



THE SOFTWARE DELIVERY EXPERTS

Unit Testing: Why and How QA Can Get Involved

Presented By: David Dang

In this training we will cover:

- What is Unit Testing?
- Unit Test Benefits
- Types of Unit Testing
- Three Pillars of Unit and Unit Integration
- Levels of Competence in Skills for Unit Testing (QA)

What is Unit Testing?

Unit testing is the smallest testable parts of programmed code (units) that are individually and independently scrutinized for proper operation.



Benefits of Unit Testing

➤ Issues are found at an early stage

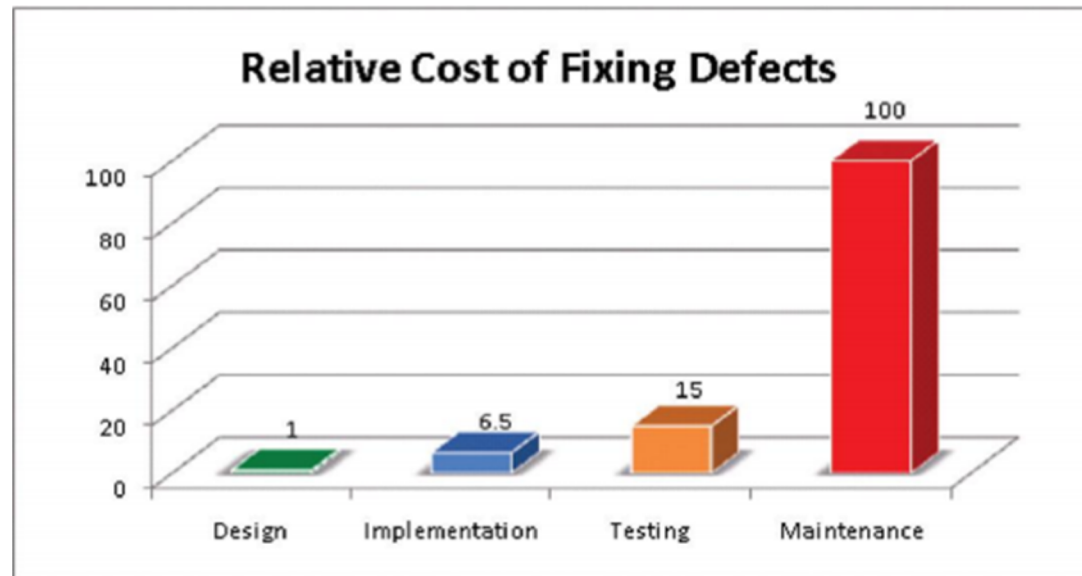
- At this point, failure is caused by bug in code or possibly with the unit test, so it is easier to trace the failure .
- Less costly to find now than later
- Working code produced faster and with fewer bugs!



Benefits of Unit Testing

➤ Helps reduce the cost of bug fixes

- Defects found in unit test are easier to locate and easier to fix.



Defects found in testing were 15 times more costly than if they were found during the design phase and 2 times more costly than if found during implementation! Source: IBM System Science Institute

Benefits of Unit Testing

➤ Reduces testing overlap between Development and QA

- Focus for Unit Testing is on individual function or piece of code, while focus for QA is Functional or Integration testing.
- Robust Unit Testing early on will allow QA to focus on Functional testing.

Benefits of Unit Testing

- Runs more often (e.g. run during every build)
 - Can be a low-level regression test suite. You can go back at any time and see not only what broke but where the bug is. Some teams run the unit test suite as part of the nightly build since it is a low effort way to catch bugs before the build goes off to QA

Benefits of Unit Testing

➤ Provides more confidence on code changes

- Verifies that the code still works as the code base changes
- Makes Integration testing easier, since you have already validated the smaller units
- As long as tests continue to pass, you will be sure that code continues to meet requirements



Benefits of Unit Testing

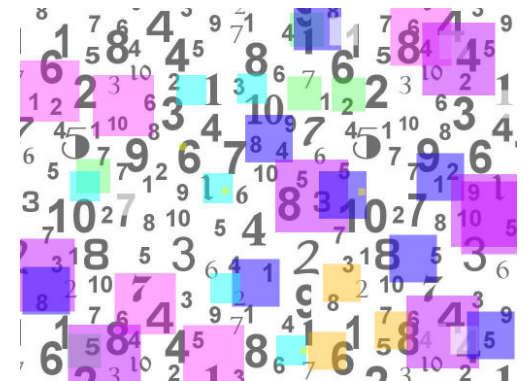
➤ Offered as a form of documentation

- Provides “living” documentation of an application. Developers can look at the unit tests to get a better understanding of the code.
- Unit tests MUST stay up to date, so unlike requirements documentation, you will always know that you are current. If the unit test happens to become outdated, the unit tests would fail!!!

