

# Vagrant

4AHIT

03/03/2014

Christian Janeczek

Mair Wolfgang

# Inhaltsverzeichnis

|  |    |
|--|----|
| <b>Inhaltsverzeichnis</b> .....                  | 1  |
| <b>Aufgabenstellung</b> .....                    | 3  |
| AUTOMATISCHES VM PROVISIONING .....              | 3  |
| <b>Designüberlegung</b> .....                    | 4  |
| Installation von Vagrant .....                   | 4  |
| Verständnis für die Konfiguration aufbauen ..... | 4  |
| Testing des Tools anhand eines Examples .....    | 4  |
| Erweitern der Konfiguration .....                | 4  |
| <b>Technologie</b> .....                         | 5  |
| About Vagrant .....                              | 5  |
| <b>Arbeitsdurchführung</b> .....                 | 6  |
| Installation des Tools Vagrant .....             | 6  |
| Provisioning via Puppet(Janeczek) .....          | 9  |
| Provisioning via Chef (Mair) .....               | 14 |

# Aufgabenstellung

## *AUTOMATISCHES VM PROVISIONING*

Vagrant ist ein System, mit dem automatisch virtuelle Maschinen (für VirtualBox, VMWare, aber auch Cloud-Systeme wie AWS) erstellt und konfiguriert werden können.

Erstellen Sie eine Vagrant-Konfigurationsdatei, die ein Basis-System installiert (precise32 oder precise64). Konfigurieren Sie Vagrant so, dass auf diesem System ein Standard-Webserver mit Wordpress (d.h. apache, PHP, MySQL) betrieben wird. Dieser Webserver soll auch von ausserhalb ihres Gastsystems erreichbar sein, daher muss bei Vagrant auch eine Port-Weiterleitung konfiguriert werden.

Arbeiten Sie in 2-Personen-Teams zusammen, wobei die Provisionierung des Systems sowohl mit Chef als auch mit Puppet durchgeführt wird. Dokumentieren Sie die Übungsdurchführung.

# Designüberlegung

## *Installation von Vagrant*

- Vagrant installieren

## *Verständnis für die Konfiguration aufbauen*

- Getting Started Guide durchlesen und verstehen
- Nach den Konfigurationsanforderungen Ausschau halten

## *Testing des Tools anhand eines Examples*

- Die automatisierte Erstellung einer Virtual Machine anhand eines Beispiels testen, sodass Ergebnisse observiert und weitere Verwendung und Anpassung der Konfigurationsdatei geleistet wird.

## *Erweitern der Konfiguration*

- Die Vagrant-Konfigurationsdatei wird sofern geändert, dass die Aufgabenstellung möglichst erfüllt wird.

# Technologie

## About Vagrant

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

Vagrant was started in January 2010 by Mitchell Hashimoto. For almost three years, Vagrant was a side-project for Mitchell, a project that he worked on in his free hours after his full time job. During this time, Vagrant grew to be trusted and used by a range of individuals to entire development teams in large companies.

In November 2012, HashiCorp was formed by Mitchell to back the development of Vagrant full time. HashiCorp builds commercial additions and provides professional support and training for Vagrant.

