# GIT - Spis podstawowych komend

**git init** - stwórz puste repozytorium w aktualnym katalogu

**git add** <folder> - dodać wszystkie zmiany z <folder> do następnego commita

**git commit** - utwórz commita ze zmianami aktualnie w "staging area", często używane z dodatkowymi przełącznikami

**git status** – pokaż, które pliki się zmieniły od ostatniego commita, a które zostały dodane (usunięte)

**git push** - wrzuca lokalne commity do zdalnego repozytorium (synchronizuje lokalne repo ze zdalnym)

**git pull** - pobiera commity ze zdalnego repozytorium do lokalnego repozytorium

**git branch <name>** - tworzy nowy branch o nazwie <name>

**git checkout <name>** - "przełącza się" na brancha <name>

**git reset --hard <commit>** - resetuje repozytorium do wersji <commit> oraz czyści working directory i staging area (czasami bezpowrotnie - zależnie od scenariuszu)

**git reset --mixed <commit>** - resetuje repozytorium do wersji <commit>, pozostawia zmiany w working directory (jest to domyślny sposób resetowania)

**git reset --soft <commit>** - resetuje repozytorium do wersji <commit>, pozostawia zmiany w staging area

**git rebase <target-commit>** - szuka commita, który był wspólnym przodkiem zarówno aktualnego brancha jak i <target-commit>, nakłada wszystkie commity, których nie ma w <target-commit> na górę <target-commit>

**git merge <target-branch>** - tworzy commita mergującego, który ma dwóch przodków, ze wspolną historią tych dwóch branchy, gdy <target-branch> ma w historii aktualnego brancha to następuje "Fast Forward", czyli następuję dodanie commitów, ale nie tworzy się commit mergujący.

**git rebase --interactive <commit>** - pozwala zamieniac ze sobą miejscami, łączyć usuwać, edytować commity od <commit> do aktualnego commita (HEAD)

**git revert <commit>** - tworzy nowego commita, który usuwa zmiany zrobione w <commit>