

DevOps Activity 1 – Part 2

Created by Steve 13-April-2018

In this version of the document, we will continue from Activity 1 – Part 1, and install two tools, VirtualBox and Vagrant. Then we will create a local VM using a **Vagrantfile** to show how we can easily with code, create a local Linux Machine in a standard way.

DevOps Activity 1 – Part 2.....	1
Activity 1 -Part 2.....	1
What is VirtualBox	2
Installing VirtualBox	2
Download VirtualBox Installer	2
Installing Vagrant	7
Download Vagrant installer	8
Run/Open the Vagrant installer	9
Verify Vagrant is working	14
Creating an Ubuntu Linux Machine using Vagrant	16
Installing a localised Linux (Ubuntu) VM using Vagrant	16
vagrant init.....	17
Vagrant up.....	17
Vagrant status.....	18
Vagrant ssh.....	19
Stopping and Starting VMs using vagrant.....	21
Start a VM using vagrant.....	21
Stop a VM using Vagrant.....	24
Using the VirtualBox GUI.	25
Stop a VM.....	25
To Start a stopped VM	26
Summary	26

Activity 1 -Part 2

These screen capture examples are using Chrome, so this is what the download looks like in Chrome...If you are using Microsoft Edge or Internet Explorer or even Firefox this may look slightly

different. I would expect nowadays that almost everyone has downloaded and installed software on a Windows Desktop.

This document presumes you have followed [DevOps Activity 1 – Part 1](#) guide and you have [Notepad++](#) installed, and [Git for Windows](#) installed and configured correctly.

You have also cloned a [Git](#) repo from online as instructed in Part 1 to get this guide.

In this guide, we will install and use the following

- Oracle VirtualBox
- Vagrant

What is VirtualBox

VirtualBox is an open source tool provided by Oracle that allows a user to quickly create and manage localised virtual machines on your desktop for all sorts of purposes.

Installing VirtualBox

We will begin the installation, by locating, and getting an appropriate download.

Download VirtualBox Installer

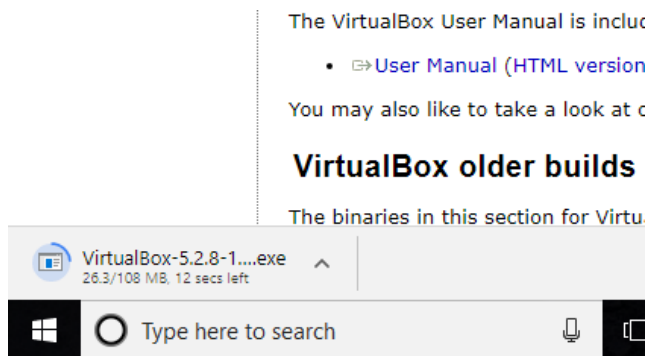
Using a browser, download the VirtualBox installer from the following URL

<https://www.virtualbox.org/wiki/Downloads>

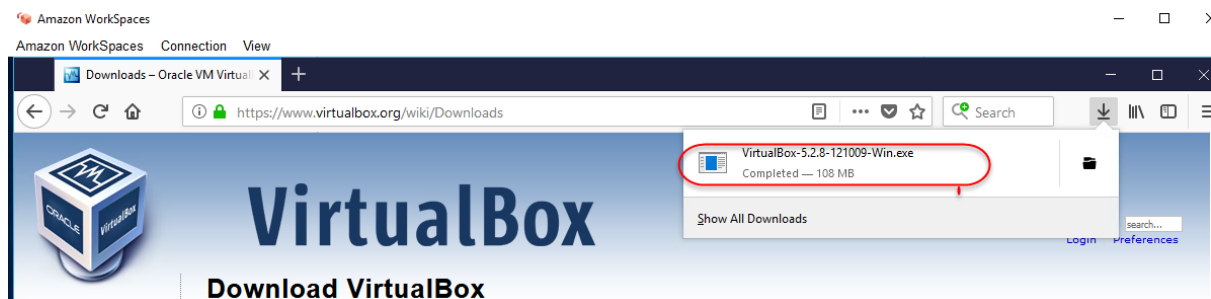


Select [Windows hosts](#) and click to download the installer.

On windows 10 we will see the following download

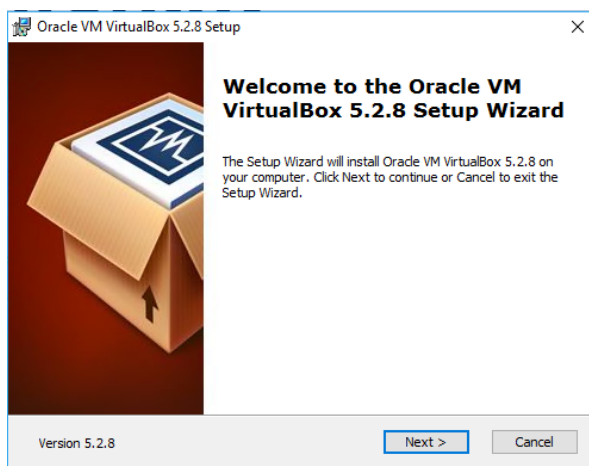
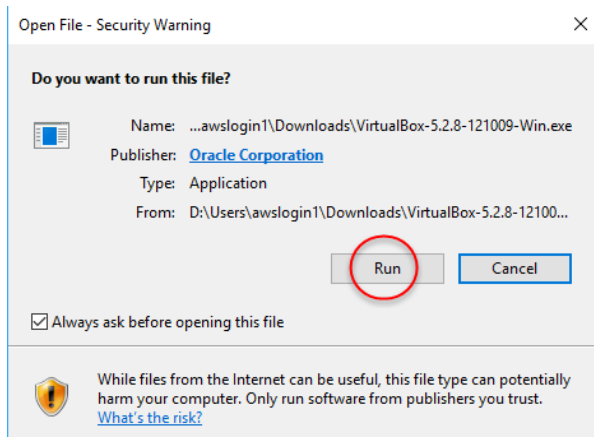


Note: Depending on the browser you use, it may look like...

