UnitTest Introdution

Phan Viết Trung
Software Engineer at Zalo Zinstant

Agenda

UnitTest

1. What
2. Why
3. When
4. How

How to write a UnitTest the right way.

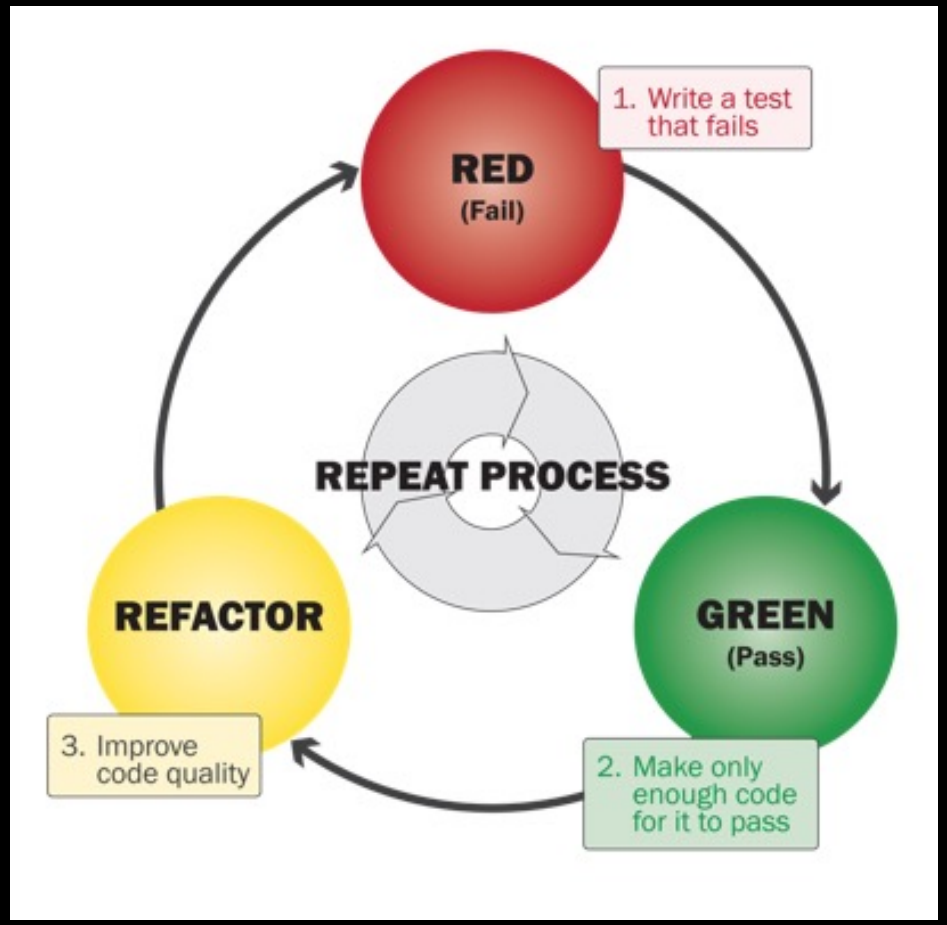
Keyword.

Demo

Q&A

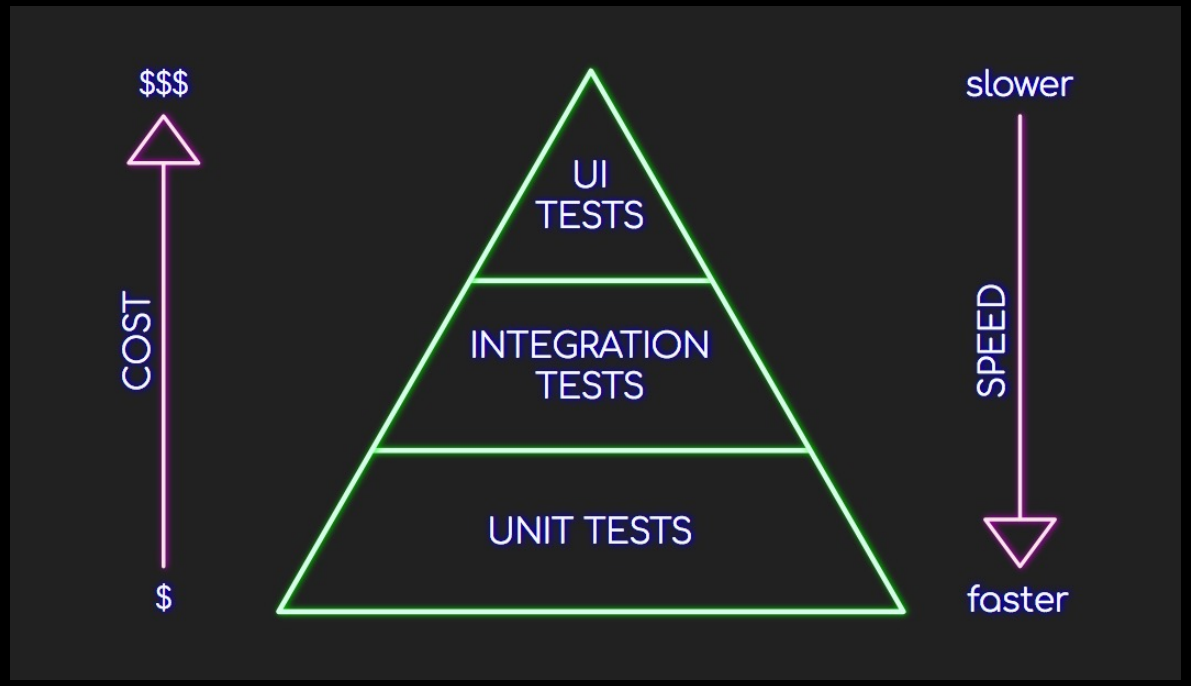
What is UnitTest

- **Process of testing a small pieces of code. Warranty it always work correct, follow the business logic.**



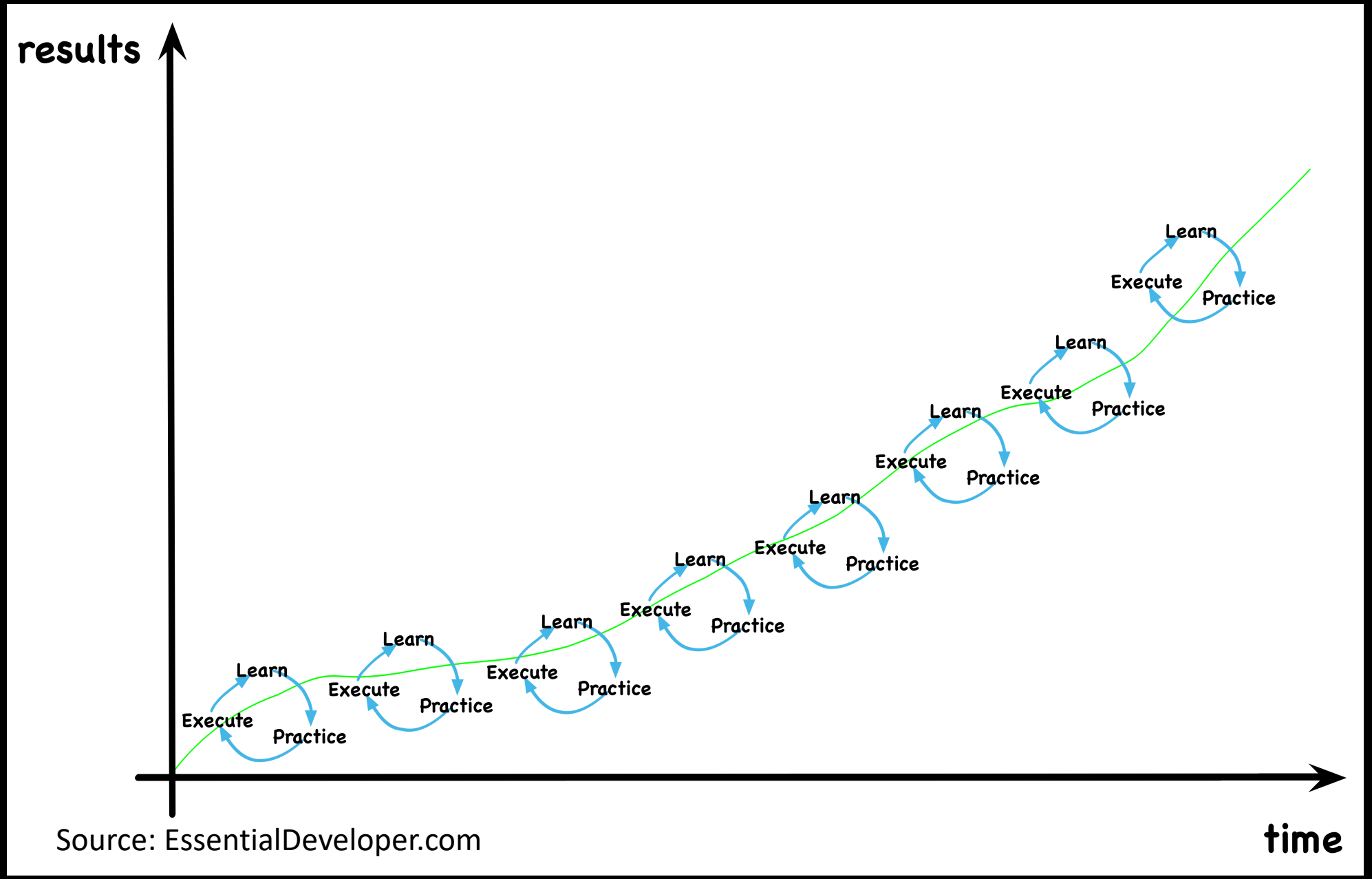
Why UnitTest

- Warranty our code working as expected.
- Cover all the edge cases.
- Protect again future changed.
- Raise an error when something wrong with our system.
- Make we write a better code. Become a real SE
- Cost and time efficency



When

- Now
- Later mean never.
- 1% everyday make a big improvement.
- I’m working on legacy project, I can’t apply TDD, apply test since the high cupling between modules.
  1. Apply for new feature first.
  2. Don’t make a breaking change, keep current codebase.
  3. Do a small cycle of Red-Green-Refactor for existed features when maintain.



Source: EssentialDeveloper.com

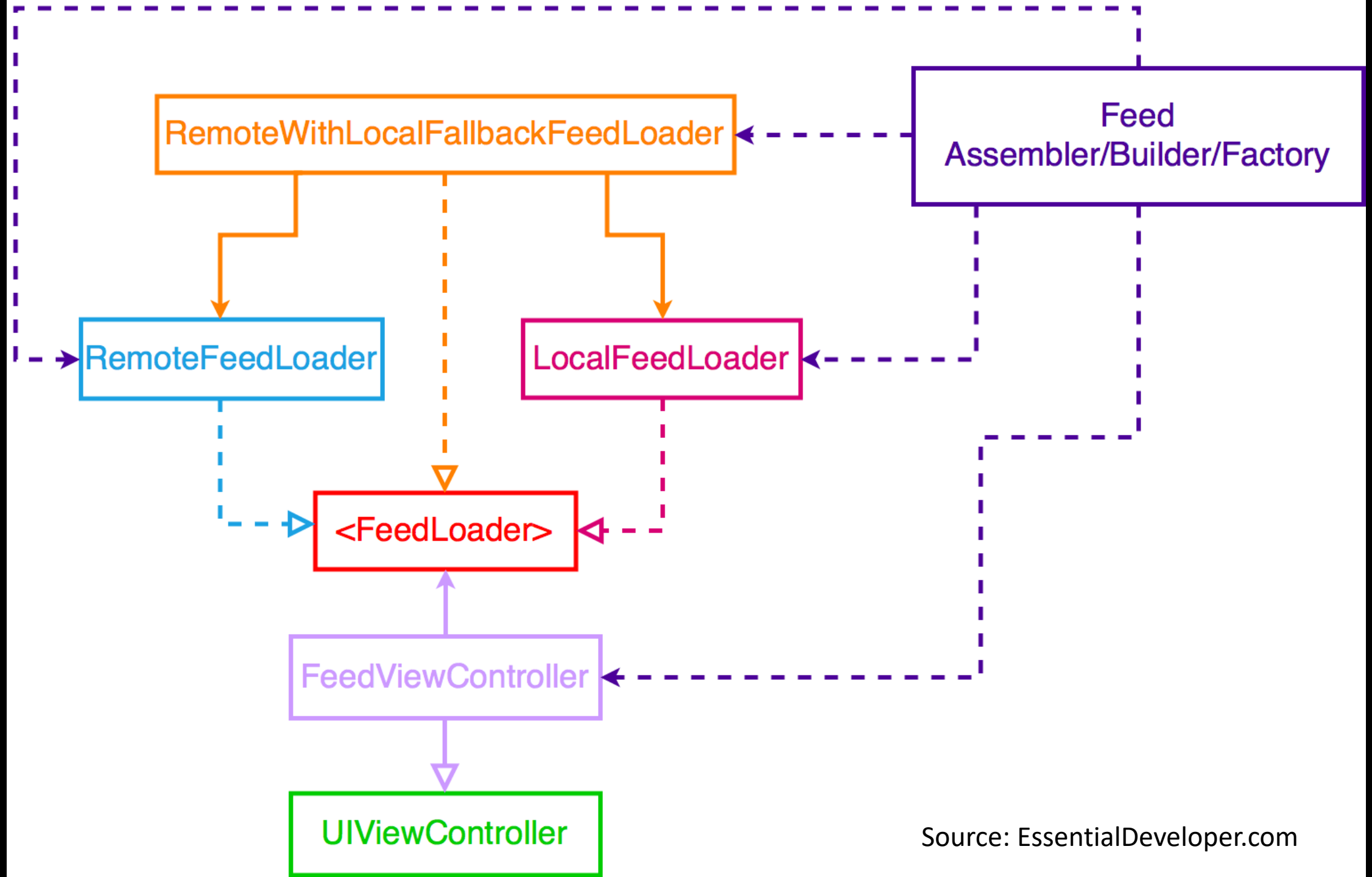
How can I write a UnitTest

BDD, what is it?

- Understand, validate the requirements.
- The Client know nothing about tech side.

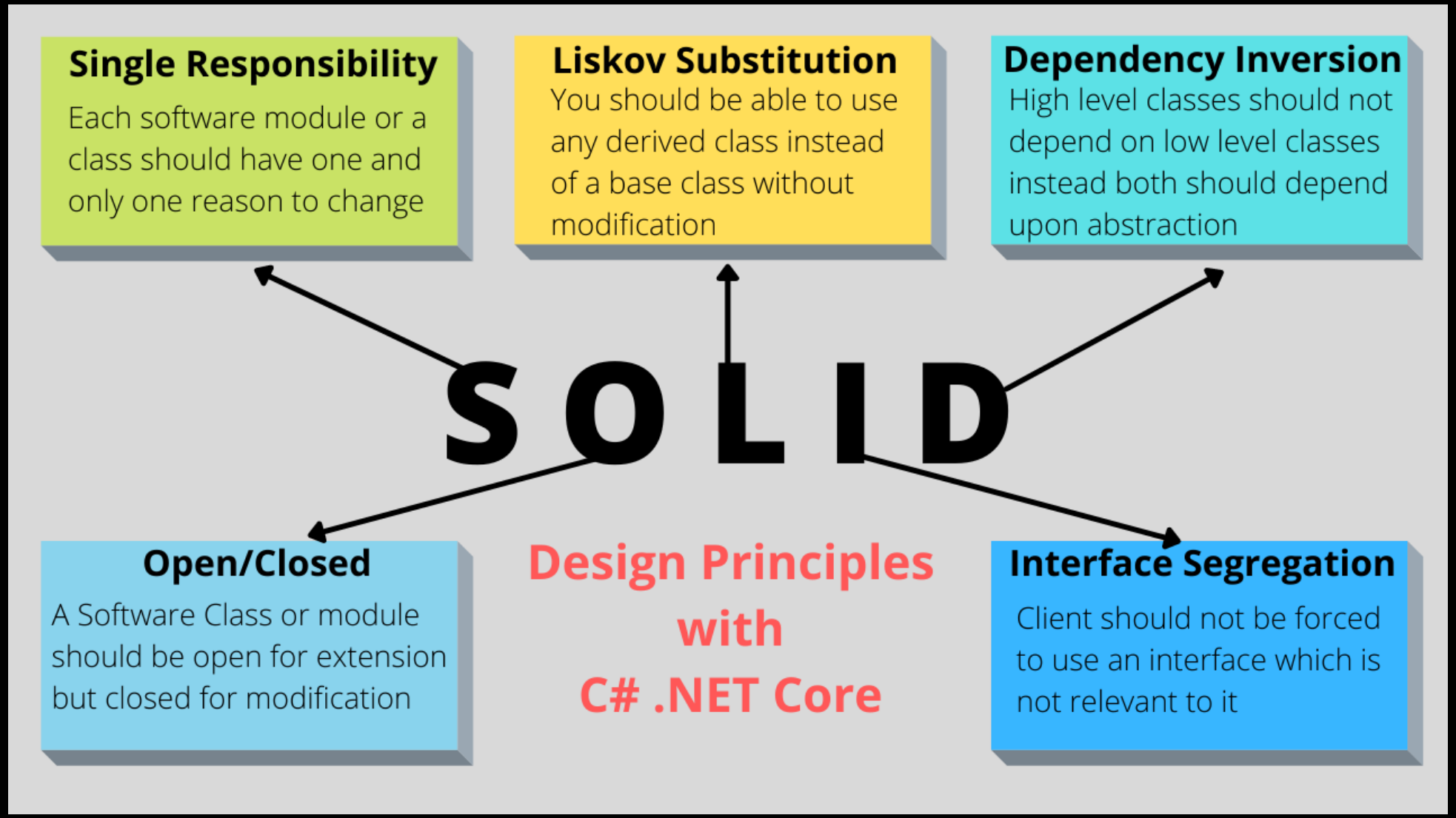
TDD

- Draw a Dependency Graph
- Red-Green-Refactor Cycle
- Code should archived 5 factors:
    1. Enable later binding.
    2. Maintainability
    3. Extendability.
    4. Testability
    5. Enable parallel development.



Source: EssentialDeveloper.com

How to enable 5 factors



Stable API

Naming a test

- **func test_feature_ExpectedBehavior**
- func test_insert_ShouldNotTriggerDeleteCommandOnNewInsertion()
- func test_insert_ShouldTriggerDeleteCommandOnExistedItem()

Structure of a test:

