Introduction to PHY 224

1 Introduction PHY 224 block course

The PHY 224 is a block course dedicated for teaching how to program in C/C++ language in a linux environment. The course will cover some of the basic tools available in C and C++. The outline of the course can be found below:

1. Introduction to programming in Linux.
2. Introduction to basics of programming.
3. Data processing in C/C++.
4. Data handling in C/C++.
5. Memory management in C/C++.
6. Handling large data in C and C++.
7. Objects and Classes.
8. Scripting using ROOT 5 and 6.
9. Programming using ROOT 5 and 6 packages.
10. Parallel programming in C/C++.

Each lecture will be held in the classroom Y36-K-08 using the following timetable:

- 0900∼0915: In the lecture room, the lecturer will provide introduction to and some examples of tools to be used in the following practical session.
- 0915∼1015: The students will divided into 3 groups. In each group, either a lecturer or a TA will be available for the students to ask questions. The students are expected to go through the exercises given in the lecture notes during this time.
- 1015∼1030: There will be a 15 minutes break before continuing with the lecture.
- 1030∼1045: In the lecture room, the lecturer will provide introduction to and some examples of more tools to be used in the following practical session.
- 1045∼1145: The students will again be divided into 3 groups. In each breakout room, either a lecturer or a TA will be available for the students to ask questions. The students are expected to go through the exercises given in the lecture notes during this time.
- 1145∼1200: The students may ask any of the remaining questions that they may have regarding the lecture material or the exercises. The lecturer will give the students a concluding remark and end the lecture for the day.

The course will start on 2021-08-23 and end on 2021-09-03. The lectures will be held on each weekday from 0900 until 1200 until the last day of the course.

1.1 Structure of the lecture notes

All of the lecture notes are designed to be used during the course. The lecturer will display the lecture notes as well as a linux terminal during the lecture to demonstrate some examples live. The students are expected to never copy-and-paste the materials in the lecture notes. The PDF has been encoded to be incompatible to be copied-and-pasted for programming purposes.
However the students are welcome to type the given example codes letter-by-letter from the lecture notes. This is to encourage the students to write a proper programs from scratch rather than copying-and-pasting.
Within the lecture notes students will find that the examples of codes and scripts have been colour-coded. There are three colours:

```
bashcommand
```

is used for demonstrating what to type into a linux terminal (commands).

```
1 bashscript
```

Is used for typing into a text editor and saving as a file to be executed from a linux terminal (scripts).

```
1 c++code
```

Is used for proper c++ code written into a text editor, saved then compiled into an executable and executed from the linux terminal (code).
More information on the above can be found in the first lecture note.

1.2 Course requirements

The students must have access to a computer with internet access. The students must also be able to access some of the local computers located physically in the physics institute through secure shell (ssh) remotely. You will find in the next section a set of instructions on how to use SSH from a computer with any linux operating system, Windows 10 and MacOS X. However, if you have any questions regarding how to gain access to the machines in the physics institute please contact one of our lecturers Dr. Roland Bernet https://www.physik.uzh.ch/en/researcharea/lhcb/team/senior-researchers/bernet.html.

2 Secure shell (SSH)

Secure shell is a very basic tool available for many operating systems by default. SSH allows users to open up a "shell" or an encrypted enclosed environment, allowing users to log in and operate the computers remotely. However there are multiple versions and functions of SSH, and for the purpose of this course, you need to be able to forward the X display. In linux environment, the X-server handles display by default (unless reconfigured) and forwarding X display is typically not an issue. However some external tools are required such as XMing (http://www.straightrunning.com/XmingNotes/) or XQuartz (https://www.xquartz.org/) for Windows and MacOS platforms. Below you will find instructions on how to use them. Please note, for the purpose of this course, only download the .exe and .dmg available on the course website https://www.physik.uzh.ch/de/lehre/PHY224/FS2021.html.

