Project0: Project Environment Setup Tutorial

Introduction to Data Communication Networks (2023 Fall)

Instructor: Prof. Kyugnhan Lee

TAs: Jongseok Park, Dongsu Kwak

Deadline: September 17, 2023

Objective

In this project, we will guide you through the steps of setting up your development environment for the upcoming projects for our class.

Please follow the “Guide 1: Environmental Setup” section depending on your preferred environment.

Then follow the “Guide 2: Testing the environment” section to check if your environment is behaving correctly.

Submit the results on ETL, following the instructions in the “Submission” section.

Guide 1: Environmental Setup

Windows:

(A video tutorial is provided for Windows. Please check “**DCN_WSL_Tutorial.mp4**”.)

We assume you are using Windows 10 or 11. We will be using WSL (Windows Subsystem for Linux) to install the Linux development environment on Windows.

A. Installing WSL.

1. Type “Powershell” in the “Search” menu of your taskbar. Run Powershell with administrator privileges.
2. A command terminal will pop up. Type “wsl –install” and hit enter on your keyboard.
3. Wait until the installation is complete.
4. Restart your computer.
5. When you restart your computer, another black command terminal will automatically pop up.
6. Wait until it prompts you to “Enter a new UNIX username”. Enter your preferred username.
7. Enter your password and retype your password. (It is normal that nothing shows up on the terminal when entering your password.)
8. When you are greeted with a text prompt ending with “~$”, you have successfully installed WSL on your Windows PC! However, we must also update and install the necessary programs on WSL.

9. To update WSL, type “sudo apt update && sudo apt -y upgrade” into the terminal and hit enter. You will have to provide your password for this.
10. After updating WSL, type “sudo apt install gcc gdb make” and hit enter. Provide your password once again.

B. Installing VS code.

1. Use your web browser and search “VScode” on your preferred search engine. Enter the website for “Visual Studio Code” and go to the “Download” tab. (link)
2. Download the installer and install VScode on your computer.
3. Run VScode. Click on the “Extensions” tab (Icon with four blocks) on the left menu bar, and search “WSL”. Install the WSL extension from Microsoft.
4. If you want, you can also install the Korean language pack extension. Search “Korean Language Pack” on the “Extension” tab to install it. You will have to restart VScode after installation.
5. After all extensions are installed, click the blue icon with “>” and “<” stacked on top of one another in the bottom left corner. A menu will pop up. Click “Connect to WSL” to connect VScode to your WSL.

C. Using WSL.

WSL is basically a Linux distro running on Windows. You can use most Linux commands and some additional commands on your WSL terminal. We already used “apt” to update and install software to WSL. Here are some more.

- Use “ls” to list all files and directories on your current working directory.
- Use “mkdir <NAME>” to create a new directory (folder) with the name <NAME>.
- Use “cd <NAME>” to change your current working directory to <NAME>.
- Use “cd ../” to change your current working directory to the parent directory.
- Use “explorer.exe .” to open the Windows File Explorer on the current directory.

There are some usage examples in the “DCN_WSL_Tutorial.mp4” video.

D. Moving files between WSL & Windows.

We will now move the included source files for our echo server from Windows to WSL.

1. Unzip “Proj0_0_echo_server.zip” on Windows.
2. Type “mkdir echo_server” on the WSL terminal to create the directory (folder) for the echo server.
3. Type “cd echo_server" on the WSL terminal to change directory to the created “echo_server" directory.
4. Type “exploler.exe .” on the WSL terminal to open Windows File Explorer inside the WSL directory.
5. Drag & drop the unzipped echo server files inside the WSL’s directory.

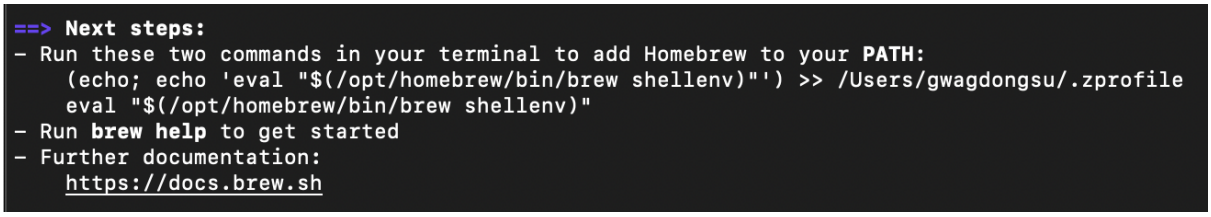
6. Type “ls” on WSL. “Makefile echo.c socket_util.c socket_util.h” should pop up.

