Linux:
An introduction

iT Centre Learning

iT services

UNIVERSITY OF OXFORD

How to Use this User Guide

This handbook accompanies the taught sessions for the course. Each section contains a brief overview of a topic for your reference and then one or more exercises.

Exercises are arranged as follows:

- A title and brief overview of the tasks to be carried out
- A numbered set of tasks, together with a brief description of each
- A numbered set of detailed steps that will achieve each task

Some exercises, particularly those within the same section, assume that you have completed earlier exercises. Your teacher will direct you to the location of files that are needed for the exercises. If you have any problems with the text or the exercises, please ask the teacher or one of the demonstrators for help.

This book includes plenty of exercise activities – more than can usually be completed during the hands-on sessions of the course. You should select some to try during the course, while the teacher and demonstrator(s) are around to guide you. Later, you may attend follow-up sessions at ITLP called Computer8, where you can continue work on the exercises, with some support from IT teachers. Other exercises are for you to try on your own, as a reminder or an extension of the work done during the course.

Text Conventions

A number of conventions are used to help you to be clear about what you need to do in each step of a task.

- In general, the word *press* indicates you need to press a key on the keyboard. *Click*, *choose* or *select* refer to using the mouse and clicking on items on the screen. If you have more than one mouse button, click usually refers to the left button unless stated otherwise.
- Names of keys on the keyboard, for example the Enter (or Return) key are shown like this: ENTER.
- Multiple key names linked by a + (for example, CTRL+Z) indicate that the first key should be held down while the remaining keys are pressed; all keys can then be released together.
- Words and commands typed in by the user are shown **like this**.
- Labels and titles on the screen are shown like this.
- Drop-down menu options are indicated by the name of the options separated by a greater-than arrow; for example, *File* > *Print*. In this example you need to select the option *Print* from the *File* menu or tab. To do this, click when the mouse pointer is on the *File* menu or tab name; move the pointer to *Print*; when *Print* is highlighted, click the mouse button again.
- A button to be clicked will look **like this**.
- The names of software packages are identified *`like this`*, and the names of files to be used “like this”.

Contents

Exercises....3
1 Introduction....4
1.1 Aims for Today....4
1.2 Course Outline....4
2 Overview of Linux....6
2.1 Desktop layout....6
2.2 General settings....8
3 Programs/applications....9
3.1 Basic Linux applications....9
3.2 Office applications....9
3.3 Program installation....11
3.4 Working with proprietary programs....15
4 The file system....17
4.1 The structure of the file system....17
4.2 External drives....19
4.3 File/directory permissions....20
5 Basic command line operations....20
6 System configuration....29
6.1 Config files....29
6.2 Exploring other desktop environments....30
7 Installing Linux for personal use....31

Exercises

Ex. 1: Edit the taskbar 6
Ex. 2: Make more extensive changes to the layout 7
Ex. 3: Take a screenshot and save it to the desktop 9
Ex. 4: Create several different types of office documents 11
Ex. 5: Install a few programs of your choice 15
Ex. 6: Install a program through PlayOnLinux 16
Ex. 7: Configure the file manager 18
Ex. 8: Open the terminal in a directory from within the file manager 20
Ex. 9: Do some basic terminal operations 24
Ex. 10: Install Vim from the command line 25
Ex. 11: Edit a file in Vim 27
Ex. 12: Adding aliases to Bash 29

1 Introduction

Today's course is divided into six parts each of which consists of a presentation followed by exercises.

- Overview of Linux and the organization of the operating system
- Some common programs and their installation
- The file system
- Basic command line operations
- System configuration
- Installing Linux for personal use

1.1 Aims for Today

The course is designed to help you become a confident Linux user. The topics covered are described in the Course Outline below.

1.2 Course Outline

1.2.1 Session One: Overview

This session will cover:

- A brief history of Linux.
- What is Open Source software?
- Who uses Linux.
- Familiarizing yourself with the desktop

Exercises:

- Exploring and configuring the Linux desktop

1.2.2 Session Two: Common programs

This session will cover:

- Basic Linux applications
- Office applications
- Program installation
- Working with proprietary programs

Exercises:

- Exploring different Linux applications
- Installing a program graphically and from the terminal

1.2.3 Session Three: The file system

This session will cover:

- The structure of the file system

- Using the file manager
- External drives
- File/directory permissions

Exercises:

- Change the file manager layout
- Create new files in user and root directories
- Search for files
- Open a terminal in a specific directory

1.2.4 Session Four: Basic command line operations

This session will cover:

- Moving around the file system
- Getting information about your system
- Listing, copying, and moving files
- Running a program from the terminal

Exercises:

- Change into different directories, list files, create and edit a file, copy/move it

1.2.5 Session Five: System configuration

This session will cover:

- Navigating the system settings
- Common configurations
- Dot files
- Trying other desktop environments

Exercises:

- Add/change some keyboard shortcuts
- Change different aspects of the interface
- Change some settings in the bash terminal config file

1.2.6 Session Six: Installing Linux for personal use

This session will cover:

- Different options for Linux distros
- Creating a Live USB installer
- Booting from USB and installing
- Dual booting Windows and Linux

Exercises

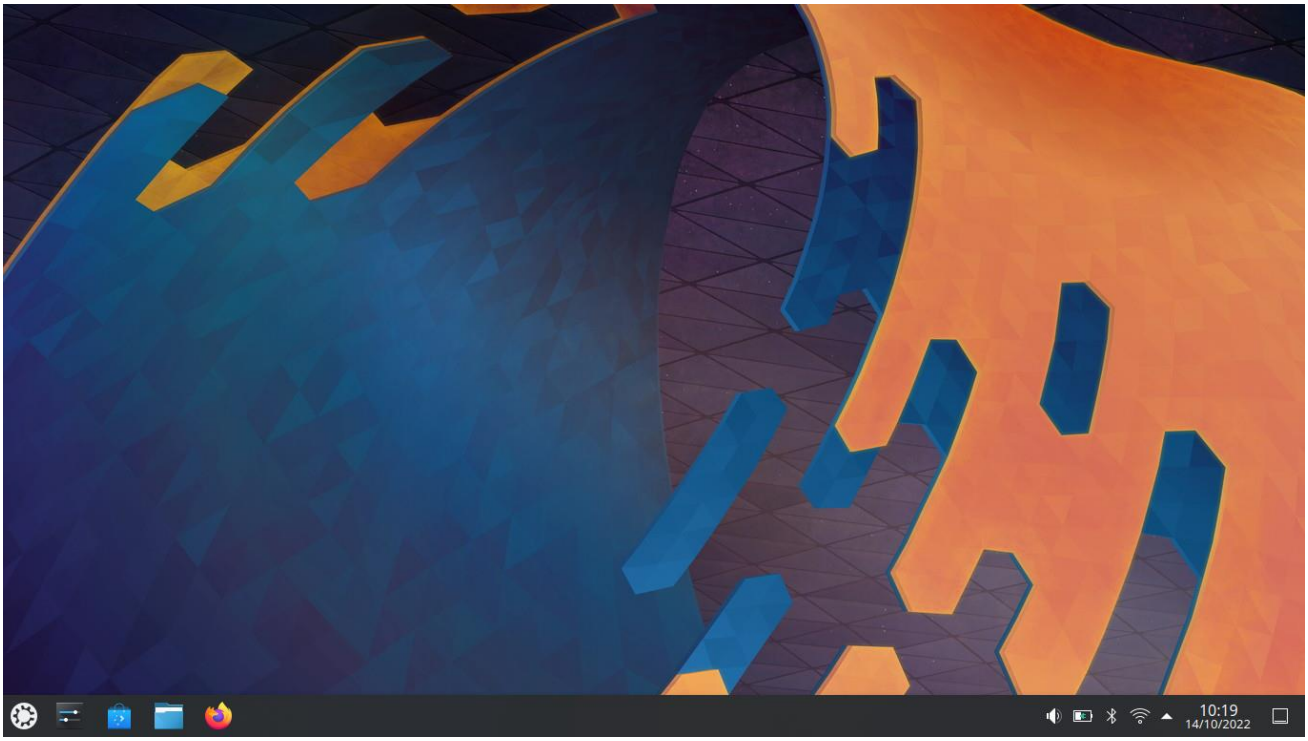
- *We won't do any exercises for this session, but you are free to use this time to ask any specific questions about your own personal setup.*

2 Overview of Linux

Here are some things to try to help you get to know Ubuntu and the KDE Desktop.

2.1 Desktop layout

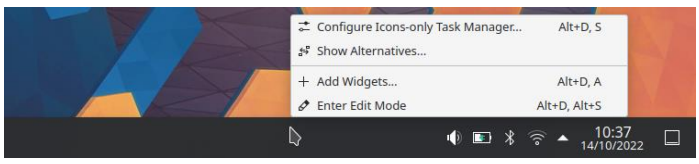
When you first open Kubuntu (Ubuntu with the KDE Desktop) on a fresh installation, you should see a screen that looks like this (Note this is for Version 22.04; future versions may look a bit different but will broadly feature the same layout).



If you've used Windows before, this layout should be very familiar. On the left (the gear icon, which is Kubuntu's logo) you have the button to launch the start menu, where you can find all programs, settings, recent documents, and so on. Next to the menu you have several icons for applications that have already been added to the taskbar (you can add more just like you would on Windows): the settings application, software centre / update manager, file manager, and web browser (Firefox).

On the right you have all the important status icons, including volume control, battery monitor, Bluetooth, network manager (Wifi selector and settings), date/time, and "show desktop" (minimize all windows). New icons for particular applications and notifications will also appear here, and there are more control icons in the system tray if you click the arrow next to the date.

This entire taskbar can be configured, including what items and tools it contains, where it is placed on the screen (bottom, top, left, right), its size, color/transparency, and many more. To edit the taskbar, right click on it and select "Enter Edit mode". You can also directly add items/tools by clicking "+ Add Widgets".

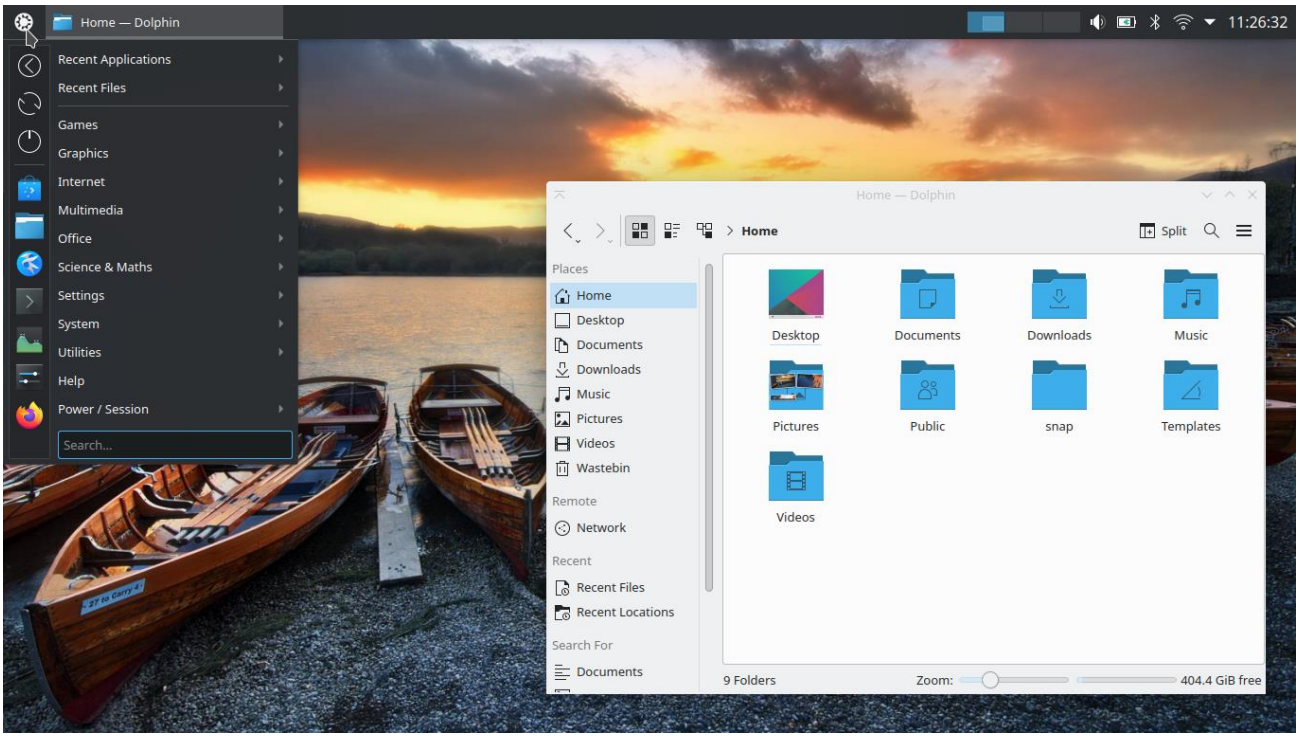


Exercise 1: Make some changes to the taskbar. Try moving it to a new position with a new style and new widgets.

Other basic layout configurations you might want to try are:

- Changing the background (just right-click on the desktop and select “Configure Desktop and Wallpaper”)
- Changing the taskbar position (after right-clicking to edit, you can drag it where you want)
- Workspaces: You can have different virtual desktops where you can place different sets of applications you are currently working on (this is a way of compartmentalizing work). Just search “workspaces” or “virtual desktops” in the menu and you should find the setting.
- Change the time display (I usually just like to see the time, not the date)
- Change the application menu type (there are 3 different types; if you hover over the menu icon while in edit mode, you should see a button for “Show alternatives”)
- You can also change the icon taskbar to one that shows information about the opened windows, as shown below (again, do this by hovering over the taskbar in edit mode and selecting “Show alternatives”)
- You can change several aspects of the theme by going into the Settings app and selecting “Appearance” (you can also just search “theme” in the menu, and it should turn up several options)
    - For theming, you can select global themes, but also change specific components of the theme, such as the colors of the taskbar, windows, and accents, the buttons that appear at the top of a window, and much more.
- Get rid of that bouncing cursor, which is always turned on by default (at this point it’s a joke in the KDE community, because people have been asking for the default to change for years and they just keep preserving it for fun). You can experiment around with where to find this setting, but as in most cases the search is your friend.

Exercise 2: Make more extensive changes to the layout. If you want, you can try to make your desktop look like this (this is generally how I like to configure mine). Or you can do something completely different. That’s the beauty of Linux, and in particular the KDE desktop: most things are configurable to how you like them and what best suits your workflow.

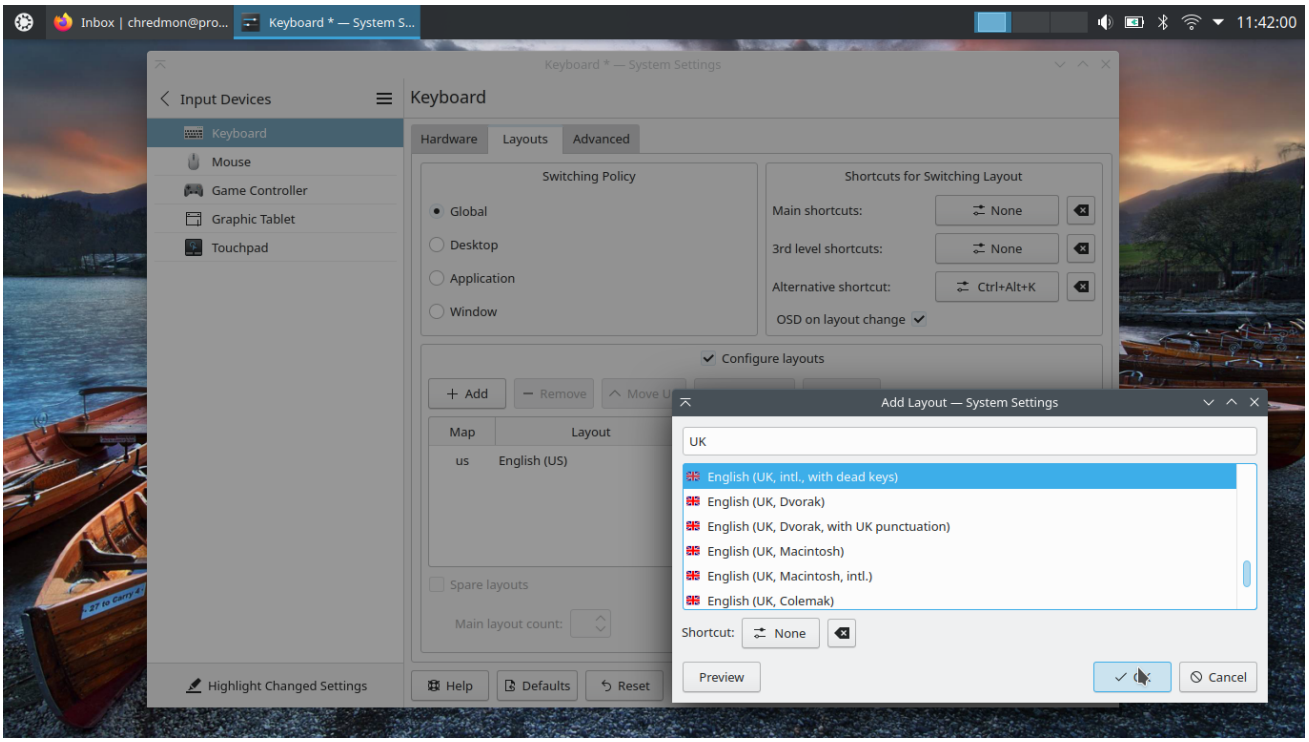


2.2 General settings

There are a number of other settings beyond the basic look and feel that you may want to change. All of these can be found in the “System Settings” application (search “settings” in the menu). This application can be a bit overwhelming for some because there are so many options and it’s not always clear which setting will be in which submenu. But here KDE has made things easy for us and all settings can be easily searched for. You can also find extensive documentation online in KDE or Ubuntu forums (a Google search will work in most cases).

Changing the keyboard layout

The keyboard is currently set up with the US-style layout. To set it to a UK layout, find the Keyboard settings (search “keyboard” in System Settings), then select the “Layout” tab, check the “Configure layouts” box, and then click “+Add” to add the UK layout. This is also what you will use if you want to add keyboard layouts for any other languages.



