|  |  |
| --- | --- |
| **Comando** | **Descrição** |
| git add <directory> | Encenou todas as mudanças no ***<diretório>*** para o próximo compromisso. Substitua <direção> por um <file> alterar um arquivo específico. |
| git branch | Liste todas as filiais em seu repo. Adicione um argumento ***<branch>*** para criar uma nova filial com o nome ***<branch>***. |
| git checkout -b <branch> | Crie e confira uma nova filial chamada <branch>. Solte a bandeira -b para verificar uma filial existente. |
| git clean -n | Mostra quais arquivos seriam removidos do diretório de trabalho. Use a bandeira -f no lugar da bandeira ***-n*** para executar a limpeza. |
| git clone <repo> | Clone repo localizado em <repo> na máquina local. O repo original pode ser localizado no sistema de arquivos local ou em uma máquina remota via HTTP ou SSH. |
| git commit --amend | Substitua o último compromisso com as alterações encenadas e o último compromisso combinado. Use sem nada encenado para editar a mensagem do último compromisso. |
| git commit -m "<message>" | Comprometa o instantâneo encenado, mas em vez de lançar um editor de texto, use ***<mesage>*** como a mensagem de compromisso. |
| git config --global --edit | Abra o arquivo de configuração global em um editor de texto para edição manual. |
| git config user.name <name> | Defina o nome do autor a ser usado para todos os compromissos no repo atual. Os devs geralmente usam ***--bandeira global*** para definir opções de configuração para o usuário atual. |
| git config --global alias. <alias-name> <git-command> | Crie atalho para um comando Git. Por exemplo ***, alias.glog "log --gráfico*** --oneline" definirá "git glog" equivalente a "git log --gráfico --oneline. |
| git config --global user.email <email> | Defina o e-mail do autor a ser usado para todos os compromissos pelo usuário atual. |
| git config --global user.name <name> | Defina o nome do autor a ser usado para todos os compromissos pelo usuário atual. |
| git config --system core.editor <editor> | Defina o editor de texto usado por comandos para todos os usuários da máquina. ***<editor>*** arg deve ser o comando que lança o editor desejado (por exemplo, vi). |
| git diff | Mostre mudanças não encenadas entre seu índice e diretório de trabalho. |
| git diff --cached | Mostrar diferença entre as mudanças encenadas e o último compromisso |
| git diff HEAD | Mostre diferença entre diretório de trabalho e último compromisso. |
| git fetch <remote> <branch> | Busca um ***<branch***> específico, do repo. Deixe de fora ***<branch>*** para buscar todos os árbitros remotos. |
| git init <directory> | Crie o Git repo vazio no diretório especificado. Corra sem argumentos para inicializar o diretório atual como um repositório git. |
| git log | Exibir todo o histórico de compromissos usando o formato padrão. Para personalização, consulte opções adicionais. |
| git log -- <file> | Somente os compromissos de exibição que têm o arquivo especificado. |
| git log <since>..<until> | Os compromissos de show ocorrem entre ***<saté>*** e ***< até>***. Args pode ser um ID de compromisso, nome de filial, ***HEAD*** ou qualquer outro tipo de referência de revisão. |
| git log --grep=”<pattern>” | Procure por compromissos com uma mensagem de compromisso que ***corresponda <pattern>***. |
| git log -<limit> | Limite o número de compromissos por ***<limite>***. |
| git log --author= ”<pattern>” | Busca por compromissos de um autor específico. |
| git log --graph --decorate | --a bandeira de gráfico desenha um gráfico baseado em texto de compromissos no lado esquerdo do msgs de compromisso. ***--decorar*** adiciona nomes de ramos ou tags de compromissos mostrados. |
| git log --oneline | Condensar cada um se comprometer com uma única linha. |
| git log -p | Exibir a difusão completa de cada compromisso. |
| git log --stat | Inclua quais arquivos foram alterados e o número relativo de linhas que foram adicionadas ou excluídas de cada uma delas. |
| git merge <branch> | Mescle ***<branch>*** na filial atual. |
| git pull <remote> | Buscar a cópia do controle remoto especificado do ramo atual e fundi-o imediatamente na cópia local. |
| git pull --rebase <remote> | Pegue a cópia do controle remoto do ramo atual e rebasee-o na cópia local. Usa rebase git em vez de mesclar para integrar os ramos. |
| git push <remote> <branch> | Empurre o ramo para<remote>, juntamente com os compromissos e objetos necessários. Cria ramo nomeado no repo remoto se ele não existir.. |
| git push <remote> --all | Empurre todas as suas filiais locais para o controle remoto especificado. |
| git push <remote> ***--force*** | Força o ***empurrão git*** mesmo que resulte em uma fusão não-rápida. Não use a bandeira ***da força*** a menos que tenha certeza absoluta de que sabe o que está fazendo. |
| git push <remote> --tags | As etiquetas não são automaticamente empurradas quando você empurra um galho ou usa a bandeira ***-tudo***. A bandeira ***-tags*** envia todas as suas tags locais para o repo remoto. |
| git rebase <base> | Rebasee o ramo atual em ***<base>***. ***<base>*** pode ser um ID de compromisso, nome da filial, uma tag ou uma referência relativa ao ***HEAD***. |
| git rebase -i <base> | Rebasear interativamente o ramo atual em ***<base>***. Lança editor para inserir comandos de como cada compromisso será transferido para a nova base. |
| git reflog | Mostre um registro de alterações no HEAD do repositório local. |
| git remote add <name> <url> | Crie uma nova conexão com um repo remoto. Depois de adicionar um controle remoto, você pode usar <name> como um atalho para <url> outros comandos. |
| git reset | Redefinir a área de preparação para corresponder ao compromisso mais recente, mas deixe o diretório de trabalho inalterado. |
| git reset <commit> | Mova a atual ponta da filial para < ***compromisso>***, reinicie a área de preparação para combinar, mas deixe o diretório de trabalho em paz. |
| git reset <file> | Remova ***<file>*** da área de encenação, mas deixe o diretório de trabalho inalterado. Isso desapresentará um arquivo sem substituir quaisquer alterações. |
| git reset ***--hard*** | Redefinir a área de encenação e o diretório de trabalho para corresponder ao compromisso mais recente e **substituir todas as mudanças** no diretório de trabalho. |
| git reset ***--hard <commit>*** | O mesmo que antes, mas redefine tanto a área de encenação quanto o diretório de trabalho para combinar. **Exclui alterações** não comprometidas e **todos se comprometem após *<compromisso>.*** |
| git revert <commit> | Criar um novo compromisso que desfaça todas as alterações feitas em ***<compromisso>*** e, em seguida, aplicá-lo ao ramo atual. |
| git status | Liste quais arquivos são encenados, não encenados e não rastreados. |

NOTA: A base do documento original para a criação desse arquivo foi retirado do site <https://www.atlassian.com/br/git/tutorials/atlassian-git-cheatsheet>, e traduzido para o português Brasil utilizando o google tradutor por mim Jociel Alves, email, jocielalvesdejesus@gmail.com, posteriormente foi formatado gerando este arquivo presente.

MPA