Testing

Chris Zou

Overview

Due to complexity of software, bugs are a fact of life.

It will never be possible to test software completely. (Be Better? Yes. Be Perfect? No.)

Test Cases

- It contains what you feed to software (input);
- What the software should output in response (output);

Levels of Testing

- **Smoke Tests** (obvious bugs like crash)
- **Unit Tests**
- **Regression Testing**
- **Stress Tests** (check how components or systems behave when under pressure)
- **System Tests** (run on entire system, verify that everything works right)

Unit Testing(low-level tests to verify the functionality of a single class each time)

- focus on the unit being tested (no dependent on database, networking)
- easy to run by anyone
- easy to write (a few minutes per test)

Unit tests come with lots of baggage: like requirement, test behaviour, use mock objects

Example: Junit Testing

Regression Testing

- it can be categorized as functional tests or Unit Tests. Functional tests exercise the complete program with various inputs. Unit tests exercise individual functions.
- **Purpose: to ensure new feature/patch does not introduce new bugs**. (use previous test suits, that's why called regression**)**
- Contrast with non-regression testing, which aims to new feature/patch work or not.
- This will ensure the whole project will not break when big bug occurs at the end of stage.(you can easily step back!)

Stress Testing

- keyword: realibility

System Testing

- A testing conducted on a complete, integrated system to evaluate the system’s compliance with its specified requirements.

Code Coverage **(which lines of the program are covered (executed) during the tests)**

- Getting 100% statement coverage is generally unrealistic, even 100%, it could go wrong 1/0.
- “dead code” refer to code that can never be executed. (waste effort to maintain code if not removed)
- As test cases are added, diminishing returns occur. (cover rate decreases as more test cases)