Git & GitHub – Kompletny przewodnik

Inicjalizacja repozytorium

git init – tworzy nowe lokalne repozytorium
git clone <url> – pobiera repozytorium z GitHub

Praca z plikami

git status – pokazuje status zmian
git add <plik> – dodaje plik do obszaru staging
git add . – dodaje wszystkie zmienione pliki
git commit -m "komentarz" – zapisuje zmiany do historii

Wysyłanie i pobieranie

git push – wysyła zmiany na zdalne repozytorium
git pull – pobiera i scala zmiany z GitHub
git fetch – tylko pobiera zmiany bez scalenia

Na GitHub można zaakceptować pull requesty przez interfejs web.

Branchowanie

git branch – pokazuje listę gałęzi
git branch <nazwa> – tworzy nową gałąź
git checkout <nazwa> – przełącza na gałąź
git switch <nazwa> – alternatywa dla checkout
git checkout -b <nazwa> – tworzy i przełącza na nową gałąź

W interfejsie GitHub można utworzyć nową gałąź przez Branches → New branch.

Merge i rebase

git merge <gałąź> – scala gałąź z obecną
git rebase <gałąź> – przepisuje historię nad gałęzią

Resetowanie commitów

git reset --soft HEAD~1 – cofa commit, zostawia zmiany w staging
git reset --mixed HEAD~1 – cofa commit i staging (opcja domyślna)

git reset --hard HEAD~1 – cofa commit i usuwa wszystkie zmiany
git revert <commit> – tworzy commit odwracający zmiany bez usuwania historii

Historia i logi

git log – pokazuje pełną historię commitów
git log --oneline – skrócona wersja historii
git diff – pokazuje różnice między wersjami

Czyszczenie i usuwanie

git clean -fd – usuwa nieśledzone pliki i katalogi
git rm <plik> – usuwa plik z repozytorium

Remote - zdalne repozytoria

git remote -v – pokazuje skonfigurowane zdalne repozytoria
git remote add origin <url> – dodaje nowe zdalne repozytorium
git push -u origin <branch> – wysyła zmiany i ustawia śledzenie gałęzi

Zaawansowane operacje na historii

git rebase -i HEAD~N – interaktywna edycja ostatnich N commitów
git push --force – wymusza nadpisanie historii zdalnego repozytorium
git push --force-with-lease – bezpieczniejsze nadpisanie historii

Funkcje interfejsu GitHub

Przez interfejs web GitHub można wykonać następujące operacje:

- Tworzenie nowych repozytoriów
- Tworzenie i zarządzanie gałęziami
- Mergowanie pull requestów
- Przeglądanie historii commitów
- Edycja plików online
- Zarządzanie współpracownikami

Staging i szczegółowe porównania

git diff – różnice między katalogiem roboczym a staging area
git diff --cached – różnice między staging area a ostatnim commitem
git diff <branch1>..<branch2> – różnice między gałęziami

git restore <plik> – przywraca plik do wersji z ostatniego commita
git restore --staged <plik> – usuwa plik z staging area

Interaktywna praca z plikami

git add -p – pozwala wybierać fragmenty plików do commita
git stash – tymczasowo odkłada zmiany
git stash pop – przywraca ostatnio odłożone zmiany

Reflog - odzyskiwanie utraconych commitów

git reflog – pokazuje historię wszystkich operacji Git
git checkout <hash> – powrót do konkretnego commita z reflog

Cherry-pick i bisect

git cherry-pick <hash> – przenosi konkretny commit na aktualną gałąź
git bisect start – rozpoczyna wyszukiwanie błędnego commita
git bisect good/bad <hash> – oznacza commit jako działający lub błędny

Podpisy GPG

git commit -S -m "msg" – tworzy podpisany commit z kluczem GPG

GitHub oznacza podpisane commity jako "Verified".

Tagowanie wersji

git tag v1.0 – tworzy tag lokalnie
git push origin v1.0 – wysyła tag na GitHub
git tag -d v1.0 – usuwa tag lokalnie
git push origin :refs/tags/v1.0 – usuwa tag ze zdalnego repozytorium

Pull Requesty i forki

Na GitHub można wykonać fork repozytorium przyciskiem "Fork" oraz utworzyć pull request przez "New pull request".

gh pr create – tworzy pull request z linii poleceń (GitHub CLI)

SSH i autoryzacja

ssh-keygen -t ed25519 – generuje klucz SSH

Publiczny klucz należy dodać w GitHub → Settings → SSH & GPG keys.

Submoduły

git submodule add <url> – dodaje inne repozytorium jako podmoduł
git submodule update --init --recursive – pobiera i inicjalizuje wszystkie podmoduły

Konfiguracja

git config --global user.name "Imię Nazwisko" – ustawienia globalne
git config --local user.email "email@example.com" – ustawienia dla konkretnego repo
git config --list – wyświetla wszystkie ustawienia

Synchronizacja z głównym repozytorium

git fetch upstream – pobiera dane z głównego repozytorium
git rebase upstream/main – przepisuje lokalną gałąź na najnowszej wersji

Workflow dla Pull Requestów

```bash
git checkout -b nowa-funkcja
# wprowadzenie zmian
git commit -m "dodano nową funkcję"
git push origin nowa-funkcja
# następnie na GitHub: Compare & pull request
```

Dobre praktyki

- Zawsze wykonuj `git pull` przed `git push`
- Unikaj `--force` w projektach zespołowych bez konsultacji
- Twórz osobne gałęzie dla nowych funkcji
- Rób commity często z opisowymi komunikatami
- Używaj znaczących nazw dla gałęzi i commitów

Przydatne aliasy

Dodaj do pliku `.gitconfig`:

ini

```ini
[alias]
  st = status
  ci = commit
  co = checkout
  br = branch
  lg = log --oneline --graph --decorate
```