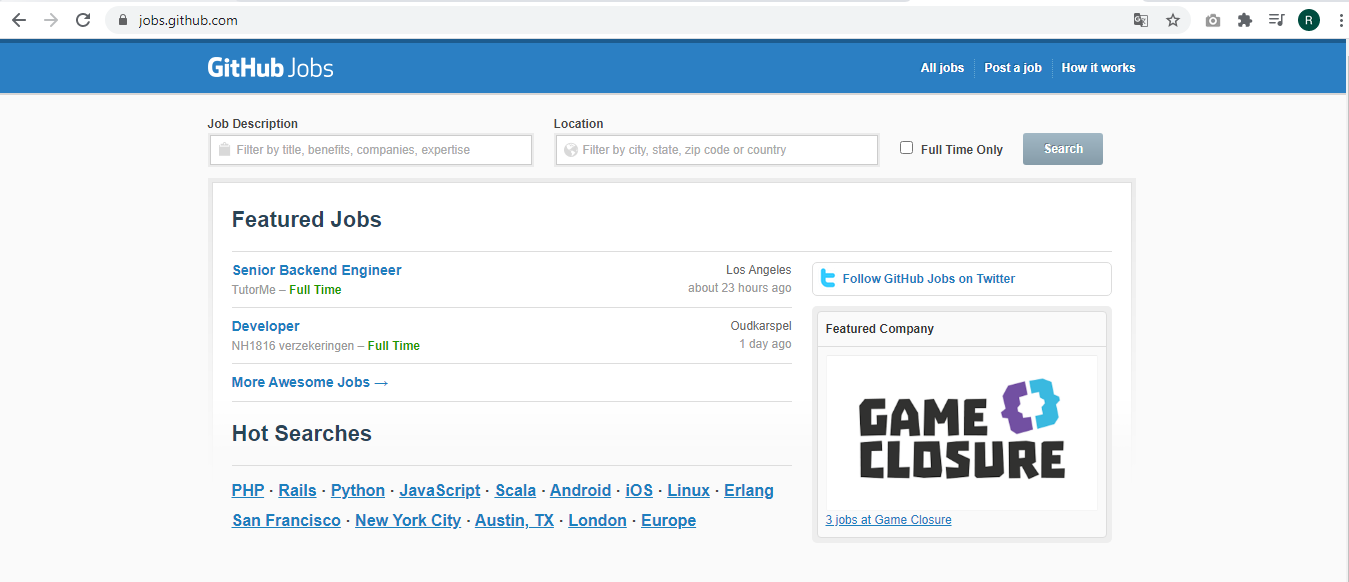
Clique em “Settings”.

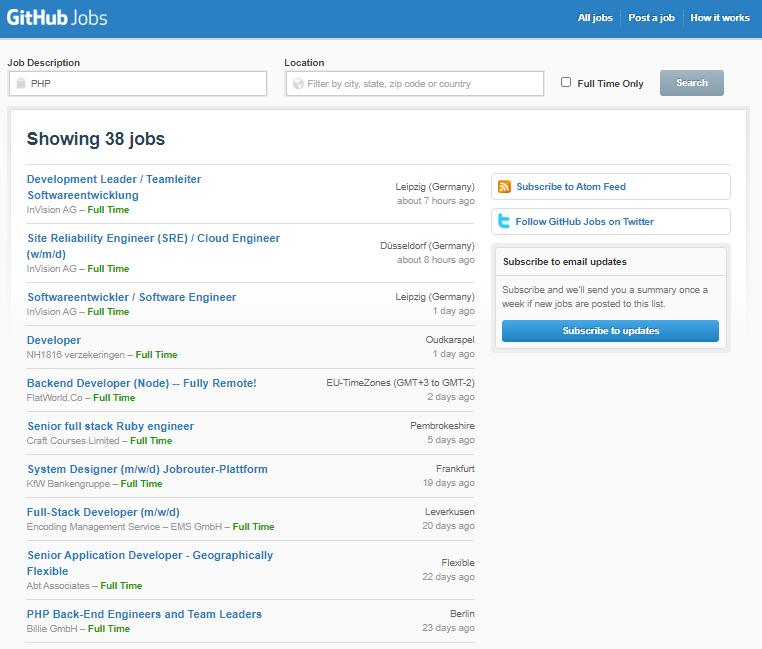




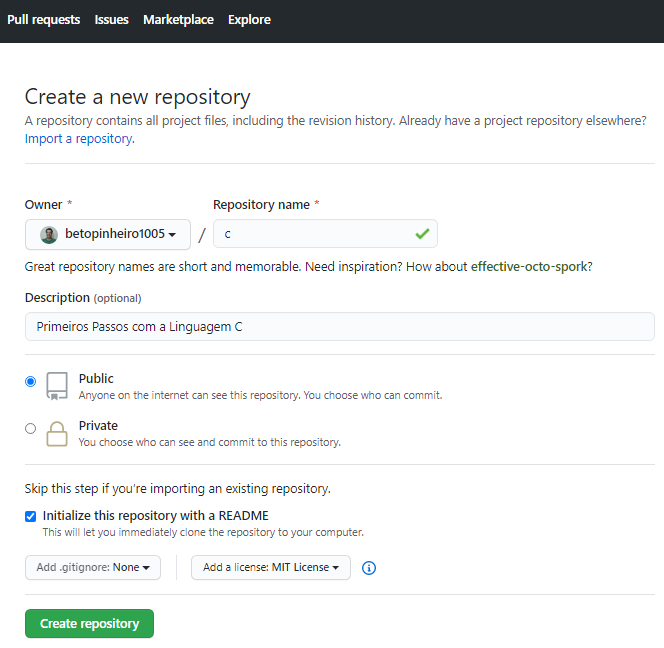


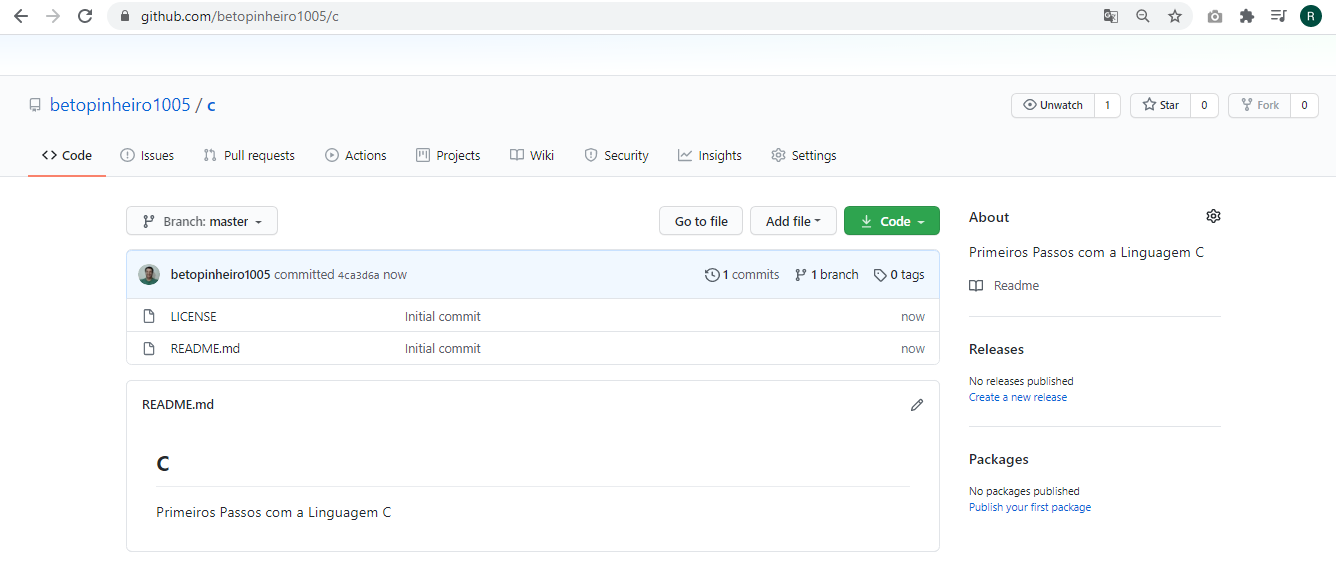
### GitHub Jobs





## Criando um novo repositório



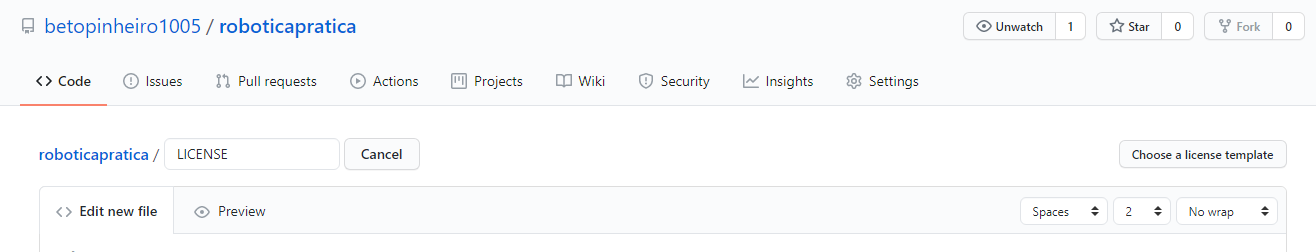


### Alterando o tipo de licença de um repositório

- Abra o seu repositório.

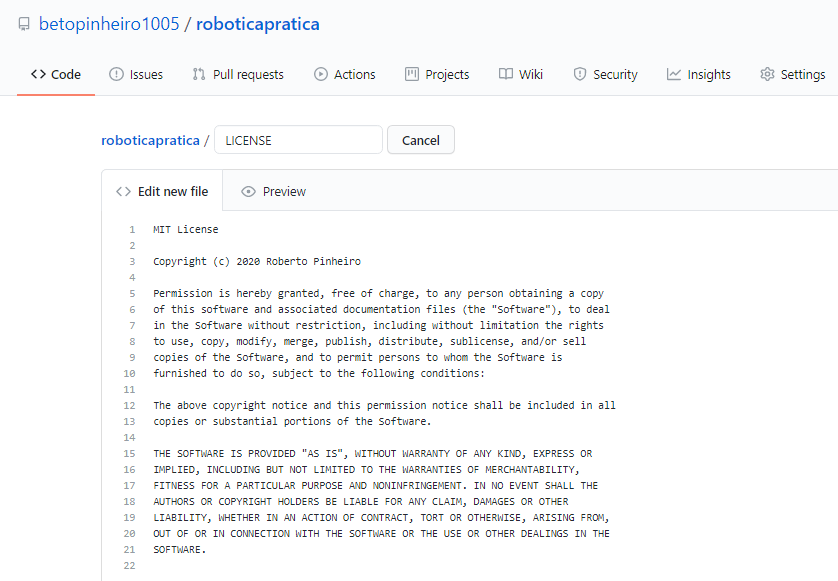
- Clique em “Add file” e em seguida em “Create new file”.

