Kreatywne programowanie

Zajęcia nr 1

GIT - System Kontroli wersji
https://git-scm.com

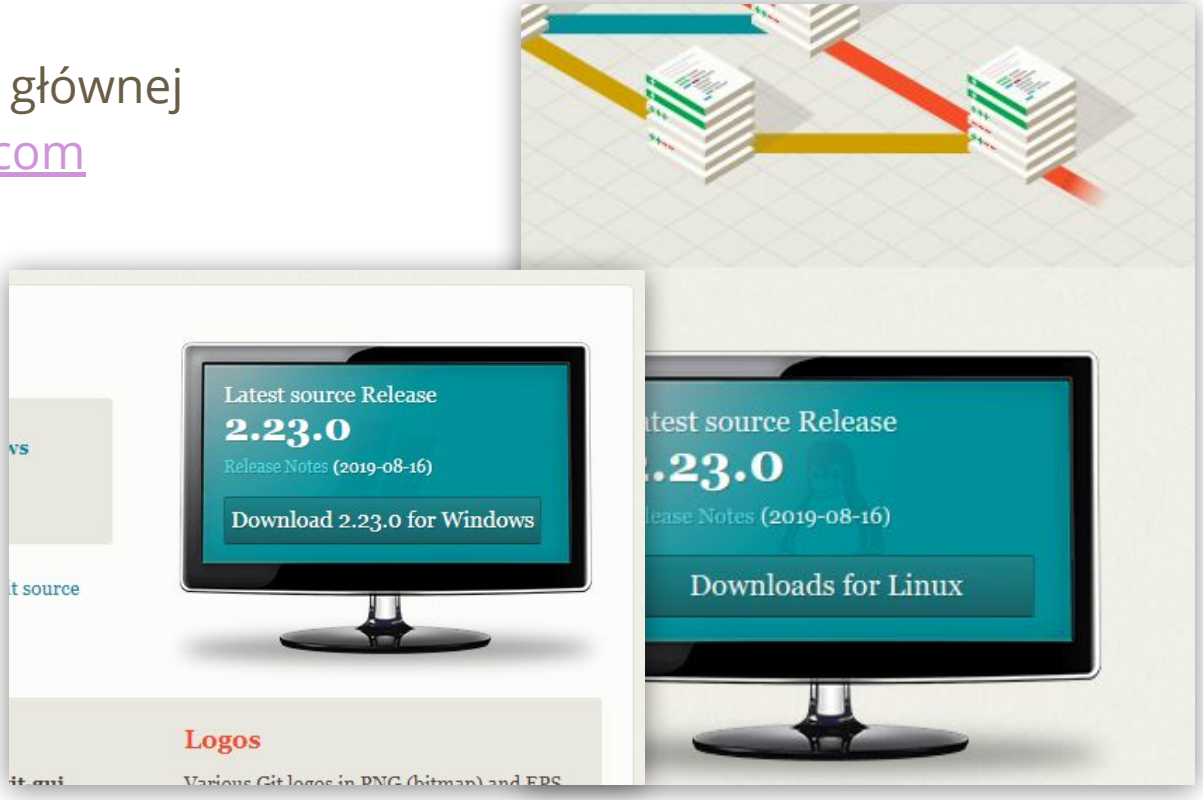
Piotr Radzikowski
ptr.radzikowski@gmail.com
radzikowski.one.pl

Organizacja pracy przy projekcie

- trello.com
- facebook.com
- git (github.com)

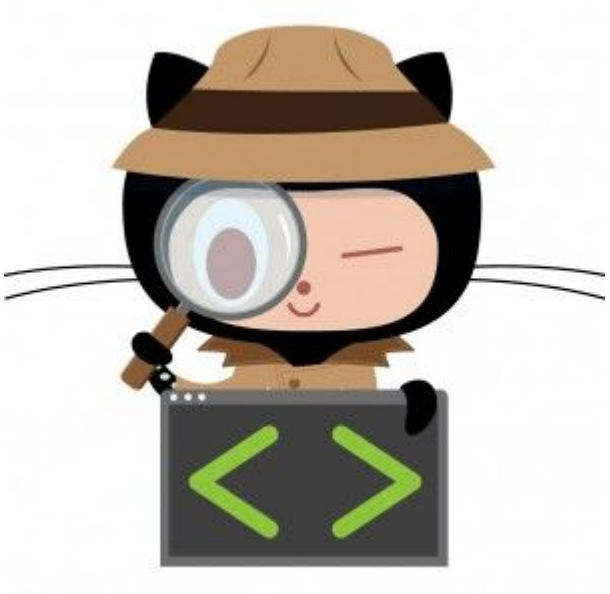
Git jest git

Ściągnijmy gita ze strony głównej projektu https://git-scm.com



VSC - Version Control System

System Kontroli Wersji *śledzi wszystkie zmiany dokonywane na pliku (lub plikach) i umożliwia przywołanie dowolnej wcześniejszej wersji.*

