

Compiling and running programs on the Computer Science Linux machines

The purpose of this lab work is to ensure that you understand the basics of compiling a C program, and of using the Unix command line. The following instructions take you step by step through creating and compiling various simple programs using Unix commands.

Logging in onto Linux

One way to login to Linux machine from the PCs in the lab.

Go to the start menu → "X2Go Client for Windows" → "X2Go Client".

The X2Go control window will then pop up.

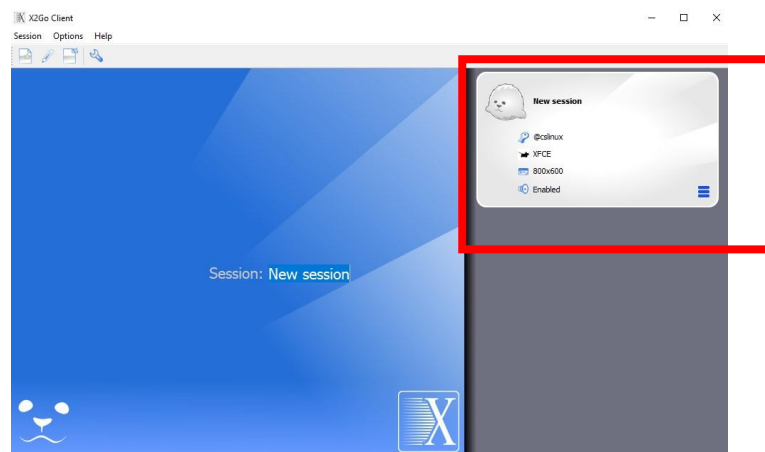


Figure 1: X2Go main screen.

The configuration for a session to UNNC's linux server should already be set on the right hand side (as shown in the red square box in Figure 1). Clicking on this will pop up a login box asking for your username and password (Figure 2). These are your standard UNNC username and password combination. Once they have been entered, X2Go will make a connection to the linux machine.

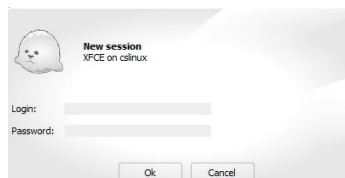


Figure 2: X2Go login screen.

Note: If your X2Go main screen does not have any configurations at all, then you should create a new session and follow the configuration as shown in Figure 3. To create a new session,

Go the menu bar on the top of the X2Go Client → “Session” → “New session ...”.

Note: If your X2Go main screen does not have the correct preset configurations, then you should change the configuration by clicking the “hamburger menu” (see the red square in Figure 1) and change the configuration as shown in Figure 3.

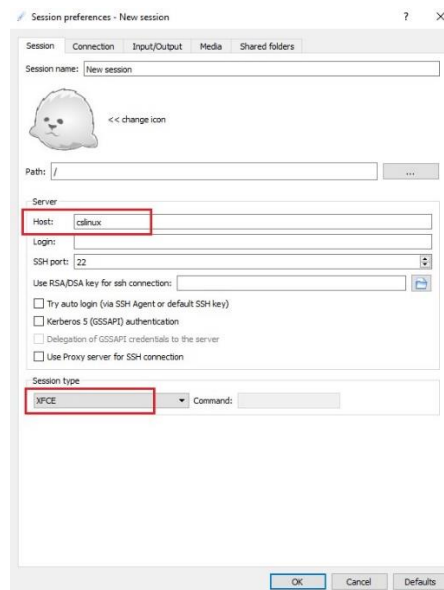


Figure 3: X2Go session configurations.

Note: Make sure the "Host" setting is "cslinux" (all one word, lowercase). The "Session type" at the bottom should be "XFCE".

Once connected, you will be shown an entire new "virtual" desktop that behaves in a similar way to the Windows desktop (see Figure 4). Everything within this window is running on the linux server.

Note: The Windows taskbar is still at the very bottom of the screen. You can minimise the linux window and swap between that and other windows as normal.

