Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №1

з дисципліни «Технології розробки програмного забезпечення»

Системи контроля версій. Розподілена система контролю версій «Git»

Виконав:
студент групи ІА-34
Швець Роман Вадимович

Перевірив:
асистент кафедри ІСТ
Мягкий Михайло Юрійович

Київ - 2025

Теоретичні відомості

Git - це розподілена система керування версіями, яка дає змогу відстежувати історію змін у проєкті, працювати з різними варіантами файлів та організовувати ефективну командну співпрацю розробників.

Переваги використання Git у роботі програміста:

- Керування версіями - усі зміни зберігаються, тому завжди можна відновити попередній стан коду.
- Спільна робота - кілька розробників можуть одночасно вносити зміни в один проєкт без ризику втрати інформації.
- Гнучке тестування - створюються окремі гілки для нових можливостей, які можна перевірити, а після цього інтегрувати в основний код.

Основні поняття:

- Репозиторій - це сховище проєкту, яке містить усі файли та повну історію змін. Створюється за допомогою команди: git init.
- Коміт (commit) - збережений знімок (версія) стану проєкту в певний момент часу. Сукупність комітів формує історію змін. Виконується командою: git commit -m "Опис змін".
- Гілка (branch) - окрема лінія розвитку проєкту, що дає змогу працювати над різними задачами незалежно одна від одної.
- Перегляд доступних гілок: git branch.
- Перемикання між гілками: git switch branch_name або git checkout branch_name.
- Основна гілка - зазвичай має назву main або master і зберігає стабільну робочу версію проєкту.

Створення комітів

Git відслідковує зміни у два кроки:

1. Індекс (staging area) — проміжне середовище, куди додаються змінені файли перед збереженням. Приклад: git add file.txt
2. Фіксація змін (commit) — створення коміту, який зберігає всі файли, додані в індекс. Приклад: git commit -m "Додано файл file.txt"

За потреби можна зробити й порожній коміт (без доданих файлів): git commit --allow-empty -m "Initial commit"

Робота з гілками

Існує низка способів створити нову гілку:

1. git branch b1 – створює гілку b1 без переходу в неї.
2. git checkout -b b1 – створює гілку b1 і переходить в неї.
3. git switch -c b1 – сучасна команда для створення гілки і переходу в неї.

Для перевірки поточних гілок використовується команда git branch.

Хід роботи

Створюємо новий локальний репозиторій:

```
C:\Users\User\Documents\GitHub> git init labrob1
Initialized empty Git repository in C:/Users/User/Documents/GitHub/labrob1/.git/
```

Додаємо довільний текстовий файл:

```
C:\Users\User\Documents\GitHub> cd labrob1

C:\Users\User\Documents\GitHub\labrob1> echo "text" > lab1.txt
```

Фіксуємо додавання файлу:

```
C:\Users\User\Documents\GitHub\labrob1> git add lab1.txt

C:\Users\User\Documents\GitHub\labrob1> git commit -m "Text added"
[master (root-commit) 7945a77] Text added
 1 file changed, 1 insertion(+)
 create mode 100644 lab1.txt
```

Створюємо нову директорію та додаємо туди довільний файл:

```
C:\Users\User\Documents\GitHub\labrob1>mkdir extra_dir

C:\Users\User\Documents\GitHub\labrob1>cd extra_dir

C:\Users\User\Documents\GitHub\labrob1\extra_dir> echo "dir" > dir.txt
```

Фіксуємо додавання директорії з файлом:

```
C:\Users\User\Documents\GitHub\labrob1>git add extra_dir

C:\Users\User\Documents\GitHub\labrob1>git commit -m "Add extra directory with file"
[master 479c906] Add extra directory with file
 1 file changed, 1 insertion(+)
 create mode 100644 extra_dir/dir.txt
```

Створюємо нову гілку та переходимо на неї:

```
C:\Users\User\Documents\GitHub\labrob1>git branch feature_branch

C:\Users\User\Documents\GitHub\labrob1>git checkout feature_branch
Switched to branch 'feature_branch'
```

Тепер у новоствореній гілці видаляємо додану директорію:

```
C:\Users\User\Documents\GitHub\labrob1>rmdir /s /q extra_dir

C:\Users\User\Documents\GitHub\labrob1>cd extra_dir
The system cannot find the path specified.
```

Фіксуємо зміни:

```
C:\Users\User\Documents\GitHub\labrob1>git add .

C:\Users\User\Documents\GitHub\labrob1>git commit -m "Removed extra_dir"
[feature_branch 65e7024] Removed extra_dir
 1 file changed, 1 deletion(-)
 delete mode 100644 extra_dir/dir.txt
```

Робимо злиття змін з основною гілкою:

```
C:\Users\User\Documents\GitHub\labrob1>git checkout master
Switched to branch 'master'

C:\Users\User\Documents\GitHub\labrob1>git merge feature_branch
Updating 479c906..65e7024
Fast-forward
 extra_dir/dir.txt | 1 -
 1 file changed, 1 deletion(-)
 delete mode 100644 extra_dir/dir.txt
```

Виводимо історію змін:

```
C:\Users\User\Documents\GitHub\labrob1>git log
commit 65e7024ef0f8b1c2d82f6c4945027bfbfc50de35 (HEAD -> master, feature_branch)
Author: rshvtss <r.shvetsss@gmail.com>
Date:   Fri Sep 12 21:16:12 2025 +0300

    Removed extra_dir

commit 479c9069893f08a09d11b9fabb7b2296460f061e
Author: rshvtss <r.shvetsss@gmail.com>
Date:   Fri Sep 12 21:00:04 2025 +0300

    Add extra directory with file

commit 7945a7776256ee0be0bd735932d75e4d64624a00
Author: rshvtss <r.shvetsss@gmail.com>
Date:   Fri Sep 12 20:19:07 2025 +0300

    Text added
```

Висновок: в ході виконання лабораторної роботи я навчився працювати із системою контролю версій git, створював репозиторії, створював та зливав між собою гілки.