ATAD | Software Requirements

This course requires the use of the **GNU Compiler Collection** (`gcc`) *toolchain* and some additional utilities.

The complete list of software is the following:

- Windows Subsystem for Linux (Ubuntu LTS 20.04 or later) – **windows-only**
- GNU Compiler Collection (GCC)
- GNU Project Debugger (GDB)
- GNU Make
- Git (already installed in WSL distro and Linux)
- Valgrind
- Doxygen
- Visual Studio Code (IDE):
  - (Extension) C/C++
  - (Extension) Doxygen Documentation Generator

There are two main ways to get all the required software at your disposal, presented in the following two sections:

1. By manually installing it - only in Windows/WSL and Linux environments, and;
2. By using a pre-generated *docker container* with all the software installed - works in **any operating system** (I'm looking at you, MacOS…).

Then, we'll need VS Code installed with some initial extensions, depending on the above choice, described in section:

3. Install VS Code with Extensions

And, finally, make sure we have `git` installed:

4. Install git.

At the end of this tutorial, proceed to Environment.pdf.

1 | Manual installation (Windows/WSL or Linux)

Requirements

The overall system requirements are the following:

- An updated installation of Windows 10/11 or a recent Linux distribution (all mainstream Linux distributions have the necessary packages);
- 8GB of RAM;
- 5GB of disk space for Windows/WSL (you'll be installing a base Linux environment), or an additional 500MB for a Linux machine.

Installing

> **Linux machine?**
>
> **Go to step 6 if you're on a native Linux environment**. The commands provided are for *Ubuntu/Debian* systems, but you should easily find the required packages for other distributions and transpose the commands to your package manager.

Weblink: Install Linux on Windows with WSL | Microsoft Docs

Prerequisites:

- Make sure your operating system is updated;
- Make sure *Virtualization* is enabled in your computer's BIOS (google: "<your machine brand> enable virtualization bios", e.g., "thinkpad enable virtualization bios").

Quick procedure:

You should be able to install WSL+Ubuntu 20.04 in a single command, as per the *link* above. Open PowerShell or Windows Command Prompt in administrator mode by right-clicking and selecting "Run as administrator", enter the wsl –install command:

```
PS> wsl --install
```

then restart your machine and go to **step** 4.

Alternative procedure (manual):

1. Open PowerShell as Administrator, run the following command and restart.

```
PS> Enable-WindowsOptionalFeature -Online -FeatureName
 ↪  Microsoft-Windows-Subsystem-Linux
```

2. Open **Microsoft Store** and search for "Ubuntu". Install **Ubuntu LTS 20.04 LTS**.
3. Run *Ubuntu* application and wait for installation.
4. During the installation you'll be asked for a **default UNIX user account**. Enter one (no spaces allowed and I recommend all lowercase letters) and your **password** (do not forget this password!)
5. The CLI will be presented afterwards.

Updating Ubuntu image

6. Update installed packages with:

```
$> sudo apt update && sudo apt upgrade
```

… this may take a few minutes. Drink some ☕.

Installing packages

7. Run the following command to install the necessary packages:

```
$> sudo apt install gcc gdb make valgrind doxygen
```

8. Make sure *git* is already installed (if it is, there is no need for reinstall):

```
$> sudo apt install git
$> git --version
```

9. Close the Ubuntu terminal. Done.

2 | Docker Container (All OSes)

This method will give you a containerized development solution and, if you have success, will give you a complete workflow using VS Code and the complete development toolchain (gcc, gdb, valgrind and doxygen).

Requirements

The overall system requirements are the following:

- An updated installation of Windows 10/11, MacOS or Linux distribution;
- 8GB of RAM;
- 3GB of disk space.

Installation

1. Follow the **Docker Desktop** installation for your operating system:
    - https://docs.docker.com/desktop/install/windows-install/
        - **Note:** In Windows you'll still need to enable WSL2 beforehand. See step 1 at "Manual installation (Windows/WSL or Linux").
    - https://docs.docker.com/desktop/install/mac-install/
2. Open a terminal and pull the `brunomnsilva/docker-atad` image:

```
$> docker pull brunomnsilva/docker-atad:latest
```

3. Done.

Important note: The integration of VS Code and *Docker* will be made through a specific extension and configuration folder.

3 | Install VS Code with Extensions

1. Download and install **System installer 64bit** from Download Visual Studio Code - Mac, Linux, Windows.
2. Run VS Code for the first time.

- If you are using WSL, install the `WSL` extension:

> Name: **WSL**
> ID: ms-vscode-remote.remote-wsl
> Description: Open any folder in the Windows Subsystem for Linux (WSL) and take advantage of Visual Studio Code's full feature set.
> Publisher: Microsoft

- If you are using *Docker*, install the `Dev Containers` extension:

> Name: **Dev Containers**
> ID: ms-vscode-remote.remote-containers
> Description: Open any folder or repository inside a Docker container and take advantage of Visual Studio Code's full feature set. Publisher: Microsoft

4 | Install Git

See: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Finally, proceed to Environment.pdf.

Troubleshooting

WSL

Most of the times the problems arise from not enabling *Virtualization* on your BIOS settings.

References:

- https://docs.microsoft.com/en-us/windows/wsl/troubleshooting

Docker: High memory usage on Windows?

If you are experience heavy RAM usage on Windows, this is due to the default behavior of memory allocation of the WSL2/docker backend (e.g., may utilize 80% of available RAM memory, which is way too much for our needs)

If that is the case, create a `.wslconfig` file in you user directory, e.g., `C:\Users\Bruno Silva` with the following contents:

```
[wsl2]
memory=2GB
```

Adjust the maximum memory according to your resources.

None of the methods work?

Check Software_AlternativeMethods.pdf for alternative (unsupported) methods.

Author and support

Bruno Silva (bruno.silva@estsetubal.ips.pt)

You should ask your PL teacher for any help regarding these contents and procedures.