

# Git

**Трішки води, історії і для чого це все  
потрібно :)**



# Локальні системи контролю версій

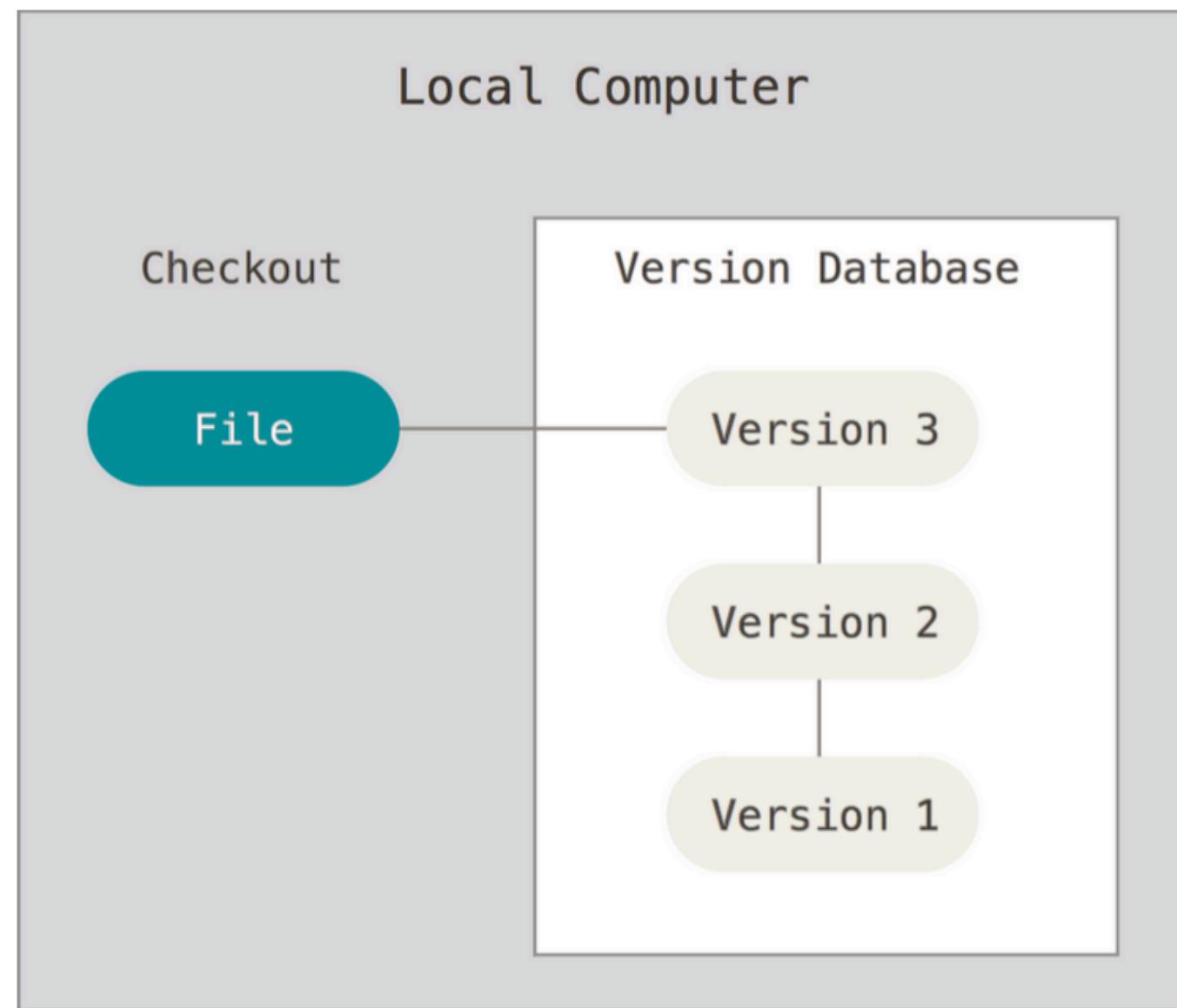


Схема локальної СКВ

# Централізовані системи контролю версій

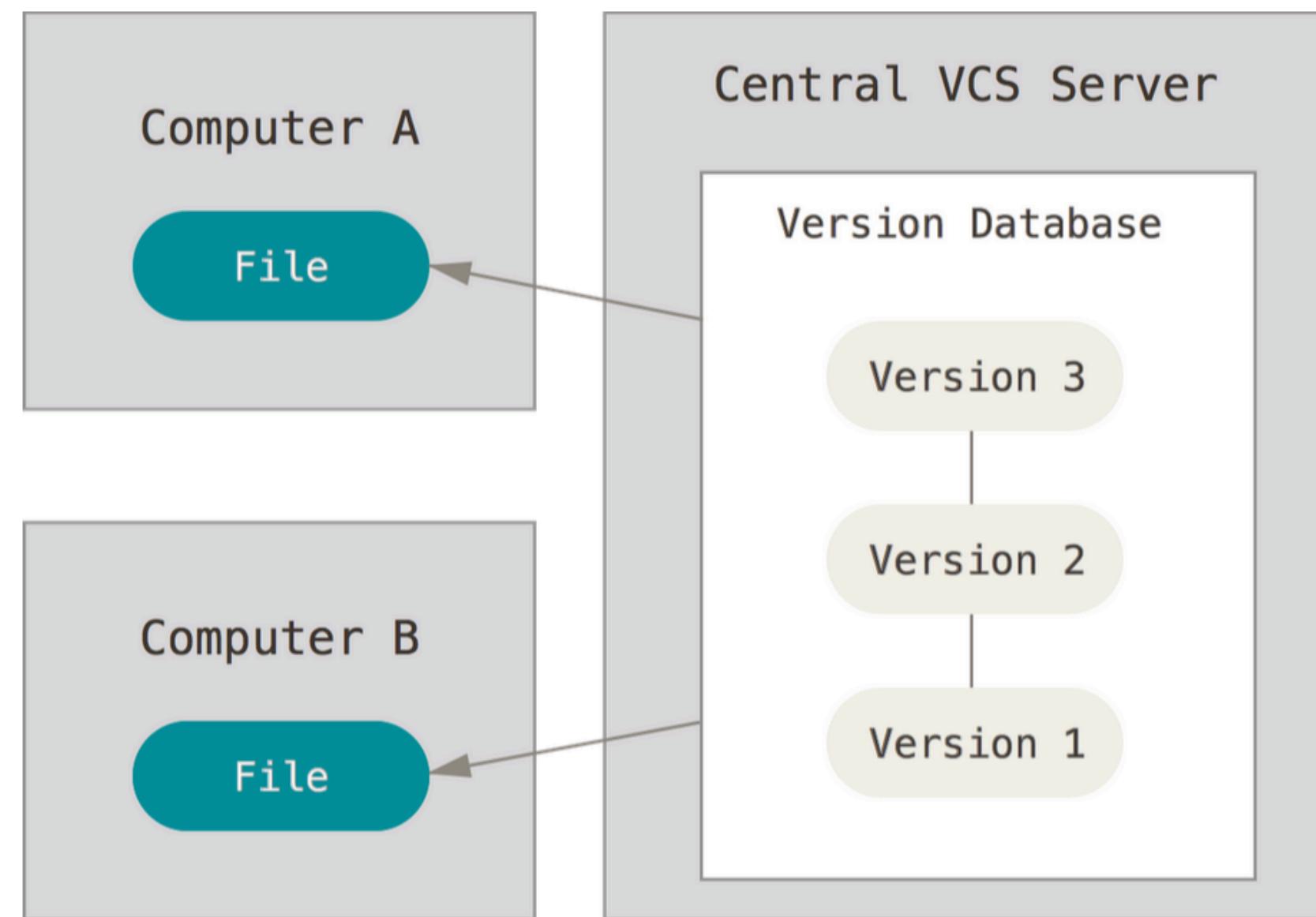


Схема централізованої СКВ

