Scripts demonstrating the Python ‘unittest’ module, a unit testing framework.

Unittest supports test automation, sharing of setup and shutdown code for tests, aggregation of tests into collections, and independence of the tests from the reporting framework.

unittest.mock is a library for testing in Python. It allows you to replace parts of your system under test with mock objects and make assertions about how they have been used.

unittest.mock provides a core Mock class removing the need to create a host of stubs throughout your test suite. After performing an action, you can make assertions about which methods / attributes were used and arguments they were called with. You can also specify return values and set needed attributes in the normal way.

Additionally, mock provides a patch() decorator that handles patching module and class level attributes within the scope of a test, along with sentinel for creating unique objects. See the quick guide for some examples of how to use Mock, MagicMock and patch().

Mock is very easy to use and is designed for use with unittest. Mock is based on the ‘action -> assertion’ pattern instead of ‘record -> replay’ used by many mocking frameworks.

Compiled and presented by Vakindu Philliam.