The Puppet Development Kit (PDK) helps streamline the development of Puppet modules by providing a standardized folder structure and tools for testing and managing Puppet code. Understanding this structure is key to effectively using PDK. Here’s a breakdown of the typical folder structure created by PDK for a Puppet module:

**1. Root Directory**

* **Gemfile**: Lists the Ruby gems required for the module, including those for testing (like rspec-puppet, puppetlabs\_spec\_helper). It is used by Bundler to install the necessary gems.
* **Gemfile.lock**: Automatically generated file that lists the exact versions of gems installed, ensuring consistent environments across different systems.
* **metadata.json**: Contains metadata about the Puppet module, such as its name, version, author, and dependencies. This file is crucial for managing module dependencies and compatibility.
* **pdk-update**: A command used to update your module’s structure and dependencies when new PDK versions are released.
* **README.md**: Provides information about the module, including usage instructions, installation guidelines, and any other relevant details.

**2. Module Directories**

* **manifests/**: Contains the main manifest files for your Puppet module. These files define the classes, defines, and other resources that the module provides.

**Typical Files:**

* + **init.pp**: The main manifest file, which often includes the primary class for the module.
* **lib/**: Contains custom Ruby code that your module might use. This is where you would place custom functions or custom facts.

**Typical Subdirectories:**

* + **facter/**: For custom Facter facts.
  + **puppet/**: For custom Puppet functions, types, and providers.
* **templates/**: Holds template files used to generate configuration files dynamically. Templates are typically written in ERB (Embedded Ruby).
* **files/**: Contains static files that your module may need to deploy, such as configuration files, scripts, or other resources.

**3. Spec Directories**

* **spec/**: The main directory for RSpec tests. This directory contains various subdirectories and files for unit and integration testing of your module.

**Typical Subdirectories:**

* + **classes/**: Contains spec files that test Puppet classes.
  + **defines/**: Contains spec files for testing Puppet defines.
  + **params/**: Contains spec files for testing parameters and their validation.
  + **support/**: For custom RSpec helpers, matchers, or shared examples.

**Important Files:**

* + **spec\_helper.rb**: Configuration file for RSpec. It includes setup code for testing, such as configuring the module path and including necessary libraries.
  + **spec\_helper\_acceptance.rb**: Configuration for acceptance tests, if you're using them.

**4. Other Directories**

* **acceptance/**: (Optional) Contains acceptance tests for your module. These tests ensure that the module works correctly when applied to real systems. Acceptance tests often use Beaker, a testing tool for running tests against real or virtual systems.
* **build/**: (Optional) Contains build-related scripts or configurations if you’re using tools like puppet-module-build.

**Example Folder Structure**

Here’s an example of what a typical PDK Puppet module folder structure might look like:

my\_module/

├── Gemfile

├── Gemfile.lock

├── metadata.json

├── README.md

├── manifests/

│ └── init.pp

├── lib/

│ ├── facter/

│ └── puppet/

├── templates/

├── files/

├── spec/

│ ├── classes/

│ │ └── my\_class\_spec.rb

│ ├── defines/

│ ├── params/

│ ├── support/

│ ├── spec\_helper.rb

│ └── spec\_helper\_acceptance.rb

├── acceptance/

└── build/

**Summary**

The PDK Puppet module structure standardizes how modules are organized and tested, making it easier to manage and maintain Puppet code. Here’s a quick recap:

* **Root Directory**: Contains essential files like Gemfile, metadata.json, and README.md.
* **Manifests Directory**: Contains Puppet manifest files defining your module's resources.
* **Lib Directory**: For custom Ruby code and Facter facts.
* **Templates and Files Directories**: For templates and static files used by the module.
* **Spec Directory**: For RSpec tests, with subdirectories for classes, defines, and parameters.
* **Acceptance Directory**: For acceptance tests (optional).
* **Build Directory**: For build-related configurations (optional).

By following this structure, you can ensure that your Puppet modules are well-organized, maintainable, and ready for testing and deployment.

Top of Form

Bottom of Form