

# Sprawozdanie z Laboratorium SCR – sieci komputerowe

## Git – system kontroli wersji

Wykonał:

Mateusz Perdek

Prowadząca:

Mgr. Inż. Paulina Russak

Kod grupy:

E02-43i

Termin zajęć:

wt/TN godz. 17:05-18:45

### 1. Wstęp teoretyczny

Git – rozproszony system kontroli wersji. Stworzył go Linus Torvalds jako narzędzie wspomagające rozwój jądra Linux. Git stanowi wolne oprogramowanie i został opublikowany na licencji GNU GPL w wersji 2. Pierwsza wersja narzędzia Git została wydana 7 kwietnia 2005 roku, by zastąpić poprzednio używany w rozwoju Linuksa, niebędący wolnym oprogramowaniem, system kontroli wersji BitKeeper.

### 2. Cele ćwiczenia

Zapoznanie się z systemem kontroli wersji GIT

### 3. Przydatne polecenia w systemie GIT

1. git status - status wszystkich plików
2. git add - rozpoczęcie śledzenia pliku
3. git init - inicjalizacja repozytorium
4. git commit - wprowadzenie zmian
5. git diff - wpokazuje co dokładnie zostało zmienione
6. git rm - usunięcie pliku z gita

### 4. Przebieg laboratorium

- Zainstalowano gita ze strony : <http://msysgit.github.com/> dla windows.

- Stworzono nowe repozytorium

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner: Mutungamiri / Repository name:  ✓

Great repository names are short and memorable. Need inspiration? How about [redesigned-chainsaw?](#)

Description (optional):

☒ **Public**  
Anyone can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore:  | Add a license:  ⓘ

[Create repository](#)

- W folderze zaczynamy od inicjalizacji repozytorium komendą git init.

```
Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git init
Initialized empty Git repository in C:/Users/Mutungamiri/Desktop/ScrLab4/.git/

Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$
```

- Teraz musimy się połączyć z naszym zdalnym repozytorium za pomocą komendy git remote.

```
Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git remote https://github.com/Mutungamiri/SCR-lab
error: Unknown subcommand: https://github.com/Mutungamiri/SCR-lab
usage: git remote [-v | --verbose]
       or: git remote add [-t <branch>] [-m <master>] [-f] [--tags | --no-tags]
       mirror=<fetch|push> <name> <url>
```

- Sprawdzamy, czy wszystko idzie zgodnie z planem komendą git status.

```
Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git status
On branch master

No commits yet

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

        index.html
        readme

nothing added to commit but untracked files present (use "git add" to track)

Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
```

- Używamy komendy `git add .` – kropka oznacza dodanie wszystkich plików w folderze do utworzonego repozytorium.

```

index.html
readme

nothing added to commit but untracked files present (use "git add" to track)
utungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git add .
utungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git status
On branch master

no commits yet

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

        new file:   index.html
        new file:   readme

utungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$

```

- Podajemy dane do logowania z Githuba, a następnie dodajemy nasz pierwszy commit oraz krótki komentarz co się dzieje w danym commicie. ( `git config` i `git commit` ).

```

Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git config user.name "Mutungamiri"

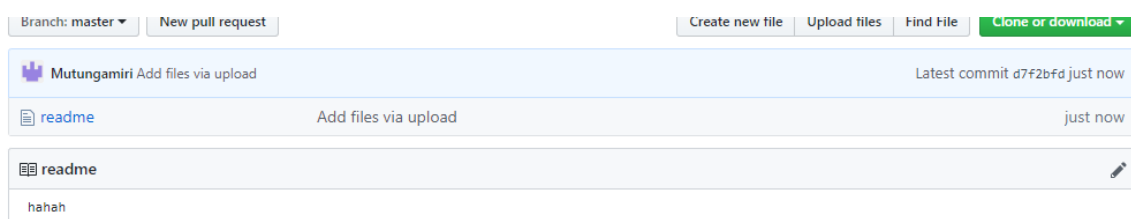
Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$ git commit -m "nowepliki"
[master (root-commit) 0ad2114] nowepliki
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
 create mode 100644 readme

Mutungamiri@DESKTOP-VO60HKV MINGW64 ~/desktop/ScrLab4 (master)
$

```

- Przy użyciu komendy `git push` wpychamy dokonane zmiany (w tym wypadku utworzone pliki) na zewnątrz.

- Teraz są już one widoczne w naszym repozytorium na GitHubie.



- Pobieramy nasze repozytorium przy użyciu komendy `git clone` do innego folderu.
- Wprowadzamy zmiany w pliku `index.html`, dodajemy plik `README.md` i wpychamy z nowego miejsca zmiany na zewnątrz (`git push`).

## 4. Wnioski

- GIT jest bardzo przydatnym i łatwym do obsługi systemem do kontroli wersji.
- Służy do śledzenia zmian w plikach i koordynowania pracy nad nimi.
- Wersjonowanie nie ogranicza się do kodu źródłowego, równie dobrze może tam trafić praca dyplomowa, wpisy z pamiętnika czy teksty z bloga.
- Bardzo przydatne narzędzie dla programistów.