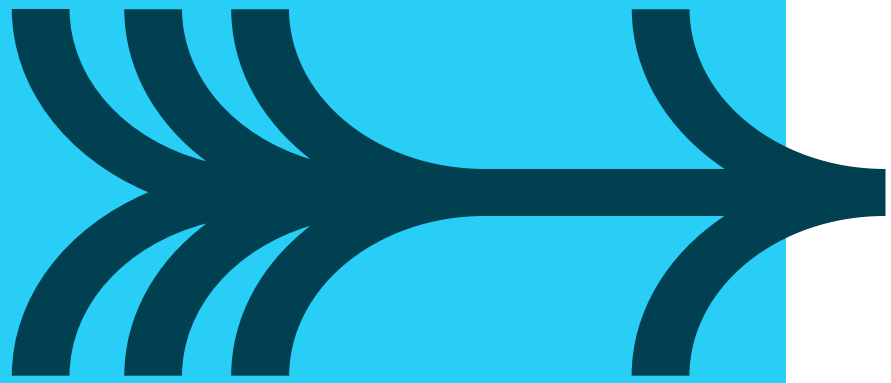
POWERING POTENTIAL

TESTING TECHNIQUES

Test-Driven Development

WHAT IS TDD?

- Repeating cycle of turning requirements into test cases and improving code to pass the tests
- Core practice of XP
- Can be adopted within other methodologies
- Test written before implementation
- Tests drive design of API

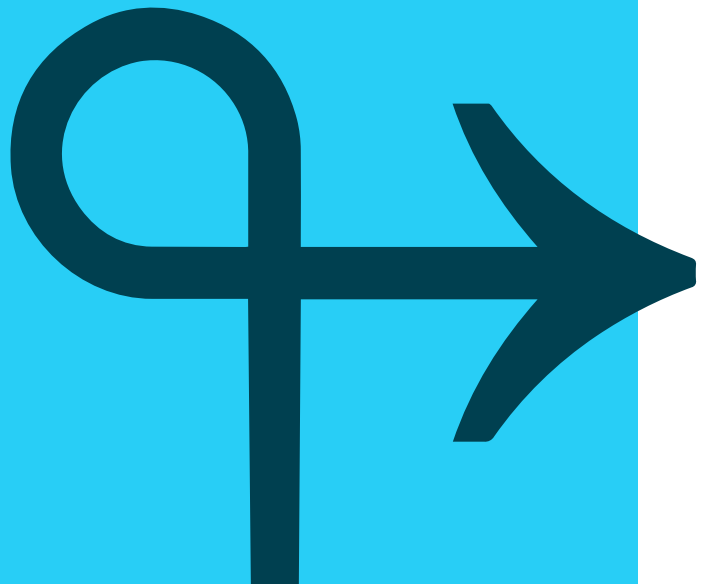


TDD BENEFITS

- **Build up library of small tests** that protect against regression bugs*
- **Extensive code coverage**
  - No code without a test
  - No code that isn't required
- **Almost completely eliminates debugging**, which makes up for time spent developing tests
- **Tests as developer documentation**
- **Confidence, not fear**
  - Confidence in quality of the code
  - Confidence to refactor

* A ***regression bug*** is a defect which stops some bit of functionality working, after an event such as a code release, or refactoring.

TDD STEPS

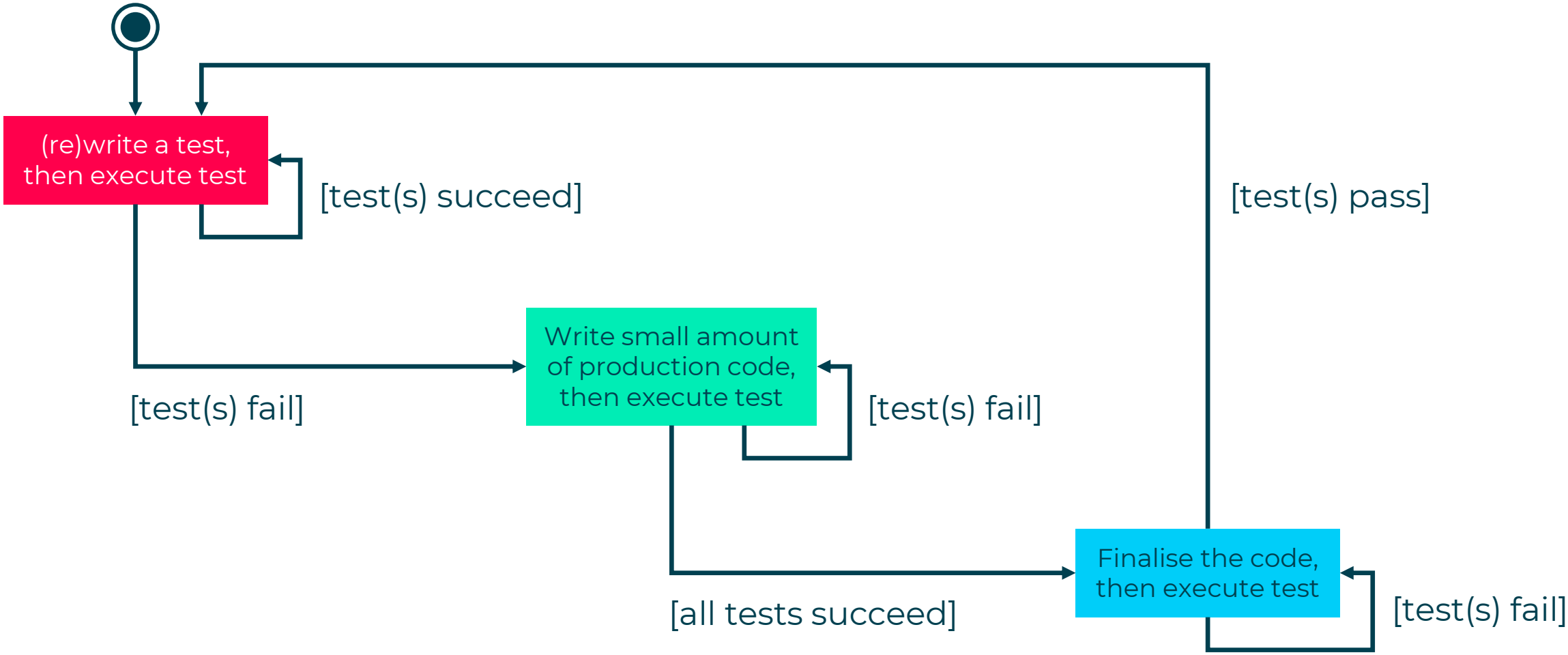


1. Write a failing test.
2. Write just enough code to pass the test.
3. Pass the test.
4. Refactor.

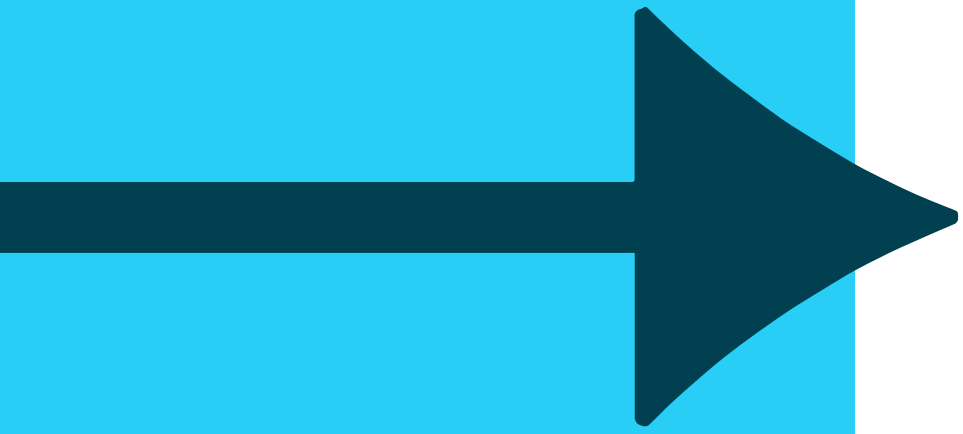
Different to traditional development...

- Tests as an afterthought
- High level of defects
- Lengthy testing phase
- High time and monetary costs
- Poor maintainability

The Red - Green - Refactor Workflow



BECK'S TESTING HEURISTICS



1. **Test List:**
   Write a list of all tests you know you must write.
2. **Starter Test:** Start with the case where the output should be the same as the input.
3. **One Step Test:** Start with the test that'll teach you something and you're confident you can implement.
4. **Explanation Test:**
   Ask for and give explanations in terms of tests.
5. **Learning Tests:**
   Check your understanding of a new API by writing tests.

LAB

"Test Driven Development (TDD)" Lab

Afterwards, we'll discuss what you thought...