

ns-3 simulator (Brief Tutorial)

Outline

- Introduction ([slide 3](#))
- Installation ([slide 5](#))
 - Starting with Ubuntu installation ([slide 6](#))
 - Starting with ns-3 installation ([slide 30](#))
- Brief tutorial ([slide 49](#))
- Homework-3 ([slide 64](#))

Introduction

- The goal of this lecture is to experience a simulator
 - There are no annoying coding and debugging assignments (just run the scripts)
- The ns-3 simulator is a discrete-event network simulator
- ns-3 is an open source project
- ns-3 is not an officially supported software product of any company
- ns-3 is designed as a set of libraries that can be combined together and also with other external software libraries
- ns-3 is primarily used on Linux or macOS systems

Resources

- Main web site
 - <https://www.nsnam.org>
- **Tutorial ★**
 - <https://www.nsnam.org/docs/release/3.36/tutorial/html/index.html>
- Manual (Release 3.36)
 - <https://www.nsnam.org/docs/release/3.36/manual/singlehtml/index.html>
- **API documentation ★ ★**
 - <https://www.nsnam.org/docs/release/3.36/doxygen/index.html>
- Git
 - <https://gitlab.com/nsnam/ns-3-dev>
- Google groups (You can check Q&A history)
 - <https://groups.google.com/g/ns-3-users>
- TA email
 - jtzlbsj4a@naver.com

Installation

- Ubuntu installation for Windows users
 - Those who are ready for the Ubuntu environment can move on to slide 34
 - 👉 For Mac users, refer <https://www.nsnam.org/wiki/Installation#macOS>
- Ubuntu
 - One of the popular Linux distributions
 - <https://releases.ubuntu.com/20.04/>
- VirtualBox
 - Powerful tool that can virtualize almost any operating system (OS)
 - <https://www.virtualbox.org/wiki/Downloads>

Installation

- Ubuntu 20.04 <https://releases.ubuntu.com/20.04/>
 - Download Ubuntu 20.04 desktop image
 - It takes up 3.1GB of memory and takes a long time (5+ minutes) to download

← → ↻ releases.ubuntu.com/20.04/ 🔒 ☆ 🌐 ⚙️

Select an image

Ubuntu is distributed on three types of images described below.

Desktop image

The desktop image allows you to try Ubuntu without changing your computer at all, and at your option to install it permanently later. This type of image is what most people will want to use. You will need at least 2048MiB of RAM to install from this image.

64-bit PC (AMD64) desktop image

Choose this if you have a computer based on the AMD64 or EM64T architecture (e.g., Athlon64, Opteron, EM64T Xeon, Core 2). Choose this if you are at all unsure.

Server install image

The server install image allows you to install Ubuntu permanently on a computer for use as a server. It will not install a graphical user interface.

64-bit PC (AMD64) server install image

Choose this if you have a computer based on the AMD64 or EM64T architecture (e.g., Athlon64, Opteron, EM64T Xeon, Core 2). Choose this if you are at all unsure.

A full list of available files, including [BitTorrent](#) files, can be found below.

If you need help burning these images to disk, see the [Image Burning Guide](#).

Name

Last modified

Size

Description

Installation

- VirtualBox <https://www.virtualbox.org/wiki/Downloads/>
- Download VirtualBox for Windows hosts



VirtualBox

Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

If you're looking for the latest VirtualBox 6.1 packages, see [VirtualBox 6.1 builds](#). Version 6.1 will remain supported until December 2023.

VirtualBox 7.0.8 platform packages

- [Windows hosts](#)
- [macOS / Intel hosts](#)
- [Developer preview for macOS / Arm64 \(M1/M2\) hosts](#)
- [Linux distributions](#)
- [Solaris hosts](#)
- [Solaris 11 IPS hosts](#)

The binaries are released under the terms of the GPL version 3.

See the [changelog](#) for what has changed.

You might want to compare the checksums to verify the integrity of downloaded packages. *The SHA256 checksums should be favored as the MD5 alg*

- [SHA256 checksums](#), [MD5 checksums](#)

Note: After upgrading VirtualBox it is recommended to upgrade the guest additions as well.

[About](#)
[Screenshots](#)
[Downloads](#)
[Documentation](#)
 [End-user docs](#)
 [Technical docs](#)
[Contribute](#)
[Community](#)

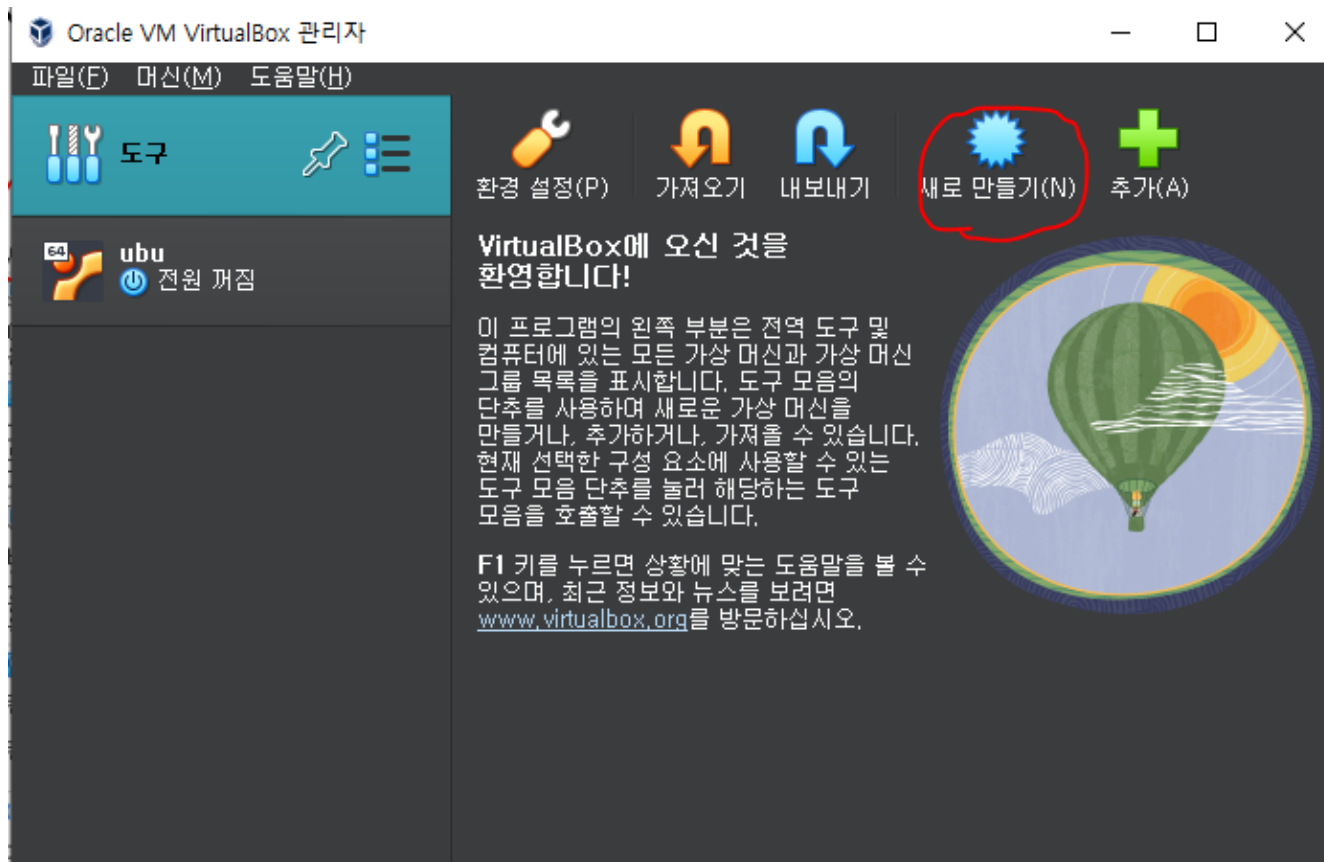
Installation

- VirtualBox
 - Executes the VirtualBox setup file
 - 'Press Next' is all you need until the successful installation



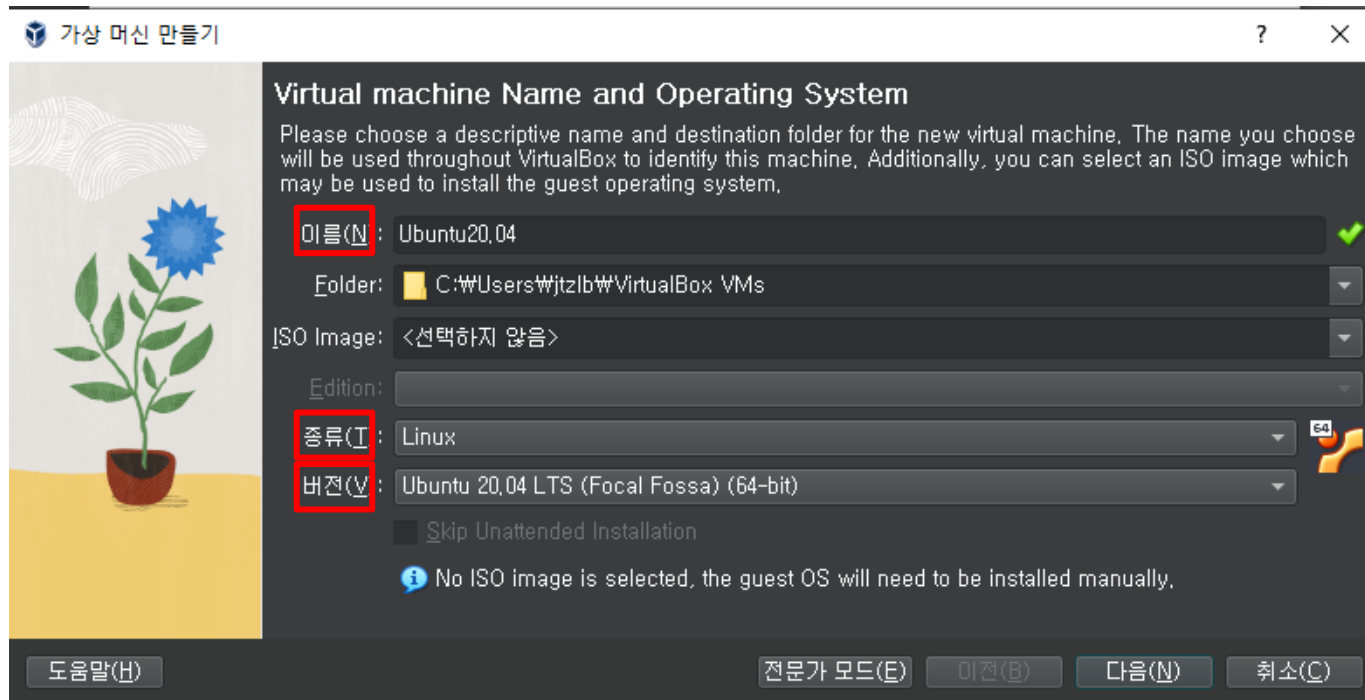
Installation

- VirtualBox setting
 - Executes the VirtualBox after the installation
 - Then, press the New (새로 만들기) button



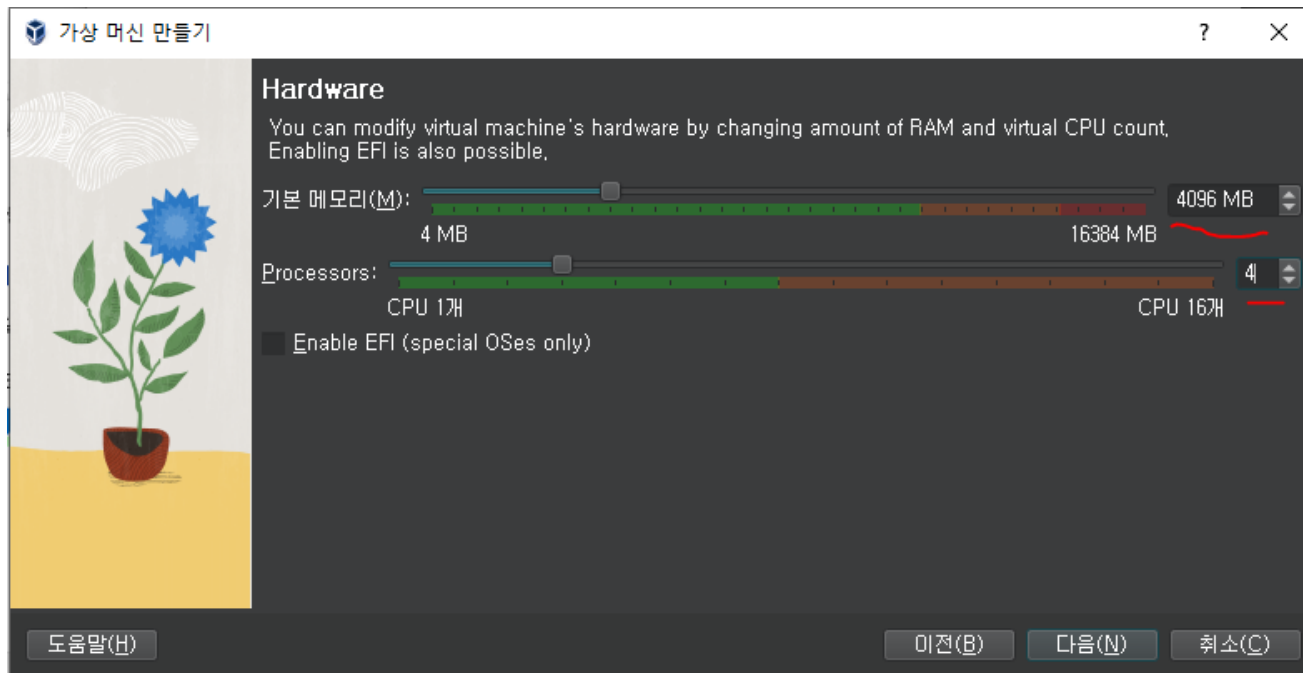
Installation

- VirtualBox setting
 - Type anything you want in the name field
 - Select Linux and version Ubuntu 20.04 LTS
 - Press the Next button



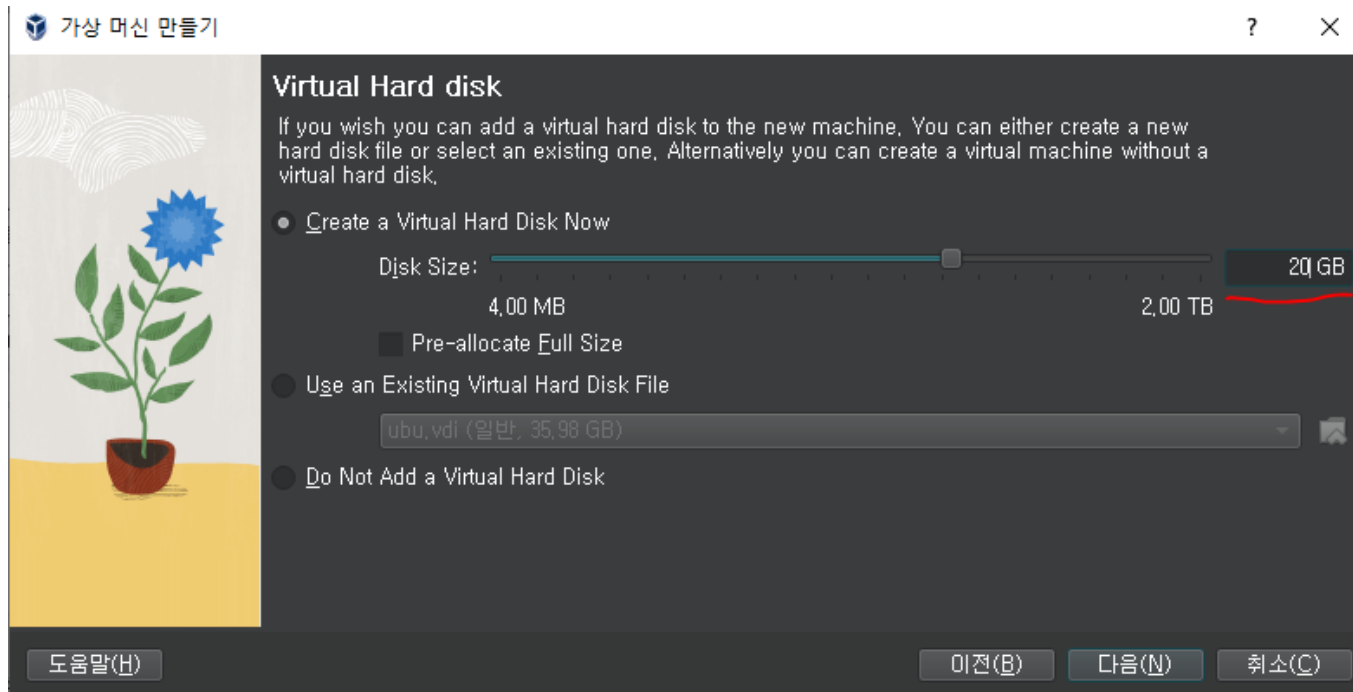
Installation

- VirtualBox setting
 - Allocate sufficient RAM and processors for the virtual machine
 - If you give it less, it will slow down
 - Press the Next button