- Crie um arquivo com o nome de LICENSE:

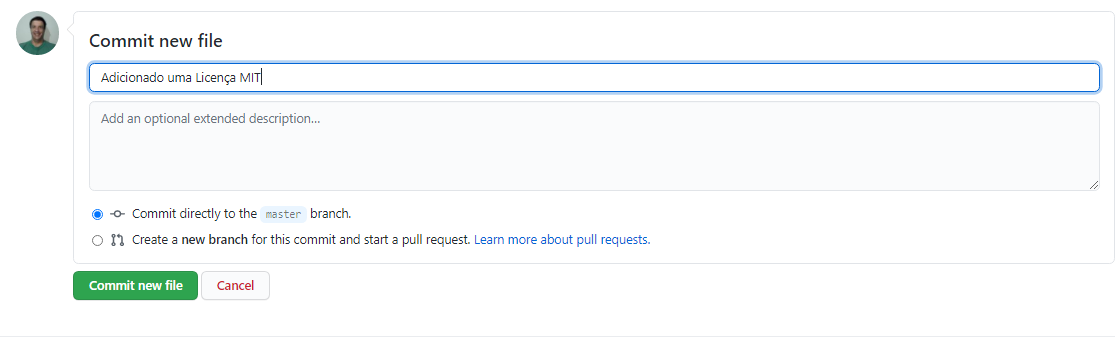


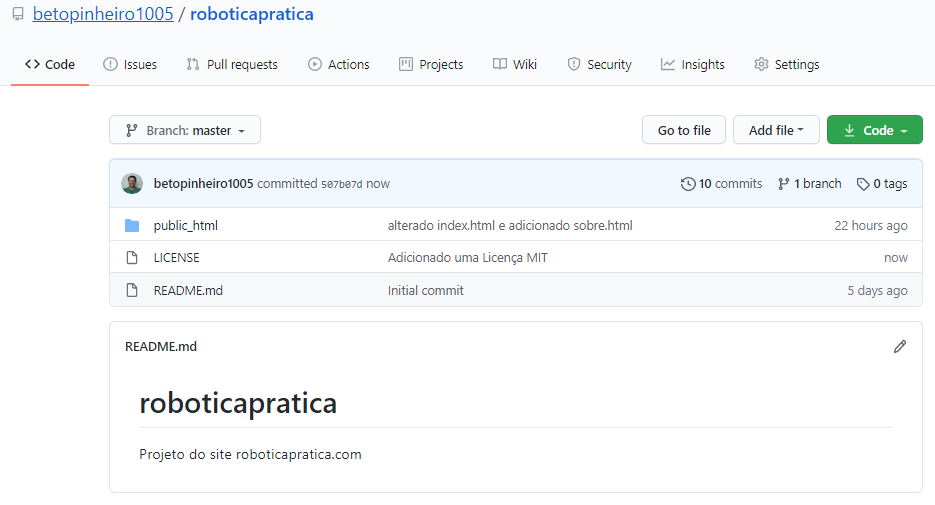
- Clique no botão “Choose a license template” e selecione a licença.

- Clique no botão “Review and Submit”.

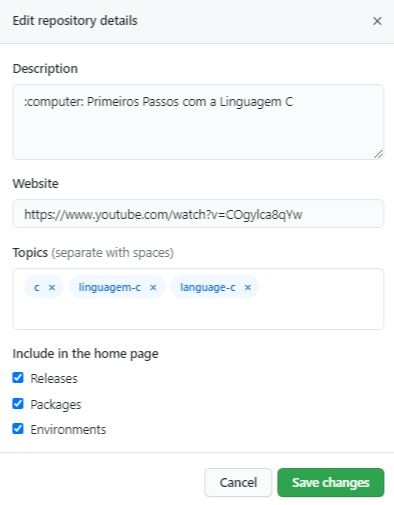


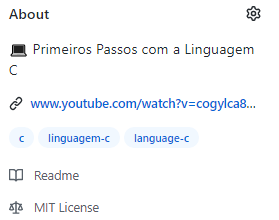
- Faça o commit. Escolha o ramo Master:





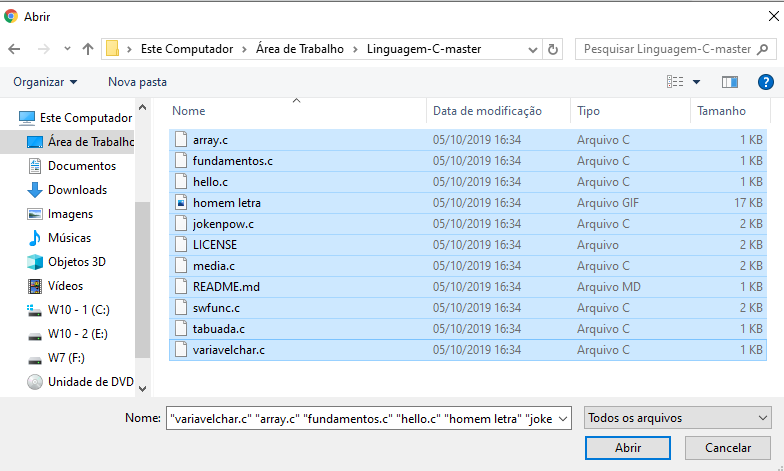
### Adicionando um emoji à descrição





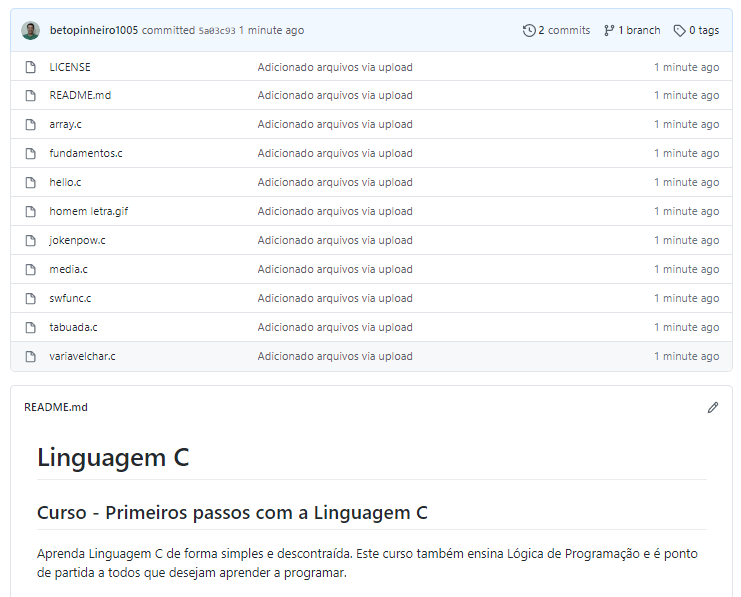
### Fazendo o upload de arquivos do projeto