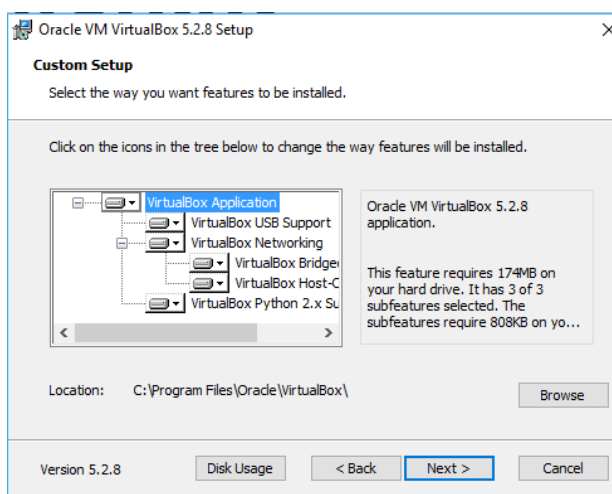


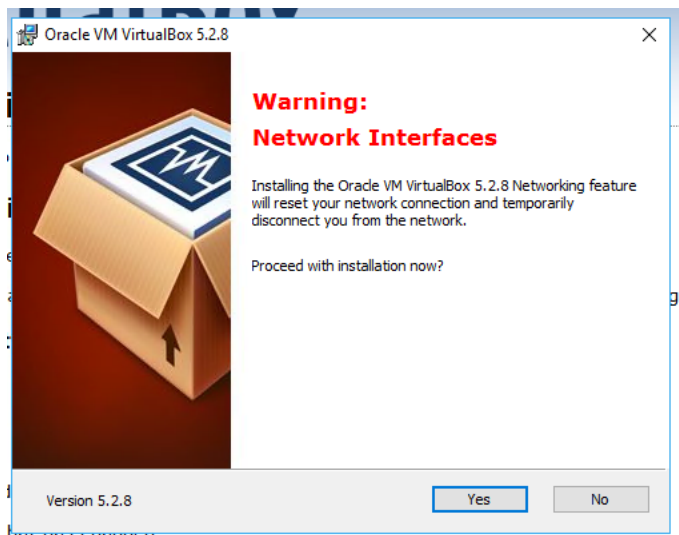
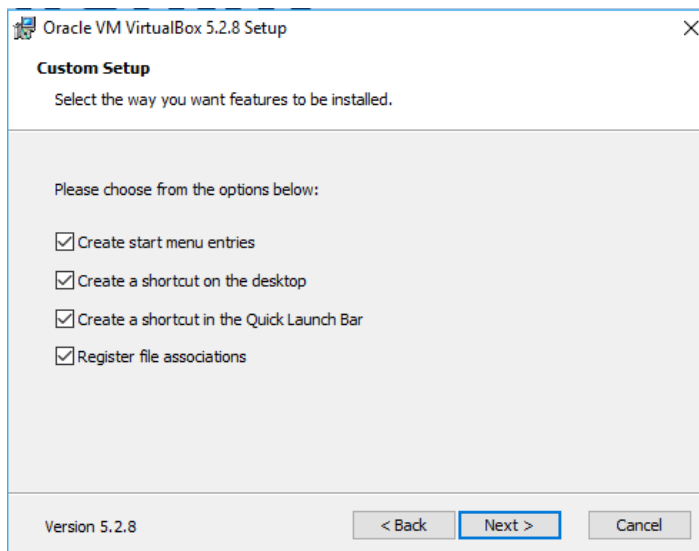
Launch the VirtualBox Installer (MSI)

If you get prompted for Admin Privileges, then acknowledge, and Run

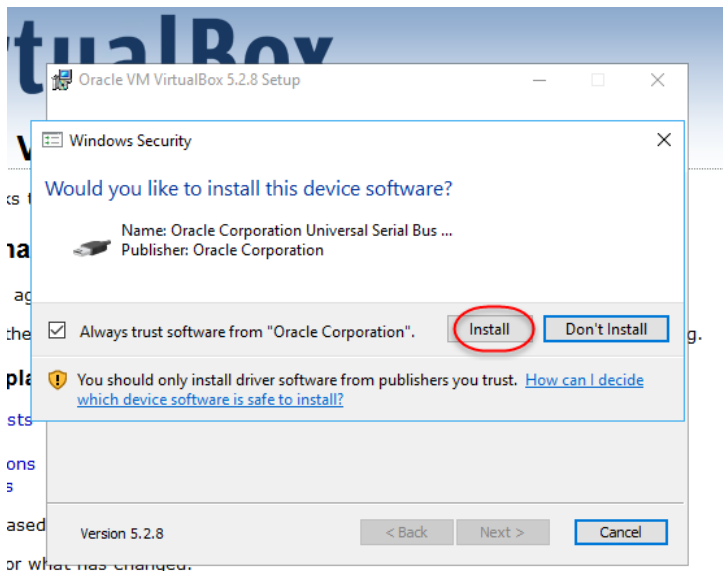
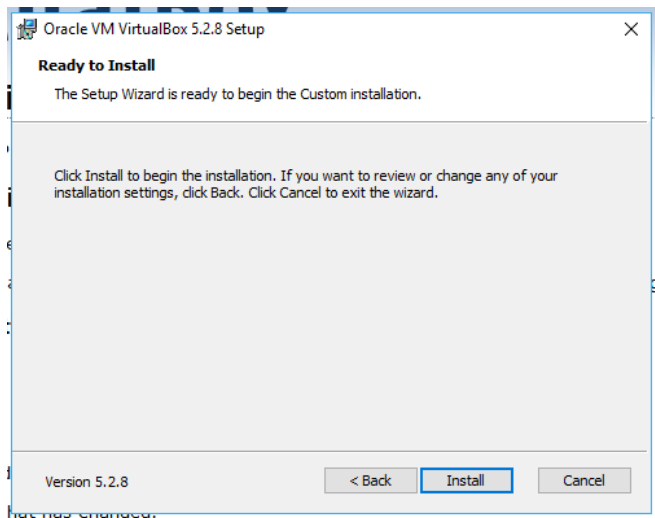


Follow the default steps i.e. **Next**, **Next** etc as shown in the following screen shots.

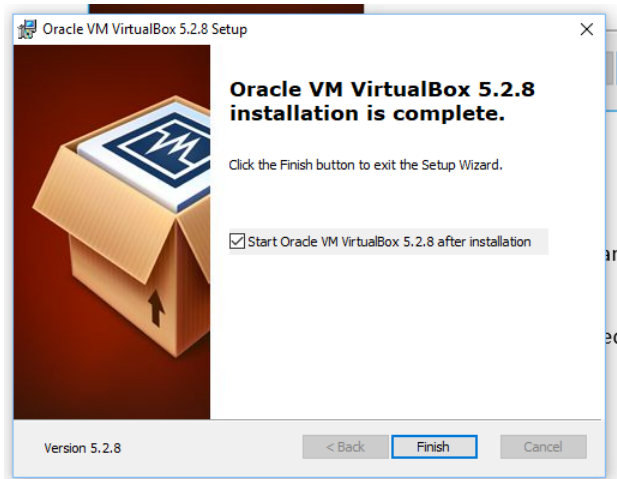




Click **Yes** to install the **Network Interface** Libraries



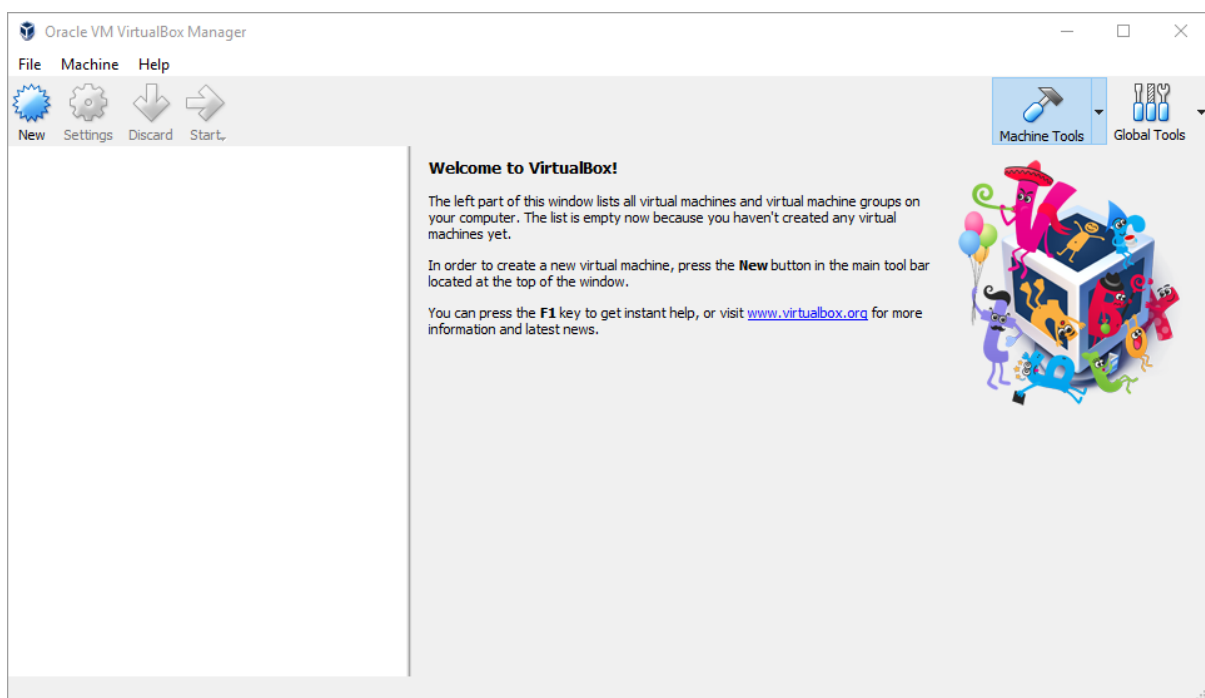
Click **Install** to confirm that you trust Oracle



We will allow the installer to Launch Virtual Box, to check that it has installed with no errors.

Click **Finish**

This is what the VirtualBox GUI looks like.



We are not going to use the VirtualBox GUI on this course but be aware that you can create local VMs on your Windows desktop using wizards and downloaded CDROM images.

