

Pytest and coverage Tool

py.test <files>	Run py.test tool
--maxfail=X	Exit after failing X tests
--durations=10	Use this to Profile your tests (time taken)
--cov=path	Run coverage within the project path
--cov-report term-missing	Add this to show which lines were not covered

Packages to install:

pip install pytest
pip install pytest-sugar
pip install pytest-cov

Pytest with Flask

client = app.test_client()	Gives you an Test Client object
client.get(url)	Makes a GET request
client.post(url, data=[])	Makes a POST request
client.put(url, data=[])	Makes a PUT request

You can create a test client to test your Flask app

Coverage Configuration File

.coveragerc	Default filename looked for
[report]	Section defining the report
exclude_lines =	Exclude lines defined after

<http://coverage.readthedocs.io/en/latest/config.html>

Write after exclude_lines = the lines as they are in Python code



By amicheletti
cheatography.com/amicheletti/

Published 17th July, 2017.
Last updated 20th July, 2017.
Page 1 of 1.

Sponsored by [CrosswordCheats.com](http://crosswordcheats.com)
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>

Fixture

@pytests fixture()	Fixture decorator
scope= "session"	Scope for the fixture: Can be session, module, class or function
params=[]	For each value, the fixture will be called with that value (this makes multiple calls)
autouse=True	All tests in the session use the fixture automatically
def my_fixture(request)	Putting request object in fixture gives you access to the pytest request (e.g. put a finalizer)
pytest --fixtures	See all available fixtures

The purpose of test fixtures is to provide a fixed baseline upon which tests can reliably and repeatedly execute.

pytest-mock

In short, mocking is creating objects that simulate the behaviour of real objects.