# READY TO CODE

Automate Your Development Environments

# READY TO CODE/HIRE

## Automate Your Development Environments

# READY TO CODE/HIRE/TEST

Automate Your Environments

# READY TO CODE/HIRE/TEST/DEPLOY

Automate Your Environments

```
dave_kerber = {name: ‘Dave Kerber’,
               wife: ‘Abbie Kerber’,
               kids: [‘Katherine’,
                      ‘Elizabeth’,
                      ‘James (July 2013)’],
               twitter: ‘davidjkerber’,
               company: ‘Agape Red’,
               shirt_color: ‘red’}
```

# THE GOAL

- You leave this talk equipped with the knowledge to setup your own virtualized environments

# WELCOME TO YOUR FIRST DAY

# THE PROBLEM:

Starting on existing companies, existing projects, or new computers is hard:

- undocumented dependencies
- unclear installation procedures
- long, tedious and brittle documentation

# WE ALREADY LOVE (SERVER) VIRTUALIZATION!

- It is faster and quicker to create new servers
- Higher usage utilization
- Cheap temporary environments
- Worry free upgrades
- Automated creation and destruction of resources as needed

# WHY DON’T WE LOVE DEV ENVIRONMENT VIRTUALIZATION?

# VIRTUALIZATION ROUND 1

## Shared Virtual Box Images with Full Ubuntu Install

# VIRTUALIZATION ROUND 1 – THE GOOD

- Bringing new people on board is easy
- No need for complex documentation
- The computer becomes a delivery mechanism
- Consistency of development environment (it’s always Linux)

# VIRTUALIZATION ROUND 1 – THE BAD

- Needs LOTS of disk space
- Difficult to share
- Upgrades are hard
- It sometimes shares too much
- Context switching between operating systems
- The code was in the Virtual Machine

# VIRTUALIZATION ROUND 2

VAGRANT!

# VAGRANT

- Lightweight, sharable virtual machine configuration
- Works with VirtualBox and VMWare, and AWS EC2
- Configure Virtual Machines with shell scripts, Chef, or Puppet

# HOW IT WORKS

- Boxes – A base image that your VM is cloned from
- Vagrantfile – A Ruby file that initiates all your configuration

# INSTALL IT!

- Vagrant- http://downloads.vagrantup.com
- Virtual Box - https://www.virtualbox.org/wiki/Downloads
- vagrant-vbguest - https://github.com/dotless-de/vagrant-vbguest

# LETS BUILD A VM

vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm

~~vagrant init~~

mate Vagrantfile

# LETS BUILD A VM

vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm

~~vagrant init~~

mate Vagrantfile

# LETS BUILD A VM

```
vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm

~~vagrant init~~

mate Vagrantfile
```

# LETS BUILD A VM

vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm

~~vagrant init~~

mate Vagrantfile

# LETS BUILD A VM

```
vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm

~~vagrant init~~

mate Vagrantfile
```

# LETS BUILD A VM

```
vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm
```

~~vagrant init~~

```
mate Vagrantfile
```

# LETS BUILD A VM

```
vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm
```

~~vagrant init~~

```
mate Vagrantfile
```

# LETS BUILD A VM

vagrant box add precise64 http://files.vagrantup.com/precise64.box

mkdir my_first_vm

cd my_first_vm

~~vagrant init~~

mate Vagrantfile

```
Vagrant.configure("2") do |config|
    config.vm.box = "precise64"
    config.vm.box_url = "http://files.vagrantup.com/precise64.box"
end
```

```
Vagrant.configure("2") do |config|
    config.vm.box = "precise64"
    config.vm.box_url = "http://files.vagrantup.com/precise64.box"
end
```

```
Vagrant.configure("2") do |config|
  config.vm.box = "precise64"
  config.vm.box_url = "http://files.vagrantup.com/precise64.box"
end
```

```
Vagrant.configure("2") do |config|
    config.vm.box = "precise64"
    config.vm.box_url = "http://files.vagrantup.com/precise64.box"
end
```

