

Workshop Git/GitHub

Tópicos

- O que é git?
 - Repositório
 - Branch
 - Commit
- Ciclo de Trabalho prático(clone, init, pull, push, commit)
- Funcionalidades
 - Stash
 - Conflito
 - Merge
 - Pull Request
- Prática 2

Setup de ambiente

Git Terminal

C:\user> git

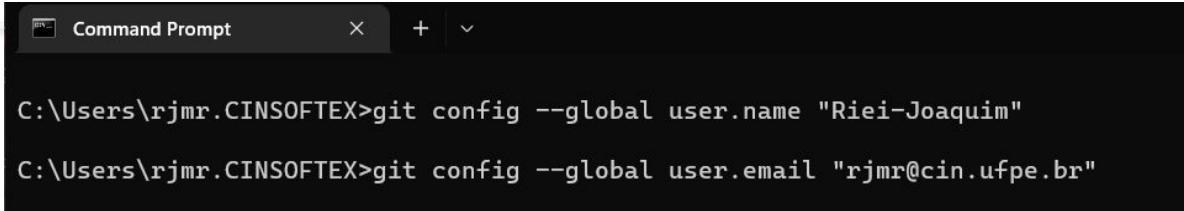
```
Command Prompt + | ^ Microsoft Windows [Version 10.0.22621.1105] (c) Microsoft Corporation. All rights reserved. C:\Users\rjmr.CINSOFTEX>git usage: git [--version] [--help] [-C <path>] [-c <name>=<value>] [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path] [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare] [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>] [--super-prefix=<path>] [--config-env=<name>=<envvar>] <command> [<args>] These are common Git commands used in various situations: start a working area (see also: git help tutorial) clone     Clone a repository into a new directory init      Create an empty Git repository or reinitialize an existing one work on the current change (see also: git help everyday) add       Add file contents to the index mv        Move or rename a file, a directory, or a symlink restore   Restore working tree files rm        Remove files from the working tree and from the index
```

Nota: Verificar git no path (se retornar erro avisem)

Configuração de Usuário

```
C:\user> git config --global user.name "seu nome"
```

```
C:\user> >git config --global user.email "seu_email@example.com"
```



```
Command Prompt

C:\Users\rjmr.CINSOFTEX>git config --global user.name "Riei-Joaquim"
C:\Users\rjmr.CINSOFTEX>git config --global user.email "rjmr@cin.ufpe.br"
```

Configuração de Usuário

```
C:\user> git config --list
```

```
C:\Users\rjmr.CINSOFTEX>git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Riei-Joaquim
user.email=rjmr@cin.ufpe.br
```

Credenciamento do Terminal

```
C:\user> ssh-keygen -t ed25519 -C "your_email@example.com"
```

```
C:\Users\rjmr.CINSOFTEX>ssh-keygen -t ed25519 -C "your_email@example.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\rjmr.CINSOFTEX/.ssh/id_ed25519):
C:\Users\rjmr.CINSOFTEX>ssh-keygen -t ed25519 -C "rjmr@cin.ufpe.br"
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\rjmr.CINSOFTEX/.ssh/id_ed25519):
Created directory 'C:\\\\Users\\\\rjmr.CINSOFTEX/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\rjmr.CINSOFTEX/.ssh/id_ed25519
Your public key has been saved in C:\Users\rjmr.CINSOFTEX/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:+BMLWHYTmkZSpXFH0Zv5zltHtK+DeapdYbe801R8MsY rjmr@cin.ufpe.br
The key's randomart image is:
+--[ED25519 256]--+
| ..+.+.+o |
| o * o . |
| B o = ... |
| = o . + E.= |
| . o S oo+= |
| o o .o=o |
| + =oo+ |
| . .o.=+o |
| ..ooo=o |
+---[SHA256]---
```

Nota: dê enter para deixar path default e sem senha

Credenciamento do Terminal

```
C:\user> start-ssh-agent.cmd
```

```
C:\Users\rjmr.CINSOFTTEX>start-ssh-agent.cmd
Removing old ssh-agent sockets
Starting ssh-agent: done
Identity added: /c/Users/rjmr.CINSOFTTEX/.ssh/id_ed25519 (rjmr@cin.ufpe.br)
```

```
C:\user> ssh-add
```

```
C:\Users\rjmr.CINSOFTTEX>ssh-add
Identity added: C:\Users\rjmr.CINSOFTTEX/.ssh/id_ed25519 (rjmr@cin.ufpe.br)
```

```
C:\user> clip < .ssh/id_ed25519.pub
```

```
C:\Users\rjmr.CINSOFTTEX>clip < .ssh/id_ed25519.pub
```

Credenciamento do Terminal(2^a opção)

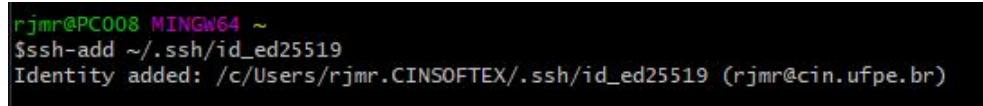
Com o git bash aberto

```
C:\user> eval "$(ssh-agent -s)"
```



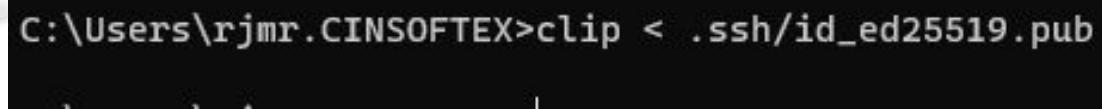
```
rjmr@PC008 MINGW64 ~
$ eval "$(ssh-agent -s)"
```

```
C:\user> ssh-add ~/.ssh/id_ed25519
```



```
rjmr@PC008 MINGW64 ~
$ ssh-add ~/.ssh/id_ed25519
Identity added: /c/Users/rjmr.CINSOFTEX/.ssh/id_ed25519 (rjmr@cin.ufpe.br)
```

