

APRENDA A USAR O GIT HUB

Autor: *João Henrique Domingos*

I.A utilizada: *chatgpt*



O QUE É O GIT HUB:

O GitHub é uma plataforma de hospedagem de código baseada em Git. Ele permite que desenvolvedores armazenem, gerenciem e colaborem em projetos de software. Com ele, é possível versionar código, trabalhar em equipe, revisar alterações e automatizar processos. Além disso, o GitHub facilita o desenvolvimento open source e a integração com diversas ferramentas de CI/CD.



CRIANDO UMA CONTA NO GIT HUB:

Acesse [GitHub](https://github.com) e cadastre-se. Após isso, confirme seu e-mail e configure seu perfil.



CRIANDO UM REPOSITORIO:

No GitHub:

1. Clique no botão "New repository"
2. Dê um nome ao repositório
3. Escolha entre público ou privado
4. Clique em "Create repository"

Top repositories




Find a repository...

Create a new repository

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

Required fields are marked with an asterisk ().*

Owner *

 joaohenriquedo ▾

Repository name *

/

Great repository names are short and memorable. Need inspiration? How about **fictional-goggles** ?



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:



Add a README file

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

Add .gitignore

.gitignore template: None ▾

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

Choose a license

License: None ▾

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

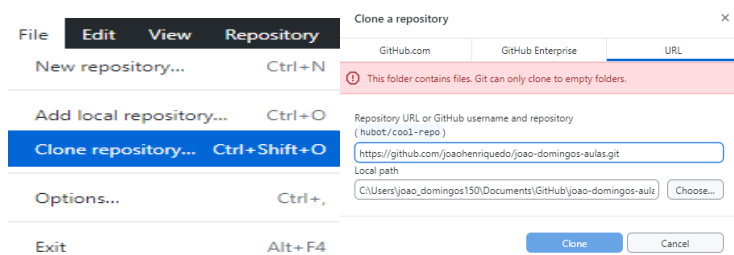
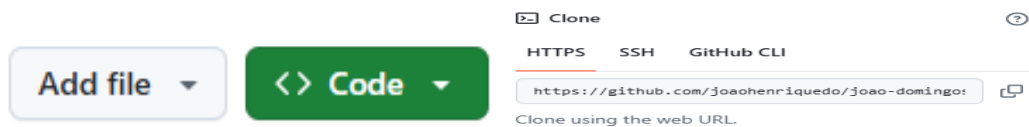


You are creating a public repository in your personal account.

Create repository






CLONANDO UM REPOSITÓRIO:





1. Acesse o repositório no GitHub.
2. Clique no botão verde "Code".
3. Copie a URL fornecida (HTTPS ou SSH).
4. Abra o GitHub Desktop.
5. Clique em "File" > "Clone Repository" ou pressione `Ctrl + Shift + O` (`Cmd + Shift + O` no Mac).
6. Escolha a pasta onde deseja salvar o projeto.
7. Clique em "Clone" e aguarde o download dos arquivos.
- 8.



EDITANDO UM ARQUIVO DIRETAMENTE NO GIT HUB:

- 1 Acesse o repositório no [GitHub](#).
- 2 Clique no arquivo que deseja editar.
- 3 Clique no ícone "Editar" (lápiz) no canto superior direito.
- 4 Faça as alterações desejadas.
- 5 Role para cima e clique em "Commit changes" para salvar

-  joaohenriquedo/joao-domingos-aulas
-  joaohenriquedo/projeto-senai-aplicativos
-  joaohenriquedo/atividade001
-  RuanAlves07/SenaiPython
-  joaohenriquedo/joao-domingos-aules

 .gitattributes	Initial commit	yesterday
 exe01.py	Update exe01.py	yesterday
 exe02.py	diploid inicial	yesterday
 exe03.py	diploid inicial	yesterday



Cancel changes

Commit changes...

Spaces

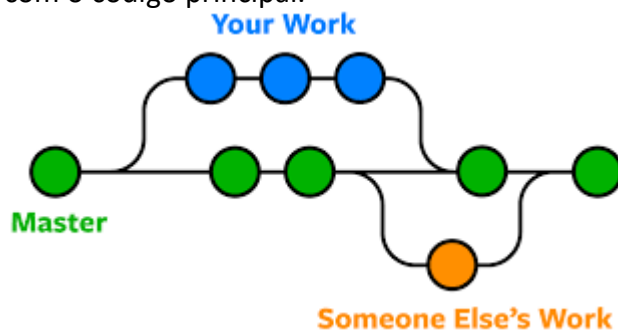
2

No wrap

O QUE É BRANCH:

Um branch (ou "ramo") no Git é como uma linha paralela de desenvolvimento dentro de um repositório. Ele permite que você trabalhe em novas funcionalidades sem afetar o código principal.

