**GitHub**

é uma plataforma de hospedagem de código-fonte com controle de versão usando o Git

-> qualquer coisa no repositorio local é reversivel

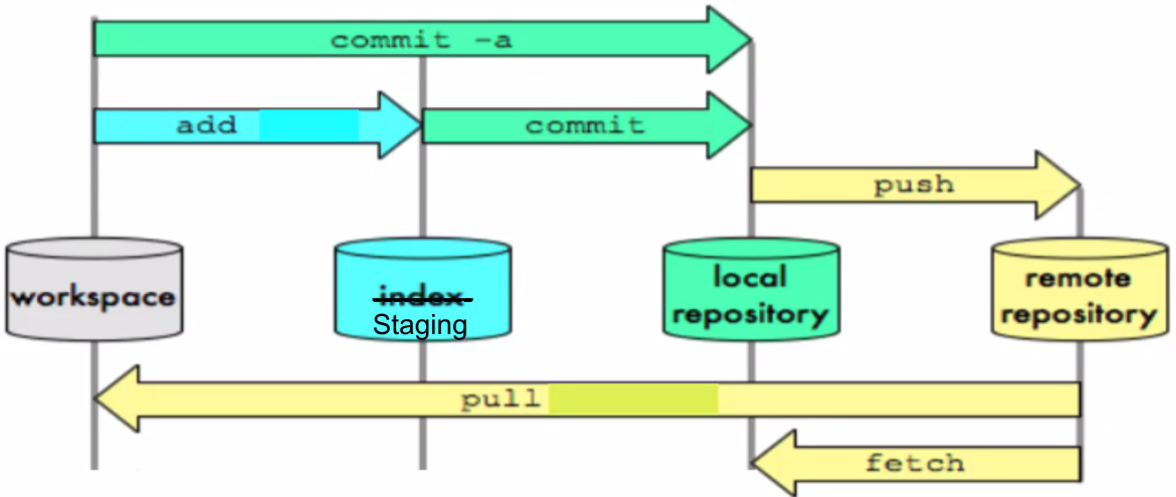
instalando o git: (no teminal do linux)  
sudo apt-get install git-all

configurando identidade:  
git config --global user.name "Seu nome"  
git config --global user.email "seuemail@exemplo.br"

configurando editor (editor de código usado, no caso o VSCode)  
git config --global core.editor "code --wait"  
 // abrir VScode no stage do git  
 code .

abre arquivo de configuração do git  
code .gitconfig

fluxo:



|  |  |
| --- | --- |
| **comando** | **Função** |
| git --version | verifica versão do git. |
| git init | inicia .git no diretório atual (fica oculto). |
| git clone urlDoRepositório | inicia .git no diretório atual clonando o conteudo da url (fica oculto). |
| git status | verifica status do git. |
| git log | mostra os logs de commits que foram feitos. |
| git add file.txt | adiciona arquivo na area de stage. (add . Para todos arquivos do diretório) |
| git commit -m "mensagem" | realiza o commit. |
| rm -rf .git | remove toda estrutura .git (cuidado) |
| git rm arquivo.txt | remove arquivo.txt (git commit -m "apagado") |
| git branch ramoDude | cria branch ramoDude. |
| git branch | verifica as branch. (--all --remote) |
| git branch -d <branchName> | delete branch locally |
| git push origin --delete <branchName> | delete branch remotely |
| git branch -m "ulala" | renomeia branch atual para "ulala" |
| git checkout ramoDude | seleciona a ramoDude |
| git checkout -b ramoDude | cria branch (-b) ramoDude e a seleciona |
| git reset <file> | se add arquivo por acidente antes do commit |
| git restore <branch> | restaura <branch> ficando igual do github |
| git push -u origin ramoDude | envia para github a branch ramoDude |

// checkout da branch do colega  
git checkout -b ana-branchName origin/ana-branchName --track

// push da sua branch apartir da branch do colega  
git push -u origin sua-branchName

Fluxo 1 - Após git init :  
 Criar seu primeiro commit ;  
 Criar o repositório remoto na sua conta do GitHub ;  
 Adicionar a URL do repositório remoto;  
 Executar o famoso push .

Fluxo 2 - Após git clone :  
 Criar sua branch dentro do repositório clonado;  
 Fazer checkout na sua branch;  
 Criar seu primeiro commit ;  
 Executar o famoso push .

// iniciar um repositório local (no diretorio atual)  
git init   
 // clonando repositorio  
 git clone urlDoRepositório

// criar um repositório remoto (pelo github)  
 // GitHub e procure o ícone com um sinal + na barra superior  
 // Create a new repository  
 // clicar no botão SSH e então copiar a (URL gerada)\*.