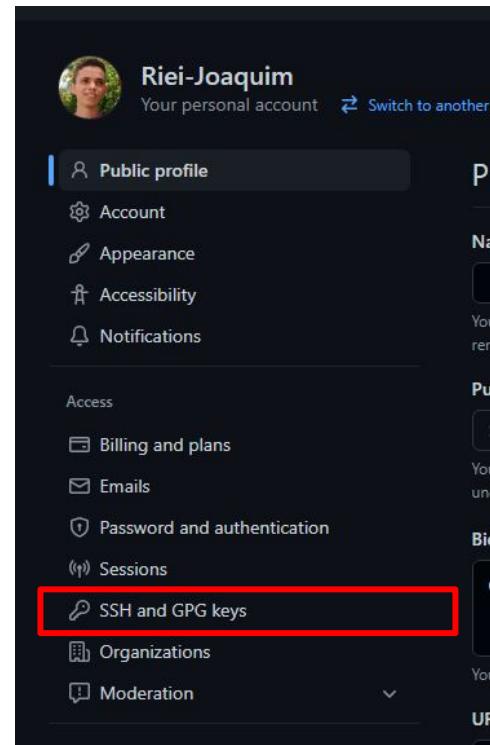
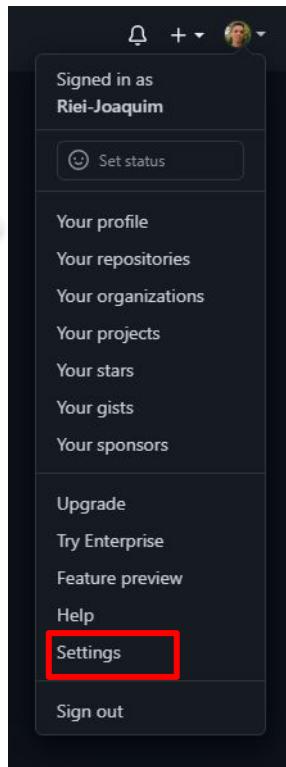
```
C:\user> clip < .ssh/id_ed25519.pub
```



```
C:\Users\rjmr.CINSOFTEX>clip < .ssh/id_ed25519.pub
```

Adicionar chave ao github

Acesse: <https://github.com/>



Adicionar chave ao github

Acesse: <https://github.com/>

The screenshot shows the GitHub 'SSH keys / Add new' page. At the top right is a green 'New SSH key' button with a red border. Below it, the 'Title' field contains the placeholder 'your Title' with a red border. The 'Key type' dropdown is set to 'Authentication Key'. The 'Key' field contains a long string of characters starting with 'ssh-ed25519 AAAAC3NzaC1IzDI1NTE5AAAAIfscSue7MsZy6+SDFSDFSDDfdfsvewJ/pt+7xQsdAWFFSAEF' and ends with 'you_email@example.com', also with a red border. At the bottom left is a green 'Add SSH key' button with a red border.

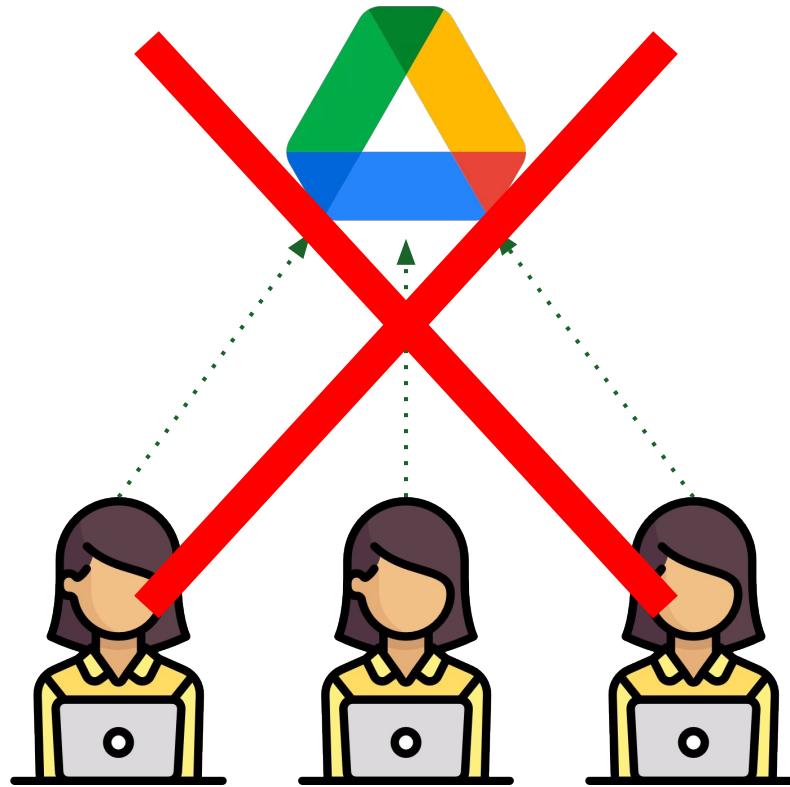
Adicionar chave ao github

```
C:\user> ssh -T git@github.com
```

```
C:\Users\rjmr.CINSOFTEX>ssh -T git@github.com
The authenticity of host 'github.com (20.201.28.151)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Hi Riei-Joaquim! You've successfully authenticated, but GitHub does not provide shell access.
```

Conceitos Básicos

O que é Git?



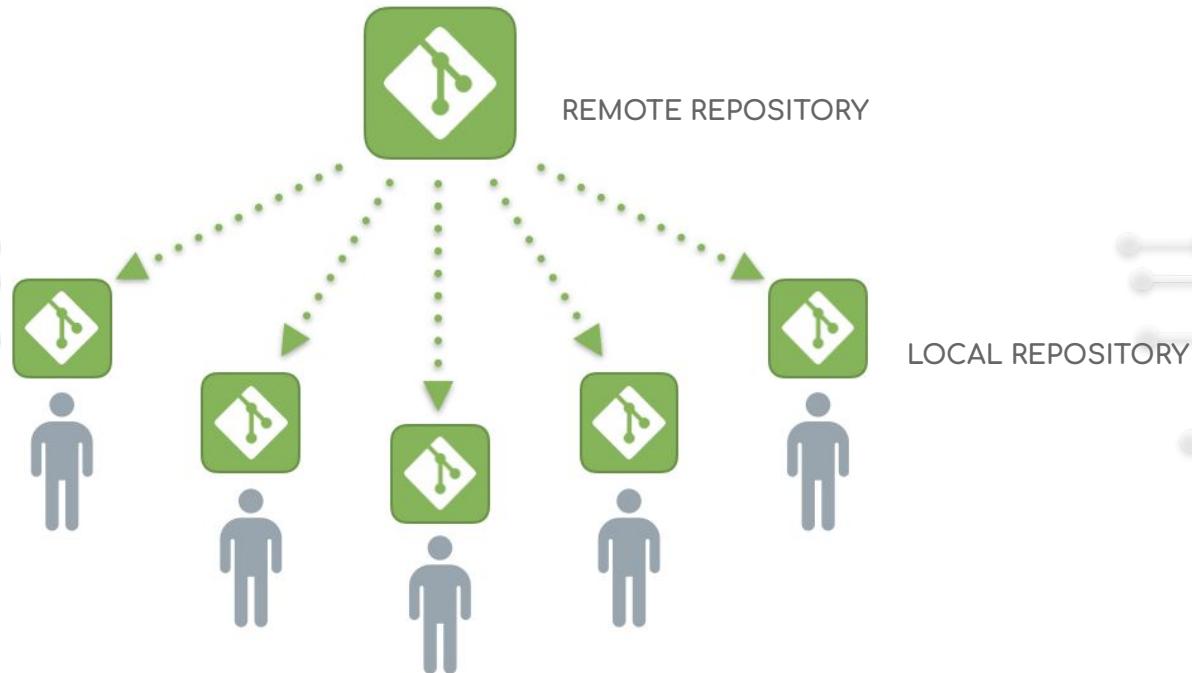
O que é Git?

“O Git é um sistema de controle de versão, projetado basicamente para facilitar a vida de quem quer executar projetos em equipe, permitindo que duas ou mais pessoas trabalhem juntas.”

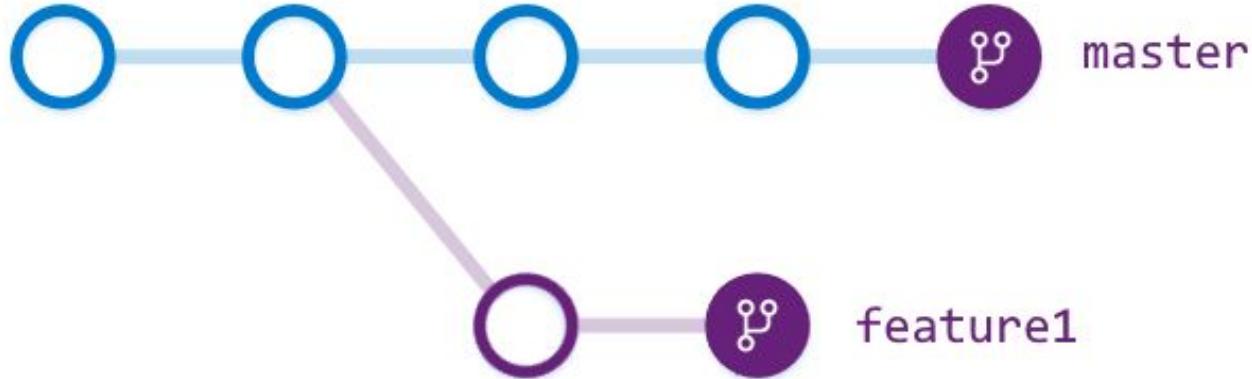
“Um sistema de controle de versão faz o papel de “juntar” as partes do projeto, de modo que, cada membro da equipe faça uma parte, e, utilizando este sistema seja possível juntar tudo no final.”

