Lesson Title: Why Vagrant?

Rationale

Vagrant is getting a lot attention as a useful tool for creating customized developer environments, but it's not appropriate for every situation. In this lesson we cover the hardware limitations you should be aware of before we dive into this learning series.

Lesson Outcomes

Major Objective

By the end of this lesson, you will be able to describe the advantages, and disadvantages for creating a local development environment with Vagrant.

Student Self-Check Tasks

- Do you want to proceed with Vagrant?
- Does it sound like it will solve your problems? [yes|no]
- Do you have the necessary hardware to proceed with the lessons?
- Or should you stick with WAMP/MAMP?

Lesson Summary

This learning series requires the following hardware capacity:

- 4G RAM *minimum*, 8G preferred.
- At least 5-10GB of disk space available per virtual machine.

Lesson Title: Installing Vagrant and VirtualBox

Rationale

Before we jump into the automation of creating a new developer environment, we need to get the building blocks we'll be working with. You will need to download and install both Vagrant and VirtualBox.

You might use a number of different provisioning scripts, or configuration options, or etc, but you need to start with the software installed. "Anything hard should be practiced, not avoided." Try to get into the habit of creating robust, repeatable decisions instead of a brittle infrastructure that can't be easily recreated.

Lesson Outcomes

Major Objective

By the end of this lesson, you will be able to install Vagrant and VirtualBox on a host platform of OSX, or Windows; and create and destroy a Vagrant instance.

Student Self-Check Tasks

- locate command line;
- successfully initialize a directory for use with Vagrant (confirm that the file Vagrantfile has been created);
- successfully intialize a new VM (confirm no fatal errors when running `vagrant up` and ensure a new binary was created in your virtual machines folder);
- successfully destroy a vagrant instance (confirm the binary file was deleted from the VirtualBox binary folder).

Lesson Summary

1. Install VirtualBox
2. Install Vagrant
3. Create a new directory for your configuration scripts. e.g. Websites, or Work-environments. This folder will not contain the VirtualBox binary files, just the Vagrant configuration scripts.
4. From the command line, run the following commands:

- `$ vagrant init precise32 http://files.vagrantup.com/precise32.box`
- `$ vagrant up`
- `$ vagrant ssh` Note: this command will not work for Windows. You will need to use PuTTY to log into your machine. This is covered in the Extras lesson. You're now inside your Ubuntu server!

5. To return to your host machine, run the command:

- `$ exit`

6. To destroy the VM and remove the binary disk image:

- `$ vagrant destroy` This command must be run from the directory which contains the Vagrantfile for the box you wish to destroy. It will not remove any of your configuration files, to recreate the machine, use the commands in step 4.

Gotchas

There are some gotchas to be aware of:

- Linux and Windows *may* need a reboot because of the kernel drivers being added.
- Sometimes upgrades don't work, and re-installing is your best bet. If you have automatic software updates turned on, and Vagrant stops working for you without an obvious reason, try removing everything and re-install using the instructions in this lesson. Removing Vagrant and VirtualBox are covered in one of the extra lessons.

Resources

- https://www.virtualbox.org/wiki/Downloads
- http://www.vagrantup.com/downloads.html
- link to extra lesson: SSH lesson for Windows
- link to extra lesson: Removing Vagrant and VirtualBox

Lesson Title: Adding a Web Server

Rationale

This is something that ultimately we want to have in a provisioning script, but this shows you the virtual machine is just like any other server you might have worked with in the past. Any time you want to add new software, you can always come back to this lesson if you (for some reason) don't want to create a recipe and re-provision your machine. Hopefully by the end of this series you'll see why you should never *want* to use this method again.

Lesson Outcomes

Major Objective

By the end of this lesson, you will be able to log into your virtual machine, and use the command line to install new software.

Student Self-Check Tasks

- Log into the server via SSH
- Add an AMP stack.

Lesson Summary

If you don't want to start with the Ubuntu LTS (precise32) base box, you can choose a different one.