# Git - розподілена система контролю версій

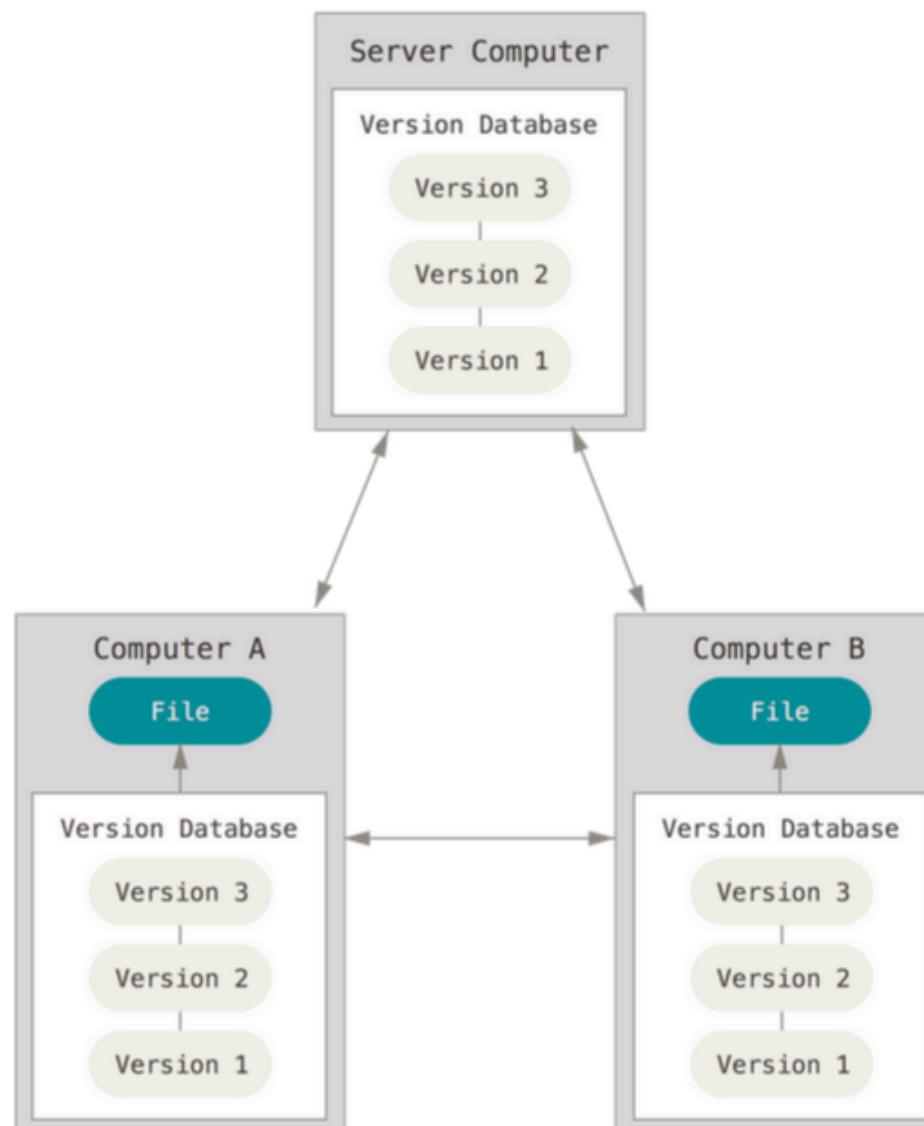
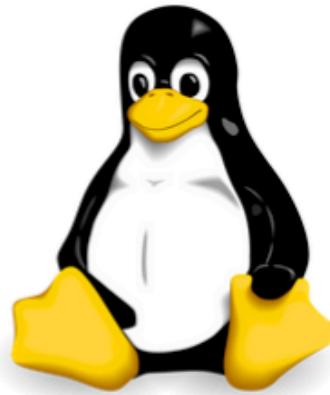


Схема розподіленої СКВ

# Звідки GIT

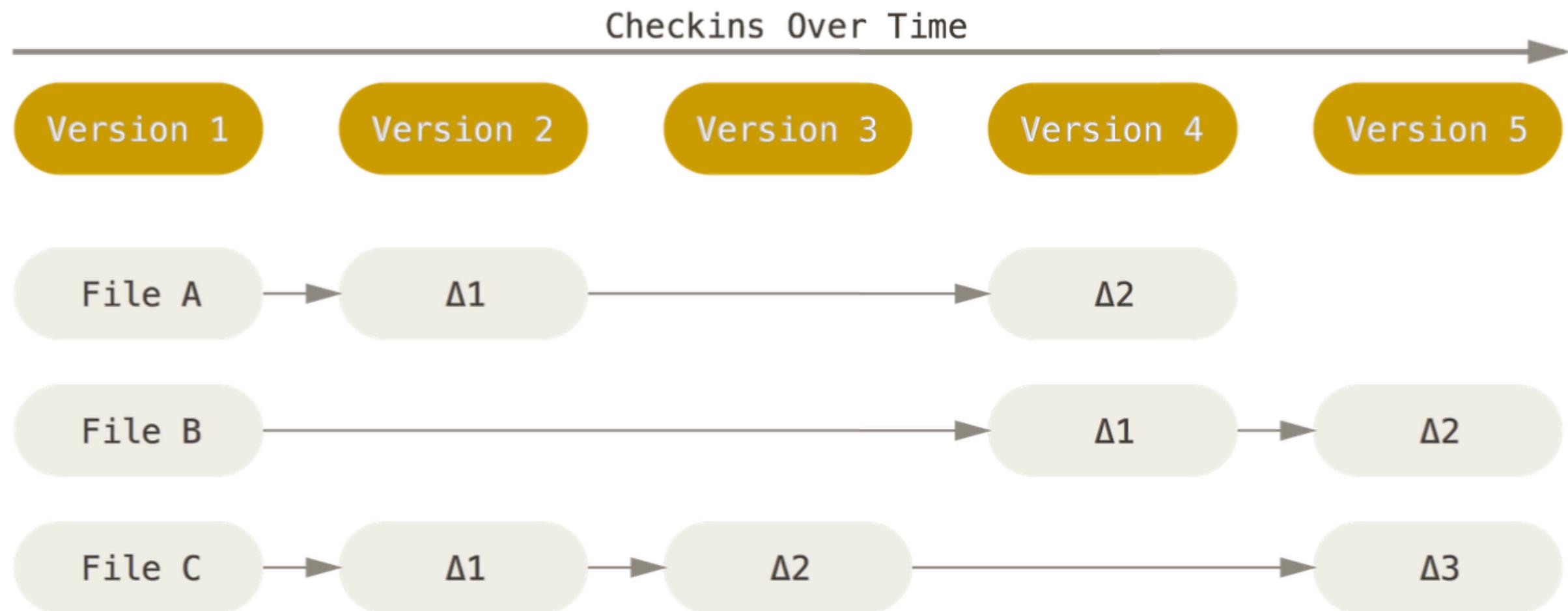
- Проект ядра linux
- Повинен бути надійним, швидким і більш природнім
- Здатен працювати з великими проектами



