

# GIT WARSZTATY

## Wprowadzenie do GIT



[www.gitwarsztaty.pl](http://www.gitwarsztaty.pl)

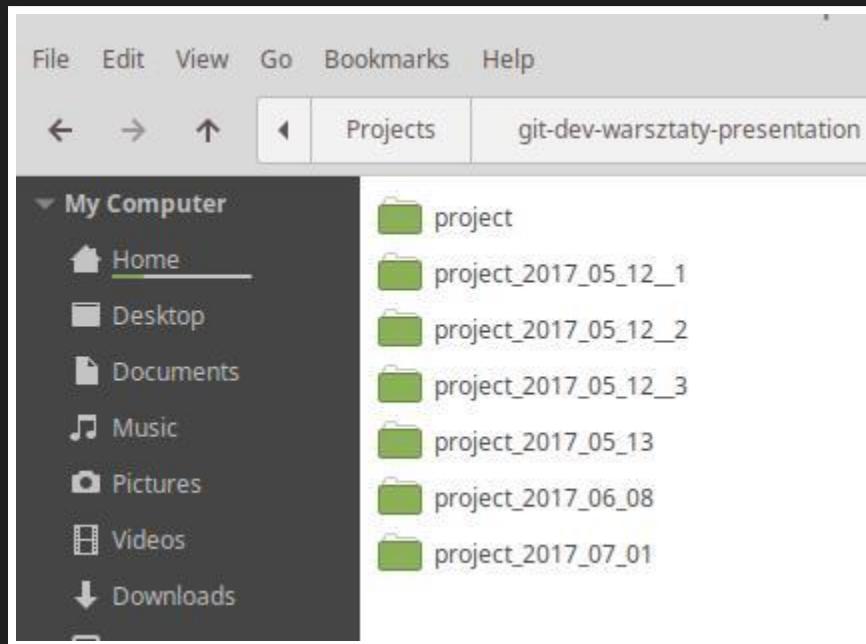
Krzysztof Morcinek & Tomasz Skraskowski

[www.gitwarsztaty.pl](http://www.gitwarsztaty.pl)

# KWESTIE ORGANIZACYJNE

- Prośba o zalogowanie się na kanał slackowy
- Po warsztatach prośba o uzupełnienie ankiety

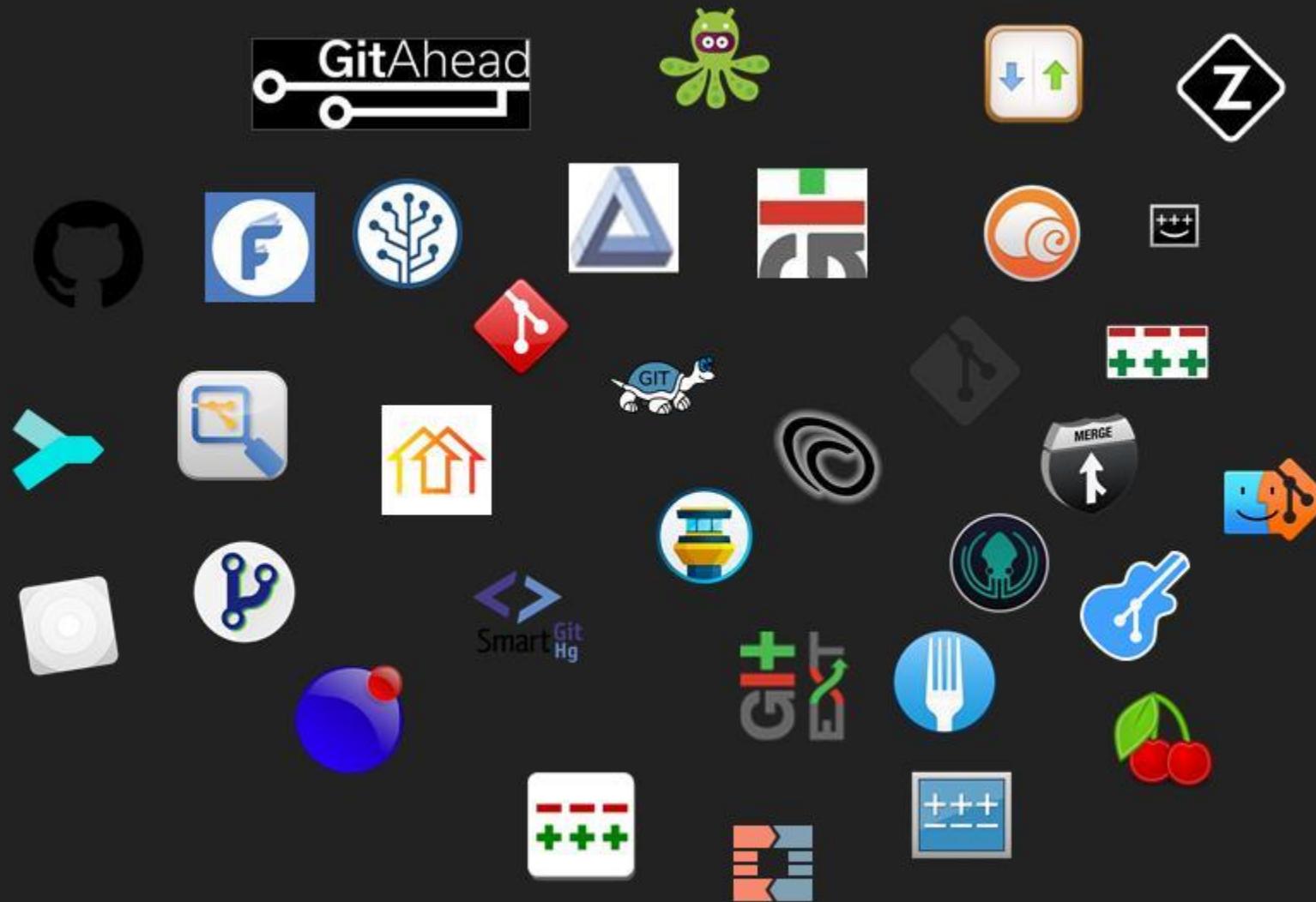
# VCS TO DOBRA RZECZ



Tak to się kiedyś robiło ;)

Polecamy korzystanie z systemu kontroli wersji nie tylko do programowania

# KONSOLA & GITK & KDIFF3



Wielu ludzi upiera się, że zarządzania gitem z konsoli nie spróbuje, że pisanie zamiast klikania to przeżytek

Ostatnio przeczytane

Krzysztof Porębski 16:37  
powiem Ci, że zmiana na konsolce  
w gitcie mocno na plus



mozesz agitowac  
podajac mnie jako przyklad  
bo sie bronilem rekami i nogami

16:37  
Cieszy mnie to 😊

Wpisz nową wiadomość

A, C, 😊, GIF, 💬, 📱, ...

▶

# KONFIGURACJA ŚRODOWISKA



# RÓŻNE EDYTORY TEKSTOWE

- vim

```
git config --global core.editor vim
```

- Linux Mint - xed

```
git config --global core.editor xed
```

- Ubuntu - gedit

```
git config --global core.editor gedit
```

- Windows - notepad

```
git config --global core.editor notepad
```

- Windows - notepad++

```
git config --global core.editor "'C:/Program  
Files/Notepad++/notepad++.exe' -multiInst -notabbar -nosession -noPlugin"
```

- Windows - Visual Studio Code

```
git config --global core.editor "'C:\Program Files\Microsoft VS  
Code\code.exe' -n -w"
```

# GIT CONFIG --GLOBAL -E

The screenshot shows a text editor window with the title ".gitconfig (~)". The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar contains icons for new file, open, save, undo, redo, cut, copy, paste, find, and search. The main content area displays the following configuration file:

```
[user]
email = tometchy@gmail.com
name = Tometchy

[core]
editor = xed

[merge]
tool = kdiff3
albo, albo

[diff]
guitool = kdiff3
Linux

[merge]
tool = kdiff3
[mergetool "kdiff3"]
path = c:/Program Files/KDiff3/kdiff3.exe
[diff]
tool = kdiff3
guitool = kdiff3
[difftool "kdiff3"]
path = c:/Program Files/KDiff3/kdiff3.exe
```

Two sections of the configuration file are highlighted with red boxes:

- Linux**: Contains the [merge] and [diff] sections.
- Windows**: Contains the [merge] section under [mergetool "kdiff3"] and the [difftool "kdiff3"] section.

A red double-headed arrow labeled "albo, albo" connects the two highlighted sections.

At the bottom of the editor window, there are buttons for .ini, Tab Width: 4, Ln 10, Col 1, and INS.

# TRZY POZIOMY KONFIGURACJI

- **--system** (brany pod uwagę na końcu)  
Linux: /etc/gitconfig  
  
Windows: %ProgramFiles(x86)%\Git\etc\gitconfig  
  
Windows: %ProgramFiles%\Git\mingw64\etc\gitconfig
- **--global** (brany pod uwagę w drugiej kolejności)  
Linux: ~/.gitconfig  
  
Windows: %USERPROFILE%\.gitconfig
- **--local** (brany pod uwagę w pierwszej kolejności)  
.git/config

# MODYFIKOWANIE KONFIGURACJI

- **git config --POZIOM SEKCJA.NAZWA "TREŚĆ"**  
Np. `git config --global user.name "John Doe"`
- **git config --POZIOM --edit**  
Np. `git config --global -e`

Domyślny poziom przy zapisywaniu to local

# ODCZYTYWANIE KONFIGURACJI

- W edytorze tekstowym: `git config (--POZIOM) --edit`  
Np. `git config --global -e`
- Konkretny wpis: `git config (--POZIOM) SEKCJA.NAZWA`  
Np. `git config --get user.email`
- Listowanie wpisów: `git config (--POZIOM) --list`  
Np. `git config --list`

Bez podania poziomu, git przyjmuje obecnie brany pod uwagę poziom

Pytania?

# GIT TO DOBRY WYBÓR :)

- jest szybki
- jest rozproszony
- ułatwia rozwiązywanie konfliktów
- wspiera nieliniowy development (branche)
- działa offline
- pozwala na pracę nad jakością commitów
- bardzo łatwy do użytku domowego
- jest bezpieczny

# JEST BEZPIECZNY - JEST WIARYGODNY

Suma kontrolna każdego commita opiera się na

- Zawartości i nazwach wszystkich plików
- ID parent commit(ów)
- Wiadomości (opisie) commita
- Autorze i/lub commiterze

Git się zorientuje przy nieprawidłowości w danych  
(np. po uszkodzeniu dysku albo próbie sabotażu)

Analogicznie wiarygodne są przelewy w bitcoin

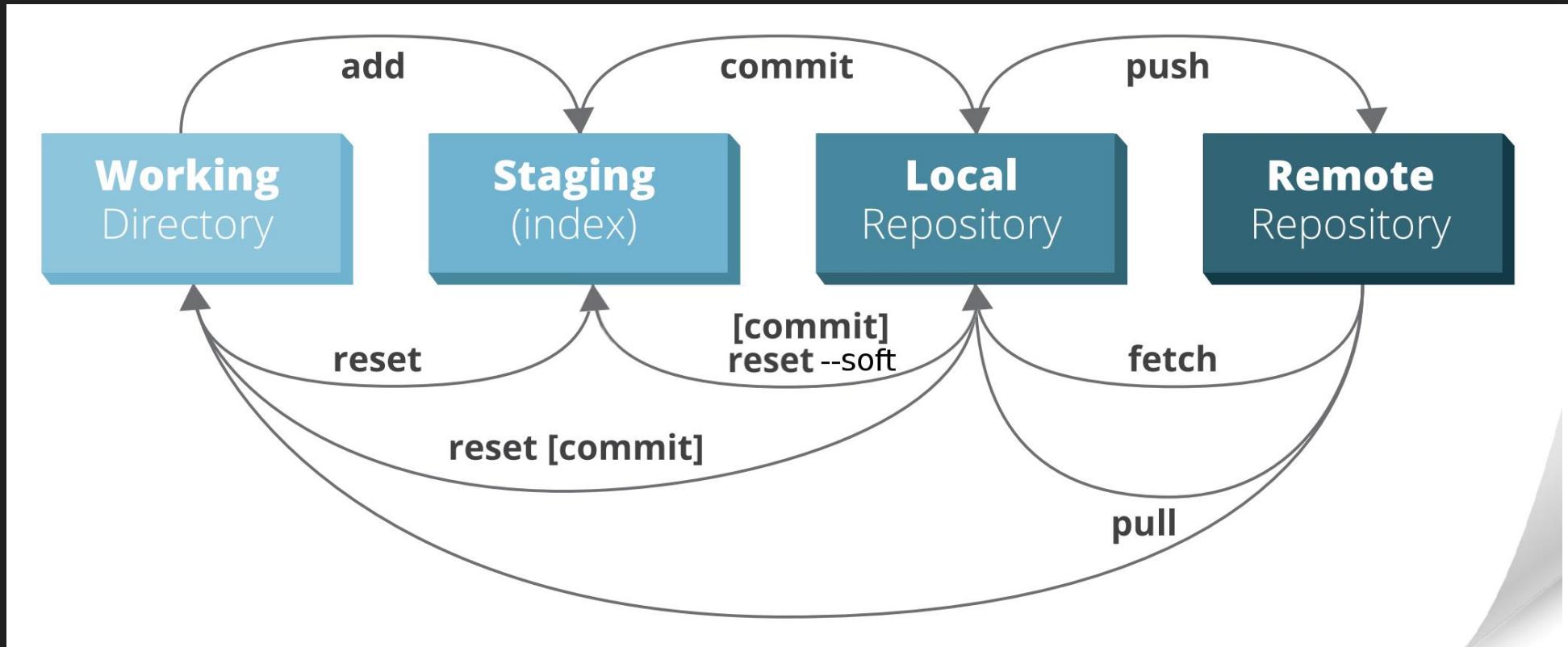
Pytania?

