

GIT E GITHUB

Sistema de Controle de Versão – SCV

Autor: Wadd Franklin

O QUE É GIT?

Git é um sistema de controle de versão de arquivos.

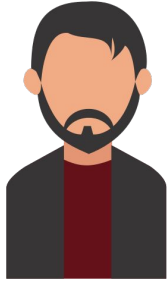
Com ele é possível:

- que várias pessoas trabalhem no mesmo arquivo simultaneamente;
- controlar as versões dos arquivos;
- desenvolvimento em paralelo (branches)



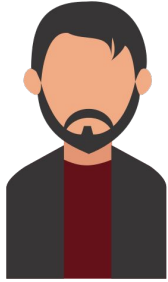


Samuel



Samuel





Samuel

```
printf("Hello World!");
```





Samuel

```
printf("Hello World!");
```



Samara



Samuel

```
printf("Hello World!");
```



Samara

```
while(!is_empty(stack)){  
    pop(stack);  
}
```



Samuel

```
printf("Hello World!");
```



Samara

```
while(!is_empty(stack)){  
    pop(stack);  
}
```

VAI OCORRER PERDA DE
CÓDIGO?



Samuel

```
printf("Hello World!");
```



Samara

```
while(!is_empty(stack)){  
    pop(stack);  
}
```

VAI OCORRER PERDA DE
CÓDIGO?

COM O GIT, NÃO!

O QUE É GITHUB?

GitHub é um serviço web que oferece várias funcionalidades extras aplicadas ao git.

Com o GitHub podemos:

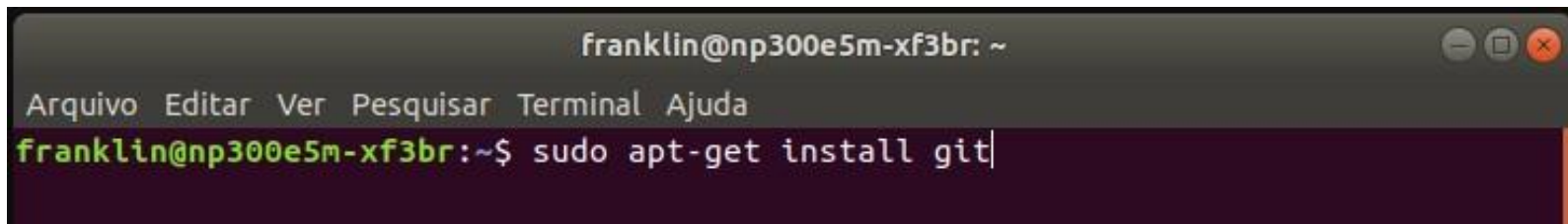
- hospedar o código fonte dos seus projetos
- Ter acesso ao código de outros usuários do GitHub
- Contribuir enviando correções de projetos de outras pessoas



INSTALANDO O GIT

Abra o terminal e digite:

\$ sudo apt-get install git

A screenshot of a terminal window. The title bar at the top reads 'franklin@np300e5m-xf3br: ~' and includes standard window control buttons (minimize, maximize, close). Below the title bar is a menu bar with the options 'Arquivo', 'Editar', 'Ver', 'Pesquisar', 'Terminal', and 'Ajuda'. The main area of the terminal shows a green prompt 'franklin@np300e5m-xf3br:~\$' followed by the command 'sudo apt-get install git|' in white text, with a cursor at the end of the line.

```
franklin@np300e5m-xf3br: ~  
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda  
franklin@np300e5m-xf3br:~$ sudo apt-get install git|
```


CRIANDO CONTA DO GITHUB


Para ter acesso aos serviços do GitHub, basta criar uma conta no site:


<https://github.com/>

Join GitHub

The best way to design, build, and ship software.

**Step 1:**
Create personal account

**Step 2:**
Choose your plan

**Step 3:**
Tailor your experience

Create your personal account

Username

Username is already taken

Email address

We'll occasionally send updates about your account to this inbox. We'll never share your email address with anyone.

Password

Use at least one lowercase letter, one numeral, and seven characters.

Verify account

You'll love GitHub

- Unlimited** collaborators
- Unlimited** public repositories
- ✓ Great communication
- ✓ Frictionless development
- ✓ Open source community

CRIANDO UM NOVO REPOSITÓRIO

Create a new repository

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

Owner



WaddFranklin ▾

/

Repository name

auladegit



Great repository names are short and memorable. Need inspiration? How about [glowing-octo-waddle](#).

Description (optional)

Repositório criado para apresentação do Git e GitHub



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

CRIANDO UM NOVO REPOSITÓRIO

Create a new repository

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

Owner



WaddFranklin ▾

Repository name

auladegit ✓

nome do repositório

Great repository names are short and memorable. Need inspiration? How about **glowing-octo-waddle**.

Description (optional)

Repositório criado para apresentação do Git e GitHub



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

CRIANDO UM NOVO REPOSITÓRIO

Create a new repository

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

Owner



WaddFranklin ▾

Repository name

auladegit ✓

nome do repositório

Great repository names are short and memorable. Need inspiration? How about **glowing-octo-waddle**.

Description (optional)

Repositório criado para apresentação do Git e GitHub

descrição



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

CRIANDO UM NOVO REPOSITÓRIO

tipo do repositório

Create a new repository

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

Owner



WaddFranklin ▾

Repository name

auladegit ✓

nome do repositório

Great repository names are short and memorable. Need inspiration? How about **glowing-octo-waddle**.

Description (optional)

Repositório criado para apresentação do Git e GitHub

descrição



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

CRIANDO UM NOVO REPOSITÓRIO

Create a new repository

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

Owner



WaddFranklin ▾

Repository name

auladegit ✓

nome do repositório

Great repository names are short and memorable. Need inspiration? How about **glowing-octo-waddle**.

Description (optional)

Repositório criado para apresentação do Git e GitHub

descrição

tipo do repositório



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.

gitignore

Add .gitignore: None ▾

Add a license: None ▾



Create repository

REPOSITÓRIO CRIADO!

Nesta tela temos várias funcionalidades pois ela é a tela principal do seu projeto. Por enquanto, vamos apenas nos concentrar no botão **"Clone or Download"**.

The screenshot shows a GitHub repository page for 'WaddFranklin / auladegit'. The repository is described as 'Repositório criado para apresentação do Git e GitHub'. It has 1 commit, 1 branch, 0 releases, and 1 contributor. The 'master' branch is selected. There are buttons for 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The file list shows '.gitignore' and 'README.md', both with 'Initial commit' status. The 'README.md' file is expanded, showing the repository name 'auladegit' and the same description.

WaddFranklin / auladegit

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Repositório criado para apresentação do Git e GitHub Edit

Add topics

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

WaddFranklin Initial commit Latest commit 89e226c just now

.gitignore	Initial commit	just now
README.md	Initial commit	just now

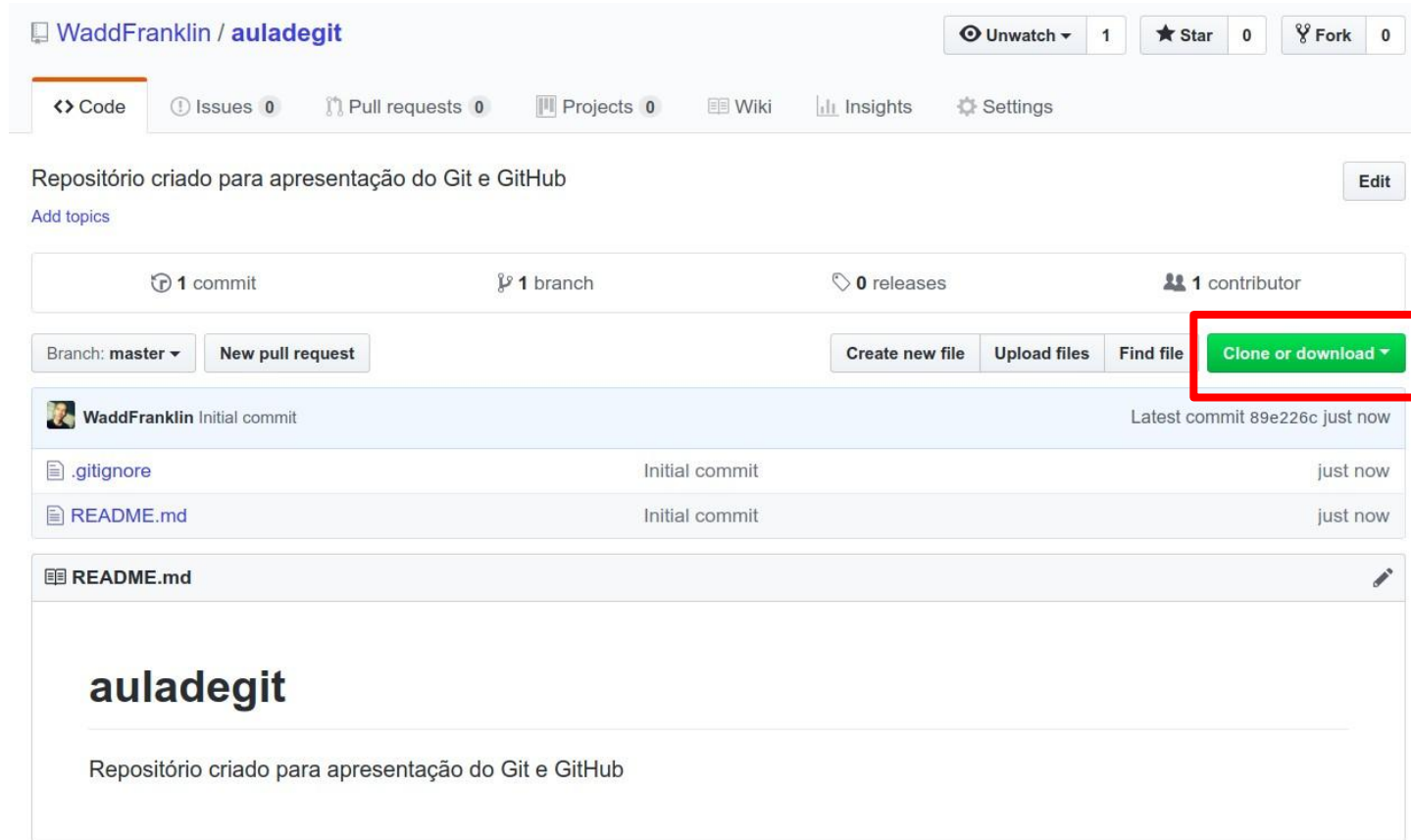
README.md

auladegit

Repositório criado para apresentação do Git e GitHub

REPOSITÓRIO CRIADO!

Clique neste botão e em seguida copie o link que vai aparecer. Nosso próximo passo é clonar este repositório para nossa máquina.



The screenshot shows a GitHub repository page for 'WaddFranklin / auladegit'. The repository is described as 'Repositório criado para apresentação do Git e GitHub'. It has 1 commit, 1 branch, 0 releases, and 1 contributor. The 'Clone or download' button is highlighted with a red box. Below the repository information, there is a table of files and their commit history. The 'README.md' file is shown with its commit history.

WaddFranklin / auladegit

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Repositório criado para apresentação do Git e GitHub

Add topics Edit

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

WaddFranklin Initial commit	Latest commit 89e226c just now
.gitignore Initial commit just now	
README.md Initial commit just now	

README.md

auladegit

Repositório criado para apresentação do Git e GitHub

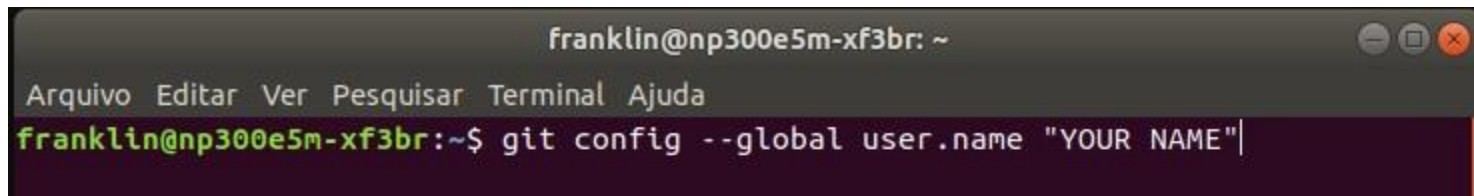
CONFIGURANDO O GIT

Antes de começar a utilizar o git, faremos algumas configurações que economizarão nosso tempo durante o trabalho.

