Lab 0

> Auteur: Badr TAJINI - DevOps Data for SWE - ESIEE-IT - 2024/2025

Open Source Code Examples

if you are new to Git, check out the Git for Beginners tutorial here or please go to [Chapter 4] of this course.

An important note for Windows users

While the example code included in this course should work on any operating system, the course also includes many example terminal commands that you run locally. These terminal commands are mostly written in Bash, so to run them, you need either a computer with Unix, Linux, or macOS, or, if you're on Windows, you can use the Windows Subsystem for Linux or Cygwin.

Choose **cygwin.mirror.constant.com** as your website from which to download the libraries, keep the default installation and include vim and nano in your installation.

Opinionated Code Examples

The *core concepts* in the course—e.g., managing infrastructure as code, CI/CD, networking, secrets management, etc—are relatively ubiquitous and applicable across the entire software industry. The *code samples*, however, represent just *one opinionated way* to implement these core concepts. The examples are there to give you hands-on practice, and to help with learning: they are *not* there as a claim that this is the only way or even the best way to do things.

In the real world, there is no single "best" way that applies to all circumstances. All technology choices are trade-offs, and some solutions will be a better fit in some situations than others. The goal of this course is to teach you the underlying concepts and techniques of DevOps and software delivery, and not a specific set of tools or technologies, so once you understand the basics, feel free to explore other technologies and approaches, and always use your judgment to pick the right tool for the job.

You Have to Get Your Hands Dirty

Reading a course is not enough to become an expert at DevOps and software delivery. That's what the code examples in this course are for. Instead of only reading, you get to *learn by doing*.

So don’t just skim the code examples: write the code, run it, and get it working. Moreover, you’ll see sections like the following throughout the labs:

"""

Exercice [Practice]

A list of exercises to try at home.

The examples in this course will get you to the point where you have something working; these “Exercice [Practice]” sections are an opportunity for you to take those examples and tweak them, customize them to your needs, break things, figure out how to fix them, and so on.

"""

For Windows users : Add Cygwin to VSCode

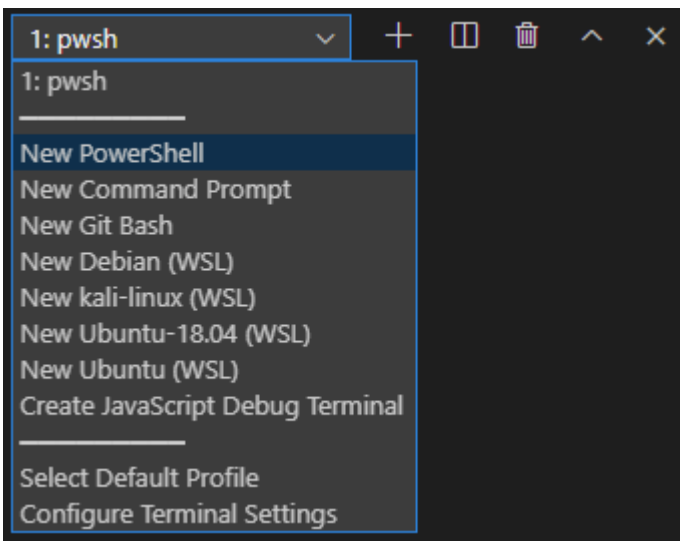
> (Linux and MacOs users are not concerned, skip the Cygwin section)

You'll need to copy and paste a few commands to integrate Cygwin, but before you do, read the new VSCode update as below.

Integrated Terminal [This section is for reading only]

Terminal profiles [This section is for reading only]

