DIGITAL INOVATION ONE

BOOTCAMP SANTANDER

FULLSTACK DEVELOPER 2022

MODULO 1 – Introdução ao git e github

# INTRODUÇÃO

* GIT – Sistema de versionamento de arquivo
  + Software é construído de maneira colaborativa

# NAVEGAÇÃO VIA COMAND LINE

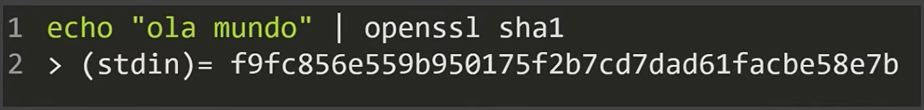
* Comandos de navegação
  + Windows / Linux – comando / ação

|  |  |  |
| --- | --- | --- |
| WINDOWS | LINUX | AÇÃO |
| dir | ls | Lista diretório |
| cd\ | cd\ | sobre diretório |
| cd “nome da pasta” | cd “nome da pasta” | entra na pasta |
| cd.. | cd.. | volta diretório |
| cls | clear (crtl+l) | limpa tela do terminal |
| tab |  | auto completar comando |
| mkdir “nome da pasta” | mkdir “nome da pasta” | cria pasta nova |
| echo “nome arquivo” | echo “nome arquivo”.txt | redireciona o fluxo para o nome arquivo digitado |
| echo “nome arquivo” > “nome arquivo.txt” | echo “nome arquivo” > “nome arquivo.txt” | Verifica se na pasta existe um arquivo com o nome “nome arquivo”, se não tiver cria o arquivo nome arquivo.txt |
| rm “nome da pasta” /s /q | rm -rf “nome da pasta” | deleta repositório “nome da pasta” |

# INSTALAÇÃO DO GIT

* Instalação Windows
  + Normal
* Instalação Linux - comandos baixados do site do git
  + $ sudo add-apt-repository ppa:git-core/ppa -y
  + $sudo apt update
  + $ sudo apt install git
  + git –version

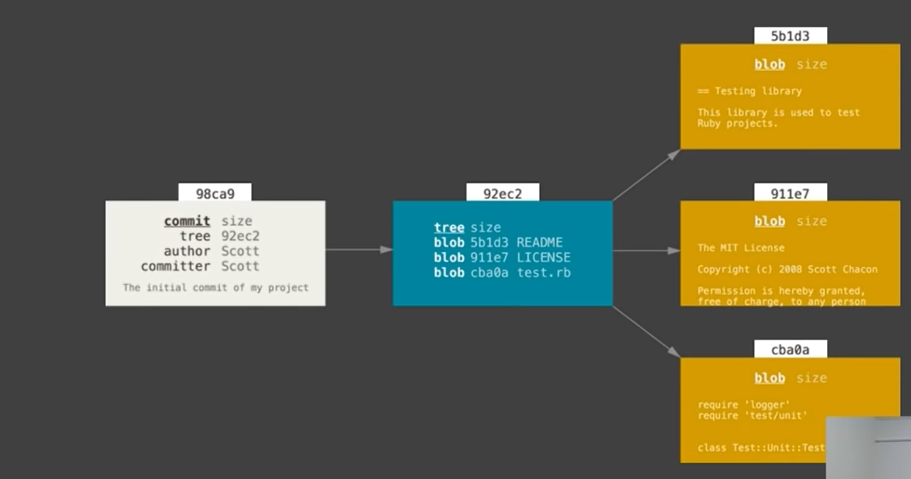
# ENTENDENDO COMO FUNCIONA O GIT

* SHA1 – Secure Hash Algorithm – Algoritimo de hash Seguro
  + Conjunto de funções criptografadas
  + Dados encriptados geram conjunto de caracteres de 40 digitos
    - Único e serve como identificação
    - Openssl sha1
    - 
  + Git Bash
    - Botão direito na barra de títulos/options/locks/theme
    - Objetos básicos do git
      * Blob - bolha
        + Tipo do objeto
        + Tamanho
        + \0
        + Conteúdo
      * Tree – arvore
        + Armazena blobs e aponta para tipos de blos diferentes e commits e outras trees
        + Monta toda a estrutura de localização dos arquivos
        + Estrutura

Tree

\0

Blob “sha1” “nome do arquivo”

* + - * Commit – Objeto que junta tudo
        + Possui sha1
        + tamanho
        + Aponta para tree/ parente/ autor/ mensagem/ timestamp
        + 
    - 

|  |  |
| --- | --- |
| Comandos | Ação |
| openssl sha1 “nome do arquivo” | Gera o arquivo sha1 |
| echo ‘“nome do arquivo”’ | git hash-object –stdin | Gera o arquivo sha1 para “nome do arquivo” |
| echo – ‘blo 9\0”nome do arquivo”’|openssl sha1 | Gera um tipo blob e a sha1 para o arquivo |