# NOW YOU HAVE A WORKING VIRTUAL MACHINE!

run `vagrant ssh` to ssh into the box

# NOW LETS TWEAK SOME THINGS

```
Vagrant.configure("2") do |config|
 config.vm.box = "precise64"
 config.vm.box_url = "http://files.vagrantup.com/precise64.box"
 config.vm.provider :virtualbox do |vb|
  vb.customize ["modifyvm", :id, "--memory", "512", "--cpus", "2"]
 end
end
```

# HOW DO I GET MY CODE ON THERE?

```
Vagrant.configure("2") do |config|
 config.vm.box = "precise64"
 config.vm.box_url = "http://files.vagrantup.com/precise64.box"
 config.vm.provider :virtualbox do |vb|
   vb.customize ["modifyvm", :id, "--memory", "512", "--cpus", "2"]
 end
 config.vm.synced_folder "/Users/dkerber/", "/host_folder"
end
```

# INSIDE THE HOST MACHINE

```
Dave-Kerbers-MacBook-Pro-Of-Doom:~ dkerber$ ls
Desktop             Phone to Mac Transfers   projects
Documents           PhpstormProjects         rvm
...
```

# INSIDE THE VM

```
vagrant@precise64:~$ cd /host_folder/
vagrant@precise64:/host_folder$ ls
Desktop             Phone to Mac Transfers   projects
Documents           PhpstormProjects         rvm
...
```

# NOW LETS TALK TO IT

```
Vagrant.configure("2") do |config|
 config.vm.box = "precise64"
 config.vm.box_url = "http://files.vagrantup.com/precise64.box"
 config.vm.provider :virtualbox do |vb|
   vb.customize ["modifyvm", :id, "--memory", "512", "--cpus", "2"]
 end
 config.vm.synced_folder "/Users/dkerber/", "/host_folder"
 config.vm.network :forwarded_port, guest: 3000, host: 3000
end
```

# HOW DO WE CONFIGURE IT?

# CONFIGURATIONS OPTIONS

- shell - Shell scripting
- puppet - Automated configuration management software
  - http://puppetlabs.com
- chef - Automation platform for managing infrastructure
  - http://www.opscode.com/chef/

# SHELL PROVISIONING

```
Vagrant.configure("2") do |config|
  ...
  config.vm.network :forwarded_port, guest: 3000, host: 3000
  config.vm.provision :shell, :inline => %Q{
    apt-get install -y vim git
    su - vagrant
    /usr/bin/git config -f /home/vagrant/.gitconfig color.ui true
  }
end
```

```
Vagrant.configure("2") do |config|
 ...
 config.vm.network :forwarded_port, guest: 3000, host: 3000
 config.vm.provision :shell, :inline => %Q{
   apt-get install -y vim git
   su - vagrant
   /usr/bin/git config -f /home/vagrant/.gitconfig color.ui true
 }
end
```

# INLINE SHELLS? GROSS!

**setup.sh**

```
apt-get install -y vim git
su - vagrant
/usr/bin/git config -f /home/vagrant/.gitconfig color.ui true
```

**Vagrantfile**

```
Vagrant.configure("2") do |config|
 ...

 config.vm.provision :shell, :path => 'setup.sh'
end
```

**setup.sh**

```
apt-get install -y vim git
su - vagrant
/usr/bin/git config -f /home/vagrant/.gitconfig color.ui true
```

**Vagrantfile**

```
Vagrant.configure("2") do |config|
 ...

config.vm.provision :shell, :path => 'setup.sh'
end
```

# PUPPET PROVISIONING

```
Vagrant.configure("2") do |config|

 ...

 config.vm.network :forwarded_port, guest: 3000, host: 3000

 config.vm.provision :puppet do |puppet|
   puppet.manifests_path = "manifests"
   puppet.manifest_file  = "base.pp"
 end

end
```

```
Vagrant.configure("2") do |config|

  ...

  config.vm.network :forwarded_port, guest: 3000, host: 3000

  config.vm.provision :puppet do |puppet|
    puppet.manifests_path = "manifests"
    puppet.manifest_file  = "base.pp"
  end

end
```

```
Vagrant.configure("2") do |config|

  ...

  config.vm.network :forwarded_port, guest: 3000, host: 3000

  config.vm.provision :puppet do |puppet|
    puppet.manifests_path = "manifests"
    puppet.manifest_file  = "base.pp"
  end

end
```

