GIT

System kontroli wersji

Krzysztof Dziuban

GIT - Agenda

- Rys historyczny
- Podstawowe komendy
- Poznanie popularnych serwisów
- Zaawansowana obsługa
- Nakładki graficzne
- GitFlow

Autor: Krzysztof Dziuban

GIT – podstawowa zasada

Autor: Krzysztof Dziuban

Rys historyczny i założenia

- 7.04.2005 – wydanie pierwszej wersji
- Stworzony przez Linusa Torvaldsa (w 3 dni!)
- Używany w rozwoju Linuxa, jako zastępstwo dla płatnego BitKeepera

Założenia:

- Przykład CSV czego nie robić
- Powinien być szybki
- Powinien być chroniony przed błędami w repozytorium
- Powinien być szybki

Autor: Krzysztof Dziuban

Podstawowe terminy

Repozytorium

Obiektowa baza projektu przechowująca wszystkie pliki i foldery projektu oraz dane gita (informacje o wersjach, commitach itp).

Commit

Pojedyncze zatwierdzenie zmian wprowadzonych do repozytorium

Branch

Odbicie od głównego pnia linii rozwoju

Autor: Krzysztof Dziuban

Rodzaje repozytoriów

Repozytorium lokalne

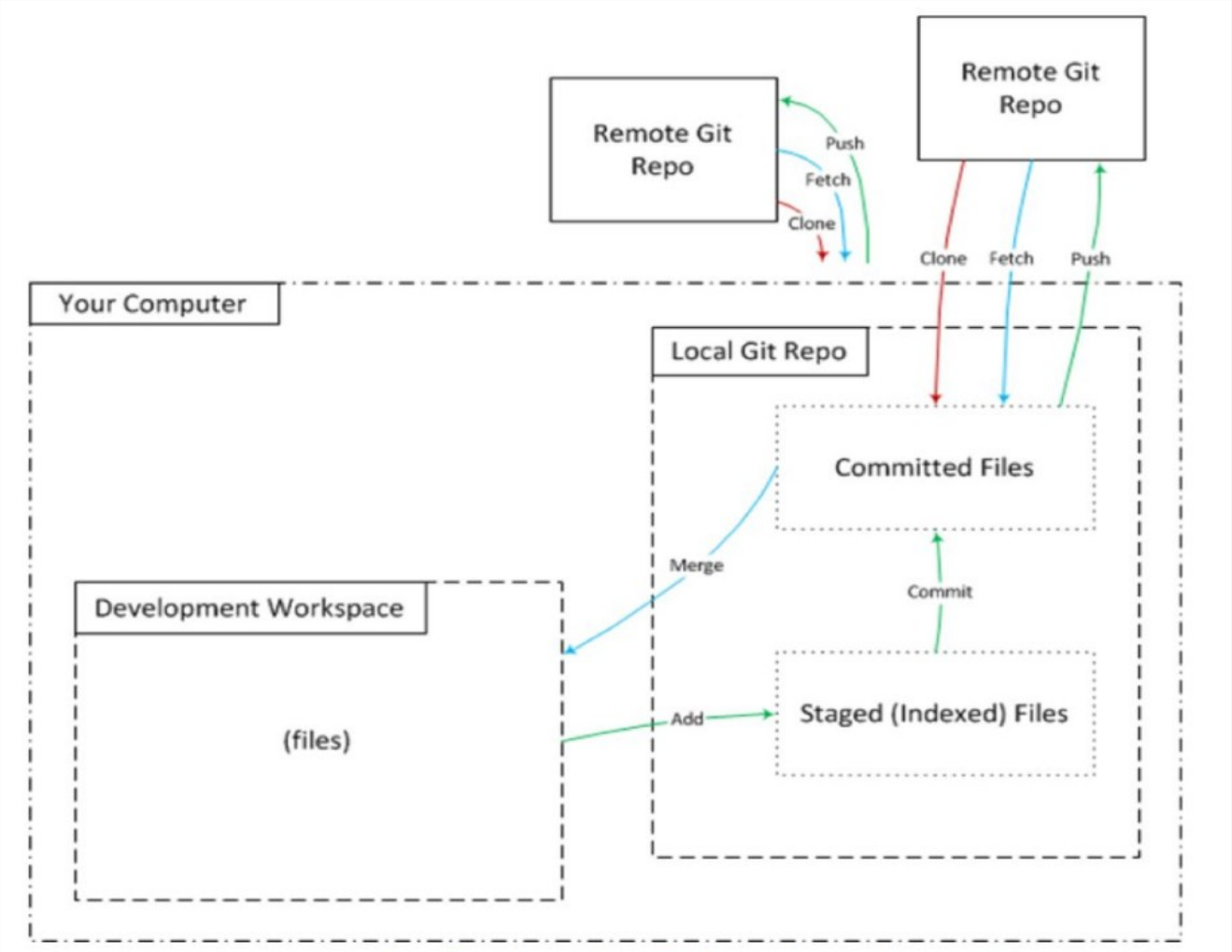
- Przechowywane na komputerze użytkownika