Network settings

You may also need to change your network settings. You can do this through the Wifi icon in the taskbar, or by going to “Connections” under “Network” in the System Settings. Here you will be able to select a different network, add a new one, configure passwords, and so on.

User information

You can change user information (name, user ID, privileges) and add users through the “Users” “Notifications” section in System Settings. This is also an important area to understand in case you find yourself unable to make certain changes to the system (indicating that you may not have an administrator account).

Other settings

Other items you may want to check out on your own are configuring notifications, shortcuts, accessibility, backups, and startup/shutdown.

3 Programs/applications

This section will get you acquainted with a number of common programs/applications on Linux, as well as covering how to install new programs, update software, and use programs that were not made for Linux but can still be accessed from within Linux.

3.1 Basic Linux applications

Here's a list of some of the default applications in Kubuntu. Note that each Linux distribution will have a different set of applications that come by default, but most systems should have some variation on the ones listed below. Each of these applications can be found in the menu. And don't be confused or discouraged by all the different names. These programs have different non-intuitive names because there is a wealth of software in the Linux ecosystem, so calling a program simply "Calculator" wouldn't distinguish between the tens of different calculator applications available on Linux. Whatever distribution you're using, you'll get used to the names of each program.

Core

- **Dolphin:** file manager
- **Konsole:** terminal

Accessories/Utilities

- **KCalc:** basic calculator
- **Kate:** feature-rich text editor (useful for both basic text editing and more advanced programming)
- **Spectacle:** screenshot tool
- **KDE Connect:** a way of sharing files between devices (similar to AirDrop on Mac)

Internet

- **Firefox:** web browser
- **Thunderbird:** email client
- **KTorrent:** torrent file download tool

Media

- **Elisa:** music player
- **VLC:** video player
- **Gwenview:** photo browser and basic editor

Office

- **LibreOffice:** full open-source office suite (MSOffice replacement)
- **Okular:** PDF reader

System

- **System Monitor:** monitor system resources (CPU, Memory, Network)
- **KDE Partition Manager:** configure hard drive (create/format partitions, external disks)
- **Discover:** software center
- **Muon Package Manager:** alternative software manager (more geared at administrators)

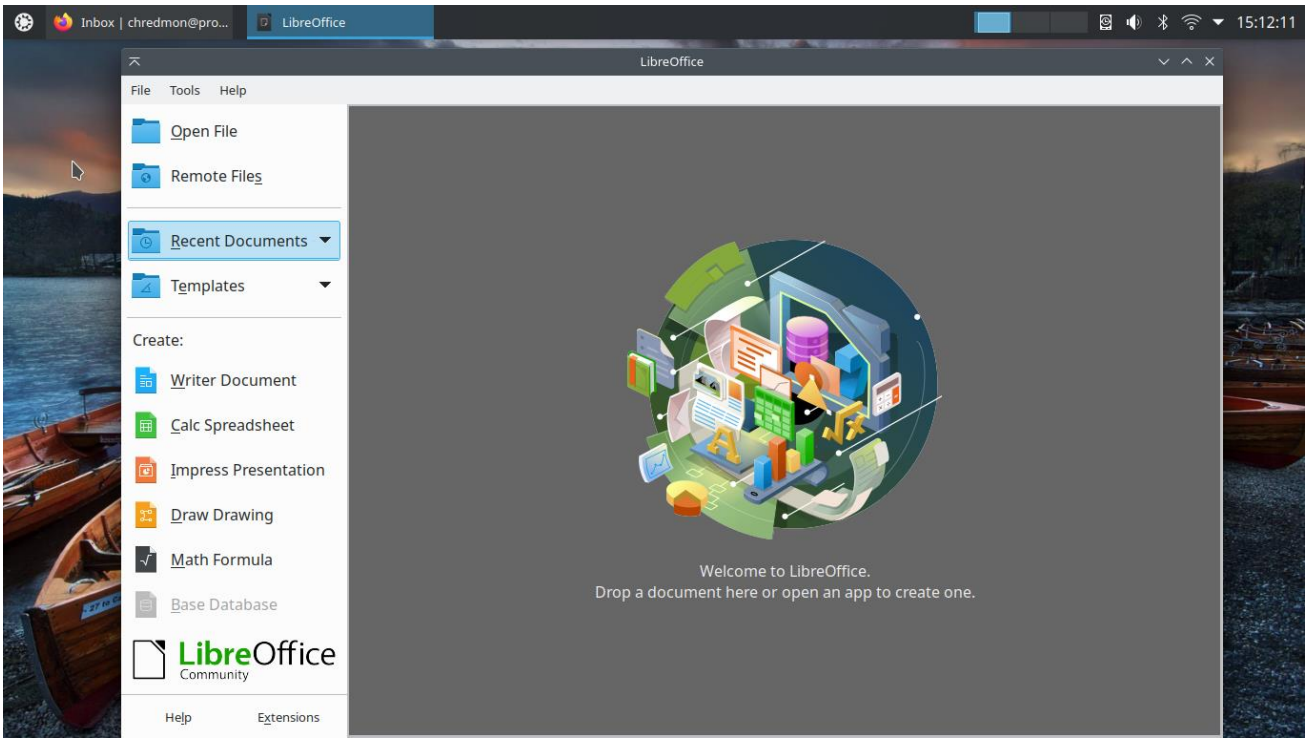
Exercise 3: Take a screenshot and save it to the Desktop. We will use this file later for exercises in the terminal.

3.2 Office applications

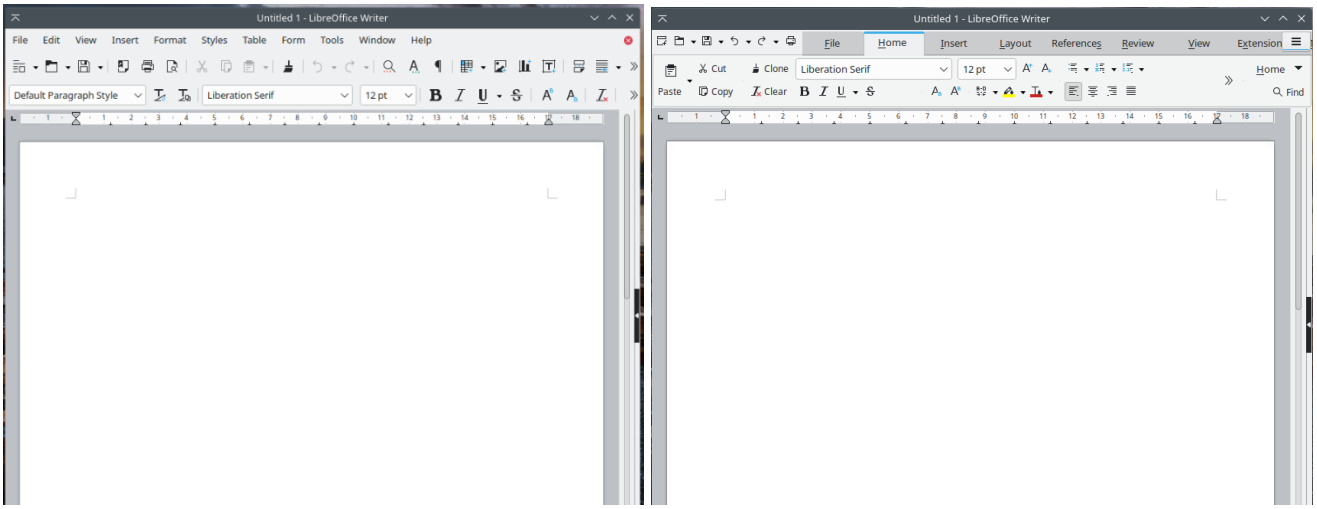
Here we'll go into a bit more detail on office applications. The primary application used in place of MSOffice on Linux is the open-source office suite: LibreOffice. Below you can see a picture of the opening page for the unified application, but there are separate links in the main application menu

for each individual component. The mapping between LibreOffice applications and MSOffice is as follows:

- Writer ←→ Word
- Calc ←→ Excel
- Impress ←→ Powerpoint
- Draw (a drawing/diagramming tool; not in MSOffice)
- Math (an equation editor; not in MSOffice)



The core MSOffice replacements (Writer, Calc, and Impress) have nearly all of the functionality of Word, Excel, and Powerpoint, and can import and export .docx/.xlsx/.pptx files as well. They are only missing a few of the most recent/advanced features, and are constantly working to keep pace with Microsoft. The Impress-Powerpoint relationship is probably the weakest of the three (advanced Excel operations may also be difficult, though LibreOffice has its own set of functions and macros). But one advantage of these programs is they also are great about backwards compatibility. So if you got really comfortable with the Word interface from the XP era, you can replicate that in Writer, but also if you like the newer 365-style layout you can implement that interface as well (Writer offers several different options).



Exercise 4: Create 3 documents: a Writer (Word) document, a spreadsheet (Calc), and a presentation (Impress), and save them in both native (Open Document) and Microsoft (.docx/.xlsx/.pptx) formats. You can also try out the Draw and Math applications if you have time.

3.3 Program installation

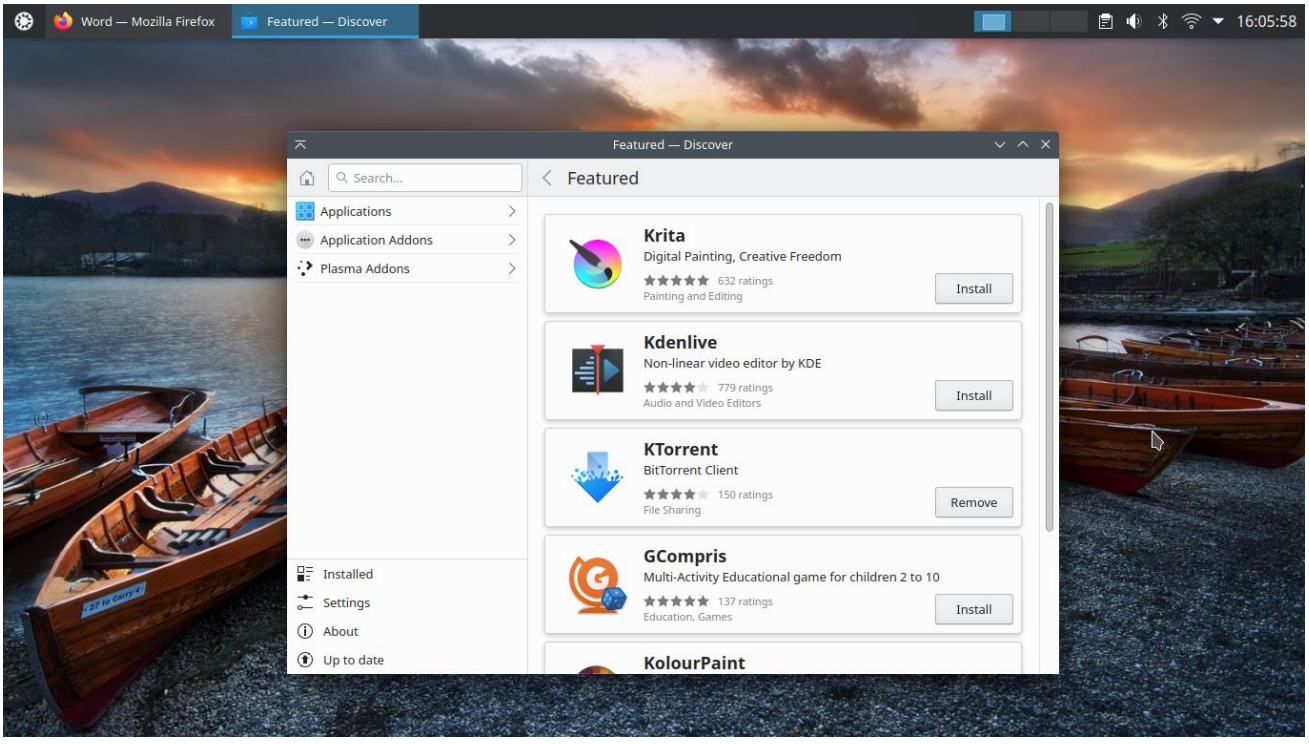
The default applications are enough to cover a fair bit of use cases, but of course you will want to install new programs as well. There are several ways to do this and we will walk you through each.

These days with app stores for phones and tablets, all major operating systems have a program you can install software from, but not too long ago the standard way to install programs on Windows or Mac was to browse the web, download an installer file (exe/dmg), and then run that file to install the program. Linux, by comparison, has always managed software through distro-specific repositories. This means that most software can be installed from within the system without ever having to visit a website (you can do this through a graphical interface or in the terminal). But you can also download installers from websites (on Ubuntu these will be .deb files).

Finally, a recent development in the Linux ecosystem has been the creation of a few new program formats: Snap, Flatpak, and AppImage. You don't need to know what these are at the moment but these were created to make it easier for developers to package software for different distributions. Ubuntu primarily uses Snap packages when not using software from its own repositories (the .deb format), and you will see in the software center that sometimes you have 2 different options for the installation. In general, we recommend going with what is already in the repository (the default), but sometimes you may need to install a different version such as the Snap version if that version is more recent or if it's the only one available.

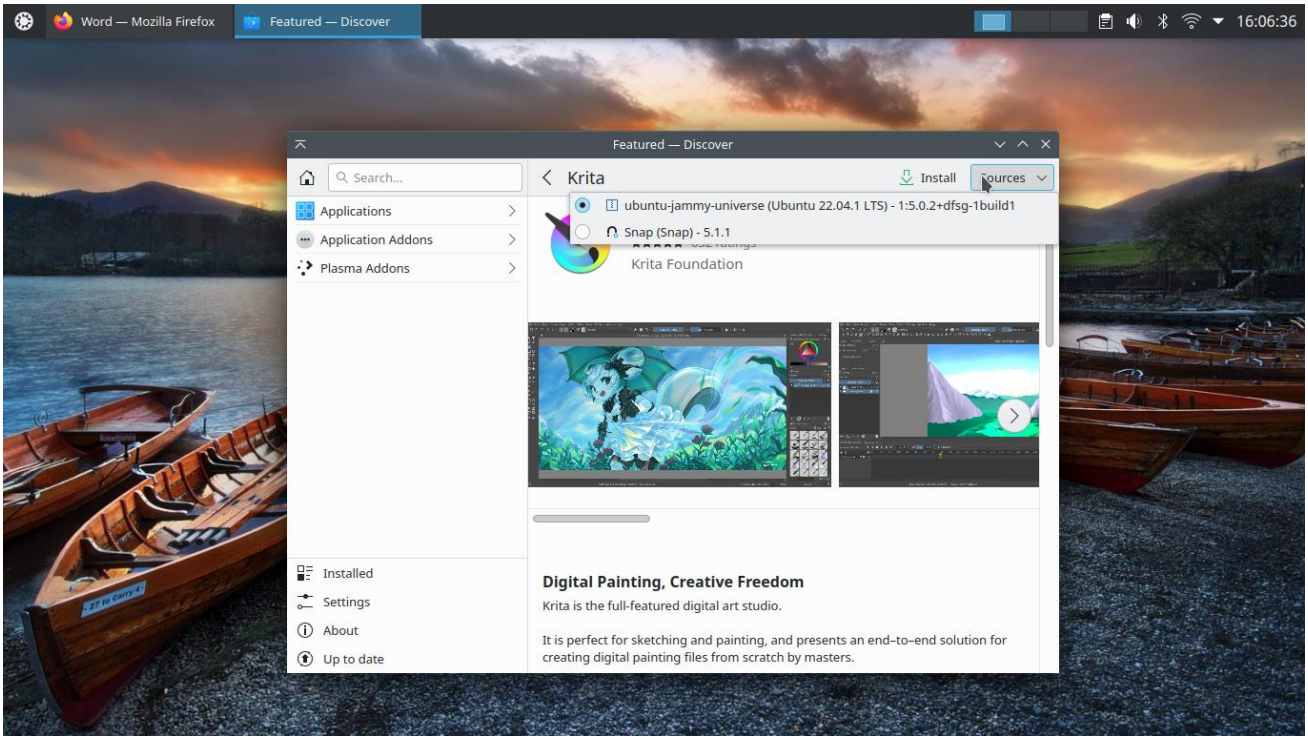
Using the software center

The simplest way to install software is through the software center, which is called "Discover" on KDE. This is also where you can check for and apply software updates.



