University College of Engineering, (BIT CAMPUS),Tiruchirappalli

Department of Computer Science & Engineering

CS8711 CLOUD COMPUTING LABORATORY

VII SEMESTER CSE A&B

Prepared by

Mrs.T.Mithila & Mrs.R.Kavitha

July-Dec 2021

CS8711 CLOUD COMPUTING LABORATORY L T P C 0 0 3 2

OBJECTIVES:

The student should be made to:

- Be exposed to tool kits for grid and cloud environment.
- Be familiar with developing web services/Applications in grid framework
- Learn to run virtual machines of different configuration.
- Learn to use Hadoop

LIST OF EXPERIMENTS:

Course Objective:

- To develop web applications in cloud
- To learn the design and development process involved in creating a cloud based application
- To learn to implement and use parallel programming using Hadoop

Exercises:

1. Install Virtualbox/VMware Workstation with different flavours of linux or windows OS on top of windows7 or 8.
2. Install a C compiler in the virtual machine created using virtual box and execute Simple Programs
3. Install Google App Engine. Create hello world app and other simple web applications using python/java.
4. Use GAE launcher to launch the web applications.
5. Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.
6. Find a procedure to transfer the files from one virtual machine to another virtual machine.
7. Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)
8. Install Hadoop single node cluster and run simple applications like wordcount.

Course Outcome:

On completion of this course, the students will be able to:

- Configure various virtualization tools such as Virtual Box, VMware workstation.
- Design and deploy a web application in a PaaS environment.
- Learn how to simulate a cloud environment to implement new schedulers.
- Install and use a generic cloud environment that can be used as a private cloud.
- Manipulate large data sets in a parallel environment.

TOTAL: 45 PERIODS

CLOUD COMPUTING

EX.No:1 Install Virtualbox/VMware Workstation

Aim:

Find procedure to Install Virtualbox/VMware Workstation with different flavours of linux or windows OS on top of windows7 or 8.

This experiment is to be performed through portal.

PROCEDURE TO INSTALL

Step 1- Download Link
Link for downloading the software is https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html. Download the software for windows. Good thing is that there is no signup process. Click and download begins. Software is around 541 MB.

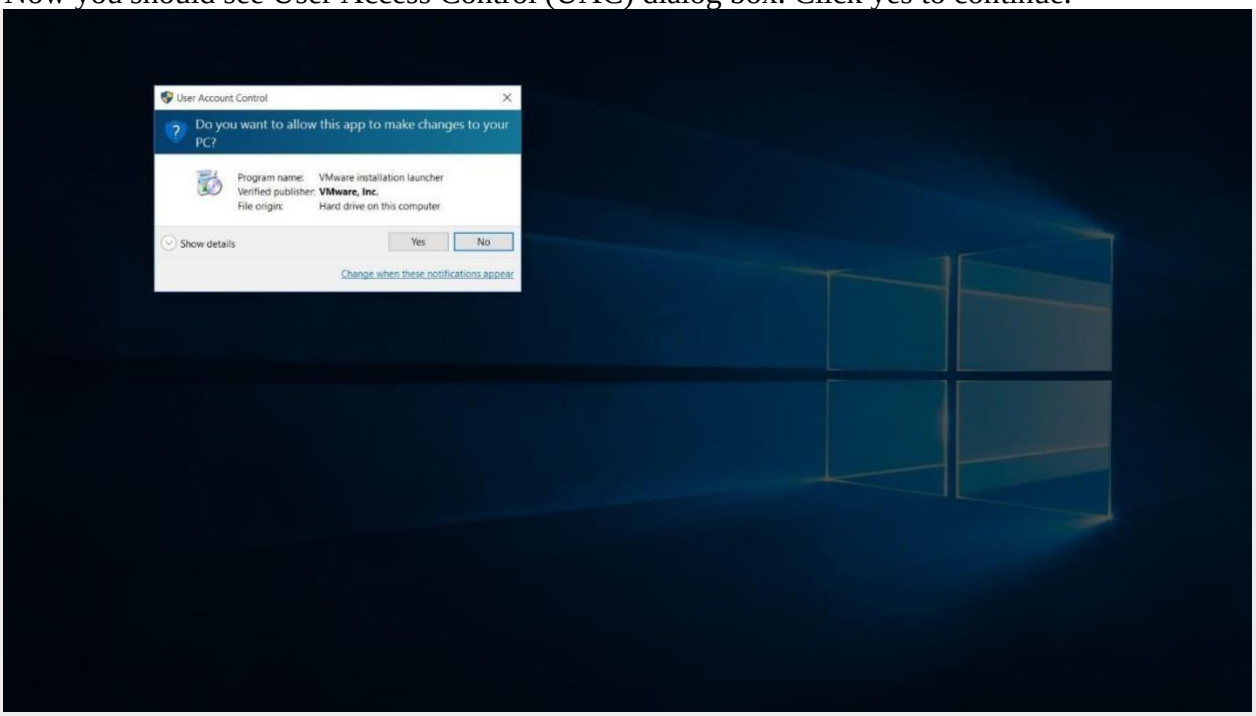
Step 2- Download the installer file
It should probably be in the download folder by default, if you have not changed the settings in your browser. File name should be something like VMware-workstation-full-15.5.1-15018445.exe. This file name can change depending on the version of the software currently available for download. But for now, till the next version is available, they will all be VMware Workstation 15 Pro.

Step 3- Locate the downloaded installer file
For demonstration purpose, I have placed the downloaded installer on my desktop. Find the installer on your system and double click to launch the application.

VMware workstation 15 pro for windows 10 installer file screenshot.

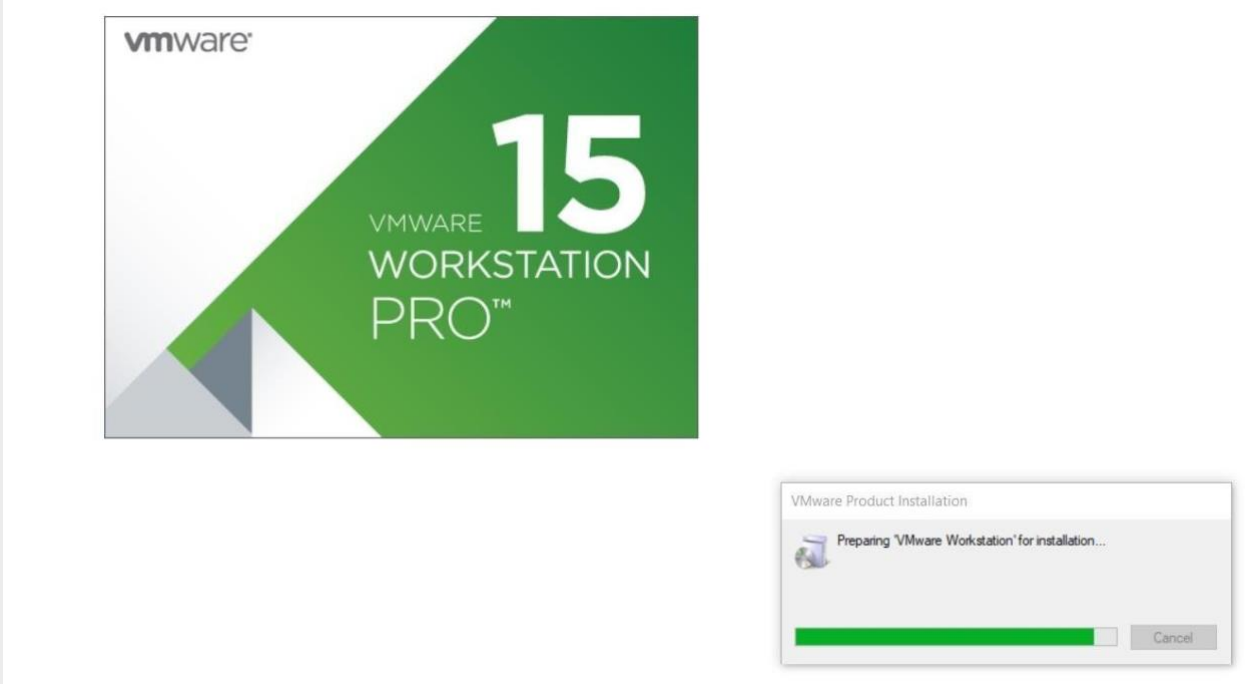
Step 4- User Access Control (UAC) Warning

Now you should see User Access Control (UAC) dialog box. Click yes to continue.



VMware Workstation 12 Pro installer windows 10 UAC screenshot

Initial Splash screen will appear. Wait for the process to complete.



VMware Workstation 15 Installation Splash Screen

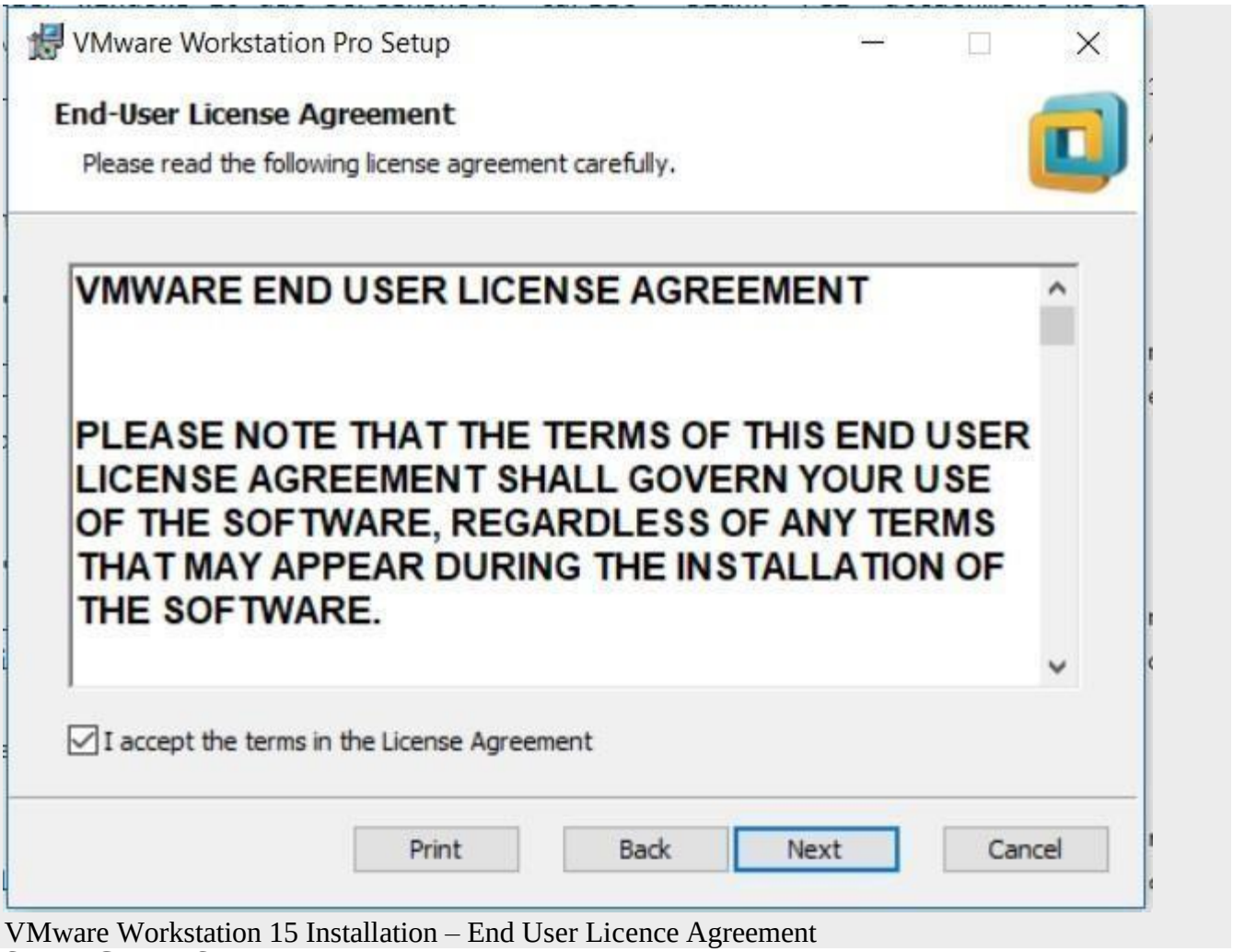
Step 5- VMware Workstation Setup wizard

Now you will see VMware Workstation setup wizard dialog box. Click next to continue.

VMware Workstation 15 Installation – Setup Wizard

Step 6- End User Licence Agreement

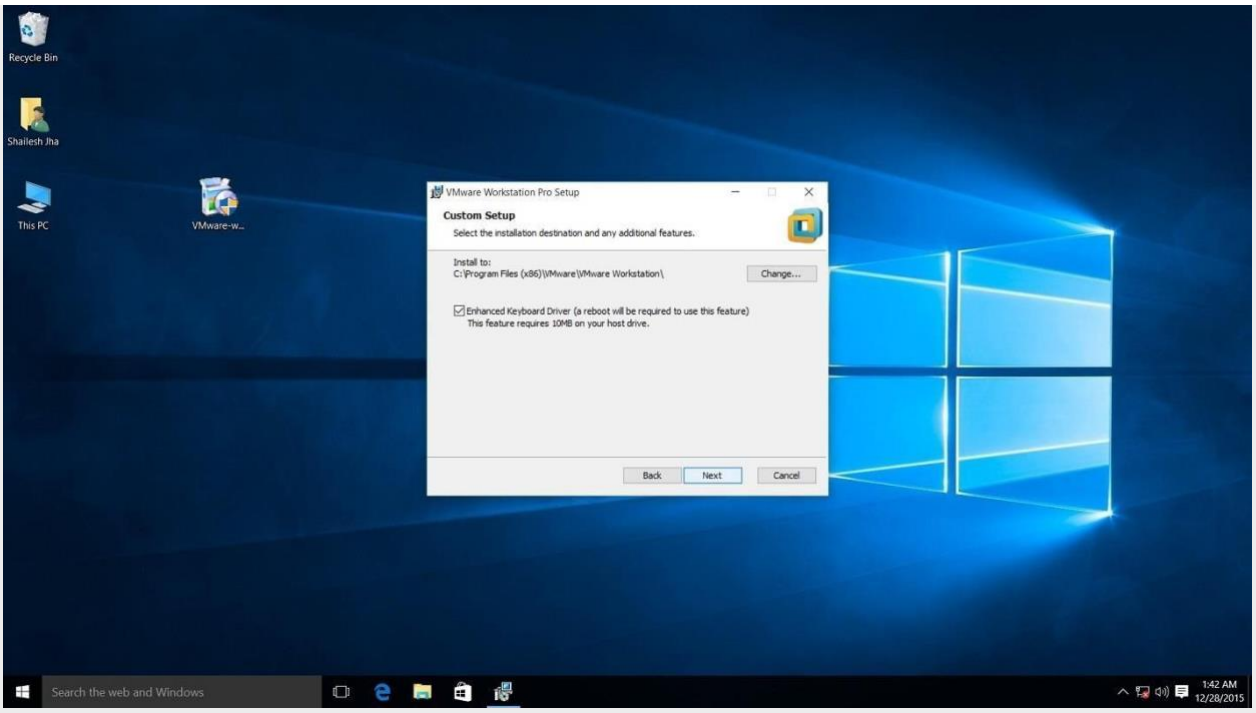
This time you should see End User Licence Agreement dialog box. Check “I accept the terms in the Licence Agreement” box and press next to continue.



VMware Workstation 15 Installation – End User Licence Agreement

Step 7- Custom Setup options

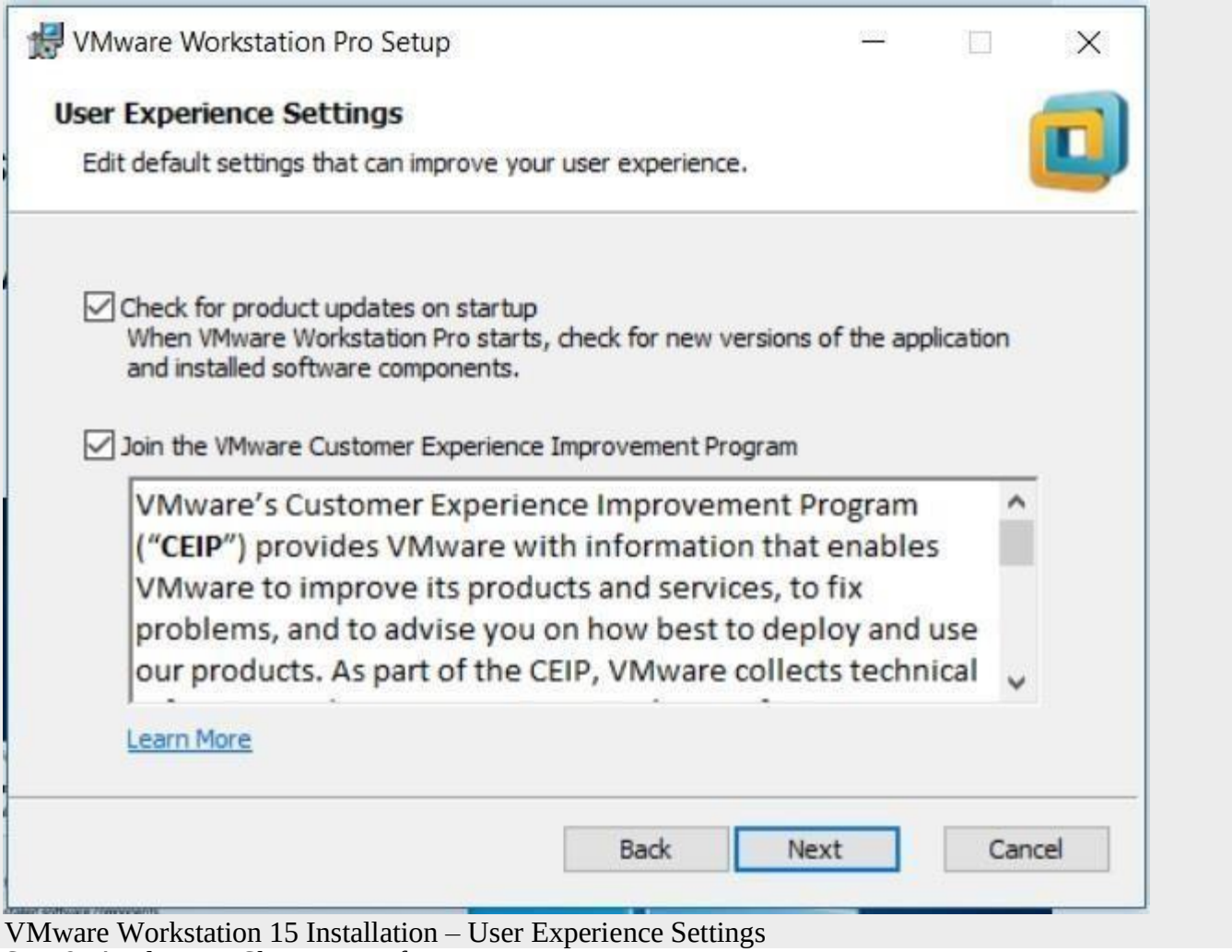
Select the folder in which you would like to install the application. There is no harm in leaving the defaults as it is. Also select Enhanced Keyboard Driver check box.



VMware Workstation 15 Pro installation – select installation folder

Step 8- User Experience Settings

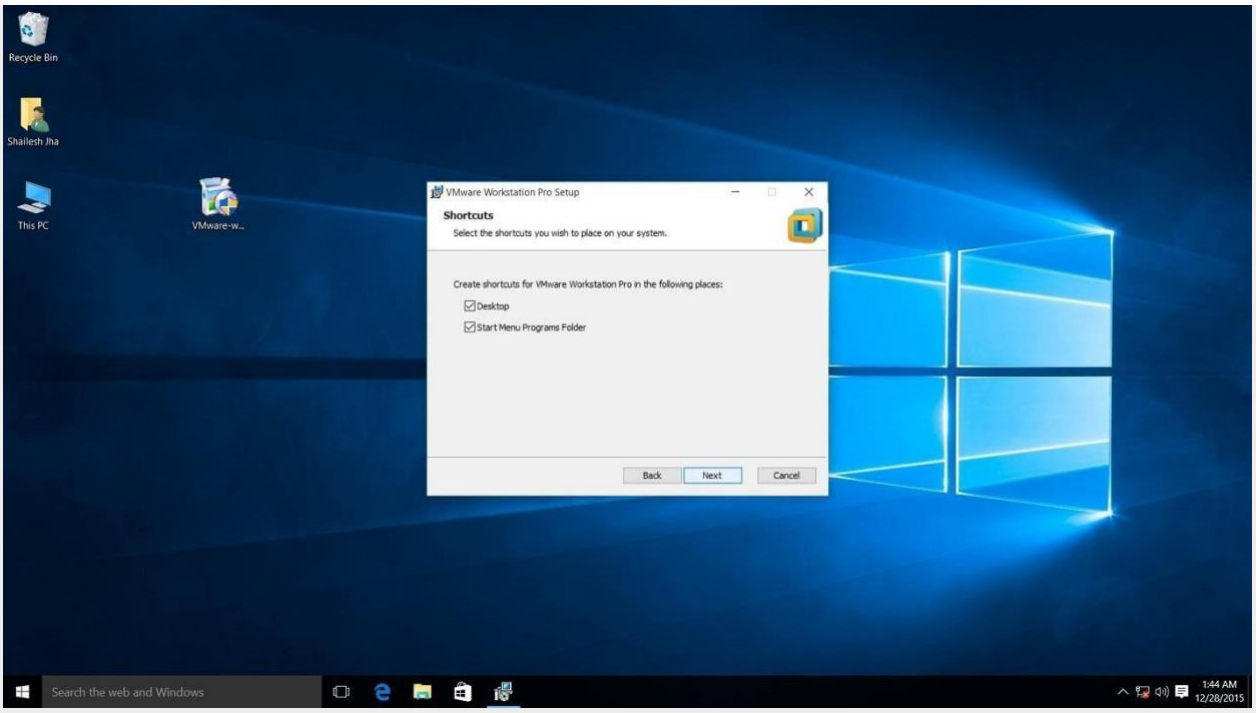
Next you are asked to select “Check for Updates” and “Help improve VMware Workstation Pro”. Do as you wish. I normally leave it to defaults that is unchecked.



VMware Workstation 15 Installation – User Experience Settings

Step 9- Application Shortcuts preference

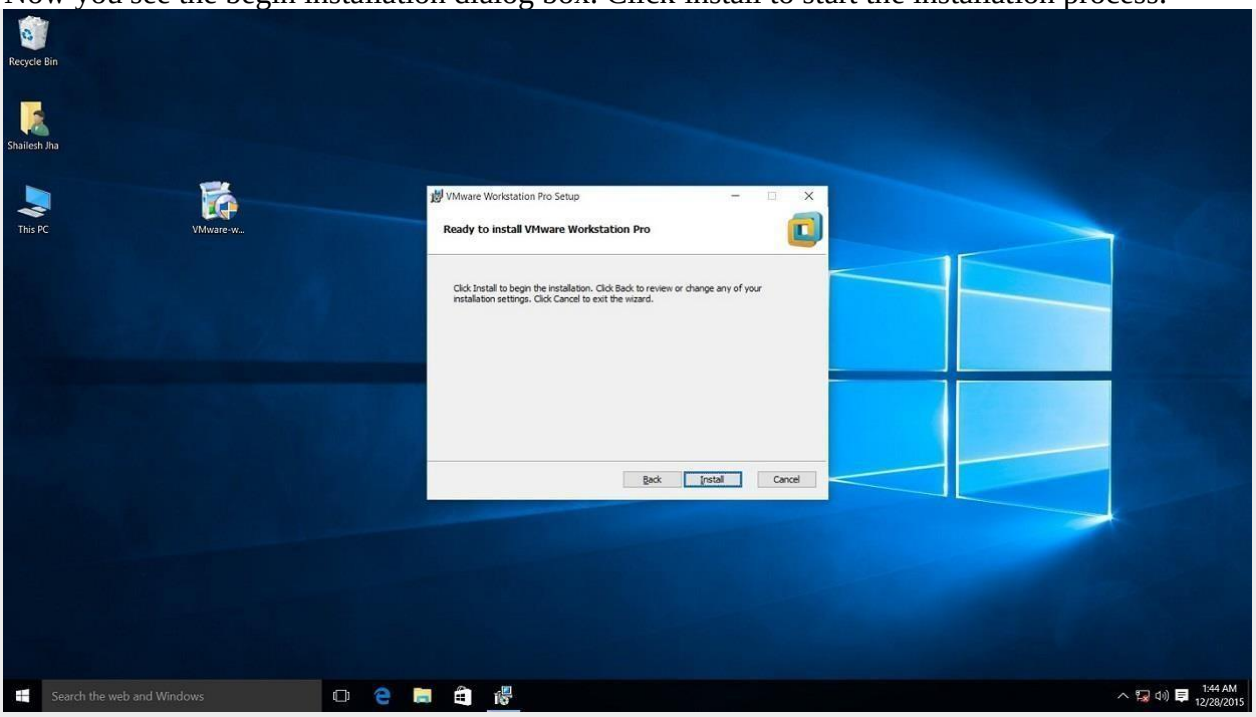
Next step is to select the place you want the shortcut icons to be placed on your system to launch the application. Please select both the options, desktop and start menu and click next.



VMware workstation 15 pro installation shortcut selection checkbox screenshot.

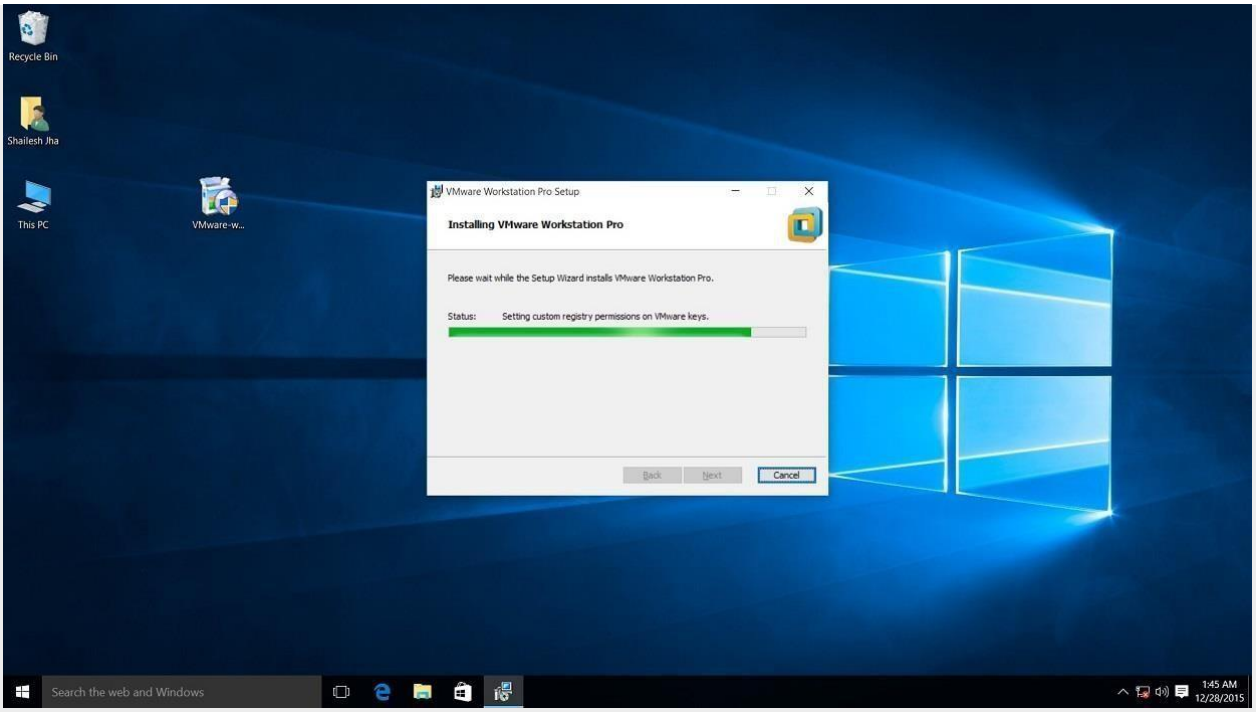
Step 10- Installation begins

Now you see the begin installation dialog box. Click install to start the installation process.



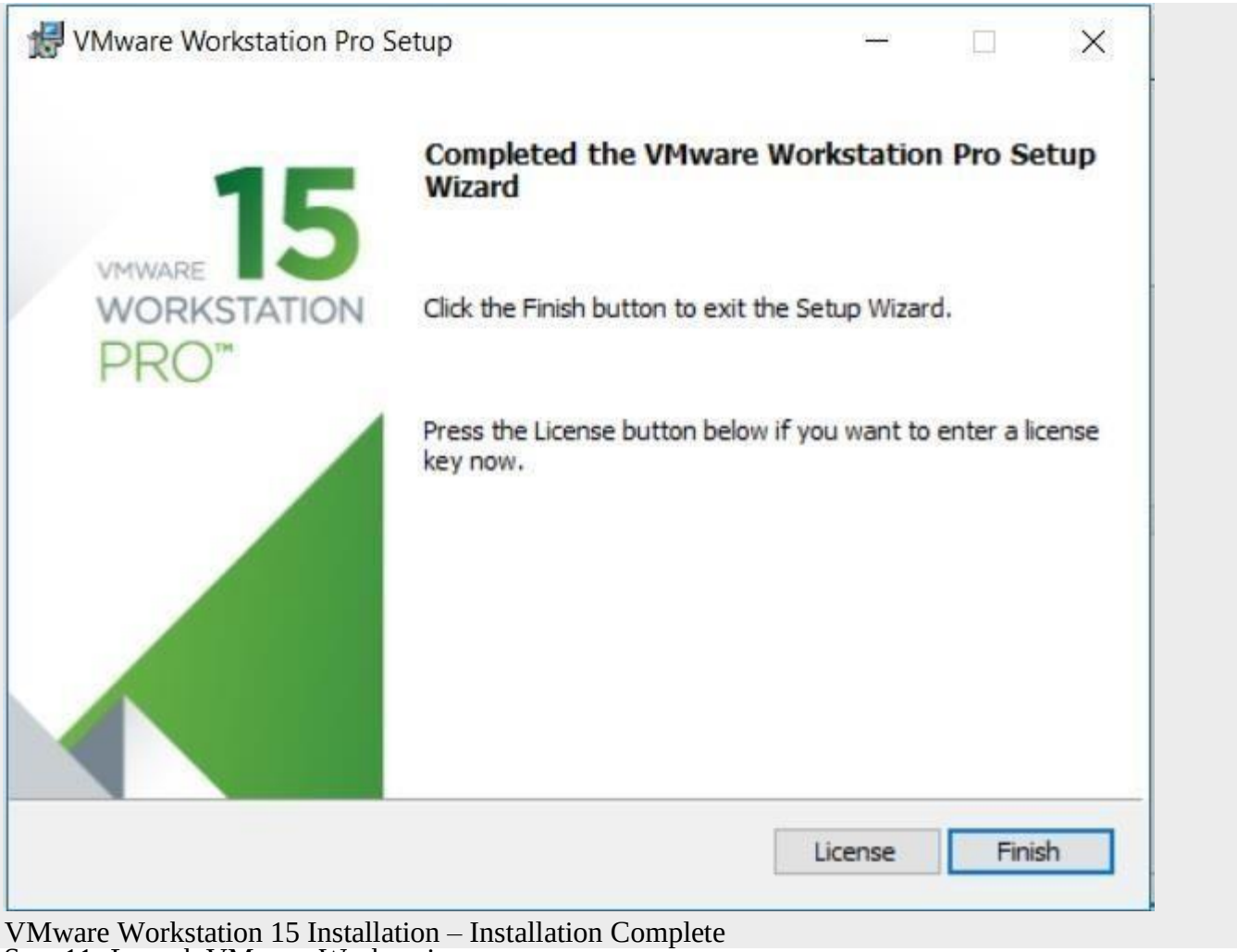
Screenshot for VMware Workstation 15 pro installation begin confirmation dialog box on windows 10.

Below screenshot shows Installation in progress. Wait for this to complete.



Screenshot for VMware Workstation 15 pro installation process.

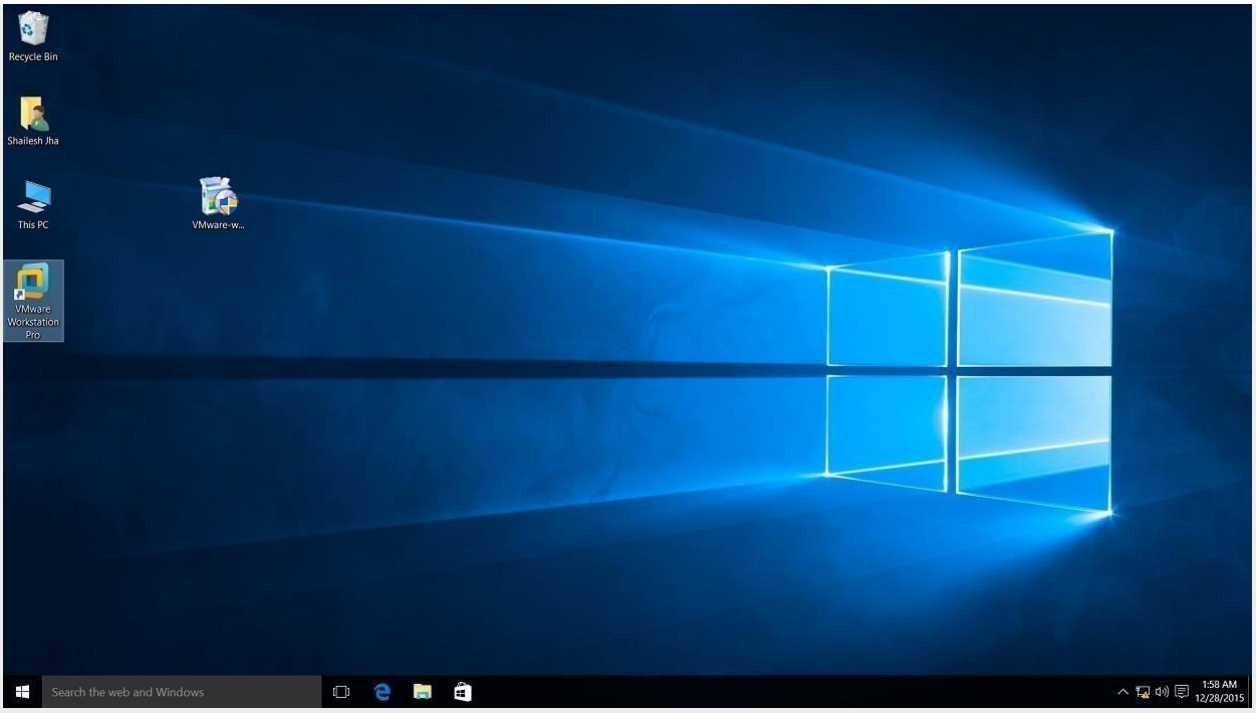
At the end you will see installation complete dialog box. Click finish and you are done with the installation process. You may be asked to restart your computer. Click on Yes to restart.



VMware Workstation 15 Installation – Installation Complete

Step 11- Launch VMware Workstation

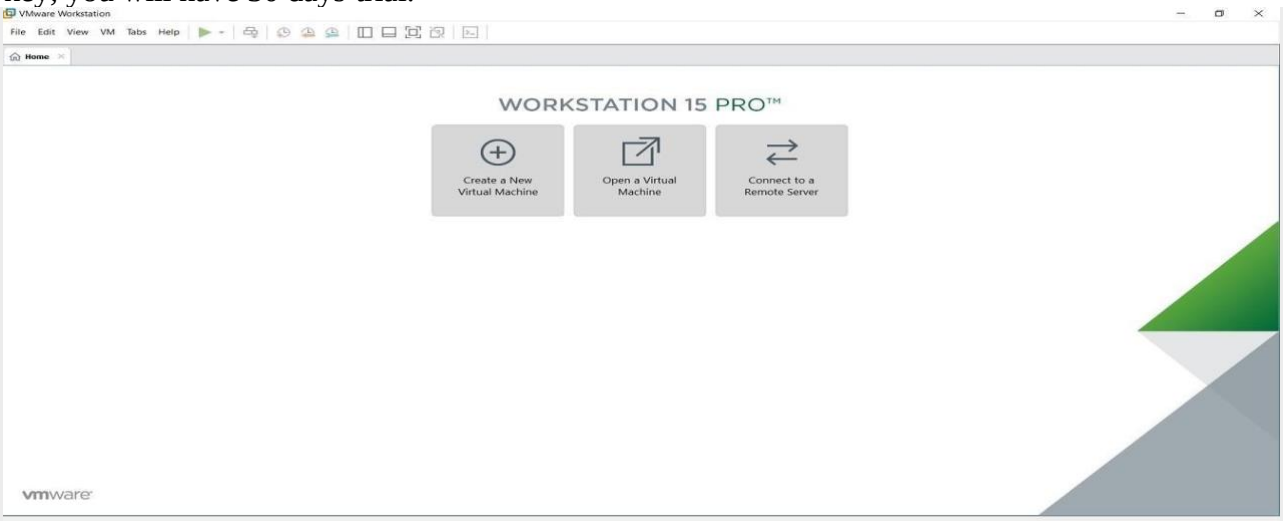
After the installation completes, you should see VMware Workstation icon on the desktop. Double click on it to launch the application.



Screenshot for VMware Workstation 15 Pro icon on windows 10 desktop.

Step 12- Licence Key

If you see the dialog box asking for licence key, click on trial or enter the licence key. Then what you have is the VMware Workstation 15 Pro running on your windows 10 desktop. If don't have the licence key, you will have 30 days trial.



VMware Workstation 15 Pro home screen

Step 13- At some point if you decide to buy

At some point of time if you decide to buy the Licence key, you can enter the Licence key by going to **Help->Enter a Licence Key**

You can enter the 25 character licence key in the dialog box shown below and click OK. Now you have the licence version of the software.

Setting up C Programming Environment

Before you start..

- Please note that this is "NOT" a required part of the course and is not a homework.

- This manual is written for someone who has never programmed / never used Linux before.
  If you have background knowledge, you can ignore this.

- If you have any questions, please email
  Yoonji Shin ys2476@columbia.edu

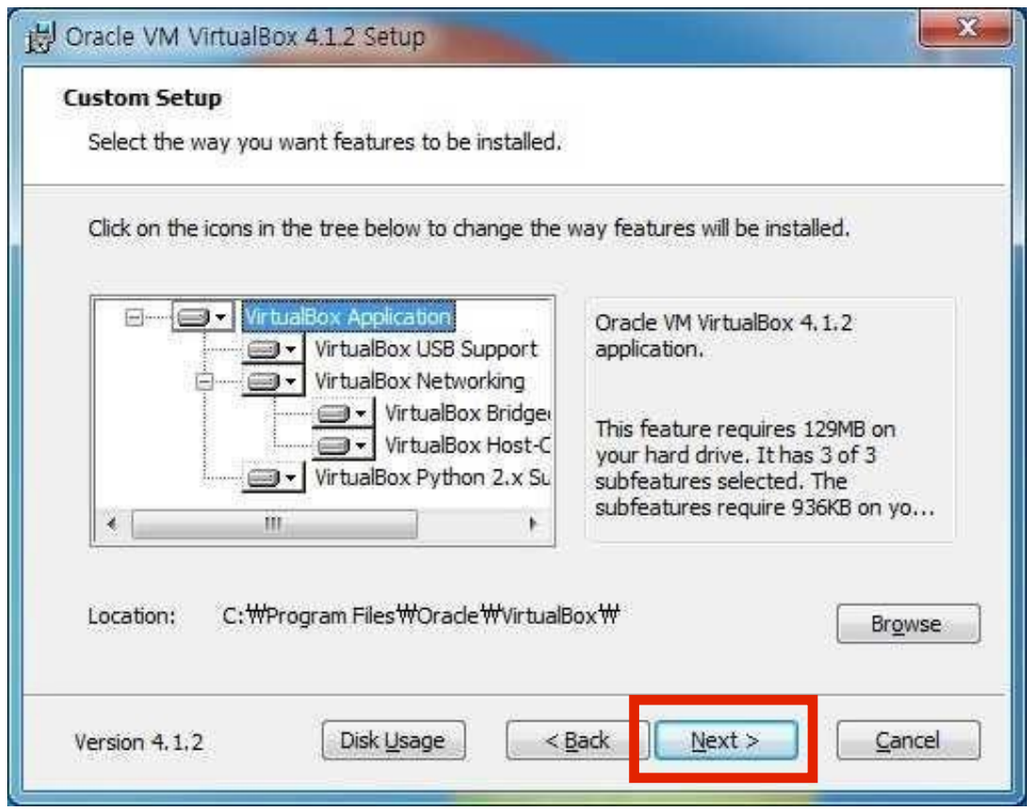
- Don't be scared, programming is fun! Enjoy :)

Install VirtualBox

1. Visit http://www.virtualbox.org/wiki/downloads
2. Download VirtualBox platform packages for your OS
3. Open the Installation Package by double clicking

MAC

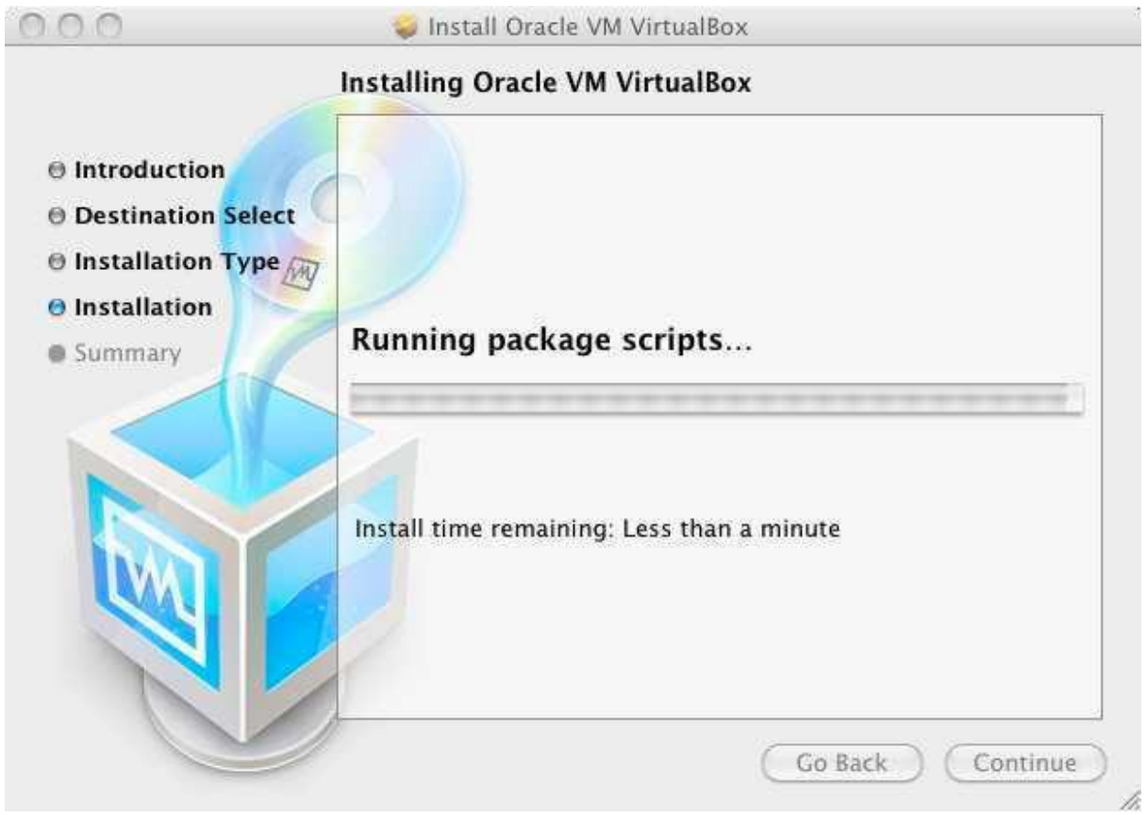
PC



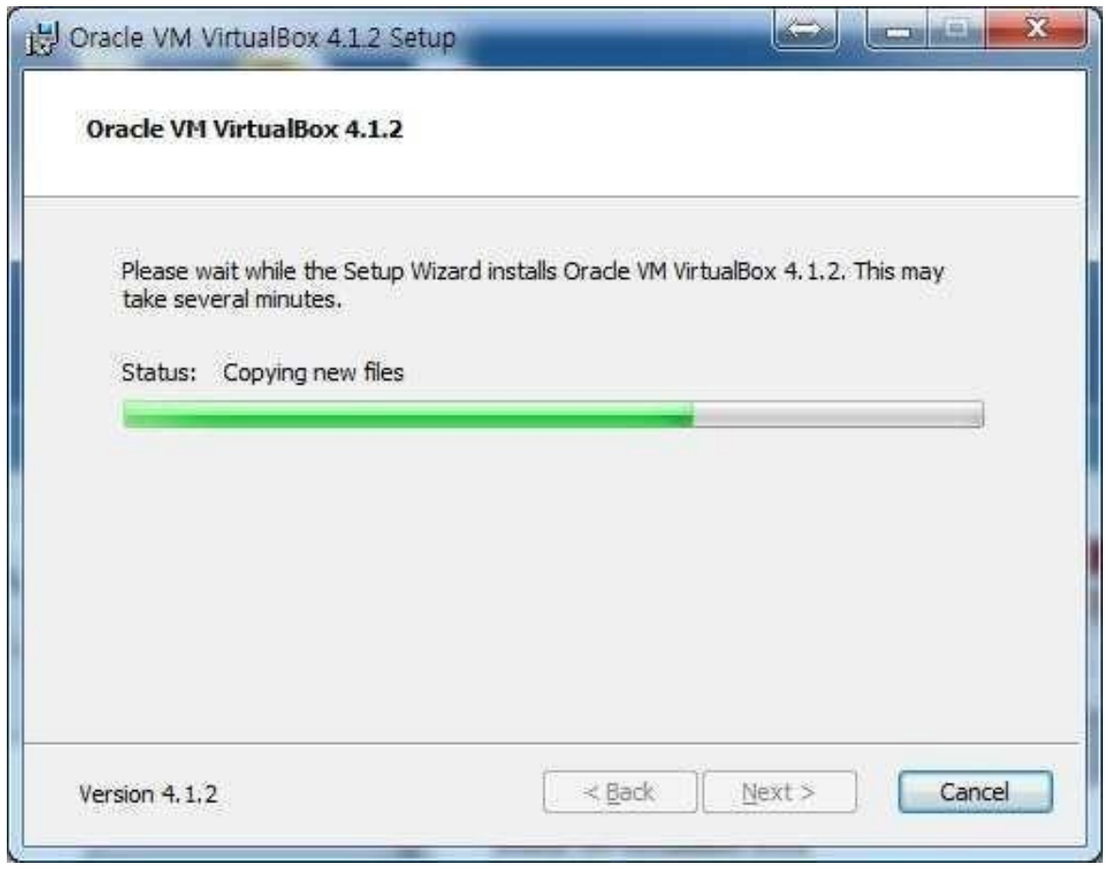
Install VirtualBox

4. Click continue and finish installing VirtualBox

MAC



PC

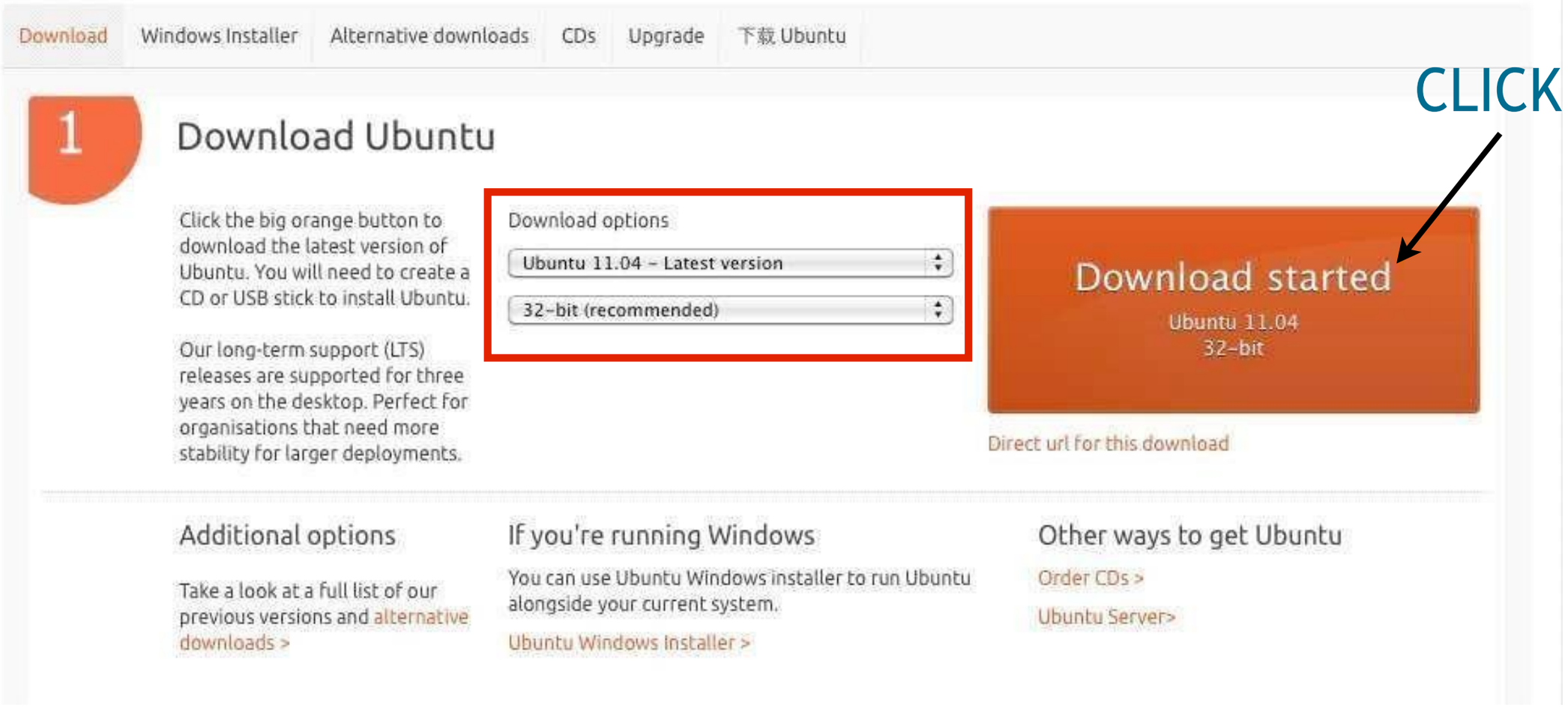


5. When finished installation, close the window.

Download Linux

I. Visit the page
http://www.ubuntu.com/download/ubuntu/download

2. Choose the Latest version of Ubuntu and 32-bit and click “Start Download”

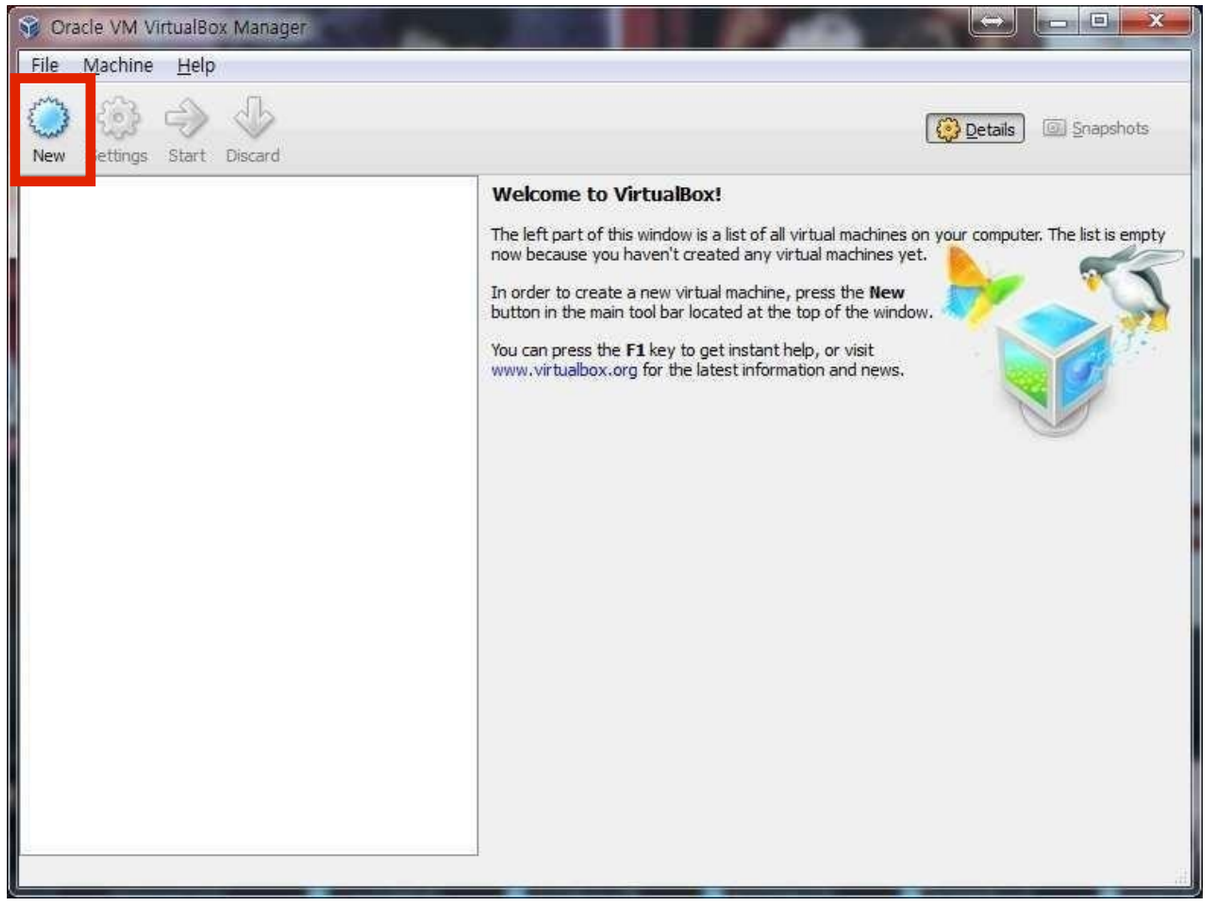


Install Linux using Virtual Box

1. Run VirtualBox by double-clicking the icon
2. Click “New” button on the top left corner

MAC

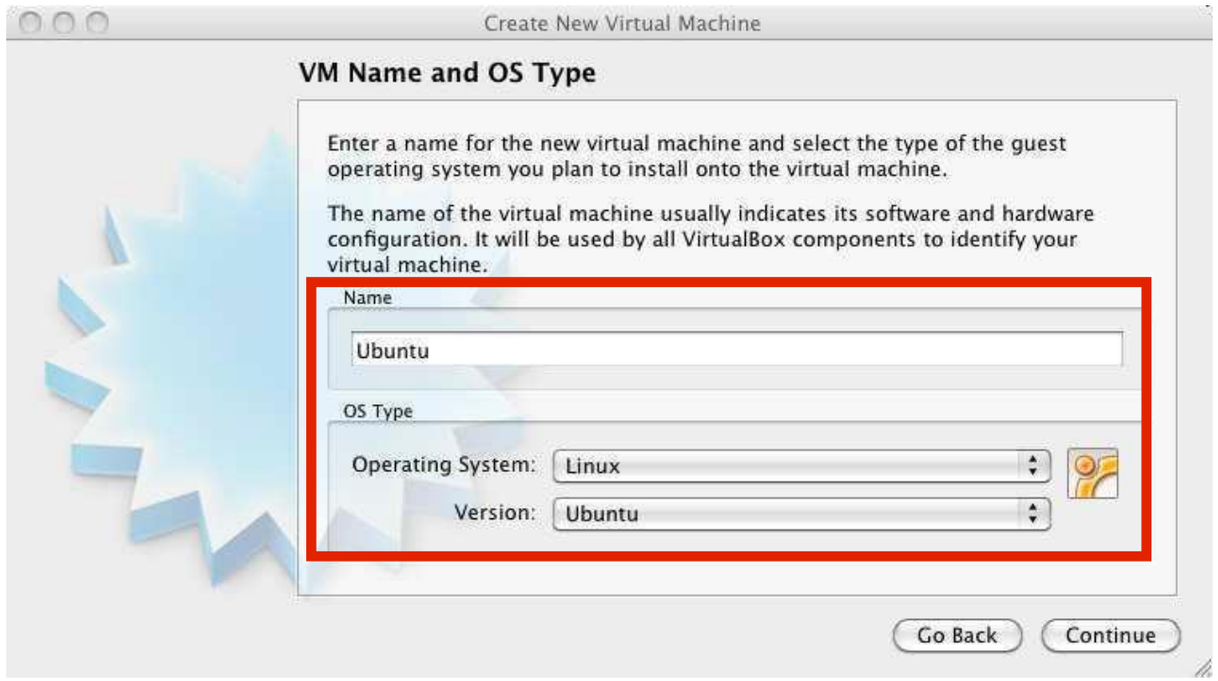
PC



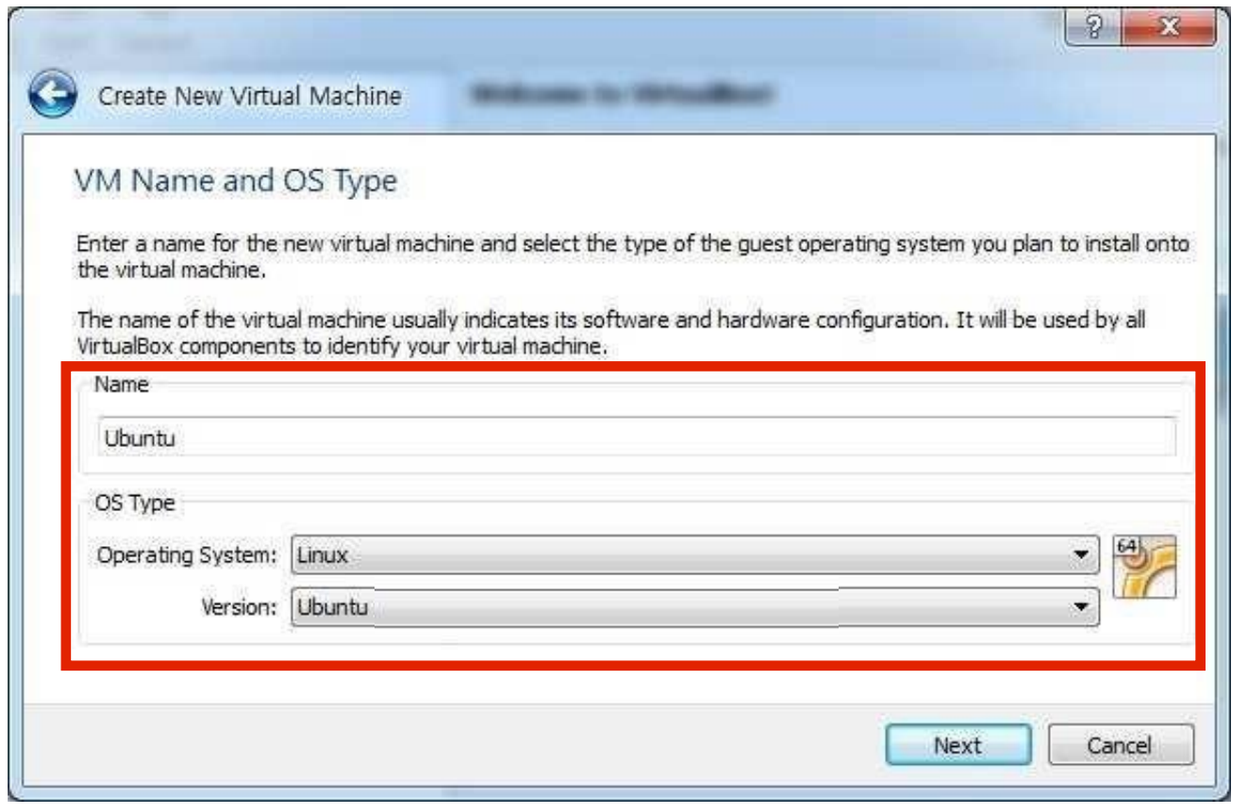
Install Linux using Virtual Box

3. Click “Continue” on the pop-up window
4. Type VM name, select “Linux” for the OS and choose “Ubuntu” for the version.

MAC



PC

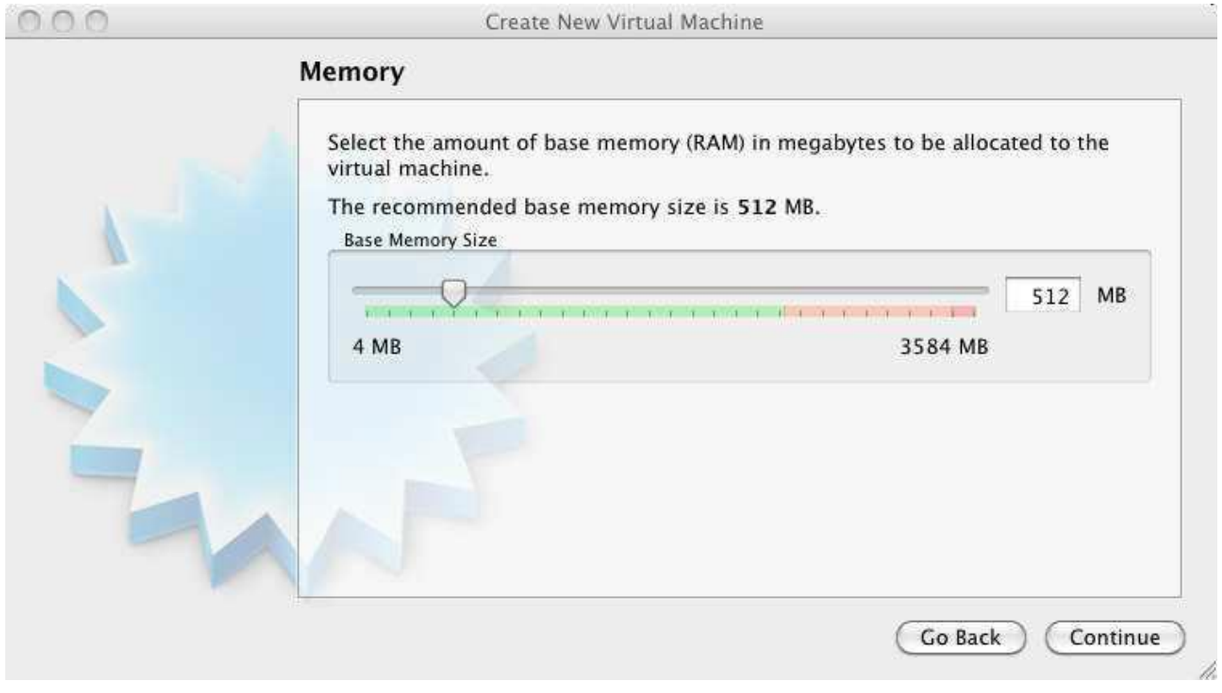


Install Linux using Virtual Box

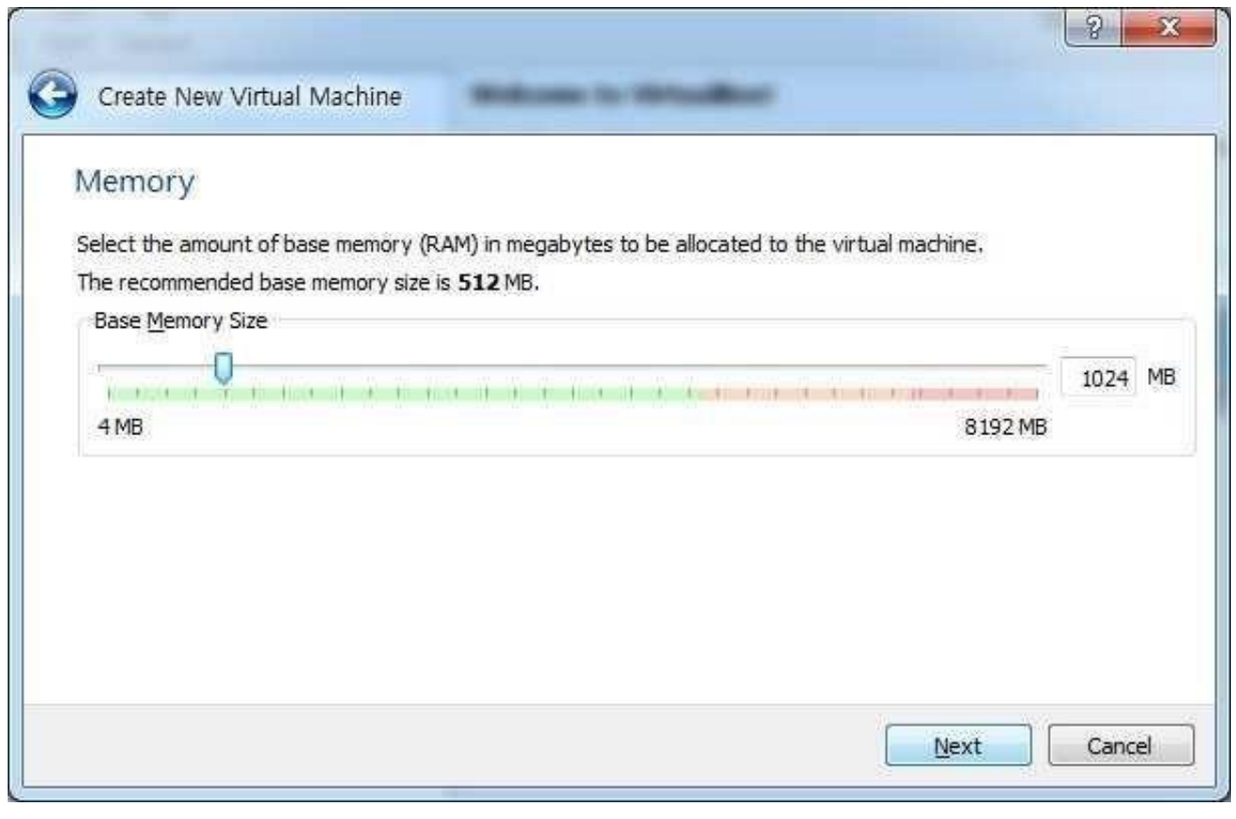
5. Choose the amount of memory to allocate (I suggest choosing between 512 MB to 1024 MB)