Repozytorium zdalne

- Hostowane w sieci (GitHub, GitLab, Bitbucket itp)

Autor: Krzysztof Dziuban

GIT – schemat działania



Źródło: https://www.intertech.com/Blog/introduction-to-git-concepts/

Autor: Krzysztof Dziuban

Instalacja

Pliki instalacyjne dostępne na:

https://git-scm.com/downloads

Dokumentacja:

https://git-scm.com/doc

Autor: Krzysztof Dziuban

Inicjalizacja lokalnego repozytorium

git init

Utworzenie lokalnego repozytorium w wybranym katalogu

- Używa się tylko raz!

git config

Konfiguracja środowiska

- git config --global user.name USER_NAME
- git config --global user.email USER_EMAIL

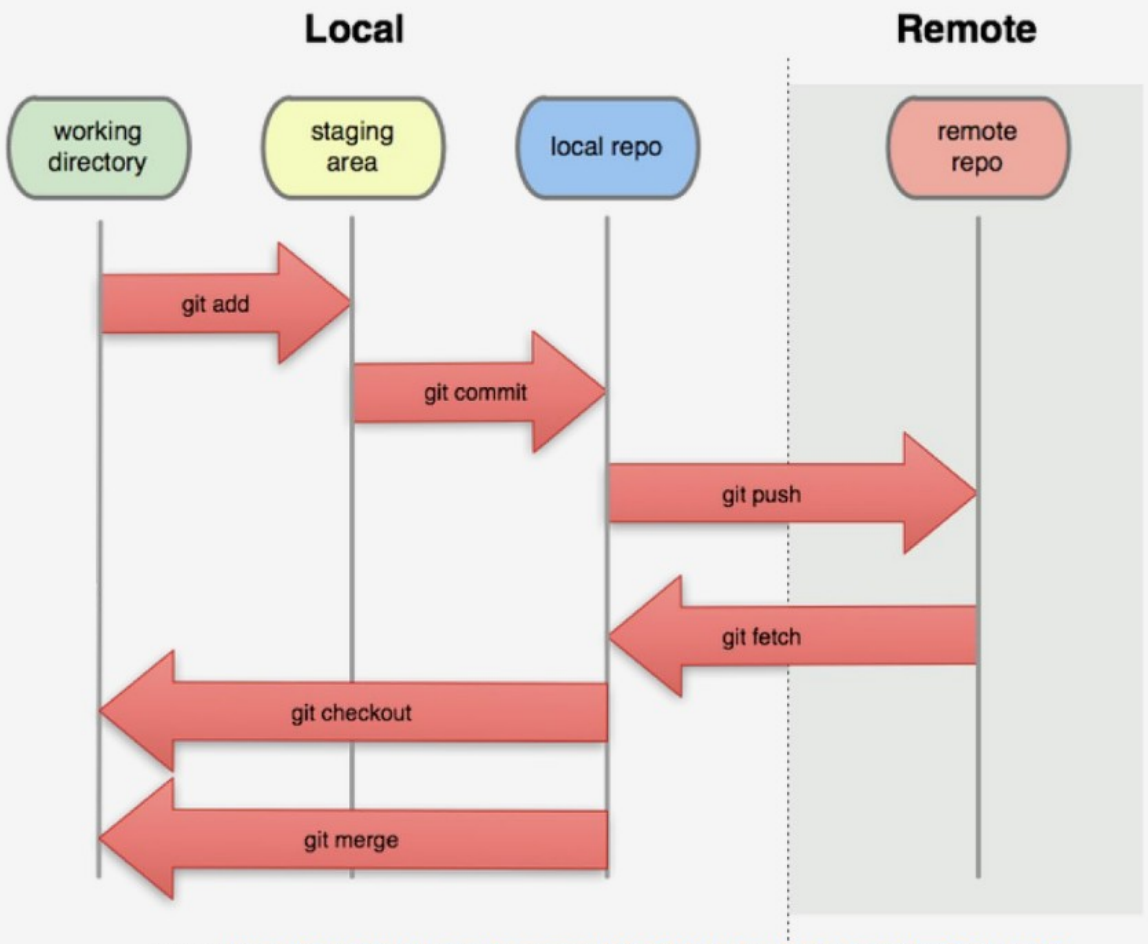
Autor: Krzysztof Dziuban

Ćwiczenie 1

1) Zainstalować git
2) Sprawdzić poprawność instalacji i wersję (git --version)
3) Utworzyć repozytorium w wybranym katalogu (git init)
4) Ustawić wybrane user_name i user_email (git config)

Autor: Krzysztof Dziuban

GIT – idea przechowalni



Źródło: https://greenido.files.wordpress.com/2013/07/git-local-remote.png?w=696&h=570

Autor: Krzysztof Dziuban

Zarządzanie plikami

git status

Sprawdzenie statusu projektu (lista zmodyfikowanych i nowych plików)

git log

Wyświetlenie listy commitów

git add

Dodawanie plików(zmian) do przechowalni

- git add [wyrażenie]

git reset

Usunięcie plików z przechowalni

- git add [wyrażenie]

Autor: Krzysztof Dziuban

Zarządzanie plikami cd.

git checkout -- <file_name>

Usunięcie zmian w pliku

git checkout <branch_name>

Przełączenie na istniejącego brancha

- git checkout -b <branch_name> - tworzy nowy branch

git commit -m “MESSAGE”

Dodawanie zmian do lokalnego repozytorium

- git add [wyrażenie]