// conectando repositorio local com o remoto  
git remote add origin git@github.com:usar URL gerada   
 // origin == apelido do repositorio + (URL gerada)\*  
 // verificar se funcionou  
 git remote -v   
 // resultado:  
 // origin git@github.com:user-github/repo-name.git (fetch)  
 // origin git@github.com:user-github/repo-name.git (push)

// busca e compara repositorios (local e github)  
git fetch -> (depois git merge) == git pull  
 git status

// busca e compara repositorios (local e github) e puxa + faz merge // verificar se esta na branch main e:  
 git pull // main atualizado com a do GitHub  
 // git pull origin main // (se não estiver no main)  
 // agora pode-se atualizar as branch

// ressuscitar arquivo  
git log --diff-filter=D -- summary // diferencia filtro por = deletados sumario   
 // pega o \*numero do commit os 4x primeiros  
 // usar o comando para trazer arquivo deletado:  
 git checkout \*numero do commit~1 nomeArquivoDeletado.txt  
 git add nomeArquivoDeletado.txt  
 git commit -m "arquivo nomeArquivoDeletado.txt foi ressuscitado"

// ignorar arquivos que não quer adicionar no git  
touch .gitignore -> criado arquivo de ignorados (arquivo fica oculto)  
cat > .gitignore -> add no arquivo ignorados  
\*.txt -> ignorar todos os .txt (nomeDiretorio/ -> ignora diretorio)  
ctrl + d -> encerra  
 git status -> os .txt não estão mais presentes

// site gitignore io (toptal)  
 // visual studio  
 // sera criado linhas para ignorar arquivos do VScode   
 // linhas deverão ser add no .gitignore

Pull Request (github)  
você garante que o branch-padrão terá todo seu trabalho concluído e aprovado realizando todas as novas alterações em uma branch separada.

// fazendo o merge  
git merge "nomeProjeto"

// conexão entre local com o do github  
git remote add origin https://github.com/linkURL/oldcavalheiro/Trybe.git

// git diff  
ve diferença entre arquivos.

SSH ou Secure Shell : é um protocolo de criptografia de rede que serve para transferir dados de forma segura mesmo em redes inseguras. Usando o protocolo SSH, você pode se conectar ao GitHub sem precisar digitar seu nome e chave de acesso pessoal a cada comando executado.

HTTPS ou Hypertext Transfer Protocol Secure : é uma extensão do protocolo de internet HTTP ( você verá mais sobre esse protocolo no dia de conteúdo que trata sobre internet! ) que utiliza certificados digitais para autenticar os dados e permitir que eles sejam cripografados de forma segura.

gerando chave SSH

SSH ou Secure Shell:

é um protocolo de criptografia de rede que serve para transferir dados de forma segura mesmo em redes inseguras.

Usando o protocolo SSH, você pode se conectar ao GitHub sem precisar digitar seu nome e chave de acesso pessoal a cada comando executado.

HTTPS ou Hypertext Transfer Protocol Secure :

é uma extensão do protocolo de internet HTTP ( você verá mais sobre esse protocolo no dia de conteúdo que trata sobre internet! ) que utiliza certificados digitais para autenticar os dados e permitir que eles sejam cripografados de forma segura.

// gerando chave ssh  
ssh-keygen -t rsa -b 4096 -C "seuemail@gmail.com"  
 // exibe msg "Enter file in which to save the key"  
 // se der o enter cria arquivo chave em local padrão:  
 /home/usuarioDoPc/.ssh/id\_rsa  
 // pede para colocar a senha (Enter passphrase:)  
 // pede para colocar a mesma senha novamente. (Enter same passphrase again:)

// adicionar chave SSH ao ssh-agent  
 // iniciar ssh-agent em background  
 eval "$(ssh-agent -s)"  
 // retorna o PID do processo. (Agent pid "numProcesso")  
 // adicionar sua chave privada SSH ao ssh-agent  
 ssh-add ~/.ssh/id\_rsa  
 // vai pedir a senha. (Enter passphrase)  
 // após digitar senha correta ele adiciona.  
 (Identitya dded)

// adicionar chave SSH na conta GitHub  
 // instale o xclip para copiar a chave  
 sudo apt-get install xclip  
 // copie a chave  
 xclip -sel clip < ~/.ssh/id\_rsa.pub  
 // para copiar manualmente a chave. (sem o xclip)   
 cat ~/.ssh/id\_rsa.pub

//Entre no GitHub (site)  
No canto superior direito do GitHub, clique na sua foto de perfil e clique em Settings;  
 Na barra lateral esquerda, clique em SSH and GPG keys;  
 Clique em New SSH key ou Add SSH key;  
 No campo Título, adicione descrição para nova chave;  
 Cole sua chave dentro do campo Key;  
 Clique em Add SSH key