# Базові принципи

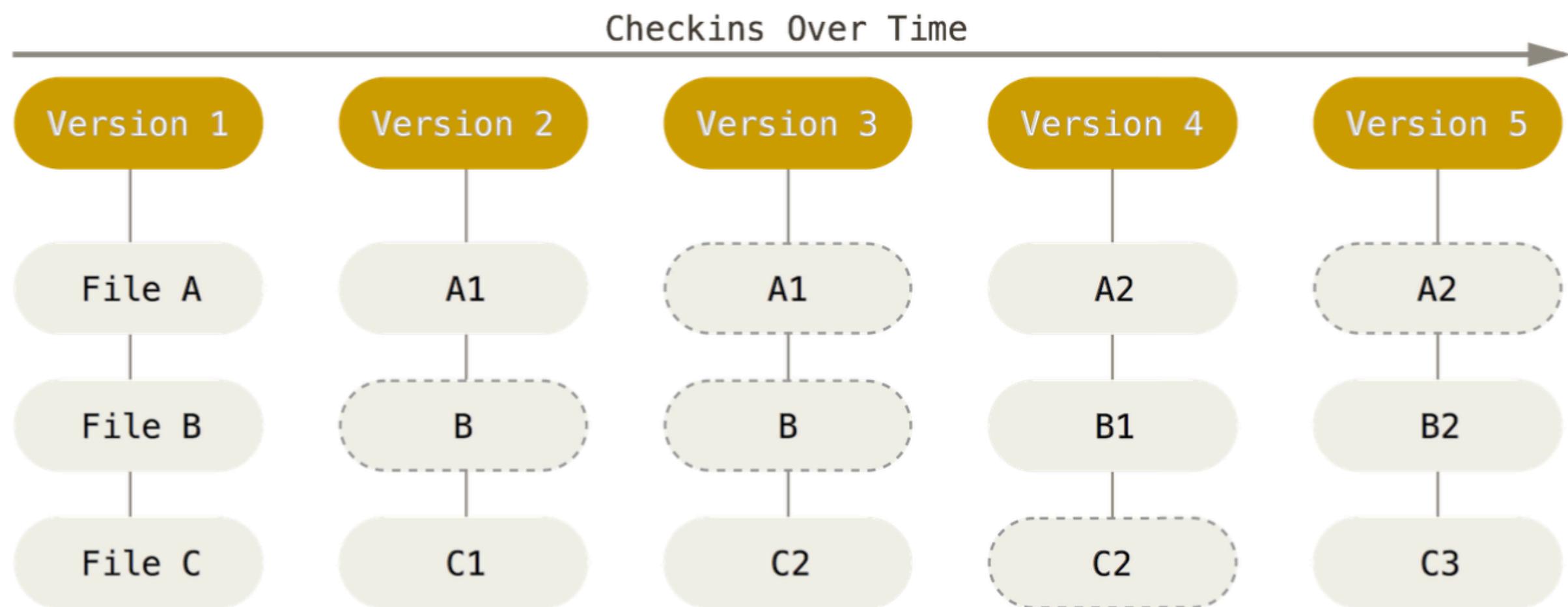
1. Зліпки замість різниці файлів(дельти)
2. Три основні стани файлу
3. Повна копія проекту у кожного користувача