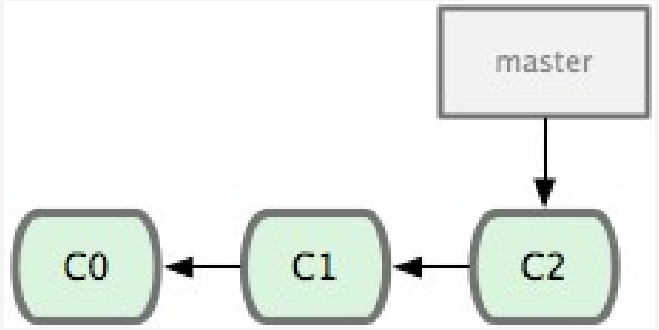
Autor: Krzysztof Dziuban

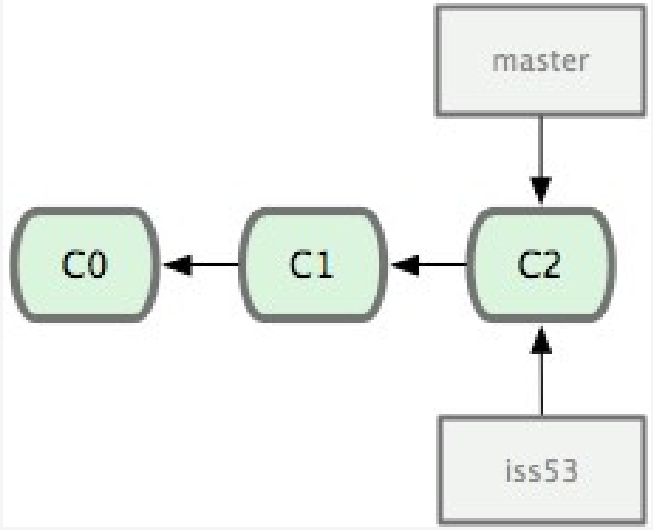
Ćwiczenie 2

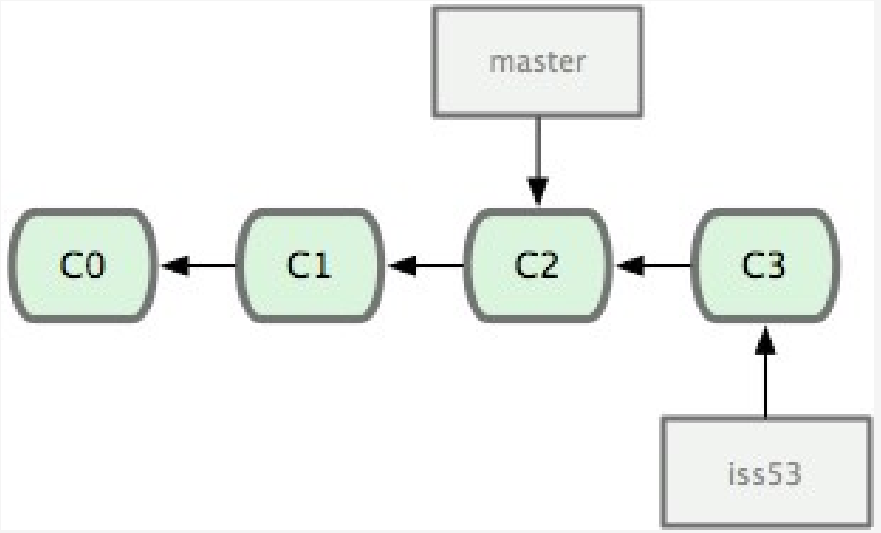
1) Sprawdzić stan repozytorium (git status)
2) Wylistować commity (git log)
3) Utworzyć plik README.txt
4) Ponownie sprawdzić stan repozytorium
5) Dodać plik do przechowalni (git add)
6) Usunąć plik z przechowalni (git reset)
7) Ponownie dodać plik do przechowalni i dodać commit (git commit)
8) Sprawdzić log

