Przydatne polecenia GIT

basics

- **git init** - inicjalizuje repozytorium GIT w katalogu
- **git clone {adres repozytorium}** - klonuje repozytorium do katalogu
- **git status** - pokazuje status repozytorium (pokazuje informację o zmodyfikowanych, nowych, usuniętych oraz nie należące do repozytorium plikach)
- **git add {ścierzka do pliku}** - dodaje plik do repozytorium (np. `git add folder/plik.php`)
- **git add -A** - dodaje wszystkie nie należące do repozytorium pliki
- **git config --global color.ui auto** - włącza koloryzowanie wyników w konsoli
- **git config --global core.pager '{nazwa}'** - ustawia program do przeglądania logów (brak w konsoli)

repo

- **git fetch -p** - kasuje branche już nie istniejąca na głównym repo
- **git fetch {nazwa remota}** - pobiera listę zmian z innego repozytorium (w tym pokazuje nowe gałęzie)
- **git remote -v** - lista repo
- **git remote remove {repo}** - usuwa wskazane repo
- **git remote set-url {nazwa repo} {url}** - zmienia adres dla podanego repo
- **git remote add {jakaś nazwa} {adres repozytorium}** - dodaje repozytorium innego użytkownika (`git remote add upstream https://github.com/bluetree-service/idylla.git`)
- **git remote -v** lista wszystkich zewnetrznych repozytoriów
- **git remote rm {nazwa dla remota}** - usuwa zewnętrzne repozytorium
- **git pull** - pobiera zmiany z aktualnej gałęzi
- **git pull {nazwa gałęzi}** - pobiera zmiany z wybranej gałęzi
- **git pull {nazwa remota} {nazwa gałęzi}** - pobiera zmiany z wybranej gałęzi wybranego zewnętrznego repozytorium
- **git pull --all --prune** - ściąga zmiany z repo + kasuje nieużywane branche
- **git pull --tags** - ściąga tagi
- **git push** - wypycha zmiany na aktualnie wybraną gałąź
- **git push {nazwa gałęzi}** - wypycha zmiany na wskazaną gałąź

- **git push {nazwa remota} {nazwa gałęzi}** - wypycha zmiany na gałąź wskazanego repozytorium
- **git push --tags** - wysyła tagi na repo
- **git revert -- {plik}** - revert pojedyńczego pliku
- **git reset --soft HEAD^** - cofa zmiany bez usuwania dodanych plików
- **git reset --soft {numer commita}** - cofa zmiany bez usuwania dodanych plików do wskazanego commita (`git reset --soft b87dcea`)
- **git reset --hard {numer commita}** - cofa zmiany włącznie z usunięciem plików do wskazanego commita (`git reset --hard b87dced`)
- **git reset --merge ORIG_HEAD** - resetuje zmiany z ostatniego merg-a
- **git checkout -- {plik}** - przywraca oryginalny plik
- **git checkout {commit} -- {plik}** - przywraca stan pliku ze wskazanego commita
- **git ls-files** - lista plików z ich ścierzkami w repo (-md + zmodyfikowane i usunięte)
- **git rm {plik}** - kasuje z repo plik
- **git rm --cached (plik/katalog)** - usuwa plik/katalog z repozytorium, pozostawiając go na dysku **-r** - dla całych katalogów

commit

- **git commit** - tworzy commita z aktualnie zmienionych plików
- **git commit -m "wiadomosc"** - tworzy commmita z podaną w cudzysłowach wiadomością
- **git commit --amend -m "{wiadomość}"** - umożliwia zmianę ostatniego commita
- **git commit --amend -m "dsfsdf"** - modyfikuje komentarz ostatniego commita
- **git commit --date="2017-08-18T13:23:41" -m ""** - comit ze wskazaną datą
- **git commit -n** - pomija git hooks
- **git revert {numer commita}** - tworzy nowego commita z cofnięciem zmian ze wskazanego commita
- **git amend** - zmienia poprzedniego commita
- **git shortlog -sn** - ile commitów zrobionych przez userów

log

- **git log** - wyświetla listę commitów (od najnowszego)

- **git log -{numer}** wyświetla podaną liczbę ostatnich commitów
- **git log --oneline** - wyświetla commity w postaci skróconej
- **git log -{numer} --oneline** wyświetla podaną liczbę ostatnich commitów w postaci skróconej
- **git log --graph --decorate --oneline** - pokazuje graficzny obraz zmian
- **git log --author={nazwa użytkownika}** - pokazuje commity danego użytkownika
- **git shortlog** - lista commitów użytkowników
- **git shortlog -s -n** - lista użytkowników repozytorium
- **git llog --after=2016-08-16 --before=2016-08-30** - commity z podanego zakresu (--until starsze od podanej daty, --since z pred podanej daty)
- **git log --name-status** - podaje statuz zmian przy nazwie pliku (add, mod, delete)
- **git log --stat** - + statystyki zmian w plikach (--shortstat bez ++++---)
- **git log {commit1}..{commit} --no-merges** - pokazuje zmiany pomiędzy 2 commitami bez info o mergach
- **git log -- {plik/katalog}** - log dla pojedyńczego pliku lub wszystkich plików z katalogu
- **git log -5 --pretty=tformat: --numstat** - satystyki zmian w 5 commitach
- **git log --no-merges --pretty=format:'%C(yellow)%h %Cred%ad %Cblue%an%Cgreen%d %Creset%s' --date=iso** -
- **git log --pretty=format:'* %s (%an)' -n 10** - pokazuje tylko nazwy commitów
- **git log --pretty=oneline -15 | awk '{print $2}' | sort | uniq | grep -i {ticket} | sed 's/[(.*)]/\1/g'** - pokaże tylko nazwy ticketów (gdy message zgody z formatem [NAME-111] some message)
- **git log --grep {nazwa}** - szuka commita zawierającego podany tekst
- **git log --author={autor} --name-only** - pokazuje commity wykonane przez autora wraz ze zmodyfikowanymi plikami
- **git log master..develop** - pokazuje różnicę między branchami
- **git log --pretty=format:'%Cred%h%Creset %C(bold blue)<%an>%Creset%C(yellow)%d%Creset %Cgreen(%cr)%Creset%n%w(80,8,8)%s' --graph** - drzewko logów
- **git log --pretty=format:'%C(yellow)%h %Cred%ad %Cblue%an%Cgreen%d %Creset%s' --date=iso** - pokazuje logi (hash, data+czas, autor, opis)
- **log --pretty=format:'%C(yellow)%p..%h %C(white dim)%cd %<|(49,trunc)%an %C(reset)%s' --date=short --abbrev=8 --no-merges** - logi z zakresem branchy

- **git log --oneline {branch1} --not {branch2}** - pokazuje różnice w commitach między branchami (branche których brakuje w branch2 a są w branch1)
- **git log --oneline --grep {branch} --name-only | grep -v {branch} | sort | uniq** - pokazuje tylko zmienione pliki dla podanego brancha

merge

- **git merge {nazwa gałęzi}** - dołączenie zmian ze wskazanej gałęzi
- **git merge {nazwa remota}/{nazwa gałęzi}** - dołączenie zmian ze wskazanego remota i gałęzi
- **git merge --abort** - przerywa łączenie (możliwe, gdy wystąpią konflikty)
- **git merge --continue** - po rozwiązaniu konfliktów zapisuje zmiany
- **git merge --revert** - cofa wszystkie wprowadzone zmiany

rebase

- **git rebase {nazwa gałęzi}** - dołączenie zmian ze wskazanej gałęzi z zachowaniem kolejności wprowadzania zmian
- **git rebase {nazwa remota}/{nazwa gałęzi}** - dołączenie zmian ze wskazanego repozytorium i gałęzi z zachowaniem kolejności wprowadzania zmian
- **git rebase --abort** - przerywa łączenie (możliwe, gdy wystąpią konflikty)
- **git rebase --continue** - po rozwiązaniu konfliktów zapisuje zmiany
- **git rebase --interactive {commit}** - pozwala wybrać commity które zostaną dołączone (lub modyfikować)
- **git rebase --interactive '{hash}^'** - umożliwia edycję commitów do podanego hasha

diff

- **git diff --name-only {gałąź 1} {gałąź 2}** - porównanie dwóch gałęzi
- **git diff --cached** - pokazuje wszystkie gotowe do commitu zmiany
- **git diff --cached | grep -wi {fraza}** - szuka podanej frazy w commicie
- **git diff --cached | grep -wiHn -C 10 {fraza}** - jw ale pokazuje 10 lini przed i po znalezieniu + numery linni i nazwę pliku
- **git diff --name-only HEAD HEAD~14** - pokazuje zmieniony pliki z 14 ostatnich commitów
- **git diff {commit1}..{commit2}** - pokazuje różnicę między 2 commitami