Fonte: <http://www.contagia.com.br/blog/git-pra-que-serv%C3%A9/>

Repositório

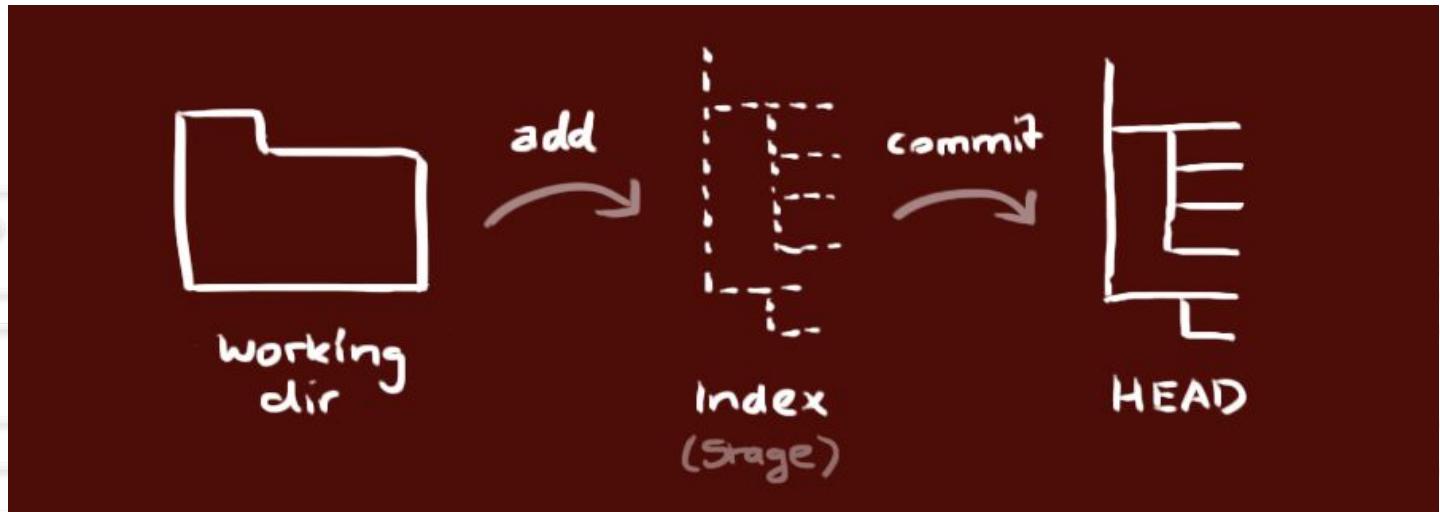


Branch

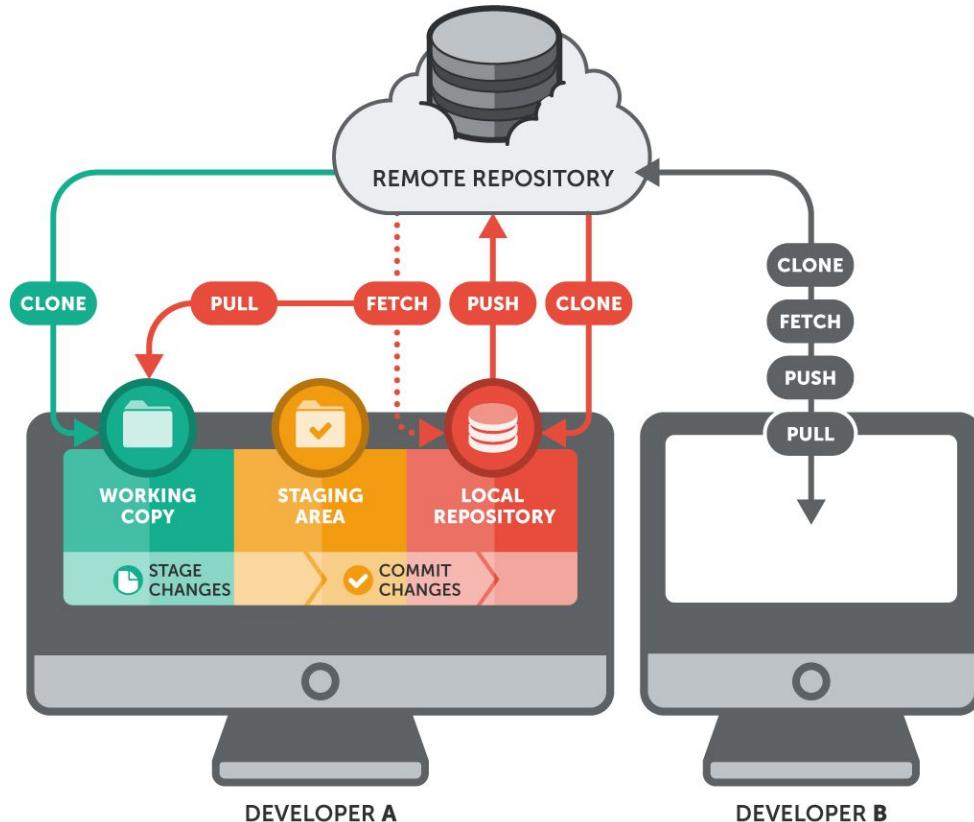


- A branch main é um porto seguro de software.
- Desenvolver diretamente na branch main reduz drasticamente as funcionalidades do Git.

Commit



Ciclo de Trabalho



Ciclo de Trabalho

- **Clone:** Baixar repositório remoto para o seu PC, criando o repositório local.
C:\user> git clone [link]
- **Pull:** Sincronizar repositório local com a versão remota mais recente.
C:\user\pasta_base_do_repo> git pull
- **Status:** Verificar mudanças não mapeadas no repositório local.
C:\user\pasta_base_repo> git status