# **DEMO I ĆWICZENIE PROSTE FLOW**

[cwiczenia/#simple-flow](#)

Tworzenie commitów z konsoli

# FLOW



# WORKING DIRECTORY, STAGING AREA, COMMIT

Twoje pliki - absolutnie każdy plik który utworzysz/edytujesz/usuniesz, zawsze jest w którymś z 3 "obszarów" gita

- obszarze roboczym (working directory)
- 'indeksie' (staged files)
- repozytorium (commicie dodanym do historii)

# OBSZAR ROBOCZY

Obszar roboczy (working directory) to obszar w którym jest praca którą wykonałeś, ale o której jeszcze nie powiedziałeś nic gitowi

- nowe pliki których wcześniej nie commitowałeś
- zmiany w plikach które wcześniej commitowałeś
- zmiana nazwy pliku lub jego usunięcie

**Git nie wie nic o Twojej pracy w working directory, uważaj żeby jej nie stracić dopóki mu o niej nie powiesz**

Zmiany w working directory odnoszą się do staging area, a jeśli danego pliku nie ma w staging area, wówczas bezpośrednio do obecnej wersji w lokalnym repozytorium

# INDEX (STAGED FILES)

Staging area to obszar w którym przygotowujesz sobie które zmiany zostaną zaccommutowane, czyli doddane do Twojego lokalnego repozytorium

Po zastagowaniu zmiany, czyli przygotowaniu jej do commita, dalej można plik edytować, wówczas zmiany w working directory pokazywane są względem zastagowanej wersji

Co więcej, od razu można wybrać żeby zastagować tylko część zmian

Dla przykładu na dole pliku zaczęliśmy dopisywać nowy kod, a w środku pliku zobaczyliśmy literówkę w tekście. Wówczas można od razu nieprzerywając pracy na dole pliku, zastagować jedynie poprawienie literówki i dodać commita "Fix typo in module A".

# COMMIT

Ukazanie różnicy pomiędzy ostatnią wersją, a jednocześnie snapshot wszystkich plików w projekcie, posiadający identyfikator SHA-1.

```
commit 7cb1df774081fc1b9d0c97c262cbefc202d79ffa
Author: Tometchy <tometchy@gmail.com>
Date:   Wed May 16 20:36:23 2018 +0200

    Add info that git doesn't track files

    slides/drafts/git-doesnt-track.xcf | Bin 0 -> 109979 bytes
    slides/img/git-doesnt-track.png   | Bin 0 -> 32647 bytes
    slides/index.html                |      5 ++++++
  3 files changed, 5 insertions(+)
```

# PRZYKŁAD STAGOWANIA (GIT ADD)



```
Terminal
tommy@Bielak ~/Projects/Website (master) $
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
```

# PRZYKŁAD STAGOWANIA (GIT ADD)



```
Terminal
tommy@Bielak ~/Projects/Website (master) $
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
nothing to commit, working directory clean
```

# PRZYKŁAD STAGOWANIA (GIT ADD)



```
Terminal
tommy@Bielak ~/Projects/Website (master) $
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
nothing to commit, working directory clean
tommy@Bielak ~/Projects/Website (master) $ echo 'We hope you like it here' >> index.html
```

# PRZYKŁAD STAGOWANIA (GIT ADD)



```
Terminal
tommy@Bielak ~/Projects/Website (master) $ 
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
nothing to commit, working directory clean
tommy@Bielak ~/Projects/Website (master) $ echo 'We hope you like it here' >> index.html
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
tommy@Bielak ~/Projects/Website (master) $
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
nothing to commit, working directory clean
tommy@Bielak ~/Projects/Website (master) $ echo 'We hope you like it here' >> index.html
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
tommy@Bielak ~/Projects/Website (master) $ git add --all
```



# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
tommy@Bielak ~/Projects/Website (master) $ 
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
nothing to commit, working directory clean
tommy@Bielak ~/Projects/Website (master) $ echo 'We hope you like it here' >> index.html
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
tommy@Bielak ~/Projects/Website (master) $ git add --all
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html
```



# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
tommy@Bielak ~/Projects/Website (master) $
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
nothing to commit, working directory clean
tommy@Bielak ~/Projects/Website (master) $ echo 'We hope you like it here' >> index.html
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
tommy@Bielak ~/Projects/Website (master) $ git add --all
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
We hope you like it here
```



# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
no changes added to commit (use "git add" and/or "git commit -a")
tommy@Bielak ~/Projects/Website (master) $ git add --all
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
We hope you like it here
tommy@Bielak ~/Projects/Website (master) $ echo 'And we hope you will stay for a while' >> index.html
```

# PRZYKŁAD STAGOWANIA (GIT ADD)



```
Terminal
no changes added to commit (use "git add" and/or "git commit -a")
tommy@Bielak ~/Projects/Website (master) $ git add --all
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
We hope you like it here
tommy@Bielak ~/Projects/Website (master) $ echo 'And we hope you will stay for a while' >> index.html
tommy@Bielak ~/Projects/Website (master) $ cat index.html
```

# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
no changes added to commit (use "git add" and/or "git commit -a")
tommy@Bielak ~/Projects/Website (master) $ git add --all
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
We hope you like it here
tommy@Bielak ~/Projects/Website (master) $ echo 'And we hope you will stay for a while' >> index.html
tommy@Bielak ~/Projects/Website (master) $ cat index.html
Welcome at our website
We hope you like it here
And we hope you will stay for a while
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html
```



# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
We hope you like it here
And we hope you will stay for a while
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html

tommy@Bielak ~/Projects/Website (master) $ git diff
diff --git a/index.html b/index.html
index ded25b6..fb6438e 100644
--- a/index.html
+++ b/index.html
@@ -1,2 +1,3 @@
 Welcome at our website
 We hope you like it here
+And we hope you will stay for a while
```



# PRZYKŁAD STAGOWANIA (GIT ADD)

```
Terminal
We hope you like it here
And we hope you will stay for a while
tommy@Bielak ~/Projects/Website (master) $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   index.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   index.html

tommy@Bielak ~/Projects/Website (master) $ git diff
diff --git a/index.html b/index.html
index ded25b6..fb6438e 100644
--- a/index.html
+++ b/index.html
@@ -1,2 +1,3 @@
 Welcome at our website
 We hope you like it here
+And we hope you will stay for a while
tommy@Bielak ~/Projects/Website (master) $ git diff --staged
diff --git a/index.html b/index.html
index 04dae32..ded25b6 100644
--- a/index.html
+++ b/index.html
@@ -1 +1,2 @@
 Welcome at our website
+We hope you like it here
tommy@Bielak ~/Projects/Website (master) $
```



Pytania?

# HEAD

HEAD to referencja do obecnie zcheckoutowanego commita,  
zazwyczaj będącego ostatnim commitem w branchu

- HEAD - ostatni commit (*pierwszy od końca*) - np.  
*ed8ab42[...]*
- HEAD~1 - przedostatni commit - np. *daa6b92[...]*
- HEAD^ - przedostatni commit - np. *a6bc32b[...]*
- HEAD~3 - czwarty od końca commit - np. *7cb16f7[...]*
- HEAD^^^ - czwarty od końca commit - np. *343ebb3[...]*

Pytania?

# EDYTOWANIE WPROWADZONYCH ZMIAN

- Przed opublikowaniem innym
- Po opublikowaniu innym

# EDYCJA 'OSTATNICH' COMMITÓW PRZED OPUBLIKOWANIEM

- `git reset`
  - `git reset --mixed`
  - `git reset --soft`
  - `git reset --hard`
- `commit --amend`
- `rebase`
- `rebase --interactive`

# EDYCJA 'OSTATNICH' COMMITÓW PO OPUBLIKOWANIU

**git revert**

np:

- git revert HEAD
- git revert HEAD~4
- git revert 6e5av3a

# MODYFIKACJA WORKING DIRECTORY I STAGING AREA

- git checkout \*PATH\*
- git reset --hard
- git clean (-d -f -n)

## **ĆWICZENIE WSPÓŁNIE Z DEMO *RESET*, *REVERT*, *COMMIT AMMEND***

[cwiczenia/#reset-revert-commit-ammend](#)

Pytania?

# REBASE --INTERACTIVE

```
|pick 95a10ac New cars
pick 91b7878 Add wrong car
pick d294edd Add Mazdaaa car
pick 2973b7f Add Skoda
pick 2c053d1 Correct typo
pick 5ae3037 Add description