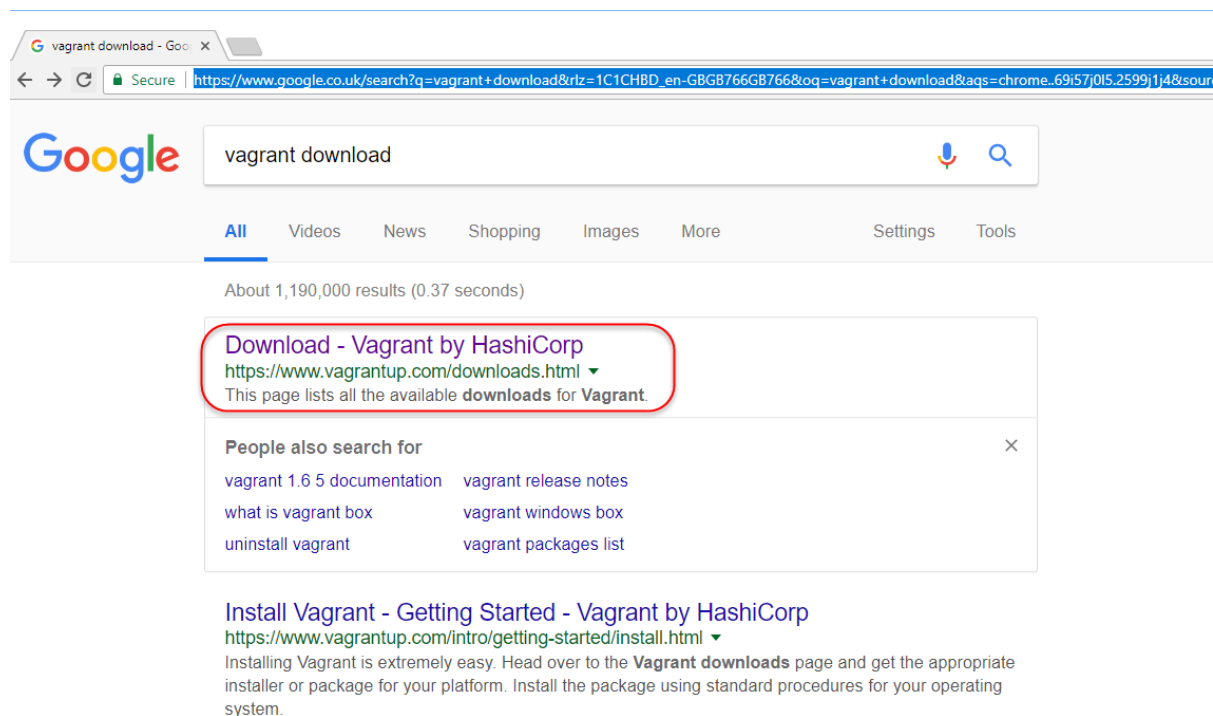
Exit/Close the VirtualBox GUI

Installing Vagrant

Next, we are going to install a command-line tools called **Vagrant** and it will need to locate Virtual Box libraries so it can manage VMs, hence the reason why we installed VirtualBox.

Download Vagrant installer

Do a Google search and locate the HashiCorp Vagrant Download page



You should get to this page (URL)

<https://www.vagrantup.com/downloads.html>

Note: Due to issues with Vagrant and Windows 10 and PowerShell conflicts we cannot use the latest version of Vagrant unless we patch with many complicated steps, this is too difficult to demonstrate at this time, so we will install Vagrant Version 2.0.2 as of 14-April-2018

Download Vagrant

Below are the available downloads for the latest version of Vagrant (2.0.3). Please download the proper package for your operating system and architecture.

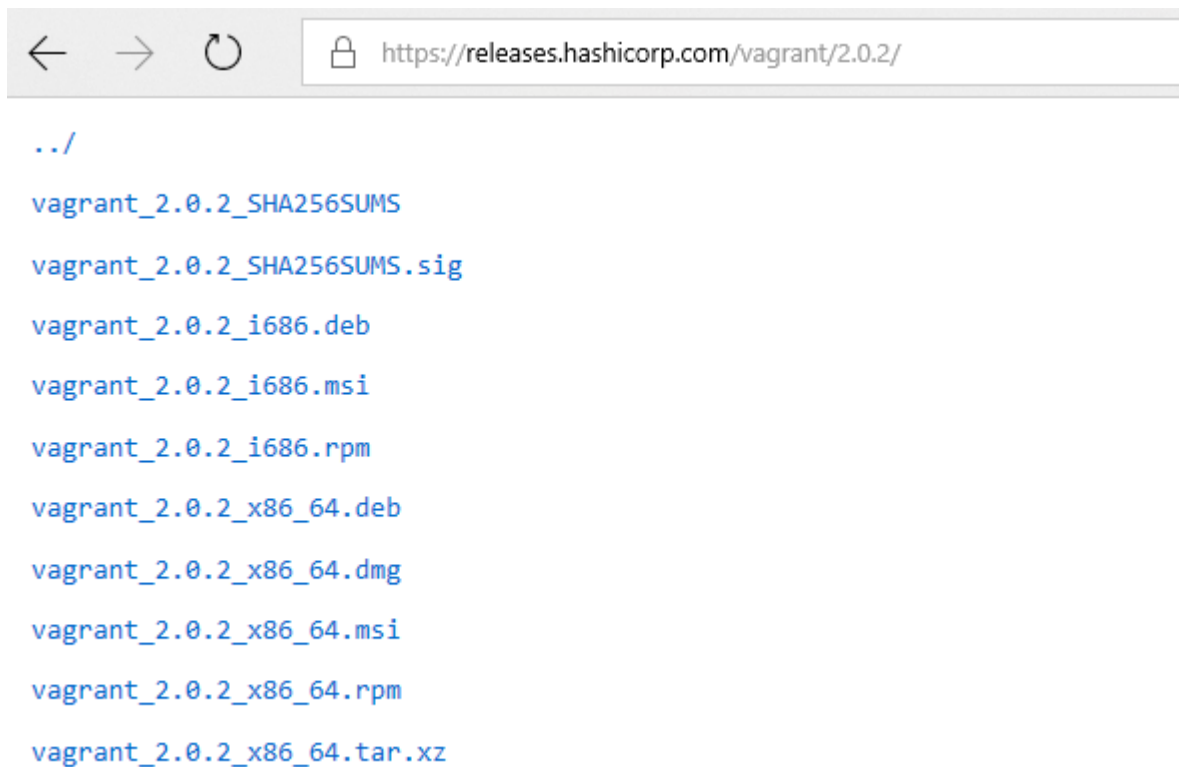
You can find the [SHA256 checksums for Vagrant 2.0.3](#) online and you can [verify the checksum's signature file](#), which has been signed using [HashiCorp's GPG key](#). You can also [download older versions of Vagrant](#) from the releases service.

Click [download older version of Vagrant](#)

Click the 64-bit download, unless your local desktop is 32bit, usually nowadays you use 64bit downloads.