**manifests/base.pp**

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config –f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

## manifests/base.pp

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config -f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

**manifests/base.pp**

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config –f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

## manifests/base.pp

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config –f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

## manifests/base.pp

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config –f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

## manifests/base.pp

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config –f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

**manifests/base.pp**

```
package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config –f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}
```

# WHAT ABOUT RUBY?

# PUPPET MODULES

- Puppet Modules are self contained bundles of code and data.
- Typically kept in the ‘modules’ folder

# RVM MODULE

- Use the RVM puppet module to install RVM, Rubys, and Gems
- https://github.com/blt04/puppet-rvm ( https://github.com/davekerber/puppet-rvm )
- install:

  ```
  mkdir modules
  git submodule add git://github.com/davekerber/puppet-rvm.git modules/rvm
  ```

# RVM MODULE

- Use the RVM puppet module to install RVM, Rubys, and Gems

- https://github.com/blt04/puppet-rvm ( https://github.com/davekerber/puppet-rvm )

- install:

  ```
  mkdir modules
  git submodule add git://github.com/davekerber/puppet-rvm.git modules/rvm
  ```

# RVM MODULE

- Use the RVM puppet module to install RVM, Rubys, and Gems
- https://github.com/blt04/puppet-rvm ( https://github.com/davekerber/puppet-rvm )
- install:
  mkdir modules
  git submodule add git://github.com/davekerber/puppet-rvm.git modules/rvm

```
Vagrant.configure("2") do |config|

 ...

 config.vm.provision :shell, :inline => 'apt-get update'

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

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

```
Vagrant.configure("2") do |config|

 ...

 config.vm.provision :shell, :inline => 'apt-get update'

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

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

**manifests/base.pp**

```
include rvm

package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config -f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}

rvm::system_user { vagrant: ; }
```

**manifests/base.pp**

```
include rvm

package { ['vim','git'] :
  ensure => present
}

exec { 'configure git' :
  command => "/usr/bin/git config -f /home/vagrant/.gitconfig color.ui true ; ",
  user => 'vagrant',
  cwd => '/home/vagrant',
  require => [Package['git']]
}

rvm::system_user { vagrant: ; }
```

```
Vagrant.configure("2") do |config|

  ...

  config.vm.provision :shell, :inline => 'apt-get update'

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

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

**manifests/rvm.pp**

```
include rvm

if $rvm_installed == "true" {
    rvm_system_ruby { 'ruby-1.9.3-p392':
      ensure => 'present',
      default_use => true;
    }

    ...
}
```

**manifests/rvm.pp**

```
include rvm

if $rvm_installed == "true" {
    rvm_system_ruby { 'ruby-1.9.3-p392':
      ensure => 'present',
      default_use => true;
    }

    ...
}
```

**manifests/rvm.pp**

```
include rvm

if $rvm_installed == "true" {
    rvm_system_ruby { 'ruby-1.9.3-p392':
      ensure => 'present',
      default_use => true;
    }

    ...
}
```

## manifests/rvm.pp

```
include rvm

if $rvm_installed == "true" {
      rvm_system_ruby { 'ruby-1.9.3-p392':
        ensure => 'present',
        default_use => true;
      }

      ...
}
```

**manifests/rvm.pp**

```
include rvm

if $rvm_installed == "true" {
      rvm_system_ruby { 'ruby-1.9.3-p392':
        ensure => 'present',
        default_use => true;
      }

      ...
}
```

## manifests/rvm.pp

```
...
if $rvm_installed == "true" {
    ....
    rvm_gem {
      'ruby-1.9.3-p392@global/puppet':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

      'ruby-1.9.3-p392@global/bundler':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

     'ruby-1.9.3-p392@global/mailcatcher':
     ensure => 'present',
     require => Rvm_system_ruby['ruby-1.9.3-p392'];
    }
}
```

## manifests/rvm.pp

```
...
if $rvm_installed == "true" {
    ....
    rvm_gem {
      'ruby-1.9.3-p392@global/puppet':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

      'ruby-1.9.3-p392@global/bundler':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

     'ruby-1.9.3-p392@global/mailcatcher':
     ensure => 'present',
     require => Rvm_system_ruby['ruby-1.9.3-p392'];
    }
}
```

# manifests/rvm.pp

```
...
if $rvm_installed == "true" {
    ....
    rvm_gem {
      'ruby-1.9.3-p392@global/puppet':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

      'ruby-1.9.3-p392@global/bundler':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

     'ruby-1.9.3-p392@global/mailcatcher':
     ensure => 'present',
     require => Rvm_system_ruby['ruby-1.9.3-p392'];
    }
}
```

## manifests/rvm.pp

```
...
if $rvm_installed == "true" {
        ....
        rvm_gem {
            'ruby-1.9.3-p392@global/puppet':
            ensure => 'present',
            require => Rvm_system_ruby['ruby-1.9.3-p392'];

            'ruby-1.9.3-p392@global/bundler':
            ensure => 'present',
            require => Rvm_system_ruby['ruby-1.9.3-p392'];

           'ruby-1.9.3-p392@global/mailcatcher':
           ensure => 'present',
           require => Rvm_system_ruby['ruby-1.9.3-p392'];
        }
}
```

## manifests/rvm.pp

```
...
if $rvm_installed == "true" {
    ....
    rvm_gem {
      'ruby-1.9.3-p392@global/puppet':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

      'ruby-1.9.3-p392@global/bundler':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

     'ruby-1.9.3-p392@global/mailcatcher':
     ensure => 'present',
     require => Rvm_system_ruby['ruby-1.9.3-p392'];
    }
}
```

**manifests/rvm.pp**

```
...
if $rvm_installed == "true" {
    ....
    rvm_gem {
      'ruby-1.9.3-p392@global/puppet':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

      'ruby-1.9.3-p392@global/bundler':
      ensure => 'present',
      require => Rvm_system_ruby['ruby-1.9.3-p392'];

     'ruby-1.9.3-p392@global/mailcatcher':
     ensure => 'present',
     require => Rvm_system_ruby['ruby-1.9.3-p392'];
    }
}
```

# WE HAVE RUBY!

# WHAT IF I NEED SOMETHING CUSTOM?

- Custom puppet modules allow you to do nearly anything
- Let’s make one

- mkdir modules/pg
- mkdir modules/pg/files
- mkdir modules/pg/manifests

**modules/pg/manifests/init.pp**

```
class pg(){
  user {"postgres" :
    password => 'password'
  }

  package { ['postgresql-9.1','libpq-dev'] :
    ensure => present
  }


  ...
}
```

## modules/pg/manifests/init.pp

```
class pg(){
  user {"postgres" :
    password => 'password'
  }

  package { ['postgresql-9.1','libpq-dev'] :
    ensure => present
  }

  ...
}
```

**modules/pg/manifests/init.pp**

```
class pg(){
  user {"postgres" :
    password => 'password'
  }

  package { ['postgresql-9.1','libpq-dev'] :
    ensure => present
  }


  ...
}
```

**modules/pg/manifests/init.pp**

```
class pg(){
  user {"postgres" :
    password => 'password'
  }

  package { ['postgresql-9.1','libpq-dev'] :
    ensure => present
  }

  ...
}
```

**modules/pg/manifests/init.pp**

```
class pg(){

  ...

  file { '/etc/postgresql/9.1/main/pg_hba.conf' :
    source => 'puppet:///modules/pg/pg_hba.conf',
    require => [ Package['postgresql–9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }

  ...

}
```

**modules/pg/manifests/init.pp**

```
class pg(){

  ...

  file { '/etc/postgresql/9.1/main/pg_hba.conf' :
    source => 'puppet:///modules/pg/pg_hba.conf',
    require => [ Package['postgresql-9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }

  ...

}
```

**modules/pg/manifests/init.pp**

```
class pg(){

  ...

  file { '/etc/postgresql/9.1/main/pg_hba.conf' :
    source => 'puppet:///modules/pg/pg_hba.conf',
    require => [ Package['postgresql-9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }

  ...

}
```

**modules/pg/manifests/init.pp**

```
class pg(){

  ...

  file { '/etc/postgresql/9.1/main/pg_hba.conf' :
    source => 'puppet:///modules/pg/pg_hba.conf',
    require => [ Package['postgresql-9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }

  ...

}
```

# NOW YOU CAN START WORKING!

...

```
vagrant@precise64:/host_folder/vagrant_data/rbmw/
rmw-server$ bundle exec rake db:create

PG::Error: ERROR:  new encoding (UTF8) is incompatible
with the encoding of the template database (SQL_ASCII)

HINT:  Use the same encoding as in the template
database, or use template0 as template.
```

**modules/pg/manifests/init.pp**

```
class pg(){
  ...
  file { '/etc/postgresql/9.1/main/alter_default_db.sql' :
    source => 'puppet:///modules/pg/alter_default_db.sql',
    require => [ Package['postgresql-9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }
  exec { 'udpdate_default_db' :
    command => "/usr/bin/psql -U postgres < /etc/postgresql/9.1/main/
alter_default_db.sql",

    require => [Package['postgresql-9.1'], File['/etc/postgresql/9.1/main/
pg_hba.conf'], File['/etc/postgresql/9.1/main/alter_default_db.sql'] ]

  }

}
```

**modules/pg/manifests/init.pp**

```
class pg(){
  ...
  file { '/etc/postgresql/9.1/main/alter_default_db.sql' :
    source => 'puppet:///modules/agapered_pg/alter_default_db.sql',
    require => [ Package['postgresql-9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }
  exec { 'udpdate_default_db' :
    command => "/usr/bin/psql -U postgres < /etc/postgresql/9.1/main/
alter_default_db.sql",

    require => [Package['postgresql-9.1'], File['/etc/postgresql/9.1/main/
pg_hba.conf'], File['/etc/postgresql/9.1/main/alter_default_db.sql'] ]

  }

}
```

**modules/pg/manifests/init.pp**

```
class pg(){
  ...
  file { '/etc/postgresql/9.1/main/alter_default_db.sql' :
    source => 'puppet:///modules/agapered_pg/alter_default_db.sql',
    require => [ Package['postgresql-9.1'], [User["postgres"]] ],
    owner => 'postgres',
    group => 'postgres'
  }
  exec { 'udpdate_default_db' :
    command => "/usr/bin/psql -U postgres < /etc/postgresql/9.1/main/
alter_default_db.sql",

    require => [Package['postgresql-9.1'], File['/etc/postgresql/9.1/main/
pg_hba.conf'], File['/etc/postgresql/9.1/main/alter_default_db.sql'] ]

  }

}
```

But there is a problem...

## modules/pg/manifests/init.pp

```
class pg(){
  ...
  exec { 'restart_postgres' :
    command => '/usr/bin/sudo /usr/sbin/service postgresql restart',
    require => [Package['postgresql-9.1'],File['/etc/postgresql/9.1/main/
pg_hba.conf']]
  }

  exec { 'update_default_db' :
    command => "/usr/bin/psql -U postgres < /etc/postgresql/9.1/main/
alter_default_db.sql",
    require => [Package['postgresql-9.1'], File['/etc/postgresql/9.1/main/
pg_hba.conf'], File['/etc/postgresql/9.1/main/alter_default_db.sql'],
Exec['restart_postgres'] ]

  }

}
```

**modules/pg/manifests/init.pp**

```
class pg(){
  ...
  exec { 'restart_postgres' :
    command => '/usr/bin/sudo /usr/sbin/service postgresql restart',
    require => [Package['postgresql-9.1'],File['/etc/postgresql/9.1/main/
pg_hba.conf']]
  }

  exec { 'update_default_db' :
    command => "/usr/bin/psql -U postgres < /etc/postgresql/9.1/main/
alter_default_db.sql",
    require => [Package['postgresql-9.1'], File['/etc/postgresql/9.1/main/
pg_hba.conf'], File['/etc/postgresql/9.1/main/alter_default_db.sql'],
Exec['restart_postgres'] ]

  }

}
```

# JAVA

**files/install_java.sh**

**files/jdk-6u32-linux-x64.bin**

**manifests/init.pp**

```
class java(){
  file { '/tmp/install_java':
    source => "puppet:///modules/java",
    recurse => true
  }

  exec { 'install java' :
    cwd => '/tmp/install_java',
    command => '/usr/bin/sudo /tmp/install_java/install_java.sh',
    creates => '/usr/lib/jvm/jdk1.6.0_32/jre/bin/java',
    require => [File['/tmp/install_java']]
  }
}
```

**files/install_java.sh**

**files/jdk-6u32-linux-x64.bin**

**manifests/init.pp**

```
class java(){
  file { '/tmp/install_java':
    source => "puppet:///modules/java",
    recurse => true
  }

  exec { 'install java' :
    cwd => '/tmp/install_java',
    command => '/usr/bin/sudo /tmp/install_java/install_java.sh',
    creates => '/usr/lib/jvm/jdk1.6.0_32/jre/bin/java',
    require => [File['/tmp/install_java']]
  }
}
```

**files/install_java.sh**

**files/jdk-6u32-linux-x64.bin**

**manifests/init.pp**

```
class java(){
  file { '/tmp/install_java':
    source => "puppet:///modules/java",
    recurse => true
  }

  exec { 'install java' :
    cwd => '/tmp/install_java',
    command => '/usr/bin/sudo /tmp/install_java/install_java.sh',
    creates => '/usr/lib/jvm/jdk1.6.0_32/jre/bin/java',
    require => [File['/tmp/install_java']]
  }
}
```

# CAN YOU ONLY HAVE ONE?

## Vagrantfile

```
...
config.vm.box = "precise64"
...
config.vm.define :web do |web|
  web.vm.network :private_network, ip: "10.0.0.1"
  web.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "web.pp"
    ...
  end
end
config.vm.define :db do |db|
  db.vm.network :private_network, ip: "10.0.0.2"
  db.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "base.pp"
    ...
  end
end
```

# Vagrantfile

```
...
config.vm.box = "precise64"
...
config.vm.define :web do |web|
  web.vm.network :private_network, ip: "10.0.0.1"
  web.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "web.pp"
    ...
  end
end
config.vm.define :db do |db|
  db.vm.network :private_network, ip: "10.0.0.2"
  db.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "base.pp"
    ...
  end
end
```

## Vagrantfile

```
...
config.vm.box = "precise64"
...
config.vm.define :web do |web|
  web.vm.network :private_network, ip: "10.0.0.1"
  web.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "web.pp"
    ...
  end
end
config.vm.define :db do |db|
  db.vm.network :private_network, ip: "10.0.0.2"
  db.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "base.pp"
    ...
  end
end
```

# Vagrantfile

```
...
config.vm.box = "precise64"
...
config.vm.define :web do |web|
  web.vm.network :private_network, ip: "10.0.0.1"
  web.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "web.pp"
    ...
  end
end
config.vm.define :db do |db|
  db.vm.network :private_network, ip: "10.0.0.2"
  db.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "base.pp"
    ...
  end
end
```

## Vagrantfile