# Rebase ff03ffb..5ae3037 onto ff03ffb (6 commands)
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
```

# REBASE --INTERACTIVE

Przenosimy linie tekstu w edytorze

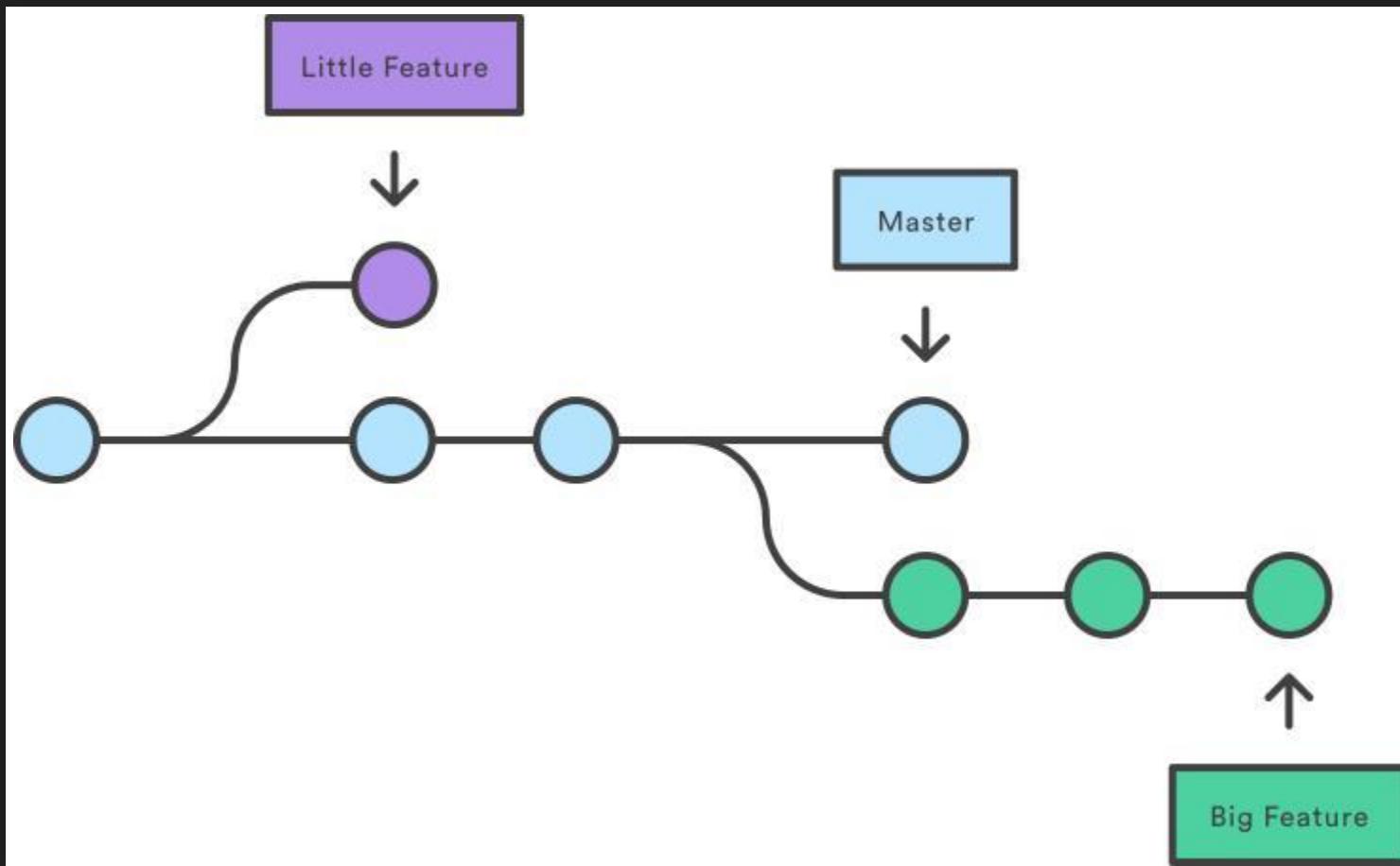
Zmieniamy *komendy* z początków linii w edytorze

Zawsze wnikliwie czytamy podpowiedzi git, piszemy **git status** gdy znikną

[cwiczenia/#rebase-interactive-demo](#)

Pytania?

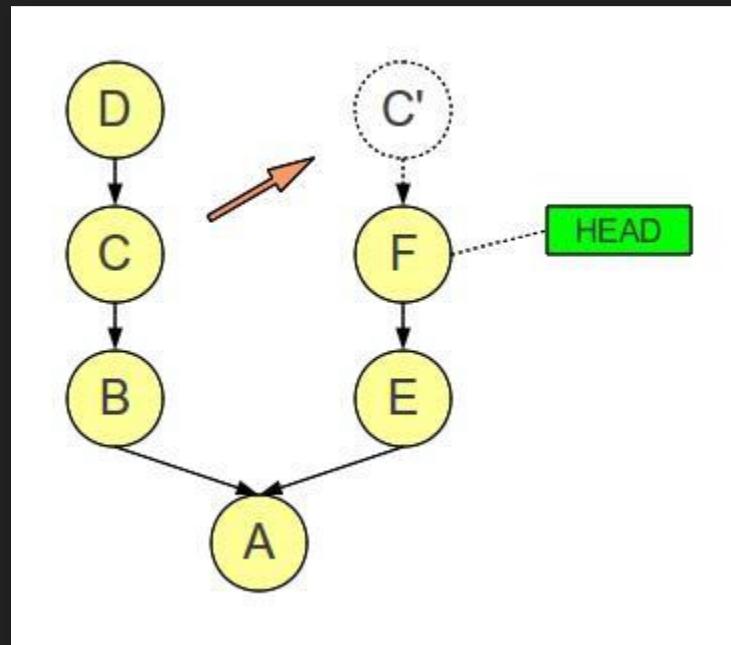
# BRANCHE



Learning branching

# CHERRY-PICK

Commity między branchami można przekładać



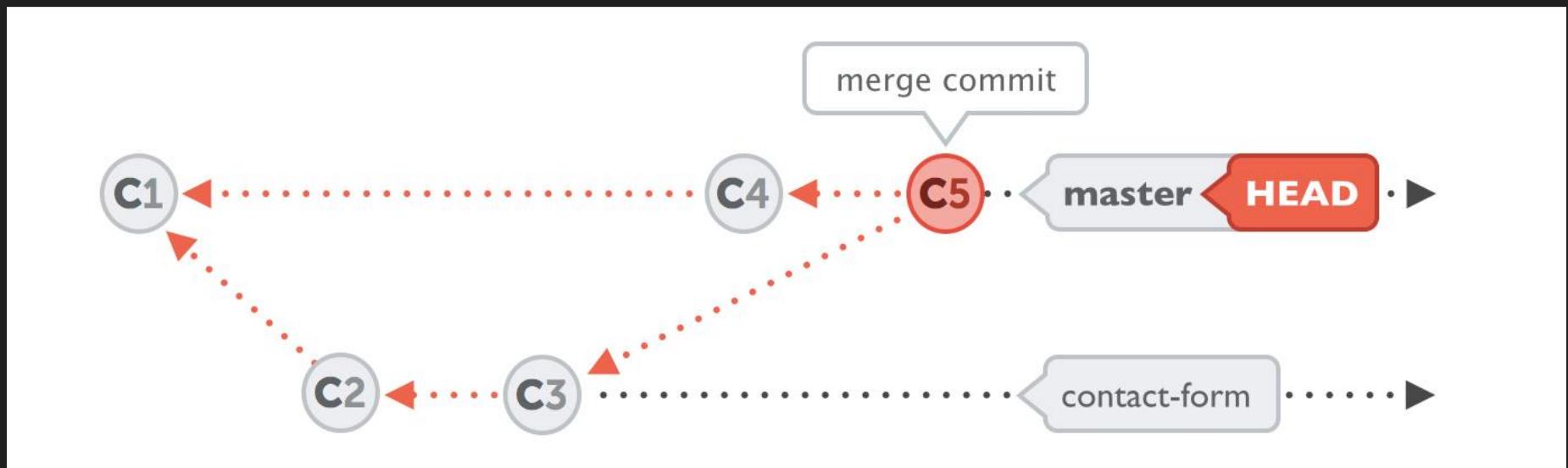
# DEMO

pokazanie jak zmiana brancha wpływa na zawartość  
katalogów (git checkout)

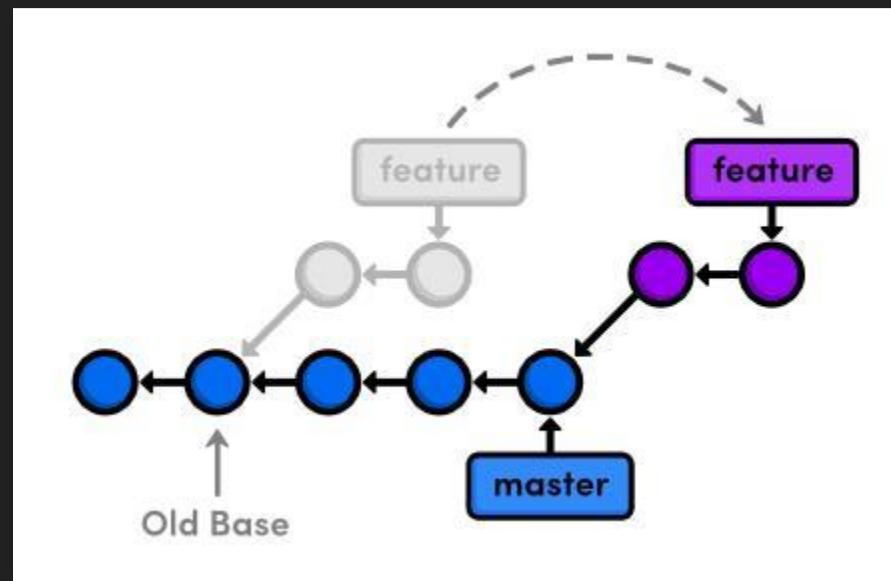
stosowanie cherry-pick w praktyce

Pytania?

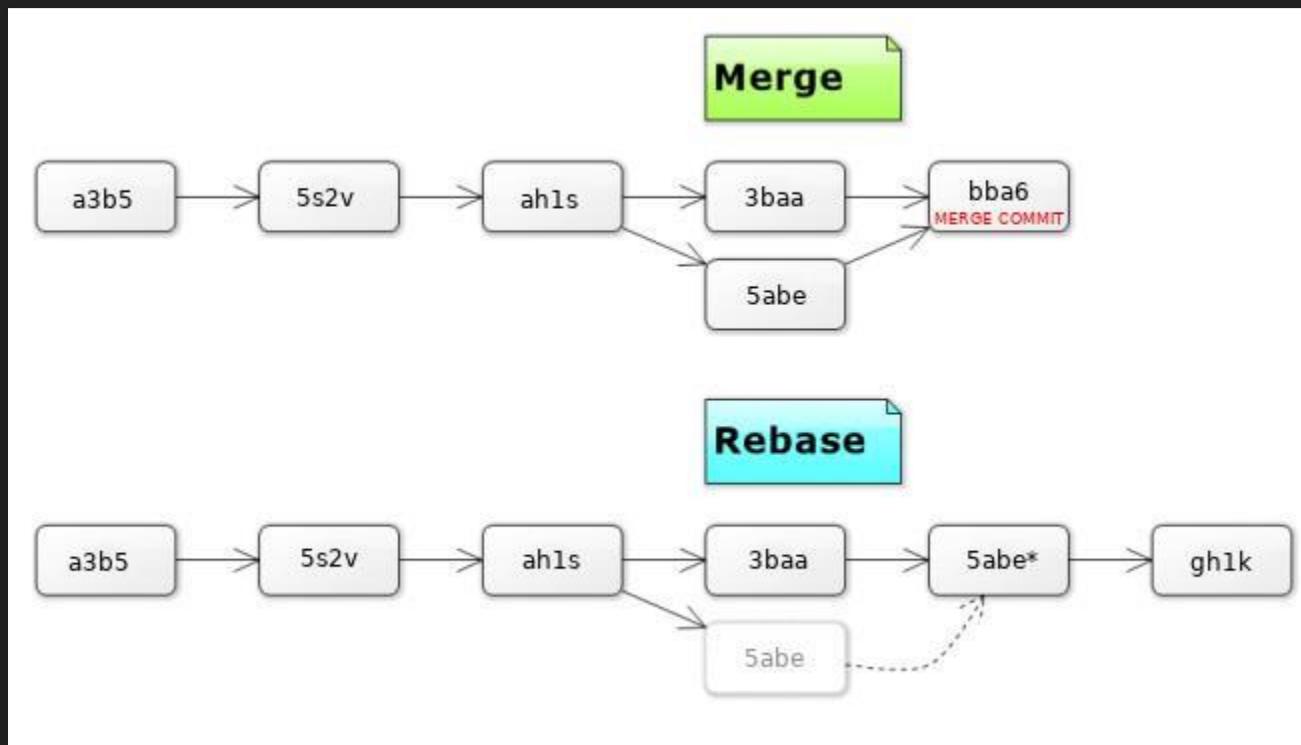
# MERGE



# REBASE



# MERGE VS REBASE



# GIT GUITAR HERO



Ćwiczenie: zrób merge i rebase

[cwiczenia/#rebase-vs-merge](#)

# REBASE TO ŚWIETNE NARZĘDZIE...

... ale w niektórych sytuacjach trzeba uważać.

Czasami łatwiej przerwać rebasowanie i jeżeli flow projektu na to pozwala, jednak zrobić zwykłego merge.

(głównie przy *starych*, mocno *rozjedzanych* branchach)

- rebase zakłamauje historię
- rebase może wywołać konflikty których by nie było przy mergu
  - a w związku z tym, może wprowadzić błąd na produkcję
  - pomijając, że to dokładna dodatkowej pracy

Pytania?

# KONFLIKTY

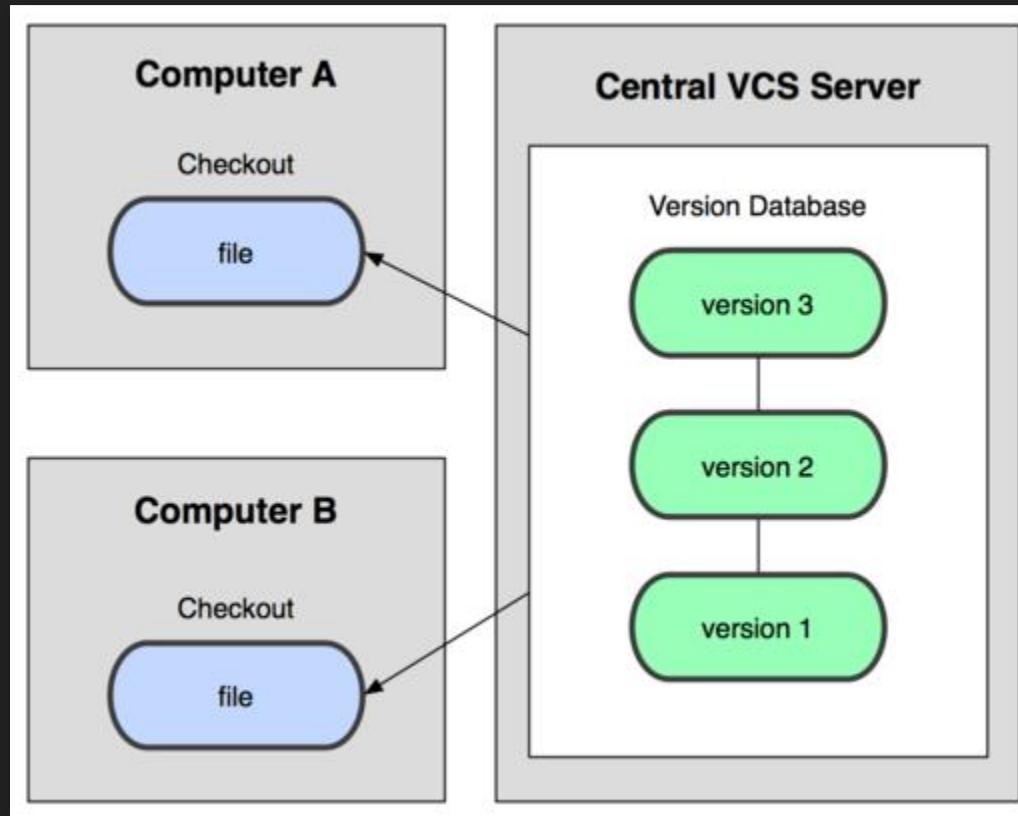
- Są czymś naturalnym w środowisku gdzie niezależne osoby działają
- Tak wygląda powstały konflikt w kodzie