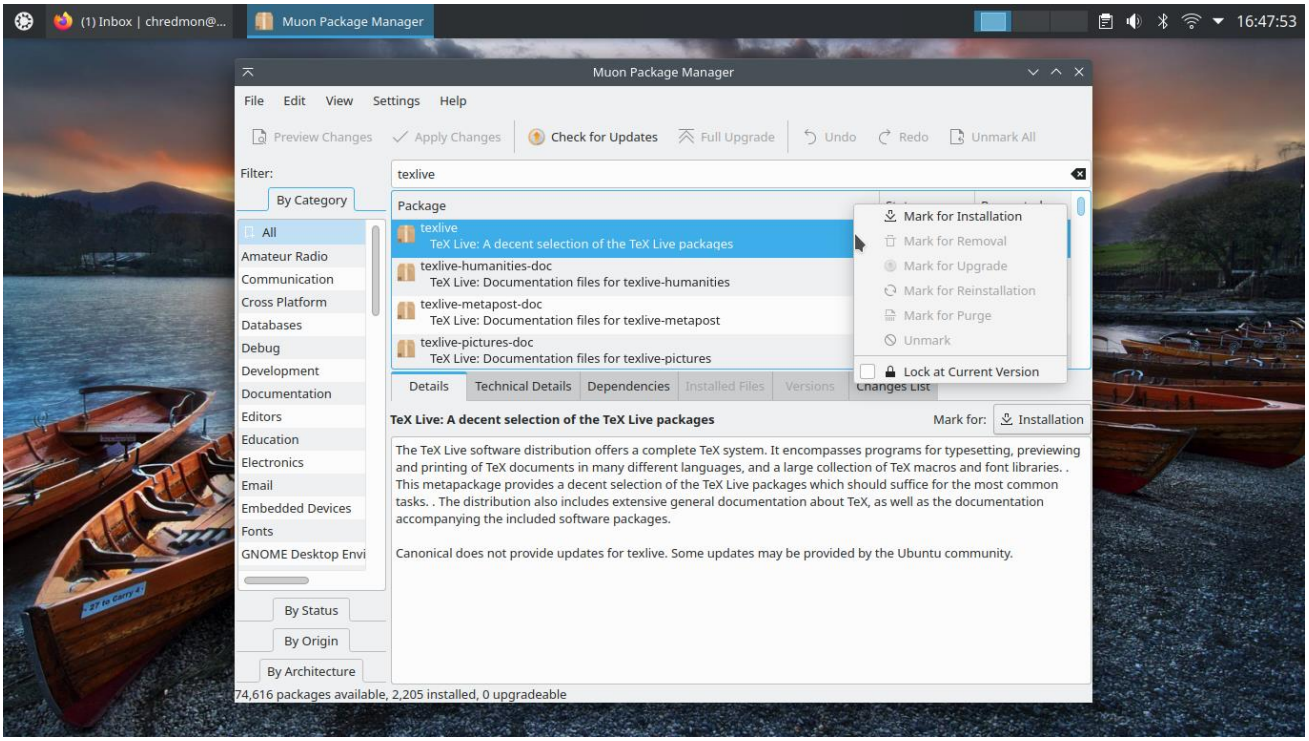
Here if we can search for applications, browse by category, and install applications directly by clicking the "Install" button. If you want to be sure where you are installing from, click on the

application first (e.g., Krita), and then if there are multiple versions you can click “Sources” to see which one is presently selected. Below we can see that it has selected the repository version by default (“ubuntu-jammy-universe (Ubuntu 22.04.1 LTS)”), but you can also click on the Snap version to install that instead. Here we can also see that the snap version is slightly newer (5.1.1) than the repository version (5.0.2). These differences are probably not meaningful but there may be some cases where the versions are very different, or where particular features are not available in one of the versions so you want to install the other.



Using the graphical package manager

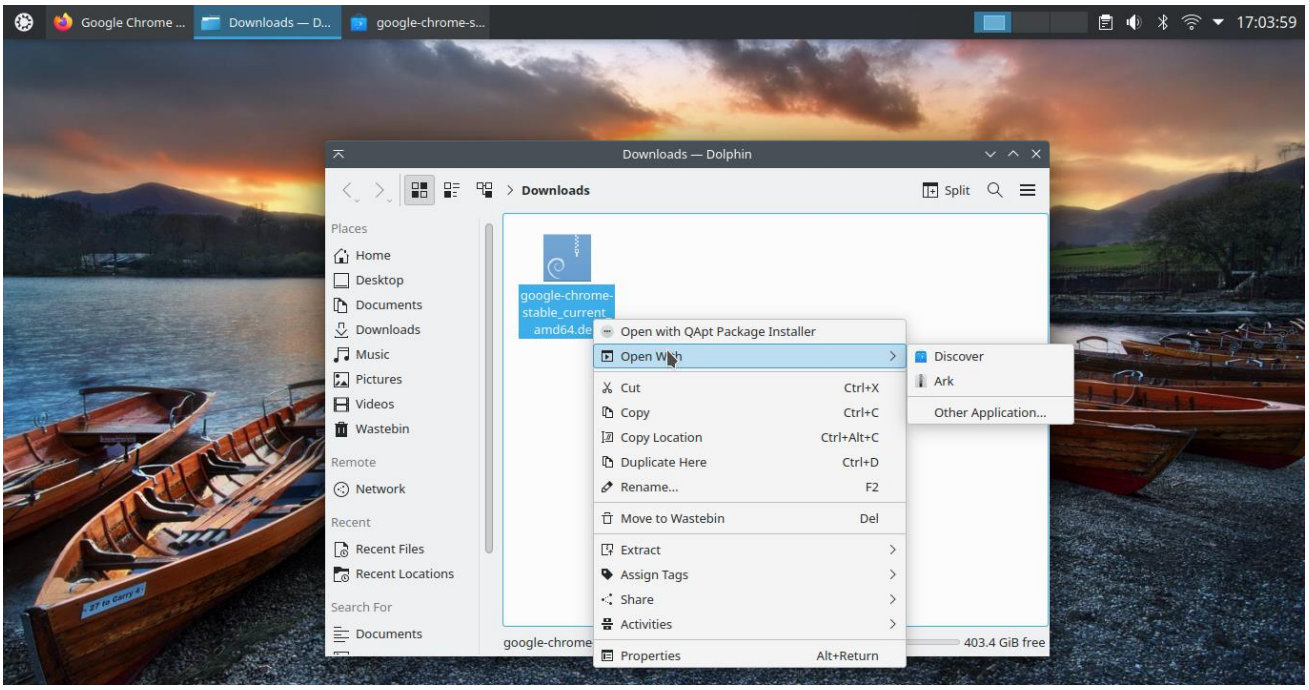
Sometimes you need a specific package that is not in the software center (often these are back-end utilities). In KDE the commonly used program for this purpose is called Muon. Here we are installing “texlive”, a package that implements the LaTeX typesetting system.

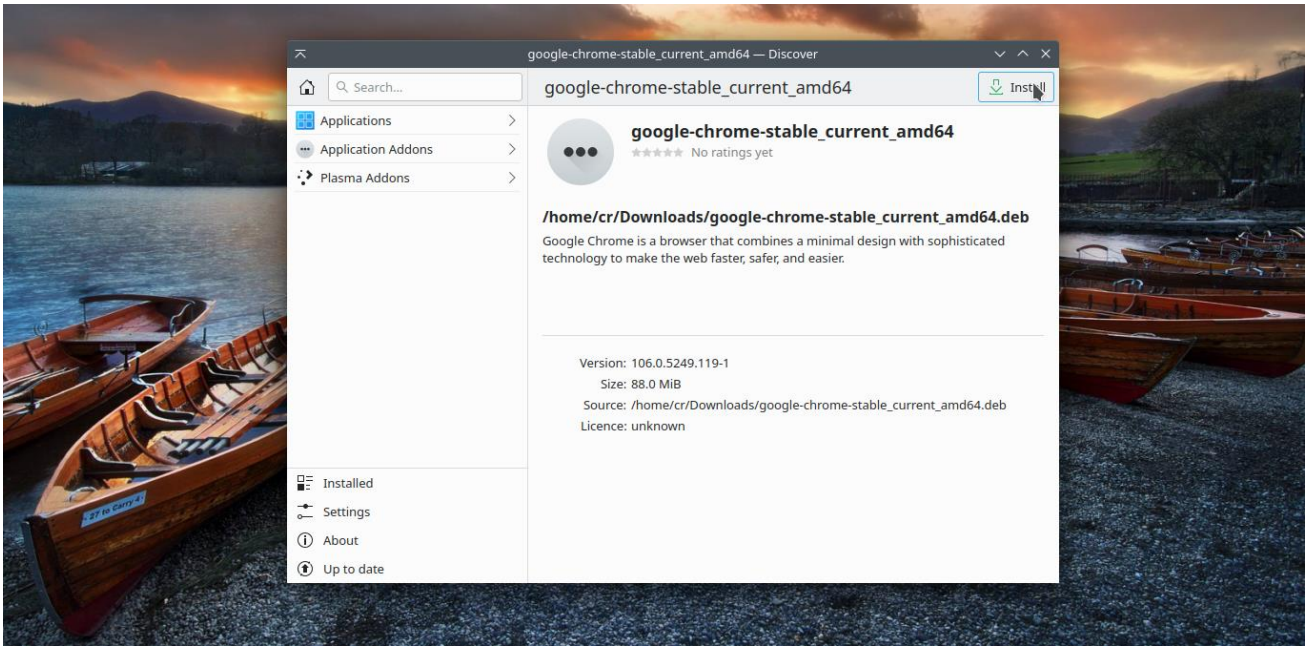


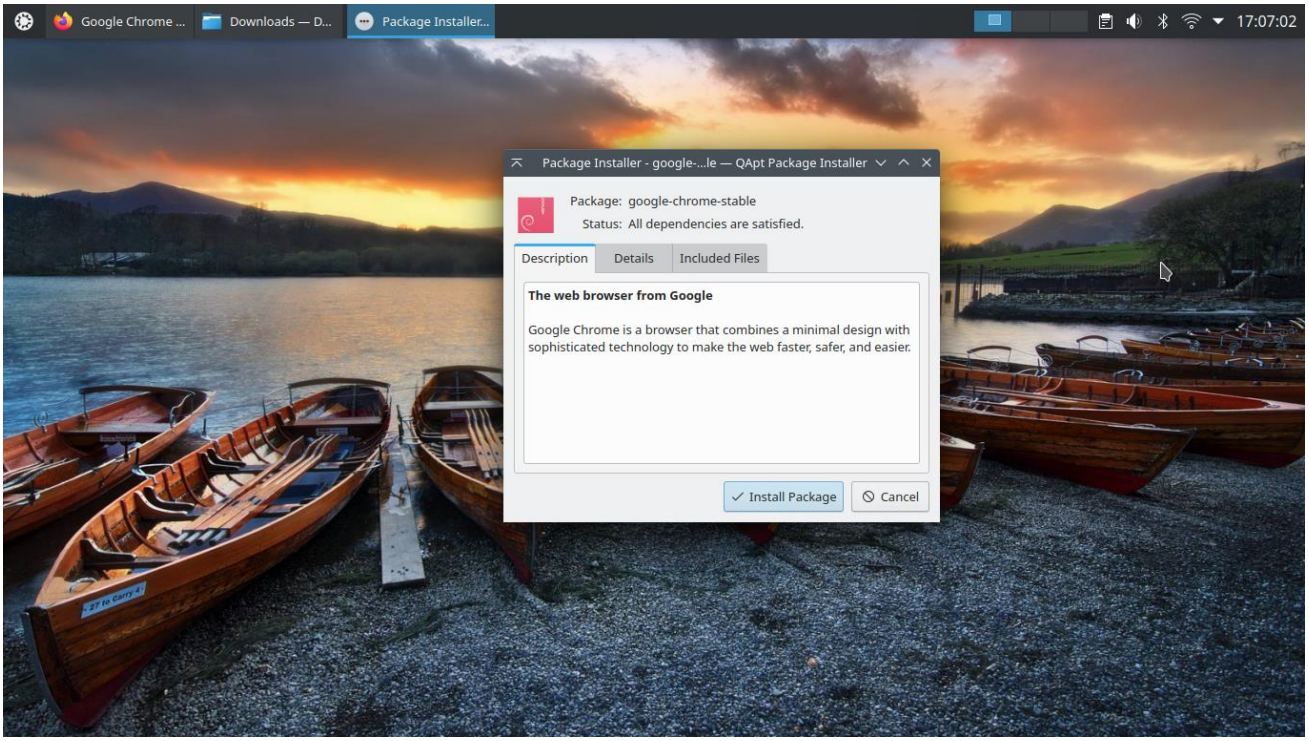
Installing software downloaded from the web

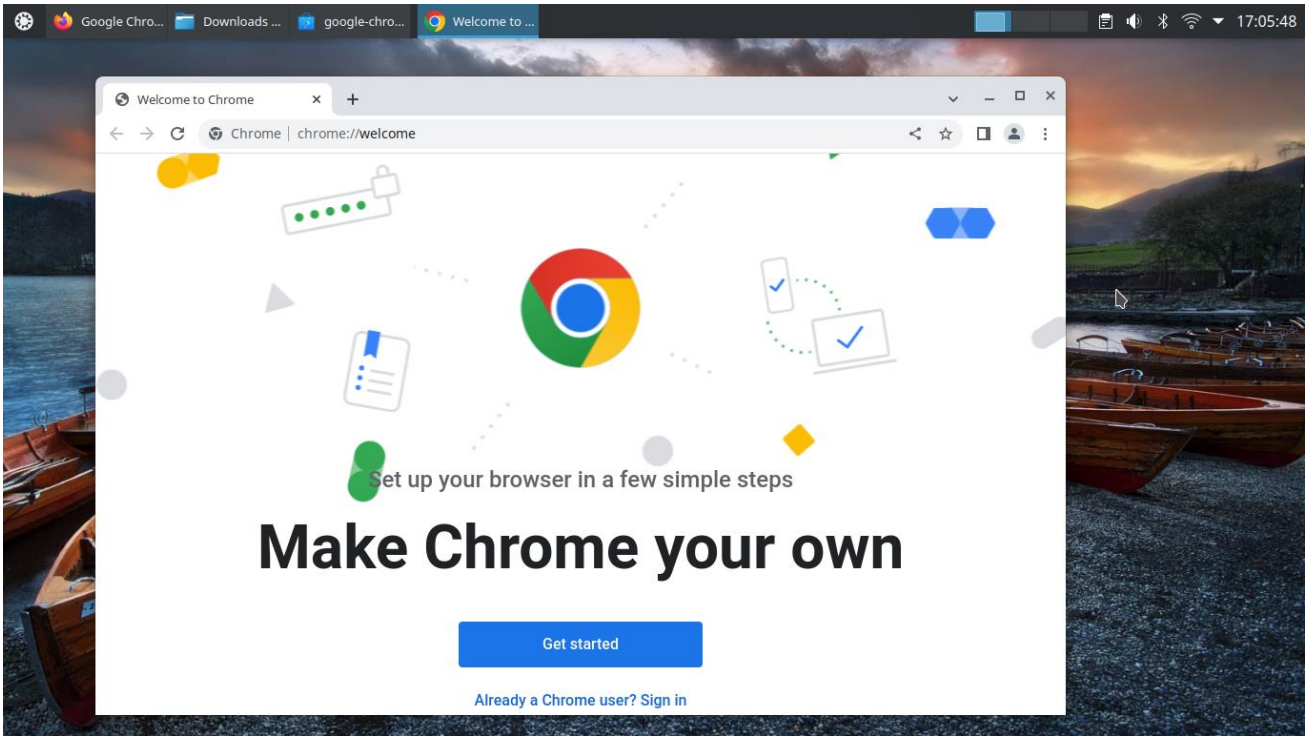
Sometimes you may need software that is only distributed on the developer/company's website. These will usually be ".deb" packages, but they could be Snap packages as well. As with anything downloaded from the internet, you should be careful doing this and make sure you can verify that the source is trustworthy. Linux doesn't generally have problems with viruses like Windows does (this is partly due to its own security protocols and partly just because it is not as popular and therefore not worth writing viruses for), but you should still be careful about installing random software from the web.

Here we have downloaded a .deb file for installing the Chrome browser. Right-clicking the file gives us a couple of options for opening it: the QApt Package Installer and Discover. Both work (you can also do this in the terminal but we'll introduce that later). QApt is a simpler installer just for these types of packages, while Discover is the full software installation and updating suite we introduced earlier. First we show what this looks like in Discover, then in QApt.









Installing Snap packages is also easy but works a bit differently. Usually you will be given a code to type in the terminal, such as the following:

```
sudo snap install audacity
```

You can try this now. It will ask you for your password (*sudo* is a command that elevates user privileges to that of an administrator) first before installing. Don't be worried if it doesn't show anything on the screen as you type. This is an additional security measure. We'll cover terminal commands later, including installing programs from Ubuntu's repository directly from the terminal.

Exercise 5: Install a few programs of your choice. Now would be a good time to test out installing some of the programs you often use and seeing what's available in Discover, what's in Muon, and what you need to find online.

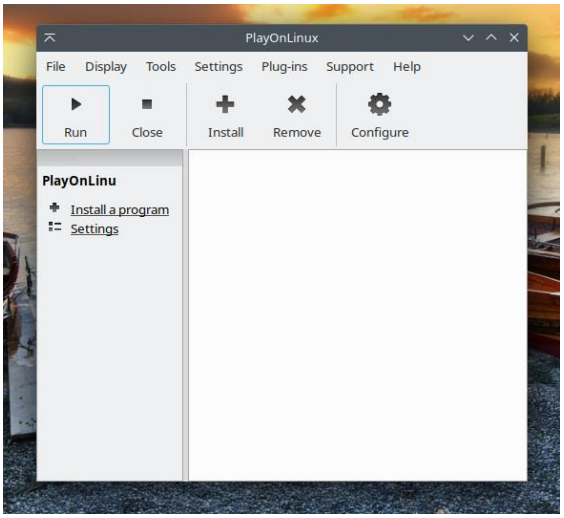
3.4 Working with proprietary programs

Probably the most frustrating thing in Linux, and the biggest barrier to entry, is working with programs that are proprietary and which the companies have not made available on Linux (even as paid versions). This is often done because the Linux user base is so small that companies may not feel it's worth their time developing and supporting software on it, though a lot of this is changing with new frameworks like Electron that allow cross-platform apps to be made much more easily by containing everything in a browser (this software is often much heavier and uses more resources, but it's better than not having any option at all).

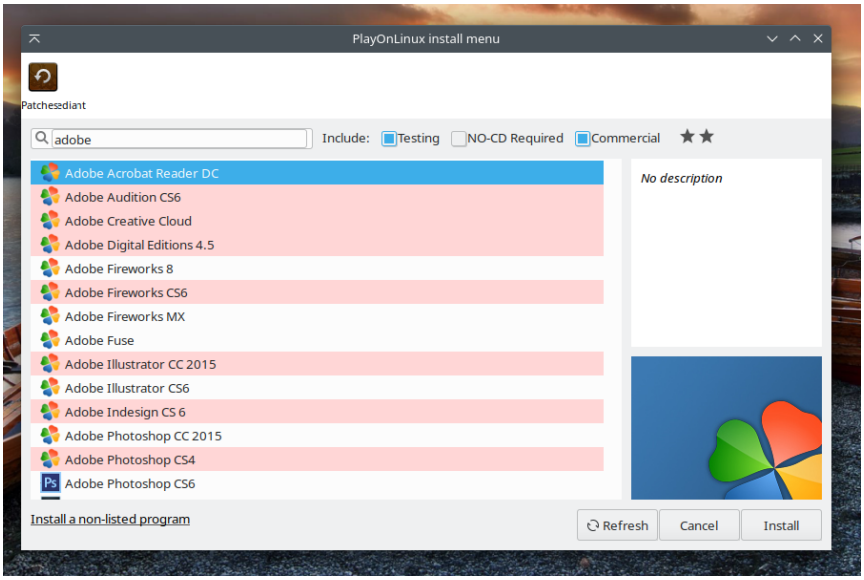
The most commonly used proprietary software that is not available on Linux is the Microsoft Office suite. We have already discussed alternatives to MSOffice, but sometimes for compatibility with colleagues you need to use MSOffice (of course, if everyone committed to using open-source software like LibreOffice there wouldn't be compatibility issues and all documents would be freely readable, but that's a fantasy for another time). Today the best solution to this problem is to use Office365 in a web browser. Alternatively, you can dual-boot Windows and Linux and switch over to Windows when you need to check/fix any compatibility issues. I still do most of my editing in LibreOffice and only jump to MSOffice for final editing.

