**Prompt de comando**

**Windows**

**\***linux e mac muda alguns comandos

**Terminal de comandos – CMD**

**Dir –** ver tudo que tem na pasta

**cd / -** voltar para a pasta principal c:\

**cd “nome” –** entrar na pasta especifica

**cd “N”+tab –** já acha a pasta com a primeira inicial

**cd.. -**  volta para pasta anterior

**cls –** apagar tela para não ficar poluído

**mkdir “ “ –** criar uma pasta (se não aparecer nenhuma info, quer dizer que criou. “Silence is success”)

**echo > “ “ –** comando para replicar algo que esta escrevendo e que pode ser criado como arquivo. Echo > hello > hello.txt vai criar um arquivo txt na pasta, caso já não haja outro com mesmo nome.

**Del “ “ –** del mais o nome da pasta, n vai deletar a pasta, mas sim, os arquivos dentro dela

**Rmdir “ “ –** deletar a pasta. Caso tenha algo na pasta, e queira deletar tudo, **rmdir “ “ /s /q**

**GIT e GIT HUB**

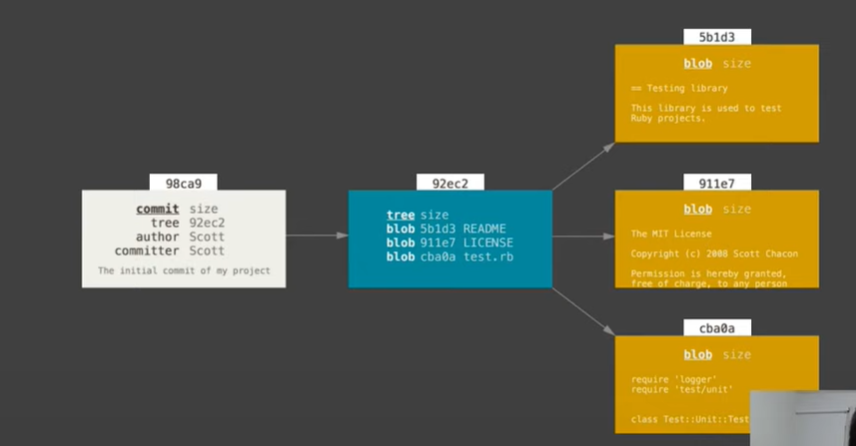
Sha1 – sistema de encriptação que lê um arquivo e encriptografa em 40 digitos. Qualquer alteração feita, muda os 40 digitos. Caso volte na mesma alteração, ele retoma os outros 40 digitos.

Objetos internos do GIT

**Blobs –** bloco que contem sha do aquivo: tipo do arquivo, tamanho e conteúdo

**Tree –** armazena os blobs, nome, localização dos arquivos e até mesmo outra tree, pois pode ter outro diretório dentro. E possui seu próprio sha, que pertence a tudo que esta na Tree.

**Commit –** aponta para a arvore, pro autor, pra mensagem (que pode ser um comunicado para as alterações, o responsável, e datas). Também possuem um sha1.