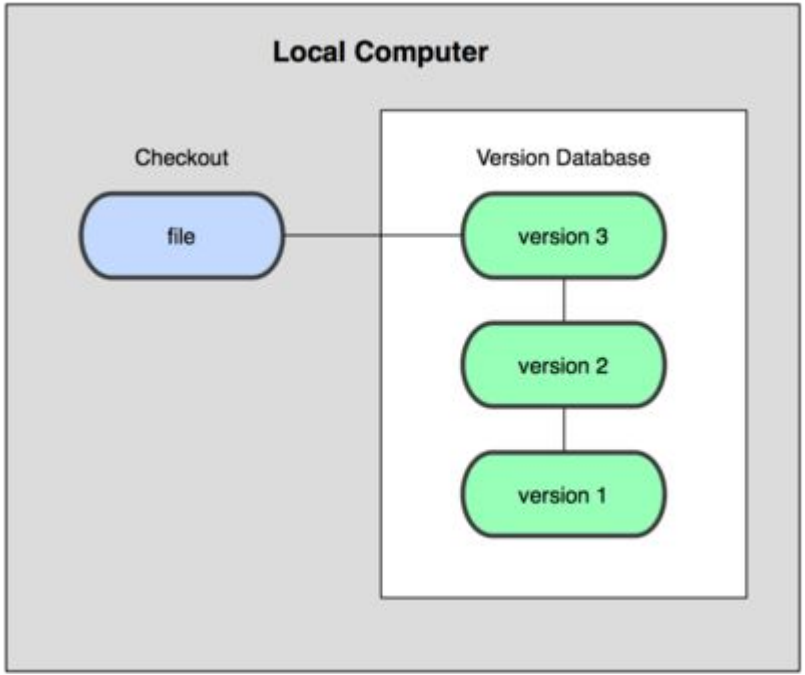


Wyróżniamy kilka typów systemów kontroli wersji

- lokalny system kontroli wersji
- scentralizowany system kontroli wersji
- rozproszony system kontroli wersji

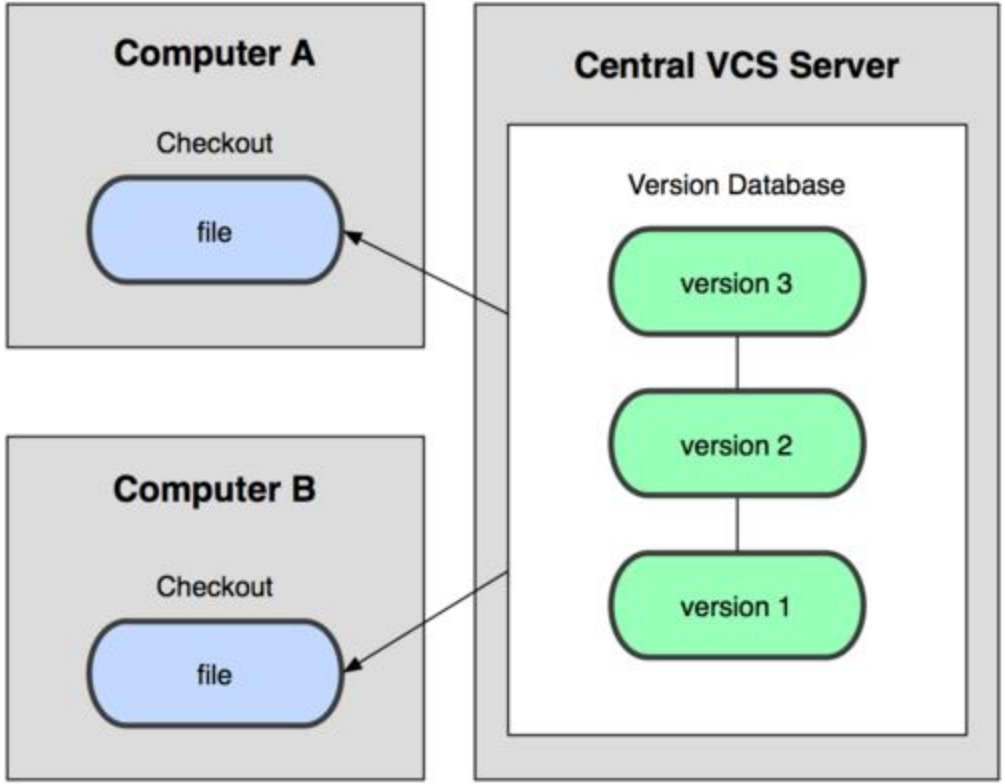
Local Control System Version

Metoda kontroli wersji polegająca na kopiowaniu plików do innego katalogu (może nawet oznaczonego datą, jeśli są sprytni).



