

INSTITUTO MAUÁ DE TECNOLOGIA



# Linguagens I

GIT

Prof. Tiago Sanches da Silva  
Prof. Murilo Zanini de Carvalho

# O que é GIT?

# O que é GIT?

Git é um sistema de controle de versão distribuído e um sistema de gerenciamento de código fonte. O Git foi inicialmente projetado e desenvolvido por Linus Torvalds para o desenvolvimento do kernel Linux (2005), mas foi adotado por muitos outros projetos.

# O que é Github?

O GitHub estende a ferramenta Git com uma ampla gama de recursos colaborativos e oferece hospedagem gratuita para projetos open source.



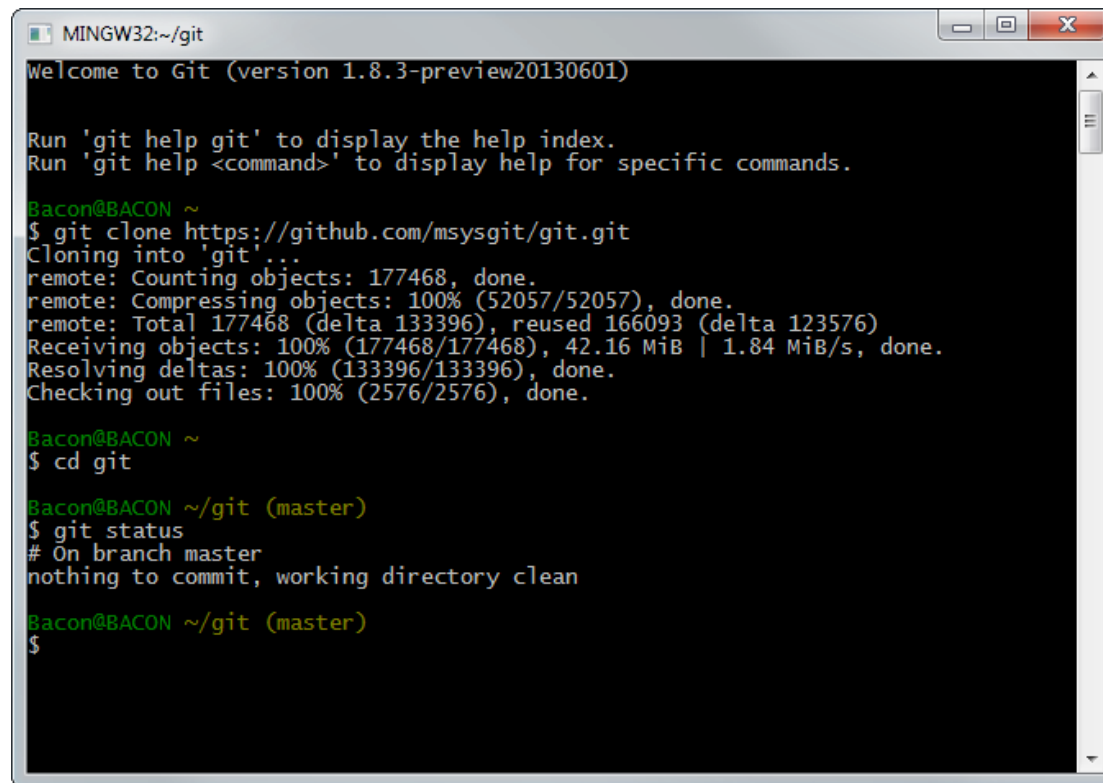
# Existem outros?

SIM



# Download

Faça o download da ferramenta: <https://gitforwindows.org/>



```
MINGW32:~/git
Welcome to Git (version 1.8.3-preview20130601)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Bacon@BACON ~
$ git clone https://github.com/msysgit/git.git
Cloning into 'git'...
remote: Counting objects: 177468, done.
remote: Compressing objects: 100% (52057/52057), done.
remote: Total 177468 (delta 133396), reused 166093 (delta 123576)
Receiving objects: 100% (177468/177468), 42.16 MiB | 1.84 MiB/s, done.
Resolving deltas: 100% (133396/133396), done.
Checking out files: 100% (2576/2576), done.

Bacon@BACON ~
$ cd git

Bacon@BACON ~/git (master)
$ git status
# On branch master
nothing to commit, working directory clean

Bacon@BACON ~/git (master)
$
```