```
1 <<<<< HEAD
2 using System.IO;
3
4 function foo() {
5     "hello world";
6     bar();
7 =====
8 using System.Drawing;
9
10 function foo() {
11     "hello world";
12     "Witaj swiecie"
13 >>>>> Tomek commit
14 }
```

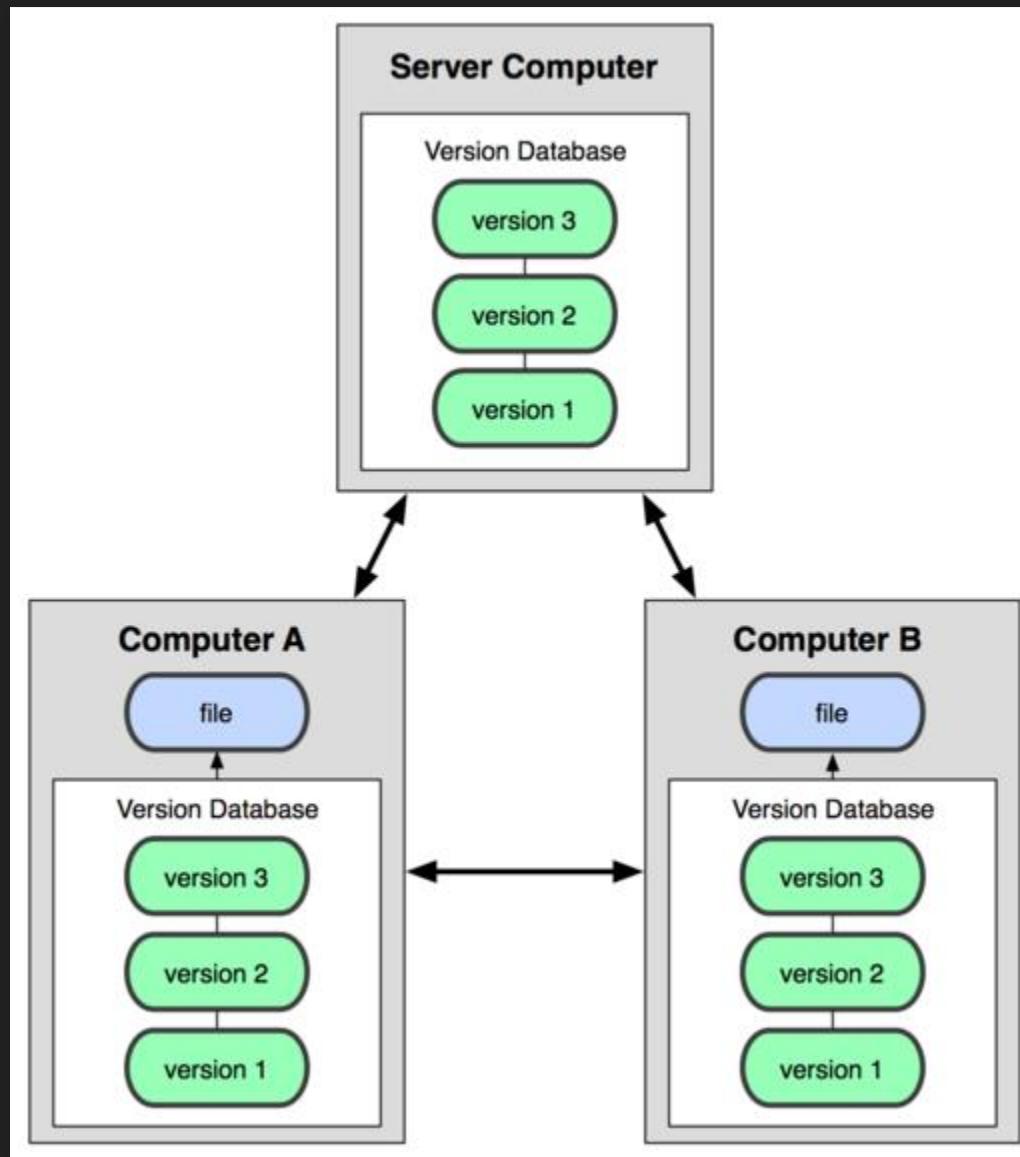
Ćwiczenie: rozwiąż konflikty  
[cwiczenia/#resolve-conflicts](#)

Pytania?

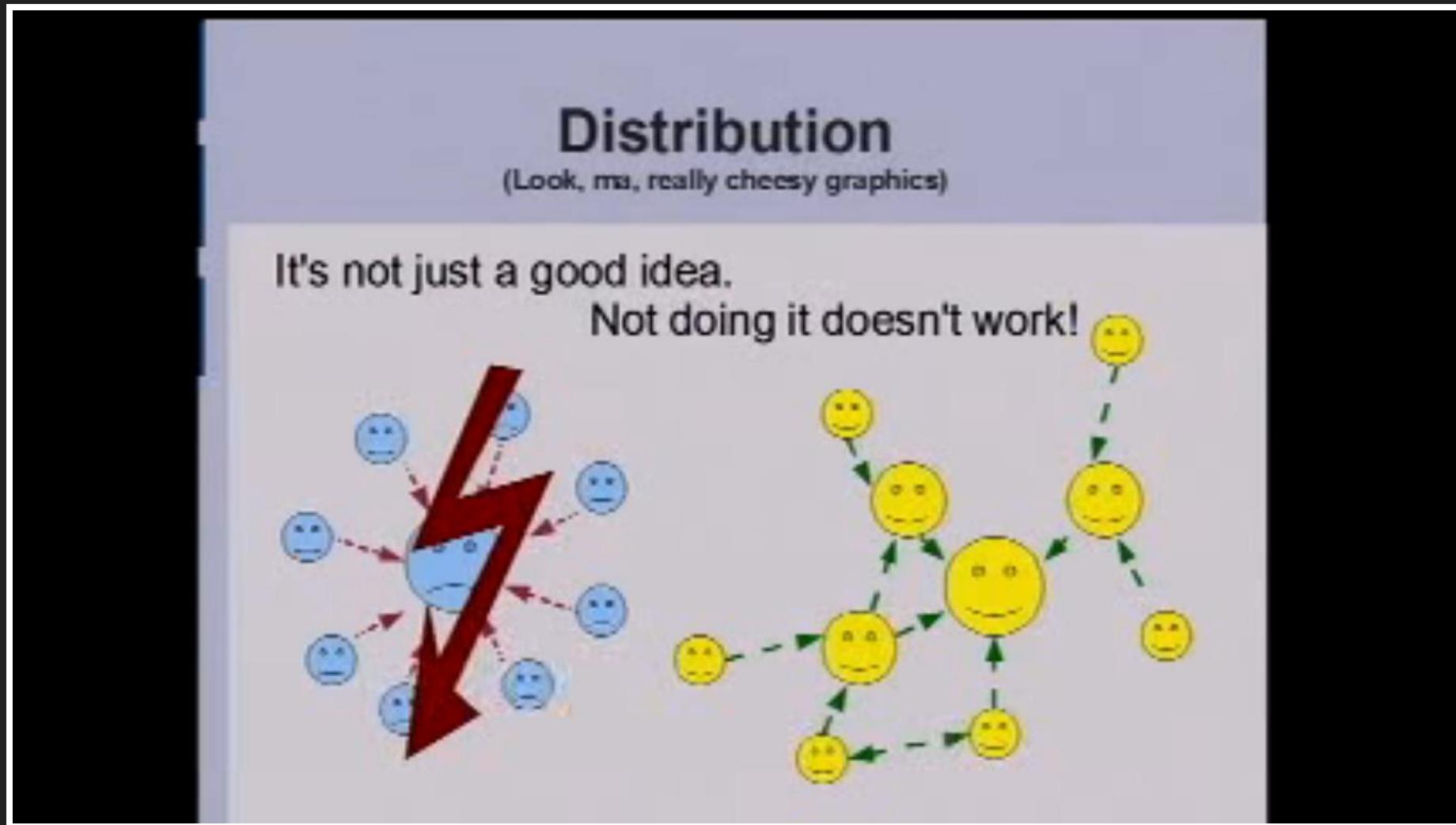
# Git nie jest scentralizowanym systemem kontroli wersji



# Git jest zdecentralizowanym systemem kontroli wersji



Git jest zdecentralizowanym systemem kontroli wersji



# GIT TO W ZASADZIE BAZA DANYCH

Git to VCS na który można patrzeć, jak na rozproszoną bazę danych, opartą na streamie snapshotów plików



Pytania?

# PUSHOWANIE & PULLOWANIE ZMIAN

Synchronizacja repozytoriów (np. lokalnego z wersją na GitHubie) odbywa się za pomocą 'wypychania' i 'ciągnięcia' commitów

- git push
- git pull
  - git fetch
  - git merge

## PULL Z REBASE

Często dobrą opcją jest zastosowanie komendy która od razu za nas zrebasuje commity

```
git pull --rebase  
git config --global --bool pull.rebase true
```

Ćwiczenie: pushowanie do wspólnego repozytorium  
[cwiczenia/#push-at-the-same-time](#)

## PUSH --FORCE

Jeżeli jesteś pewny co robisz, możesz wymusić pusha

`git push --force`

Ale jak już wymuszasz, to zawsze z zabezpieczeniem

`git push --force-with-lease`

Pytania?

# PRZEGŁĄDANIE HISTORII

- git log
- gitk

Git log oraz gitk domyślnie pokazują historię od *obecnego momentu* - HEAD, ale możliwe jest wskazanie dowolnego momentu w historii, np. nazwy brancha, SHA commita lub odniesienia SHA<sup>^</sup> lub SHA~N

np. HEAD<sup>^</sup>, master~5

# POPULARNE PRZEŁĄCZNIKI DO GIT LOG / GITK

- git log -4
- git log --oneline
- git log --graph
- git log --decorate
- git log --stat
- git log --patch
- git log --since=X,git log --after=X,git log --before=X
- git log X --not Y,git log Y..X,git log X...Y
- git log --author="xyz"
- git log --grep="xyz" -i
- git log --no-merges
- git log --all
- git log --pretty="FORMAT"
- git log -S"text" -i
- Wyjście ze strumienia logów - klawisz *q*

# GIT LOG Z GRAFEM

```
Terminal
tommy@Bielak ~/Desktop/temp/material-components-android (master) $ git log --graph --decorate
* commit 768659d5f861efe278c96947dfa29023895a6970 (HEAD -> master)
| Author: Tometchy <tometchy@gmail.com>
| Date:   Fri May 18 22:03:32 2018 +0200
|
|       Update readme file
|
* commit 2478632c421929ef1bf96ca1ad8b48e3211d2277
| Merge: 7fa43fc 8261465
| Author: Tometchy <tometchy@gmail.com>
| Date:   Fri May 18 22:02:19 2018 +0200
|
|       Merge branch 'travisci'
|
* commit 8261465a10a0289f4b1e717038760c1a36662091 (origin/travisci, travisci)
| Author: Cameron Ketcham <cketcham@google.com>
| Date:   Thu May 17 17:08:42 2018 -0400
|
tommy@Bielak ~/Desktop/temp/material-components-android (master) $
```

# GIT LOG Z JEDNOLINIWYMI WYNIKAMI

```
Terminal
| * 53ba33d Fix android-P repo
| * d8b646b Remove flaky test
tommy@Bielak ~/Desktop/temp/material-components-android (master) $ git log -5 --oneline
392aecा Update readme file
2478632 Merge branch 'travisci'
7fa43fc Add null check before validating attribute set to avoid NPE.
1cad400 Roll forward box background fix.
eb89b76 Prevent user from setting start/end compound drawable on Chips. They should be set via app:chipIcon
and app:closeIcon instead.
tommy@Bielak ~/Desktop/temp/material-components-android (master) $ git log -5 --oneline --graph
* 392aecा Update readme file
* 2478632 Merge branch 'travisci'
| \
| * 8261465 Add lynx back
| * 53ba33d Fix android-P repo
tommy@Bielak ~/Desktop/temp/material-components-android (master) $ git log -5 --oneline --graph --decorate
* 392aecा (HEAD -> master) Update readme file
* 2478632 Merge branch 'travisci'
| \
| * 8261465 (origin/travisci, travisci) Add lynx back
| * 53ba33d Fix android-P repo
| * d8b646b Remove flaky test
tommy@Bielak ~/Desktop/temp/material-components-android (master) $
```

# GIT LOG Z POKAZANĄ STATYSTYKĄ I DIFFEM

```
Terminal
tommy@Bielak ~/Desktop/temp/material-components-android (master) $ git log -1 --stat --patch
commit 392aecab5aff1467bc329af9b60d02e7e6a94915
Author: Tometchy <tometchy@gmail.com>
Date:   Fri May 18 22:03:32 2018 +0200

    Update readme file
---
 README.md | 2 ++
 1 file changed, 2 insertions(+)

diff --git a/README.md b/README.md
index b2e8da6..3de54ea 100644
--- a/README.md
+++ b/README.md
@@ -1,3 +1,5 @@
+# Brand new section
+Brand new description
 [![Build Status](https://img.shields.io/travis/material-components/material-components-androi
d/master.svg)](https://travis-ci.org/material-components/material-components-android)
 [![Chat](https://img.shields.io/discord/259087343246508035.svg)](https://discord.gg/material-
components)

tommy@Bielak ~/Desktop/temp/material-components-android (master) $ █
```

# GITK

material-components-android: All files - gitk

**File Edit View Help**

master Update readme file  
Merge branch 'travisci'  
travisci remotes/origin/travisci Add lynx back  
Fix android-P repo  
Remove flaky test  
Modify travis build to cache android sdk  
Remove flaky test  
Remove flaky test  
Only run one set of tests  
Remove tests  
Add findbugsден for tests

SHA1 ID: d8b646bb3842161879bbcf7a0f9231580dba4b9e

Find commit containing:  Exact All fields

Diff Old version New version Lines of context:

Author: Cameron Ketcham <cketcham@google.com> 2018-05-18 22:03:32  
Committer: Cameron Ketcham <cketcham@google.com> 2018-05-18 22:02:19  
Parent: [9dbf97998002b1b6ec13878ea5ddce1199f52e05](#) (Modif)  
Child: [53ba33d800788116945eb06fd34f26e89f9292ad](#) (Fix)  
Branches: master, remotes/origin/travisci, travisci  
Follows: [1.0.0-alpha1](#)  
Precedes:  
  
Remove flaky test

Tometchy <tometchy@gmail.com> 2018-05-18 22:03:32  
Tometchy <tometchy@gmail.com> 2018-05-18 22:02:19  
Cameron Ketcham <cketcham@google.com> 2018-05-17 23:08:42  
Cameron Ketcham <cketcham@google.com> 2018-05-17 22:50:12  
Cameron Ketcham <cketcham@google.com> 2018-05-17 22:45:16  
Cameron Ketcham <cketcham@google.com> 2018-05-17 22:40:47  
Cameron Ketcham <cketcham@google.com> 2018-05-17 22:28:04  
Cameron Ketcham <cketcham@google.com> 2018-05-17 22:13:38  
Cameron Ketcham <cketcham@google.com> 2018-05-17 21:50:27  
Cameron Ketcham <cketcham@google.com> 2018-05-17 19:36:30  
Cameron Ketcham <cketcham@nonale.com> 2018-05-17 17:52:56

Comments  
tests/javatests/com/google/android/material/textfield/TextInputLayoutTest.java

# SZUKANIE W ŁADNYCH TOOLACH

- Jest *ładne*
- Jest ograniczone
- Wymaga *oderwania się* od konsoli

# PRZYGOTOWANIE DO ĆWICZEŃ Z PRZEGŁĄDANIEM HISTORII

1. Przejdź do katalogu ćwiczeń na pulpicie:

```
cd ~/Desktop/cwiczenia
```

2. Sklonuj przykładowe repozytorium (z projektem open source), do katalogu log-demo:

```
git clone https://gitlab.com/terrakok/gitlab-client log-demo
```

3. Przejdź do katalogu repo: cd log-demo

4. Zresetuj repozytorium do wersji:

```
7ad14ecda2f97a3ff7c3cca44cc7605484b578b5
```

```
git reset --hard 7ad14
```

*Żeby każdy miał tę samą wersję do ćwiczeń*

**SPRAWDŹ TYTUŁ OSTATNIEGO COMMITA AUTORA Z  
NAZWISKIEM GULYA**

**GITK**

**gitk --author=gulya -1**

**KONSOLA**

**git log --author=gulya -1**

**Odpowiedź:**

Upgrade Markwon to version 2.0.0. Replace ImageSizeResolver workaround with upstream one.

**Pytania?**

Start Stop Restart

**00:00:00**

# SPRAWDŹ AUTORA COMMITA ZAWIERAJĄCEGO W OPISIE FRAZĘ *UPDATE STUB*

GITK

```
gitk --all --grep="Update Stub" -i -1
```

KONSOLA

```
git log --all --grep="Update Stub" -i -1
```

Odpowiedź:  
terrakok

Pytania?

Start Stop Restart

00:00:00

# SPRAWDŹ KTO PODNIÓSŁ WERSJĘ PRODUKTU Z 9 NA 10

(TEXT W KODZIE: VERSIONCODE 10)

**GITK**

Da się wyklikać :)

**KONSOLA**

```
git log -S "versionCode 10" -i -p
```

Odpowiedź:

*terrakok, w commicie*

6d289b10a07bb13d75a7b9767b9b0ae52b0421ce

**Pytania?**

Start Stop Restart

00:00:00

# SPRAWDŹ JAKIE PLIKI MODYFIKOWAŁ COMMIT

B9272A3BE3D4F9182E6893A0F9A2F7B9FF0B6923

## GITK

gitk b927 -1

## KONSOLA

git log b927 -1 --stat

Odpowiedź:

.../model/repository/session/SessionRepository.kt | 9 ++++++++

.../terrakok/gitlabclient/presentation/auth/AuthPresenter.kt | 12 +++++-----

Pytania?

Start Stop Restart

00:00:00

# SPRAWDŹ CO DOKŁADNIE ZMIENIŁ COMMIT

0E02008383A59F6AB56A14DF4688070AAB925765

## GITK

gitk 0e020 -1

## KONSOLA

