Testing

- Topics
  - **When, why and kinds of testing used in software development**
  - **Functional (Black-box)** vs. Structural (white-box testing)
  - Stubs and mock objects
  - Dependency injection and testing

Bold is Mon Feb 29.
Dr. Marc Palyart
will take over on Wed.

Testing

- Learning objectives:
    - Be able to explain when, how and the kinds of testing used in software development
    - Be able to recognize and write a black-box unit test
    - Be able to recognize and write a white-box unit test
    - Be able to write appropriate stubs and/or mock objects
    - Be able to use dependency injection for testing
    - Understand the following terms:
        - Unit test, integration test, system test, acceptance test, verification, validation, test case, test suite, regression tests

Motivation

Ideally, we would be able to prove that a software system behaves as intended.

```
// x and y are guaranteed to be > 0
static int someMethod(int x, int y) {
   if(x > y) {
        return x - y;
   } else if(x < y) {
        return y - x;
   } else {
        return x + y;
   }
}
```

Can this function ever return a negative number?

Is proving ever used for software?

Researchers prove kernel is secure

An Australian research organization says it has absolute mathematical proof of the security of an operating system core.

Proofs are…

- static (program is logically analyzed)
- theoretically can show absence of bugs
- applicability is practically limited
- extremely costly

Tests are…

- dynamic (program is executed)
- builds confidence (can only show the presence, not the absence of bugs)
- used widely in practice
- costly

Testing Goals

Verification: “Did we build the system right?”

Discover situations in which the behavior of the software is incorrect or undesirable

Validation: “Did we build the right system?”

Demonstrate to the developer and the customer that the software meets its requirements

Testing Terminology

Testing:

Execute a program or program unit with a test set to determine if the expected output is produced

Test Case:

A set of value assignments to each input parameter and expected values for each output

Test Set/Test Suite:

A set of test cases

Exhaustive testing:

A test set which covers all possible inputs. Almost always infeasible.

“Testing can show the presence, but not the absence of errors”

— Edgar Dijkstra

Kinds of Testing

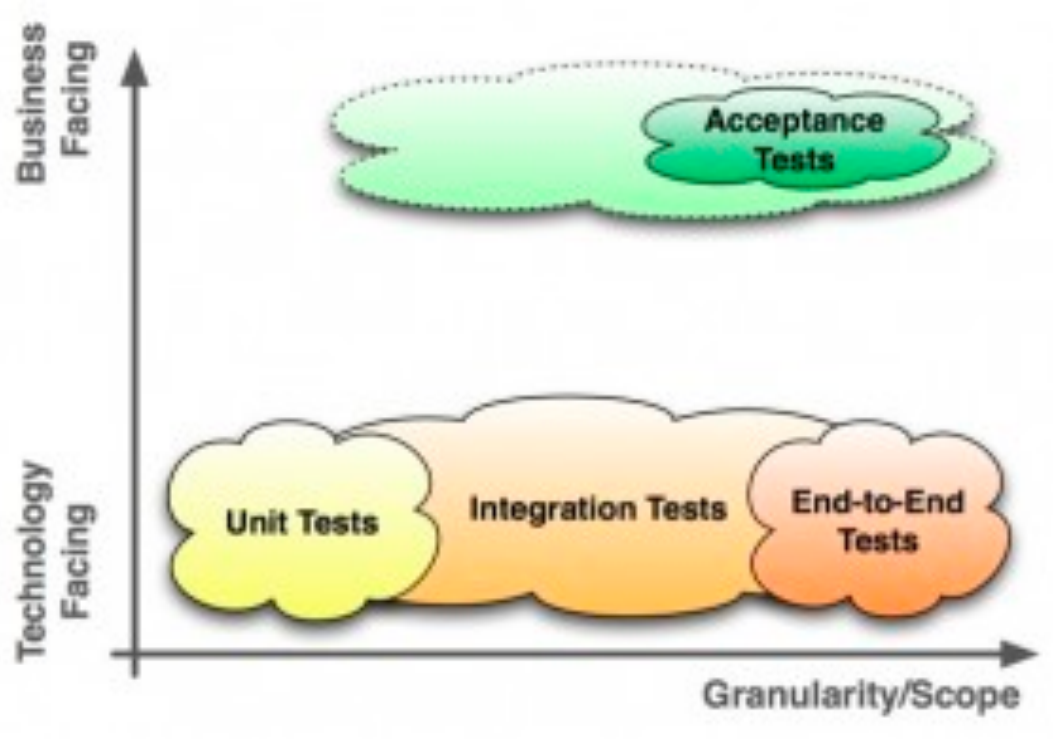
Unit

Regression

Integration

System

Acceptance



Unit Testing

A unit is the smallest testable part of an application.

