



Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря
Сікорського” Факультет інформатики та обчислювальної
техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №1

із дисципліни *«Технології розроблення програмного забезпечення»*

Тема: «Використання основних команд Git»

Виконав: Студент
групи ІА-23 Хохол
М.В.

Перевірив:
Мягкий М.Ю.

Мета: У межах даної лабораторної роботи передбачено освоїти використання локальної системи контролю версій Git.

Відповіді на питання:

1. Що таке система контролю версій (СКВ)

Система контролю версій (СКВ) — це програмне забезпечення, яке дозволяє відстежувати зміни у файлах та координувати роботу над проектом між кількома розробниками. Основне завдання СКВ — зберігати історію змін, підтримувати різні версії файлів і забезпечувати можливість повернутися до попередніх версій, якщо це необхідно.

2. Відмінності між розподіленою та централізованою СКВ

Централізована СКВ підходить для невеликих команд з простим робочим процесом, де є необхідність контролювати доступ до змін через єдиний сервер. Розподілена СКВ забезпечує більшу гнучкість, надійність та незалежність, особливо для великих проєктів з багатьма учасниками, де важлива автономна робота та можливість працювати офлайн.

3. Різниця між stage та commit в Git

Stage (індекс): це підготовка змін для коміту. Файли додаються в індекс за допомогою команди `git add`, але ще не збережені в історії.

Commit: це фактичне збереження змін в історію репозиторію. Після виконання `git commit` зміни, що були додані в індекс, фіксуються у вигляді нового коміту з коментарем.

4. Як створити гілку в Git

Для створення гілок є наступні команди **`git branch <name>`** — створює нову гілку, не перемикаючи на неї; **`git checkout -b <name>`** — створює нову гілку і перемикає на неї (старіший спосіб); **`git switch -c <name>`** — створює нову гілку і перемикає на неї (сучасний спосіб).

5. Як створити або скопіювати репозиторій Git з віддаленого серверу

create a new repository on the command line

```
echo "# trpz" >> README.md
```

```
git init
```

```
git add README.md
```

```
git commit -m "first commit"
```

```
git branch -M main
```

```
git remote add origin https://github.com/Maxim-Khokhol/trpz.git
```

```
git push -u origin main
```

push an existing repository from the command line

```
git remote add origin https://github.com/.../...
```

```
git branch -M main
```

```
git push -u origin main
```

6. Що таке конфлікт злиття, як створити конфлікт, як вирішити конфлікт?

Конфлікт злиття виникає, коли Git не може автоматично об'єднати зміни з двох різних гілок, оскільки ці зміни конфліктують. Найчастіше це трапляється, коли два або більше розробників редагують один і той самий рядок в одному файлі, або коли один видаляє файл, який інший модифікував.

Git сигналізує про конфлікт, коли під час виконання команди `git merge` або `git rebase` не може об'єднати зміни без ручного втручання.

Вирішити конфлікт: Вручну видалити конфліктні позначки (`<<<<<<<`, `=====`, `>>>>>>>`) і обрати, які зміни залишити. Можна залишити одне із двох, об'єднати обидва варіанти або внести власні корективи.

Зберегти файл і додати його до індексу: Після вирішення конфлікту потрібно зберегти файл і додайте його в індекс

Завершити злиття: Завершити процес `—continue`

7. Ситуації використання команд: merge, rebase, cherry-pick

`git merge`: використовується для об'єднання двох гілок із збереженням повної історії.

`git rebase`: використовується для чистої, лінійної історії шляхом переписування комітів поверх іншої гілки.

`git cherry-pick`: використовується для вибіркового копіювання окремих комітів з однієї гілки в іншу.

8. Як переглянути історію змін Git репозиторію в консолі?

`git log`

9. Як створити гілку в Git не використовуючи команду `git branch`

`git checkout -b <name>` — створює нову гілку і перемикає на неї (старіший спосіб); `git switch -c <name>` — створює нову гілку і перемикає на неї (сучасний спосіб).

10. Як підготувати всі зміни в поточній папці до коміту?

переглянути статус, занести в індекс всі файли які повинні бути в комміті

11. Як підготувати всі зміни в дочірній папці до коміту?

Перевірити статус репозиторію (`git status`).

