GIT

Git init: inicia ou reinicializa um repositório git na pasta

Git Status: lista o status dos arquivos da working tree

Git show : mostra vários tipos de objetos

Git commit -m"mensagem": salva as alterações em um commit na branch atual

cat <nome do arquivo>: exibe o conteúdo do arquivo no prompt

BRANCH

Git Branch: lista as branches

Git Branch <nome da branch> : usado para criar branches novas.

Git Branch -d <nome da branch> : usado para deletar branches

Git Branch -d -r origin/<remote branch name>: deleta uma tracking branch remota

Git Checkout <nome da branch>: usado para alternar de branch

Git Checkout -b <nome da branch>: cria e alterna para a branch

Git Checkout <nome do commit>: usado para alterar o HEAD da branch pro commit

BRANCH LABELS

local master branch label: aponta para o ultimo commit na branch master local

tracking master branch label: aponta para o ultimo commit usado com um network command

remote master branch label: aponta para o último commit na branch master remota

Git remote set-head origin <branch remota>: define essa branch como a tracking branch default, assim podendo utilizar apenas "origin" ao invés de "origin/<branch remota>"

LOG

Git log : lista os commits da branch atual

Git log <nome da branch>: mostra os commits da brach especificada

Git log --all: lista todos os commits de todas as branches

Git log --oneline: lista os commits da branch atual mais resumido em uma linha.

Git log --graph: lista os commits com um grafico

TAGS

**OBS: tags não explicitadas o commit, usam por default o mais recente.**

TAG SIMPLES

Git tag <nome da tag> <referência do commit>

TAG ANOTADA

Git tag -a -m"mensagem" <referência do commit> nome da tag

Git tag: lista as tags.

Git tag -d <nome da tag> : deleta a tag

Git push origin <nome da tag> : faz o push da tag

REFERÊNCIAS

**Descrição:** identificações para encontrar objetos na branch

HEAD: ref q sempre aponta para o commit mais recente na branch atual

apêndice ~(numero): ref para commits anteriores

apêndice^(numero): ref para a ordem do parente do commit.

Git Reflog: mostra um histórico dos últimos commits por onde o HEAD passou

RE-ESCREVENDO A HISTÓRIA

**Git Commit --amend** : permite atualizar o último commit da brach podendo mudar sua mensagem usando o parâmetro -m””, e ou, alterando arquivos os adicionado na stage area antes de lançar o comando, para manter a mensagem do commit anterior e apenas atualizar seus arquivos utilize o parâmetro --no-edit no comando. (isso na verdade substitui o commit anterior, ou seja mudando seu SHA1)

**git rebase <base branch>** : reordena os commits para que o primeiro commit da branch "topic" troque seu parent para o último commit da "Base Branch", isso permite que possa dar merge para a Branch Base usando Fast-Forward(PORÉM ISSO ALTERA TODOS OS ID'S DOS COMMITS FEITOS NA TOPIC BRANCH, TAMBÉM PODE CAUSAR MERGE CONFLICTS CASO O MESMO HUNK NO HEAD DA MASTER BRANCH ESTIVER ALTERADO)

**git rebase -i <commit anterior aos que você precisa modificar>**: realiza um rebase interativo na mesma branch que você está, você pode escolher quais commits deseja modificar e após salvar o arquivo o rebase acontecerá, parando nos arquivos demarcados para você fazer a alteração.

**git merge --squash <nome da branch topic>:** realiza um merge porém não mantém o histórico da branch topic na branch base. no caso de (Fast-forward merge) ele aplica um único novo commit a base branch com o mesmo conteúdo do último commit realizado na topic.