- Clique em “Add files” e posteriormente em “Upload files”



- Em “Commit changes” entre com:

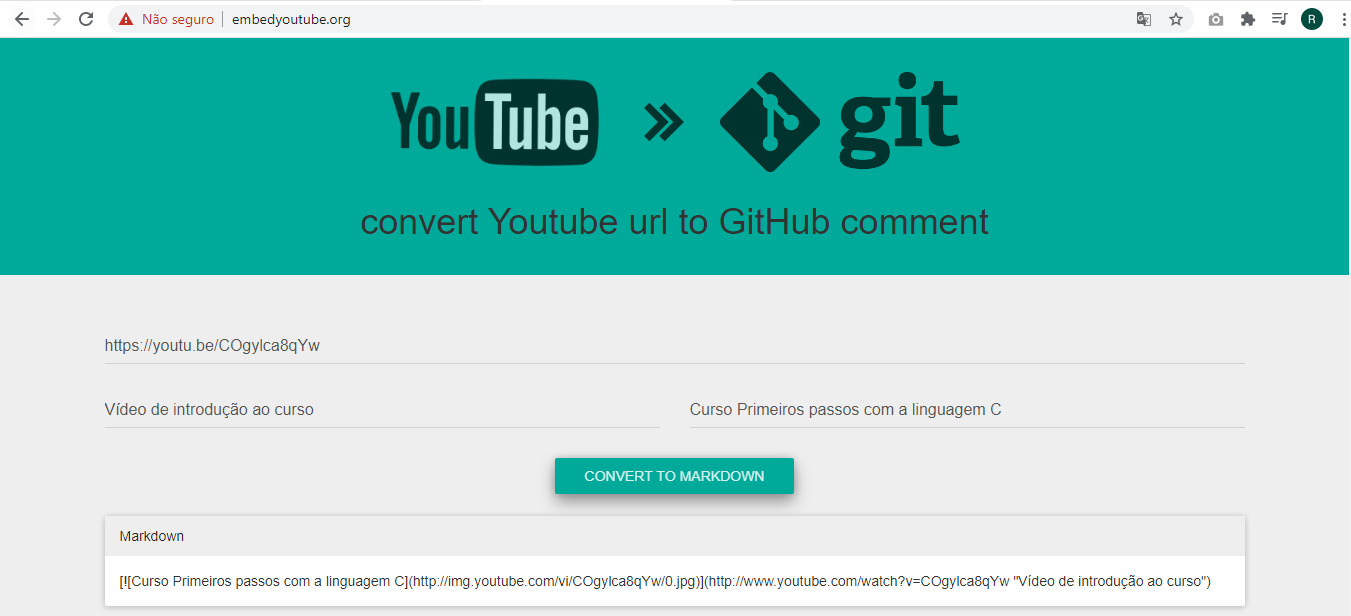
Adicionado arquivos via upload.



O próximo passo é personalizar o arquivo README.md

# Aula 12 - Personalizando o README.md

http://embedyoutube.org/





# Linguagem C

## Curso - Primeiros passos com a Linguagem C

Aprenda Linguagem C de forma simples e descontraída. Este curso também ensina Lógica de Programação e é ponto de partida a todos que desejam aprender a programar.

![Homem letra](https://github.com/professorjosedeassis/c/blob/master/homem%20letra.gif)

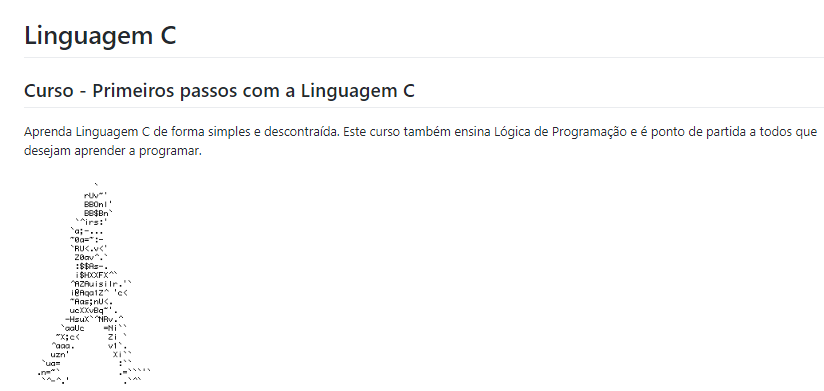
### Assistir ao vídeo de introdução

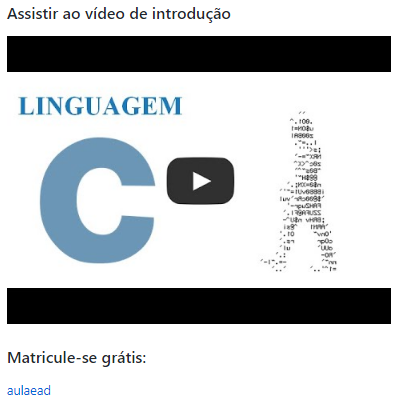
[![Curso primeiros passos com a Linguagem C](http://img.youtube.com/vi/COgylca8qYw/0.jpg)](http://www.youtube.com/watch?v=COgylca8qYw "Vídeo de introdução ao curso")