Another common proprietary suite of software is the Adobe suite. This is more of a problem on Linux because Adobe doesn't currently have a complete online suite like Office does. For now if you are a heavy Adobe user, and can't switch to open-source alternatives to Photoshop (GIMP) or Illustrator (Inkscape), then I would suggest keeping a Windows/Mac partition on your computer for this work. For smaller Adobe applications like Adobe reader, it is possible to use the solution below. This is not needed for general PDF usage, because Okular is a very powerful PDF reader, but if you are given a document with Adobe-specific forms to fill out then you may need to switch to Adobe.

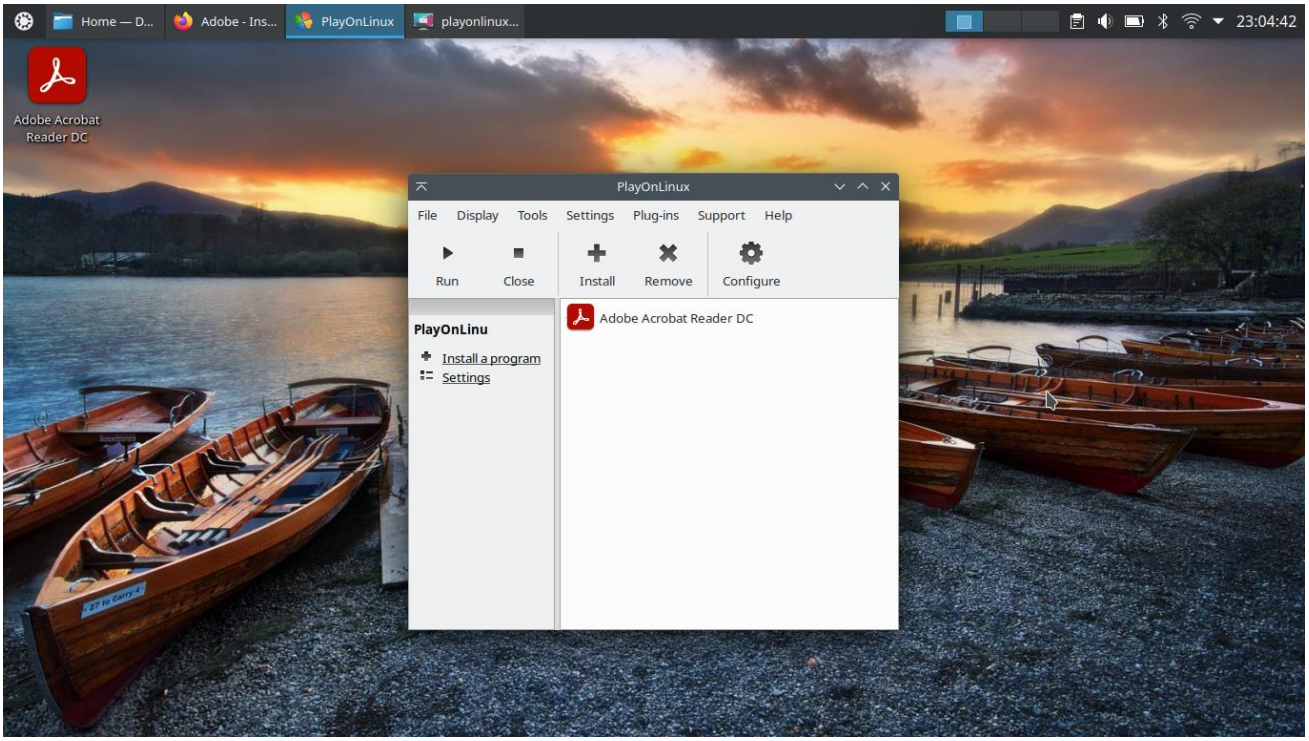
MSOffice and Adobe are the two biggest players in this space, and unfortunately they still remain a major barrier to entry for Linux. However, if you want to use other Windows software there are a couple of options to consider. The first is to run Windows in a virtual environment (we will discuss this in the final section of the course). The second and simpler way, though more variable in its performance, is to use Wine/PlayOnLinux. Wine is a Windows compatibility layer for Linux, and PlayOnLinux is a friendly user interface to use Wine. You can install PlayOnLinux directly from the software center. After installing, it looks like this:



We can then install a program by clicking “Install a program” in the sidebar and following the instructions.



If we want to install Adobe Reader DC we select it in the options, which helps configure Wine for the installation. Later you’ll be prompted to select the .exe file you downloaded from Adobe, and after clicking through the installation guide Adobe Reader will be installed on your system and you’ll get both a link within PlayOnLinux and a shortcut on the Desktop.



Exercise 6: Try installing PlayOnLinux and downloading and installing a Windows (.exe) program. Depending on the program this will either work perfectly or be very buggy. It also depends on how recent the program is. Older programs for Windows XP (especially games) tend to work the best.

4 The file system

Understanding the Linux file system may not seem like something necessary for using Linux, and by and large you can get by without knowing much about the structure, but a few key ideas will go a long way and toward helping you get the most out of your system.

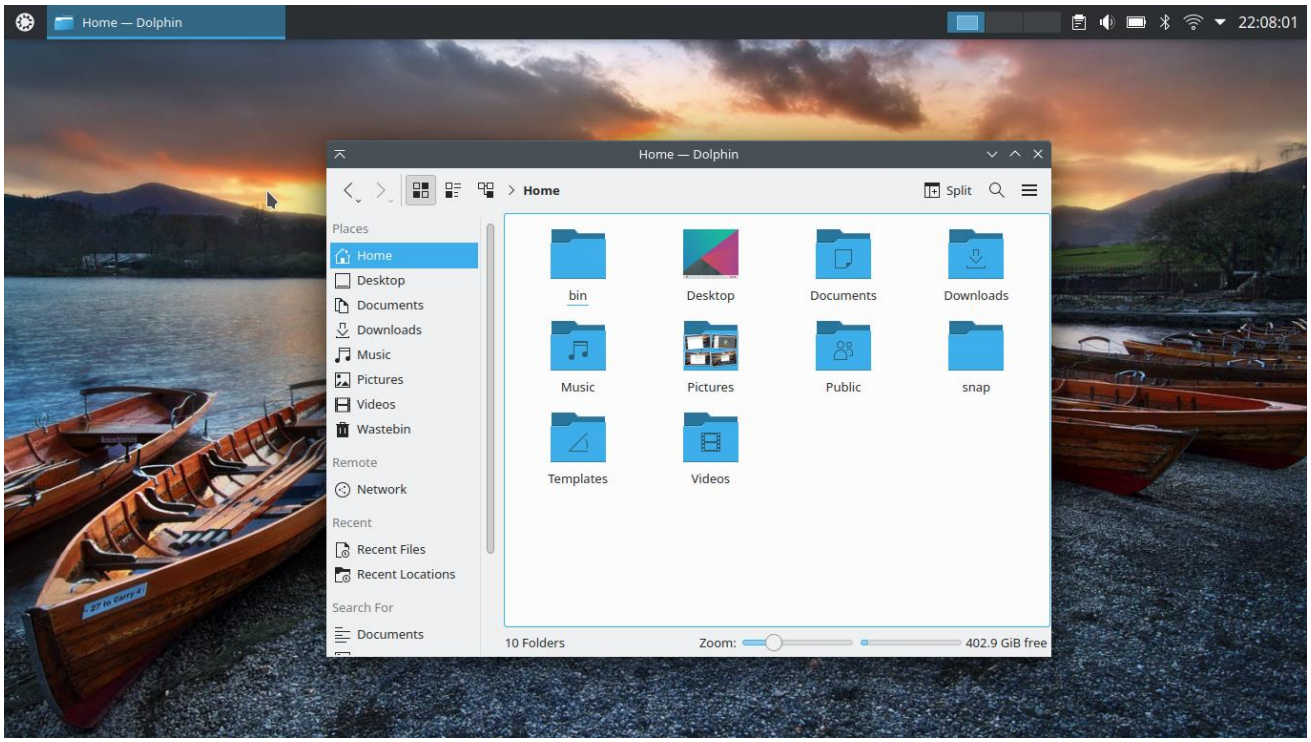
4.1 The structure of the file system

All files and directories on Linux are organized into a single tree. At the very top, housing all other directories, is the "root" directory, symbolized with the forward slash: "/". This means that all subdirectories will be pre-pended with a forward slash (e.g., "/etc", "/home", "/media").

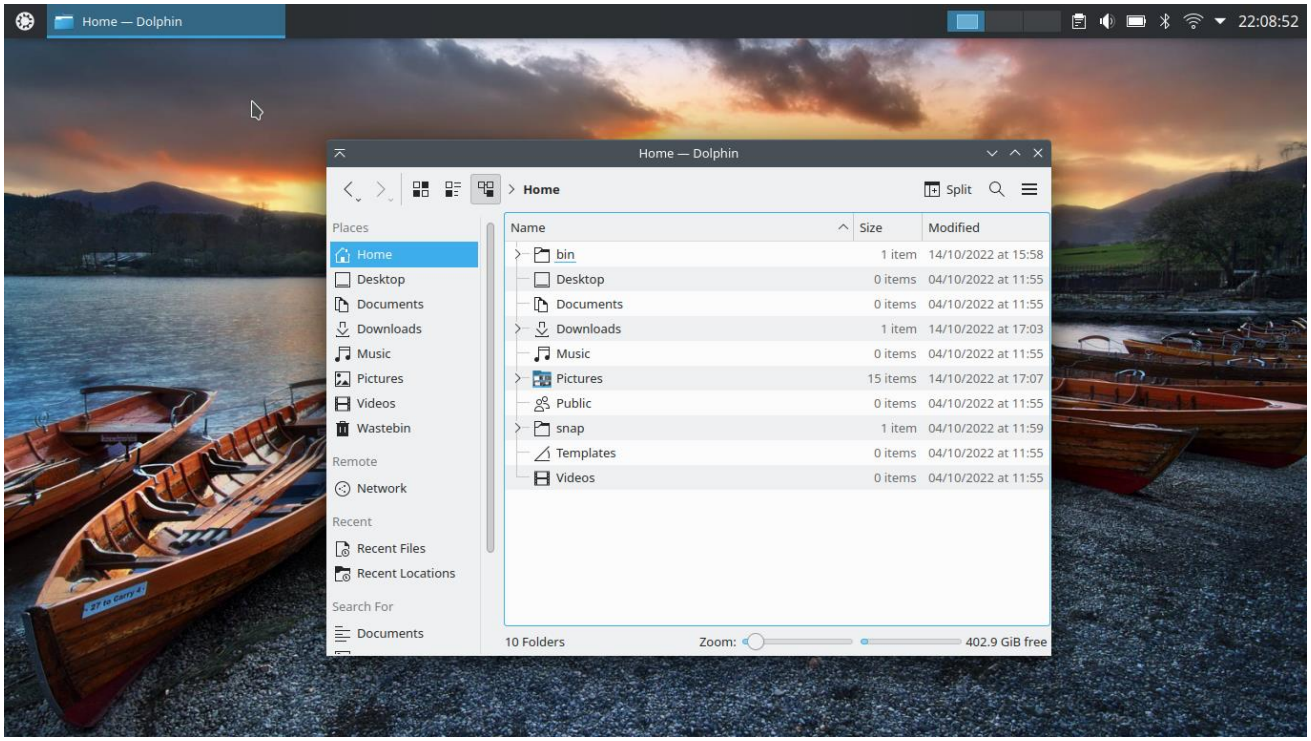
Most of your work will likely be in the home directory, which is located in "/home/[username]" (e.g., if your username is "newuser" your home directory will be in "/home/newuser").

This information is not so important at the start, but will come up more in the next section when we start our work in the terminal.

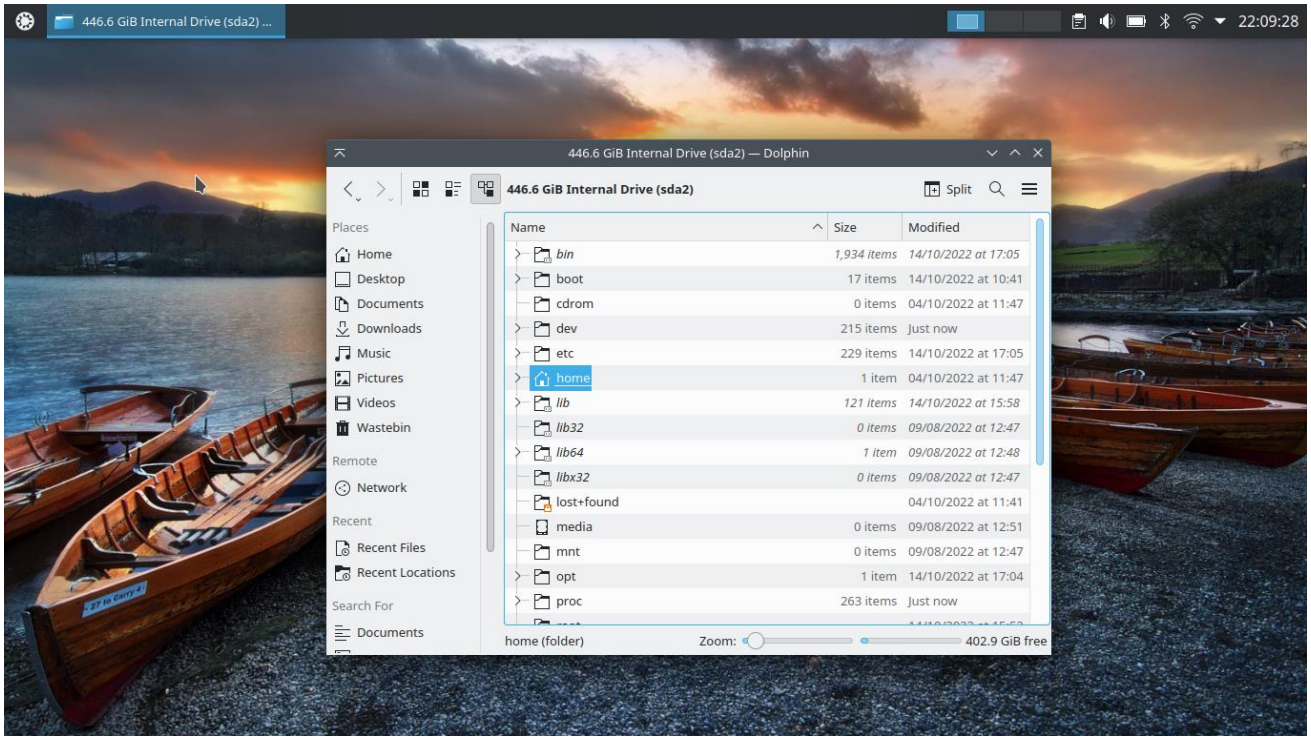
For now, open up the file manager ("Dolphin" on KDE), and you should see something like the following.



As with everything we've shown so far, the file manager is highly configurable. I don't like these big icons so I generally shift to a layout like the one below that shows the tree and files/directories in consecutive lines. This also provides information on file size, date of creation, etc. Again, this should all be familiar from Windows or Mac.



If you now move up a couple of levels in the tree (click the arrow next to "Home" and you should see a hierarchy you can click through), you will be in the root directory, shown below.

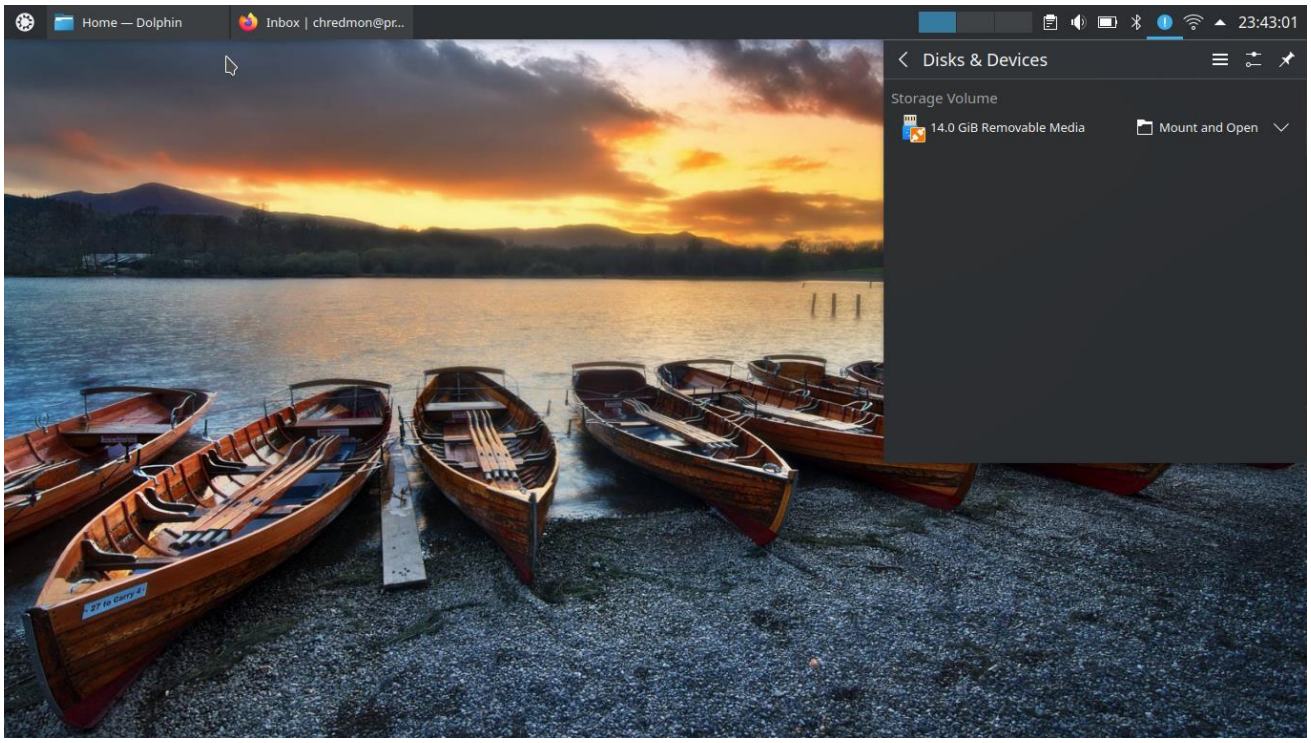


Most of these directories you won't ever need to look through, as they mostly contain program and system files. These all require administrator privileges to access, but you can access anything on the system that you want. This is Linux after all: nothing is locked-down and inaccessible to the user. But be careful working with system files because you could mess up your system if you don't know what you're doing.