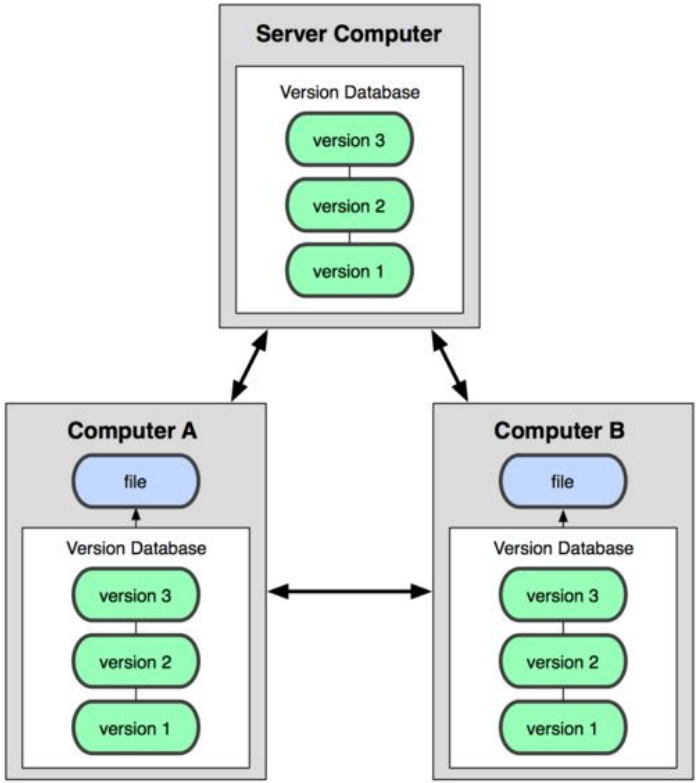
CVCS - Central Version Control System

Przykładem jest Subversion(SVN) lub TFS czy kiedyś bardzo popularny CVS



DVCS - Distributed Version Control System

Rozproszony system kontroli wersji
(DVCS - Distributed Version Control System).
Przykładem jest np. Git i Mercurial (zwany też HG)



Każda maszyna przechowuje całe repozytorium

GIT

- rozproszony system kontroli wersji (DVCS)
- “narodził się” 7 kwietnia 2005 roku
- autor: Linus Torvalds
- rewolucja w systemie kontroli wersji

Instalacja gita

- zaznaczenie opcji integracji z domyślny cmd

Konfiguracja gita

Zmienne identyfikacyjne i konfiguracja gita

- przełącznik “global”

```
✗ git config --global user.email "your email@example.com"
```

```
✗ git config --global user.name "Imie Nazwisko"
```

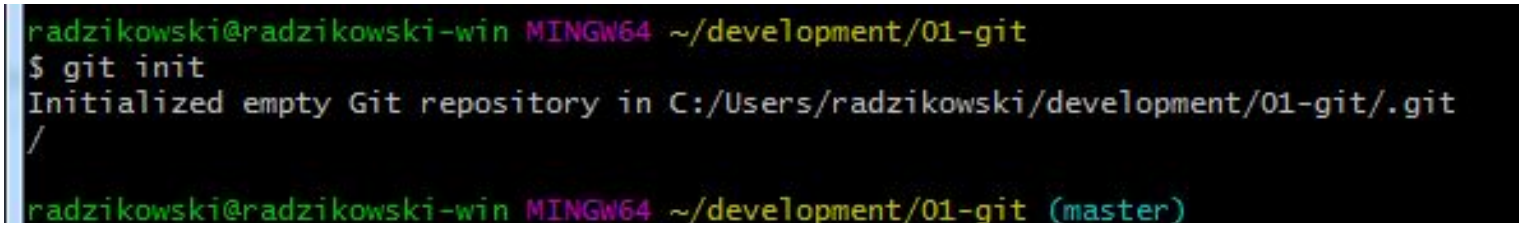
Repozytorium (potocznie: repo)

Katalog zawierający wszystkie informacje o aktualnym stanie projektu i jego historii.

Może zawierać odwołania do innych repo.

git init

Tworzenie (inicjalizacja) projektu opartego na git, możliwe jest wykonanie tej operacji w dowolnym katalogu, pustym lub z istniejącym już projektem.



