Vagrant is a tool for building and managing virtual machine environments in a single workflow. With an easy-to-use workflow and focus on automation, Vagrant lowers development environment setup time, increases production parity, and makes the "works on my machine" excuse a relic of the past.

**Why Vagrant?**

Vagrant provides easy to configure, reproducible, and portable work environments built on top of industry-standard technology and controlled by a single consistent workflow to help maximize the productivity and flexibility of you and your team.

To achieve its magic, Vagrant stands on the shoulders of giants. Machines are provisioned on top of VirtualBox, VMware, AWS, Then, industry-standard provisioning table such as shell scripts, Chef, or Puppet, can automatically install and configure software on the virtual machine.

# Vagrantfile

The primary function of the Vagrantfile is to describe the type of machine required for a project, and how to configure and provision these machines. Vagrantfiles are called Vagrantfiles because the actual literal filename for the file is Vagrantfile (casing does not matter unless your file system is running in a strict case sensitive mode).

Vagrant is meant to run with one Vagrantfile per project, and the Vagrantfile is supposed to be committed to version control. This allows other developers involved in the project to check out the code, run vagrant up, and be on their way. Vagrantfiles are portable across every platform Vagrant supports.

**Lookup Path**

When you run any vagrant command, Vagrant climbs up the directory tree looking for the first Vagrantfile it can find, starting first in the current directory. So if you run vagrant in /home/mitchellh/projects/foo, it will search the following paths in order for a Vagrantfile, until it finds one:

/home/mitchellh/projects/foo/Vagrantfile

/home/mitchellh/projects/Vagrantfile

/home/mitchellh/Vagrantfile

/home/Vagrantfile

/Vagrantfile

This feature lets you run vagrant from any directory in your project.

You can change the starting directory where Vagrant looks for a Vagrantfile by setting the VAGRANT\_CWD environmental variable to some other path.

# Boxes

Boxes are the package format for Vagrant environments. A box can be used by anyone on any platform that Vagrant supports to bring up an identical working environment.

The vagrant box utility provides all the functionality for managing boxes. You can read the documentation on the [vagrant box](https://www.vagrantup.com/docs/cli/box) command for more information.

The easiest way to use a box is to add a box from the [publicly available catalog of Vagrant boxes](https://vagrantcloud.com/boxes/search). You can also add and share your own customized boxes on this website.

Boxes also support versioning so that members of your team using Vagrant can update the underlying box easily, and the people who create boxes can push fixes and communicate these fixes efficiently.

You can learn all about boxes by reading this page as well as the sub-pages in the navigation to the left.

## [»](https://www.vagrantup.com/docs/boxes#discovering-boxes)Discovering Boxes

The easiest way to find boxes is to look on the [public Vagrant box catalog](https://vagrantcloud.com/boxes/search) for a box matching your use case. The catalog contains most major operating systems as bases, as well as specialized boxes to get you up and running quickly with LAMP stacks, Ruby, Python, etc.

The boxes on the public catalog work with many different [providers](https://www.vagrantup.com/docs/providers/). Whether you are using Vagrant with VirtualBox, VMware, AWS, etc. you should be able to find a box you need.

Adding a box from the catalog is very easy. Each box shows you instructions with how to add it, but they all follow the same format:

$ vagrant box add USER/BOX

For example: vagrant box add hashicorp/bionic64. You can also quickly initialize a Vagrant environment with vagrant init hashicorp/bionic64.

**Namespaces do not guarantee canonical boxes!** A common misconception is that a namespace like "ubuntu" represents the canonical space for Ubuntu boxes. This is untrue. Namespaces on Vagrant Cloud behave very similarly to namespaces on GitHub, for example. Just as GitHub's support team is unable to assist with issues in someone's repository, HashiCorp's support team is unable to assist with third-party published boxes.

## [»](https://www.vagrantup.com/docs/boxes#official-boxes)Official Boxes

HashiCorp (the makers of Vagrant) publish a basic Ubuntu 18.04 64-bit box that is available for minimal use cases. It is highly optimized, small in size, and includes support for VirtualBox, Hyper-V, and VMware. You can use it like this:

$ vagrant init hashicorp/bionic64

or you can update your Vagrantfile as follows:

Vagrant.configure("2") do |config|

config.vm.box = "hashicorp/bionic64"

end

Note:

You need Virtualbox to Start vagrant, to install it use the below.

* wget -q https://www.virtualbox.org/download/oracle\_vbox\_2016.asc -O- | sudo apt-key add –
* sudo apt-key add -sudo add-apt-repository "deb [arch=amd64]
* http://download.virtualbox.org/virtualbox/debian $(lsb\_release -cs) contrib"
* sudo apt update
* sudo apt install virtualbox-6.0

Starting Vagrant:

1. vagrant init ubuntu/trusty64
2. vagrant up
3. (in case error appears as bsdtar not available, enter the command): sudo apt install libarchive-tools
4. Login into the machine using: vagrant ssh

Older version of Ubuntu use below:

1. sudo apt install virtualbox

**Installing Vagrant:**

sudo apt-get update  
sudo apt-get install unzip  
wget <https://releases.hashicorp.com/vagrant/2.2.14/vagrant_2.2.14_linux_amd64.zip>  
unzip vagrant\_2.2.14\_linux\_amd64.zip  
sudo mv vagrant /usr/local/bin/  
vagrant --version