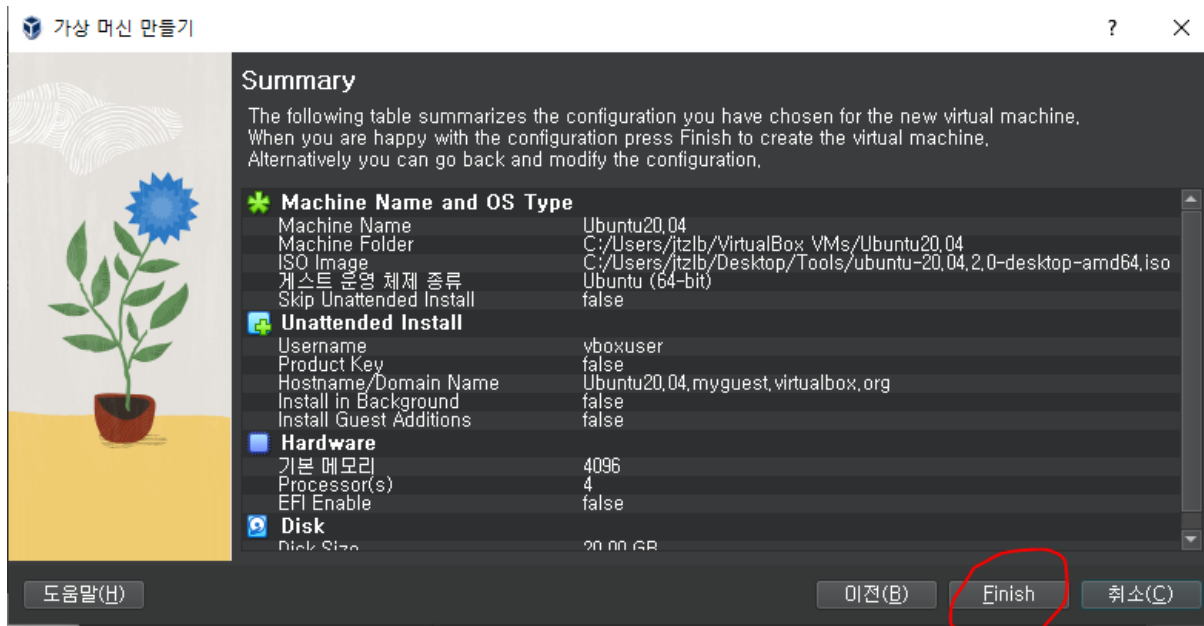
Installation

- VirtualBox setting
 - Allocate sufficient memory for the virtual machine, then press the Next button



Installation

- VirtualBox setting
 - Finish



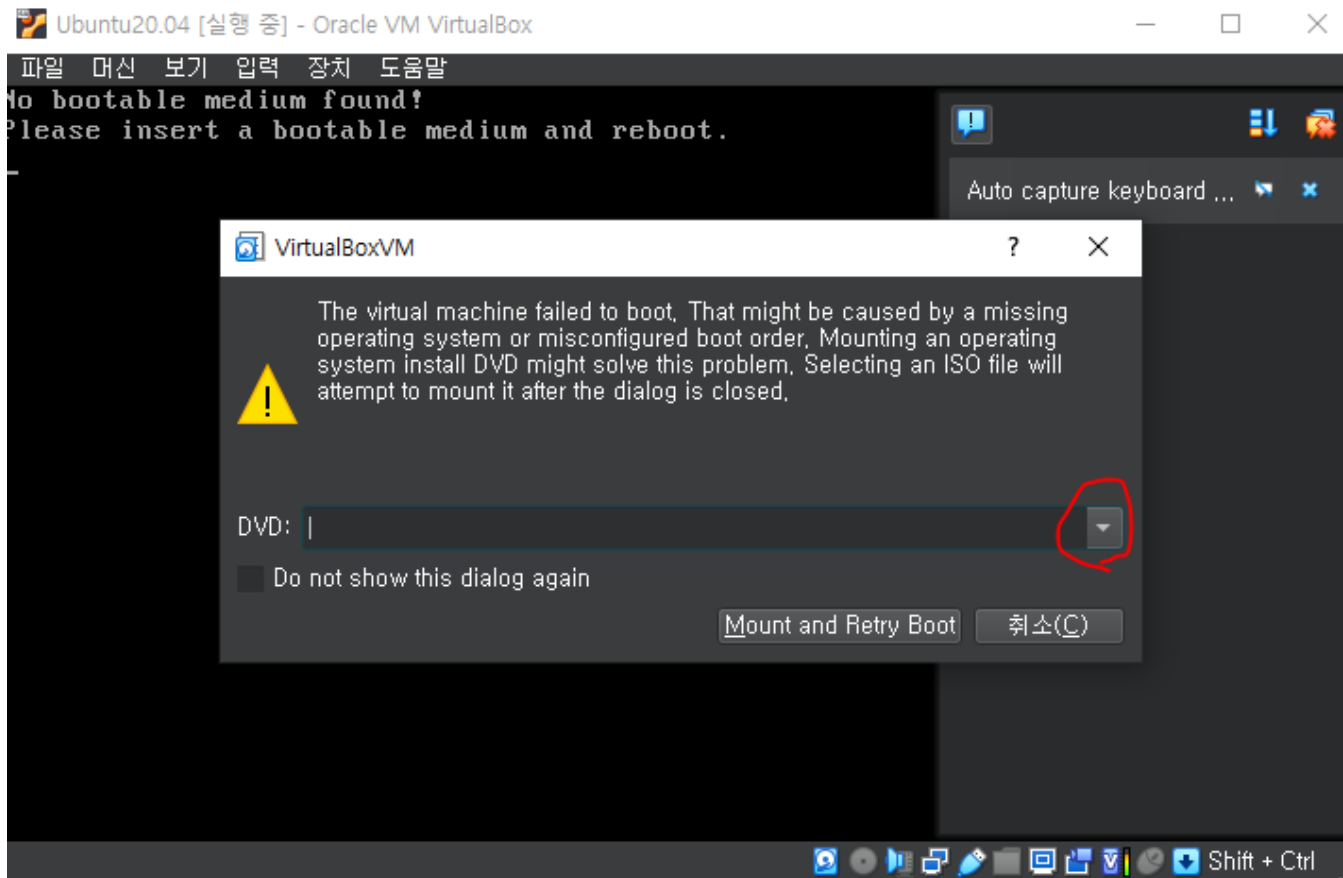
Installation

- VirtualBox setting
 - After the basic settings, select the newly created virtual machine and click the Start button



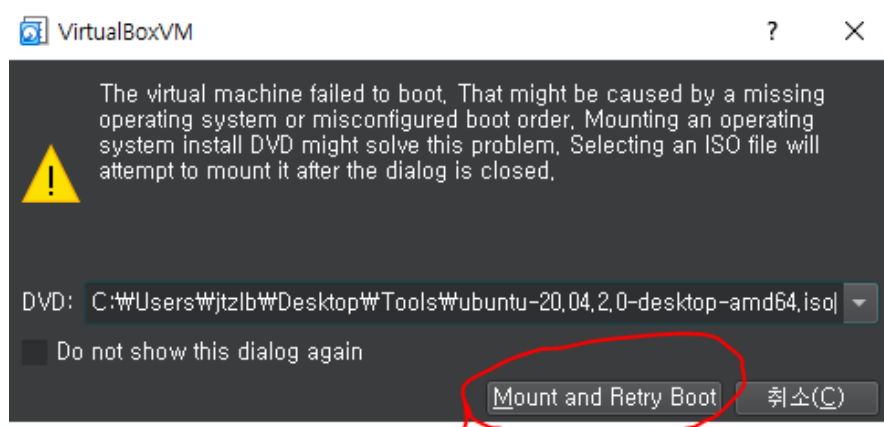
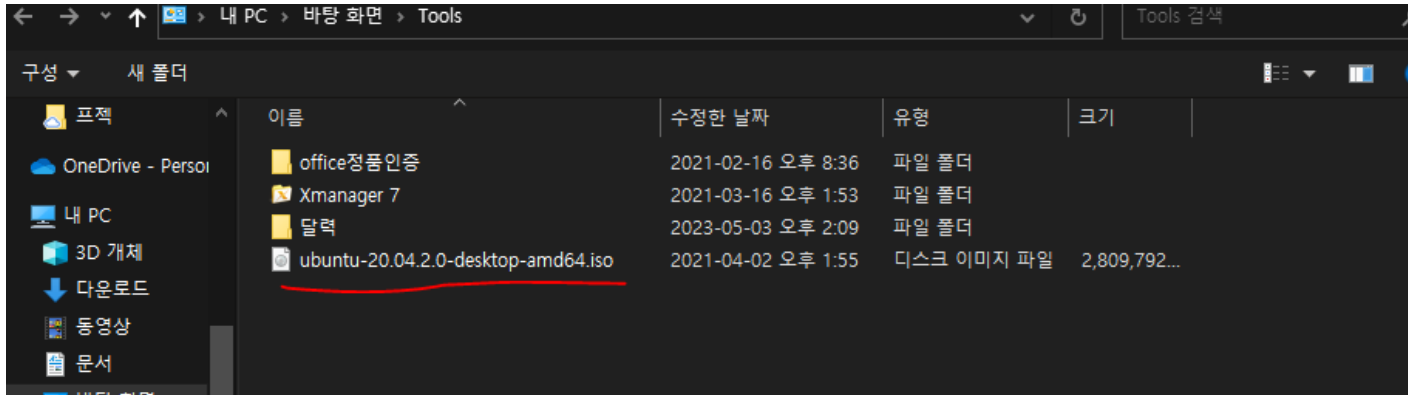
Installation

- VirtualBox setting
 - It's time to plug-in the Ubuntu ISO image
 - Press the dropdown button



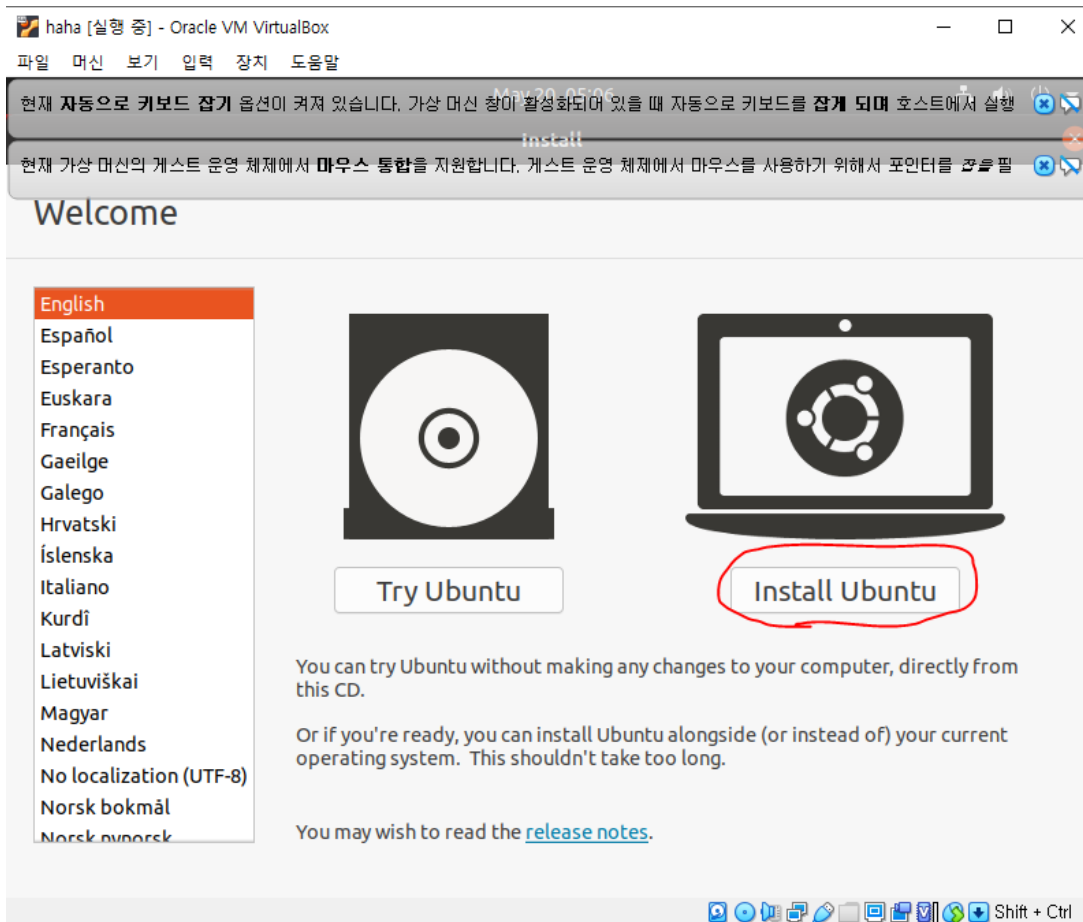
Installation

- VirtualBox setting
 - Find & select the iso image
 - Then, press the mount and retry boot button



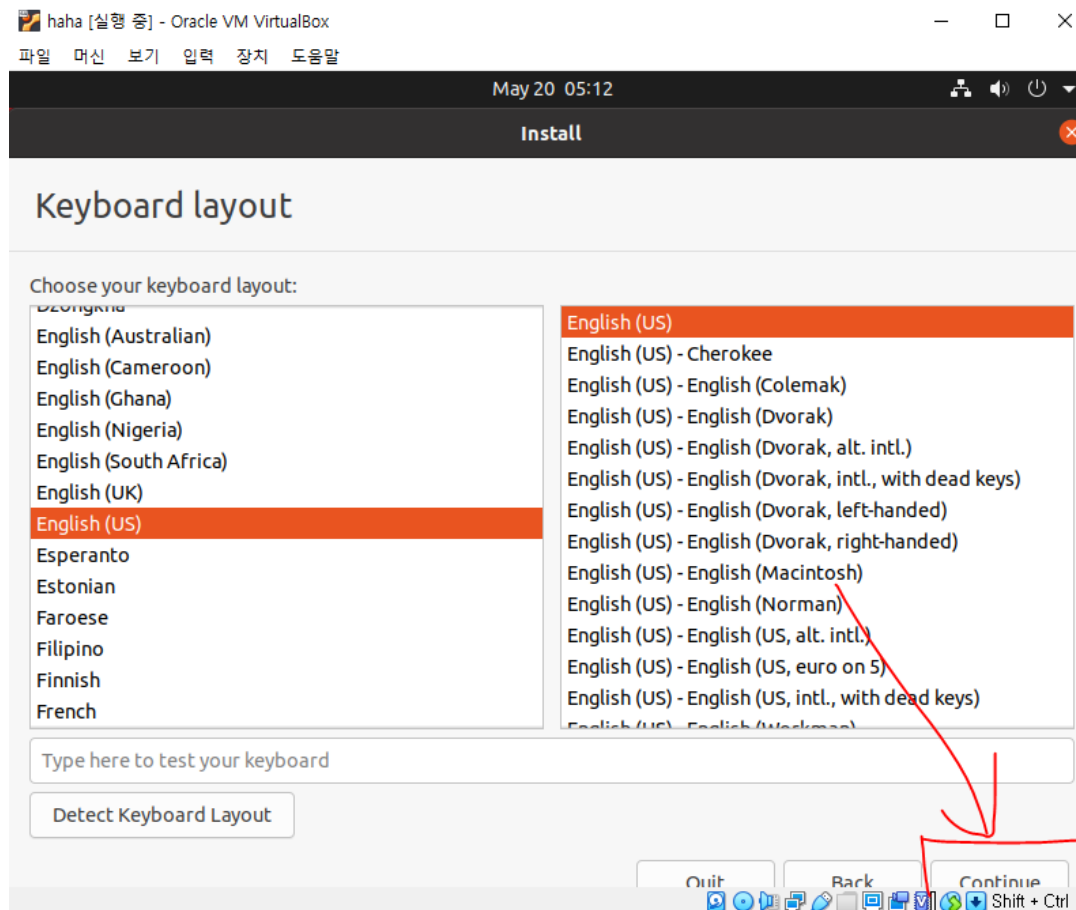
Installation

- Ubuntu setup
 - This screen appears a few moments later
 - Let's press the install Ubuntu



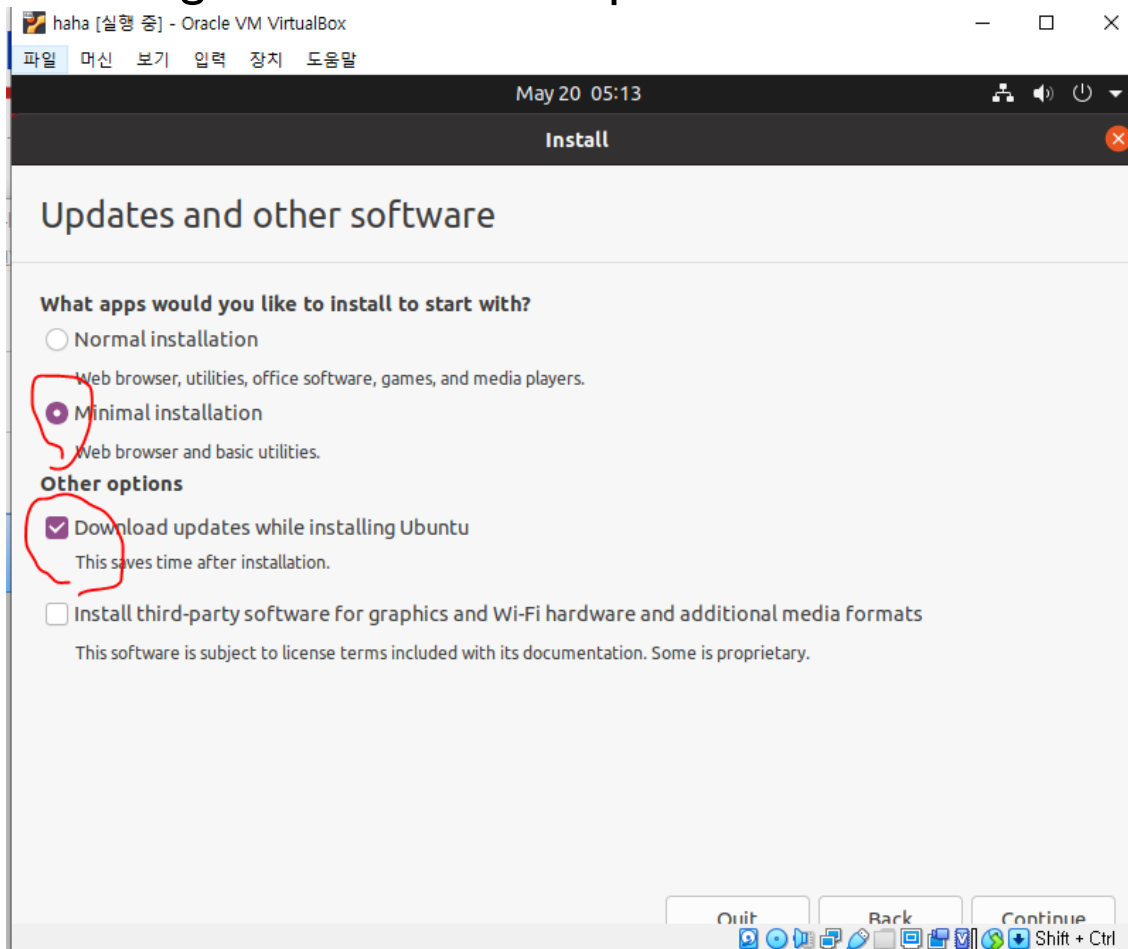
Installation

- Ubuntu setup
 - Press the Continue button



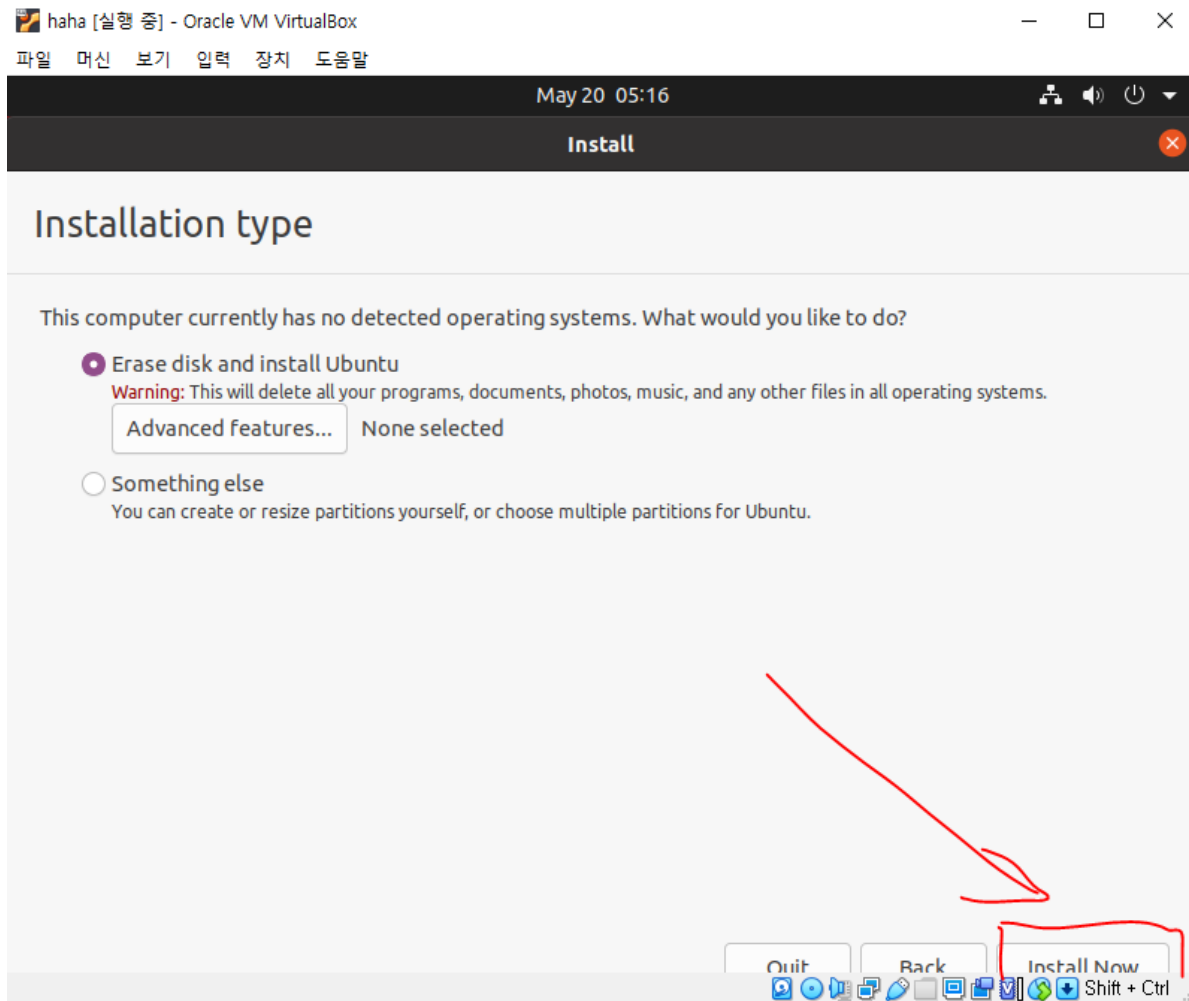
Installation

- Ubuntu setup
 - Select the Minimal installation option, Download update while installing Ubuntu, and then press the Continue button



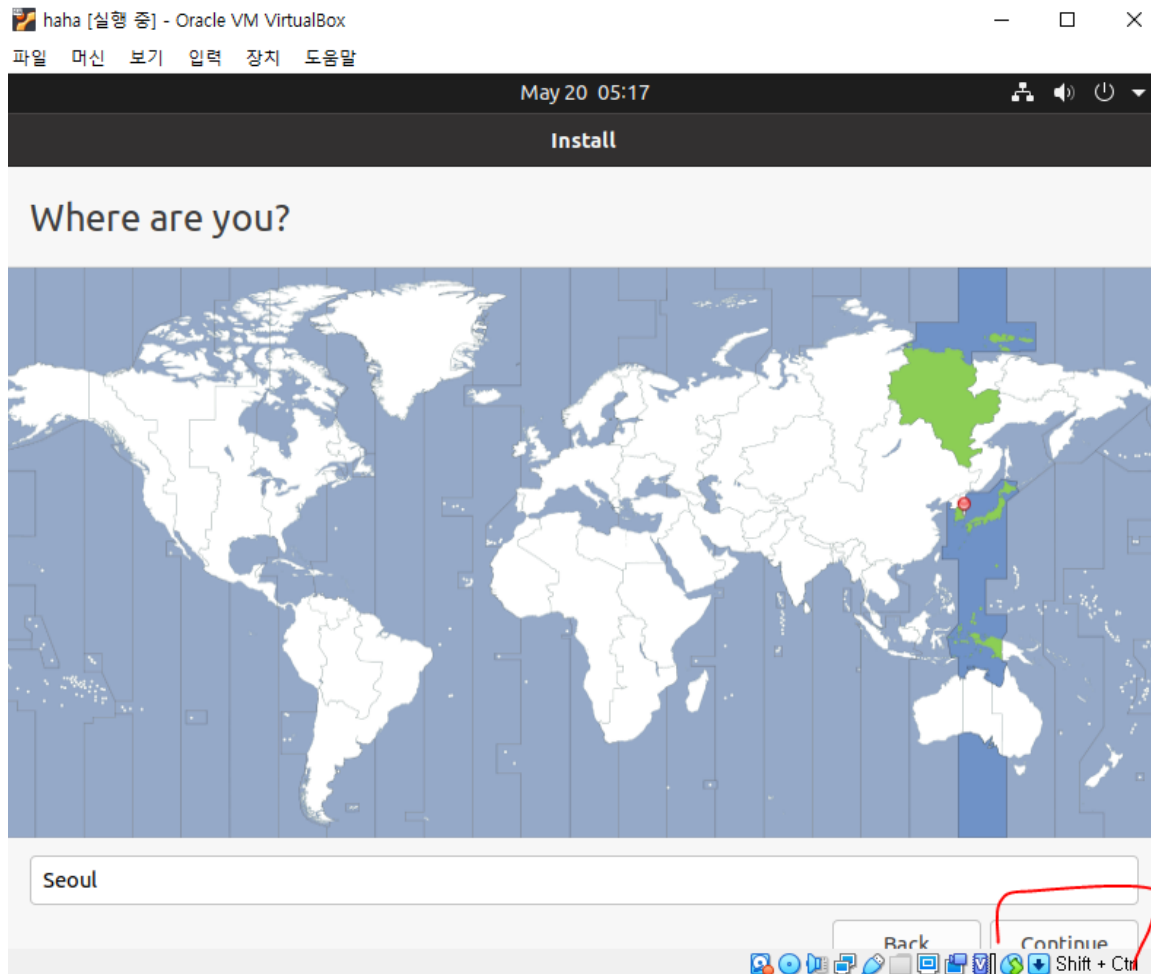
Installation

- Ubuntu setup
 - Press the Install Now



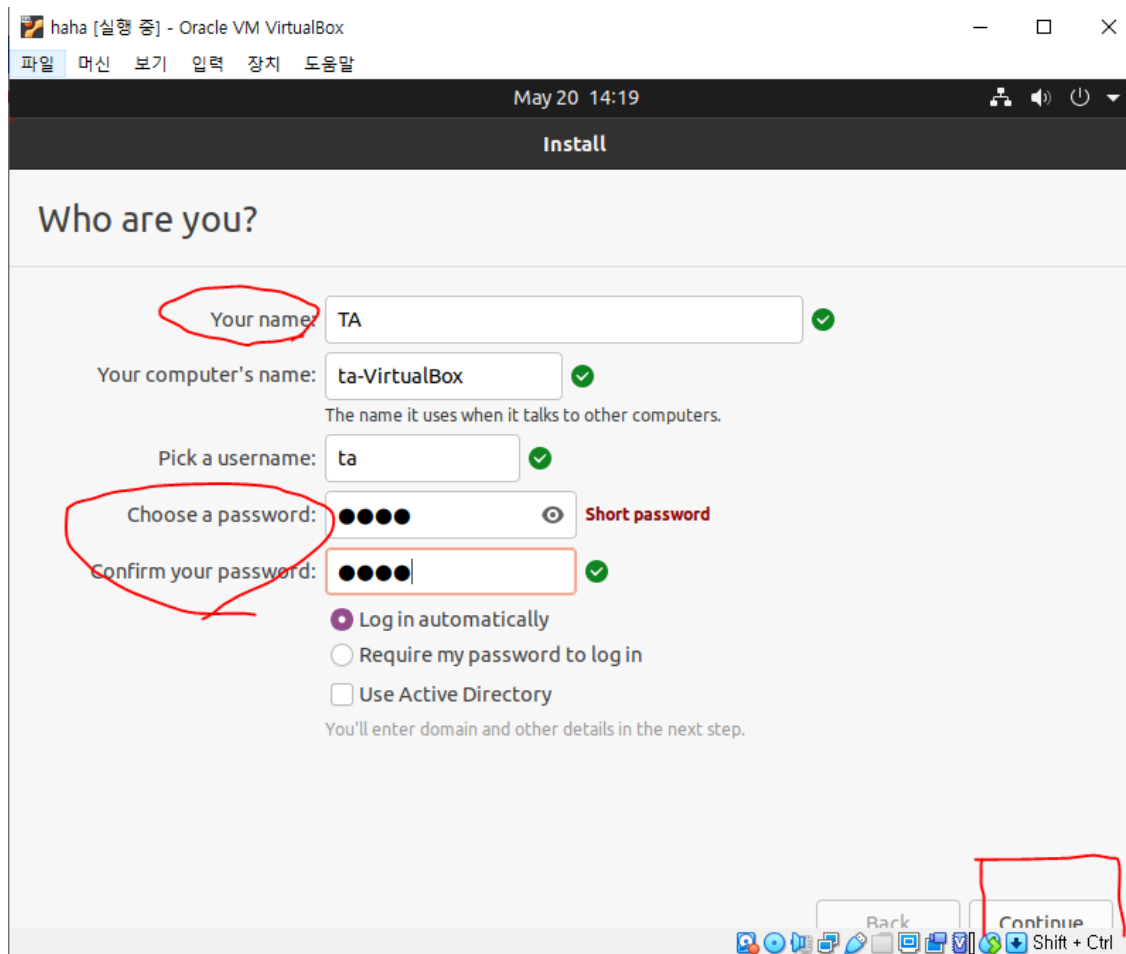
Installation

- Ubuntu setup
 - Press the Continue button



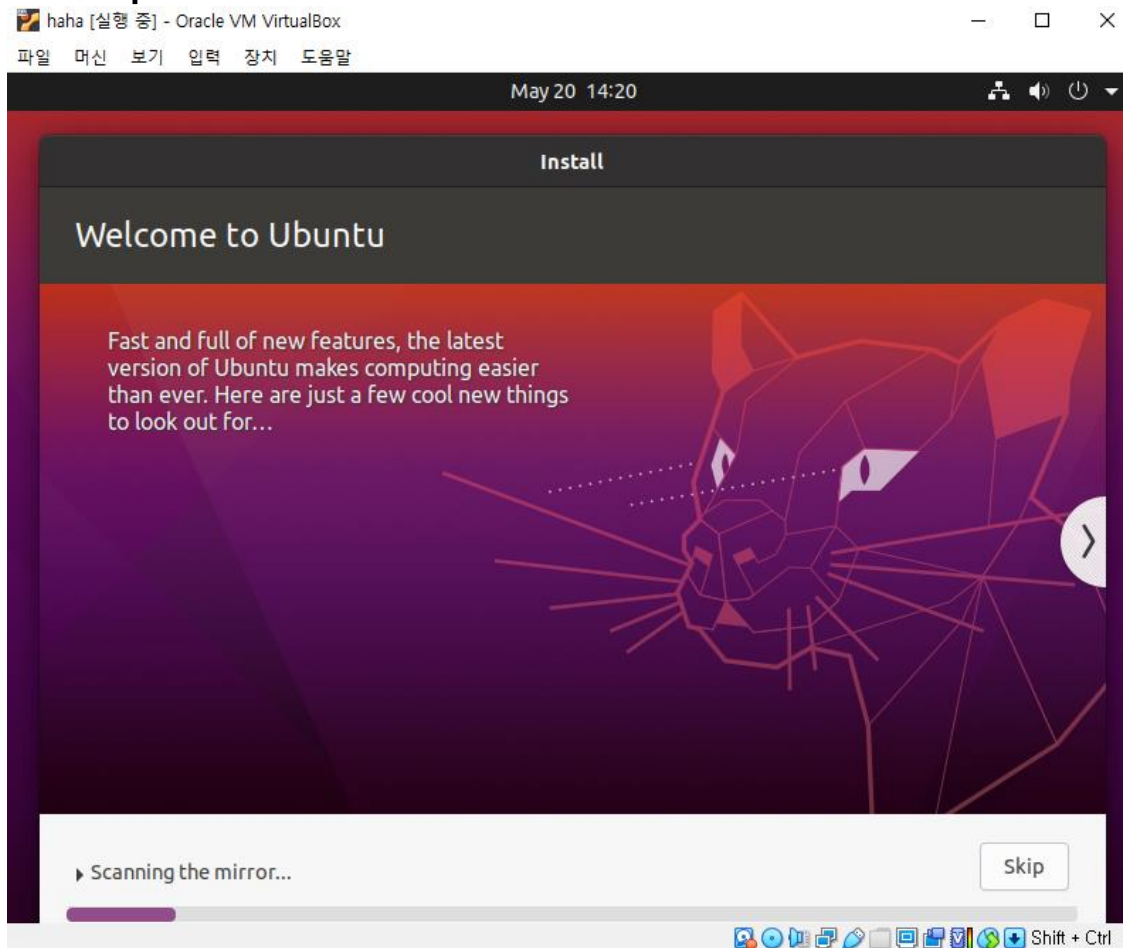
Installation

- Ubuntu setup
 - Who are you?



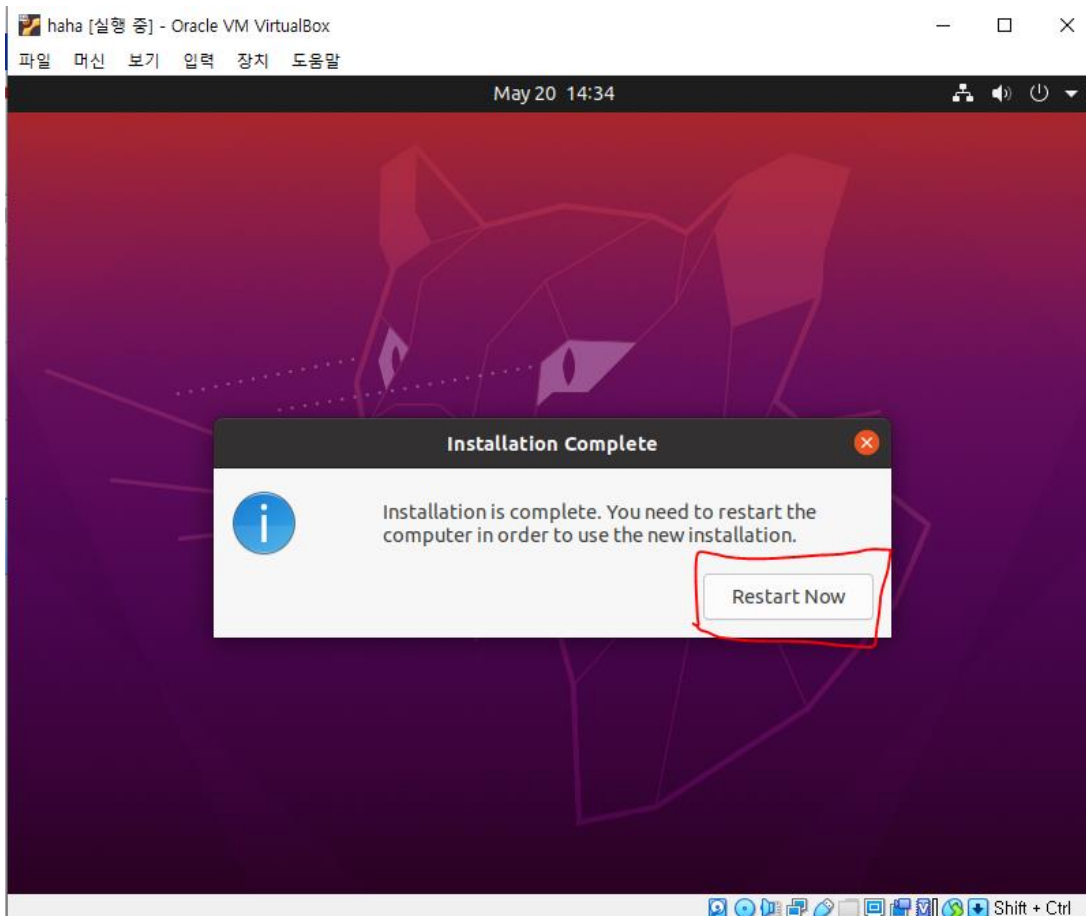
Installation

- Ubuntu setup
 - Welcome to Ubuntu... It takes a long time (10+ minutes) to complete the installation



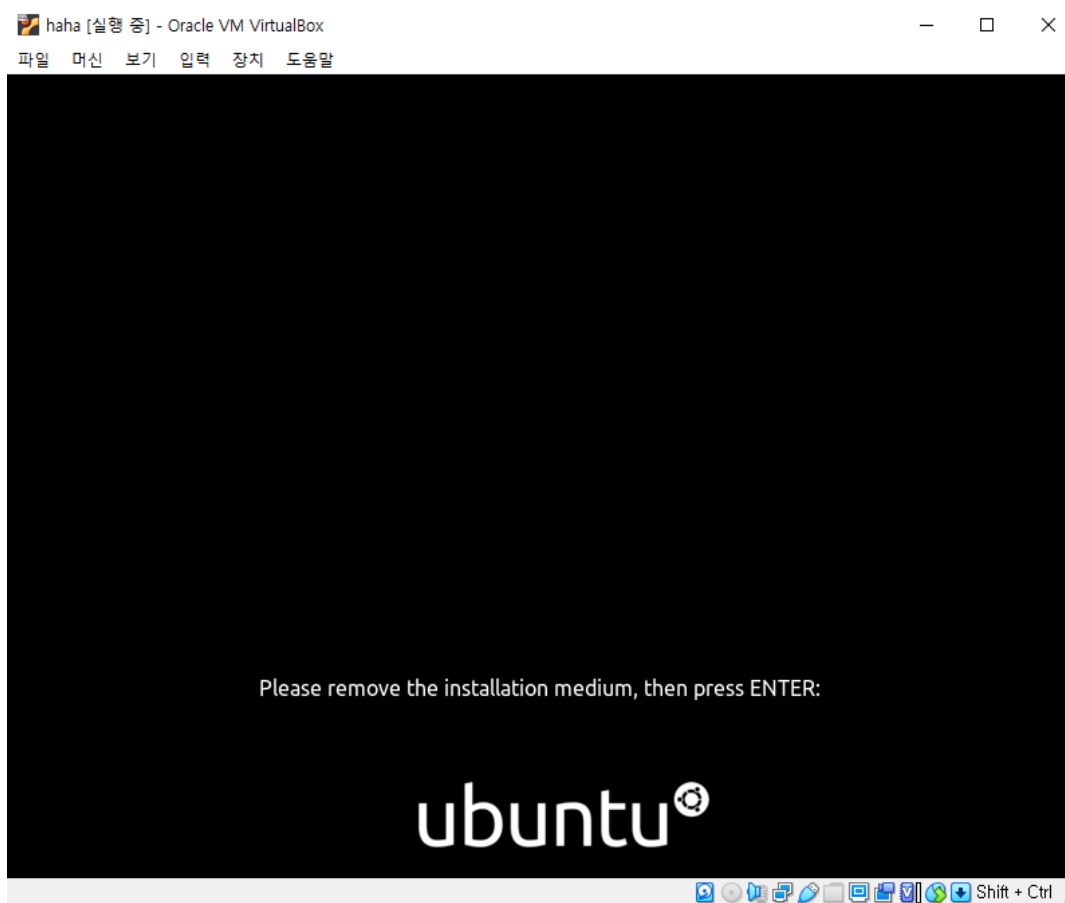
Installation

- Ubuntu setup
 - Congratulations. You have completed the Ubuntu installation.
 - Restart now



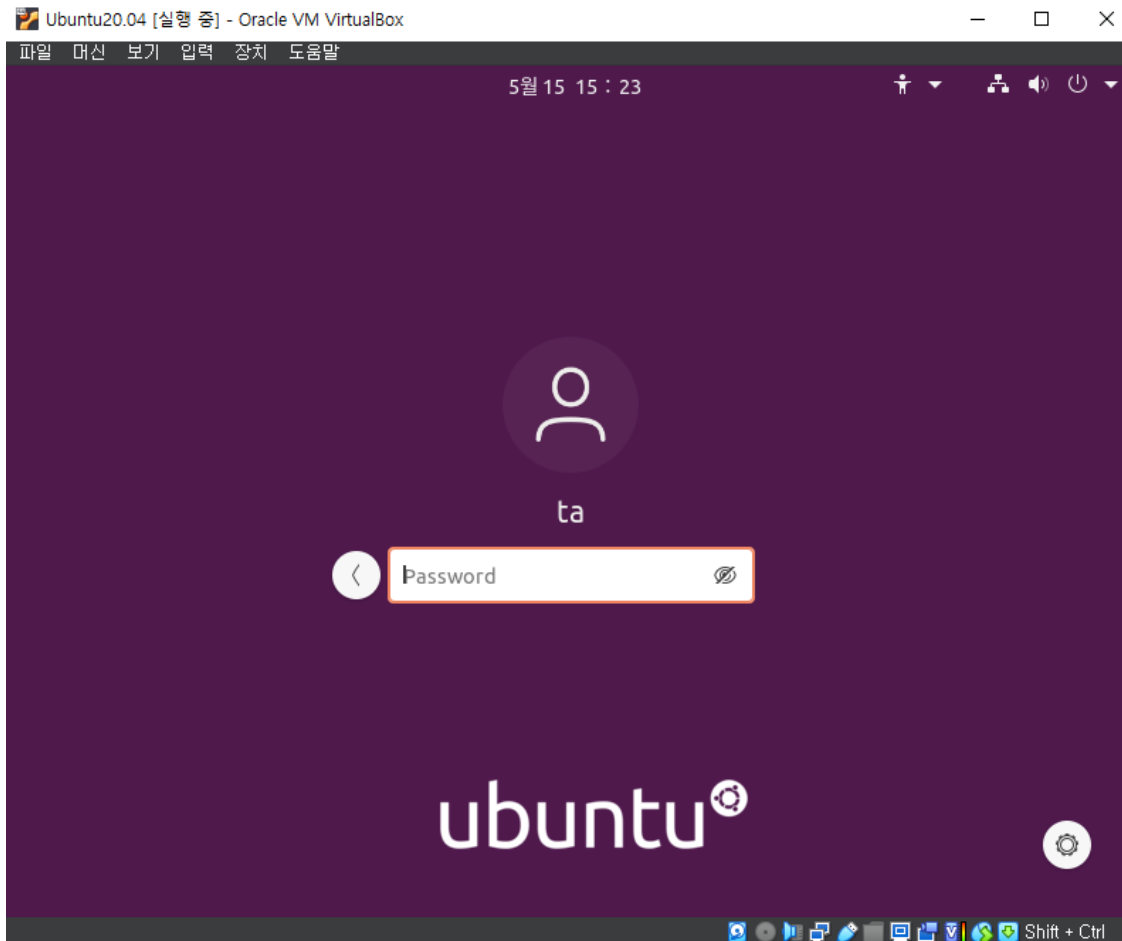
Installation

- Ubuntu setup
 - Just press ENTER after restarting



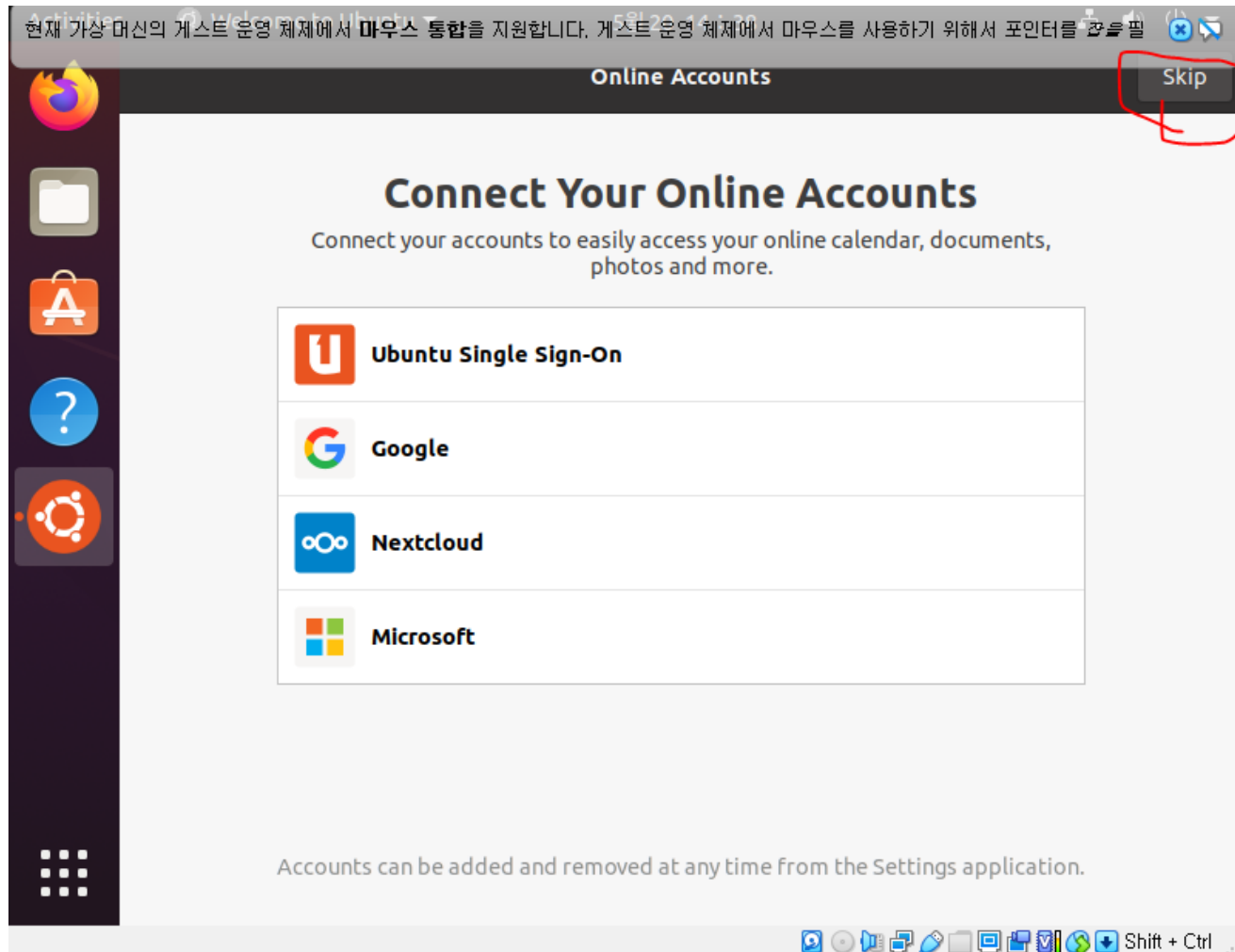
Installation

- Ubuntu setup
 - Log-in



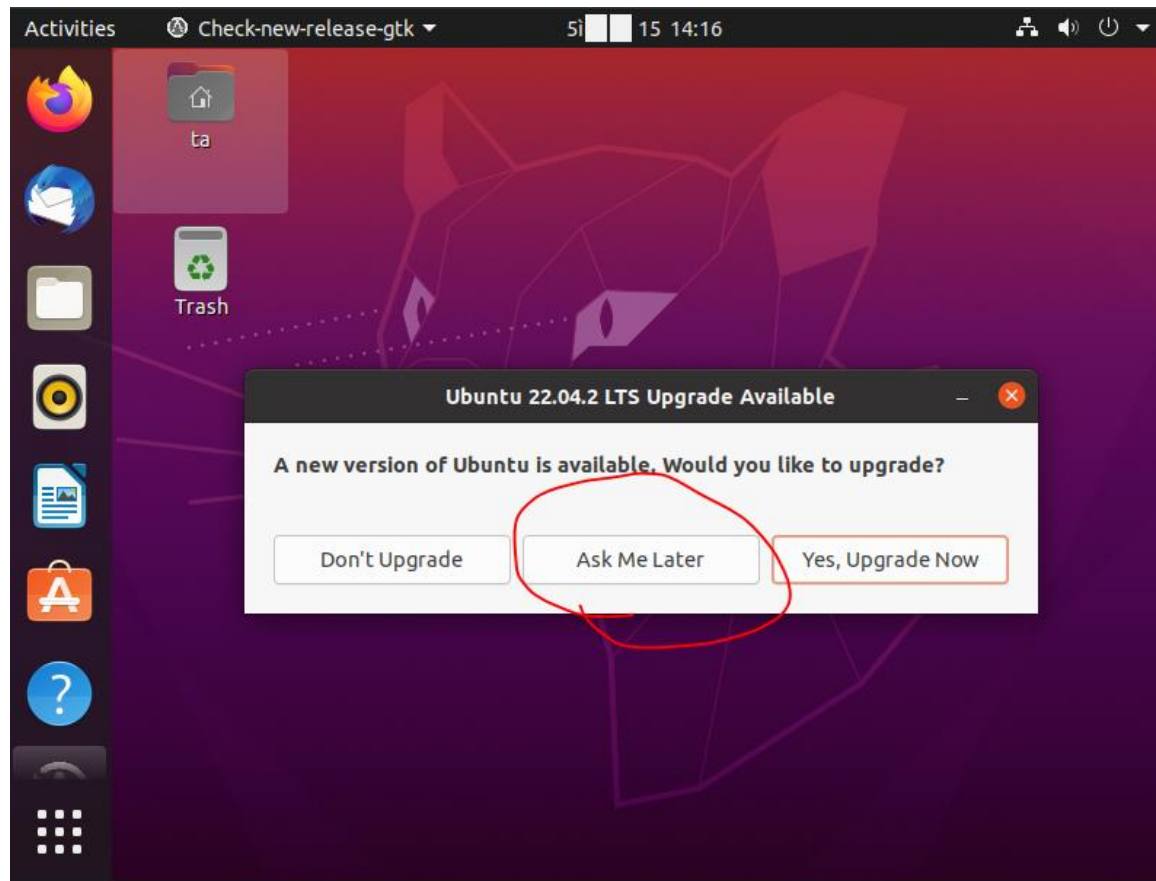
Installation

- Ubuntu setup
 - Skip all the suggestions



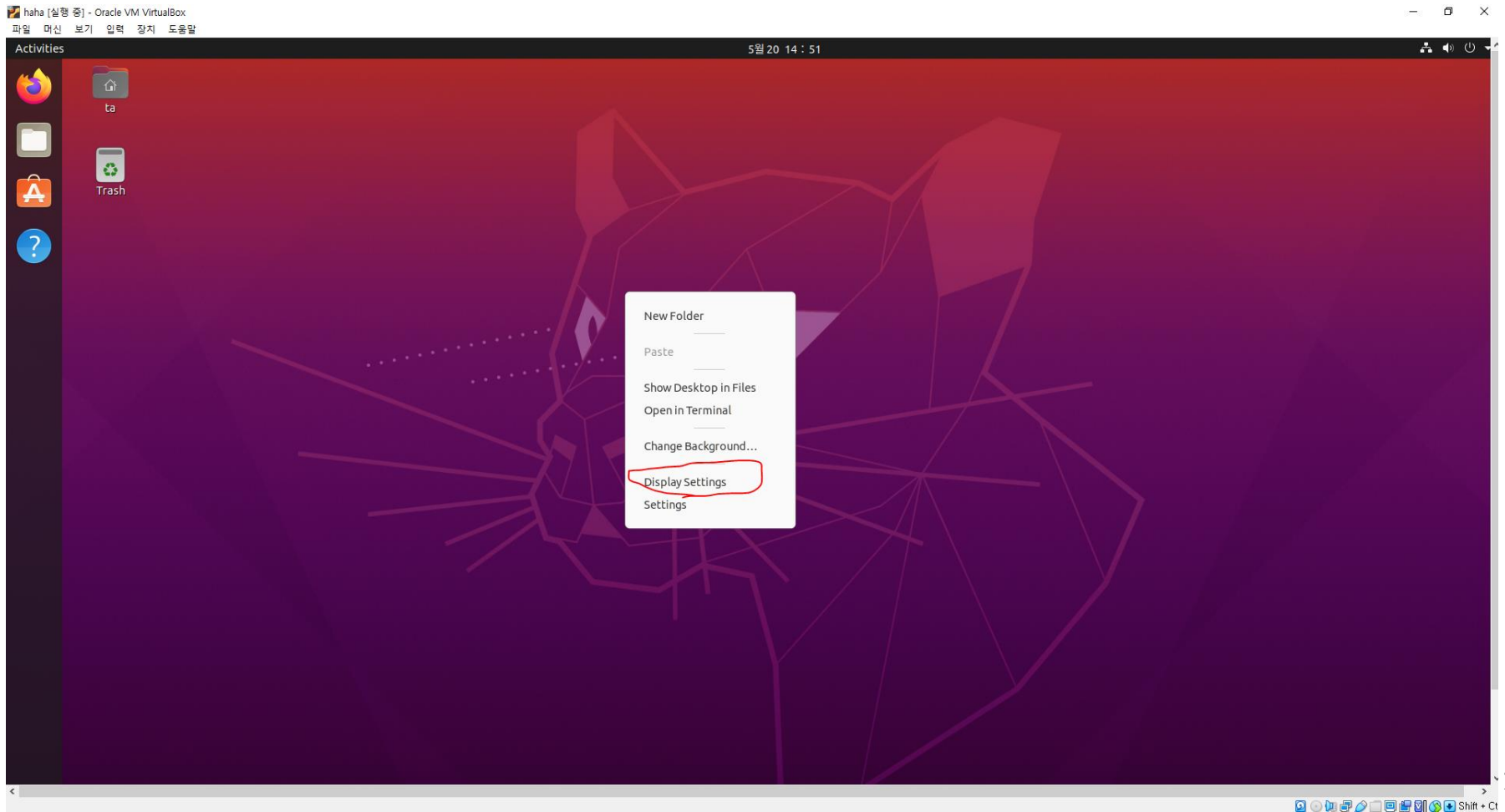
Installation

- Ubuntu setup
 - If you see a pop-up window like this, press the Ask Me Later



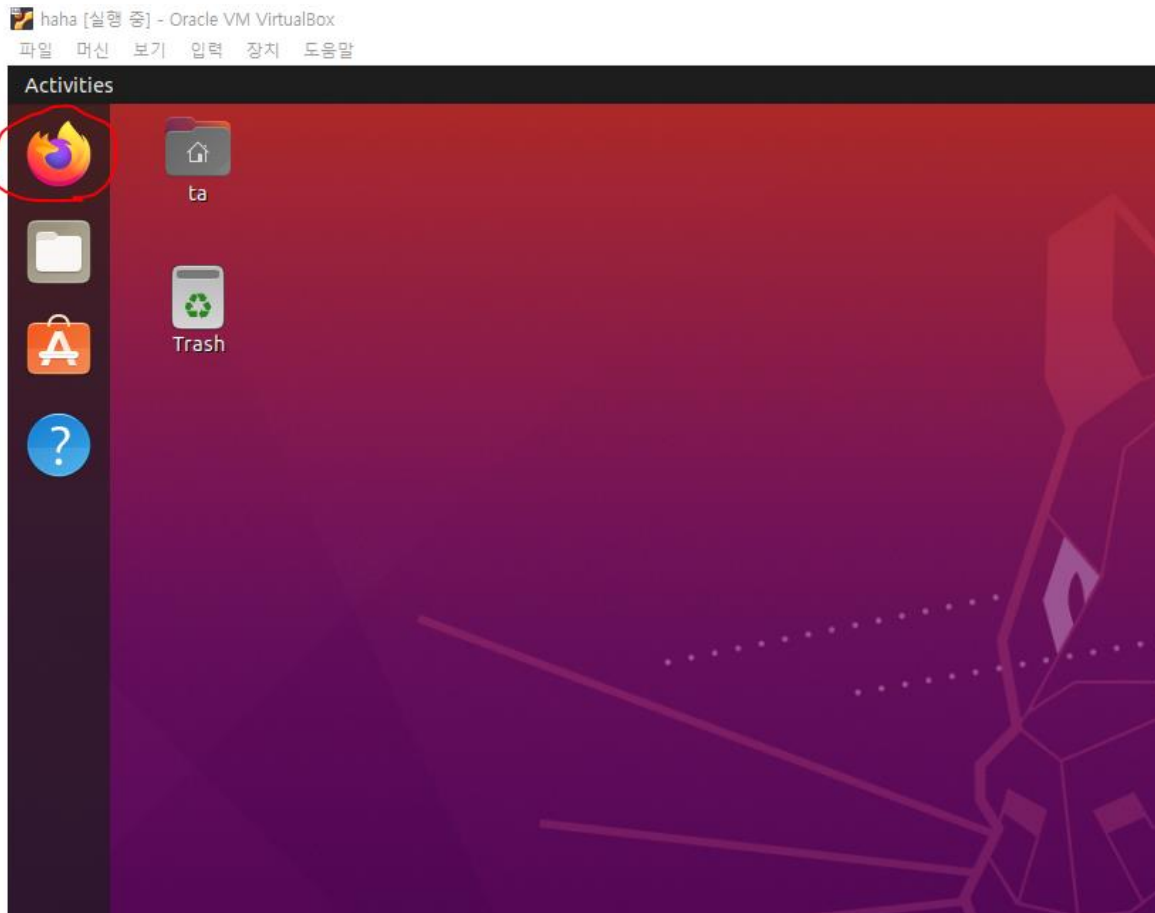
Installation

- Optional
 - You can open display settings by right-clicking the background



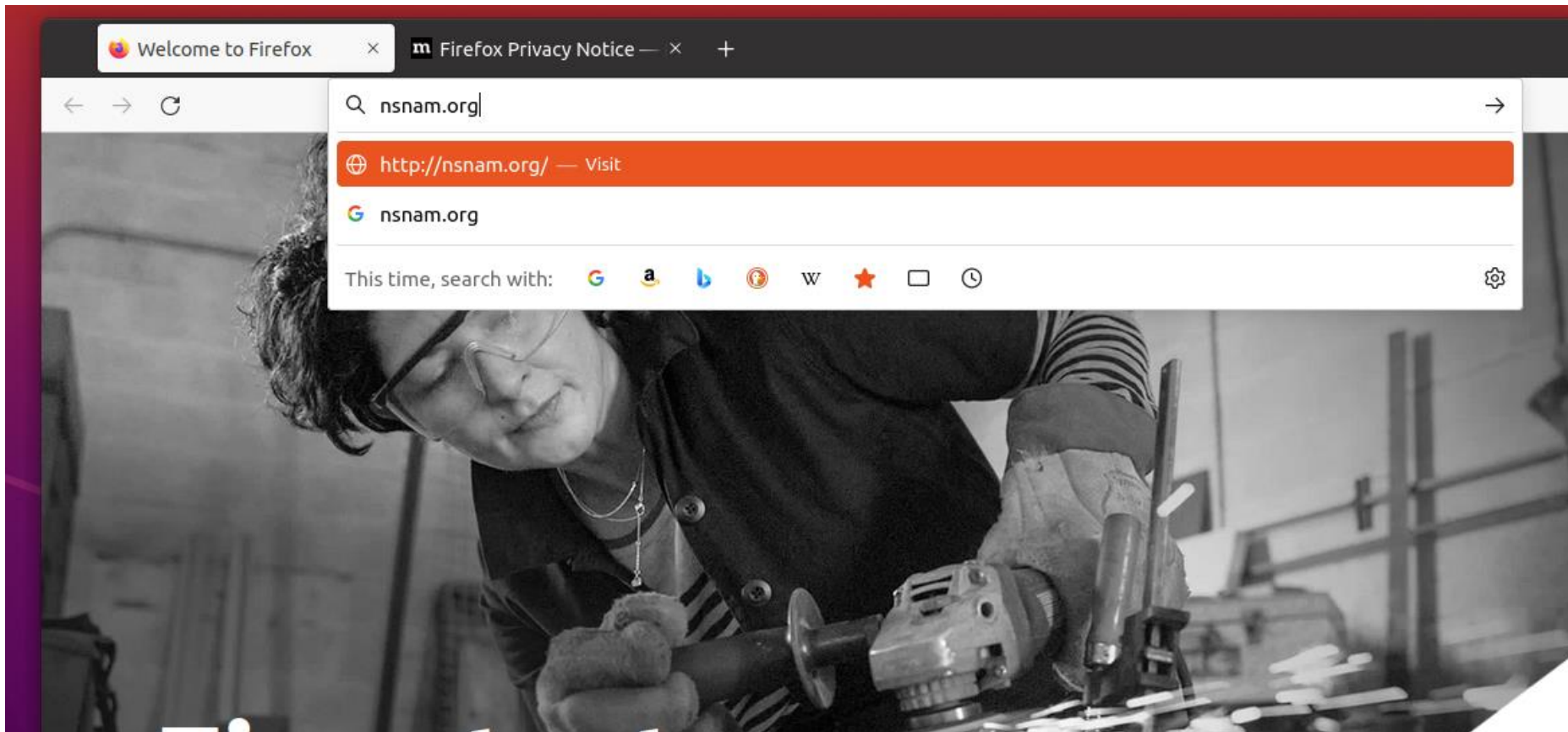
Installation

- ns-3 installation
 - Open Firefox web browser on the left side



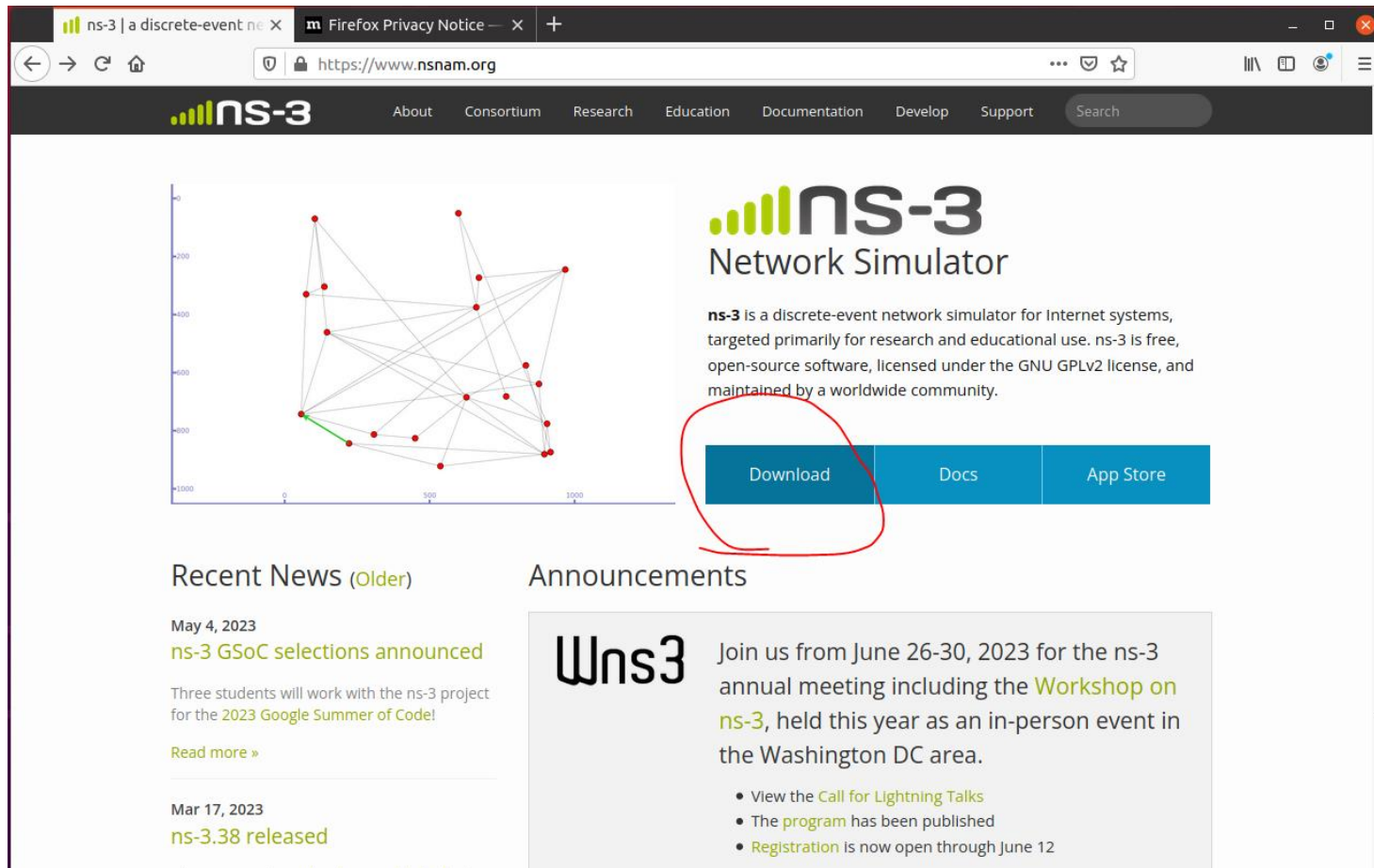
Installation

- ns-3 installation
 - Visit to nsnam.org



Installation

- ns-3 installation
 - Press the Download button



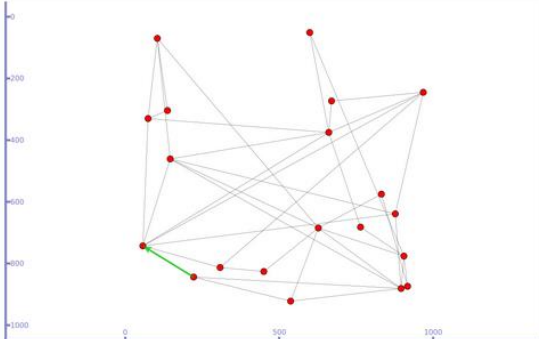
The screenshot shows the ns-3 website homepage in a browser window. The browser's address bar displays "https://www.nsnam.org". The website's navigation menu includes "About", "Consortium", "Research", "Education", "Documentation", "Develop", "Support", and a search bar. The main content area features a network graph on the left and the ns-3 logo and title on the right. Below the logo, a paragraph describes ns-3 as a discrete-event network simulator for Internet systems, targeted for research and educational use. At the bottom of this section, three buttons are visible: "Download", "Docs", and "App Store". The "Download" button is circled in red. Below the main content, there are sections for "Recent News (Older)" and "Announcements". The "Recent News" section lists two items: "ns-3 GSoC selections announced" (dated May 4, 2023) and "ns-3.38 released" (dated Mar 17, 2023). The "Announcements" section features a large "Wns3" logo and text inviting users to join the ns-3 annual meeting from June 26-30, 2023, in the Washington DC area. A list of bullet points provides further details about the meeting, including a call for lightning talks, the published program, and open registration through June 12.

ns-3 | a discrete-event network simulator

Firefox Privacy Notice

https://www.nsnam.org

ns-3 About Consortium Research Education Documentation Develop Support Search



ns-3 Network Simulator

ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for research and educational use. ns-3 is free, open-source software, licensed under the GNU GPLv2 license, and maintained by a worldwide community.

Download Docs App Store

Recent News (Older)

May 4, 2023
ns-3 GSoC selections announced

Three students will work with the ns-3 project for the 2023 Google Summer of Code!

[Read more »](#)

Mar 17, 2023
ns-3.38 released

Announcements

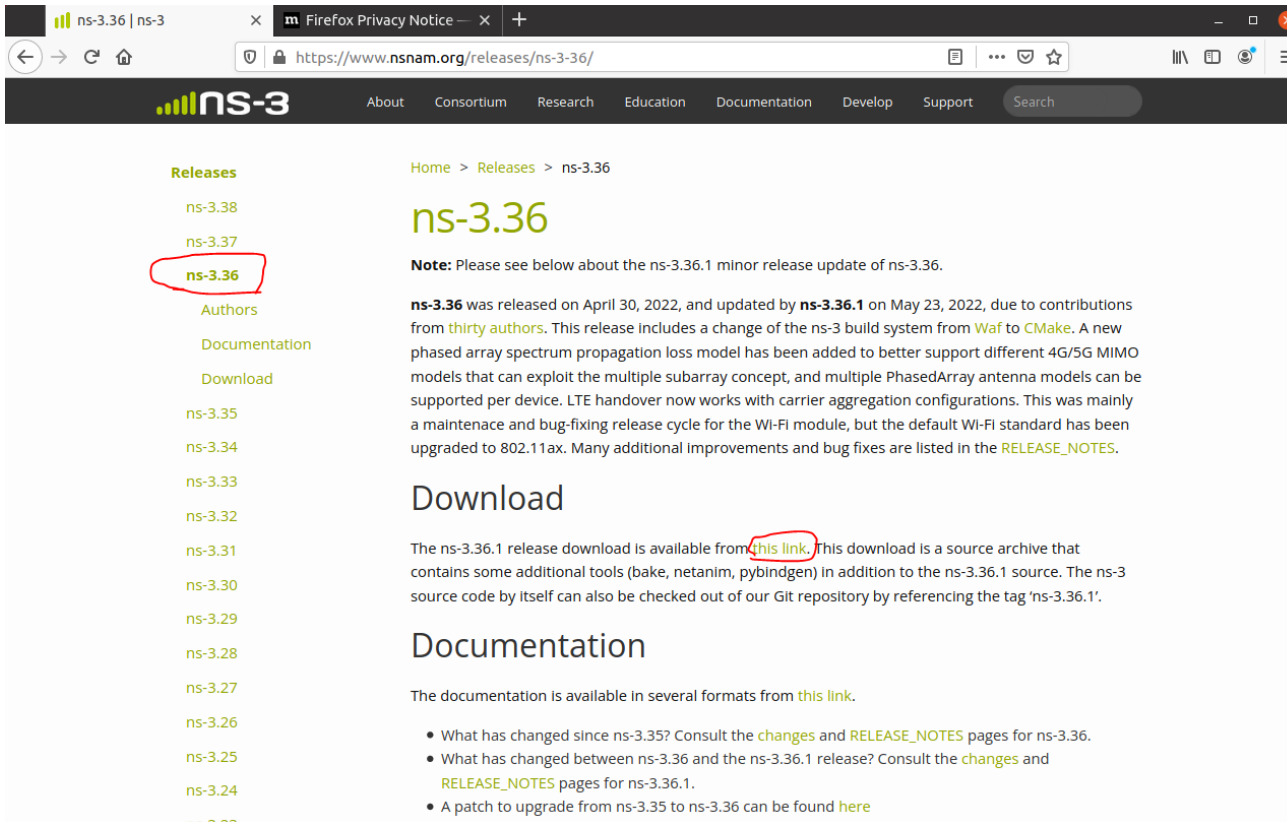
Wns3

Join us from June 26-30, 2023 for the ns-3 annual meeting including the **Workshop on ns-3**, held this year as an in-person event in the Washington DC area.

- View the [Call for Lightning Talks](#)
- The [program](#) has been published
- [Registration](#) is now open through June 12

Installation

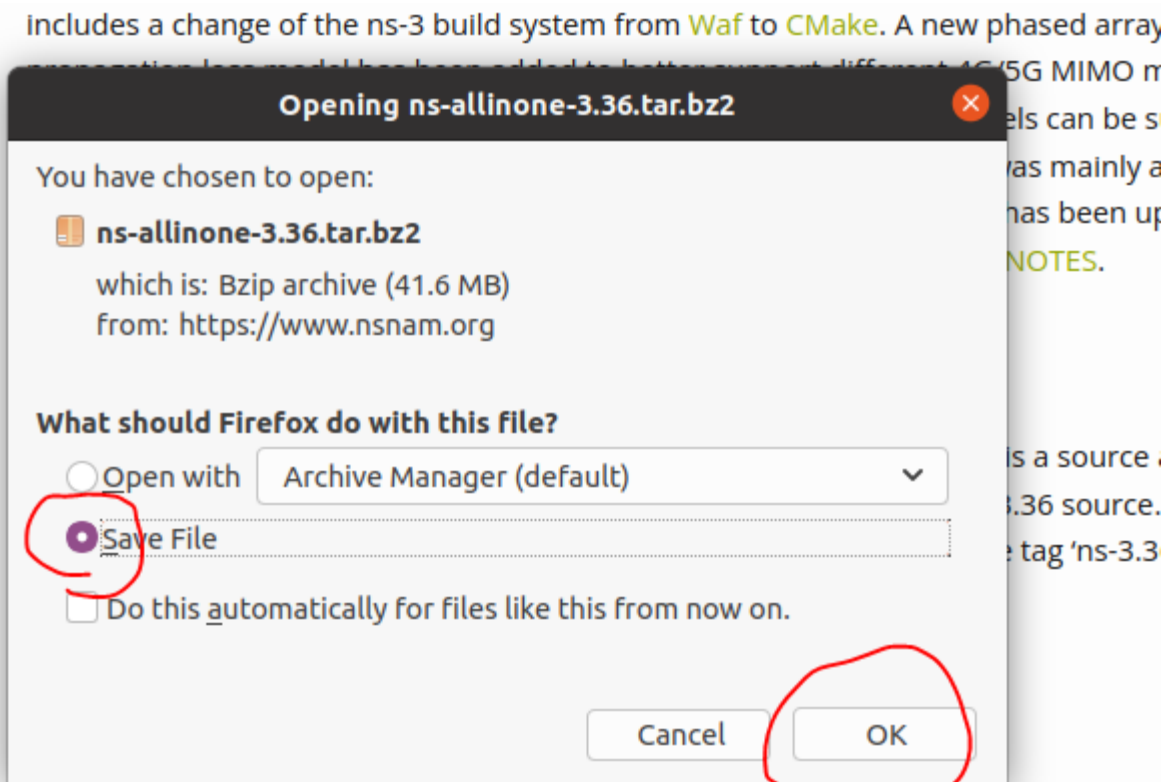
- ns-3 installation
 - Select the 'ns-3.36', and then, Press the word 'this link' in the Download section



The screenshot shows the ns-3 website's release page for ns-3.36. The browser address bar shows the URL <https://www.nsnam.org/releases/ns-3.36/>. The page features a navigation menu with links for About, Consortium, Research, Education, Documentation, Develop, and Support. A search bar is also present. On the left sidebar, under the 'Releases' section, the version 'ns-3.36' is highlighted with a red circle. The main content area displays the title 'ns-3.36' and a note about a minor update. Below this, the 'Download' section contains the text: 'The ns-3.36.1 release download is available from [this link](#).' The phrase 'this link' is circled in red. The 'Documentation' section follows, stating that documentation is available from [this link](#).

Installation

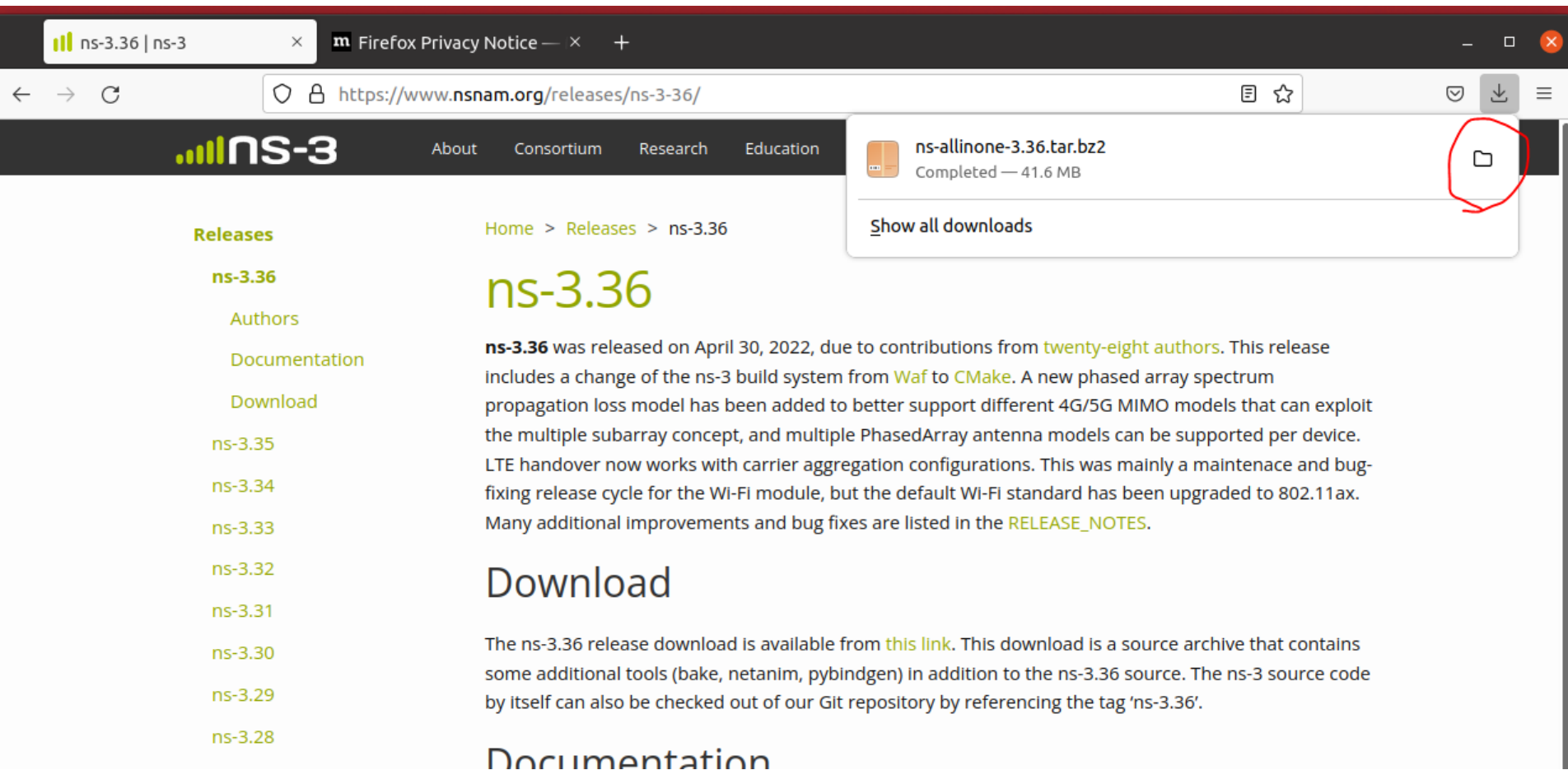
- ns-3 installation
 - Select the Save file option and press the OK



- What has changed since ns-3.35? Consult the [changes](#) and [RELEASE_NOTES](#) page
- A patch to upgrade from ns-3.35 to ns-3.36 can be found [here](#)
- Errata containing any late-breaking information about the release can be found

Installation

- ns-3 installation
 - When the download is complete, open the Downloads directory



The screenshot shows a web browser window displaying the ns-3 website. The browser's address bar shows the URL <https://www.nsnam.org/releases/ns-3-36/>. The website's navigation bar includes links for "About", "Consortium", "Research", and "Education". The main content area features a "Releases" section with a list of versions from ns-3.28 to ns-3.36. The current page is for ns-3.36, which includes a "Download" section. A download notification is visible in the top right corner of the browser, showing a file named "ns-allinone-3.36.tar.bz2" (41.6 MB) with a folder icon circled in red. The notification also includes a link to "Show all downloads".

ns-3.36 | ns-3

Firefox Privacy Notice

https://www.nsnam.org/releases/ns-3-36/

ns-3

About Consortium Research Education

Releases

ns-3.36

Authors

Documentation

Download

ns-3.35

ns-3.34

ns-3.33

ns-3.32

ns-3.31

ns-3.30

ns-3.29

ns-3.28

Home > Releases > ns-3.36

ns-3.36

ns-3.36 was released on April 30, 2022, due to contributions from [twenty-eight authors](#). This release includes a change of the ns-3 build system from [Waf](#) to [CMake](#). A new phased array spectrum propagation loss model has been added to better support different 4G/5G MIMO models that can exploit the multiple subarray concept, and multiple PhasedArray antenna models can be supported per device. LTE handover now works with carrier aggregation configurations. This was mainly a maintenance and bug-fixing release cycle for the Wi-Fi module, but the default Wi-Fi standard has been upgraded to 802.11ax. Many additional improvements and bug fixes are listed in the [RELEASE_NOTES](#).

Download

The ns-3.36 release download is available from [this link](#). This download is a source archive that contains some additional tools (bake, netanim, pybindgen) in addition to the ns-3.36 source. The ns-3 source code by itself can also be checked out of our Git repository by referencing the tag 'ns-3.36'.

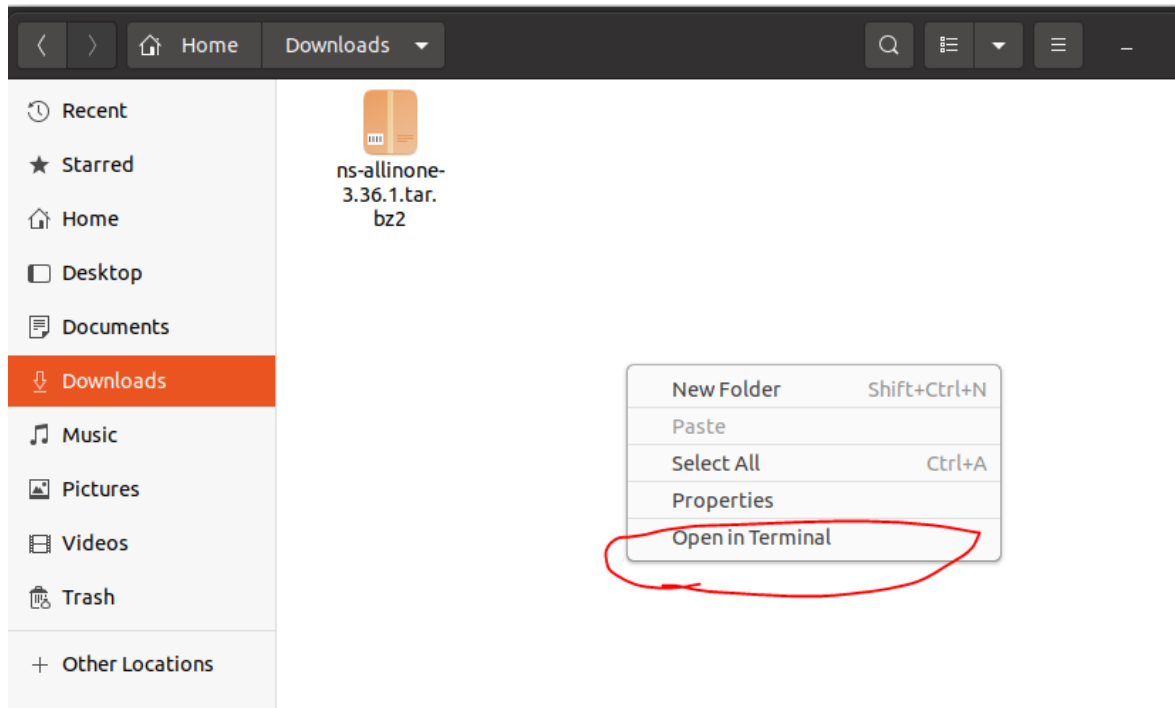
Documentation

ns-allinone-3.36.tar.bz2
Completed — 41.6 MB

Show all downloads

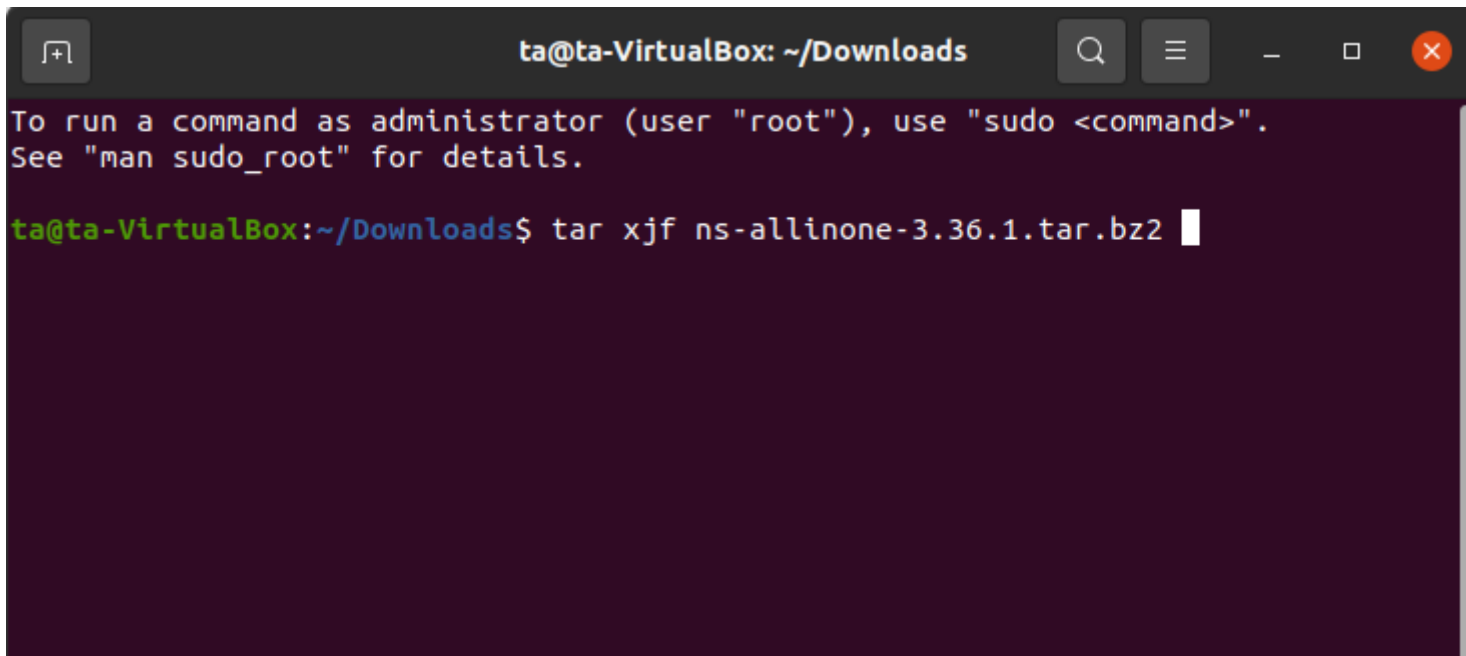
Installation

- ns-3 installation
 - Right-click on the blank space in the Downloads directory and press "Open in Terminal"