README.md

markdown - pseudo język do formatowania tekstu.

```
It's very easy to make some words **bold** and other words
*italic* with Markdown. You can even
[link to Google!](http://google.com)
```

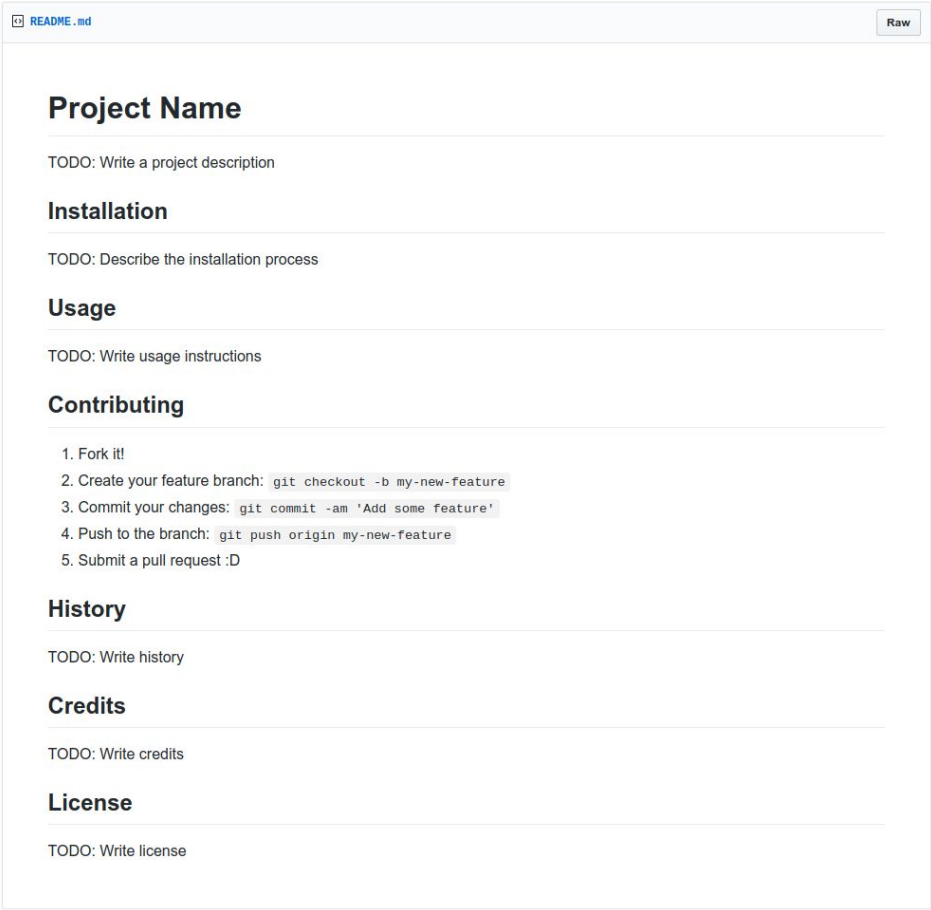
It's very easy to make some words **bold** and other words *italic* with Markdown. You can even link to Google!

https://guides.github.com/features/mastering-markdown/

README.md

Plik opisuje jak korzystać z projektu, oraz dodatkowe przydatne informacje na temat projektu, np.

- jak go uruchomić na komputerze
- jak uruchomić testy
- z czego składa się projekt (biblioteki)


README.md

Raw

Project Name

TODO: Write a project description

Installation

TODO: Describe the installation process

Usage

TODO: Write usage instructions

Contributing

1. Fork it!
2. Create your feature branch: `git checkout -b my-new-feature`
3. Commit your changes: `git commit -am 'Add some feature'`
4. Push to the branch: `git push origin my-new-feature`
5. Submit a pull request :D