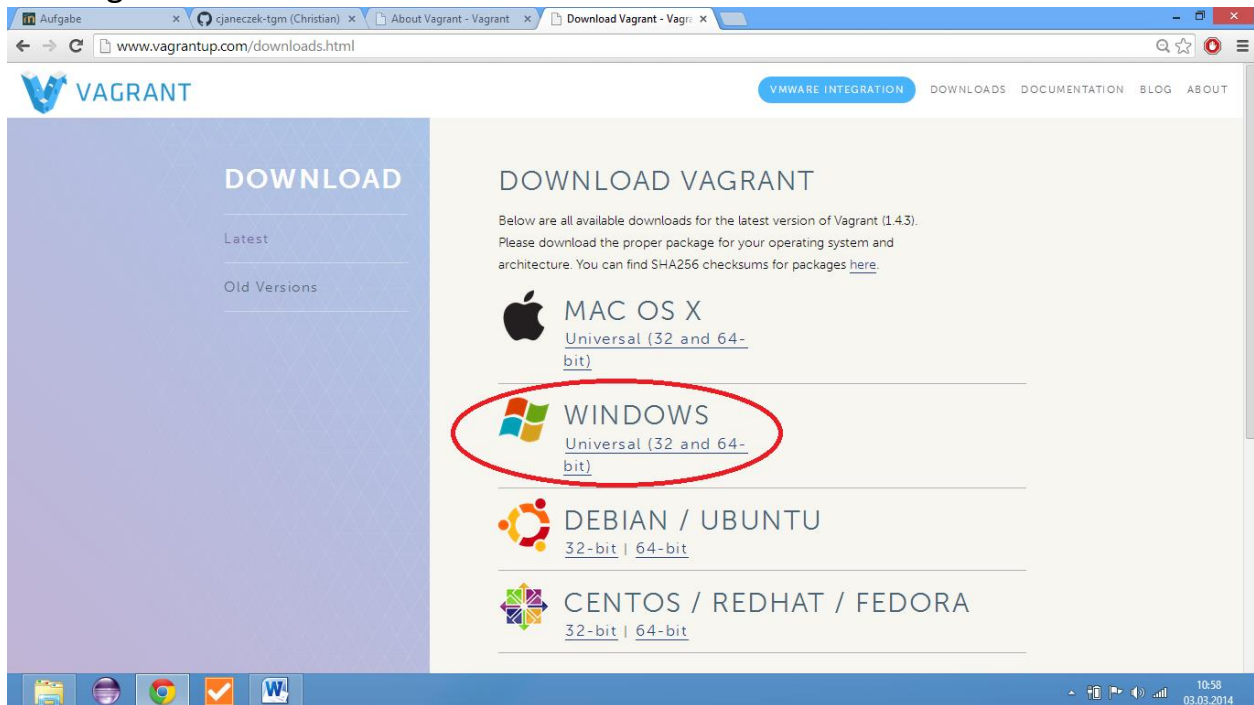
Vagrant remains and always will be a liberally licensed open source project. Each release of Vagrant is the work of hundreds of individuals' contributions to the open source project.

# Arbeitsdurchführung

## Installation des Tools Vagrant

Wir haben uns dafür entschieden, die Universal-Version x64bit des Tools „Vagrant“ für das Betriebssystem Windows zu installieren.

Für diesen Vorgang einfach die folgende Seite aufrufen und den Download Button betätigen:



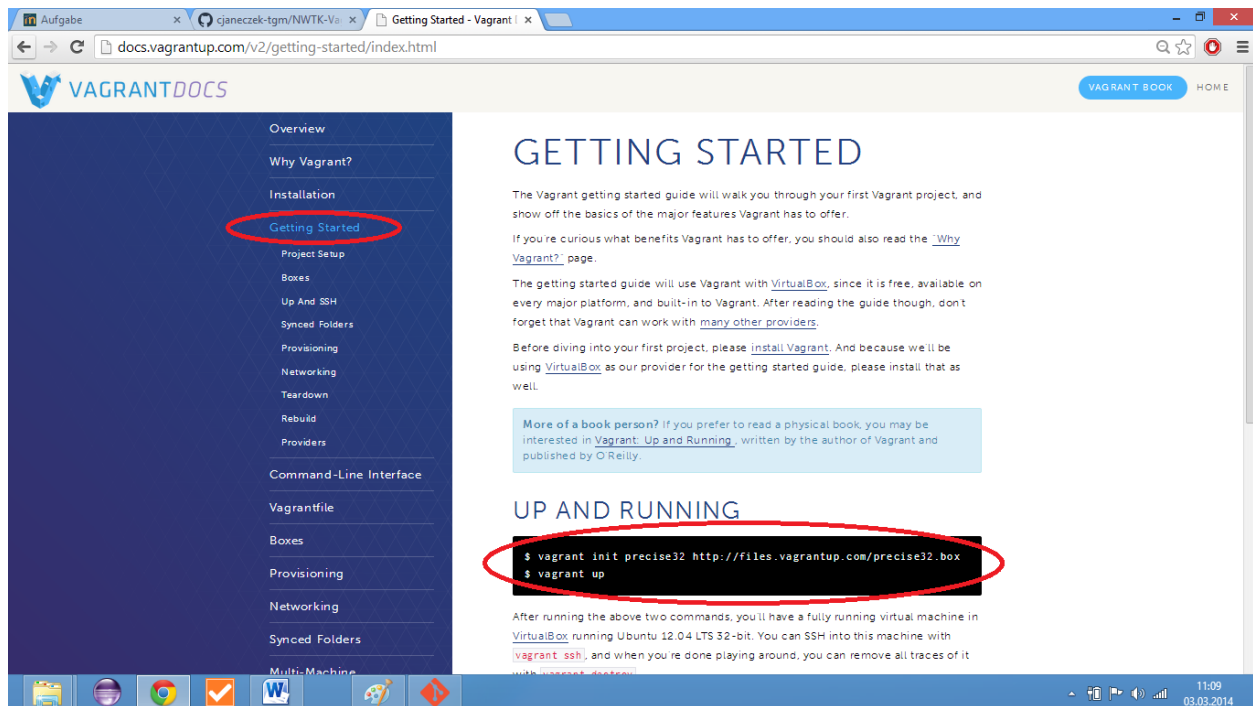
Bei der Installation immer die Standard-Eigenschaften wählen und nach der Installation muss ein Neustart des Systems erfolgen.

Wie in dem Guide mit dem Namen „Getting Started“ beschrieben müssen für den Vorgang „Up and Running“ Vagrant sowie VirtualBox installiert sein.

Zuerst muss der Download der precise64.box Datei erfolgen.

Danach kann das Vagrant Tool mit folgendem Befehl initialisiert werden:

```
C:\Users\Christian\Desktop\vagrant-nwtk>vagrant init precise64
A 'Vagrantfile' has been placed in this directory. You are now
ready to 'vagrant up' your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
'vagrantup.com' for more information on using Vagrant.
```



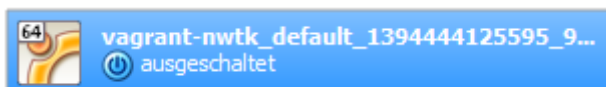
Das Tool wurde nun erfolgreich initialisiert.

Mit dem Befehl „vagrant up“ wird benötigt die precise64.box Datei für die VirtualBox zu importiert und gebootet.

```
C:\Users\Christian\Desktop\vagrant-nwtk>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
[default] Importing base box 'precise64'...
[default] Matching MAC address for NAT networking...
[default] Setting the name of the VM...
[default] Clearing any previously set forwarded ports...
[default] Clearing any previously set network interfaces...
[default] Preparing network interfaces based on configuration...
[default] Forwarding ports...
[default] -- 22 => 2222 (adapter 1)
[default] Booting VM...
[default] Waiting for machine to boot. This may take a few minutes...
DL is deprecated, please use Fiddle
[default] Machine booted and ready!
[default] Mounting shared folders...
[default] -- /vagrant
```

Wie man  
bereits

erkennen kann wurde eine Virtuelle Instanz generiert, jedoch mit einem gegebenen Grafikspeicher von 8MB(ziemlich wenig). Diesen haben wir zu 128MB geändert um eine erleichterte Bedienung zu ermöglichen.



Die Zugangsdaten sind(Nur für die Testung notwendig):

Username: **vagrant**

Passwort: **vagrant**

Um die Netzwerkerreichbarkeit zu gewährleisten muss das Vagrantfile folgendermaßen angepasst werden:

```
# Create a forwarded port mapping which allows access to a specific
port
# within the machine from a port on the host machine. In the example
below,
# accessing "localhost:8080" will access port 80 on the guest machine.
config.vm.network :forwarded_port, guest: 80, host: 8080
```

Localhost kann über den Port 80 angesprochen werden.

```
# Create a public network, which generally matched to bridged network.
# Bridged networks make the machine appear as another physical device
on
# your network.
config.vm.network :public_network
```

Erstellt einen Bridged-Network Adapter, welcher automatisch eine IPv4 und IPv6 Adresse des Hostnetzwerkes zur Verfügung gestellt bekommt.

```
C:\Users\Christian\Desktop\vagrant-nwtk>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
[default] Clearing any previously set forwarded ports...
[default] Clearing any previously set network interfaces...
[default] Preparing network interfaces based on configuration...
[default] Forwarding ports...
[default] -- 22 => 2222 (adapter 1)
[default] -- 80 => 8080 (adapter 1)
[default] Booting VM...
[default] Waiting for machine to boot. This may take a few minutes...
DL is deprecated, please use Fiddle
[default] Machine booted and ready!
[default] Configuring and enabling network interfaces...
[default] Mounting shared folders...
[default] -- /vagrant
[default] VM already provisioned. Run 'vagrant provision' or use '--provision' to force it
```

Mit einem vagrant up wird die Virtuelle Instanz den neuen Einstellungen angepasst.



## Provisioning via Puppet(Janeczek)

Wie in der Aufgabenstellung beschrieben, sollen die Dienste apache, mySQL sowie php auf der Virtuellen Instanz vorinstalliert sein.

Für diesen Prozess verwende ich das Plug-in „Puppet“.

Folgende Zeilen wurden im Vagrantfile auskommentiert:

```
config.vm.provision :puppet do |puppet|
  puppet.manifests_path = "manifests"
  puppet.module_path = "modules"
  puppet.manifest_file = "site.pp"
end
```

Im neu-erstellten Ordner mit dem Namen „manifests“ muss nun ein File „site.pp“ erstellt werden, welches auf unsere html Files verweisen soll.

Der Inhalt von **manifests/site.pp** ist:

```
exec { "apt-get update":
  path => "/usr/bin",
}
package { "apache2":
  ensure => present,
  require => Exec["apt-get update"],
}
service { "apache2":
  ensure => "running",
  require => Package["apache2"],
}
file { "/var/www/syt-app":
  ensure => "link",
  target => "/vagrant-nwtk/syt-app",
  require => Package["apache2"],
  notify => Service["apache2"],
}
```

Der Dienst Apache2 wird nach einem apt-get update installiert und das File im Verzeichnis **/vagrant-nwtk/syt-app** wird via Apache2 als index.html festgelegt.

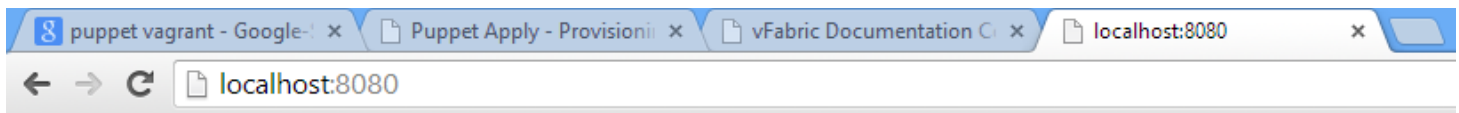
Beim Starten der automatisiert-erstellten Virtuellen Instanz kann man nun auf die Webapplikation zugreifen:

```

C:\Users\Christian\Desktop\vagrant-nwtk>notepad Vagrantfile

C:\Users\Christian\Desktop\vagrant-nwtk>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
[default] Clearing any previously set forwarded ports...
[default] Clearing any previously set network interfaces...
[default] Available bridged network interfaces:
1) Qualcomm Atheros AR5BWB222 Wireless Network Adapter
2) TAP-Windows Adapter V9
What interface should the network bridge to? 1
[default] Preparing network interfaces based on configuration...
[default] Forwarding ports...
[default] -- 22 => 2222 (adapter 1)
[default] -- 80 => 8080 (adapter 1)
[default] Booting VM...
[default] Waiting for machine to boot. This may take a few minutes...
DL is deprecated, please use Fiddle
[default] Machine booted and ready!
[default] Configuring and enabling network interfaces...
[default] Mounting shared folders...
[default] -- /vagrant
[default] -- /tmp/vagrant-puppet-1/manifests
[default] -- /tmp/vagrant-puppet-1/modules-0
[default] VM already provisioned. Run 'vagrant provision' or use '--provision' to force it

```



## It works!

This is the default web page for this server.

