

Setting Up

A GUIDE TO SETTING UP YOUR VIRTUAL MACHINE FOR PYTHON

UPDATED 12/31/16

Why use a virtual machine?

Before we begin, some motivation. Python can be installed on your host OS and many of the editors, tools, and packages we use in this class might work on your host OS as well. However, there's no guarantee!

To make sure we are all using the same environment, I strongly suggest that you set up a VM as directed here. There are two advantages to this:

1. You can be sure that I will never require you to use a package or tool that is unavailable or problematic for the environment you're using.
2. If you encounter any problems in this environment, I will always be able to help you.

If you decide to install and use Python directly in your host OS X on your Mac, for instance, you will be on your own later in the semester when (not “if”, “when”) you run into problems.

Get A Virtual Box!

You can use whichever virtualization software you want, but Virtual Box is recommended.

Go to <https://www.virtualbox.org/wiki/Downloads> and choose the appropriate installer.

Follow the installation instructions and start up Virtual Box.

Choose a platform!



The screenshot shows the VirtualBox website's 'Downloads' page. At the top is a large blue header with the 'VirtualBox' logo. Below it is a sub-header 'Download VirtualBox'. A green callout box highlights the 'VirtualBox 5.1.12 platform packages' section, which lists 'Windows hosts', 'OS X hosts', 'Linux distributions', and 'Solaris hosts'. The 'VirtualBox 5.1.12 Oracle VM VirtualBox Extension Pack' section is also visible.

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 I

- **VirtualBox 5.1.12 platform packages.** The binaries are released u
 - o ↗ Windows hosts
 - o ↗ OS X hosts
 - o ↗ Linux distributions
 - o ↗ Solaris hosts
- **VirtualBox 5.1.12 Oracle VM VirtualBox Extension Pack** ↗ All su
 - Support for USB 2.0 and USB 3.0 devices, VirtualBox RDP, disk encry
 - The Extension Pack binaries are released under the [VirtualBox Person](#)
 - Please install the extension pack with the same version as your inst*

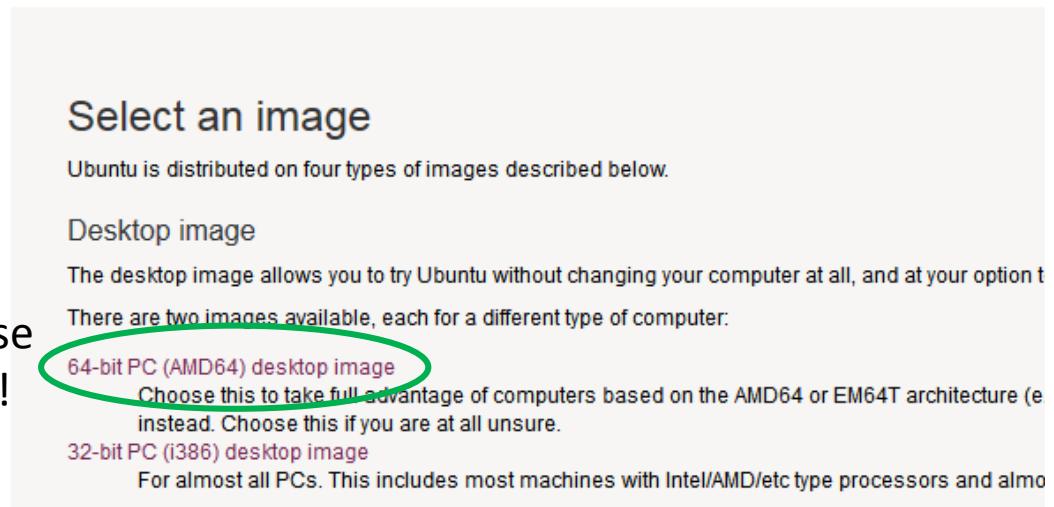
Choose an OS

You are encouraged to use Ubuntu 16.04, the latest LTS release of Ubuntu. You could alternatively use a lighter-weight Debian-based Linux distribution, but your mileage may vary.

Download the Ubuntu 16.04 iso from here: <http://mirror.pnl.gov/releases/xenial/>

Ubuntu 16.04.1 LTS (Xenial Xerus)

Choose
me!



Select an image

Ubuntu is distributed on four 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.

There are two images available, each for a different type of computer:

64-bit PC (AMD64) desktop image Choose this to take full advantage of computers based on the AMD64 or EM64T architecture (e.g. most new PCs). Choose this if you are at all unsure.

32-bit PC (i386) desktop image For almost all PCs. This includes most machines with Intel/AMD/etc type processors and almost

This may take some time so be prepared to just let it download for a while...

Make sure your VirtualBox version supports 64-bit Linux machines. You can check with the step on the next slide.

Set Up the VM

Now, create a virtual machine inside of Virtual Box.

1. Click the New button up in the top left.



Name and operating system

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

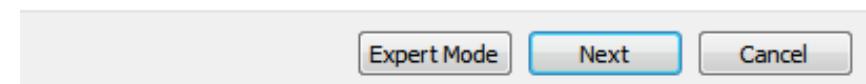
Name: 64

Type: ▼

Version: ▼

2. Fill in the basic information about your new virtual machine.

Note that 32 bit may be the only option. If this is the case, and you have a Windows machine, [click here](#).



Set Up the VM

3. Choose the RAM capacity of your machine.

The capacity you choose will depend on the physical RAM of your host machine. You should reserve at least 2 GB if possible.

Be aware that this RAM is reserved for the running virtual machine, whether it is using all of it or not. When the machine is suspended or shut off, you won't need to worry about sharing it.

Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **1024 MB**.



Next **Cancel**

Set Up the VM

4. Create a virtual hard drive.

You should choose the default option to create a new virtual hard drive.

5. Choose the default hard drive file type, VDI.

VDI stands for VirtualBox Disk Image.

6. Choose the dynamically-allocated option.

Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **8.00 GB**.

- Do not add a virtual hard disk
- Create a virtual hard disk now
- Use an existing virtual hard disk file

PythonMachine.vdi (Normal, 8.00 GB)



Create

Cancel

Set Up the VM

7. Finalize the hard drive. Go ahead and use the default name if you'd like, but it is recommended that you reserve at least 25 GB for the virtual hard drive.

File location and size

Please type the name of the new virtual hard disk file into the box below or click on the folder icon to select a different folder to create the file in.

PyMachine 

Select the size of the virtual hard disk in megabytes. This size is the limit on the amount of file data that a virtual machine will be able to store on the hard disk.



Create

Cancel

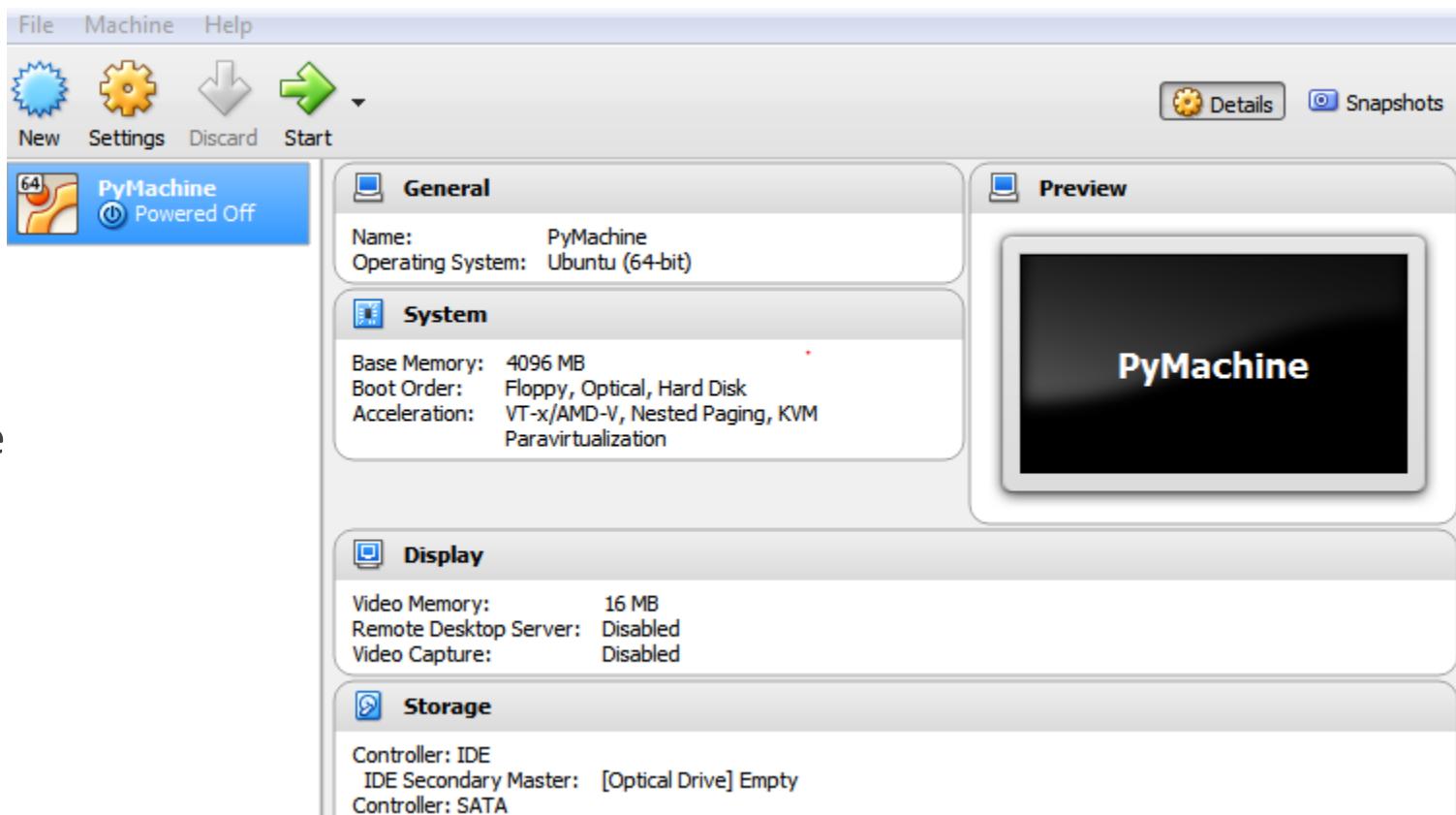
Set Up the VM

You're almost all done!

Your new virtual machine should appear in the main menu now along with some details about the machine.

We still haven't installed the OS. We've only created a new machine for the particular environment we want to create.

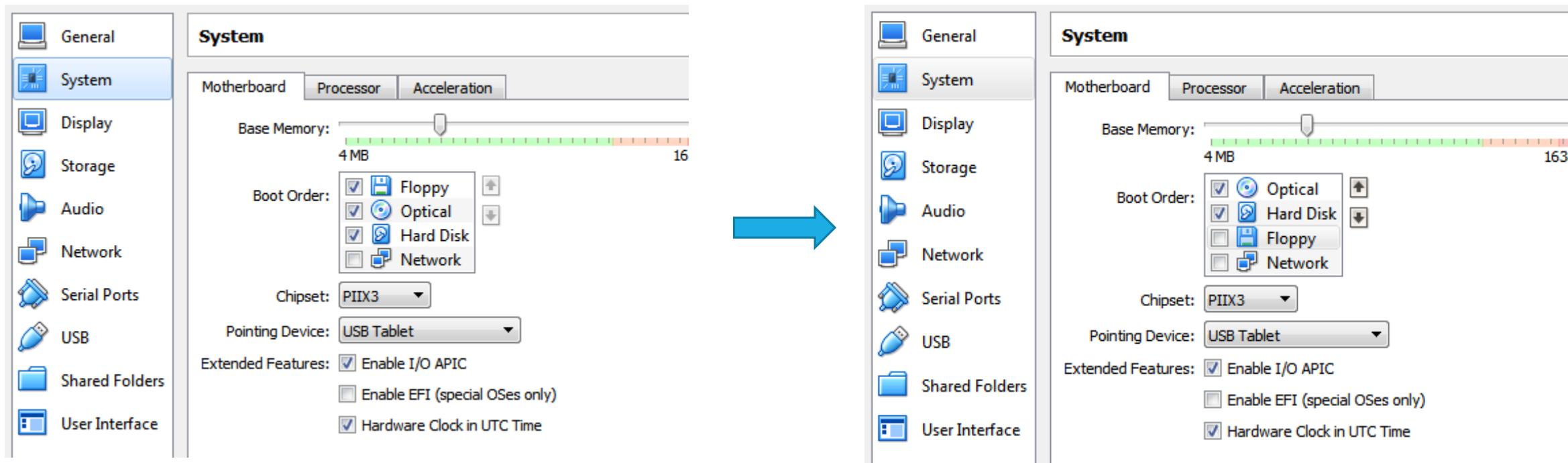
Right-click on the machine and choose Settings.



Set Up the OS

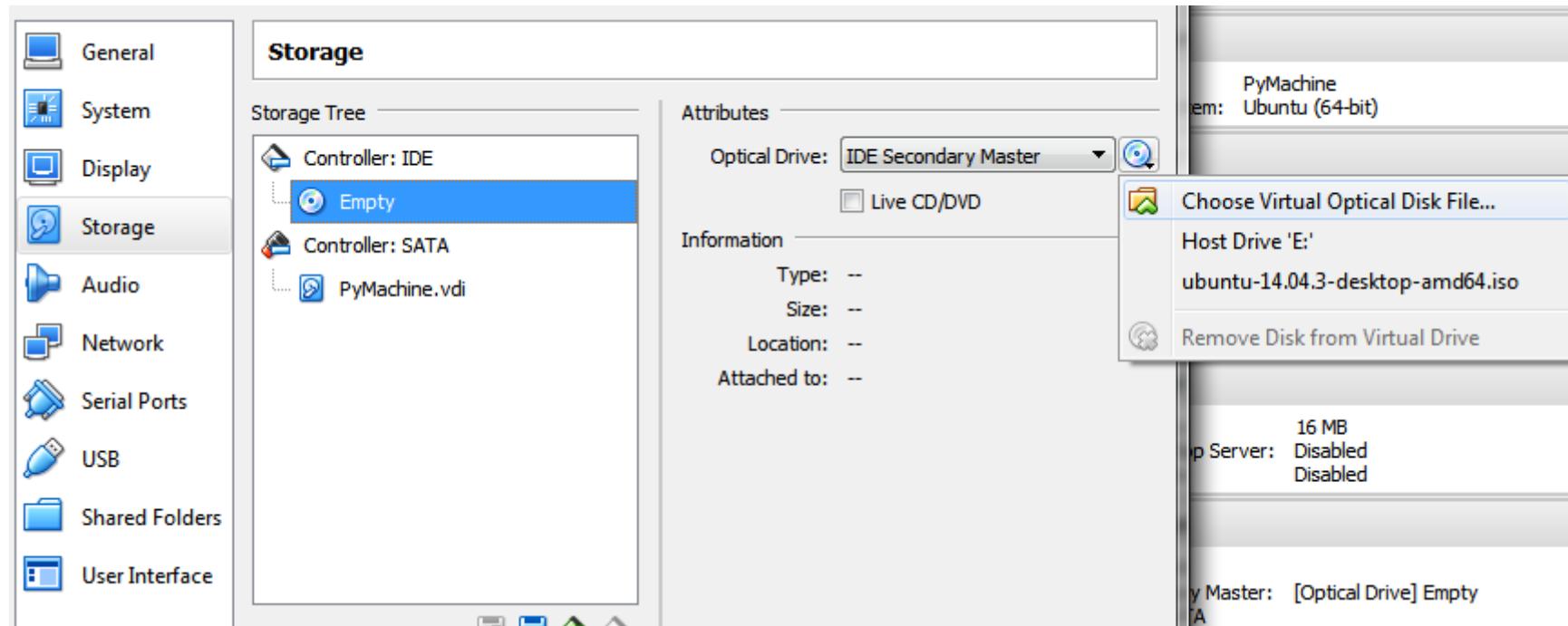
Choose System on the left side of the menu.

Uncheck Floppy and move it down in the boot order.



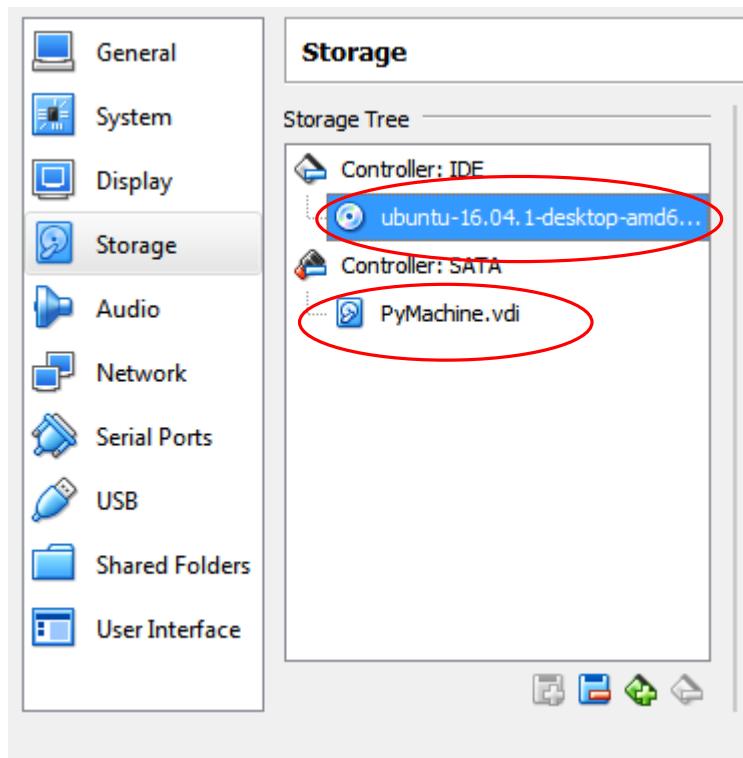
Set Up the OS

Now, choose Storage on the left-side menu. Select the Empty slot in the Controller Tree and then select *Choose Virtual Optical Disk File...* on the right-hand side as shown below.



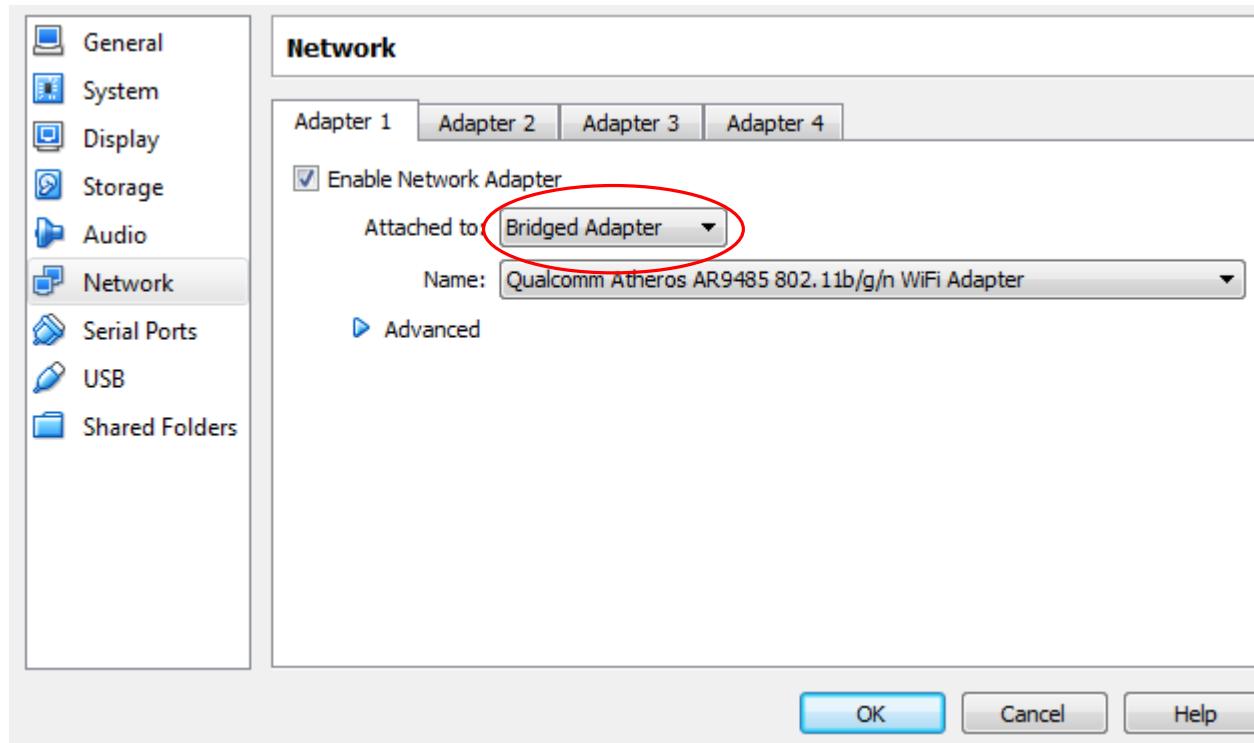
Set Up the OS

After navigating to select the iso you downloaded before, you should see your iso and newly created hard drive file selected.



Set Up the OS

Lastly, select Network on the left-side menu. Select Bridged Adaptor from the drop-down menu.



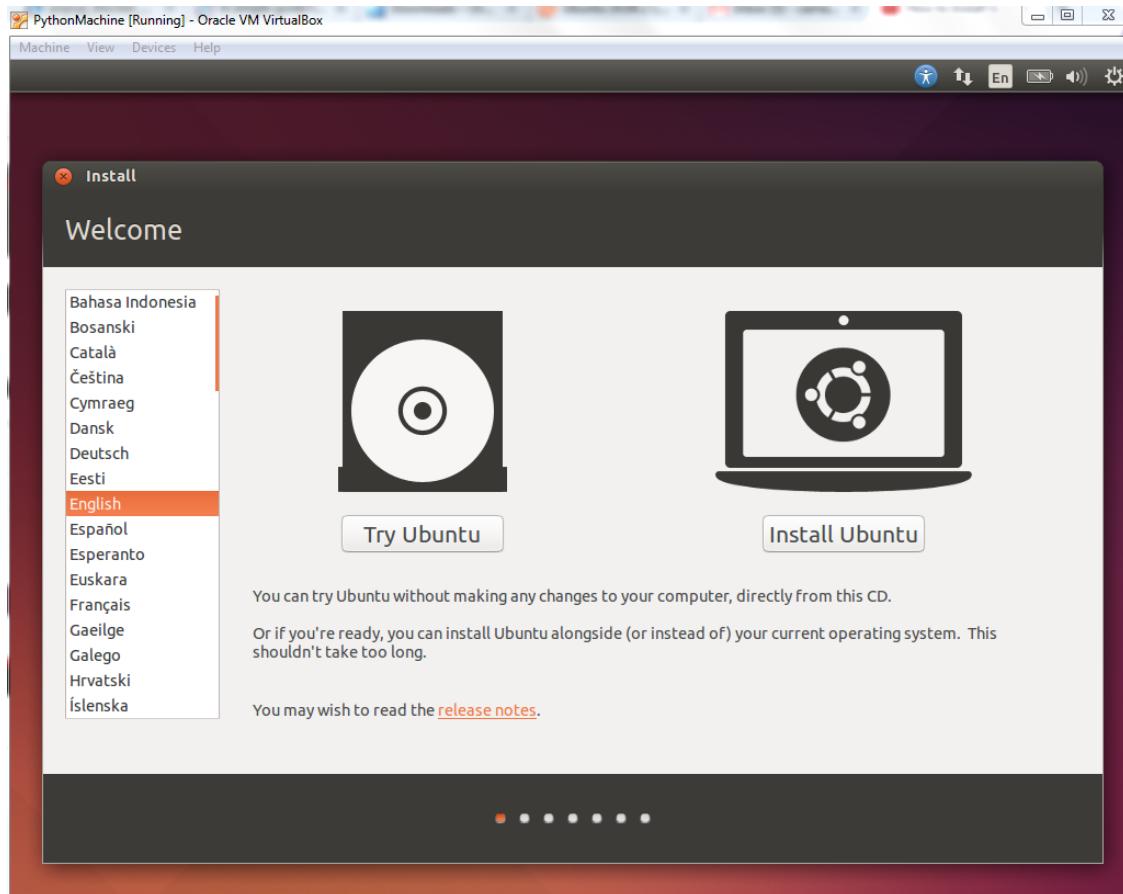
Set Up the OS

Now, exit the pop-up menu and select your virtual machine and press the Start button.

Wait for your machine to boot up. You may see some error messages but usually these are not fatal. Just give it some time.