Installation

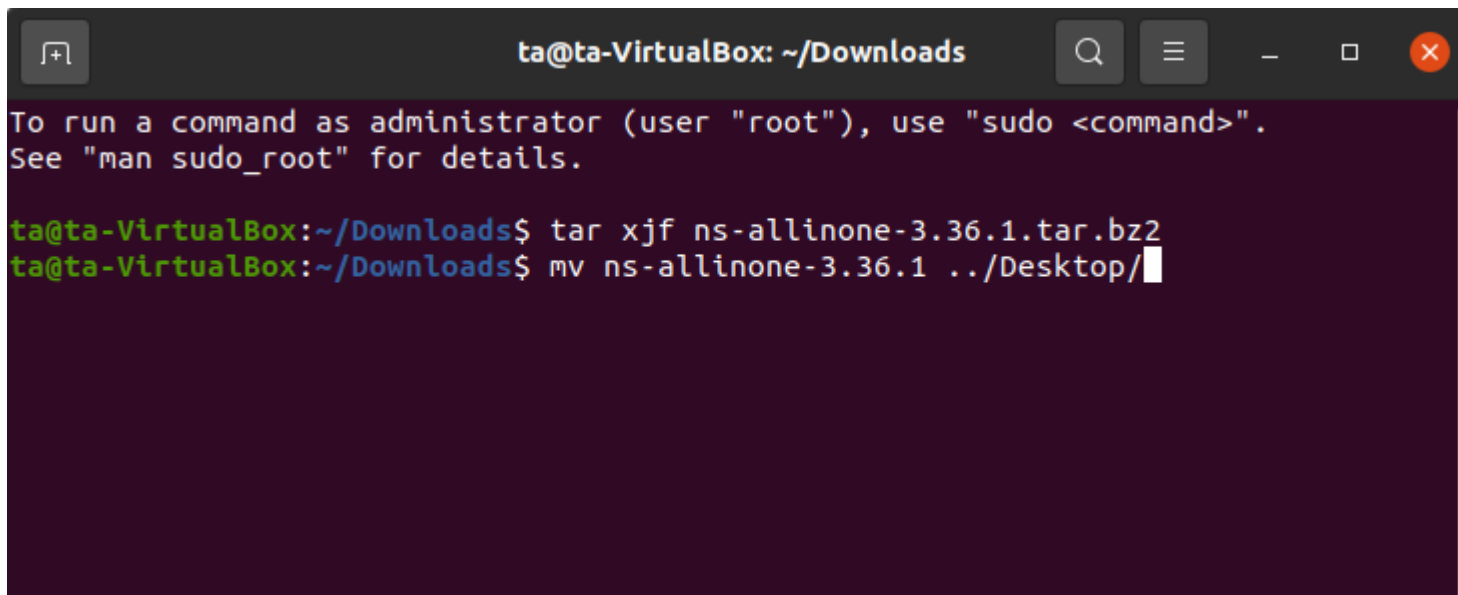
- ns-3 installation
 - Type `tar xjf ns-allinone-3.36.1.tar.bz2` in the terminal
 - Tip: Type up to `tar xjfs ns` and press Tab key to complete the filename automatically.



```
ta@ta-VirtualBox: ~/Downloads
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
ta@ta-VirtualBox:~/Downloads$ tar xjf ns-allinone-3.36.1.tar.bz2
```

Installation

- ns-3 installation
 - Type `mv ns-allinone-3.36.1 ../Desktop/` in the terminal
 - This command will move (mv) ns-3 directory to Desktop directory
 - `..` stands for the parent directory
 - Tip: Type up to `mv ns` and press Tab key to complete the filename automatically.



```
ta@ta-VirtualBox: ~/Downloads
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
ta@ta-VirtualBox:~/Downloads$ tar xjf ns-allinone-3.36.1.tar.bz2
ta@ta-VirtualBox:~/Downloads$ mv ns-allinone-3.36.1 ../Desktop/
```

Installation

- ns-3 installation
 - Type `cd ../Desktop/ns-allinone-3.36.1/ns-3.36.1/`
 - The command `cd` means change directory, so it will change your working directory to the `ns-3.36.1` directory
 - Note that I changed directory to `ns-3.36.1`, not `ns-allinone-3.36.1`

```
ta@ta-VirtualBox:~/Downloads$ tar xjf ns-allinone-3.36.1.tar.bz2
ta@ta-VirtualBox:~/Downloads$ mv ns-allinone-3.36.1 ../Desktop/
ta@ta-VirtualBox:~/Downloads$ cd ../Desktop/ns-allinone-3.36.1/ns-3.36.1/
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$
```

Installation

- ns-3 installation
 - Download minimal requirements to run ns-3
 - Type `sudo apt install python3 g++ cmake`
 - `sudo` means run the command with super user authority, and `apt install` command is used for download the packages
 - `g++` is a c++ compiler and `cmake` is a tool for build, test and package software
 - Then enter your login password

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ sudo apt install python3 g++ cmake  
[sudo] password for ta: █
```

- Enter `y`, if the question below appears

```
After this operation, 103 MB of additional disk space will be used.  
Do you want to continue? [Y/n] y  
Get:1 http://kr.archive.ubuntu.com/ubuntu focal-updates/main amd64
```


Installation

- ns-3 installation
 - Type these two commands

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ ./ns3 clean
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ ./ns3 configure --enable-test --enable-examples
```

- `./ns3 clean`
 - `./ns3` means execute `ns3` file in the current working directory (`./`)
 - `clean` for clean out the previous build is not usually strictly necessary, but is a good practice
- `./ns3 configure --enable-test --enable-examples`
 - `configure` is for change the build configuration of `ns-3` projects, such as logging options and compiler optimization (details are in the `ns-3` manual)
 - Further, You can use the `--enable-test` and `example` options to build a project that includes examples and tests
- Then type `./ns3 build` to build the `ns-3` project
 - It takes a long long time (30+ minutes)

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ ./ns3 build
```

Installation

- When the build is complete, enter the command below
`./ns3 run hello-simulator`

```
[100%] Linking CXX shared library ../../../../build/lib/libns3.36.1-lte-test-default.so
Scanning dependencies of target test-runner
[100%] Building CXX object utils/CMakeFiles/test-runner.dir/test-runner.cc.o
[100%] Linking CXX executable ../../build/utils/ns3.36.1-test-runner-default
Finished executing the following commands:
cd cmake-cache; cmake --build . -j 3 ; cd ..
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ ./ns3 run hello-simulator
Hello Simulator
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$
```

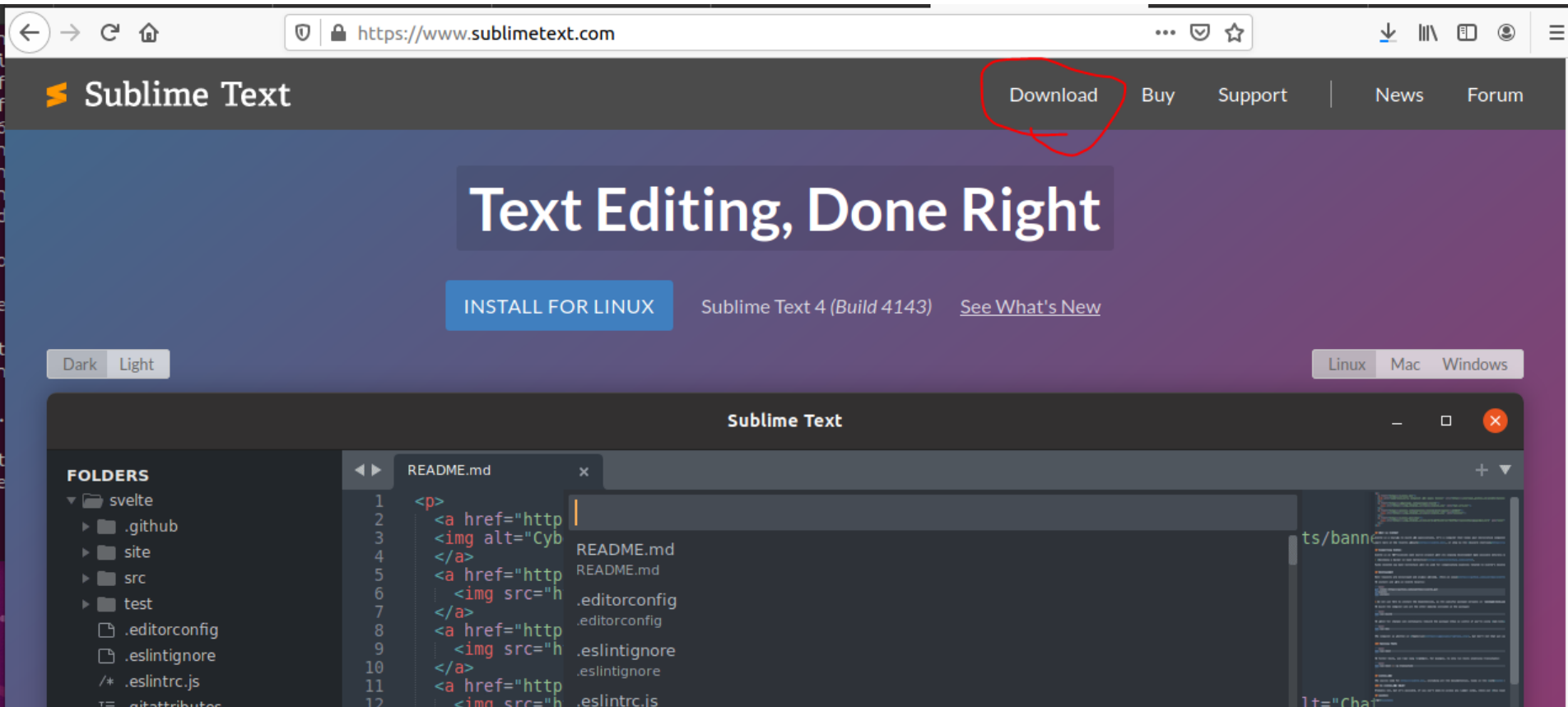
Congratulations! You are now an ns-3 user

Installation

- Optional
 - It is recommended that you use your favorite text editor or IDE
 - Popular text editors
 - Vim <https://www.vim.org/>
 - Emacs <https://www.gnu.org/software/emacs/>
 - Sublime text <https://www.sublimetext.com/>
 - Nano <https://www.nano-editor.org/>
 - IDE
 - VS code <https://code.visualstudio.com/>
 - CLion <https://www.jetbrains.com/clion/>
- This lecture uses Sublime text

Installation

- Optional
 - Sublime text installation



Installation

- Optional
 - Sublime text installation

Download

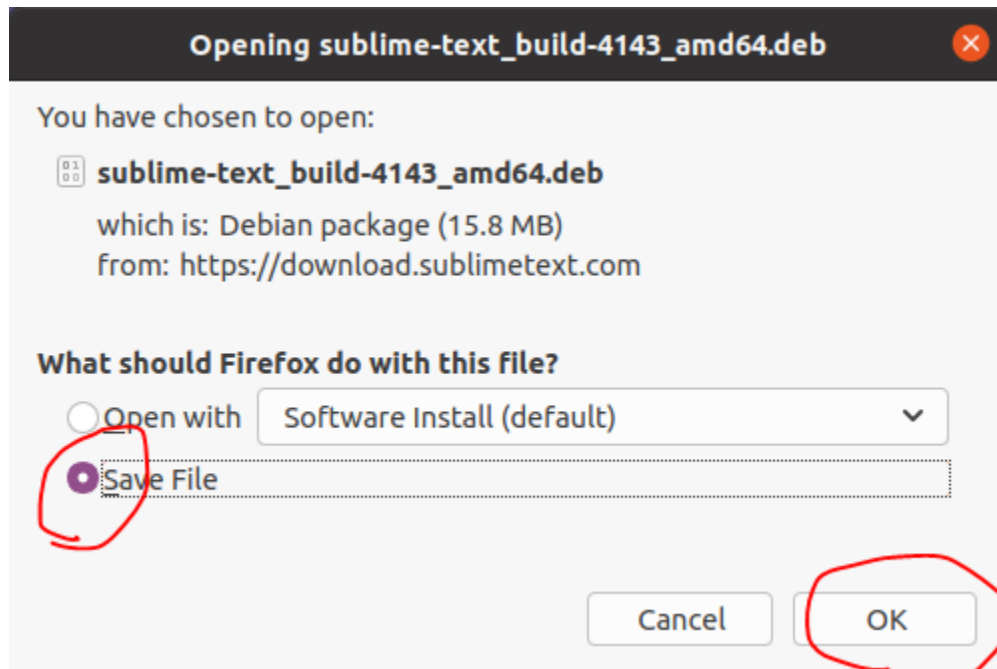
Sublime Text 4 is the current version of Sublime Text. For bleeding-edge releases, see the [dev builds](#).

Version: Build 4143

- [macOS](#)
- [Windows](#) - also available as a [portable version](#)
- [Linux repos - direct downloads](#)
 - [64 bit .deb](#) - [sig](#), [key](#)
 - [64 bit .rpm](#) - [signed](#), [key](#)
 - [64 bit .pkg.tar.xz](#) - [sig](#), [key](#)
 - [64 bit .tar.xz](#) - [sig](#), [key](#)
 - [ARM64 .deb](#) - [sig](#), [key](#)
 - [ARM64 .tar.xz](#) - [sig](#), [key](#)

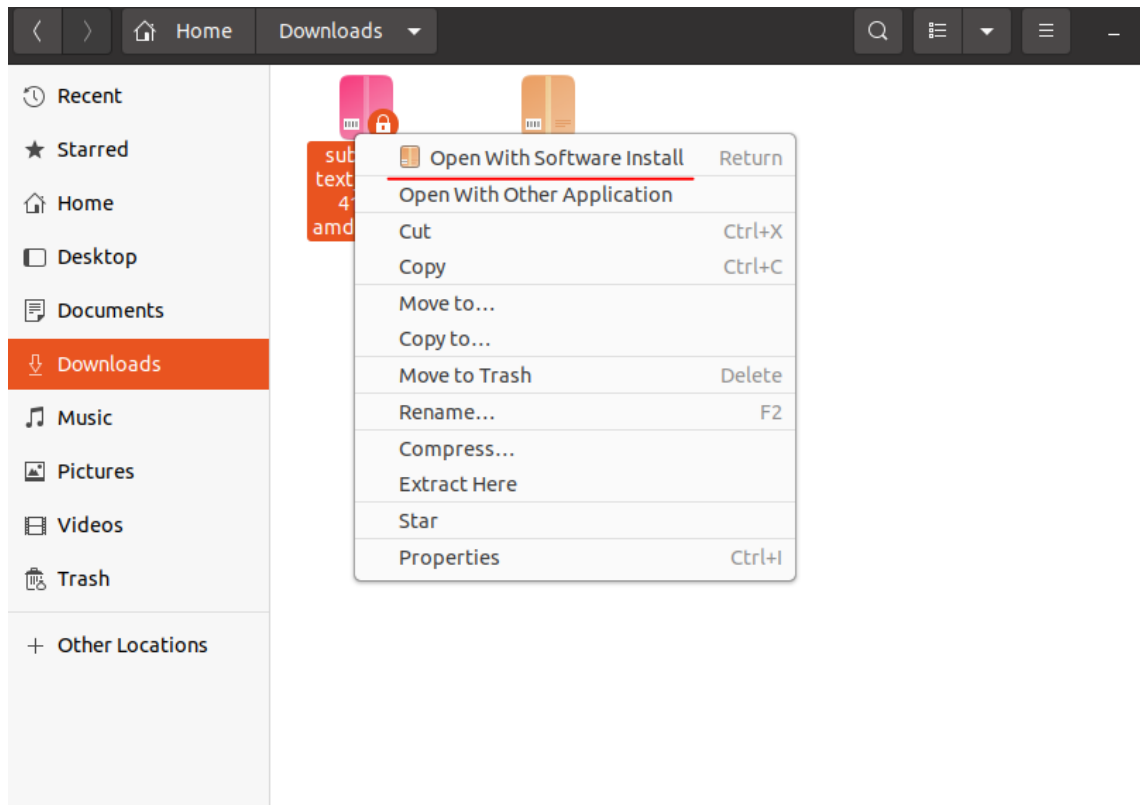
Installation

- Optional
 - Sublime text installation



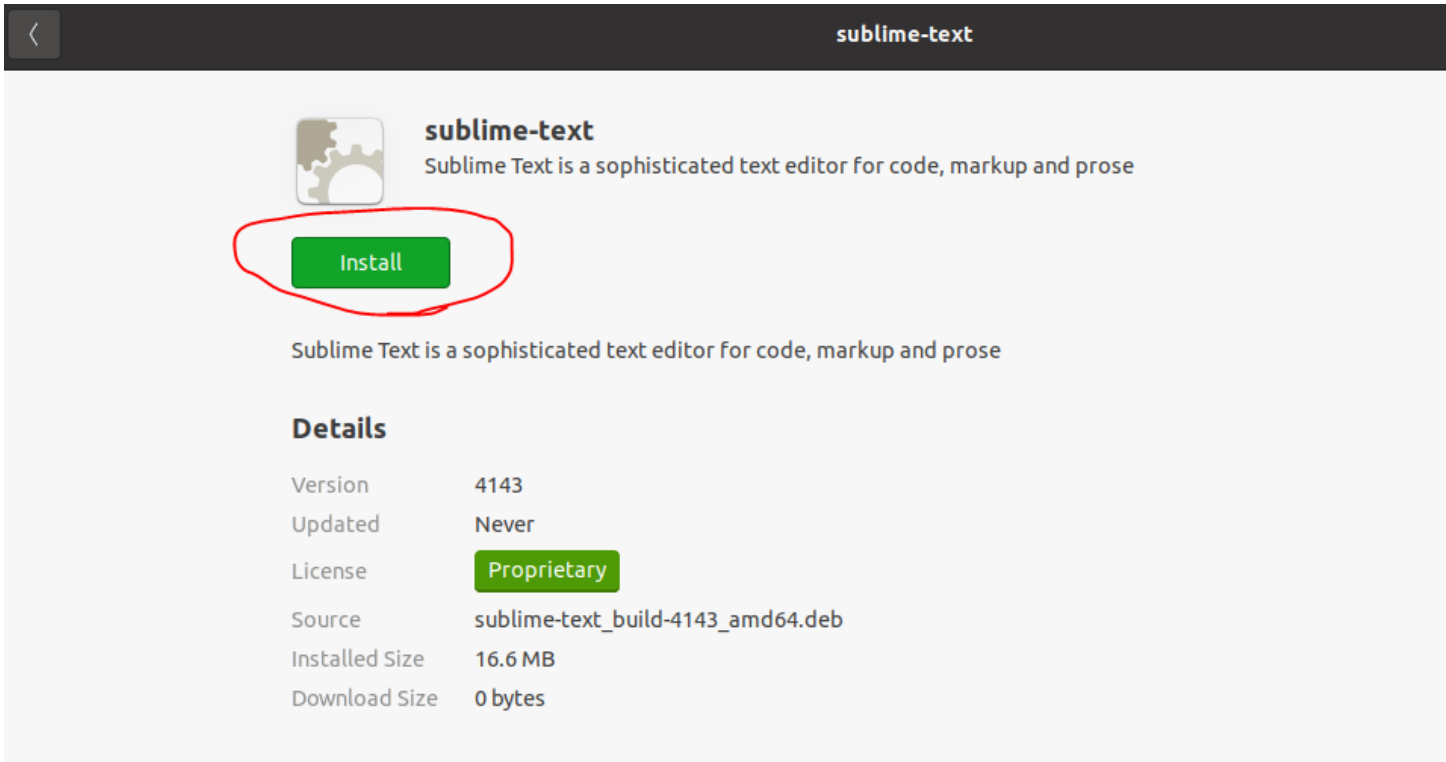
Installation

- Optional
 - Sublime text installation



Installation

- Optional
 - Sublime text installation



The screenshot shows the package page for 'sublime-text' in a package manager. At the top, there is a dark header with a back arrow on the left and the package name 'sublime-text' on the right. Below the header, on the left, is a square icon with a gear and a sun. To the right of the icon, the package name 'sublime-text' is displayed in bold, followed by the description 'Sublime Text is a sophisticated text editor for code, markup and prose'. A green 'Install' button is positioned below the icon and description, and it is circled in red. Below the 'Install' button, the same description is repeated. Further down, a 'Details' section is shown, containing a list of package attributes and their values.

Version	4143
Updated	Never
License	Proprietary
Source	sublime-text_build-4143_amd64.deb
Installed Size	16.6 MB
Download Size	0 bytes

Brief tutorial

- ns-3 key abstraction
 - Node
 - A computing device that connects to a network
 - Application
 - ns-3 applications run on ns-3 Nodes to drive simulations in the simulated world
 - Basic abstraction for a user program that generates some activity
 - Channel
 - The media over which data flows in the network
 - NetDevice
 - It can be regarded as a network interface card (NIC)
 - A net device is “installed” in a Node in order to enable the Node to communicate with other Nodes in the simulation via Channels
 - Topology helpers
 - It provides many convenient operations, such as create a NetDevice, add a MAC address, assigning IP address, connect the NetDevice to a Channel, etc.

Brief tutorial

- Tutorial script
 - From now on, ns-allinone-3.36.1/ns-3.36.1 is considered as a base directory
 - You can find tutorial scripts in the examples/tutorial directory

```
ta@ta-VirtualBox: ~/Desktop/ns-allinone-3.36.1/ns-3.36.1/examples/tutorial
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ ls
AUTHORS  build-support  CMakeLists.txt  doc      ns3      scratch  testpy.supp  VERSION
bindings  CHANGES.md   contrib         examples  README.md  src      utils
build    cmake-cache   CONTRIBUTING.md  LICENSE  RELEASE_NOTES.md  test.py  utils.py
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ cd examples/tutorial
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1/examples/tutorial$ ls
CMakeLists.txt  first.cc  hello-simulator.cc  seventh.cc  third.py
examples-to-run.py  first.py  second.cc           sixth.cc   tutorial-app.cc
fifth.cc        fourth.cc  second.py          third.cc   tutorial-app.h
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1/examples/tutorial$
```

- In this lecture, we will study one tutorial script
- Let's open the first.cc with a sublime text (or with your favorite editor)
subl first.cc

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1/examples/tutorial$ subl first.cc
```

Note: ls (list segment), allows to list directory

subl (sublime text), open file or directory with the sublime text editor

Brief tutorial

- first.cc
 - Module includes

```
17 #include "ns3/core-module.h"
18 #include "ns3/network-module.h"
19 #include "ns3/internet-module.h"
20 #include "ns3/point-to-point-module.h"
21 #include "ns3/applications-module.h"
```

- ns-3 provides many modules to make it much easier for users to write simulation scripts
- Each of the ns-3 include files is placed in a /build/include/ns3 directory
 - You can take a look at the contents of these files

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36.1/ns-3.36.1/build/include/ns3$ ls
a2-a4-rsrq-handover-algorithm.h    lte-rrc-header.h
a3-rsrq-handover-algorithm.h      lte-rrc-protocol-ideal.h
aarfcd-wifi-manager.h             lte-rrc-protocol-real.h
aarf-wifi-manager.h               lte-rrc-sap.h
abort.h                            lte-spectrum-phy.h
acoustic-modem-energy-model.h      lte-spectrum-signal-parameters.h
acoustic-modem-energy-model-helper.h lte-spectrum-value-helper.h
address.h                          lte-stats-calculator.h
address-utils.h                    lte-ue-ccm-rrc-sap.h
adhoc-aloah-noack-ideal-phy-helper.h lte-ue-cmac-sap.h
adhoc-wifi-mac.h                  lte-ue-component-carrier-manager.h
aloah-noack-mac-header.h          lte-ue-cphy-sap.h
aloah-noack-net-device.h          lte-ue-mac.h
ampdu-subframe-header.h           lte-ue-net-device.h
ampdu-tag.h                        lte-ue-phy.h
amrr-wifi-manager.h               lte-ue-phy-sap.h
amsdu-subframe-header.h           lte-ue-power-control.h
angles.h                           lte-ue-rrc.h
```

Brief tutorial

- first.cc
 - Namespace declaration

```
30 using namespace ns3;
```

- After this declaration, you will not have to type ns3:: scope resolution operator before all of the ns-3 code in order to use it
 - If you are not familiar with the concept of the namespace in C++, please visit this site: <https://www.cplusplus.com/doc/oldtutorial/namespaces/>

- Log component define

```
32 NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");
```

- This line declares a logging component called FirstScriptExample that allows you to enable and disable console message logging by reference to the name
 - Details about logging are discussed later

- Time resolution setting

```
40 Time::SetResolution (Time::NS);
```

- This line sets the time resolution to one nanosecond
- The resolution is the smallest time value that can be represented

Brief tutorial

- first.cc

- Enable two logging components

```
41 LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);  
42 LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
```

- These two lines of code enable debug logging at the INFO level for echo clients and servers
 - Details about logging are discussed later

- Node container

```
44 NodeContainer nodes;  
45 nodes.Create (2);
```

- The **Node**Container topology helper provides a convenient way to create, manage and access any **Node** objects

- PointToPointHelper

```
47 PointToPointHelper pointToPoint;  
48 pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));  
49 pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
```

- The first line instantiates a `PointToPoint` **Helper** object to configure and connect `PointToPointNetDevice` and `PointToPointChannel` objects
 - The second line tells the `PointToPointHelper` object to use the value “5Mbps” as the “DataRate” when it creates a `PointToPointNetDevice` object

Brief tutorial

- first.cc

- The third line tells the `PointToPointHelper` to use the value “2ms” as the value of the propagation delay of when it creates `PointToPointChannel`

- `NetDeviceContainer`

```
51 NetDeviceContainer devices;  
52 devices = pointToPoint.Install (nodes);
```

- The `Install` method of the `PointToPointHelper` takes a `NodeContainer` as a parameter, and creates `NetDeviceContainer`
- For each node in the `NodeContainer` a `PointToPointNetDevice` is created
- A `PointToPointChannel` is created and the two `PointToPointNetDevices` are attached
- After executing the `PointToPoint.Install (nodes)` call, we will have two **nodes**, each with an installed point-to-point **net device** and a single point-to-point **channel** between them

- `InternetStackHelper`

```
54 InternetStackHelper stack;  
55 stack.Install (nodes);
```

- It will install an Internet Stack (TCP, UDP, IP, etc.) on each of the **nodes** in the node container.

Brief tutorial

- first.cc
 - Ipv4AddressHelper

```
57 Ipv4AddressHelper address;  
58 address.SetBase ("10.1.1.0", "255.255.255.0");
```

- We need to associate the **devices** on our **nodes** with IP addresses
- The code above declares an address **helper** object and tell it that it should begin allocating IP addresses from the network 10.1.1.0 using the mask 255.255.255.0

```
60 Ipv4InterfaceContainer interfaces = address.Assign (devices);
```

- address.Assign performs the address assignment
- It makes the association between an IP address and a **device** using an Ipv4Interface object
- To contain a list of Ipv4Interface objects for future reference, we use Ipv4InterfaceContainer

Brief tutorial

- first.cc
 - UdpEchoServerHelper

```
62 UdpEchoServerHelper echoServer (9);
63
64 ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
65 serverApps.Start (Seconds (1.0));
66 serverApps.Stop (Seconds (10.0));
```

- The first line, declares an **UdpEchoServerHelper** and gets a port number **9** as a parameter
- The **Install** method takes a **NodeContainer** as a parameter just as the other **Install** methods
- **Install** will return a container that holds pointers to all of the applications created by the helper
- Applications require a time to “start” and may take an optional time to “stop”
- We set the echo server application to **Start** (enable itself) at one second into the simulation and to **Stop** (disable itself) at ten seconds into the simulation.

Brief tutorial

- first.cc
 - UdpEchoClientHelper

```
68 UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
69 echoClient.SetAttribute ("MaxPackets", UIntegerValue (1));
70 echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
71 echoClient.SetAttribute ("PacketSize", UIntegerValue (1024));
72
73 ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
74 clientApps.Start (Seconds (2.0));
75 clientApps.Stop (Seconds (10.0));
```

- The first line, declares an **UdpEchoClientHelper** and gets IP address and port number of the server as a parameter
 - Recall that we used an `Ipv4InterfaceContainer` to keep track of the IP addresses we assigned to our devices
- **SetAttribute**
 - **MaxPackets**: maximum number of packets we allow client to send
 - **Interval**: it tells the client how long to wait between packets
 - **PacketSize**: it tells how large client's packet payloads should be
- As with the `UdpServer` case, we set the start and end times for the `UdpClient` application

Brief tutorial

- first.cc
 - Simulator
 - Running the simulation is done by using the global function `Simulator::Run`

```
77 Simulator::Run ();
```

- Recall that we scheduled events in the 1, 2 and 10 seconds (with UdpEcho Client & Server applications)
- When `Simulator::Run` is called, the system will begin looking through the list of scheduled events and
- The act of sending the packet to the server will trigger a chain of events that will be automatically scheduled executing them
- When there are no further events to process, `Simulator::Run` returns

```
78 Simulator::Destroy ();
```

- Finally, `Simulator::Destroy` calls will deal with the hard part of destroying all created objects

Brief tutorial

- Building your first script
 - Let's copy (cp) examples/tutorial/first.cc into the **scratch** directory
 - The command is executed in the base directory

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ cp examples/tutorial/first.cc scratch/myfirst.cc
```

- Then, build your first example script using ns3:

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ ./ns3
```

- You can now run the example:

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ ./ns3 run scratch/myfirst
At time +2s client sent 1024 bytes to 10.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9
```

- You can see the logging component on the echo client indicate that it has sent a 1024 bytes packet to the Echo Server on 10.1.1.2
- You can also see the logging component on the echo server say that it has received the 1024 bytes from 10.1.1.1
- The echo server silently echoes the packet and you can see the echo client log that it has received its packet back from the server

Brief tutorial

- Logging
 - ns-3 supports some kind of message logging facility
 - ns-3 provide a selectable, multi-level (7) approach to message logging
 - LOG_ERROR: log error messages
 - LOG_WARN: log warning messages
 - LOG_DEBUG: log debugging messages
 - LOG_INFO: Log informational messages about program progress
 - LOG_FUNCTION: Log a message describing each function called
 - LOG_LOGIC: Log messages describing logical flow within a function
 - LOG_ALL: Log everything mentioned above
 - ns-3 also provides an unconditional logging macro that is always displayed
 - NS_LOG_UNCOND
 - Recall that myfirst.cc script contains the code below:

```
41 LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);  
42 LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
```

- We can increase the logging level and get more information without changing the script and recompiling by setting the NS_LOG environment variable like this:

```
~/Desktop/ns-allinone-3.36/ns-3.36$ export NS_LOG=UdpEchoClientApplication=level_all
```

- export NS_LOG=UdpEchoClientApplication=level_all

Brief tutorial

- Logging
 - Now, let's run myfirst.cc script again:

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ ./ns3 run scratch/myfirst
UdpEchoClientApplication:UdpEchoClient(0x55e290805ed0)
UdpEchoClientApplication:SetDataSize(0x55e290805ed0, 1024)
UdpEchoClientApplication:StartApplication(0x55e290805ed0)
UdpEchoClientApplication:ScheduleTransmit(0x55e290805ed0, +0ns)
UdpEchoClientApplication:Send(0x55e290805ed0)
At time +2s client sent 1024 bytes to 10.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
UdpEchoClientApplication:HandleRead(0x55e290805ed0, 0x55e290817f50)
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9
UdpEchoClientApplication:StopApplication(0x55e290805ed0)
UdpEchoClientApplication:DoDispose(0x55e290805ed0)
UdpEchoClientApplication:~UdpEchoClient(0x55e290805ed0)
```

- You can see that additional debug information provided by the application
- Let's find out what log information UdpEchoClientApplication provides

```
~/Desktop/ns-allinone-3.36/ns-3.36$ find . -name '*.cc' | xargs grep UdpEchoClientApplication
```

- `find . -name '*.cc'` command finds every .cc (*.cc) file from current directory (.)
- The following command after pipeline (|) gets result of the find command as an input and check if there is a word UdpEchoClientApplication in the input file and display the word if it exists

Brief tutorial

- Logging

```
./src/lte/examples/lena-ipv6-ue-rh.cc: LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_ALL);  
./src/applications/model/udp-echo-client.cc: NS_LOG_COMPONENT_DEFINE ("UdpEchoClientApplication");  
./src/nix-vector-routing/examples/nix-simple.cc: LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
```

- The result shows that UdpEchoClientApplication log component is defined in ./src/applications/model/udp-echo-client.cc. Lets take a look at the file:

```
~/Desktop/ns-allinone-3.36/ns-3.36$ subl ./src/applications/model/udp-echo-client.cc
```

- If you search NS_LOG in the script using ctrl+F command, you can find many logging code in it

```
UdpEchoClient::SetRemote (Address ip, uint16_t port)  
{  
    NS_LOG_FUNCTION (this << ip << port);
```

```
        NS_LOG_INFO ("At time " << Simulator::Now ().As (Time::S) << " client received "  
                    Inet6SocketAddress::ConvertFrom (from).GetIpv6 () << " port " <<  
                    Inet6SocketAddress::ConvertFrom (from).GetPort ());  
    }  
    socket->GetSockName (localAddress);  
    m_rxTrace (packet);
```

NS_LOG

Brief tutorial

- Logging
 - From now, let's add logging code to myfirst script
 - Open myfirst.cc script and then, add Info level logging code in line 43 (and save)

```
42 LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
43 NS_LOG_INFO("Creating topology");
44 NodeContainer nodes;
45 nodes.Create (2);
```

- build myfirst script again and reset the NS_LOG variable again

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ ./ns3
Scanning dependencies of target scratch_myfirst
[ 0%] Building CXX object scratch/CMakeFiles/scratch_myfirst.dir/myfirst.cc.o
[ 0%] Linking CXX executable ../../build/scratch/ns3.36-myfirst-default
Finished executing the following commands:
cd cmake-cache; cmake --build . -j 1 ; cd ..
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ NS_LOG=
```

- Recall that myfirst.cc log component is defined as FirstScriptExample
- Enable the FirstScriptExample logging level into LOG_INFO and run the script:

```
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ NS_LOG=FirstScriptExample=level_info
ta@ta-VirtualBox:~/Desktop/ns-allinone-3.36/ns-3.36$ ./ns3 run scratch/myfirst.cc
Creating topology
At time +2s client sent 1024 bytes to 10.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9
```

- You can see the log you just made!