Resources : a Resource describes something about the state of the system. For example, whether a certain user or file exists, or a service should be running or enabled, or a package should be installed.

Format

```
resource_type { 'name_of_the_resource':
  argument => value,
  other_arg => value,
 }
```

Example 1:

```
file { 'test':
path => '/tmp/test',
content => 'An example content',
}
```

Example 2:

```
package { 'ntp':
ensure => present, }
```

Example 3:

```
service { 'ntpd':
ensure => running,
enable => true,
}
```

Manifest in Puppet

Step 1: Login to Puppet Server (Master Node)

Step 2: Master# vi /etc/puppet/manifests/site.pp

```
file { "/tmp/myFile":
ensure => "present",
owner => "root",
group => "root",
mode => "644",
content => "This is my first manifest",
}
```

Step 3: Node# puppet agent –t

Step 4: Node# cat /tmp/myFile

Conditional Statements

```
if $operatingsystem =='Ubuntu'{
  package {'apache2':
   ensure => installed,
  }
}
elsif $operatingsystem == 'CentOS' {
  package {'httpd':
   ensure => installed,
  }
}
```

Httpd Service Manifest

```
Master# vi /etc/puppet/manifests/site.pp
if $operatingsystem == 'Ubuntu'
{
  exec{'update':
   command => '/usr/bin/apt-get update',
  }
  package{'apache2':
   ensure => installed,
   require => Exec['update'],
  }
  service{'apache2':
   ensure => running,
   require => Package['apache2'],
  }
}
```

Httpd Service Manifest

```
elsif $operatingsystem == 'CentOS'
{
 exec{'up':
  command => '/bin/yum -y update',
 }
 package{'httpd':
  ensure => installed,
  require => Exec['up'],
 }
 service{'httpd':
  ensure => running,
  require => Package['httpd'],
 }
}
```

Master# vi /etc/puppet/manifests/site.pp

```
exec { 'apt-update':
    command => '/usr/bin/apt-get update',
  }
 package { ‘openjdk-7-jdk':
        ensure => 'installed',
 }
 package { 'tomcat7':
        ensure => 'installed',
 }
 service { 'tomcat7':
        ensure => 'running',
        enable => 'true',
}
```

Mysql Service Manifest

```
Master# vi /etc/puppet/manifests/site.pp
exec { 'apt-update':
  command => '/usr/bin/apt-get update'
}
package { 'mysql-server':
  require => Exec['apt-update'],
  ensure => installed,
}
service { 'mysql':
  ensure => running,
}
Node1# puppet agent –t
```

Classes : Classes contain code blocks that can be invoked from anywhere in the code. Classes provide the facility of code reuse.

Example:

```
class class_name {
::::::::::Puppet code goes here:::::::::
}
Include class_name
```

Class names : Class names must start with a lowercase letter, and can contain lowercase letters, numbers, and underscores. Class names can also use a double colon (::) as a namespace separator.

Classes are introduced with the class keyword, and their contents are wrapped in curly braces.

```
Master# vi /etc/puppet/manifests/site.pp
class unix {
    file { '/etc/passwd':
            owner => 'root',
            group => 'root',
            mode => 644;
    }
    file { '/etc/shadow':
            owner => 'root',
            group => 'root',
            mode => 440;
    }
}
Include unix
```

```
Master# vi /etc/puppet/manifests/site.pp
class tomcat {
   package { 'tomcat':
        ensure => 'installed'
   }
   service { 'tomcat':
        ensure => 'running',
        enable => 'true'
   }
}
Include tomcat
```

```
Master# vi /etc/puppet/manifests/site.pp
class ubnt
{
 exec{'update':
  command => '/usr/bin/apt-get update',
 }
 package{'mysql-server':
  ensure => installed,
  require => Exec['update'],
 }
 service{'mysql':
  ensure => running,
  require => Package['mysql-server'],
 }
}
```

```
class cent
{
  exec{'up':
    command => '/bin/yum -y update',
  }
  package{'mariadb-server':
    ensure => installed,
    require => Exec['up'],
  }
  service{'mariadb':
    ensure => running,
    require => Package['mariadb-server'],
  }
}
```

```
if $operatingsystem == 'Ubuntu'
{
 include ubnt
}
elsif $operatingsystem == 'CentOS'
{
 include cent
}
```

Agent-1# puppet agent –t

Agent-2# puppet agent –t

Inheritance

- Classes also support a simple form of object inheritance. For those not acquainted with programming terms,this means that we can extend the functionality of the previous class without copy/pasting the entire class.
- Inheritance allows subclasses to override resource settings defined in parent classes. A class can only inherit from one other class, not more than one.

Ex:

```
class freebsd inherits unix {
File['/etc/passwd'] { group => wheel }
File['/etc/shadow'] { group => wheel }
}
```

```
class unix
{
 file{'/tmp/demo.txt':
   ensure => present,
   owner => root,
   group => root,
   content => 'Hello Good day!',
 }
}
class linux inherits unix
{
 File['/tmp/demo.txt']{
   group => puppet,
   mode => 0777,
   content => 'SATHYA TECH HYD',
 }
}
include linux
```