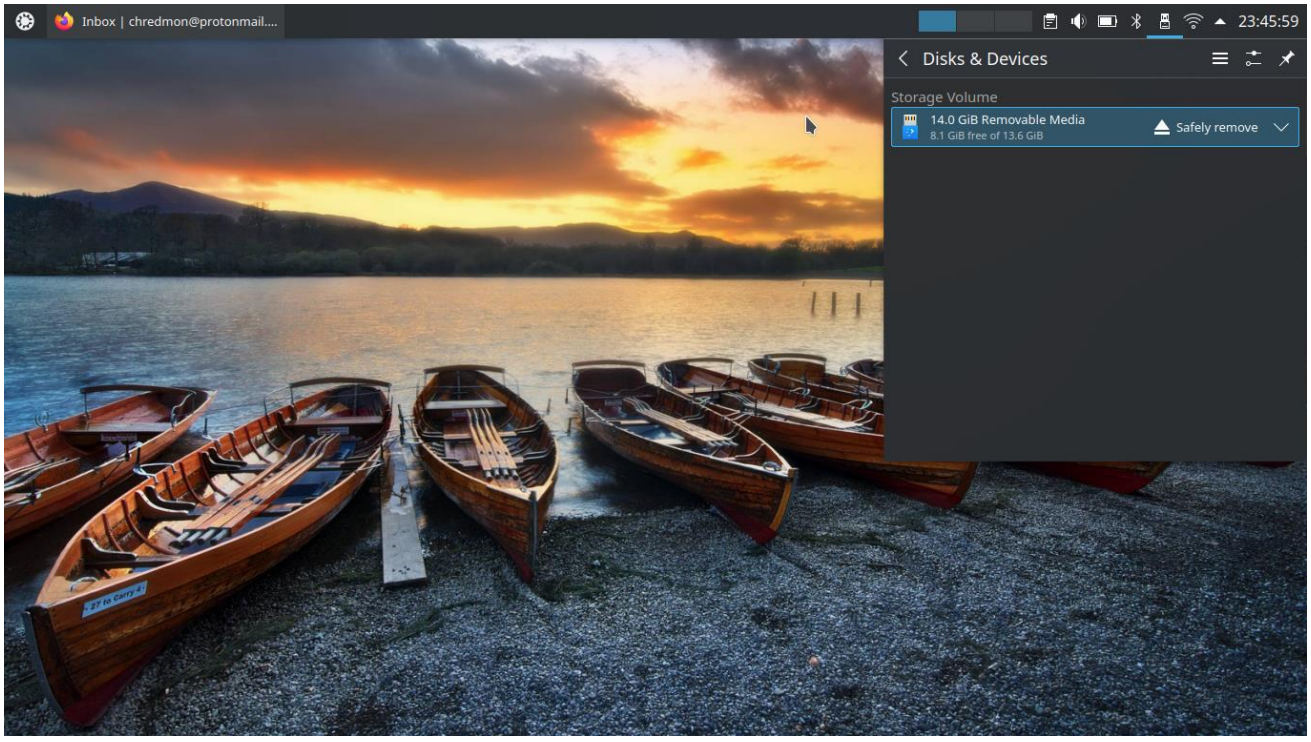
Exercise 7: Reconfigure the file manager and explore the different tools and settings.

4.2 External drives

Any external drive (USB flash drive, external hard disk) will show up in /media/[username] once it's plugged in and mounted. Just like on other operating systems, when you plug in an external drive you will be prompted with a notification and an option to mount and open the drive.



Once mounted and opened you should be able to access your external files from within the file manager or in the terminal from "/media/[username]/[name of drive]". To remove the drive just click the icon in the panel as shown below.

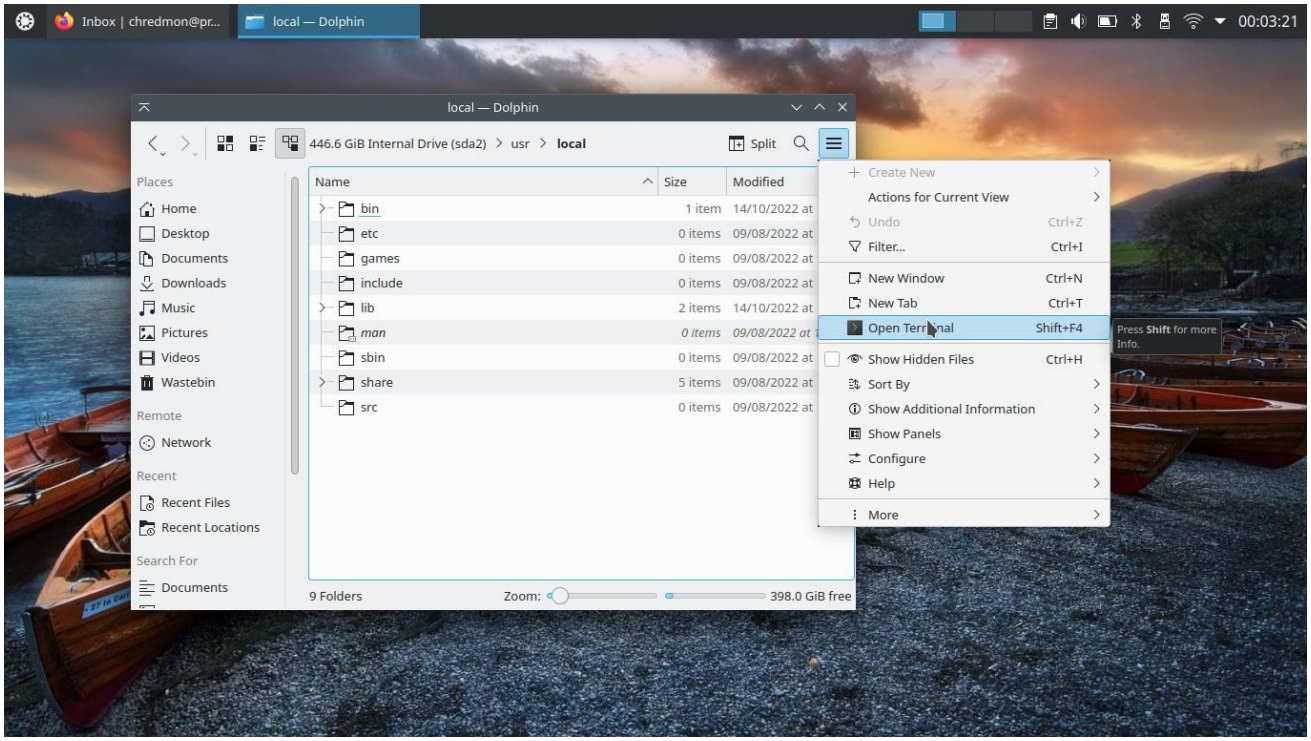


4.3 File/directory permissions

Files and directories on Linux have different access restrictions and thus require different permissions to open, read, write, and delete. If you are using Linux on your personal desktop you will likely only be working with two levels: your user account and the administrator account. Since you set up the system, you have access to both, but you will need to enter your password every time you wish to elevate yourself to administrator.

Everything in the home directory is accessible to you without elevated privileges, so you are free to create, delete, and edit files as much as you want. The only exception to this is if you want to run a script or program in the home directory that you haven't made "executable" before, but we will cover this later in the terminal operations section.

If you wish to open a folder in root you will have to do so through the terminal, as Dolphin doesn't allow administrator access to the program (this is for security purposes). In the example below we want to enter /usr/local through the terminal. We could do this directly by changing directories in the terminal, but we can also browse to the directory we want and then click "Open in terminal" from the menu.



Once you're in the terminal you can switch to administrator (with the *sudo su* command) and make any changes you wish, but we'll cover that in the next section.

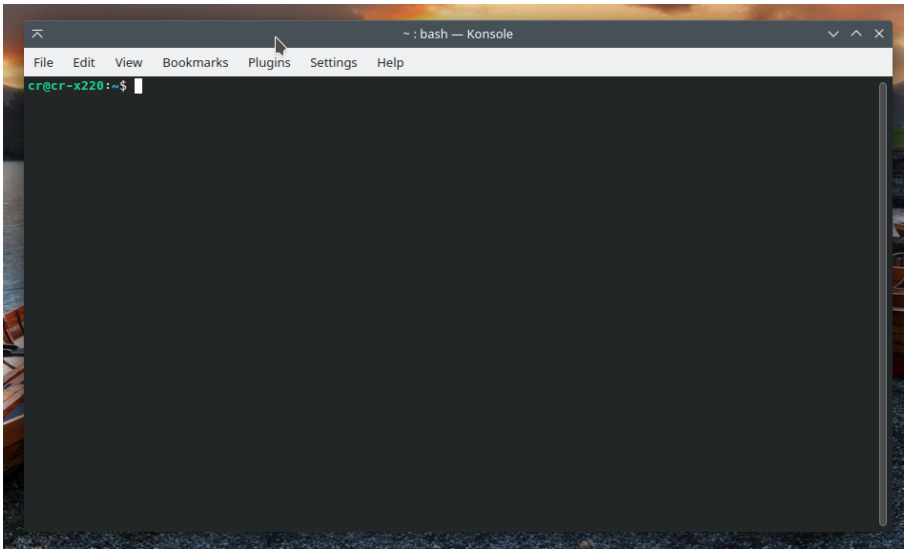
Exercise 8: Browse to a directory and open it in the terminal.

5 Basic command line operations

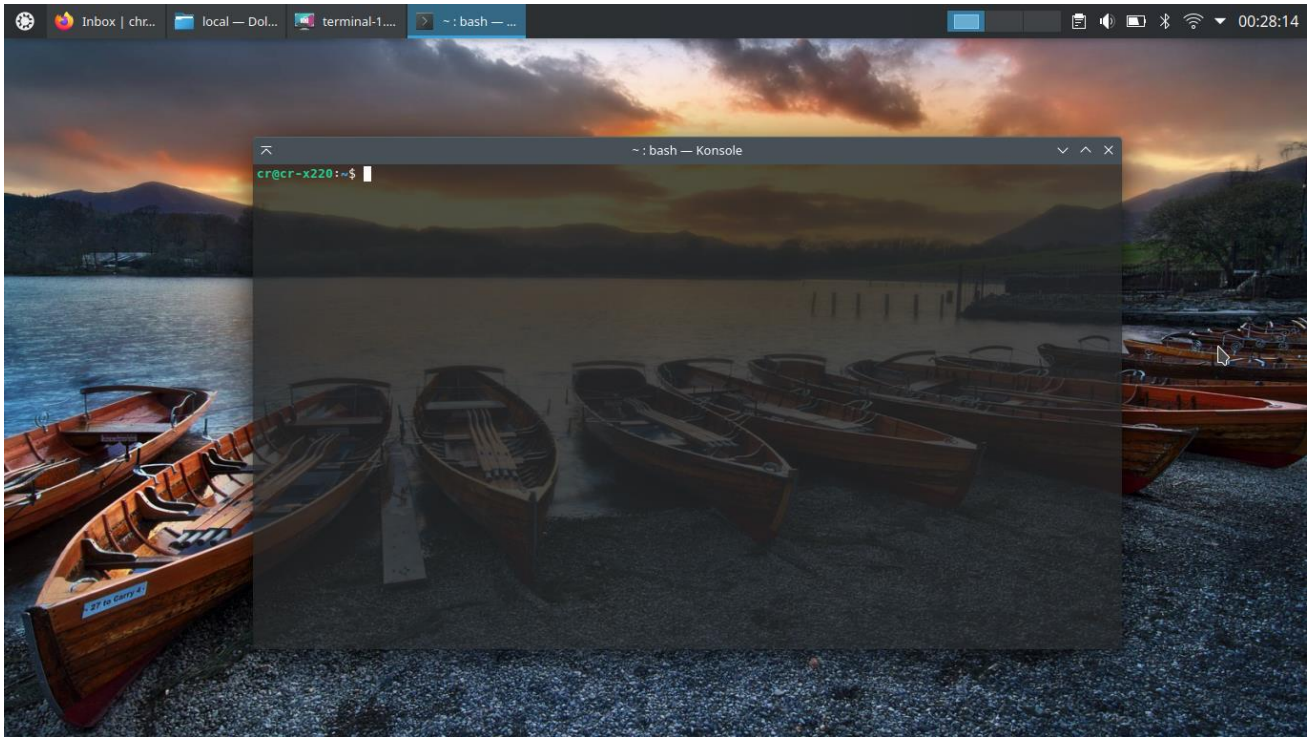
The terminal (command line) is a versatile tool that may be a bit intimidating at first but once you get familiar with the basics you'll be able to accomplish a lot while also understanding the operation of your system much better. To begin, let's open the terminal (you should be able to do this by now; just look for it in the menu like you would any other application). Alternatively, you can hit the key combination CTRL+ALT+T (this doesn't work on every Linux system but is a common shortcut that

works on Ubuntu). Finally, you can hit ALT+SPACE to open Krunner, a program launcher similar to Spotlight on Mac. From there you can search for “terminal” and find the Konsole application.

Once you have the terminal open you should see something like this:



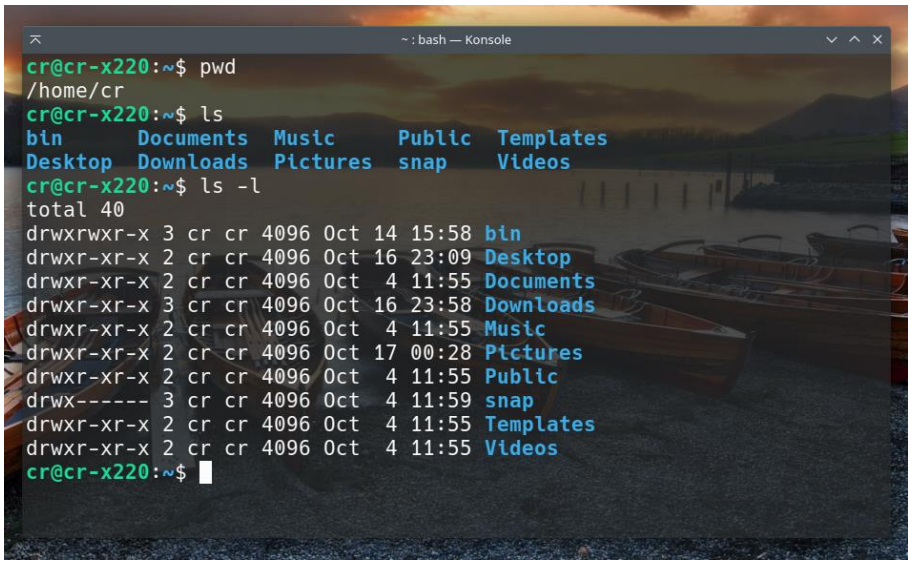
As always, you may want to modify the settings a bit first. I generally don’t like seeing the menu or scrollbar, and like some transparency to the background so that I can read information behind the terminal while I’m typing in commands (e.g., online documentation or StackExchange comments).



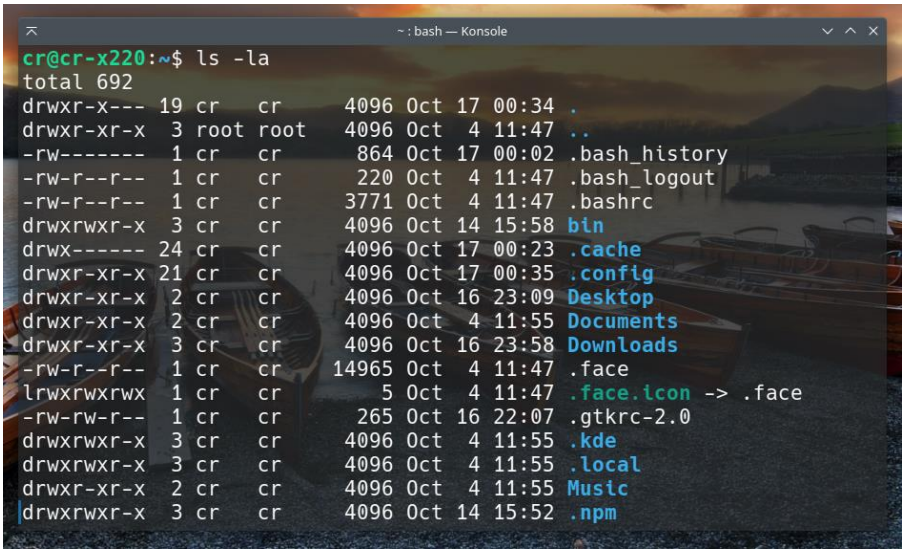
The terminal is a program that runs a shell application, in this particular case (the default on most Linux distributions), we are using the *bash* shell. Which shell you are using doesn’t matter for most of the core commands (and these should also be very close to the ones you find on Mac), but there are some differences that may turn up if you’re reading a guide from someone using *zsh* or *fish*.

The first thing we'll do is get comfortable moving around the file system from within the terminal. If you've never done this before this may be a strange thing to do (rather than clicking through folders), but if you just remind yourself you're doing everything you would do in the file manager you shouldn't get disoriented.