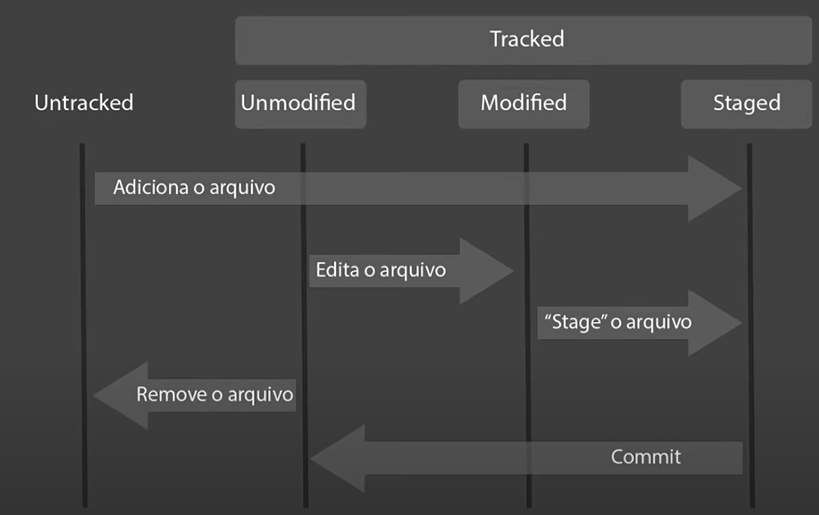
* CHAVES SSH E TOKENS
  + Chaves ssh
    - Conexão segura e confiável entre seu computador e o github
    - Dentro do github
      * No Git Bash
        + ssh-keygen -t ed25519 -C “e-mail do github”
        + programa gera local aonde a chave foi salva publica (.pub) e privada
        + cd “caminho especificado aonde gerou a chave”.ssh/
        + ls – lista os arquivos
        + cat “nome da chave .pub”
        + copiar a chave
      * No site do git
        + Profile/setting/ssh and gpg key/new ssh key

Title: “nome para identificação do pc”

Key: colar a chave copiada

* + - * Inicializar ssh agente – entidade que lida com as chaves
        + eval $(ssh-agent -s)
        + ssh-add “nome da chave privada”

# PRIMEIROS COMANDOS COM GIT





* Iniciar repositório do git
  + Criar uma pasta á ser utilizada e iniciar o repositório para depois mandar um commit
* git init
  + navegar até a pasta de trabalho do projeto e digitar o comando para inicializar um repositório vazio
* Configurar o git para realizar commits
  + git config –global user.email “e-mail do github”
  + git config –global user.name “nome do usuário”
* git add – área de stage
  + git add \*
  + git add “nome do arquivo” ./nome da pasta
    - adiciona os arquivos e pastas modificados
  + git restore --staged
    - tira os arquivos da área de stage
* git status
  + monitora o estado dos arquivos
* git commit
  + git commit -m “commit inicial”

# USANDO O GIT HUB

* CRIAR UM REPOSITORIO
* LINKAR O REPOSITORIO CRIADO COM O CRIADO NO COMPUTADOR
  + copiar link do repositório
  + git remote add origin “link do repositório”
  + git remote -v
    - exibe os origins
  + git push origin master

# CONFLITOS NO GITHUB

* gir pull
  + git pull origin master
    - puxa o arquivo do repositório do github
* git clone “link do repositório”

|  |  |
| --- | --- |
| COMANDOS GERAIS GIT | |
| COMANDO | AÇÃO |
| ssh-keygen -t ed25519 -C “email do github” | Gera chave ssh |
| cd “caminho da pasta” | Move para a paste especificada |
| ls | Lista arquivos da pasta |
| clear | Limpa tela do gitbash |
| cat “nome da chave” | Exibe a chave ssh |
| pwd | Exibe local complete de onde vc esta |
| eval $(ssh-agent -s) | Inicializa agente gerenciador de chaves |
| ssh-add “nome da chave privada” | Passa chave privada para o gerenciador |
| git clone “caminho ssh” | Clona o caminho ssh para a pasta em que vc esta |
| git init | Inicia um repositório |
| ls -a | Mostra arquivos resolutos |
| cd .. | Volta diretório |
| git config –global user.email “email do github” |  |
| git config –global user.name “nome do usuario” |  |
| git add \* |  |
| git commit -m “nome do commit” |  |
| mkdir “nome da pasta” | cria diretório |
| mv “nome do arquivo” ./”nome da pasta” | move o arquivo para a pasta |
| git config –list | lista as configs |
| q | sai do config |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |