Git - Comandos explicados

git config

```
git config --global user.name "nome_do_usuario"
```

Faz a configuração do nome de usuário.

```
git config --global user.email "email_do_usuario"
```

Faz a configuração do e-mail do usuário.

```
git config --global --list
```

Faz a listagem de todos os valores atribuídos na configuração e mais configurações padrões.

git init

```
git init
```

Inicializa repositório git. O diretório usado para a criação não precisa necessariamente estar vazio, pode existir arquivos antigos. Isso torna possível fazer o controle de versão em um projeto já existente.

git add

```
git add --all
```

Adiciona todos os arquivos (adicionados, modificados e removidos) na área de preparação (staging area) e os deixa preparado para o commit, indicando ao git que esses arquivos serão rastreados.

```
git add <nome_do_arquivo.extensão>
```

Adiciona único arquivo na área de preparação (staging area).

git commit

```
git commit -m "mensagem_commit"
```

A partir das mudanças que estão na área de preparação, salva mudanças no repositório local juntamente com a mensagem de commit informada, o nome do usuário, e-mail, data e horário.

git status

```
git status
```

Verifica quais os estados os arquivos se encontram. Se existe algum arquivo modificado, adicionado ou removido. Além disso, caso esteja rastreado com uma branch no servidor, verifica se a versão local está a frente ou atrás da versão do servidor.

git log

```
git log
```

Mostra histórico de alterações em ordem cronológica juntamente com o hash do commit, nome, email, data e hora da alteração.

```
git log --stat
```

Mostra histórico de alterações em ordem cronológica juntamente com o hash do commit, nome, email, data e hora da alteração. Além disso, mostra quais os arquivos alterados no commit.

git diff

```
git diff <commit_1> <commit_2>
```

Faz a comparação entre dois commits. Mostra quais foram os arquivos alterados, novos e removidos. Além disso, mostra também quais foram as linhas alteradas.

git diff

Faz a comparação entre o que está na área de trabalho e commit mais recente.

git diff --cached ou git diff --staged

Faz a comparação entre o que está na área de preparação e commit mais recente.

git checkout

git checkout <nome>

Caso o <nome> seja um branch:

- Muda o código para a branch selecionada. Alterações devem ser salvas (commited) antes de fazer a troca de branch.

Caso o <nome> seja um arquivo:

- Desfaz as alterações no arquivo, porém arquivo já deve ter sido adicionado nas mudanças.

Caso o <nome> seja uma tag:

- Muda o versionamento para onde a tag foi criada.

Caso o <nome> seja o hash de um commit:

- Muda o versionamento para o commit especificado.

git checkout -b <branch_name>

Faz a criação de uma nova branch e já faz a troca do versionamento para a nova branch.

git reset

git reset --hard

Desfaz todas as alterações que aconteceram em arquivos rastreados.

git reset --hard HEAD~1

Desfaz o último commit. Caso o commit esteja local, todas as alterações serão perdidas. Caso o commit já esteja no servidor, as alterações não são perdidas, somente são desfeitas localmente.

git clean

> *git clean -n*

Mostra uma lista de arquivos que serão apagados.

> *git clean -f*

Apaga todos os arquivos incluídos que ainda não estão rastreados.

> *git clean -i*

Mostra uma lista de opções para apagar somente alguns arquivos.

git clone

> *git clone <origem>*

Faz a clonagem do repositório para a pasta corrente. O valor da origem pode ser uma pasta local ou pode ser uma URL para o código ser baixado.

git pull

> *git pull*

Atualiza repositório local com a última versão da origem da branch remota. Necessário commitar as mudanças para executar essa ação.

Se for no servidor:

- Branch já deve estar rastreado no servidor. Assim, o git sabe que deve comparar a versão local com o servidor.

Se for no repositório local:

- Deve ter sido clonado de outra pasta para a pasta destino. Só assim, o repositório terá um 'pai' que poderá ser verificado para atualização de novas informações.