To begin, let's check what directory we're currently in. To do this type in the *pwd* command. By default you should be in the home directory. A good way to orient yourself in the file system is with the *ls* command. This command will print out all the files/directories in the present directory, though by default it will not show so-called "hidden" files. You can also add an argument *-l* to *ls* (i.e., *ls -l*) to list each file on its own line with other information about permissions, ownership, and date modified. Below we show each of these commands in succession.



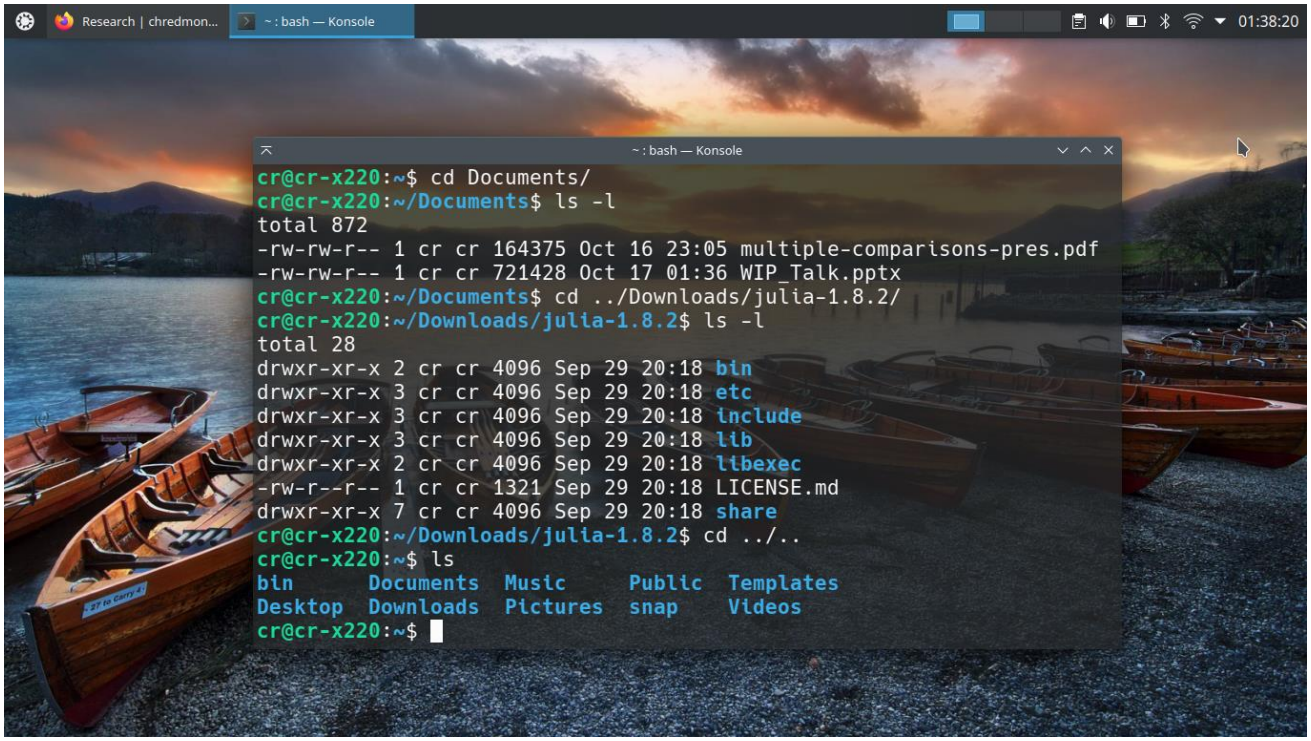
If you want to see all the files in a directory, including the hidden ones, you need to add the *a* argument, which can be used on its own (*ls -a*) or tacked on to the end of another argument (*ls -la*). Here we can see that there are many new files/directories we haven't seen before, all pre-pended with a dot. These are sometimes called dotfiles for that reason. We'll cover what some of these files are for later. For now you can ignore them.



Next we'll try moving around the file system by changing directories. This is accomplished with the *cd* command. Just type *cd* and the name of the directory you wish to go to. For the directory name you can enter either the *relative path* or the *absolute path*. The relative path makes reference to the current directory (e.g., typing in "Desktop" makes sense in the home directory where there is a folder called "Desktop", but wouldn't work in the "usr" directory). The absolute path contains the full directory tree (e.g., "/home/[username]/Desktop"), and will work no matter which directory you're currently in.

For example, consider the sequence of commands below. First we move into the "Documents" directory (*cd Documents/*; the slash on the end isn't necessary) and list out the files in that directory (again, a good way to know where you are). A useful tip for any command-line work is to use the TAB key to complete text (e.g., if you type *cd Doc* and hit TAB it will complete *Documents* because it knows *cd* is looking for a directory/file in the home directory and the "Documents" directory is the only possible completion of "Doc"; for fun, try out what happens when you hit TAB after *cd Do*).

Next we change into the "julia-1.8.2" directory within the Downloads folder. This file path is a bit more complicated because it uses the ".." notation, which stands for "one directory up". So *cd ..* would take us back into the home directory, while *cd ../Downloads* takes us up into home and then down into the Downloads directory. Finally you can see from within this folder inside Downloads we can go back up two levels into the home directory with *cd ../..* (alternatively, if you ever want to jump straight to the home directory just type in *cd* and hit ENTER).



Now that you've moved around the file system let's copy/move some files. To copy a file use the *cp* command, followed by the path of the file you want to copy and the new path representing where you want to copy it to. For example, if we wanted to copy "WIP_Talk.pptx" from "Documents" into "Downloads", we'd write:

```
cp Documents/WIP_Talk.pptx Downloads/
```

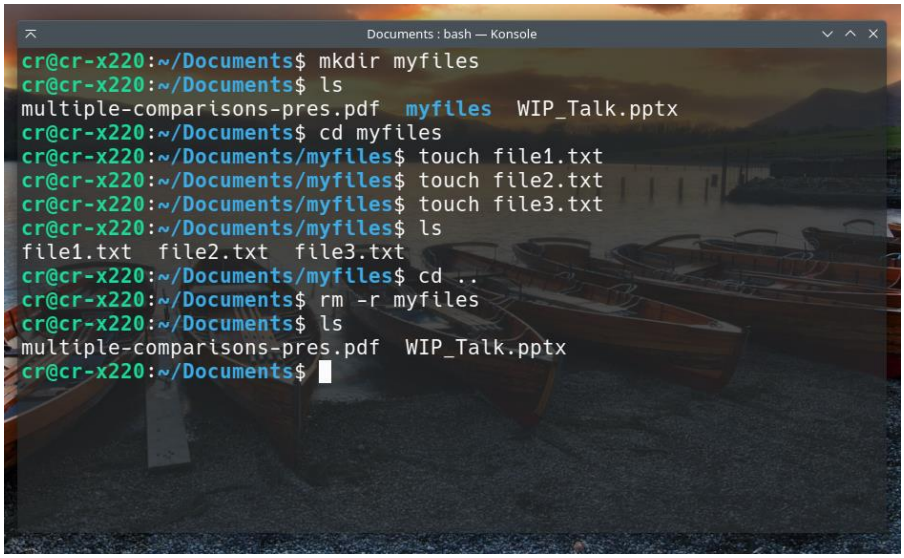
In the terminal most commands have arguments that are separated by spaces. This is one reason why many people recommend against putting spaces in file names.

If you want to move a file (similar to cutting and pasting it, which you can do in the file manager) replace *cp* with *mv*. This is also how you can rename a file in the terminal. If you keep the directory location the same and change the name of the output file (e.g., *mv Documents/WIP_Talk.pptx Documents/fMRI-presentation.pptx*).

Finally, if you want to remove a file you use the *rm* command. To remove a simple file just type *rm* and the name of the file you want to remove:

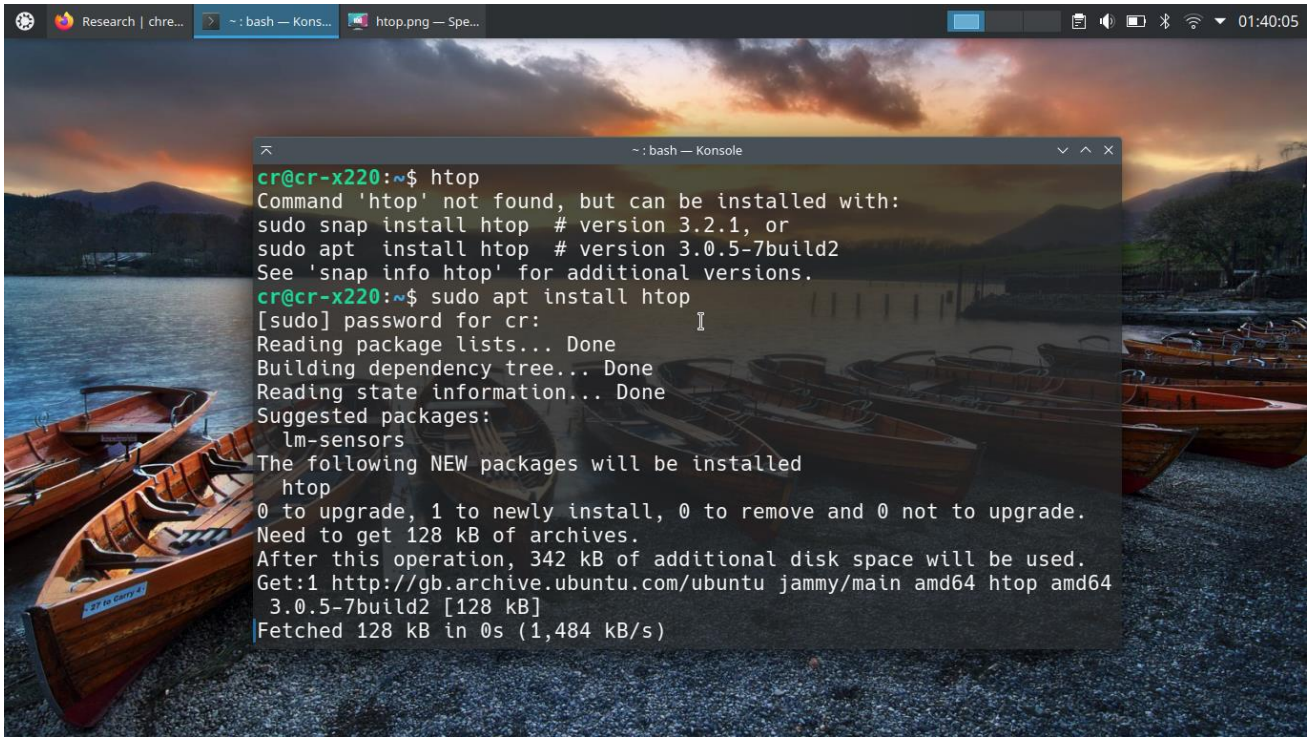
```
rm Documents/WIP_Talk.pptx
```

To remove a directory, you need to add the *-r* (recursive) flag. In the example below we introduce several new commands to test this out. First, we create a directory called "myfiles" using the *mkdir* command. Then we *cd* into that directory and create several new files using the *touch* command. These will all be empty files but this is a nice quick way to create a file that you can then edit later. Finally, we move back up a level to try out deleting this directory using *rm -r*. Sometimes you will need to add a further flag *rm -rf* (or *rm -f* for a single file) if there are files in the directory that you don't have write permission over, but be careful, this is a very powerful command and can destroy lots of data if you're not careful. Later we'll show how to add some safeguards in the terminal.

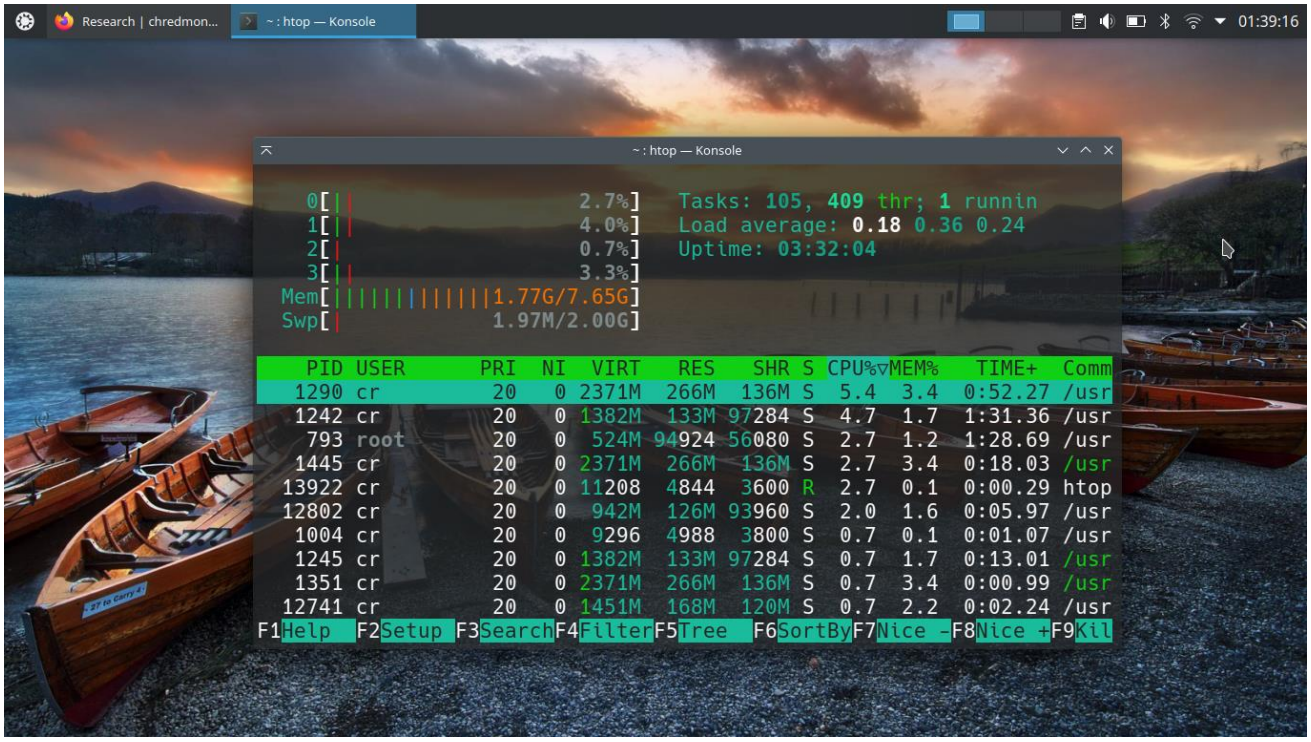


Exercise 9: In the terminal, create a directory, add some files to it, then copy some to a new directory, rename another of the files, then delete the files and directory.

Now you should be comfortable doing basic operations in the terminal. Next we'll show how to do a very common thing in the terminal, which is installing and running a program from the terminal. First let's try and run the *htop* program, a program that gives you running information on your system resources. To run *htop* just type in the name into the terminal and hit ENTER. In the example below, we try this and Ubuntu turns up an error saying that there is no such *htop* command. This is a common problem, but Ubuntu is helpful it telling us how to install the program that is missing. This will not always happen (if Ubuntu is not familiar with the program you are using it might just say "Command not found" with no other help. But here we are told we can either install the command as a Snap package or from *apt*, Ubuntu's (based on Debian's) software repository. We'll use the second method and type in *sudo apt install htop*. Let's break this command down piece by piece: *sudo* tells the system to run the command after it with elevated privileges (you will have to type in your password before it will run), *apt* is the command to access the *apt* software repository, *install* tells *apt* we are installing a program (you can also remove software, update, and do many other things in *apt*), and finally *htop* is the name of the program we want to install.

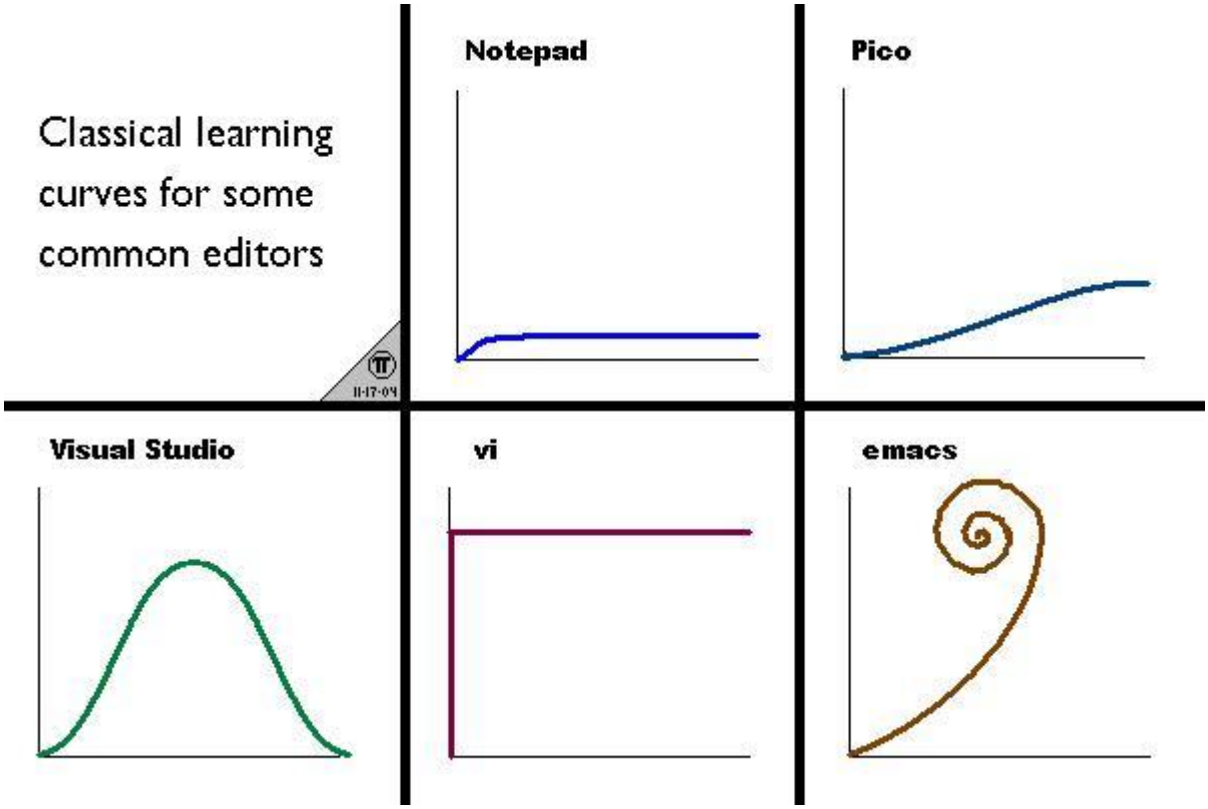


Now that htop is installed we can type in *htop* and hit ENTER to run it. Here is what it looks like:



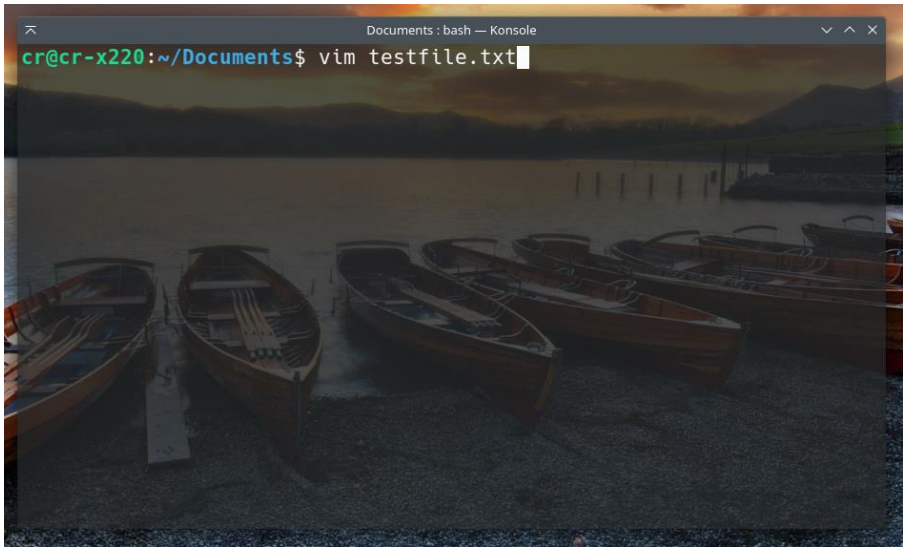
Exercise 10: Update the system and then install the program *vim*. **HINT: you can get information about a command by typing *man [command]* (e.g., *man apt* for the apt package manager).

Now that you've installed Vim, let's use it to do some basic text editing in the terminal. Vim is an old terminal text editor, based on *vi*, that is ubiquitous in computing and also famously difficult to use. Here is a classic joke about different text editors and their learning curves:

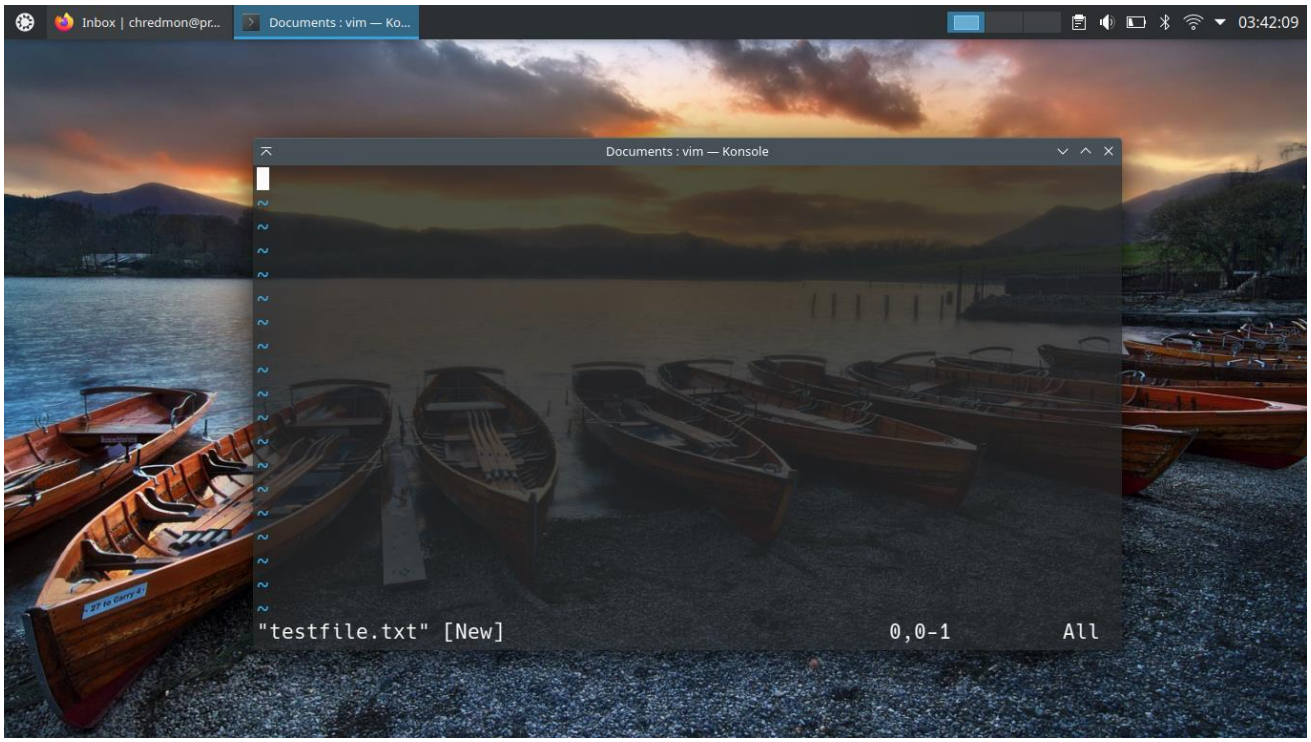


Vi/Vim is immediately difficult, so why introduce it here? There are two reasons to become at least a little familiar with Vim. First, it is often the only program installed on servers for working with text files, so if you need to access a remote server this may be your only tool. Second, many command-line programs (e.g., *git*) will drop you into Vim directly to edit a file, and a very old programming joke is "How do you generate a random sequence of characters? You drop a new user into Vim and ask them to exit." So at the very least you should know what to do if you find yourself in this situation.

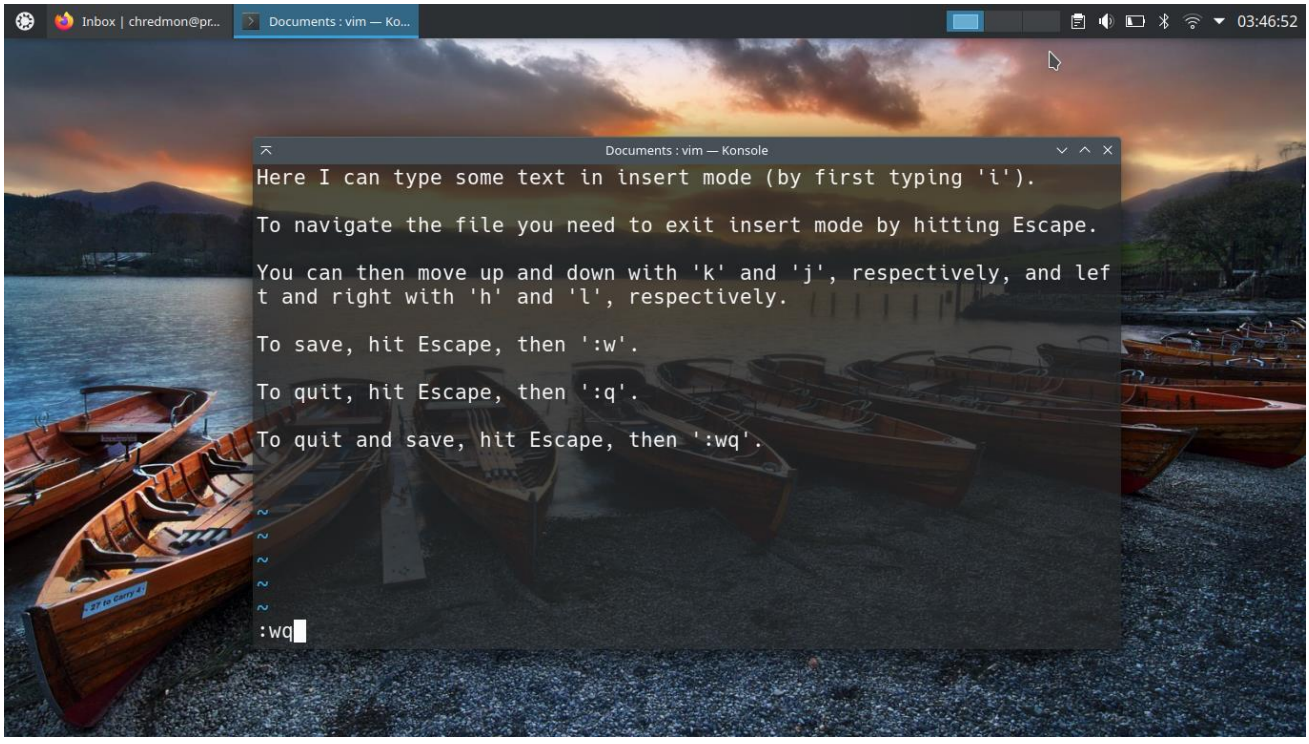
To edit a file in vim, simply type *vim [filename]*. This works even if you don't have a file yet, as Vim will create an empty one first.



Next, you should see an empty file screen like this:



Right now you are in the command mode, where you can run commands and move around the file but you can't make any changes to it yet (you'll notice that if you start typing you won't see any text appearing in the file). To edit the file, you first need to type 'i', which will take you to INSERT mode. Now you can edit all you want. The image below shows you a few other basics about moving around and finally about how to exit vim, which you do by first making sure you're in command mode (by hitting Escape) and then typing ':q' or ':wq' (the latter quits and saves).



Exercise 11: Edit a file in Vim, save, and exit.

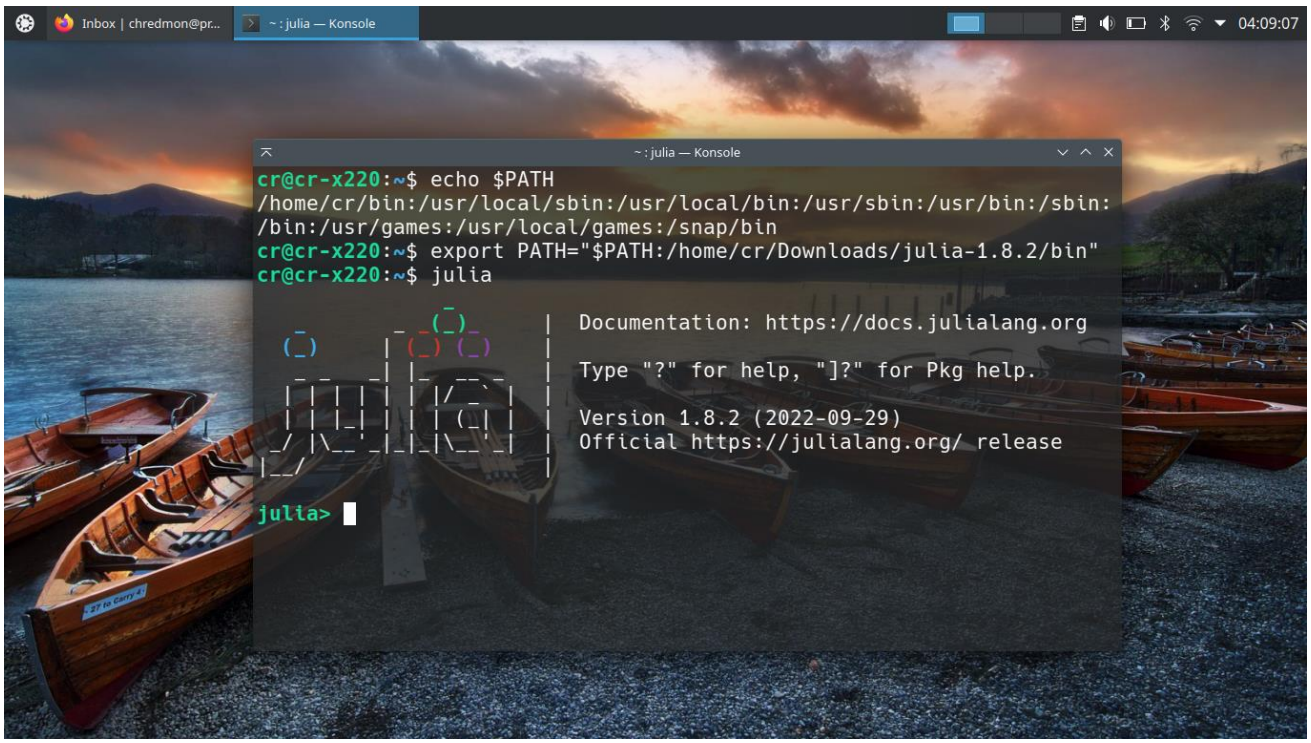
Lastly, there are a few more odds-and-ends of working in the terminal that are useful to know:

You can represent any text string with the wildcard character '*', which then allows you to apply commands to sets of files that match certain patterns. For example, if you want to copy only the PDF files in a directory you can type: *cp oldir/*.pdf newdir/*

You can send the output of one command to a second command with the pipe symbol '|'. For example, if you want to list out a set of files and count the words in each of them (using the *wc* command) you can type: *ls *.txt | wc*

To find out where a command is located in the file system, you can type *which [command name]*. This is sometimes useful if a command you want to run isn't working; it could be an error with the path variable.

In order to run a command from the terminal it must be in the system's path, contained in the PATH variable. Here we first check what is in the $PATH variable with the *echo* command, and then we add to the path with the *export* command (here I'm adding the location of the executable Julia program I downloaded, which is in a folder in Downloads). Now we can run Julia without having to go into the folder where the executable is housed.



This will not permanently add the Julia bin directory to our path, but for that we'll need to edit the *bash* configuration file (*.bashrc*). In the next section we'll cover dotfiles as a common way programs store settings and user configurations.

6 System configuration

By now you should be familiar with the System Settings, so this section will focus instead on more complex configurations like working with dotfiles and changing your desktop environment, but just to test and make sure you haven't forgotten anything, try changing the power settings to logout of the session after an hour and dim the screen after 30 minutes. Additionally, pick a program you commonly use and configure it to open on startup (i.e., once you log in the program will launch).

6.1 Config files

In this section we'll focus on a more advanced system configuration: working with config files, particularly dotfiles in the home directory. A primary example to demonstrate this in is the ".bashrc" file, which you can take a look at by either showing hidden files in the file manager and opening in a text editor like Kate, or opening in vim from the terminal via *vim .bashrc*.

This file is run every time you open the terminal, so if you need commands to persist, like our previous path-changing command, then you should add them to this file. The *.bashrc* file in Ubuntu already has a bunch of code in it, and it's a bit intimidating at first. Don't worry, you don't need to know all this at the start. We'll focus on just a couple key areas you can configure that are helpful at the beginning.

First, let's give ourselves a safeguard in the terminal to ask for confirmation before deleting any files. The *rm* command can take the argument *-i* to query the user before deleting. This is often something you forget to add, though. To make it included automatically, we'll create what's called an alias. Put the following line in the *.bashrc* file (there are a few aliases toward the bottom so you might want to group it with them).

```
alias rm = "rm -i"
```

You can see from this line that an alias simply replaces one code with another set in the background, so every time I type *rm* from now on, it will run *rm -i* in the background.

Exercise 12: Add some aliases of your own.

And of course the other critical thing to configure in *.bashrc* is the PATH variable. We already modified this before to add the location of the Julia executable to the path. You should have a line in your *.bashrc* that already configures the PATH, so rather than adding a new line you can just modify the line in the file to add this additional folder.

Alternatively, since we know that the path already includes /home/[username]/bin, we can add the executable to the *bin* folder in our home directory and not need to configure the path at all. The best way to do this is with a symbolic link, because the Julia executable may need to access other files in its directory. Type in the following command in the terminal (assuming you're in the home directory).

```
ln -s bin/julia Downloads/julia-1.8.2/bin/julia
```

What we've done with this command is create a kind of shortcut in the /home/[username]/bin folder that runs the file in the julia-1.8.2/bin folder every time it's called.

6.2 Exploring other desktop environments

We've done enough configuring of the terminal at this point. Let's finally try out some different desktop environments to see what else is out there. There are so many options in Linux, so you should take full advantage of finding exactly what works best for you. Here are two you can install and try out: *Xfce* and *Openbox*. The first is still quite easy to use but lightweight and more of an older style than KDE. The second is extremely minimal and likely doesn't work like anything you're used to.

