DEFINITIONS

## **estrutura**

Entendam a estrutura de uma definição.

### **definição**

define nome($var1, $var2) {  
 código  
}

### **declaração**

Para executar essa definição devemos declarar da seguinte forma em um manifest

nome { 'titulo' :  
 var1 => 'valor1',  
 var2 => 'valor2',  
}

## **casos reais**

### **proxy**

Agora vamos pegar um exemplo real de proxy

#### **definition**

define proxy::squid($http\_port, $squid\_mode, $squid\_hostname, $cache\_mem, $maximum\_object\_size\_in\_memory, $maximum\_object\_size, $memory\_replacement\_policy, $cache\_replacement\_policy, $cache\_dir, $cache\_mgr, $cache\_effective\_user, $cache\_effective\_group, $dns\_nameservers, $ips\_squid) {  
   
 ### pacotes ####################################  
   
 package { 'squid':  
 ensure => present,  
 }  
   
 ### servicos ##########################  
   
 service { 'squid':  
 ensure => running,  
 enable => true,  
 hasrestart => true,  
 hasstatus => true,  
 require => Package['squid'],  
 }  
   
 ### arquivos ############################  
   
 # configuracao do servico squid  
   
 file { '/etc/squid/squid.conf':  
 ensure => present,  
 content => template('proxy/squid.conf.erb'),  
 owner => 'root',  
 group => 'root',  
 mode => 644,  
 require => Package['squid'],  
 notify => Exec['squid-reconfigure'],  
 }  
   
 ### execs ##########################  
   
 exec { 'squid -k reconfigure':  
 alias => 'squid-reconfigure',  
 path => '/usr/bin:/usr/sbin:/bin',  
 refreshonly => true,  
 }  
   
}

Veja que uma definição é muito parecida com uma classe, porém declaramos os atributos que ela vai receber no início.

Essa definição faz o seguinte:

* Declara o pacote squid
* Declara a configurações para o serviço squid
* Declara um arquivo squid.conf com referencia para um template squid.erb
* Declara um comando do squid

E os atributos que ele vai receber são:

$http\_port,   
$squid\_mode,   
$squid\_hostname,   
$cache\_mem,   
$maximum\_object\_size\_in\_memory,   
$maximum\_object\_size,   
$memory\_replacement\_policy,   
$cache\_replacement\_policy,   
$cache\_dir,   
$cache\_mgr,   
$cache\_effective\_user,   
$cache\_effective\_group,   
$dns\_nameservers,   
$ips\_squid

Entendido, agora veremos como declarar a execução dessa definição.

#### **declarando a definição**

No manifest do seu nome, você vai declarar a definição, definir um título e especificar o valor de cada atributo, como o exemplo abaixo:

proxy::squid { 'proxyFilial' :  
 http\_port => '3128',  
 squid\_mode => 'transparent',  
 squid\_hostname => 'proxy.ebc',  
 cache\_mem => '2 GB',  
 maximum\_object\_size\_in\_memory => '6 MB',  
 maximum\_object\_size => '128 MB',  
 memory\_replacement\_policy => 'heap GDSF',  
 cache\_replacement\_policy => 'heap LFUDA',  
 cache\_dir => 'aufs /var/spool/squid 1024 16 256',  
 cache\_mgr => 'monitora@ebc.com.br',  
 cache\_effective\_user => 'proxy',  
 cache\_effective\_group => 'proxy',  
 dns\_nameservers => '127.0.0.1 10.61.12.2 172.16.1.1',  
 ips\_squid => '127.0.0.1 192.168.12.3',  
 }

Aqui outro exemplo para o proxy de uma matriz.

proxy::squid { 'proxyMatriz' :  
 http\_port => '3128',  
 squid\_mode => 'transparent',  
 squid\_hostname => 'proxy.ebc',  
 cache\_mem => '4 GB',  
 maximum\_object\_size\_in\_memory => '6 MB',  
 maximum\_object\_size => '128 MB',  
 memory\_replacement\_policy => 'heap GDSF',  
 cache\_replacement\_policy => 'heap LFUDA',  
 cache\_dir => 'aufs /var/spool/squid 4098 16 256',  
 cache\_mgr => 'monitora@ebc.com.br',  
 cache\_effective\_user => 'proxy',  
 cache\_effective\_group => 'proxy',  
 dns\_nameservers => '127.0.0.1 10.61.12.2 172.16.1.1',  
 ips\_squid => '127.0.0.1 192.168.12.3',  
 }

Veja como é prático declarar os atributos e usar uma definição para diferentes cenários.

Usar definições desse jeito fará mais sentido no momento em que estivermos estudando o puppet em modo client/servidor.

Mas o importante é que você já entendeu como funcionam definições.

#### **template squid.erb**

Agora vejam um trecho do arquivo de template.

...  
http\_port <%= http\_port %> <%= squid\_mode %>  
visible\_hostname <%= squid\_hostname %>  
dns\_nameservers <%= dns\_nameservers %>  
...  
cache\_mem <%= cache\_mem %>  
cache\_replacement\_policy <%= cache\_replacement\_policy %>  
cache\_dir <%= cache\_dir %>  
cache\_mgr <%= cache\_mgr %>  
cache\_effective\_user <%= cache\_effective\_user %>  
cache\_effective\_group <%= cache\_effective\_group %>  
...  
acl ips\_squid src <%= ips\_squid %>  
...

## **dicas definition**

As definições são fundamentais para o reaproveitamento de código, o funcionamento não é diferente das classes, um arquivo para cada classe, um arquivo para cada definição, arquivos dentro do diretório manifests, e a parte de namespace dos subdiretórios é a mesma coisa das classes, o esquema de referencia não muda, com exceção da declaração dos atributos, característica que é única das definições.