### Matricule-se grátis:

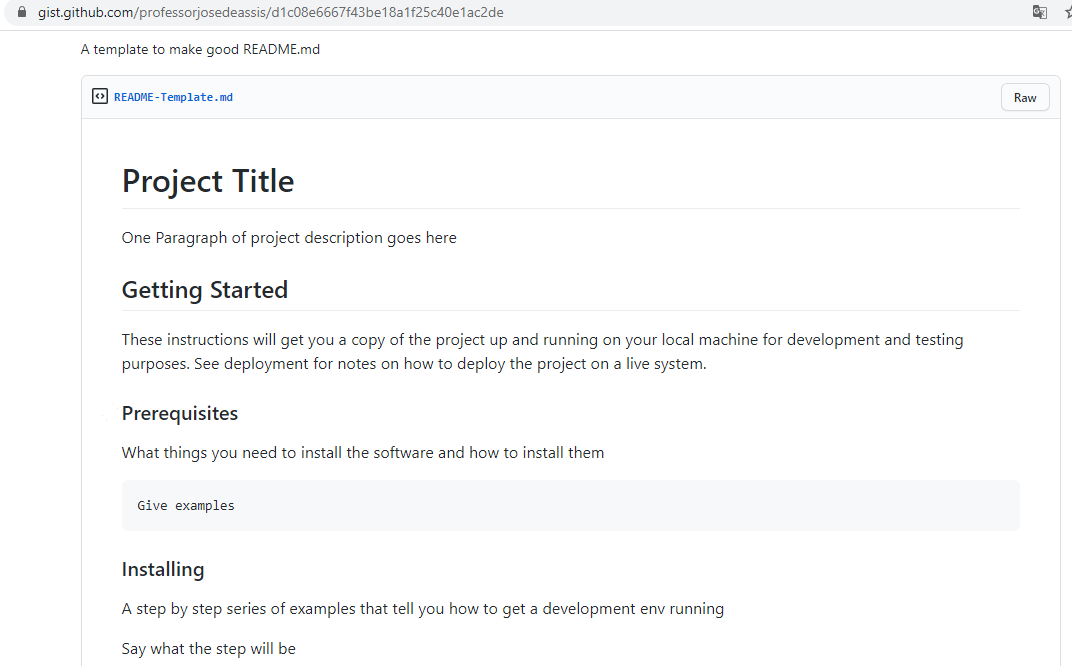
[aulaead](https://www.aulaead.com/courses/curso-gratis-linguagem-c)

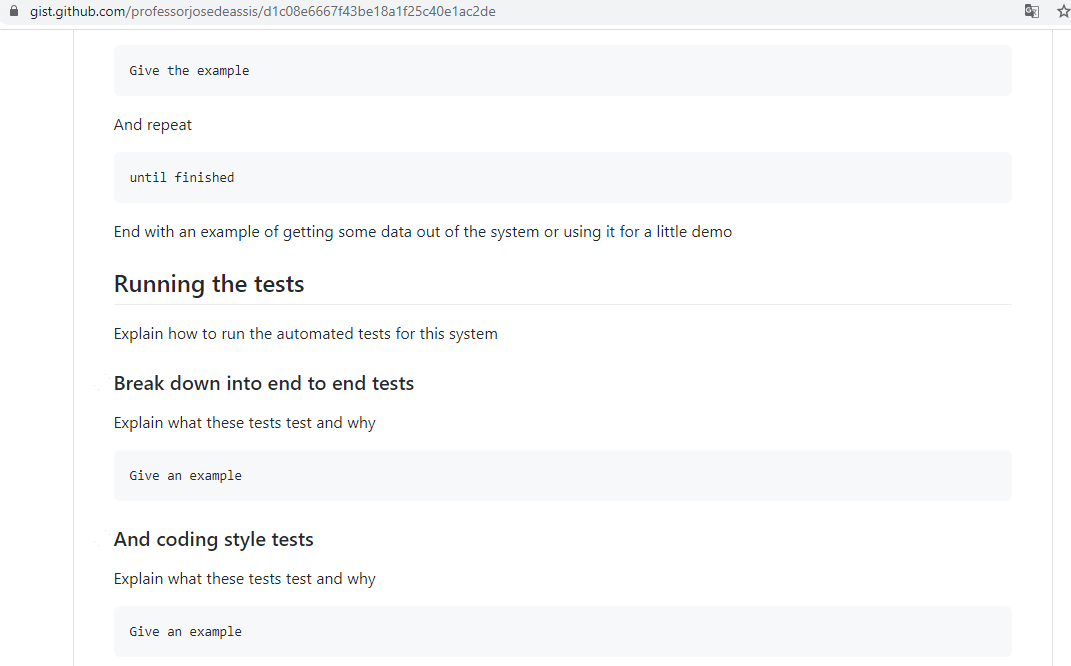
Posteriormente faça o commit (em commit changes).