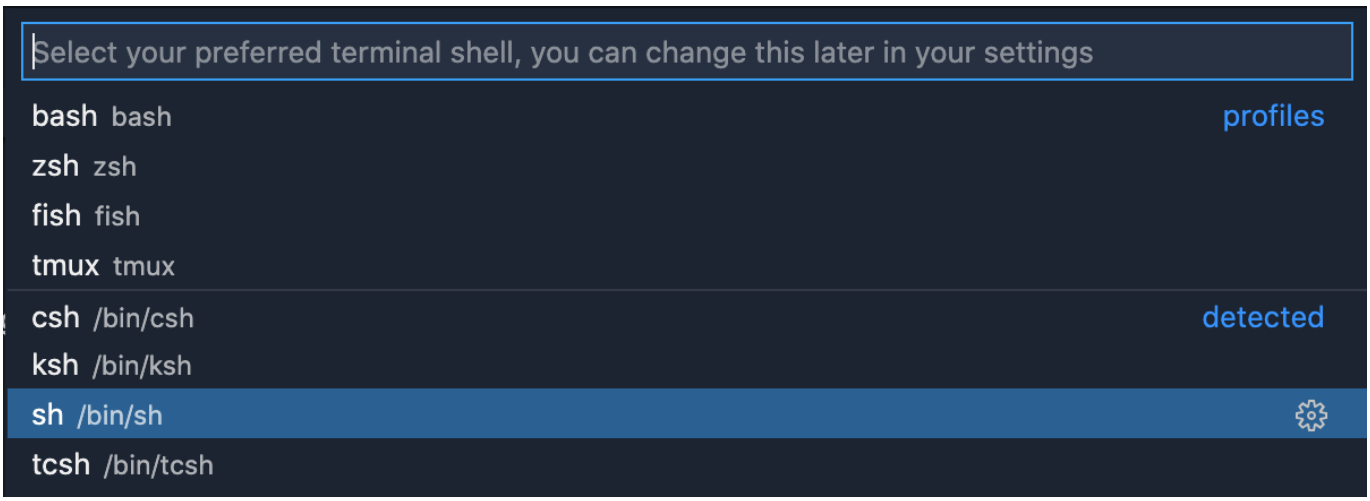
The terminal now supports defined profiles, which appear in the terminal's dropdown, to conveniently launch non-default shells:



VS Code will automatically detect and surface some of the more commonly used shells through this menu, but they can also be configured using the `terminal.integrated.profiles.<platform>` setting. With this setting, it's possible to add new profiles, change existing profiles and remove default profiles. For example:

```
"terminal.integrated.profiles.windows": {
  // Add a PowerShell profile that doesn't run the profile
  "PowerShell (No Profile)": {
      // Some sources are available which auto detect complex cases
      "source": "PowerShell",
      "args": ["-NoProfile"],
      // Name the terminal "PowerShell (No Profile)" to differentiate it
      "overrideName": true
  },
  // Remove the builtin Git Bash profile
  "Git Bash": null,
  // Add a Cygwin profile
  "Cygwin": {
    "path": "C:\\cygwin64\\bin\\bash.exe",
    "args": ["--login"]
  }
}
```

The recommended way to initially add a profile is via the **Select Default Profile** command, which allows creating profiles based on either an existing profile or other detected shells.



Practice [Start here]

Start by opening `settings.json` in Visual Studio Code, follow these steps:

- 1- Open the Command Palette by pressing `Ctrl+Shift+P`.

- 2- Type `> Preferences: Open User Settings (JSON)` and select it. This will open the settings.json file where you can edit your settings directly.
- Copy-paste in the `settings.json` :

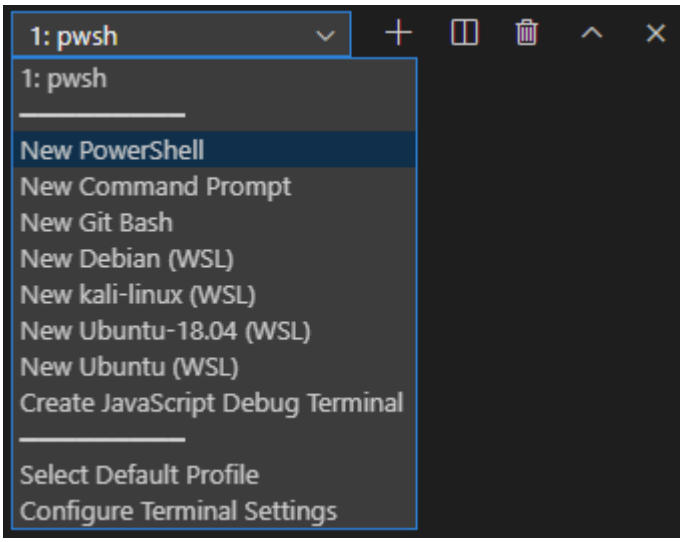
```
// Add a Cygwin profile
"Cygwin": {
  "path": "C:\\cygwin64\\bin\\bash.exe",
  "args": ["--login"]
```

- You'll see something similar in your settings:

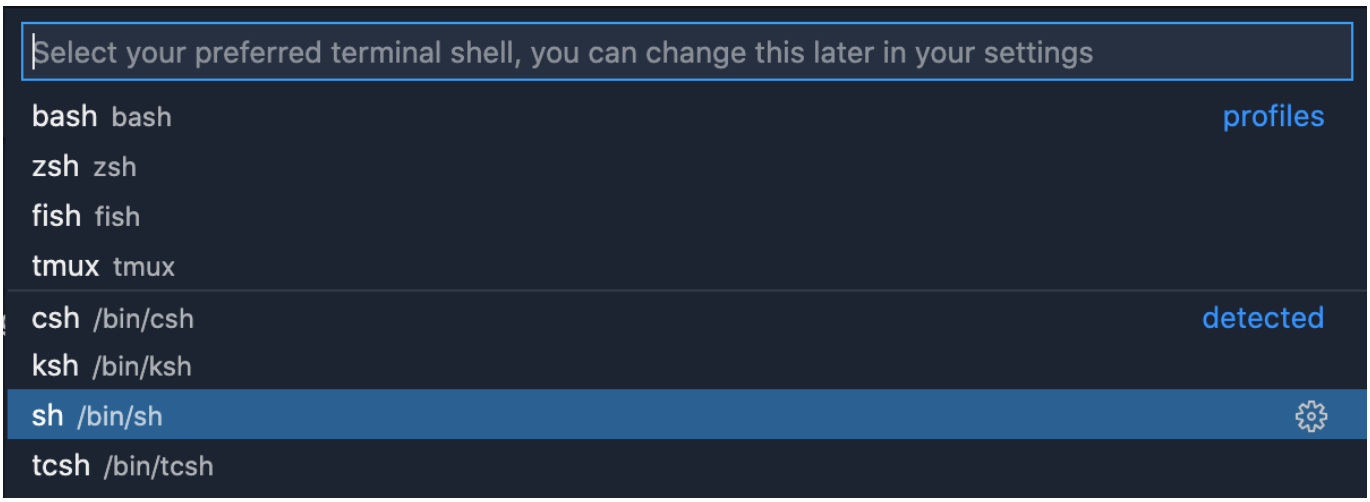
- Then :

Click on the terminal's dropdown, to conveniently launch non-default shells. Add a profile is via the **Select Default Profile** command, which allows creating profiles based on either an existing

profile or other detected shells.

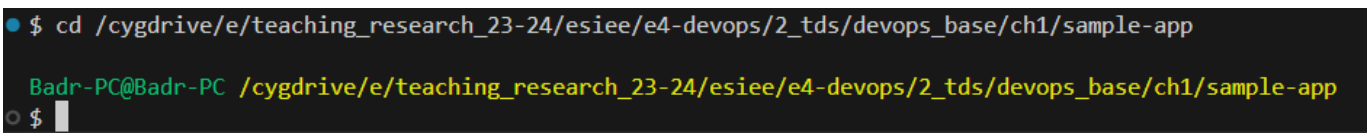


Cygwin will be detected automatically. Choose it as the new default terminal in the list as below :



- The terminal will be activated as shown below:

- To access a folder on your local disk (C or E or D in Windows), you need to go to the `cygdrive` folder as shown below:



- Then, to install other tools more easily, I strongly recommend using Chocolatey in Windows. To do this, try opening PowerShell as administrator and copy and paste this command :

```
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
```

- Restart PowerShell, Then install Nodejs as below :

  > For Mac and Linux users : please follow instructions here

  ```
  # download and install Node.js
  choco install nodejs --version="23.2.0"
  # verifies the right Node.js version is in the environment
  node -v # should print `23`
  # verifies the right npm version is in the environment
  npm -v # should print `10.9.0`
  ```

- And now, to hook up Node Manager with the Cygwin kernel in your terminal. You have to set aliases to `nodejs` and `npm` in your `bash_profile` :


```
nano ~/.bash_profile
```

- Copy and paste the command script below into your `bash_profile` :

  ```
  # write at the end : (or echo "" >> ~/.bash_profile)
  alias npm="/cygdrive/c/Program\ Files/nodejs/npm.cmd"
  alias node="/cygdrive/c/Program\ Files/nodejs/node.exe"
  ```

- Next, try to source it and run the following commands:

```
# source ~/.bash_profile
node -v
# v23.2.0
npm -v
# 10.9.0
```

So now you're ready to get your first laboratory up and running!