# Git local

## Primeira configuração

# Primeira configuração

Você deverá configurar o nome e e-mail que assinarão seus *commits*.

```
$ git config --global user.name "John Doe"  
$ git config --global user.email johndoe@example.com
```

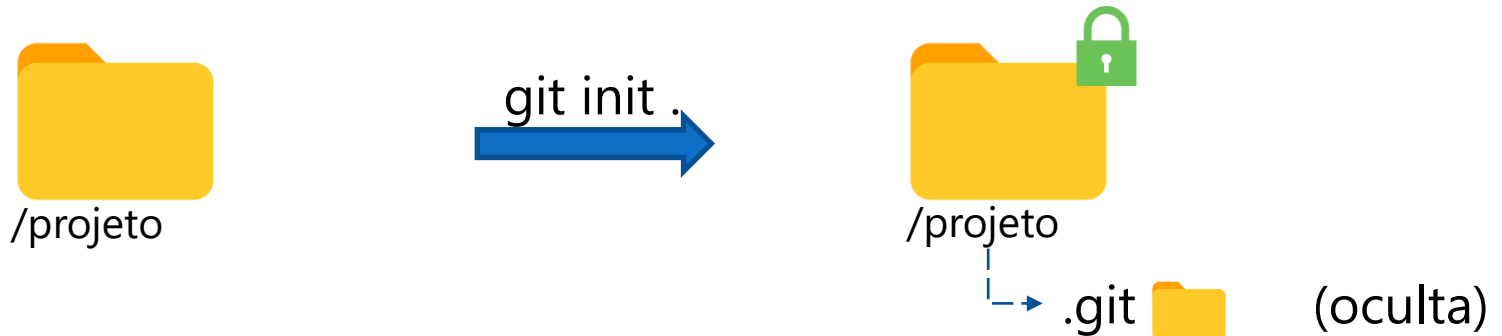
Esta configuração deve ser feita apenas uma vez em cada máquina de trabalho.



# Repositório

# Repositório

Cada diretório de **trabalho do Git** é um repositório com um histórico completo e habilidade total de acompanhamento das revisões, não dependente de acesso a uma rede ou a um servidor central.



# Cenário 1

# Cenário 1

- Criar o repositório local usando: `git init` .



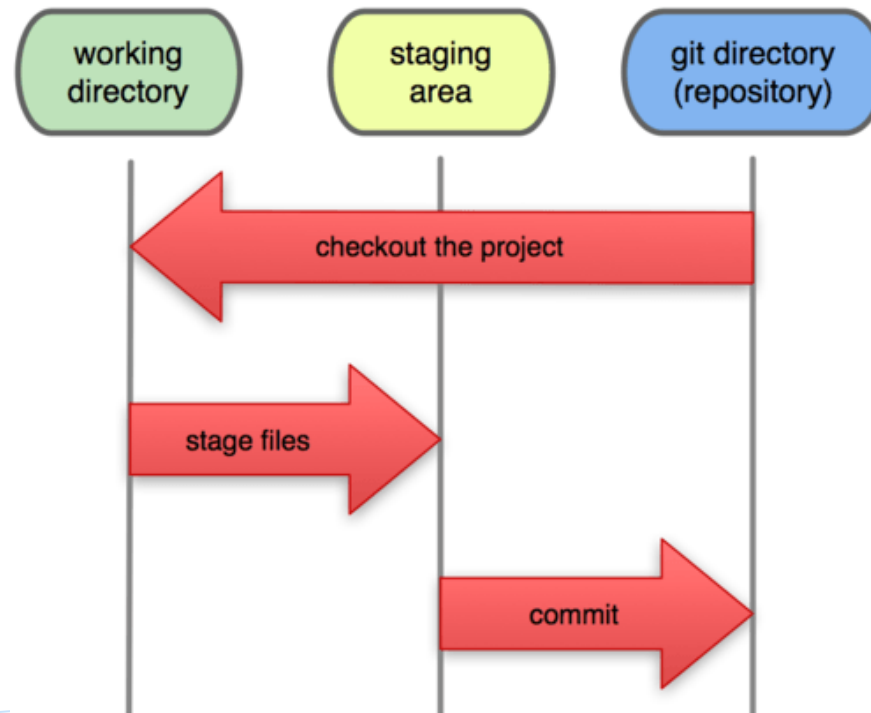
- Trabalhe
- Adicione o(s) arquivo para entrar no seu próximo **commit**
- Realize um **commit**

