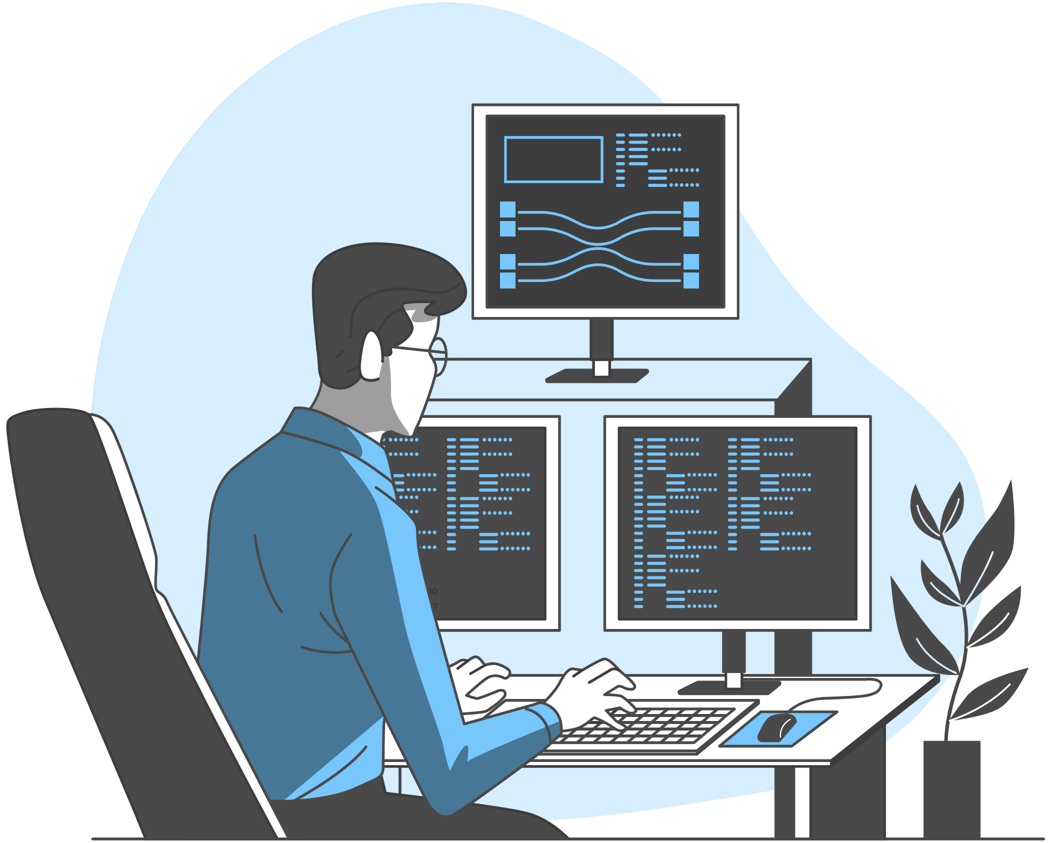
جامعة هواري بومدين للعلوم والتكنولوجيا
USTHB



# Free and open source software

Faculty of biological sciences
Cell pharmacology and signaling team

ABDELAZIZ Ismahane

## WINDOWS

**STEP 1: DOWNLOAD**

1. Open your browser

2. Go to: [https://git-scm.com/download/win]

3. The download starts automatically
• File: Git-2.42.0-64-bit.exe
• Size: ~50 MB

4. If the download does not start automatically:

→ Click **"64-bit Git for Windows Setup"**

...

## PART 1

Launch Git-2.42.0-64-bit.exe
Screen 1: License
→ Next

Screen 2: Installation folder
→ Keep C:\Program Files\Git
→ Next

Screen 3: Components
✓ Windows Explorer integration
✓ Git Bash Here
✓ Git GUI Here
✓ Add a Git Bash Profile
→ Next
…

PART 1

## Screen 4: Start Menu
→ Keep "Git"
→ Next

...

## PART 2

## ⚠ CRITICAL SCREENS ⚠

Screen 5: Default editor
→ Choose "Use Notepad as Git's default editor«
(or VS Code if installed)
→ Next

Screen 6: Initial branch name
→ Select "Override the default"
→ Type: main
→ Next

Screen 7: PATH environment ⚠️ CRUCIAL
→ Choose the MIDDLE option:
"Git from the command line and also 3rd-party«
→ Next

## PART 3

Screen 8: SSH executable
→ Use bundled OpenSSH
→ Next

Screen 9: HTTPS transport
→ Use the OpenSSL library
→ Next

Screen 10: Line ending conversions
→ "Checkout Windows-style, commit Unix-style«
(first option)
→ Next

Screen 11: Terminal emulator
→ Use MinTTY
→ Next

## PARTIE 4

Écran 12: git pull behavior
→ Default (fast-forward or merge)
→ Next
Écran 13: Credential helper
→ Git Credential Manager
→ Next
Écran 14: Extra options
✓ Enable file system caching
✓ Enable symbolic links
→ Next
Écran 15: Experimental
→ Ne rien cocher
→ Install

 Installation en cours... (2-3 minutes)
✓ Launch Git Bash → Finish

## CHOOSE YOUR TERMINAL

Several OPTIONS AVAILABLE:

🎯 Use Git Bash to avoid problems!