```
...
config.vm.box = "precise64"
...
config.vm.define :web do |web|
  web.vm.network :private_network, ip: "10.0.0.1"
  web.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "web.pp"
    ...
  end
end
config.vm.define :db do |db|
  db.vm.network :private_network, ip: "10.0.0.2"
  db.vm.provision :puppet do |puppet|
    puppet.manifest_file  = "base.pp"
    ...
  end
end
```

# YOU’RE READY TO GO

You can now start working with Ruby and Postgres

WE NEED TO SHARE THE GOODNESS

# IGNORE .VAGRANT

**.gitignore**

```
.vagrant
```

# INITIALIZE SUBMODULES AUTOMATICALLY

```
`git submodule init`
`git submodule update`
```

```
Vagrant.configure("2") do |config|
...
```

# SOME THINGS SHOULD BE CONFIGURED

**~/vagrant_setup.rb**

```
require 'ostruct'

USER_CONFIG = OpenStruct.new({
  data_dir: '/Users/dkerber/vagrant_data/',
});
```

## ~/vagrant_setup.rb

```
require 'ostruct'

USER_CONFIG = OpenStruct.new({
  data_dir: '/Users/dkerber/vagrant_data/',
});
```

```
if File.file? (File.expand_path '~/vagrant_setup.rb')
  require '~/vagrant_setup.rb'
else
  require 'ostruct'
  USER_CONFIG = OpenStruct.new(data_dir: File.expand_path(‘~’) )
end

Vagrant.configure("2") do |config|
  ...
  if USER_CONFIG.data_dir
    config.vm.synced_folder USER_CONFIG.data_dir, "/host_folder"
  end
  ...
end
```

```
if File.file? (File.expand_path '~/vagrant_setup.rb')
  require '~/vagrant_setup.rb'
else
  require 'ostruct'
  USER_CONFIG = OpenStruct.new(data_dir: File.expand_path(‘~’) )
end

Vagrant.configure("2") do |config|
  ...
  if USER_CONFIG.data_dir
    config.vm.synced_folder USER_CONFIG.data_dir, "/host_folder"
  end
  ...
end
```

```
if File.file? (File.expand_path '~/vagrant_setup.rb')
  require '~/vagrant_setup.rb'
else
  require 'ostruct'
  USER_CONFIG = OpenStruct.new(data_dir: File.expand_path(‘~’) )
end

Vagrant.configure("2") do |config|
  ...
  if USER_CONFIG.data_dir
    config.vm.synced_folder USER_CONFIG.data_dir, "/host_folder"
  end
  ...
end
```

```
if File.file? (File.expand_path '~/vagrant_setup.rb')
  require '~/vagrant_setup.rb'
else
  require 'ostruct'
  USER_CONFIG = OpenStruct.new(data_dir: File.expand_path(‘~’) )
end

Vagrant.configure("2") do |config|
  ...
  if USER_CONFIG.data_dir
    config.vm.synced_folder USER_CONFIG.data_dir, "/host_folder"
  end
  ...
end
```

**~/vagrant_setup.rb**

```
require 'ostruct'

USER_CONFIG = OpenStruct.new({
  data_dir: '/Users/dkerber/vagrant_data/',
  name: 'David Kerber',
  email: 'dave@agapered.com',
  ssh_private_key: '~/.ssh/id_rsa',
  ssh_public_key: '~/.ssh/id_rsa.pub',
  startup_script: '~/custom_vagrant_setup.sh'
});
```

# DEMO!

This is not a drill

# VAGRANT – THE GOOD

- All the usual benefits of regular virtualization
- Uses very little disk space
- Easy to share
- Cheap to create and destroy new environments
- Executable documentation of configuration
- No context switching while editing

# VAGRANT - THE BAD

- Sometimes getting the configuration right is tricky, and the feedback cycle can be long
- Things change fast
- Sometimes disk access speed is a problem (recommend NFS)

# SO HOW CAN YOU USE THIS?

- Developers
- Managers
- Testers
- Deployment/Configuration Managers

# GO FORTH AND CONQUER!

- http://docs.vagrantup.com/v2/
- Example Config Available At:
  https://github.com/davekerber/vagrant-example

# THANKS!

I’d love to hear from you.

@davidjkerber

dave@agapered.com

agape red