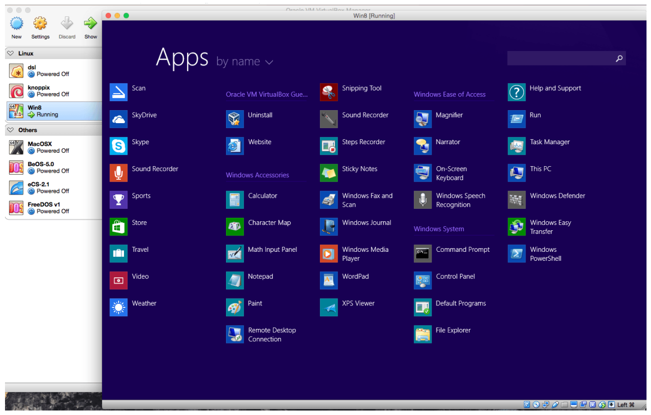
Computational Methods Class

What operating system does your computer have? OS X, Windows, Linux, Solaris? Did you know that your computer (called the *host*), say a Mac, can behave like another (called the *guest*), say a Windows PC? This is achieved through virtual machines (VMs).

VirtualBox and Vagrant

VirtualBox (first released in 2007) is a virtualization tool; that is, it supports the creation and management of guest virtual machines running versions of operating systems other than your host machine. The first thing to note is that it installs on your existing Intel or AMD-based computers, whether they are running Windows, Mac, Linux or Solaris operating systems. It extends the capabilities of your existing computer so that it can run multiple operating systems (inside multiple virtual machines) at the same time. So, for example, you can run Windows and Linux on your Mac, run Windows Server 2008 on your Linux server, run Linux on your Windows PC, and so on, all alongside your existing applications. You can install and run as many virtual machines as you like -- the only practical limits are disk space and memory. However, you need to build each VM from scratch every time you want one.



This shows how VirtualBox, installed on a Mac computer, is running Windows 8 in a virtual machine window.

Vagrant (first released in 2010) helps you to automate the process of building virtual machines inside your computer. Vagrant can control a variety of virtualization tools, like VirtualBox. Vagrant is a command line utility for managing the lifecycle of virtual machines. Vagrant provides easy to configure, reproducible, and portable work environments built on top of industry-standard technology (like VirtualBox) and controlled by a single consistent workflow to avoid the unnecessary maintenance and setup time and to help maximize the productivity and flexibility of you and your team.

VirtualBox Download: https://www.virtualbox.org/wiki/Downloads
Vagrant Download: https://www.vagrantup.com/downloads.html

VirtualBox Installation: https://www.virtualbox.org/manual/ch02.html
Vagrant Installation: https://www.vagrantup.com/docs/installation/

Installation Videos: Mac (https://youtu.be/RhhF8Yh7OnE), Windows (*VirtualBox* https://youtu.be/r1rYoH_XwFc, *Vagrant* https://youtu.be/_LctHOLN604), Linux (*VirtualBox* https://youtu.be/nD4vCxtWRdk, *Vagrant* https://youtu.be/VI5m1UyNBiE)

Setting Up a Virtual Machine

We will use Vagrant to automate the process of installing Ubuntu (a distribution of Linux) in a virtual machine hosted in your computer.

Almost all interaction with Vagrant is done through the command-line interface, so you need a command-line bash shell (e.g. Terminal in a Mac). The interface is available using the `vagrant` command. The `vagrant` command in turn has many subcommands, such as `vagrant up` and `vagrant halt`.

Please follow the steps below.

1. Create a folder for the class in your computer called 'CompMethods' and locate there the file 'Vagrantfile' and the folder 'vm_provision' that you can download from https://github.com/pavelsolis/Computational_Methods_VM/

2. Open the Terminal

3. Enter each of the commands below in the Terminal and hit enter (following the order)

In the Terminal type (do NOT type $ nor #)	What does the command do?
`$ vagrant –v`	In case you want to verify that Vagrant is already installed in your computer, this command tells you the version of Vagrant installed.
`$ cd /users/…/CompMethods/`	Go to the folder created for the class (fill in the dots with your working directory).
`$ vagrant box add ubuntu/trusty64`	Every Vagrant development environment requires a box. A box is a template for building a virtual machine. The box we will use is a distribution of Linux called Ubuntu.
`$ vagrant up`	It sets up the box (using the information in Vagrantfile). In other words, it brings the virtual machine up, which means that your computer will host a VM running Linux. (You can see this by opening VirtualBox, which will show an entry for the VM just created; you don't need VirtualBox open though.)
`$ vagrant ssh`	This gives you access into the VM (i.e. you will be in a Linux session), which can be controlled through the Terminal. Note that the beginning of the command line changes to 'vagrant@...'

`$ su – self`	Switches the user from ‘vagrant’ to the user ‘self’. It will request for the password, type ‘JHUEcon’ (*without* the apostrophes).
`$ passwd`	This allows you to change the password for the user ‘self’. When prompted, type ‘JHUEcon’ in the (current) password. Then enter a password you would like to assign for the user ‘self’ (remember it!).
`$ exit`	Logs you out of the ‘self’ user.
`$ su – setup`	Switches the user from ‘vagrant’ to the user ‘setup’. It will request for the password, type ‘JHUEcon’.
`$ exit`	Logs you out of the ‘setup’ user.
`$ sudo su`	Switches you into a user with administrator privileges (‘root’ user).
`# mkdir -p /usr/sync/tools`	Creates a folder (in Linux terminology is called directory) that will contain tools Prof. Carroll will be constructing for the class.
`$ exit`	Logs you out of the ‘root’ user.
`$ cd /`	Takes you to the top level of your directory.
`$ ls`	Lists all the existing directories in the VM. Note the vagrant directory.
`$ ls vagrant/`	Lists what is inside the vagrant directory: Vagrantfile and vm_provision directory. Note that they are in the VM! They are actually the *same* file and directory that are in your CompMethods folder in your actual host computer. Indeed, all the files you put in the CompMethods folder in the host machine will be synched in the guest machine.
`$ cd usr/sync/tools/`	Locates you into the directory for the class.
`$ exit`	Takes you back to your computer (i.e. you are no longer in the Linux session although the VM is still running; if you want to go back to the Linux session type `vagrant ssh`).
`$ vagrant halt`	Shuts down the guest operating system and powers down the guest machine. You can use `vagrant up` to boot it again.

What just happened? You set up a VM running Linux in your computer. In that VM there is a directory for the class (usr/sync/tools) and two user accounts called ‘setup’ (identical across students with password ‘JHUEcon’) and ‘self’ (unique to you). The commands that will be used on a regular basis are those in the shaded cells.