History

TODO: Write history

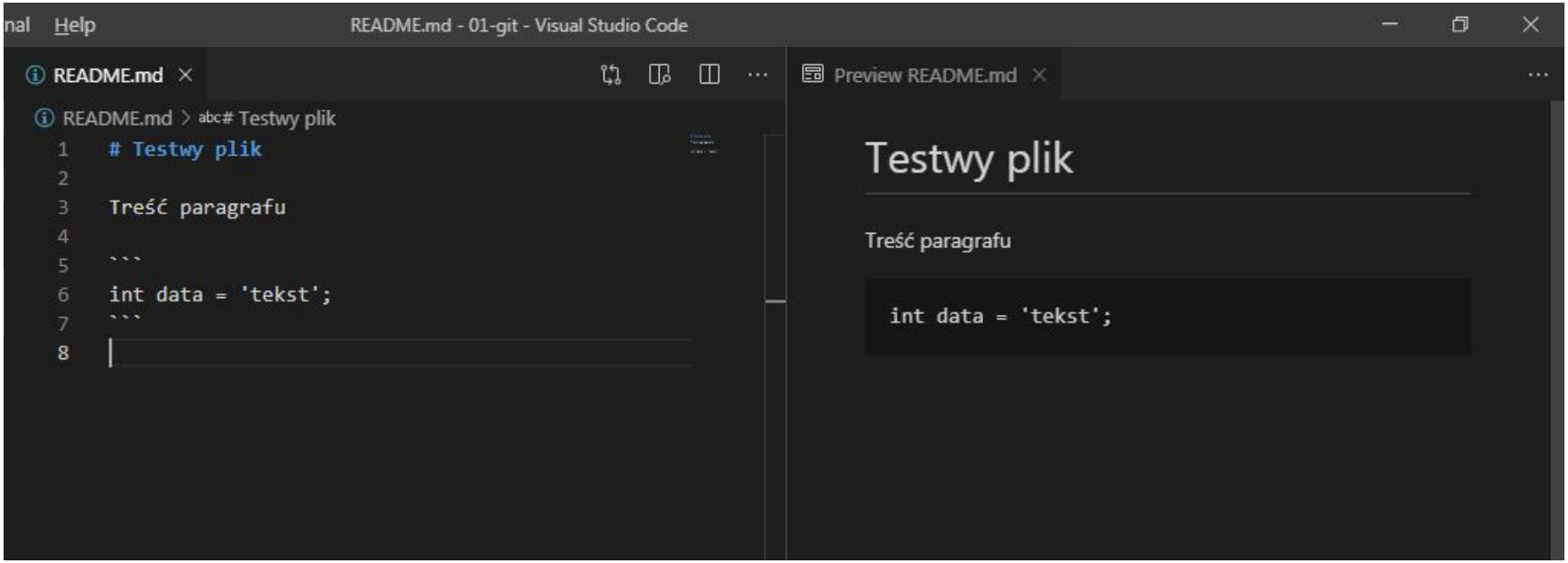
Credits

TODO: Write credits

License

TODO: Write license

https://guides.github.com/features/mastering-markdown/


nal Help
README.md - 01-git - Visual Studio Code
README.md
Preview README.md
README.md > abc# Testwy plik
1 # Testwy plik
2
3 Treść paragrafu
4
5 ```
6 int data = 'tekst';
7 ```
8
Testwy plik
Treść paragrafu
int data = 'tekst';

git status

Polecenie do ukazania zmian plików wraz z kolorowaniem ukazującym jakie pliki zostały zmienione.

```
→  01-git git:(master) ✗ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        readme.md

nothing added to commit but untracked files present (use "git add" to track)
→  01-git git:(master) ✗
```

git add

Polecenie służy do dodawania pliku/plików do naszego repozytorium.

```
→  01-git git:(master) ✗ git add
Nothing specified, nothing added.
Maybe you wanted to say 'git add .'?
→  01-git git:(master) ✗
```

```
→  01-git git:(master) ✗ git add readme.md
→  01-git git:(master) ✗
```

```
→  01-git git:(master) ✗ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        nowy plik:     readme.md

→  01-git git:(master) ✗
```

git commit

Zapisanie wersji względem zmian, tworzenie “historii” zmian w repozytorium

```
→  01-git git:(master) ✗ git commit
```

```
→  01-git git:(master) git commit -m "init readme file"
```

```
✗ git commit . -m "init readme file"
```

```
✗ git commit .
```

core.editor

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

Basic Vim Commands

- :w Write the current file
- :wq Write the current file and exit.
- :q! Quit without writing
- To change into insert mode: i or a
  - Use escape to exit
- search forward /, repeat the search backwards: N
- Basic movement:
  - h l k j character left, right; line up, down (also arrow keys)
  - b w word/token left, right
  - ge e end of word/token left, right
  - 0 $ jump to first/last character on the line
- x delete
- u undo

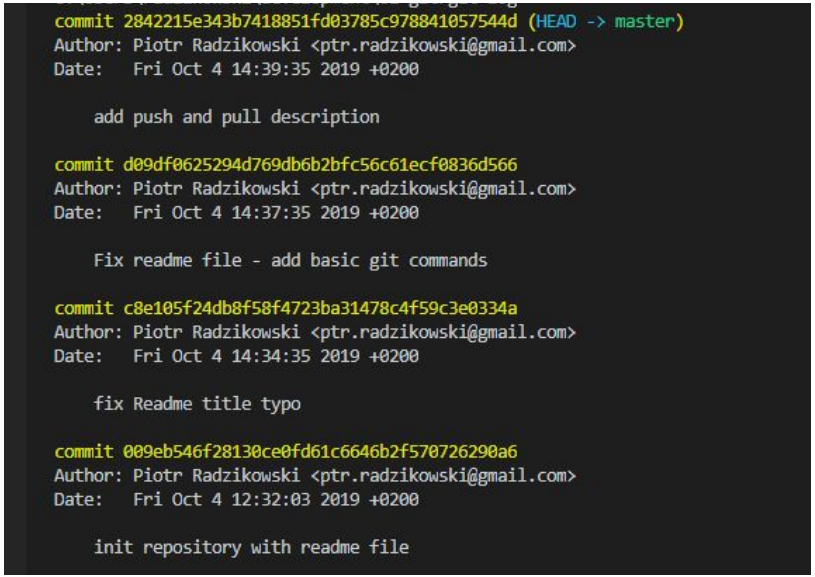
https://wiki.gentoo.org/wiki/Vim/Guide and http://tnerual.eriogerg.free.fr/vimqrc.pdf