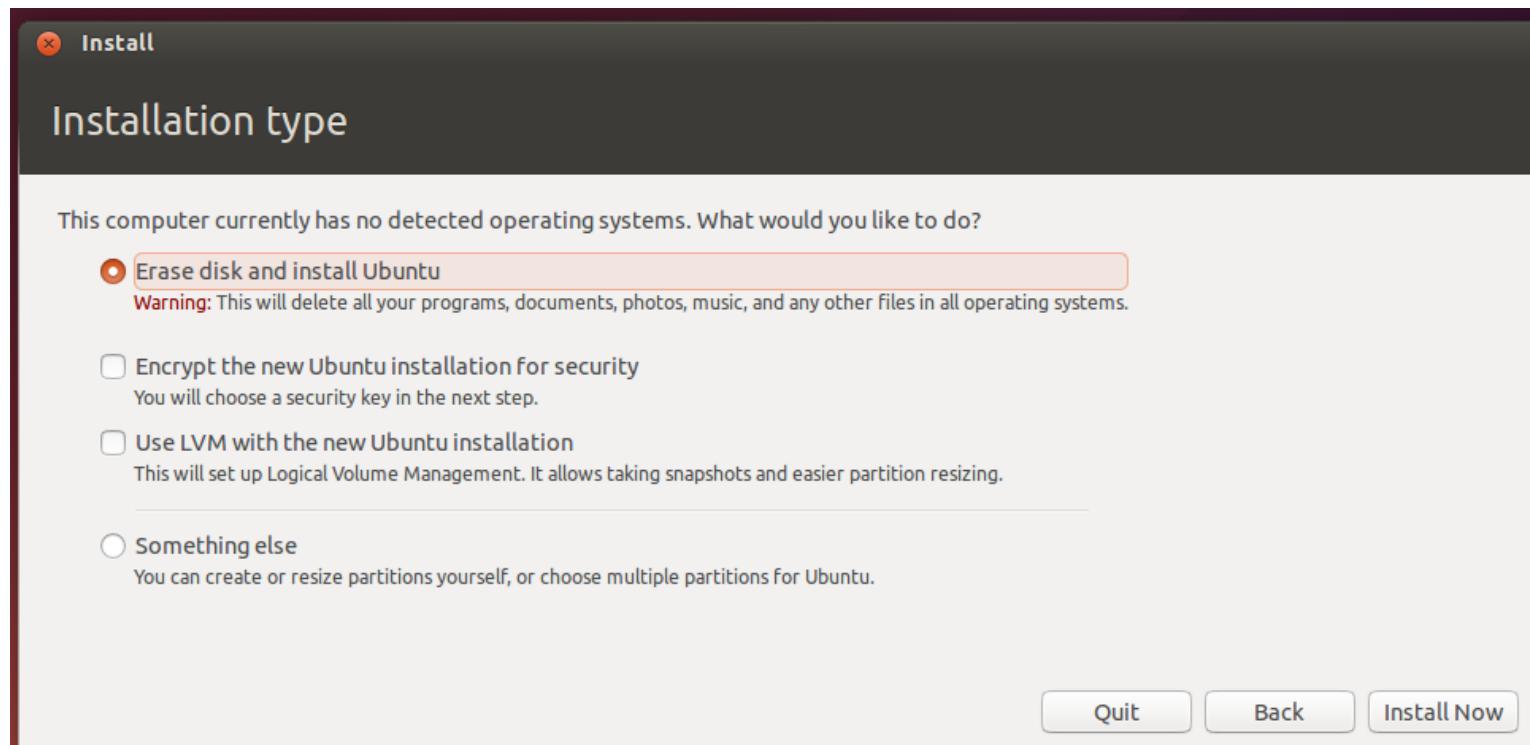
Install Ubuntu!

Note: I recommend *not* installing updates right away.



Set Up the OS

Continue with installation. Make sure you choose the option to Erase Disk and Install Ubuntu. Don't worry! – it won't affect your host OS. Allow the changes to the drive and install!



Supersize the VM

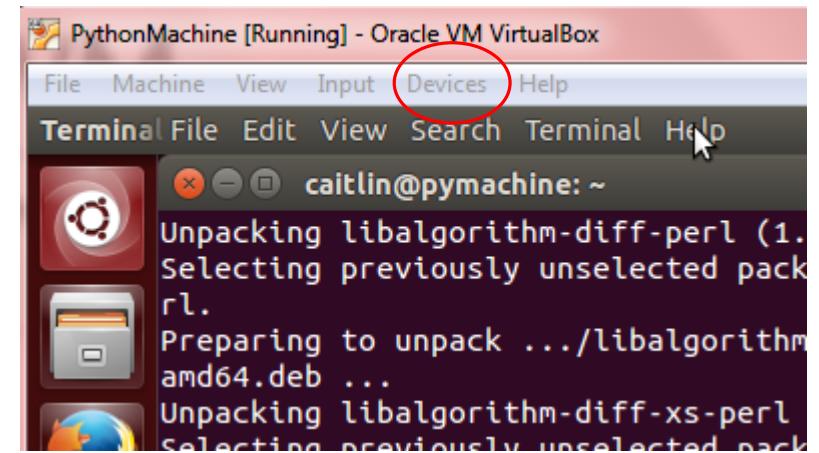
You may end up with a tiny screen for your VM. If this is the case, in Ubuntu pull up a terminal (ctrl-alt-t) and issue the following:

```
$ sudo apt-get update  
$ sudo apt-get upgrade  
$ sudo apt-get install dkms  
$ sudo apt-get install build-essential
```

Go to the Devices menu, tell it to install Guest Additions, and choose the run option.

Now, restart the VM.

