Git

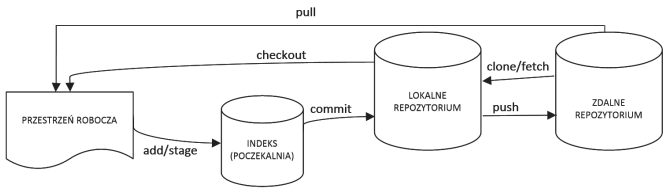
TABLICE INFORMATYCZNE • Daniel Krasnokucki

Helion

WPROWADZENIE

Przepływ

Pracując w zespole lub nad dużym projektem, chcemy mieć porządek w repozytorium kodu. Git bardzo ułatwia zapanowanie nad poszczególnymi wersjami, historią i współpracą przy tworzeniu oprogramowania. Każdy z użytkowników będzie pracował w swojej przestrzeni roboczej (pliki będzie zmieniał lokalnie), a dopiero po dodaniu zmian do indeksu i zatwierdzeniu ich do repozytorium lokalnego można wrzucić pliki na serwer. Podstawowy przepływ plików w Gicie pokazany został poniżej na rysunku.



Słownik

branch — odgałęzienie.
commit — stan projektu (migawka) w danej chwili.
indeks (ang. *staging area*) — poczekalnia. Trzyma zmiany, które zostaną zacommitowane do nowej wersji.
HEAD — wskaźnik do gałęzi, w której się obecnie znajdujesz.
katalog roboczy (ang. *working directory*) — katalog, w którym dokonujemy zmian.
master — zwyczajowa nazwa głównego odgałęzienia.
merge — scalanie/łączenie plików lub gałęzi w jedną.
obiekt — coś, co jest w repozytorium (obiektami są: blob, drzewo, commit, tag).
origin — zwyczajowa nazwa głównego repozytorium.
nazwa — nazwa obiektu, dwudziestocyfrowy skrót SHA-1 dla danego obiektu.
ORIG_HEAD — wskaźnik do obecnej gałęzi repozytorium zdalnego.
ref — referencja, wskaźnik na commit (może to być gałąź, tag itd.).
repozytorium — struktura danych zawierająca historię projektu.
repozytorium lokalne — miejsce przechowywania zacommitowanych plików.
repozytorium zdalne — wersja projektu utrzymywana na serwerze.
revision — rewizja, kolejna wersja plików w repozytorium.
tag — zrozumiała dla człowieka nazwa obiektu wraz z informacją o osobie tagującej.

PODSTAWOWE POLECENIA

Pomoc

`--help` — wyświetlenie pomocy dla dowolnego polecenia.

Przykład: `git config --help`

Konfiguracja repozytorium

`git init` — inicjalizacja repozytorium, czyli stworzenie w miejscu wywołania podkatalogu *.git* zawierającego szkielet repozytorium.
`git init --bare` — inicjalizacja repozytorium zdalnego w obecnym katalogu.
`git init <TwojaLokalizacja>` — inicjalizacja repozytorium w podanej lokalizacji.

Przykład: `git init C:\Git\TwojeRepozytorium\`

`git config` — zmiana ustawień konfiguracyjnych.
`--global` — zmiana ustawień globalnych.
`--local` — zmiana ustawień lokalnych.
`--system` — zmiana ustawień systemowych (dla wszystkich użytkowników).
`--list` — wyświetlenie obecnych ustawień.

Więcej opcji ustawień w części „Konfiguracja".

`git remote add origin https://TwojAdres/twoje_repozytorium.git` — ustawienie zdalnego repozytorium.
`git clone <zdalne_repozytorium> <lokalny_katalog>` — klon repozytorium zdalnego do katalogu lokalnego. Automatycznie ustawia *origin* na klonowane repozytorium zdalne.

Operacje w repozytorium

`git add` — dodanie plików do indeksu (pliki będą gotowe do commita).
`-a` — dodaje wszystkie pliki (nowe, zmodyfikowane, usunięte).
`.` — dodaje nowe i zmodyfikowane pliki (bez usuniętych).
`-u` — dodaje pliki zmodyfikowane i usunięte (bez nowych).
`git commit` — zatwierdzenie zmian w indeksie, dodanie do lokalnego repozytorium.