Units may be:

- functions / methods
- classes
- composite components with defined interfaces used to access their functionality

Unit testing is often performed by developers.

The execution (and sometimes generation - e.g., Pex) of unit testing is often automated (e.g., JUnit).

Regression Testing

Check that changes have not ‘broken’ previously working code.

Not new tests, but repetition of existing tests.

Often at, but not limited to, units.

Performed by software developers and QA personnel.

Integration Testing

Individual units are combined and tested as a group.

Shows that major subsystems that make up the project work well together.

Typically performed by software development teams.

System Testing

Testing large parts of the system, or the whole system, for functionality.

Test the system in different configurations.

Typically performed by QA team.

Acceptance Testing

Formal testing with respect to user needs and requirements conducted to determine whether or not the software satisfies the acceptance criteria.

Does the end product solve the problem it was intended to solve?

Performed by QA and customers.

Testing Tactics

- Tests based on spec
- Treats unit as atomic
- Covers as much specified behaviour as possible

- Tests based on code
- Examines system internals
- Covers as much implemented behaviour as possible

A black-box unit test example

```
boolean isLeapYear(int year)
```

What input values should we test with?

Techniques for choosing input values in black-box testing

Equivalence Class Partitioning (ECP)

Boundary Value Analysis

Equivalence Class Partitioning (ECP)

Divide the test conditions into sets or groups that are considered the same by the unit/system under test. Consider partitions of both valid and invalid input.

Consider Translink and the charging software.

What are the partitions?

Regular Fares

Weekdays from start of service to 6:30 p.m. All bus travel is a one-zone fare.

Zone	Adult Price	Concession Price
1 Zone	$2.75	$1.75
2 Zone	$4	$2.75
3 Zone	$5.50	$3.75

Weekdays after 6:30 p.m. and all day Saturday, Sunday and Holidays.

Zone	Adult Price	Concession Price
All Zones	$2.75	$1.75

Equivalence Class Partitioning…

Translink example…

Partition
All bus rides, regular fare
1 zone bus and non-bus rides before 6:30pm on week-day, regular fare
2 zone bus and non-bus rides before 6:30pm on week-day, regular fare
3 zone bus and non-bus rides before 6:30pm on week-day, regular fare
Any ride all zones after 6:30pm on weekday or weekend or holiday, regular fare
All bus rides, concession fare
1 zone bus and non-bus rides before 6:30pm on week-day, conc. fare
2 zone bus and non-bus rides before 6:30pm on week-day, conc. fare
3 zone bus and non-bus rides before 6:30pm on week-day, conc. fare
Any ride all zones after 6:30pm on weekday or weekend or holiday, conc. fare
Bus rides before the system start running (an example of a value in an invalid partition)

Boundary Value Analysis

Test boundary values between partitions. Based on observations that errors often happen at the boundaries,

Consider Translink again. What would be examples of boundary values?

Leap years

Leap years were introduced by Julius Caesar. In the Julian calendar, any year divisible by 4 was a leap year.

The Julian calendar provided too many leap years and eventually the Julian calendar was 24 days out of sync with the fixed dates for equinoxes, etc.

The Gregorian calendar was adopted in Italy, Portugal and Spain in 1582.

It used a new formula for calculating leap years. In October 1582, 10 days were "dropped".

For info on programming time see:
https://www.youtube.com/watch?v=-5wpm-gesOY

http://www.timeanddate.com/calendar/gregorian-calendar.htm

ECP and Boundary Value Analysis

What kind of inputs would you test a Gregorian leap year function with if you apply ECP and Boundary Value Analysis?

ECP and Boundary Value Analysis

Better than random testing

But only as good as your partitions!

No exploration of combinations of inputs.