# Cenário 1

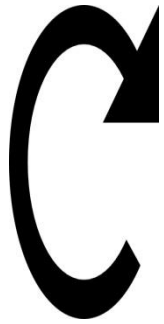


- Trabalhe
- Adicione o(s) arquivo para entrar no seu próximo *commit*
- Realize um *commit*

## Local Operations



# Cenário 1



- Trabalhe
- Adicione o(s) arquivo para entrar no seu próximo commit
- Realize um commit

```
git add <arquivo>
```

```
git add *
```

```
git commit -m "comentários das alterações"
```

```
git status
```

```
git status
```

```
git status
```

```
git add -A .
```

# Cenário 1

## git status

```
TiagoLow@TiagoLow-PC MINGW64 /e/Mauá/Disciplinas (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   ecm251-Linguagens/aulas/aula1/GIT.pptx

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

        ecm251-Linguagens/aulas/aula1/~$GIT.pptx

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

```
TiagoLow@TiagoLow-PC MINGW64 /e/Mauá/Disciplinas (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   ecm251-Linguagens/aulas/aula1/GIT.pptx

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

        ecm251-Linguagens/aulas/aula1/~$GIT.pptx
```

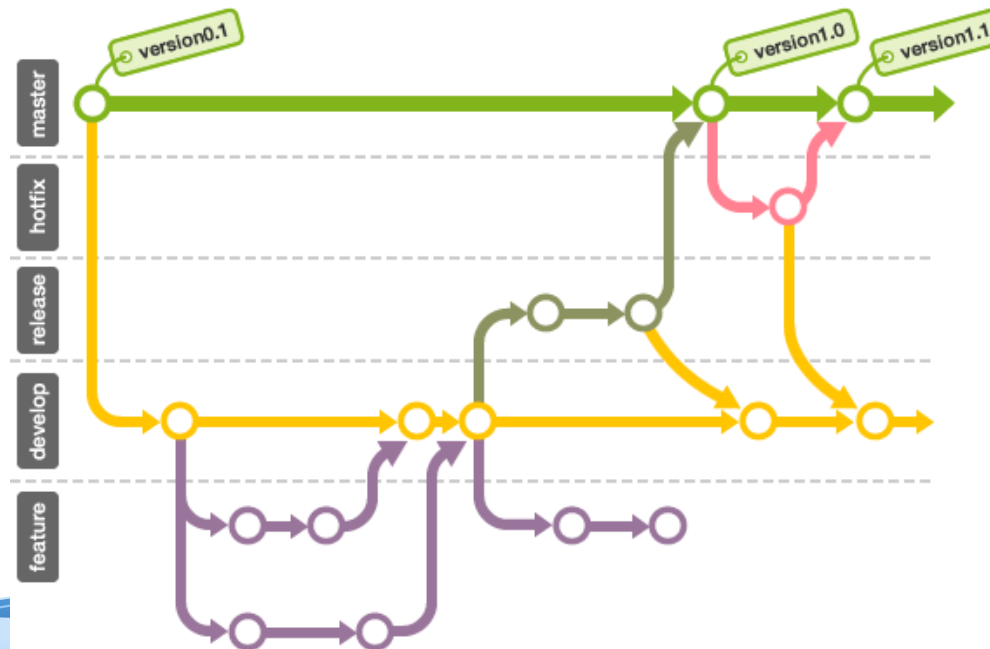
# Ramificação



# branches

Branches ("ramos") são utilizados para desenvolver funcionalidades isoladas umas das outras.

O **branch master** é o **branch** "padrão" quando você cria um repositório. Use outros **branches** para desenvolver e mescle-os (merge) ao **branch master** após a conclusão.



# branches

crie um novo branch chamado "funcionalidade\_x" e selecione-o usando