```
git log 0e020 -1 --patch  
lub  
git show 0e020
```

## Odpowiedź:

Linie dodane do pliku PrivacyPolicyFragment.kt:  
override val parentScopeName = DI.APP\_SCOPE  
 pusta  
 Toothpick.inject(this, scope)

Linia usunięta z pliku PrivacyPolicyFragment.kt:  
Toothpick.inject(this, Toothpick.openScope(DI.APP\_SCOPE))

Pytania?

Start Stop Restart

00:00:00

# SPRAWDZ ID COMMITÓW Z DRUGIEGO GRUDNIA 2018

## GITK

```
gitk --after="2018-12-02 00:01" --before="2018-12-02 23:59"
```

## KONSOLA

```
git log --after="2018-12-02 00:01" --before="2018-12-02 23:59" --oneline
```

## Odpowiedź:

```
6167018 Merge branch 'task/upgrade-markwon-and-remove-image-loading-crutch' into 'develop'  
fdb2e6c Merge branch 'project_milestone_tab' into 'develop'  
158bba8 Delete unused "todo".  
6d45888 Add account id for fixing same userId for difference servers.  
0e02008 Fix scope for PrivacyPolicyFragment.  
c46c806 Merge branch 'feture/multi_account' into develop  
45c0836 Delete code style from git history.  
5752be0 Add scopes to each fragment for more powerful memory management.
```

## Pytania?

Start   Stop   Restart

00:00:00

Sprawdź jakie commity są lokalnie na branchu develop, a których nie ma na branchu develop na serwerze  
**GITK**

```
gitk develop --not origin/develop  
lub  
gitk origin/develop..develop
```

## KONSOLA

```
git log develop --not origin/develop  
lub  
git log origin/develop..develop
```

Odpowiedź:

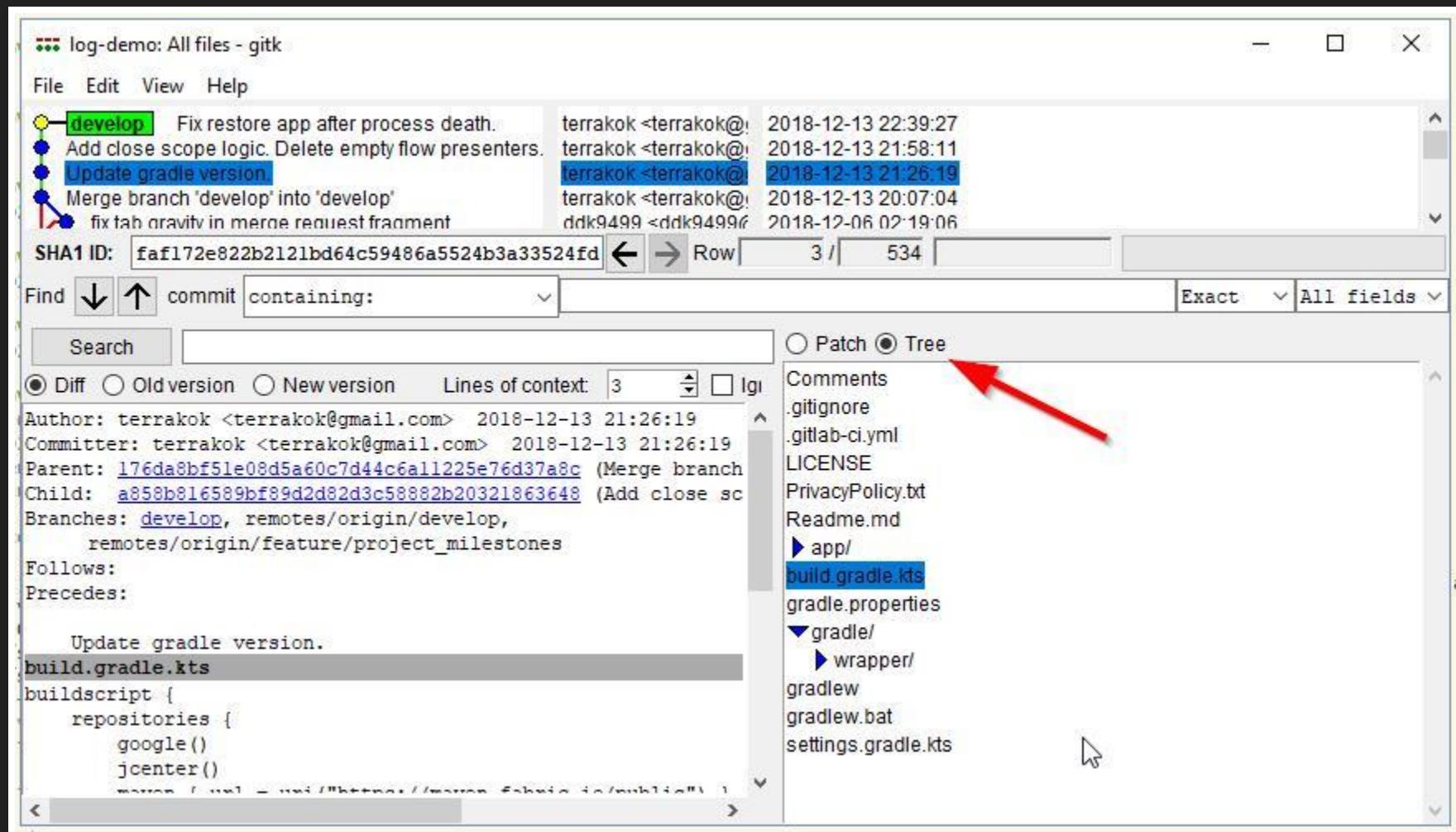
Nie ma takich

Pytania?

Start Stop Restart

00:00:00

# GITK POKAZUJĄCY CAŁY STAN PROJEKTU



Przejrzeć drzewo można również w konsoli, ale to się przydaje głównie do skryptów  
git ls-tree -r 0e020 --name-only

Pytania?

# IGNOROWANIE PLIKÓW

Ścieżki do plików/katalogów które chcemy ignorować dla repozytorium, trzymamy w pliku .gitignore

Plik ten wersjonujemy w repozytorium

# PRZYKŁADOWE REGUŁY IGNOROWANIA

- `*.orig`
- `**/[Pp]ackages/*`
- `bin/`
- `!src/Todo.csproj.user` - wykluczenie od reguły  
(zaczyna się od wykryznika)

[git-scm.com/docs/gitignore#\\_pattern\\_format](https://git-scm.com/docs/gitignore#_pattern_format)

## DODAJĘ REGUŁĘ DO .GITIGNORE, A "GIT STATUS" DALEJ POKAZUJE ZMIANY

Jest to scenariusz na który każdy przedzej czy później się natknie

Powód jest prosty - plik już jest w repozytorium,  
.gitignore ignoruje tylko nie śledzone pliki

Wystarczy plik... usunąć z repozytorium

# USUWANIE PLIKU Z REPOZYTORIUM, ŻEBY "POSŁUCHAŁ" REGUŁ .GITIGNORE

- Tradycyjnie:  
usunięcie pliku -> git add -> git commit
- Za pomocą komendy:  
`git rm file1.txt`  
`git commit -m "remove file1.txt"`
- Zostawiając plik na dysku:  
`git rm --cached file1.txt`  
`git commit -m "remove file1.txt"`

# GOTOWE REGUŁY .GITIGNORE

- [www.gitignore.io](http://www.gitignore.io)
- checkbox przy tworzeniu nowego projektu
- [Krzyśka Morcinka ;\).gitignore](http://Krzyśka-Morcinka-.gitignore)

Pytania?

# 7 ZASAD DOBREGO COMMIT MESSAGE

1. Oddziel tytuł od ciała pustą linią
2. Ogranicz tytuł do **50 znaków**
3. Stosuj wielkie litery w tytule
4. Nie kończ tytułu commita kropką
5. Zapisuj tytuł w trybie rozkazującym
6. Ogranicz 'szerokość' ciała do 72 znaków
7. W ciele opisz *co* i *dlaczego*, a nie *jak*

<https://chris.beams.io/posts/git-commit/>

Pytania?

# GIT REFGLOG