Git Status

```
C:\Users\rjmr.CINSOFTEX\ssl-unification>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 restore <file>..." to discard changes in working directory)
      modified:   src/Modules/Modules.h

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

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

Ciclo de Trabalho

- **Add:** Mapear mudanças para serem indexadas
C:\user> git add .
C:\user> git add <caminho relativo para o arquivo>
- **Pull:** Sincronizar repositório local com a versão remota mais recente.
C:\user\pasta_base_do_repo> git pull
- **Commit:** Indexa e empacota as modificações localmente.
C:\user\pasta_base_do_repo> git commit -m "you message"
- **Push:** Pública os commits locais no repositório remoto.
C:\user\pasta_base_do_repo> git push
C:\user\pasta_base_do_repo> git push -u origin <nome-da-branch>

Ciclo de Trabalho

- Fetch: Sincroniza branches remotas disponíveis.
C:\user\pasta_base_do_repo> git fetch
- Checkout: Troca a branch atual e atualiza o repositório local com as suas mudanças.
C:\user\pasta_base_do_repo> git checkout <nome-da-branch>
C:\user\pasta_base_do_repo> git checkout -b <nome-da-nova-branch>
- Reset: Reset das mudanças locais, voltando para o último estado sincronizado com o repositório remoto.
C:\user\pasta_base_do_repo> git reset --hard

Atividade prática

Atividade prática 1

1. Clonar o repositório WorkshopGit;
2. Abrir o repositório no VScode e verificar o README.md;
3. Criar e publicar uma branch a partir da main com o seu nome;
4. Adicionar arquivo .txt e preencher dados pessoais;
5. Commitar na branch com seu nome;
6. Voltar na main;
7. Dar fetch e ver as outras branches;

1. git clone
<git@github.com:Curso-de-Robotica-e-IA/WorkshopGit.git>
2. cd WorkshopGit
 - a. git status
 - b. code .
3. git checkout -b seuNome
 - a. git push -u origin seuNome
4. (VScode) direito do mouse, selecionar New File
5. git add .
 - a. git commit -m "you commit message"
 - b. git push
6. git checkout main
7. git branch
 - a. git fetch
 - b. git branch

Funcionalidades

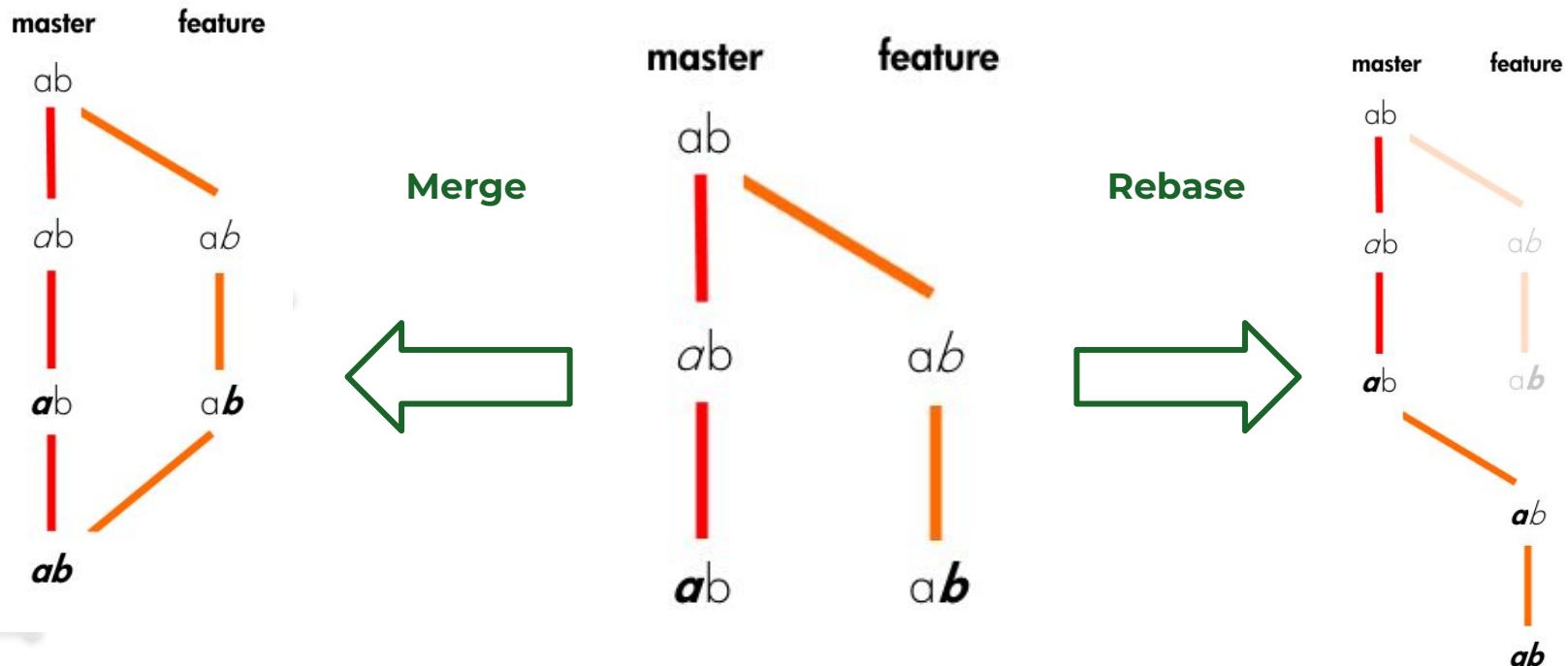
Stash

- Armazenamento temporário de mudanças locais, “guardar para mais tarde”.
- Trocas de branch só podem ocorrer quando não há mudanças locais sem serem mapeadas.

Stash

- **Save:** Empilha modificações não comitadas com um nome
C:\user\pasta_base_do_repo> git stash save “title”
- **List:** Lista os stashes locais no repositório, com os títulos e indexes
C:\user\pasta_base_do_repo> git stash list
- **Pop:** Desempilha modificações guardada pelo indexe
C:\user\pasta_base_do_repo> git stash pop index
- **Apply:** Aplica as modificações guardadas ao repositório sem desempilhar
C:\user\pasta_base_do_repo> git stash apply index
- **Drop:** Deleta modificações guardada pelo indexe
C:\user\pasta_base_do_repo> git stash drop index

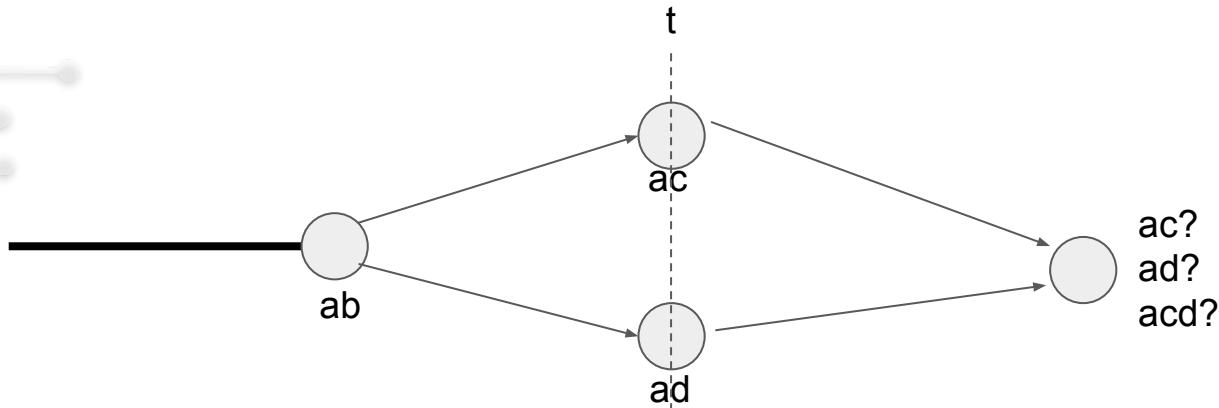
Integração de branches

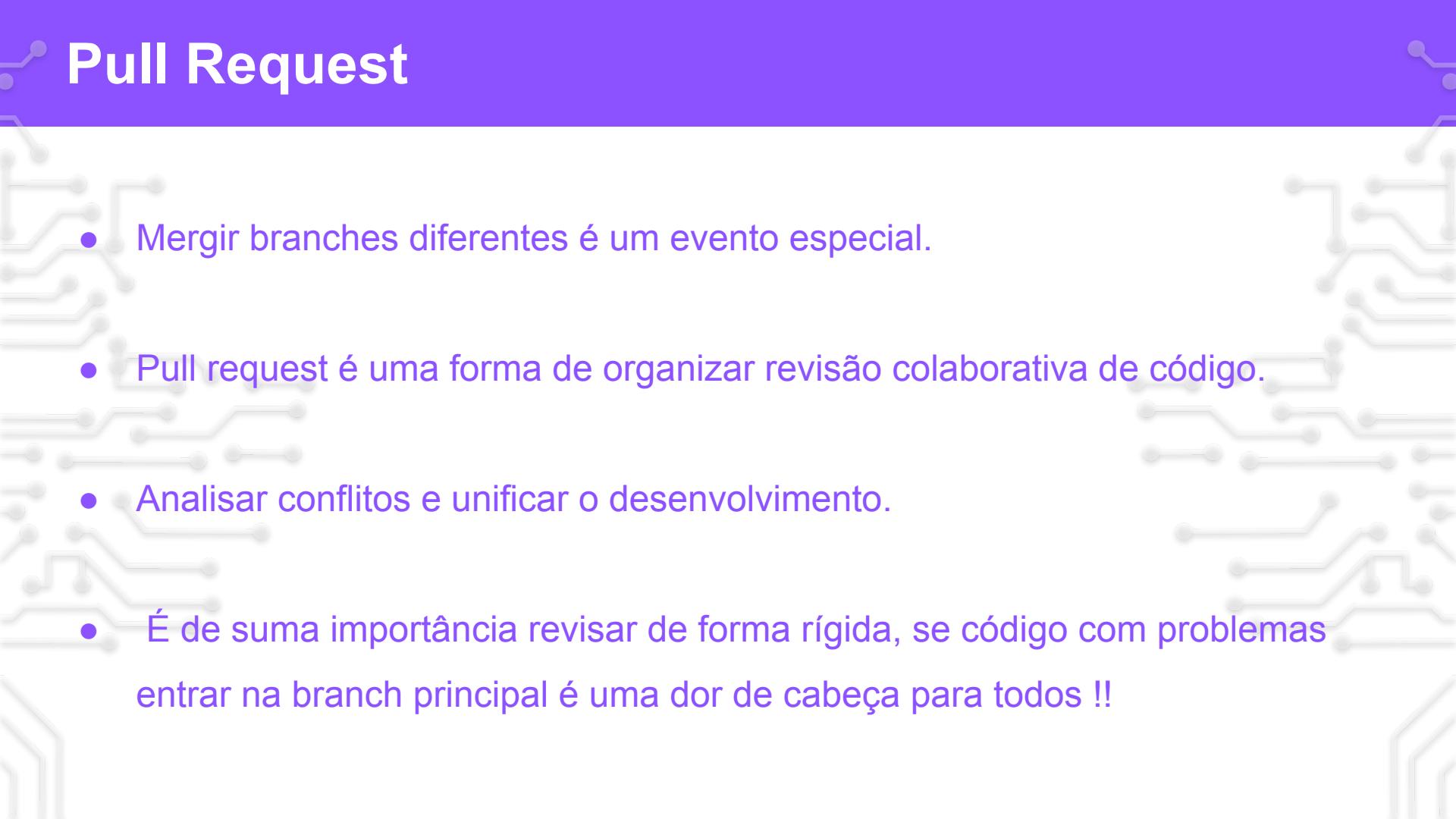


- Merge: Cria um commit mesclando a branch base na destino.
- Rebase: Replica os commits da branch base na destino.

Conflito

- Ocorre quando não é possível unificar de forma automática o histórico de mudanças.
- Necessita que o desenvolvedor saiba quais mudanças realmente devem estar na versão final.
- Edição manual dos arquivos para unificar a versão final.