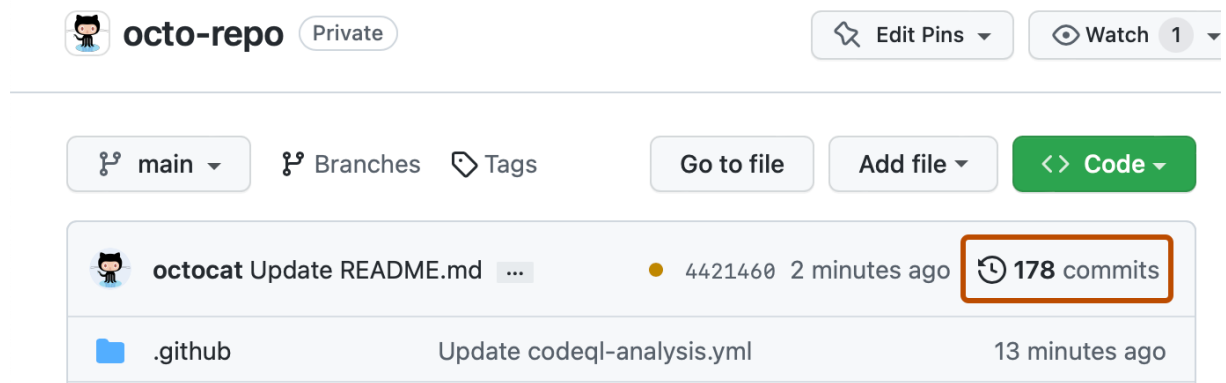
Exemplo: Imagine que você tem um projeto funcionando e quer adicionar uma nova funcionalidade. Em vez de modificar diretamente o código principal (geralmente chamado de main ou master), você cria um branch separado, faz suas alterações lá e só depois junta com o código principal.



O QUE É UM COMMIT:

Um **commit** no Git é como um "salvamento" no histórico do seu projeto. Ele registra as mudanças feitas nos arquivos e cria um ponto de restauração, permitindo que você volte para esse estado no futuro, se necessário.

Pense no commit como uma foto do seu código naquele momento!



The screenshot shows the GitHub interface for a repository named 'octo-repo'. At the top, there's a header with the repository name, a 'Private' badge, and buttons for 'Edit Pins' and 'Watch' (with a count of 1). Below this is a navigation bar with 'main' (selected), 'Branches', and 'Tags'. To the right are buttons for 'Go to file', 'Add file', and 'Code'. The main content area displays a list of commits. The first commit is by 'octocat' titled 'Update README.md', with a commit hash '4421460' and the text '2 minutes ago'. A box highlights the '178 commits' link. Below this, a file change is shown for '.github' with the title 'Update codeql-analysis.yml' and the text '13 minutes ago'.

Commit Hash	Author	Message	Time	Commits
4421460	octocat	Update README.md	2 minutes ago	178 commits
		Update codeql-analysis.yml	13 minutes ago	

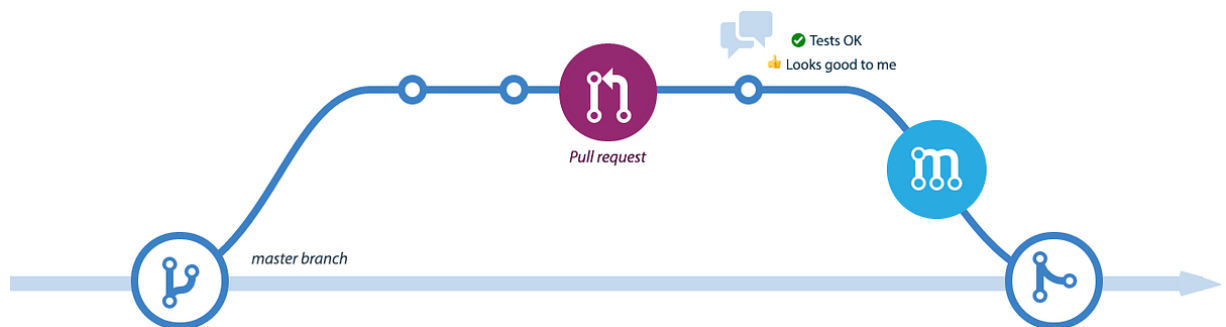
O QUE É UM PULL REQUEST:

Um Pull Request (PR) é um pedido para mesclar (juntar) as mudanças de um branch para outro no GitHub.

Geralmente, é usado para sugerir e revisar mudanças antes de adicioná-las ao código principal (main ou master).

COMO FUNCIONA UM PULL REQUEST:

- 1 ☐ Você cria um branch e faz alterações no código.
- 2 ☐ Faz commits e envia (push) para o GitHub.
- 3 ☐ Abre um Pull Request para sugerir a mudança.
- 4 ☐ Outros desenvolvedores revisam o código e podem aprovar, pedir ajustes ou recusar.
- 5 ☐ Se aprovado, o PR é mesclado (merged) no branch principal.



O QUE É UM MAIN:

O main (ou master em versões mais antigas) é o branch principal de um repositório Git. Ele geralmente contém a versão mais estável e pronta para produção do projeto.

