**Comandos Básicos de GIT**

**git config**

Um dos comandos git mais usados é o **git config** que pode ser usado para definir valores de configuração específicos do usuário como e-mail, algoritmo preferido para diff, nome de usuário e formato de arquivo etc. Por exemplo, o seguinte comando pode ser usado para definir o email:

git config --global user.email [sam@google.com](mailto:sam@google.com)

**git init**

Este comando é usado para criar um novo repositório GIT. Uso:

git init

**git add**

O comando **git add** pode ser usado para adicionar arquivos ao índice. Por exemplo, o seguinte comando irá adicionar um arquivo chamado temp.txt presente no diretório local para o índice:

git add temp.txt

**git clone**

O comando **git clone** é usado para fins de verificação de repositório. Se o repositório estiver em um servidor remoto, use:

git clone [alex@93.188.160.58:/path/to/repository](mailto:alex@93.188.160.58:/path/to/repository)

Por outro lado, se uma cópia de trabalho de um repositório local for criada, use:

git clone /path/to/repository

**git commit**

O comando **git commit** é usado para confirmar as alterações na cabeça. Tenha em atenção que quaisquer alterações efetuadas não irão para o repositório remoto. Uso:

git commit –m “coloque sua mensagem aqui”

**git status**

O comando **git status** exibe a lista de arquivos alterados juntamente com os arquivos que ainda não foram adicionados ou confirmados. Uso:

git status

**git push** é outro dos comandos git básicos mais usados. Um simples envio envia as alterações feitas para o ramo mestre do repositório remoto associado ao diretório de trabalho. Por exemplo:

git push origin master

**git checkout**

O comando **git checkout** pode ser usado para criar ramos ou alternar entre eles. Por exemplo, o seguinte cria um novo ramo e muda para ele:

command git checkout -b <branch-name>

Para simplesmente mudar de um ramo para outro, use:

git checkout <branch-name>

**git remote**

O comando **git remote** permite que um usuário se conecte a um repositório remoto. O comando a seguir lista os repositórios remotos atualmente configurados:

git remote –v

Esse comando permite que o usuário se conecte a um servidor remoto:

git remote add origin <93.188.160.58>

**git branch**

O comando **git branch** pode ser usado para listar, criar ou excluir ramos. Para listar todos os ramos presentes no repositório, use:

git branch

Para excluir um ramo:

git branch –d <branch-name>

**git pull**

Para mesclar todas as alterações presentes no repositório remoto para o diretório de trabalho local, o comando pull é usado. Uso:

git pull

**git merge**

O comando **git merge** é usado para mesclar uma ramificação no ramo ativo. Uso:

git merge <branch-name>

**git diff**

O comando **git diff** é usado para listar os conflitos. Para visualizar conflitos com o arquivo base, use

git diff --base <file-name>

O seguinte comando é usado para exibir os conflitos entre ramos about-to-be-merged antes de mesclá-los:

git diff <source-branch> <target-branch>

Para simplesmente listar todos os conflitos atuais, use:

git diff

**git tag**

A marcação é usada para marcar compromissos específicos com alças simples. Um exemplo pode ser:

git tag 1.1.0 <insert-commitID-here>

**git log**

Executar o comando **git log** exibe uma lista de compromissos em uma ramificação, juntamente com os detalhes pertinentes. Um exemplo de saída pode ser:

commit 15f4b6c44b3c8344caasdac9e4be13246e21sadw  
Author: Alex Hunter [< HYPERLINK "mailto:%3Calexh@gmail.com"alexh@gmail.com](mailto:%3Calexh@gmail.com)>  
Date: Mon Oct 1 12:56:29 2016 -0600

**git reset**

Para redefinir o índice e o diretório de trabalho para o estado do último commit, o comando **git reset** é usado. Uso:

git reset --hard HEAD

**git rm**

**git rm** pode ser usado para remover arquivos do índice e do diretório de trabalho. Uso:

git rm filename.txt

**git stash**

Provavelmente um dos menos conhecidos comandos git básicos é **git stash** que ajuda a salvar as mudanças que não devem ser cometidos imediatamente, mas em uma base temporária. Uso:

git stash

**git show**

Para visualizar informações sobre qualquer objeto git, use o comando **git show**. Por exemplo:

git show

**git fetch**

**git fetch** permite que um usuário obtenha todos os objetos do repositório remoto que atualmente não residem no diretório de trabalho local. Exemplo de uso:

git fetch origin

**git ls-tree**

Para exibir um objeto de árvore juntamente com o nome e o modo de cada item e o valor SHA-1 do blob, use o comando **git ls-tree**. Por exemplo:

git ls-tree HEAD

**git cat-file**

Usando o valor SHA-1, exiba o tipo de um objeto usando o comando **git cat-file**. Por exemplo:

git cat-file –p d670460b4b4aece5915caf5c68d12f560a9fe3e4

**git grep**

**git grep** permite que um usuário procure através das árvores de conteúdo frases e / ou palavras. Por exemplo, para pesquisar [www.hostinger.com](http://www.hostinger.com/) em todos os arquivos use:

git grep "[www.hostinger.com](http://www.hostinger.com/)"

**gitk**

**gitk** é a interface gráfica para um repositório local que pode ser invocado digitando e executando:

gitk

**git instaweb**

Com o comando **git instaweb**, um servidor web pode ser executado em interface com o repositório local. Um navegador da Web também é automaticamente direcionado para ele. Por exemplo:

git instaweb –httpd=webrick

**git gc**

Para otimizar o repositório através da coleta de lixo, que irá limpar arquivos desnecessários e otimizá-los, use:

git gc

**git archive**

O comando **git archive** permite que um usuário crie um arquivo zip ou tar contendo os componentes de uma única árvore de repositório. Por exemplo:

git archive --format=tar master

**git prune**

Através do comando **git prune**, os objetos que não têm ponteiros de entrada são excluídos. Uso:

git prune

**git fsck**

Para executar uma verificação de integridade do sistema de arquivos git, use o comando **git fsck**. Todos os objetos corrompidos são identificados:

git fsck

**git rebase**

O comando **git rebase** é usado para reaplicação de compromissos em outro ramo. Por exemplo:

git rebase master