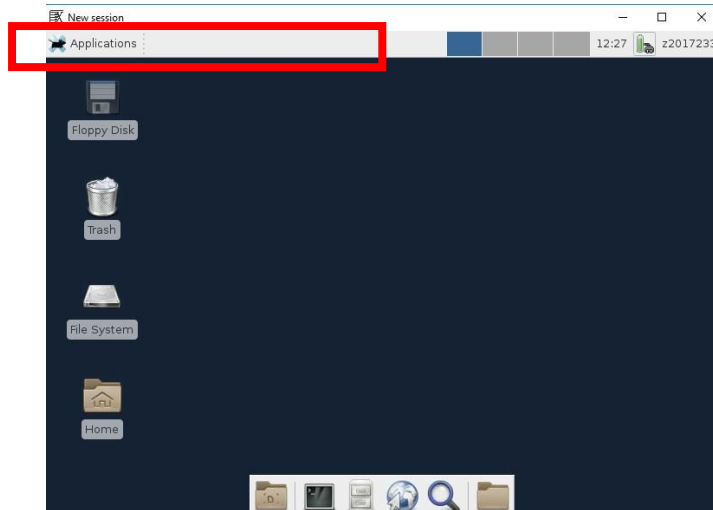


Figure 4: Linux virtual machine.

Note: If the application menu in the top left is missing, you might be missing the entire top panel. You can get the panel back by

Right-clicking on the desktop and selecting "Applications" → "Settings" → "Panel".

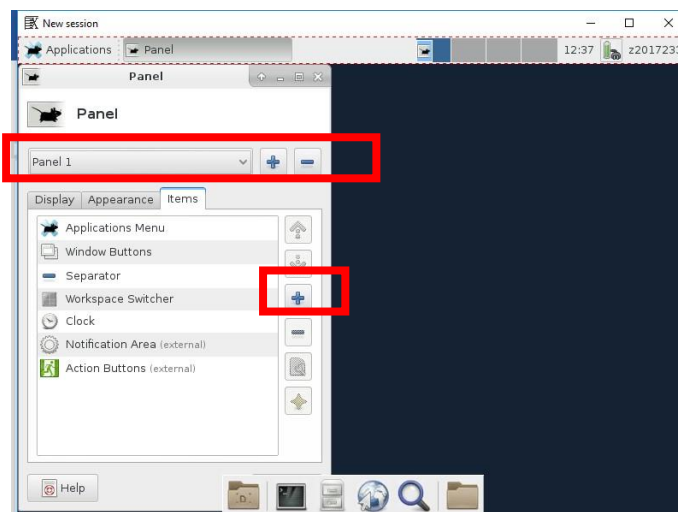


Figure 5: Linux panel setting.

Note: You can then use the "+" button to add a new panel and drag it to the top of the screen. Change the "Length" to 100%. Then go to the "Items" tab and click the "+" button there. Select "Applications Menu" and click the "Add" button. You should then see the applications menu on the panel and can close each of the open windows.

Creating source code

In the top-left corner of your Linux virtual desktop, there is an applications menu.

Go to the "Accessories" → "gedit".

This starts a text editor in a new window, which can be used this to create and edit files. Type in the code as shown below.

Note: Be careful to type it exactly as it appears.

```
#include <stdio.h>

int main(int argc, char* argv[])
{
    printf("Hello world!\n");
    return 0;
}
```

Once you have typed it in, click the save button in the toolbar. Since this is a new file, it will open a file dialog for you to choose where to save the file to. Click on "Home" in the left hand column to select your home directory as the location to save the file. You then need to change the name of the file at the top of the dialog. Give it the name 'hello.c'. Finally, click the save button and the file will be saved and the dialog will close.

Basic Unix commands

In the top-left corner of your Linux virtual desktop, there is an applications menu. To open a terminal,

Go to “Applications” → “Terminal Emulator”.

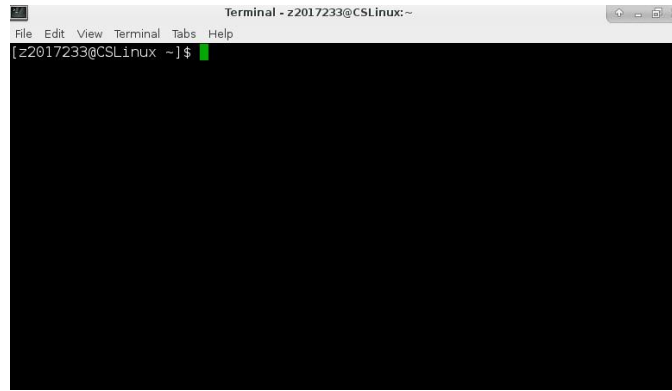


Figure 6: Linux's terminal.