1. Navigate to www.vagrantbox.es.
2. Locate a base box you would like to use. Ensure you are matching the "provider" to what you have. e.g. VirtualBox for these lessons (not VMware).
3. At the command lines, initialize your new server:

- `$ vagrant init <box name> <box URL>`
  - box name: ringtail64
  - box URL (mind the line wrap): http://cloud-images.ubuntu.com/vagrant/raring/current/raring-server-cloudimg-amd64-vagrant-disk1.box
- `$ vagrant up`

Ensure your server is up and running, and then log in with vagrant ssh. This must be done from the directory which contains the file Vagrantfile.

Install an AMP server using the directions for your base box. We'll use the same technique as is covered in Addi's lesson "Installing a Web Server on Ubuntu".

- `$ sudo apt-get install tasksel`
- `$ sudo tasksel install lamp-server`

This will install Apache, MySQL, and PHP.

To confirm the server is running, use the command: `pstree`.

Gotchas

- OSX can use vagrant ssh; Windows will need to install PuTTY and use ssh vagrant.
- If you get the error `tasksel: aptitude failed (100)`:

1. `$ sudo apt-get update`
2. `$ sudo tasksel install lamp-server` (I got this error on a precise32 base box with a Windows 8 host running inside of Parallels on a Mac OSX machine.)

Resources

- Installing a Web Server on Ubuntu
- SSH tutorial

Lesson Title: Connecting to Your Web Server

Rationale

Ultimately local Web development is only effective if you can actually see the results of your work in a Web browser. We need to configure match the configuration in Vagrantfile with our host's networking configuration so that the two know where to find each other.

Lesson Outcomes

Major Objective

By the end of this lesson you will be able to correctly configure your host's network settings to make the virtual machine available via a browser.

Student Self-Check Tasks

In a web browser see "It Works" when you pull up http://localhost.

Lesson Summary

1. Configure port forwarding in Vagrantfile. `# config.vm.network "forwarded_port", guest: 80, host: 4567`
2. Reload the vagrant instance. `$ vagrant reload`
3. Log into the server, confirm Apache is running. (Hint: it's not running.) `$ curl 'http://localhost:80'`
4. Restart Apache. `$ /etc/init.d/apache2 start`
5. In a Web browser, navigate to: `http://127.0.0.1:4567` (Hopefully) It works!

Gotchas

This can sometimes be the tricky bit, especially if you already have a lot of custom network configuration. If this lesson doesn't work for you, you will need to do an audit of your system to find any software which might be interfering with the connection. We're going to limit ourselves to one VM running at a time (all the same ports). This is mostly because of the hardware limitations I expect you'll have. If your machine is more powerful, great! But we're not covering it in this video series.

Lesson Title: Sharing Files Between Your Guest and Host Machines

Rationale

Your Web server will be located inside the Vagrant instance, in order to upload files to the server, you can either use SCP or enable file sharing. With file sharing enabled you will be able to transfer files into specific directories within your Vagrant instance using your local file manager (e.g. Finder or Windows Explorer) and continue working on them with your favorite IDEs and code editing tools. Technically, enabling file sharing isn't necessary, but it is much more convenient than SCPing files into your local server.

Lesson Outcomes

Major Objective

By the end of this lesson you will be able to share files between your host and guest machines without the use of SCP.

Student Self-Check Tasks

After enabling file sharing, you should be able to copy a file into the shared folder in your Vagrantfile and have it automatically appear inside the Vagrant instance. e.g. you can load a static web page created on your local machine without having to use SCP.

Lesson Summary

As long as you're using basic file sharing (no NFS), it should work "out of the box" for Windows 8, and OSX.

1. Configure synced folder in Vagrantfile.

- Locate and remove: `# config.vm.synced_folder "../data", "/vagrant_data"`
- In its place, add: `config.vm.synced_folder "docroot", "/var/www"`

2. Create a new directory using the name identified in the file Vagrantfile.

- `$ mkdir docroot`

3. Reload the vagrant instance (and restart the server). The contents of `/var/www` will appear in `docroot`.

- `$ vagrant reload`
- `$ vagrant ssh`
- `$ sudo /etc/init.d/apache start`
- `$ exit`

4. Edit the Web page from the host machine.
5. Refresh the browser to see the changed file.

Gotchas

No known gotchas. If you experience problems, create an issue for the project and we can update the notes.