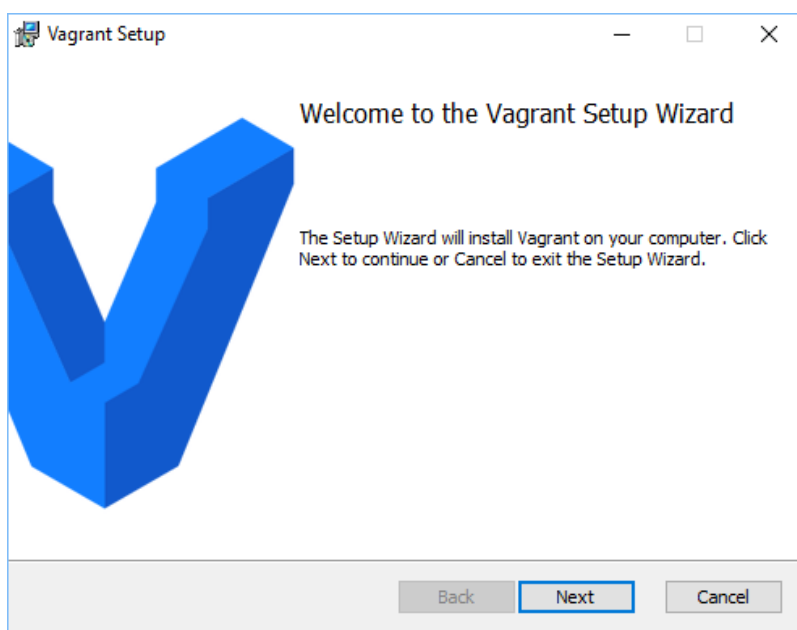
Locate Version 2.0.2

<https://releases.hashicorp.com/vagrant/2.0.2/>

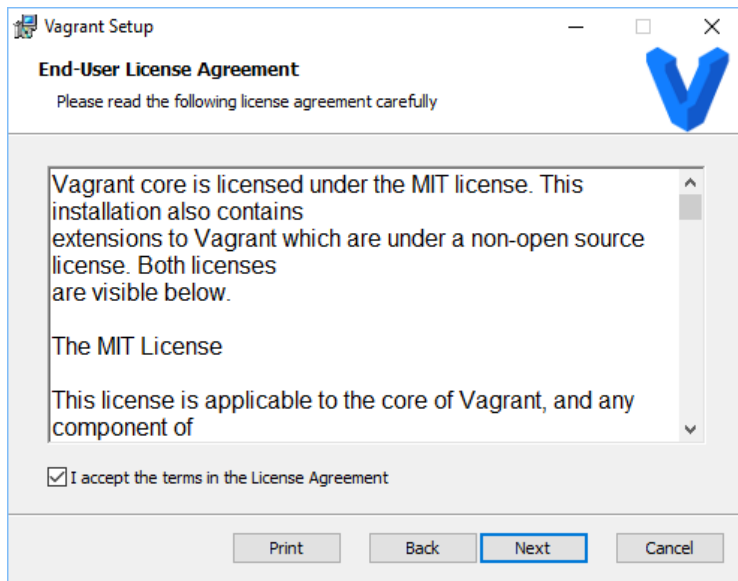


Download the [vagrant_2.0.2_x86_64.msi](#) file as seen above.

Run/Open the Vagrant installer

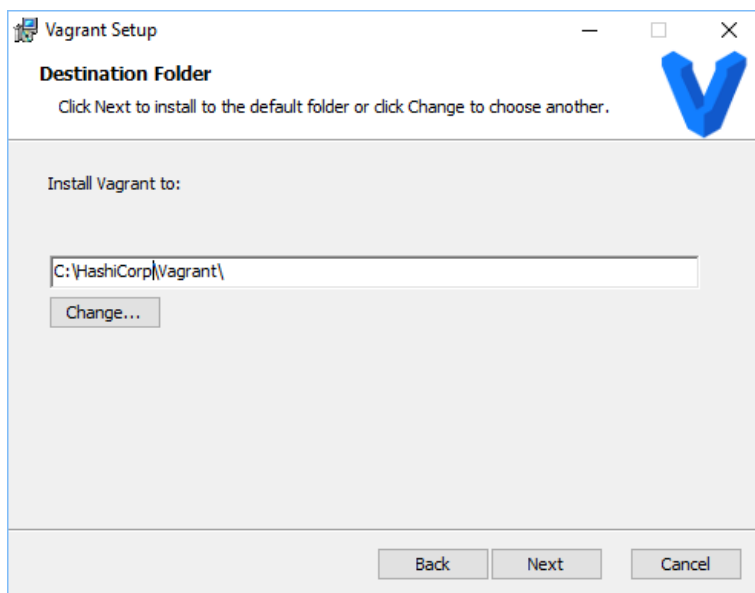


Click **Next**

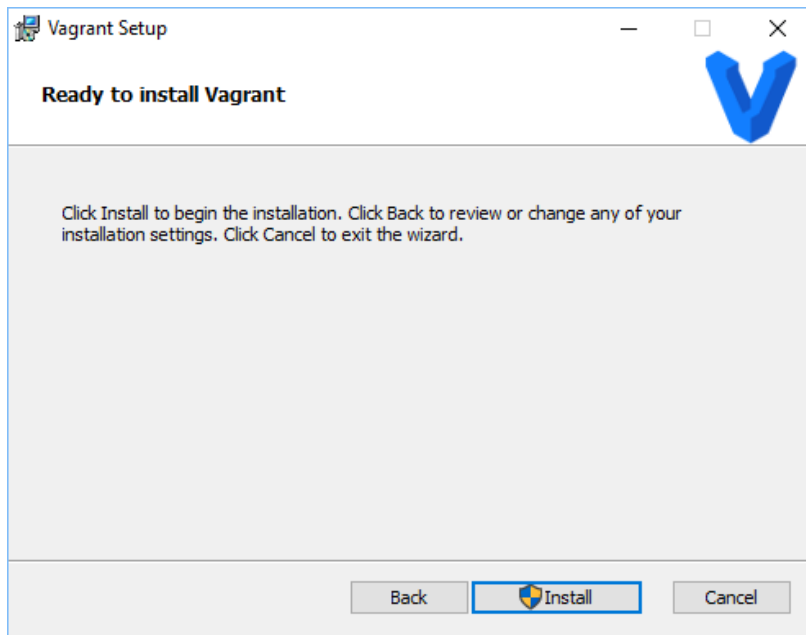


Click Next to accept the license

We will leave the default installer location as the default

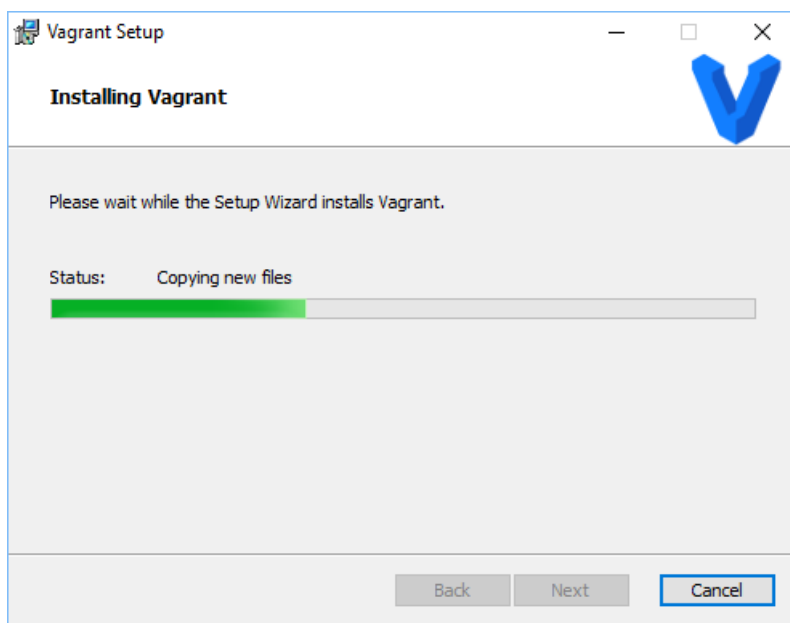


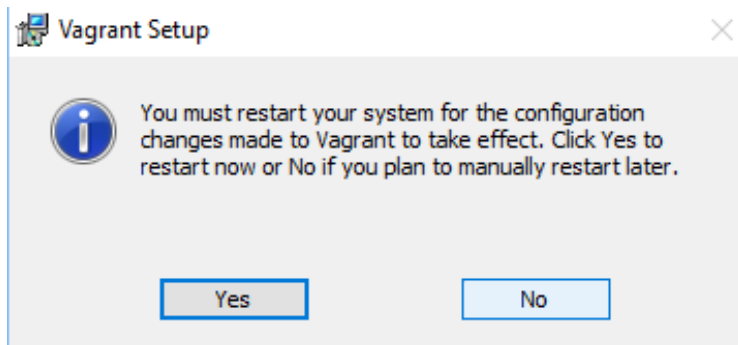
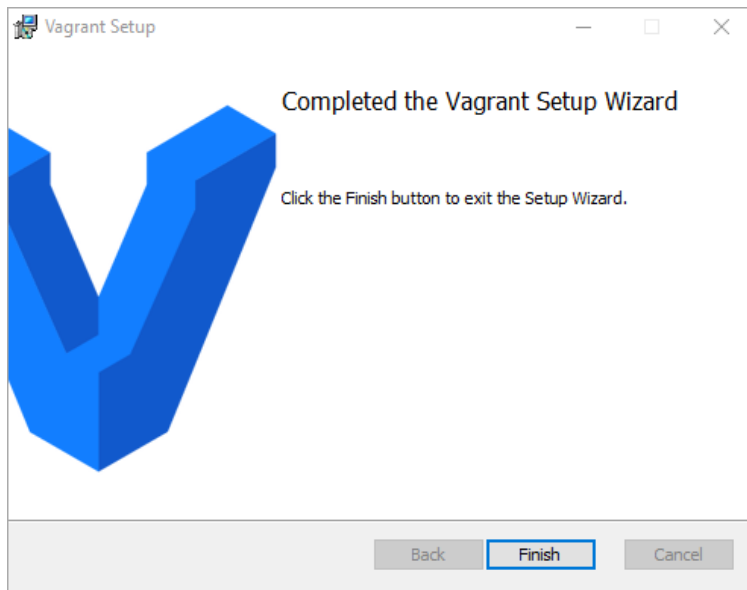
Click **Next**