You should then have a window with [your_username@CSLinux ~]\$ prompt as shown in Figure 6. In this example, z2017233 is the username, which will change to your username when you open your terminal.

1. pwd

At the \$ prompt, type 'pwd' (print working directory) to see the directory name of the directory that you are currently in.

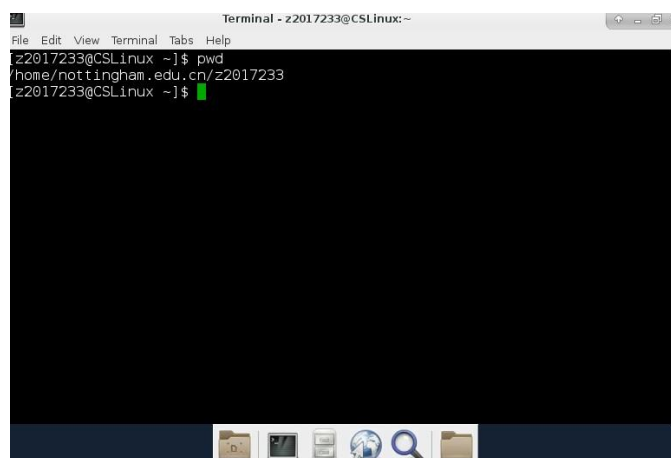


Figure 7: pwd command.

2. ls

At the prompt, type 'ls' to get a list of files.

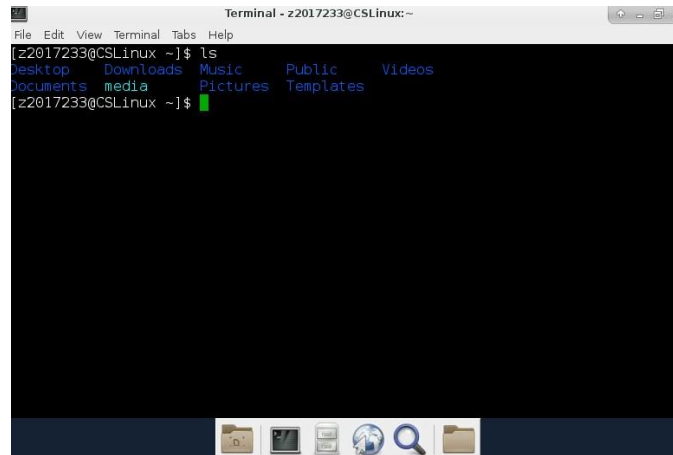
A terminal window titled "Terminal - z2017233@CSLinux:~" with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal shows the command [z2017233@CSLinux ~]\$ ls and its output: Desktop Downloads Music Public Videos Documents media Pictures Templates. The prompt [z2017233@CSLinux ~]\$ is shown again on the next line. The terminal has a dark background and a light blue cursor. The window has a standard Linux desktop environment with a taskbar at the bottom showing icons for a file manager, terminal, and other applications.

Figure 8: ls command.

Note: your files and folders may look different from what is shown on Figure 8.

3. mkdir

At the \$ prompt, type in 'mkdir osc' to create a directory for your AE2OSC work.