Przykład: `git commit -m "Opis zmian"`

`git mv` — przeniesienie lub zmiana nazwy pliku i zapis w indeksie.
`git rm` — usuwa pliki z katalogu roboczego.
`git status` — wyświetla informację o tym, które zmiany w katalogu roboczym zostały zapisane w indeksie, a które nie.
`-s` — wyświetla krótszy opis.

Niektóre statusy:

STATUS	SKRÓT	OPIS
untracked	??	plik nieśledzony przez Gita/niewersjonowany
modified	M	zmieniony plik
added	A	plik dodany
deleted	D	plik usunięty
copied	C	plik skopiowany
renamed	R	nazwa pliku została zmieniona
updated but unmerged	U	pliki zmienione, ale niescalone

Poprawki

`git checkout -- <nazwa_pliku>` — przywrócenie pliku do wersji z dowolnego commita.
`git reset` — usunięcie plików z indeksu, ustawienie bieżącej wersji (`HEAD`). Pliki w katalogu roboczym nie zmieniają się.
`git reset --hard` — skasowanie wszystkich zmian, przywrócenie stanu z ostatniej rewizji.
`git reset --soft` — skasowanie commitów, bez zmian w indeksie i katalogu roboczym.
`git commit --amend` — wprowadzenie zmian do ostatniej rewizji (poprawienie ostatniego commita).

Praca z gałęziami

`git branch` — wyświetla gałęzie w lokalnym repozytorium.
`git branch <nazwa>` — tworzy nowe odgałęzienie.
`git branch -d <nazwa>` — usuwa gałąź o podanej nazwie. Wersja z tej gałęzi będzie jednak nadal dostępna z innych gałęzi.
`git checkout` — przywraca pliki do stanu z ostatniego commita.
`git checkout <nazwa_gałęzi>` — zmienia aktywną gałąź na wybraną przez użytkownika.
`git checkout -b <nazwa_gałęzi>` — tworzy nową gałąź na podstawie aktywnej i przełącza się na nią.
`git merge` — scalanie dwóch lub więcej wersji z projektem.
`git merge --no-ff` — scalenie zmian ze szczegółowym opisem w specjalnie utworzonym commicie.
`git mergetool` — uruchamia narzędzie do scalania plików.
`git stash` — zapamięta wszelkie niezatwierdzone zmiany i przywróci stan kopii roboczej do ostatniego commita (ukryje nasze zmiany w skrytce).
`git stash apply` — polecenie odwrotne do poprzedniego. Odkrywa schowane pliki.
`git stash pop` — przywrócenie ze skrytki *stash* poprzedniego stanu katalogu roboczego i indeksu.
`git stash drop` — usunięcie zapamiętanego stanu ze stosu skrytki.
`git stash show` — wyświetlenie zmian w indeksie oraz w plikach w najnowszym wpisie w skrytce.
`git stash list` — wyświetlenie stosu przechowywanych w skrytce kontekstów, zaczynając od najnowszego.
`git tag` — wyświetla wszystkie dostępne etykiety.

Przykład: `git tag -l 'v1.1.2.*'` — wyszukiwanie etykiet wersji 1.1.2.

Aktualizacja projektu

`git fetch` — pobieranie zmian z odpowiednich gałęzi zdalnych do gałęzi lokalnych je śledzących bez scalania zmian.
`git pull` — pobranie zmian ze zdalnego repozytorium do siebie. Próbuje automatycznie scalić (wmergować) zmiany.
`git push` — zapis śledzonych gałęzi do zdalnego repozytorium.
`git push -f` — wysyła zmiany do zdalnego repozytorium, ignorując konflikty; oznacza to, że jeśli wystąpią konflikty, pliki zostaną nadpisane obecną wersją.
`git push [remote-name] [branch-name]` — zapisuje konkretne odgałęzienie w repozytorium zdalnym.

Przykład: `git push origin master`

`git remote` — wyświetlanie listy repozytoriów zdalnych.
`git submodule add` — dodanie zewnętrznego projektu jako modułu zależnego.