Abra o terminal com o git instalado e digite:

```
$ git config --global user.name "YOUR NAME"
```

onde "YOUR NAME" é o seu login no github.

A screenshot of a terminal window with a dark background. The title bar at the top reads "franklin@np300e5m-xf3br: ~". Below the title bar is a menu bar with the options "Arquivo", "Editar", "Ver", "Pesquisar", "Terminal", and "Ajuda". The main area of the terminal shows the command "franklin@np300e5m-xf3br:~\$ git config --global user.name \"YOUR NAME\"" being entered. The prompt character is a green dollar sign. The command is in white text, and the cursor is at the end of the line.

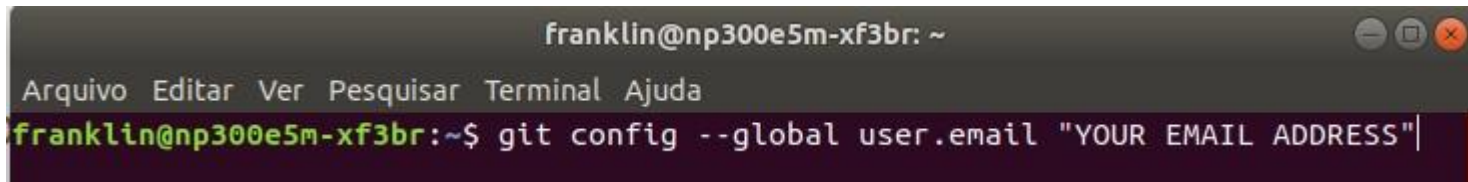
```
franklin@np300e5m-xf3br: ~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
franklin@np300e5m-xf3br:~$ git config --global user.name "YOUR NAME"
```

CONFIGURANDO O GIT

O próximo comando é:

```
$ git config --global user.email "YOUR EMAIL ADDRESS"
```

onde "YOUR EMAIL ADDRESS" é o email que você cadastrou no github.

A screenshot of a terminal window with a dark background. The title bar at the top reads "franklin@np300e5m-xf3br: ~" and includes standard window control buttons (minimize, maximize, close). Below the title bar is a menu bar with the options "Arquivo", "Editar", "Ver", "Pesquisar", "Terminal", and "Ajuda". The main area of the terminal shows a command prompt "franklin@np300e5m-xf3br:~\$" followed by the command "git config --global user.email \"YOUR EMAIL ADDRESS\"". The cursor is positioned at the end of the command line.

```
franklin@np300e5m-xf3br: ~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
franklin@np300e5m-xf3br:~$ git config --global user.email "YOUR EMAIL ADDRESS"
```

CLONANDO UM REPOSITÓRIO

Depois da configuração inicial, estamos prontos para clonar nosso repositório.

No git, clonar significa que estamos fazendo uma cópia do repositório que está no github para nossa máquina. Assim, nós podemos fazer alterações nesse repositório localmente e subir as alterações para o repositório remoto.

CLONANDO UM REPOSITÓRIO

Para clonar um repositório, digite no terminal:

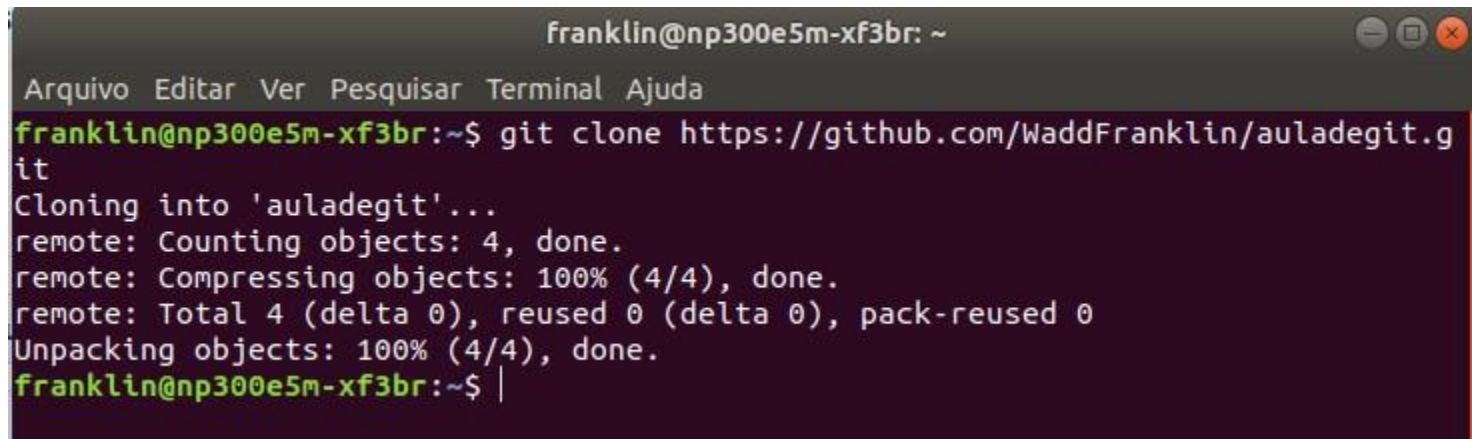
```
$ git clone <link do repositório>
```

onde "link do repositório" é aquele link que nós copiamos quando criamos nosso repositório lá no github. Em seguida pressione ENTER.

CLONANDO UM REPOSITÓRIO

Para clonar um repositório, digite no terminal:

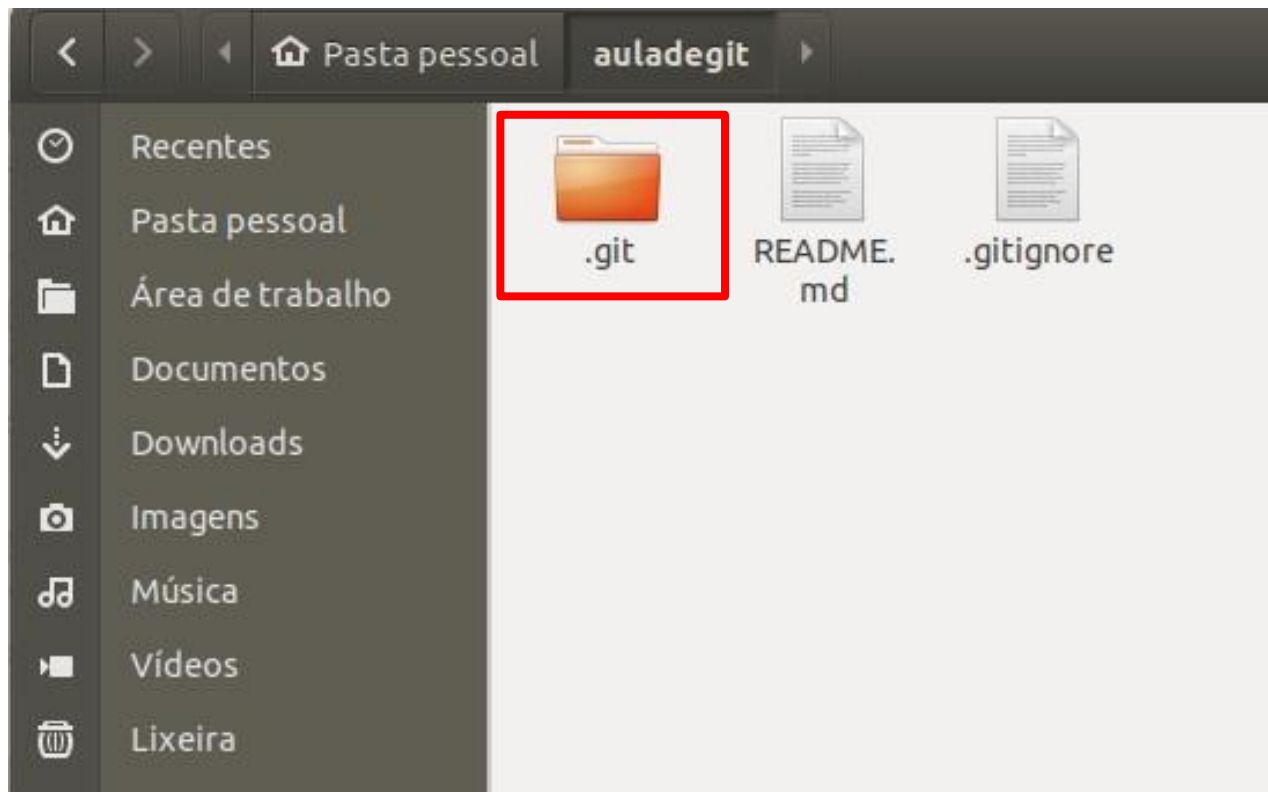
\$ git clone <link do repositório>

A screenshot of a terminal window titled 'franklin@np300e5m-xf3br: ~'. The window has a menu bar with 'Arquivo', 'Editar', 'Ver', 'Pesquisar', 'Terminal', and 'Ajuda'. The terminal shows the command 'git clone https://github.com/WaddFranklin/auladegit.git' being executed. The output shows the cloning process: 'Cloning into 'auladegit'...', 'remote: Counting objects: 4, done.', 'remote: Compressing objects: 100% (4/4), done.', 'remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0', and 'Unpacking objects: 100% (4/4), done.'. The prompt returns to 'franklin@np300e5m-xf3br:~\$' with a cursor at the end.

```
franklin@np300e5m-xf3br: ~  
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda  
franklin@np300e5m-xf3br:~$ git clone https://github.com/WaddFranklin/auladegit.g  
it  
Cloning into 'auladegit'...  
remote: Counting objects: 4, done.  
remote: Compressing objects: 100% (4/4), done.  
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (4/4), done.  
franklin@np300e5m-xf3br:~$ |
```

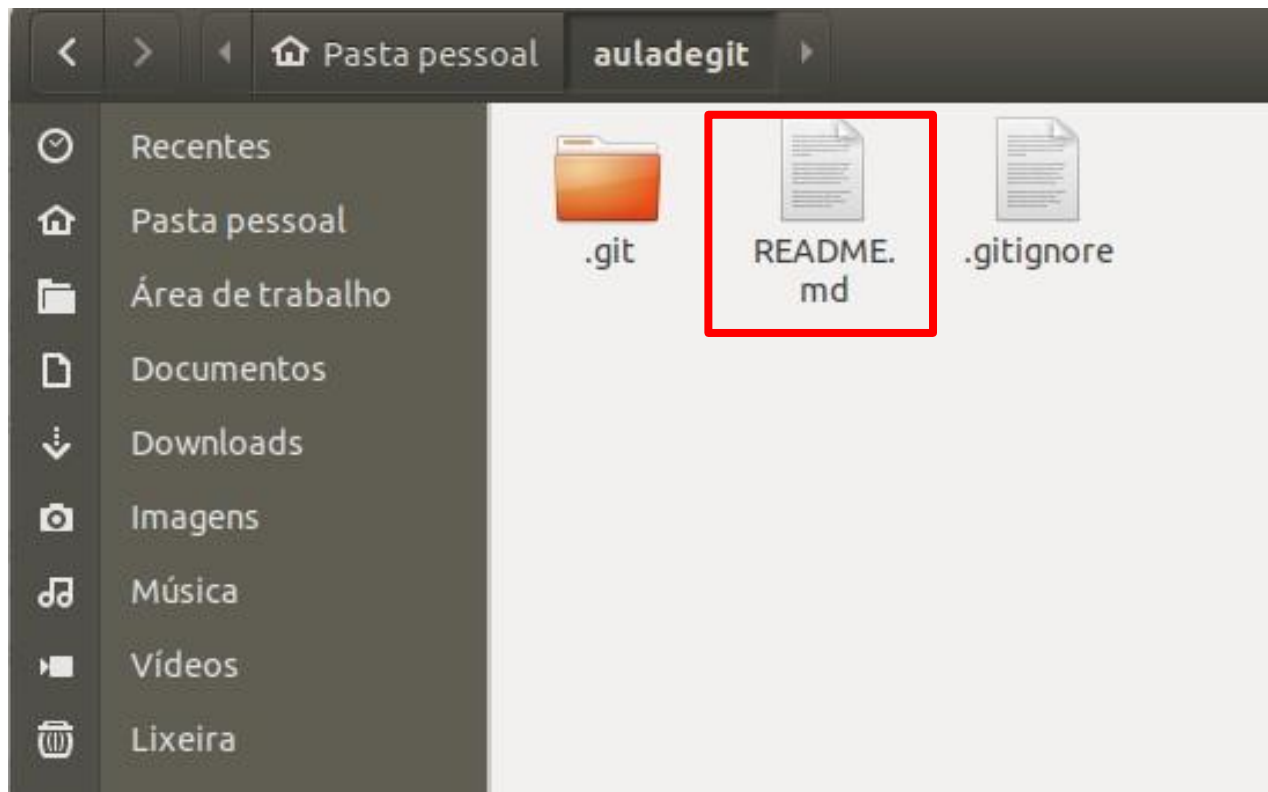

.GIT

Pasta com os principais arquivos de configuração do git.