Autor: Krzysztof Dziuban

Podstawy rozgałęziania

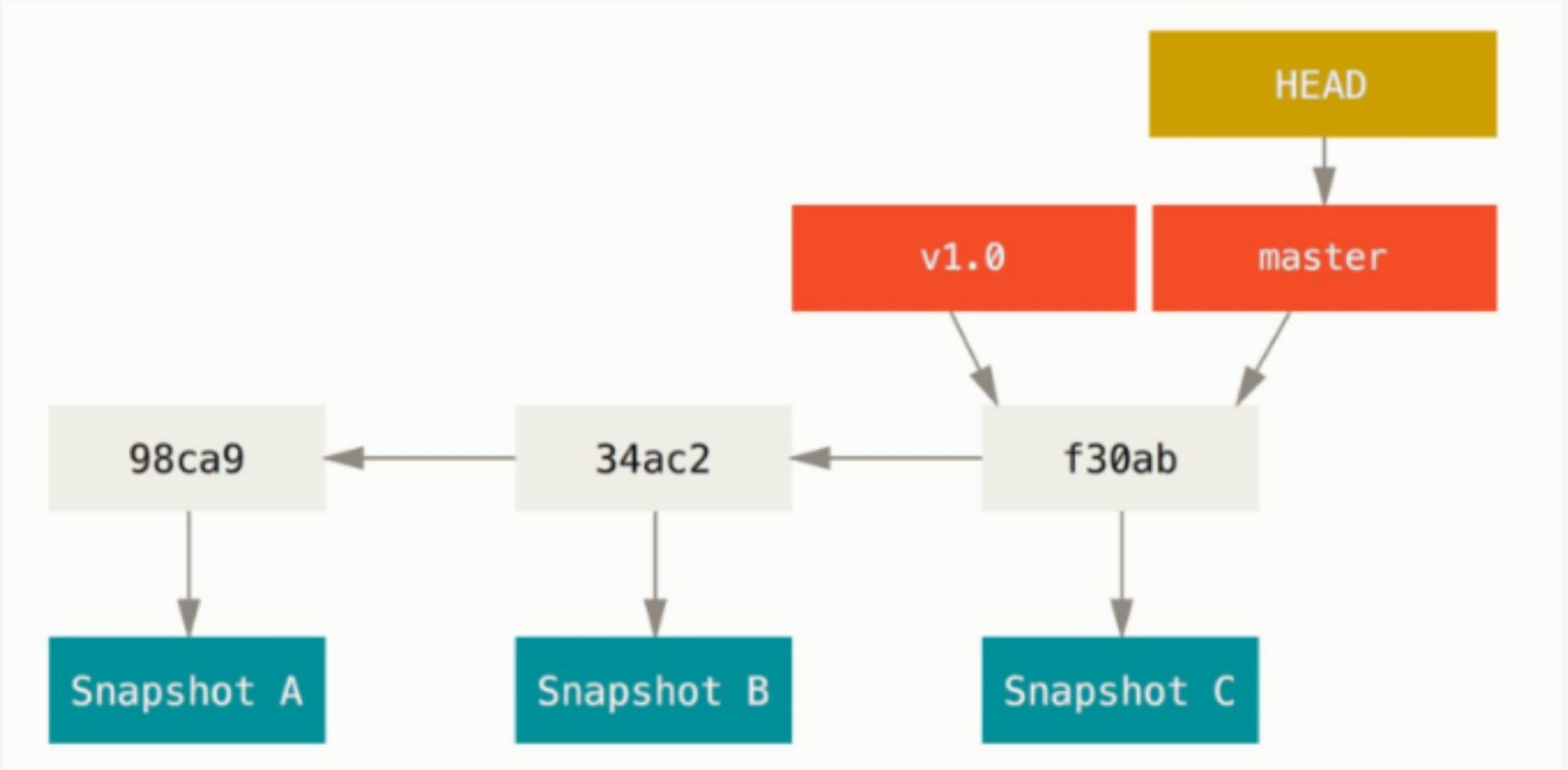






Autor: Krzysztof Dziuban

Czym są branche?



Źródło: https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell

Autor: Krzysztof Dziuban

HEAD vs head

- HEAD – wskaźnik na branch na której jesteśmy
- head – wskaźnik na ostatni commit (tip) na branchu

Jest tyle headów ile branchy, ale tylko jeden HEAD!!

Autor: Krzysztof Dziuban

Podstawy rozgałęziania cd.

git branch

Wyświetla listę lokalnych branchy

git branch -r

Wyświetla listę zdalnych branchy

git branch <branch_name>

Tworzy lokalnie branch

git branch -d <branch_name>

Usuwa lokalnie branch

- git branch -D <branch_name> - force delete!!

Autor: Krzysztof Dziuban

Ćwiczenie 3

1) Wyświetlić listę branchy (git branch)
2) Dodać nowy branch (git branch second)
3) Przełączyć się na utworzony branch (git checkout second)
4) Ponownie wyświetlić listę branchy i sprawdzić aktywny
5) Zmodyfikować plik README.txt i dodać do przechowalni
6) Wykonać commit (git commit)
7) Sprawdzić status repozytorium

Autor: Krzysztof Dziuban

Schowek

git stash

Wrzuca zmiany do schowka

git stash apply

Przywraca zmiany

git stash list

Wyświetla listę zmian ze schowka

git stash drop <stash>

Usuwa wybrany stash (jeśli brak nazwy, usuwa ostatni)

- git stash clear - czyści całą listę

Ćwiczenie 4

1) Zmodyfikować plik README.txt
2) Dodać plik temp.txt
3) Sprawdzić status
4) Wrzucić zmiany do schowka (git stash) i ponownie sprawdzić status
5) Sprawdzić listę ze schowka (git stash list)
6) Usunać ostatnie zmiany ze schowka
7) Powtórzyć punkty 1-5 (z wyłączeniem 2), tym razem przywrócić zmiany (git stash apply)

Autor: Krzysztof Dziuban

Merge i konflikty

git merge <branch_name>

Łączy aktualny branch z branch_name

Uwaga!!! Możliwe konflikty!!

Ćwiczenie 5

1) Stworzyć branch ala
2) Sprawdzić, czy aktualny branch to master
3) Dodać plik ala.txt
4) Zacommitować go
5) Przełączyć się na branch ala
6) Dodać ponownie plik ala.txt i zacommitować
7) Przełączyć się z powrotem na master i wykonać merge (git merge ala)
8) Rozwiązać konflikt i ponownie zacommitować plik (git add […] && git commit)
9) Sprawdzić logi (powinien być dodatkowy commit z mergem)

Autor: Krzysztof Dziuban

Praca ze zdalnym repozytorium

git remote -v

Wyświetla linki do zdalnego repozytorium

git remote add origin <link>

Dodawanie urla do zdalnego repozytorium

git remote set-url origin <link>

Ustawianie urla do zdalnego repozytorium

Autor: Krzysztof Dziuban

Praca ze zdalnym repozytorium cd.

git clone <link>

Sklonowanie(ściągnięcie) zdalnego repo do lokalnego

git fetch

Sprawdzenie różnic pomiedzy repozytorium zdalnym i lokalnym

git pull

Ściągnięcie danych z repozytorium zdalnego do lokalnego

Autor: Krzysztof Dziuban

Ćwiczenie 6

1) Założyć konto na http://www.github.com
2) Stworzyć repozytorium git_tutorial z plikiem README.md
3) Dodać link do repozytorium zdalnego do lokalnego (git remote)
4) Zaktualizować repozytorium lokalne (git pull)

Autor: Krzysztof Dziuban

Rodzaje git push

git push

Przesłanie zmian do repozytorium zdalnego

git push -f

Nadpisywanie zdalnego repozytorium (tylko dla konkretnego brancha)

git push --set-upstream origin <branch_name>

Przesłanie zmian do zdalnego repozytorium z utworzeniem brancha

git push --delete <branch_name>

Usuwa branch na zdalnym repozytorium

Autor: Krzysztof Dziuban

Ćwiczenie 7

1) Przesłać dotychczasowe zmiany do zdalnego repozytorium (git push)
2) Stworzyć nowy branch my_new_branch i przełączyć się na niego
3) Dodać dowolny nowy plik, zacommitować lokalnie
4) Spróbować ponownie przesłać do zdalnego repozytorium
5) Przesłać do zdalnego repo z utworzeniem nowego brancha

Autor: Krzysztof Dziuban

Etykiety

Tag – odnośnik do specyfinczego punktu w historii projektu

Rodzaje tagów:

- lightweight – podstawowy typ taga, zalecany jako prywatny
- annotated – rozszerzony tag, zawierający dodatkowe informacje (nazwa osoby tagującej, email, data itp)

Autor: Krzysztof Dziuban

Etykiety cd.

git tag

Wyświetla listę tagów

git tag –l <wildcard>

Umożliwia wyszukiwanie po nazwach z wykorzystaniem metaznaków

git tag <tag_name>

Utworzenie lightweight tag

git tag –a <tag_name>

Utworzenie annotated tag

Autor: Krzysztof Dziuban

Etykiety cd.

git tag <tag_name> <commit_hash>

Umożliwia przypisanie taga do konkretnego commita

git tag –f <tag_name> <commit_hash>

Umożliwia zastąpienie taga innym

git checkout <tag_name>

Przełączenie się na konkretnego taga

Uwaga !! – wykonane commity będą odłączone (dostępne będą tylko poprzez odwołanie do hasha – powinno się utworzyć nowego brancha przed commitowaniem zmian)

Autor: Krzysztof Dziuban

Etykiety cd.

git tag –d <tag_name>

Usuwa taga

git push origin <tag_name>

Przesyła taga do zdalnego repozytorium

git push --tags

Przesyła zmiany wraz z wszystkimi tagami

Autor: Krzysztof Dziuban

Ćwiczenie 8

1) Utwórzyć tag v0.9 (git tag)
2) Wylistować obecne tagi
3) Dodać nowy plik, zacommitować
4) Dodać annotated tag v1.0 (git tag –a)
5) Ponownie wylistować tagi, wyświetlić tylko ostatnio utworzony
6) Dodać tag v0.8 dla jednego z wcześniejszych commitów
7) Zastąpić tag v0.8 tagiem v0.8a
8) Podpiąć się pod taga v0.8a
9) Wrócić do master branch i usunąć tag v0.8a
10) Wykonać push wpierw dla taga v1.0, a potem dla pozostałych

Autor: Krzysztof Dziuban

Rozwiązywanie problemów

git blame <file_name>

Wylistowanie zawartości pliku, wraz z commitami

git blame –L min,max <file_name>

Wyświetlenie linii z podanego zakresu

git blame –e <file_name>

Podmienia username na email

Autor: Krzysztof Dziuban

Rozwiązywanie problemów cd.