# Дельта станів



Інші системи зберігають данні як зміни до базової версії файла

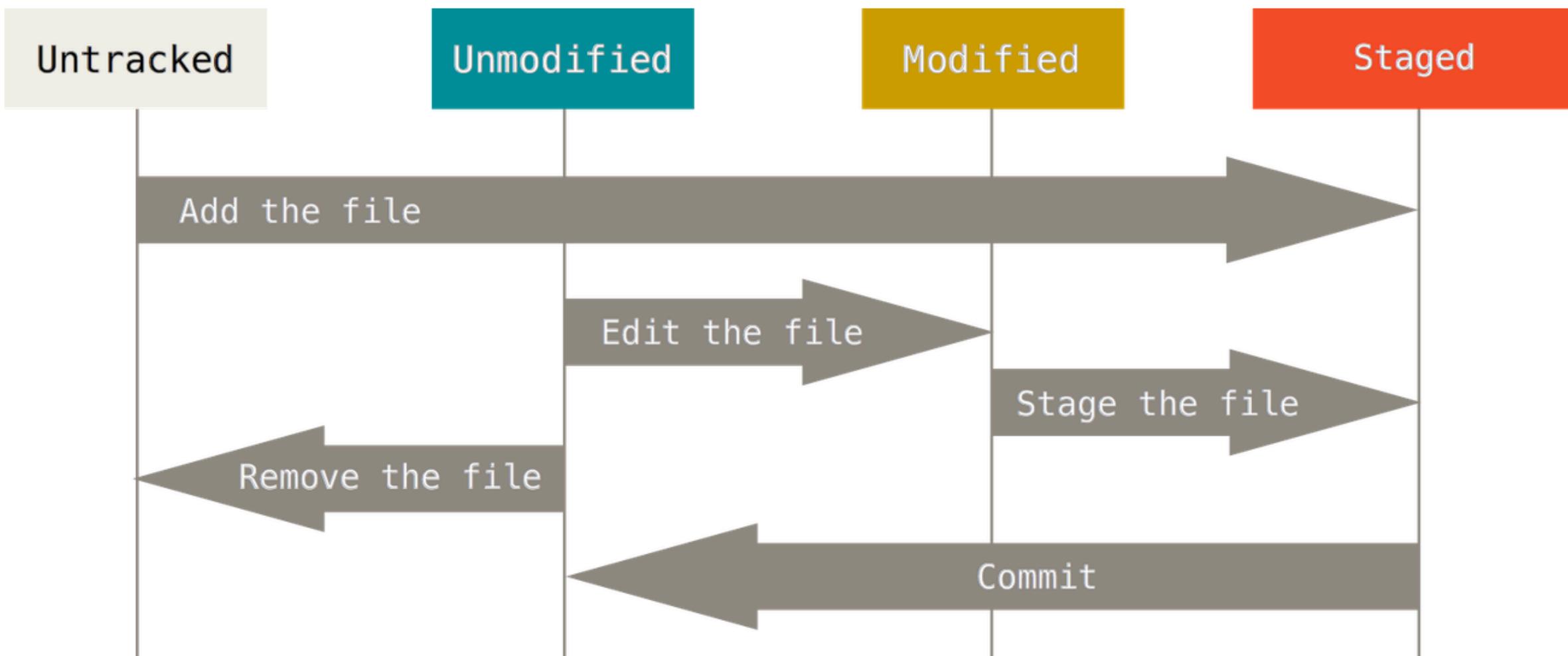
# Зліпки



Git зберігає данні як зліпки станів проекта в часі

# Стани файлу

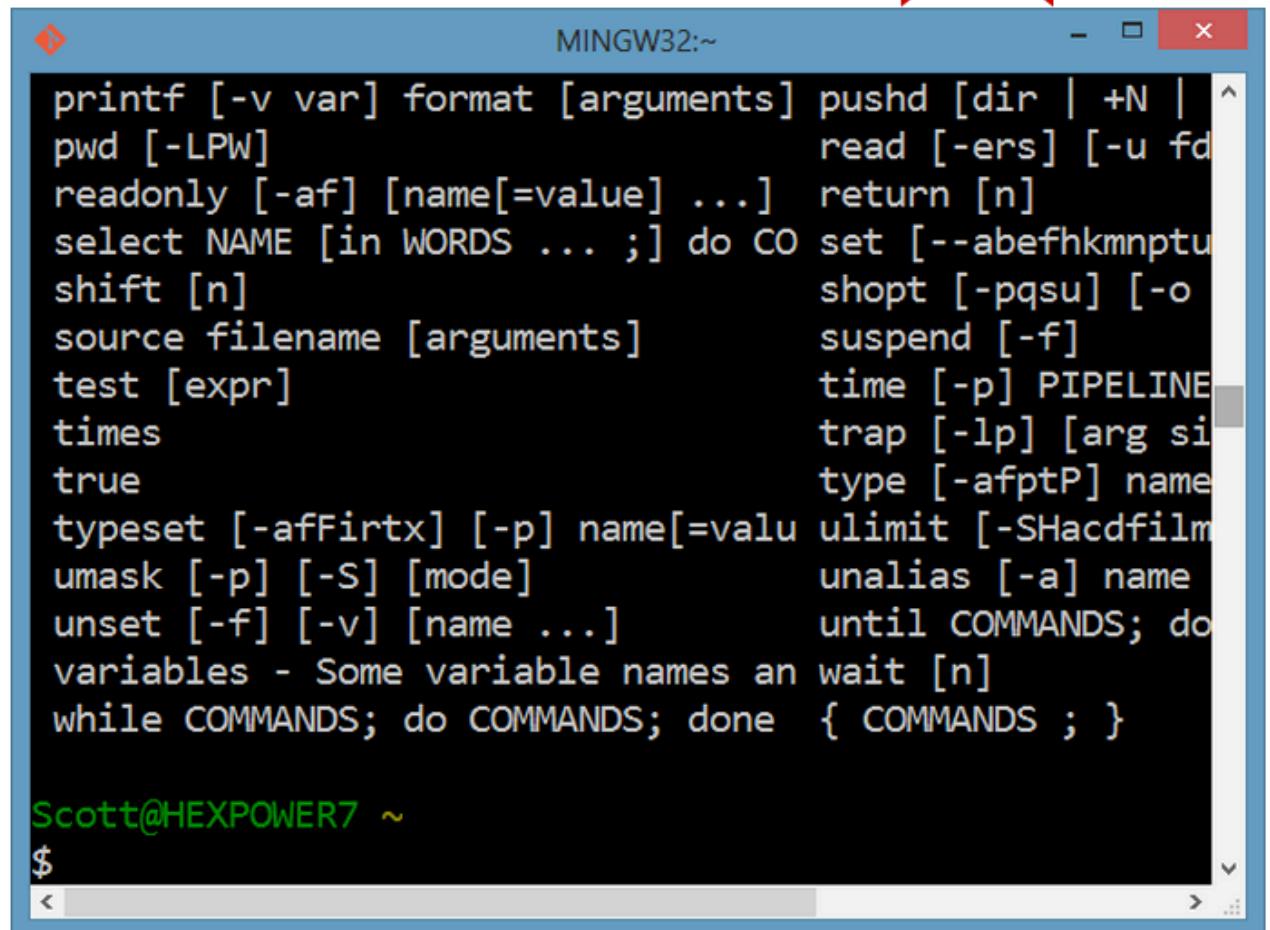
**При роботі з файлами це потрібно пам'ятати**



**Життєвий цикл станів файлу**

# Робота з GIT

- Command line interface
- Багато GUI засобів доступно
- Офіційний сайт <http://git-scm.com/>



The screenshot shows a terminal window titled "MINGW32:~". The window contains a list of shell commands, likely from the zsh shell, displayed in white text on a black background. The commands listed include: printf [-v var] format [arguments] pushd [dir | +N | ^...], pwd [-LPW], readonly [-af] [name[=value] ...], select NAME [in WORDS ... ;] do CO, shift [n], source filename [arguments], test [expr], times, true, typeset [-afFirtx] [-p] name[=valu], umask [-p] [-S] [mode], unset [-f] [-v] [name ...], variables - Some variable names an wait [n], while COMMANDS; do COMMANDS; done { COMMANDS ; }, read [-ers] [-u fd], return [n], set [--abefhkmnptu], shopt [-pqsu] [-o suspend [-f], time [-p] PIPELINE, trap [-lp] [arg si, type [-afptP] name, ulimit [-SHacdfilm, unalias [-a] name, until COMMANDS; do, done { COMMANDS ; }.

Scott@HEXPOWER7 ~

\$

# GIT HELP

01. \$ git help

02. \$ git help config

# Базове налаштування GIT

01. \$ git config --global user.name "Valera"
02. \$ git config --global user.email "valera.ogon@gmail.com"
03. \$ git config --global color.ui true

# Ініціалізуємо репозиторій

01. \$ mkdir readme

02. \$ cd readme

03. \$ git init

```
Covert@COVERTNOTE /C/Users/Covert/readme
$ git init
Initialized empty Git repository in c:/Users/Covert/readme/.git/
```

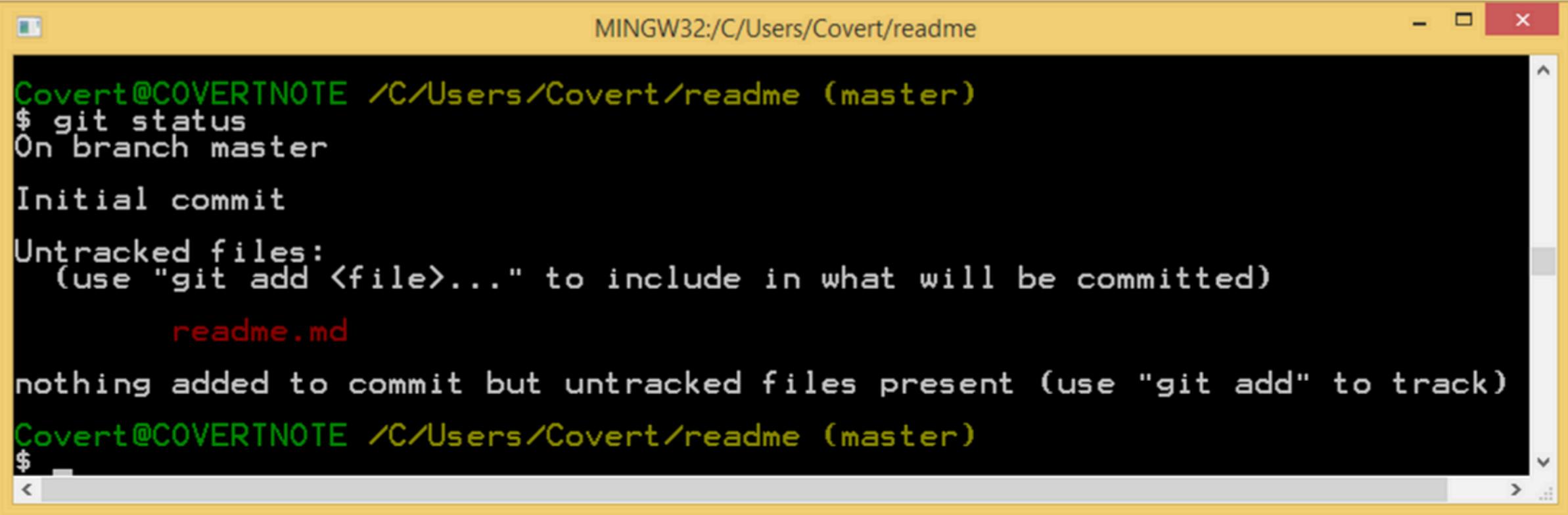
ініціалізуємо GIT

# Послідовність роботи з GIT

- Валера створив файл readme.txt
- Додав до зони підготовки до коміту "staging area"
- "Commit" (фіксуємо зміни)
  
