12/12/21

M3 - Introdução ao Git e ao GitHub

A1.1) Entendendo o que é Git

√ Por que estou aprendendo esta tecnologia? De que forma a tecnologia Git pode me ajudar?

1. Criado em 2005 por Linus Torvalds, é um softwware de versionamento de código distribuído

a. O Git foi "criado", pois Linus se deparou com problema, estava desenvolvendo o sistema "Linux" de forma colaborativa com a comunidade de DEV do mundo todo.

b. Como saber qual a versão mais recente do softwware? Quais foram as últimas modificações? Como versionar o terminal?

2. Após a criação do softwware de versionamento de código Git, onde irei armazenar os dados e informações de um projeto?

a. Surge então, o GitHub (microsoft), que é um repositório online (Host do código).

A2.1) Comandos básicos no terminal

√ A grande parte dos softwwares e aplicativos, hoje possuem interface gráfico de interação com o usuário.

12/12/21

√ Mas no início do desenvolvimento da computação, os computadores eram (CLI - Command Line Interface) e algumas tecnologias, principalmente voltadas para desenvolvimento são CLI;

√ Comandos básicos no Bash:

- listar diretório em uma pasta: ls
- navegação entre pastas: cd
- retornar "um nível" em pastas: cd ..
- limpar tela do terminal: clear ou Ctrl + l
- criar uma pasta: mKdir nome da pasta
- criar arquivo: echo nome > nome.extensão
- apagar diretório: rm -rf nome diretório/

A3.1) Entendendo o funcionamento do Git

√ SHA significa Algoritmo de Hash Seguro, é um conjunto de funções hash criptográficas projetadas pela NSA (Agência de segurança nacional dos EUA);

1. No geral, é um algoritmo que pega seu arquivo e embaralha de forma específica. A saída da encriptação gera um conjunto de caracteres único com 40 dígitos;

12/12/21

A3.2) Objetos internos do Git

√ Os objetos básicos do Git responsáveis pelo versionamento de código são: Blobs, Trees, Commits;

SHA1 ax1s412fd....

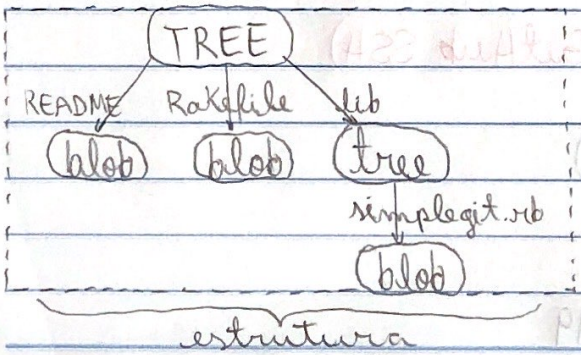
Blob TAMANHO42
\0
Ola Mundo

echo -e 'conteudo' | openssl sha1
≠
echo -e 'blob 9\0conteudo' | openssl sha1
=
echo 'conteudo' | git hash-object --stdin

√ O Git guardar os arquivos fazendo o SHA específico de encriptação, mas também, guarda metadados nestes objetos;

√ As Trees armazenam Blobs, ela aponta para um Blob ou mais juntamente com o SHA1 deste Blob específico, mas também conseguem guardar o nome do arquivo;

1. As Trees são responsáveis por montar toda estrutura de onde está o arquivo, além de ser um objeto recursivo;



√ Já o Commit junta tudo e dá sentido a alteração que você está fazendo;

12/12/21

A4.1) Iniciando Git e criando Commit

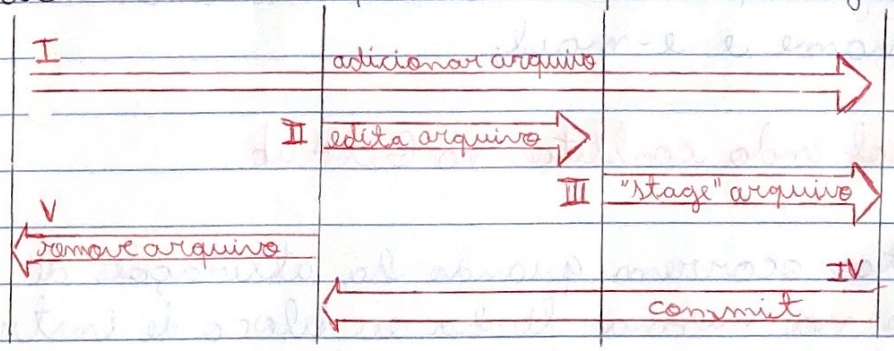
√ No geral, passaremos por três instruções gerais: git init, git add, git commit, git push;

- Visualizar pastas ocultas: $ ls -a
- configuração: $ git config --global user.email "email"
  $ git config --global user.name "nome"

- $ git init
- $ git add or git add * (todas modificações)
- $ git commit "name"

A5.1) Ciclo de vida dos arquivos no Git

√ O comando 'git init', além de criar a pasta .git este comando inicializa o conceito de repositório dentro da pasta em questão;



√ O comando 'git init' nos fornece duas classificações dentro do repositório servidor e ambiente de desenvolvimento, respectivamente;

12/12/21

- situação atual: $ git status
- mover arquivo: $ mv arquivo ./pasta/
- criar arquivo: touch nome arq. extensão

A6.1) Trabalhando com GitHub

√ Após criar uma conta no GitHub, iremos criar um repositório (servidor online) para empurrar nossos arquivo com o Git;

- $ git remote add origin https:// link repositório
- $ git branch -M main
- $ git push -u origin main
- $ git push (commit já criados)

√ O GitHub nos fornece um SHA1 associado ao repositório, além de fornecer informações como:

1. datação do último commit, quais foram as realizadas, quais os conflitos, dados do autor como name e e-mail;

A7.1) Resolvendo conflitos no GitHub

√ Conflitos ocorrem quando há alterações de um arquivo na mesma linha ou bloco de instruções. Isto ocorre constantemente quando há pelo menos duas pessoas trabalhando em um arquivo. Após alterações, uma das pessoas terá um arquivo desatualizado em relação ao GitHub;

$ git pull origin main or master