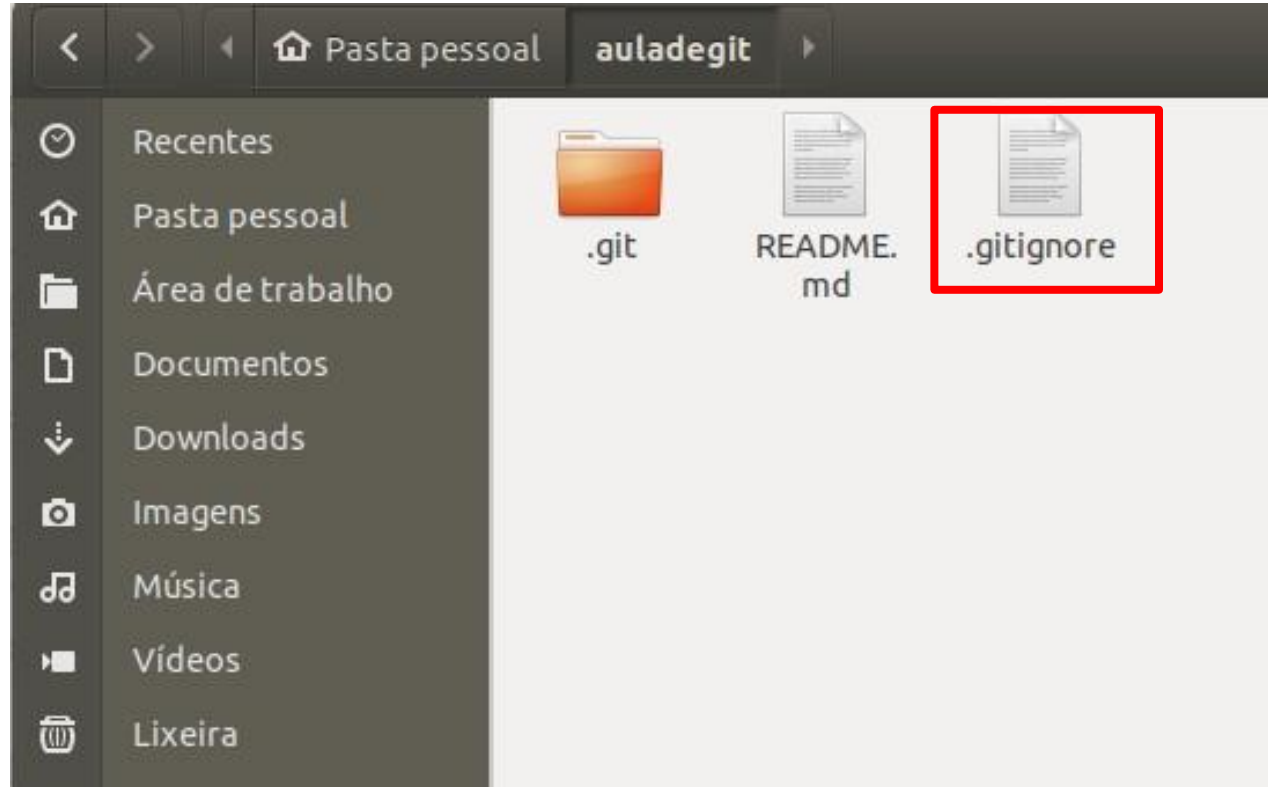
README.MD

Arquivo com a descrição do projeto.



.GITIGNORE

Arquivo contendo todas as extensões que não devem ser monitoradas pelo git.



OS 3 ESTÁGIOS

- **Untracked files:** as alterações feitas não estão prontas para serem commitadas
- **Staged:** as alterações feitas estão prontas para serem commitadas
- **Committed:** uma nova versão do projeto foi gerada e está pronta para ser inserida no master

4 COMANDOS BÁSICOS

Vamos aprender 4 comandos que usaremos com bastante frequência enquanto estivermos trabalhando com o git.

```
$ git add <arquivos...>
```

Este comando adiciona o(s) arquivo(s) em uma área do git onde possamos enviá-los para o github. Esta operação também é conhecida como **stage**.

É como se estivéssemos dizendo ao git que estes arquivos devem ser preparados para serem enviados para o repositório remoto.

```
$ git commit -m "message"
```

Este comando pega os arquivos staged e adiciona a estes um número (hash) e um comentário que poderão ser vistos por todos. Esta operação é chamada de **commit**.

\$ git status

Exibe o status atual do repositório. São exibidas informações como:

- Arquivos unstaged
- Arquivos staged
- Arquivos criados, alterados e removidos


```
$ git push
```

Literalmente "empurra" os arquivos que foram commitados para o repositório remoto.

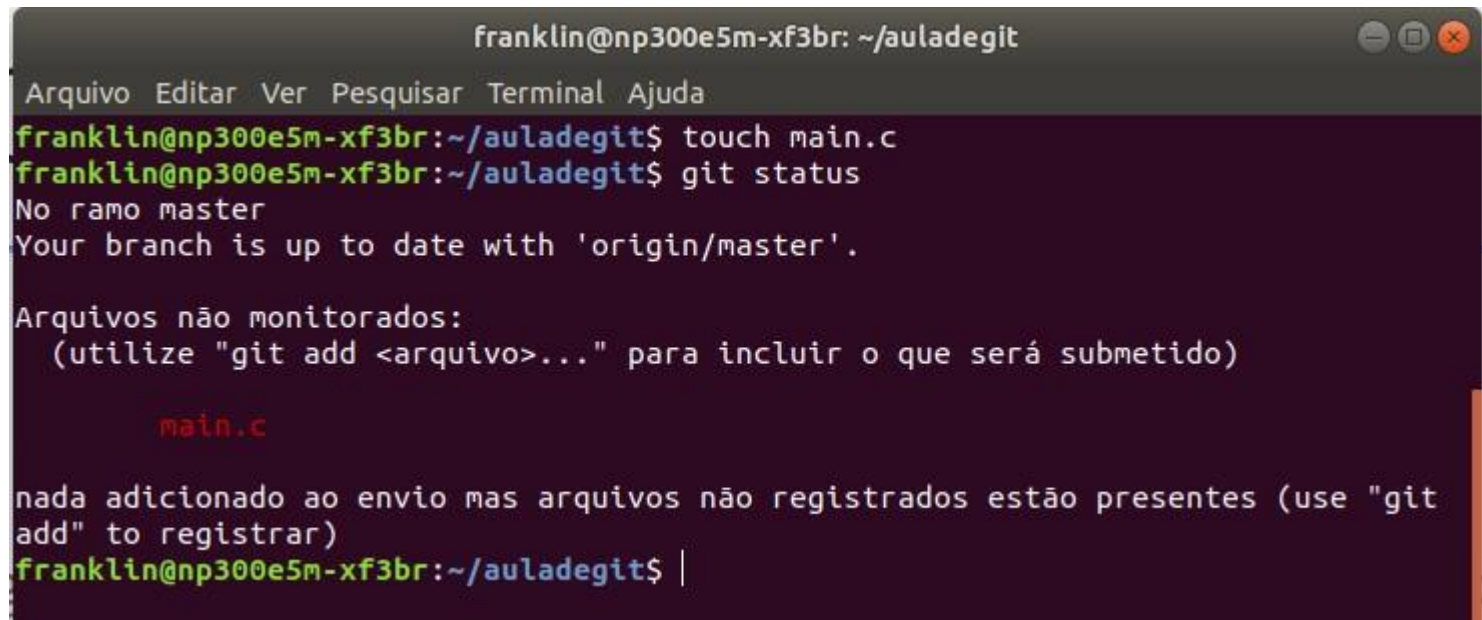
CRIANDO UM ARQUIVO NOVO

Agora vamos ver na prática como esses comandos funcionam.

Dentro da pasta auladegit:

1. digite: `$ touch main.c` para criar um arquivo em branco.
2. digite: `$ git status` para ver o status atual do seu repositório.

CRIANDO UM ARQUIVO NOVO



```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
franklin@np300e5m-xf3br:~/auladegit$ touch main.c
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Arquivos não monitorados:
  (utilize "git add <arquivo>..." para incluir o que será submetido)

    main.c

nada adicionado ao envio mas arquivos não registrados estão presentes (use "git
add" to registrar)
franklin@np300e5m-xf3br:~/auladegit$ |
```

CRIANDO UM ARQUIVO NOVO

Note que o git detectou que o arquivo `main.c` foi criado, porém ele não está na lista de arquivos staged, ou seja, a lista dos arquivos cujo versionamento está sendo controlado pelo git.

Dizemos que os arquivos nesse estágio são chamados de **untracked files**.

ALTERANDO E FAZENDO STAGE

Vamos adicionar algum conteúdo no nosso arquivo main.c

1. digite: `$ nano main.c` para abrir o arquivo com o editor de texto nano.
2. Digite alguma coisa no conteúdo do arquivo e salve-o.
3. digite: `$ git add main.c` para fazer o stage do arquivo.
4. digite: `$ git status` para ver o status atual.

ALTERANDO E FAZENDO STAGE

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo Editar Ver Pesquisar Terminal Ajuda
franklin@np300e5m-xf3br:~/auladegit$ nano main.c
franklin@np300e5m-xf3br:~/auladegit$ git add main.c
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Mudanças a serem submetidas:
  (use "git reset HEAD <file>..." to unstage)

        new file:   main.c

franklin@np300e5m-xf3br:~/auladegit$ |
```

ALTERANDO E FAZENDO STAGE

Agora o versionamento do nosso arquivo está sendo controlado pelo git. O arquivo `main.c` está staged, ou seja, pronto para receber commit e ser enviado para o repositório remoto.

COMMITANDO

Vamos commitar nosso arquivo. O commit deve ser uma descrição resumida e de fácil entendimento das alterações que foram feitas até esse ponto.

1. digite: `$ git commit -m "escrevendo o corpo do programa main.c"` seguido de ENTER.
2. digite: `$ git status` para ver o status atual.

COMMITANDO