Now you are ready to test your development environment! Go to the ‘Guide 2: Testing the environment” section to build and run the echo server on your WSL environment.

Apple Mac:

We assume you are using Mac with M Series chip (M1 or M2).

1. Open your terminal.
2. Download Homebrew (link). If you have once downloaded Homebrew, you can skip below two instructions.
   A. Open your terminal. Type “sudo /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)" ” to download Homebrew.
   B. Type two command below “Next steps” in termal. Below is an example figure, after typing previous command.



3. Type “brew update && brew upgrade” to update your installs.
4. Type “brew install gcc make” to install the necessary programs.
5. Install VScode from the internet. Refer to Section B 1~4 of the Windows guide.
6. Create a working directory named “echo_server".
7. Unzip “Proj0_0_echo_server.zip” and move the contents to your created “echo_server" directory.
8. There should be Makefile, echo.c, socket_util.c, and socket_util.h inside your directory.

Linux:

We assume you are using Ubuntu. However, the steps will be similar in other Linux distros as well.

1. Open your terminal. Type “sudo apt update && sudo apt -y upgrade” to update your installs.
2. Type “sudo apt install gcc gdb make” to install the necessary programs.

3. Install VScode from the internet. Refer to Section B 1~4 of the Windows guide.
4. Create a working directory named “echo_server".
5. Unzip “Proj0_0_echo_server.zip” and move the contents to your created “echo_server" directory.
6. There should be Makefile, echo.c, socket_util.c, and socket_util.h inside your directory.

Guide 2: Testing the environment.

We will use VScode to build and test our echo server. Please follow the included “DCN_WSL_Tutorial.mp4” from the 4m5s mark for better visualization.

A. Opening the working directory in VScode. (4:05 of the video)

1. Click on the “Open Folder…” menu on VScode. It should be under the “File” tab of the top left menu list. If your screen is small, the “File” tab may be hidden under the “≡” or “**…**” tab.
2. Navigate through your menu and open the “echo_server" directory on your VScode. (The “..” tab on the menu means the parent directory.)

B. Opening a terminal and installing extensions. (4:14 of the video)

1. Use the “Terminal” tab and open a new terminal on VScode. The “Terminal” tab may be hidden under the “≡” or “**…**” tab. A terminal will pop up in VScode.
2. You may move your terminal to a preferred location by right-clicking on the terminal menu and using the “Panel Position” tab.
3. Click on the “Extensions” tab (Icon with four blocks) on the left menu bar. Search for “C/C++ Extension Pack” and “Makefile Tools” and install them.

C. Building and running the echo server. (5:12 of the video)

1. Type “make” on the terminal to build our echo server. After the build, an “obj” directory and “echo”, which is the echo server program, should be created. Type “ls” to check this.
2. Type “./echo” to run the created echo server. It should return the usage guide of the echo server.
3. Click on the icon with two adjacent blocks on the terminal menu. It should create a new terminal.
4. Type “./echo server 127.0.0.1 12345” on one of the terminals. This will run the echo server program in the server mode.
5. Type “./echo client 127.0.0.1 12345” on the other terminal. This will run the echo server program in the client mode. The client program should automatically connect to the server program.

6. Type any text in the client program and hit enter. The server program should receive the text and return the text to the client as is.
7. Check if the client has received the same text it sent. If it did, your environment setup is successful!

If you reach this stage, your environment is set up correctly.

If you have any problems, please carefully follow the guide and the video tutorial one more time. If the problem persists, let us know on Thursday's (9/14) lab class!

Submission

Using VScode, modify the source files of the echo server such that the echo server returns a reversed text of what the client has sent.

To be specific, **you only need to fill the "server_function" function** in the file "echo.c" such that it reverses the char* str that it received.

For example, if the client has sent "Hello World!" the server should send "!dlroW olleH" back to the client. You can use this link as a hint if you are having trouble.

We also included an ample number of comments throughout the code. While not necessary, you can study and tinker around with the source code and get a better understanding of socket programming before starting the more serious projects.

You will have to build your program again ("make") to run your updated code.

Please take a screenshot of the client and server exchanging reversed texts, and submit it on ETL, under "<YOUR_NAME>_<YOUR_STUDENT_ID>_Proj0.jpg".

We have included screenshot examples on the next page.

Example screenshot: