NXP PRESENTS

AiM
ARTIFICIAL INTELLIGENCE in MOBILITY

ONLINE DESIGN CHALLENGE

INSTALLATION GUIDE

IN COLLABORATION WITH

Time Of Sports®

INSTALLING UBUNTU Ver – 20.04

PREREQUISITES

1. System requirements (recommended):
    - 2 GHz dual-core processor
    - 4GB memory
    - 25GB available disk space for storage (less if installing the minimal version)
    - DVD drive or USB port

2. At least a 4GB USB drive

Note: It is advised to take a full system backup before proceeding with the installation as faulty installation can lead to loss/corruption of data

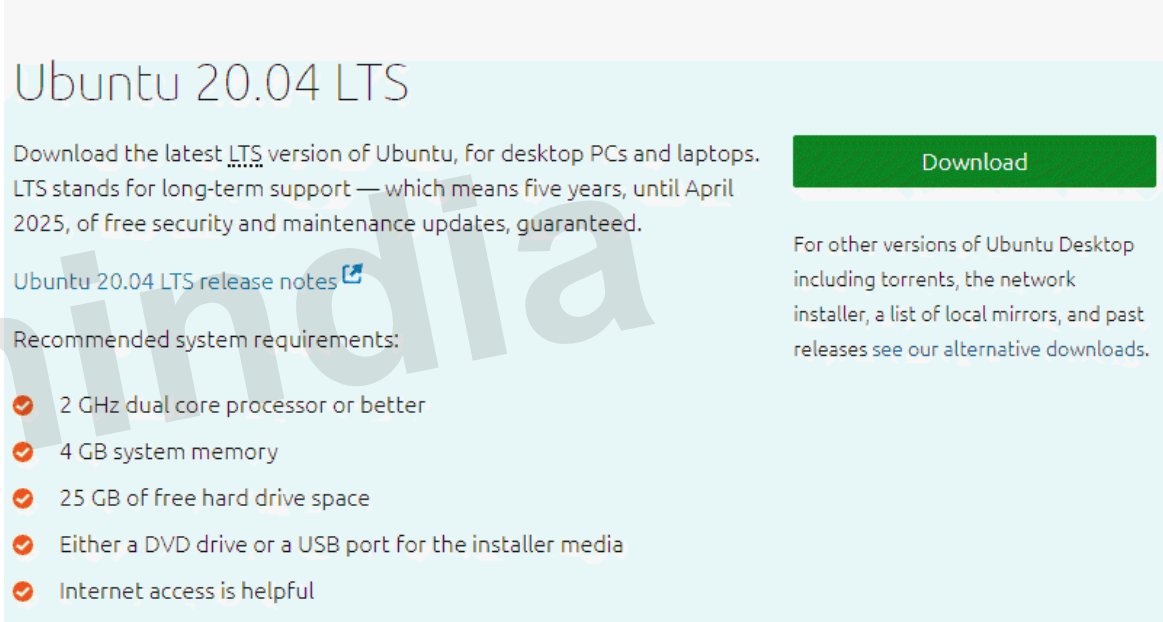
STEP 1: DOWNLOAD THE INSTALLATION MEDIA

1. In a web browser, visit the Ubuntu download page and pick a version suitable for your machine. The most popular versions include:
   - Ubuntu Desktop
   - Ubuntu Server
   - Ubuntu Derivatives

2. Once you find the version you need, click the green **Download** button. You'll be taken to a thank-you page, and your download should start. (We will download and install Ubuntu 20.04 for desktops.)

The download is an **.iso** file. You can use it to create a bootable USB drive.

3. Save the file to a location of your choice.

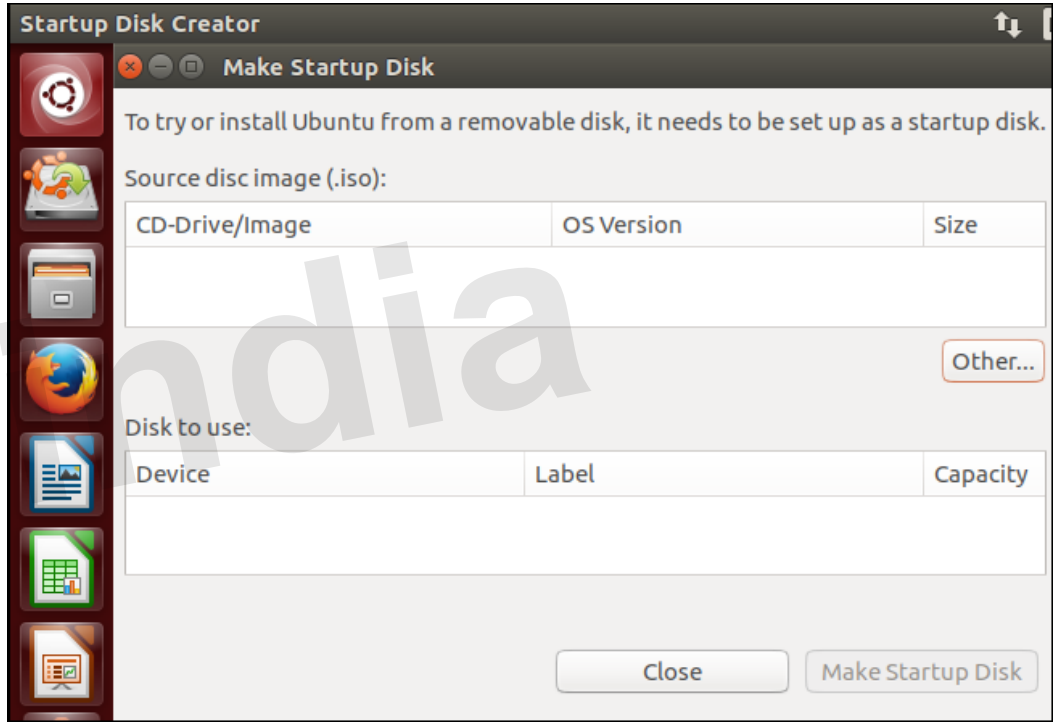


STEP 2: CREATE BOOTABLE USB

You will need a USB drive with 4GB or more. **This process will delete all data on the USB drive.** Make sure to backup any existing data on the USB drive.

Option 1: Create a Bootable USB Drive on Ubuntu

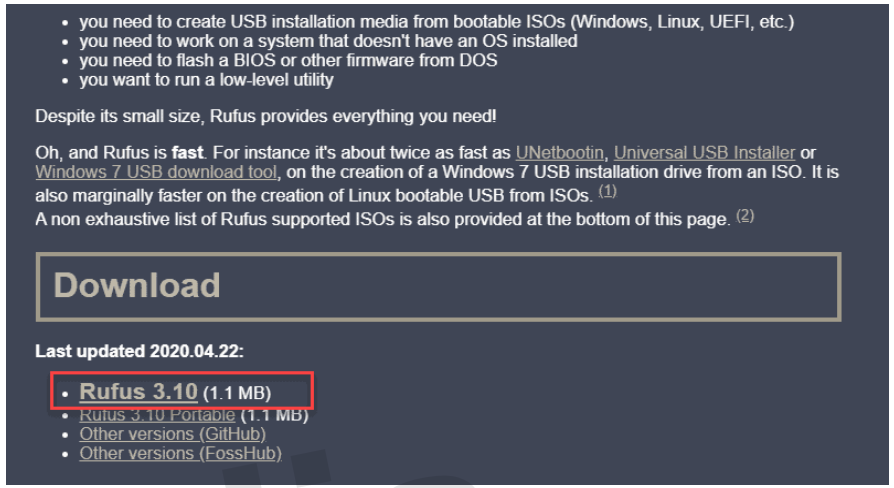
- Use the **Create startup disk** tool:
- Open a **search dialog**, and type *create startup*.
- If it's not installed, the Software Center will offer the option to install it – choose the option for USB drive, then open the utility.
- In the top pane, click **Other**, then browse and select the Ubuntu 20.04 .iso file you downloaded.
- In the bottom pane, select your USB drive.
- Click **Make startup disk.**

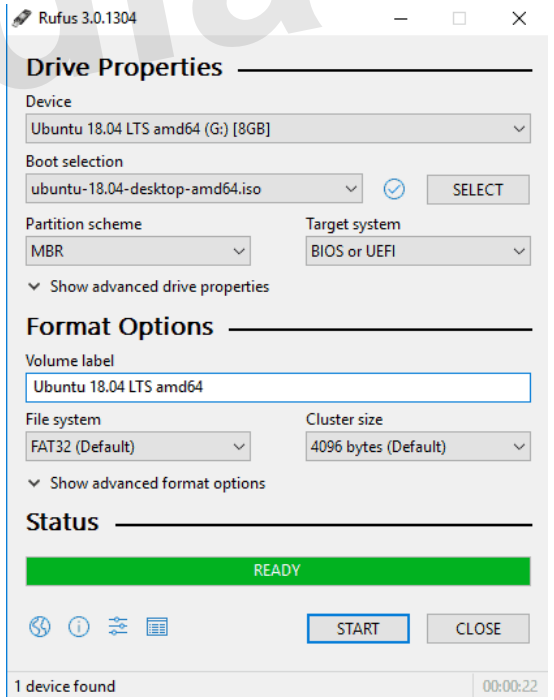


Option 2: Create Bootable USB Drive on Windows

You'll need to install a third-party utility called **Rufus** to create a USB bootable drive.

1. Download the Rufus utility. Scroll down to the download section and click the link to download the latest version of Rufus.
2. Run the file once downloaded.
3. A pop-up dialog opens. You will be prompted whether you want to check for online updates. Select **No**.
4. The Rufus utility launches. Plug in the USB drive – you should see the drive pop up in the device field.
    - Set the USB as the device you wish to write to.
    - In the *Boot Selection* drop-down, click **Disk or ISO Image.**
    - Click the **Select** button to the right.
    - Browse and select the .iso Ubuntu file you downloaded earlier.
5. Click **Start**.





STEP 3: BOOT UP UBUNTU FROM USB

1. **Turn off your system**. Make sure you remove all other USB devices, such as printers, memory cards, etc.
2. **Insert the Ubuntu USB drive** into the system and turn on your machine.
There are two possible scenarios:
    - The computer boots the USB drive automatically.
    - You need to manually configure USB booting in the **Boot Menu** or **BIOS/UEFI**.
3. To manually configure the boot order, tap the boot menu key about once or twice per second as soon as the computer powers on.
4. Once you see your boot menu, use the arrows to pick the Ubuntu media to boot from. For a DVD, the entry will usually have DVD or Optical in the name. USB is usually labeled USB.

Your system should start loading the Ubuntu live disc menu.

The boot menu key may be different depending on your computer manufacturer. Below is a list of common boot keys associated to a brand:

Asus	**F8 or Esc**
Acer	**F12, F9 or Esc**
Compaq	**F9 or Esc**
Dell	**F12**
eMachines	**F12**
Fujitsu	**F12**
HP	**F9 or Esc**
Lenovo	**F8, F10 or F12**
Samsung	**F2, F12 or Esc**
Toshiba	**F12**

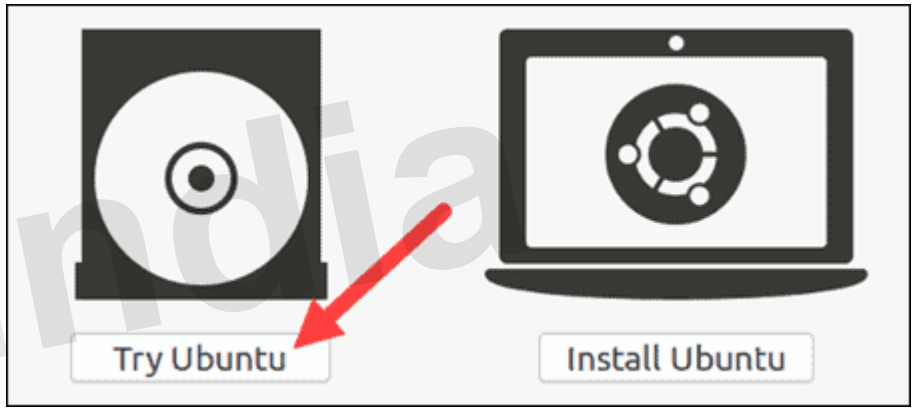
Note: If you are experiencing issues when booting the USB from the boot menu, try to boot the USB from BIOS/UEFI.

STEP 4: RUN UBUNTU

You can test Ubuntu 20.04 before you commit to installing it.

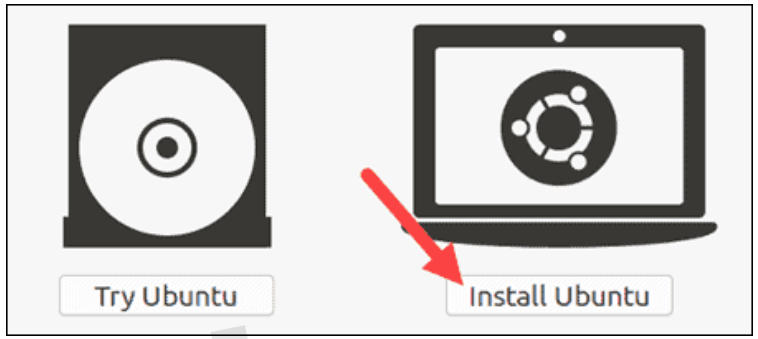
The .iso includes a live mode that only runs in memory.

Launch this mode by clicking **Try Ubuntu**.



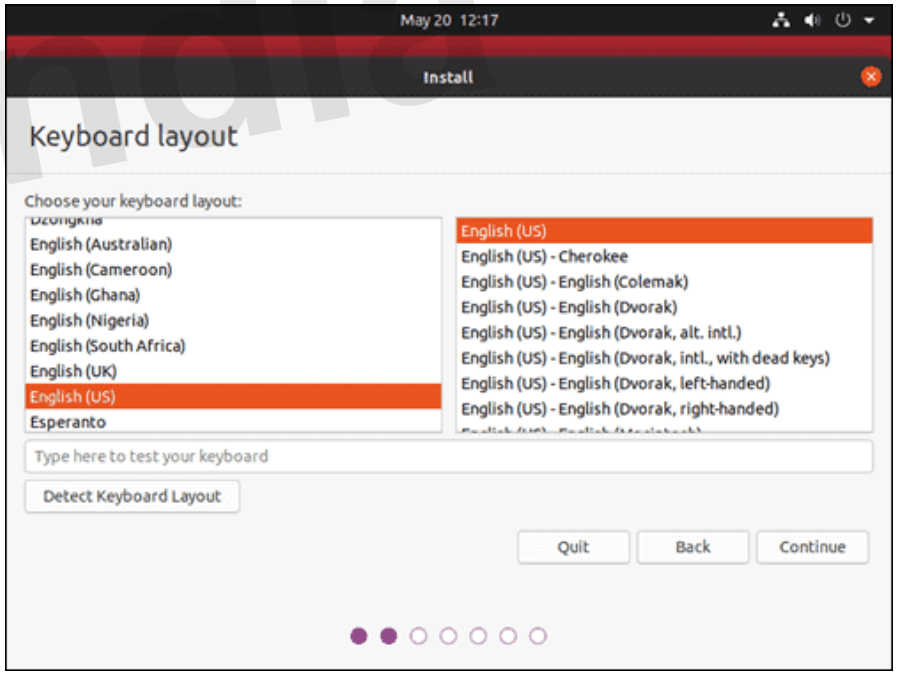
STEP 5: INSTALL UBUNTU 20.04 LTS DESKTOP

To begin the installation, click **Install Ubuntu**.



Choose Keyboard Layout

By default, the system will select English and English.
If you have a non-standard keyboard, you can select it in the list.
Alternately, click **Detect Keyboard Layout** and the system will automatically choose your keyboard.
If you need to test your keyboard, use the labeled field.
When you're ready, click **Continue**.

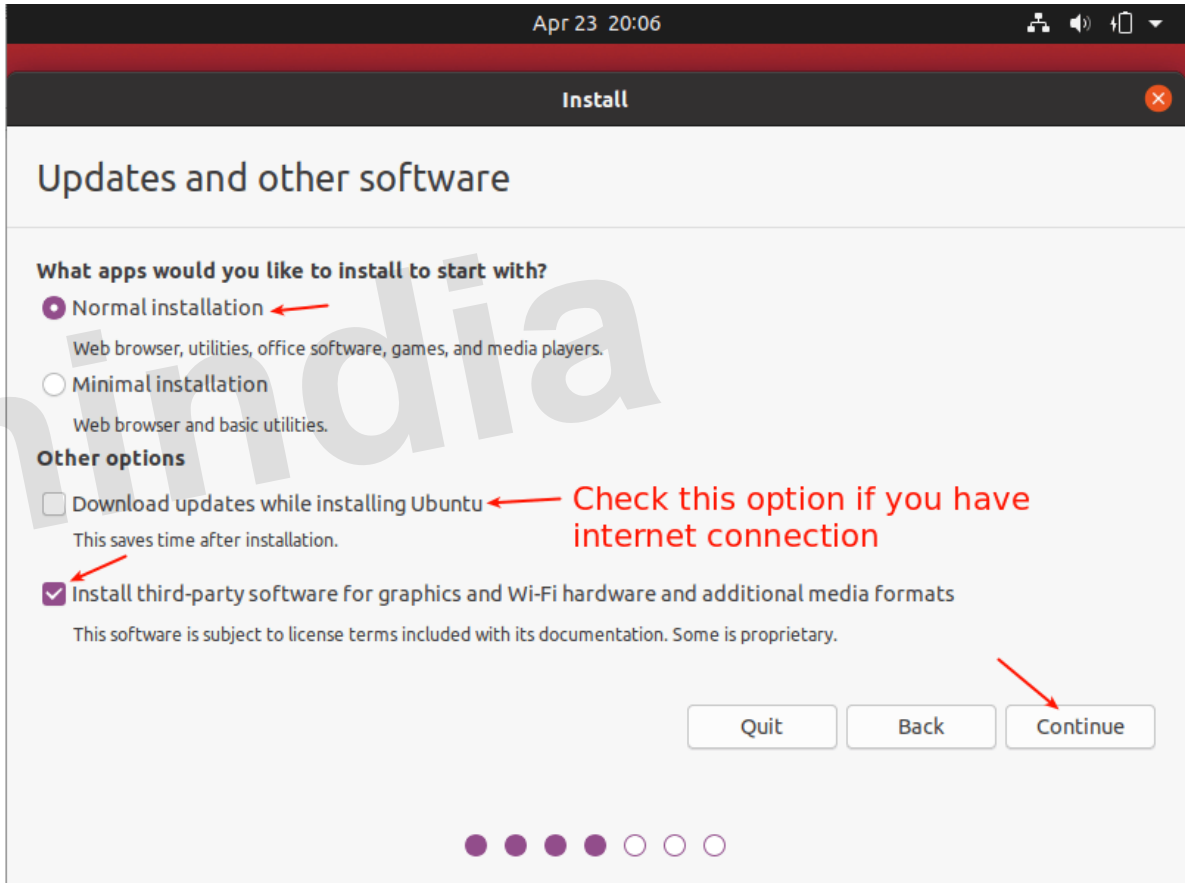


Choose Starting Applications

- **Normal Installation(Recommended)** – This is the full Ubuntu Desktop experience, with office software, games, and media players.
- **Minimal Installation –** Choose this to save disk space, especially if you won't be using media players or productivity software.

You'll also be asked to confirm other options:

- **Download updates while installing Ubuntu** – This does the work of downloading large package files during the installation. Once the installation finishes, the packages will be ready to apply as updates.
- **Install third-party software for graphics and Wi-Fi hardware and additional media formats –** Some hardware, like graphics cards and wi-fi cards, do not have open-source driver support. Also, some media formats, such as .wmv, do not fall under the GPL license. If you need support for these, you'll need to agree to additional terms of use.

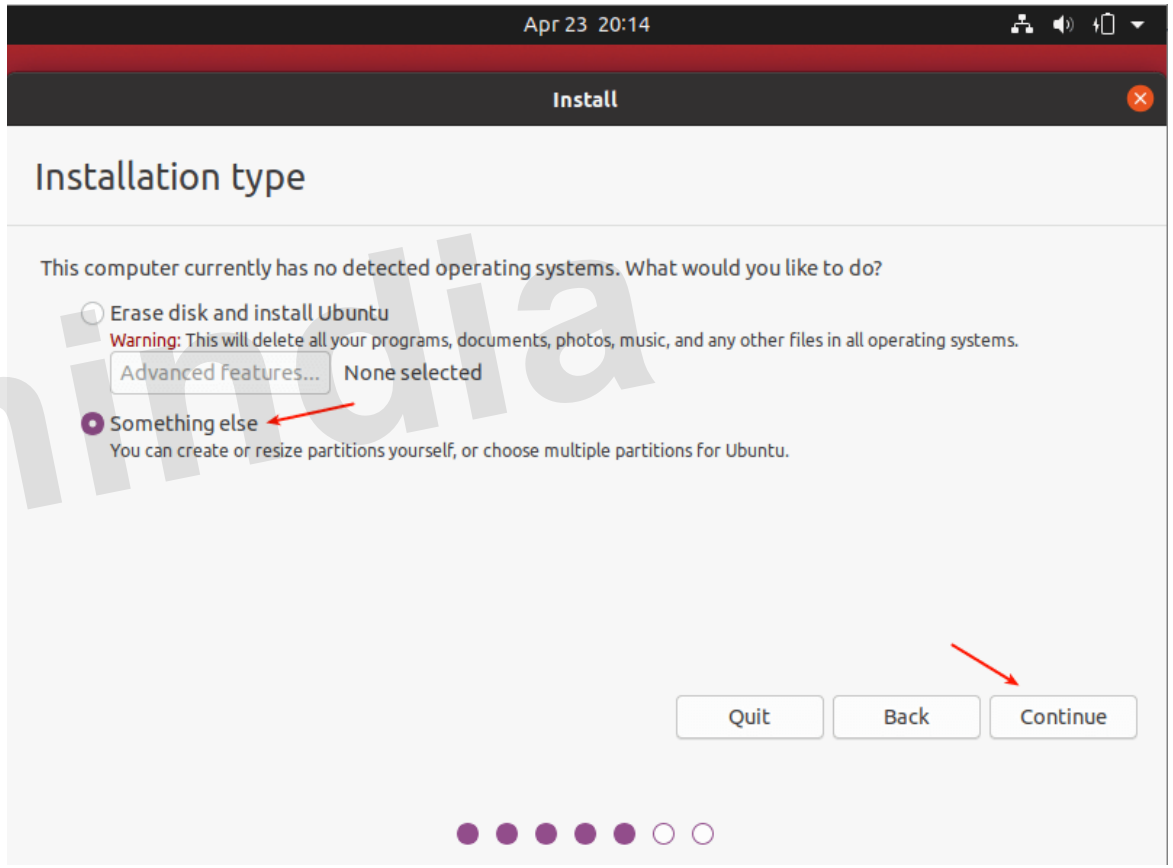


DISK PARTITIONING

Next, you'll be presented with an **Installation Type** dialog.

Scenario 1: Keep only ubuntu ver 20 on hard disk:
You can wipe the hard drive clean prior to installing Ubuntu by clicking **Erase disk and install Ubuntu**. If you go this route, skip ahead to the next step.
Advanced users may want to edit **Advanced Features**.

Scenario 2: Using an Unpartitioned Harddrive (without an OS installed)
For this scenario, you need to set up partitions manually so choose **Something else** and click **Continue**.

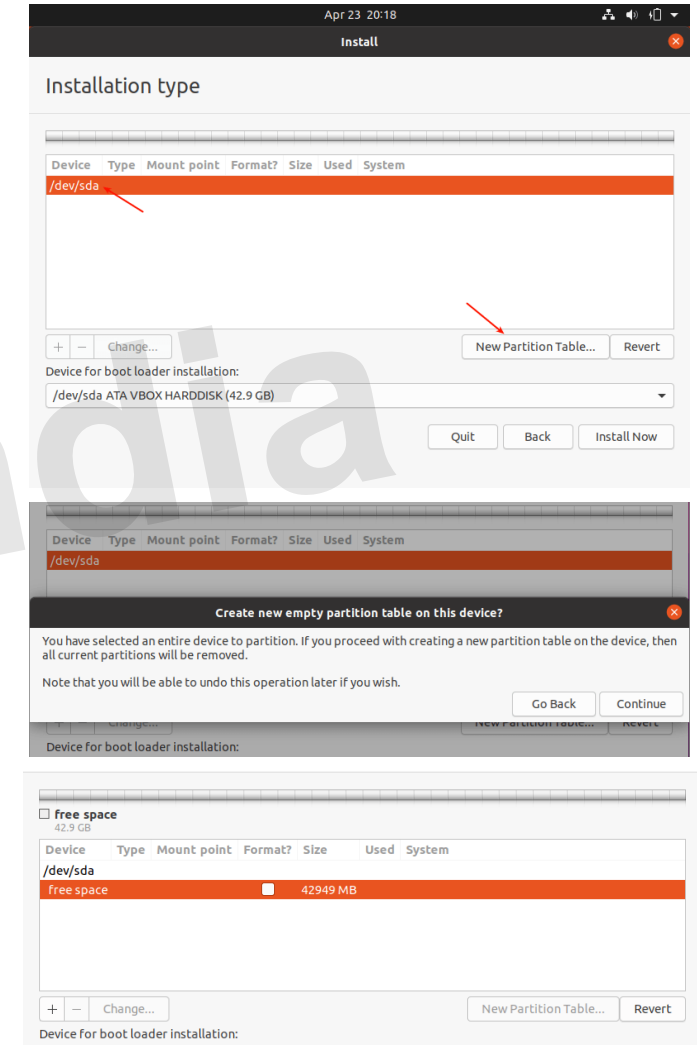


SCENARIO 2 CONTINUED…

Now you need to partition your hard drive for the installation. Simply select/click on the unpartitioned storage device from the list of available storage devices. Then click **New Partition Table**.
Note that the installer will auto-select the device on which the boot-loader will be installed as shown in the following screenshot.

Next, click **Continue** from the pop-up window to create an empty partition table on the device

Now you should be able to see the free space created equivalent to the capacity of the hard drive. Double click on the free space to create a partition as described next.

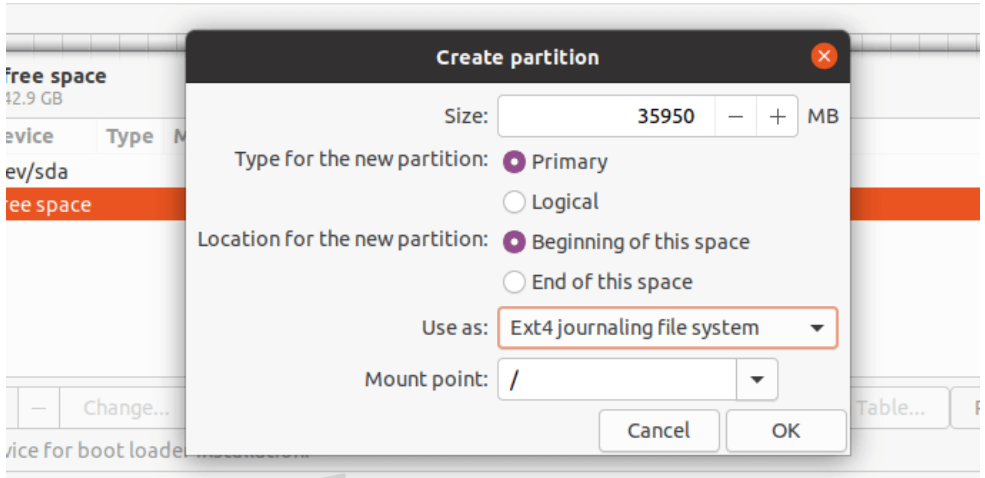


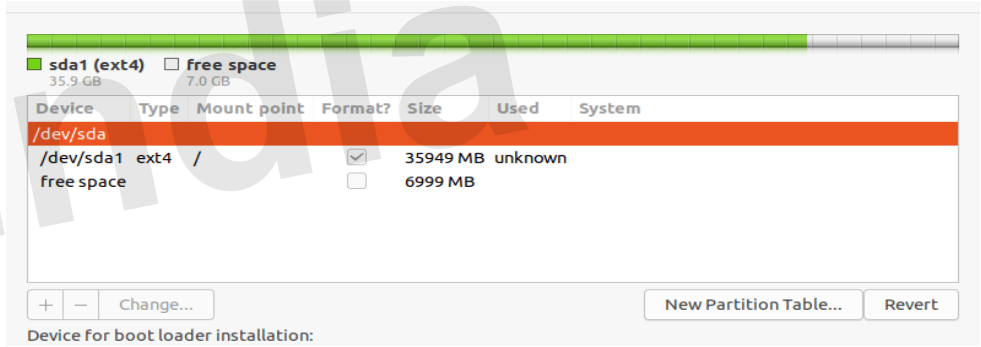
SCENARIO 2 CONTINUED…

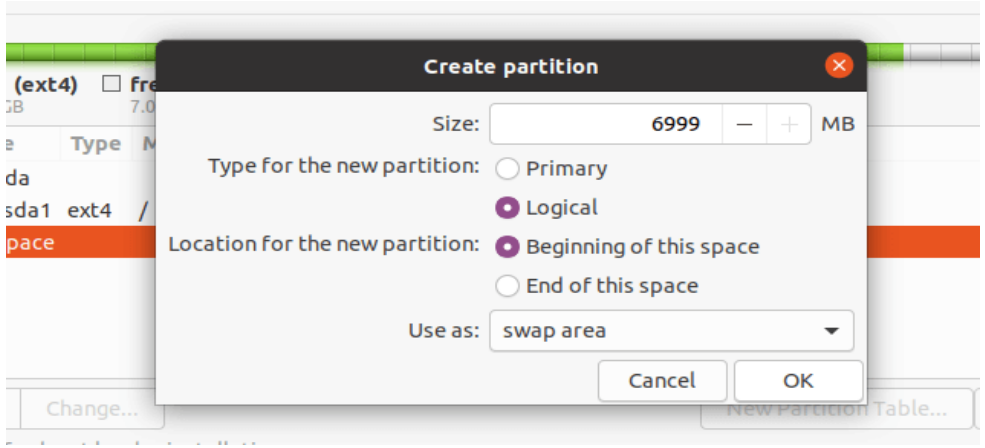
To create a root(/) partition (where the base system files will be installed), enter the size of the new partition out of the total free space. Then set the file system type to **EXT4** and the mount point to / from the drop-down list.

Now the new partition should appear in the list of partition as shown in the next screenshot.

Next, you need to create a **swap** partition/area. Double click on the current free space to create a new partition to be used as swap area. Then enter the swap **partition size** and set **swap area** as shown in the following screenshot.



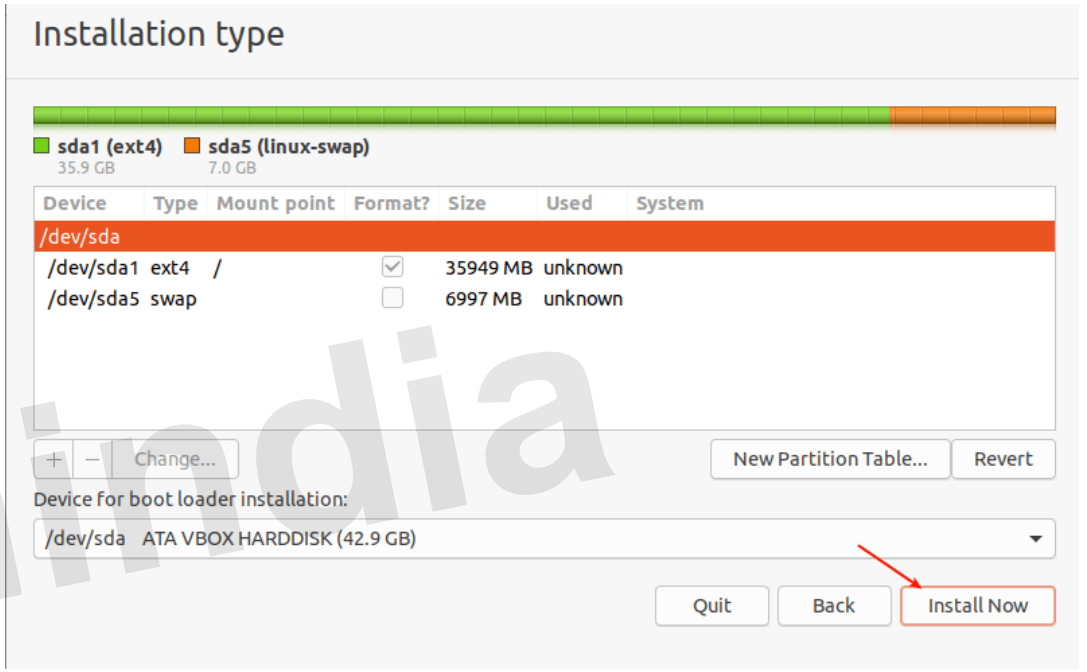


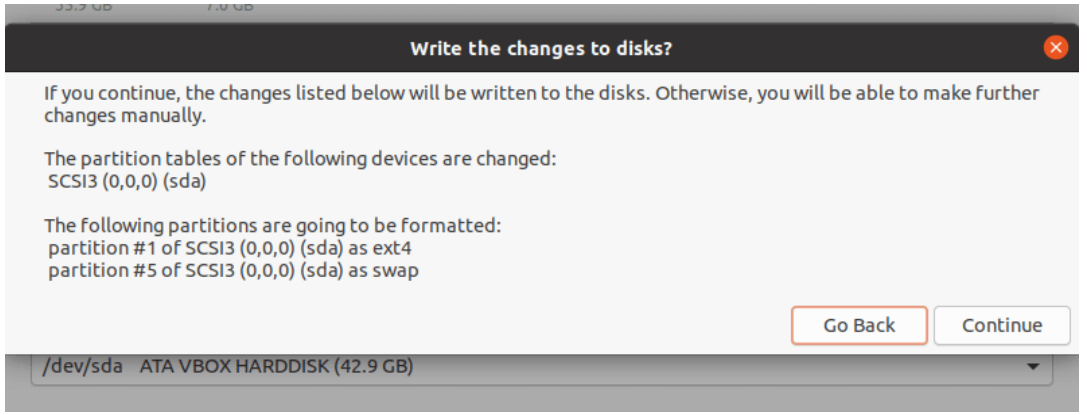


SCENARIO 2 CONTINUED…

At this point, you should see two partitions created, the root partition and the swap partition. Next, click **Install Now**

You will be prompted to permit the installer to write the recent changes concerning partitioning to disk. Click **Continue** to proceed.

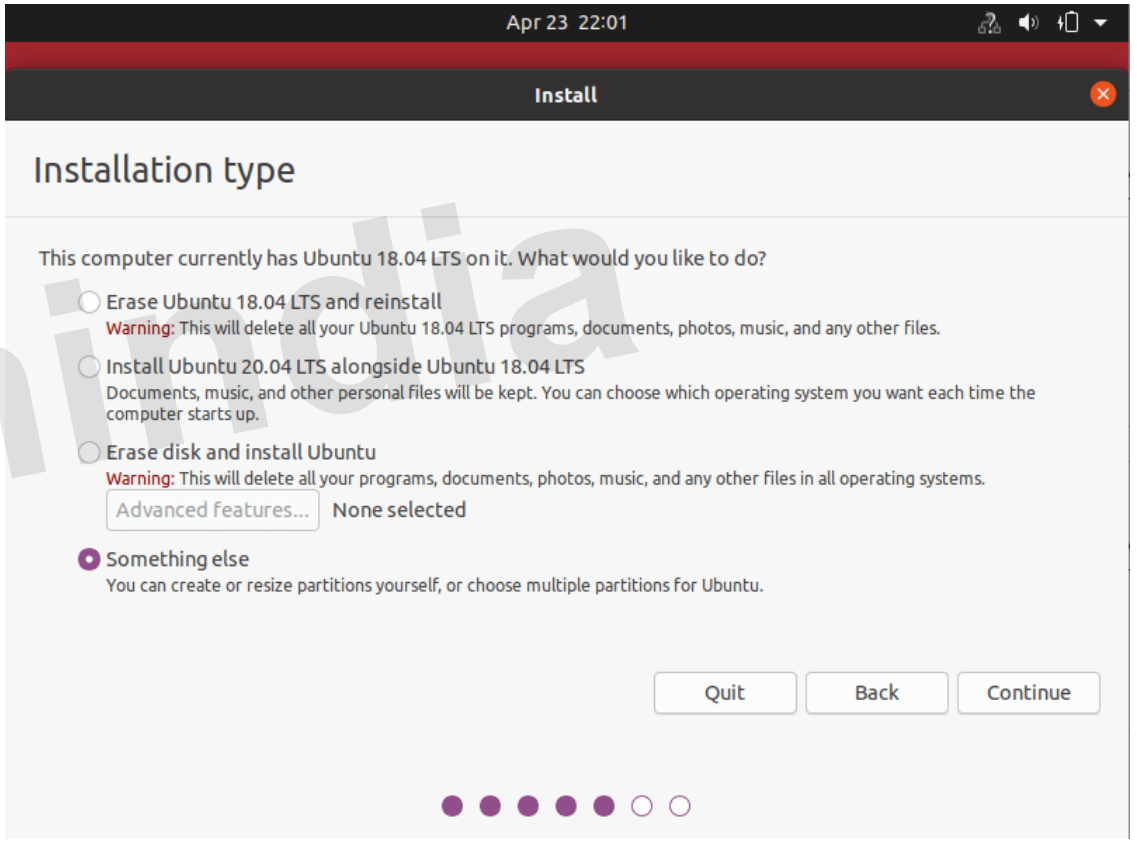




SCENARIO 3: USING AN ALREADY PARTITIONED HARDDRIVE (WITH AN EXISTING OS INSTALLED)

Option 1: Direct installation
Select option **Install Ubuntu 20.4 alongside xxxxxx.** This will enable the system to have more than one OS on it, thoe all system settings will be done automatically

Option 2:
For this scenario, you will use the existing partitions, choose **Something** else and click **Continue**.



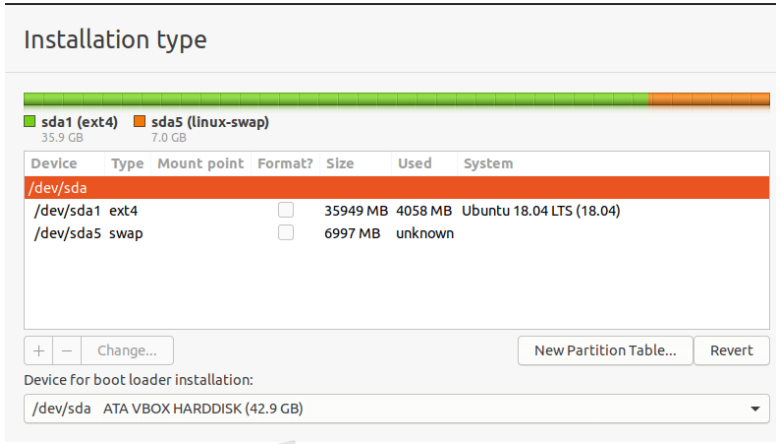
SCENARIO 2 CONTINUED…

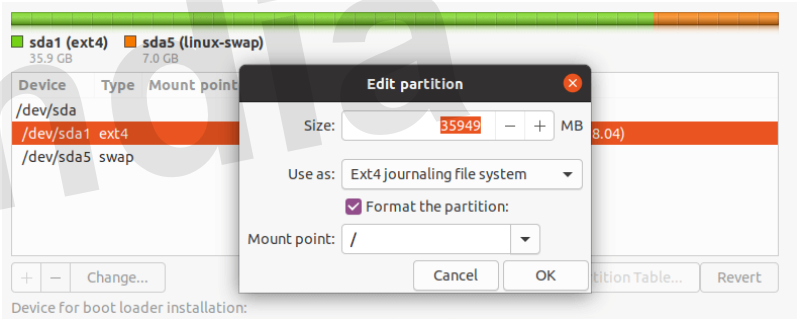
Then you should see your existing partitions for example, as shown in the following screenshot. Double click on the partition with the previous OS installation, **Ubuntu 18.04** in our case.

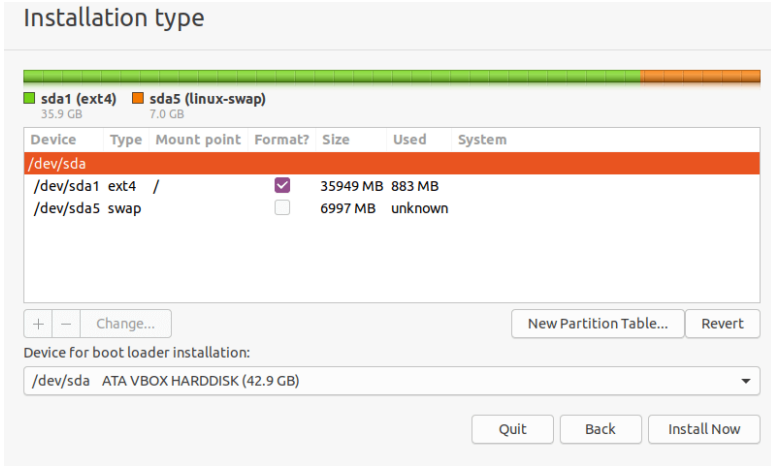
Next, edit the partition and set the file system size, file system type to **Ext4**, and then check the format option and set the mount point to root(/).

Accept the changes in the hard drive partition table, in the next pop-up window by clicking **Continue**.

Now you should have a **root** and **swap** partition as shown in the following screenshot. Note that the swap partition will be auto-detected by the installer. So click **Install Now** to proceed.





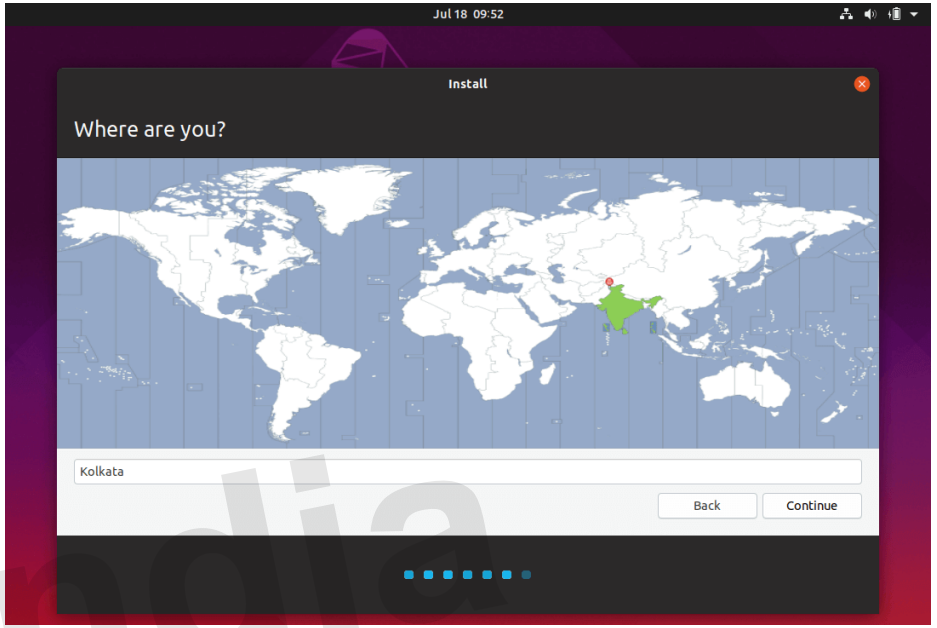


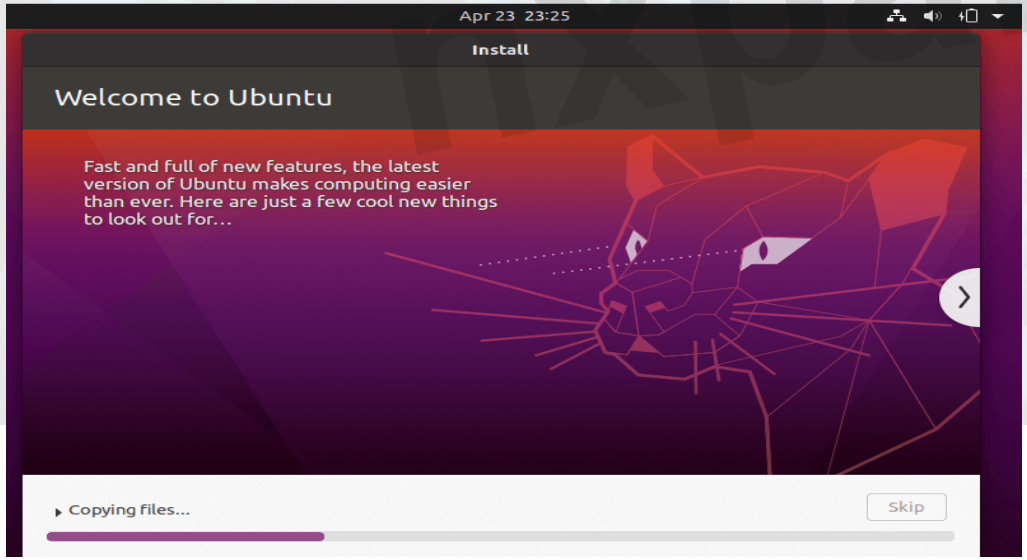
INSTALLATION CONTINUED…

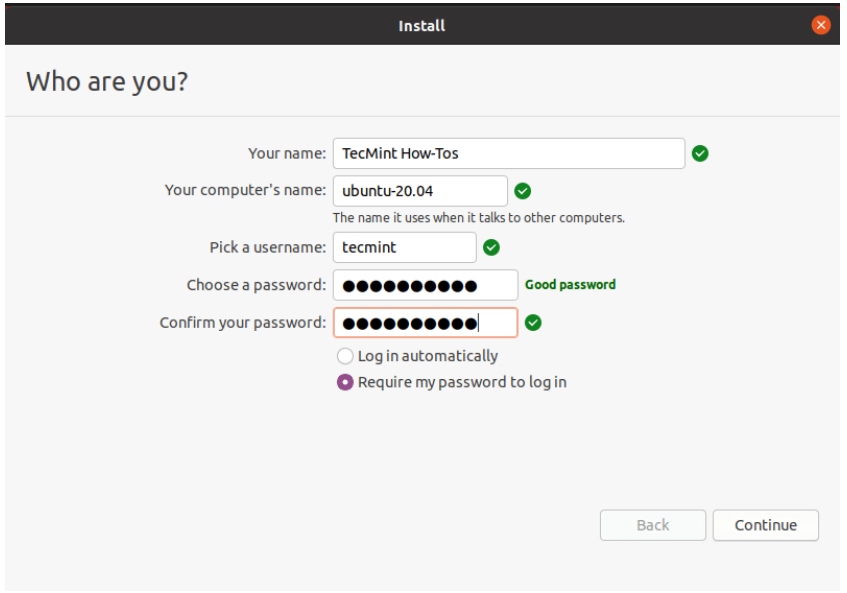
Next, select your **location** and click **Continue**.

Then provide your user details for system account creation. Enter your full name, computer name and username, and a strong, secure password as shown in the following screenshot. Then click Continue.

Now the actual base system installation will begin as shown in the screenshot. Wait for it to finish.



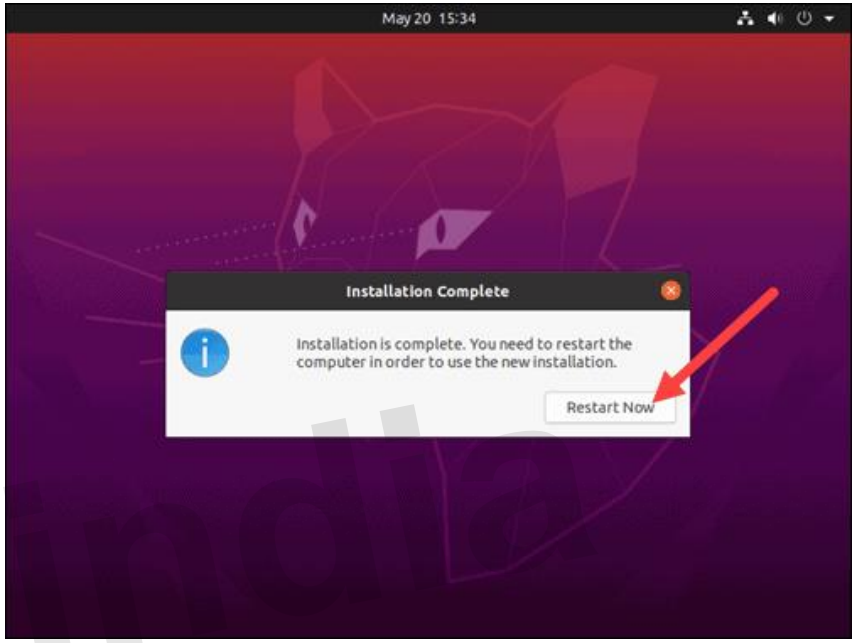


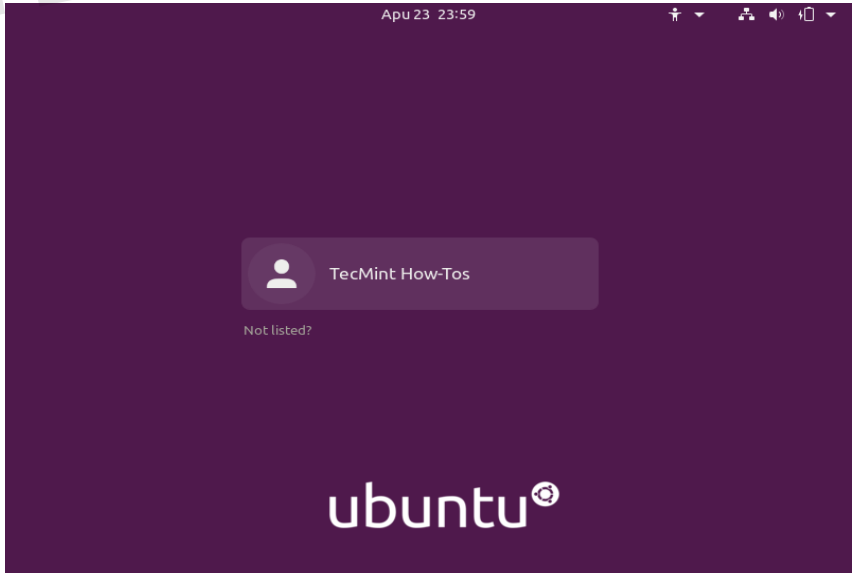


INSTALLATION CONTINUED…

Once the system installation is complete, **reboot** your system by clicking **Restart Now**. Remember to remove the installation media, otherwise, the system will still boot from it.

The system should boot into your fresh install of Ubuntu 20.04.



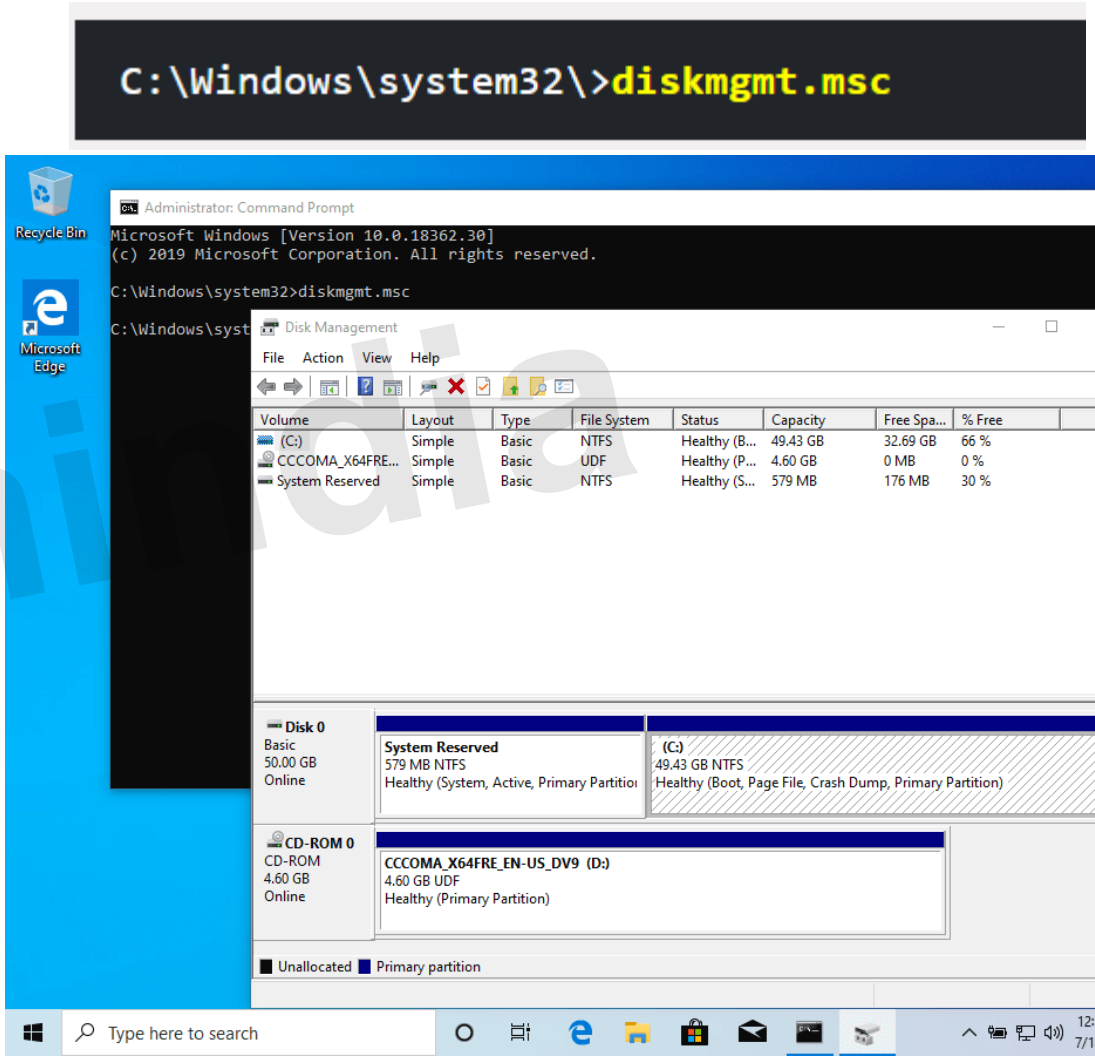


HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT

Step 1: Prepare Windows Machine for Dual-Boot

1. The first thing you need to take care of is to create free space on the computer hard disk in case the system is installed on a single partition.
Log in to your Windows machine with an administrative account and right-click on the **Start Menu -> Command Prompt** (Admin) in order to enter Windows Command-Line.

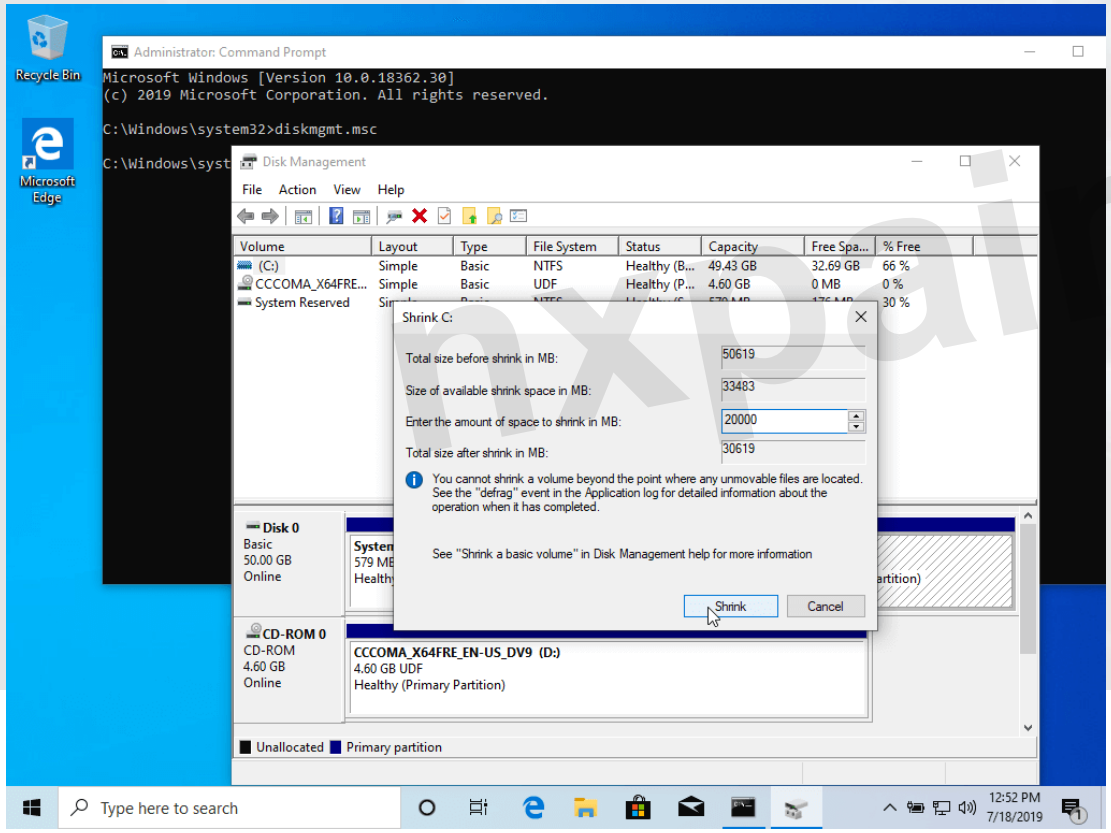
Once in **CLI**, type diskmgmt.msc on prompt and the **Disk Management** utility should open. From here, right-click on C: the partition and select **Shrink Volume** in order to resize the partition.

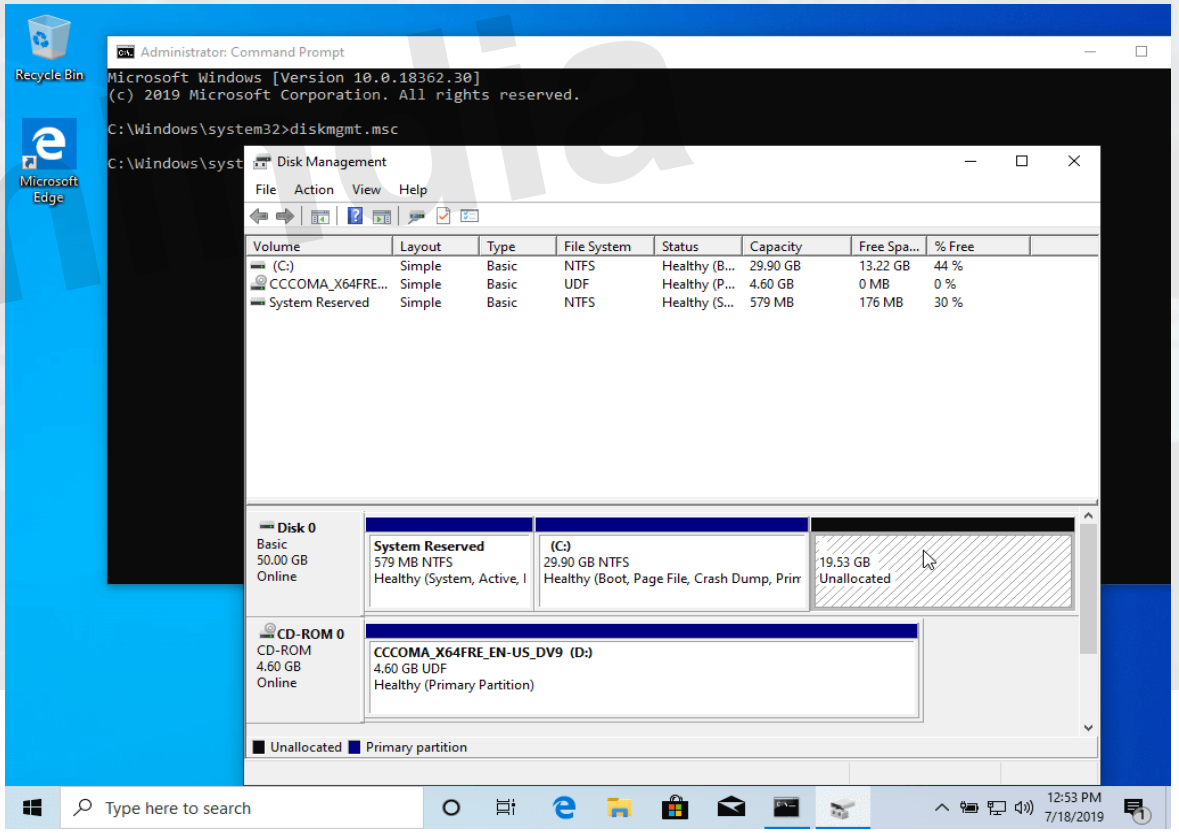


HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED…

On Shrink c: enter a value on space to shrink in MB (use at least **20000 MB** depending on the **C:** partition size) and hit **Shrink** to start partition resize as illustrated below (the value of space shrink from below image is lower and only used for demonstration purposes).

Once space has been resized you will see a new unallocated space on the hard drive. Leave it as default and reboot the computer in order to proceed with the Ubuntu installation.





HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

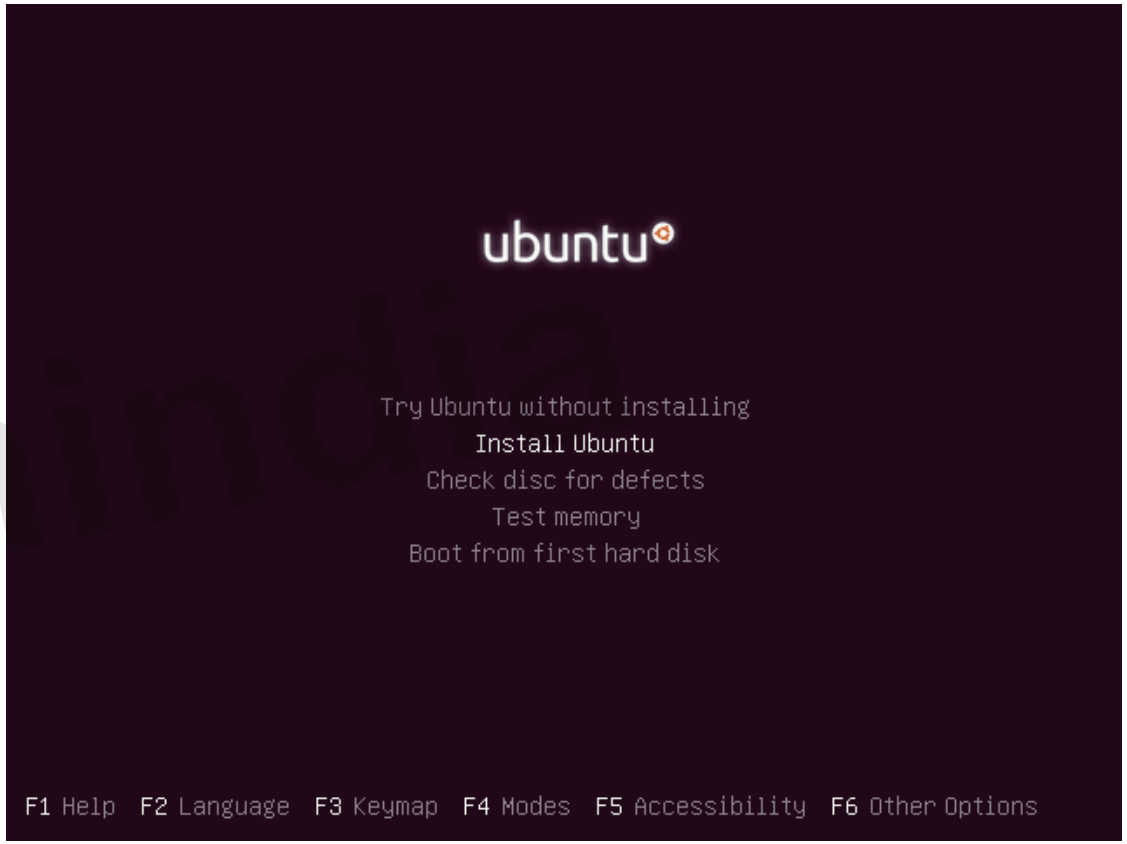
Step 2: Install Ubuntu with Windows Dual-Boot

Go the download link from the topic description and grab **Ubuntu Desktop 20.04 ISO** image.

Burn the image to a DVD or create a bootable USB stick using a utility such as **Universal USB Installer** (BIOS compatible) or **Rufus** (UEFI compatible).

Place the USB stick or DVD in the appropriate drive, reboot the machine and instruct the **BIOS/UEFI** to boot-up from the DVD/USB by pressing a special function key (usually **F12**, **F10** or **F2** depending on the vendor specifications).

Once the media boot-up a new grub screen should appear on your monitor. From the menu select **Install Ubuntu** and hit **Enter** to continue.



HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

Step 2: Install Ubuntu with Windows Dual-Boot

4. Go the download link from the topic description and grab **Ubuntu Desktop 20.04 ISO** image.

Burn the image to a DVD or create a bootable USB stick using a utility such as **Universal USB Installer** (BIOS compatible) or **Rufus** (UEFI compatible).

Place the USB stick or DVD in the appropriate drive, reboot the machine and instruct the **BIOS/UEFI** to boot-up from the DVD/USB by pressing a special function key (usually **F12**, **F10** or **F2** depending on the vendor specifications).

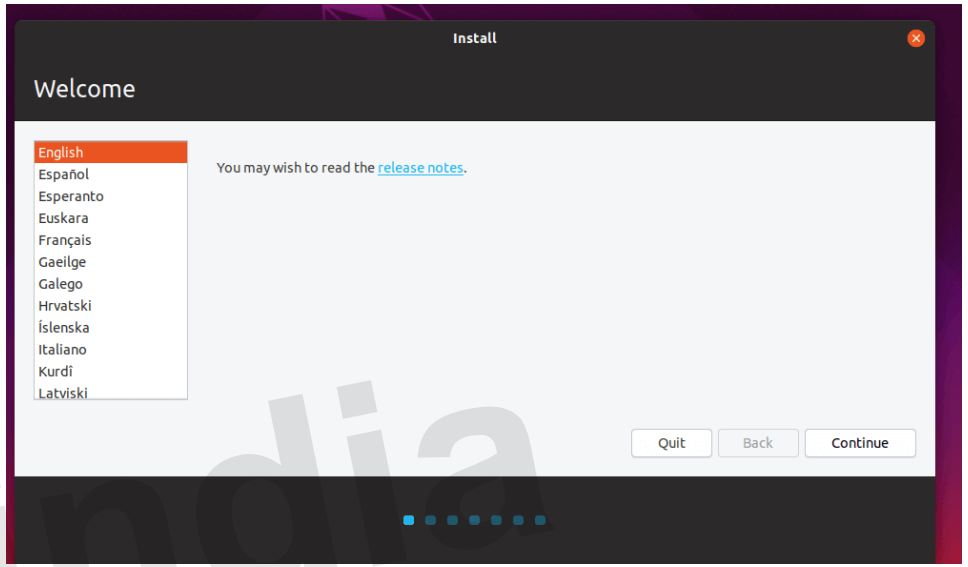
Once the media boot-up a new grub screen should appear on your monitor. From the menu select **Install Ubuntu** and hit **Enter** to continue.

HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

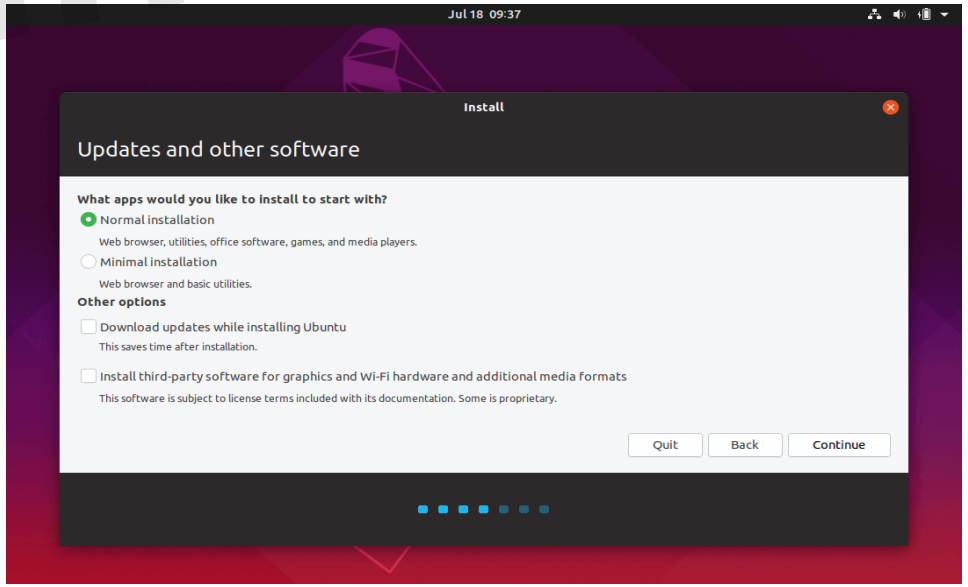
5. After the boot media finishes loading into RAM you will end-up with a completely functional Ubuntu system running in live-mode.

On the Launcher hit on the second icon from top, **Install Ubuntu 20.04 LTS**, and the installer utility will start.

Choose the language you wish to perform the installation and click on the **Continue** button to proceed further.



6. Next, choose the first option "**Normal Installation**" and hit on the **Continue** button again.

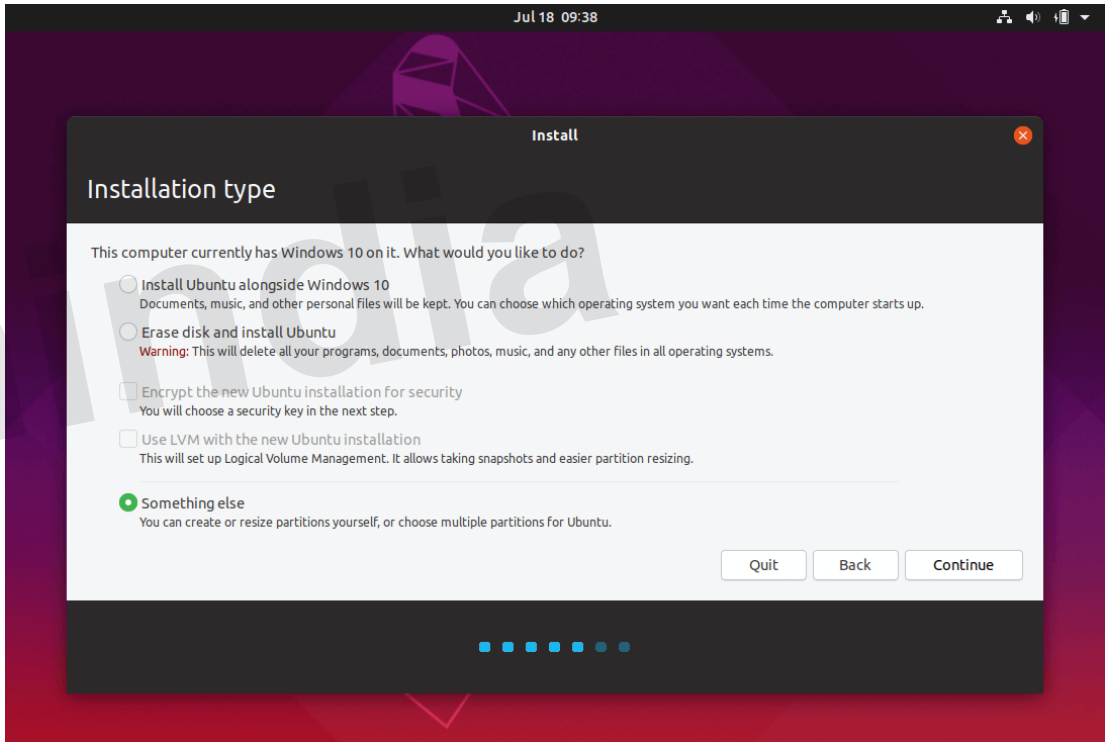


HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

7. Now it's time to select an Installation Type.
You can choose to **Install Ubuntu** alongside **Windows Boot Manager**, an option that will automatically take care of all the partition steps.

Use this option if you don't require a personalized partition scheme. In case you want a custom partition layout, check the **Something else** option and hit on the **Continue** button to proceed further.

The option **Erase disk** and install Ubuntu should be avoided on dual-boot because is potentially dangerous and will wipe out your disk.



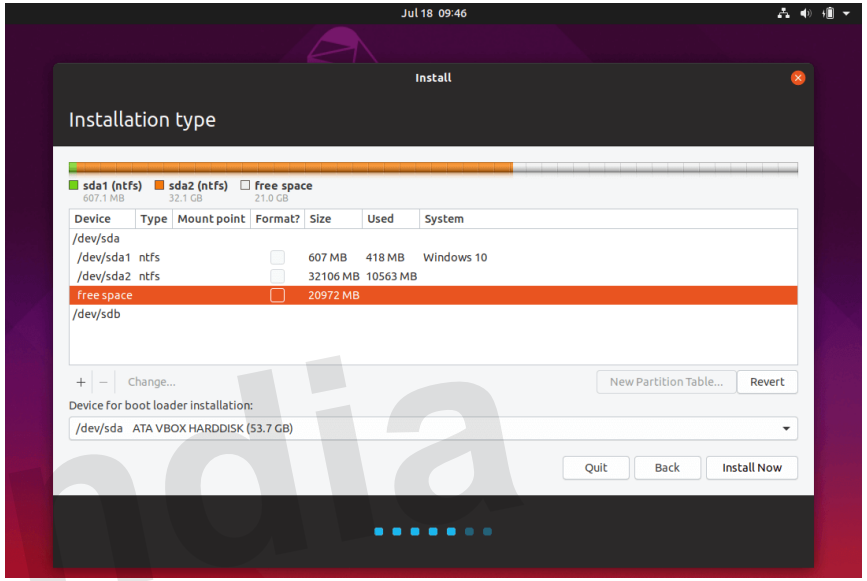
HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

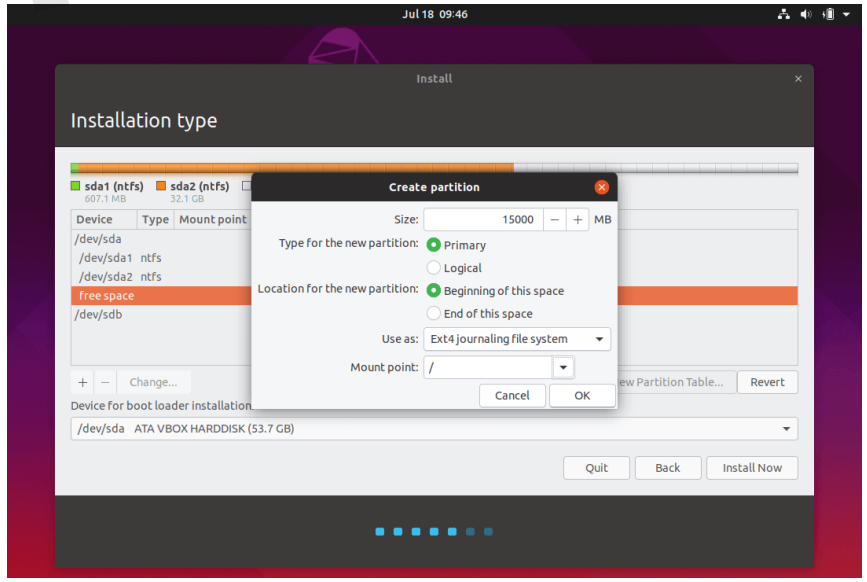
8. On this step, we'll create our custom partition layout for **Ubuntu**. This guide will recommend that you create two partitions, one for root and the other for home accounts data and no partition for swap (use a swap partition only if you have limited RAM resources or you use a fast SSD).

To create the first partition, the root partition, select the free space (the shrinking space from Windows created earlier) and hit on the + icon below.

On partition settings use the following configurations and hit **OK** to apply changes:

- Size = at least **20000** MB
- Type for the new partition = **Primary**
- Location for the new partition = **Beginning**
- Use as = **EXT4** journaling file system
- Mount point = **/**



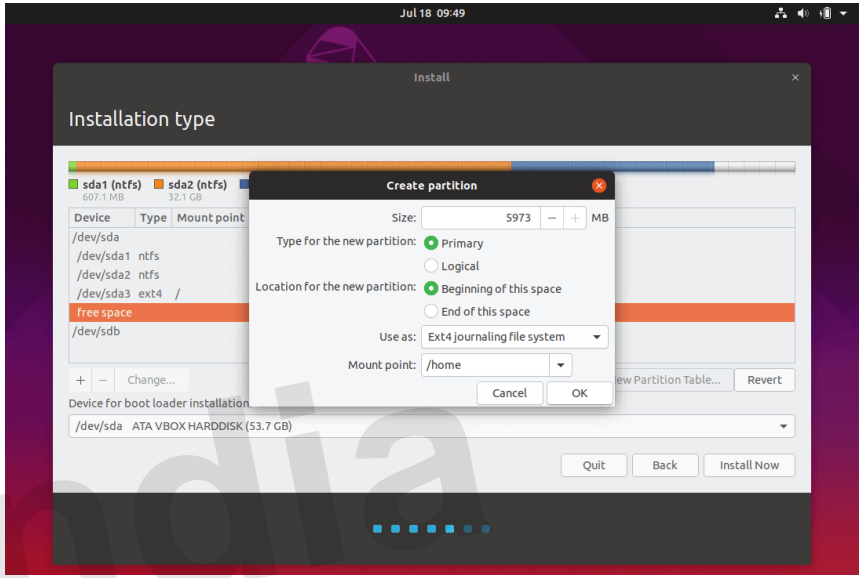


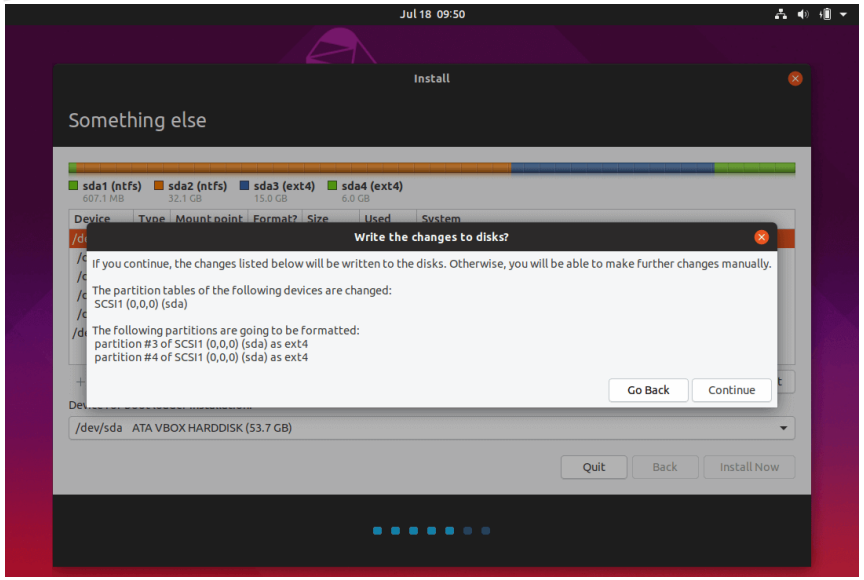
HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

Create the home partition using the same steps as above. Use all the available free space left for the home partition size. The partition settings should look like this:

- Size = all remaining free space
- Type for the new partition = **Primary**
- Location for the new partition = **Beginning**
- Use as = **EXT4** journaling file system
- Mount point = **/home**

9. When finished, hit the **Install Now** button in order to apply changes to disk and start the installation process.
A pop-up window should appear to inform you about **swap** space. Ignore the alert by pressing on the **Continue** button.
Next, a new pop-up window will ask you if you agree with committing changes to disk.
Hit **Continue** to write changes to disk and the installation process will now start.

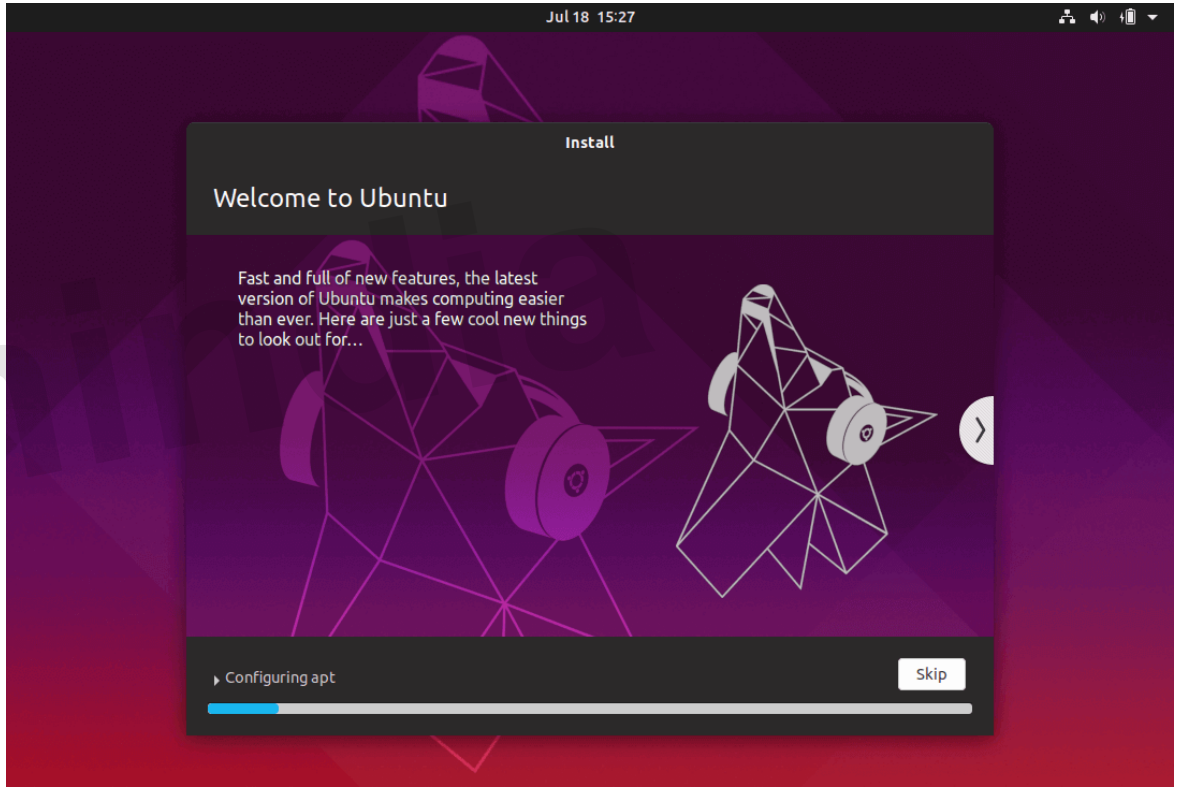




HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

10. On the next screen adjust your machine physical location by selecting a city nearby from the map. When done hit **Continue** to move ahead.

11. Pick up a username and password for your administrative **sudo** account, enter a descriptive name for your computer and hit **Continue** to finalize the installation. These are all the settings required for customizing the **Ubuntu** installation. From here on the installation process will run automatically until it reaches the end.

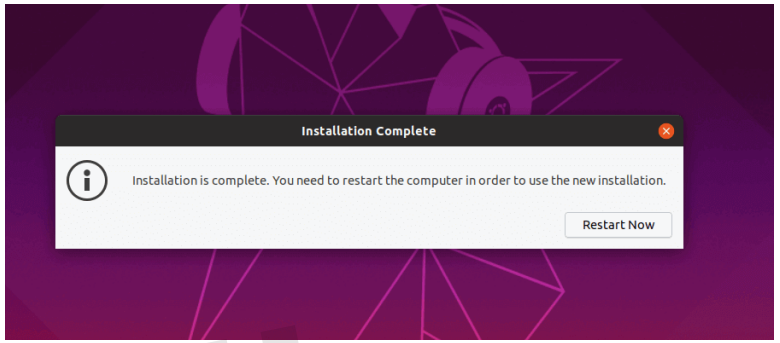


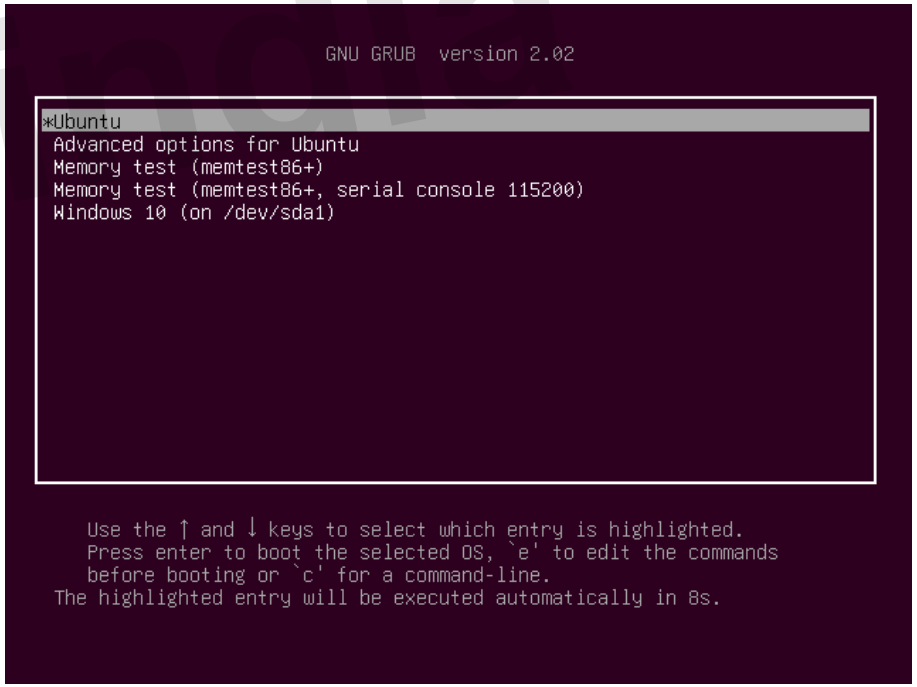
HOW TO INSTALL UBUNTU ALONGSIDE WITH WINDOWS 10 OR 8 IN DUAL-BOOT CONTINUED...

12. After the installation process reaches its end hit on the **Restart Now** button in order to complete the installation.

The machine will reboot into the **Grub** menu, where for ten seconds, you will be presented to choose what OS you wish to use further: **Ubuntu 20.04** or **Microsoft Windows**.

Ubuntu is designated as default OS to boot from. Thus, just press **Enter** key or wait for those **10** seconds timeout to drain.





USEFUL LINKS

- https://phoenixnap.com/kb/install-ubuntu-20-04
- https://www.tecmint.com/install-ubuntu-20-04-desktop/
- https://www.tecmint.com/install-ubuntu-alongside-with-windows-dual-boot/

VIDEOS:

- https://www.youtube.com/watch?v=Z-Hv9hOaKso
- https://www.youtube.com/watch?v=G7ffzC4S0A4&t=133s

ONLINE DESIGN CHALLENGE

Platform Creation Guide

USEFUL LINKS

Gazebo is a full-fledged simulation environment that is compatible with PX4. In 2021, NXP Cup contestants may use the Gazebo simulation for an extra challenge at home. The benefit to using the Gazebo simulation environment is that you can test your code without crashing or damaging your actual NXP Cup car. The code modules that you run on your actual NXP Cup car can be ported to the Gazebo simulation environment with ease - and if you're using a brushless Cup car, you'll essentially be running the same exact code.

PREREQUISITES:

To run Gazebo you will need an Ubuntu Linux (20.04) environment. You can run the Ubuntu as a native OS or as a virtual machine. Recommended specs for running the Gazebo simulation environment are as follows:

Component	Amount
Processor	Any modern quad-core processor or better
RAM	8GB of RAM or better
Hard Disk Space	~20GB of space (SSD recommended)

INSTALLATION OF NXP GAZEBO

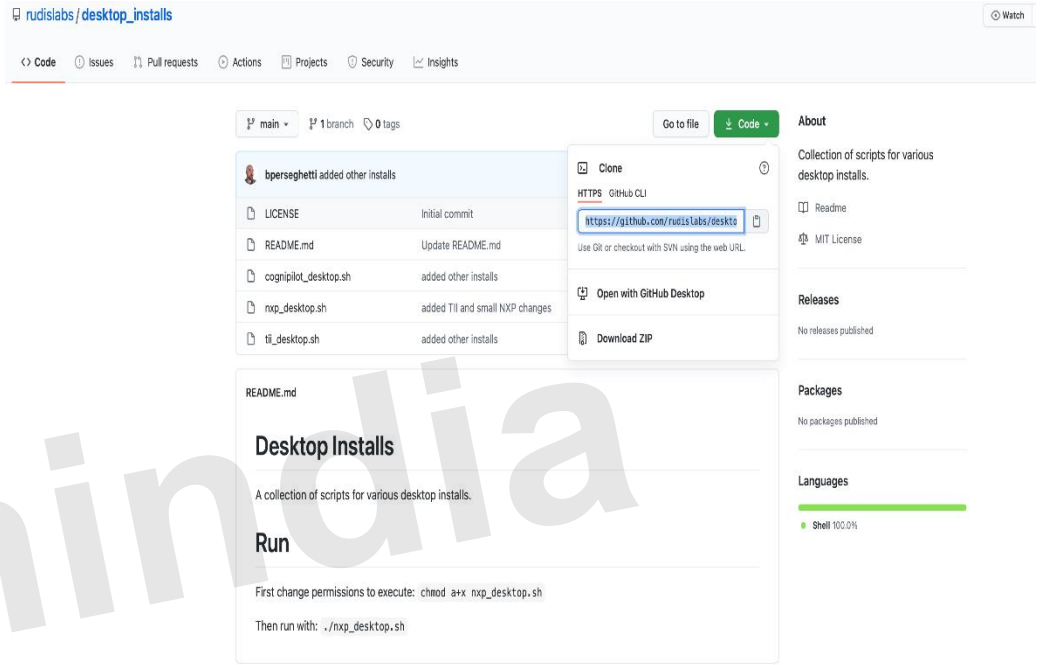
Setting up your Ubuntu desktop environment

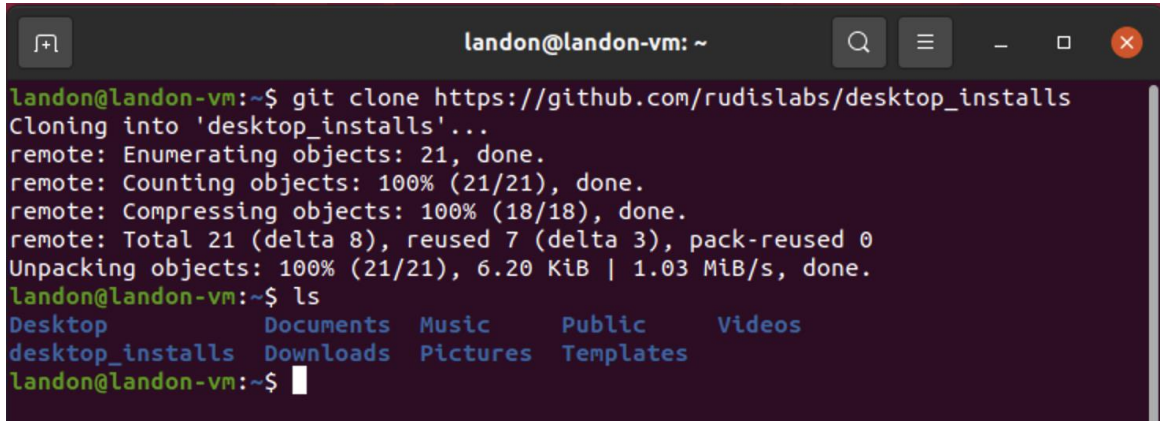
You can install by using nxp_desktop.sh shell by downloading the file at https://github.com/rudislabs/desktop_installs

You can do so easily by running the following git clone command using the link from the git repo:

$ git clone https://github.com/rudislabs/desktop_installs

Now you should see the desktop_installs folder in your home folder





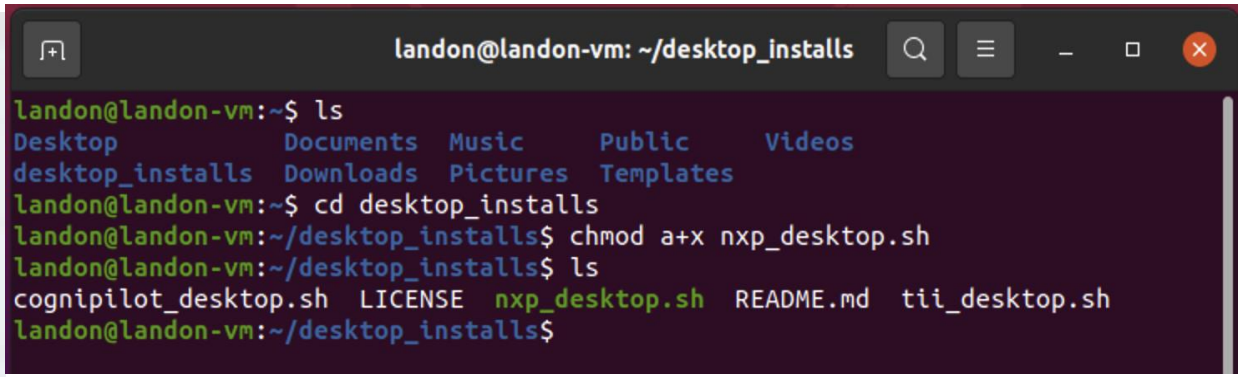
After downloading the file you may need to make it executable. You can do so by entering the *desktop_installs* directory and running a *chmod* command:

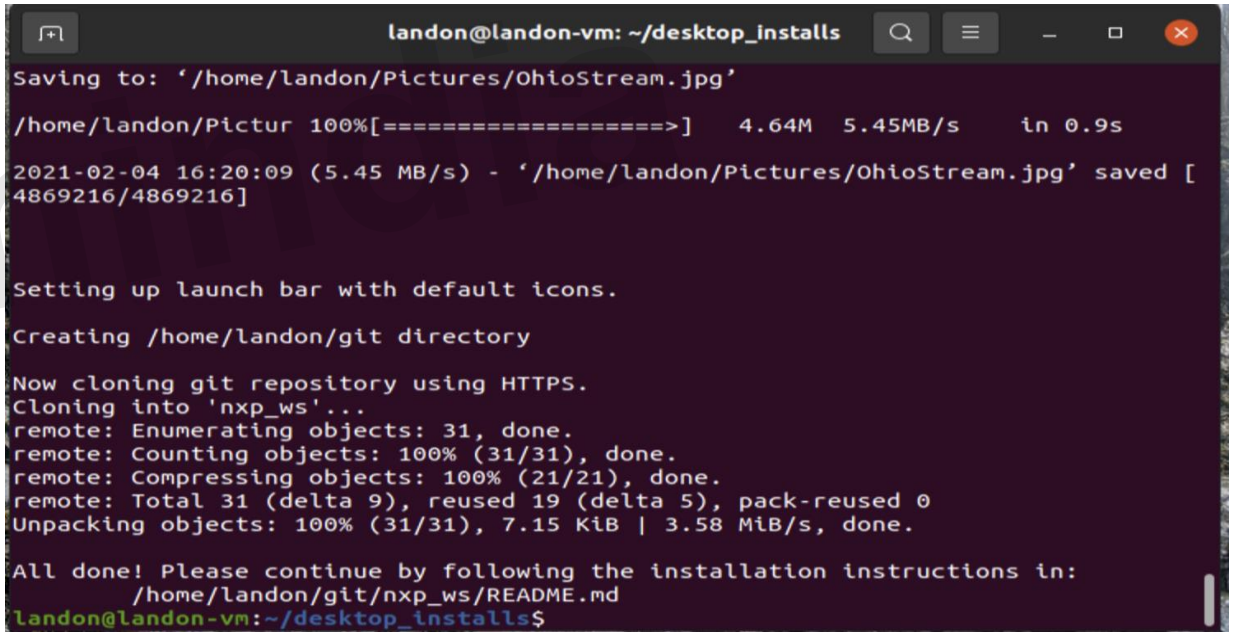
$ cd desktop_installs
$ chmod a+x nxp_desktop.sh
$ ls

Now you should see that the *nxp_desktop.sh* file is highlighted green. This means that it has executable permissions

Now, we want to run this file to set up our desktop environment. It will ask you for your Ubuntu user password; just type that in when it asks for it. To run the file, run the following command and you should get the expected output

$ ./nxp_desktop.sh





You should now see that your background has changed and the command has given you the ***All done!*** signal.
Great! Now let's run the workspace setup to get Gazebo, PX4, and ROS installed.

SETTING UP NXP CUP SIMULATION SOFTWARE

Running the ROS2 Foxy install script

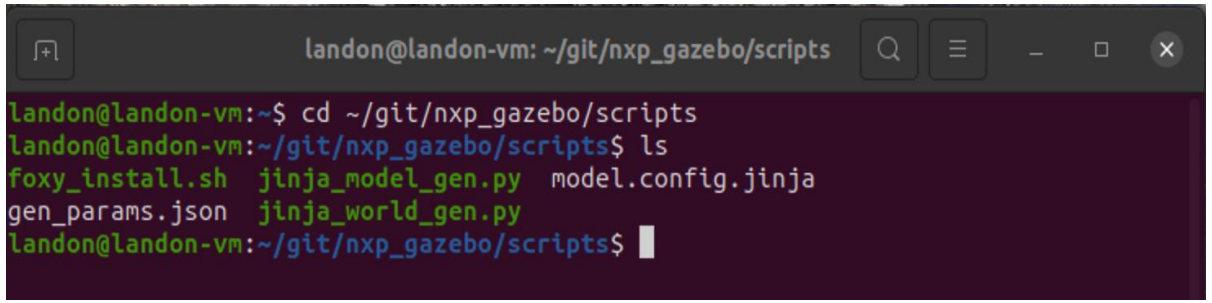
First, we will need to navigate to the new workspace folder created by the nxp_desktop.sh install script. The workspace folder is at *~/git/*.
Inside of that workspace folder, there is a script that will set up ROS2 and all necessary dependecies for you. That install script is located at *~/git/nxp_gazebo/scripts/*. To get there, run the following commands:

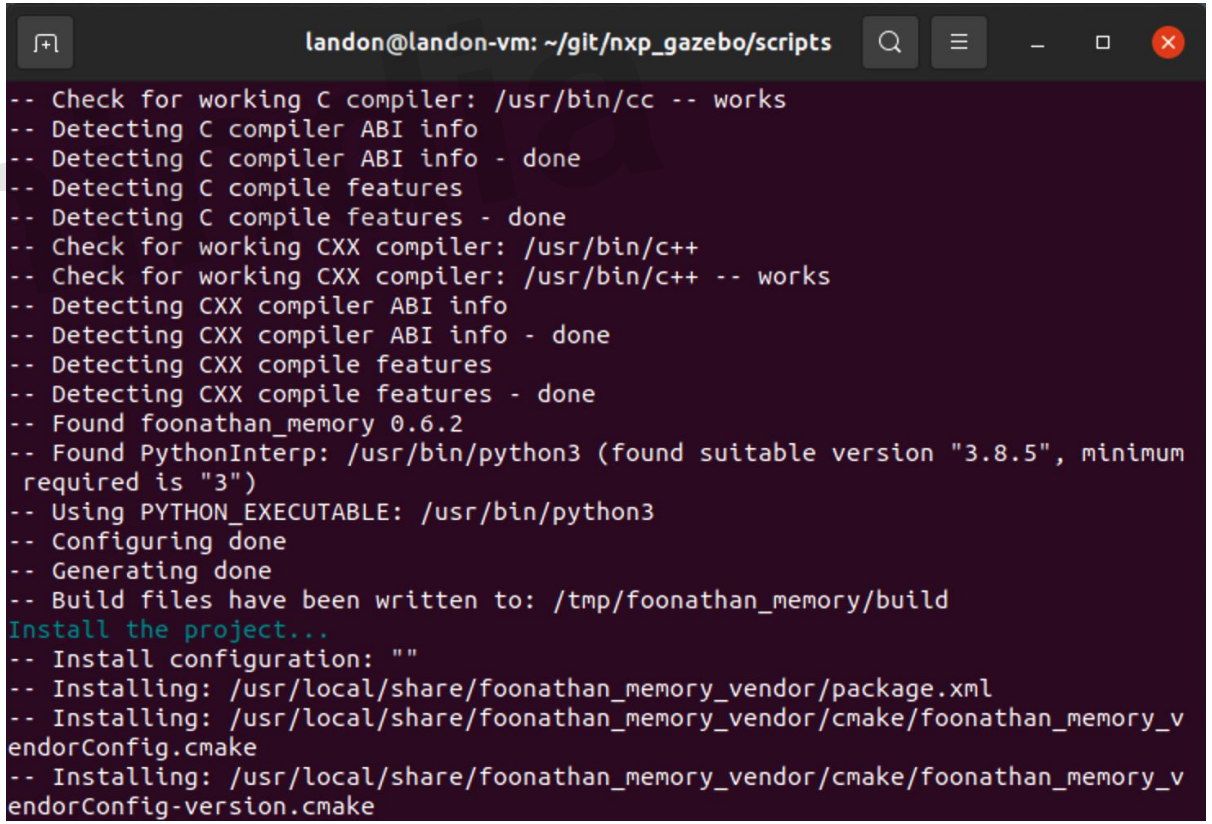
$ cd ~/git/nxp_gazebo/scripts
$ ls

Now, we will need to run the *foxy_install.sh* file to set up our software. To start, run the following command and make sure you get the expected output:

$ ./foxy_install.sh

Note: This command will take a long time to finish since it is installing ROS2.





SETTING UP NXP CUP SIMULATION SOFTWARE

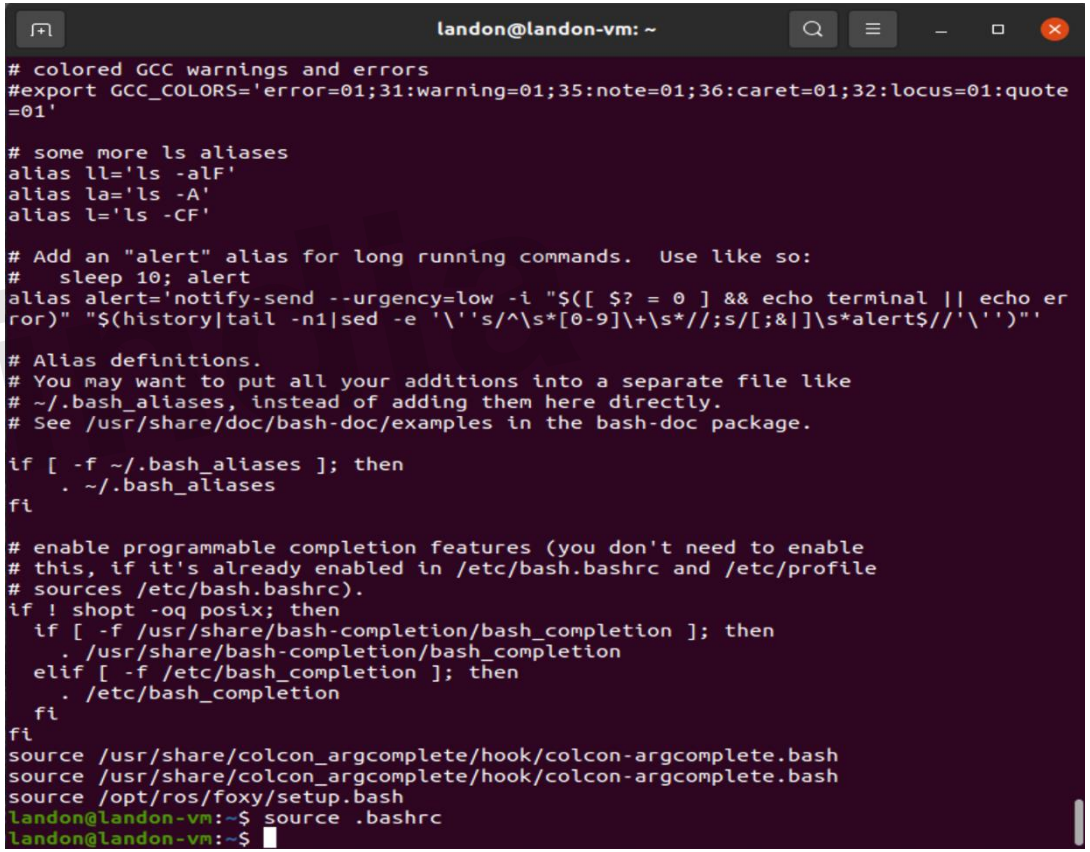
Running the ROS2 Foxy install script continued…

Once you get the expected output from ./foxy_install.sh, you'll want to add a line to your ~/.bashrc. Follow the instructions below:

$ nano ~/.bashrc
[go to end of file]
Add: source /opt/ros/foxy/setup.bash
[Press ctrl+x, type Y, press enter]

Then, run the following commands and make sure you get the expected output:

$ cat ~/.bashrc
$ source ~/.bashrc



SETTING UP NXP CUP SIMULATION SOFTWARE

Installing RTPS

In order to transfer simulated Pixy camera data from ROS2 to simulated PX4, we need to install some software.

Installing Fast-RTPS (DDS)

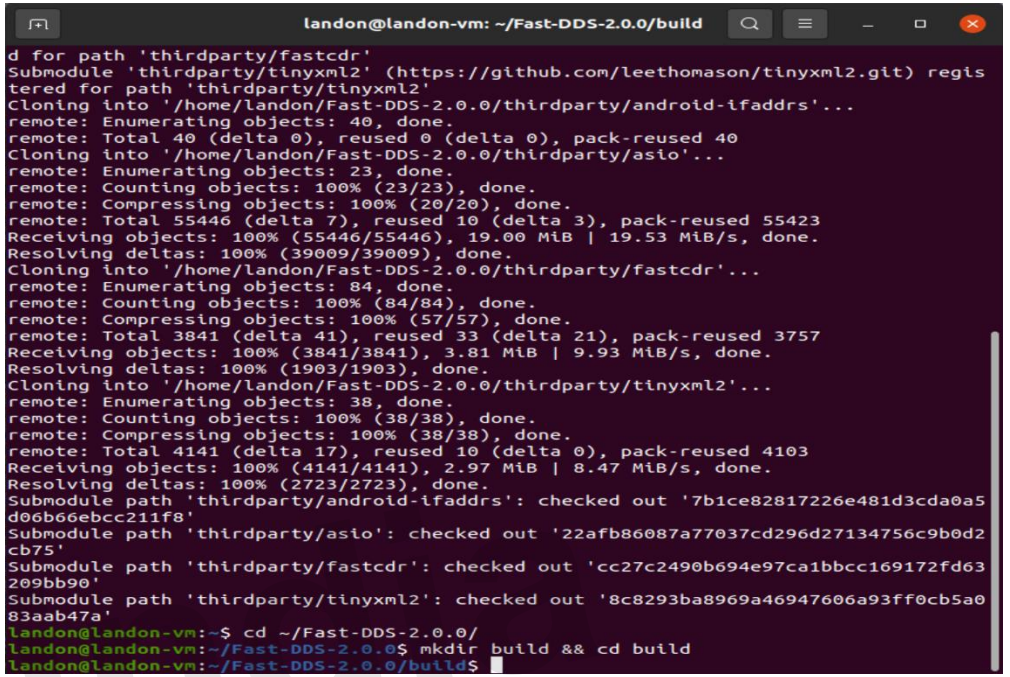
Clone the project from Github:

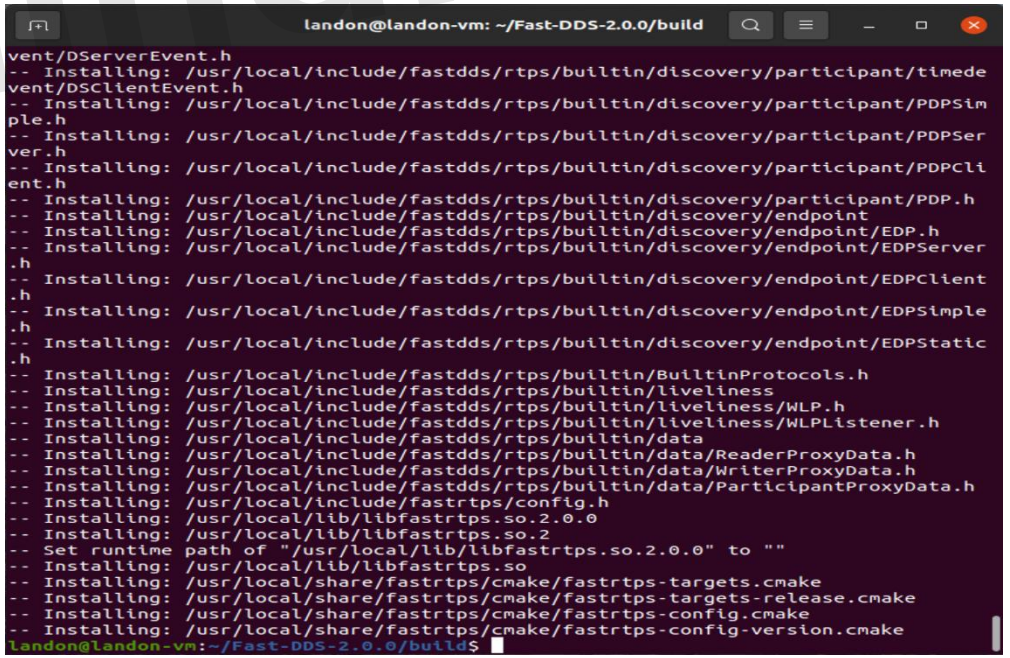
```
$ git clone --recursive https://github.com/eProsima/Fast-DDS.git -b v2.0.0 ~/FastDDS-2.0.0
$ cd ~/FastDDS-2.0.0
$ mkdir build && cd build
```

Next, we will run some commands to build and install Fast-DDS. Run the following commands:

```
$ cmake -DTHIRDPARTY=ON -DSECURITY=ON ..
$ make -j$(nproc --all)
$ sudo make install
```

You should get a message stating *[100%] Built target fastrtps* once make *-j$(nproc --all)* is finished, and you should get the expected output shown after you run *sudo make install*:





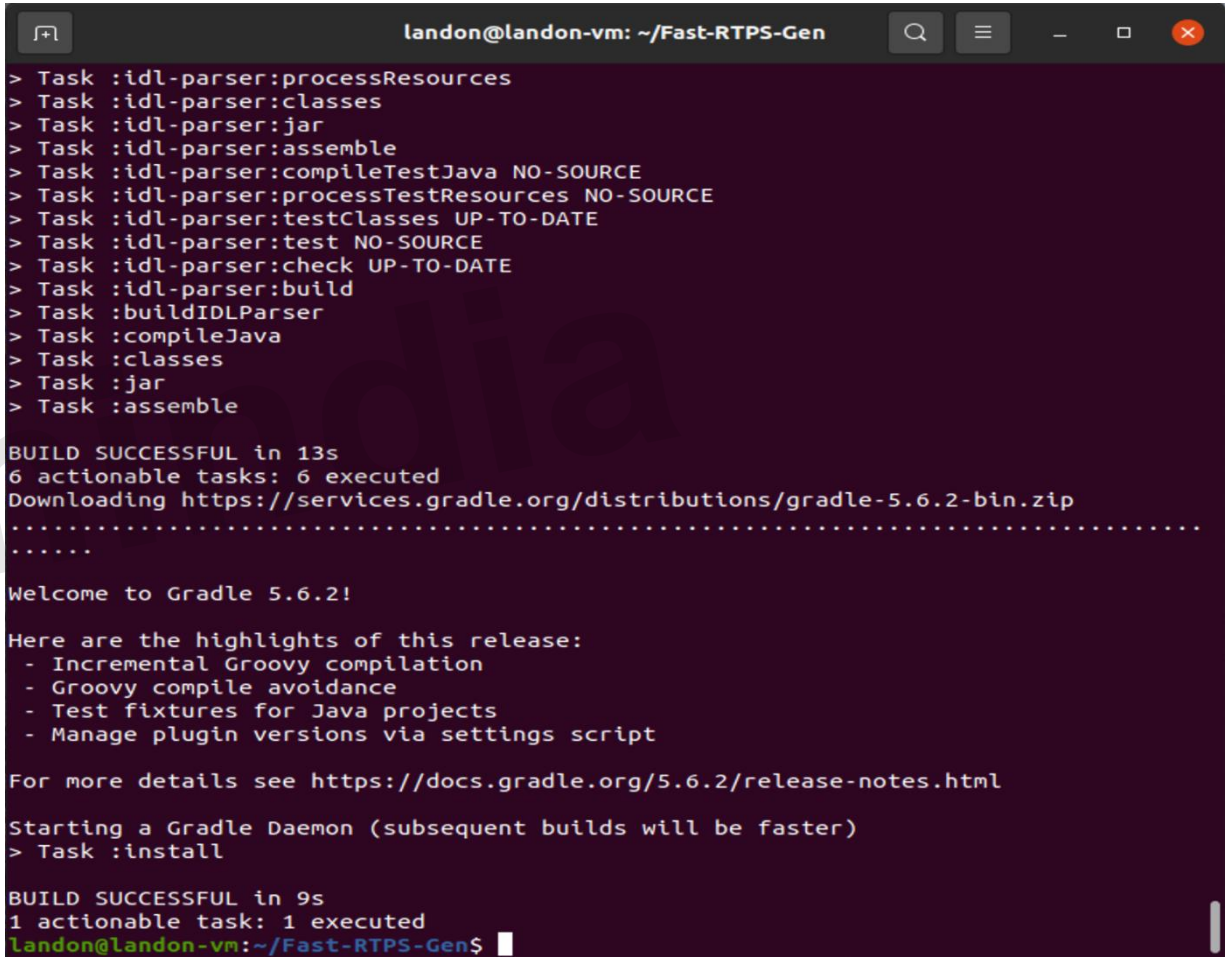
SETTING UP NXP CUP SIMULATION SOFTWARE

Installing Fast-RTPS-Gen

Next, we will install Fast-RTPS-Gen. To do so, run the following command and make sure you get the expected output:

```
$ git clone --recursive https://github.com/eProsima/Fast-DDS-Gen.git -b v1.0.4 ~/Fast-RTPS-Gen \    && cd ~/Fast-RTPS-Gen \    && ./gradlew assemble \    && sudo ./gradlew install
```

If you got the expected output, great! We will move on to building our ROS2 workspace.



SETTING UP NXP CUP SIMULATION SOFTWARE

Building the ROS2 workspace

Now, we will want to navigate to the ROS2 workspace and build all of the packages inside it. First, we will change directories to the *~/git/nxp_ros2_ws/* and then run some build commands. Run the commands below and make sure that you get the expected output:

$ colcon build --packages-select px4_msgs --symlink-install
$ colcon build --packages-select px4_ros_com --symlink-install
$ colcon build --packages-select nxp_cup_vision --symlink-install
$ colcon build --packages-select nxp_cup_bringup --symlink-install

```
landon@landon-vm: ~/git/nxp_ros2_ws
landon@landon-vm:~/git/nxp_ros2_ws$ colcon build --packages-select px4_msgs --symlink-install
Starting >>> px4_msgs
[Processing: px4_msgs]
[Processing: px4_msgs]
[Processing: px4_msgs]
[Processing: px4_msgs]
[Processing: px4_msgs]
[Processing: px4_msgs]
Finished <<< px4_msgs [3min 11s]

Summary: 1 package finished [3min 11s]
landon@landon-vm:~/git/nxp_ros2_ws$
```

```
landon@landon-vm: ~/git/nxp_ros2_ws/log/build_2021-02-24_12-15-27
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.20.04, mixed mode)
openjdk version "11.0.10" 2021-01-19
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.20.04, mixed mode)
openjdk version "11.0.10" 2021-01-19
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.20.04, mixed mode)
openjdk version "11.0.10" 2021-01-19
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.20.04, mixed mode)
---
Finished <<< px4_ros_com [2min 28s]

Summary: 1 package finished [2min 28s]
  1 package had stderr output: px4_ros_com
landon@landon-vm:~/git/nxp_ros2_ws$ ls
```

```
landon@landon-vm:~/git/nxp_ros2_ws$ colcon build --packages-select nxp_cup_vision --symlin
k-install
Starting >>> nxp_cup_vision
Finished <<< nxp_cup_vision [0.35s]

Summary: 1 package finished [0.40s]
landon@landon-vm:~/git/nxp_ros2_ws$ colcon build --packages-select nxp_cup_bringup --symli
nk-install
Starting >>> nxp_cup_bringup
Finished <<< nxp_cup_bringup [0.52s]

Summary: 1 package finished [0.57s]
landon@landon-vm:~/git/nxp_ros2_ws$
```

SETTING UP NXP CUP SIMULATION SOFTWARE

Once we have built all of our ROS2 workspace, we will want to source the setup.bash for the workspace. We also want to make sure that all of our library paths are properly sourced. To do so, we will add some lines to the end of our *~/.bashrc.* Add the following lines to the end of your *~/.bashrc*:

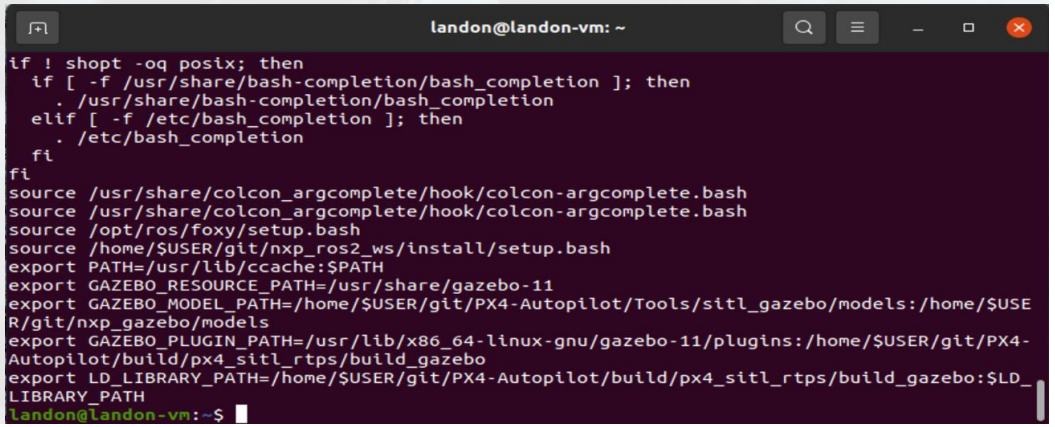
```
source /home/$USER/git/nxp_ros2_ws/install/setup.bash
export PATH=/usr/lib/ccache:$PATH
export GAZEBO_RESOURCE_PATH=/usr/share/gazebo-11
export GAZEBO_MODEL_PATH=/home/$USER/git/PX4-Autopilot/Tools/sitl_gazebo/models:/home/$USER/git/nxp_gazebo/models
export GAZEBO_PLUGIN_PATH=/usr/lib/x86_64-linux-gnu/gazebo-11/plugins:/home/$USER/git/PX4-Autopilot/build/px4_sitl_rtps/build_gazebo
export LD_LIBRARY_PATH=/home/$USER/git/PX4-Autopilot/build/px4_sitl_rtps/build_gazebo:$LD_LIBRARY_PATH
```

You can do so with *nano* or *vim*. You can follow the nano instructions from earlier in the guide if you need help doing so. To make sure your ~/.bashrc looks correct, run the following command and then check if you get the expected output:

```
$ cat ~/.bashrc
```

Now, you'll want to source that ~/.bashrc by running the following command:

```
$ source ~/.bashrc
```



SETTING UP NXP CUP SIMULATION SOFTWARE

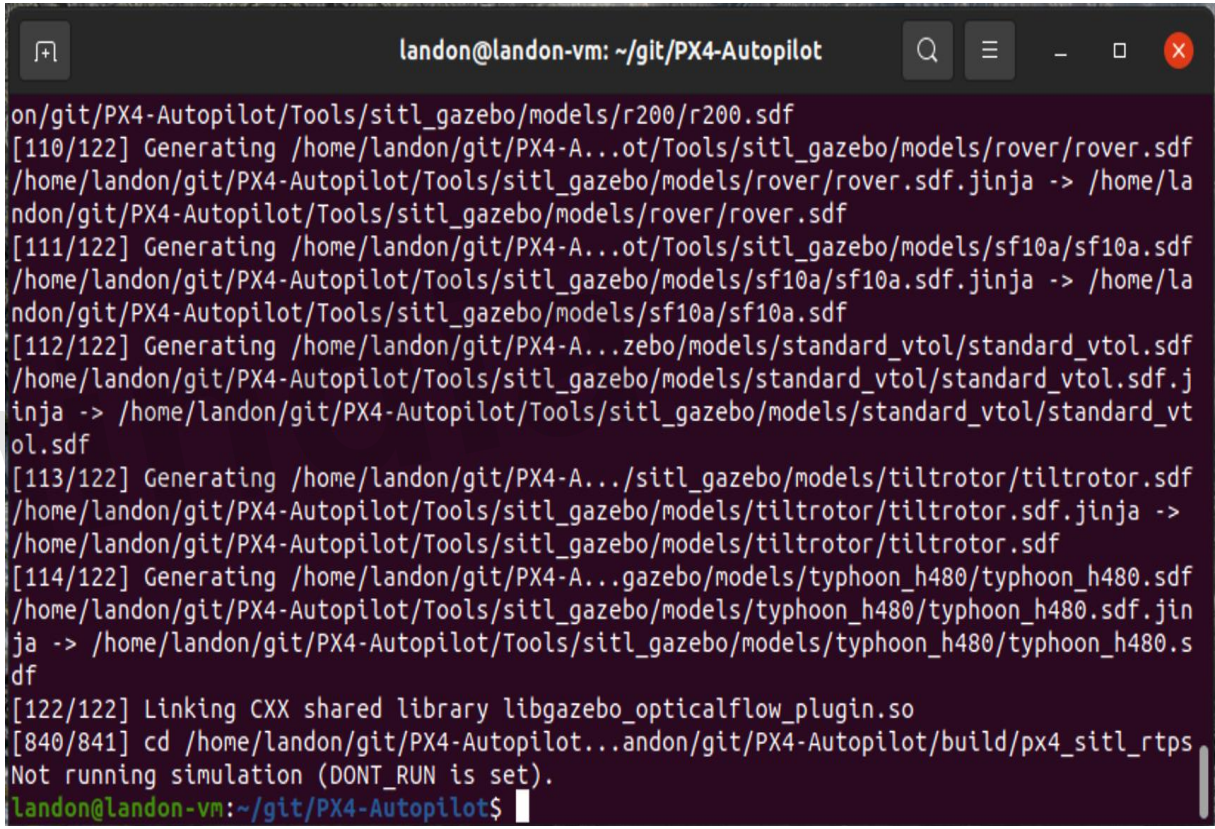
Building the PX4 binary

The PX4 source code is located at ~/git/PX4-Autopilot. To build the PX4 source, we will want to change directories and run the build command for *px4_sitl_rtps gazebo*. This will give the ROS2 workspace a PX4 binary to run for our simulated NXP Cup car. Run the following commands to build PX4 and make sure you get the expected output:

```
$ cd ~/git/PX4-Autopilot
$ DONT_RUN=1 make px4_sitl_rtps gazebo
```

If you get the expected output, stay vigilant! We only have one more command to run. We need to install *xterm* for the PX4 shell that runs when we boot up the simulation. To do so, run the following command:

```
$ sudo apt install xterm
```



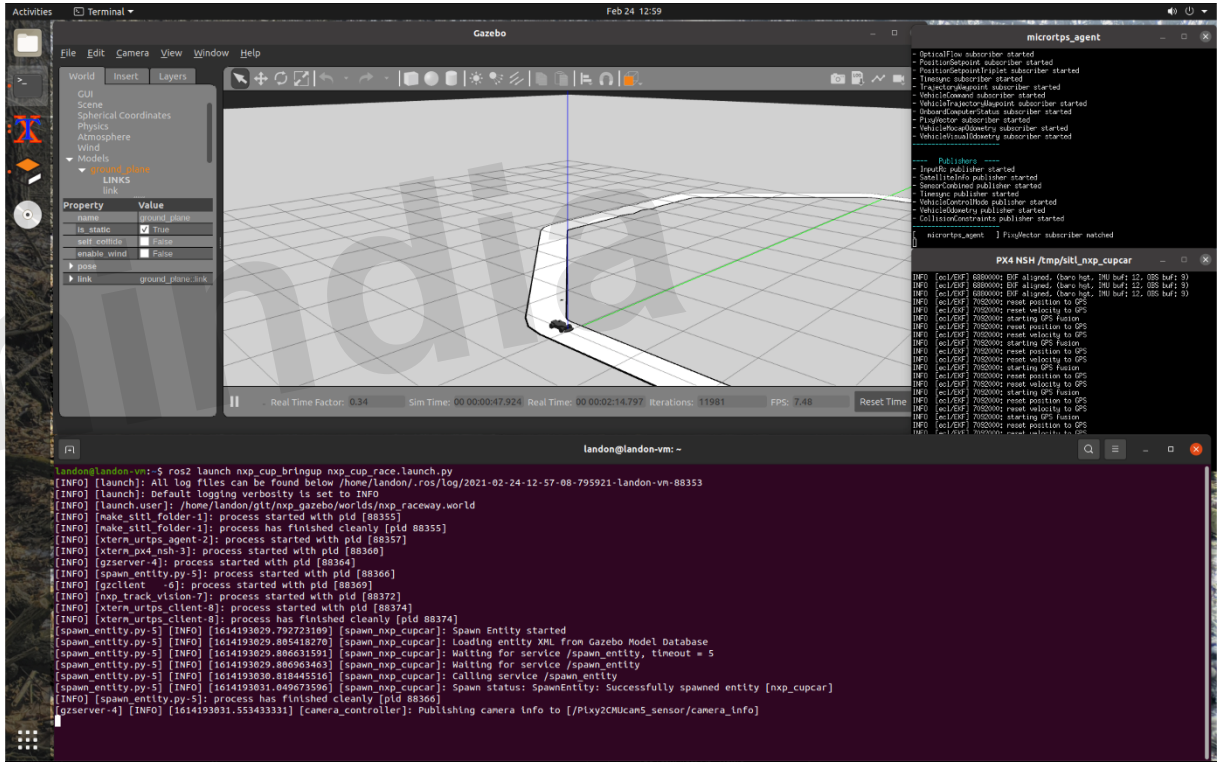
Once you have done that, you're finished setting up the NXP Cup simulation!

RUNNING THE GAZEBO SIMULATION

To run the Gazebo simulation, we suggest you first restart your VM or PC and/or log out and log back in. Once you've done so, you can run the following command to start the simulation stack and check to see if you get the expected result:

```
$ ros2 launch nxp_cup_bringup nxp_cup_race.launch.py
```

The black terminal window is your PX4 shell, and the purple terminal window is the ROS console. The PX4 shell works just like a real PX4 shell - meaning that you can run your example programs and watch uORB topics just like you would on the real brushless NXP Cup car.



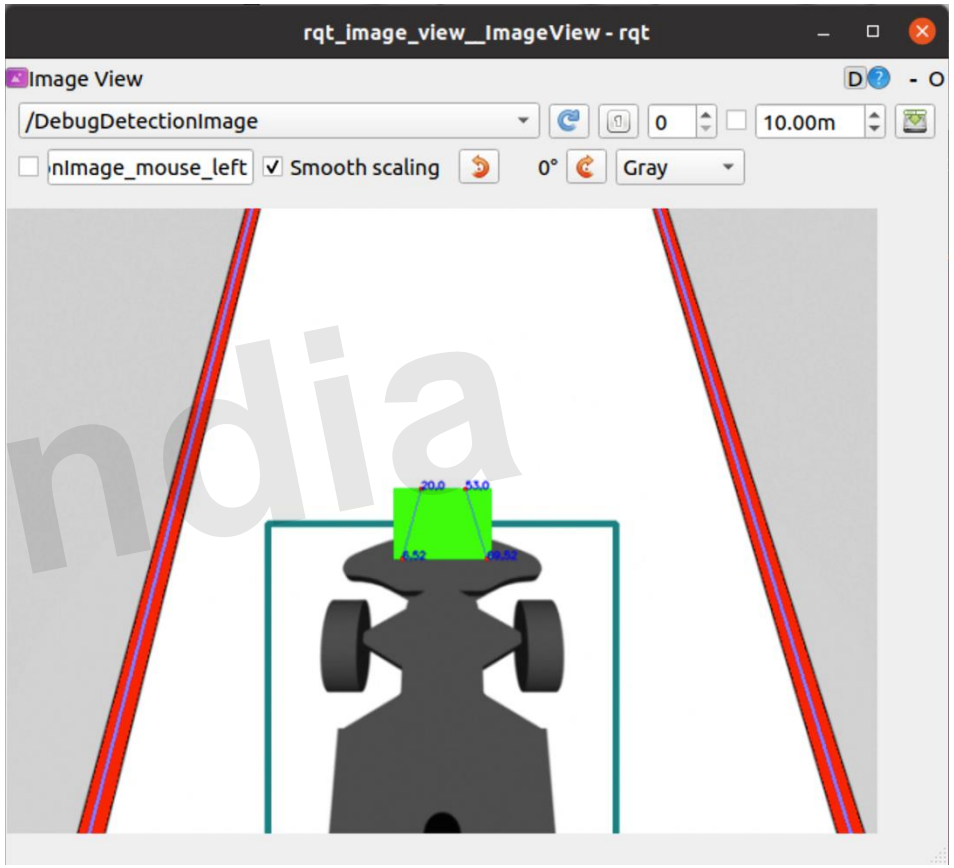
VIEWING THE SIMULATED PIXY CAMERA

Preface
In order to provide a true-to-life simulated environment for NXP Cup contestants, we have written a simulated Pixy camera module that detects lines and outputs vector data just like the real Pixy camera. The source code for the simulated Pixy camera uses OpenCV to fit vectors to detected lines in simulation.

Guide
To view the output of the simulated Pixy camera, you can open a separate tab in your Ubuntu terminal window and run the following command:

```
$ ros2 run rqt_image_view rqt_image_view
```

This will open a new window that shows the debug output of the simulated Pixy camera. Here's what it looks like:

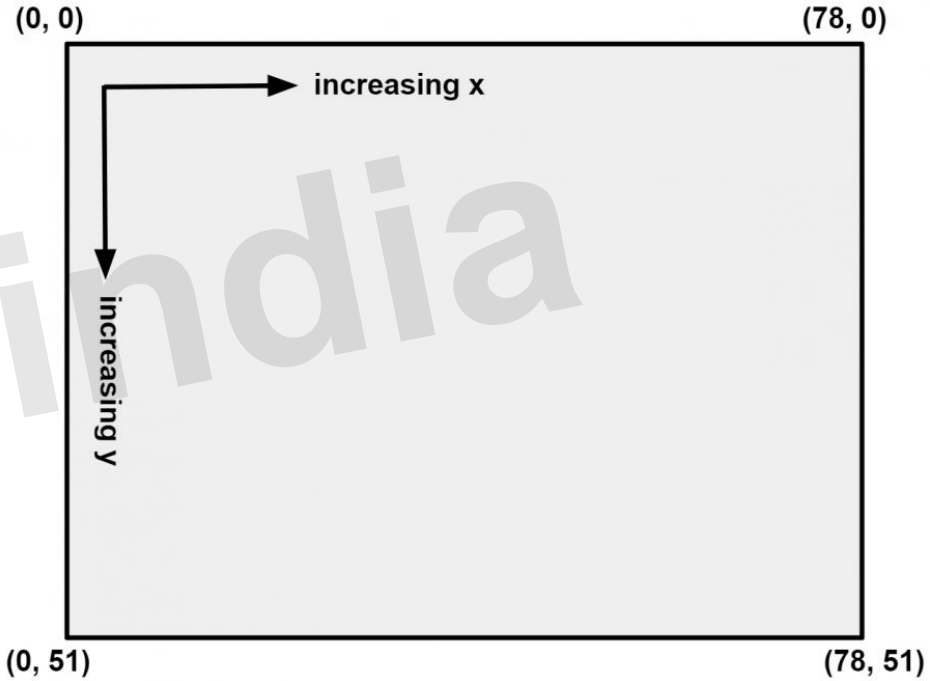


If you do not see the simulated Pixy camera output, use the drop down at the top left of the window and select /DebugDetectionImage.

The simulated Pixy camera detects the black lines in the environment and fits lines to to them. Then, it will use those lines to create a simulated Pixy camera vector output in the Pixy camera frame space as seen :

As you can see in the simulated pixy camera output, the vector data returned is identical to the vector data that the real Pixy camera sends over I2C (vector head and tail coordinates for each vector). This allows contestants to use the same algorithms that are on their real NXP Cup cars.

The source code for this simulated Pixy camera is located at *~/git/nxp_ws/src/nxp_gazebo/scripts/track_vision.py*. You are free to edit this code if you see any potential areas of improvement!



VIEWING THE SIMULATED PIXY CAMERA

To run the example self-driving algorithm, run the following command in the PX4 shell that is opened when you boot up the NXP Gazebo simulation:

pxh> nxpcup start
pxh> commander arm –f

When you run this command, a thread will be activated that should successfully drive the simulated NXP Cup car around the track

When you run the example code (*nxpcup start*) it will start printing data very quickly in the PX4 shell. This is debug information. When you go to run *commander arm -f*, you will not see the text in the shell because the prints are pushing it off the screen too fast. You can still type in the *commander arm -f* command and press enter to run it.

NOTE: The code supplied does not currently drive around the track successfully.

VIEWING THE SIMULATED PIXY CAMERA

NXP Cup Supported Material: https://community.nxp.com/t5/University-Programs-Knowledge/NXP-CUP-2019-20-Support-Material/ta-p/1104379

Pixy 2 Wiki: https://docs.pixycam.com/wiki/doku.php?id=wiki:v2:start

PX4 User Guide: https://docs.px4.io/master/en/

QGroundControl User Guide: https://docs.qgroundcontrol.com/master/en/index.html

PX4 Github: https://github.com/PX4

Gazebo Tutorials: http://gazebosim.org/tutorials
(Important Categories: Build a Robot, Model Editor,Build a World, Plugin, Write a Plugin & Sensors)

NXP PRESENTS

AIM ARTIFICIAL INTELLIGENCE in MOBILITY

IN COLLABORATION WITH

Time Of Sports®

THANK YOU

www.nxpaimindia.com

support@nxpaimindia.com