- Валера змінив файл readme.txt та додав LICENSE.txt
- Додав обидва файла до зони підготовки до коміту "staging area"
- "Commit" (фіксуємо зміни)

# Перевіряємо стан файлів

\$ git status



The screenshot shows a terminal window titled "MINGW32:/C/Users/Covert/readme". The command \$ git status is run, resulting in the following output:

```
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git status
On branch master
Initial commit
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)
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

результат в консолі

# Додаємо файл до зони підготовки до коміту

```
$ git add README.txt
```



The screenshot shows a terminal window titled "MINGW32:/C/Users/Covert/readme". The command \$ git status is run, resulting in the following output:

```
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git status
On branch master
Initial commit
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   README.md

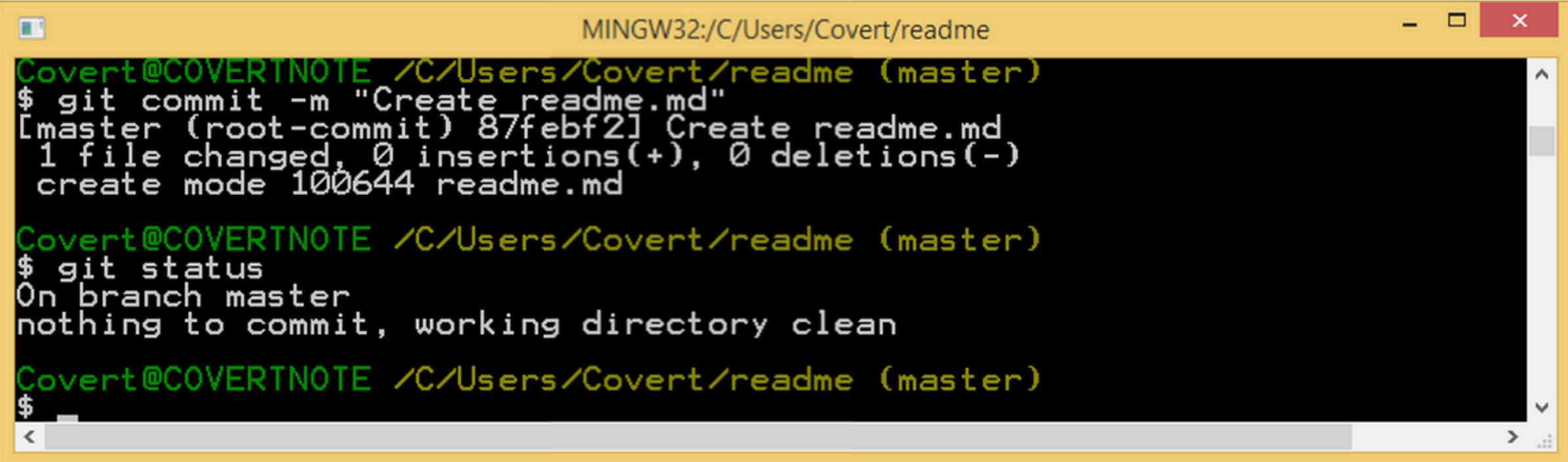
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

The terminal window has a yellow border and a red title bar.

перевірили статус

# Робимо зліпок (commit)

```
$ git commit -m 'create README'
```



The screenshot shows a terminal window with a yellow title bar and a black body. The title bar reads "MINGW32:/C/Users/Covert/readme". The terminal output is as follows:

```
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git commit -m "Create README.md"
[master (root-commit) 87febf2] Create README.md
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 README.md

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git status
On branch master
nothing to commit, working directory clean

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

результат в консолі

## Валера змінив файл `readme.md` та додав `license.txt`

```
$ git add readme.md license.txt
```

або

```
$ git add --all
```

або

```
$ git add .
```

## Закомітмо зміни

```
$ git commit -m "Add license, update readme"
```

# Історія комітів

\$ git log



The screenshot shows a Windows Command Prompt window titled "MINGW32:/C/Users/Covert/readme". The command \$ git log is run, displaying two commits. The first commit adds a license and updates the README, while the second creates a README.md file. Both commits were made by Valera on Wednesday, April 8, 2015, at 23:02:59 +0300.

```
MINGW32:/C/Users/Covert/readme
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git log
commit 20f34d51e4cb60b3a84e5d5e8b6dfcec59173d28
Author: Valera <m.dekhtiarenko@gmail.com>
Date:   Wed Apr 8 23:02:59 2015 +0300

    Add license, update readme

commit 87febfb24c94d3243a1077cc69f56cc823718b238
Author: Valera <m.dekhtiarenko@gmail.com>
Date:   Wed Apr 8 22:06:03 2015 +0300

    Create README.md

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ _
```

результат в консолі

# GIT DIFF

01. \$ git diff

02. \$ git diff --staged



The screenshot shows a terminal window titled "MINGW32:/C/Users/Covert/readme". The command entered is "\$ git diff". The output is a diff report comparing two versions of the file "readme.md". The report shows a single change where a line has been deleted and a new line has been added. The deleted line is shown in red, and the new line is shown in green. The terminal window has a yellow border and a black background.

```
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git diff
diff --git a/readme.md b/readme.md
index e9248a7..8cad9b3 100644
--- a/readme.md
+++ b/readme.md
@@ -1 +1 @@
-k jf jk lwlwg ljer gjk ergler lgj ergl erj gler
\ No newline at end of file
+k jf jk lwlwg ljer gjk ergler lgj ergl erj gler weto i jweltw;et wef wf wefw e
\ No newline at end of file