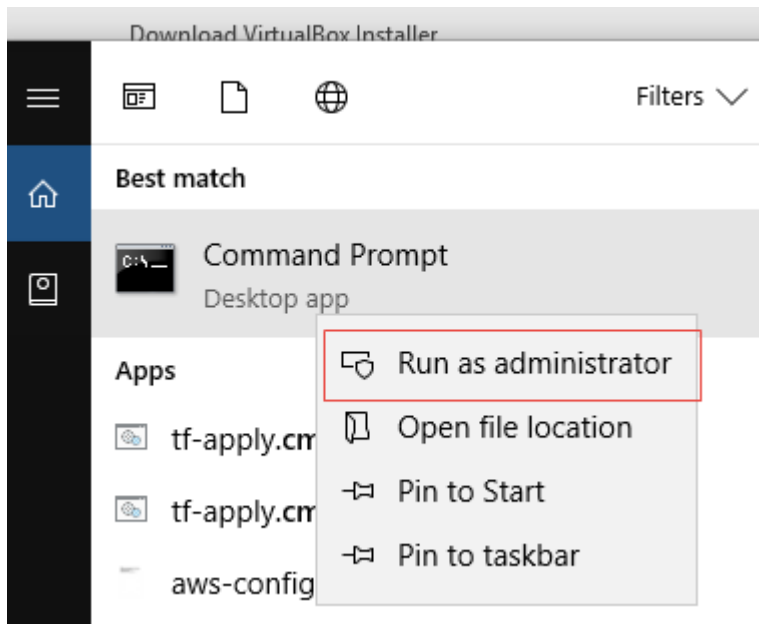
Click **Install**

Accept any Administrator privileges prompts and continue





Before we restart we need to Disable Hyper-V by running the following command: in an Administration CMD prompt



Type the following command to check the Windows 10 Hyper V Launch Type. We do not want Auto.

`bcdedit`

```
Administrator: Command Prompt
isolatedcontext      Yes
fverecoverysurl      https://account.activedirectory.windowsazure.com/n/#/devices
default              {current}
resumeobject         {93bdeb4f-5a28-11e7-a374-bc8385e8d1a1}
displayorder         {current}
toolsdisplayorder    {memdiag}
timeout              30

Windows Boot Loader
-----
identifier            {current}
device                partition=C:
path                  \WINDOWS\system32\winload.efi
description            Windows 10
locale                en-GB
inherit                {bootloadersettings}
recoverysequence      {d1090286-bfc3-11e7-a670-a940aef7479b}
displaymessageoverride Recovery
recoveryenabled        Yes
badmemoryaccess       Yes
isolatedcontext        Yes
allowedinmemorysettings 0x15000075
osdevice              partition=C:
systemroot             \WINDOWS
resumeobject          {93bdeb4f-5a28-11e7-a374-bc8385e8d1a1}
nx                    OptIn
bootmenupolicy         Standard
hypervisorlaunchtype  Auto

C:\WINDOWS\system32>
```

If you see `hypervisorlaunchtype = Auto`, then type

`bcdedit /set hypervisorlaunchtype off`

Type `bcdedit` again to verify changes

```
Administrator: Command Prompt
isolatedcontext      Yes
fvererecoveryurl     https://account.activedirectory.windowsazure.com/n/#/devices
default              {current}
resumeobject         {93bdeb4f-5a28-11e7-a374-bc8385e8d1a1}
displayorder         {current}
toolsdisplayorder    {memdiag}
timeout              30

Windows Boot Loader
-----
identifier           {current}
device                partition=C:
path                  \WINDOWS\system32\winload.efi
description           Windows 10
locale                en-GB
inherit               {bootloadersettings}
recoverysequence      {d1090286-bfc3-11e7-a670-a940aef7479b}
displaymessageoverride Recovery
recoveryenabled       Yes
badmemoryaccess       Yes
isolatedcontext       Yes
allowedinmemorysettings 0x15000075
osdevice              partition=C:
systemroot            \WINDOWS
resumeobject          {93bdeb4f-5a28-11e7-a374-bc8385e8d1a1}
nx                    OptIn
bootmenupolicy        Standard
hypervisorlaunchtype  Off

C:\WINDOWS\system32>
```

Now we can initiate a restart of the machine.

Please restart your machine. If we do this now there are less chances of installation issues later.

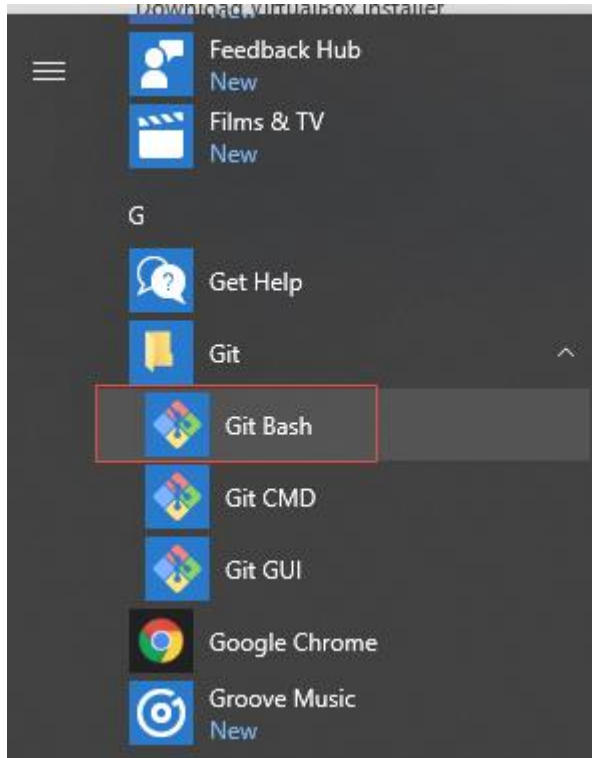
Wait for restart to occur...

Once the computer has restarted we will return to this guide and validate Vagrant.

Verify Vagrant is working

Note: In this course we do not want to run Git commands or Vagrant commands from within a Windows Command Prompt, please always check that you have opened a Git Bash terminal not a Windows command terminal. The reason of for this is that switching between command prompt can confuse for course learning due to conflicts between tools.

Open a [Git Bash](#) command prompt using your preferred method, for example the start menu



Please change to the repo location by issuing the following command:

```
cd ~/local-repos/devops-course-activity1
```

Once in the correct location, we will check the vagrant version by typing the following command:

```
vagrant --version
```

Result:

We can now prove that we have installed Vagrant, but we still need to validate that it will run correctly with the current operating system.

Type and run `vagrant validate` from within a Git Bash terminal

```
vagrant validate
```

Result:

```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant validate
Vagrantfile validated successfully.

Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$
```

If you see the following error message, then there is an issue with Vagrant version. Often a conflict. In this case for Windows 10, and other installed tools. Please double check the version of Vagrant you have downloaded.

The version of powershell currently installed on this host is less than the required minimum version. Please upgrade the installed version of powershell to the minimum required version and run the command again.