```
git checkout -b funcionalidade_x
```

retorne para o master usando

```
git checkout master
```

e remova o branch da seguinte forma

```
git branch -d funcionalidade_x
```

um branch *não está disponível a outros* a menos que você envie o

branch para seu repositório remoto

```
git push origin <funcionalidade_x>
```

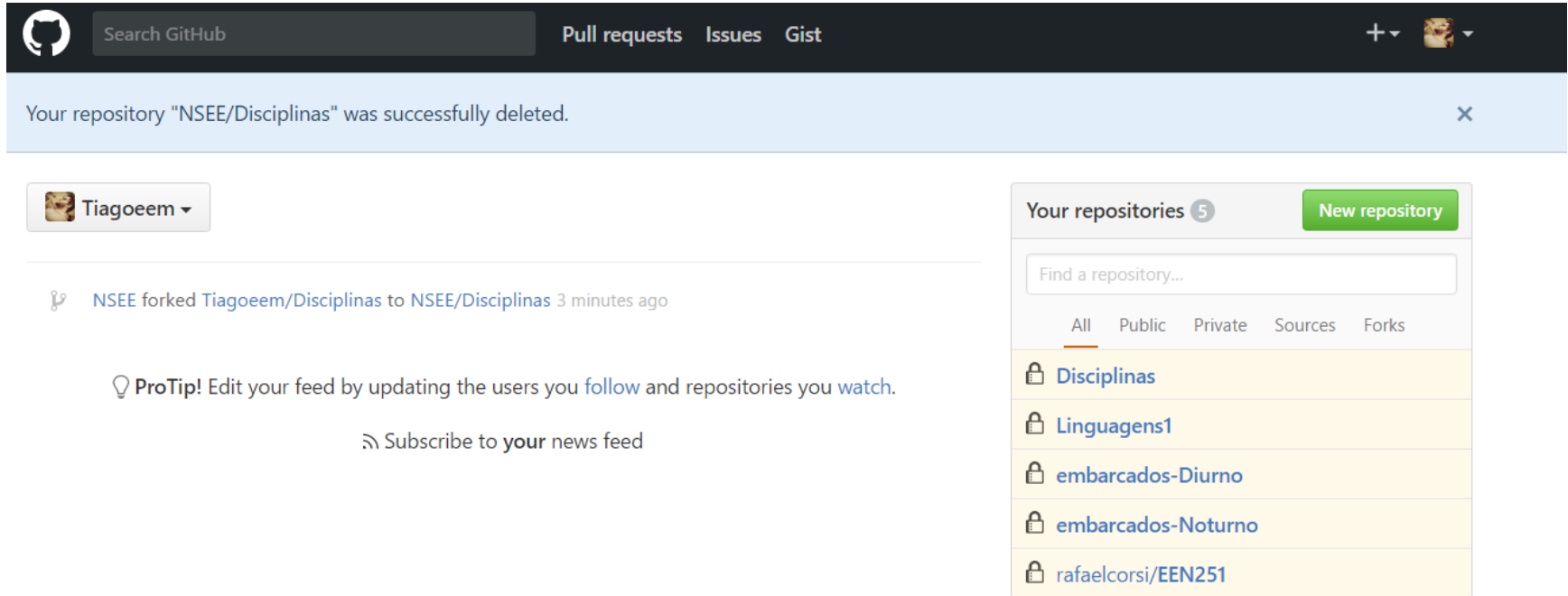
# Mergear branches distintos

Para mergear outro *branch* no seu atual ativo:

```
git merge <branch>
```

E se der conflito?

# Podemos explorar o Github?



The screenshot shows the GitHub web interface. At the top, there's a dark header with the GitHub logo, a search bar, and links for Pull requests, Issues, and Gist. A notification banner states: "Your repository 'NSEE/Disiplinas' was successfully deleted." Below this, the user profile for "Tiagoem" is visible. A feed item shows "NSEE forked Tiagoem/Disiplinas to NSEE/Disiplinas 3 minutes ago". A "ProTip!" suggests editing the feed by updating followed users and watched repositories. A "Subscribe to your news feed" link is also present. On the right, the "Your repositories" section shows 5 repositories: Disciplinas, Linguagens1, embarcados-Diurno, embarcados-Noturno, and rafaelcorsi/EEN251.

Search GitHub

Pull requests Issues Gist

Your repository "NSEE/Disiplinas" was successfully deleted.

Tiagoem

NSEE forked Tiagoem/Disiplinas to NSEE/Disiplinas 3 minutes ago

ProTip! Edit your feed by updating the users you follow and repositories you watch.

Subscribe to your news feed

Your repositories 5

New repository

Find a repository...

All Public Private Sources Forks

- Disciplinas
- Linguagens1
- embarcados-Diurno
- embarcados-Noturno
- rafaelcorsi/EEN251



“Sim, podemos!”

git remote

# git remote

Verificar se o servidor remoto está configurado no seu repositório GIT.

```
git remote -v
```

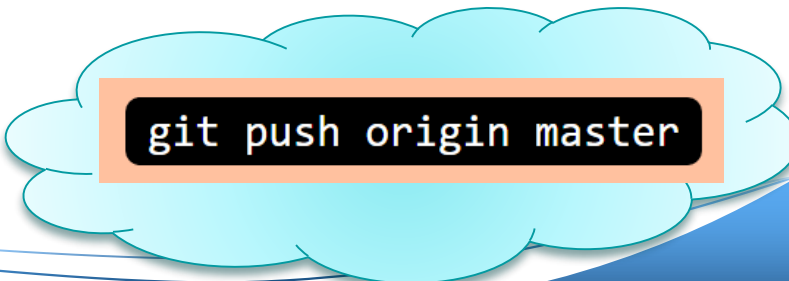
Caso esteja:

```
origin  git@github.com:Tiagoem/Disciplinas.git (fetch)  
origin  git@github.com:Tiagoem/Disciplinas.git (push)
```

Caso não:



Cada repositório **git local**, possui seu próprio **remote (server remoto)**



```
git push origin master
```

# git remote

git                      remote                      add                      **origin**  
git@gitserver:/opt/git/project.git

```
git remote -v
```

```
git push origin master
```

# Cenário 2



# Cenário 2

- Criar o repositório no github.
- “Anotar” o nome do repositório **remoto**.
- Clonar na maquina local o repositório remoto. (inicializa e configura o servidor remoto)
  - ❖ Será criado uma pasta com o nome do repositório.



- Trabalhe
- Adicione o(s) arquivo para entrar no seu próximo *commit*
- Realize um *commit*
- Atualize o repositório remoto

# Cenário 2

- Criar o repositório no github.

## Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: Tiagoem / Repository name: OlaMundo ✓

Great repository names are short and memorable. Need inspiration? How about `solid-octo-guacamole`.

### Description (optional)

Meu primeiro repositóriozinho :D

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

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

☐ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** | Add a license: **None** ⓘ

Create repository

https ou ssh?

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

Set up in Desktop or **HTTPS** **SSH**

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

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

```
echo "# OlaMundo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/Tiagoem/OlaMundo.git
git push -u origin master
```

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

```
git remote add origin https://github.com/Tiagoem/OlaMundo.git
git push -u origin master
```

...or import code from another repository

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

Import code

- “Anotar” o nome do repositório **remoto**.

# Cenário 2

- Clonar na maquina local o repositório remoto. (inicializa e configura o servidor remoto)
  - ❖ Será criado uma pasta com o nome do repositório.

```
git clone usuário@servidor:/caminho/para/o/repositório
```

*Ex: git clone https://github.com/Tiagoeem/teste.git*

- Verifique agora que o *remote* já vem configurado e aponta para o repositório do seu projeto no github.

```
git remote -v
```

# Cenário 2

- Atualize o repositório remoto

```
git push origin master
```

Alias (“apelido”) padrão do repositório remoto

Nome do *branch* que você quer atualizar

→ <https://github.com/Tiagoeem/teste.git>

Atualizar repositório local

# Atualizar repositório

Atualizar seu repositório local com a mais nova versão, disponível no repositório remoto.