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

результат в консолі

# А якщо ми консерватори?

```
$ git checkout -- readme.md
```



The screenshot shows a terminal window titled "MINGW32:/C/Users/Covert/readme". The command \$ git diff was run, comparing two versions of the file "readme.md". The output shows a diff between the file's content, highlighting changes made. The terminal window has a yellow border and a black background.

```
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git diff
diff --git a/readme.md b/readme.md
index e9248a7..8cad9b3 100644
--- a/readme.md
+++ b/readme.md
@@ -1 +1 @@
-kjfjklwklwgljergjkerglerlgjergjlerjgler
\ No newline at end of file
+kjfjklwklwgljergjkerglerlgjergjlerjglerwetoijweltw;et wef wf wefwe ew
\ No newline at end of file

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

результат в консолі

# Корисні команди

01. \$ git reset --soft HEAD^
02. \$ git commit --amend -m 'New message'
03. \$ git reset --hard HEAD^
04. \$ git reset --hard HEAD^^

# Як поділитися?

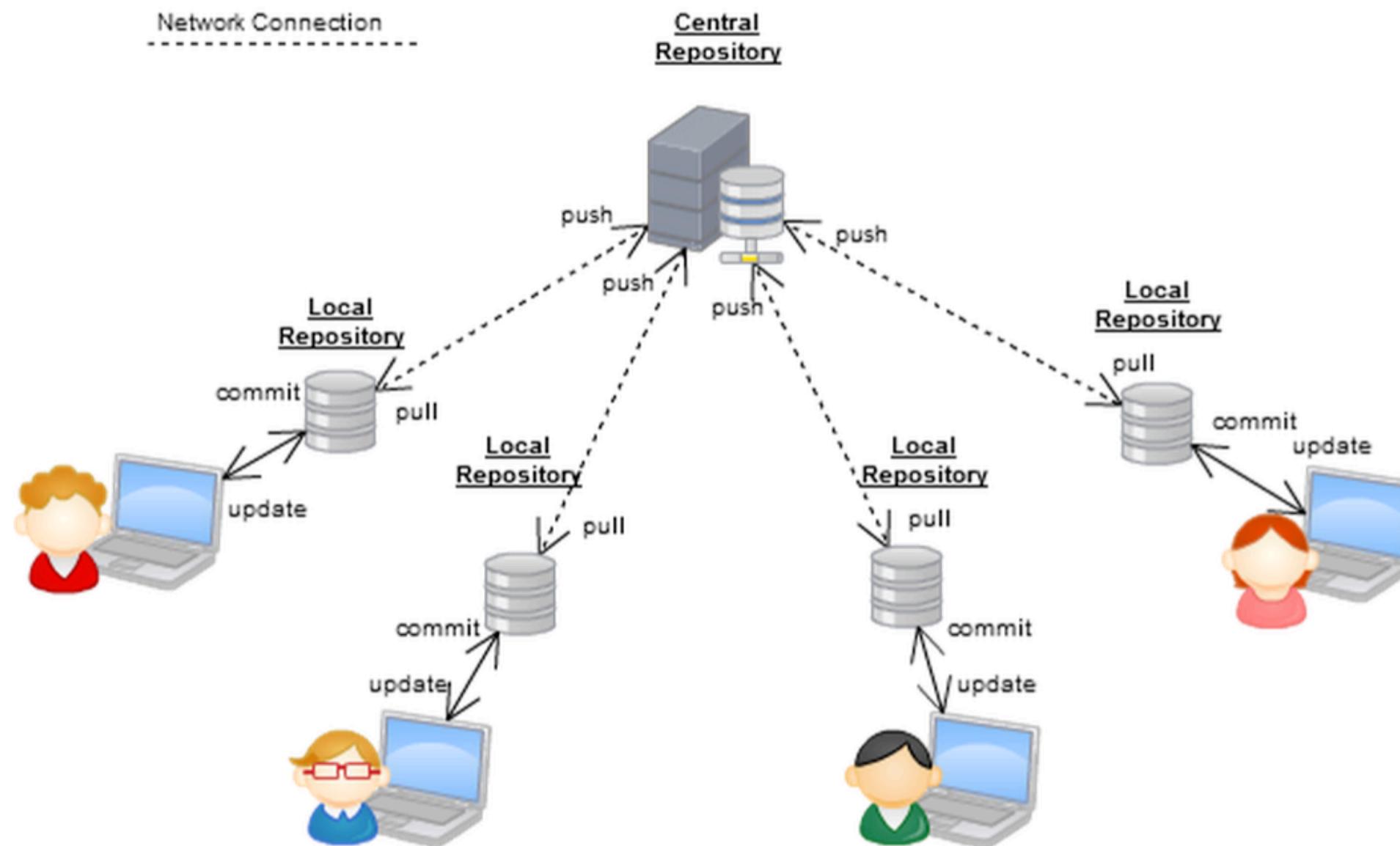


схема GIT

# GIT не піклується про контроль доступу

Потрібен додатковий софт



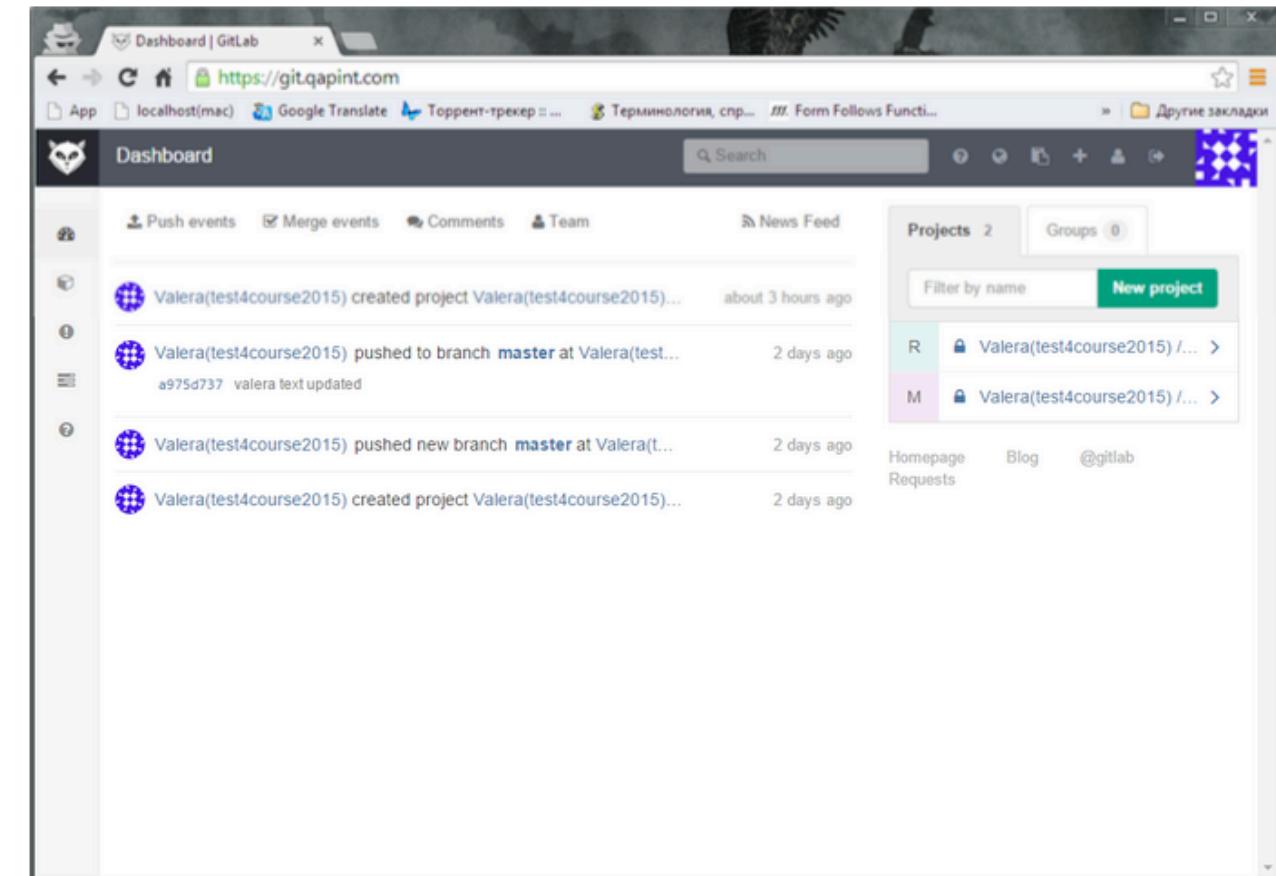
# Хостинг для віддалених git репозиторів

## Довіритись великим дядям

- GitHub
- BitBucket

## Бути самостійним

- GitLab
- Gitosis
- Gitorious

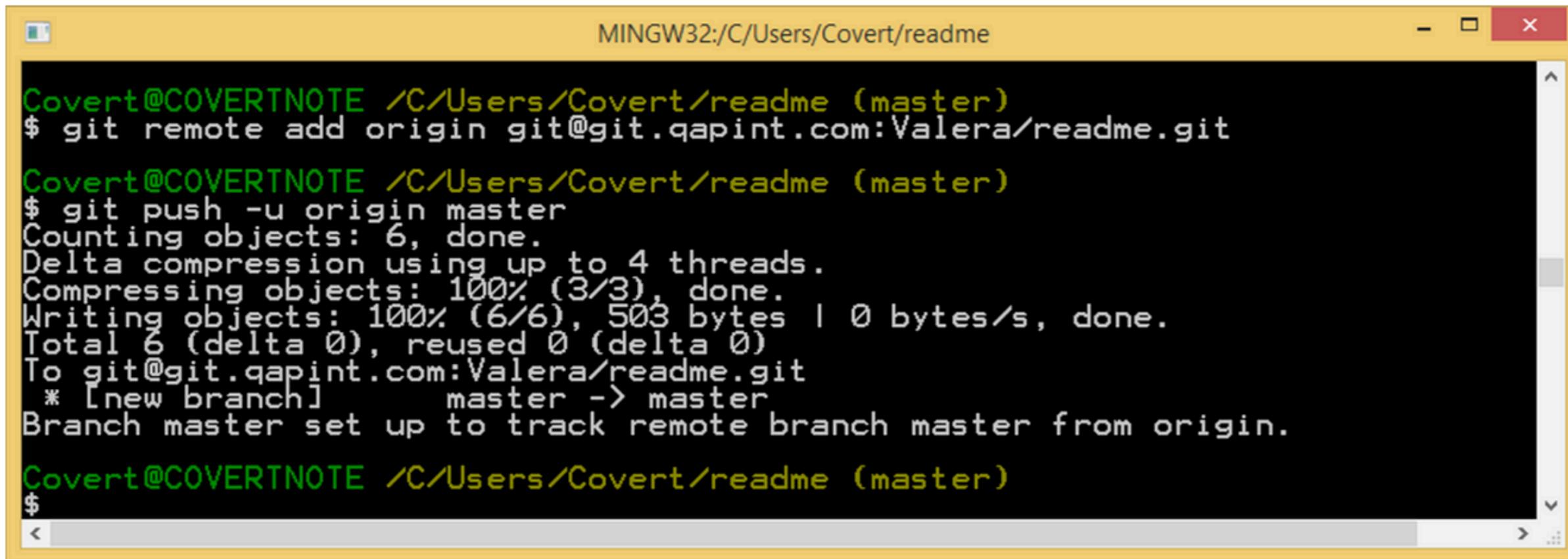


## З'єднуємо локальне та віддалене сховище

01. \$ git remote add origin git@git.qapint.com:Valera/readme.git
02. \$ git remote -v

# Заливаємо на віддалений репозиторій