```
franklin@np300e5m-xf3br:~/auladegit$ git commit -m "escrevendo o corpo do programa main.c"
[master 78cece9] escrevendo o corpo do programa main.c
 1 file changed, 6 insertions(+)
 create mode 100644 main.c
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Seu ramo está à frente de 'origin/master' por 1 submissão.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
franklin@np300e5m-xf3br:~/auladegit$ |
```

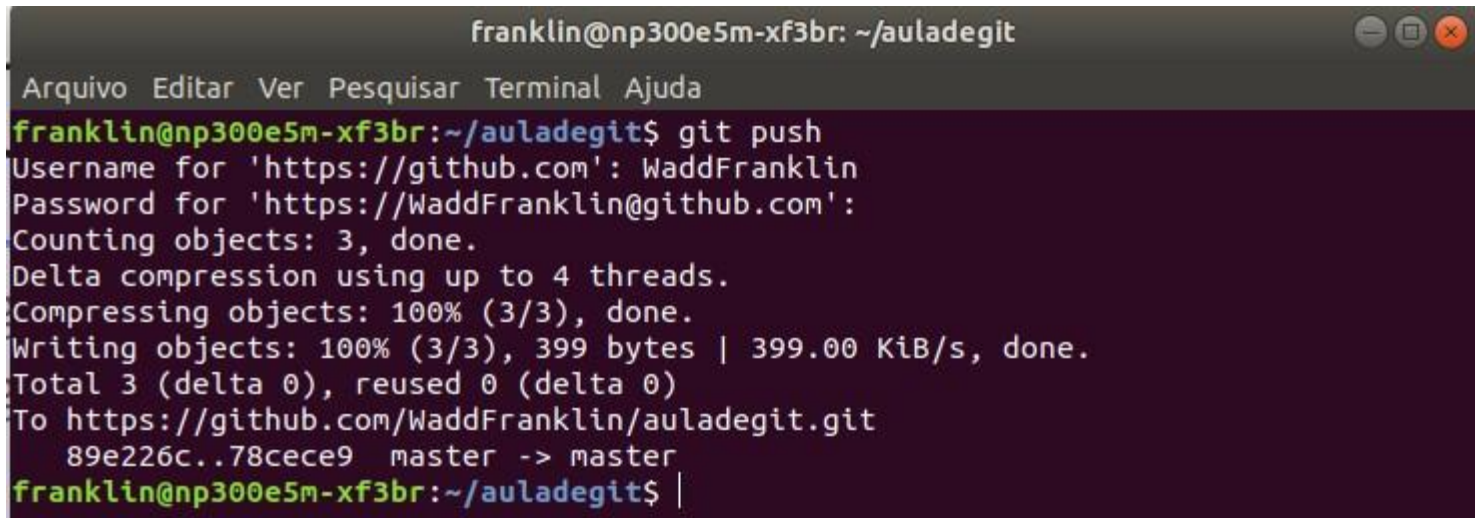
COMMITANDO

Pronto! Nosso arquivo já foi commitado.

Como não temos mais nenhum arquivo esperando para ser commitado, podemos enviar esta versão do nosso projeto para o repositório remoto através do comando **git push**.

FAZENDO PUSH PARA O REPOSITÓRIO REMOTO

1. digite: **\$ git push** seguido de ENTER.
2. O git vai solicitar seu login e senha da conta github.
Entre com suas credenciais e tecle ENTER.

A terminal window titled 'franklin@np300e5m-xf3br: ~/auladegit' with standard window controls. The terminal shows the execution of 'git push' and its output. The prompt is 'franklin@np300e5m-xf3br:~/auladegit\$'. The output includes prompts for username and password, progress bars for counting and compressing objects, and the final commit hash and branch name.

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
franklin@np300e5m-xf3br:~/auladegit$ git push
Username for 'https://github.com': WaddFranklin
Password for 'https://WaddFranklin@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 399 bytes | 399.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/WaddFranklin/auladegit.git
 89e226c..78cece9  master -> master
franklin@np300e5m-xf3br:~/auladegit$ |
```

<> Code

! Issues 0

🔗 Pull requests 0

📁 Projects 0

📖 Wiki

📊 Insights

⚙️ Settings

Repositório criado para apresentação do Git e GitHub

Edit

[Add topics](#)

🕒 2 commits

🌿 1 branch

📦 0 releases

👤 1 contributor

Branch: master ▾

New pull request

Create new file

Upload files

Find file

Clone or download ▾



WaddFranklin escrevendo o corpo do programa main.c

Latest commit 78cece9 11 minutes ago

📄 .gitignore

Initial commit

3 hours ago

📄 README.md

Initial commit

3 hours ago

📄 main.c

escrevendo o corpo do programa main.c

10 minutes ago

📖 README.md



auladegit

Repositório criado para apresentação do Git e GitHub

E SE EU QUISER VER TODOS OS COMMITS DO PROJETO?

É SE EU QUISER VER TODOS OS COMMITS DO PROJETO?

```
$ git log
```

E SE EU QUISER VER TODOS OS COMMITS DO PROJETO?

Com o comando **git log**, é possível ver todos os commits feitos bem como a hash de identificação de cada um, data e hora em que foram commitados.

E SE EU QUISE VER TODOS OS COMMITS DO PROJETO?

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo Editar Ver Pesquisar Terminal Ajuda
franklin@np300e5m-xf3br:~/auladegit$ git log
commit f197ce06698df88067cf07a5eeba7b6e1fbd5373 (HEAD -> master, origin/master,
origin/HEAD)
Author: Wadd Franklin <waddinsohn@gmail.com>
Date:   Sun Sep 9 22:44:51 2018 -0300

    Create tree.c

commit 78cece9af2353b8c3bf80c93e6ab498d3f7621bb
Author: Wadd Franklin <waddinsohn@gmail.com>
Date:   Sun Sep 9 22:09:00 2018 -0300

    escrevendo o corpo do programa main.c

commit 89e226c52a9181eb9ed8b9891add933c7f044fde
Author: Wadd Franklin <waddinsohn@gmail.com>
Date:   Sun Sep 9 19:12:01 2018 -0300

    Initial commit
franklin@np300e5m-xf3br:~/auladegit$ |
```


ERREI A MENSAGEM DO COMMIT... E AGORA?

ERREI A MENSAGEM DO COMMIT... E AGORA?

```
$ git commit --amend
```

ERREI A MENSAGEM DO COMMIT... E AGORA?

Com o **git commit --amend**, você pode editar a mensagem de um commit que já foi feito.

COMO FAZER O "UNSTAGE" DE UM ARQUIVO?

Suponha que você tenha um arquivo staged, ou seja, um arquivo pronto para commit, e queira que este arquivo retorne ao estado anterior (**untracked file**).

Para isso usamos o comando **git reset HEAD <arquivo>**.

COMO FAZER O "UNSTAGE" DE UM ARQUIVO?

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo Editar Ver Pesquisar Terminal Ajuda
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Mudanças a serem submetidas:
  (use "git reset HEAD <file>..." to unstage)

    new file:   texto.txt

franklin@np300e5m-xf3br:~/auladegit$ git reset HEAD texto.txt
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Arquivos não monitorados:
  (utilize "git add <arquivo>..." para incluir o que será submetido)

    texto.txt

nada adicionado ao envio mas arquivos não registrados estão presentes (use "git
add" to registrar)
franklin@np300e5m-xf3br:~/auladegit$ |
```

COMO FAZER O "UNSTAGE" DE UM ARQUIVO?

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo Editar Ver Pesquisar Terminal Ajuda
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Mudanças a serem submetidas:
  (use "git reset HEAD <file>..." to unstage)

    new file:   texto.txt

franklin@np300e5m-xf3br:~/auladegit$ git reset HEAD texto.txt
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Arquivos não monitorados:
  (utilize "git add <arquivo>..." para incluir o que será submetido)

    texto.txt

nada adicionado ao envio mas arquivos não registrados estão presentes (use "git
add" to registrar)
franklin@np300e5m-xf3br:~/auladegit$ |
```

arquivo staged



COMO FAZER O "UNSTAGE" DE UM ARQUIVO?

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo Editar Ver Pesquisar Terminal Ajuda
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Mudanças a serem submetidas:
  (use "git reset HEAD <file>..." to unstage)

    new file:   texto.txt

franklin@np300e5m-xf3br:~/auladegit$ git reset HEAD texto.txt
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Arquivos não monitorados:
  (utilize "git add <arquivo>..." para incluir o que será submetido)

    texto.txt

nada adicionado ao envio mas arquivos não registrados estão presentes (use "git
add" to registrar)
franklin@np300e5m-xf3br:~/auladegit$ |
```

COMO FAZER O "UNSTAGE" DE UM ARQUIVO?

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo Editar Ver Pesquisar Terminal Ajuda
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Mudanças a serem submetidas:
  (use "git reset HEAD <file>..." to unstage)

    new file:   texto.txt

franklin@np300e5m-xf3br:~/auladegit$ git reset HEAD texto.txt
franklin@np300e5m-xf3br:~/auladegit$ git status
No ramo master
Your branch is up to date with 'origin/master'.

Arquivos não monitorados:
  (utilize "git add <arquivo>..." para incluir o que será submetido)

    texto.txt

nada adicionado ao envio mas arquivos não registrados estão presentes (use "git
add" to registrar)
franklin@np300e5m-xf3br:~/auladegit$
```

arquivo staged

arquivo unstaged

SUPONHA A SEGUINTE SITUAÇÃO...

SUPONHA A SEGUINTE SITUAÇÃO...

repositório remoto



main.c



.gitignore



README.md



tree.c

SUPONHA A SEGUINTE SITUAÇÃO...

repositório remoto



main.c



.gitignore



README.md



tree.c

repositório local



main.c



.gitignore



README.md

SUPONHA A SEGUINTE SITUAÇÃO...

repositório remoto



main.c



.gitignore



README.md



tree.c

repositório local



main.c



.gitignore



README.md

Nosso repositório remoto tem uma arquivo a mais do que o repositório local.

Solução...

```
$ git pull
```

\$ git pull

Este comando serve para atualizarmos nosso repositório local de acordo com o remoto.

Assim, quem estiver trabalhando no projeto pode obter a versão mais atual do mesmo.

1. Suponha que tenhamos criado o arquivo `tree.c` no repositório remoto.
2. Para atualizar o repositório local basta, estando dentro da pasta `auladegit`, digitar o comando: `$ git pull`.

Repositório criado para apresentação do Git e GitHub


[Edit](#)[Add topics](#) 3 commits 1 branch 0 releases 1 contributor

Branch: master ▾

[New pull request](#)[Create new file](#)[Upload files](#)[Find file](#)[Clone or download ▾](#)

WaddFranklin Create tree.c

Latest commit f197ce0 5 minutes ago

 .gitignore	Initial commit	4 hours ago
 README.md	Initial commit	4 hours ago
 main.c	escrevendo o corpo do programa main.c	41 minutes ago
 tree.c	Create tree.c	5 minutes ago

 README.md

auladegit

Repositório criado para apresentação do Git e GitHub

franklin@np300e5m-xf3br: ~/auladegit

Arquivo Editar Ver Pesquisar Terminal Ajuda

```
franklin@np300e5m-xf3br:~/auladegit$ git pull
```

```
remote: Counting objects: 3, 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), done.
```

```
From https://github.com/WaddFranklin/auladegit
```

```
78cece9..f197ce0 master -> origin/master
```

```
Updating 78cece9..f197ce0
```

```
Fast-forward
```

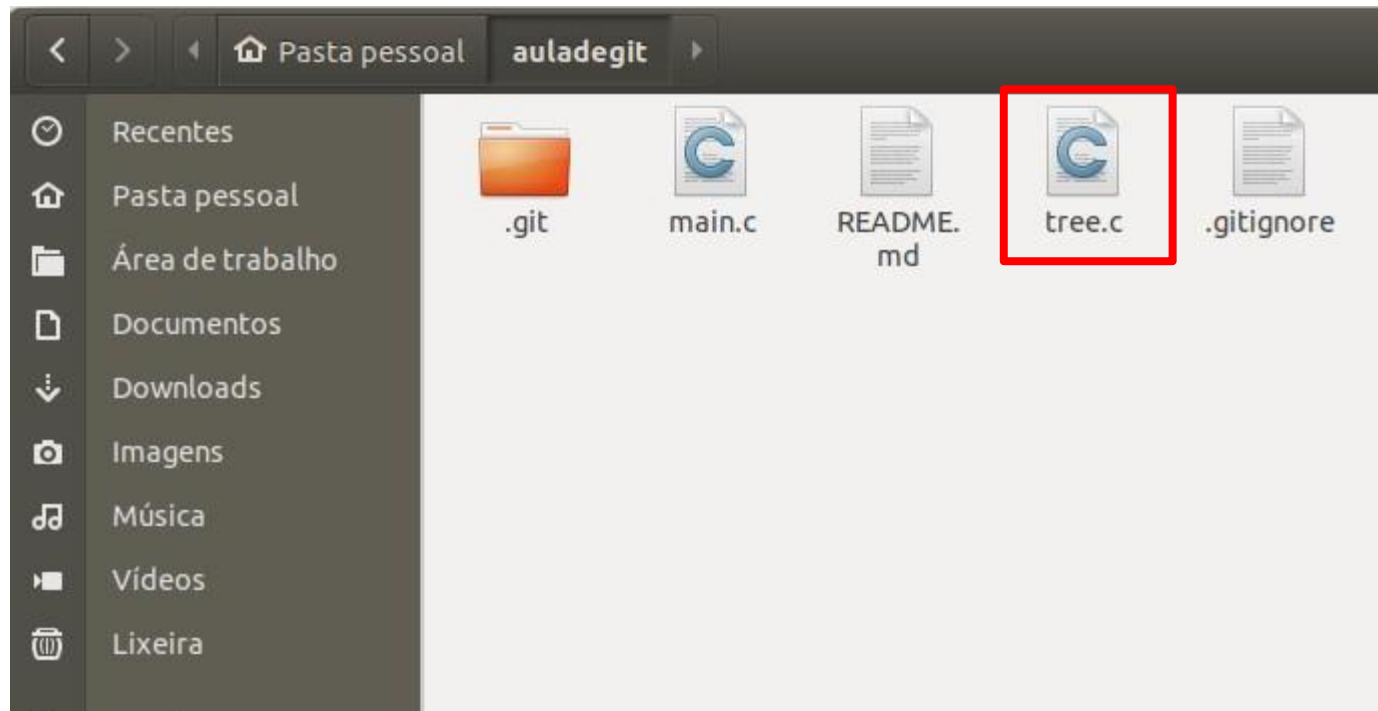
```
tree.c | 5 +++++
```