- **git diff {commit1}..{commit2} {plik}** - pokazuje różnicę między 2 commitami dla podanego pliku
- **git diff {commit} -- plik** - pokazuje zmiany w pliku od podanego commita
- **git diff-index --name-only --cached --diff-filter=ACMR HEAD**
- **git diff {commit}** - różnica od podanego commita
- **git diff {gałąź 1} {gałąź 2} -- {plik}** - dif dla pojedynczego pliku między gałęziami
- **git diff {plik}** - pokazuje zmiany dokonane na pliku (nie zacommitowane)
- **git diff .** - pokazuje zmiany dokonane na wszystkich zmienionych plikach
    - **--color-words** - pokaże bez +/-
- **git diff -p -R --no-color | grep -E "^(diff|(old|new) mode)" --color=never | git apply** - resetuje zmiany w atrybutach plików

show

- **git show {commit}** - szczegóły podanego commita
- **git show --name-only {commit}** - nazwy zmodyfikowanych plików w commicie
- **git show --name-only {commit}** - pokazuje tylko listę zmodyfikowanych plików z commita
- **git show {commit}** - pokazuje zmiany w commicie
- **git show HEAD:{plik}** - pokazuje zmiany tylko w konkretnym pliku
- **git show {commit} --name-only -p -5** - pokazuje 5 poprzednich comitów od podanego
    - **--color-words** - pokaże bez +/-

branch

- **git branch** - lista gałęzi w repozytorium
- **git branch -a** - pokazuje listę wszystkich gałęzi (łącznie z tymi z repo, same z repo `-r`)
    - **-r** - tylko gałęzie zdalne
- **git branch -d {nazwa gałęzi}** - usuwa wskazaną gałąź
- **git branch --merged** - lista zmergowanych branchy
- **git branch --merged | git branch -d** - kasuje wszystkie zmergowane branche
- **git branch -r | awk '{print $1}' | egrep -v -f /dev/fd/0 <(git branch -vv | grep origin) | awk '{print $1}' | xargs git branch -d** - kasuje wszystkie nie używane/ nie istniejące branche
- **git branch rename {1} {2}** - zmiana nazwy brancha

- **git branch | grep -v "master" | xargs git branch -D** - kasuje wszystkie branche z wyjątkiem mastera
- **git checkout {nazwa gałęzi}** - przełącza na podaną gałąź
- **git checkout -b {nazwa gałęzi}** - tworzy nową gałąź o podanej nazwie i automatycznie przełącza się na niego
- **git checkout -b {nazwa gałęzi} {nazwa remota}/{nazwa gałęzi}** - tworzy nową gałąź o podanej nazwie, pobiera zmiany ze wskazanego repozytorium i gałęzi i automatycznie przełącza się na niego
- **git checkout {nazwa pliku}** - cofa zmiany na podanym pliku
- **git branch rename {stara nazwa} {nowa nazwa}** - zmiana nazwy brancha
- **git branch -m {stara nazwa} {nowa nazwa}** - zmiana nazwy brancha

stash

- **git stash** - zapisuje nowe i zmodyfikowane pliki do pamięci podręcznej
- **git stash pop** - przywraca zapisane pliki z pamięci podręcznej
- **git stash pop --index 1**
- **git stash pop --index 454aa619**
- **git stash pop --index stash@{1}**
- **git stash pop 1**
- **git stash pop 454aa619**
- **git stash pop stash@{1}**
- **git stash save "{tekst komentarza}"** - zapisuje stash z komentarzem
- **git stash show stash@{1}**- pokazuje zachowane zmiany
- **git stash list** - lista zachowanych zmian
- **git stash branch {name}** - stworzy nową gałąź, pobierze ostatnią wersję plików
- **git stash push -m {message} {plik}** - stashuje z komentarzem wskazany plik

tag

- **git tag -l** - lista tagów
- **git tag -a {} -m '{}'** -
- **git tag --sort=v:refname | tail -2 | xargs printf "%s..%s" | xargs git log --no-merges --pretty=format:%s** - wyświetla wszystkie commity między 2 ostatnimi tagami (| grep -o "SOC-[0-9]*" | sort --unique** - tickety)
- **git tag -l "{pattern}"** - lista tagów pasująca do wzorca

- **git tag -d {tag} && git push origin :refs/tags/{tag}** - kasuje taga lokalnie + repo
- **git lasttag** - pokazuje ostatniego taga

Inne

- **git reset --soft HEAD~3**; **git commit -m** - pozwala na cofnięcie się 3 commity do tyłu, i połączenie ich w jeden (git commit --amend)
- **git rebase -i {commit}** - j/w ale commity wybierane ręcznie
- **git log -i -1 --pretty="format::%an <%ae>\n" --author="$1"** - info o userze
- **git show -s --pretty='tformat::%h (%s, %ad)' --date=short** - info o branchu
- **git log -a --pretty=oneline | wc -l** - ilość commitów
- **git fetch && git log --oneline HEAD..origin/$1** - ostatnie zmiany na podanym branchu
- **git shortlog HEAD..origin/$0** - kto ostatnio robił zmiany i jakie
- **for branch in** `git branch -r | grep -v HEAD`**;do echo** `git show -s --format=\"%Cred%ci %C(green)%h %C(yellow)%cr %C(magenta)%an %C(blue)\" $branch | head -n 1` **\\t$branch; done | sort -r** - jakie branche sa na orginie, jak dawno i kto je tworzył

Extra