Tổng quan về modules

- Module bản thân nó chứa code và data. Nó có thể sử dụng lại, có thể chia sẻ

- Gần như tất cả các bundles đều nằm trong modules, Ngoại lệ duy nhấy là site.pp manifest, nó chứa site-wide và node-specific code.

Using modules

- Puppet sử dụng modules để tìm các class và type nó được sử dụng – nó có thể tự động load các class và type lưu trữ trong modules

- Có thể định nghĩa class hoặc các loại với tên riêng nằm trong manifest hawocj trong 1 ENC

# /etc/puppetlabs/code/environments/production/manifests/site.pp

node default {

include apache

class {'ntp':

enable => false;

}

apache::vhost {'personal\_site':

port => 80,

docroot => '/var/www/personal',

options => 'Indexes MultiViews',

}

}

- Puppet cũng có thể tự động load plugin từ modules

- Có tạo 1 modules mới cho puppet, lưu trữ nó trong direc được định nghĩa tại modulepath và chắc chắn nó được định nghĩa chính xác

- Có thể cài đặt các modules có sẵn từ puppet module (được lưu trữ lại source có sẵn của puppet)

Module layout

- Cấu trúc của 1 modules

<MODULE NAME>

• manifests

• files

• templates

• lib

• facts.d

• examples

• spec

• functions

• types

Ví dụ:

my\_module — This outermost directory’s name matches the name of the module.

* manifests/ — Contains all of the manifests in the module.
  + init.pp — Contains a class definition. This class’s name must match the module’s name.
  + other\_class.pp — Contains a class named my\_module::other\_class.
  + my\_defined\_type.pp — Contains a defined type named my\_module::my\_defined\_type.
  + implementation/ — This directory’s name affects the class names beneath it.
    - foo.pp — Contains a class named my\_module::implementation::foo.
    - bar.pp — Contains a class named my\_module::implementation::bar.
* files/ — Contains static files, which managed nodes can download.
  + service.conf — This file’s source => URL would be puppet:///modules/my\_module/service.conf. Its contents can also be accessed with the file function, like content => file('my\_module/service.conf').
* lib/ — Contains plug-ins, like custom facts and custom resource types. These are used by both the Puppet master server and the Puppet agent service, and they are synced to all agent nodes whenever they request their configurations.
* facts.d/ — Contains [external facts](https://docs.puppet.com/facter/3.7/custom_facts.html#external-facts), which are an alternative to Ruby-based [custom facts](https://docs.puppet.com/facter/3.7/custom_facts.html). These will be synced to all agent nodes, so they can submit values for those facts to the Puppet master. (Requires Facter 2.0.1 or later.)
* templates/ — Contains templates, which the module’s manifests can use. See [“Templates”](https://docs.puppet.com/puppet/5.0/lang_template.html) for more details.
  + component.erb — A manifest can render this template with template('my\_module/component.erb').
  + component.epp — A manifest can render this template with epp('my\_module/component.epp').
* examples/ — Contains examples showing how to declare the module’s classes and defined types.
  + init.pp
  + other\_example.pp — Major use cases should have an example.
* spec/ — Contains spec tests for any plug-ins in the lib directory.
* functions/ — Contains custom functions written in the Puppet language.
* types/ — Contains type aliases.

Manifests

- Mỗi manifest trong modules sẽ định nghĩa 1 class or 1 type. File name của manifest sẽ map tới 1 class or 1 type

- init.pp manifest là 1 dạng đặc biết, nó luôn chứa class cùng tên với modules, ko phải giờ tạo class với tên init

- Tất cả manifest sẽ chứa class và type theo

| Name of module | :: | Other directories:: (if any) | Name of file (no extension) |
| --- | --- | --- | --- |
| my\_module | :: |  | other\_class |
| my\_module | :: | implementation:: | foo |

Lưu ý:

* my\_module::other\_class would be in the file my\_module/manifests/other\_class.pp
* my\_module::implementation::foo would be in the file my\_module/manifests/implementation/foo.pp

Allowed module names

- Modules names nên chứa 1 lowercase letter, number, dấu gạch dưới. Thỏa mãn re exp: [a-z][a-z0-9\_]\*

Files in modules

- có thể down file trên puppet

| Protocol | 3 slashes | “Modules”/ | Name of module/ | Name of file |
| --- | --- | --- | --- | --- |
| puppet: | /// | modules/ | my\_module/ | service.conf |

Templates in modules

- Có 2 loại template, dành cho ruby và dsl (ERB or EPP)

| Template function | (‘ | Name of module/ | Name of template | ’) |
| --- | --- | --- | --- | --- |
| template | (' | my\_module/ | component.erb | ') |
| epp | (' | my\_module/ | component.epp | ') |

- So template('my\_module/component.erb') would render the template my\_module/templates/component.erb, and epp('my\_module/component.epp') would render my\_module/templates/component.epp.

Writing modules

- Để viêt 1 module, sử dụng “puppet module generate” cmd

- Để chỉ định 1 modules “puppet module generate <USERNAME>-<MODULE NAME>”

VD:

$ puppet module generate examplecorp-mymodule

We need to create a metadata.json file for this module. Please answer the

following questions; if the question is not applicable to this module, feel free

to leave it blank.

Puppet uses Semantic Versioning (semver.org) to version modules.

What version is this module? [0.1.0]

--> 0.1.0

Who wrote this module? [examplecorp]

--> Pat

What license does this module code fall under? [Apache 2.0]

--> Apache 2.0

How would you describe this module in a single sentence?

--> It examples with Puppet.

Where is this module's source code repository?

--> https://github.com/examplecorp/examplecorp-mymodule

Where can others go to learn more about this module?

--> https://forge.puppetlabs.com/examplecorp/mymodule

Where can others go to file issues about this module?

-->

{

"name": "examplecorp-mymodule",

"version": "0.1.0",

"author": "Pat",

"summary": "It examples with Puppet.",

"license": "Apache 2.0",

"source": "https://github.com/examplecorp/examplecorp-mymodule",

"project\_page": "(https://forge.puppetlabs.com/examplecorp/mymodule)",

"issues\_url": null,

"dependencies": [

{

"name": "puppetlabs-stdlib",

"version\_requirement": ">= 1.0.0"

}

]

}

About to generate this metadata; continue? [n/Y]

--> Y

Notice: Generating module at /Users/Pat/Development/mymodule...

Notice: Populating ERB templates...

Finished; module generated in mymodule.

mymodule/manifests

mymodule/manifests/init.pp

mymodule/metadata.json

mymodule/Rakefile

mymodule/README.md

mymodule/spec

mymodule/spec/classes

mymodule/spec/classes/init\_spec.rb

mymodule/spec/spec\_helper.rb

mymodule/examples

mymodule/examples/init.pp

Beginner's guide to writing modules

- Để viết được 1 modules cần

+ hiểu biết về kiến trúc, cấu tạo ngôn ngữ, modules

+ Trước khi bắt đầu cần hiểu cơ bản về Puppet language, cách tạo class, cách đứa chúng vào 1 modules