Lesson Title: Configuration Management

Rationale

Configuration management (CM) allows administrators to establish and maintain consistency of a computer system throughout its life against a specified set of rules. CM is what allows you to maintain consistency across all developer environments, and also from your development environment to your server. Configuration management automates the setup of machines so that there are no inconsistencies. It also makes it faster for your to spin up a virtual environment without having to manually install a Web server, like we did in a previous lesson.

Lesson Outcomes

Major Objective

By the end of this lesson, you will be able to explain how configuration management relates to virtual machines, and why you would want to provision new virtual machines.

Student Self-Check Tasks

Describe in which communities Chef, Puppet, and Ansible are popular.

Lesson Summary

- Vagrant is a wrapper which creates virtual development environments from base boxes. It acts as a wrapper around a provider, such as VirtualBox, and configuration management software, such as Chef.
- Configuration Management: Configuration management is a process which allows administrators to establish and maintain consistency of a computer system throughout its life against a specified set of rules. Configuration management enables consistency across one, or many, machines.
- Popular open source configuration managaement tools supported by Vagrant include: Puppet, Chef, and Ansible. Chef tends to be popular among Drupal developers. Puppet tends to be popular among system administrators (who are not necessarily Web developers first-and-foremost).
- Provisioning is the process of creating, or altering, a system in order to prepare it for service.
- Vagrant allows you to use one or more configuration management tools from the same Vagrantfile to provision a machine; as well as use shell scripts.

Resources

- Configuration Management
- Provisioning
- Comparison of Configuration Management Tools
- Chef Overview
- Chef Cookbooks

Lesson Title: Provisioning with Chef

Rationale

Because you want to be able to have stuff pre-installed when you turn on the machine. It's not necessary, but who wants to apt-get all the things? Hint: not me. The scripts we'll install will get you AMP (you already have Linux); and drush.

Lesson Outcomes

Major Objective

By the end of this lesson you will be able to add a Chef recipe, and re-provision Vagrant.

Student Self-Check Tasks

I can issue *drush* commands necessary to install Drupal.

Lesson Summary

In the video, I've already setup Chef recipes, roles, and cookbooks for you. So instead of having to download all the pieces separately, you can just start with this repository. The outline here, however, includes each of the steps that I went through to prepare the configuration Chef recipes and roles that you're downloading.

Assembling Chef Cookbooks

1. Locate Chef cookbooks for AMP stack, drush, and Drupal-specific PHP libraries you want pre-loaded.

- http://community.opscode.com/cookbooks/

2. Place downloaded cookbooks into a sub-folder in your project directory. For example: `chef-recipes/cookbooks`.
3. Create a role file to include all of the recipes (from each of the cookbooks) you want to load. This file can be placed into its own roles sub-directory. For example: `chef-recipes/roles/lamp_stack.rb`.
4. Update the Vagrantfile to include:

- path to cookbooks folder
- path to roles folder
- role(s) to load for this server
- new values for any of the default settings for your recipes

5. Provision the machine to enable / set-up / trigger the Chef configuration settings:

- `$ vagrant provision`

The machine is now configured, but does not have Drupal installed. This is on purpose. You probably have a specific Drupal project you're working on.

Configuration Files

The role files contain a list of individual recipes, and/or roles that must be run to provision a specific type of server. Generally there are three or four parts to them.

- name
- description
- run list (recipes and roles to add)
- default settings for recipes (optional)

You should investigate the contents of the folder `chef-recipes/roles`. There is a detailed README tailored to this learning series, as well as the two roles that were created for this lesson.