git blame –w <file_name>

Ignoruje zmiany związane z białymi znakami

git blame –M <file_name>

Wykrywa przeniesione lub skopiowane linie i wyświetla oryginalnego autora

git blame –C <file_name>

Wykrywa przeniesione lub skopiowane linie z innych plików i wyświetla oryginalnego autora

Autor: Krzysztof Dziuban

Inne przydatne komendy

git show

Pokazuje zawartość ostatniego commita

git show <commit_hash>

Pokazuje zawartość wybranego commita

git blame –C <file_name>

Wykrywa przeniesione lub skopiowane linie z innych plików i wyświetla oryginalnego autora

Autor: Krzysztof Dziuban

Inne przydatne komendy cd.

git cherry-pick <commit-hash>

Przenosi commita na aktualnego brancha

git rebase

Zmienia bazę brancha

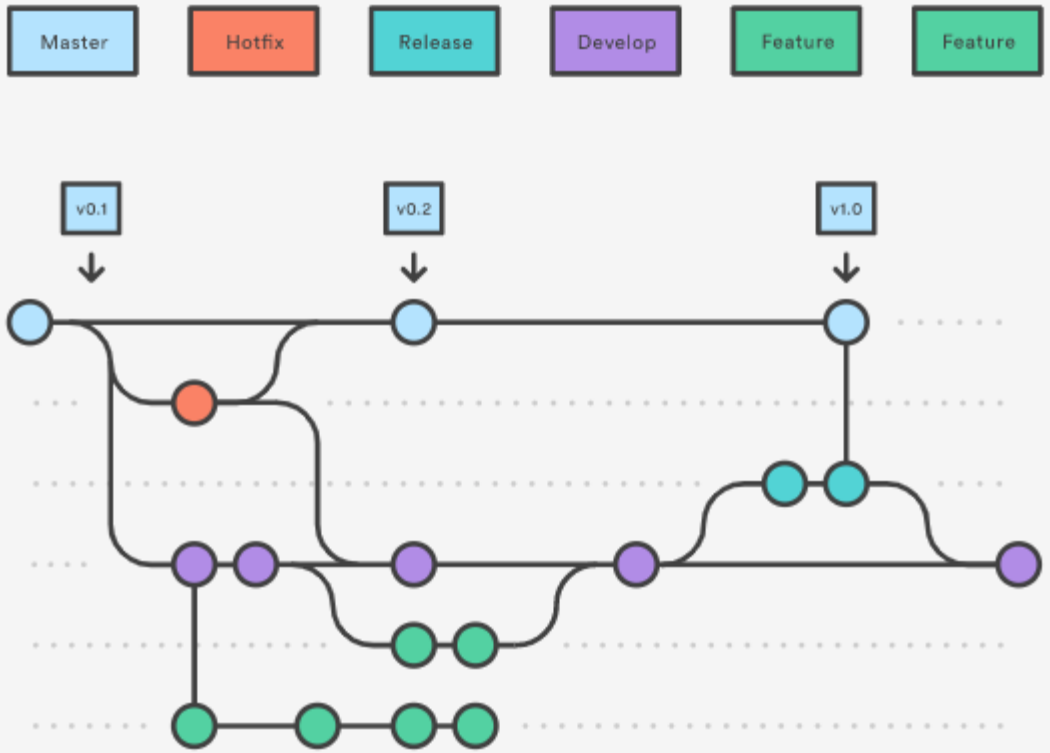
git rebase –i

Umożliwia edycję historii

Autor: Krzysztof Dziuban

GitFlow

GitFlow – metodyka pracy oparta o tworzenie określonego typu branchy



Autor: Krzysztof Dziuban

GitFlow

Typy branchy:

- master – główny branch
- develop – podstawowy branch developerski
- feature – branch poświęcony konkretnej historyjce/elementowi aplikacji
- task – branch poświęcony konkretnemu zadaniu
- release – branch poświęcony konkretnej wersji
- hotfix – branch poświęcony konkretnej poprawce

Autor: Krzysztof Dziuban

Dodatkowe tutoriale

https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud

Autor: Krzysztof Dziuban

Dziękuję za uwagę

Autor: Krzysztof Dziuban