6. Click Continue or Next

MAC



PC

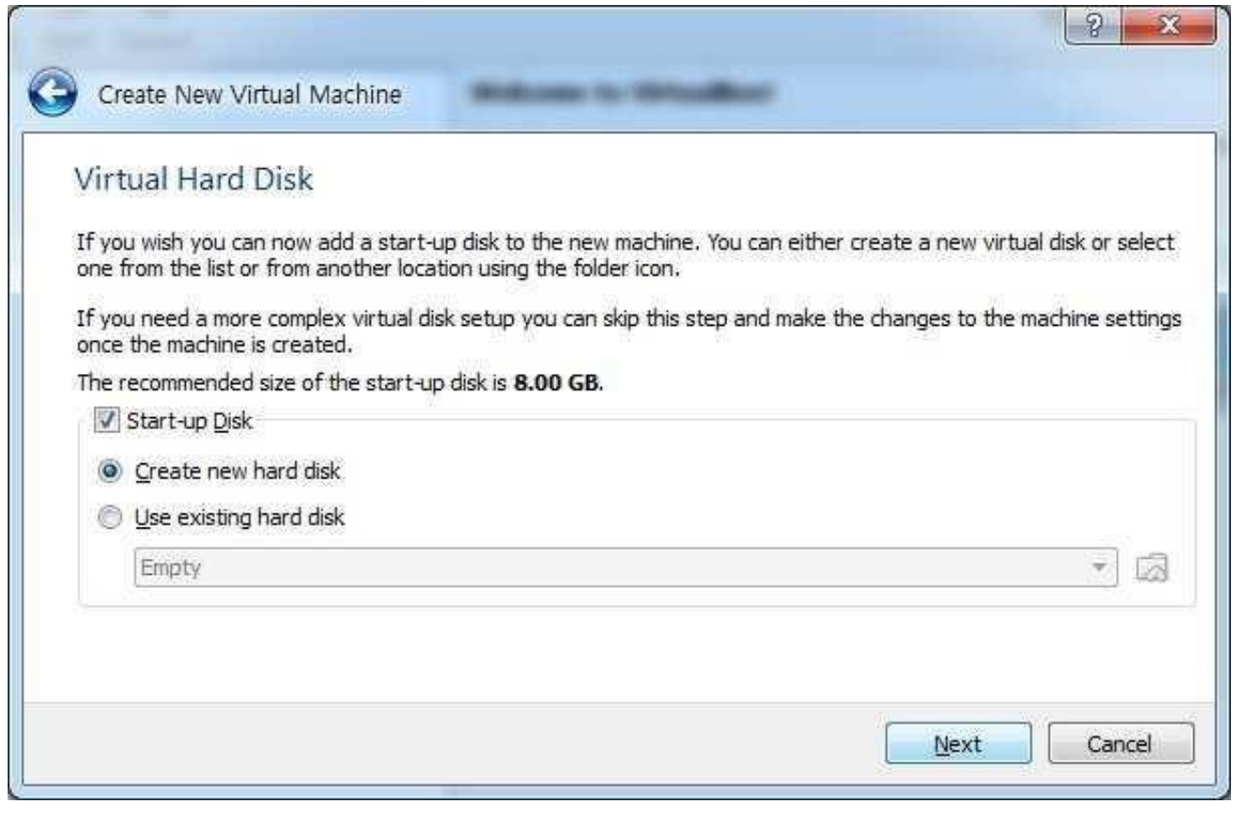


Install Linux using Virtual Box

7. Choose create a new virtual hard disk
8. Click Continue or Next

MAC

PC



Install Linux using Virtual Box

9. Choose VDI (VirtualBox Disk Image)

I0. Click Continue or Next

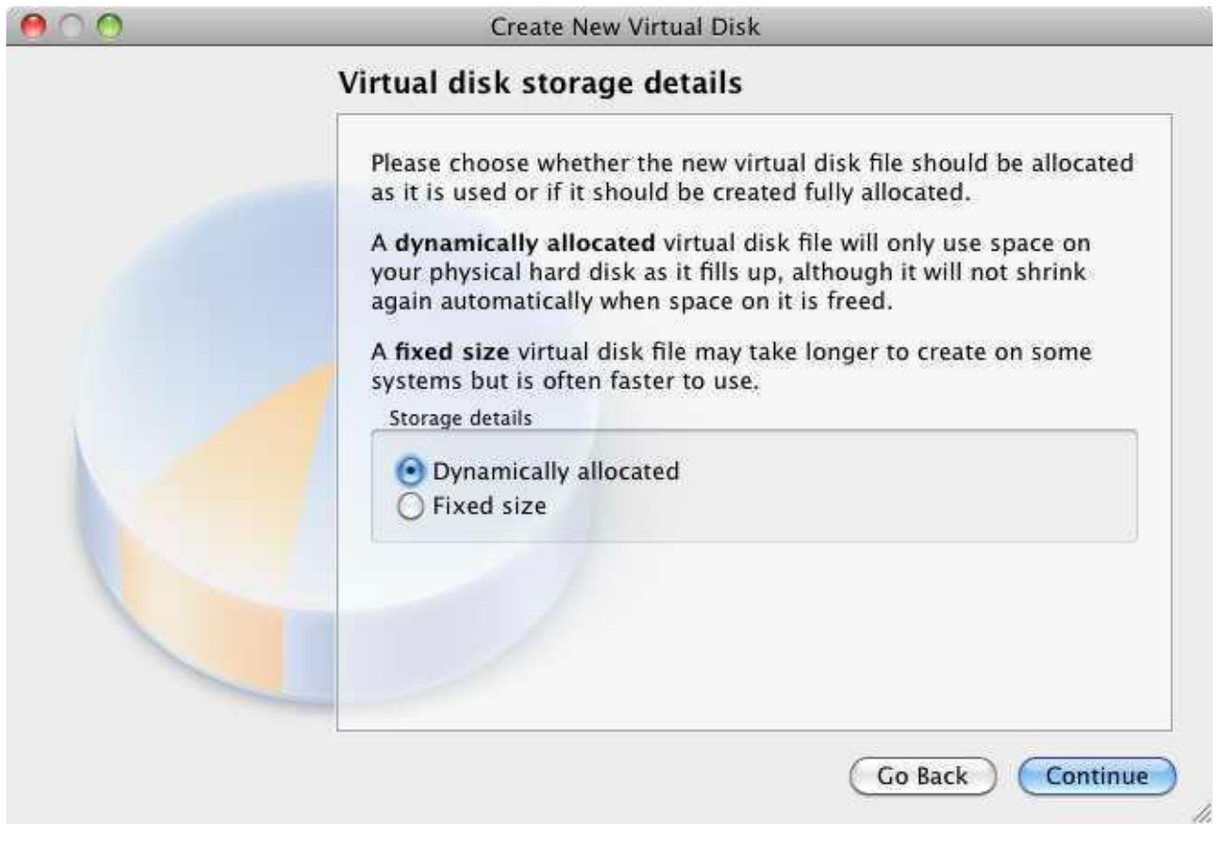
MAC

PC

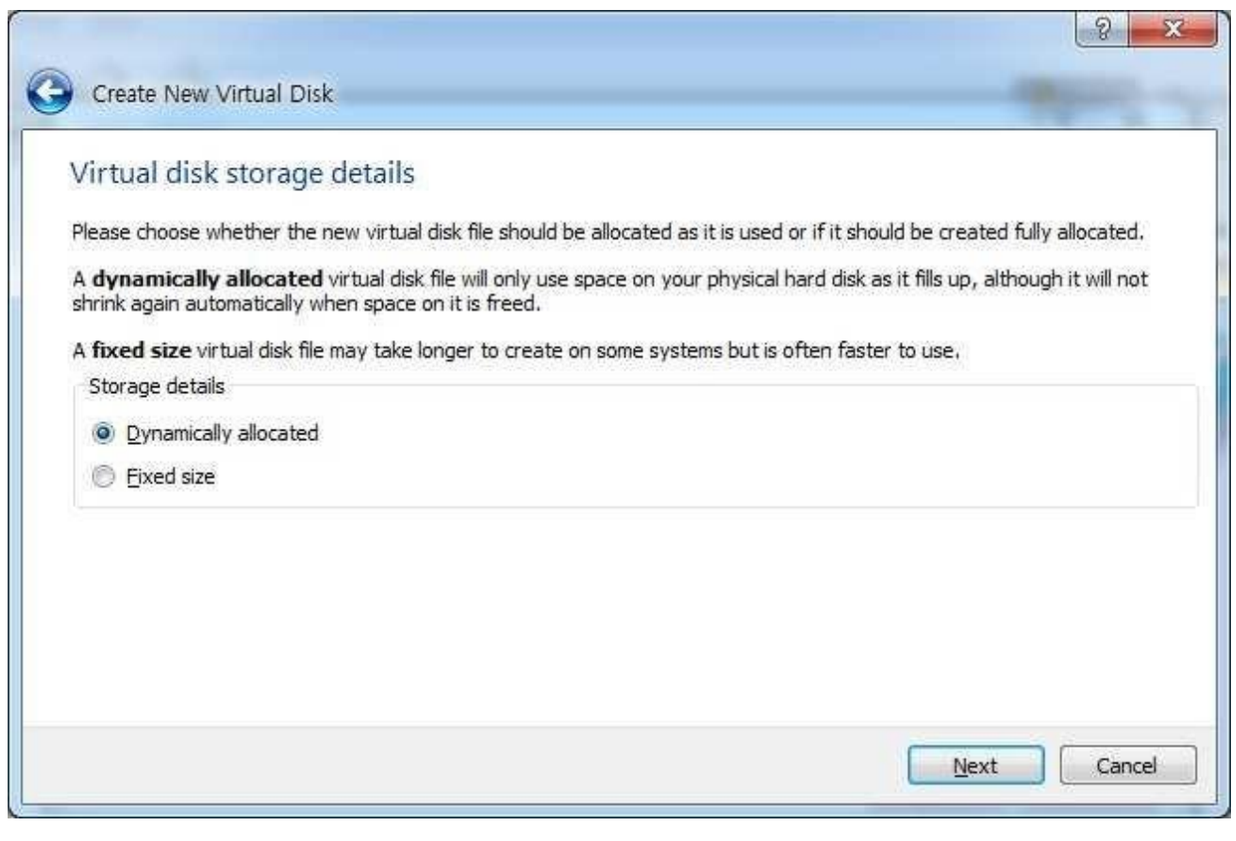
Install Linux using Virtual Box

II. Choose “DynamicallyAllocated” click continue.
This way, the size of yourVirtual Hard Disk will grow as you use.

MAC



PC

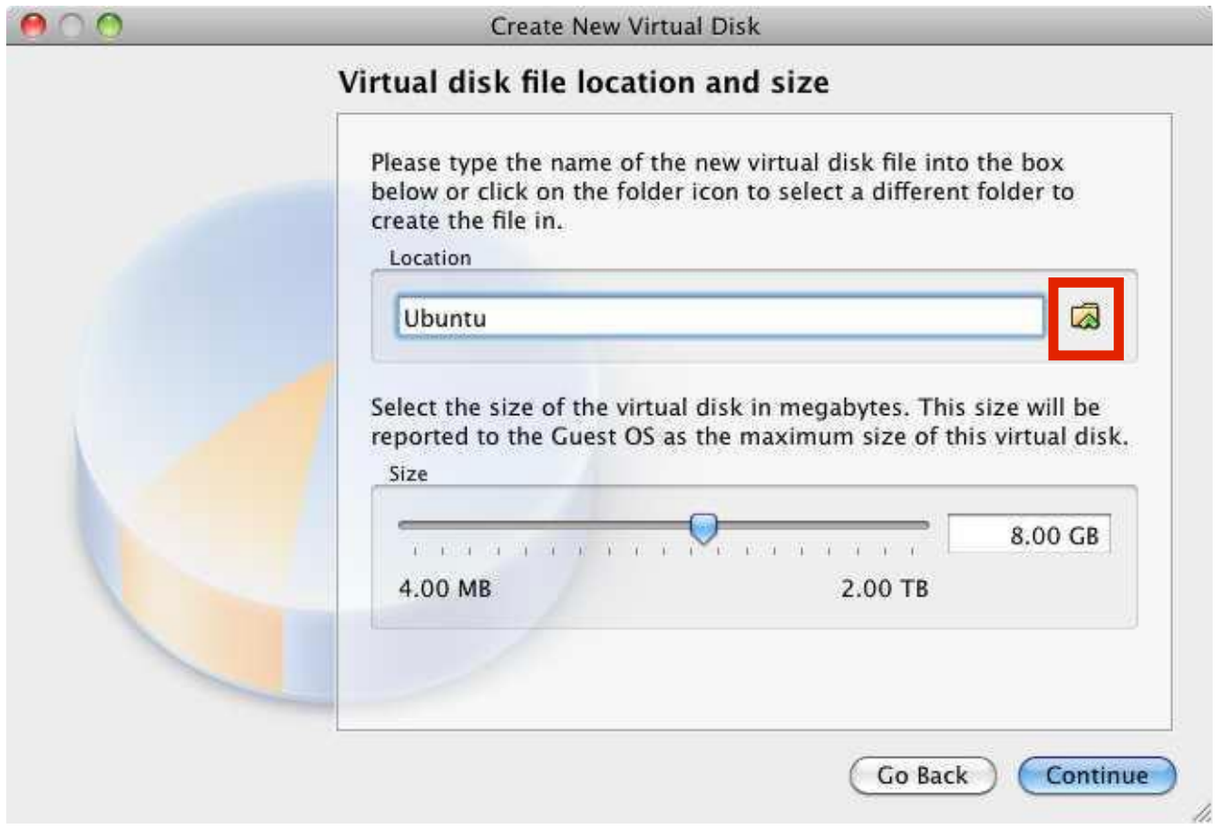


Install Linux using Virtual Box

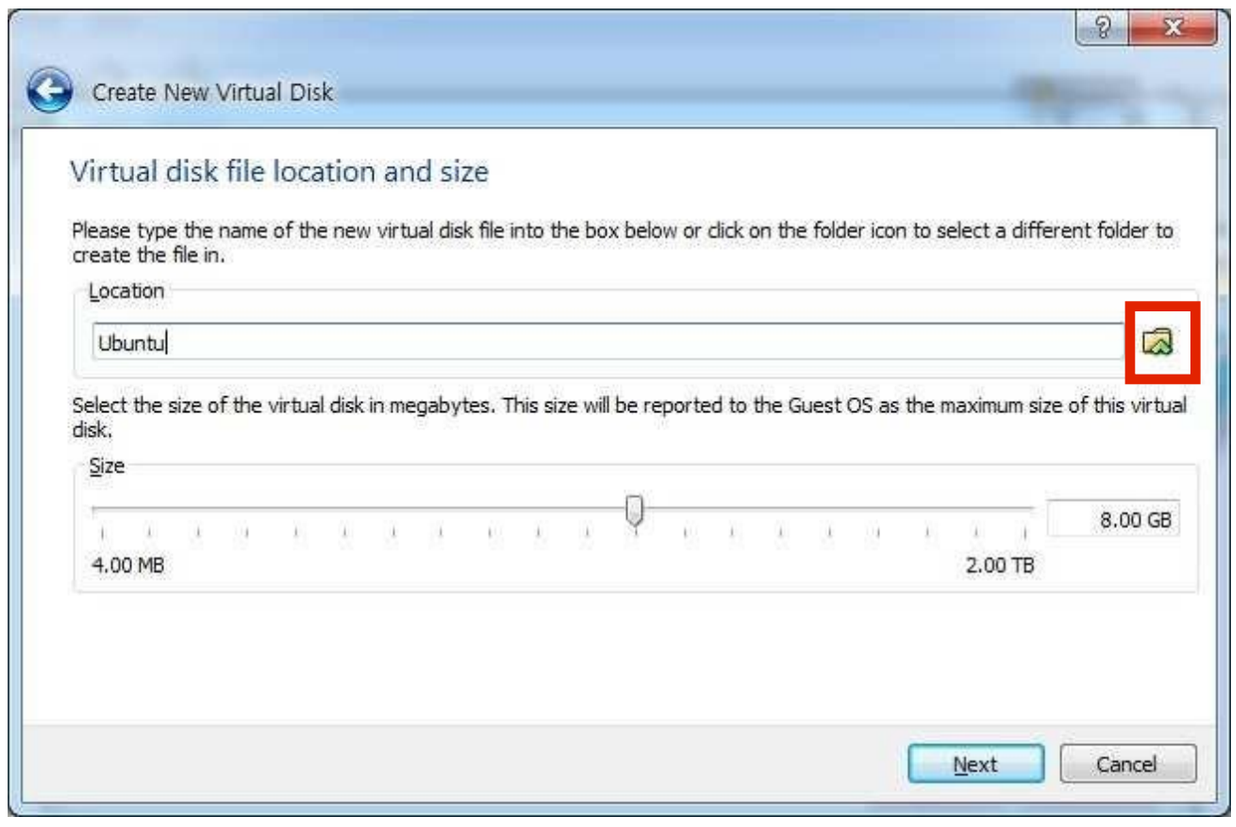
I2. Click the folder icon and choose the ubuntu iso file you downloaded.

I3. Select the size of the Virtual Disk (I recommend choosing 8 GB) and click continue

MAC



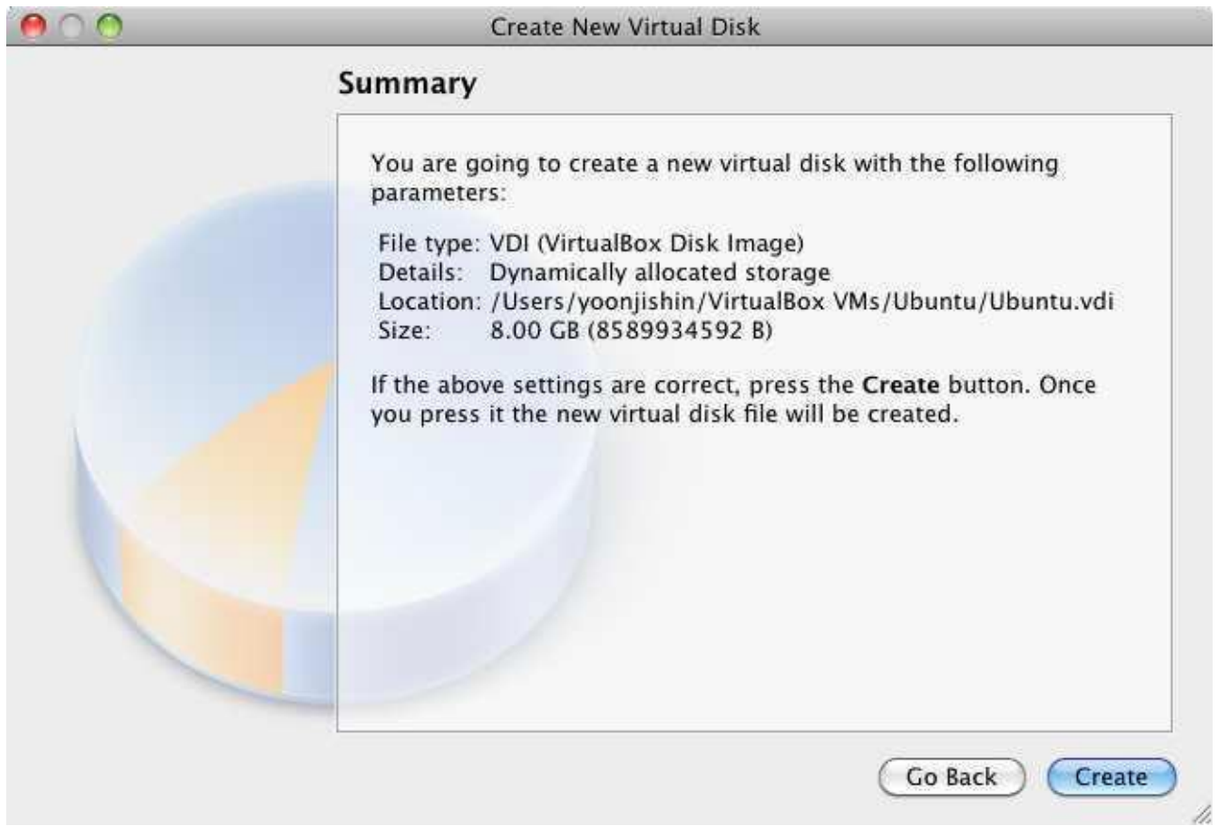
PC



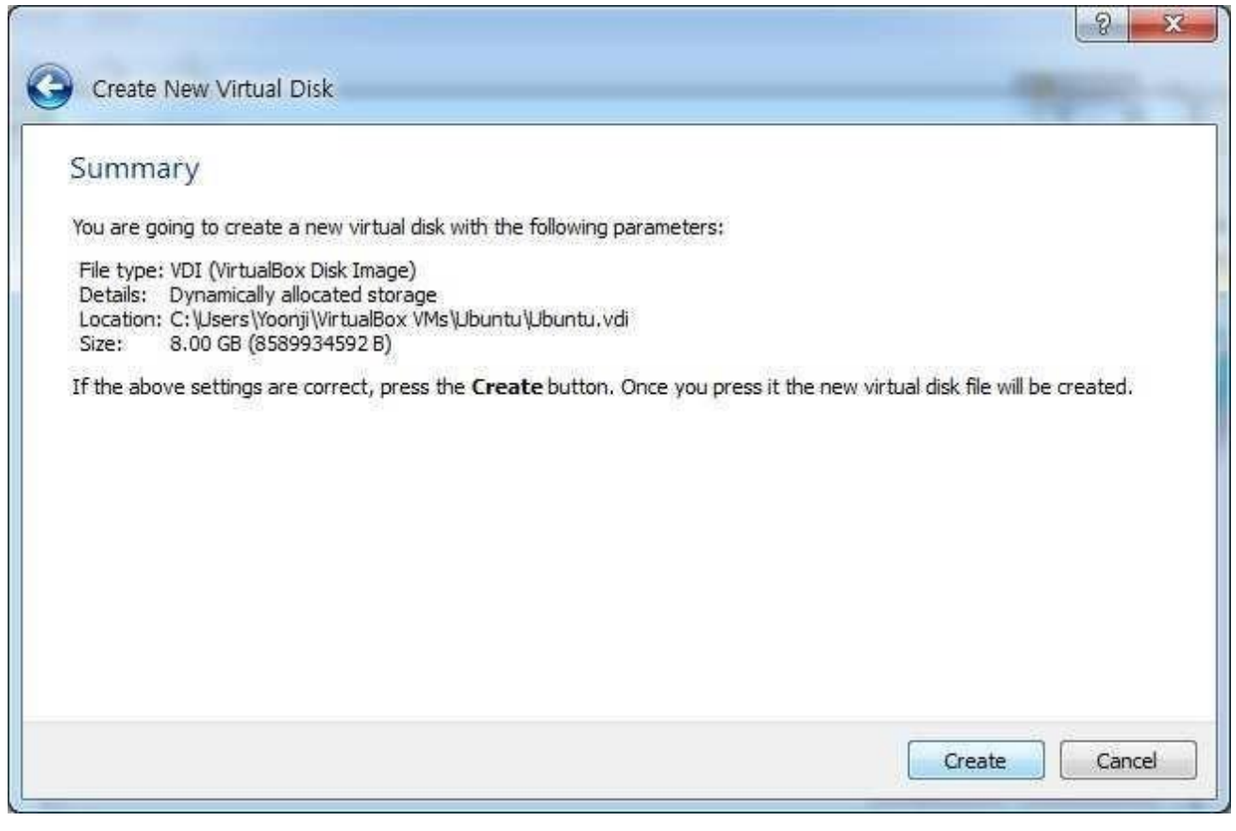
Install Linux using Virtual Box

14. Click Create

MAC



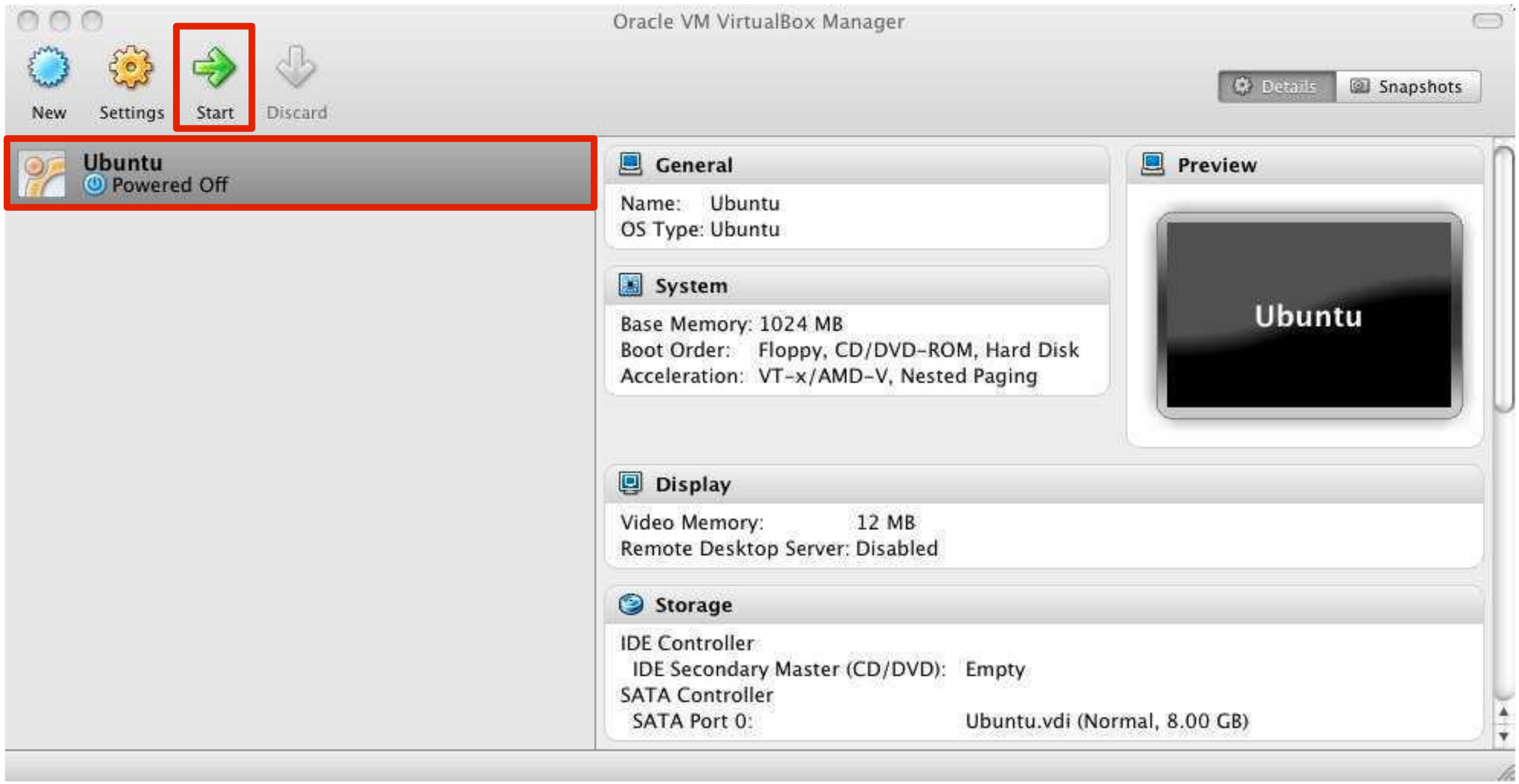
PC



Running Linux

I. Choose Ubuntu from left column and click Start

MAC & PC



Running Linux

2. Click continue on pop-up window

MAC

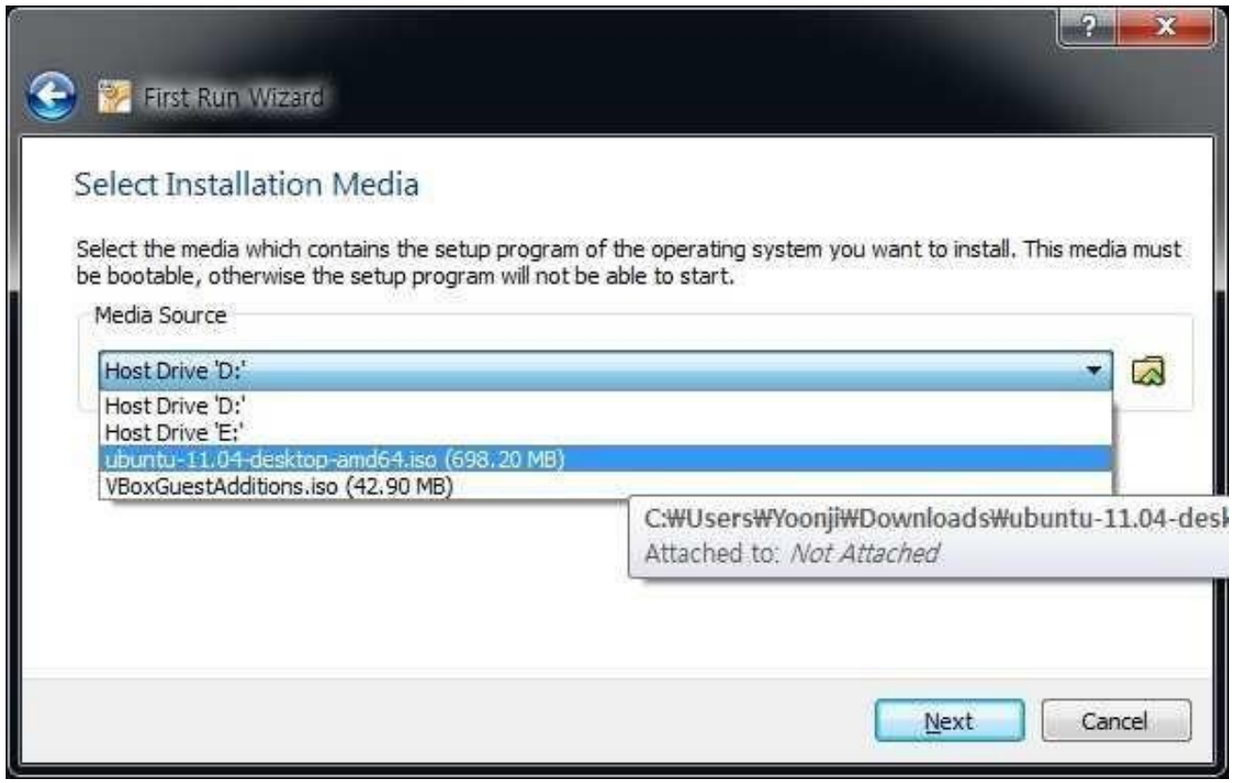
PC

Running Linux

3. Click the folder icon and choose the ubuntu iso file you downloaded and click continue and start

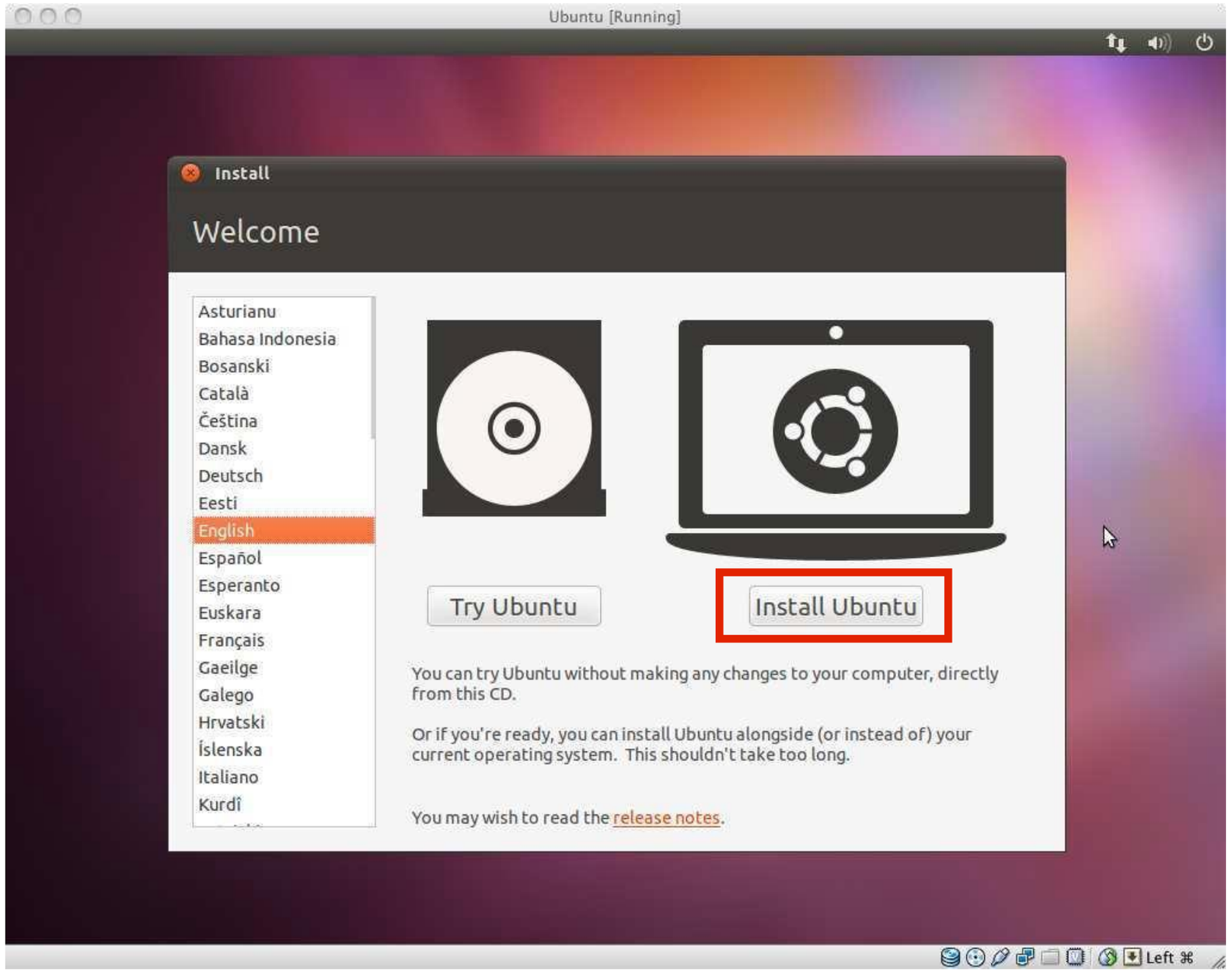
MAC

PC



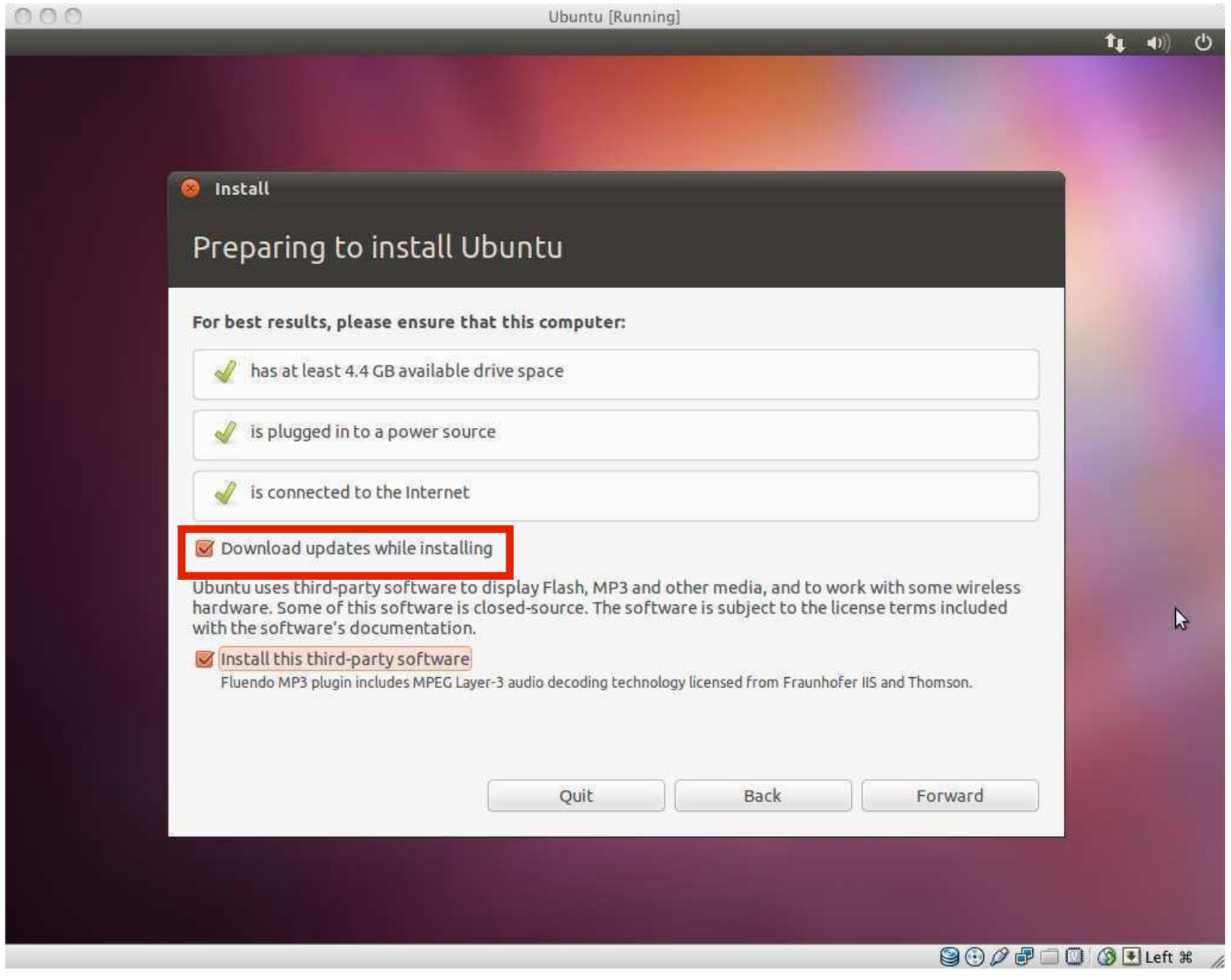
Running Linux

4. Click Install Ubuntu



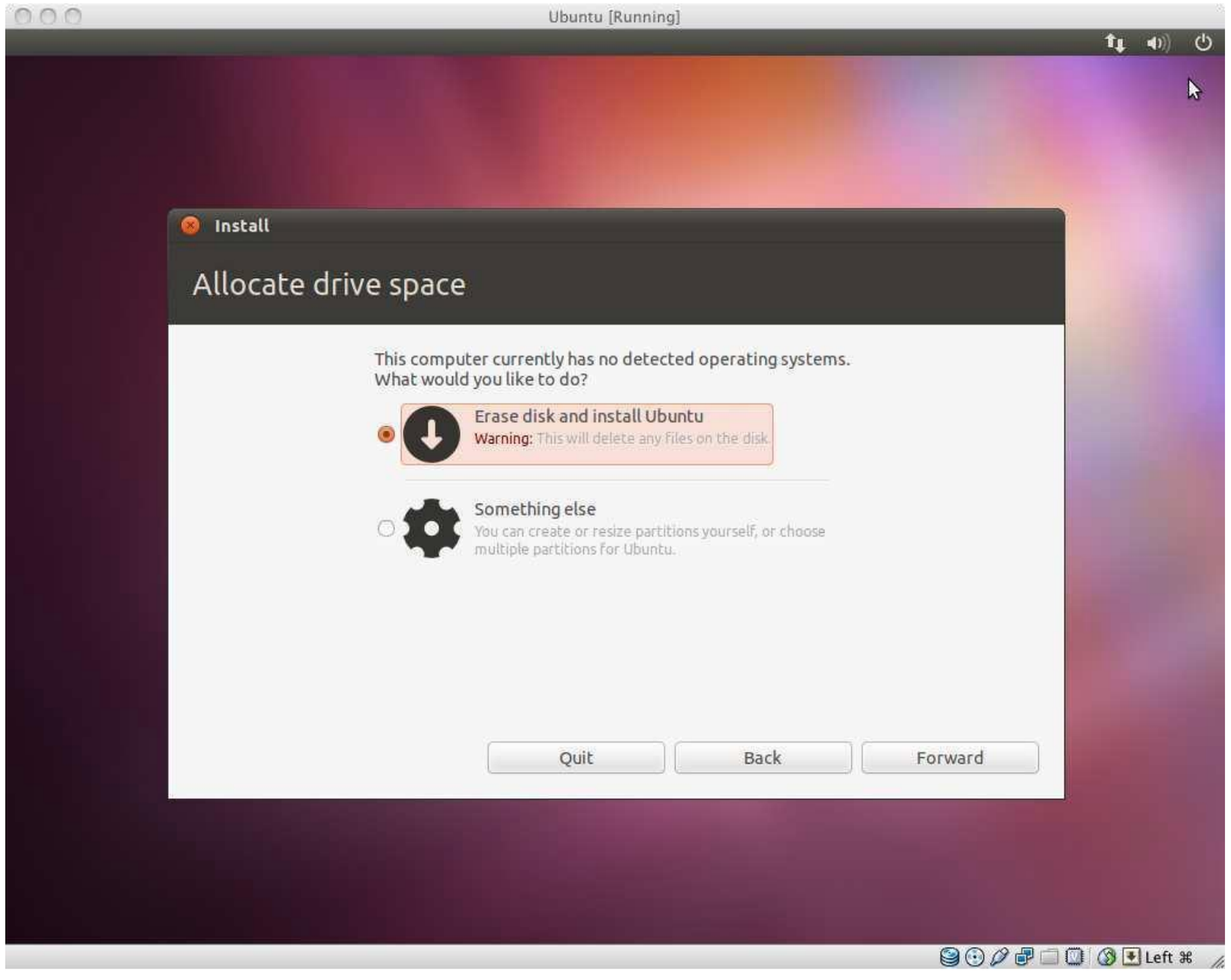
Running Linux

4. Check “Download updates” and click Forward



Running Linux

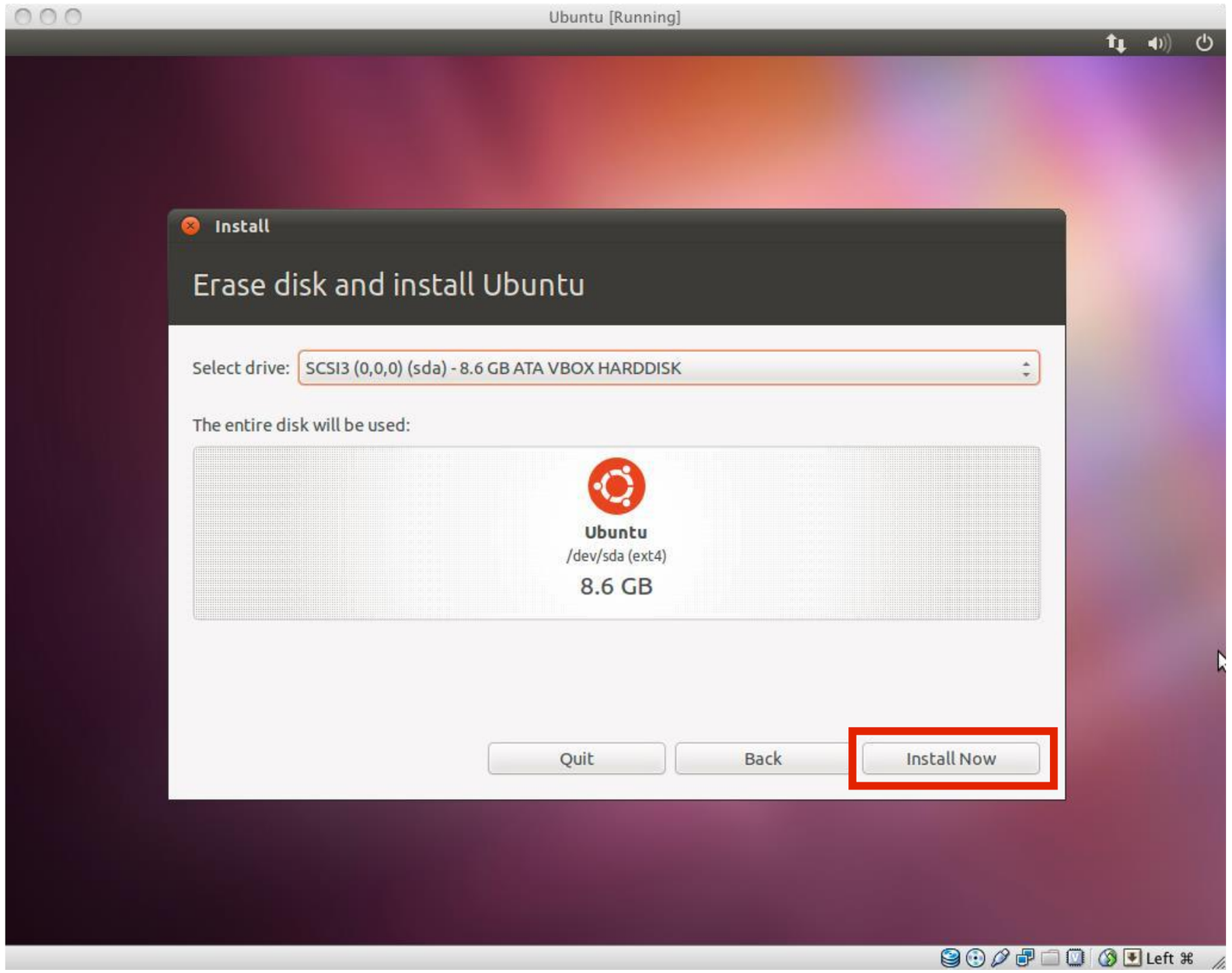
5. Choose “Erase disk and install Ubuntu” and click Forward (Don’t worry, it won’t wipe your computer)



Running Linux

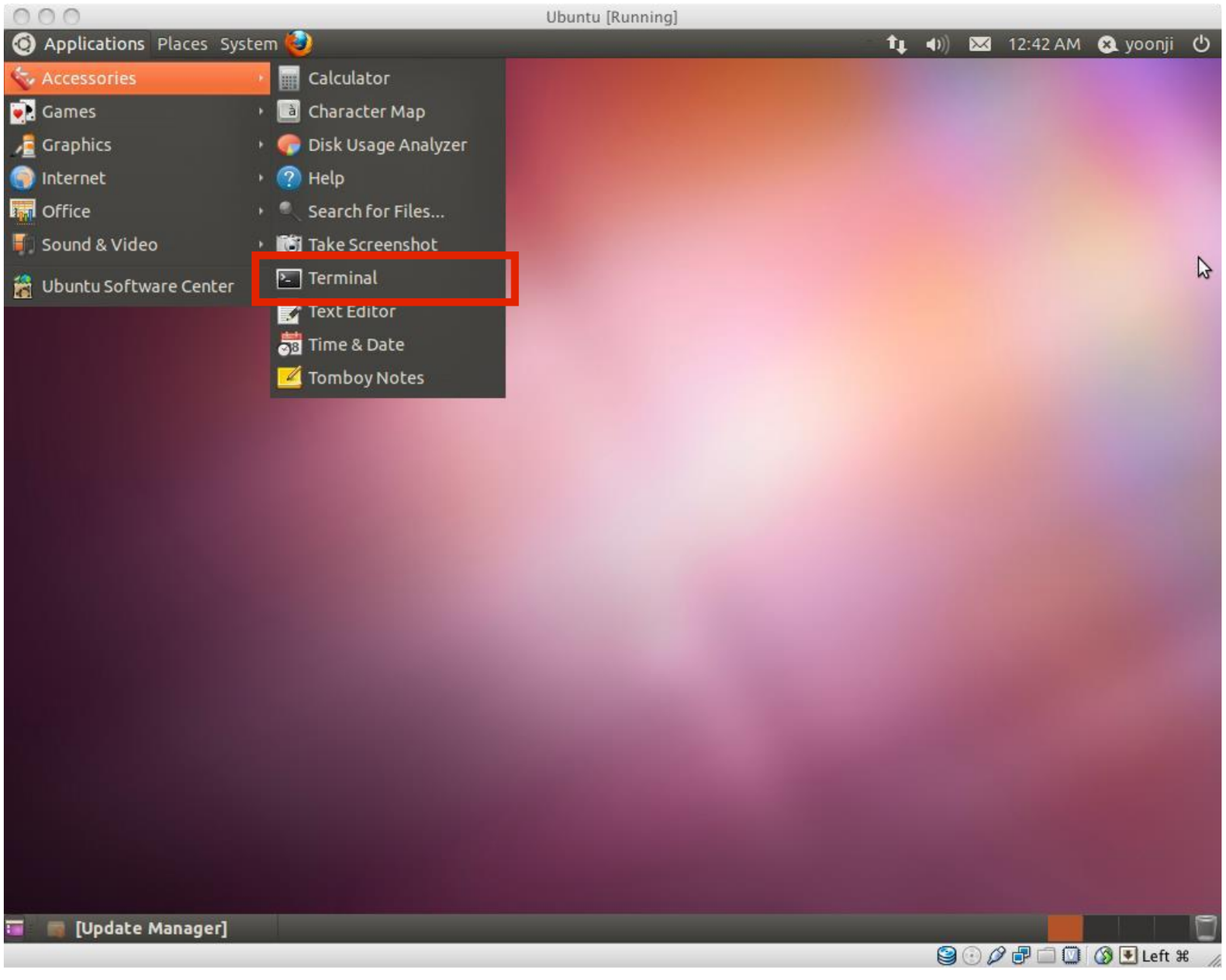
6. Click “Install Now” and wait. Maybe grab asnack.

7. When finished, click Restart and press Enter.



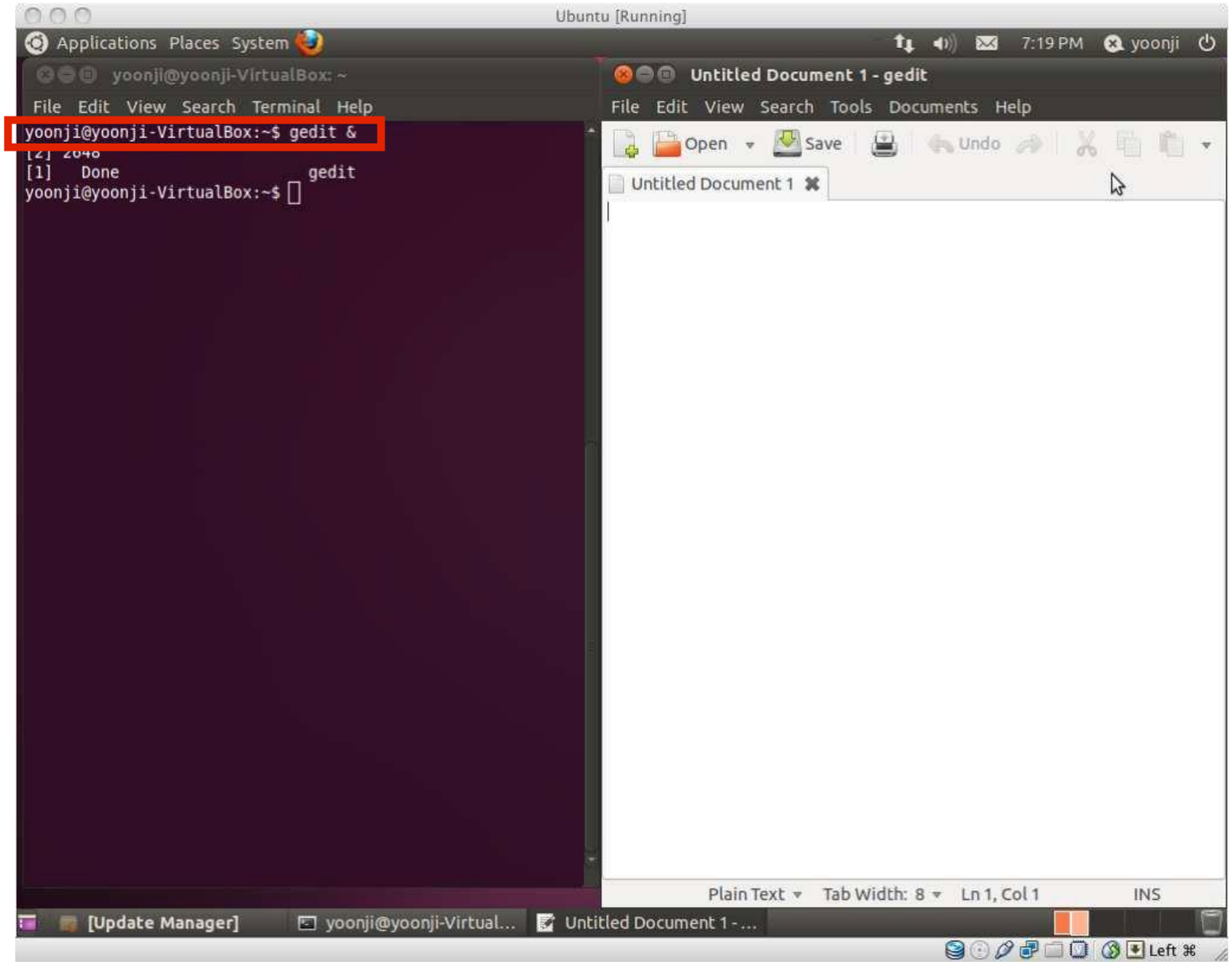
C Programming on Linux

I. OpenTerminal (Applications-Accessories-Terminal)



C Programming on Linux

2. Open gedit by typing “gedit &” on terminal
   (You can also use any other Text Editor application)

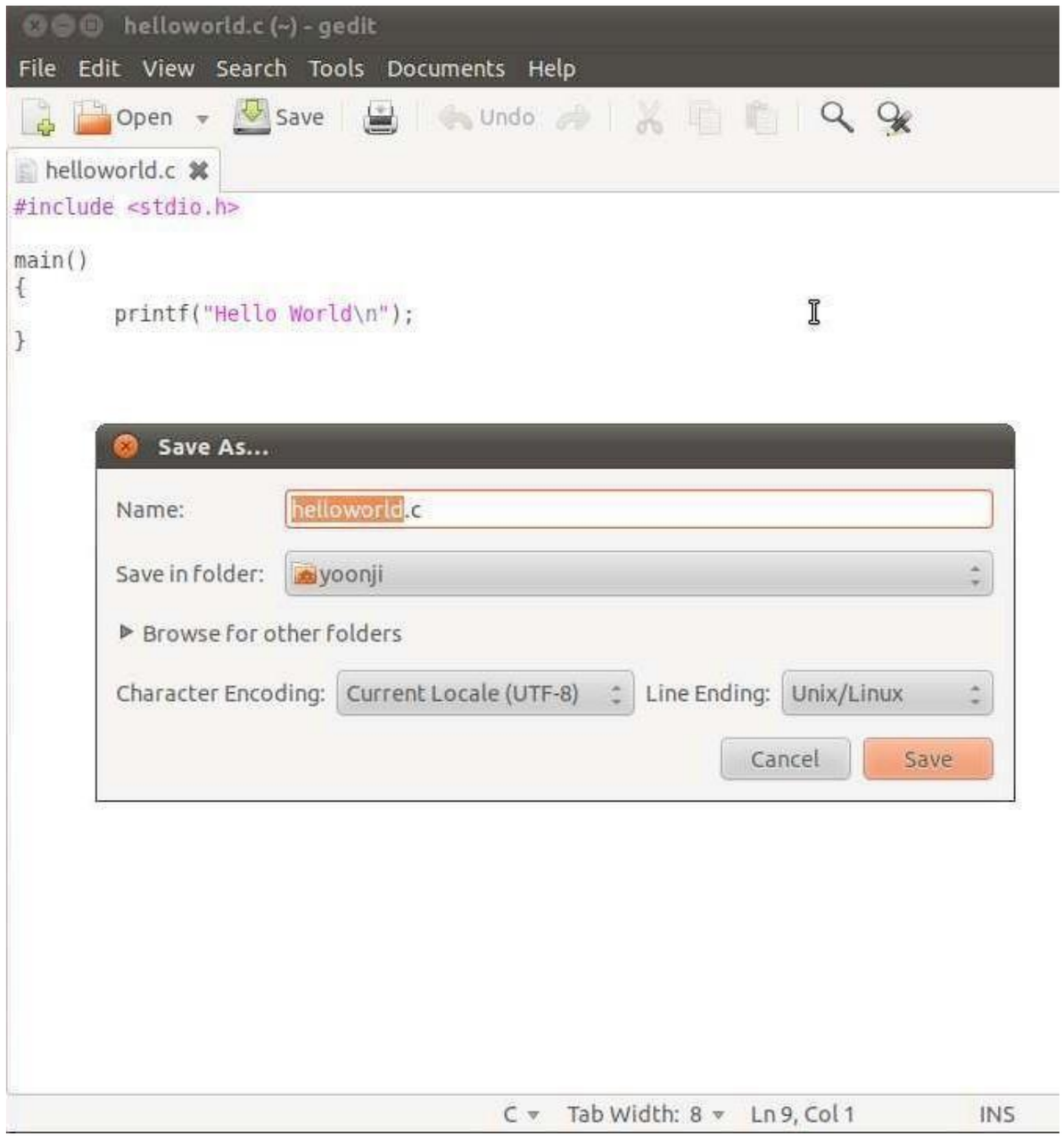


C Programming on Linux

3. Type the following on gedit (or any other text editor)

```
#include<stdio.h>

main()
{
    printf("Hello World\n");
}
```

4. Save this file as “helloworld.c”

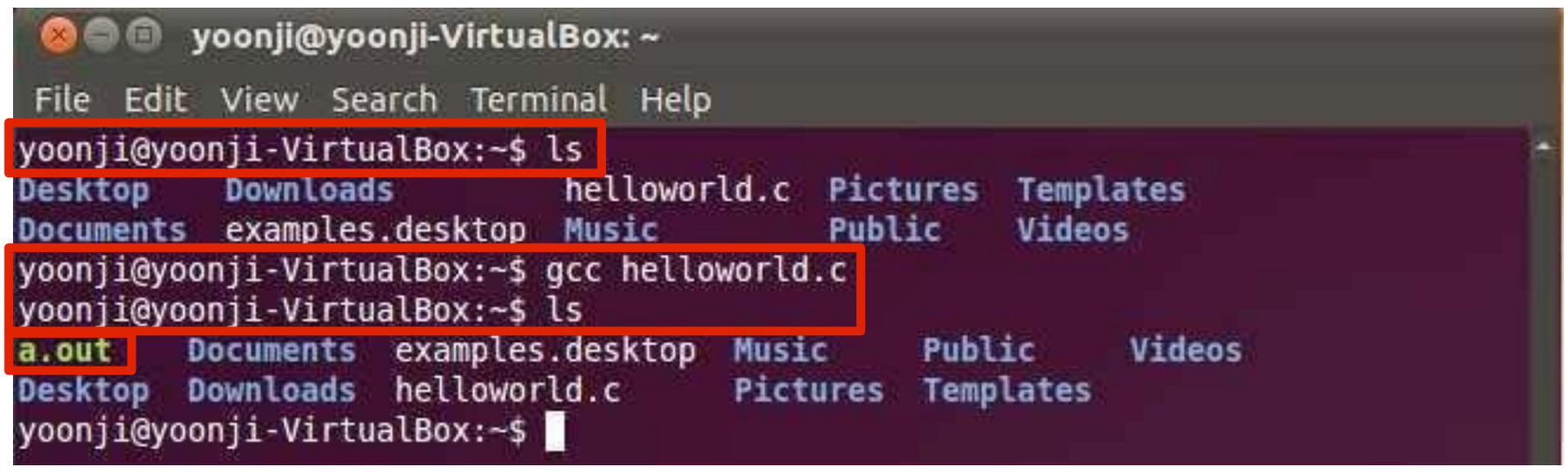


C Programming on Linux

5. Type “ls” on Terminal to see all files under current folder

6. Confirm that “helloworld.c” is in the current directory.
   If not, type cd DIRECTORY_PATH to go to the
   directory that has “helloworld.c”

7. Type “gcc helloworld.c” to compile, and type “ls” to
   confirm that a new executable file “a.out” is created

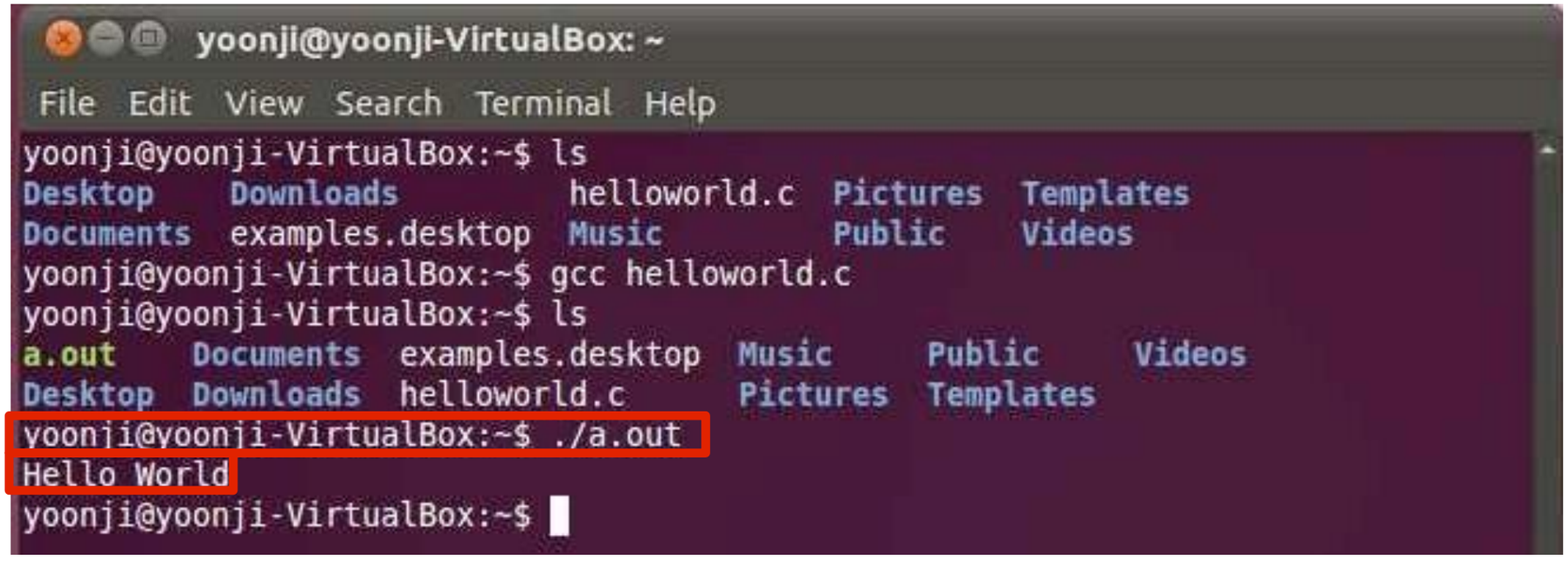


C Programming on Linux

8. Type "./a.out" on Terminal to run the program

9. If you see "HelloWorld" on the next line,
   you just successfully ran your first C program!

10. Try other codes from "A Shotgun Introduction to C" on professor Edwards's webpage. You can also find many C programing guides online. (just google it!) Enjoy :)



Installing and Running the Google App Engine On Windows

This document describes the installation of the Google App Engine Software Development Kit (SDK) on a Microsoft Windows and running a simple "hello world" application.

The App Engine SDK allows you to run Google App Engine Applications on your local computer. It simulates the run---time environment of the Google App Engine infrastructure.

Pre-•Requisites: Python 2.5.4

If you don't already have Python 2.5.4 installed in your computer, download and Install Python 2.5.4 from:

http://www.python.org/download/releases/2.5.4/

Download and Install

You can download the Google App Engine SDK by going to:

http://code.google.com/appengine/downloads.html

and download the appropriate install package.

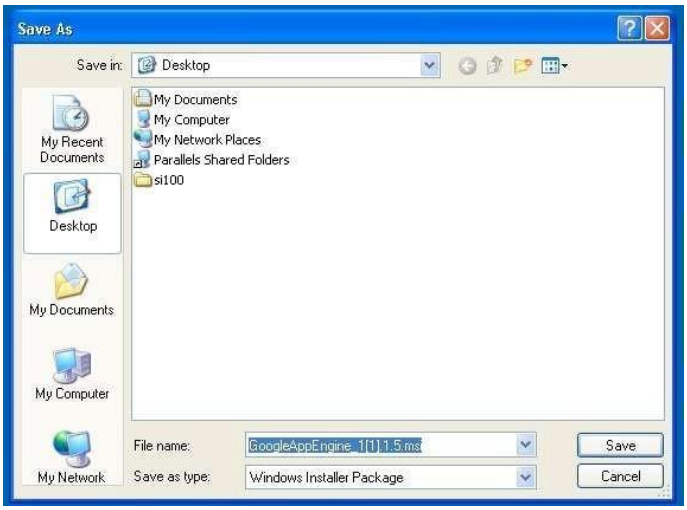
Download the Google App Engine SDK

Before downloading, please read the Terms that govern your use of the App Engine SDK.

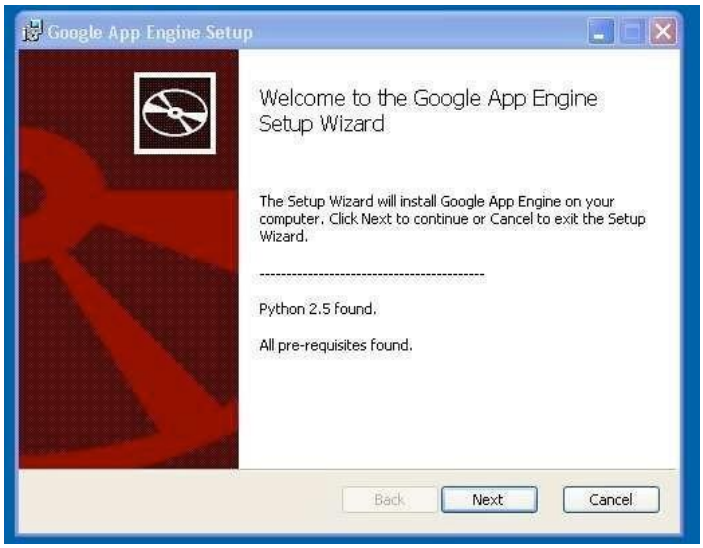
Please note: The App Engine SDK is under **active development**, please keep this in mind as you explore its capabilities. See the SDK Release Notes for the information on the most recent changes to the App Engine SDK. If you discover any issues, please feel free to notify us via our Issue Tracker.

Platform	Version	Package	Size	SHA1 Checksum
Windows	1.1.5 - 10/03/08	GoogleAppEngine_1.1.5.msi	2.5 MB	e974312b4aefc0b3873ff0d93eb4c525d5e88c30
Mac OS X	1.1.5 - 10/03/08	GoogleAppEngineLauncher-1.1.5.dmg	3.6 MB	f62208ac01c1b3e39796e58100d5f1b2f052d3e7
Linux/Other Platforms	1.1.5 - 10/03/08	google_appengine_1.1.5.zip	2.6 MB	cbb9ce817bdabf1c4f181d9544864e55ee253de1

Download the Windows installer – the simplest thing is to download it to your Desktop or another folder that you remember.



Double Click on the **GoogleApplicationEngine** installer.



Click through the installation wizard, and it should install the App Engine. If you do not have Python 2.5, it will install Python 2.5 as well.

Once the install is complete you can discard the downloaded installer

Making your First Application

Now you need to create a simple application. We could use the "+" option to have the launcher make us an application – but instead we will do it by hand to get a better sense of what is going on.

Make a folder for your Google App Engine applications. I am going to make the Folder on my Desktop called "**apps**" – the path to this folder is:

C:\Documents and Settings\csev\Desktop\apps

And then make a sub-folder in within **apps** called "**ae-01-trivial**" – the path to this folder would be:

C:\ Documents and Settings \csev\Desktop\apps\ae-01-trivial

Using a text editor such as JEdit (www.jedit.org), create a file called **app.yaml** in the **ae-01-trivial** folder with the following contents:

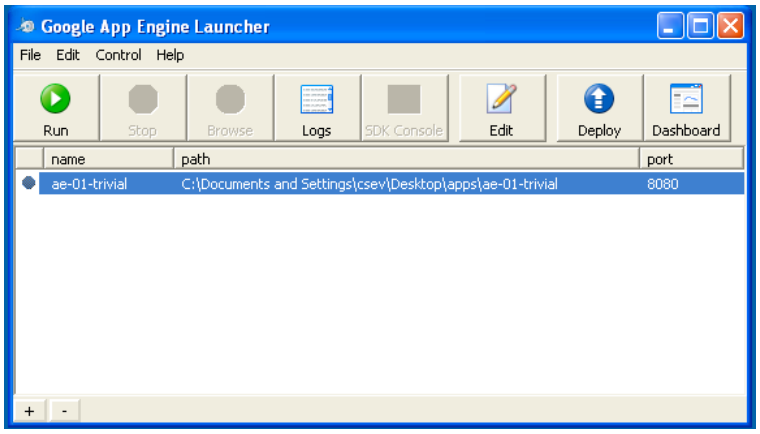
```
application: ae-01-trivial
version: 1
runtime: python
api_version: 1

handlers:
- url: /.*
  script: index.py
```

Note: Please do not copy and paste these lines into your text editor – you might end up with strange characters – simply type them into your editor.

Then create a file in the **ae-01-trivial** folder called **index.py** with three lines in it:

```
print 'Content-Type: text/plain'
print ''
print 'Hello there Chuck'
```

Then start the **GoogleAppEngineLauncher** program that can be found under **Applications**. Use the **File -> Add Existing Application** command and navigate into the **apps** directory and select the **ae-01-trivial** folder. Once you have added the application, select it so that you can control the application using the launcher.



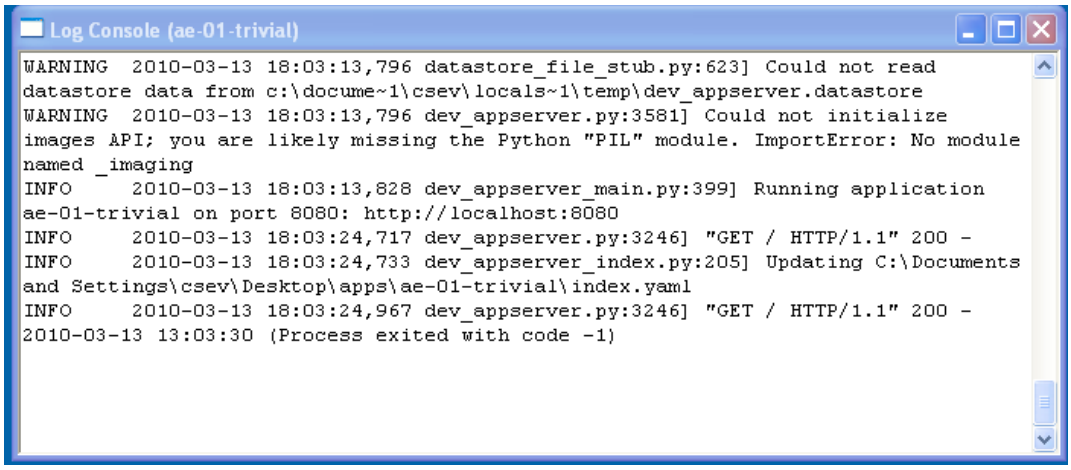
Once you have selected your application and press **Run**. After a few moments your application will start and the launcher will show a little green icon next to your application. Then press **Browse** to open a browser pointing at your application which is running at **http://localhost:8080/**

Paste **http://localhost:8080** into your browser and you should see your application as follows:

Just for fun, edit the **index.py** to change the name "Chuck" to your own name and press Refresh in the browser to verify your updates.

Watching the Log

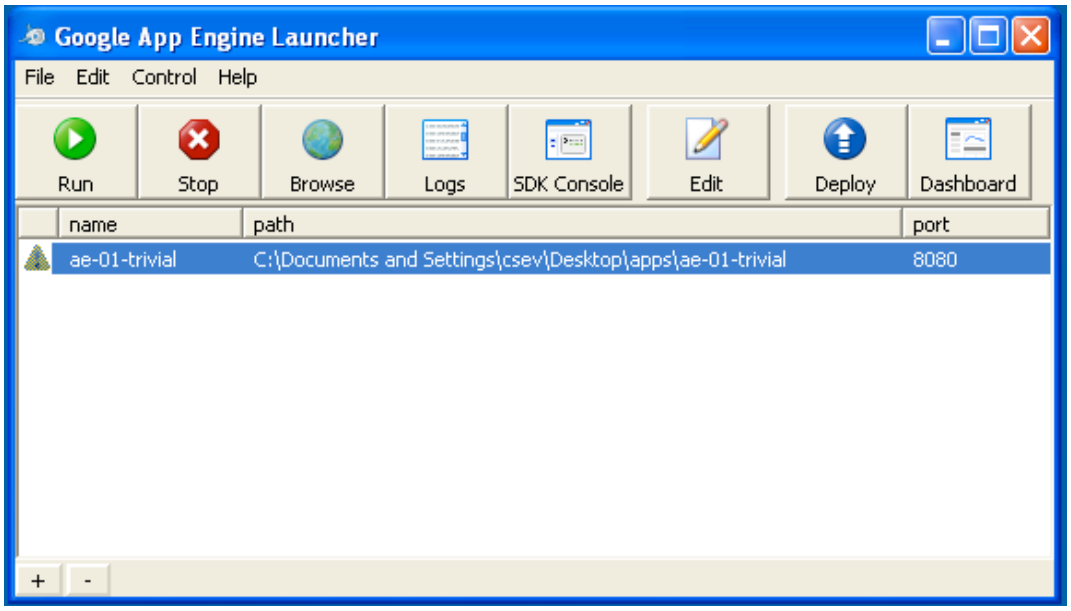
You can watch the internal log of the actions that the web server is performing when you are interacting with your application in the browser. Select your application in the Launcher and press the **Logs** button to bring up a log window:



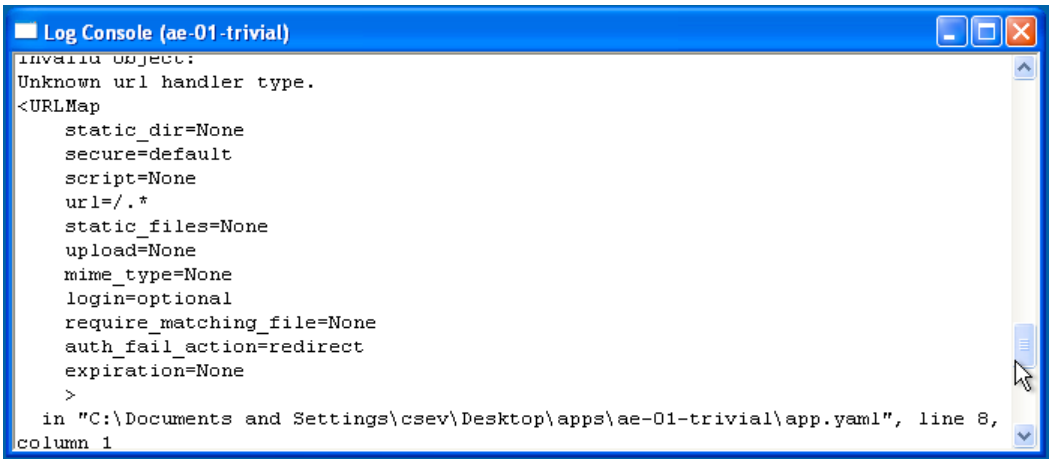
Each time you press **Refresh** in your browser – you can see it retrieving the output with a **GET** request.

Dealing With Errors

With two files to edit, there are two general categories of errors that you may encounter. If you make a mistake on the **app.yaml** file, the App Engine will not start and your launcher will show a yellow icon near your application:

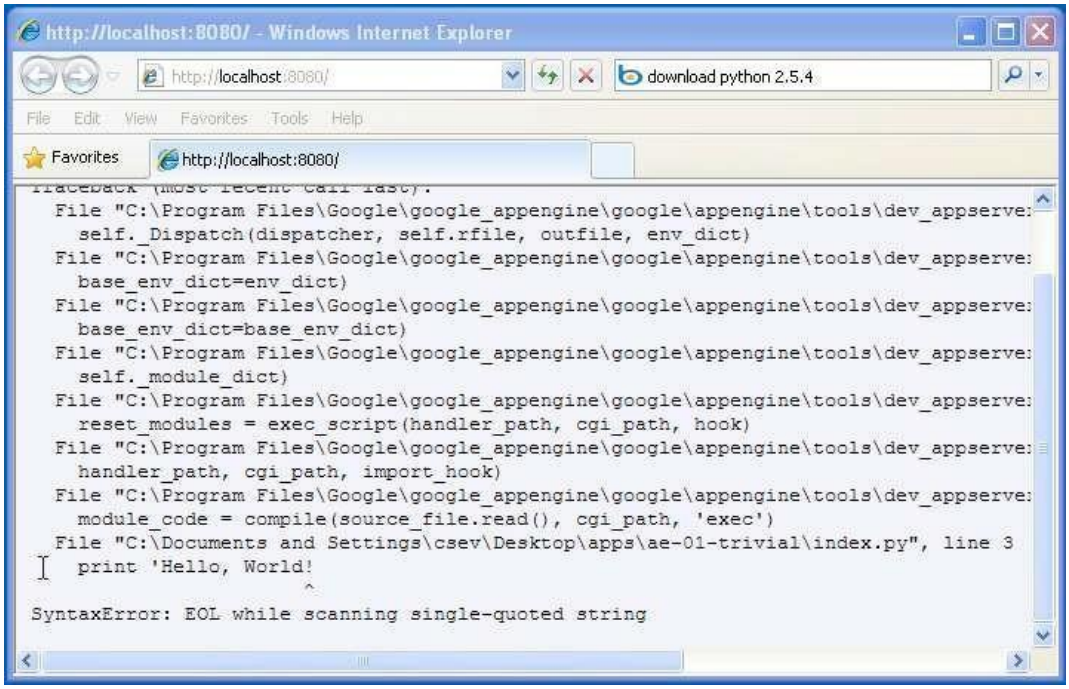


To get more detail on what is going wrong, take a look at the log for the application:



In this instance – the mistake is mis-indenting the last line in the **app.yaml** (line 8).

If you make a syntax error in the **index.py** file, a Python trace back error will appear in your browser.



The error you need to see is likely to be the last few lines of the output – in this case I made a Python syntax error on line one of our one-line application.

Reference: http://en.wikipedia.org/wiki/Stack_trace

When you make a mistake in the **app.yaml** file – you must the fix the mistake and attempt to start the application again.

If you make a mistake in a file like i**ndex.py**, you can simply fix the file and press refresh in your browser – there is no need to restart the server.

Shutting Down the Server

To shut down the server, use the Launcher, select your application and press the **Stop** button.

Comments and questions to csev@umich.edu www.dr-chuck.com

How to use CloudSim in Eclipse

CloudSim is written in Java. The knowledge you need to use CloudSim is basic Java programming and some basics about cloud computing. Knowledge of programming IDEs such as Eclipse or NetBeans is also helpful. It is a library and, hence, CloudSim does not have to be installed. Normally, you can unpack the downloaded package in any directory, add it to the Java classpath and it is ready to be used. Please verify whether Java is available on your system.

To use CloudSim in Eclipse:

1. Download CloudSim installable files from *https://code.google.com/p/cloudsim/downloads/list and unzip*
2. Open Eclipse
3. Create a new Java Project: File -> New
4. Import an unpacked CloudSim project into the new Java Project
5. The first step is to initialise the CloudSim package by initialising the CloudSim library, as follows:

```
CloudSim.init(num_user, calendar, trace_flag)
```

6. Data centres are the resource providers in CloudSim; hence, creation of data centres is a second step. To create Datacenter, you need the DatacenterCharacteristics object that stores the properties of a data centre such as architecture, OS, list of machines, allocation policy that covers the time or spaceshared, the time zone and its price:

```
Datacenter datacenter9883 = new Datacenter(name, characteristics, new VmAllocationPolicySimple(hostList), s
```

7. The third step is to create a broker:

```
DatacenterBroker broker = createBroker();
```

8. The fourth step is to create one virtual machine unique ID of the VM, userId ID of the VM's owner, mips, number Of Pes amount of CPUs, amount of RAM, amount of bandwidth, amount of storage, virtual machine monitor, and cloudletScheduler policy for cloudlets:

```
Vm vm = new Vm(vmid, brokerId, mips, pesNumber, ram, bw, size, vmm, new
CloudletSchedulerTimeShared())
```

9. Submit the VM list to the broker:

```
broker.submitVmList(vmlist)
```

10. Create a cloudlet with length, file size, output size, and utilisation model:

```
Cloudlet cloudlet = new Cloudlet(id, length, pesNumber, fileSize, outputSize, utilizationModel, utilizationMode
```

11. Submit the cloudlet list to the broker:

```
broker.submitCloudletList(cloudletList)
```

12. Start the simulation:

```
CloudSim.startSimulation()
```

Sample Output from the Existing Example:

```
Starting CloudSimExample1...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>null
Broker is starting...
Entities started.
    : Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
    : Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 0 to VM #0
400.1: Broker: Cloudlet 0 received
       : Broker: All Cloudlets executed. Finishing...
: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
========== OUTPUT ==========
Cloudlet ID  STATUS Data center ID  VM ID  Time  Start Time  Finish Time
    0         SUCCESS    2          0      400          0.1        400.1
*****Datacenter: Datacenter_0*****
User id     Debt
    3         35.6

CloudSimExample1 finished!
```

1. You can copy few (or more) lines with ***copy & paste*** mechanism.
   For this you need to share clipboard between host OS and guest OS, installing **Guest Addition** on both the virtual machines (probably setting *bidirectional* and restarting them).
   You *copy* from *guest OS* in the clipboard that is shared with the *host OS*.
   Then you *paste* from the *host OS* to the second *guest OS*.
2. You can enable **drag and drop** too with the same method (Click on the machine, settings, general, advanced, drag and drop: set to *bidirectional*)
3. You can have **common *Shared Folders*** on both virtual machines and use one of the directory shared as buffer to copy.
   Installing **Guest Additions** you have the possibility to set Shared Folders too. As you put a file in a shared folder from *host OS* or from *guest OS*, is immediately visible to the other. (Keep in mind that can arise some problems for date/time of the files when there are different clock settings on the different virtual machines).
   If you use the same folder shared on more machines you can exchange files directly copying them in this folder.
4. You can use **usual method to copy files between 2 different computer** with client-server application. (e.g. scp with sshd active for linux, winscp... you can get some info about SSH servers e.g. here)
   You need an active server (sshd) on the receiving machine and a client on the sending machine. Of course you need to have the authorization setted (via password or, better, via an automatic authentication method).
   Note: many Linux/Ubuntu distribution install sshd by default: you can see if it is running with pgrep sshd from a shell. You can install with sudo apt-get install openssh-server.
5. You can **mount part of the file system** of a virtual machine via NFS or SSHFS on the other, or you can **share file and directory** with Samba.
   You may find interesting the article Sharing files between guest and host without VirtualBox shared folders with detailed step by step instructions.

You should remember that you are dialling with a little network of machines with different operative systems, and in particular:

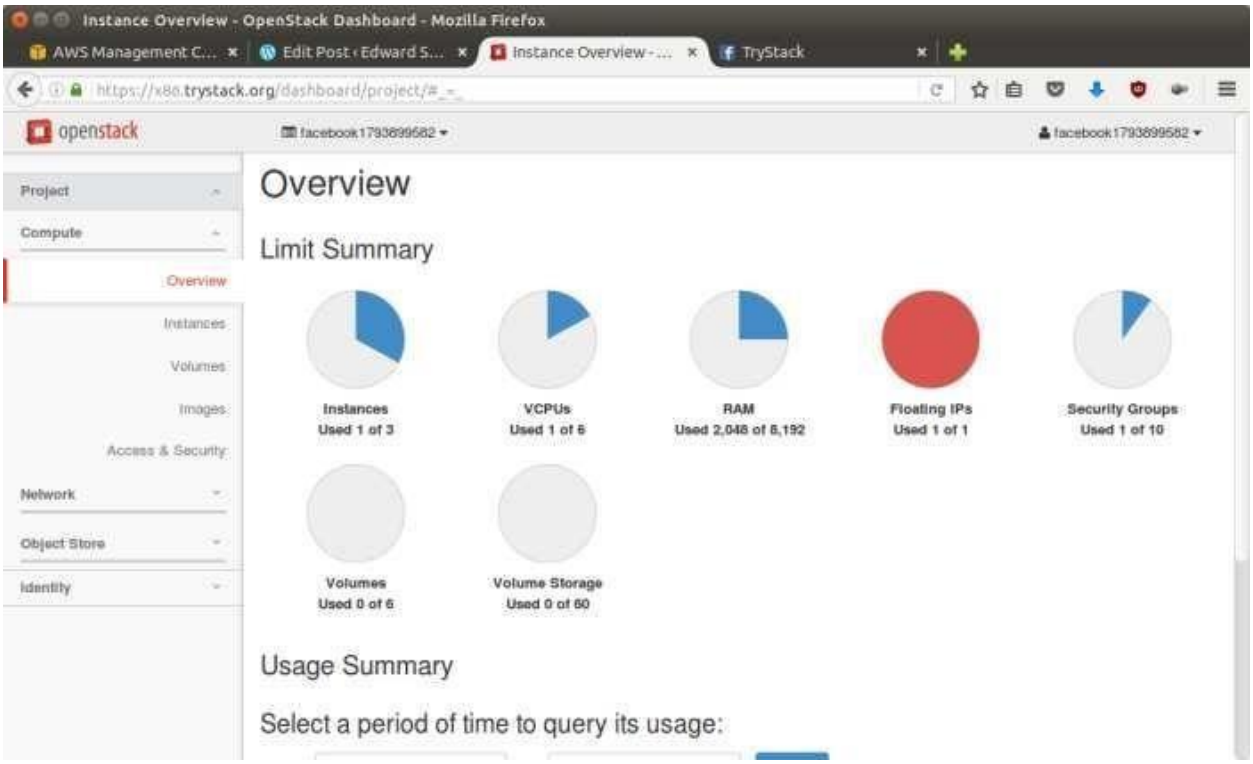
- Each virtual machine has its own operative system running on and acts as a physical machine.
- Each virtual machine is an instance of a program *owned* by an *user* in the hosting operative system and should undergo the restrictions of the *user* in the *hosting OS*.
  E.g Let we say that Hastur and Meow are users of the hosting machine, but they did not allow each other to see their directories (no read/write/execute authorization). When each of them run a virtual machine, for the hosting OS those virtual machine are two normal programs owned by Hastur and Meow and cannot see the private directory of the other user. This is a restriction due to the *hosting OS*. It's easy to overcame it: it's enough to give authorization to read/write/execute to a directory or to chose a different directory in which both users can read/write/execute.
- Windows likes mouse and Linux fingers. :-)
  I mean I suggest you to enable *Drag & drop* to be cosy with the Windows machines and the *Shared folders* or to be cosy with Linux.
  When you will need to be fast with Linux **you will feel the need** of ssh-keygen and

to Generate once SSH Keys to copy files on/from a remote machine without writing password anymore. In this way it functions bash auto-completion remotely too!

OpenStack is an open-source software cloud computing platform. OpenStack is primarily used for deploying an infrastructure as a service (IaaS) solution like Amazon Web Service (AWS). In other words, you can *make your own AWS* by using OpenStack. If you want to try out OpenStack, **TryStack** is the easiest and free way to do it.
In order to try OpenStack in TryStack, you must register yourself by joining TryStack Facebook Group. The acceptance of group needs a couple days because it's approved manually. After you have been accepted in the TryStack Group, you can log in TryStack.

TryStack.org Homepage

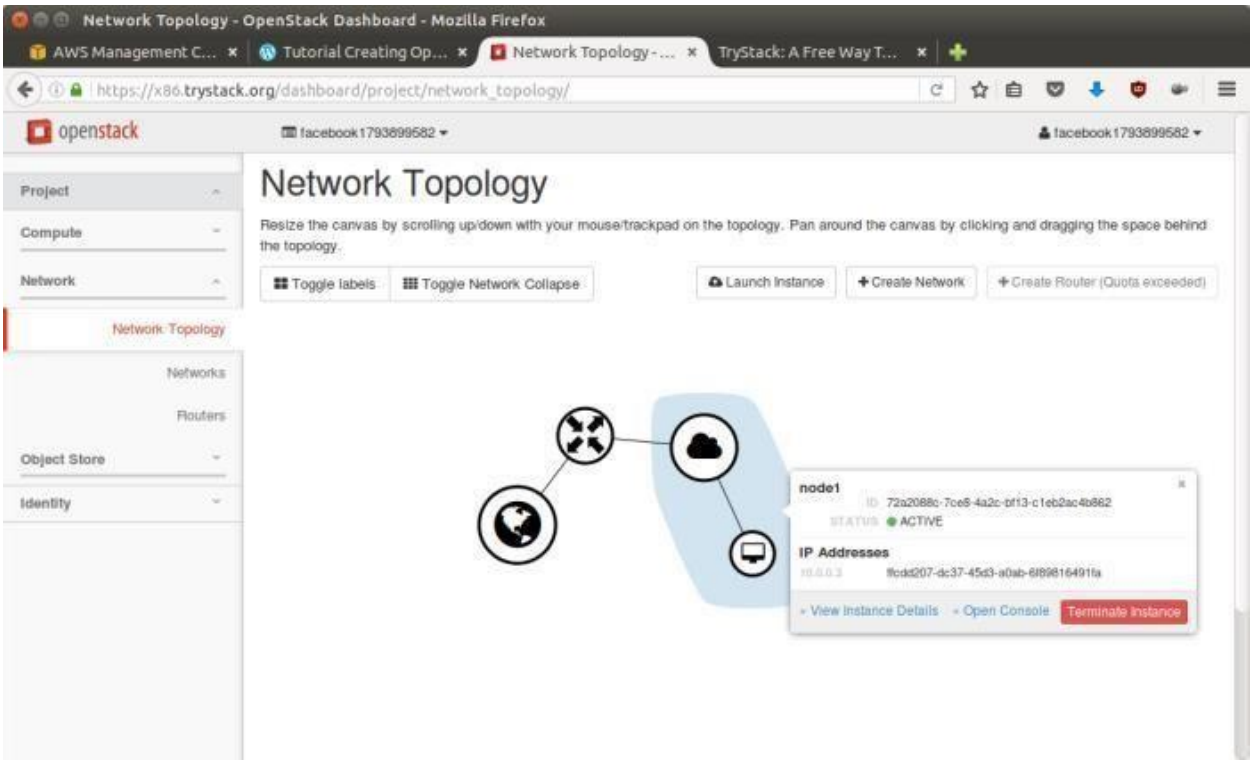
I assume that you already join to the Facebook Group and login to the dashboard. After you log in to the TryStack, you will see the Compute Dashboard like:



OpenStack Compute Dashboard

Overview: What we will do?

In this post, I will show you how to run an OpenStack instance. The instance will be accessible through the internet (have a public IP address). The final topology will like:



Network topology

As you see from the image above, the instance will be connected to a local network and the local network will be connected to internet.

Step 1: Create Network

Network? Yes, the network in here is our own local network. So, your instances will be not mixed up with the others. You can imagine this as your own LAN (Local Area Network) in the cloud.

1. Go to **Network > Networks** and then click **Create Network**.
2. In **Network** tab, fill **Network Name** for example internal and then click **Next**.
3. In **Subnet** tab,
    1. Fill **Network Address** with appropriate CIDR, for example 192.168.1.0/24. Use private network CIDR block as the best practice.
    2. Select **IP Version** with appropriate IP version, in this case IPv4.
    3. Click **Next**.
4. In **Subnet Details** tab, fill **DNS Name Servers** with 8.8.8.8 (Google DNS) and then click **Create**.

Step 2: Create Instance

Now, we will create an instance. The instance is a virtual machine in the cloud, like AWS EC2. You need the instance to connect to the network that we just created in the previous step.

1. Go to **Compute > Instances** and then click **Launch Instance**.
2. In **Details** tab,
    1. Fill **Instance Name**, for example Ubuntu 1.
    2. Select **Flavor**, for example m1.medium.
    3. Fill **Instance Count** with **1**.
    4. Select **Instance Boot Source** with **Boot from Image**.
    5. Select **Image Name** with **Ubuntu 14.04 amd64 (243.7 MB)** if you want install Ubuntu 14.04 in your virtual machine.
3. In **Access & Security** tab,
    1. Click [+] button of **Key Pair** to import key pair. This key pair is a public and private key that we will use to connect to the instance from our machine.
    2. In **Import Key Pair** dialog,
        1. Fill **Key Pair Name** with your machine name (for example Edward-Key).
        2. Fill **Public Key** with your **SSH public key** (usually is in ~/.ssh/id_rsa.pub). See description in Import Key Pair dialog box for more information. If you are using Windows, you can use **Puttygen** to generate key pair.
        3. Click **Import key pair**.
    3. In **Security Groups**, mark/check **default**.
4. In **Networking** tab,
    1. In **Selected Networks**, select network that have been created in Step 1, for example internal.
5. Click **Launch**.
6. If you want to create multiple instances, you can repeat step 1-5. I created one more instance with instance name Ubuntu 2.

Step 3: Create Router

I guess you already know what router is. In the step 1, we created our network, but it is isolated. It doesn't connect to the internet. To make our network has an internet connection, we need a router that running as the gateway to the internet.

1. Go to **Network > Routers** and then click **Create Router**.
2. Fill **Router Name** for example router1 and then click **Create router**.
3. Click on your **router name link**, for example router1, **Router Details** page.
4. Click **Set Gateway** button in upper right:
    1. Select **External networks** with **external**.
    2. Then **OK**.
5. Click **Add Interface** button.
    1. Select **Subnet** with the network that you have been created in Step 1.
    2. Click **Add interface**.
6. Go to **Network > Network Topology**. You will see the network topology. In the example, there are two network, i.e. external and internal, those are bridged by a router. There are instances those are joined to internal network.

Step 4: Configure Floating IP Address

Floating IP address is public IP address. It makes your instance is accessible from the internet. When you launch your instance, the instance will have a private network IP, but no public IP. In OpenStack, the public IPs is collected in a pool and managed by admin (in our case is TryStack). You need to request a public (floating) IP address to be assigned to your instance.

1. Go to **Compute > Instance**.
2. In one of your instances, click **More > Associate Floating IP**.
3. In **IP Address**, click Plus [+].
4. Select **Pool** to **external** and then click **Allocate IP**.
5. Click **Associate**.
6. Now you will get a public IP, e.g. 8.21.28.120, for your instance.

Step 5: Configure Access & Security

OpenStack has a feature like a firewall. It can whitelist/blacklist your in/out connection. It is called *Security Group*.

1. Go to **Compute > Access & Security** and then open **Security Groups** tab.
2. In **default** row, click **Manage Rules**.
3. Click **Add Rule**, choose **ALL ICMP** rule to enable ping into your instance, and then click **Add**.
4. Click **Add Rule**, choose **HTTP** rule to open HTTP port (port 80), and then click **Add**.
5. Click **Add Rule**, choose **SSH** rule to open SSH port (port 22), and then click **Add**.
6. You can open other ports by creating new rules.

Step 6: SSH to Your Instance

Now, you can SSH your instances to the floating IP address that you got in the step 4. If you are using Ubuntu image, the SSH user will be ubuntu.

Install Hadoop

Step 1: Click here to download the Java 8 Package. Save this file in your home directory.

Step 2: Extract the Java Tar File.

***Command*:** tar -xvf jdk-8u101-linux-i586.tar.gz

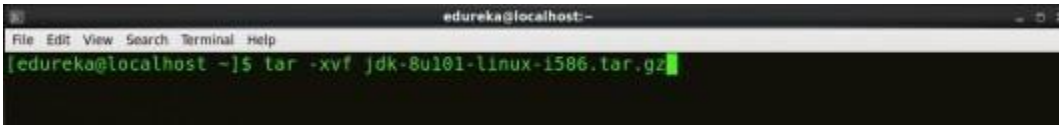


Fig: Hadoop Installation – Extracting Java Files

Step 3: Download the Hadoop 2.7.3 Package.

***Command*:** wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz

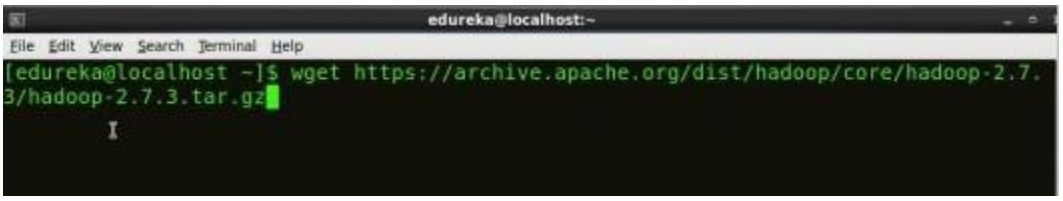


Fig: Hadoop Installation – Downloading Hadoop

Step 4: Extract the Hadoop tar File.

Command: tar -xvf hadoop-2.7.3.tar.gz

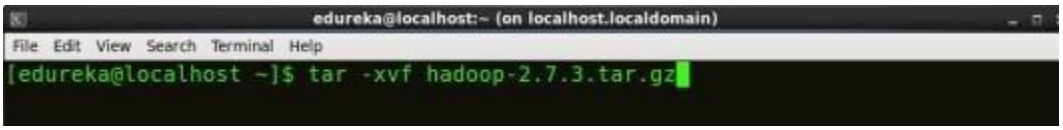


Fig: Hadoop Installation – Extracting Hadoop Files

Step 5: Add the Hadoop and Java paths in the bash file (.bashrc).

Open**. bashrc** file. Now, add Hadoop and Java Path as shown below.

***Command*:** vi .bashrc

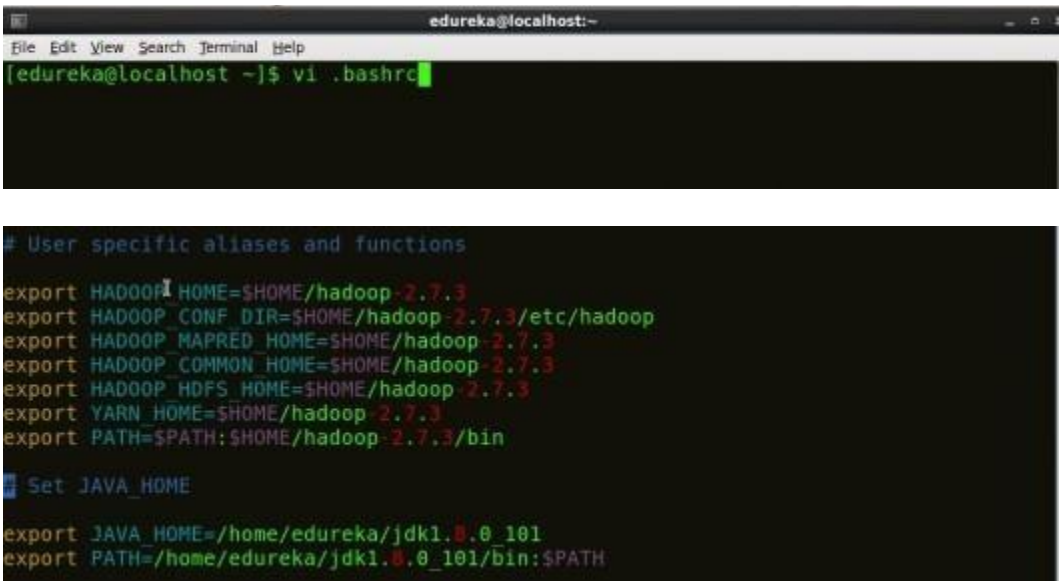


Fig: Hadoop Installation – Setting Environment Variable

Then, save the bash file and close it.

For applying all these changes to the current Terminal, execute the source command.

***Command*:** source .bashrc

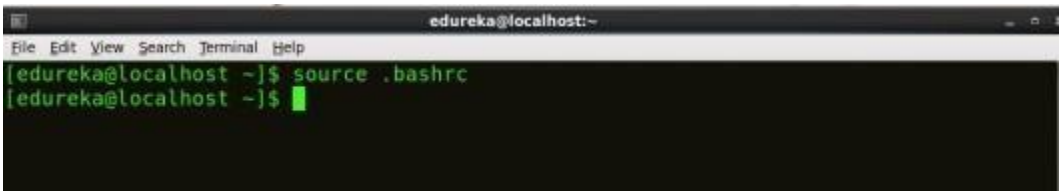


Fig: Hadoop Installation – Refreshing environment variables

To make sure that Java and Hadoop have been properly installed on your system and can be accessed through the Terminal, execute the java -version and hadoop version commands.

***Command*:** java -version

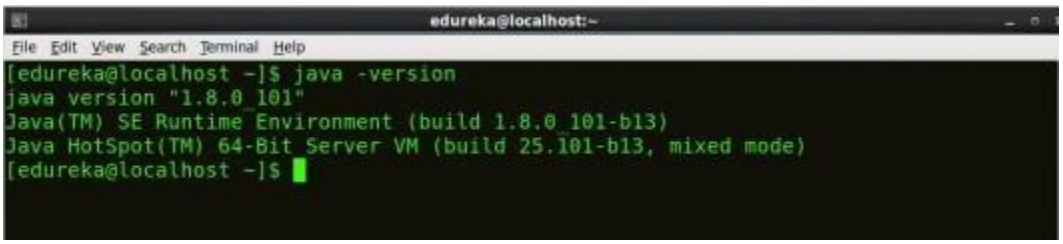


Fig: Hadoop Installation – Checking Java Version

***Command*:** hadoop version

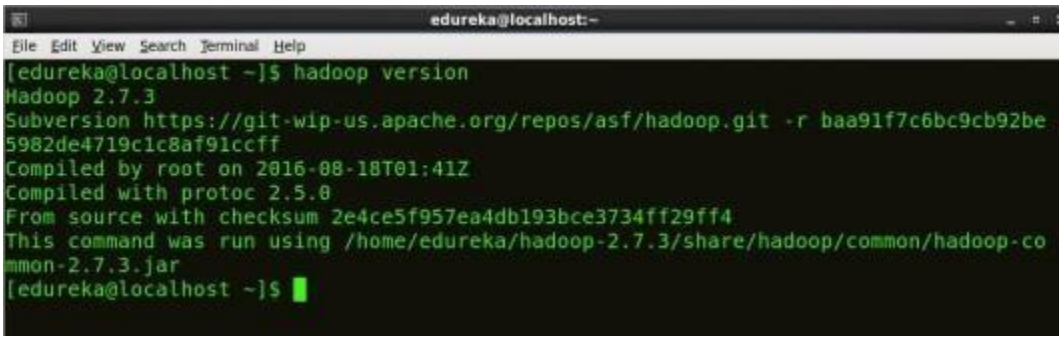


Fig: Hadoop Installation – Checking Hadoop Version

Step 6: Edit the **Hadoop Configuration files**.

Command: cd hadoop-2.7.3/etc/hadoop/

Command: ls

All the Hadoop configuration files are located in **hadoop-2.7.3/etc/hadoop** directory as you can see in the snapshot below:

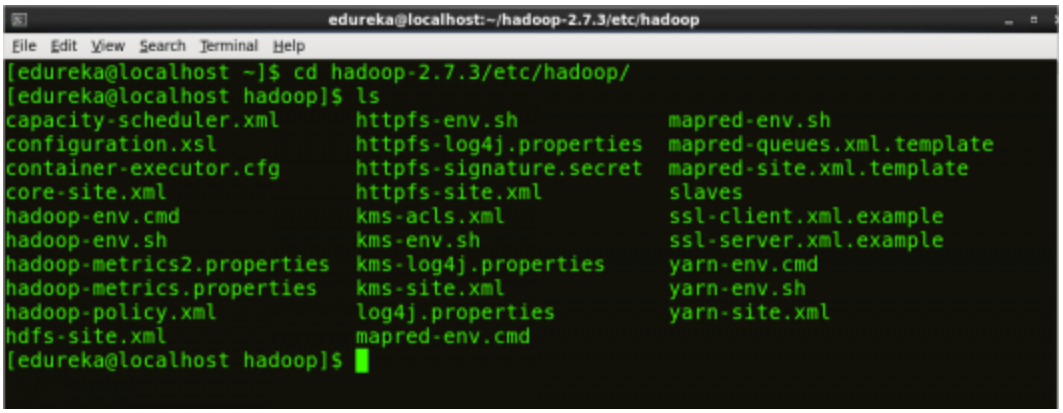


Fig: Hadoop Installation – Hadoop Configuration Files

Step 7: Open *core-site.xml* and edit the property mentioned below inside configuration tag:

core-site.xml informs Hadoop daemon where NameNode runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce.

***Command*:** vi core-site.xml

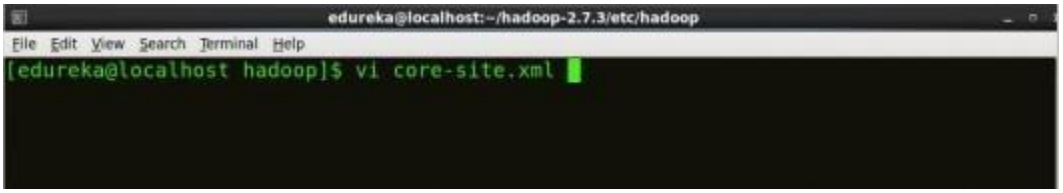


Fig: Hadoop Installation – Configuring core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
```

Step 8: Edit *hdfs-site.xml* and edit the property mentioned below inside configuration tag:

hdfs-site.xml contains configuration settings of HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.

***Command*:** vi hdfs-site.xml

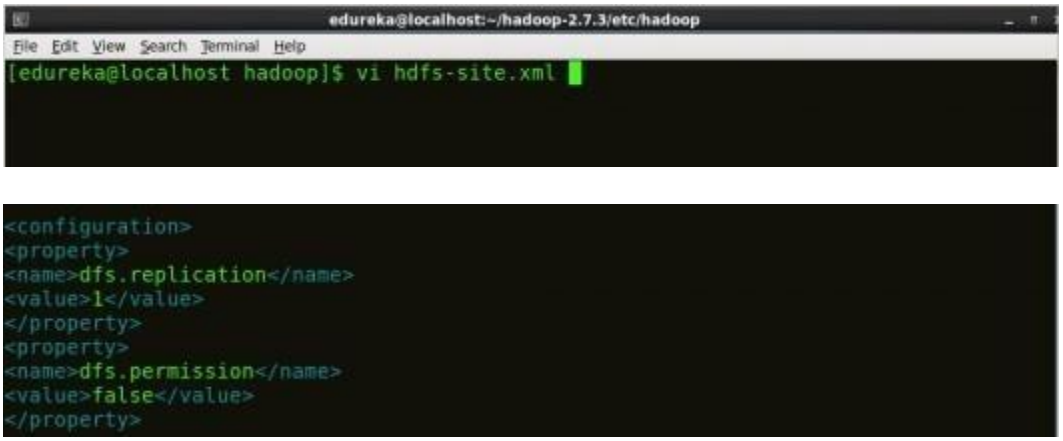


Fig: Hadoop Installation – Configuring hdfs-site.xml

```
1
2           <?xml version="1.0" encoding="UTF-8"?>
3  <?xml-stylesheet  type="text/xsl"  href="configuration.xsl"?>
4                        <configuration>
5                          <property>
6                  <name>dfs.replication</name>
7                        <value>1</value>
8                          </property>
9                          <property>
10                 <name>dfs.permission</name>
11                    <value>false</value>
12                         </property>
                         </configuration>
```

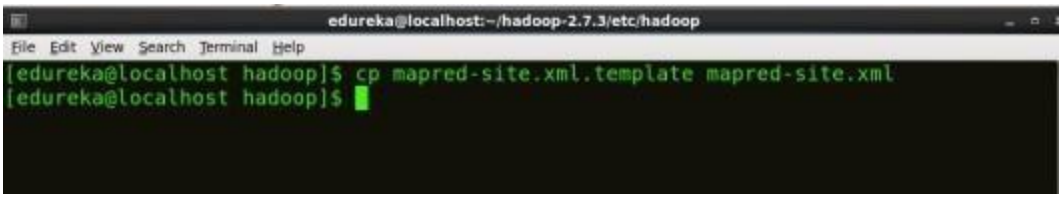
Step 9: Edit the *mapred-site.xml* file and edit the property mentioned below inside configuration tag:

mapred-site.xml contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.

In some cases, mapred-site.xml file is not available. So, we have to create the mapred-site.xml file using mapred-site.xml template.

***Command*:** cp mapred-site.xml.template mapred-site.xml

***Command*:** vi mapred-site.xml.



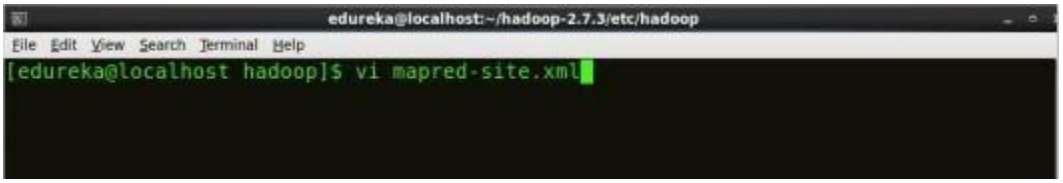


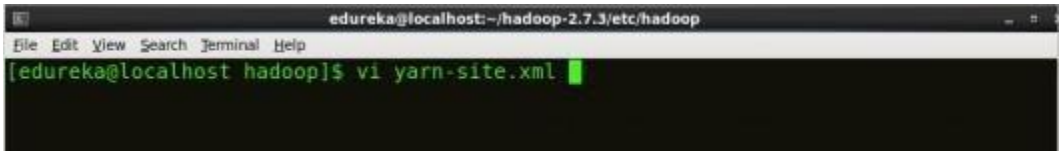
Fig: Hadoop Installation – Configuring mapred-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

Step 10: Edit *yarn-site.xml* and edit the property mentioned below inside configuration tag:

yarn-site.xml contains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.

***Command*:** vi yarn-site.xml



```
<configuration>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
</configuration>
```

Fig: Hadoop Installation – Configuring yarn-site.xml

```
<?xml version="1.0">
<configuration>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</
name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
</configuration>
```

Step 11: Edit *hadoop-env.sh* and add the Java Path as mentioned below:

hadoop-env.sh contains the environment variables that are used in the script to run Hadoop like Java home path, etc.

***Command*:** vi hadoop–env.sh

```
edureka@localhost:~/hadoop-2.7.3/etc/hadoop
File Edit View Search Terminal Help
[edureka@localhost hadoop]$ vi hadoop-env.sh
```

```
# The java implementation to use.
export JAVA_HOME=/home/edureka/jdk1.8.0_101
```

Fig: Hadoop Installation – Configuring hadoop-env.sh

Step 12: Go to Hadoop home directory and format the NameNode.

***Command*:** cd

***Command*:** cd hadoop-2.7.3

***Command*:** bin/hadoop namenode -format

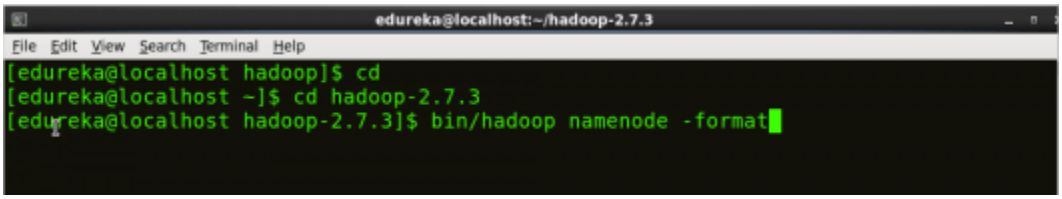


Fig: Hadoop Installation – Formatting NameNode

This formats the HDFS via NameNode. This command is only executed for the first time. Formatting the file system means initializing the directory specified by the dfs.name.dir variable.

Never format, up and running Hadoop filesystem. You will lose all your data stored in the HDFS.

Step 13: Once the NameNode is formatted, go to hadoop-2.7.3/sbin directory and start all the daemons.

Command: cd hadoop-2.7.3/sbin

Either you can start all daemons with a single command or do it individually.

Command: ./start-all.sh

The above command is a combination of ***start-dfs.sh, start-yarn.sh*** & ***mr-jobhistory-daemon.sh***

Or you can run all the services individually as below:

Start NameNode:

The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks all the file stored across the cluster.

Command: ./hadoop-daemon.sh start namenode

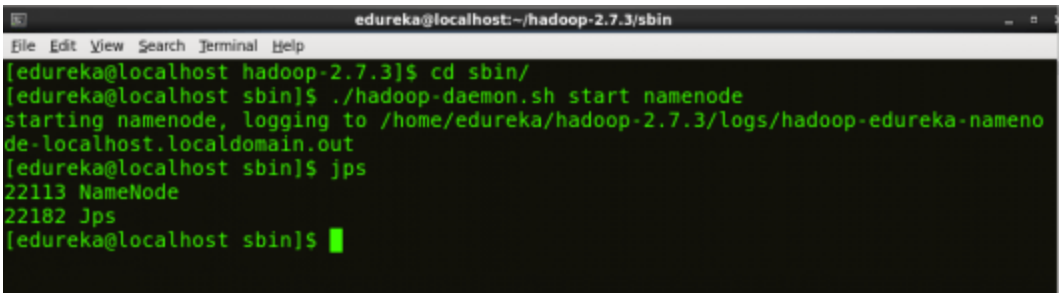


Fig: Hadoop Installation – Starting NameNode

Start DataNode:

On startup, a DataNode connects to the Namenode and it responds to the requests from the Namenode for different operations.

Command: ./hadoop-daemon.sh start datanode

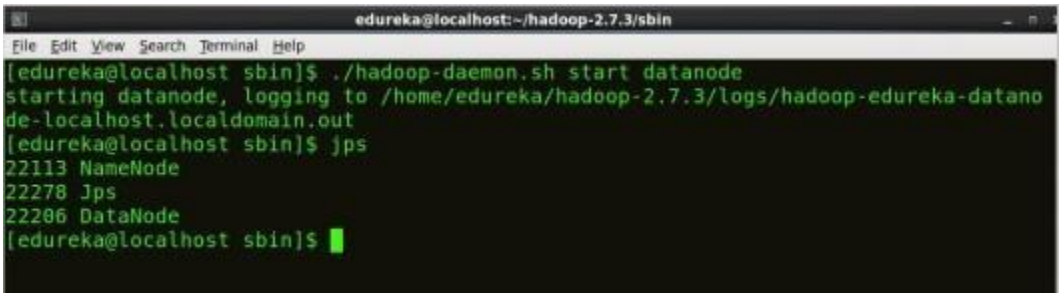


Fig: Hadoop Installation – Starting DataNode

Start ResourceManager:

ResourceManager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system.Its work is to manage each NodeManagers and the each application's ApplicationMaster.

Command: ./yarn-daemon.sh start resourcemanager

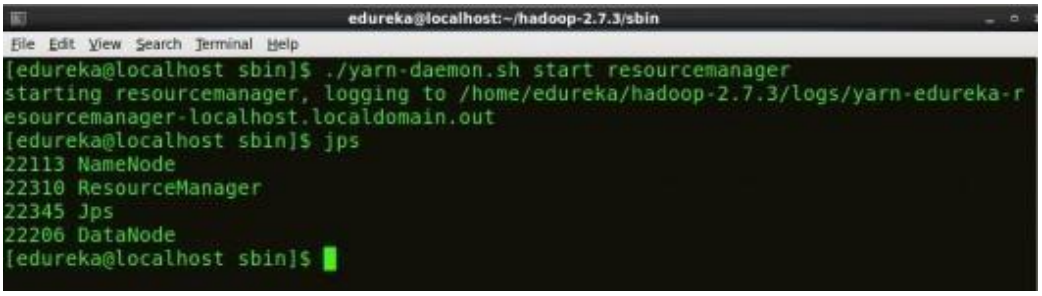


Fig: Hadoop Installation – Starting ResourceManager

Start NodeManager:

The NodeManager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the ResourceManager.

Command: ./yarn-daemon.sh start nodemanager

Fig: Hadoop Installation – Starting NodeManager

Start JobHistoryServer:

JobHistoryServer is responsible for servicing all job history related requests from client.

***Command*:** ./mr-jobhistory-daemon.sh start historyserver

Step 14: To check that all the Hadoop services are up and running, run the below command.

Command: jps

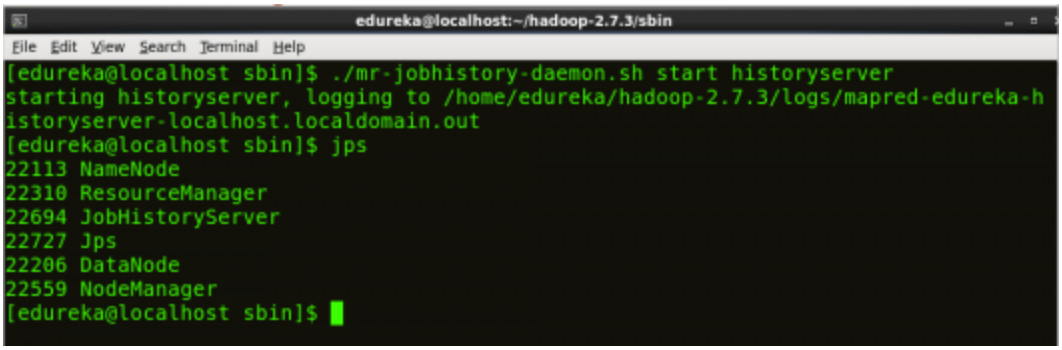


Fig: Hadoop Installation – Checking Daemons

Step 15: Now open the Mozilla browser and go to **localhost**:**50070/dfshealth.html** to check the NameNode interface.

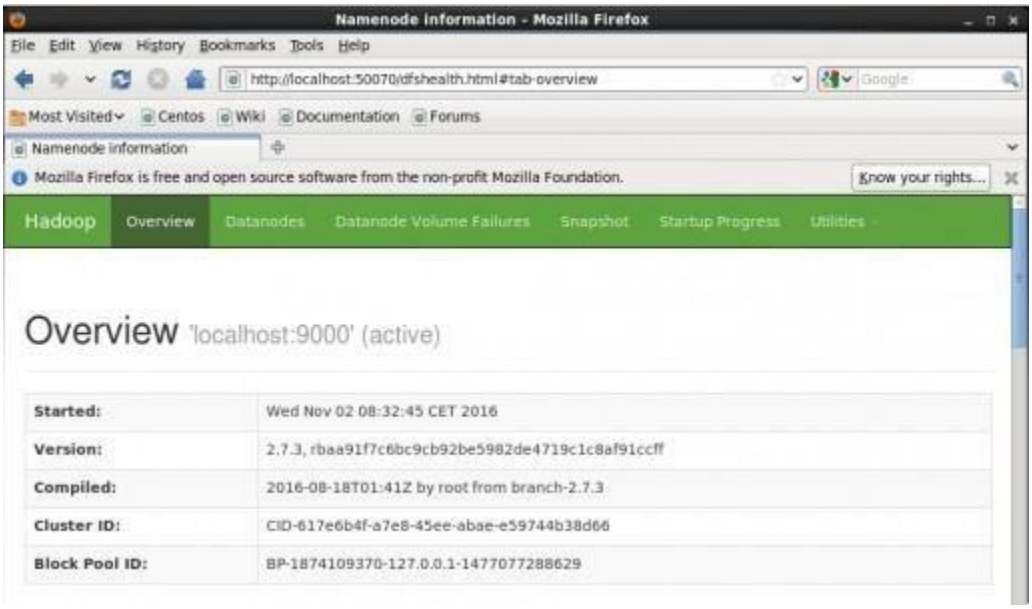


Fig: Hadoop Installation – Starting WebUI

Congratulations, you have successfully installed a single node Hadoop cluster

CLOUD COMPUTING

CONTENT BEYOND SYLLABUS

LIST OF EXPERIMENTS:

1. Develop a new Web Service for Calculator.
2. Develop new OGSA-compliant Web Service.
3. Using Apache Axis develop a Grid Service.
4. Develop applications using Java or C/C++ Grid APIs

1. Develop a new Web Service for Calculator.

OBJECTIVE:
To develop a new Web service for Calculator applications.

PROCEDURE:
When you start Globus toolkit container, there will be number of services starts up. The service for this task will be a simple Math service that can perform basic arithmetic for a client.

The Math service will access a resource with two properties:
1. An integer value that can be operated upon by the service
2. A string values that holds string describing the last operation

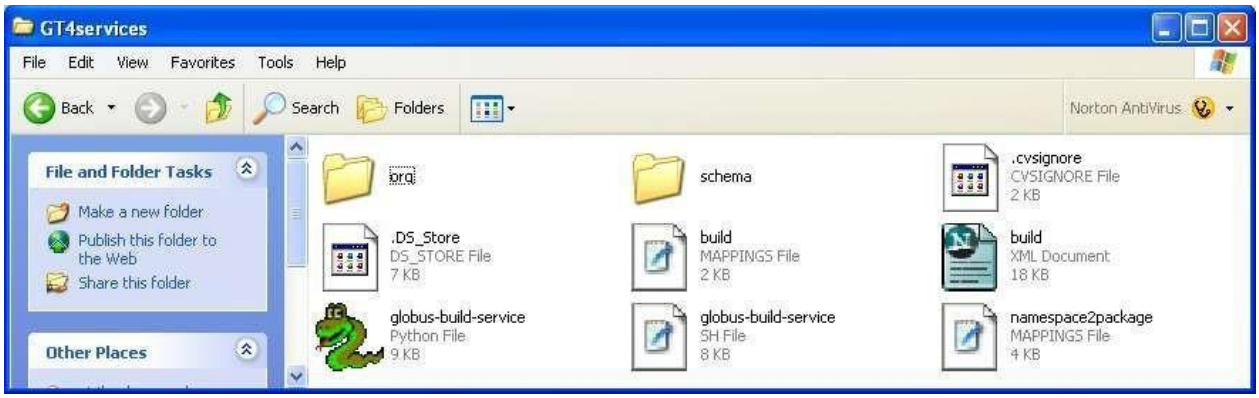
The service itself will have three remotely accessible operations that operate upon *value*:
(a) add, that adds *a* to the resource property *value*.
(b) subtract that subtracts *a* from the resource property *value*.
(c) getValueRP that returns the current value of *value*.

Usually, the best way for any programming task is to begin with an overall description of what you want the code to do, which in this case is the service interface. The service interface describes how what the service provides in terms of names of operations, their arguments and return values. A Java interface for our service is:

```
public interface Math {
public void add(int a);
public void subtract(int a);
public int getValueRP();
}
```

It is possible to start with this interface and create the necessary WSDL file using the standard Web service tool called Java2WSDL. However, the WSDL file for GT 4 has to include details of resource properties that are not given explicitly in the interface above. Hence, we will provide the WSDL file.

Step 1 Getting the Files
All the required files are provided and comes directly from [1]. The MathService source code files can be found from http://www.gt4book.com (http://www.gt4book.com/downloads/gt4book-examples.tar.gz)
A Windows zip compressed version can be found at http://www.cs.uncc.edu/~abw/ITCS4146S07/gt4book-examples.zip. Download and uncompress the file into a directory called **GT4services**. Everything is included (the java source WSDL and deployment files, etc.):



WSDL service interface description file -- ***The WSDL service interface description file is provided within the GT4services folder at: GT4Services\schema\examples\MathService_instance\Math.wsdl***
This file, and discussion of its contents, can be found in Appendix A. Later on we will need to modify this file, but first we will use the existing contents that describe the Math service above.
Service code in Java -- For this assignment, both the code for service operations and for the resource properties are put in the same class for convenience. More complex services and resources would be defined in separate classes. The Java code for the service and its resource properties is located within the GT4services folder at:

GT4services\org\globus\examples\services\core\first\impl\MathService.java.
Deployment Descriptor -- The deployment descriptor gives several different important sets of information about the service once it is deployed. It is located within the **GT4services** folder at:
GT4services\org\globus\examples\services\core\first\deploy-server.wsdd.

Step 2 – Building the Math Service
It is now necessary to package all the required files into a GAR (Grid Archive) file. The build tool ant from the Apache Software Foundation is used to achieve this as shown overleaf:
Generating a GAR file with Ant (from http://gdp.globus.org/gt4-tutorial/multiplehtml/ch03s04.html)
Ant is similar in concept to the Unix make tool but a java tool and XML based.
Build scripts are provided by Globus 4 to use the ant build file. The windows version of the build script for MathService is the Python file called **globus-build-service.py**, which held in the **GT4services** directory. The build script takes one argument, the name of your service that you want to deploy. To keep with the naming convention in [1], this service will be called **first**.
In the *Client Window*, run the build script from the **GT4services** directory with:
globus-build-service.py first
The output should look similar to the following:
Buildfile: build.xml
.
.
.
.
.
BUILD SUCCESSFUL
Total time: 8 seconds
During the build process, a new directory is created in your **GT4Services** directory that is named **build**. All of your stubs and class files that were generated will be in that directory and

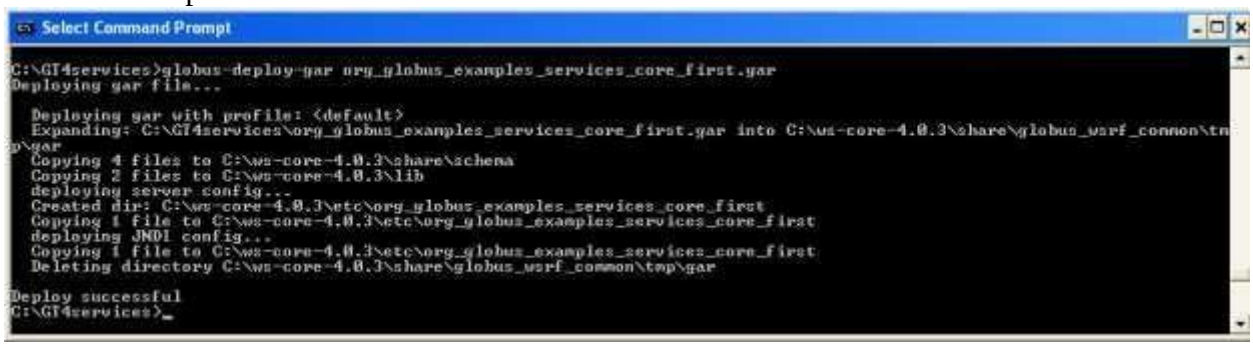
its subdirectories. More importantly, there is a GAR (Grid Archive) file called **org_globus_examples_services_core_first.gar**. The GAR file is the package that contains every file that is needed to successfully deploy your Math Service into the Globus container. The files contained in the GAR file are the Java class files, WSDL, compiled stubs, and the deployment descriptor.

Step 3 – Deploying the Math Service

If the container is still running in the Container Window, then stop it using Control-C. To deploy the Math Service, you will use a tool provided by the Globus Toolkit called **globus-deploy-gar**. In the *Container Window*, issue the command:

globus-deploy-gar org_globus_examples_services_core_first.gar

Successful output of the command is :



The service has now been deployed.

Check service is deployed by starting container from the *Container Window*:

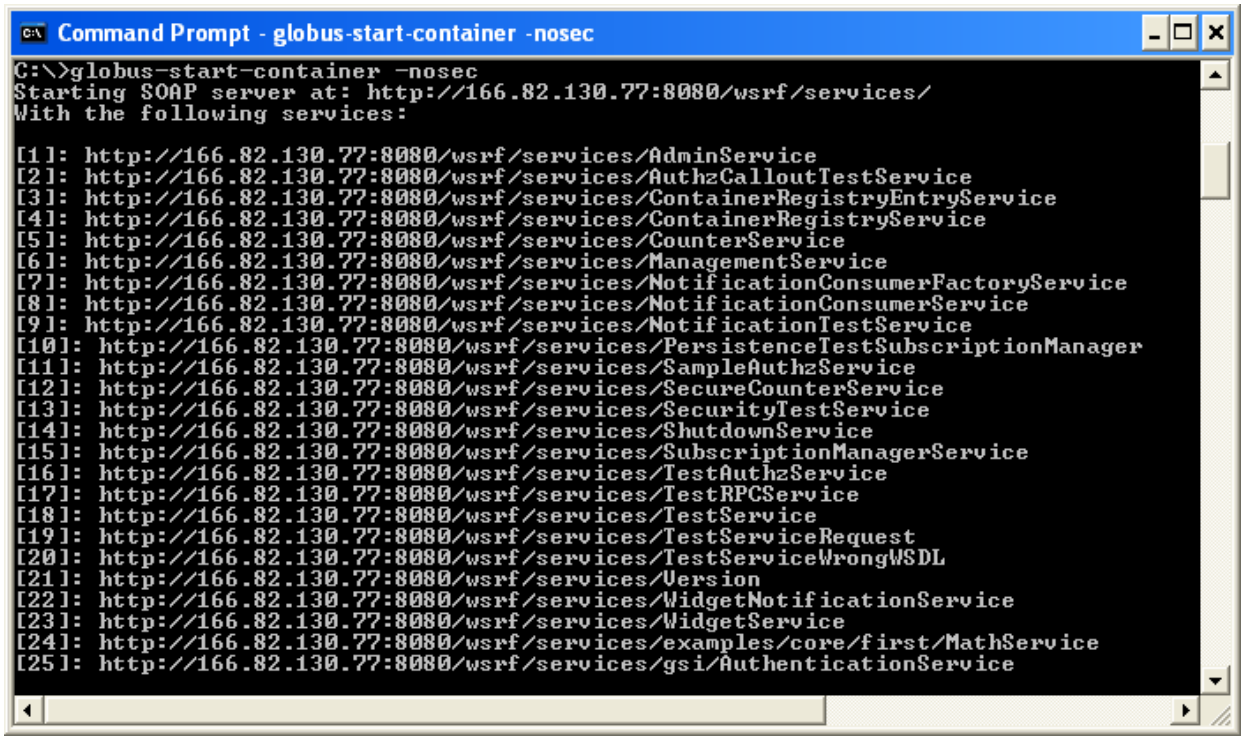
You should see the service called **MathService**.

Step 4 – Compiling the Client

A client has already been provided to test the Math Service and is located in the **GT4Services** directory at:

GT4Services\org\globus\examples\clients\MathService_instance\Client.java

and contains



You should see the service called **MathService**.

Step 4 – Compiling the Client
A client has already been provided to test the Math Service and is located in the **GT4Services** directory at:
GT4Services\org\globus\examples\clients\MathService_instance\Client.java

and contains the following code:

```
package org.globus.examples.clients.MathService_instance;
import org.apache.axis.message.addressing.Address;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.globus.examples.stubs.MathService_instance.MathPortType;
import org.globus.examples.stubs.MathService_instance.GetValueRP;
import
org.globus.examples.stubs.MathService_instance.service.MathServiceAddressingL
ocator;
public class Client {
public static void main(String[] args) {
MathServiceAddressingLocator locator = new
MathServiceAddressingLocator()
try {
String serviceURI = args[0];
// Create endpoint reference to service
EndpointReferenceType endpoint = new
EndpointReferenceType();
endpoint.setAddress(new Address(serviceURI));
MathPortType math;
// Get PortType
math = locator.getMathPortTypePort(endpoint);
// Perform an addition
math.add(10);
// Perform another addition
math.add(5);
// Access value
System.out.println("Current value: "
+ math.getValueRP(new GetValueRP()));
// Perform a subtraction
math.subtract(5);
// Access value
System.out.println("Current value: "
+ math.getValueRP(new GetValueRP()));
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

When the client is run from the command line, you pass it one argument. The argument is the URL that specifies where the service resides. The client will create the end point rerference and incorporate this URL as the address. The end point reference is then used with the **getMathPortTypePort** method of a **MathServiceAdressingLocator** object to obtain a reference to the Math interface (portType). Then, we can apply the methods available in the

service as though they were local methods Notice that the call to the service (add and subtract method calls) must be in a "**try {} catch(){}**" block because a "RemoteException" may be thrown. The code for the "**MathServiceAddressingLocator**" is created during the build process. (Thus you don't have to write it!)

(a) Settting the Classpath

To compile the new client, you will need the JAR files from the Globus toolkit in your CLASSPATH. Do this by executing the following command in the Client Window:

%GLOBUS_LOCATION%\etc\globus-devel-env.bat

You can verify that this sets your CLASSPATH, by executing the command:

echo %CLASSPATH%

You should see a long list of JAR files.

Running **\gt4\etc\globus-devel-env.bat** only needs to be done *once* for each *Client Window* that you open. It does *not* need to be done each time you compile.

(b) Compiling Client

Once your CLASSPATH has been set, then you can compile the Client code by typing in the following command:

javac -classpath
build\classes\org\globus\examples\services\core\first\impl\:%CLASSPATH%
org\globus\examples\clients\MathService_instance\Client.java

Step 5 – Start the Container for your Service

Restart the Globus container from the *Container Window* with:

globus-start-container -nosec

if the container is not running.

Step 6 – Run the Client

To start the client from your **GT4Services** directory, do the following in the *Client Window*, which passes the GSH of the service as an argument:

java -classpath
build\classes\org\globus\examples\services\core\first\impl\:%CLASSPATH%
org.globus.examples.clients.MathService_instance.Client
http://localhost:8080/wsrf/services/examples/core/first/MathService

which should give the output:

Current value: 15
Current value: 10

Step 7 – Undeploy the Math Service and Kill a Container

Before we can add functionality to the Math Service (Section 5), we must undeploy the service. In the *Container Window*, kill the container with a Control-C. Then to undeploy the service, type in the following command:

globus-undeploy-gar org_globus_examples_services_core_first

which should result with the following output:

Undeploying gar...
Deleting /.
.
.
Undeploy successful

In this final task, you are asked to modify the Math service and associated files so the srvice supports the multiplication operation. To do this task, you will need to modify:

- Service code (**MathService.java**)
- WSDL file (**Math.wsdl)**

The exact changes that are necessary are not given. You are to work them out yourself. You will need to fully understand the contents of service code and WSDL files and then modify them accordingly. Appendix A gives an explanation of the important parts of these files. Keep all file names the same and simply redeploy the service afterwards. You will also need to add a code to the client code (**Client.java**) to test the modified service to include multiplication.

Result:

Thus the Develop a new Web Service for Calculator was executed successfully.

2. Develop new OGSA-compliant Web Service

OBJECTIVE:

To develop a new OGSA-compliant web service.

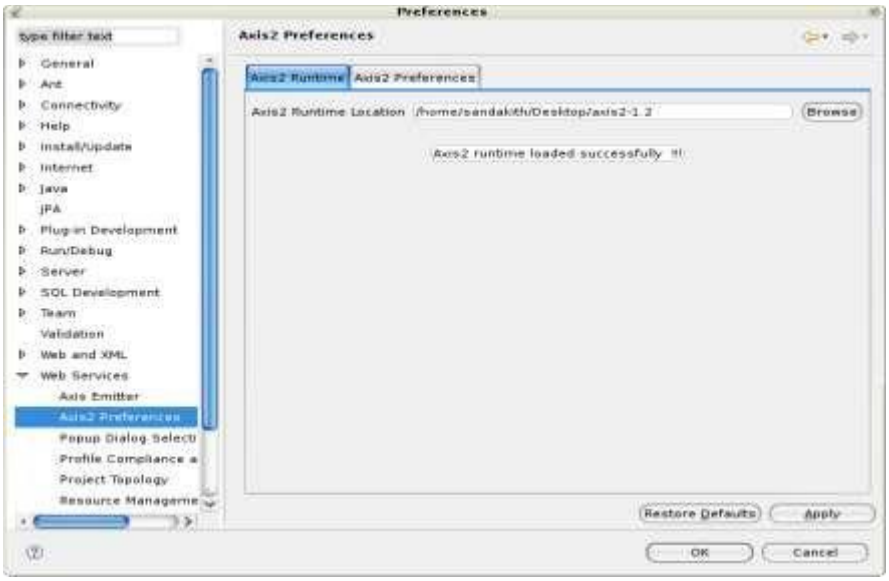
PROCEDURE:

Writing and deploying a WSRF Web Service is easier than you might think. You just have to follow five simple steps

1. Define the service's interface. This is done with *WSDL*
2. Implement the service. This is done with *Java*.
3. Define the deployment parameters. This is done with *WSDD* and *JNDI*
4. Compile everything and generate a GAR file. This is done with *Ant*
5. Deploy service. This is also done with *a GT4 tool*

To run this program, as a minimum you will be required to have installed the following prerequisite software

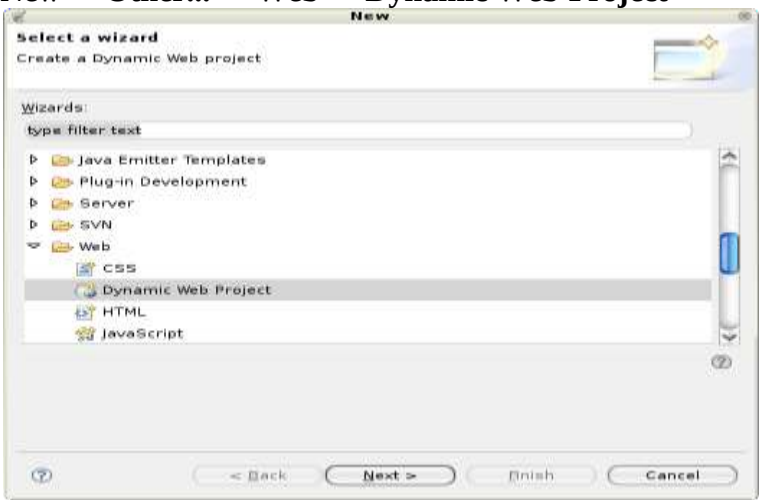
a. Download the latest Axis2 runtime from the above link and extract it. Now we point Eclipse WTP to downloaded Axis2 Runtime. Open **Window -> Preferences -> Web Services -> Axis2 Emitter**



Select the Axis2 Runtime tab and point to the correct Axis2 runtime location. Alternatively at the Axis2 Preference tab, you can set the default setting that will come up on the Web Services Creation wizards. For the moment we will accept the default settings.

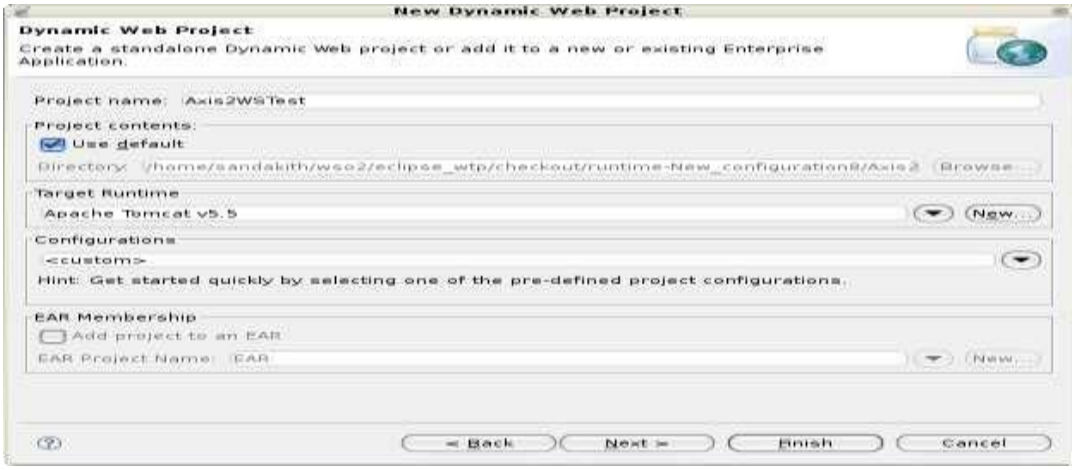
b. Click OK.

c. Next we need to create a project with the support of Axis2 features. Open **File -> New -> Other... -> Web -> Dynamic Web Project**



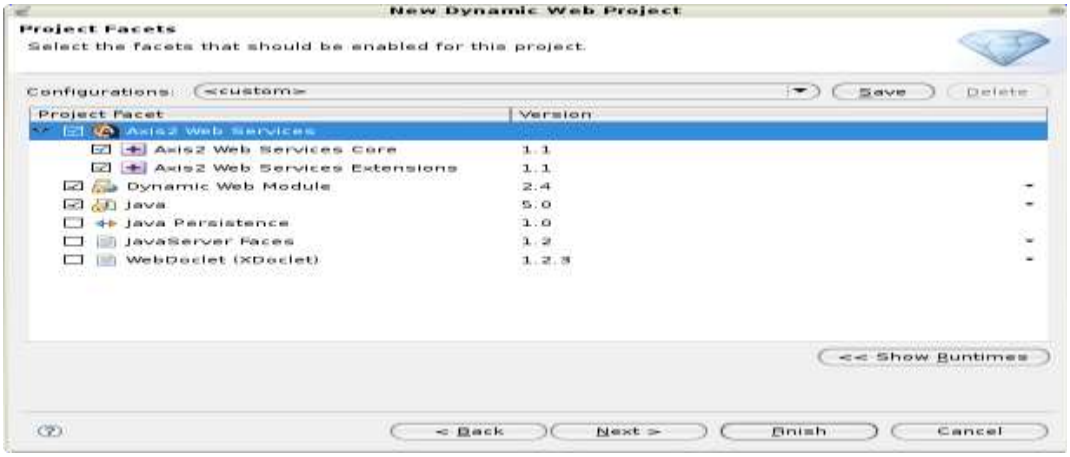
Click next

d. Select the name **Axis2WSTest** as the Dynamic Web project name (you can specify any name you prefer), and select the configured Tomcat runtime as the target runtime.



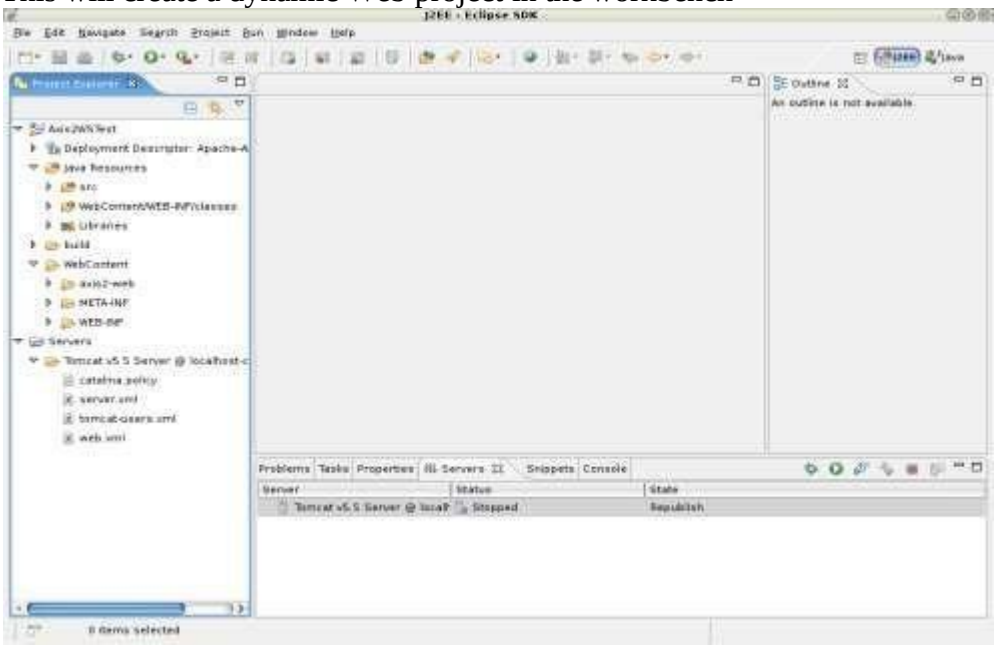
Click next.

e. Select the Axis2 Web service facet

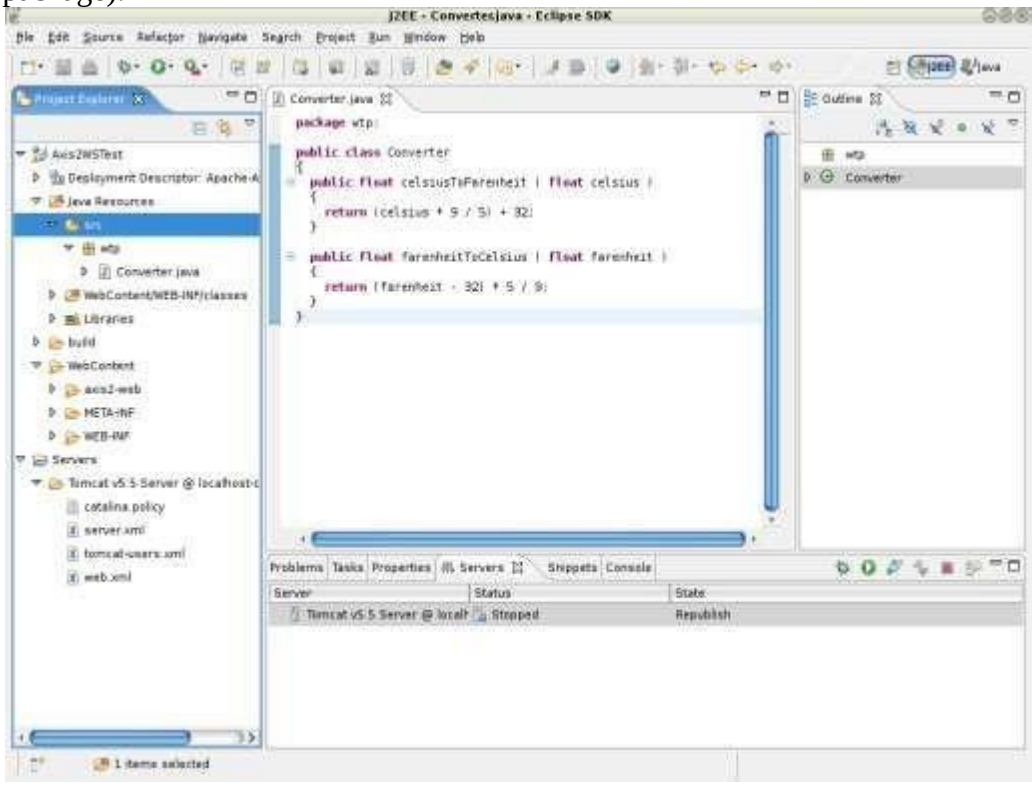


Click Finish.

f. This will create a dynamic Web project in the workbench

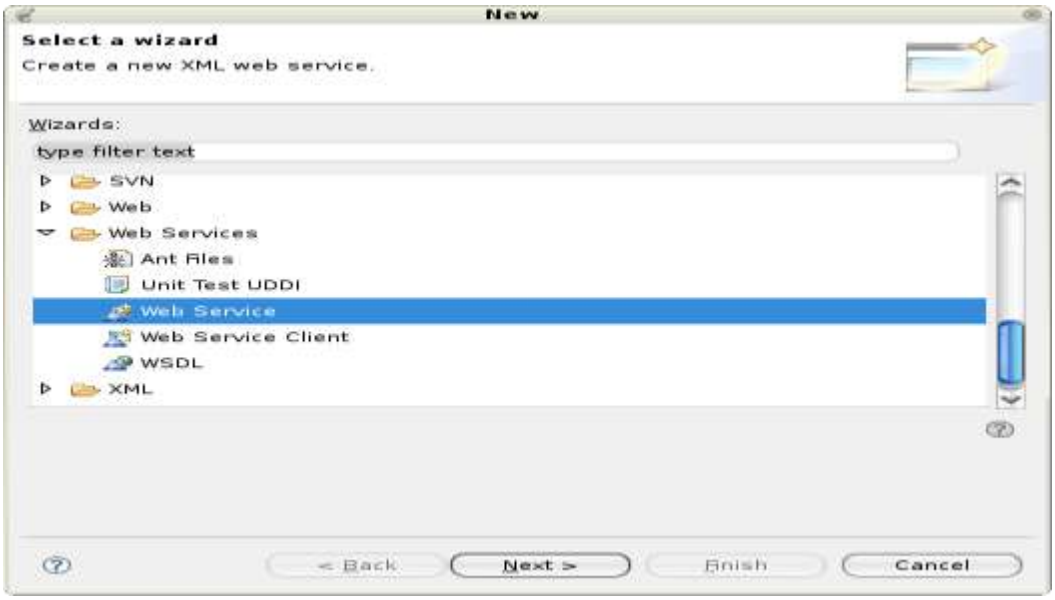


g. Import the wtp/Converter.java class into Axis2WSTest/src (be sure to preserve the package).



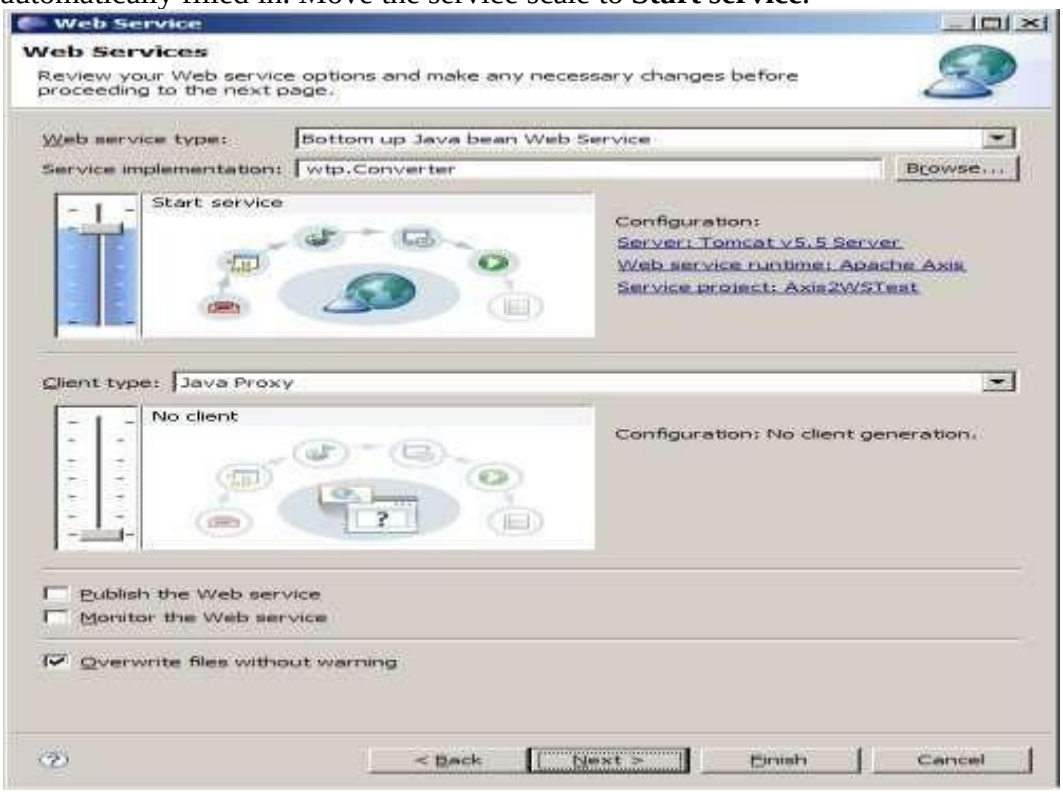
Build the Project, if its not auto build.

h. Select Converter.java, open File -> New -> Other... -> Web Services -> Web Service

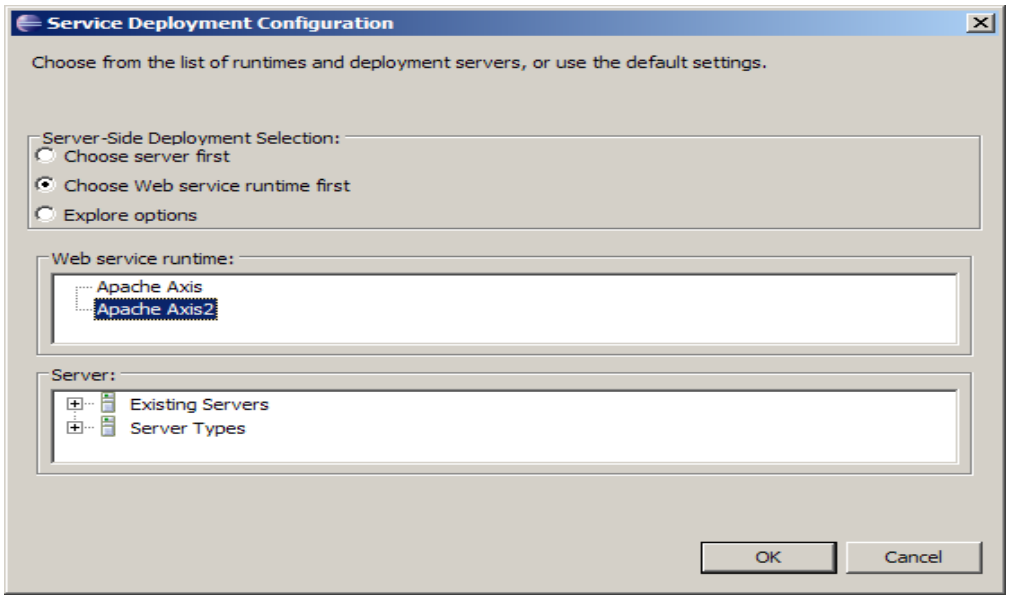


Click next.

i. The Web service wizard would be brought up with Web service type set to **Bottom up Java bean Web Service** with the service implementation automatically filled in. Move the service scale to **Start service**.

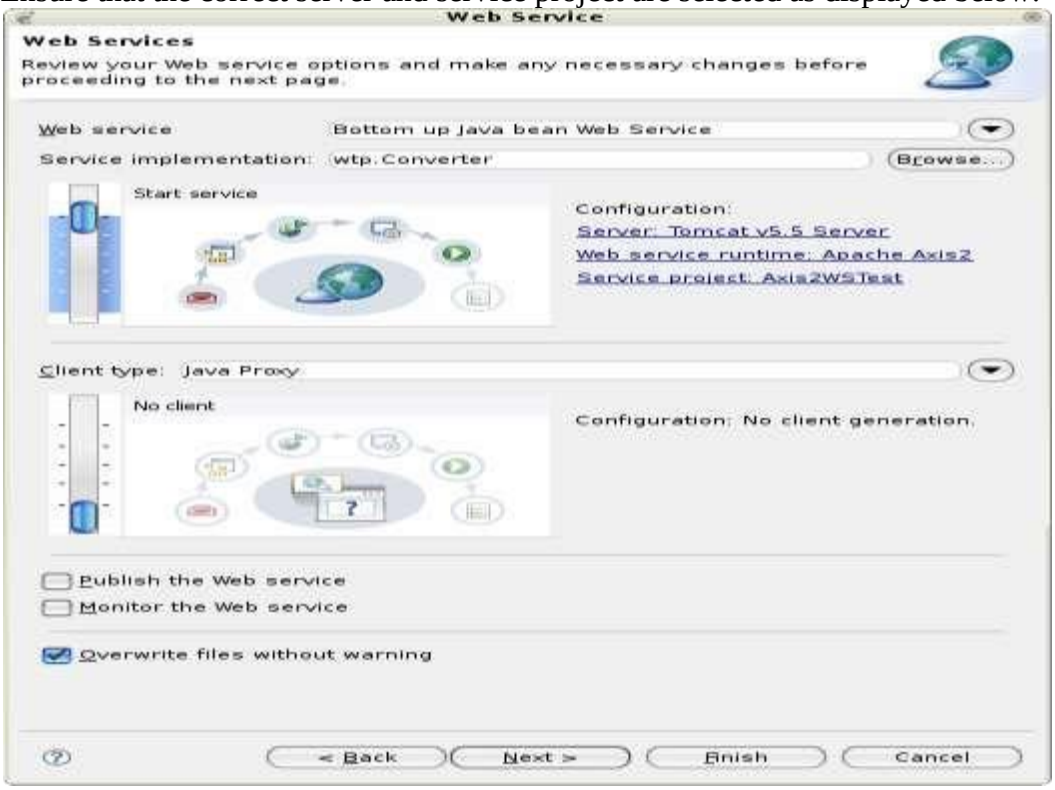


j. Click on the **Web Service runtime** link to select the Axis2 runtime.



Click OK.

k. Ensure that the correct server and service project are selected as displayed below.



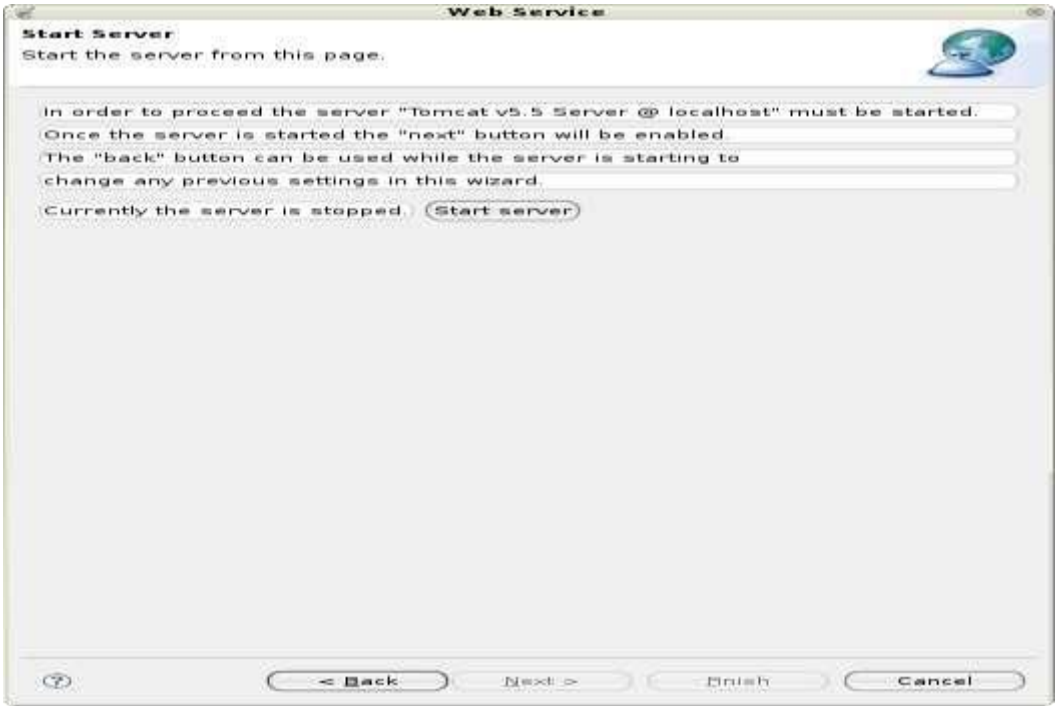
Click next.

l. This page is the service.xml selection page. if you have a custom services.xml, you can include that by clicking the **Browse** button. For the moment, just leave it

at the default.

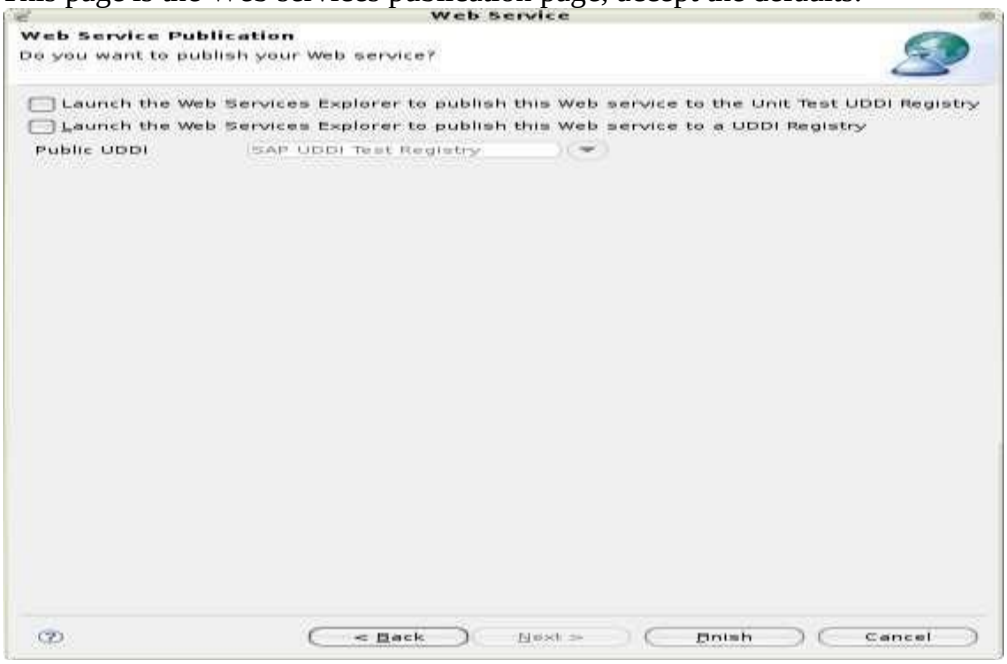
Click next.

m. This page is the Start Server page. It will be displayed if the server has not been started. Click on the **Start Server** button. This will start the server runtime.



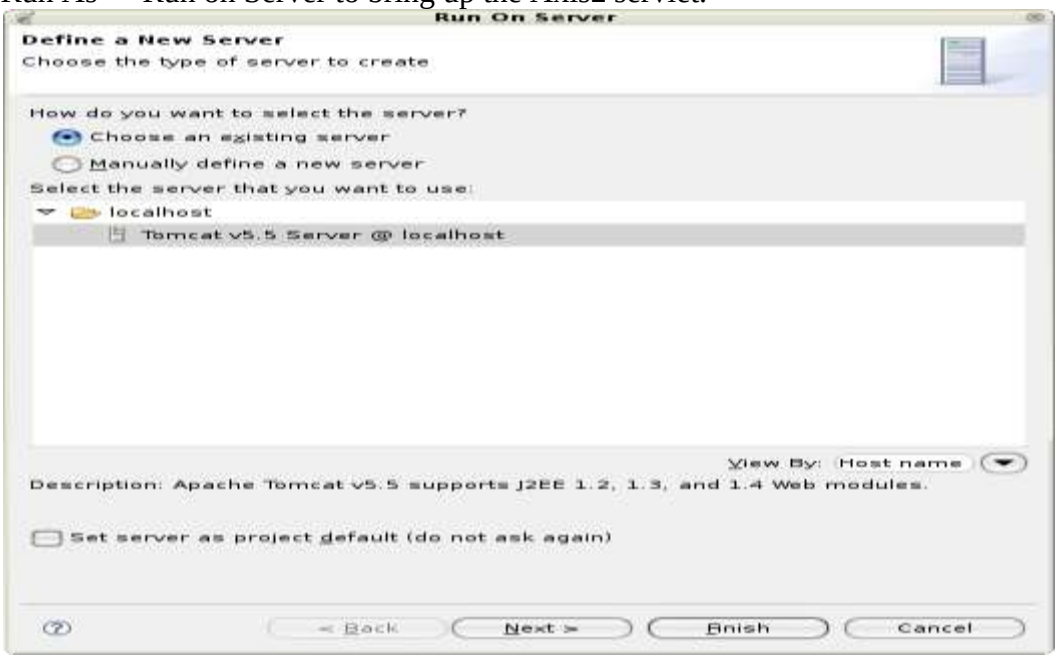
Click next.

n. This page is the Web services publication page, accept the defaults.



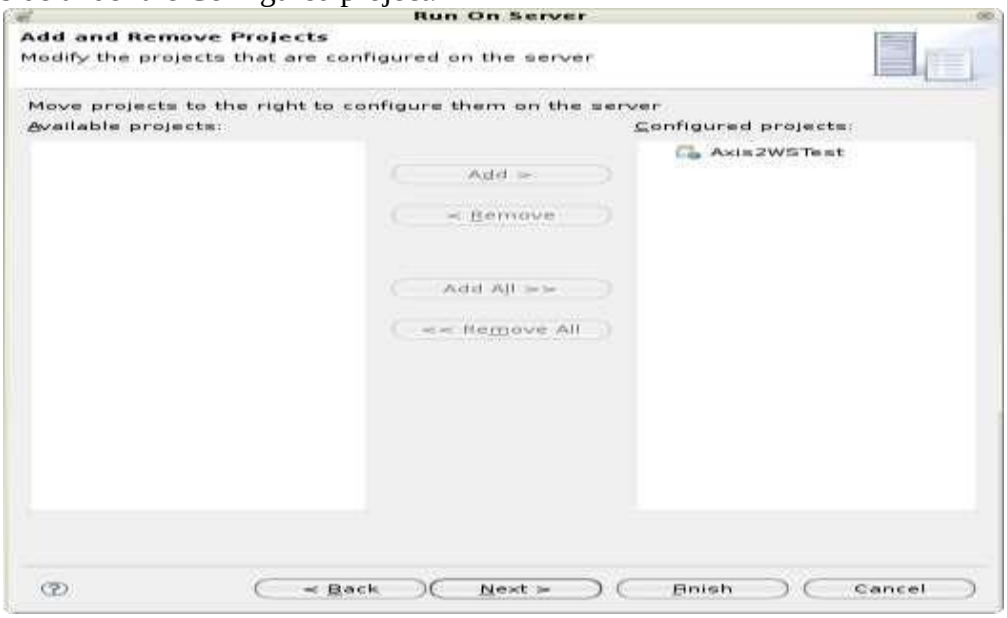
Click Finish.

o. Now, select the **Axis2WSTest** dynamic Web project, right-click and select Run -> Run As -> Run on Server to bring up the Axis2 servlet.



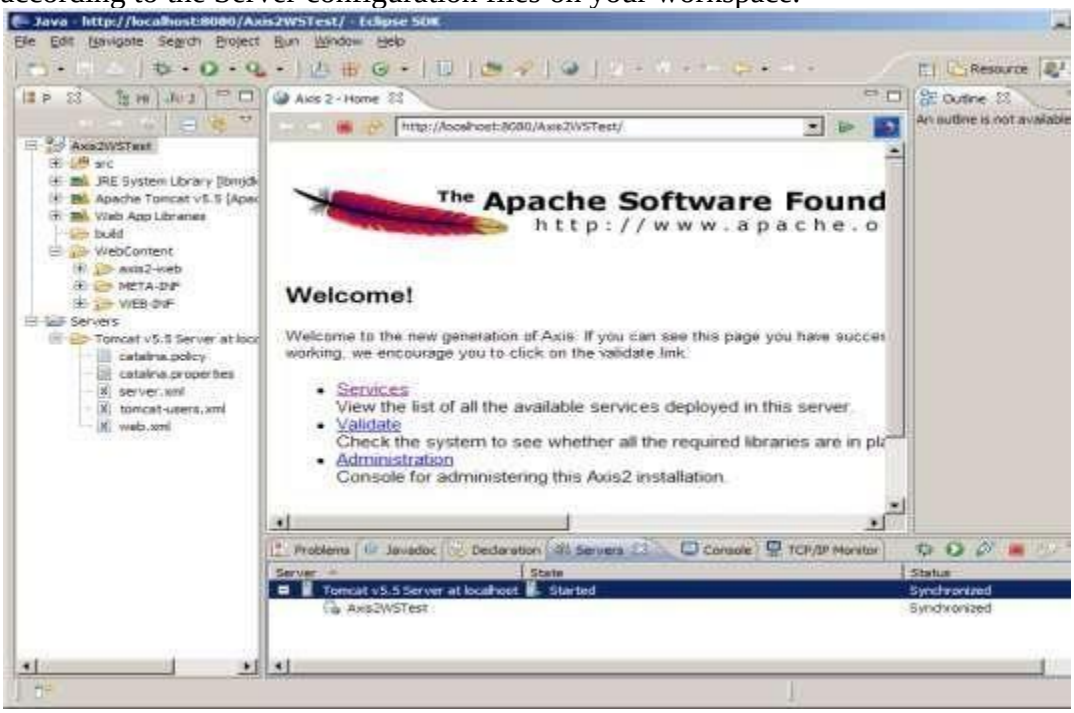
Click Next.

p. Make sure you have the **Axis2WSTest** dynamic Web project on the right-hand side under the Configured project.

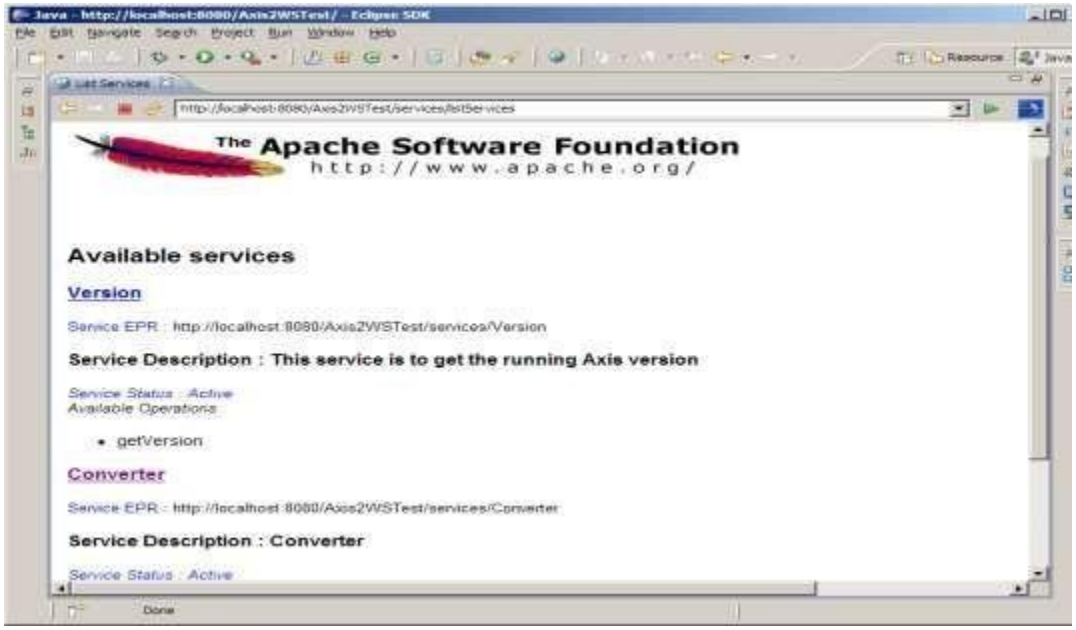


Click Finish.

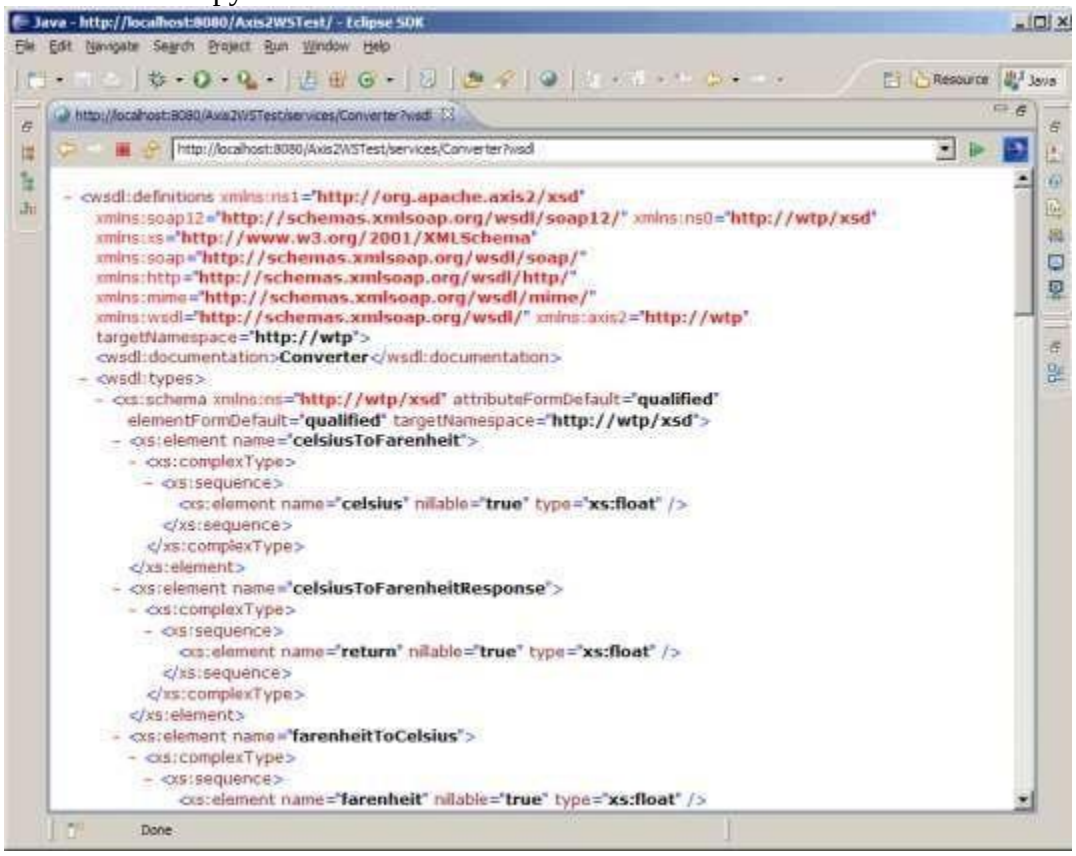
q. This will deploy the Axis2 server webapp on the configured servlet container and will display the Axis2 home page. Note that the servlet container will start up according to the Server configuration files on your workspace.



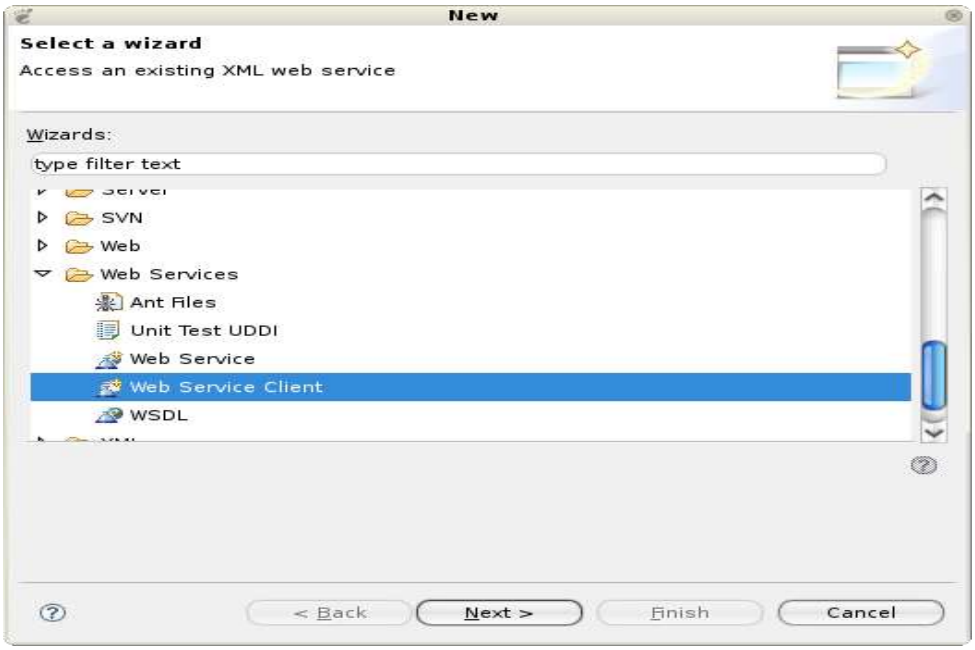
r. Click on the **Services** link to view the available services. The newly created converter Web service will be shown there.



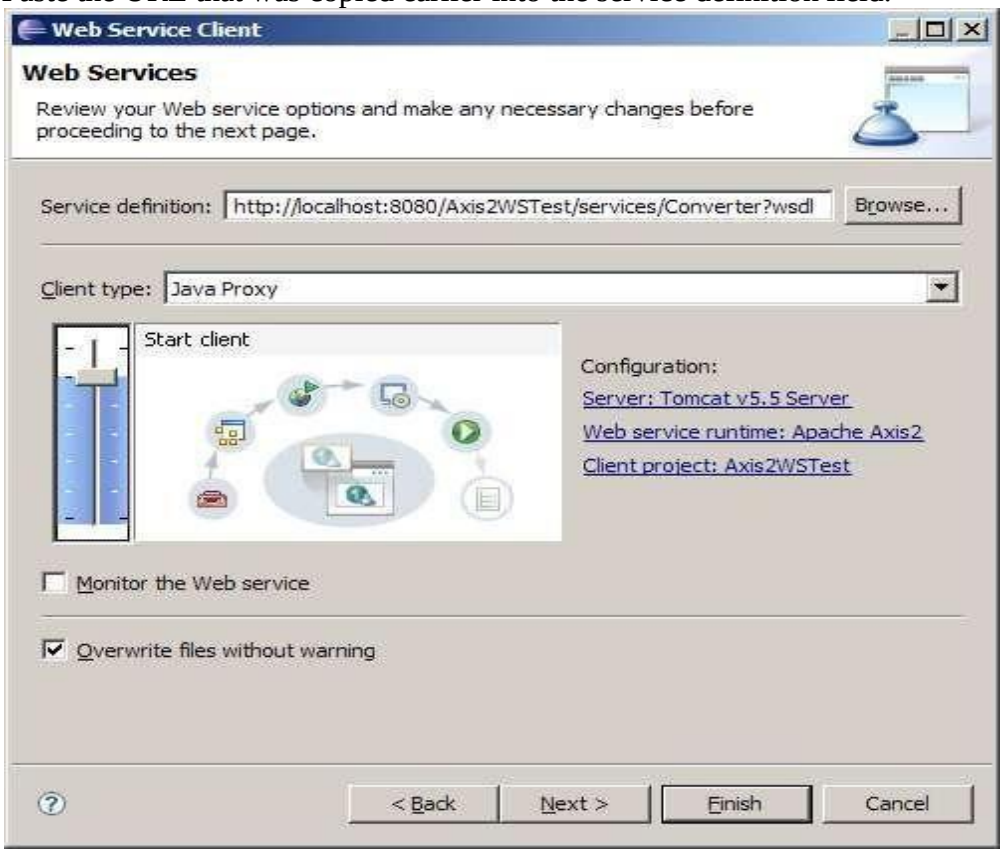
s. Click on the **Converter Service** link to display the wsdl URL of the newly created Web service. Copy the URL.



