Notes for puppet

- *Installing a package can be done by…*

```
package { "<packageName>":
  ensure => "installed",
}
```

- *Starting a service can be done by…*

```
service { "<serviceName>":
  ensure => "running|stopped",
  enable => true,
}
```

- *Managing a file/directory by…*

```
file { "<filename|path>":
  [path => "<optionalTargetPath>",]
   ensure => "file|directory",
  [recurse => true],
  [content => template("<moduleName>/<file>",]
  [source => "puppet:///modules/<moduleName>/<file>",]
}
```

- *Running a command by…*

```
exec {"install_something":
  command => "sh /tmp/install.sh",
  path => ["/bin","/usr/bin","puppet:///modules/test/"],
  [onlyif => "test -f /tmp/install.sh",]
  [require => File["installFile"],]
}
```

- *Creating a new module can be done by…*
    - cd /private/etc/puppetlabs/puppet/modules
    - puppet module generate agray-installtest
    - cd agray-installtest
    - mkdir files templates (files for distro, templates for template erbs)
    - cd manifests
    - vi init.pp (embed logic) for example….

```
class test {
  include test::params
```

```
$packnameDir = "${test::params::package_name}_dir"


define create_dirs {
  file { "/tmp/${title}":
    ensure => "directory",
  }
  file { "/tmp/${title}/package":
    ensure => "directory",
  }
}

create_dirs { ["test1","test2","test3"]: }

file { "$packnameDir":
  path => "/tmp/${packnameDir}",
  ensure => "directory",
}

service { "tomcat-stop" :
  provider => "init",
  ensure => stopped,
  start => "${test::params::tomcat_home}/bin/startup.sh",
  stop => "${test::params::tomcat_home}/bin/shutdown.sh",
  status => "",
  restart => "",
  hasstatus => false,
  hasrestart => false,
  before => File["deployWar"],
}

file { "deployWar":
  path => "/tmp/${packnameDir}/test.war",
  source => "puppet:///modules/test/test.war",
  require => Service["tomcat-stop"],
}

service { "tomcat-start" :
  provider => "init",
  ensure => running,
  start => "${test::params::tomcat_home}/bin/startup.sh",
  stop => "${test::params::tomcat_home}/bin/shutdown.sh",
  status => "",
  restart => "",
  hasstatus => false,
  hasrestart => false,
  require => File["deploy"],
```

```
  }

  file {"installFile":
    path => "/tmp/install.sh",
    source => "puppet:///modules/test/install.sh",
    require => Service["tomcat-start"],
  }

  exec {"install_something":
    command => "sh /tmp/install.sh",
    path => ["/bin","/usr/bin","puppet:///modules/test/"],
    onlyif => "test -f /tmp/install.sh",
    require => File["installFile"],
  }

  $varName = "lala"
  $varNameEx = "lala1"

  file {"subFile":
    path => "/tmp/varTest.txt",
    content => template("test/varTest.erb")
  }

  #package { "openssl":
  #  ensure => "installed",
  #  provider => "pkgdmg",
  #  source => "puppet:///modules/test/"
  #}
}
```

…with a parameter file…

```
class test::params {

  case $operatingsystem {
    'Darwin': {
        $package_name = 'TestDarwin'
        $tomcat_home = '/Library/Tomcat/'
    }
    default: {
        $package_name = 'TestDefault'
        $tomcat_home = '/opt/tomcat/'
    }
  }
}
```

Static files to be deployed

Static files to be deployed are placed in the files/ directory and referenced via

```
source => "puppet:///modules/test/install.sh",
```

Template files to be deployed
Template files to be deployed are placed in the templates/ directory and referenced via

```
$varName = "lala"
$varNameEx = "lala1"

file {"subFile":
     path => "/tmp/varTest.txt",
     content => template("test/varTest.erb")
}
```

With the template file using variables replaced using ERB

```
<%= @varName %>
<%= @varNameEx %>
```

Running the modules

- *Running the modules…*
    - cd /etc/puppet/manifests
    - vi site.pp and add a line…

```
import 'nodes.pp'
```

    - Then create a profile and roles file, this will help assign groups of functionality to roles and roles to nodes. Roles do not exist in puppet so you need to model them. This is the suggested way.

      Profiles – These are profiles of functionality which could be inherited

```
class profile::base {
        notify {"This is a base profile":}
}

class profile::webserver {
        notify {"This is a webserver profile":}
}

class profile::webserver::dev inherits profile::webserver {
        include test
        notify {"This is a dev webserver profile":}
}
```

      Roles – These are assigned to the roles and inherit the profiles

```
class role {
        include profile::base
```

```
}

class role::webserver inherits role {
        include profile::webserver
}

class role::webserver::dev inherits role::webserver {
        include profile::webserver::dev
}
```

- Then create a nodes.pp file with the nodes you want to process. This will then run the role you want and all the logic is in the same place

```
node 'localhost', 'Rhiannon-mac.local', 'webserver.node' {
       include role::webserver::dev
}
```

- The configurations can then be tested by doing,

  On the server…
  ```
  cd manifests
  puppet apply site.pp
  ```

  On the client…
  ```
  puppet agent --test --server <serverName>
  ```

Other module commands

Other module orientated commands are...

- To install a module...
  ```
  puppet module install <moduleName>
  ```
- To list modules that exist on the system ...
  ```
  puppet module list
  ```
- To search for modules that might be of interest from PuppetLabs ...
  ```
  puppet module search <string>
  ```