```
git fetch  
git merge
```

ou

```
git pull
```

**SOMENTE de git pull caso o diretório de trabalho esteja limpo.**

```
TiagoLow@TiagoLow-PC MINGW64 /e/Cursos_repo (master)  
$ git status  
On branch master  
Your branch is ahead of 'origin/master' by 1 commit.  
  (use "git push" to publish your local commits)  
nothing to commit, working tree clean
```

# Outras ações

Liguei meu PC, vou começar a trabalhar no código.

Será que alguém subiu alguma alteração? Será que preciso atualizar meu repositório local (git pull ou git fetch && git merge) ?

O que devo fazer?



# Outras ações

Liguei meu PC, vou começar a trabalhar no código.

Será que alguém subiu alguma alteração? Será que preciso dar uma atualizada no meu repositório local (git pull ou git fetch && git merge)?

Verifique!

```
git remote update && git status
```

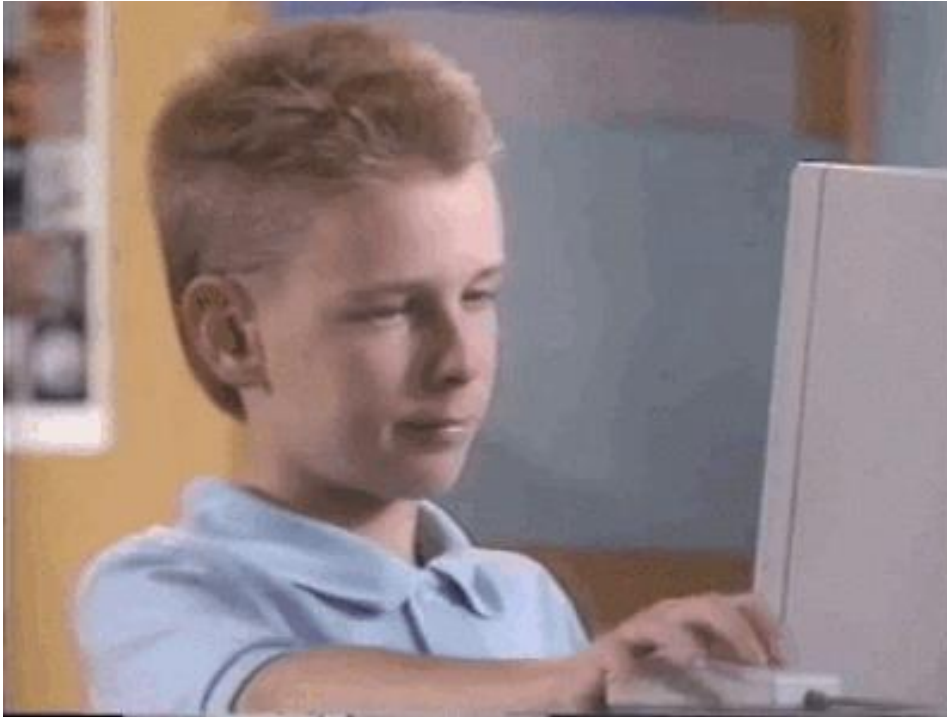


Clone ou Fork?

# Clone ou Fork: discussão em sala

Qual a diferença entre fazer o clone de um repositório e fazer um fork?

# Quer praticar mais?




1. <https://try.github.io/levels/1/challenges/1>
2. <https://learngitbranching.js.org/>

<http://comandosgit.github.io/>

## Vídeos:

- <https://www.youtube.com/watch?v=UMhskLXJuq4&t=2s>
- <https://www.youtube.com/watch?v=6Czd1Yetaac>

## PodCasts:

- <https://hipsters.tech/guia-do-iniciante-em-github-hipsters-184/>
  - <https://hipsters.tech/git-e-github-hipsters-109/>
- 

# Exercício

- Para que serve o git log?  
Para verificar os commits realizados no branch.  
(git log -oneline -n4)
- Como voltar a um commit específico?  
git checkout <hash-do-seu-commit>
- Como remover o último commit?  
<https://pt.stackoverflow.com/questions/3030/como-desfa%C3%A7o-o-%C3%BAltimo-commit-no-git>
- O que é o .gitignore?  
Arquivo que contém todas as pastas, arquivos e tipo de arquivos que devem ser ignorados pelo git.

Perguntas?