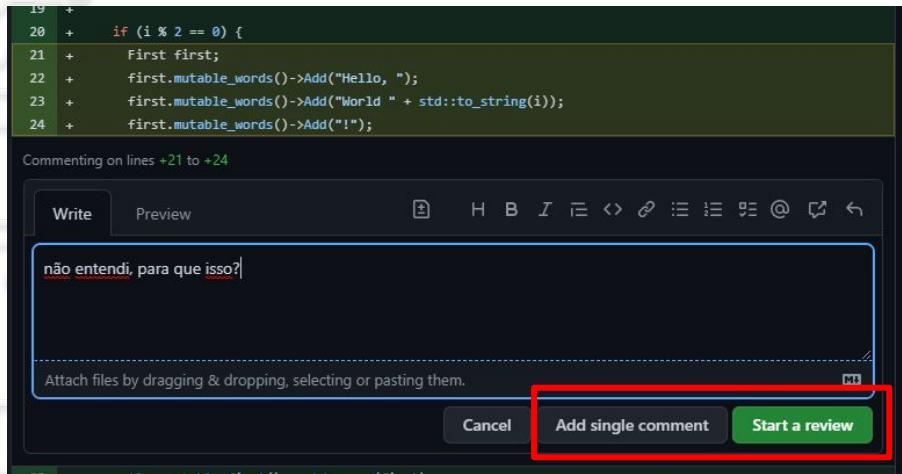




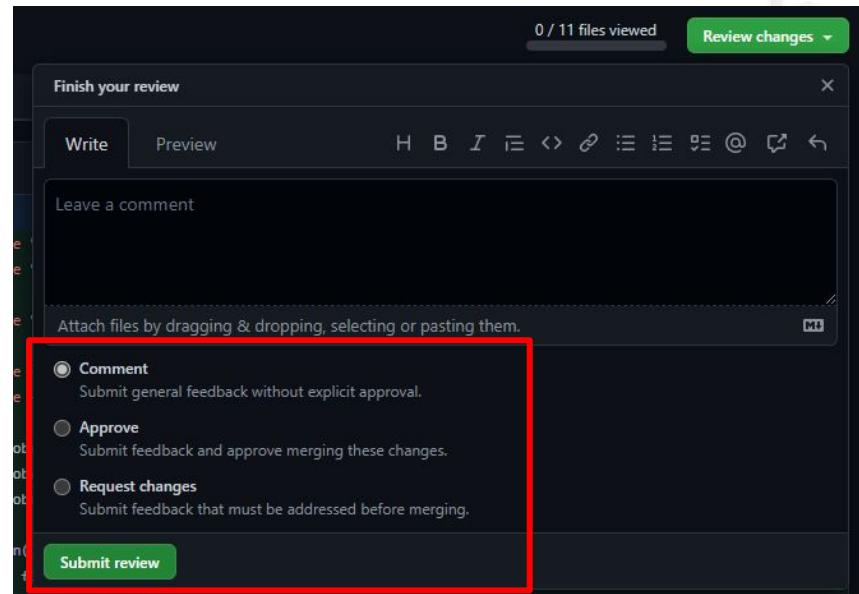
Pull Request

- Mergir branches diferentes é um evento especial.
- Pull request é uma forma de organizar revisão colaborativa de código.
- Analisar conflitos e unificar o desenvolvimento.
- É de suma importância revisar de forma rígida, se código com problemas entrar na branch principal é uma dor de cabeça para todos !!

Processo de Revisão



A screenshot of a GitHub pull request review interface. On the left, there's a code diff showing additions to lines 19 through 24. The right side shows a 'Write' tab where a comment has been typed: 'não entendi, para que isso?'. Below the comment area is a red box highlighting the 'Add single comment' and 'Start a review' buttons.



- Processo feito pelo site do github

Atividade prática 1 (continuação)

Ainda com o repositório WorkshopGit (fim do caminho feliz):

1. Pelo web browser abrir o [github.com](https://github.com/Curso-de-Robotica-e-IA/WorkshopGit), no repositório workshopGit:
<https://github.com/Curso-de-Robotica-e-IA/WorkshopGit>
2. Abrir o Pull Request e colocar o amigo do lado como revisor;
3. O seu amigo ao lado deve olhar as modificações e aprovar as modificações pelo navegador.
4. Estando aprovado o pull request basta dar merge, mas não delete a branch mergida ainda.

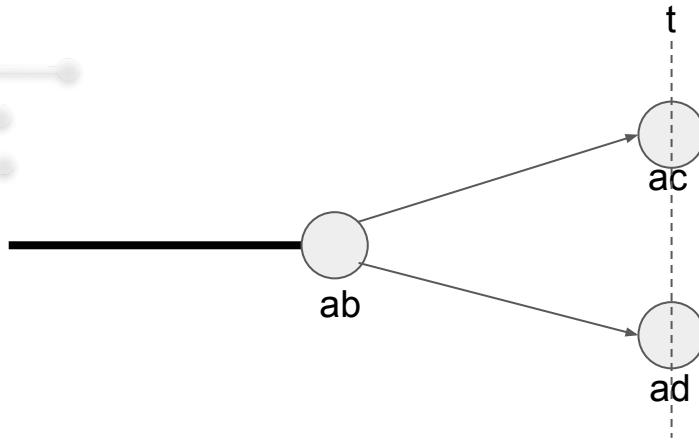
Atividade prática 2

1. Retorne para a branch main e sincronize com seu estado remoto.
2. Delete sua branch nos repositórios local e remota.
3. Crie e publique duas branches com seu nome

1. git checkout main
 - a. git pull
2. git branch -D seuNome
 - a. git push origin --delete seuNome
3. git checkout -b seuNome
 - a. git push -u origin seuNome
4. git checkout -b seuNome2
 - b. git push -u origin seuNome2

Atividade prática 2

1. Utilizando as branches criadas faça e publique no repositório remoto:
 - a. Em uma delas delete algumas linhas do README.md e adicione mais informações no .txt adicionado por você e na outra branch adicione informações no README.md e modifique as informações presentes no .txt adicionado por você;



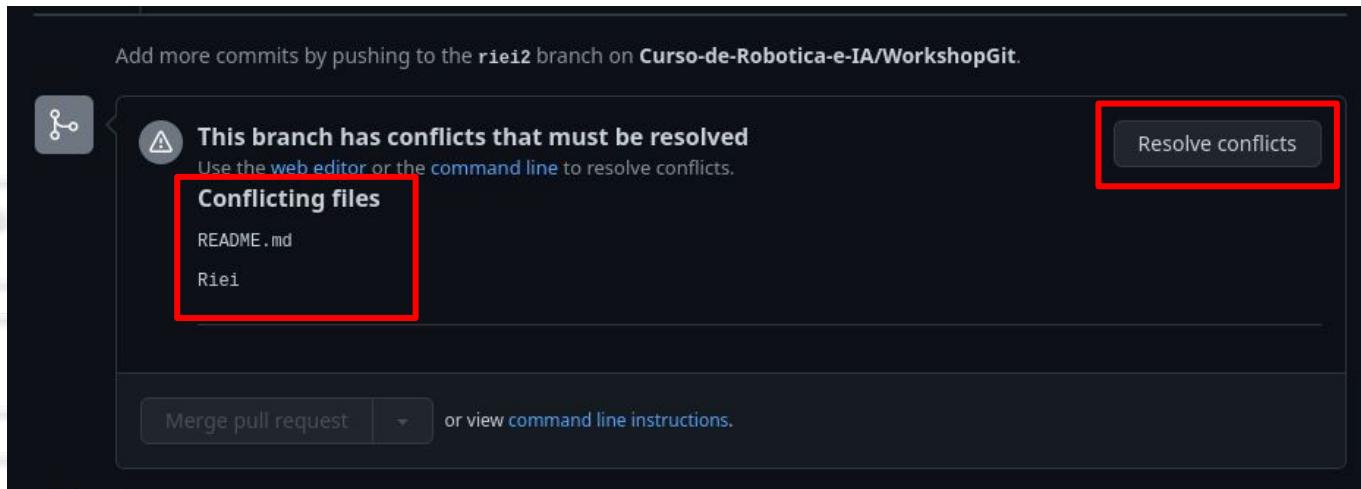
Atividade prática 2

The screenshot shows a GitHub repository named "Curso-de-Robotica-e-IA / WorkshopGit" which is private. The repository has 1 watch, 0 forks, and 0 stars. The main navigation bar includes links for Code, Issues, Pull requests (which is the active tab), Actions, Projects, Security, Insights, and more. Below the navigation bar are filters (is:pr is:open), labels (9), milestones (0), and a prominent green button labeled "New pull request".

A modal window titled "Comparing changes" is open. It displays a comparison between two branches: "base: riei" and "compare: riei2". A message indicates that "Can't automatically merge" and provides a link to learn about pull requests. At the bottom of the modal is a green button labeled "Create pull request".

Abrir o Pull request da branch seuNome2 para a branch seuNome;

Atividade prática 2



Resolver os conflitos

Atividade prática 2

Resolving conflicts between riei2 and riei and committing changes → riei2

2 conflicting files

README.md

1 conflict

Prev ^

Next v



Mark as resolved

README.md
README.md

Riei
Riei