```
$ git push -u origin master
```



The screenshot shows a terminal window titled "MINGW32:/C/Users/Covert/readme". The command \$ git push -u origin master is entered, followed by its execution output:

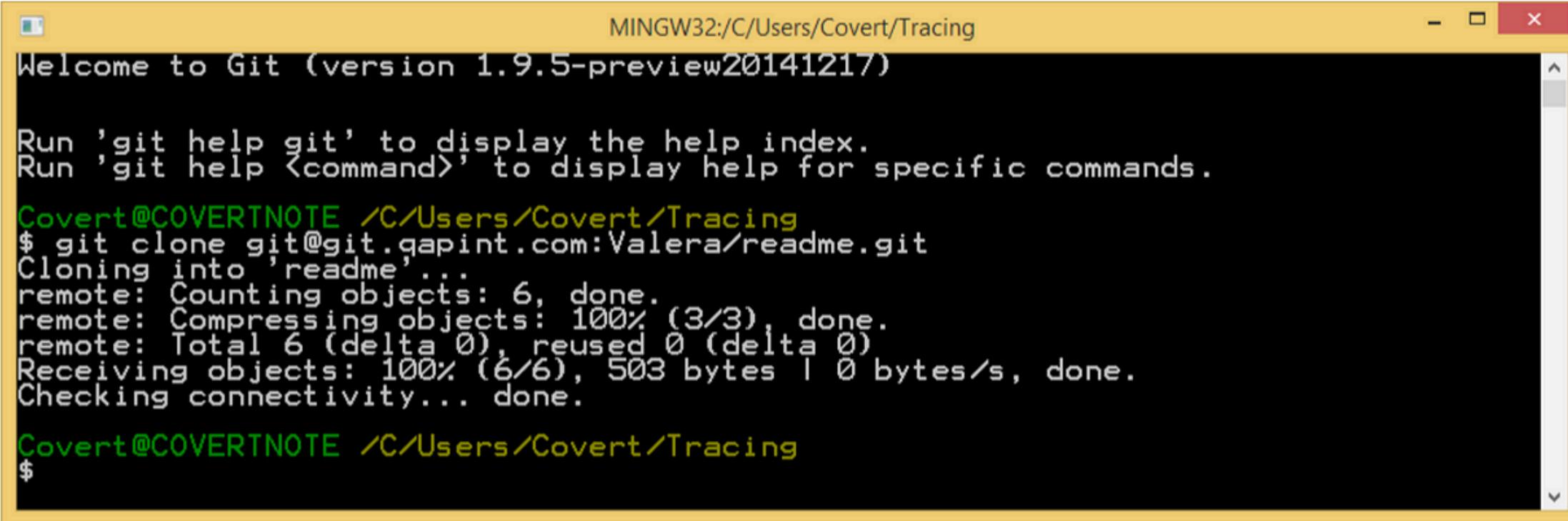
```
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git remote add origin git@git.qapint.com:Valera/readme.git
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git push -u origin master
Counting objects: 6, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 503 bytes | 0 bytes/s, done.
Total 6 (delta 0), reused 0 (delta 0)
To git@git.qapint.com:Valera/readme.git
 * [new branch] master -> master
Branch master set up to track remote branch master from origin.

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

інформація про пуш

# А якщо нам відкрили доступ до цікавого проекту

```
$ git clone git@git.qapint.com:Valera/readme.git
```



```
MINGW32:/C/Users/Covert/Tracing
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

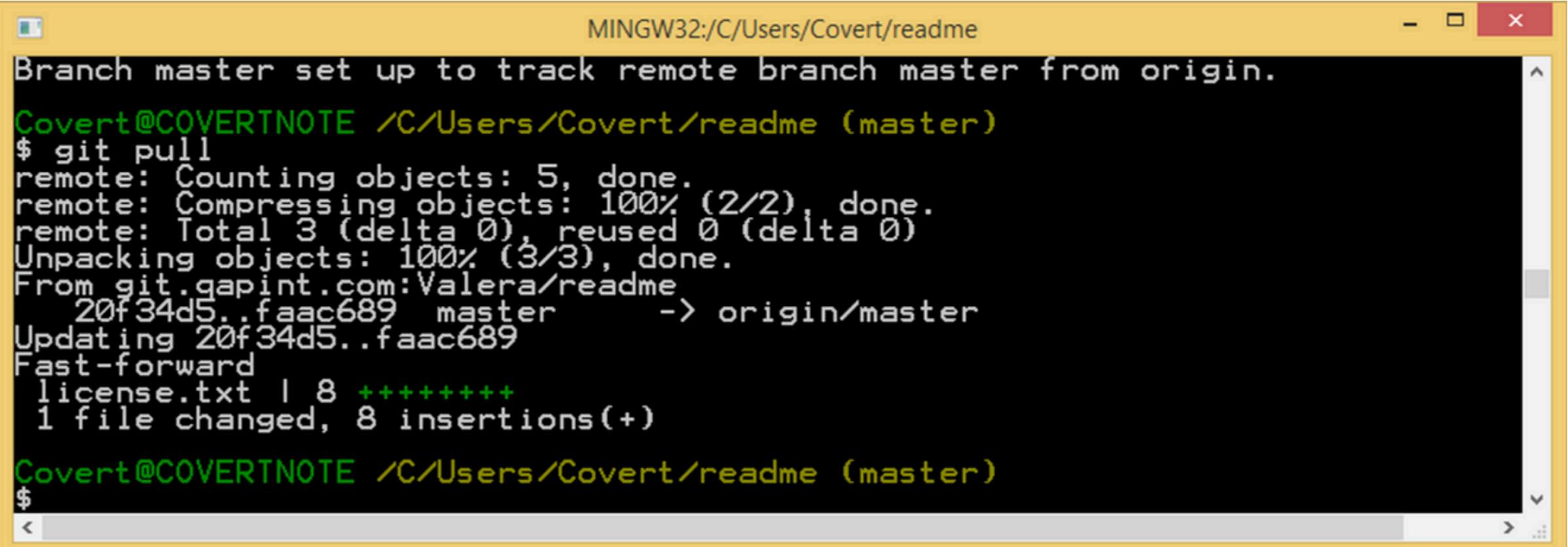
Covert@COVERTNOTE /C/Users/Covert/Tracing
$ git clone git@git.qapint.com:Valera/readme.git
Cloning into 'readme'...
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (6/6), 503 bytes | 0 bytes/s, done.
Checking connectivity... done.

Covert@COVERTNOTE /C/Users/Covert/Tracing
$
```

інформація про клонування проекту

# Оновлюємо проект

\$ git pull



The screenshot shows a terminal window titled "MINGW32:/C/Users/Covert/readme". The command \$ git pull is run, and the output shows the process of pulling from the origin/master branch. It includes details about object counting, compression, and unpacking, followed by a fast-forward update of the master branch, and a summary of file changes.

```
Branch master set up to track remote branch master from origin.
Covert@COVERTNOTE /C/Users/Covert/readme (master)
$ git pull
remote: Counting objects: 5, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
From git.qapint.com:Valera/readme
  20f34d5..faac689 master      -> origin/master
Updating 20f34d5..faac689
Fast-forward
  license.txt | 8 ++++++++
  1 file changed, 8 insertions(+)

Covert@COVERTNOTE /C/Users/Covert/readme (master)
$
```

інформація про оновлення проекту

# Практика

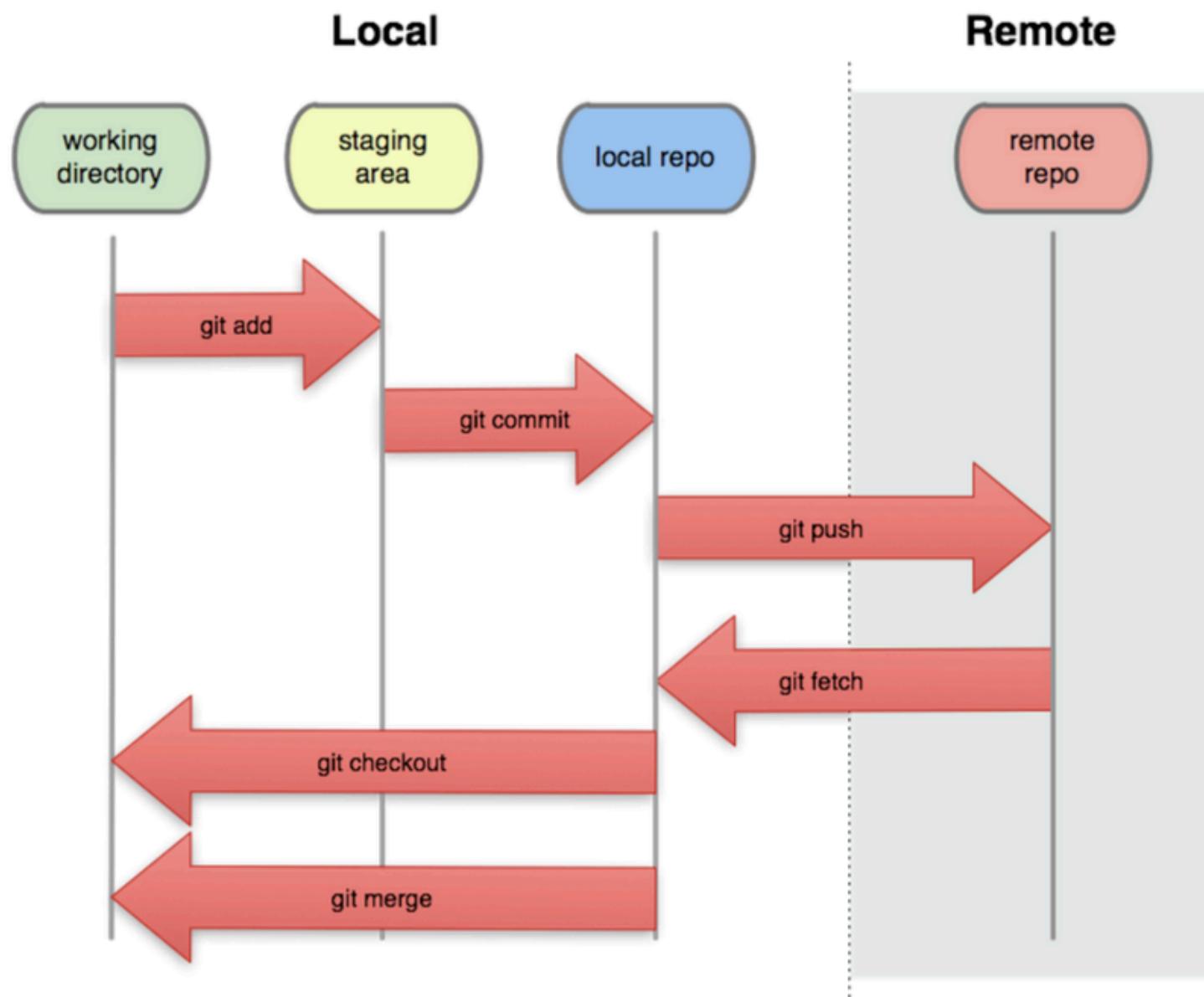


# Налаштуємо доступ до GitLab(SSH)

<http://git-scm.com/>

01. \$ ssh-keygen -t rsa -C "your\_email@example.com"
02. \$ ssh-agent -s
03. \$ ssh-add ~/.ssh/id\_rsa
04. \$ clip < ~/.ssh/id\_rsa.pub

# Шпаргалка



ЖИТТЄВИЙ ЦИКЛ GIT CLI



# Не забуваємо про домашнє завдання

1. Прочитати 2 глави з книги 'Pro Git book', written by Scott Chacon and Ben Straub (є російська версія тут <http://git-scm.com> )
2. Склонувати проект який на вас розшарив Валера
3. Створити свій проект, закомітити туди файл який знайдете в проекті Валери, ішим комітом добавити в репозиторій будь-який текстовий файл (без фанатизму), змінити його та ще раз закомітити. Результат повинен бути запущений в GitLab