Tests

Dos and don’ts

Ask yourself

- why do I want to write tests?
- will I benefit?

Don’t force yourself

A crappy test is far worse than no test.

Don’t over-test

“Whoa! Aren’t these getters and setters tested?!”

a.k.a.

“100% test coverage syndrome”

Test in isolation

Prioritize ease of maintenance.
No order dependence, no assumptions.

Write short tests

Assert one, small piece of logic in a test.

Only public methods

Think of the API. Test the API.
Non-public methods can be changed at **any time**
without breaking tests.

Execute your code

Even one, simple test runs your code and gives you more insight.

DRY

Extract small, commonly used bits of logic to helpers.