mock module in Python

The unittest module now includes a **mock** submodule as of Python 3.3. It will allow you to replace portions of the system that you are testing with mock objects as well as make assertions about how they were used. A mock object is used for simulating system resources that aren't available in your test environment. In other words, you will find times when you want to test some part of your code in isolation from the rest of it or you will need to test some code in isolation from outside services.

> *Note that if you have a version of Python prior to Python 3, you can download the Mock library and get the same functionality.*

Let's think about why you might want to use mock. One good example is if your application is tied to some kind of third party service, such as Twitter or Facebook. If your application's test suite goes out and retweets a bunch of items or "likes" a bunch of posts every time its run, then that is probably undesirable behavior since it will be doing that every time the test is run. Another example might be if you had designed a tool for making updates to your database tables easier. Each time the test runs, it will do some updates on the same records every time and could wipe out valuable data.

Instead of doing any of those things, you can use unittest's mock. It will allow you to mock and stub out those kinds of side-effects so you don't have to worry about them. Instead of interacting with the third party resources, you will be running your test against a dummy API that matches those resources. The piece that you care about the most is that your application is calling the functions it's supposed to. You probably don't care as much if the API itself actually executes. Of course, there are times when you will want to do an end-to-end test that does actually execute the API, but those tests don't need mocks!