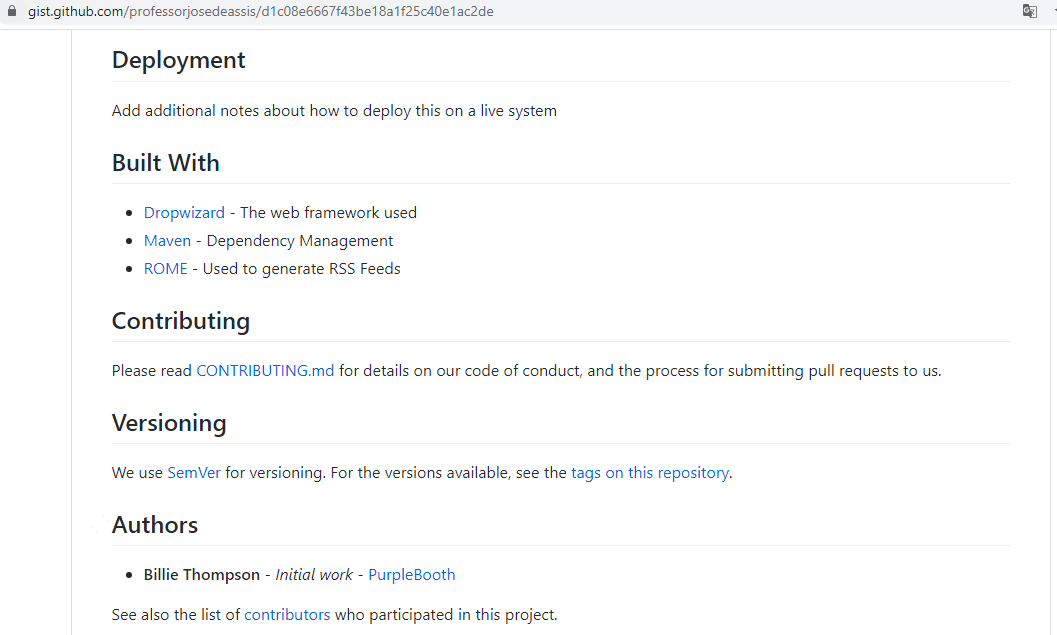


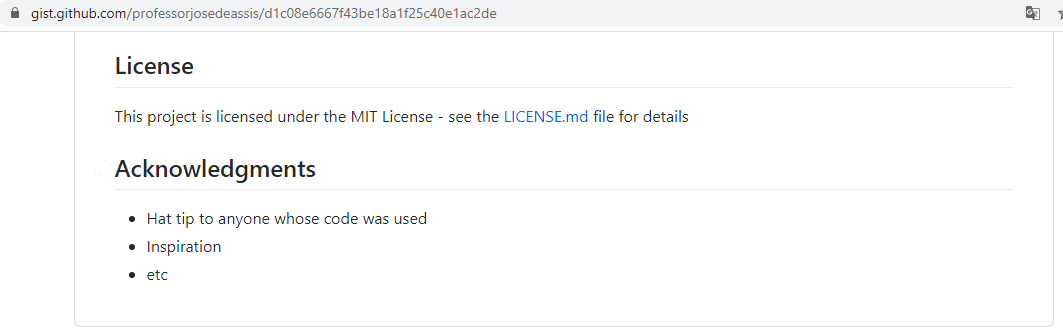


<https://gist.github.com/professorjosedeassis>









# Project Title

One Paragraph of project description goes here

## Getting Started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes. See deployment for notes on how to deploy the project on a live system.

### Prerequisites

What things you need to install the software and how to install them

```

Give examples

```

### Installing

A step by step series of examples that tell you how to get a development env running

Say what the step will be

```

Give the example

```

And repeat

```

until finished

```

End with an example of getting some data out of the system or using it for a little demo

## Running the tests

Explain how to run the automated tests for this system

### Break down into end to end tests

Explain what these tests test and why

```

Give an example

```

### And coding style tests

Explain what these tests test and why

```

Give an example

```

## Deployment

Add additional notes about how to deploy this on a live system

## Built With

\* [Dropwizard](http://www.dropwizard.io/1.0.2/docs/) - The web framework used

\* [Maven](https://maven.apache.org/) - Dependency Management

\* [ROME](https://rometools.github.io/rome/) - Used to generate RSS Feeds

## Contributing

Please read [CONTRIBUTING.md](https://gist.github.com/PurpleBooth/b24679402957c63ec426) for details on our code of conduct, and the process for submitting pull requests to us.

## Versioning

We use [SemVer](http://semver.org/) for versioning. For the versions available, see the [tags on this repository](https://github.com/your/project/tags).

## Authors

\* \*\*Billie Thompson\*\* - \*Initial work\* - [PurpleBooth](https://github.com/PurpleBooth)

See also the list of [contributors](https://github.com/your/project/contributors) who participated in this project.

## License

This project is licensed under the MIT License - see the [LICENSE.md](LICENSE.md) file for details

## Acknowledgments

\* Hat tip to anyone whose code was used

\* Inspiration

\* etc

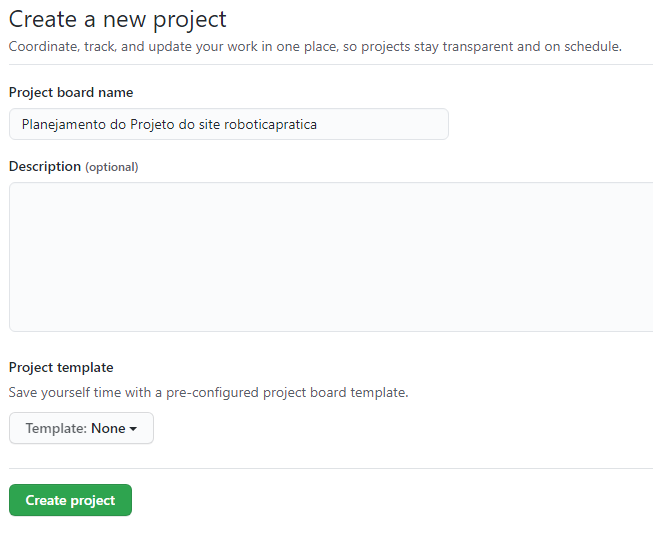
# Aula 13 - Gerenciando projetos com a ferramenta KANBAN nativa do GitHub

O Kanban é um sistema visual de gerenciamento de projetos.



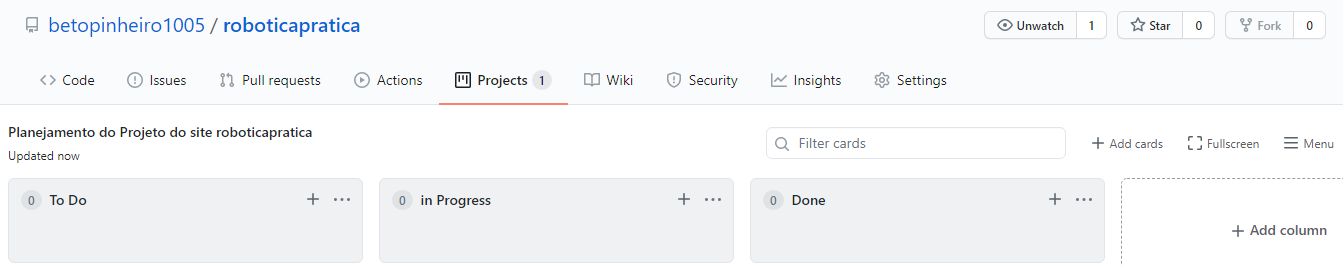
## Criando um projeto de Kanban

Clique no link “Projects” e posteriormente no botão “Create a new project”.

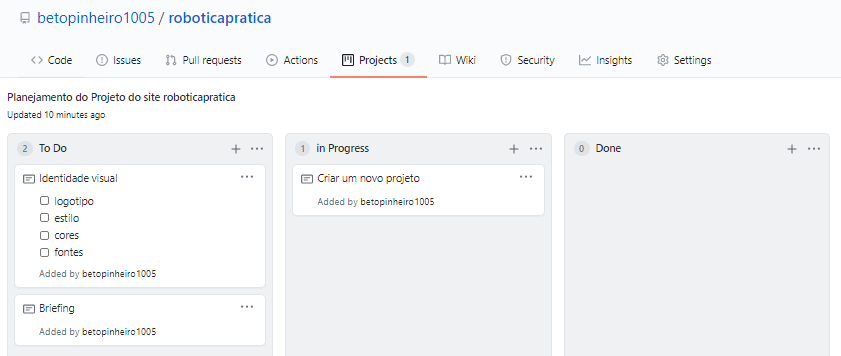


- Clique no botão “Create project”.

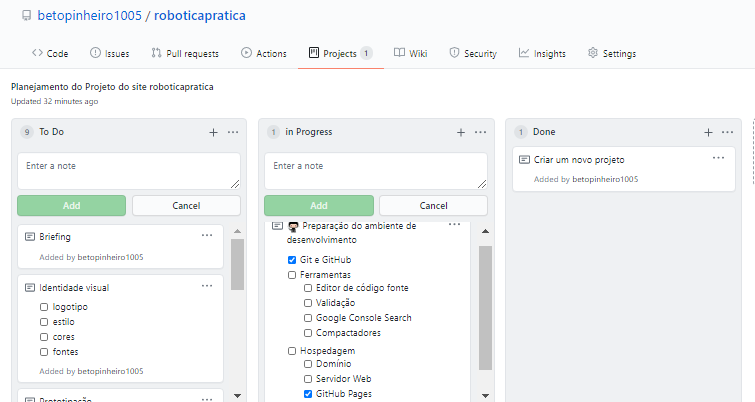
- Crie três colunas: To Do, in Progress e Done.



## Criando os cartões







# Aula 14 - Fork e Pull Request

O fork faz a clonagem de um repositório diretamente no seu GitHub. Ao fazer um fork você pode fazer qualquer tipo de alteração no projeto clonado sem interferir no projeto original.

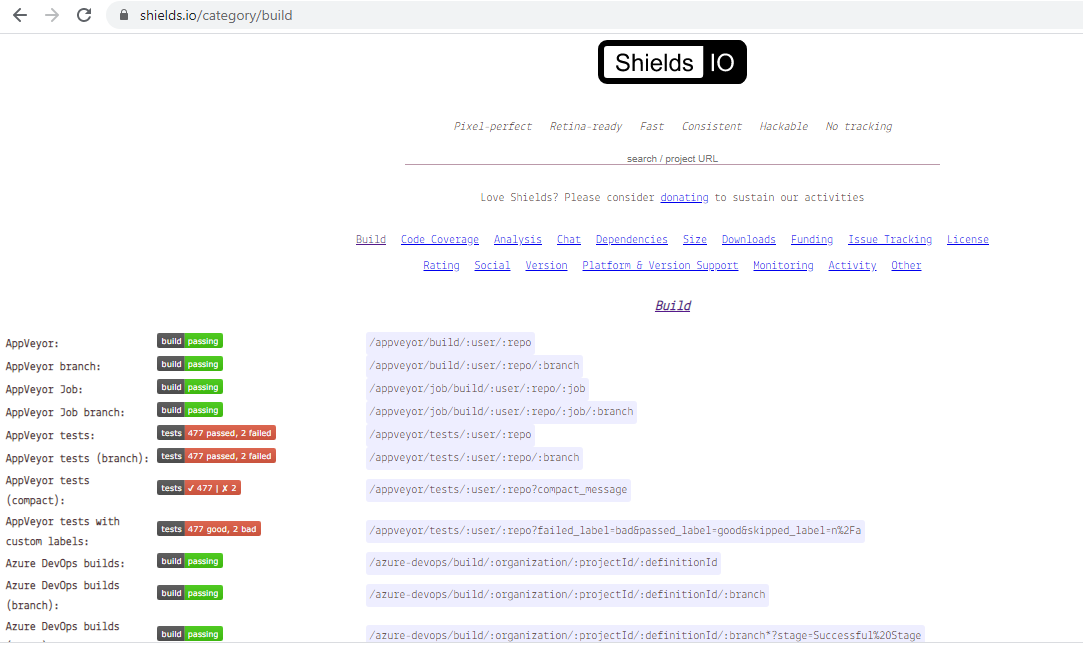
Caso queira sugerir alguma alteração no projeto original você pode fazer um “pull request” sinalizando as alterações ao dono do projeto. Quando o dono do projeto receber o pull request ele pode aceitar ou não as mudanças. O pull request é uma forma de você contribuir com projetos da comunidade.

# Aula 15 - Selos Dinâmicos de metadados usando shield.io

## Instalando selos de metadados no README.md

<https://shields.io/>

Existem muitos tipos de selos, subdivididos em categorias. Estes selos usam as informações de seus repositórios. Alguns destes selos também dependem de serviços externos para poderem funcionar.