The Vagrantfile is well commented, this section includes a quick references of the settings that have been updated in the file.

```
config.vm.provision :chef_solo do |chef|
   # default settings
   chef.cookbooks_path = "path_to_cookbooks_folder"
   chef.roles_path = "path_to_roles_folder"

   # roles and recipes to load
   # recipes are generally loaded from within roles
   chef.add_role "name_of_role"
   chef.add_recipe "name_of_recipe"

   # overrides for settings in recipes
   chef.json = {
     "name_of_recipe" => {
       "name_of_setting" => "new_value",
     },
   }
end
```

Gotchas

- Apt will only run if the cache is older than a day. The first time the machine is provisioned, this *might* be a problem. You can force updating the cache with the following setting in Vagrantfile `chef.json = { "apt" => {"compiletime" => true} }`
- You may want to update Chef with the omnibus plugin for Vagrant if the cookbook you want to add requires a newer version of Chef.

On the host machine, install the plugin: `$ vagrant plugin install vagrant-omnibus`. And edit the Vagrantfile to add the following: `config.omnibus.chef_version = :latest`. Note: Chef must be updated before it is "run". For this lesson, I opted to use older versions of the cookbooks. 20 hours of my life spent working on upgrades that didn't work. The later scripts provide more flexibility (e.g. install ANY version of Drush, not just 5.8), but they work well enough for our needs.

Lesson Title: Installing Drupal with Drush

Rationale

Not automating the installation of Drupal on our vagrant instance is *by design*. It gives us the ability to quick spin up new machines and then put any version of Drupal onto them. It also allows us to use a single server for multiple Drupal projects (just as you would with a MAMP or WAMP environment). In this lesson you will learn how to install Drupal from the command line using Drush. If you already know how to do this, you should review the Lesson Summary, but you are not required to watch the video as it will be entirely review for you.

Lesson Outcomes

Major Objective

By the end of this lesson you will be able to install Drupal via Drush from within your Vagrant instance.

Student Self-Check Tasks

Drupal is installed, and can be viewed it in a web browser at http://localhost:4567.

Lesson Summary

The machine was configured in the previous lesson, but does not have Drupal installed. This is on purpose. You probably have a specific Drupal project you're working on. If you don't, you can use the following instructions to set up a generic instance of Drupal.

- `$ vagrant ssh`
- `$ cd /var/www/docroot`
- `$ drush dl drupal`

- `$ cd drupal-XXX` (where XXX is the version number for Drupal you've just downloaded)
- `$ drush si standard --db-url=mysql://root:root@localhost/drupal7 --db-su=root --db-su-pw=root --site-name="Drupal on Vagrant"` (for the lessons, the MySQL database has the username root; and the password root, if you have deviated from these insecure passwords, you wiill need to update this line to get the site install to work).

Lesson Title: Planning Your VM Use

Rationale

I started with one box per project but found that I task switch frequently enough that this was a problem considering my hardware constraints (8G RAM = only want one machine running at a time). The shut down + startup only takes ~5minutes, but it's still dead time especially as that doesn't include pulling a fresh copy of the Git repo and updating my DB. SO! If you need to task switch frequently, consider "grouping" your projects with similar requirements, and then breaking out new boxes as you justify the split. e.g. you need two different versions of PHP.

Lesson Objectives

Major Objective

By the end of this lesson, you will be able to describe how many Vagrant instances you will use, and why.

Student Self-Check Tasks

Create a list of all your projects requiring a local environment, and group them into similar requirements (e.g. version of PHP).

Resources

- Drupal Deploy Cookbook

Lesson Title: Updating Vagrant

Rationale

Sometimes upgrading doesn't work, so it's easier to rip everything off your system in order to upgrade.

Lesson Outcomes

Major Objective

By the end of this lesson, you will be able to remove relevant Vagrant from your system to prepare yourself for a clean installation.

Student Self-Check Tasks

When I perform an "upgrade" I walk through the installation process as if I had never installed Vagrant on my system.

Lesson Title: Using SSH on Windows

Rationale

`vagrant ssh` doesn't work on windows because of path issues. Although you can update some files to fix this, running PuTTY is slightly more accessible (and has a GUI configuration screen to fill in).

Lesson Objectives

Major Objective

By the end of this lesson you will be able to SSH into a remote machine using PuTTY.

Student Self-Check Tasks

You can log into your Vagrant machine using PuTTY with the IP address listed from running the command `vagrant ssh` in the same directory as your Vagrantfile.