**Comandos do Git**

*Primeiros Passos*

* git config --global user.name nome-do-autor;
* git config --global user.email email-do-autor;
* git config --global core.editor caminho-do-editor;
* git config user.name = Exibe o nome;
* git config user.email = Exibe o email;
* git config core.editor = Exibe o caminho do editor padrão;

# *Navegação no Console*

* \* ou . = Símbolo que seleciona tudo;
* pwd = Mostra o caminho até aquele local;
* ls ou ll = Mostra tudo que tem na pasta;
* cd referência = Acessa a pasta que foi referenciada;
* cd = Volta até a pasta de usuário do pc;
* cd ../ = Volta para a pasta anterior;
* ls -a = Mostra o diretório oculto de seu repositório;

# *Criando Arquivos/pastas*

* git init = Transforma um diretório/pasta em repositório;
* git init nome-do-repositório = Cria um repositório;
* mkdir nome-do-diretório = Cria um diretório;
* touch nome-e-tipo arquivo = Cria um arquivo da sua preferência;
* Para mover arquivos de outros lugares para o repositório, a maneira mais simples é usar o Explorador de Arquivos;

# *Exibição*

* git status = Mostra as mudanças que foram feitas;
* git diff = Mostra as mudanças detalhadas que foram feitas;
* git log = panorama dos commits;
* git log --oneline = panorama dos commits resumido;

# *Vim (Editor de Texto)*

* ESC = menu;
  + I = Abre a edição;
    - selecionar = Copiar o texto;
    - Apertar Scroll = Cola o texto;
  + V = Visual;
  + : = Console do vim;
  + :/ = Pesquisar palavras no console;
    - :wq = Salva o arquivo e data;
    - :q! = sair sem salvar;

# *Commits*

* git add = Adiciona ao container;
* git commit -m "comentário" = Adiciona ao diretório ".git";
* git commit -am “comentário” = Faz um commit de um novo arquivo/alteração sem passar pelo “container”;

# Tags

* git tag = Exibe uma tag existente;
* git tag -a número-versão -m “mensagem-versão” = Implica uma tag;
* git tag -a número-versão -m “mensagem-versão” hash-commit = Implica uma tag a um commit específico;

# *Branch*

* git branch = Mostra o branch em que se encontra;
* git checkout -b nome-branch = Cria um Branch;
* git checkout nome-branch = Se desloca para o branch anexado;

# *Revertendo*

* git reset HEAD = Retira todos os arquivos/mudanças postagens no “container”;
* git checkout nome-arquivo = Retira mudanças no arquivo, com base no último commit feito, se essas alterações não estiverem no “container”;
* git reset --hard 'HEAD' = O será excluído todas as versões posteriores do HEAD vinculado;

# *Repositório Remoto*

* git remote = Verifica se há um repositório remoto;
* git remote add nome-repositório-remoto link-repositório-remoto = Adiciona um repositório remoto;
* git remote -v = Exibe o link do repositório remoto conectado;
* git push -u nome-repositório-remoto master = Envia o repositório local para o remoto;
* git pull = Receber atualizações do repositório remoto;
* git clone = Clonar um repositório;

# *Comandos Auxiliares*

'comando' --help = Mostra o que pode ser feito com o comando vinculado;

# *Grafo de Commits*

* git log --graph --all = Mostra o Grafo de Commits;
* q = Para sair do grafo;

# **Referências**

Professor José de Assis - <https://youtu.be/_mB-TShMDvY>

Git - [https://git-scm.com/docs/git#Documentation/git.txt---help](https://git-scm.com/docs/git" \l "Documentation/git.txt---help)

Stack Overflow - [https://pt.stackoverflow.com/questions/528332/o-que-%C3%A9-head-no-git#:~:text=O%20HEAD%20no%20Git%20uma,ou%20a %20%C3%BAltima%20confirma%C3%A7%C3%A3o%20registrada](https://pt.stackoverflow.com/questions/528332/o-que-é-head-no-git" \l ":~:text=O HEAD no Git uma,ou a última confirmação registrada) .

Influenttech - <https://youtu.be/EscyTZSaHXA>

Nuvem do Futuro - <https://youtu.be/seXngk2dJr8>