Installed version: N/A

Minimum required version: 3

Creating an Ubuntu Linux Machine using Vagrant

In our local-repo/devops-course-activity1 folder we have a folder called scripts.

We pulled down the repo (cloned) previously, but to ensure we have the latest updated versions, we can issue this command

```
git pull
```

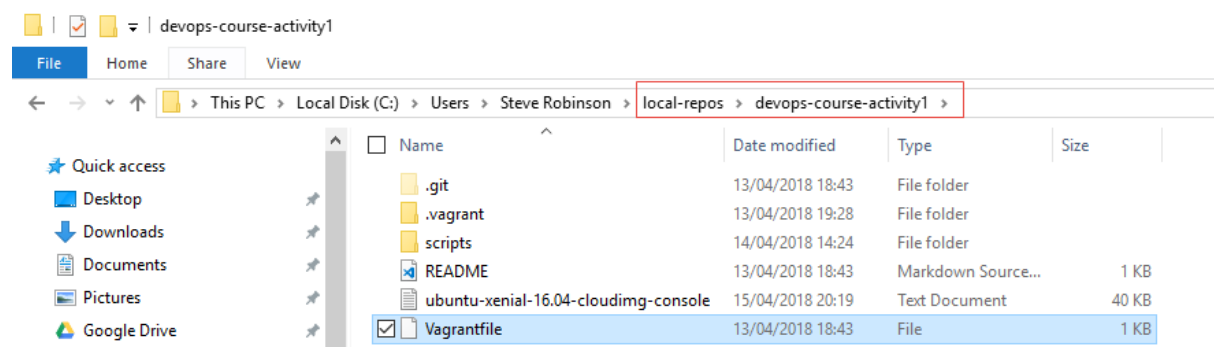
Installing a localised Linux (Ubuntu) VM using Vagrant

Inside the repo is a file called

Vagrantfile

with the following contents, feel free to locate it using file-explore and then open it using

notepad++



The file's contents is as follows:

```
# -*- mode: ruby -*-  
  
# vi: set ft=ruby :
```



```

Vagrant.configure(2) do |config|

  # Every Vagrant virtual environment requires a box to build off of.
  config.vm.box = "ubuntu/xenial64"

  config.vm.provider "virtualbox" do |vb|
    # Display the VirtualBox GUI when booting the machine
    #   vb.gui = true

    vb.name = "my_vm"

    # Example: Customize the amount of memory on the VM:
    vb.memory = "1024"

    # Example: Use VBoxManage to customize the VM. For example to change
    memory or VCPU
    vb.customize ["modifyvm", :id, "--cpus", "1"]
  end
end

```

vagrant init

We will now execute `vagrant init` command to initialise vagrant in this folder

```
vagrant init
```

Because a Vagrantfile already exists, We will see the following message:

```

Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant init
'Vagrantfile' already exists in this directory. Remove it before
running 'vagrant init'.

```

We need not worry since we already have our `Vagrantfile` from the git clone/pull we don't require Vagrant to create an empty default `Vagrantfile` (template example)

Vagrant up

We will now issue a vagrant up command, and because the default expected Vagrant Configuration file name is "`Vagrantfile`" it will execute immediately.

Type the following command:

```
vagrant up
```

Note: If you get the following error:

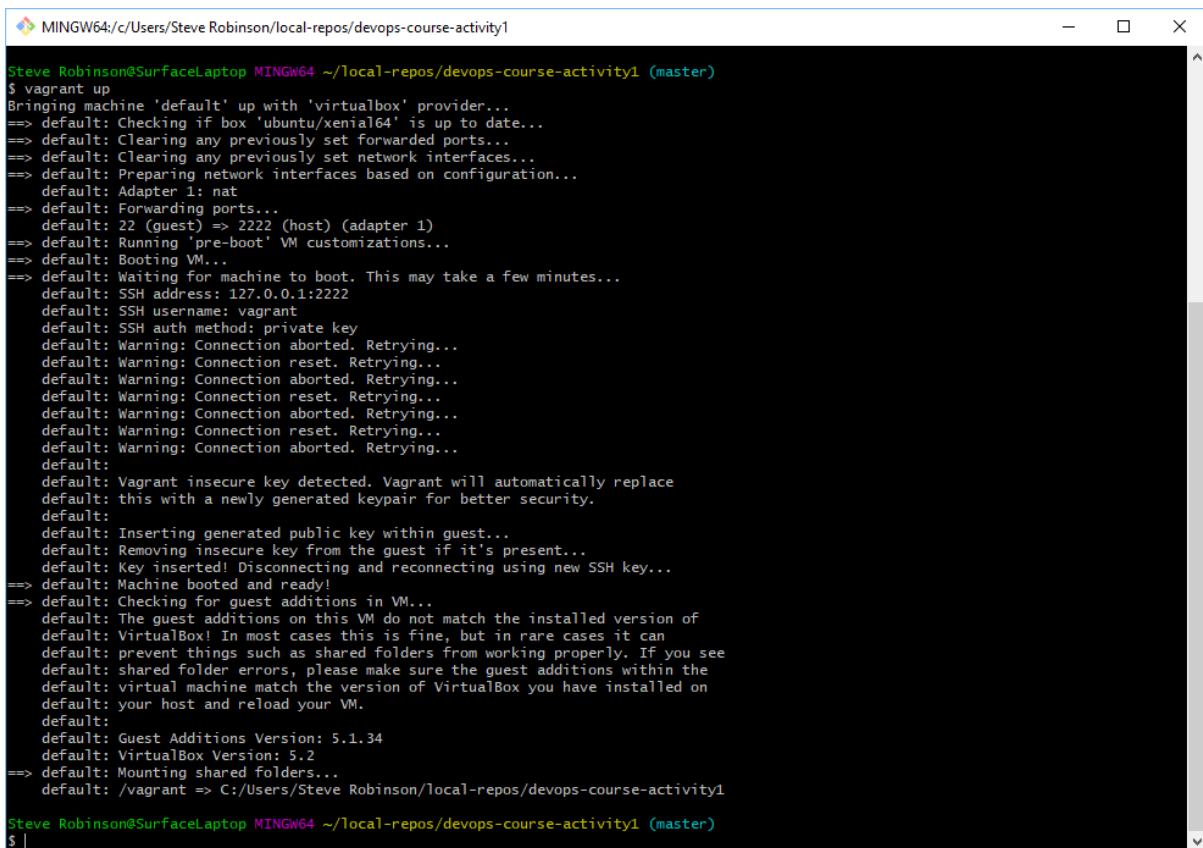
```
> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
There was an error while executing 'VBoxManage', a CLI used by Vagrant
for controlling VirtualBox. The command and stderr is shown below.

Command: ["startvm", "17b28c23-cbd5-4b65-941c-6593c80ad8f9", "--type", "headless"]

Stderr: VBoxManage.exe: error: Raw-mode is unavailable courtesy of Hyper-V. (VERR_SUPDRV_NO_RAW_MODE_HYPER_V_ROOT)
VBoxManage.exe: error: Details: code E_FAIL (0x80004005), component ConsoleWrap, interface IConsole
```

Please check that you have turn off the hyper visor launch type as explained earlier in the document.

Result of running a `vagrant up` command in a Git Bash terminal should look like



```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'ubuntu/xenial64' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
default: Adapter 1: nat
==> default: Forwarding ports...
default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
default: SSH address: 127.0.0.1:2222
default: SSH username: vagrant
default: SSH auth method: private key
default: Warning: Connection aborted. Retrying...
default: Warning: Connection reset. Retrying...
default: Warning: Connection aborted. Retrying...
default: Warning: Connection reset. Retrying...
default: Warning: Connection aborted. Retrying...
default: Warning: Connection reset. Retrying...
default: Warning: Connection aborted. Retrying...
default:
default: Vagrant insecure key detected. Vagrant will automatically replace
default: this with a newly generated keypair for better security.
default:
default: Inserting generated public key within guest...
default: Removing insecure key from the guest if it's present...
default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
default: The guest additions on this VM do not match the installed version of
default: VirtualBox! In most cases this is fine, but in rare cases it can
default: prevent things such as shared folders from working properly. If you see
default: shared folder errors, please make sure the guest additions within the
default: virtual machine match the version of VirtualBox you have installed on
default: your host and reload your VM.
default:
default: Guest Additions Version: 5.1.34
default: VirtualBox Version: 5.2
==> default: Mounting shared folders...
default: /vagrant => C:/Users/Steve Robinson/local-repos/devops-course-activity1
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$
```

Vagrant status

We should first run the `vagrant status` command to check the state of the newly created VM.

