GUI X CLI

CLI – Common Line Interface – Linhas de comando

GUI – Graphic User Interface

COMANDOS PARA APRENDER A NAVEGAR NO GIT

1 – Mudar de Pastas

2 – Listar as Pastas - Dir

3- Criar Pastas/Arquivos

4 – Deletar Pastas/Arquivos

WINDOWS X UNIX

Cd Cd Cd = Change directory (navegar nas pastas)

Dir Is

Mkdir mkdir (make directory)

Del/rmdir rm/rf

Voltar é cd .. (cd + dois pontos) (unix é cd + .- um ponto) Sai da pasta

Limpar a tela – cls (clear screen) (unix é o clear ou o ctrl + L)

TAB tem a função de auto completa

Echo – printa de volta no terminal uma frase, um texto que você passa

O símbolo > é um redirecionador de fluxo – ele pega o output (saída da função) e joga em um arquivo

O comando del se restringe apenas a deletar ARQUIVOS. Existe uma diferença entre deletar um arquivo e um repositório. Para realmente remover o repositório com todos os arquivos que tem lá dentro deve-se usar o rmdir.

A seta pra cima serve para navegar no Histórico de comandos que já executou no terminal

Flags – complementos que são passados para os comandos que acrescentam, modificam ou formatam a forma que esses comandos são devolvidos pra gente

Silence on sucess – se der tudo certo o terminal não vai te falar nada

Tópicos fundamentais para entender o funcionamento do Git

1 - SHA1

2 – Objetos fundamentais que constituem o sistema (que rodam por baixo dos panos no Git)

3 – Sistema Distribuído

4 – Segurança

1 – SHA1 – Algoritmo de encriptação – Secure Hash Algorithm |ALGORITMO DE HASH SEGURO| - É um conjunto de funções hash criptográficas projetadas pela NSA

Ele é um algoritmo que vai pegar o seu arquivo e vai embaralhar ele de uma forma muito específica

A saída dessa encriptação, esses dados encriptados gera conjunto de caracteres identificador de 40 dígitos, esse conjunto é ÚNICO

Ele identifica os arquivos de forma segura e rápida através desse número de identificação de 40 dígitos

Ele é uma forma curta de representar um arquivo e seu estado

2 – Objetos Fundamentais que constituem o sistema do Git (que rodam por baixo dos panos)

Quem São?

BLOBS

TREES

COMMITS

São os 3 tipos básicos de objetos do Git responsável pelo versionamento do nosso código

BLOBS

Os arquivos ficam guardados dentro desse objeto chamado BLOB, que contem metadados dentro dele. Ele vai ter o tipo do objeto, o tamanho, a \0 e o conteúdo desse arquivo, seja qual for o conteúdo. O blob contém metadados do Git. O blob o bloco básico de composição

TREES

Elas armazenam Blobs e apontam para tipos de Blobs diferentes e/ou para outras árvores. Também contém metadados, tem a \0. Ela também guarda o nome desse arquivo. O blob não guarda o nome do arquivo, ele só guarda o SHA1.

As TREES são responsáveis por montar toda a estrutura de onde estão os arquivos.

Elas tem SHA1 dos metadados, mudou o SHA1 de um dos arquivos dos Blobs que a TREE aponta, consequentemente muda o SHA1 dessa TREE também.

É como se fosse uma pasta que dentro tem arquivos ou outras pastas contendo arquivos.

COMMIT

É o objeto mais importante de todos, ele vai juntar tudoo. Ele aponta para uma árvore, aponta para um parente (o último commit realizado antes dele), aponta pra um autor do commit, pra uma mensagem (vai estar explicando, dando significado pra esse monte de arquivo dentro desse monte de pasta). Eles contêm também um timestamp (carimbo de tempo) – data e hora certinha de quando foi criado.

Eles também possuem um SHA1, uma encriptação que muda se mudar qualquer coisa dentro de um blob que tá dentro uma arvore que tá dentro do seu commit.

Ele é único pra cada autor

O Git é um sistema distribuído e seguro

CHAVES SSH E TOKENS – REVER ESSA AULA MAIS ADIANTE

CHAVE SSH – É uma forma de estabelecer uma conexão segura e encriptada entre duas máquinas. A gente vai se conectar com o servidor do Github e a gente vai configurar nossa máquina local como uma máquina confiável pelo Github, estabelecendo essa conexão com duas chaves (uma pública e a outra privada) – pega a pública e vai lá no Github colocando ela lá, então o github passa a conhecer nossa máquina e a sua assinatura. Nossa máquina já vai ficar configurada pra poder enviar os códigos pra lá. Onde essa chave vai estar?

TOKENS de acesso pessoal – usa o token como senha para gerar um commit

Primeiros comandos com o Git

Iniciar/criar nosso repositório Git – git init

Mover arquivos e dar inicio ao versionamento – git add

Criar commit – git commit

No terminal sempre se coloca o nome do programa na frente, chamando assim pelo terminal o git e comandos específicos do git

Flag – -a (mostra arquivos ocultos)

Arquivo HTML – É o esqueleto, a estrutura básica de qualquer página na web

Markdown – o arquivo markdown é uma forma mais humana de escrever um arquivo html

Tracked – o git tem ciência da existência deles

Unmodified – ao editar o arquivo ele vira modified

Modified – se rodar o git add de novo no arquivo a gente vai de modified para staged

Staged – Arquivos que estão se preparando para poder fazer parte de um outro tipo de agrupamento (de um commit)

Untracked – o git não tem ciência da existência deles

O git add pega um arquivo que tava untracked pq a gente tinha acabado de criar o arquivo e move para tracked. O git rm faz o oposto disso.

O comando git add moveu aquele arquivo untracked direto para staged

Remover o arquivo vai de unmodified para untracked

Snapshot tá salvo dentro do commit, assim ao ir de staged para um commit o arquivo volta para o estado unmodified

Servidor – o git é um sistema distribuído, vai ter a versão no servidor (que será o github). É o remote repository (github) que fica guardado em um servidor

Ambiente de desenvolvimento – representa tudo que tá na nossa máquina (versão que tá na nossa máquina). As alterações que você faz no seu código na sua máquina, elas não repercutem imediatamente na versão que tá no seu repositório remoto, a não ser que você execute um conjunto de códigos específicos, você empurra a alteração do seu repositório local para o seu repositório remoto.

No ambiente de desenvolvimento temos:

Working Directory – repositório de trabalho

Staging area – os arquivos vão estar sempre alterando entre essa área e o repositório de trabalho

Local Repository – quando você faz um commit ele passa a integrar o seu repositório local e o seu repositório local pode ser empurrado para o seu repositório remoto. Tudo dentro dele tem de estar commitado, ele é composto de commit, pois só assim podemos passar para um repositório remoto.

O git status é um comando que ajuda a monitorar tudo isso

O comando para mover no terminal é mv + nome do arquivo + ./ + nome do lugar que vamos colocar

Para mudar de staged para unstaged usar o comando git restore –staged + o nome do arquivo

O git add \* pega tudo que foi modificado, tudo que tá no meu working directory e adiciona para o staged para ser commitado

O git commit -m adiciona uma mensagem

O git remote add origin + colar a url gerado no Github do repositório

O git remote -v lista as listas de repositórios remotos que tem cadastrado

Conflito de Merge

Git pull origin master

Para clonar copia a url e dá um git clone + a url copiada

-a é para mostrar repositórios ocultos