2.1 From Linux

As stated above, using SSH with X display forwarding is the most straightforward. You need to open a terminal and type in the following:

```
ssh -X username@remote.com.put.er
```

Obviously you need to change the "username" and "remote.com.put.er" with your actual username and the domain. For instance

```
~> ssh -X stelee@ohm21.physik.uzh.ch
The authenticity of host 'ohm21.physik.uzh.ch (130.60.165.162)' cant be
↪  established.
ECDSA key fingerprint is SHA256:x7ERYVt/uJs0Q44o4sBqo28r96ecHthfmrD2KW9q3pQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ohm21.physik.uzh.ch,130.60.165.162' (ECDSA) to
↪  the list of known hosts.
Password: #Here, you will not see your password being typed.
Have a lot of fun...
stelee@ohm21:~>
```

To make sure that the X display is being forwarded correctly, you should try opening one of the simplest X applications.

```
xclock
```

You should be able to see the following:

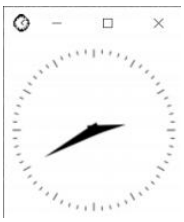

Figure 1: X-clock from your display may appear in different colours.

If you can see the X-clock, your system is good to go.

2.2 From MacOS X

The MacOS X will allow you to SSH into a linux computer remotely by default from the terminal. However, MacOS X does not have a X server installed by default. This is where you have to install something external. The instructions are as follows.

1. Download and install XQuartz (https://www.physik.uzh.ch/dam/jcr:ee978d2d-2c55-4ad5-949f-aea05edd XQuartz-2.7.11.dmg).

2. Using a text editor add the following lines

```
1  XAuthLocation /opt/X11/bin/xauth
```

   to

```
/etc/ssh/ssh_config
```

   If done properly, above file should look as follows:

3. Test your setup by connecting to a remote server and opening the X-clock.

```
#   Tunnel no
#   TunnelDevice any:any
#   PermitLocalCommand no
#   VisualHostKey no
#   ProxyCommand ssh -q -W %h:%p gateway.example.com
#   RekeyLimit 1G 1h

Host *
        SendEnv LANG LC_*
XAuthLocation /opt/X11/bin/xauth
```

Figure 2: Your /etc/ssh/ssh_config should look similar to above.

```
ssh -X username@remote.com.put.er
```

Obviously you need to change the "username" and "remote.com.put.er" with your actual username and the domain. For instance

```
ssh -X stelee@ohm21.physik.uzh.ch
```

and once logged in

```
xclock
```

You should see something similar to the following:

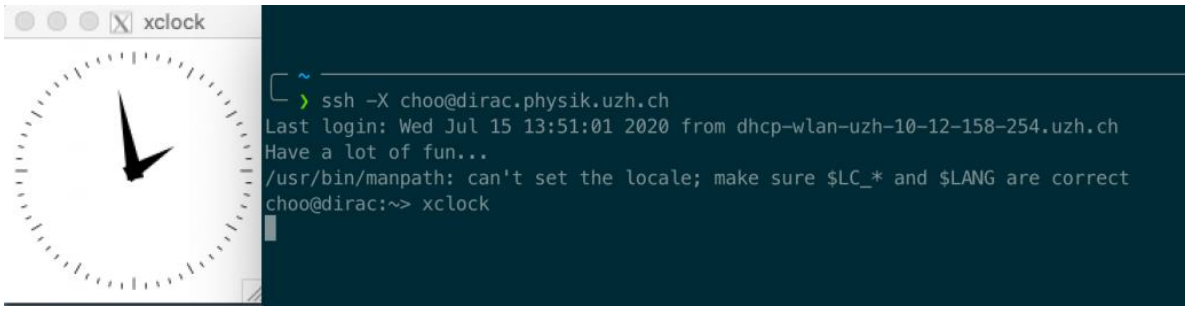

Figure 3: When you log into your own account in the physics server, your terminal output should show similar results. You should also use xclock command to check that the X display is being forwarded correctly.

If you can see the X-clock remotely, you are good to go. If you have any trouble getting X-server forwarding to work on MacOS X. Please let us know.

2.3 Pimping your Mac setup

The introduction of Mac OS X in 2001 brought about a major change: the system is based on a BSD Unix kernel, making it very similar to Linux operating systems. As a result, it can be tweaked to be very suitable for software development with Linux environment, and can almost transparently be used for most of the tasks Linux distributions can do. We will in this optional section go through the most important components.

