使用 Vagrant 在 Windows下部署开发环境

做 Web开发少不了要在本地搭建好开发环境， 虽然说目前各种脚本都有对应的 Windows版，甚至是一键安装包， 但很多时候和 Windows环境的相性并不是那么好，各麻烦的问题是实际部署的环境通常是 Linux ，常常还要面临着开发和部署环境不一致， 上线前还要大量的调试。 更要命的是， 如果有很多机器需要装的话，那就真是一个灾难了。

Windows下玩 Linux 少不了虚拟机，但装系统依旧是相当费事。在现在什么都在自动化的今天， Vagrant 就是这么一个神器，基于 Ruby开发，使用开源 VirtualBox 作为虚拟化支持，可以轻松的跨平台部署。

今天试着把几台机器的环境都换成 Vagrant ，很爽很顺畅，于是就试着整理了一下使用小结：

目前所选用的是当前的最新版本 Vagrant 1.2.7 （对应的 Vagrantfile V2版），VirtualBox 使用的是 4.2.16

准备工作：

下载安装 VirtualBox ：https://www.virtualbox.org/

下载安装 Vagrant ：http://www.vagrantup.com/

下载需要使用的 box ：

官方提供的范例： http://files.vagrantup.com/precise32.box

还可以在 http://www.vagrantbox.es/ 这里下载更多不同系统甚至是已经配置好环境直接可以用的 box，虽然可以直接在 Vagrant 直接使用网址，由 Vagrant 自动下载安装，但是考虑到网络情况，还是建议自行先下载好。

由于习惯用 CentOS，于是我就下了 CentOS6.3 x86_64 minimal ，这个 Box 根据个人情况进行选择即可。

设置环境：

安装完成并且重启好系统之后就可以开始设置环境了。

首先在本地创建好工作目录，并在命令行下切换到对应目录

view source
print?

```
1 vagrant box add base CentOS-6.3-x86_64-minimal.box
```

base 表示指定默认的 box，也可以为 box 指定名称，比如 centos63 ，使用 base 时，之后可以直接使用 vagrant init 进行初始化，如果自行指定名称，则初始化的时候需要指定 box 的名称。

CentOS-6.3-x86_64-minimal.box 是 box 对应的文件名，这里可以是本地保存 box 的路径，也可以是可以下载 box 的网址，如果是网址的话， Vagrant 会自动启动下载。

view source
print?

```
1 [vagrant] Downloading with Vagrant::Downloaders::File...
2 [vagrant] Copying box to temporary location...
3 [vagrant] Extracting box...
4 [vagrant] Verifying box...
5 [vagrant] Cleaning up downloaded box...
```

设置好 box 之后，在当前工作目录运行

view source
print?

```
1 vagrant init
```

生成对应的 Vagrantfile 。通过文本编辑器打开 Vagrantfile 可以进行一些进一步的常用配置：

网络配置：

Vagrant 的网络有三种模式

1、较为常用是端口映射，就是将虚拟机中的端口映射到宿主机对应的端口直接使用 ，在 Vagrantfile 中配置：

view source
print?

```
1 config.vm.network :forwarded_port, guest: 80, host: 8080
```

guest: 80 表示虚拟机中的 80 端口， host: 8080 表示映射到宿主机的 8080 端口。

2、如果需要自己自由的访问虚拟机，但是别人不需要访问虚拟机，可以使用 private_network ，并为虚拟机设置 IP ，在 Vagrantfile 中配置：

view source
print?
```
1 config.vm.network :private_network, ip: "192.168.1.104"
```

192.168.1.104　表示虚拟机的 IP，多台虚拟机的话需要互相访问的话，设置在相同网段即可

3、如果需要将虚拟机作为当前局域网中的一台计算机，由局域网进行 DHCP 那么在 Vagrantfile 中配置：

view source
print?
```
1 config.vm.network :public_network
```

目录映射：

既然是开发环境， 那么开发工作肯定还是需要在本地完成， 而不是都要进到虚拟机中去完成， 虚拟机就好好在后台运行服务就好了， 不然就本末倒置了， 所以这里就需要使用目录映射功能，将本地的目录映射到虚拟机的对应目录。

默认情况下， 当前的工作目录， 会被映射到虚拟机的 /vagrant 目录，当前目录下的文件可以直接在 /vagrant 下进行访问，当然也可以在通过 ln 创建软连接，如

view source
print?
```
1 ln -fs /vagrant/wwwroot /var/www
```

来进行目录映射，当然，从自动化配置的角度，能不进系统就不需要进系统，所以在 Vagrant 也可以进行目录映射的操作：

view source
print?
```
1 config.vm.synced_folder "wwwroot/", "/var/www"
```

前面的参数 “ wwwroot/ ” 表示的是本地的路径，这里使用对于工作目录的相对路径，这里也可以使用绝对路径，比如： “ d:/www/ ”