Додати зміни в дочірній папці (`git add my-subfolder/`).

12. Як переглянути перелік наявних гілок в репозиторії?

Використати `git branch` для перегляду локальних гілок.

Використати `git branch -a` для перегляду всіх гілок (локальних та віддалених).

Використати `git branch -r` для перегляду тільки віддалених гілок.

13. Як видалити гілку?

Для локальної гілки використовуйте `git branch -d <назва_гілки>` для видалення з перевіркою злиття або `git branch -D <назва_гілки>` для примусового видалення.

Для віддаленої гілки використовуйте `git push <remote> --delete <назва_гілки>`.

Хід роботи:

Крок 1: Для початку роботи потрібно проініціалізувати новий репозиторій Git.

```
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git init
Initialized empty Git repository in C:/Users/hmax0/OneDrive/Рабочий стол/trpz-1/.git/
```

Крок 2: Створити перший пустий комміт

```
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git commit --allow-empty -m "first commit"
[main (root-commit) d66f788] first commit
```

Крок 3: Створити 2 гілки двома різними способами

```
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git branch br1  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git checkout -b br2  
Switched to a new branch 'br2'
```

Крок 4: В поточній папці створити та окремо закомітити 3 файли. Остінній файл - .gitignore

```
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>echo "some text" >> text.txt  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git add .  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git commit -m "some text"  
[br2 e06bb72] some text  
1 file changed, 1 insertion(+)  
create mode 100644 text.txt  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>echo "some text2" >> text2.txt  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git add .  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git commit -m "text2"  
[br2 75d7772] text2  
1 file changed, 1 insertion(+)  
create mode 100644 text2.txt  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>echo ".log" >> .gitignore  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git add .  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git commit -m "gitignore"  
[br2 2d6d098] gitignore  
1 file changed, 1 insertion(+)  
create mode 100644 .gitignore
```

Крок 4: Видалити файл text.txt з видаленням комітів з історії (тут я поспішив та видалив через rm що призвело до появи ще одного коміту стосовно видалення файлу)

```
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git rm text.txt  
rm 'text.txt'  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git add .  
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git commit -m "delete text"  
[br2 0aabc9e] delete text  
1 file changed, 1 deletion(-)  
delete mode 100644 text.txt
```

Крок5: Спроба видалити файл з історії комітів через git rebase -i

```
C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git rebase -i br2
hint: Waiting for your editor to close the file... unix2dos: converting file C:/Users/hmax0/OneDrive/Рабочий стол/trpz-1/.git/rebase-merge/git-rebase-todo to DOS format...
dos2unix: converting file C:/Users/hmax0/OneDrive/Рабочий стол/trpz-1/.git/rebase-merge/git-rebase-todo to Unix format...
Successfully rebased and updated refs/heads/br2.

C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git rebase -i br2 "@~4"
hint: Waiting for your editor to close the file... unix2dos: converting file C:/Users/hmax0/OneDrive/Рабочий стол/trpz-1/.git/rebase-merge/git-rebase-todo to DOS format...
dos2unix: converting file C:/Users/hmax0/OneDrive/Рабочий стол/trpz-1/.git/rebase-merge/git-rebase-todo to Unix format...
Successfully rebased and updated detached HEAD.
```

Обидві спроби відкрили файл редагування проте не містили коммітів.

Під час захисту в двох командах була дописана назву гілки →

```
>git rebase -i br2 "@~4"
```

Файл редагування відкрився, проте коммітів там не було


Крок6: Після захисту я прописав

```
git rebase -i HEAD~3
```

вже без назви гілки і в файлі редагування з'явилися комміти

Я просто видалив стрічку з коммітом “some text” зберіг файл та закрив його.

Він повністю пропав з історії

<pre>C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git log --oneline 2d6d098 (HEAD) gitignore 75d7772 text2 e06bb72 some text d66f788 (main, br1) first commit</pre>		<pre>C:\Users\hmax0\OneDrive\Рабочий стол\trpz-1>git log --oneline bb02d5c (HEAD) gitignore a828234 text2 d66f788 (main, br1) first commit</pre>
---	---	---

Висновок: у даній лабораторній роботі я спробував на практиці деякі основні команди для локального контролю версій Git