**Mock:**

*“Mock objects are simulated objects that mimic the behavior of real objects in controlled ways”* - Wikipedia.

Mock objects are used in unit tests to simulate the behavior of complex, real objects and can be used when a real object is either impractical (or even impossible) to incorporate into a unit test – also from Wikipedia. They know their shit.

* A mock is meant to **simulate an expected behavior** of a complex component in the system. It is written to **fit the expectations** of the programmer from the component that the mock simulates.
* The mock is used more in Behavioral testing that check how the different components in the system interact with each other.

**Stub:**

Even though a stub can act as a mock, a stub is meant to **simulate a response** for a simplemethod.

The stub is **pre-written** before the tests and it returns a **pre-determined response** mostly without concerning the input.

Stubs provide “synthetic” answers to calls made during the tests and usually don’t respond to calls made outside of the test system, although it can save some data related to the calls.

**Bottom line**

* The Stub is a fake object that allows our tests to run smoothly, the Mock is a smarter stub that verifies that the test passes through it.
* The Mock is like the Stub but instead of returning some value, it asserts the interaction rather than the state.
* The Mock is not expected to return a value, but is expected to verify that a specific order of method calls is made.
* The mocks and stubs represent a production object in a testing environment by exposing the same interface.
* The purpose of both mocks and stubs is to **eliminate the dependencies** of a component so your **tests will be consistent** and not change with the implementation of these components.

**Manual mocks vs automated**

* Manual mocks are defined by writing a module manually by the programmer and are called specifically in the code instead of the real module.
* Because the Mock is manually written as a replacement to the real module, every change in the real module needs to be applied in the Mock, also manually. This makes maintaining the mock a problem.
* Automated mocks are mock generated by a mocking framework that creates a fake object from a set of APIs.
* The Mock framework creates a fake object without the user needing to maintain irrelevant details of the specific test. In other words, the fake object is created for a specific class and when that class adds a new method, nothing needs to change in the test.

Some frameworks create a fake object at run-time, others generate the needed code during compilation. There are a few other types but even from these there can be some complications.

* For example, a framework that uses inheritance cannot fake static methods and objects that cannot be derived.
* Once building a testing system relying on a mocking framework, replacing that mocking system is more expensive than replacing a manual Mock.

**Our Project**

In our project we user a simple manual mocking system(?)

Bibliography:

* The Art of Mocking - By Gil Zilberfeld - <http://www.methodsandtools.com/archive/archive.php?id=122>
* Wikipedia – Mock Objects - <https://en.wikipedia.org/wiki/Mock_object>