```
1 file changed, 5 insertions(+)
```

```
create mode 100644 tree.c
```

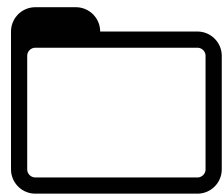
```
franklin@np300e5m-xf3br:~/auladegit$ |
```


O arquivo
tree.c foi
adicionado ao
nosso
repositório
local,
deixando os
dois
atualizados.



SUPONHA AGORA QUE...

SUPONHA AGORA QUE...

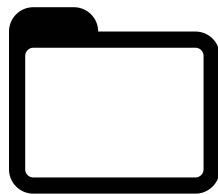


auladegit

SUPONHA AGORA QUE...



Samuel



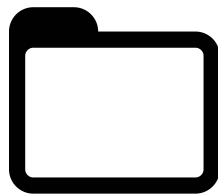
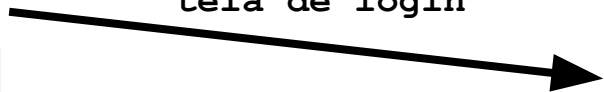
auladegit

SUPONHA AGORA QUE...



Samuel

tela de login



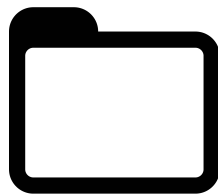
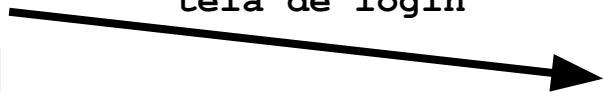
auladegit

SUPONHA AGORA QUE...



Samuel

tela de login

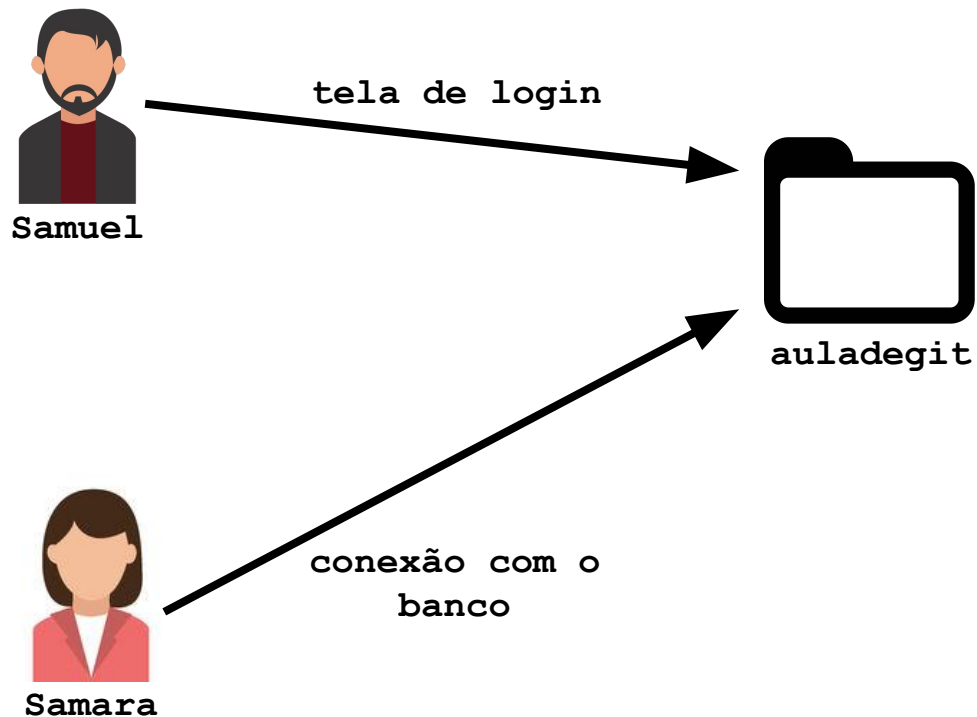


auladegit

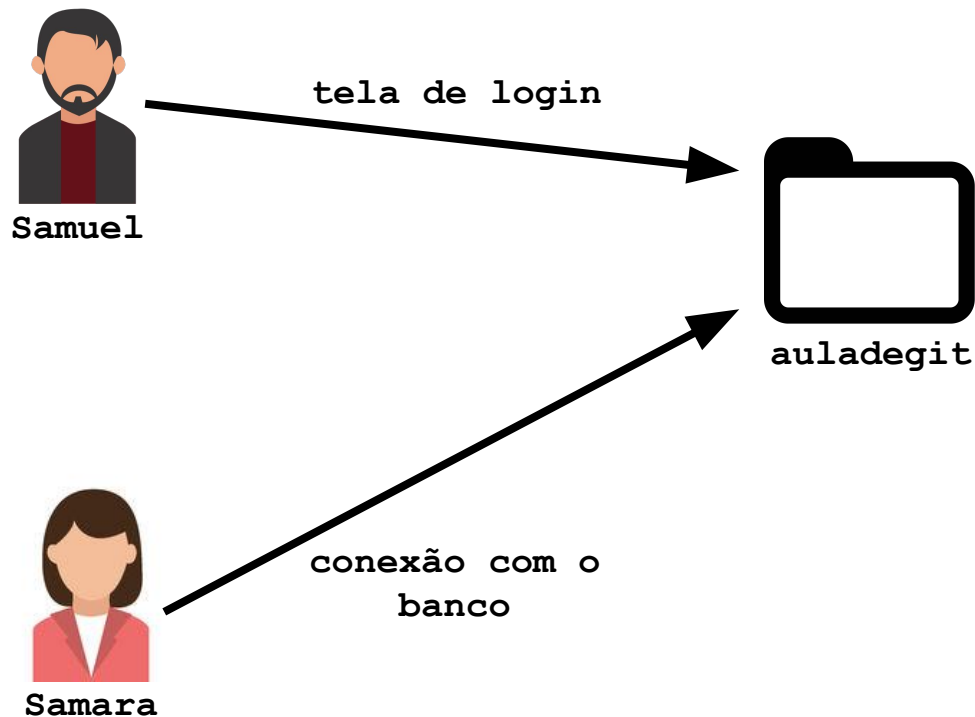


Samara

SUPONHA AGORA QUE...



SUPONHA AGORA QUE...



Como Samara e Samuel podem desenvolver funcionalidades diferentes para o projeto, ao mesmo tempo, sem comprometer o que já foi feito e está funcionando?

BRANCHES

Desenvolvimento em paralelo

master





Samuel

master



Samara



Samuel



Samara



Vou implementar a
tela de login!

Samuel

master



Samara



Vou implementar a
tela de login!

Samuel

login

master



Samara



Samuel

login

master

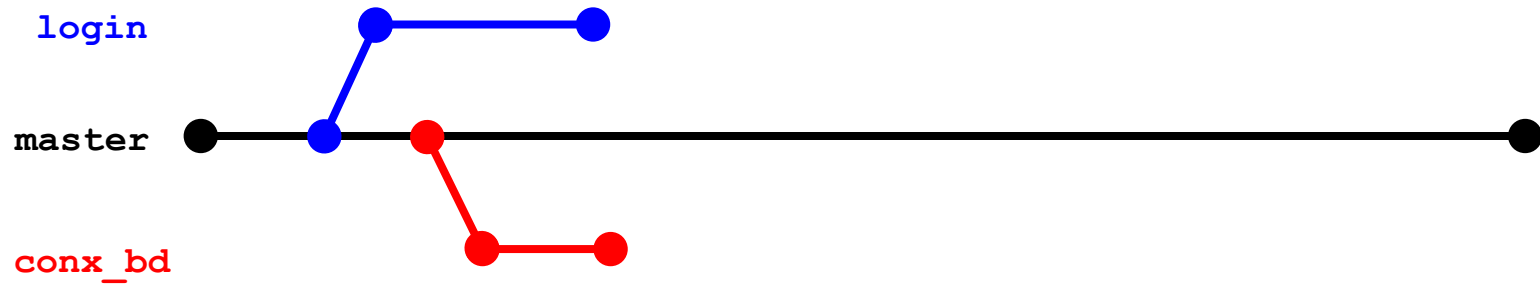


Samara

Ok! Então eu vou
implementar a
conexão com o BD!



Samuel



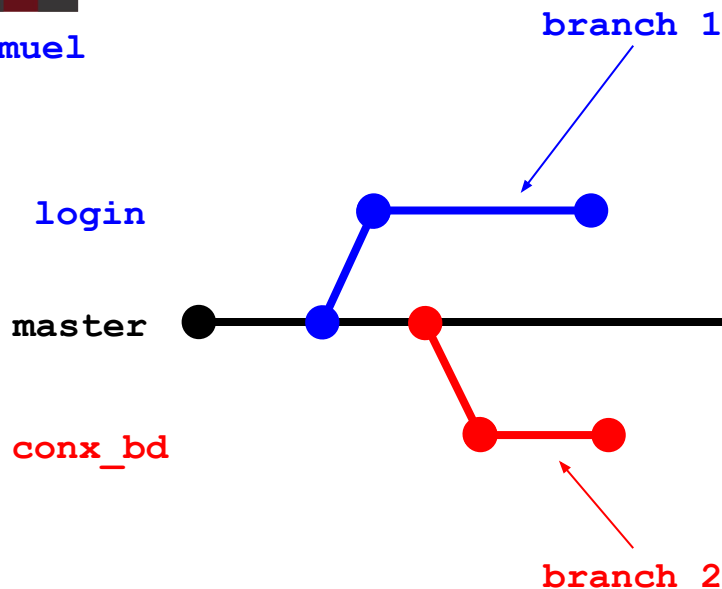
Samara

Ok! Então eu vou
implementar a
conexão com o BD!



Samuel

Agora ambos podem desenvolver suas features ao mesmo tempo sem comprometer a versão atual do projeto.



Samara

```
$ git checkout -b <nome_da_branch>
```

Cria uma nova branch com o nome desejado.

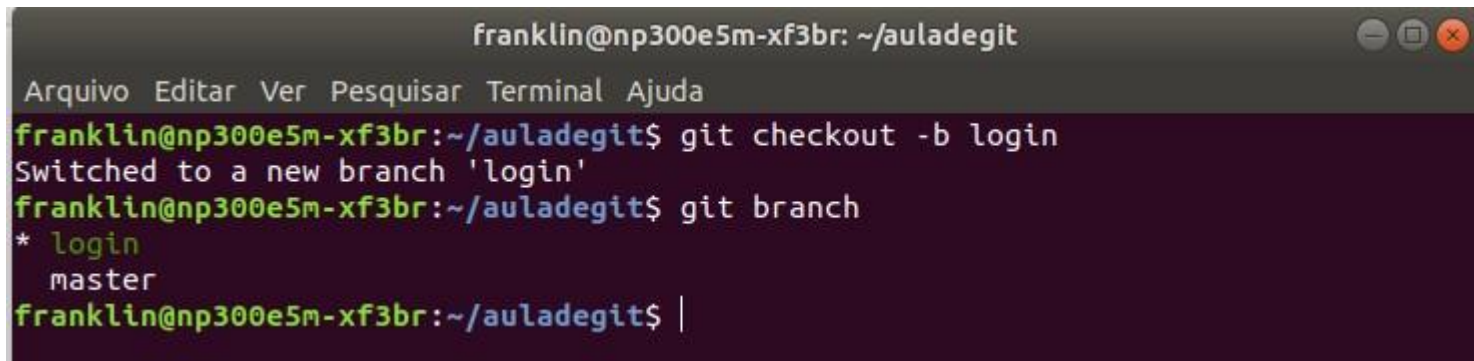
Ao criar uma branch, o git automaticamente muda a sua branch atual para a que for criada.

\$ git branch

Mostra todas as branches do projeto e em qual branch você está atualmente.

\$ git branch

Mostra todas as branches do projeto e em qual branch você está atualmente.

A terminal window titled 'franklin@np300e5m-xf3br: ~/auladegit' with standard window controls. The menu bar includes 'Arquivo', 'Editar', 'Ver', 'Pesquisar', 'Terminal', and 'Ajuda'. The terminal shows the following sequence of commands and output:

```
franklin@np300e5m-xf3br:~/auladegit$ git checkout -b login
Switched to a new branch 'login'
franklin@np300e5m-xf3br:~/auladegit$ git branch
* login
  master
franklin@np300e5m-xf3br:~/auladegit$ |
```

BRINCANDO COM AS BRANCHES

1. Agora que estamos na branch `login`, crie 3 arquivos:
`login1.txt`, `login2.txt` e `login3.txt`.
2. Com um editor de texto, digite e salve no arquivo.
`login1.txt`: "Arquivo de login 1".
3. digite `git status` para ver o status atual desta branch
4. digite `git add .` para fazer o stage de todos os arquivos.
5. Commite digitando `git commit` e ENTER.

PRONTO!

Nossos 3 arquivos de login foram criados e commitados na branch **login**.

Vamos observar uma coisa...

1. digite **git checkout master** para mudar para a branch principal.
2. digite **git log** para verificar o último commit na branch **master**

Notou algo?

PRONTO!

Nossos 3 arquivos de login foram criados e commitados na branch **login**.

Vamos observar uma coisa...

1. digite **git checkout master** para mudar para a branch principal.
2. digite **git log** para verificar o último commit na branch **master**

Notou algo?



CADÊ OS ARQUIVOS QUE CRIEI?

Os arquivos de login foram criados na branch login, ou seja, por enquanto, eles só existem naquela branch.

Para que as alterações da branch login sejam refletidas na branch master, precisamos fazer um...

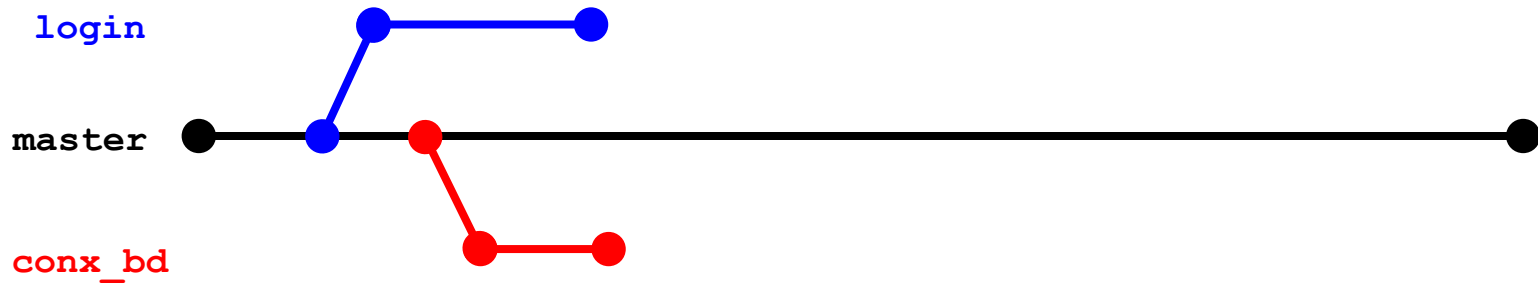
MERGE

Mescle as alterações da sua branch com a master!



Samuel

Samuel precisa
atualizar o projeto
com as funcionalidades
de login que ele
implementou

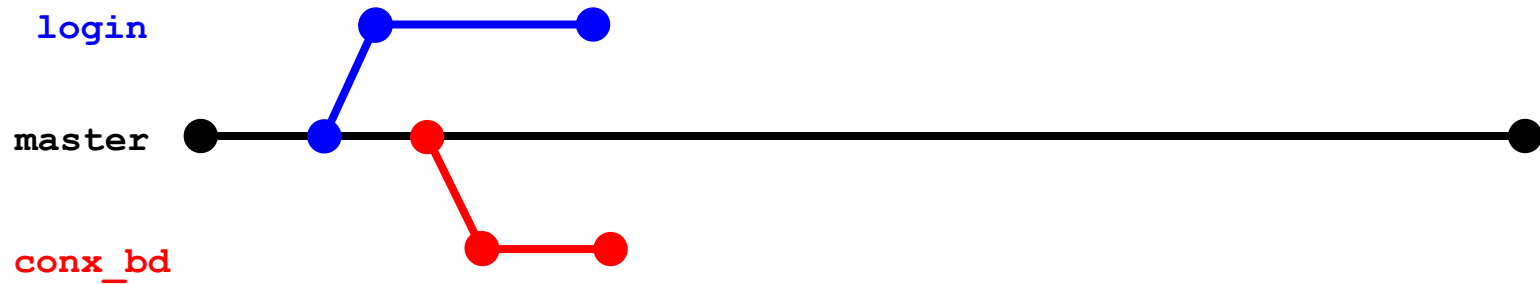


Samara



Terminei! Agora
vou mandar tudo
para o master.

Samuel

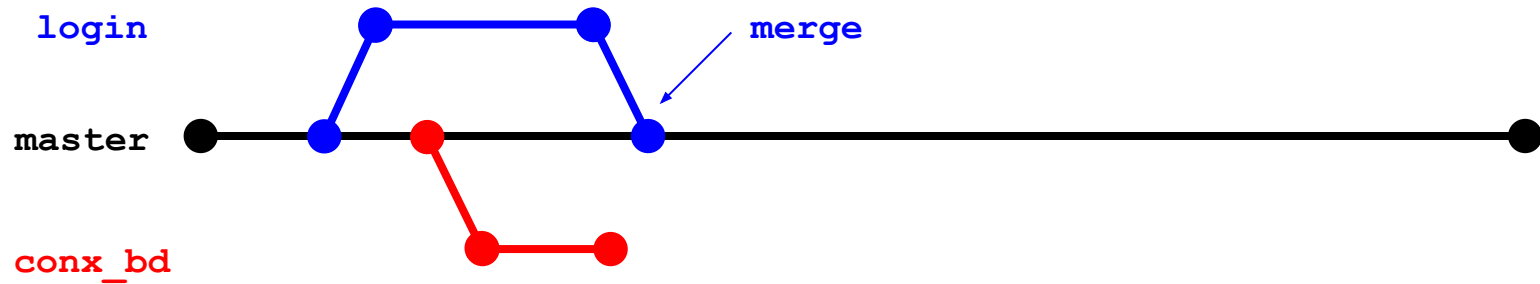


Samara



Terminei! Agora
vou mandar tudo
para o master.

Samuel



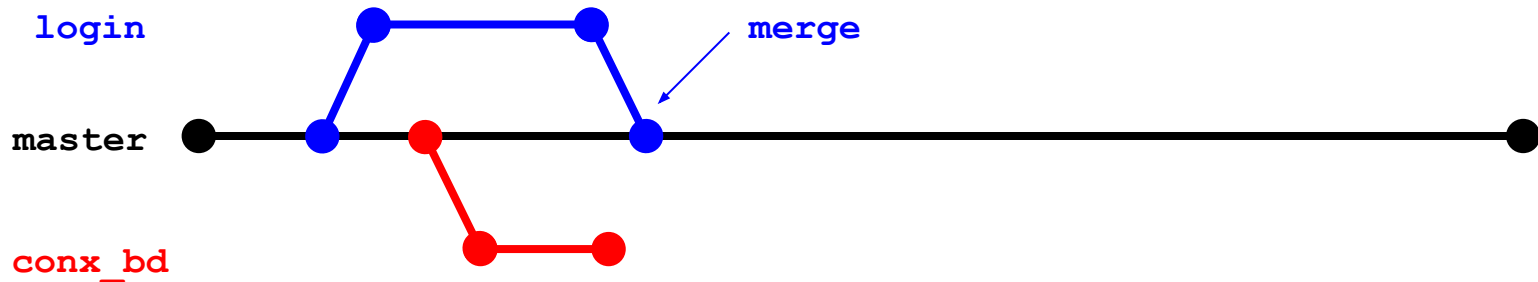
Samara



Terminei! Agora
vou mandar tudo
para o master.

Samuel

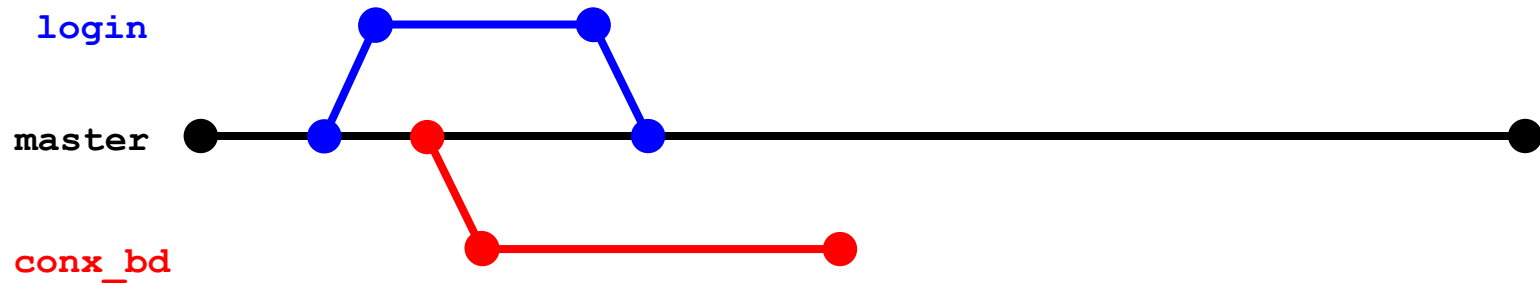
Agora o projeto possui
todas as
funcionalidades que
Samuel implementou!



Samara



Samuel

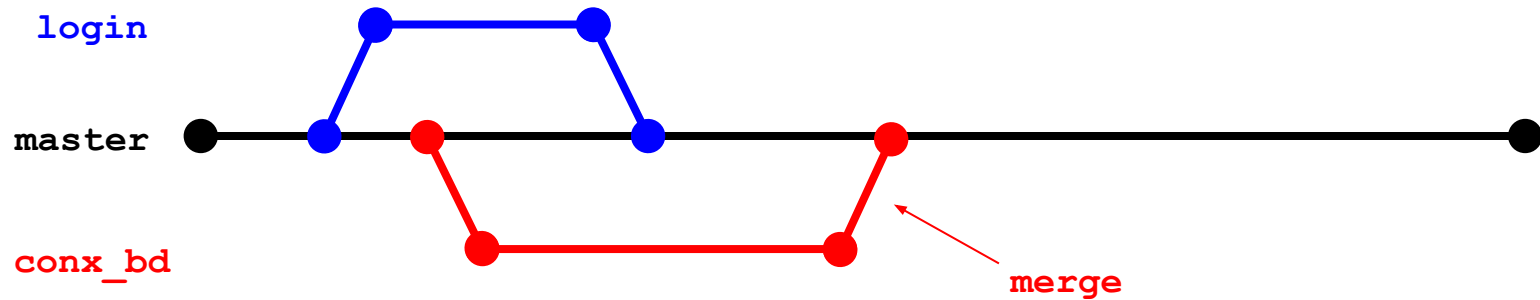


Samara

Também terminei.
Vou fazer um
merge com a
branch master.



Samuel



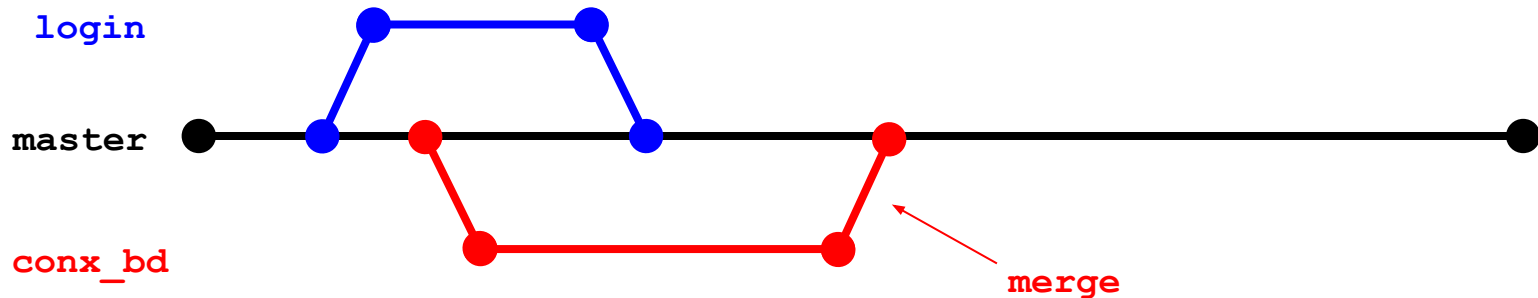
Samara

Também terminei.
Vou fazer um
merge com a
branch master.



Samuel

Pronto! Agora o projeto possui as funcionalidades implementadas por Samuel e Samara!



Samara

Também terminei.
Vou fazer um
merge com a
branch master.

NA PRÁTICA

Nossa branch **master** não possui os arquivos de login da branch **login**. Então vamos fazer um merge de **login** -> **master**:

1. Digite **git checkout master** pra mudar para a branch **master**
2. digite **git merge login** para mesclar as alterações de **login** com a **master**.
3. digite **ls** para confirmar que os arquivos de login agora também estão na branch **master**.

NA PRÁTICA

```
franklin@np300e5m-xf3br:~/auladegit$ git merge login
Updating 18aad5c..0c7965d
Fast-forward
 login1.txt | 1 +
 login2.txt | 0
 login3.txt | 0
 3 files changed, 1 insertion(+)
 create mode 100644 login1.txt
 create mode 100644 login2.txt
 create mode 100644 login3.txt
franklin@np300e5m-xf3br:~/auladegit$ ls
login1.txt login2.txt login3.txt main.c README.md texto.txt tree.c
franklin@np300e5m-xf3br:~/auladegit$ |
```

NA PRÁTICA

1. Agora mude novamente para a branch **login**
2. Abra o arquivo **login2.txt** e digite:

linha 1

linha 2

linha 3

linha 4

linha 5

3. Salve e commite esta alteração

NA PRÁTICA

1. Mude para a branch **master**
2. Abra o arquivo **login2.txt** e digite:

linha 99

linha 100

linha 101

3. Salve e commite esta alteração

NA PRÁTICA

1. Mude para a branch **master**
2. digite **git merge login**
3. O que aconteceu?

NA PRÁTICA

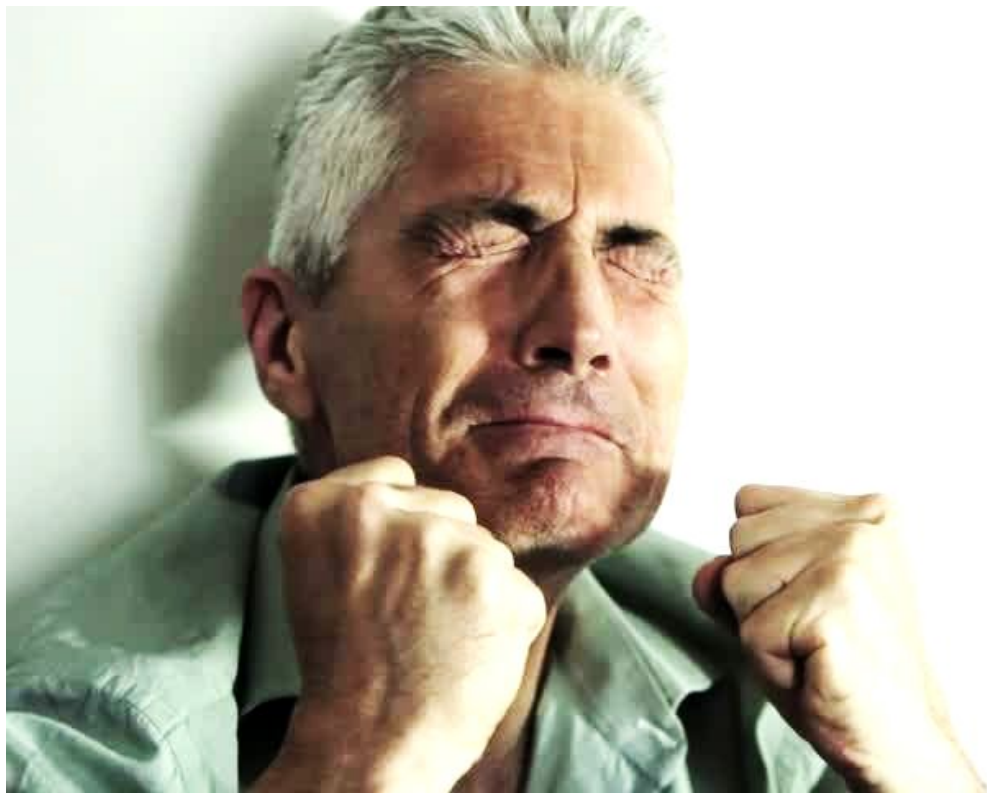
1. Mude para a branch **master**
2. digite **git merge login**
3. O que aconteceu?

```
franklin@np300e5m-xf3br:~/auladegit$ git merge login
Mesclagem automática de login2.txt
CONFLITO (conteúdo): conflito de mesclagem em login2.txt
Automatic merge failed; fix conflicts and then commit the result.
```

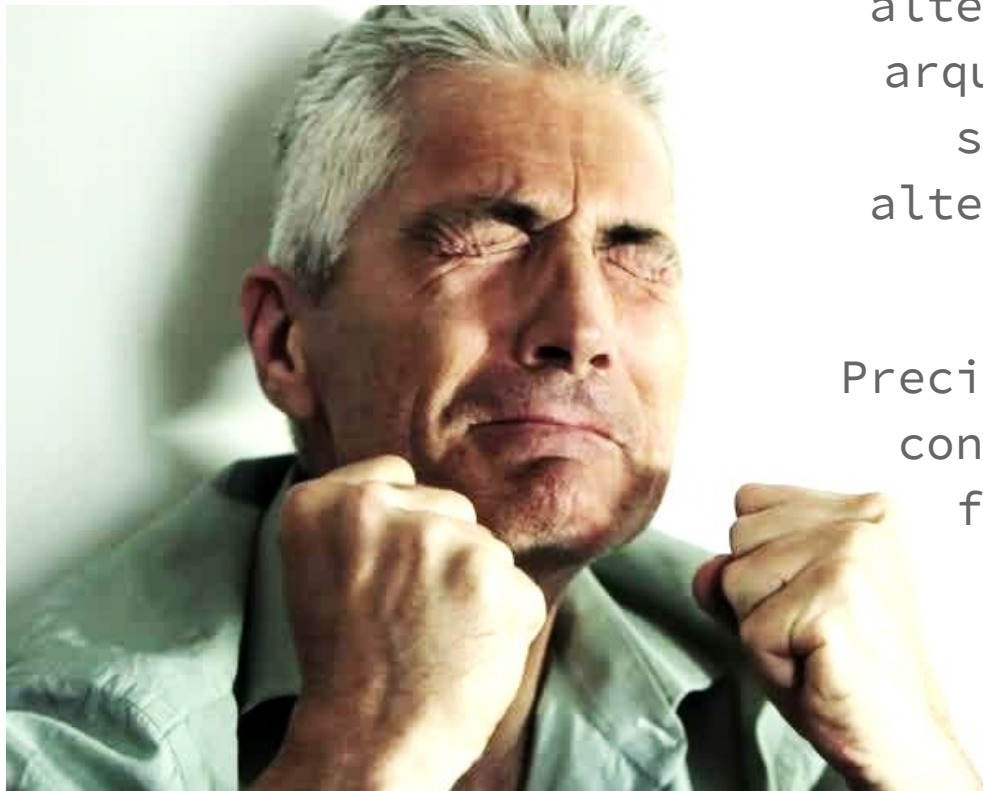
MERGE COM CONFLITO

**O mesmo arquivo possui commits diferentes em
branches diferentes**

E AGORA?



E AGORA?



Como existem duas alterações no mesmo arquivo, o git não sabe qual das alterações ele deve manter.

Precisamos resolver o conflito antes de fazer o merge novamente

O GTI FACILITA AS COISAS PARA VOCÊ! :)

Resolvendo o conflito:

1. Com o editor de texto, abra o arquivo **login2.txt**
2. Veja que o git marcou as partes conflitantes deste arquivo.
3. Edite deixando apenas as partes de código que devem ser mantidas e salve o arquivo.
4. Commite

O GIT FACILITA AS COISAS PARA VOCÊ! :)

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
[1/1] login2.txt

<<<<<<< HEAD
linha 99
lina 100
linha 101
=====
linha 1
linha 2
linha 3
linha 4
linha 5
>>>>>> login
```

```
franklin@np300e5m-xf3br: ~/auladegit
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
GNU nano 2.9.3 login2.txt Modificado

linha 99
lina 100
linha 101|
```

O GTI FACILITA AS COISAS PARA VOCÊ! :)

Resolvendo o conflito:

1. Mude para a branch **login**
2. digite **git merge master**
3. confirme que o arquivo **login2.txt** está com a versão que você escolheu quando resolveu o conflito

RECURSOS ÚTEIS DO GITHUB

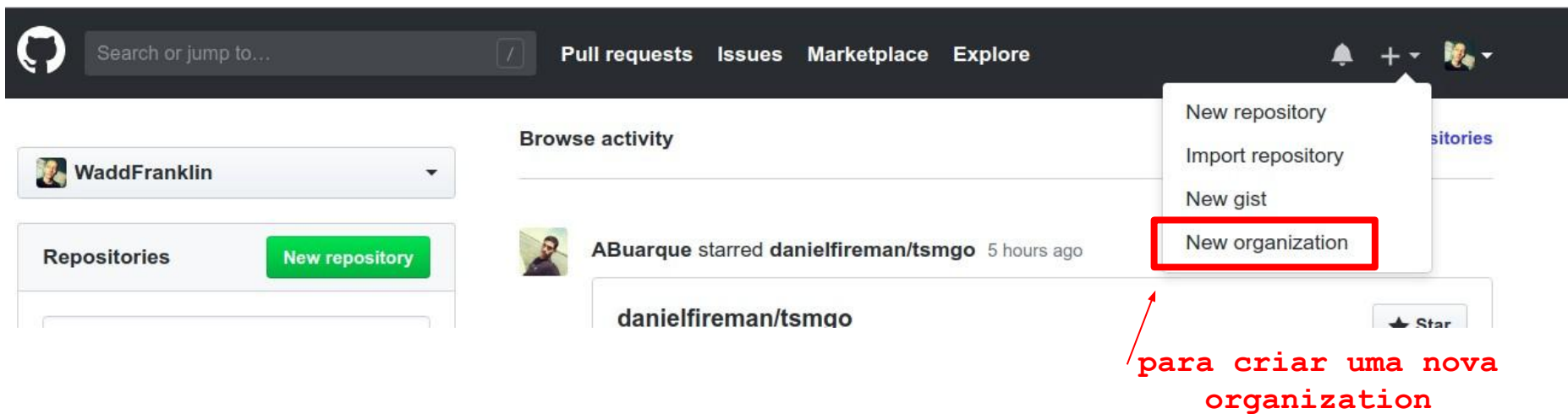
Organizations | Fork | Pull Request

ORGANIZATION

Você pode usar as organizações para criar equipes de desenvolvimento de um projeto.

O owner da organization pode convidar colaboradores para compor a equipe e contribuir com o projeto.

ORGANIZATION



The screenshot shows the GitHub homepage. At the top is a dark navigation bar with the GitHub logo, a search bar, and links for Pull requests, Issues, Marketplace, and Explore. On the left, there's a sidebar with the user's profile (WaddFranklin) and a 'New repository' button. The main content area is titled 'Browse activity' and shows a recent activity item where 'ABuarque' starred the repository 'danielfireman/tsmgo'. A dropdown menu is open from the top right, displaying options: 'New repository', 'Import repository', 'New gist', and 'New organization'. The 'New organization' option is highlighted with a red rectangle. A red arrow points from this option to the text 'para criar uma nova organization' at the bottom right.

Search or jump to...

Pull requests Issues Marketplace Explore

WaddFranklin

Repositories **New repository**

Browse activity

ABuarque starred danielfireman/tsmgo 5 hours ago

danielfireman/tsmgo

New repository
Import repository
New gist
New organization

para criar uma nova organization

ORGANIZATION



ProjetoDiscretaExcript

Repositories 1

People 2

Teams 0

Projects 0

Settings

Type: All

Language: All

Customize pinned repositories

New

Excript-2.0

Projeto apresentado na disciplina de Matemática Discreta. Time: França MacDowel, John Victor e Waddinsohn Franklin

C

2

Updated on 7 Nov 2017

Top languages

C

People

2 >



francamacdowell
França Mac Dowell



WaddFranklin
Wadd Franklin

Invite someone

FORK

Quando estamos visitando o perfil de outro usuário do github e abrimos um repositório deste usuário, é comum que os desenvolvedores desejem copiar este repositório para a sua conta.

Desta forma, é possível alterar o código, estudar e até sugerir correções de bugs ao proprietário do repositório.

Para este tipo de operação, o github possui o botão **fork**.

FORK

Com esta funcionalidade, é possível copiar ("forkar") o conteúdo de qualquer repositório público no github.

FORK

 r0drigopaes / paa

 Watch ▾

5

★ Star

12

 Fork

20

<> Code

! Issues 0

 Pull requests 1

 Projects 0

 Wiki

 Insights

Arquivos das aulas de PAA

 77 commits

 1 branch

 0 releases

 8 contributors

Branch: master ▾

New pull request

Create new file

Upload files

Find file

Clone or download ▾



Rodrigo Paes Programação Dinâmica, BFS, Componentes Conectados, Huxley 613

Latest commit 95e1b3a on 26 Apr



soma_maxima

Adicionadas as soluções em python do problema da soma máxima

a year ago



.gitignore

Add .gitignore

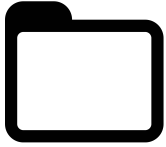
2 years ago

PULL REQUEST

PULL REQUEST



Samuel

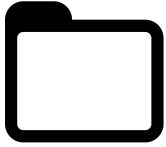


repo

PULL REQUEST



Samuel



repo

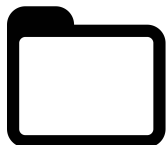


Samara

PULL REQUEST



Samuel



repo

fork!



Samara



forked

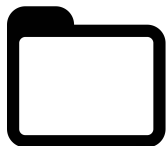
Samara agora pode
modificar o código do
repositório forkado de
repo.

Ou seja, ela possui
uma cópia de repo na
sua conta do github.

PULL REQUEST



Samuel



repo

fork!

Quanta gambiarra, meu Deus! Vou arrumar isso e mandar para o Samuel.



Samara

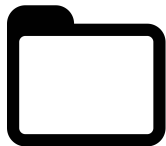


forked

PULL REQUEST



Samuel

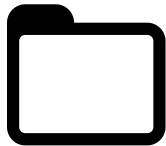


repo

Nesse momento, Samara já fez todas as melhorias que julgou necessárias e quer enviar o código refatorado para o Samuel.



Samara

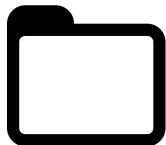


forked

PULL REQUEST



Samuel

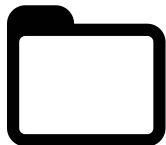


repo

Nesse momento, Samara já fez todas as melhorias que julgou necessárias e quer enviar o código refatorado para o Samuel.



Samara



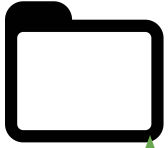
forked

Como as mudanças que Samara estão apenas no seu fork de repo, ela precisa fazer um **pull request** e Samuel deve aceitar ou recusar as mudanças.

PULL REQUEST



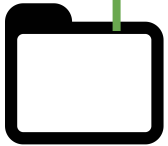
Samuel



repo



Samara



forked

pull
request

PULL REQUEST



Samuel

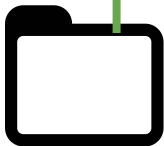
Isso que é um código de vergonha! Vou aceitar esse pull request!

repo

pull
request



Samara

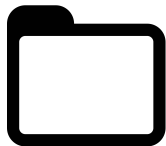


forked

PULL REQUEST



Samuel

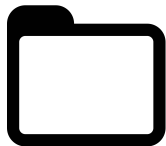


repo

Agora, como Samuel
aceitou o pull request
de Samara, todas as
modificações feitas
por ela agora estão
também no repositório
de Samuel



Samara



forked

PULL REQUEST

 r0drigopaes / paa

 Watch ▾

5

 Star

12

 Fork

20

 Code

 Issues 0

 Pull requests 1

 Projects 0

 Wiki

 Insights

Arquivos das aulas de PAA

 77 commits

 1 branch

 0 releases

 8 contributors

Branch: master ▾

New pull request

Create new file

Upload files




Find file

Clone or download ▾



Rodrigo Paes Programação Dinâmica, BFS, Componentes Conectados, Huxley 613

Latest commit 95e1b3a on 26 Apr

 soma_maxima	Adicionadas as soluções em python do problema da soma máxima	a year ago
 .gitignore	Add .gitignore	2 years ago
 BFS.cpp	Busca em grafos em C++ e Python3	a year ago

GIT GUI CLIENTS

Ferramentas git com interface gráfica



smartgit

SmartGit develop - [E:\java\smartgit\master] - SmartGit

Repository Edit View Remote Local Branch Query Changes Tools Window Help Debug

Pull Sync Push Git-Flow Merge Commit Stage Index Editor Unstage Save Stash Apply Stash Abort Discard Delete Log Blame Investigate

Repositories x Files 12,215 files hidden File Filter

- all* (smartgit-master)
- deepgit* (master)
- javal* (master)
- javal-subversion* (smartsvr)
- SmartGit develop* (develop)**
- SmartGit release* (release-1)
- SmartSynchronize* (all)
- website* (master)
- website-ruby* (syntevo)

> other

> privat

> SmartSVN

> SWT

- eclipse.platform.swt (syntevo)
- eclipse.platform.swt.binary

> test

Branches x

- develop <3 origin
- Local Branches (1)
 - all-master = origin
- origin (26) - https://
- Tags (1672)
- Stashes (6)

Ready

Name	State	Relative Directory
ScCompareInnerLineTokenComparatorTest.java	Staged	src/base/components/compare/test/com/syntevo/sc/textcompo
ScCompareInnerLineTokenComparator.java	Staged	src/base/components/compare/src/com/syntevo/sc/textcompon
ScCompareInnerLineMedia.java	Staged	src/base/components/compare/src/com/syntevo/sc/textcompon
ScCompareBlockBuildPacketTest.java	Modified	src/base/components/compare/test/com/syntevo/sc/textcompo
realWorldTooComplex1.res.1.txt	Modified	src/base/components/compare/test-resources/com/syntevo/sc/t
realWorldSkinIrrelevantSmallIdenticalAreas1.right	Untracked	src/base/components/compare/test-resources/com/syntevo/sc/t

Changes for ScCompareInnerLineTokenComparator.java - HEAD vs. Index +7-2~9

```
180         if (diffTokenIndices.contains(index)) {
181             lineDiffs[line]++;
182         }
183     }
184
185     int diffCount = 0;
186     int tokenCount = 0;
187     for (int index = 0; index < lineTokens.length; index++) {
188         final int diffs = lineDiffs[index];
189         final int tkns = lineTokens[index];
190         if (diffs == tkns ||
191             // special handling for trailing \n of
192             // without line endings and hence ScCom
193             index == lineTokens.length - 1 && diff<
194         )
195             continue;
196     }
197
198     for (int index = 0; index < lineTokens.length; index++) {
199         for (int index = minLine; index <= maxLine; index++) {
200             break;
201             maxLine--;
202         }
203     }
204
205     int diffCount = 0;
206     int tokenCount = 0;
207     for (int index = minLine; index <= maxLine; index++) {
208         final int diffs = lineDiffs[index];
209         final int tkns = lineTokens[index];
210         diffCount += diffs;
211         tokenCount += tkns;
212     }
213
214     if (tokenCount < THRESHOLD_MIN_TOKENS) {
215         return Quality.depends_on_counterpart;
216     }
217 }
```

Journal x

- origin/origin/all-master SG-11275: Compare: improved inner-changes diff (relaxed "complexity" condition) Thomas Singer 2018-02-02 09:04
- SG-11275: Compare: improved inner-changes diff (treat small, "irrelevant" areas of similarity as changed) 2018-02-02 09:04
- SG-11275: Compare: improved inner-changes diff (no-white-space block shifting should already honor line star... 2018-02-02 09:03
- develop ScCompareInnerLineTokenComparator: debug output improved 2018-02-02 09:02
- SG-11275: Compare: improved inner-changes diff (tightened "complexity" condition which prevents inline cha... 2018-02-01 19:12
- SG-11460: Compare: inner-line changes not showing up where they should (2) 2018-02-01 19:11
- SG-11461: Compare: displayed inner-line changes may be outdated (e.g. after switching options) 2018-02-01 19:10
- SG-11275: Compare: improved inner-changes diff (no more camel-hump tokenizing) 2018-02-01 19:05
- ScLicensePurchaseConfiguration -> ScLicensePurchaseConfigProvider 2018-02-01 19:04



GitKraken

libgit2 index_dirty

Undo Redo Pull Push Branch Stash Pop

Viewing 466/466 Show All Filter (⌘ + Option + f)

LOCAL 2/2

- ethomson
- index_dirty**
- master 10

REMOTE 406/406

Libgit2

- appveyor
- bindings
 - libgit2sharp
 - 020_2
 - 022_1
- brianmario
 - attr-from-tree
 - revwalk-filter
 - trailer-info
 - trailer-list
- bug
 - status_case
- clay-test
- cmn
 - atexit-skeleton
 - auth-retry
 - bump-pretend-git
 - cancellation
 - commit-on
 - commit-to-memory
 - commit-with-signature

master h h gh-pages

Merge pull req... 12 hours ago
generated docs
openssl: remove leftover #if...
Merge pull request #4640 fr...
Merge pull req... 4 days ago
worktree: add fu... 5 days ago
tests: fix issue with /tmp pat...
Merge pull req... 6 days ago
Merge pull request #4525 fr...
Merge pull request #4580 fr...
Merge pull req... a week ago
Merge pull request #4635 fr...
Merge pull request #4577 fr...
tests: free the worktree in a...
refspec: check for valid para...
remote: repo is optional here
worktree: fix calloc of the w...
local: fix a leaking reference...
Fix deletion of unrelated br...
Merge pull request #4631 fr...
fixed stack smashing due to...
index: commit the changes ...
index: test dirty index bit
index: add a dirty bit reflecti...
stash: use _an_ index not _t...
checkout: always set the ind...

commit: 934486

index: commit the changes to the index properly

Now that the index has a "dirty" state, where it has changes th at have

Edward Thomson authored 11/16/2017 @ 5:19 PM parent: 51c2aa

10 modified

Name | Path | Tree | View all files

- src/checkout.c
- src/index.c
- src/index.h
- src/status.c
- tests/checkout/tree.c
- tests/checkout/typechange.c
- tests/cherrypick/workdir.c
- tests/index/names.c
- tests/index/reuc.c
- tests/rebase/submodule.c

100% Feedback PRO

CURSO DE GIT E GITHUB ONLINE GRÁTIS

<https://www.schoolofnet.com/curso-git-e-github/>

OBRIGADO!