```
jekyll master $ git reflog
6703083... HEAD@{0}: pull origin master: Fast forward
fe71d2b... HEAD@{1}: commit: Updating README with added
eac6b03... HEAD@{2}: commit: Making sure that posts fla
f682c8f... HEAD@{3}: commit: Added publish flag to post
9478f1b... HEAD@{4}: rebase: Modifying the README a bit
7e178ff... HEAD@{5}: checkout: moving from master to 7e
cda3b9e... HEAD@{6}: HEAD~1: updating HEAD
3b75036... HEAD@{7}: merge newfeature: Merge made by re
cda3b9e... HEAD@{8}: commit: Modifying the README a bit
6981921... HEAD@{9}: checkout: moving from newfeature t
7e178ff... HEAD@{10}: commit: Rewriting history is fun
4a97573... HEAD@{11}: commit: all done with TODOs
eb60603... HEAD@{12}: commit: README
```

Pytania?

# SCHOWEK

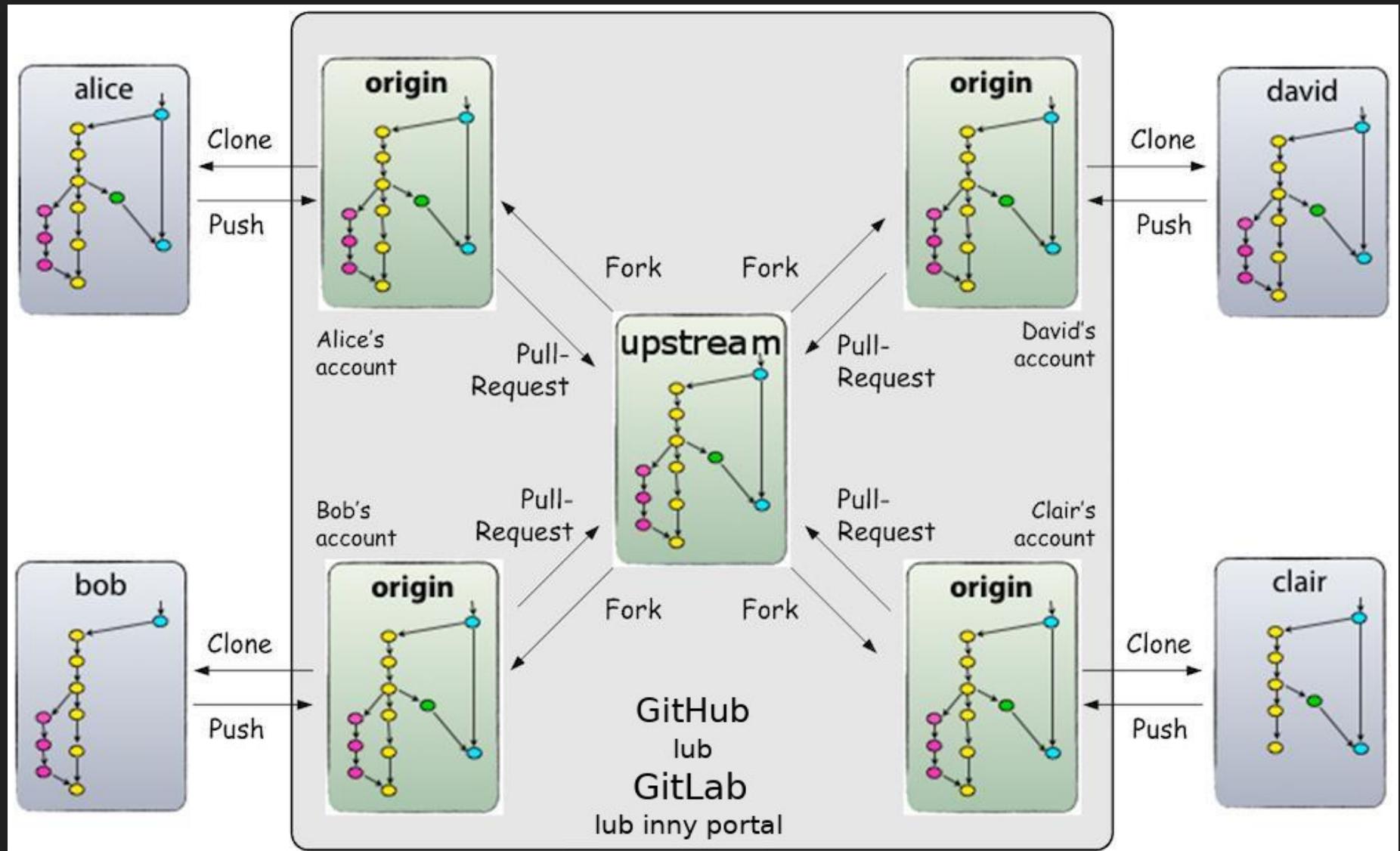
- git stash
- git stash --include-untracked
- git stash list
- git stash apply (optional: name)
- git stash pop (optional: name)
- git stash drop (optional: name)
- Różne wariacje, zachowywanie indexu, nadawanie wiadomości itd.

## Uwaga

Łatwo zapomnieć o nieśledzonych plikach

Pytania?

# FORKI & PULL REQUESTY

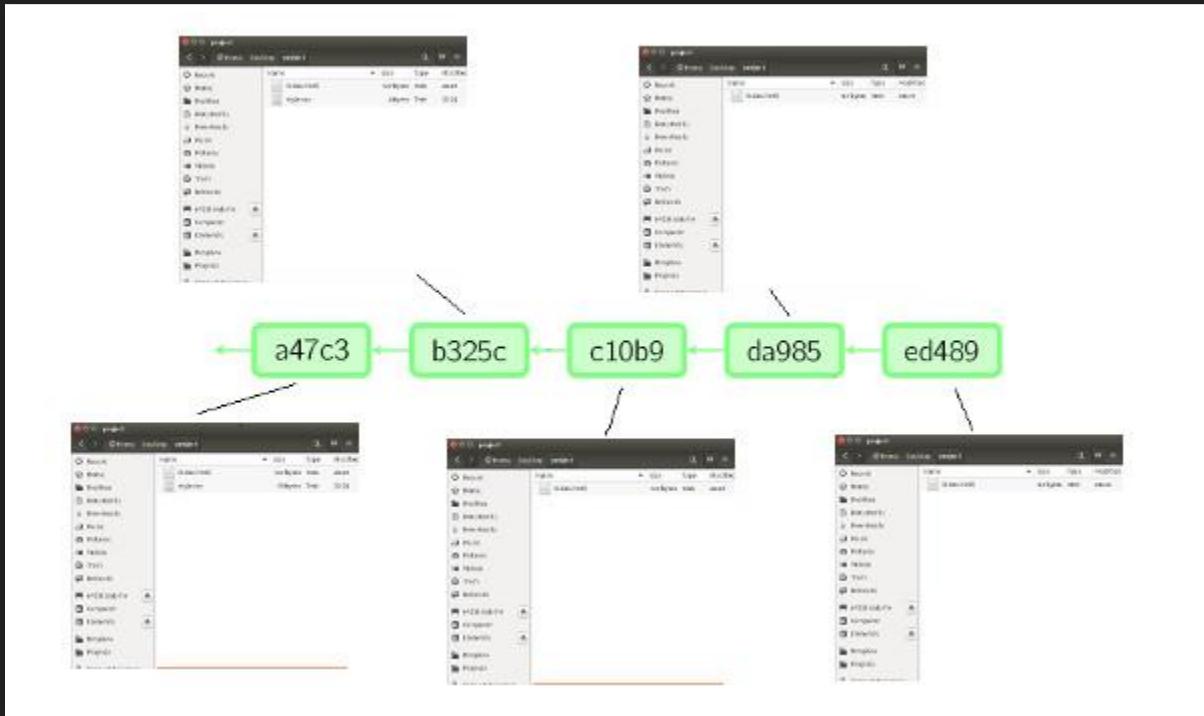


# PYTANIA? - PULL REQUESTY!

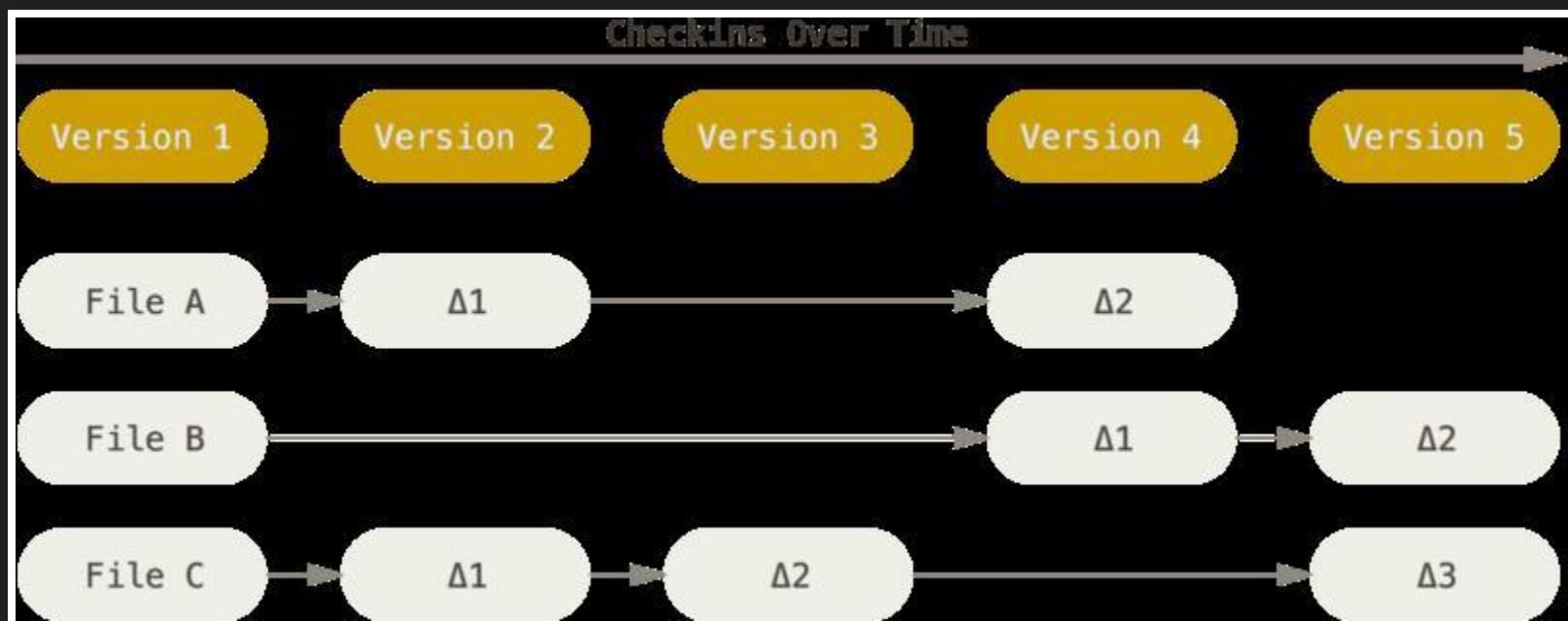
[cwiczenia/#forks-pull-request](#)

# JAK DZIAŁA GIT

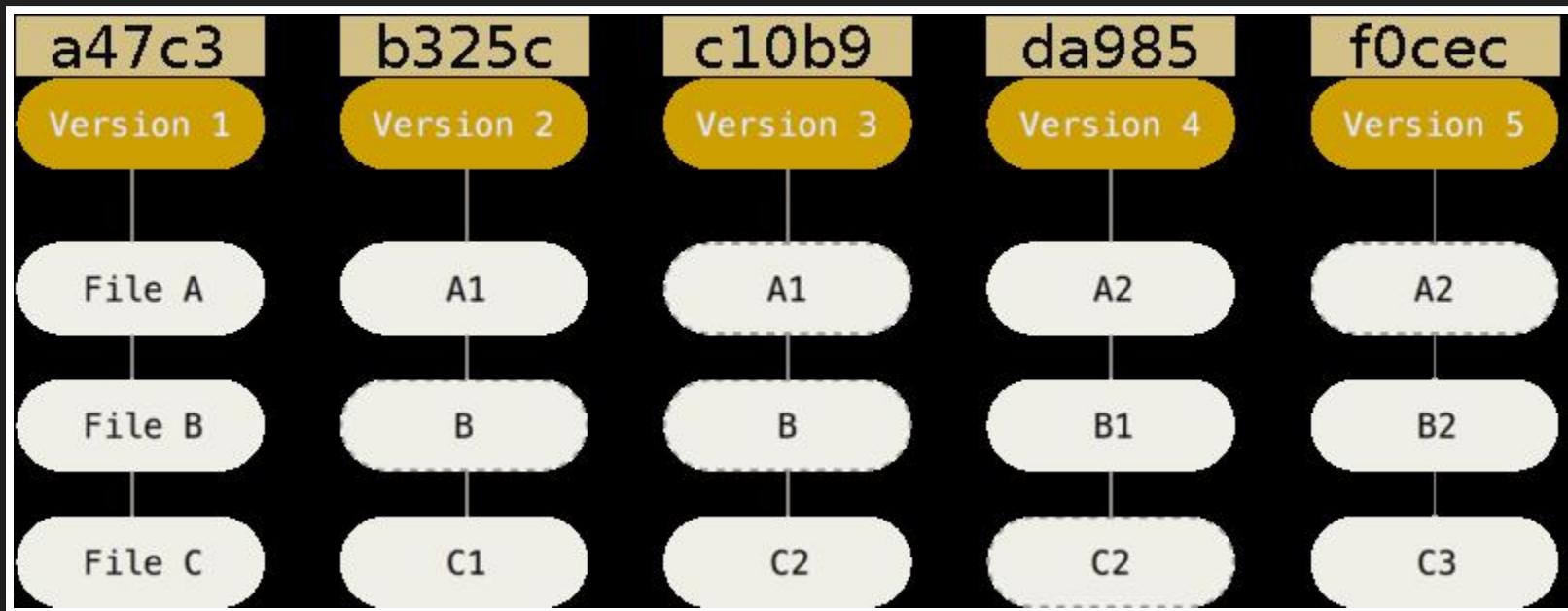
Git w podstawowym scenariuszu jedynie dodaje kolejne snapshoty plików



W svn commit to diff zmienionych plików (różnica)

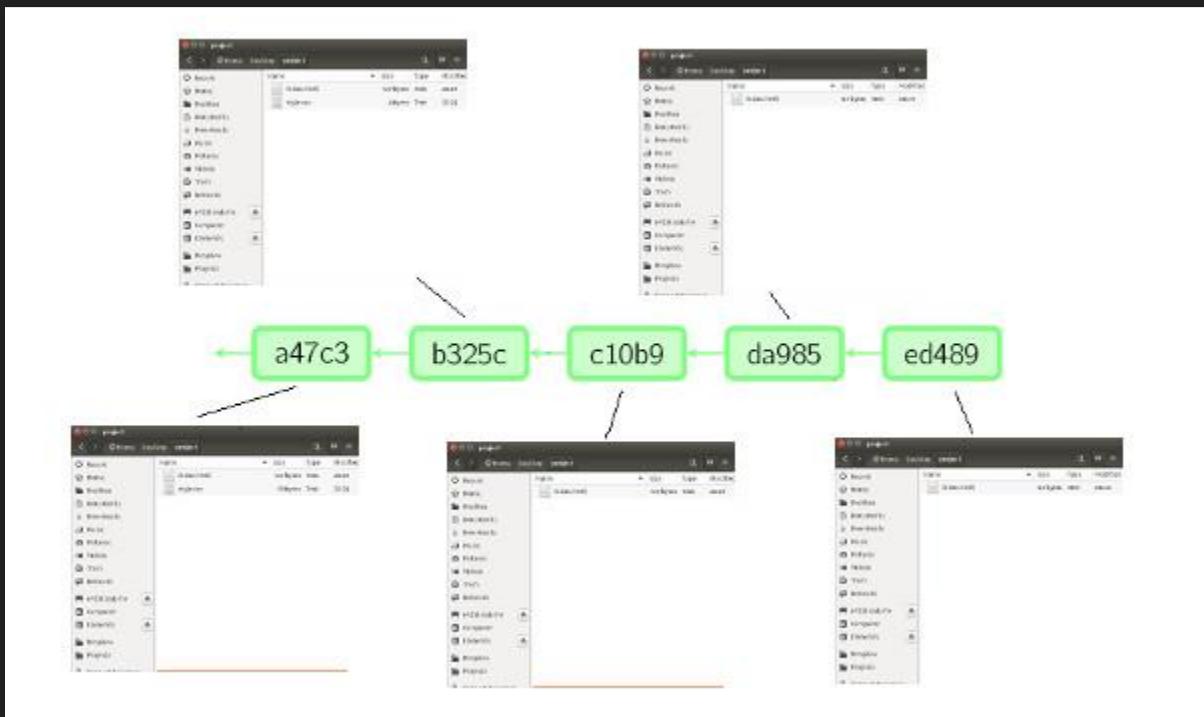


W git commit to snapshot **wszystkich** plików



reprezentowany za pomocą hasha SHA-1

Dzięki temu, można wskazać dowolny commit i podejrzeć cały stan systemu z ówczas, a nie tylko jakie diffy ten commit ze sobą niósł.



Wiedząc jaka jest wersja na produkcji, zawsze można wrócić do niej w kodzie i zrobić drobną poprawkę bez wprowadzania na produkcję kolejnych commitów.

# Toole chętnie przeglądanie snapshota plików wspierają

The screenshot shows a GitHub repository interface. At the top, there's a list of commits from a user named 'stakx' over the last 26 days. Below this is a pull request titled 'Merge pull request #819 from stakx/setupcollection'. Further down is a commit related to 'Flag setups that are known to be overridden'. The bottom part of the screenshot shows a list of messages from various authors, including Joao Moreno, isidor, Andre Weinand, Daniel Imms, and Chirag Bhatia. A dropdown menu is open over the history filter, showing options: 'Simple history (default)', 'First parent', 'Full history', and 'Full history (simplified merges)'. The 'Simple history (default)' option is currently selected.

Author	Message	Authored Date	Commit	Pull Request	Build
Joao Moreno	> support commandPalette menu...	an hour ago	14187e44		
isidor	> localization clarification...	2 hours ago	36cf071f		
Andre Weinand	update node-debug	17 hours ago	422d7ff1		
Daniel Imms	> Merge pull request #18980 from chirag64/master...	2/11/2017	f65c156e		
Chirag Bhatia	> Fixes #17701 - Integrated Terminal Context Menu is triggered via contextmenu ...	2/11/2017	184e7288		

# W PRZECIWIEŃSTWIE DO KRYPTOWALUT...

możesz zmieniać istniejące snapshoty (commity)

ALE!

wiedząc, że jakoś trzeba się z innymi zsynchronizować :)

# GIT NIE ŚLEDZI PLIKÓW

W przeciwieństwie do niektórych VCS,  
git nie śledzi plików, wyłącznie ich zawartość

```
Your branch is ahead of 'origin/gh-pages' by 1 commit.  
(use "git push" to publish your local commits)  
Changes to be committed:  
(use "git reset HEAD <file>..." to unstage)
```

