Package/Method	Description	Code Example

Packaging	To create a package, the folder structure is as follows:
	
	1. Project folder → Package name → **init**.py, module_1.py, module_2.py, and so on.
	2. In the **init**.py file, add code to reference the modules in the package.
		module1.py
		
		`def function_1(arg):`
		
		`    return <operation output>`
		
		init.py:
		
		`from . import function_1`

Python Style Guide	● Four spaces for indentation
	● Use blank lines to separate functions and classes
	● Use spaces around operators and after commas
	● Add larger blocks of code inside functions
	● Name functions and files using lowercase with underscores
	● Name classes using CamelCase
	● Name constants in capital letters with underscores separating words
		`def function_1(a, b):`
		
		`        if a > b:`
		
		`        c = c + 5`
		
		`    else:`
		
		`        c = c - 3`
		
		`    return c`
		
		`…`
		
		`c = function_1(a, b)`

		Constant Definition example `MAX_FILE_UPLOAD_SIZE`
Static Code Analysis	Use Static code analysis method to evaluate your code against a predefined style and standard without executing the code. For example, use Pylint to perform static code analysis.	**Shell code:** `pylint code.py`

Unit Testing	Unit testing is a method to validate if units of code are operating as designed.
	
	During code development, each unit is tested. The unit is tested in a continuous integration/continuous delivery test server environment.
	
	You can use different test functions to build unit tests and review the unit test output to determine if the test passed or failed.
		`import unittest`
		
		`from mypackage.mymodule import my_function`
		
		`class TestMyFunction(unittest.TestCase):`
		
		`    def test_my_function(self):`
		
		`        result = my_function(<args>)`
		
		`self.asserEqual`

		`(result, <response>)`
		
		`unittest.main()`