`vagrant status`

Result:

```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant status
Current machine states:

default                                running (virtualbox)

The VM is running. To stop this VM, you can run 'vagrant halt' to
shut it down forcefully, or you can run 'vagrant suspend' to simply
suspend the virtual machine. In either case, to restart it again,
simply run 'vagrant up'.
```

Vagrant ssh

We can now create an SSH session into the new Linux VM (Ubuntu) using the `vagrant ssh` command

```
vagrant ssh
```

Result:

```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant ssh
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.

vagrant@ubuntu-xenial:~$
```

We can see that we have now entered an `ssh` session into the Linux VM.

If we type the following command, then the VM can be updated to the latest patches for Ubuntu Xenial (Ubuntu 16.04.x)

```
sudo apt-get update
```

Type `y` to continue

```
Get:156 http://archive.ubuntu.com/ubuntu xenial-backports/universe amd64 Packages [7,104 B]
Get:157 http://archive.ubuntu.com/ubuntu xenial-backports/universe Translation-en [3,844 B]
Fetched 24.8 MB in 7s (3,106 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  apport libplymouth4 plymouth plymouth-theme-ubuntu-text python3-apport python3-problem-report
6 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 411 kB of archives.
After this operation, 4,096 B of additional disk space will be used.
Do you want to continue? [Y/n]
```

Once the update is complete, then type the command below to check the version of Ubuntu.

```
lsb_release -a
```

```
vagrant@ubuntu-xenial:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 16.04.4 LTS
Release:        16.04
Codename:       xenial
vagrant@ubuntu-xenial:~$
```

Type `exit` to leave the ssh session and return to Git Bash terminal

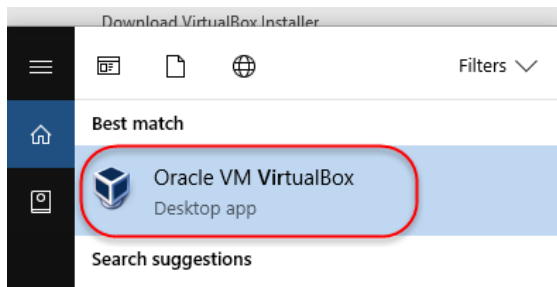
```
vagrant@ubuntu-xenial:~$ exit
logout
Connection to 127.0.0.1 closed.

Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$
```

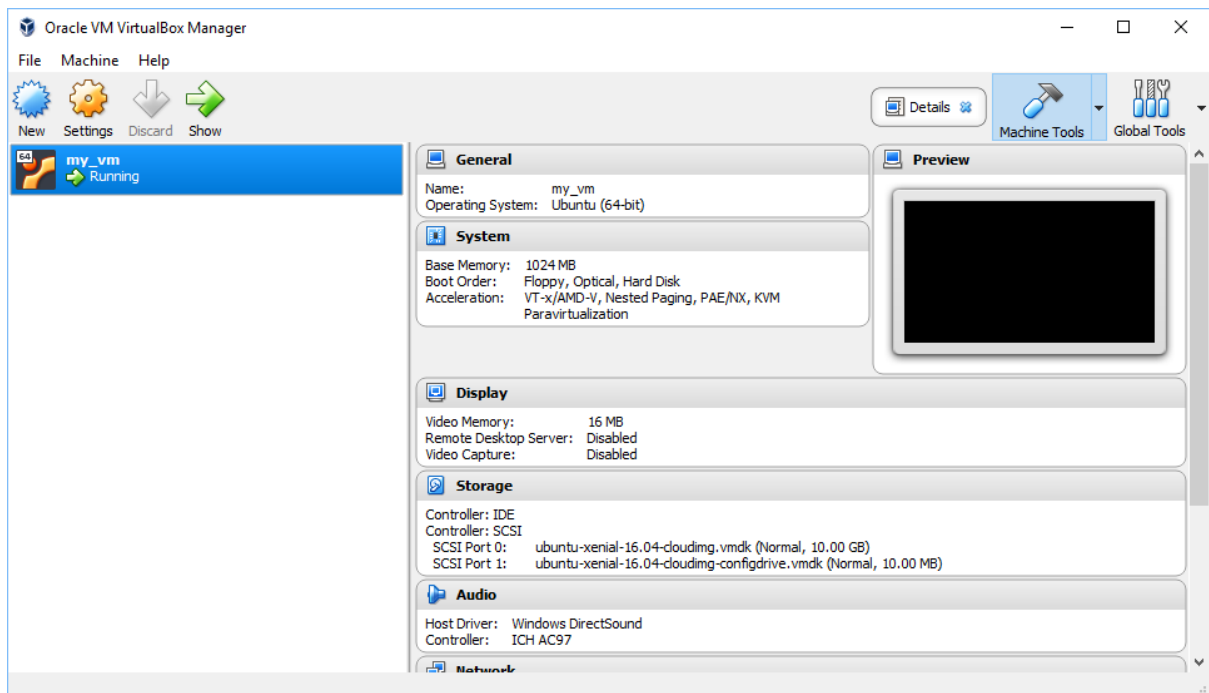
Congratulations we now have a running Linux VM.

We can also check the Virtual Box GUI, and see the VM running

Locate the Oracle Virtual Box shortcut



Launch VirtualBox, which will load the VirtualBox GUI.



Note: It is possible to stop and start the VM using vagrant commands or use the VirtualBox GUI. The next few pages demonstrate example of different ways to stop and start VirtualBox provided VMs as managed by Vagrant.

Please spend a little time testing a few of these commands.

Stopping and Starting VMs using vagrant

It is possible to stop and start Vagrant managed VMs using the command line.

Start a VM using vagrant

To start a VirtualBox VM using Vagrant

Ensuring you are in the correct directory for example using a Git Bash terminal on your Windows Desktop.

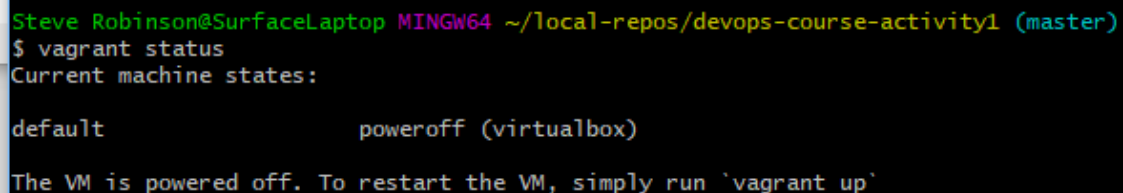
```
cd ~/local-repos/devops-course-activity1  
vagrant status
```

Result:

Current machine states:

```
default          poweroff (virtualbox)
```

The VM is powered off. To restart the VM, simply run `vagrant up`



```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)  
$ vagrant status  
Current machine states:  
  
default          poweroff (virtualbox)  
  
The VM is powered off. To restart the VM, simply run `vagrant up`
```

To start the VM we issue the `vagrant up` command as shown below

Ensuring you are in the correct directory for example using a Git Bash terminal on your Windows Desktop.

```
vagrant up
```

Result:

```
Bringing machine 'default' up with 'virtualbox' provider...  
==> default: Checking if box 'ubuntu/xenial64' is up to date...  
==> default: A newer version of the box 'ubuntu/xenial64' for provider  
'virtualbox' is  
==> default: available! You currently have version '20180410.0.0'. The  
latest is version  
==> default: '20180413.0.0'. Run `vagrant box update` to update.  
==> default: Clearing any previously set forwarded ports...  
==> default: Clearing any previously set network interfaces...  
==> default: Preparing network interfaces based on configuration...  
default: Adapter 1: nat  
==> default: Forwarding ports...  
default: 22 (guest) => 2222 (host) (adapter 1)  
==> default: Running 'pre-boot' VM customizations...  
==> default: Booting VM...  
==> default: Waiting for machine to boot. This may take a few minutes...  
default: SSH address: 127.0.0.1:2222  
default: SSH username: vagrant  
default: SSH auth method: private key
```

```

    default: Warning: Connection reset. Retrying...
    default: Warning: Connection aborted. Retrying...
    default: Warning: Connection reset. Retrying...
    default: Warning: Connection aborted. Retrying...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: The guest additions on this VM do not match the installed
version of
    default: VirtualBox! In most cases this is fine, but in rare cases it
can
    default: prevent things such as shared folders from working properly.
If you see
    default: shared folder errors, please make sure the guest additions
within the
    default: virtual machine match the version of VirtualBox you have
installed on
    default: your host and reload your VM.
    default:
    default: Guest Additions Version: 5.1.34
    default: VirtualBox Version: 5.2
==> default: Mounting shared folders...
    default: /vagrant => C:/Users/Steve Robinson/local-repos/devops-course-
activity1
==> default: Machine already provisioned. Run `vagrant provision` or use
the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always
will still run.