2.3.1 Install Xcode

The first step is to install the coding environment. Apple ships the develoment tools with Xcode IDE (Integrated Development Environment), which contains the compiler, the necessary system libraries, as well as a code editor with debugger and terminal. It is not necessary to use Xcode for coding, but it is necessary to install it in order to get the compiler and the system libraries, so let's download it from the App store:

https://apps.apple.com/us/app/xcode/id497799835?mt=12

Once Xcode is downloaded, we can install the development tools. For that, open the terminal and type:

```
xcode-select --install
```

You should now have the tools installed to build C++ projects.

2.3.2 Upgrading the terminal

The native Mac OS terminal can do everything, but newer, more feauture-rich terminals have emerged. A convenient and widely used alternative is *iTerm*, which can be downloaded here:

https://iterm2.com/downloads.html

After installing it, you can use it interchangeably with the native terminal.

2.3.3 Installing a package manager

One nice feature of Linux distributions is that they come with a package manager, which can download and install a number of precompiled binaries from a central software repository using the command line.
It turns out open source projects exist for Mac, which porvide the same features and power as on Linux (some of them maintained by former Apple employees). The two main package managers for Mac OS are *MacPorts* or *Homebrew*. The choice between the two is a personal matter, and we will try to provide some basic idea of the pros and cons of each here, but it is important to **ONLY INSTALL ONE OF THEM**. Since they are package managers, they will keep a catalogue of installed and available software, and make sure the versions of the packages match each other. Packages installed by one package manager will not be known by the other package manager, and this can result in clashes, mismatches in versions, and ultimately render your system unusable.

MacPorts
Mac ports is historically the first package manager for Mac OS, and recieved support from Apple employees. It will offer you the latest and greatest version of each software, making it the best choice for cutting edge development. As a result, it will tend to install a lot of packages and dependencies, and each update might trigger updates in already installed packages.
Instructions on installing MacPorts can be found here: https://www.macports.org/install.php

You can then install packages with the command:

```
port install [package]
```

Homebrew
The other popular alternative is Homebrew. Homebrew tries to stick as much as possible to the system libraries, and completes the OS by adding the missing components. The resulting

configuration will be closer to the native Mac OS version, and might not allow to update to the latest versions. On the other hand, the system is left closer to the initial software it shipped with, and is thus less likely to prevent optimal functioning with tools targeting specifically Mac OS.
Instructions on installing Homebrew can be found here: https://brew.sh/

Packages can then be installed with the command:

```
brew install [package]
```

2.4 From Windows 10

Windows 10 is a little bit trickier. You must install Xming and also enable Windows Subsystem for Linux, and install a linux operating system within the Windows 10 operating system. The instructions are as follows:

1. Download and install X-Ming (https://www.physik.uzh.ch/dam/jcr:0b8cc71b-7788-4a7c-b0d0-d47b228b
   Xming-6-9-0-31-setup.exe)