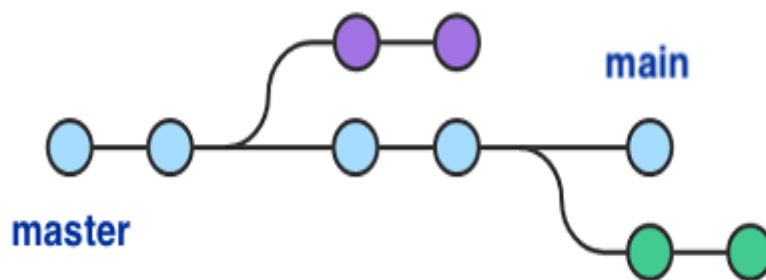
CARACTERÍSTICAS DO MAIN:

É o ponto central do desenvolvimento.

1. Normalmente, todas as alterações passam por revisão antes de serem adicionadas a ele.

Branches secundários (como feature-x ou bugfix-y) são criados para testar novas funcionalidades sem afetar o main.

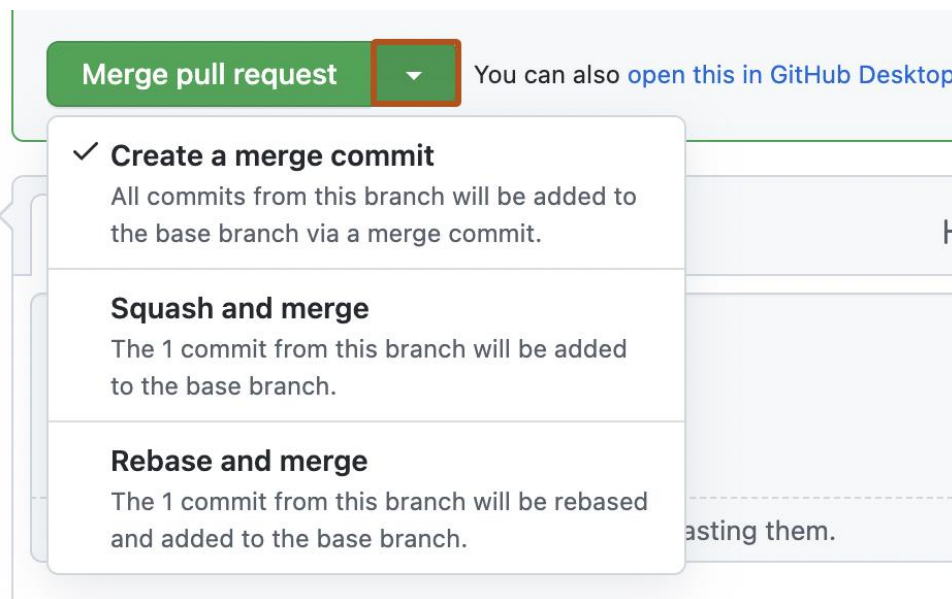
Após testes e revisões, mudanças de outros branches são mescladas (merged) no main por meio de um Pull Request.



O QUE É UM MERGE PULL REQUEST:

No contexto do desenvolvimento de software, principalmente com ferramentas como GitHub, GitLab ou Bitbucket, pull request (PR) é uma maneira de sugerir mudanças no código. Quando um desenvolvedor termina uma parte de seu trabalho em uma branch separada, ele cria um "pull request" para pedir que essas mudanças sejam revisadas e mescladas (merge) na branch principal (geralmente chamada de "main" ou "master").

Merge Pull Request significa combinar (ou "mesclar") as mudanças de uma branch (onde o trabalho foi feito) na branch principal ou em outra branch, depois que a revisão foi feita.



CONCLUSÃO:

O GitHub e o Git Desktop são ferramentas essenciais no desenvolvimento de software moderno, proporcionando uma forma eficiente de versionamento e colaboração em projetos. O GitHub se destaca como uma plataforma de hospedagem de código-fonte que oferece uma interface intuitiva e recursos avançados para facilitar o trabalho em equipe, como pull requests, issues, e integração com outros serviços. Já o Git Desktop simplifica o uso do Git, permitindo que até mesmo iniciantes possam gerenciar repositórios de forma visual e sem a necessidade de trabalhar diretamente com a linha de comando.

Ao entender e dominar essas ferramentas, os desenvolvedores conseguem otimizar o fluxo de trabalho, mantendo um controle rigoroso sobre as versões do código e colaborando de forma mais eficaz. Além disso, ao utilizar o GitHub, é possível compartilhar projetos, contribuir para projetos open-source, e até mesmo criar portfólios profissionais, o que amplia as oportunidades no mercado de trabalho.

Em resumo, o GitHub e o Git Desktop, quando utilizados corretamente, oferecem uma poderosa combinação que facilita a colaboração, o versionamento e a gestão de projetos de software, tornando-se indispensáveis para qualquer desenvolvedor que queira se destacar na área de tecnologia.