1. GIT BASH (✅ RECOMMENDED)
2. • Right-click → "Git Bash Here«
3. • Or Start Menu → Git Bash

• Interface: `$`

## OPENING GIT BASH

METHOD 1: From a folder

1. Navigate to your working folder
2. Right-click in the empty space
3. Select "Git Bash Here"

METHOD 2: From the Start Menu

1. Open the Start Menu
2. Type "Git Bash"
3. Press Enter

WHAT YOU WILL SEE:

```
MINGW64:/c/Users/YourName
$ _
```

## VERIFY INSTALLATION

In Git Bash, type:

```
$ git --version
```

Expected result

```
git version 2.51.0.windows.2
```

✓ If the version is displayed → Installation successful!

## MAC

CHECK IF GIT IS ALREADY INSTALLED

1. Open Terminal:
   - Press ⌘ + Space
   - Type "Terminal"
   - Press Enter

2. Type this command:

```
$ git --version
```

3. Results:

✓ If you see: git version 2.39.0 → Already installed!

✕ If you see: command not found → Need to install

…

## **MAC**

METHOD 1: HOMEBREW (RECOMMENDED)

STEP 1: Install Homebrew
Copy and paste in Terminal:

```
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/
install/HEAD/install.sh)"
```

STEP 2: Install Git

```
$ brew install git
```

STEP 3: Verify

```
$ git --version
```

...

## MAC

METHOD 2: XCODE COMMAND LINE TOOLS

STEP 1: Open Terminal

STEP 2: Type:

```
$ git --version
```

STEP 3: A popup appears:
"The 'git' command requires command line developer tools. Would you like to install?"

[Not Now]  [Install] ← Click this

STEP 4: Wait for installation (10-15 minutes)

STEP 5: Verify:

```
$ git --version
```

## TEST

## Initialize a Git Repository

Method 1

From the project folder

(open the terminal directly in the folder)

```
git init
```

> Create a new Git repository here

Method 2

From anywhere

```
cd path/to/my_folder
git init
```

> Move to the folder and initialize Git

## Git Configuration

### Setting Up Your Identity

Why Configure Git?

Every Git commit includes author information. Setting up your identity ensures proper attribution of your work.

## 🔧 Global Configuration

### Set your name:

```
git config --global user.name "Your Full Name“
```

### Set your name:

```
git config --global user.email "your.email@university.edu"
```

## REPOSITOTY

Why?

to track changes, collaborate safely, and keep a complete history of our project.

```
git config --global init.defaultBranch “name of your main branch”
```

# REPOSITOTY

Why?

to track changes, collaborate safely, and keep a complete history of our project.

```
$ git init
```

git → calls Git, the version control system.

init → tells Git to initialize a new repository in the current folder.

```
$ git status
```

Output/will tell you:

Untracked files (new, not added yet)

Changes not staged (edited but not added)

Changes to be committed (ready for commit)

## COMMIT

Why?

to track changes, collaborate safely, and keep a complete history of our project.

```
$ git commit -m “details"
```

Examples

```
$ git commit -m “XXXXXX"
```

Git replies

```
[main (root-commit) 675871a] first commit - xxxxxxx
3 files changed, 163 insertions(+)
create mode 100644 README.md.txt
create mode 100644 analysis.py.txt
create mode 100644 structures.txt.txt
```

## BRANCH

Why?

safely experiment and develop new features in isolation without breaking the working main code, allowing multiple people to work simultaneously without interfering with each other.

```
$ git branch Name of the new branch
$ git branch
$ git switch branch
```

### IMPORTANT

Always must COMMIT after making changes in one branch then switch to other branch like the main one

```
$ git commit -a -m “Name the modification that has been made in the branch“
```

-a (all) → Automatically stage all tracked modified files

-m "..." → Commit message inline

## MERGE

Why?

to bring our tested changes from the branch back into the main code so everyone can benefit from the new features, otherwise our work stays isolated and useless to the team.

```
$ git merge -m “DETAIL OF YOUR CHANGE" NAME BRANCH
```

git merge → Merges the history of another branch into the branch you are on

-m "..." → Adds a custom merge commit message

NAME_BRANCH → The branch you want to merge into your current branch

Example:

```
git merge -m "Merge NewsMoleculesreduction to main" NewsMolecule
```

## MERGE

🆘 Emergency Commands

Abort merge if something goes wrong

```
git merge --abort
```

See what will be merged before doing it

```
git diff main..feature-branch
```

Undo a merge (dangerous!)

```
git reset --hard HEAD~1
```

## SEND ME YOUR GIT HISTORY