2. Enable Windows Subsystem for Linux by following the red highlights in the following figure:

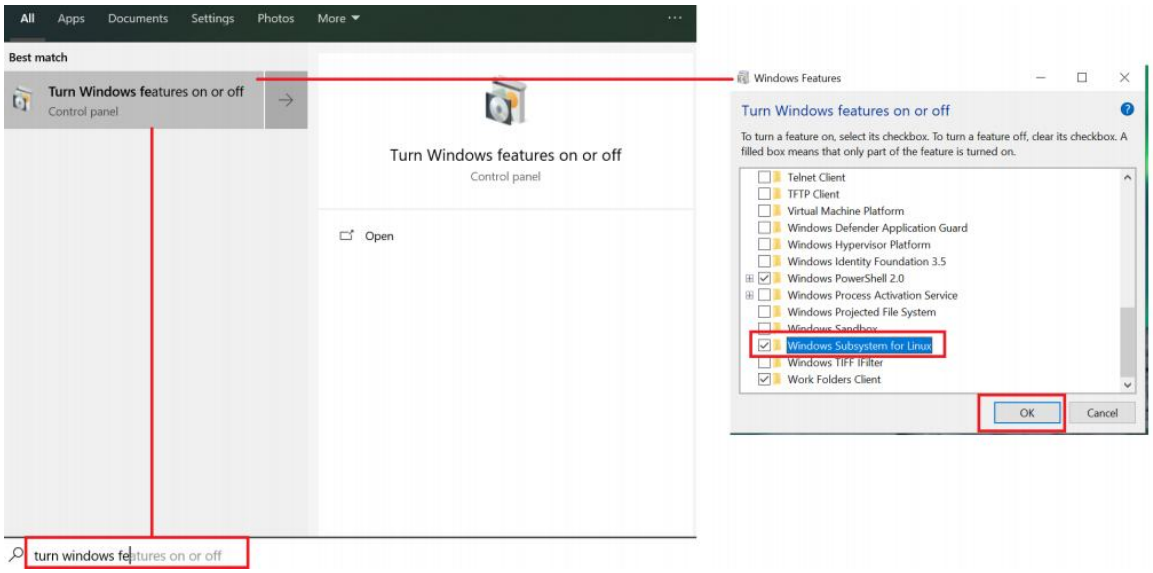

Figure 4: You must search for "Turn Windows features on or off" from Start menu, and click on "Turn Windows features on or off (Control Panel)". You need to then check "Windows Subsystem for Linux and hit OK

.

3. Windows will then install the feature and you must reboot your Windows 10 computer.

4. When the computer reboots, go to Microsoft store and install OpenSUSE LEAP 15.2 following the red highlights in the following figures

5. Once the Microsoft store installs OpenSUSE Leap 15.2, open it and follow the instructions displayed to create a user, and initialize it.

6. once initialized, become the super user or "root" by typing in

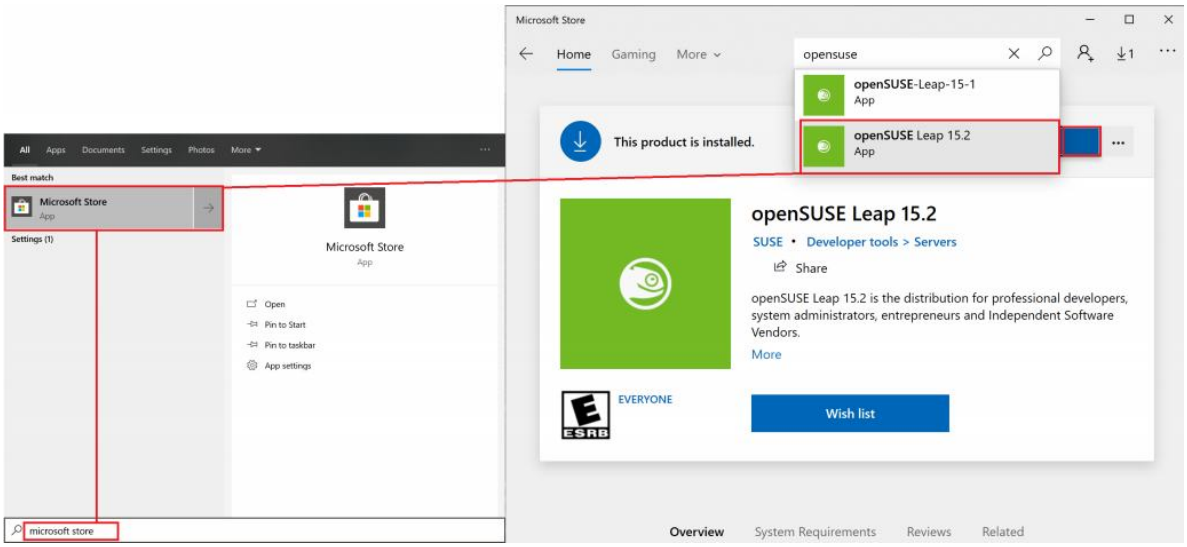

Figure 5: You must search for "Microsoft Store" on "Microsoft Store (App)". You then need to search for OpenSUSE Leap 15.2 from the Microsoft Store and install it.

```
su
```

and when you are prompted to do so, type in your password.

7. You then need to install a text editor called "emacs" using the following line.

```
zypper in emacs
```

and say yes when prompted for confirmation.

8. You then need to get out of super user by typing in

```
exit
```

or use the key combination CTRL + d.

9. You need to modify your default loading environment called ".bashrc" by the following.

```
emacs -nw ~/.bashrc
```

10. Add the following entry to the very bottom of your ".bashrc"

```
1 export DISPLAY=0:0
```

11. save and exit by pressing "Ctrl+x, Ctrl+s" then "Ctrl+x, Ctrl+c". If EMACs asks if you want to add a line at the end, or to confirm saving the file, type in yes.

