→ VAGRANT

- INSTALL VAGRANT
- INITIALIZE A PROJECT DIRECTORY:

```
$ MKDIR VAGRANT_GETTING_STARTED
$ CD VAGRANT_GETTING_STARTED
$ VAGRANT INIT HASHICORP/BIONIC64
```

- INSTALL AND SPECIFY A BOX:

  * IF YOU RAN THE INIT COMMAND ABOVE, YOU DO NOT NEED TO ADD A BOX

  $ VAGRANT ~~ADD~~ BOX ADD HASHICORP/BIONIC64

  • CONFIGURE YOUR PROJECT TO USE IT AS BASE:

  OPEN THE VAGRANTFILE AND REPLACE THE CONTENTS WITH THE FOLLOWING:

```
VAGRANT.CONFIGURE("2") DO |CONFIG|
    CONFIG.VM.BOX = "HASHICORP/BIONIC64"
END
```

```
$ ECHO 'CONTENT ABOVE' > VAGRANTFILE
$ CAT VAGRANTFILE
```

- BOOT AN ENVIRONMENT:

```
$ VAGRANT UP
```

! KERNEL DRIVER NOT INSTALLED (RC=1908)

* MAC USERS MIGHT RUN INTO THIS ISSUE, BUT THERE IS A EASY SOLUTION UNINSTALL AND REINSTALL YOUR VIRTUALBOX, → ACCESS YOUR SYSTEM PREFERENCES → SECURITY & PRIVACY → ALLOW "ORACLE AMERICA, INC" TO HAVE ACCESS TO YOUR SYSTEM

```
$ VAGRANT SSH          /HOME/VAGRANT
$ LOGOUT    OR    CTRL + D
$ VAGRANT DESTROY   → Y
$ VAGRANT BOX LIST
$ VAGRANT BOX REMOVE 'NAME'    EX: HASHICORP/BIONIC64
```

- SYNCHRONIZE LOCAL AND GUEST FILES:

```
$ LS /VAGRANT
```

* THE VAGRANTFILE IN YOUR VM IS THE SAME AS IN YOUR HOST MACHINE

```
$ TOUCH /VAGRANT/FOO
$ EXIT   → LOGOUT
$ LS VAGRANTFILE FOO
```

* NOW YOU CAN SUCCESSFULLY INTERACT WITH YOUR HOST MACHINE VIA SYNCED FOLDERS ON THE GUEST MACHIN

→ PROVISION A VIRTUAL MACHINE

YOU CAN SERVE THE SYNCED FILES ~~INTO~~ USING A WEBSERVER

- CREATE A DIRECTORY WHERE APACHE WILL LOOK FOR YOUR CONTENT

```
$ MKDIR HTML
```

- CREATE A FILE CALLED INDEX.HTML IN THE NEW DIRECTORY, AND POPULATE IT WITH THE CONTENT :

```
<!DOCTYPE HTML>
<HTML>
  <BODY>
    <H1> GETTING STARTED WITH VAGRANT ! </H1>
  </BODY>
</HTML>
```

- WRITE A PROVISIONING SCRIPT USING SHELL SCRIPT AND SAVE IT AS BOOTSTRAP.SH IN THE SAME DIRECTORY AS YOUR VARGANTFILE

```
#! /USR/BIN/ENV BASH

APT-GET UPDATE
APT-GET INSTALL -Y APACHE2
IF ! [ -L /VAR/WWW ]; THEN
  RM -RF /VAR/WWW
  LN -FS /VAGRANT /VAR/WWW
FI
```

*THIS SCRIPT WILL DOWNLOAD AND START APACHE, AND CREATE A SYMLINK BETWEEN YOUR SYNCED FILES DIRECTORY AND THE LOCATION WHERE APACHE WILL LOOK FOR CONTENT TO SERVE.

- CONFIGURE VAGRANT

```
VAGRANT.CONFIGURE ("2") DO |CONFIG|
    CONFIG.VM.BOX = "HASHICORP/BIONIC64"
    CONFIG.VM.PROVISION :SHELL, PATH: "BOOTSTRAP.SH"
END
```

- PROVISION THE WEBSERVER

```
$ VAGRANT RELOAD --PROVISION
$ VAGRANT SSH
$ WGET -QO- 127.0.0.1
$ LOGOUT
```

- CONFIGURE THE NETWORK :

NOW WE'RE GOING TO USE VAGRANT'S NETWORKING FEATURES TO PROVIDE ACCESS TO THE GUEST MACHINE FROM OUR HOST MACHINE.

- CONFIGURE PORT FORWARDING

```
VAGRANT.CONFIGURE ("2") DO |CONFIG|
    CONFIG.VM.BOX = "HASHICORP/BIONIC64"
    CONFIG.VM.PROVISION :SHELL, PATH: "BOOTSTRAP.SH"
    CONFIG.VM.NETWORK :FORWARDED_PORT, GUEST: 80, HOST: 4567
END
```

```
$ VAGRANT RELOAD
```

- ACCESS THE SERVED FILES

ONCE THE MACHINE IS RUNNING AGAIN, LOAD HTTP://127.0.0.1:4567 IN YOUR BROWSER, WHERE YOU WILL FIND A WEBPAGE THAT IS BEING SERVED FROM THE GUEST VIRTUAL MACHINE.

- SHARE AN ENVIRONMENT :

VAGRANT SHARE IS A PLUGIN THAT LETS YOU SHARE YOUR VAGRANT ENVIRONMENT TO ANYONE AROUND THE WORLD WITH AN INTERNET CONNECTION. IT WILL GIVE YOU A URL THAT WILL ROUTE DIRECTLY TO YOUR VAGRANT ENVIRONMENT FROM ANY DEVICE.

* NOT PRODUCTION-LEVEL TRAFFIC ; ONLY SHARE DEVELOPMENT OR Q/A ENVIRONMENTS

- PREREQUISITES : INSTALL NGROK

! "NGROK" CANNOT BE OPENED BECAUSE IT IS FROM AN UNDENTIFIED DEVELOPER.

SYSTEM PREFERENCES > SECURITY & PRIVACY > ALLOW "NGROK"

NGROK STEPS :

(1) UNZIP TO INSTALL OR RUN .EXE FILE $ SUDO NPM INSTALL -G NGROK

```
$ UNZIP /PATH/TO/NGROK.ZIP
```

(2) CONNECT TO YOUR ACCOUNT

```
./NGROK AUTHTOKEN <YOUR_AUTH_TOKEN>
```

! YOU MUST SIGNUP TO NGROK TO HAVE A AUTHTOKEN

(3) FIRE IT UP

```
./NGROK HELP
./NGROK HTTP 80    /TO START A HTTP TUNEL ON PORT 80
```

- INSTALL THE PLUGIN

```
$ ~~INSTALL~~ VAGRANT PLUGIN INSTALL VAGRANT-SHARE
$ VAGRANT SHARE      COPY URL OUTPUT
```

THE URL IS ROUTING DIRECTLY INTO YOUR VAGRANT ENVIRONMENT

```
$ ^C
```

- TEARDOWN AN ENVIRONMENT:

NOW THAT YOU'VE EXPLORED YOUR DEVELOPMENT ENVIRONMENT, IT IS TIME TO STOP, SHUT DOWN, AND FINALLY DESTROY THE ENVIRONMENT.

- SUSPEND THE MACHINE

```
$ VAGRANT SUSPEND  / WILL STOP AND SAVE ITS CURRENT RUNNING STATE
$ VAGRANT UP
```

- HALT THE MACHINE

```
$ VAGRANT HALT   / WILL GRACEFULLY SHUTDOWN THE GUEST OS AND POWER DOWN GUEST MACHINE
$ VAGRANT UP
```

- BESTROY THE MACHINE

```
$ VAGRANT DESTROY  / WILL REMOVE ALL TRACES OF THE GUEST MACHINE FROM YOUR SYSTEM. IT'LL
  > YES               STOP THE GUEST MACHINE, POWER IT DOWN, AND RECLAIM ITS DISK SPACE & RAM
```

- REBUILD AN ENVIRONMENT:

TO START A BOX AT ANY TIME, RUN:

```
$ VAGRANT UP
```

- EXPLORE OTHER PROVIDERS:

IN THESE TUTORIALS, YOUR PROJECT WAS BACKED WITH VIRTUALBOX. BUT VAGRANT CAN WORK WITH A VARIETY OF BACKEND PROVIDERS, SUCH AS VMWARE, HYPER-V, AND MORE.

- BOOT WITH THE NEW PROVIDER

```
$ VAGRANT UP --PROVIDER = VMWARE_DESKTOP  / NO NEED TO MODIFY YOUR VAGRANTFILE
```