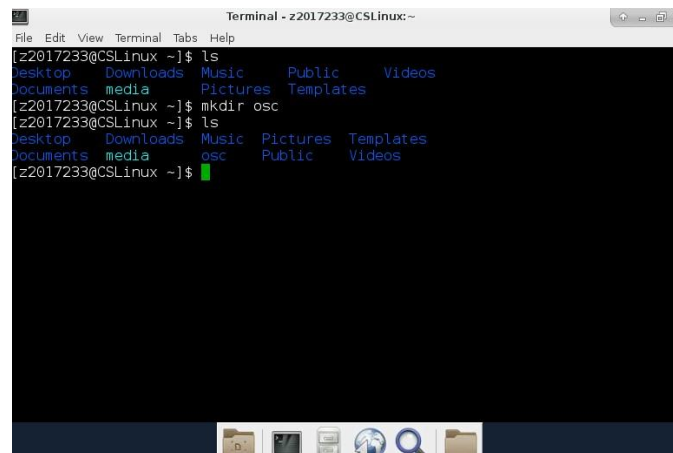
A terminal window titled "Terminal - z2017233@CSLinux:~" with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal shows the command [z2017233@CSLinux ~]\$ ls and its output: Desktop Downloads Music Public Videos Documents media Pictures Templates. Then, the command [z2017233@CSLinux ~]\$ mkdir osc is entered. This is followed by another [z2017233@CSLinux ~]\$ ls command, which shows the same output as before, but with 'osc' added to the list of directories: Desktop Downloads Music Public Videos Documents media osc Pictures Templates. The prompt [z2017233@CSLinux ~]\$ is shown again at the bottom. The terminal has a dark background and a light blue cursor. The window has a standard Linux desktop environment with a taskbar at the bottom showing icons for a file manager, terminal, and other applications.

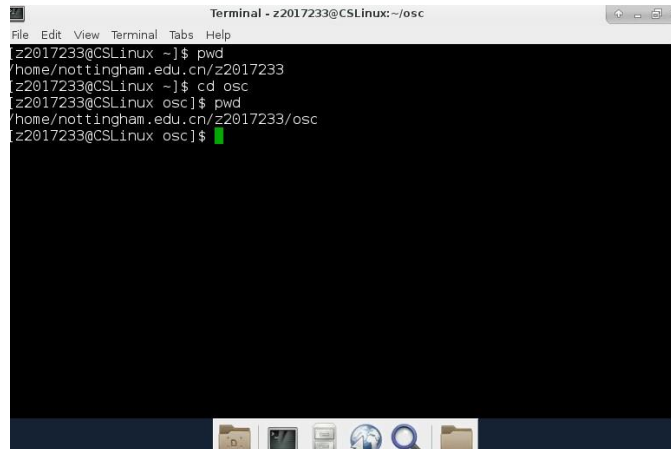
Figure 9: mkdir command.

Note: compare the result from the first 'ls' command with the second 'ls' command, which follows the 'mkdir osc'.

4. `cd`

At the `$` prompt, type '`cd osc`' to change your working directory. The working directory is the directory which you are currently in.

Note: In this example, `osc` represents the name of the folder you wish to go to.

A terminal window titled "Terminal - z2017233@CSLinux: ~/osc" showing a sequence of commands and their outputs. The user starts in the home directory, runs 'pwd' to confirm the path, then runs 'cd osc' to change the directory. A second 'pwd' command confirms the new directory. The window has a menu bar (File, Edit, View, Terminal, Tabs, Help) and a taskbar at the bottom with various application icons.

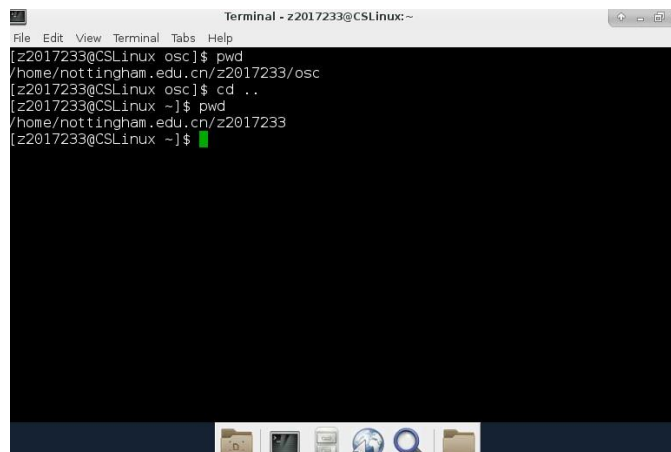
```
Terminal - z2017233@CSLinux: ~/osc
[z2017233@CSLinux ~]$ pwd
/home/nottingham.edu.cn/z2017233
[z2017233@CSLinux ~]$ cd osc
[z2017233@CSLinux osc]$ pwd
/home/nottingham.edu.cn/z2017233/osc
[z2017233@CSLinux osc]$
```

Figure 10: `cd` command.

Note: compare the result from the first '`pwd`' command with the second '`pwd`' command, which follows '`cd osc`'.