25

git log

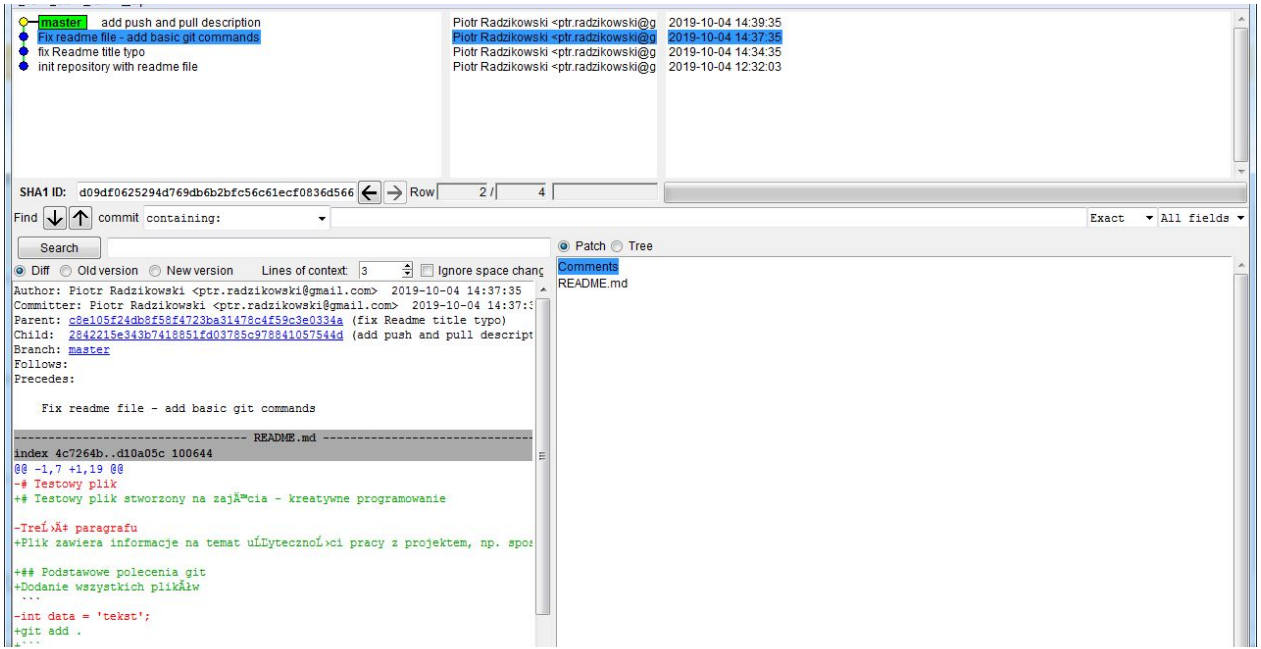
Historia kto/co/kiedy?

można prześledzić proces powstawania każdej funkcji



gitk

graficzny interfejs do reprezentacji historii i gałęzi* gita



git diff

```
✗ git diff
```

```
diff --git a/readme.md b/readme.md
index 3d79c13..66d5f7a 100644
--- a/readme.md
+++ b/readme.md
@@ -1,6 +1,13 @@
-# test
-asd
+# Testowy projekt GIT
+Projekt powstał na potrzeby zajęc z kraatywnego programowania

+## podstawowe komendy
 ```
-var code = 'abc';
+git commit -m "wiadomosc"
 ```
+```
+git push
+```
+```
+git pull
+```
\ No newline at end of file
(END)
```

git push

Komenda do wysyłania zmian które zostały przez nas zapisane (za-commit-owane) w naszym projekcie, do katalogu w internecie*

