**init**

$ git init  
ディレクトリにリポジトリを作成.

$ git init --bare  
ベアリポジトリの作成

$ git init --shared  
グループ書き込み権限を有効にする.

**add**

$ git add [filename]  
ファイルやディレクトリをインデックスに登録.

$ git add -A  
すべての変更を含むワークツリーの内容をインデックスに追加.

$ git add -u  
以前コミットしたことがあるファイルだけインデックスに追加.

**commit**

$ git commit  
インデックスに追加されたファイルをコミットする.

$ git commit -m “[comment]”  
コミットメッセージを同時に指定.

$ git commit -a  
変更されたファイル(新規を除く)をインデックスに追加し,コミットする.

$ git commit --amend  
直前のコミットを修正する.

**status**

$ git status  
前回のコミットと比較してどのファイルが変更されたかを表示.

**log**

$ git log  
コミットログを参照する.

$ git log --oneline  
コミットログの先頭７桁のコミットIDを表示する.

$ git log --decorate  
コミットログのHEADの位置を明示する.

$ git log --graph  
コミットログを縦グラフで表示する.

$ git log --grep [filter]  
指定した文字がコミットログに含まれるコミットを表示する.

**diff**

$ git diff  
インデックスとワーキングツリーの差異を表示.  
$ git diff --cache  
HEADとインデックスの差分を表示.  
$ git diff [commit id 1] [commit id 2]  
コミット間の差異を表示する.

**checkout**

$ git checkout [commit id] [filename]  
コミットされた過去のファイルを復元する.

$ git checkout [branch]  
ブランチを変更する.

$ git checkout --ours [filename]  
マージでコンフリクトしたときに情報を指定してファイル内容を採用する.

$ git checkout --theirs [filename]  
マージでコンフリクトしたときに下方を指定してファイル内容を採用する.

**show**

$ git show  
最新のコミット内容を表示.

$ git show [tagname]  
タグを指定してコミット内容を表示.

**reset**

$ git reset [commit id]  
インデックスを現在のHEADの状態にする.

$ git reset HEAD [filename]  
インデックスからファイルをアンステージする.

$ git reset --hard [commit id]  
ワークツリーを含めたすべてをコミットIDの状態に戻す.

$ git reset --hard HEAD@{[number]}  
git reflogで確認した番号の状態に戻す.

$ git reset --hard ORIG\_HEAD  
直前の状態に戻す.

**rm**

$ git rm [filename]  
ワークツリーとインデックスからファイルを削除.

$ git rm --cache [filename]  
インデックスのファイルを削除.

**mv**

$ git mv [filename 1] [filename 2]  
ファイル名を変更(インデックスとワークツリーに同ファイル存在時)

**revert**

$ git revert [commit id]  
コミットIDのコミットを取り消す.

**rebase**

$ git rebase -i [commit id]  
コミットIDから古い順にコミットが表示され,コミットの行を消すとコミットの取り消し,先頭のpickをほかのものに置き換えるとコミットメッセージなどの編集が可能になる.

$ git rebase --abort  
直前のgit rebaseの編集を中止する.

$ git rebase --continue  
git rebaseの変更を適応する.

**clone**

$ git clone [repository PATH] [new repository PATH]  
リポジトリをコピーする.

**push**

$ git push [remote repository PATH] [branch]  
リモートリポジトリに変更を書き込む.

$ git push [remote repository] --tags  
リモートリポジトリにすべてのタグをアップロードする.

$ git push [remote repository] [tagname]  
リモートリポジトリに指定したタグをアップロードする.

$ git push [remote repository] :[branch or tagname]  
指定したブランチ,もしくはタグをリモートリポジトリから削除する.

**pull**

$ git pull [remote repository PATH] [branch]  
リモートリポジトリの変更を取り込む.

**remote**

$ git remote  
リモートリポジトリの一覧表示

$ git remote add [username] [remote repository PATH]  
名前とリモートリポジトリを関連付けする(リモートリポジトリの追加).

$ git remote rename [remoterepository] [new name]  
リモートリポジトリの名前を変更する.

$ git remote show [remote repository]  
リモートリポジトリの情報を見る.

$ git remote prune [remote repository]  
リモートリポジトリで排除されたブランチをローカルからも削除する.

**fetch**

$ git fetch [remote repository]  
リモートリポジトリの最新情報を追加する.

$ git fetch --prune  
リモートリポジトリの削除情報をローカルに更新する.

**branch**

$ git branch &[new branch]  
現在のブランチの確認.&新しいブランチを作成する.

$ git branch -a  
すべてのブランチを確認する.

$ git branch -r  
リモートブランチを確認する.

$ git branch -d [branch]  
ブランチを削除する.

$ git branch -m [branch] [new branchname]  
ブランチの名前を変更する.

$ git branch --set-upstream [my branch] [other branch]  
他のユーザーのブランチと自分のブランチを関連付ける.

**merge**

$ git merge [branch]  
現在のブランチをほかのブランチとマージする.

**tag**

$ git tag  
タグの一覧を表示.

$ git tag -n[number]@  
タグとそのメッセージ[行数指定]の一覧を表示する(行数指定なしの場合１行).

$ git tag -l [filter]  
タグを,フィルターをかけて表示する.

$ git tag [tagname]  
現在のコミットIDにタグを関連付けする.

$ git tag [tagname] [commit id]  
コミットIDを指定してタグを関連付けする.

$ git tag -a [tagname]  
現在のコミットIDにメッセージ付きのタグを関連付けする.

$ git tag -d [tagname]  
指定したタグを削除する.

**stash**

$ git stash  
現在の状態を保存する.

$ git stash save “[message]”  
メッセージ付きで現在の状態を保存する.

$ git stash list  
保存した状態の一覧を表示する.

$ git stash pop  
最新の保存状態を復元する.

$ git stash pop stash@{[numbar]}  
番号を指定して保存状態を復元する.

$ git stash apply  
保存状態をリストに残したまま最新の保存状態を復元する.

$ git stash apply stash@{[number]}  
保存状態をリストに残したまま指定した番号の保存状態を復元する.

$ git stash drop stash@{[number]}  
保存状態を削除する.

$ git stash clear  
保存状態をすべて削除する.

**reflog**

$ git reflog  
過去にHEADが指していたコミット一覧をを表示する.

$ git reflog [branch]  
ブランチを指定して過去にHEADが指していたコミット一覧を表示する.

**cherry-pick**

$ git cherry-pick [commit id]  
別のブランチのコミットを現在のブランチにコピーする.

**config**

$ git config -l  
使用されるリポジトリの設定を表示する.

$ git config --global user.name [username]  
ユーザ名の設定.

$ git config --global user.email [email address]  
メールアドレスの設定.

$ git config --global color.ui auto  
出力結果を色づけする.