12. close your OpenSUSE LEAP 15.2 session by typing in:

```
exit
```

13. Turn on X-Ming from the start menu.

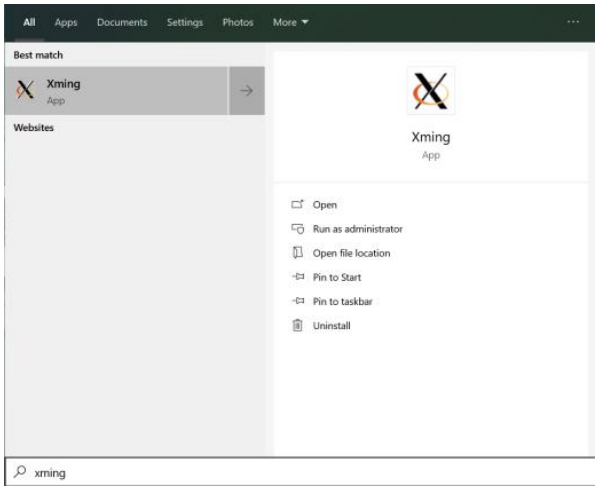

Figure 6: Search for "Xming" in start menu and click on Xming (App)

.

14. At this point you have linux installed within your Windows 10 operating system. You can now test it in the same manner as you could on linux by opening OpenSUSE Leap 15.2 application.

```
ssh -X username@remote.com.put.er
```

Obviously you need to change the "username" and "remote.com.put.er" with your actual username and the domain. For instance

```
~> ssh -X stelee@ohm21.physik.uzh.ch
The authenticity of host 'ohm21.physik.uzh.ch (130.60.165.162)' cant be
↪  established.
ECDSA key fingerprint is
↪  SHA256:x7ERYVt/uJs0Q44o4sBqo28r96ecHthfmrD2KW9q3pQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
↪  yes
Warning: Permanently added 'ohm21.physik.uzh.ch,130.60.165.162' (ECDSA)
↪  to the list of known hosts.
Password: #Here, you will not see your password being typed.
Warning: No xauth data; using fake authentication data for X11
↪  forwarding.
Have a lot of fun...
stelee@ohm21:~>
```

To make sure that the X display is being forwarded correctly, you should try opening one of the simplest x applications.

```
xclock
```

You should be able to see the following:

Once you have xclock working from Windows 10 through Xming and OpenSUSE. You are good to go. Remember to turn on X-ming before you try to forward X display, otherwise nothing will be displayed.

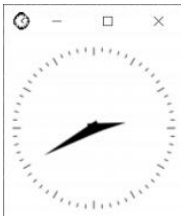

Figure 7: X-clock from your display may appear in different colours.

3 Version Control

The students should note that the only reason that we are asking you to use the local machines in the physics institute is because we know the versions and the packages installed on these machines well.
All of the exercises and the programs for this course have been tested on the machines in the institute. However your local linux, MacOS X or Windows Subsystem Linux (WSL) can just as well handle everything locally from your computer.

This is very much encouraged since we have limited number of computers available at the institute.

This means that as long as you have the correct packages and correct versions of the packages installed in your local machine, you can do everything for this course without the use of SSH. Please let us know as soon as possible that you would like to use your own local machine instead of the one at the institute remotely. Just make sure you check that your version of gcc and g++ are later than 7 by typing in:

```
gcc --version ; g++ --version
```

from your linux (or Unix[1], or WSL) terminal.
The students should also note that the 8th and 9th lecture will require them to use packages from ROOT (https://ph-root-2.cern.ch/). First, please note that "ROOT" is different from the super-user "root" and that depending on your programming background, the installation of ROOT may not be trivial.
Students are encouraged to try to install ROOT themselves in their local machines as there are many resources available online. In particular, it is available as a package in many of the operating system's package managers (including MacPorts, Homebrew, Conda, Debian https://cern.ch/amadio/root/root_6.26.10_ubuntu22_amd64.deb14, RedHat). However ROOT is already installed and available for use on the local machines at the institute.

[1] MacOS is Unix based