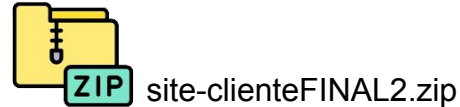


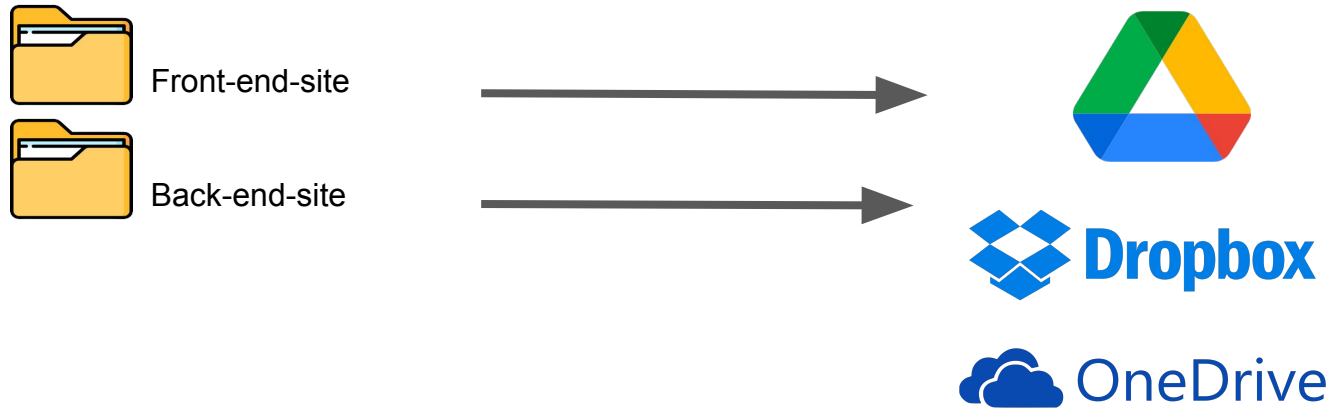


Software de controle de versão - Versionamento de código.





Software de controle de versão - Versionamento de código.





Software de controle de versão - Versionamento de código.

- Controle de histórico.
- Trabalhar em equipe no mesmo projeto
- Logs detalhados.
- Trabalhar de forma isolada em seu próprio ambiente.
- Eliminando sujeira em código e duplicidade.



Software de controle de versão,
LOCAL.

!=

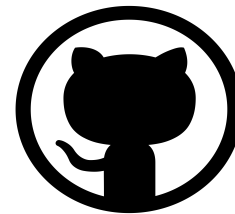


Plataforma, Repositório remoto,
rede social para devs.



git

+



GitHub

Seu repositório
local



push (envio)

Colega em outra
região (País)



push (envio)



← → ↺

github.com/uchoamaster

🔍


Sign in

Sign up

🐱

Product ▾ Solutions ▾ Open Source ▾ Pricing

Overview Repositories 178 Projects Packages Stars 300



uchoamaster

Follow

Tech Leader, Professor, Programador, Nerd e Autodidata.

185 followers · 603 following

Freeline Informática

Joinville - SC

Achievements

Beta

Send feedback

Block or Report

uchoamaster / README.md

Hi there 🙋 I'm Carlos Júnior Uchôa Oliveira 🇧🇷

A Fullstack developer from Joinville, Brazil.

LINKEDIN

INSTAGRAM

@uchoamaster

359 Articles read

JavaScript50

React30

CSS23

Tools23

Favorite publications

daily.dev

uchoamaster's GitHub Stats



O que é um Repositório?

É onde os arquivos do seu projeto ficam armazenados.

Quando iniciamos um projeto com **git**, estamos na verdade criando um repositório para aquele projeto.

Você pode enviar o repositório para alguns servidores específicos para gerenciar repositórios como **github**, **bitbucket** ou **gitlab**.



Iniciando um Repositório

Usamos o comando : **git init**

Com isso você está iniciando um repositório dessa pasta que digitou **git init**.

Todos arquivos dentro dessa pasta agora serão reconhecidos pelo git como um projeto.



GitHub

Serviço para **hospedar** / **gerenciar** repositórios.

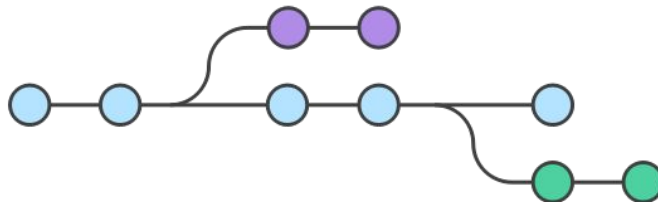
Podemos subir nossos repositórios no github e disponibilizá-lo para outros devs gerenciarem versões de nossos projetos e ou trabalhar em equipe.

Linha do tempo de repositórios adicionados, gráfico de codificação realizada e etc.

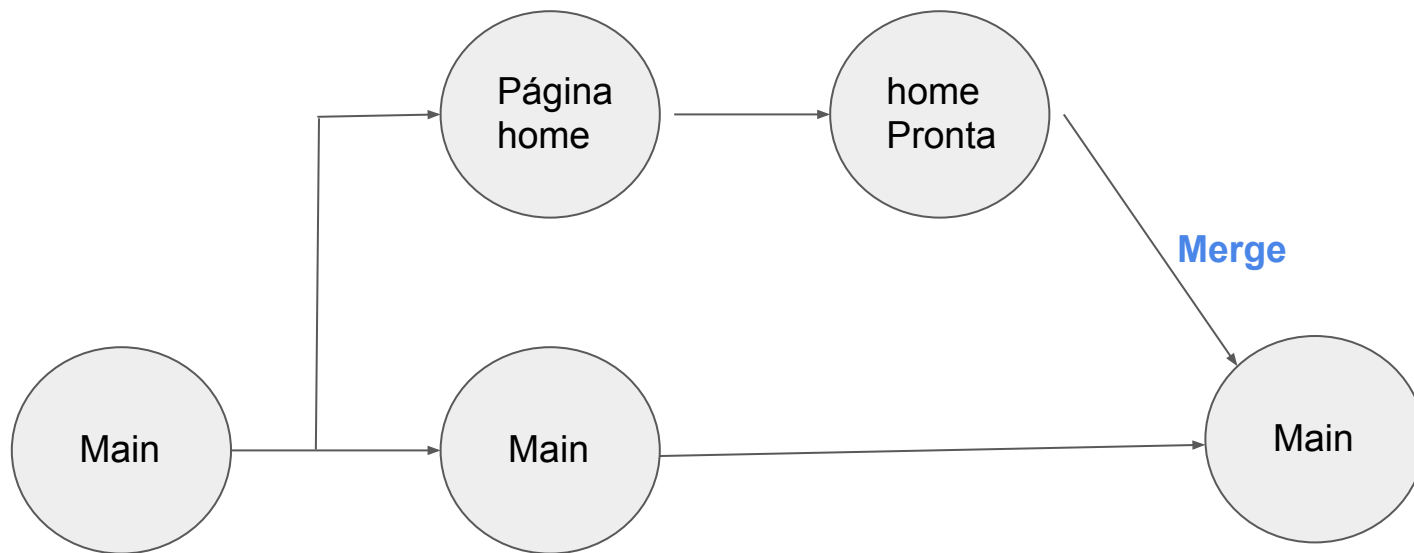
Branch

É uma maneira que o git disponibiliza para **separar versões do projeto**, **gerenciar melhor o projeto**.

Quando criamos um novo projeto ele inicia sempre na Branch **master** pelo **Git** (linha de comando) e pelo repositório no **Github** a Branch inicial sempre se chamará **main**



Branch



Criando uma nova Branch

Comando para visualizar Branches no meu projeto: **git branch**

Comando para criar uma nova Branch: **git branch page_home**

Comando para visualizar Branches no meu projeto: **git branch**

Mudando de Branch

Comando para visualizar Branches no meu projeto: **git branch**

Comando para mudar de Branch atual para outra criada: **git checkout page_home**

Comando para visualizar Branches no meu projeto: **git branch**

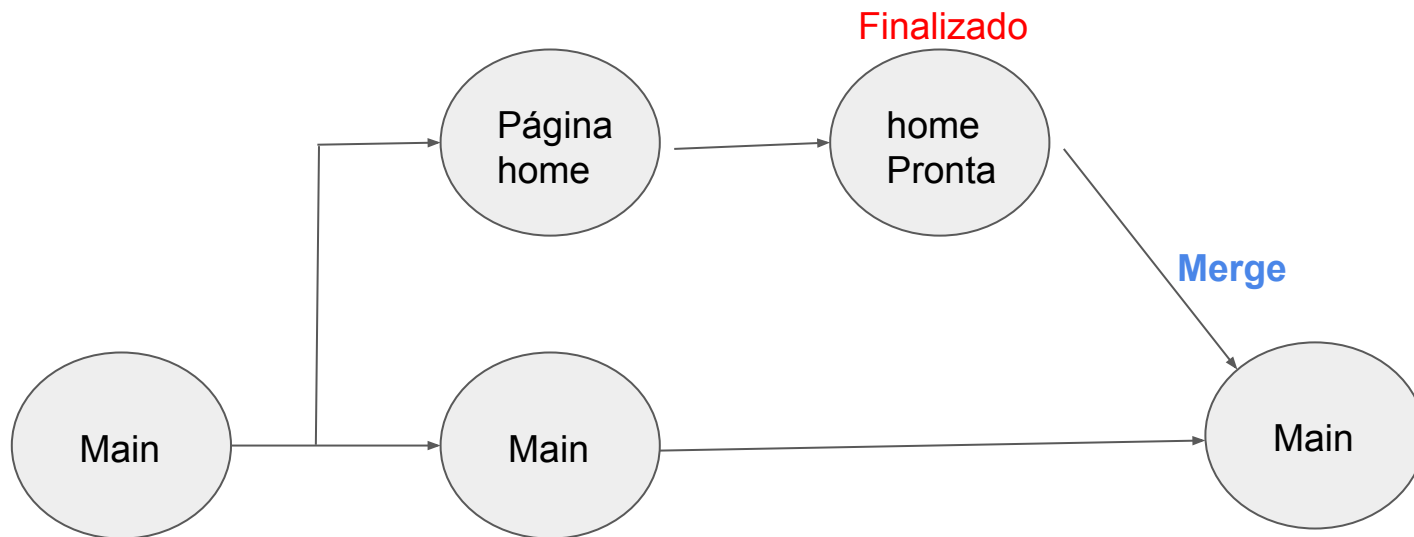
Comando para criar e já trocar para a Branch nova: **git checkout -b "page_teste"**

Deletando uma Branch

Comando para deletar uma Branch: **git branch -d page_teste**

Unindo Branches

Comando para unir Branches: **git merge nome_da_branch**



Unindo Branches

Comando para unir Branches: **git merge nome_da_branch**

Comando para trocar de Branch: **git checkout page_home**

Comando para trocar de Branch: **git checkout page_contato**

Após:

git add .

git commit -m "msg"

git push

volto para a Branch main (principal)

git merge page_home

git push enviando pro Github