```
  renamed: index.html -> main.html
```

GIT wyświetlając informację o rename pliku,  
tak naprawdę jedynie domyśla się, co  
człowiek zrobił - dla niego jedynie  
część danych się przeniosła, doszczętnie nowy  
plik, a jeden istniejący zniknął

Pytania?

# PRZYDATNE NARZĘDZIA ZWIĄZANE Z GITEM

- GitHub Pages / GitLab Pages / Netlify
  - Np. prezentacja na którą właśnie patrzymy:  
[gitwarsztaty.pl/prezentacja](http://gitwarsztaty.pl/prezentacja)
- GitHub gist

Pytania?

# TAGI

Lekkie(*lightweight*) - tylko wskaźnik z nazwą  
`git tag v1.4`

Opisane(*annotated*) - posiadają opis, sumę kontrolną,  
autora, datę itd.

`git tag -a v1.4 -m 'my version 1.4'`

# PUSHOWANIE TAGÓW

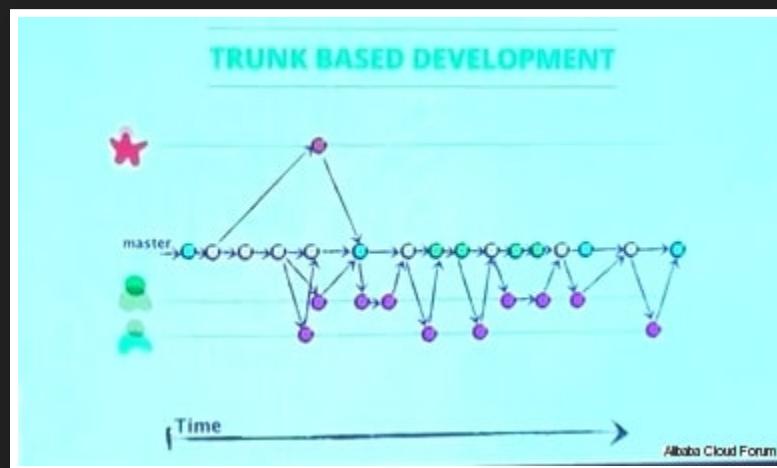
Fetch pobiera wszystkie tagi z serwera, ale domyślnie tagi się nie wypushowują

Dwa sposoby pushowania tagów

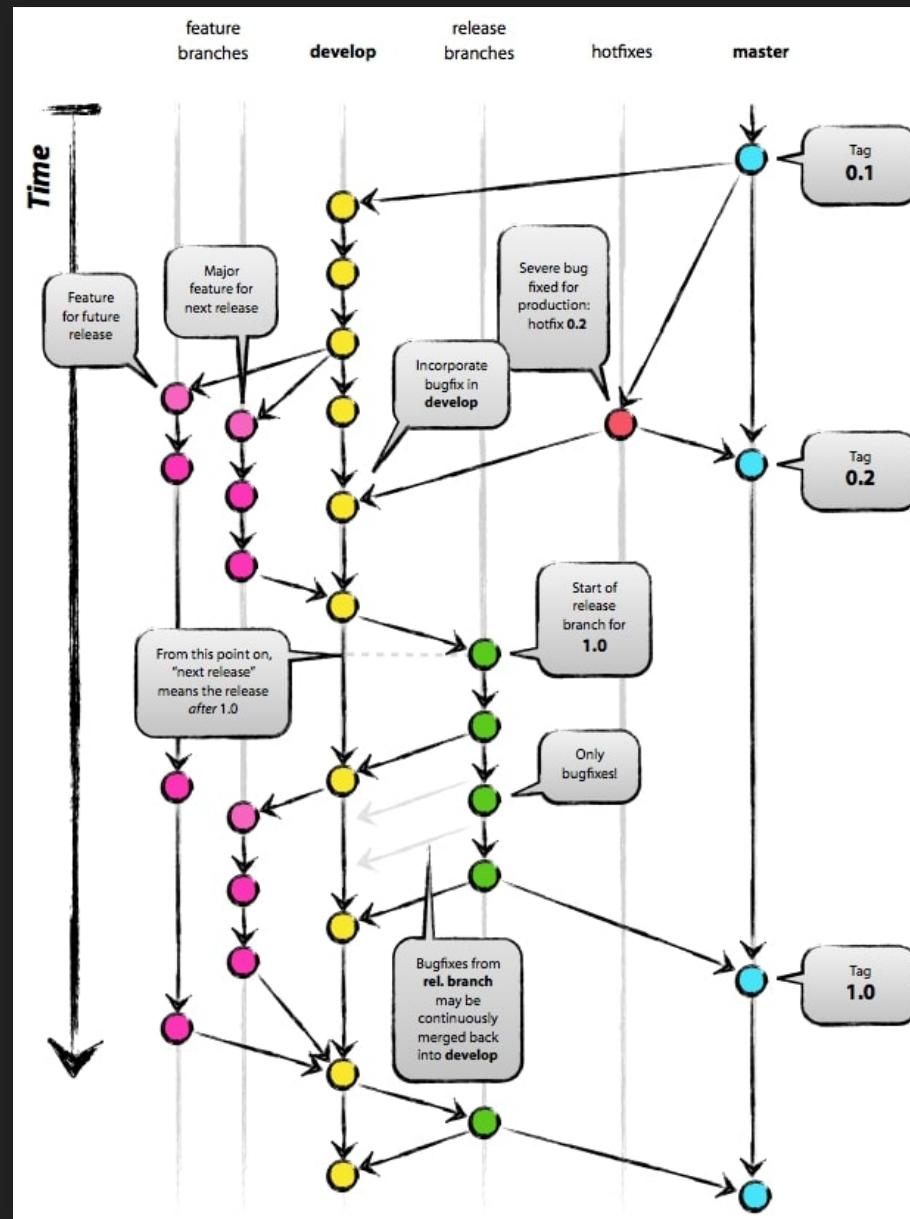
- Zalecany: `git push origin tag_name`
- Silnie odradzany: `git push --tags`

Pytania?

# TRUNK BASED DEVELOPMENT



# GIT FLOW



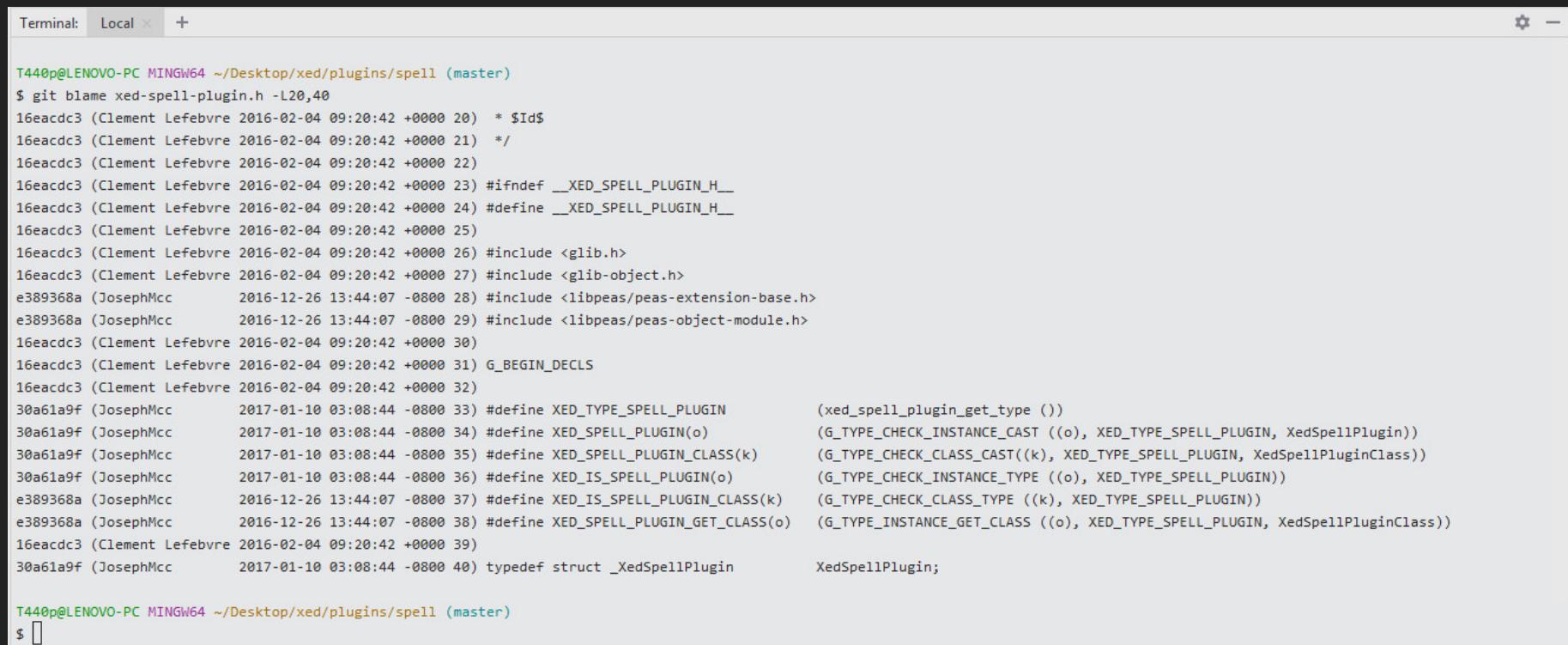
# GIT GUITAR HERO



Pytania?

# GIT BLAME

Który to ?!



The screenshot shows a terminal window with the title bar "Terminal: Local × +". The main area displays the output of the command \$ git blame xed-spell-plugin.h -L20,40. The output lists commits from Clement Lefebvre and JosephMcc, with their commit messages, dates, and times, along with the corresponding blame ranges for specific lines of code. The blame output is color-coded, and the terminal window has a standard Linux-style interface.

```
T440p@LENOVO-PC MINGW64 ~/Desktop/xed/plugins/spell (master)
$ git blame xed-spell-plugin.h -L20,40
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 20) * $Id$
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 21) */
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 22)
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 23) #ifndef __XED_SPELL_PLUGIN_H__
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 24) #define __XED_SPELL_PLUGIN_H__
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 25)
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 26) #include <glib.h>
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 27) #include <glib-object.h>
e389368a (JosephMcc 2016-12-26 13:44:07 -0800 28) #include <libpeas/peas-extension-base.h>
e389368a (JosephMcc 2016-12-26 13:44:07 -0800 29) #include <libpeas/peas-object-module.h>
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 30)
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 31) G_BEGIN_DECLS
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 32)
30a61a9f (JosephMcc 2017-01-10 03:08:44 -0800 33) #define XED_TYPE SPELL_PLUGIN (xed_spell_plugin_get_type ())
30a61a9f (JosephMcc 2017-01-10 03:08:44 -0800 34) #define XED_SPELL_PLUGIN(o) (G_TYPE_CHECK_INSTANCE_CAST ((o), XED_TYPE SPELL_PLUGIN, XedSpellPlugin))
30a61a9f (JosephMcc 2017-01-10 03:08:44 -0800 35) #define XED_SPELL_PLUGIN_CLASS(k) (G_TYPE_CHECK_CLASS_CAST((k), XED_TYPE SPELL_PLUGIN, XedSpellPluginClass))
30a61a9f (JosephMcc 2017-01-10 03:08:44 -0800 36) #define XED_IS SPELL_PLUGIN(o) (G_TYPE_CHECK_INSTANCE_TYPE ((o), XED_TYPE SPELL_PLUGIN))
e389368a (JosephMcc 2016-12-26 13:44:07 -0800 37) #define XED_IS SPELL_PLUGIN_CLASS(k) (G_TYPE_CHECK_CLASS_TYPE ((k), XED_TYPE SPELL_PLUGIN))
e389368a (JosephMcc 2016-12-26 13:44:07 -0800 38) #define XED_SPELL_PLUGIN_GET_CLASS(o) (G_TYPE_INSTANCE_GET_CLASS ((o), XED_TYPE SPELL_PLUGIN, XedSpellPluginClass))
16eacdc3 (Clement Lefebvre 2016-02-04 09:20:42 +0000 39)
30a61a9f (JosephMcc 2017-01-10 03:08:44 -0800 40) typedef struct _XedSpellPlugin XedSpellPlugin;

T440p@LENOVO-PC MINGW64 ~/Desktop/xed/plugins/spell (master)
$ 
```

Pytania?

# ALIASY

Modyfikacje aliasów można wykonywać w gitconfigu

```
[alias]  
st = status  
ci = commit -v
```

Albo za pomocą polecenia

```
git config --global alias.st status  
git config --global alias.ci 'commit -v'
```

Przykład użycia - skrócone wypisywanie statusu

```
git st
```

# INNE PRZYKŁADY ALIASÓW

```
wdiff = diff --word-diff=plain
rh1 = reset HEAD^ --hard
amend = commit --amend -aC HEAD
standup = log --since '1 day ago' --oneline
--author krzysztof.morcinek@gmail.com # hack
it with your email
cam = commit -am
jira = log --since '6am' --oneline --author
krzysztof.morcinek@gmail.com # hack it with
your email
ls = log --
pretty=format:"%C(yellow)%h%Cred%d\\\
%Creset%s%Cgreen\\ [%cn]" --decorate
mt = mergetool
```

Pytania?

# WIELE REMOTÓW

```
git remote add anotherOrigin https...
```

```
git remote -v
```

```
git push anotherOrigin
```

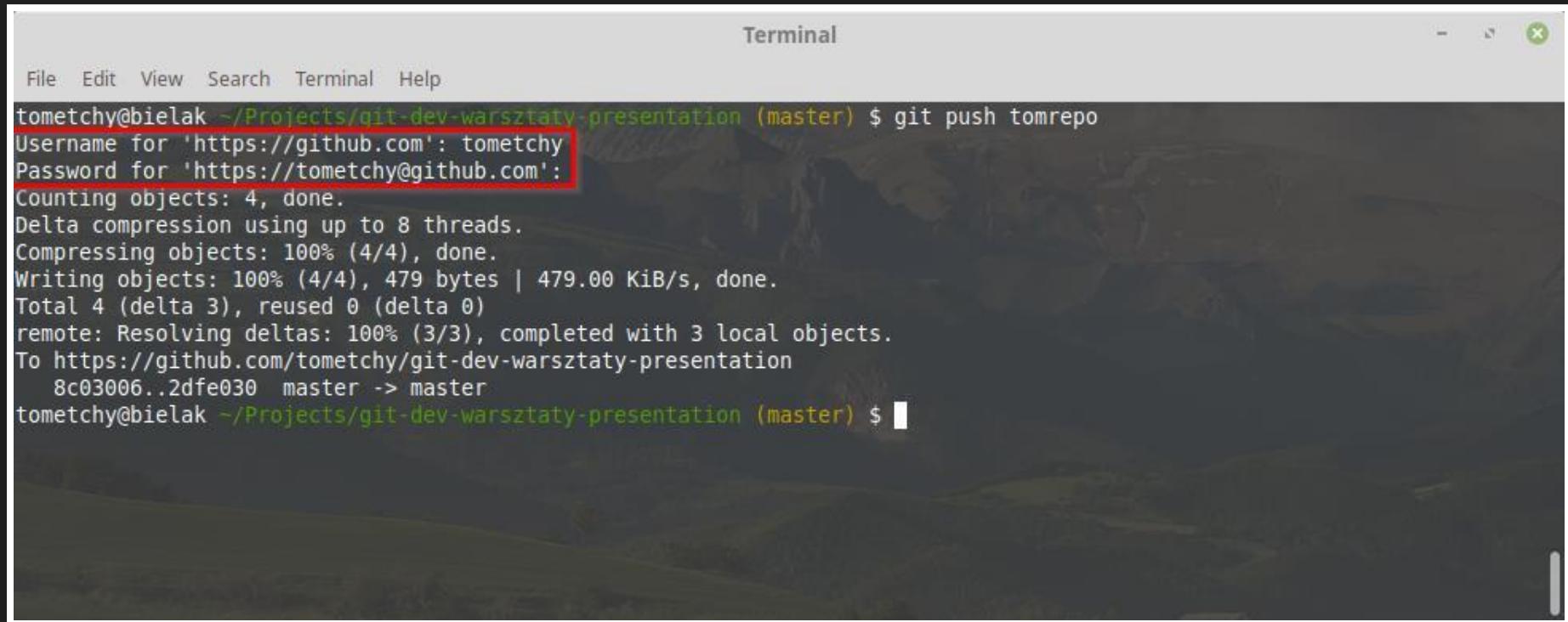
```
git fetch anotherOrigin
```

