Collaborative testing

Jean-Luc Delarbre

Software workflow requirements

- **Traceability**
  - Git
  - Merge request
- **Reproducibility**
  - CI / pipeline
  - Docker
  - Registry
  - Package manager

Creating software: a teamwork

- **System engineer – project leader**
  - Business stuff
- **Software engineer**
  - Play with code
- **There is a gap between the 2 worlds, but**
  - Seamlessly going to business deep into code
  - Building a business model
  - Finding relevant use cases, that defines software behavior

Define testable use cases / user story

- **Use cases / user story**
  - Directly related to:
    - Software users
    - Business domain
  - Defining inputs and expecting behavior
    - Allow to write detailed specifications
    - Runnable as code
- **Place for developers to work with software client**

Usage of software development tool

- **With specification define a relevant use cases**
  - Traceability between specs and tests
  - Project management tool can link issues and tests
- **Test framework allows test automation**
  - Kind of burn down chart
- **Involve non programmers**

Work with agile methods

- **3 Cs**
  - Card
  - Conversation
  - Confirmation
- **Confirm done by test**
  - User story / use case
  - Acceptance test

DSL (Domain Specific Language)

- **Dedicated programming language for a narrow purpose (still executable)**
  - Very simple
  - No structure control necessarily (not Turing complete)
- **External DSL**
  - More freedom with syntax
  - Need some tools to create it or work for a parser
- **Internal DSL**
  - No barrier with the base language
  - Fluent interface

Readable tests

- **Acceptance tests as a mirror of use cases**
  - Should be easily readable by non-programmers
- **Kind of repetition**
  - Hard to maintain

Related topics

- **BDD – Behavior driven development**
- **Cucumber**

Next objective

- **Writing code that permit use case testing**
- **Writing non software people readable tests**

- **Writing testable code is not easy**
  - Without appropriate knowledge, it could lead to test hell

Reference

- **Object-Oriented Software Engineering: A Use Case Driven Approach - Ivar Jacobson**
- **Extreme Programming Explained – Kent Beck**
- **https://www.martinfowler.com/articles/languageWorkbench.html**
- **https://martinfowler.com/bliki/SyntacticNoise.html**
- **https://martinfowler.com/books/dsl.html**
- **https://www.martinfowler.com/bliki/FluentInterface.html**