Resize the VM window via System Settings in Ubuntu or choose Full Screen Mode and it will resize the desktop screen. ☺



Set Up Python

You're all set! Let's turn our attention to making sure Python and some key packages are installed:

- Python 2.7.12 or Python 3.5.2
Open up a terminal and type
'python' or 'python3' – it's already there!
- Pip
- Pylint

```
caitlin@pymachine:~$ python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
caitlin@pymachine:~$ python3
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

Note: if you're using an older version of Ubuntu and have trouble logging in with your Ubuntu username, try some of the actions outlined here <http://askubuntu.com/questions/454576/cant-login-to-ubuntu-14-04-after-upgrade>.

Set Up Python

Next we want pip, the de-facto Python package manager. To install pip, you simply need to issue the command

```
$ sudo apt-get install python-pip python3-pip
```

This will install pip 8.1.1 for python 2.7 and python 3.5. We will be using pip frequently in this class to install third –party libraries for our Python applications.

```
caitlin@pymachine:~$ pip -V
pip 8.1.1 from /usr/lib/python2.7/dist-packages (python 2.7)
caitlin@pymachine:~$ pip3 -V
pip 8.1.1 from /usr/lib/python3/dist-packages (python 3.5)
```

Set Up Python

Lastly, we'll grab pylint.

```
$ sudo apt-get install pylint pylint3
```

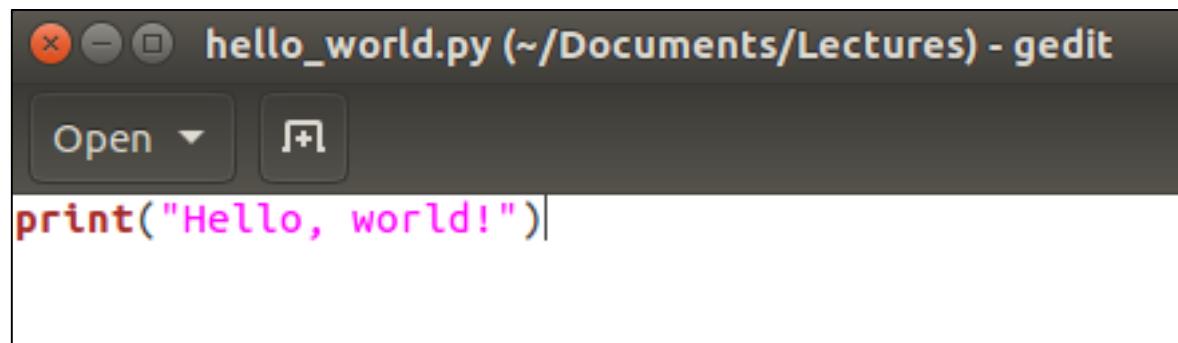
This will install pylint 1.5.2 and pylint3 1.6.4. We will be using pylint throughout the semester to make sure our code is conforming to Pythonic standards. It's important to maintain good habits while programming in Python.

Testing Our Environment

Let's create a little Python program to test everything. Open up the gedit text editor by issuing the following command:

```
$ gedit
```

Save the file as `hello_world.py` and type the following contents. Save the file.



A screenshot of the gedit text editor window. The title bar reads "hello_world.py (~/Documents/Lectures) - gedit". The menu bar has "File", "Edit", "View", "Insert", "Format", "Search", and "Help". The toolbar has "Open", "Save", "Print", and "New" buttons. The main text area contains the following Python code:

```
print("Hello, world!")
```

Testing Our Environment

We can run our Python program through the Python interpreter by issuing

```
$ python3 hello_world.py
```

or, in this case, we can also do:

```
$ python hello_world.py
```

We're all set! ☺

```
caitlin@pymachine:~/Documents/Lectures$ python3 hello_world.py
Hello, world!
caitlin@pymachine:~/Documents/Lectures$ python hello_world.py
Hello, world!
caitlin@pymachine:~/Documents/Lectures$ █
```

Its just a little coincidence that this tiny python program works for both Python 2 and Python 3. The reason why may not be immediately obvious.

In general, it is NOT the case that Python 2 and Python 3 are interchangeable!

Testing Our Environment

You should also try running the following commands and see what you get. In particular, look at the messages at the top.

```
$ pylint3 hello_world.py
```

```
$ pylint hello_world.py
```

Appendix: Enabling 64-bit Virtualization

You may encounter an issue where VirtualBox does not allow you to set up a 64-bit virtual machine and you only have 32-bit options.

To fix this, you will need to check two things:

1. Virtualization technology is enabled.

When powering on your machine, open up the BIOS menu and look for an option like "Intel Virtualization Technology" and make sure it's enabled.

2. Hyper-V or Windows Virtual PC are disabled.

Look for the tool that allows you to turn Windows features on and off. Turning off these virtualization features ensures that they are not grabbing ownership of virtualization tech at start up.