5. `cd ..`

At the `$` prompt, type '`cd ..`', to step out of the current directory.

A terminal window titled "Terminal - z2017233@CSLinux: ~" showing a sequence of commands and their outputs. The user is currently in the 'osc' directory, runs 'pwd' to confirm the path, then runs 'cd ..' to move up one directory level. A second 'pwd' command confirms the new directory. The window has a menu bar (File, Edit, View, Terminal, Tabs, Help) and a taskbar at the bottom with various application icons.

```
Terminal - z2017233@CSLinux: ~
[z2017233@CSLinux osc]$ pwd
/home/nottingham.edu.cn/z2017233/osc
[z2017233@CSLinux osc]$ cd ..
[z2017233@CSLinux ~]$ pwd
/home/nottingham.edu.cn/z2017233
[z2017233@CSLinux ~]$
```

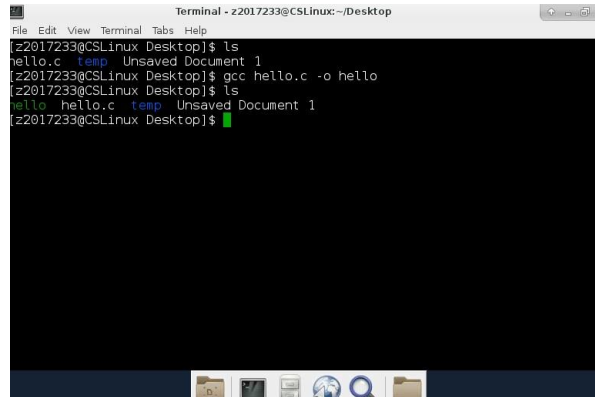
Figure 11: `cd ..` command.

Note: compare the result from the first '`pwd`' command with the second '`pwd`' command, which follows '`cd ..`'.

Compile your program

You can compile the c file using by typing in 'gcc hello.c -o hello'. This will compile the source file called 'hello.c' and write the output to a file called 'hello'. The '-o' command line option specifies the output filename. If you do not provide this a file called 'a.out' will be used. This will still contain your program, but will have a less useful name.

Note: replace 'hello' with whatever filename your program is in.

A terminal window titled 'Terminal - z2017233@CSLinux: ~/Desktop' with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal shows the following commands and output:

```
z2017233@CSLinux Desktop]$ ls
hello.c  temp  Unsaved Document 1
z2017233@CSLinux Desktop]$ gcc hello.c -o hello
z2017233@CSLinux Desktop]$ ls
hello  hello.c  temp  Unsaved Document 1
z2017233@CSLinux Desktop]$
```

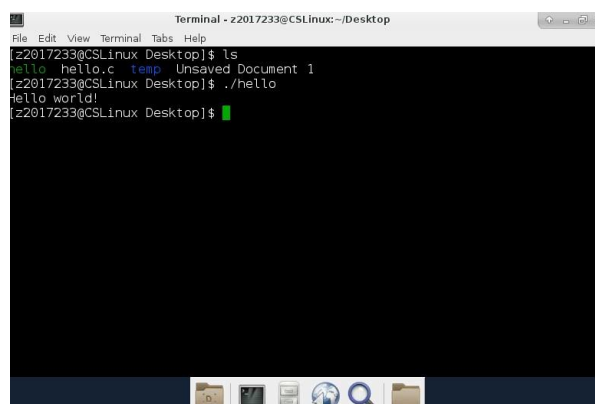
Figure 12: gcc command.

Note: compare the result from the first 'ls' command with the second 'ls' command, which follow the 'gcc hello.c -o hello'.

Execute your program

You can then execute the program by typing in './hello'. The '.' refers to the current directory, so this means execute the file called 'hello' in the current directory.

Note: replace 'hello' with whatever filename your '.out' is in.

A terminal window titled 'Terminal - z2017233@CSLinux: ~/Desktop' with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal shows the following commands and output:

```
z2017233@CSLinux Desktop]$ ls
hello  hello.c  temp  Unsaved Document 1
z2017233@CSLinux Desktop]$ ./hello
Hello world!
z2017233@CSLinux Desktop]$
```

Figure 13: execute a c program.