

Національний Технічний університет України

“Київський політехнічний інститут”

імені Ігоря Сікорського

Лабораторна робота №1

з дисципліни "ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ"

на тему "Системи контролю версій SVN, GIT"

Виконав:

студент групи Да-62

Домілевський О.С

Зміст

[Мета роботи: 2](#_Toc20264659)

[Задача: 2](#_Toc20264660)

[Завдання: 2](#_Toc20264661)

[Короткі теоретичні відомості: 2](#_Toc20264662)

[Хід роботи: 3](#_Toc20264663)

[Висновки: 4](#_Toc20264664)

## **Мета роботи:**

за допомогою системи контролю версій завантажити коди програми у репозіторій. Відтворити типовий цикл розробки програмного забезпечення з використанням системи контролю версій.

## **Задача:**

1. Вивчити основні команди роботи з репозіторіями.
2. Завантажити код програми у репозіторій.
3. Показати основний цикл роботи з програмним кодом за допомогою системи контролю версій.

## **Завдання:**

1. Обрати безкоштовну систему репозиторія для системи контролю версіями, наприклад projectlocker, або інш
2. Встановити клієнтське безкоштовне програмне забезпечення для роботи с системою контролю версій (GIT, SVN clients)
3. Протягом роботи над лабораторними роботами 2-6 використовувати систему контролю версіями
4. Описати цикл розробки програмного забезпечення з використанням системи контрою версій

## **Короткі теоретичні відомості:**

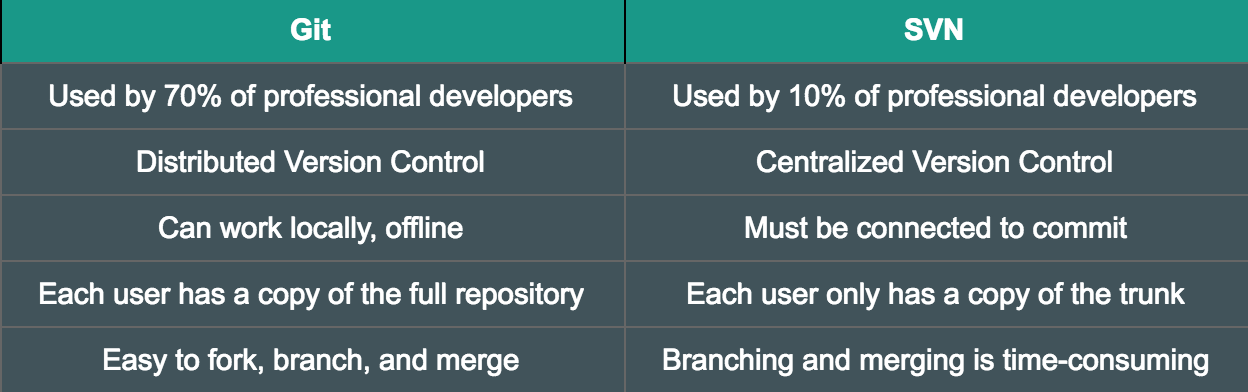
Система керування версіями (від англ Version Control System, VCS або Revision Control System.) – це спеціальне програмне забезпечення для полегшення роботи з інформацією, яка часто змінюється. Система керування версіями дозволяє зберігати декілька версій одного і того ж документа, при необхідності повертатися до більш ранніх версій, визначати, хто і коли зробив ту чи іншу зміну, керувати гілками різноманітних версій програми.

Git — [розподілена система керування версіями](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D0%BF%D0%BE%D0%B4%D1%96%D0%BB%D0%B5%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8) [файлів](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB) та спільної роботи. Проект створив [Лінус Торвальдс](https://uk.wikipedia.org/wiki/%D0%9B%D1%96%D0%BD%D1%83%D1%81_%D0%A2%D0%BE%D1%80%D0%B2%D0%B0%D0%BB%D1%8C%D0%B4%D1%81" \o "Лінус Торвальдс) для управління розробкою [ядра Linux](https://uk.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_Linux), а сьогодні підтримується [Джуніо Хамано](https://uk.wikipedia.org/wiki/%D0%94%D0%B6%D1%83%D0%BD%D1%96%D0%BE_%D0%A5%D0%B0%D0%BC%D0%B0%D0%BD%D0%BE" \o "Джуніо Хамано). Git є однією з найефективніших, надійних і високопродуктивних [систем керування версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8), що надає гнучкі засоби нелінійної розробки, що базуються на відгалуженні і злитті [гілок](https://uk.wikipedia.org/wiki/%D0%93%D1%96%D0%BB%D0%BA%D0%B0_(%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8)). Для забезпечення цілісності історії та стійкості до змін заднім числом використовуються криптографічні методи, також можлива прив'язка [цифрових підписів](https://uk.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D0%BF%D1%96%D0%B4%D0%BF%D0%B8%D1%81) розробників до тегів і комітів.