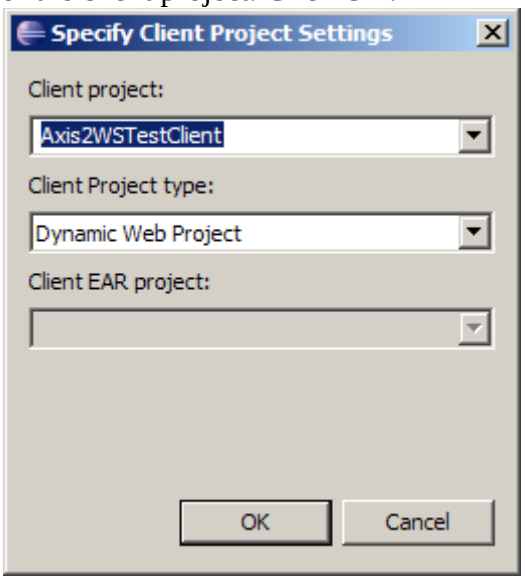
t. Now we'll generate the client for the newly created service by referring the ?wsdl generated by the Axis2 Server. Open File -> New -> Other... -> Web Services -> Web ServiceClient



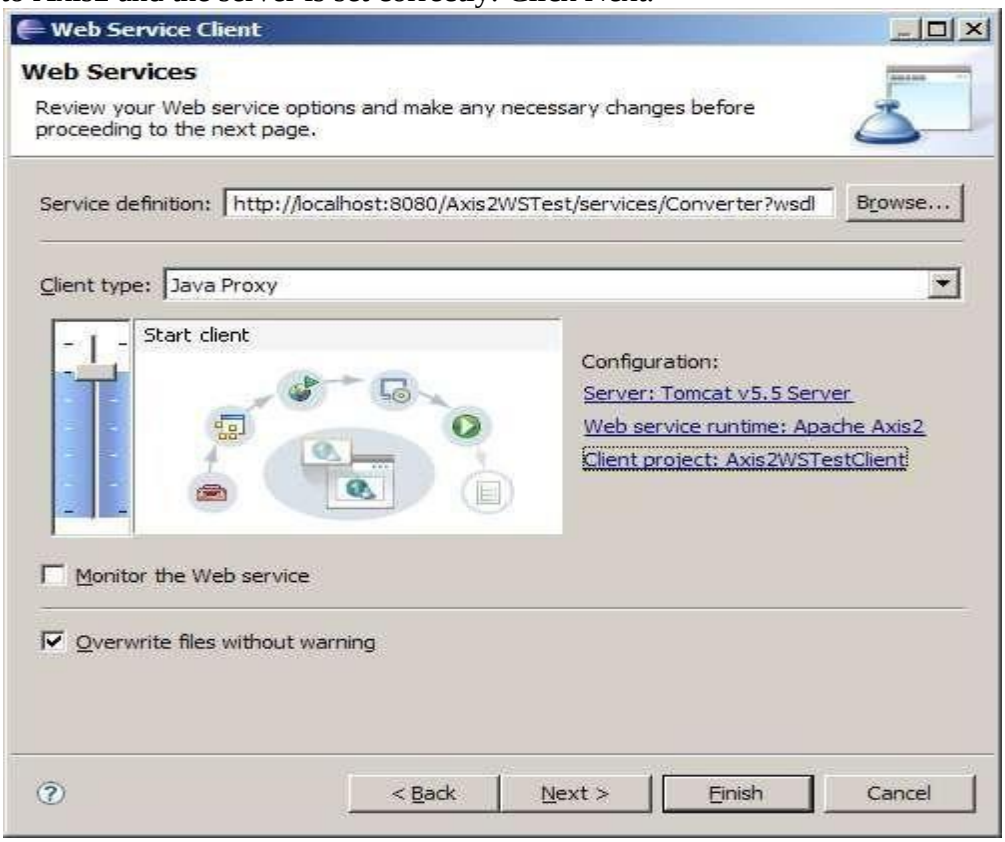
u. Paste the URL that was copied earlier into the service definition field.



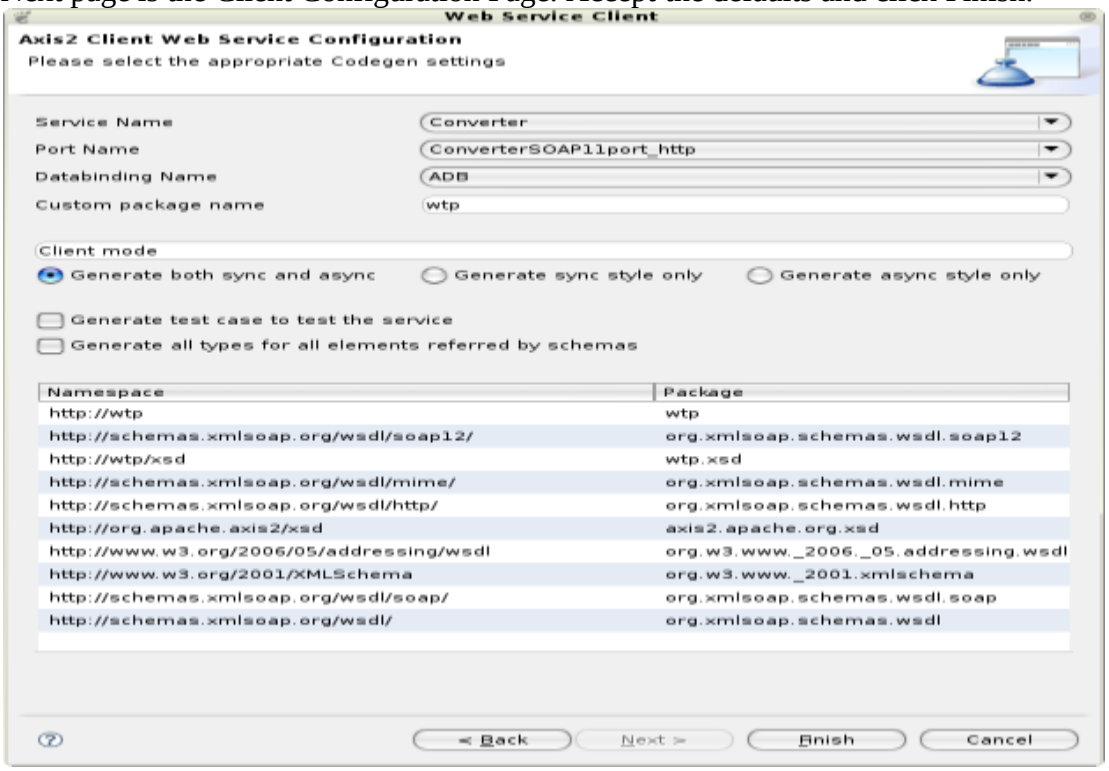
v. Click on the **Client project** hyperlink and enter **Axis2WSTestClient** as the name of the client project. Click OK.



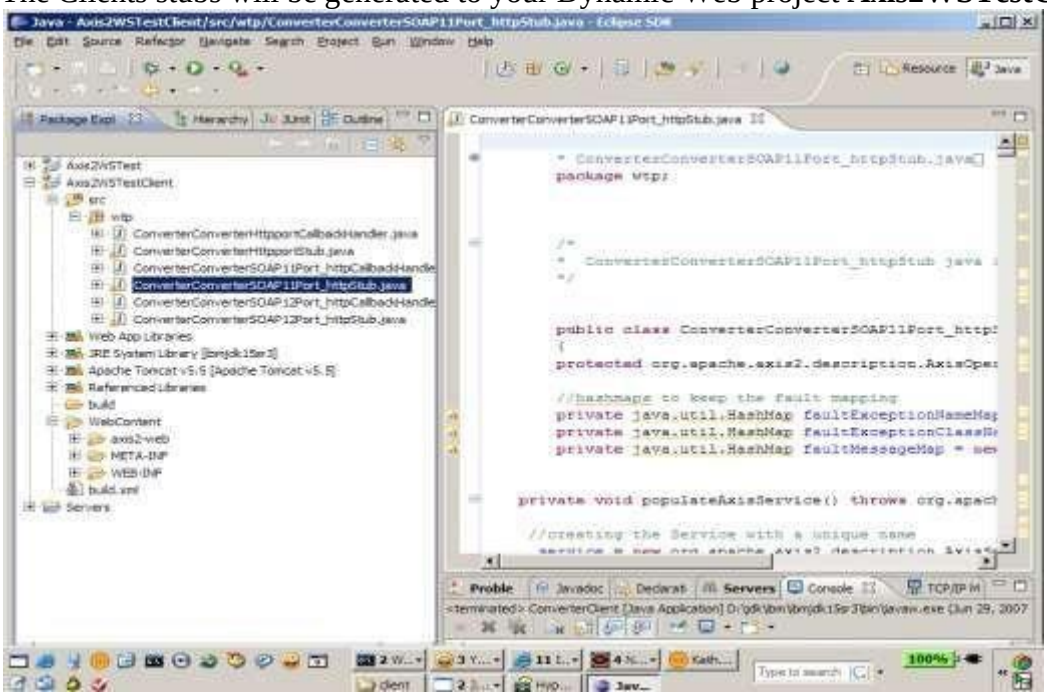
Back on the Web Services Client wizard, make sure the Web service runtime is set to Axis2 and the server is set correctly. Click Next.



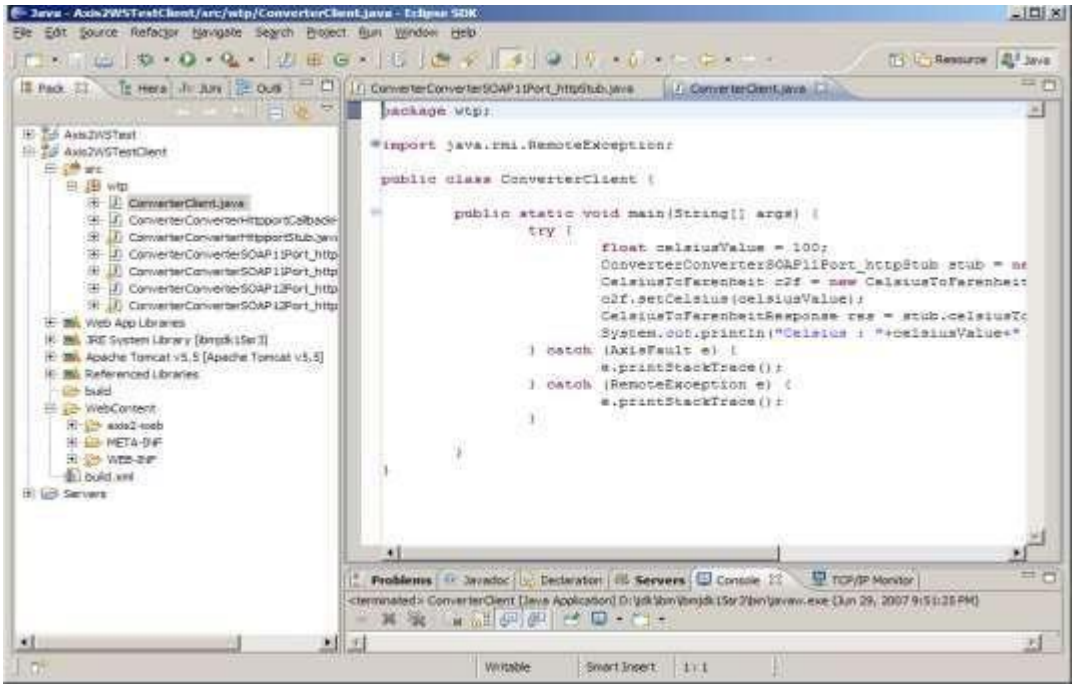
Next page is the Client Configuration Page. Accept the defaults and click Finish.



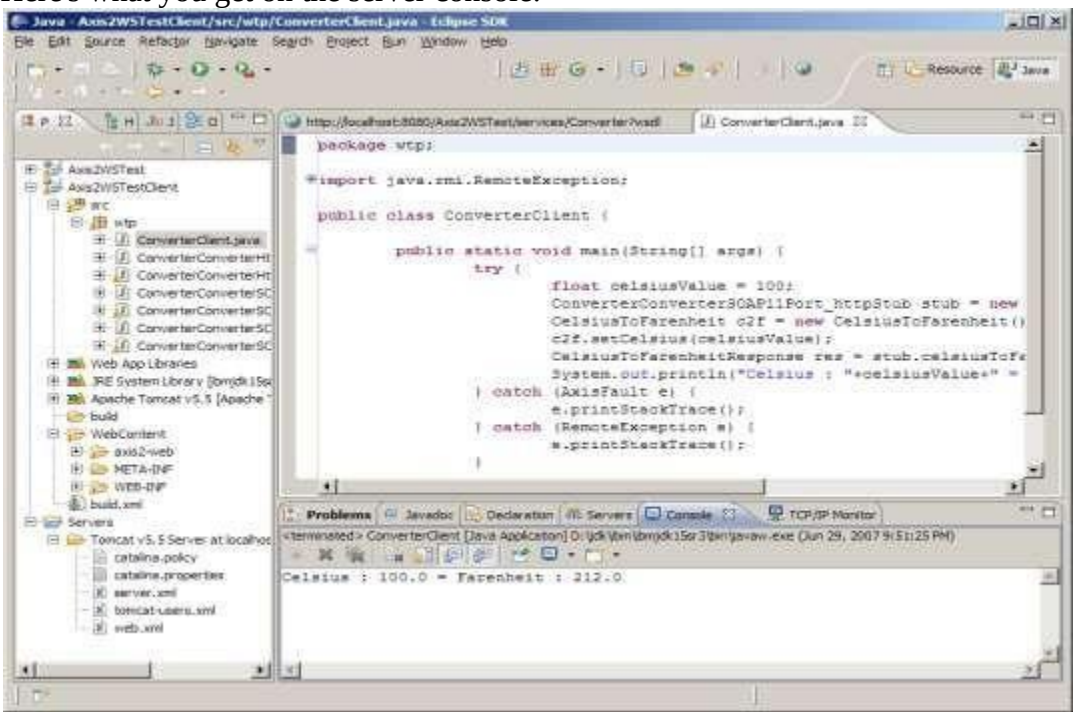
The Clients stubs will be generated to your Dynamic Web project **Axis2WSTestClient**.



Now we are going to write Java main program to invoke the client stub. Import the ConverterClient.java file to the workspace into the wtp package in the src folder of **Axis2WSTestClient**.

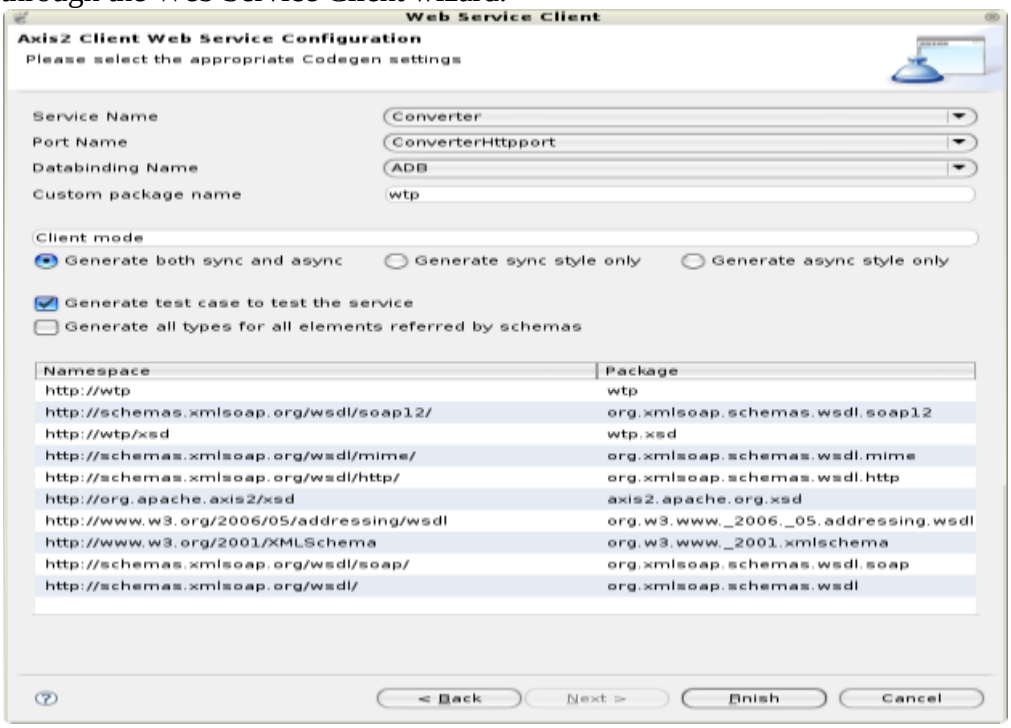


Then select the ConverterClient file, right-click and select Run As -> Java Application. Here's what you get on the server console:

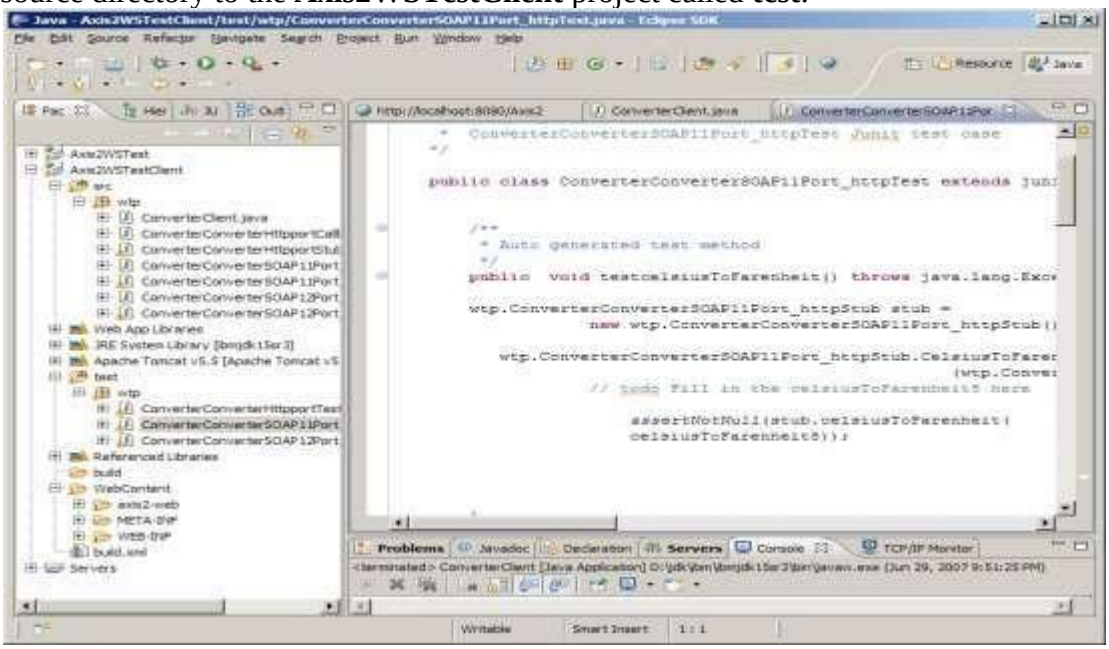


Another way to test and invoke the service is to select **Generate test case to test the service** check box on the Axis2 Client Web Service Configuration Page when going

through the Web Service Client wizard.



If that option is selected, the Axis2 emitter will generate JUnit testcases matching the WSDL we provide to the client. These JUnit testcases will be generated to a newly added source directory to the **Axis2WSTestClient** project called **test**.



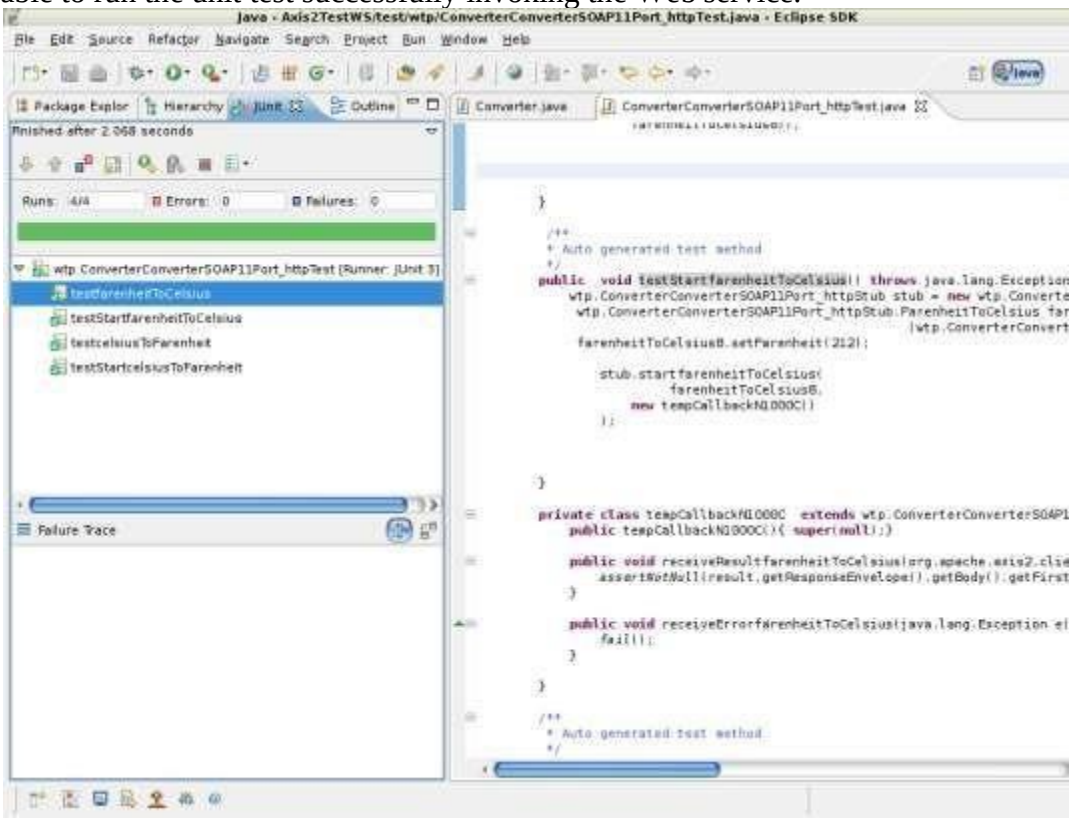
Next thing we need to do is to insert the test case with the valid inputs as the Web service method arguments. In this case, let's test the ConverterConverterSOAP11Port_httpTest.java by provide values for Celsius and Farenheit for the temperature conversion. As an example, replace the generated TODO statement in each test method to fill in the data with values as:

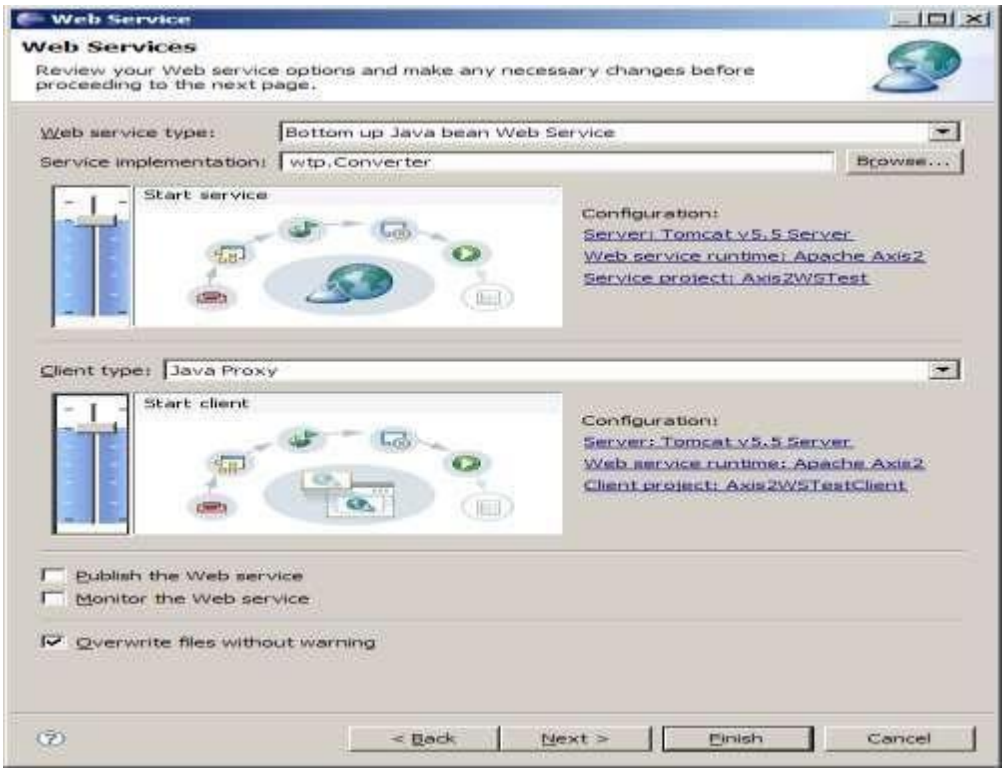
```
testfarenheitToCelsius() -> farenheitToCelsius8.setFarenheit(212);
```

```
     testStartfarenheitToCelsius() -
>farenheitToCelsius8.setFarenheit(212);
     testcelsiusToFarenheit() -> celsiusToFarenheit10.setCelsius(100);
     testStartcelsiusToFarenheit() ->
celsiusToFarenheit10.setCelsius(100);
```

Here the testcases were generated to test both the synchronous and asynchronous clients.

w. After that, select the testcase, right-click, select Run As -> JUnit Test. You will be able to run the unit test successfully invoking the Web service.





The Web Service wizard orchestrates the end-to-end generation, assembly, deployment, installation and execution of the Web service and Web service client. Now that your Web service is running, there are a few interesting things you can do with this WSDL file. Examples:

- You can choose Web Services -> Test with Web Services Explorer to test the service.
- You can choose Web Services -> Publish WSDL file to publish the service to a public UDDI registry.

RESULT:

Thus the development of a new OGSA-compliant web service was executed successfully.

3.Using Apache Axis develop a Grid Service

OBJECTIVE:

To develop a Grid Service using Apache Axis.

PROCEDURE:

You will need to download and install the following software:

1. Java 2 SDK v1.4.1, http://java.sun.com/j2se/1.4.1/download.html
2. Apache Tomcat v4.124
   http://jakarta.apache.org/builds/jakarta-tomcat-4.0/release/v4.1.24/bin/jakarta tomcat4.1.24.exe.
3. XML Security v1.0.4,
   http://www.apache.org/dist/xml/security/java-library/xmlsecurity bin1.0.4.zip
4. Axis v1.1, http://ws.apache.org/axis/dist/1_1/axis-1_1.zip

1. Java 2 SDK

- Run the downloaded executable (j2sdk-1_4_1-windows-i586.exe) which will install the
- SDK in C:\j2sdk1.4.1. Set the JAVA_HOME environment variable to point to this directory as follows:
- Click on START->CONTROL PANEL->SYSTEM
- Click on the Advanced tab
- Click on the Environment Variables button
- Click on the New… button in the user variable section and enter the details
- Add the Java binaries to your PATH variable in the same way by setting a user variable called PATH with the value "%PATH%;C:\j2sdk1.4.1\bin"

2. Apache Tomcat

- Run the downloaded executable (jakarta-tomcat-4.1.24.exe), and assume the installation directory is C:\jakarta-tomcat-4.1.24.
- Edit C:\ jakarta-tomcat-4.1.24\conf\tomcat-users.xml and create an "admin" and "manager" role as well as a user with both roles. The contents of the file should be similar to:

```
<?xml version='1.0' encoding='utf8'?>
<tomcat-users>
<role rolename="manager"/>
<role rolename="admin"/>
<user username="myuser" password="mypass" roles="admin,manager"/>
</tomcat-users>
```

- Start Tomcat by running C:\ jakarta-tomcat-4.1.24\bin\startup.bat and test it by browsing http://localhost:8080/
- Stop Tomcat by running C:\ jakarta-tomcat-4.1.24\bin\shutdown.bat.

3. XML Security

- Download and unzip
  http://www.apache.org/dist/xml/security/javalibrary/xmlsecurity-bin_1_0_4.zip
- Copy xml-sec.jar to C:\axis-1_1\lib\

- Set-up your CLASSPATH environment variable to including the following: C:\axis1_1\lib\xml-sec.jar;

4. Apache Axis

- Unzip the downloaded Axis archive to C: (this will create a directory C:\axis-1_1).
- Extract the file xmlsec.jar from the downloaded security archive to C:\axis1_1\webapps\axis\WEB-INF\lib.
- Set-up your CLASSPATH environment variable to including the following:
  - o The current working directory
  - o All the AXIS jar files as found in C:\axis-1_1\lib
    C:\jakarta-tomcat-4.1.24\common\lib\servlet.jar
- Your CLASSPATH should therefore look something like:
  C:\axis-1_1\lib\axis.jar;
  C:\axis 1_1\lib\axis-ant.jar;
  C:\axis-1_1\lib\commons-discovery.jar;
  C:\axis-1_1\lib\commons-logging.jar;
  C:\axis-1_1\lib\jaxrpc.jar;
  C:\axis-1_1\lib\log4j-1.2.8.jar;
  C:\axis-1_1\lib\saaj.jar;
  C:\axis-1_1\lib\wsdl4j.jar;
  C:\axis-1_1\lib\xercesImpl.jar
  C:\axis-1_1\lib\xmlParserAPIs.jar;
  C:\jakarta-tomcat-4.1.24\common\lib\servlet.jar
  C:\axis-1_1\lib\xml-sec.jar;
- Now tell Tomcat about your Axis web application by creating the file C:\jakarta- tomcat-4.1.24\webapps\axis.xml with the following content:
  <Context path="/axis" docBase="C:\axis-1_1\webapps\axis" debug="0" privileged="true">

  <LoggerclassName="org.apache.catalina.logger.FileLogger"prefix="axis_log." suffix=".txt" timestamp="false"/>

5. Deploy a Sample Web service packaged within Axis installations

Deploy one of the sample Web Services to test the system and to create the C:\axis-1_1\webapps\axis\WEB-INF\server-config.wsdd file. From C:\axis-1_1 issue the command (on one line):
java org.apache.axis.client.AdminClient
http://localhost:8080/axis/services/AdminService/samples/stock/deploy.wsdd
This should return the following:
.-Processing file samples/stock/deploy.wsdd
.- <Admin>Done processing</Admin>

RESULT:

Thus the development of a Grid Service using Apache Axis is executed successfully.

4. Develop applications using Java or C/C++ Grid APIs

OBJECTIVE:

To develop an applications using Java or C/C++ Grid APIs.

SAMPLE CODE:

```
import AgentTeamwork.Ateam.*;
import MPJ.*;
public class UserProgAteam extends AteamProg {
private int phase;
public UserProgAteam( Ateam o )
{ }
public UserProgAteam( )
{ }
// real const
public UserProgAteam( String[] args ) {
phase = 0;
}
// phase recovery
private void userRecovery( ) {
phase = ateam.getSnapshotId( );
}
private void compute( ) {
for ( phase = 0; phase < 10; phase++ ) {
try {
Thread.currentThread( ).sleep( 1000 );
}
catch(InterruptedException e ) {
}
ateam.takeSnapshot( phase );
System.out.println( "UserProgAteam at rank " + MPJ.COMM_WORLD.Rank( ) + " : took a
snapshot " + phase );
}
}
public static void main( String[] args ) {
System.out.println( "UserProgAteam: got started" );
MPJ.Init( args, ateam);
UserProgAteam program = null;
// Timer timer = new Timer( );
if ( ateam.isResumed( ) ) {
program = ( UserProgAteam )
ateam.retrieveLocalVar( "program" );
program.userRecovery( );
}
else
{
program = new UserProgAteam( args );
ateam.registerLocalVar( "program", program );
}
program.compute( );
MPJ.Finalize( );
}
```

```
public class UserProgAteam extends AteamProg {
// application body private void compute( ) {
for ( phase = 0; phase < 10; phase++ ) {
try {
Thread.currentThread( ).sleep( 1000 );
}
catch(InterruptedException e ) {
}
ateam.takeSnapshot( phase );
System.out.println ("UserProgAteam at rank " + MPJ.COMM_WORLD.Rank() + " : took a snapshot
" + phase );
}}
```

Socket sample code – within some function body

```
import AgentTeamwork.Ateam.GridTcp.*;
private final int port = 2000;
private GridSocket socket; private
GridServerSocket server; private InputStream
input; private OutputStream output;
for ( int i = start; i < start + trans; i++ ) {
try {
output.write( i % 128 );
} catch ( IOException e ) {
}
System.out.println ( "Sockets with " + myRank + ": " + " output[" + i + "]=" + i % 128 );
}
for ( int i = start; i < start + trans; i++ ) {
try {
System.out.println ( "Sockets with " + myRank + ": " + " input[" + i + "]=" + input.read( ) ); }
catch ( IOException e ) {
}}
```

MPI sample code

```
import AgentTeamwork.Ateam.*;
import MPI.*;
public class UserProgAteam extends AteamProg {
// application body private void compute( ) {
}
public static void main( String[] args ) {
MPJ.Init( args, ateam );
program.compute( ); MPJ.Finalize( );
}
}
```

C/C++ compile.sh – Helloworld.cpp

```
#!/bin/sh
rm -f *.class
javac -classpath MPJ.jar:Ateam.jar:. *.java
# jar cvf GridJNI.jar *.class jar -cvf
GridJNI.jar *.class javah -jni JavaToCpp
g++ -rdynamic JavaToCpp.cpp -o _libJavaToCpp.so_ -shared -ldl g++ -shared -o
```

libHelloWorld.so GridJNI_library.cpp
HelloWorld.cpp

C/C++ MPI sample code – Helloworld.cpp

```
#include <iostream.h>
using namespace std;
typedef int MPI_Request, MPI_Status, MPI_Comm;
extern void takeSnapshot(int argc);
extern int MPI_Init(int* argc, char*** argv);
extern void MPI_Finalize();
extern int MPI_Comm_rank(MPI_Comm comm, int *rank);
extern int MPI_Comm_size(MPI_Comm comm, int *size);
int main(int argc, char** argv) {
cerr << "main" << endl;
cerr << "argc = " << argc << endl;
cerr << "argv[0] = " << argv[0] << endl; cerr << "argv[1] = " <<
argv[1] << endl; MPI_Init(&argc,&argv);
cout << "MPI Init Successful!" << endl;
cout << "[HelloWorld.cpp]Calling Rank() and Size()" << endl;
int rank, size;
MPI_Comm_rank(0,&rank);
MPI_Comm_size(0,&size);
cout << "[HelloWorld.cpp]Rank = " << rank << endl;
cout << "[HelloWorld.cpp]Size = " << size << endl; cerr << "Calling
MPI_Finalize()" << endl; MPI_Finalize();
cerr << "finished" << endl;
}
```

RESULT:

Thus the development of applications using Java or C/C++ Grid APIs is executed successfully

CLOUD COMPUTING LABORATORY

VIVA QUESTIONS

1) What are the advantages of using cloud computing?

2) Mention platforms which are used for large scale cloud computing?

3) Different models for deployment in cloud computing

4) What is the difference in cloud computing and computing for mobiles?

5) How user can gain from utility computing?

6) For a transport in cloud how you can secure your data?

7) What are the security aspects provided with cloud?

8) List out different layers which define cloud architecture?

9) What are system integrators in Cloud Computing?

10) What is " EUCALYPTUS" stands for?

11) What is the use of "EUCALYPTUS" in cloud computing?

12) What is the requirement of virtualization platform in implementing cloud?

13) Before going for cloud computing platform what are the essential things to be taken in concern by users?

14) Mention some open source cloud computing platform databases?

15) What are the security laws which are implemented to secure data in a cloud ?

16) Mention the name of some large cloud providers and databases?

17) The difference between cloud and traditional datacenters?

18) What are the different modes of software as a service (SaaS)?

19) What is the use of API's in cloud services?

20) What are the different data centers deployed for cloud computing?

21) In cloud computing what are the different layers?

22) How important is the platform as a service?

23) What is a cloud service?

24) What is Grid Computing ?

25) History of Grid , Cluster , grid , Distributed Computing

26) Why is ir given the name 'Grid'?

27) Which are the companies suporting, working, and doing researching on the Grid ?

28) Which are the Academic Institutions interested and doing research on Grid ?

29) What are the terminologies that different companies use for the same term "Grid" ?

30) What are the different types of computing on the Grid ? (HPC , HTC)

31)What is the use of Grid in general ?

32) Why do we need Grid ? What are the applications of the Grid that one can explain to a naive person ?

33) What is a Virtual Organizaton ?

34) VO of a cluster , VO of Grid

35) What are the middlewares (Glues) used to build a Grid ?

36) Who are the commercial players in building a Grid ?

37) Who are the open source players in building a Grid ?

38) Which is the most used grid middleware ?

39) What are Grid Test beds ? Who started it ? Who made it ?

40) Which are the existing Grid Test Beds in the world ?