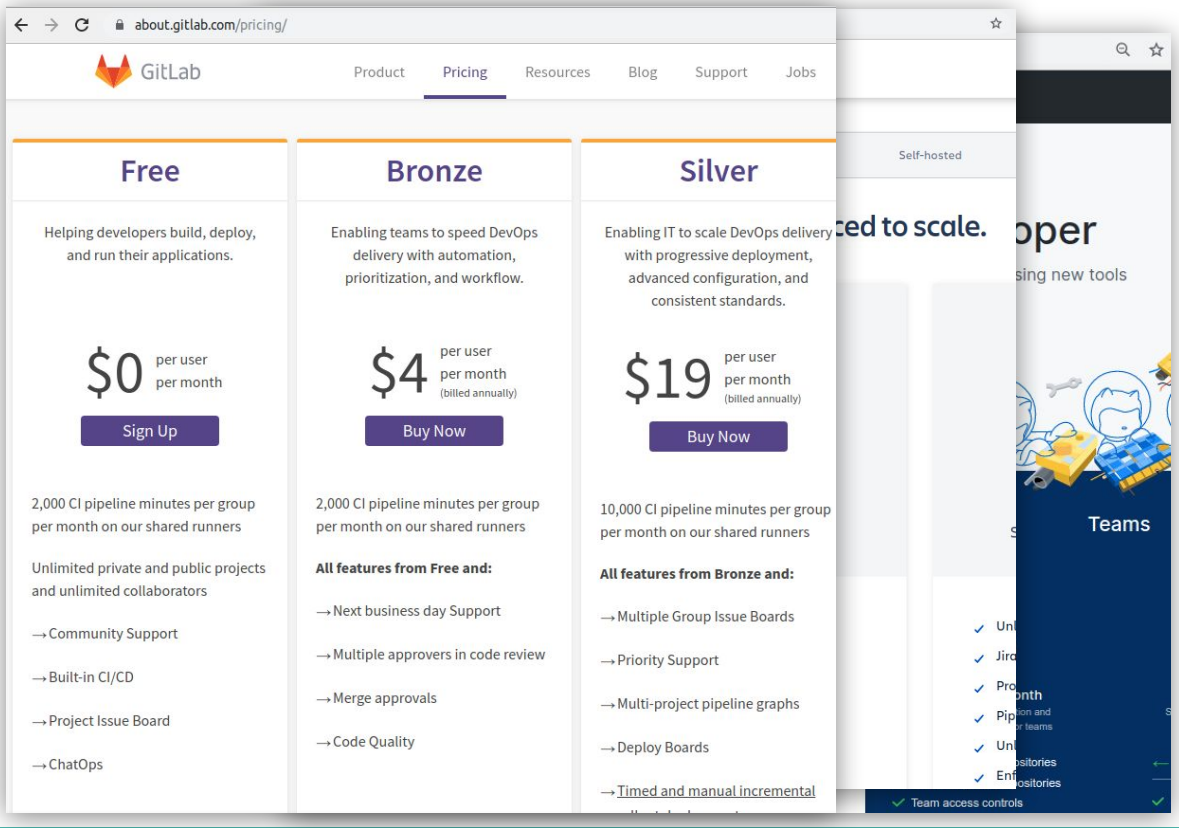
```
C:\Users\radzikowski\development\01-git>git push
fatal: No configured push destination.
Either specify the URL from the command-line or configure a remote repository using

    git remote add <name> <url>

and then push using the remote name

    git push <name>
```

Zanim zapiszemy efekty naszych prac

Musimy wskazać dla gita (git-owi :)) gdzie te zmiany muszą być wysłane

Portale które wspierają protokół GIT

- github.com
- bitbucket.org
- gitlab.com



Quick setup — if you've done this kind of thing before

or HTTPS SSH `git@github.com:radzikowski/01-kreatywne-programowanie.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# 01-kreatywne-programowanie" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:radzikowski/01-kreatywne-programowanie.git
git push -u origin master
```

…or push an existing repository from the command line

```
git remote add origin git@github.com:radzikowski/01-kreatywne-programowanie.git
git push -u origin master
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

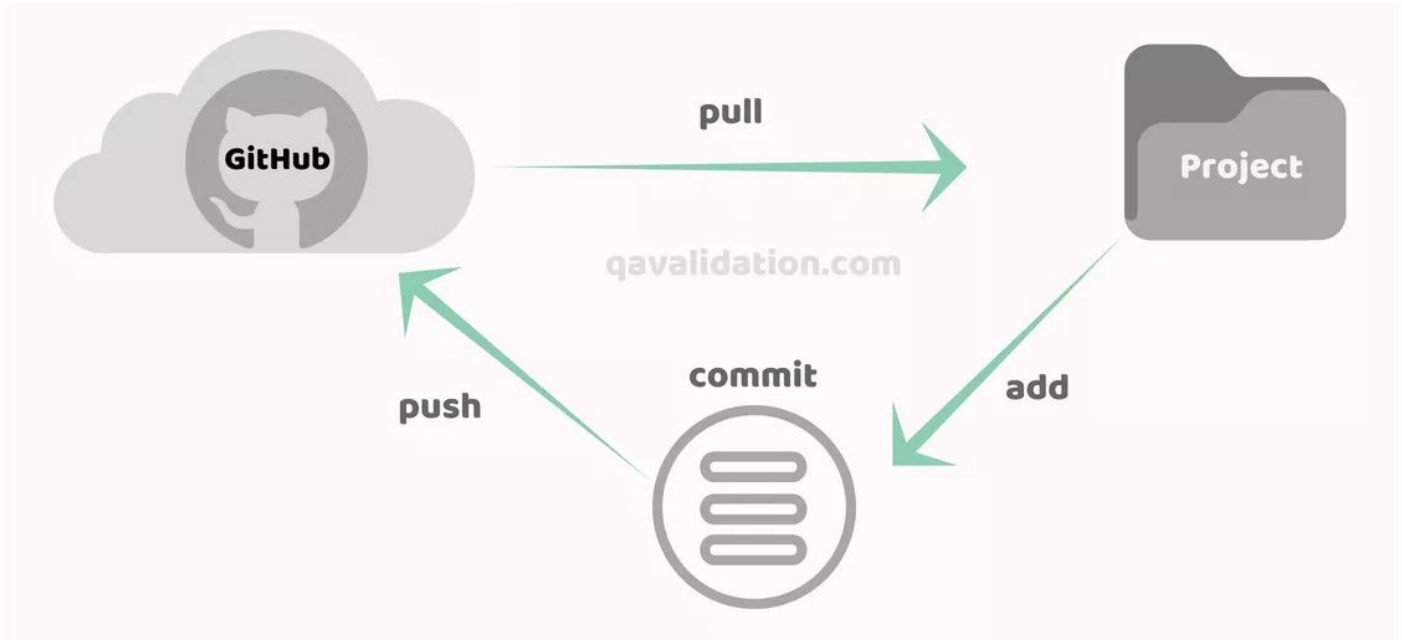
Import code

git push

Wysłanie zmian na nasze repozytorium.

git pull

Pobranie zmian z naszego repozytorium



Zapamiętanie hasła w systemie

- **Windows**: Git Windows Credentials Manager
  - git config --global credential.helper wincred

ssh-keygen

https://help.github.com/en/enterprise/2.16/user/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Generating a new SSH key

1. Open Git Bash.
2. Paste the text below, substituting in your GitHub Enterprise email address.

```
$ ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

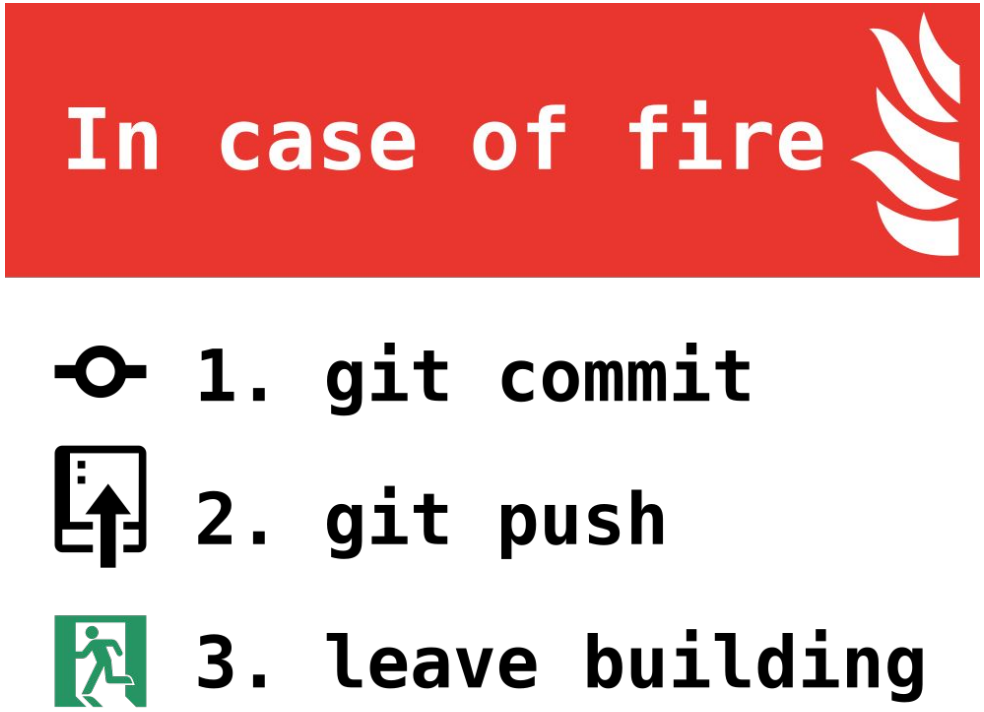
This creates a new ssh key, using the provided email as a label.

```
> Generating public/private rsa key pair.
```

Podsumowanie

Poznaliśmy komendy:

- git init
- git status
- git add
- git commit
- git log
- gitk
- git diff
- git push
- git pull



Dziękuje za uwagę!

Co na następnych zajęciach?

Piotr Radzikowski
ptr.radzikowski@gmail.com
radzikowski.one.pl