

Lesson 7 Demo 4: Create a Puppet Module

This section will guide you to:

- Create a custom Puppet module

This lab has three subsections, namely:

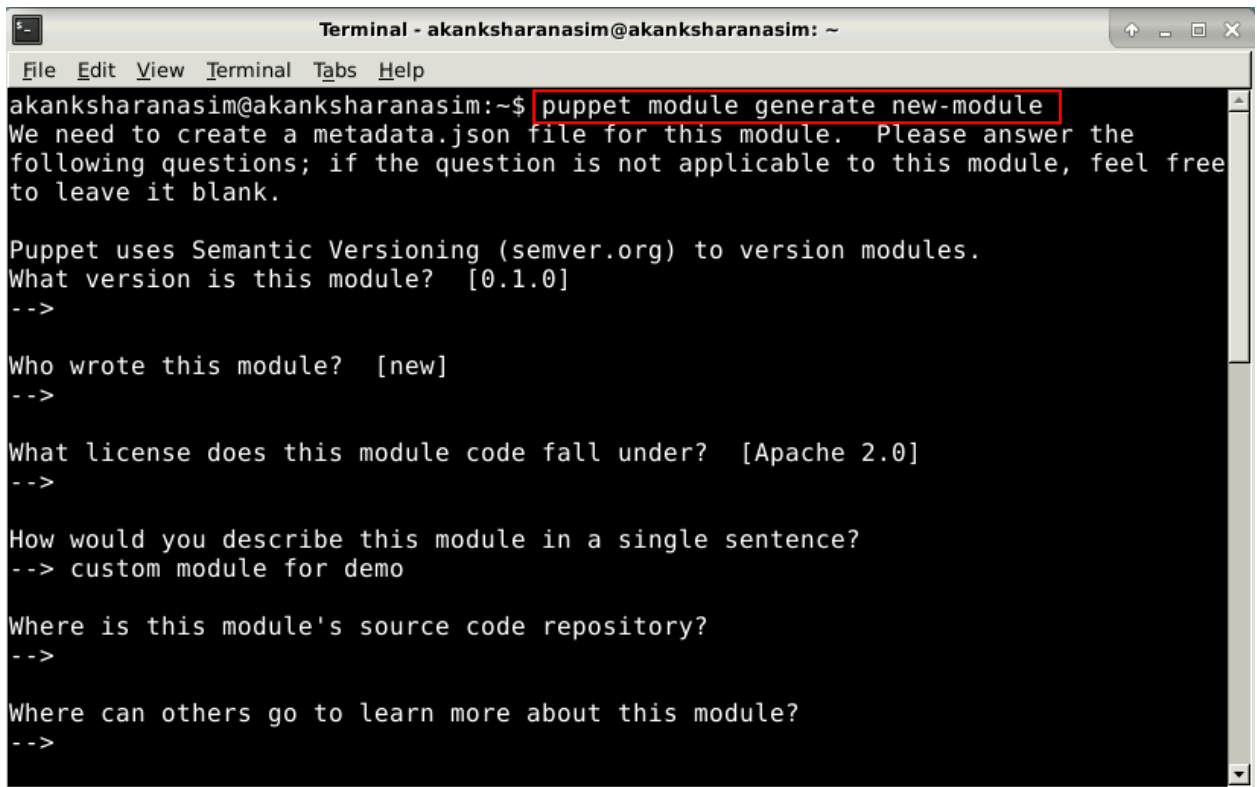
1. Creating the module structure
2. Generating metadata
3. Exploring module directory

Step 1: Creating the module structure

- Open the command line terminal on the system where Puppet has been installed
- Type in the following command to create the module structure:

```
puppet module generate <module-name>
```

- Replace <module-name> in the above command with **new-module**
- Choose a dash-separated name for the module as shown in the following screenshot:



```
Terminal - akanksharanasim@akanksharanasim: ~
File Edit View Terminal Tabs Help
akanksharanasim@akanksharanasim:~$ puppet module generate new-module
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]
-->

Who wrote this module? [new]
-->

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

How would you describe this module in a single sentence?
--> custom module for demo

Where is this module's source code repository?
-->

Where can others go to learn more about this module?
-->
```

- The above modified command will prompt a series of questions that will be used to create the metadata.json file for the module
- Press **enter** to use the default answers for the questions
- Once the json file is created it will be displayed on the terminal as shown in the screenshot below:



```
-----
{
  "name": "new-module",
  "version": "0.1.0",
  "author": "new",
  "summary": "custom module for demo",
  "license": "Apache 2.0",
  "source": "",
  "project_page": null,
  "issues_url": null,
  "dependencies": [
    { "name": "puppetlabs-stdlib", "version_requirement": ">= 1.0.0" }
  ]
}
-----
```

Step 2: Generating metadata

- Type y when prompted with the question: about to generate metadata; continue?

```
About to generate this metadata; continue? [n/Y]
--> y
Notice: Generating module at /home/akanksharanasim/new-module...
Notice: Populating templates...
Finished; module generated in new-module.
new-module/spec
new-module/spec/spec_helper.rb
new-module/spec/classes
new-module/spec/classes/init_spec.rb
new-module/tests
new-module/tests/init.pp
new-module/Gemfile
new-module/README.md
new-module/metadata.json
new-module/manifests
new-module/manifests/init.pp
new-module/Rakefile
akanksharanasim@akanksharanasim:~$
```

- The basic module structure will be created at /home/rootuser/modulename path
- The default files and templates will be populated inside the module's respective directories

Step 3: Exploring module directory

- From the home directory, type in the following command in the command line:

```
ls
```

```
Terminal - akanksharanasim@akanksharanasim: ~
File Edit View Terminal Tabs Help
akanksharanasim@akanksharanasim:~$ ls
Desktop file1.txt new-module puppet6-release-xenial.deb thinclient_drives
akanksharanasim@akanksharanasim:~$
```

- The module created in the previous step will be listed as a result, among other contents of the home directory
- Type in the following command to enter the module and list the contents of the module:

```
cd new-module
ls
```

```
Terminal - akanksharanasim@akanksharanasim: ~/new-module/manifests
File Edit View Terminal Tabs Help
akanksharanasim@akanksharanasim:~$ ls
Desktop file1.txt new-module puppet6-release-xenial.deb thinclient_drives
akanksharanasim@akanksharanasim:~$ cd new-module
akanksharanasim@akanksharanasim:~/new-module$ ls
Gemfile manifests metadata.json Rakefile README.md spec tests
akanksharanasim@akanksharanasim:~/new-module$ cd manifests
akanksharanasim@akanksharanasim:~/new-module/manifests$ ls
init.pp
akanksharanasim@akanksharanasim:~/new-module/manifests$
```

- The above command will display the default files and modules prepared by Puppet while creating the module structure