Esse comando não funciona caso o versionamento seja somente local.

git branch

git branch

Faz a listagem de todas as branches locais.

git branch -r

Faz a listagem de todas as branches remotas.

git branch -a

Faz a listagem de todas as branches locais e remotas.

git branch -d <branch_name>

Faz a remoção de um branch local.

git branch -D <branch_name>

Força a remoção de um branch local. Esse comando é necessário, pois caso exista uma branch que ainda não foi feito o merge, Git notifica e não permite apagar a branch somente com o comando 'git branch -d'.

git branch -m <nome_antigo> <nome_novo>

Renomeia a branch especificada no <nome_antigo> para <nome_novo>. Caso esteja na branch que será feita a alteração, não é necessário fornecer o nome antigo, basta executar o comando git branch -m <nome_novo>.

Essa alteração é muito simples de ser executada em branches locais, porém no caso de uma branch já estar no servidor e for o esperado renomear uma branch no servidor, é necessário apagar e fazer a criação da branch com o nome correto.

git push

git push

Faz o envio das mudanças comitadas localmente para a origem da branch rastreada.

```
git push -u origin <nome_da_branch>
```

Faz o envio da branch local para o servidor pela primeira vez. Caso a branch que está sendo enviada não exista no servidor, ela será criada. A partir desse momento, a branch local está configurada para ser rastreada com essa origem no servidor.
A partir desse primeiro comando, para versionar próximas mudanças basta que seja feito o comando 'git push' descrito acima.

git merge

```
git merge <nome_da_branch>
```

Mescla as mudanças presentes na <nome_da_branch> na branch corrente.

git tag

```
git tag
```

Faz a listagem de tags.

```
git tag <nome_da_tag>
```

Faz a criação de uma tag no último commit da branch corrente.

```
git tag <nome_da_tag> -m "mensagem"
```

Faz a criação de uma tag no último commit da branch corrente juntamente com uma mensagem.

```
git tag <nome_da_tag> <hash_do_commit>
```

Faz a criação de uma tag no commit específico.

```
git tag <nome_da_tag> <hash_do_commit> -m "mensagem"
```

Faz a criação de uma tag no commit específico juntamente com uma mensagem.

```
git push origin <nome_da_tag>
```

Faz o envio de uma única tag para o servidor.

git push --tags

Faz o envio das tags locais para o servidor.

git tag -d <nome_da_tag>

Faz a remoção da tag localmente.

git push origin --delete <nome_da_tag>

Faz a remoção de tag no servidor. Mesmo depois de apagada do servidor, tag ainda continua localmente caso não seja apagada.

git checkout -b <branch_a_partir_da_tag> <nome_da_tag>

Faz a criação de uma branch a partir de uma tag.

git stash

git stash

Salva as mudanças ainda não comitadas em uma pilha temporária para uso posterior.

git stash list

Faz a listagem das mudanças salvas na pilha temporária.

git stash apply

Traz a alteração mais recente que foi salva para a branch atual.

git stash apply stash@{X}

Traz a alteração de posição X que foi salva para a branch atual. Podem existir várias versões de códigos salvas na pilha temporária. Dessa maneira, é possível especificar qual alteração se trata.

git stash drop

Faz a remoção mais recente da pilha temporária.

git stash drop stash@{X}

Faz a remoção específica de uma versão salva na pilha temporária.

git stash pop

Traz a alteração mais recente que foi salva para a branch atual e logo em seguida já faz a remoção dessa alteração da pilha.

alias

git config --global alias.<abreviação> <comando_git>

Cria uma abreviação para o comando do git.

Ex: git config --global alias.st status

Faz a criação de um atalho para o comando status. Dessa maneira esse comando pode ser executado como 'git st'.

git config --global --unset alias.<abreviação>

Faz a remoção do alias cadastrado.

git config --global alias.<abreviação>

Verifica o comando que foi cadastrado.

git config --get-regexp

Faz a listagem de todos os alias configurados.

git remote

git remote -v

Faz a listagem dos servidores remotos que o repositório está usando associados com a URL.

git remote <URL> origin

Faz a troca da URL do servidor 'origin' para a nova URL informada. Caso exista mais de um servidor remoto que o repositório esteja sendo rastreado, a palavra 'origin' pode ser substituída.

git fetch

git fetch

Faz o download das atualizações do repositório remoto e trás para o repositório local.

git rebase

> *git rebase <branch>*

Aplica os commits da <branch> na branch corrente.