**Comandos git**

**Ls –** ver tudo que esta dentro da pasta

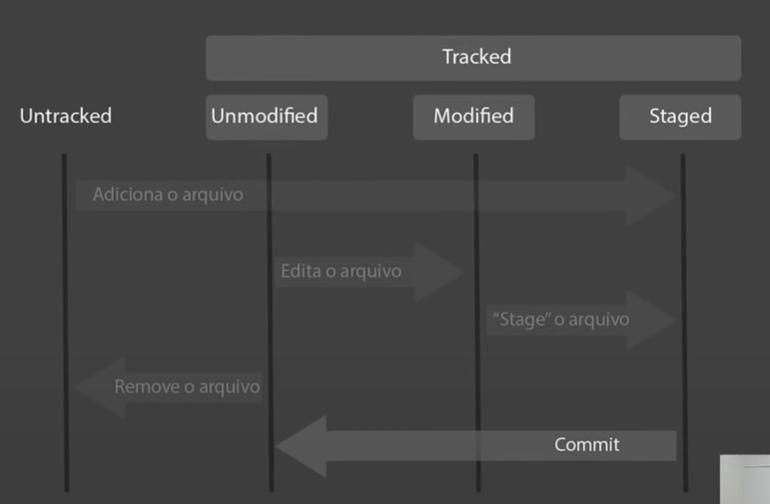
**Cd “ “** – entrar na pasta especifica

**Ctrl+l –** limpar tela

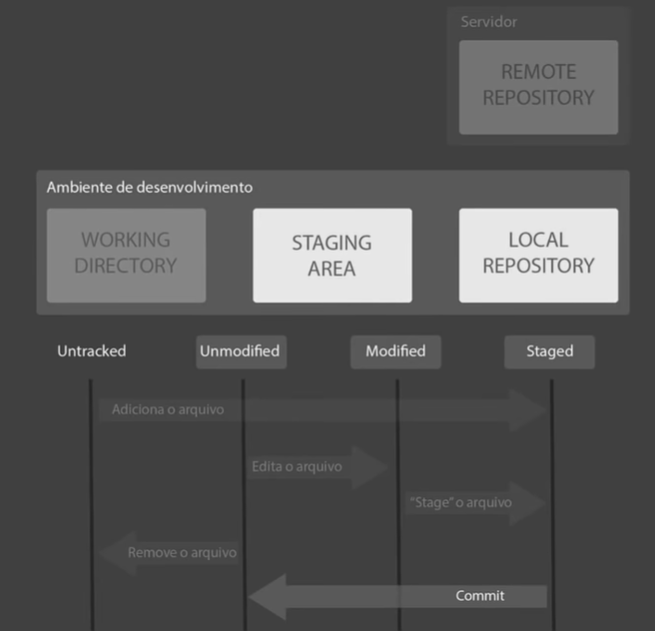
**Mkdir –** criar pasta

**Ls -a** – mostra pasta oculta

**Git init** – inicializar o git naquela pasta



Arquivo ele esta untracked, você incia git nele, ele passa para unmodified, ainda não teve alteração, assim que começa a alterar, ele vira modified e passa para staged, que esta pronto. Depois de virar um commit, por um autor, uma mensagem, ele gera um sha1, grava todas as info, e volta a ser um unmodified pra iniciar todo processo.



Os commits, ficaram disponíveis no Local repository, que é o único local que poderá ser enviado para um Remote repository, como o github.

**Git status –** comando que vai mostrar se tem algum arquivo que não foi comitado, ou qual o status.

**Mv “ “ ./ “pasta que quer mover”/**  - vai mover o arquivo para pasta informada

**Criando primeiro commit**

**git remote add origin** <https://github.com/Ginomenna/livro-receitas.git> - comando para preparar para enviar commit para remoto. O link é fornecido no site do github

**git remote -v** – mostra o link onde esta localizado o diretorio

**Git add \*** - comando para iniciar uma criação de commit naquela pasta , de tudo que esta pronto

**Git commit -m “nome do commit” –** para deixar ele como commit

**Git push origin master** – comando para subir o commit para o github.

Caso uma pessoa pegue aquele arquivo e mude, vai aparecer o erro informando que contem trabalho que não esta armazenado no seu diretório local, então você tem que puxar aquele arquivo, pra modificar e ai sim fazer o push

**Git pull origin master** – vai puxar o arquivo modificado por outra pessoa

Caso você queira fazer o download de um arquivo de alguém, você entra no github, entra no arquivo, na acaba Code, copia o link do arquivo. Vai em seu dirtorio base, workspace, e usa o conando: **git clone “ link url”**

**Criando aquivo**



Markdown é uma forma mais humana de criar um html, mais fácil. Usa # para mostrar os níveis.

**Typora -**  programa para abrir os arquivos MD

Comandos:

# espaço “ “ – para já criar a primeira linha na formatação

\*\*” “\*\* - negrito

\_” “\_ - itálico

:nome do emoticon – por emoticon

Criando primeiro commit

**Git add \*** - comando para iniciar uma criação de commit naquela pasta

**Git commit -m “nome do commit”**