zühlke
empowering ideas

Properties of Unit Tests

@arne_mertz

zühlke
empowering ideas

About me

Arne Mertz (@arne_mertz)

Software Engineer, mostly embedded

Learning C++ for almost two decades

Trainer for C++ and maintainable code

Definition of Unit Tests

There are SO many!

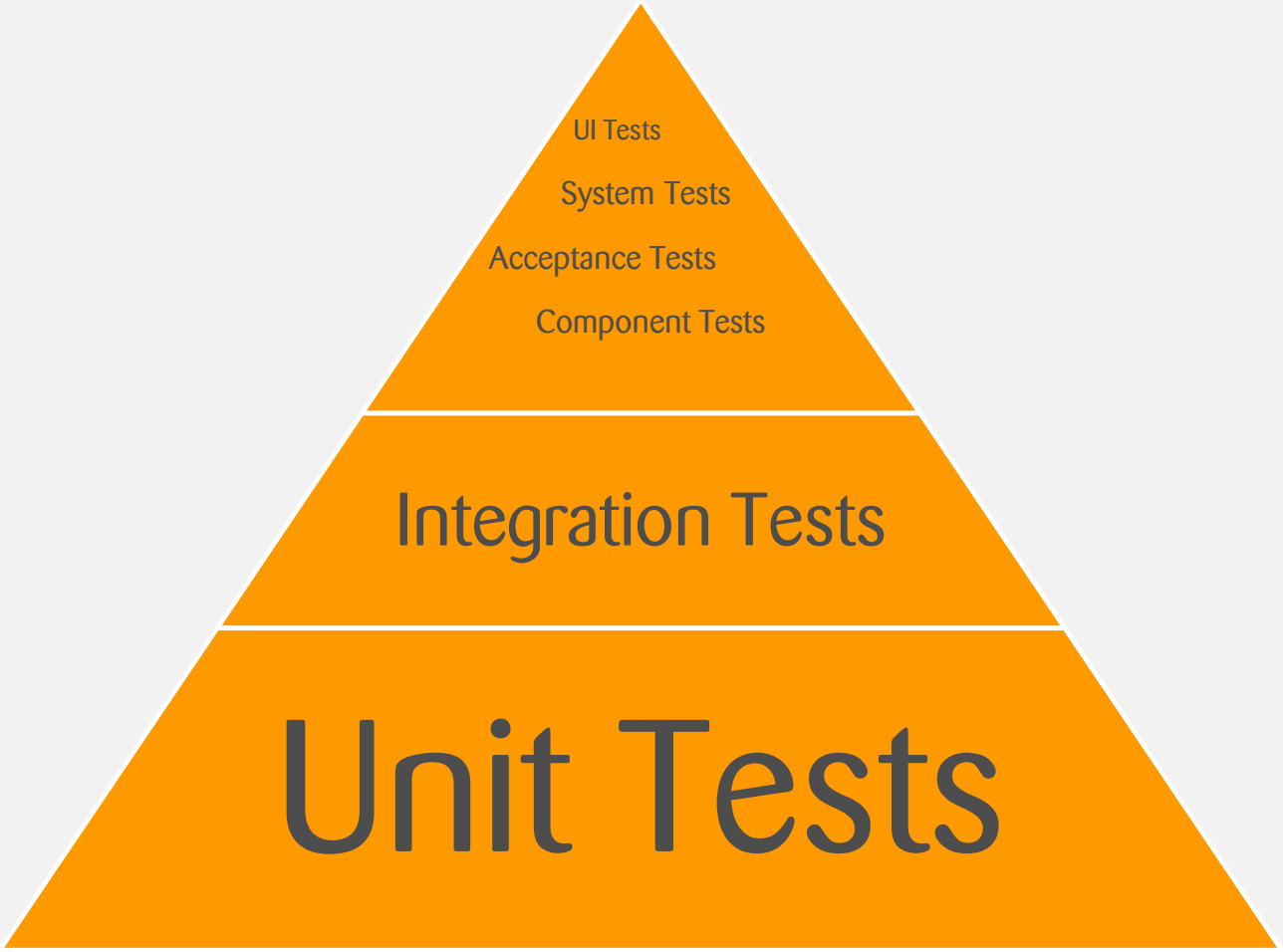
Unit Tests

- Automated by the programmer
- Cover the entire software
- Test individual units

Unit

- Small individual, logically separated piece


unit
test
software
code
module
component
method
application
individual
function
developer
program
testable
use


UI Tests
System Tests
Acceptance Tests
Component Tests
Integration Tests
Unit Tests

Uses of unit tests

QA

- Ensure correctness ✅
- Document failures 📗

Test Driven Development

- Implementation after tests ✅
- Red-green-refactor cycle 🕐

Fixing bugs

- Confirm correctness ✅
- Select specific cases 🎯

Refactoring

- Preserve correctness ✅
- Short feedback loops 🕒

Documentation

- Tests show intended use 📗
  - Co-located with code 🔍

Properties

Disclaimer

- No rules
- No guarantees
- Be pragmatic

Correctness

- Prevent bugs
- Improve confidence

Don’t reproduce logic

- Risks approving wrong behaviour

No complex logic

- Input and expect constants
- Improves maintainability
- Avoids bugs

Gray box testing

- Test against the interface
- Tests are not `friends`
- But test edge cases

Code coverage

- Goal: “good enough”
  - Document gaps

Strong typing

- Focused tests
- Less corner cases

Readability

- Understand failures
- Documentation
- Maintenance

Use Design Principles

- Meaningful names
- Abstractions, e.g. fixtures
- Short test cases

One fact per test case

- Single Responsibility Principle:

  One reason to fail

- Focus on unit under test

Usability check

- Convoluted test cases indicate problematic code

Test case order

- Constructors and basics first
  - Group by feature

Discoverability

- Finding tests to read and maintain them

Project structure

- Test file names
- Test directories
- Test case locations

Speed

- Short feedback loops improve development speed

Rough estimate (order of magnitude)

- 1kloc-10kloc affected on a busy day
- Test code to production code ~ 1:1
  - 10 loc per test case

➔ 100-1000 test cases to run

- Short feedback cycle (minutes)

➔ Tests run in seconds

➔ Milliseconds per test case

Decouple from slow code

- Separate infrastructure code & tests
- Unit test domain logic
- Integration tests

KISS & YAGNI

- Less code
- Fewer test cases
- Faster test execution

Independence

- Test individual units
- Repeat individual test cases

Filtering tests

- Naming & Tagging

Repeatable tests

- Determinism
- Timing is hard

Avoid global state

- Transfers state between tests
  - Hard to reason about
  - Including test cases

Dependency injection

- Tailor outside state
- Trace interactions

Questions

zühlke
empowering ideas

Thank you 👋 Let‘s talk!

🌍 Simplify C++! – www.arne-mertz.de

@arne_mertz

arne.mertz@zuehlke.com

💬 `#include<C++>` Discord (includecpp.org)

Emojis by Twemoji