- **Given/When/Then**

Sut – Spy – Mock

Protect again future changed

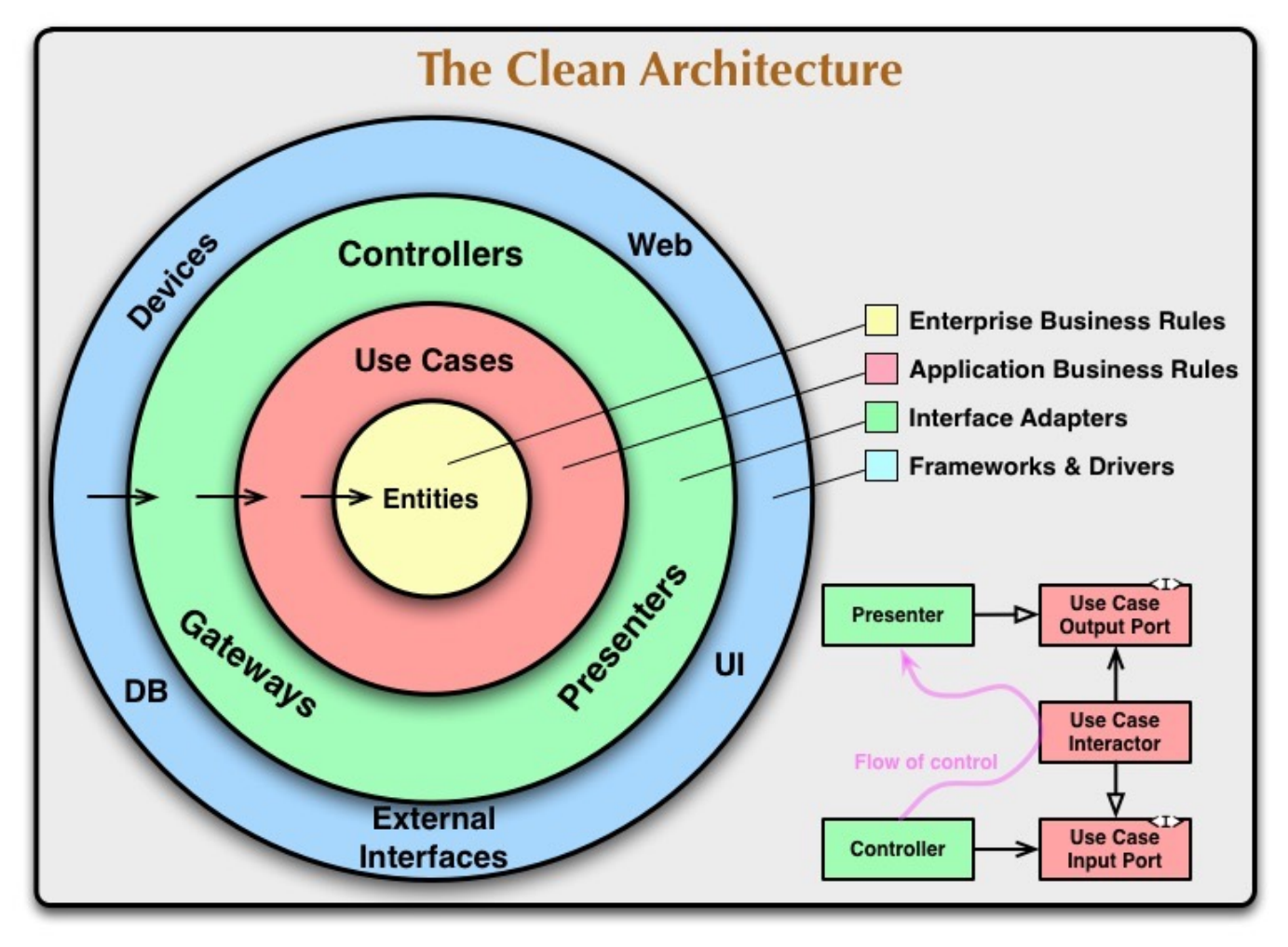
- Using Typedef Alias
- makeSUT(), makeItem()
- Using Tuple, Array to veryfy sequence of action, events, simplify test logic, improve readability.

Write test a right way

-Write a behavior code not a implement details.
(Behavior everywhere, not only test)

- Welcome to future changed.
- Easy to changed.
- Valid state of the system, not validate a single value.
- Fast.

Some off topic about Dependency Graph.



Source: CleanArchitecture by Uncle Bob

Keyword

Clean Architecture:

Clean code

TDD – BDD

Red – Green - Refactor

What is SPY-Mock-SUT

SOLID principles

Demo

- Use cases:
  1. As a user I Want to see a latest Feed list.
  2. If the network is error, retry 3 time, otherwise I can see Feed I had seen before. <= Cached
  3. If there is no cached, show the error empty list screen. <= Network error + No cached.

Q&A

- Thank you for listening.
- trungpv3@vng.com.vn
- For Slide and Sample source code please visit: https://github.com/viettrungphan/UnitTestIntroduction