```
1 <<<< riei2
2 dgfgdfgfdgdfg
3 =====
4 # wewfwf
5 >>>> riei
6
```

2 conflicting files

README.md

✓ Resolved

README.md

```
1 dgfgdfgfdgdfg
2
```

Resolving conflicts between riei2 and riei and committing changes → riei2

Commit merge

Resolver os conflitos

2 conflicting files

RIEI

✓ Resolved

README.md



README.md



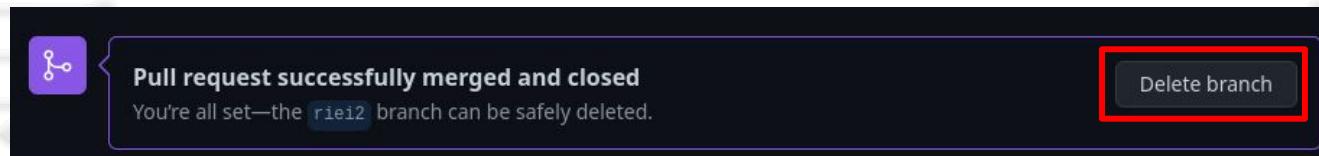
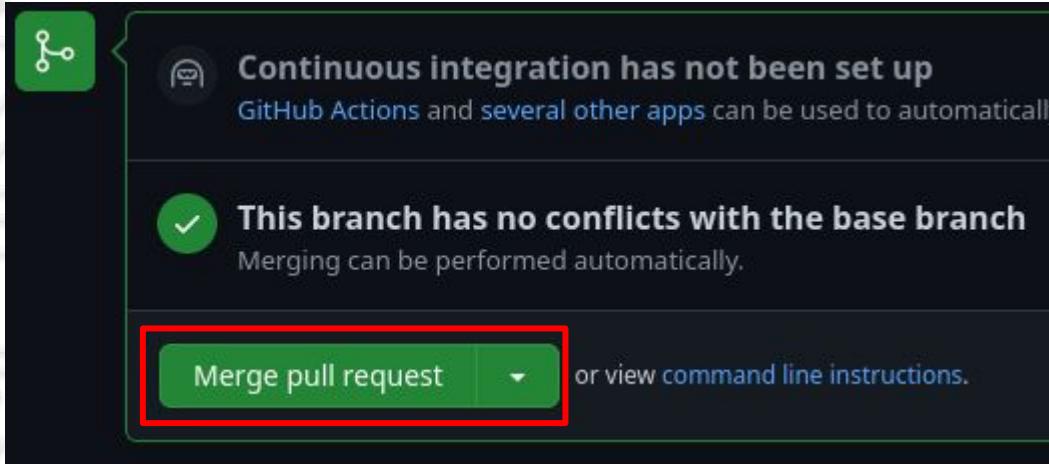
Riei



Riei