```

```

MINGW64~/c:/Users/Steve Robinson/local-repos/devops-course-activity1
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'ubuntu/xenial64' is up to date...
==> default: A newer version of the box 'ubuntu/xenial64' for provider 'virtualbox' is
==> default: available! You currently have version '20180410.0.0'. The latest is version
==> default: '20180413.0.0'. Run `vagrant box update` to update.
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key

    default: Warning: Connection reset. Retrying...
    default: Warning: Connection aborted. Retrying...
    default: Warning: Connection reset. Retrying...
    default: Warning: Connection aborted. Retrying...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: The guest additions on this VM do not match the installed version of
    default: VirtualBox! In most cases this is fine, but in rare cases it can
    default: prevent things such as shared folders from working properly. If you see
    default: shared folder errors, please make sure the guest additions within the
    default: virtual machine match the version of VirtualBox you have installed on
    default: your host and reload your VM.
    default:
    default: Guest Additions Version: 5.1.34
    default: VirtualBox Version: 5.2
==> default: Mounting shared folders...
    default: /vagrant => C:/Users/Steve Robinson/local-repos/devops-course-activity1
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.

```

Note: We might see output lines containing, the following text:

```
default: Warning: Connection reset. Retrying...
```

This is vagrant polling/waiting for VirtualBox (our default VM provider) to complete starting the VM.

To re-check status again, we will issue the vagrant status command as shown below:

```
vagrant status
```

Result

```
Current machine states:
```

```
default                running (virtualbox)
```

```
The VM is running. To stop this VM, you can run `vagrant halt` to
shut it down forcefully, or you can run `vagrant suspend` to simply
suspend the virtual machine. In either case, to restart it again,
simply run `vagrant up`.
```

Stop a VM using Vagrant

To stop a Vagrant managed VM, we can issue the following command to allow a graceful stop.

```
vagrant halt
```

Result:

```
=> default: Attempting graceful shutdown of VM...
```

Result:

We can see that it does not return status, as it just lets the VM take it time as required.

If we wish to pull the power plug as it were and force the VM to halt immediately we can add the **--force** (**-f** or **--force**) switch.

```
$ vagrant halt -f
```

Typical result:

```
=> default: Attempting graceful shutdown of VM...
```

Note: You can always learn more about extra options using help for example...


```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant halt --help
Usage: vagrant halt [options] [name|id]

Options:
  -f, --force                Force shut down (equivalent of pulling power)
  -h, --help                Print this help
```

Here is an example of a forced immediate stop using multiple commands on the command-line ie force a stop, and also immediately display status

```
vagrant halt -f && vagrant status
```

Result:

```
vagrant halt -f && vagrant status
==> default: Forcing shutdown of VM...
Current machine states:
```

```
default                poweroff (virtualbox)
```

The VM is powered off. To restart the VM, simply run `vagrant up`

```
Steve Robinson@SurfaceLaptop MINGW64 ~/local-repos/devops-course-activity1 (master)
$ vagrant halt -f && vagrant status
==> default: Forcing shutdown of VM...
Current machine states:

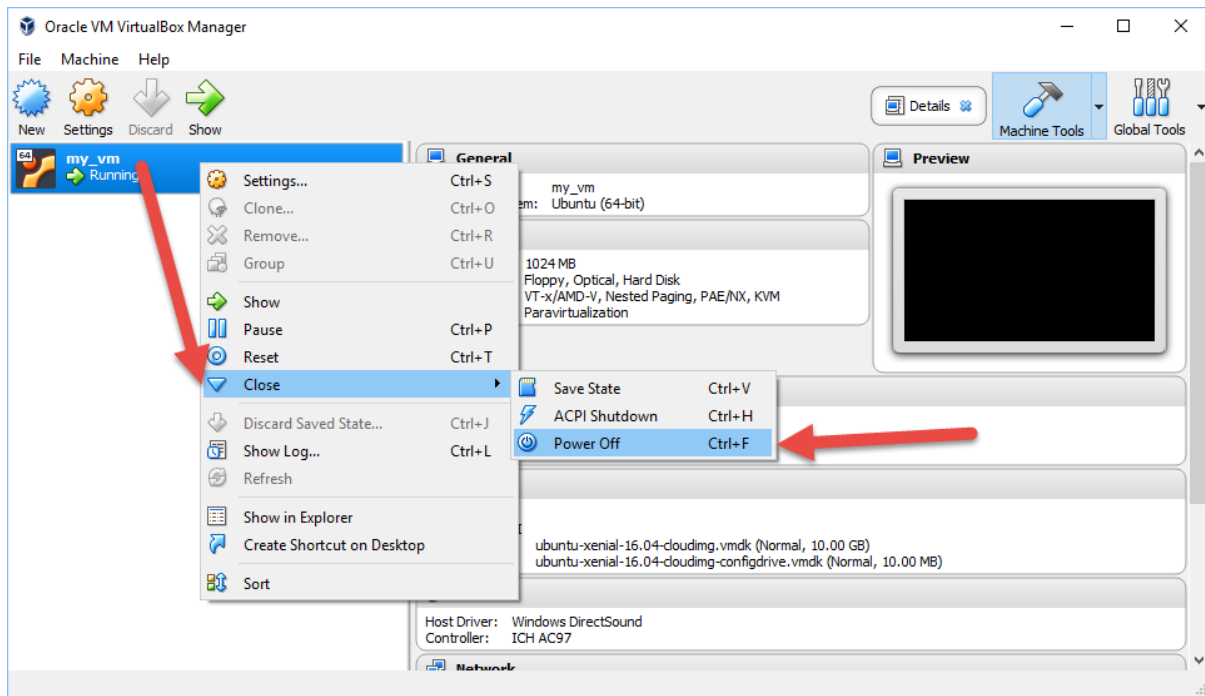
default                poweroff (virtualbox)

The VM is powered off. To restart the VM, simply run `vagrant up`
```

Using the VirtualBox GUI.

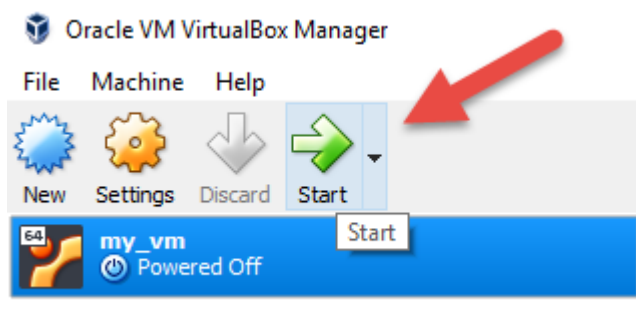
Stop a VM

Right-mouse click on VM and locate the Close submenu and choose Power Off.



To Start a stopped VM

Locate the VM, select it, then Click the Start button in the main menu.



Note: When you start a VM using the VirtualBox GUI, it will launch a terminal session as a human interface as it is not a headless action.

When we use vagrant, it is a headless action i.e. no VirtualBox session GUI is launched.

Summary

We have completed the Activity 1, Part 2 exercise, we will now move onto the Part 3 document.

Open the [DevOpsActivity1-Part3.pdf](#) file to continue on with the Activity.