后面的参数 “ /var/www ” 表示虚拟机中对应映射的目录。

运行脚本：

虽然不是必须， 但是如果有需要在启动时运行一些脚本 （环境的安装或者有些服务的启动需要在完成目录映射之后进行），可以编辑脚本，类似如下（摘自 Vagrant Document ）：

view source
print?

```
1 #!/usr/bin/env bash
2
3 apt-get update
4 apt-get install -y apache2
5 rm -rf /var/www
6 ln -fs /vagrant /var/www
```

保存在和 Vagrantfile 相同目录，文件名自取（如 boot.sh ），然后在 Vagrantfile 中添加：

view source
print?

```
1 config.vm.provision :shell, :path => "boot.sh"
```

当初次使用基本的设置都完成则之后，则可以使用 vagrant up 启动虚拟机

view source
print?

```
01 Bringing machine 'default' up with 'virtualbox' provider...
02 [default] Setting the name of the VM...
03 [default] Clearing any previously set forwarded ports...
04 [default] Creating shared folders metadata...
05 [default] Clearing any previously set network interfaces...
06 [default] Preparing network interfaces based on configuration...
07 [default] You are trying to forward to privileged ports (ports < = 1024). Most operating systems restrict this to only privileged process (typicallyprocesses running as an administrative user). This is a warning in case
08 the port forwarding doesn't work. If any problems occur, please try a port higher than 1024.
09 [default] Forwarding ports...
10 [default] -- 22 => <strong>2222</strong> (adapter 1)
11 [default] -- 80 => 8080 (adapter 1)
12 [default] Booting VM...
```

13 [default] Waiting for VM to boot. This can take a few minutes.

14 [default] VM booted and ready for use!

[default] The guest additions on this VM do not match the installed version of VirtualBox! In most cases this is fine, but in rare cases
15 it can cause things such as shared folders to not work properly. If you see shared folder errors, please update the guest additions within the virtual machine and reload your VM.

16

17 Guest Additions Version: 4.1.18

18 VirtualBox Version: 4.2

19 [default] Mounting shared folders...

20 [default] -- /var/www

21 [default] -- /vagrant

22 [default] Running provisioner: shell...

虚拟机启动之后则可以通过 vagrant ssh 联入虚拟机进行进一步的环境配置，或者软件安装相关的工作， 在 Windows系统下，并不能直接通过 vagrant ssh 连到虚拟机，需要使用 Putty ，Xshell 等第三方工具进行连接

连接的 IP 和端口根据网络环境配置的不同有所不同，如果是默认使用端口映射的话，一般是连接 本地的 2222 端口。

登录的帐号密码均为 vagrant ，登录之后如果需要 su root ，密码也是 vagrant

注：使用 vagrant ssh 时，会提示可以使用密钥进行登录， 如果需要使用 putty 进行密钥登录的话， 需要下载 puttygen 将 ssh 的密钥转换为 ppk 文件才能使用。

登录 ssh 完成环境的配置，如果在开发环境中使用 webserver （nginx/apache ）为了避免一些静态文件处理的问题，可能还需要进行一些额外的配置： Vagrant 下共享目录静态文件 (js/jpg/png 等）“ 缓存 ” 问题

在不进入虚拟机的情况下，还可以使用下面的命令对 虚拟机进行管理：

vagrant up （启动虚拟机）
vagrant halt （关闭虚拟机——对应就是关机）
vagrant suspend （暂停虚拟机——只是暂停，虚拟机内存等信息将以状态文件的方式保存在本地，可以执行恢复操作后继续使用）
vagrant resume （恢复虚拟机 —— 与前面的暂停相对应）
vagrant destroy （删除虚拟机，删除后在当前虚拟机所做进行的除开 Vagrantfile 中的配置都不会保留）

当在启动 Vagrant 后，对于虚拟机有进行过安装环境相关的配置，　如果并不希望写在 Vagrant 的启动 shell　里面每次都重新安装配置一遍，　可以将当前配置好的虚拟机打包成 box，

view source
print?
```
1 vagrant package --output NAME --vagrantfile FILE
2
3 可选参数：
4
5 --output NAME　：　（可选）设置通过　NAME来指定输出的文件名
6
7 --vagrantfile FILE　　：（可选）可以将　Vagrantfile　直接封进 box 中
```

注：如果网络模式中使用　private_network　的话，在打包之前需要清除一下 private_network　的设置，避免不必要的错误：

view source
print?
```
1 sudo rm -f /etc/udev/rule.d/70-persistent-net.rules
```

制作完成之后直接将　box 文件拿到其他计算机上配置即可使用。

更多信息可以参考官方文档：　http://docs.vagrantup.com/v2/