```
1 psjdpfojsodhgpsfjbpbvsif
2 adadasdssfsqdfnhgjkuyvgs
3 zxczxczx
4
```

Atividade prática 2



Integrar o Pull Request

GitHub desktop

- Baixar através do Link:
 - <https://desktop.github.com/>

Coisas para fazer no GitHub desktop

1. Logar
2. Clonar repositório
3. Commit
4. Pull/Push
5. Pull Request
6. Stash
7. Conflito
8. Merge

Recomendações

1. Usar o repositório pessoal dentro da organização (solicitação enviada por email);
2. Obrigatório commitar ao fim do dia;
3. Criar uma pasta por atividade;
4. O repositório será usado para avaliação.
5. Crie um repositório pessoal para as suas atividades de onboarding
<https://classroom.github.com/a/S-r5S78a>

=)

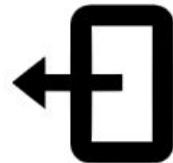
In case of fire



1. git commit



2. git push



3. leave building

Referências

- http://rogerdudler.github.io/git-guide/index.pt_BR.html
- <http://www.contagia.com.br/blog/git-pra-que-serve/>
- <https://meteo.unican.es/trac/wiki/versionControl/git>
- <https://gist.github.com/dirkdunn/2ff2784e8c780f0f7c2cc2a909fc299c>
- <https://hackernoon.com/git-merge-vs-rebase-whats-the-diff-76413c117333>
- <https://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/>



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