To install *Xfce* (specifically the *Xubuntu* version, which is specifically customized for Ubuntu), run the following command:

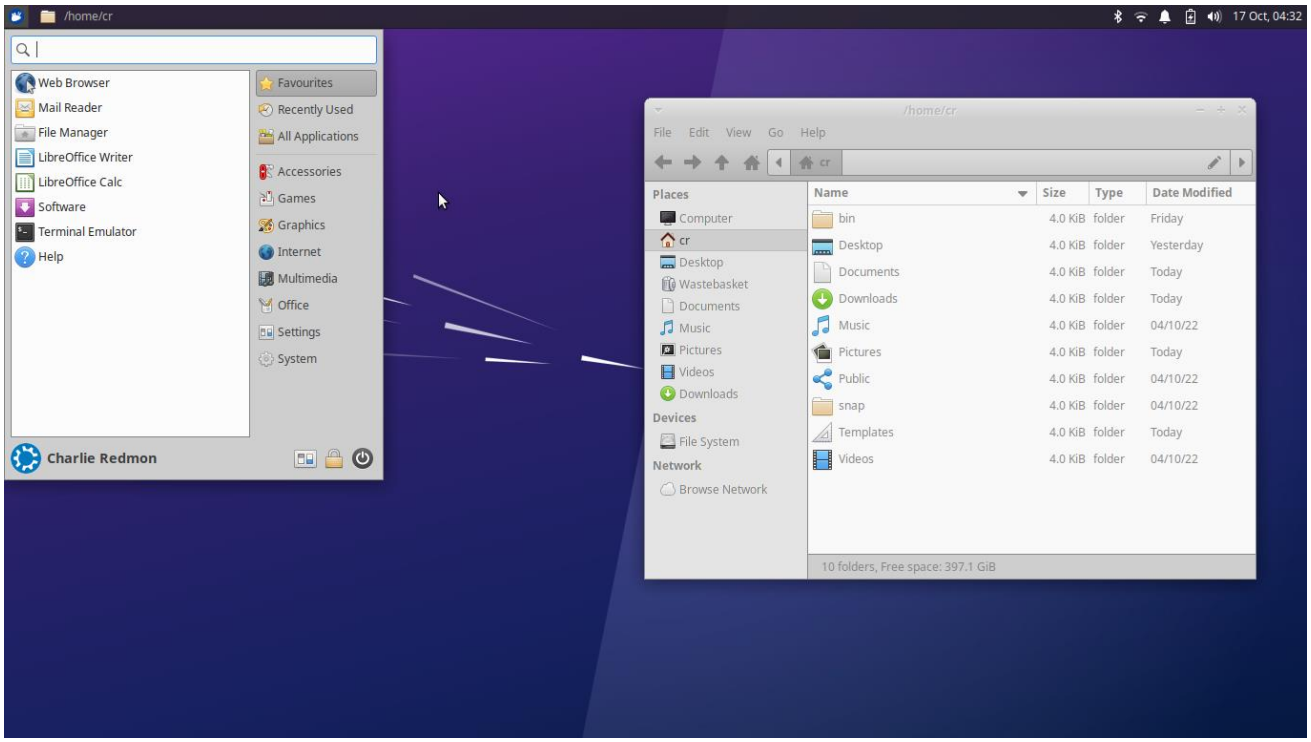
```
sudo apt install xubuntu-desktop
```

To install Openbox, type:

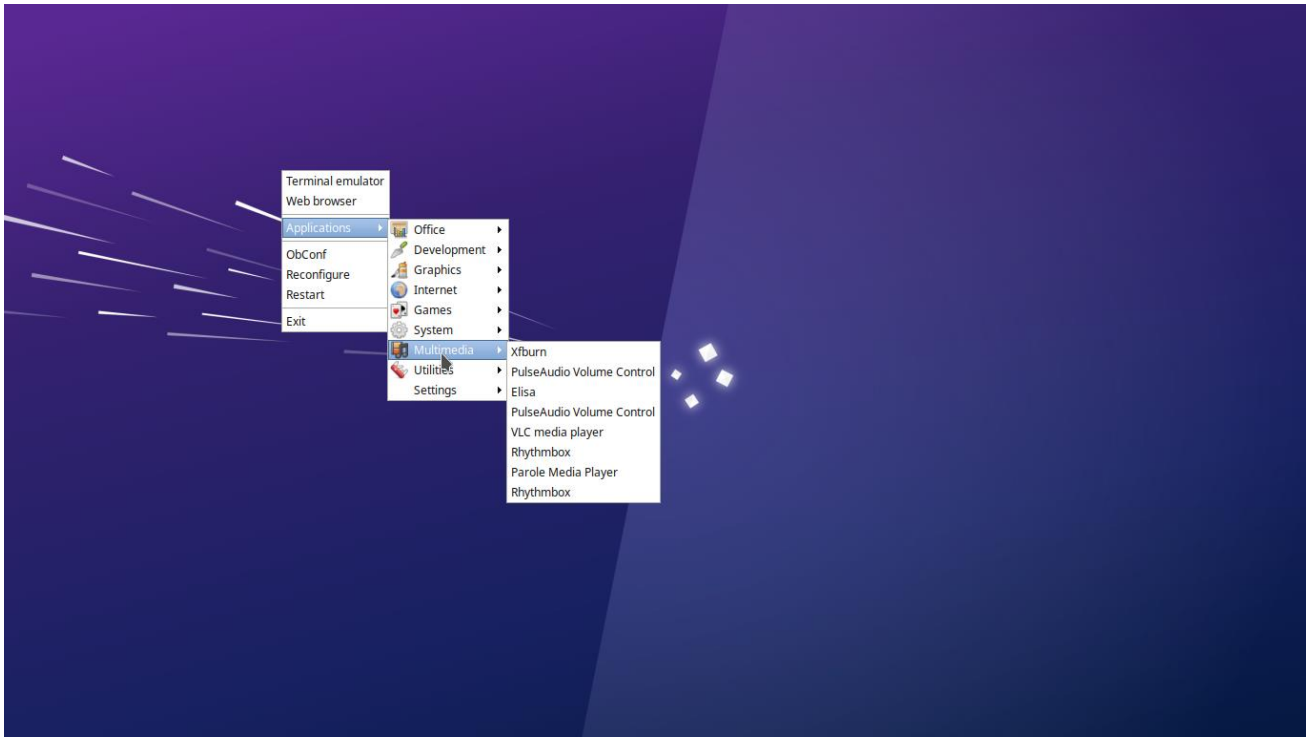
```
sudo apt install openbox
```

These will install a lot of additional packages (the first one in particular). Once you're done with the install, log out and then when you log back in there is a button in the upper right corner that allows you to choose your session. Here you can select not just KDE Plasma (as we've been using presently), but now you also have the options "Xubuntu" and "Openbox". Here's what each looks like at the beginning:

Xfce:



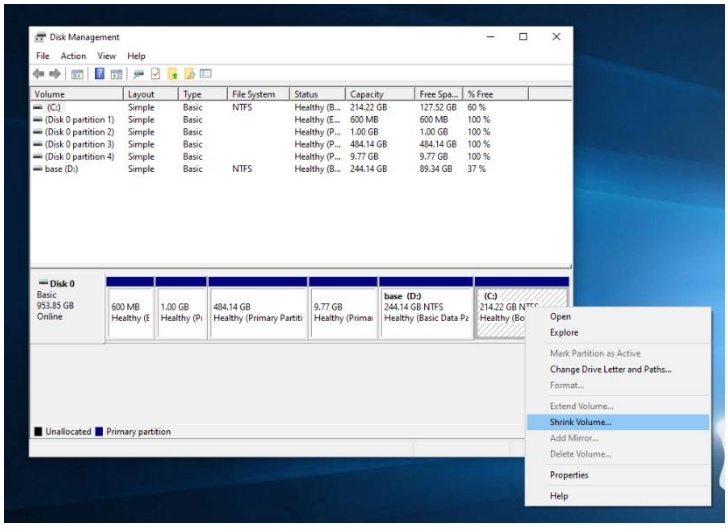
Openbox:



7 Installing Linux for personal use

Finally, we will give a brief overview of how to install Linux on your own system. There are basically just a few steps to this procedure:

1. Get an external USB drive (flash drive, pen drive, whatever you call it)
2. Install a program for writing disk images to USB (Rufus is a good one on Windows)
3. Download the Live USB Image for your Linux distro of choice (it should be a .iso file)
4. Use the program (e.g., Rufus) to write the .iso file to your USB stick (make sure there is nothing currently on your USB drive because the program will completely wipe the data.
5. This step is only for dual-booting Windows and Linux: Partitioning your hard drive. Open up the Disk Management tool in Windows, right click on the main partition (should be a much simpler configuration than the one I have below, and select “Shrink volume…” You can then create a large chunk of unallocated space to install Linux in.



6. Now boot from USB. This is kind of tricky and used to be much easier. Basically, every computer has a particular key that opens up its boot menu on startup. Shut down your computer and then hold down one of the following while it's starting up: Escape, F1, F9, F10, F11, F12. You may need to check your computer's manufacturer to find out which it is. Once you have a boot menu, you can select your USB and then it will boot directly from the USB stick. This pops you into a live linux environment where you can test out your system before installing.
7. Once you're ready to install, there should be a shortcut on the desktop to install Linux to your machine. From there you just need to follow the menus. Most installers these days give you the option to "Install alongside other system" rather than needing to manually set up the Linux partitions yourself, so I would do this if you're new. And of course, the usual warning to BACK UP ALL DATA BEFORE INSTALLING.

And that's it. After going through the installer you should be good to go and the next time you start up your computer it will either boot you directly into Linux or give you a menu to choose Linux or Windows.

Happy Linuxing!

Slide 1

Linux: An introduction

Charles Redmon
charles.redmon@ling-phil.ox.ac.uk

MT 2022

Slide 2

Your safety is important

- Where is the fire exit?
- Beware of hazards:
  - Tripping over bags and coats
- Please report any equipment faults to us
- Let us know if you have any other concerns

Slide 3

Your comfort is important

- The toilets are along the corridor outside the lecture rooms.
- The rest area has vending machines and a water cooler.
- The seats at the computers are adjustable.
- You can adjust the monitors for height, tilt and brightness.

Slide 4

Session 1

Introducing
the Linux desktop

Slide 5

IT Services Linux Courses

- Today's course is divided into six parts each consisting of:
  - A guided group tour of new concepts and methods
  - Individual exercises
- The six parts are
  - Overview of Linux
  - Some common programs and their installation
  - The file system
  - Basic command line operations
  - System configuration
  - Installing Linux for personal use

Slide 6

Session 1

- What is Linux?
  - Open source and why it is important
  - Where it came from and how it is made
- What is Linux used for?
- How to get Linux
- Getting acquainted with the Linux desktop

Slide 7

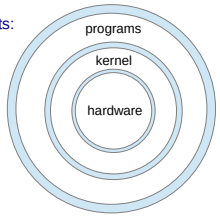
How Linux changed things

- Linux does not come from a single large corporation
- It offers an alternative approach
  - Freedom of choice
  - Freedom to understand and change
  - Software written with quality rather than profit as a goal
- Competition and alternative approaches benefit users and consumers
  - Software developments: Firefox for example
  - Increased awareness of open standards by Microsoft
  - Easier to work using different systems

Slide 8

What is Linux?

- Linux is an operating system designed by Linus Torvalds in the early 1990s.
- An operating system protects the user from the hardware and vice versa.
- An operating system has several parts:
  - kernel
  - programs
- Other Operating Systems include
  - Windows (11, 7, XP)
  - MacOS
  - BSD



Slide 9

Who makes Linux?

Linus Torvalds created the first Linux kernel in August 1991:

"Hello everybody out there using minix-
I'm doing a (free) operating system
(just a hobby, won't be big and
professional like gnu) for 386 (486) AT
clones. This has been brewingsince
april, and is starting to get ready."

Text of original email from Linus to the comp.os.unix newsgroup sent on 25 August 1991.

Now linux is developed by the Linux Foundation.

Slide 10

How is Linux developed?

- Linux is maintained and developed both by businesses and private individuals
  - Hundreds are involved in work on the Linux kernel
  - Thousands work on the applications which come with Linux distributions
- It is known as open source software
- Linux or GNU/Linux?
  - Linux is the kernel
  - GNU is the project to create a Unix-like operating system
- GNU stands for **G**NU's **N**ot **U**nix (terrible Unix joke)

Slide 11

What is Linux used for?

- Linux was originally designed for desktop use but this did not make a significant impact on the market
- Linux did quickly gain a large following for
  - Servers – for example, web servers
  - Internet services (DNS, routers etc)
  - Programming
  - High performance computing
- Significantly improved user interface and economic advantages mean that Linux is gaining popularity on the desktop

Slide 12

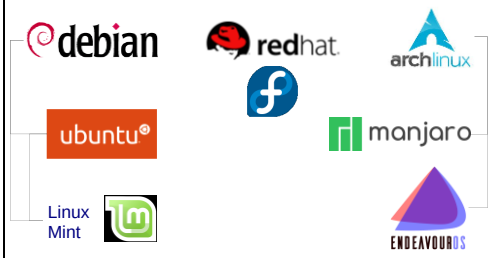
Getting Linux

- The easiest way to get Linux is to download or to buy a "distro" or distribution.
- **A distribution = a kernel + applications + installation software + support + documentation**
- Typical applications are
  - Office software
  - Databases
  - Programming languages and tools
  - Web browsers
  - Email readers
  - Internet services (www, DNS, NIS, firewalls)
  - Games

Slide 13

Linux distros

The three big distros are Debian, RedHat/Fedora, and Arch:



Slide 14

Linux desktop environments

Distros determine the supported software and system configuration tools, but the *Desktop Environment* is what determines your ultimate user interface experience on Linux, and many distros offer a variety of desktop environments.

Here are a few of the big ones:

Slide 15

Ubuntu / Kubuntu

- For this course we will be using the Ubuntu distro, specifically Kubuntu, which is Ubuntu with the KDE/Plasma desktop environment

kubuntu

- This version in particular is running directly from a USB drive, which is pretty handy because it means you can run Linux without touching the computer's hard drive.
- Ubuntu is a Debian-based version of Linux, providing an easy-to-use desktop and straightforward installation. Specifically, we are using Kubuntu 22.04 LTS (Long Term Support, Jammy Jellyfish)

Slide 16

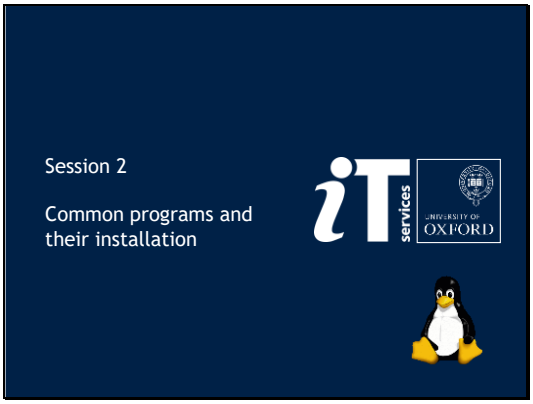
Linux and Windows

- You can install Linux on your PC without removing Windows (or any other operating system).
  - At boot, choose which to use.
  - Could use emulation software (VMWare, wine).
- Linux will read most Windows file systems.
- Windows won't support Linux file systems.

Slide 17

Session 2

Common programs and their installation

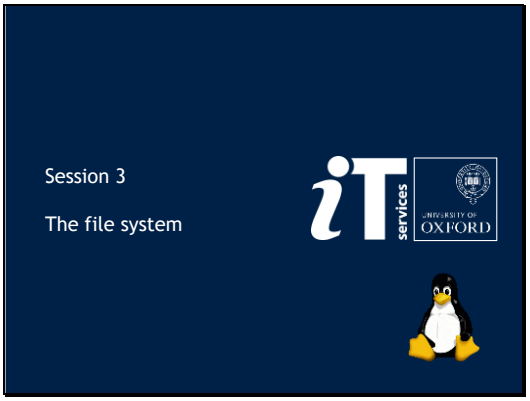


Slide 18

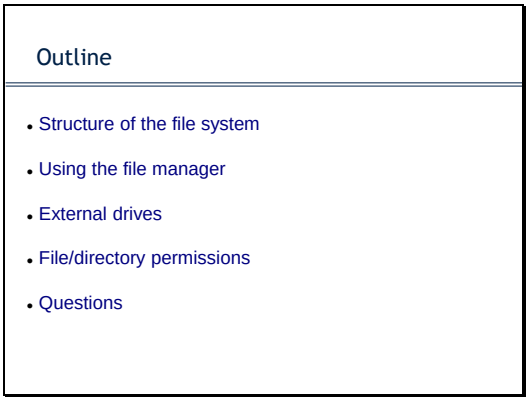
Outline

- Basic Linux applications
- Office applications
- Program installation
- Working with proprietary programs
- Questions

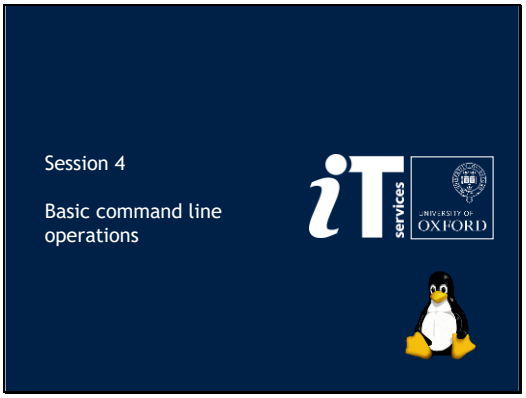
Slide 19



Slide 20



Slide 21

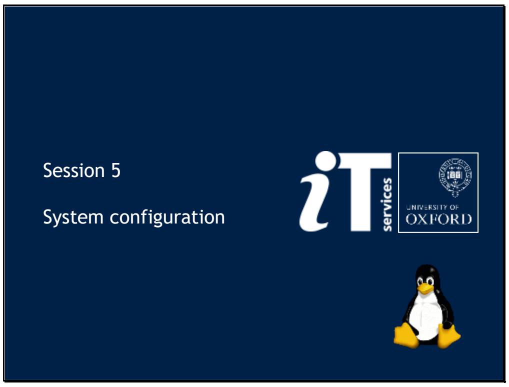


Slide 22

Outline

- Moving around the file system
- Listing, copying, and moving files
- Installing and running programs from the terminal
- Basic file editing in Vim
- Questions

Slide 23

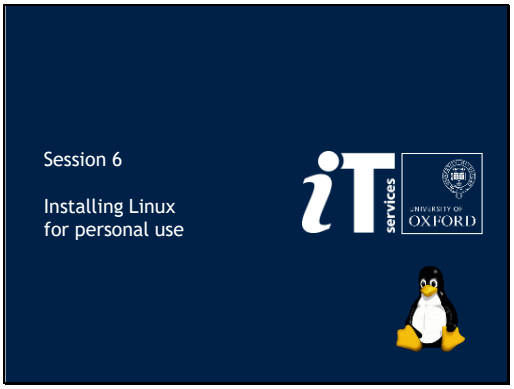


Slide 24

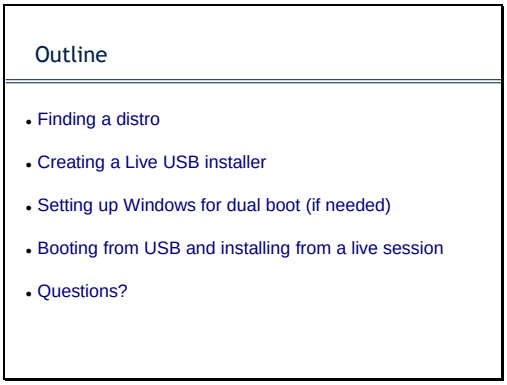
Outline

- Review of system settings
- Config files
- Configuring *.bashrc*
- Questions

Slide 25



Slide 26



Slide 27