The web server software is running but no content has been added, yet.

Die Virtuelle Instanz wird nun automatisch mit Erreichbarkeit und einem vorinstallierten Apache2 Service generiert.

Für die Installation von MySQL, PHP und Wordpress müssen folgende Files erstellt werden:

- mysql.pp
- php.pp
- wordpress.pp

Der Inhalt der **mysql.pp** ist folgender:

```
package { "mysql-server":  
  ensure => present,  
  require => Exec["apt-get update"],  
}  
service { "mysql":  
  ensure => "running",  
  require => Package["mysql-server"],  
}
```

Der Inhalt der **php.pp** ist folgender:

```
package { "php5":  
  ensure => present,  
  require => Exec["apt-get update"],  
}
```

Der Inhalt der **wordpress.pp** ist folgender:

```
package { "wordpress":  
  ensure => present,  
  require => Exec["apt-get update"],  
}
```

Der Inhalt der **site.pp**, welche alle Installationen zusammenfasst muss folgendermaßen angepasst werden:

```
import "mysql.pp"  
import "php.pp"  
import "wordpress.pp"  
  
exec { "apt-get update":  
  path => "/usr/bin",  
}  
package { "apache2":  
  ensure => present,  
  require => Exec["apt-get update"],  
}  
service { "apache2":  
  ensure => "running",  
  require => Package["apache2"],  
}  
file { "/var/www/syt-app":  
  ensure => "link",  
  target => "/vagrant-nwtk/syt-app",  
}
```

```
require => Package["apache2"],  
notify  => Service["apache2"],  
}
```

Die import-Befehle importieren den Inhalt der jeweiligen .pp Dateien und diesen werden weitergehend ausgeführt.

```
vagrant@precise64:~$ sudo apt-get install apache2  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
apache2 is already the newest version.  
0 upgraded, 0 newly installed, 0 to remove and 159 not upgraded.
```

```
vagrant@precise64:~$ sudo apt-get install php5  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
php5 is already the newest version.  
0 upgraded, 0 newly installed, 0 to remove and 159 not upgraded.
```

```
vagrant@precise64:~$ sudo apt-get install mysql-server  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
mysql-server is already the newest version.  
0 upgraded, 0 newly installed, 0 to remove and 159 not upgraded.
```

```
vagrant@precise64:~$ sudo apt-get install wordpress  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
wordpress is already the newest version.  
0 upgraded, 0 newly installed, 0 to remove and 159 not upgraded.
```

Man muss in das Verzeichnis **/usr/share/doc/wordpress/examples** switchen und dort den **setup-mysql** Befehl ausführen.

Dies läuft folgendermaßen:

```
sudo bash setup-mysql -n wordpress localhost
```

Danach muss der Inhalt des Verzeichnis **/usr/share/wordpress** nach **/var/www** kopiert werden. Folgender Befehl ist von Nöten:

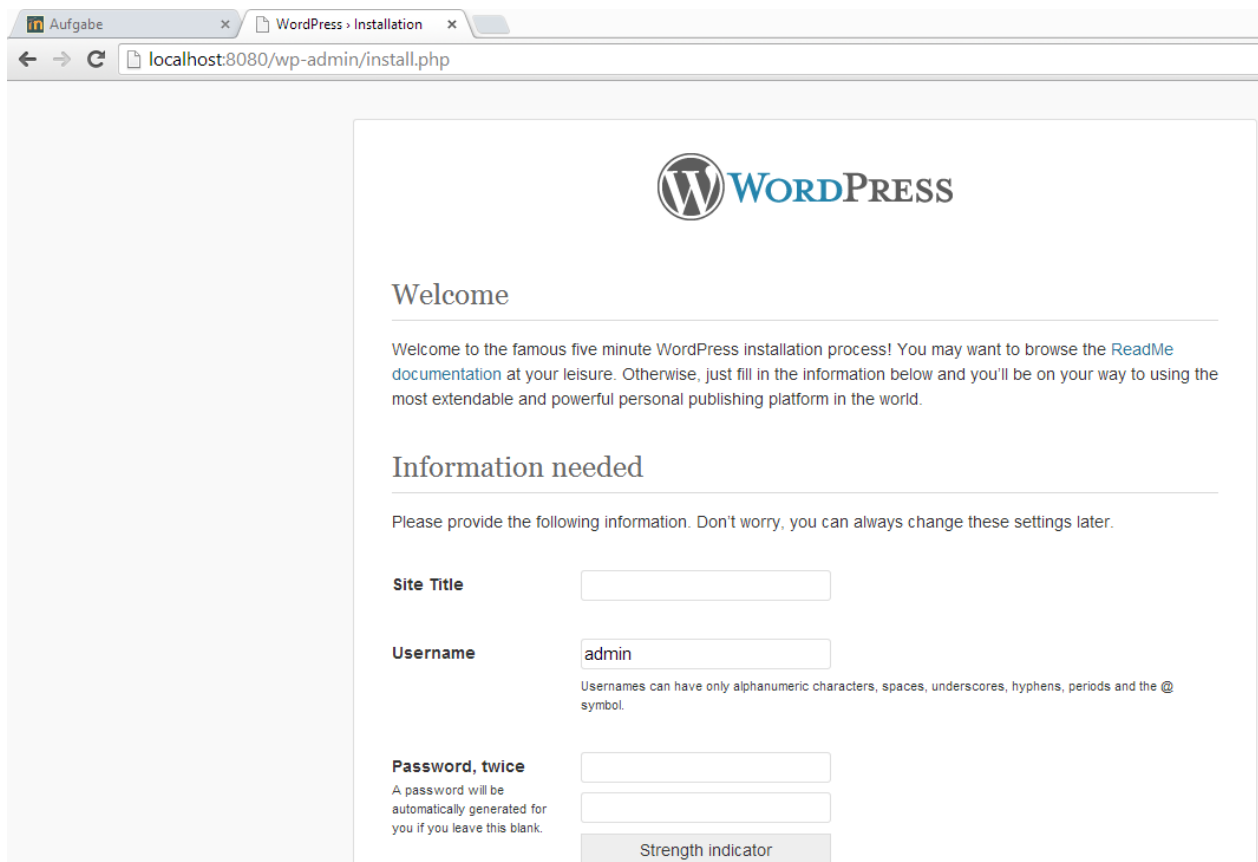
```
sudo cp -R * /var/www
```

Desweiteren muss die Default index.html Datei gelöscht werden, sodass die index.php Datei automatisch gewählt wird.

Der komplette Inhalt von /var/www sieht dann so aus:


```
vagrant@precise64:/usr/share/wordpress$ ls
index.php          wp-comments-post.php  wp-links-opml.php  wp-settings.php
readme.html        wp-config.php         wp-load.php        wp-signup.php
wp-activate.php    wp-config-sample.php  wp-login.php       wp-trackback.php
wp-admin           wp-content            wp-mail.php        xmlrpc.php
wp-app.php         wp-cron.php           wp-pass.php
wp-blog-header.php wp-includes           wp-register.php
vagrant@precise64:/usr/share/wordpress$ s_
```

Ein einfacher Zugriff des Host-Systems auf das Gast-System erfolgt mit **localhost:8080** im Webbrowser.



Aufgabe x WordPress - Installation x

localhost:8080/wp-admin/install.php

 **WORDPRESS**

### Welcome

Welcome to the famous five minute WordPress installation process! You may want to browse the [ReadMe documentation](#) at your leisure. Otherwise, just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

### Information needed

Please provide the following information. Don't worry, you can always change these settings later.

**Site Title**

**Username**

Username can have only alphanumeric characters, spaces, underscores, hyphens, periods and the @ symbol.

**Password, twice**

A password will be automatically generated for you if you leave this blank.

Strength indicator

Success!!

## Provisioning via Chef (Mair)

### Inhalt des Vagrantfiles zum erstellen der VM

*# Vagrantfile API/syntax version. Don't touch unless you know what you're doing!*  
 VAGRANTFILE\_API\_VERSION = "2"

Vagrant.configure(VAGRANTFILE\_API\_VERSION) do |config|

*# Every Vagrant virtual environment requires a box to build off of.*  
 config.vm.box = "precise64"

*# The url from where the 'config.vm.box' box will be fetched if it  
 # doesn't already exist on the user's system.*  
 config.vm.box\_url = "http://files.vagrantup.com/precise64.box"

*# Create a forwarded port mapping which allows access to a specific port  
 # within the machine from a port on the host machine. In the example below,  
 # accessing "localhost:8080" will access port 80 on the guest machine.*  
 config.vm.network :forwarded\_port, guest: 80, host: 8080

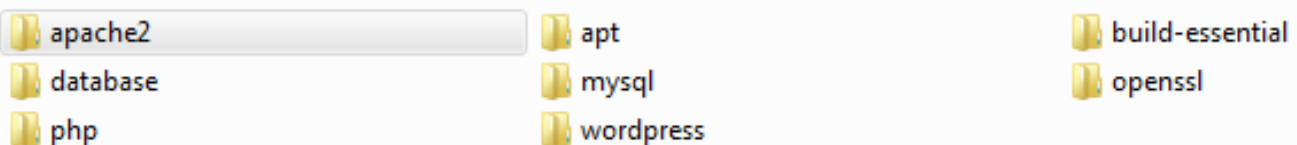
```
config.vm.provision "chef_solo" do |chef|
  chef.add_recipe "apt"
  chef.add_recipe "php"
  chef.add_recipe "apache2"
  chef.add_recipe "mysql::server"
  chef.add_recipe "wordpress"

  chef.json = {
    :mysql => {
      :server_root_password => "Passwort",
      :server_repl_password => "Passwort",
      :server_debian_password => "Passwort"
    }
  }
end
```

end

### Cookbooks die benötigt werden

Link zu den Cookbooks: <https://github.com/opscode-cookbooks>



## Wichtige Punkte auf die geachtet werden müssen

In Version 10.x von Vagrant muss bei Wordpress kleine Änderungen durchgeführt werden damit das automatisierte erstellen funktioniert.

Datei -> attributes/default.rb :

```
if platform_family?('windows') -> Löschen  
if node['platform'] == 'windows' -> Hinzufügen
```

Datei -> recipes/default.rb :

```
template "#{node['wordpress']['dir']}/wp-config.php" do  
  source 'wp-config.php.erb'  
+ mode 0644 -> Hinzufügen  
  variables(  
    :db_name => node['wordpress']['db']['name'],  
    :db_user => node['wordpress']['db']['user'],
```

Auch zu finden unter den Link:

<https://github.com/brint/wordpress-cookbook/pull/27/files>

Außerdem gibt es auch Probleme mit dem build-essential Cookbook das für WordPress benötigt wird. Dieses enthält nämlich in der aktuellen Version einen Fehler. Es sollte daher Version 1.4.4 von build-essential verwendet werden (git checkout c1f1166).

## Testen der erstellten VM

Beim öffnen der VM kann man ganz einfach mittels des suchen nach den Programmen feststellen ob das Installieren erfolgreich war.

```
vagrant@precise64:/$ whereis apache2
apache2: /usr/sbin/apache2 /etc/apache2 /usr/lib/apache2 /usr/share/apache2 /usr/share/man/man8/apache2.8.gz
```

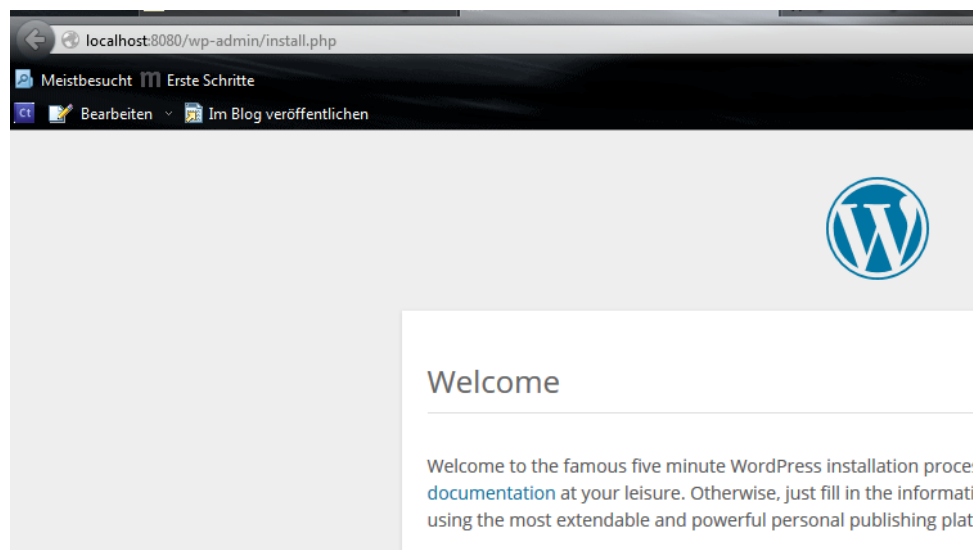
Das Ergebniss wenn man nach Apache2 sucht. Wie man sieht ist es vorhanden und installiert.

```
vagrant@precise64:/var/www/wordpress$ ls
index.php      wp-blog-header.php  wp-cron.php      wp-mail.php
license.txt    wp-comments-post.php wp-includes       wp-settings.php
readme.html    wp-config.php       wp-links-opml.php wp-signup.php
wp-activate.php wp-config-sample.php wp-load.php       wp-trackback.php
wp-admin       wp-content          wp-login.php      xmlrpc.php
```

Eine weitere Möglichkeit ist in den Installierten Ordner des Programmes nachzuschauen ob alle Dateien vorhanden sind. Wie man sieht ist Wordpress vorhanden.

```
vagrant@precise64:~$ whereis mysql
mysql: /usr/bin/mysql /etc/mysql /usr/lib/mysql /usr/include/mysql /usr/share/mysql /usr/share/man/man1/mysql.1.gz
```

Auch Mysql wurde erfolgreich installiert, wie man bei dem Bild oben erkennen kann.



Beim aufrufen des Localhosts von dem Home-PC erkennt man das alles funktioniert.



## Analyse des Vagrantfiles

*# Vagrantfile API/syntax version. Don't touch unless you know what you're doing!*

```
VAGRANTFILE_API_VERSION = "2"
```

Legt die API Version des Vagrantfiles fest.

```
Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
```

Start der Konfiguration

*# Every Vagrant virtual environment requires a box to build off of.*

```
config.vm.box = "precise64"
```

Zeigt welches System automatisch generiert werden soll, hier ist es 64-Bit

*# The url from where the 'config.vm.box' box will be fetched if it  
# doesn't already exist on the user's system.*

```
config.vm.box_url = "http://files.vagrantup.com/precise64.box"
```

Beschreibt wo die box herunterzuladen ist falls sie noch nicht existiert

*# Create a forwarded port mapping which allows access to a specific port  
# within the machine from a port on the host machine. In the example below,  
# accessing "localhost:8080" will access port 80 on the guest machine.*

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

Erlaubt das das Hostsystem die Virtual Machine mittels Localhost:8080 aufrufen kann

```
config.vm.provision "chef_solo" do |chef|
```

```
  chef.add_recipe "apt"
```

```
    chef.add_recipe "php"
```

```
  chef.add_recipe "apache2"
```

```
    chef.add_recipe "mysql::server"
```

```
    chef.add_recipe "wordpress"
```

Inkludiert die Cookbooks die zu der Konfiguration benötigt werden

```
  chef.json = {
```

```
    :mysql => {
```

```
      :server_root_password => "Passwort",
```

```
      :server_repl_password => "Passwort",
```

```
      :server_debian_password => "Passwort"
```

```
    }
```

Setzen der Passwörter für Mysql, da sie nicht automatisch generiert werden

```
end
```

```
end
```

Schließen der Konfigurationsaufgaben