GitHub — один з найбільших [веб-сервісів](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) для спільної розробки [програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F). Існують безкоштовні та платні тарифні плани користування сайтом. Базується на [системі керування версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8) [Git](https://uk.wikipedia.org/wiki/Git" \o "Git) і розроблений на [Ruby on Rails](https://uk.wikipedia.org/wiki/Ruby_on_Rails" \o "Ruby on Rails) і [Erlang](https://uk.wikipedia.org/wiki/Erlang" \o "Erlang) компанією GitHub, Inc (раніше Logical Awesome).

Сервіс безкоштовний для проектів з відкритим вихідним кодом, з наданням користувачам усіх своїх можливостей (включаючи SSL), а для окремих індивідуальних проектів пропонуються різні платні тарифні плани.

Subversion (з 2010 називається Apache Subversion) — вільна [система управління версіями](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8). Subversion розроблена спеціально для заміни [CVS](https://uk.wikipedia.org/wiki/CVS), найпоширенішої відкритої системи управління версіями. Вона має всі основні функції CVS (хоча деякі з них виконує іншими способами) і вільна від ряду її недоліків. Subversion часто називають «svn», по назві клієнтської програми, що входить в її дистрибутив.



**Git** — розподілена система керування версіями файлів, а **Subversion** — централізована система. Git не має потрібності у мережевому з’єднанні, з її гілками легше працювати, але не можливо контролювати доступ до файлів та відсутня можливість їх блокування. SVN має загальну нумерацію версій та можливість керування доступом до файлів, проте оновлення та збереження змін можливе лише при наявному мережевому з’єднанні з основним сервером.

## **Хід роботи:**

Основні команди GIT:

git clone адрес репозитория - клонировать репозиторий на локальный

компютер

git commit -m 'initial commit' - комит изменений в локальный репозиторий

git push -u origin master - только первый раз отправка изменений в удаленный репозиторий

git push - все последующие разы отправка изменений в удаленный репозиторий

git pull - скачивается актуальная версия удаленного репозитория и все изменения применяются к локальному репозиторию

checkout - перейти в другую ветку

discard - не отправлять в репозиторий те изменения которые нам не нравятся

revert - откатить существующие комиты

hard reset -откатить целую пачку комитов

merge - сначала подтягиваем в свою ветку из master все изменения

merge - потом тестируем и делаем слияние переходим (checkout) в master и применяем (merge) все изменения из нашей ветки

git commit -a -m 'commit all edited files'- закомитить все измененные файлы без использования команды git add

git clone - скачивание репозитория

git status - просмотр текущих изменений

git add - добавить файл в локальный репозиторий

git commit - коммит в локальный репозиторий

git push - коммит в удаленный репозиторий

git branch -a - посмотреть в какой ветке мы находимся и показать все ветки ключ -а all

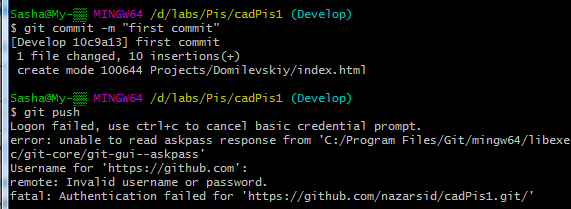
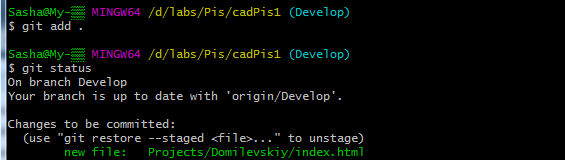
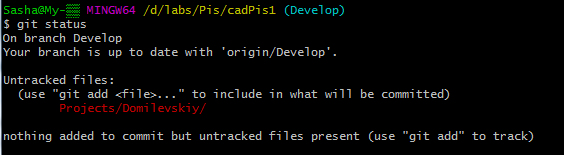
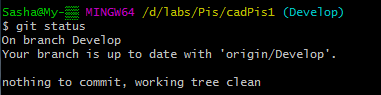
git branch newbranch - создать новую ветку с именем newbranch

git checkout newbranch - перейти в новую ветку

git push origin newbranch - (сначала переключиться на мастер) отправить в удаленный репозиторий ветку newbranch

git branch -d newbranch - удалить ветку newbranch из локального репозитория

git push origin --delete newbranch - удалить ветку newbranch в удаленном репозитории



## **Висновки:**

Під час виконання лабораторної роботи, було проведено порівняння систем контролю версій Git та SVN. Отримано навички роботи з системою контролю версій Git з засвоєнням основних команд.