```
git push -u anotherOrigin your_branch
```

Pytania?

# CREDENTIALS HELPER

Każdy push, fetch (czyli również pull), wymaga autoryzacji...



A screenshot of a terminal window titled "Terminal". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows the following command and its execution:

```
tometchy@bielak ~/Projects/git-dev-warsztaty-presentation (master) $ git push tomrepo
Username for 'https://github.com': tometchy
Password for 'https://tometchy@github.com':
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 479 bytes | 479.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/tometchy/git-dev-warsztaty-presentation
  8c03006..2dfe030 master -> master
tometchy@bielak ~/Projects/git-dev-warsztaty-presentation (master) $ █
```

The password input field is highlighted with a red rectangle.

# WBUDOWANE CREDENTIALS HELPERY

- **cache**

```
git config --global credential.helper 'cache --timeout=300'  
Bezpieczny, ale tylko tymczasowy (in memory).  
Opcjonalnie własny timeout w sekundach (domyślny 900 - 15minut).
```

- **store**

```
git config --global credential.helper store  
Wygodny, ale niebezpieczny - credentiale zapisane w pliku tekstowym niezaszyfrowane.
```

# LIBSECRET - WYGODNY I BEZPIECZNY CREDENTIALS HELPER DLA LINUX

Przykładowa instalacja dla Linux Mint / Ubuntu

1. sudo apt-get install libsecret-1-0 libsecret-1-dev
2. cd /usr/share/doc/git/contrib/credential/libsecret
3. sudo make

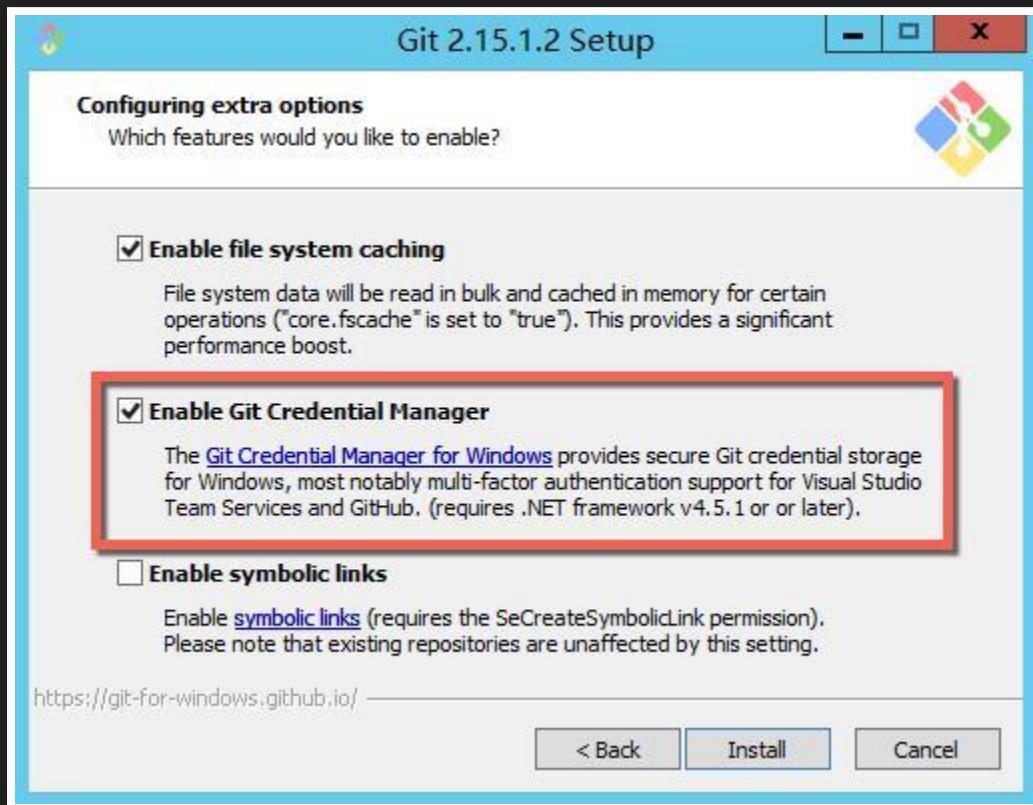
Konfiguracja (gdy już zainstalowany)

```
git config --global credential.helper  
/usr/share/doc/git/contrib/credential/libsecret/git-credential-libsecret  
Szerszy opis na blogu SoftwareDeveloper.Blog
```

# WYGODNY I BEZPIECZNY CREDENTIALS HELPER DLA WINDOWS

Instalator: [github.com/Microsoft/Git-Credential-Manager-for-Windows/releases](https://github.com/Microsoft/Git-Credential-Manager-for-Windows/releases)

Lub *checkbox* w trakcie instalacji git-a:



Powinno ustawić: `git config --global credential.helper manager`  
albo ścieżkę do exe

Pytania?

# STRATEGIE MERGOWANIA

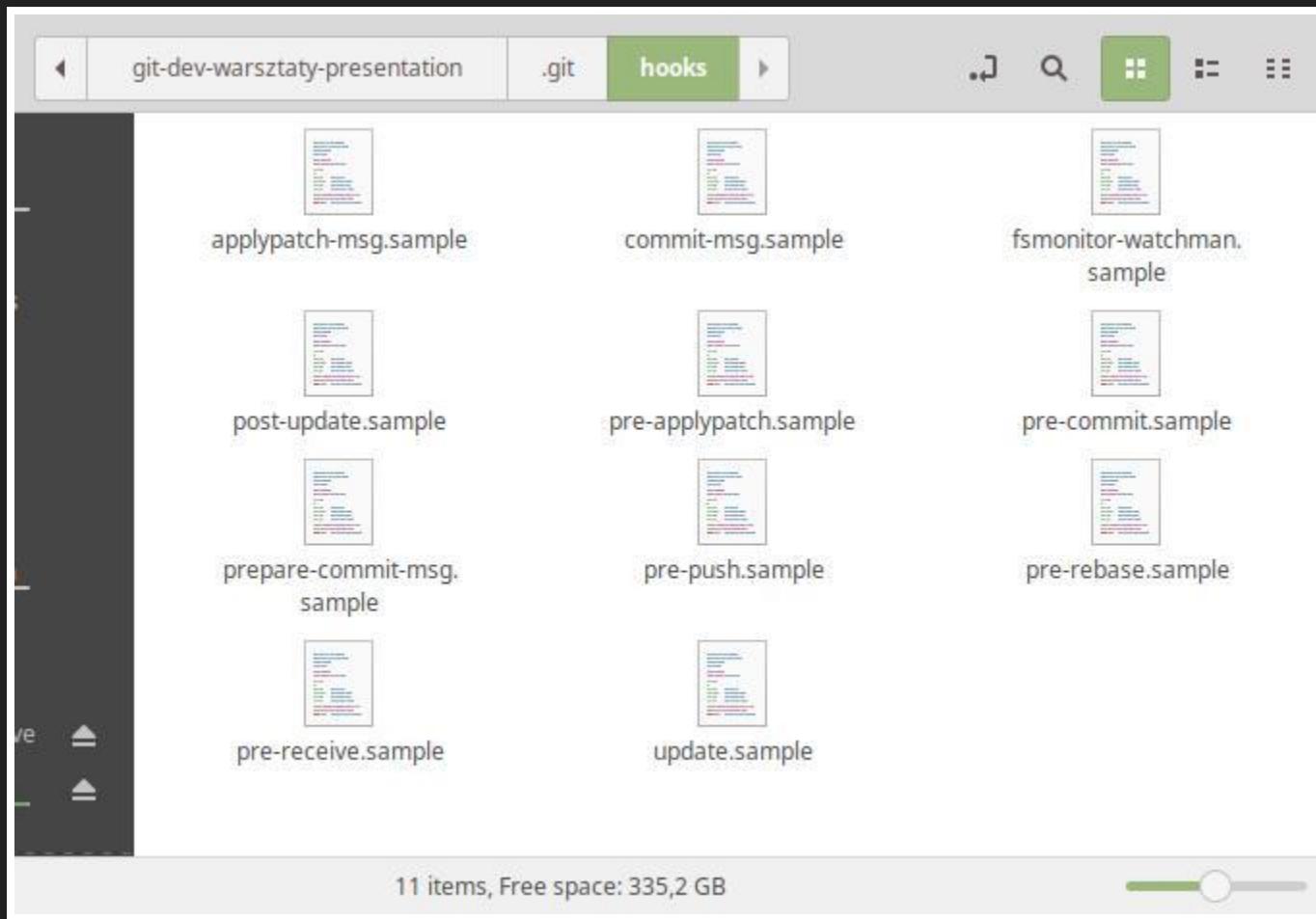
- resolve
- **recursive**
  - ours
  - theirs
  - patience
  - diff-algorithm=[patience|minimal|histogram|myers]
  - ignore-space-change
  - ignore-all-space
  - ignore-space-at-eol
  - ignore-cr-at-eol
  - renormalize
  - no-renormalize
  - no-renames
  - **find-renames[=n]**
  - rename-threshold=*n*
  - subtree[=*path*]
- octopus
- ours
- subtree

# PODPIS KRYPTOGRAFICZNY

Commity i tagi można podpisywać kryptograficznie, jednak wymaganie tego w workflow jest kłopotliwe

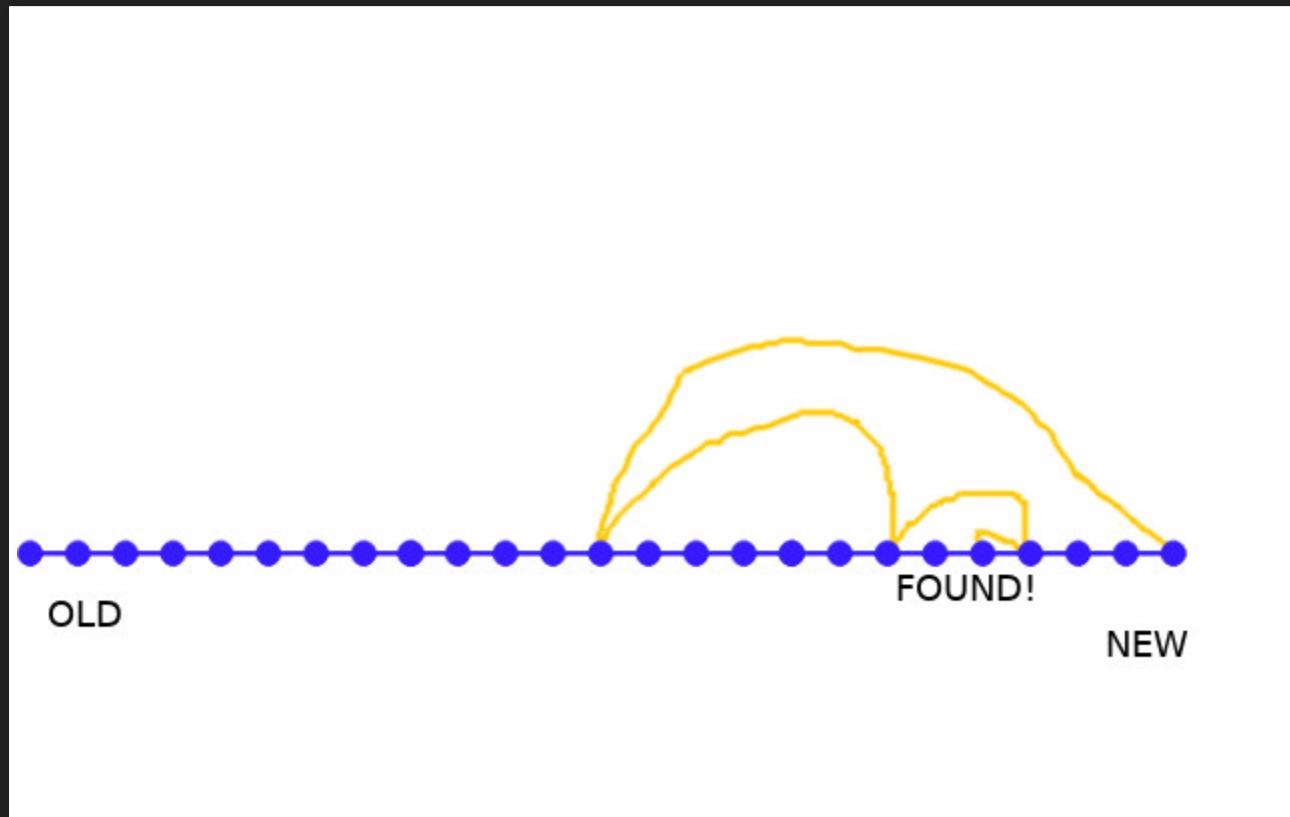
```
$ git log --show-signature
commit 5c3386cf54bba0a33a32da706aa52bc0155503c2
gpg: Signature made Wed Jun 4 19:49:17 2014 PDT using RSA key ID 0A46826A
gpg: Good signature from "Scott Chacon (Git signing key) <schacon@gmail.com>"
Author: Scott Chacon <schacon@gmail.com>
Date: Wed Jun 4 19:49:17 2014 -0700
Add new function
```

# GIT HOOKS



# GIT BISECT

AUTOMATYZACJA SZUKANIA COMMITA, KTÓRY  
WPROWADZIŁ BŁĄD



# Linki

Pro Git book (Scott Chacon, Ben Straub)

---

How to Write a Git Commit Message

(The seven rules of a great Git commit message)

---

Atlassian Tutorials

---

Learn git branching

---

Visualise git with D3

---

Oh shit, git!

---

How to undo (almost) anything with git

---

GIT Illustrated Cheatsheet

---

Artykuł na co uważać przy rebase

---

# DZIĘKUJEMY!