## Categoria Size



### GitHub repo size

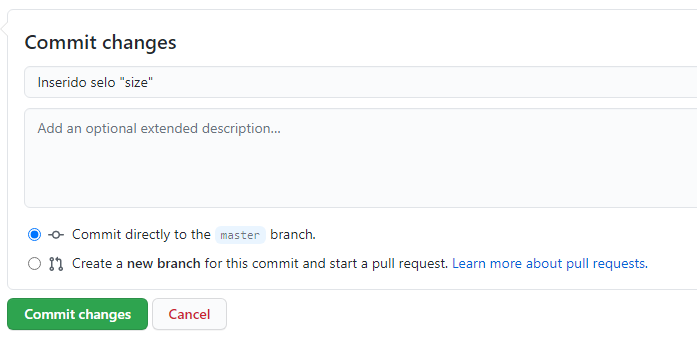


Selecione a opção “Copy Markdown”.





- Faça o commit.



## Categoria Analysis

### GitHub language count



Selecione a opção “Copy Markdown”.

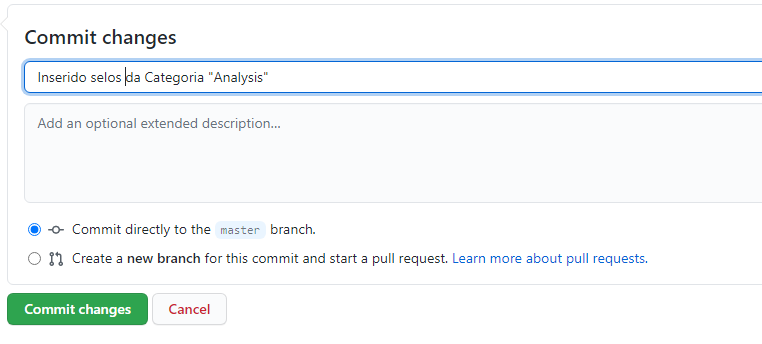
### GitHub top language



- Selecione a opção “Copy Markdown”.



- Faça o commit.





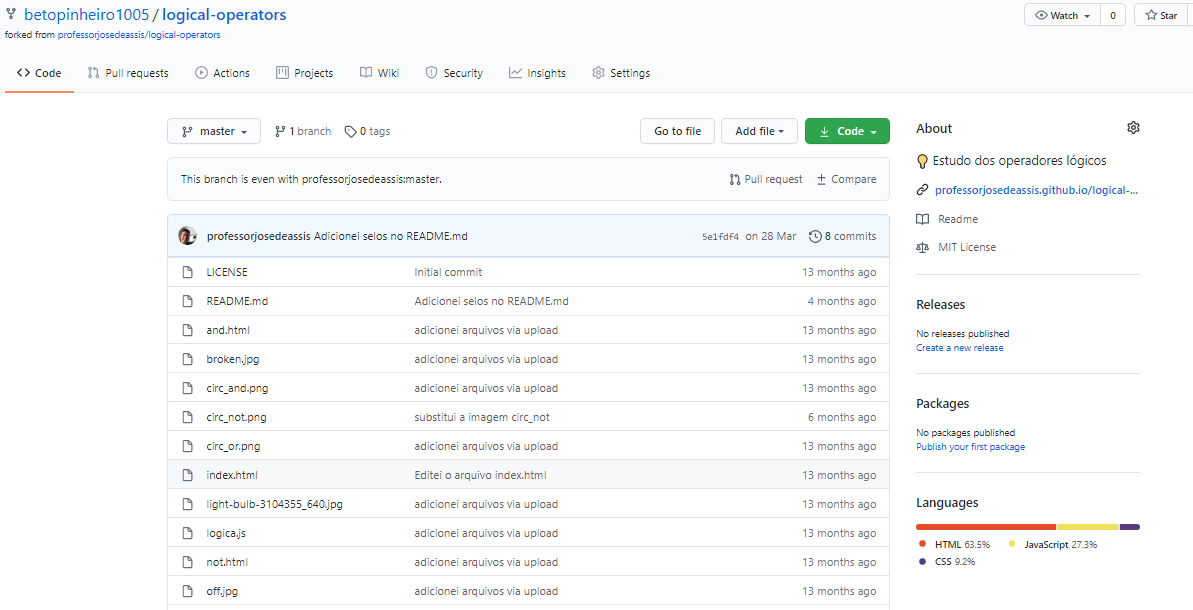
# Aula 16 - Hospedagem de site grátis no GITHUB

## GitHub Pages

O GitHub Pages permite a hospedagem de um site estático, ou seja: com arquivos HTML, CSS, Javascript sem suporte ao PHP, sem um banco de dados, sem a possibilidade de instalar, por exemplo, o WordPress.

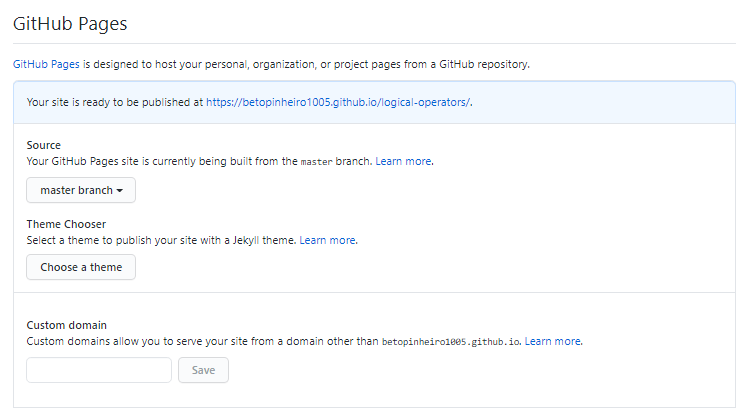
Limite de no máximo 1 Gb para seu site e um limite de largura de banda de 100G/mês. Não permite gerar nenhum tipo de comércio eletrônico.

- No github do Prof. José de Assis, faça o fork do projeto logical-operators.



- Clique em Settings:

- Em Source, selecione a opção “master branch”.



Endereço URL gerado:

<https://betopinheiro1005.github.io/logical-operators/>



<https://www.credential.net/credential-redirect/20499081#gs.9znsft>

<https://www.linkedin.com/pulse/certificado-curso-git-e-github-roberto-pinheiro>