

Este trabalho consiste na elaboração de um dicionário técnico dos códigos dos códigos do GIT, o programa utilizado foi o Git Bash. Para a separação dos códigos foram colocados em grupos que apresentam uma relação logica e dentro destes grupos seguem uma ordem alfabética.

Toda a pesquisa foi baseada nos 6 primeiros vídeos do curso gratuito, Curso prático GIT e GITHUB, disponibilizado no *youtube* pelo professor José de Assis Filho.

## 1. Navegação Git Bash.

- 1.1. *cd "pasta"*: vai até o diretório escolhido (CURSO..., 2018).
- 1.2. *cd*: volta para o diretório do usuario (CURSO..., 2018).
- 1.3. *clear*: limpa os dados do terminal (CURSO..., 2018).
- 1.4. *CTRL+scroll do mouse*: aumenta o tamanho da fonte do terminal (CURSO..., 2018).
- 1.5. *ls -a*: lista todos os elementos ocultos dentro do diretório (CURSO..., 2018).
- 1.6. *ls*: lista todos os elementos dentro do diretório (CURSO..., 2018).
- 1.7. *pwd*: retorna o endereço do diretório atual (CURSO..., 2018).

## 2. Manipulação Git Bash.

- 2.1. *mkdir "exemplo"*: cria um novo diretório dentro do diretório atual (CURSO..., 2018).
- 2.2. *touch "exemplo.txt"*: cria um novo arquivo dentro do diretório atual (CURSO..., 2018).

## 3. Configuração do Git Bash.

- 3.1. *core .* : abre o programa de edição que foi escolhido (CURSO..., 2019).
- 3.2. *git config core.editor*: retorna o programa de edição que foi definido (CURSO..., 2019).
- 3.3. *git config --global core.editor "caminho do seu editor.exe"*: define o programa de edição por meio do seu endereço no computador (CURSO..., 2019).
- 3.4. *git config --global user.email "correio eletrônico"*: define o *e-mail* do usuario (CURSO..., 2019).
- 3.5. *git config --global user.name "nome"*: define o nome do usuario (CURSO..., 2019).
- 3.6. *git config user.email*: retorna o *e-mail* que foi definido (CURSO..., 2019).
- 3.7. *git config user.name*: retorna o nome que foi definido (CURSO..., 2019).

#### 4. Repositório Local Git Bash.

- 4.1. *(git add .)* ou *(git add \*)*: adiciona todos os arquivos novos ou modificados, do diretório, em um “container” (CURSO..., 2019).
- 4.2. *git add “arquivo”*: adiciona o arquivo novo ou modificado, do diretório, num “container” (CURSO..., 2019).
- 4.3. *git commit -am “comentário”*: faz a adição ao container e faz um novo commit (CURSO..., 2019).
- 4.4. *git commit -m “comentário”*: cria uma cópia dos arquivos do diretório, adicionados ao “container”, dentro do repositório (CURSO..., 2019).
- 4.5. *git diff*: indica as mudanças que foram feitas em um arquivo que ainda não foi atualizado no repositório (CURSO..., 2019).
- 4.6. *git init*: cria um novo repositório “.git” no diretório atual (CURSO..., 2019).
- 4.7. *git log --oneline*: retorna um histórico de alterações do repositório de forma simplificada (CURSO..., 2019).
- 4.8. *git log*: retorna um histórico de alterações do repositório com os dados do usuário de forma detalhada (CURSO..., 2019).
- 4.9. *git reset HEAD “nome do arquivo”*: retira o arquivo armazenado no container (CURSO..., 2019).
- 4.10. *git reset HEAD*: retira todos os dados armazenados no container (CURSO..., 2019).
- 4.11. *git status*: identifica o que está armazenado dentro do “container” (CURSO..., 2019).

## 5. Manipulação de projeto Git Bash

- 5.1. *git branch*: indica os ramos do projeto e a localização do HEAD sendo o nodo atual que está sendo usado, caso este tenha sido deslocado para um nodo anterior indica seu ID (CURSO..., 2019).
- 5.2. *git checkout "ID do commit anterior"*: pega todos os dados dos de versões anterior, presentes no nodo indicado, carregando no diretório, desta forma podendo recuperar uma versão anterior do projeto sem modificar as novas (CURSO..., 2019).
- 5.3. *git checkout "nome do arquivo"*: pega os dados dentro do nodo, deste arquivo, e os coloca dentro do diretório, então caso tenha perdido os dados do arquivo no diretório podem ser recuperados por meio do repositório (CURSO..., 2019).
- 5.4. *git checkout "nome do ramo"*: diferente do código 5.2. que direcionava para um nodo anterior, este irá apontar para o final da ramificação que você indicar, salvando assim os próximos commits nela (CURSO..., 2019).
- 5.5. *git checkout -b "nome da nova ramificação"*: cria uma ramificação do projeto com o nome atribuído, após sua criação todos os novos commit serão salvos nele por padrão. esta ramificação herda todos os dados do nodo anterior (CURSO..., 2019).
- 5.6. *git log --graph*: retorna uma forma descritiva dos nodos em forma de grafo, detalhado (CURSO..., 2019).
- 5.7. *git log --oneline --graph*: retorna uma forma descritiva dos nodos em forma de grafo, simplificado (CURSO..., 2019).
- 5.8. *git reset --hard "ID do nodo"*: este comando é usado para retornar a uma versão anterior do projeto e removendo as mais recentes (CURSO..., 2019).

## 6. Git Bash → GitHub

6.1. *git remote*: retorna se existe algum repositório remoto (CURSO..., 2019).

6.2. *git remote -v* : retorna detalhes do repositório remoto (CURSO..., 2019).

Após ser feito o cadastro no GitHub é fazer a criação de um novo repositório no site, podemos por meio do HTTPS passar os dados do repositório local para o remoto. O próprio site indica como fazer isso.

6.3. *git branch -m “novo nome”*: este comando redefine o nome do nodo em que se está trabalhando (CURSO..., 2019).

6.4. *git push -u origin “nome do repositório”*: o comando envia todos os dados do repositório local para o repositório remoto que foi registrado (CURSO..., 2019).

6.5. *git remote add origin “endereço do repositório remoto”*: cria uma ligação entre o repositório local e o remoto (CURSO..., 2019).

```
git remote add origin https://github.com/jorgebandeo/Dicionario-tecnico.git
```

```
git branch -m main
```

```
git push -u origin main
```

- **Referencias**

- CURSO GIT e GITHUB - Como instalar e configurar o GIT no windows. Direção: José de Assis Filho. Produção: José de Assis Filho. Youtube: [s. n.], 2018. Disponível em: <https://www.youtube.com/watch?v=SOxafinthys&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjglj&index=2>. Acesso em: 1 out. 2021.
- CURSO GIT e GITHUB - Comandos básicos (init status add commit log). Direção: José de Assis Filho. Produção: José de Assis Filho. Youtube: [s. n.], 2019. Disponível em: <https://www.youtube.com/watch?v=v8-4rl0cJXE&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjglj&index=3>. Acesso em: 1 out. 2021.
- CURSO GIT e GITHUB - Rastreando e recuperando versões anteriores do projeto (checkout). Direção: José de Assis Filho. Produção: José de Assis Filho. Youtube: [s. n.], 2019. Disponível em: [https://www.youtube.com/watch?v=\\_mB-TShMDvY&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjglj&index=4](https://www.youtube.com/watch?v=_mB-TShMDvY&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjglj&index=4). Acesso em: 1 out. 2021.
- CURSO GIT e GITHUB - Criando ramificações do projeto (branch e merge) e resolução de conflitos. Direção: José de Assis Filho. Produção: José de Assis Filho. Youtube: [s. n.], 2019. Disponível em: <https://www.youtube.com/watch?v=iRs6sQOPcvg&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjglj&index=5>. Acesso em: 1 out. 2021.
- CURSO GIT e GITHUB - Enviando um projeto local para o GITHUB. Direção: José de Assis Filho. Produção: José de Assis Filho. Youtube: [s. n.], 2019. Disponível em: <https://www.youtube.com/watch?v=j2exng2k3z4&list=PLbEOwbQR9lqzK14I7OOeREEIE4k6rjglj&index=6>. Acesso em: 1 out. 2021.

Link dos Repositórios:

[https://github.com/bernaction/Aula\\_Mobile](https://github.com/bernaction/Aula_Mobile)

[https://github.com/jorgebando/Aula\\_Mobile](https://github.com/jorgebando/Aula_Mobile)

[https://github.com/felipehoffmeister/Aula\\_Mobile](https://github.com/felipehoffmeister/Aula_Mobile)