Benefits of Unit Testing

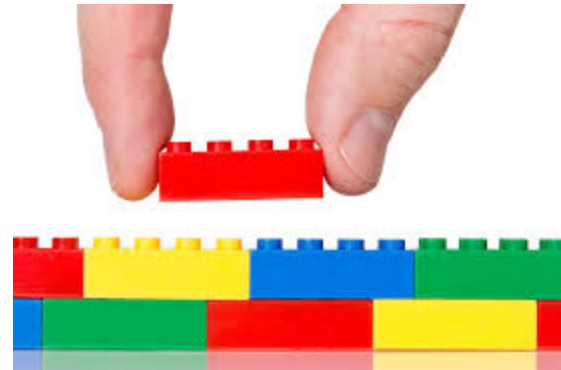
➤ Covers more combinations & permutations quicker

- When Developer has to write unit test, he/she must really think about the design and what needs to be accomplished. Keeps focus and makes for better design.
- In order to Unit Test, need to break down the design into the single functions of code. This will improve the overall code design, unit interfaces and the design, itself.



Benefits of Unit Testing

- Helps to validate for testability, complexity and reusability
 - Since Unit is small, test is easier to design, execute, record and analyze test results than larger chunks of code are.
 - Code becomes more reusable

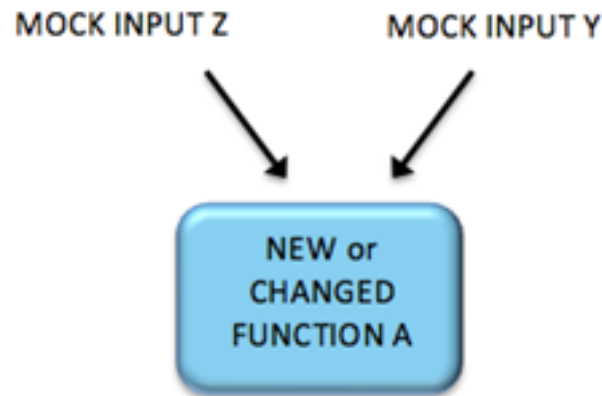


Two Types of Unit Testing

- Unit
- Unit Integration

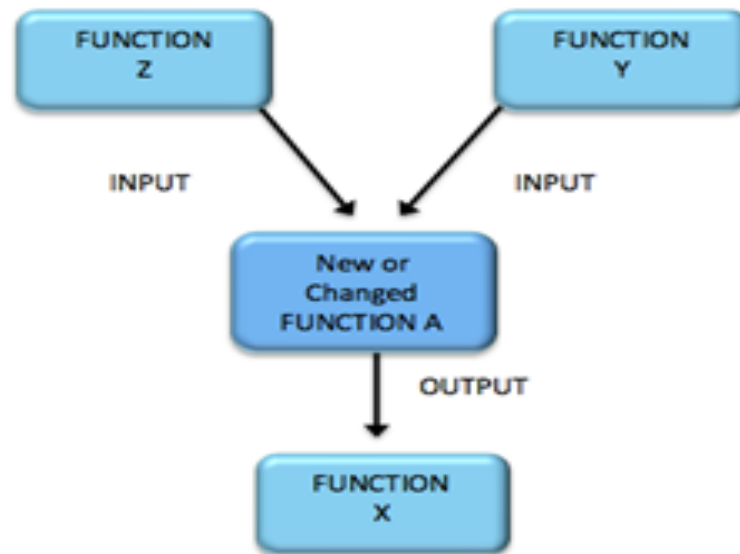
Unit Testing

- Tests a single function, class, method, procedure, etc by itself independently of any other programmed code

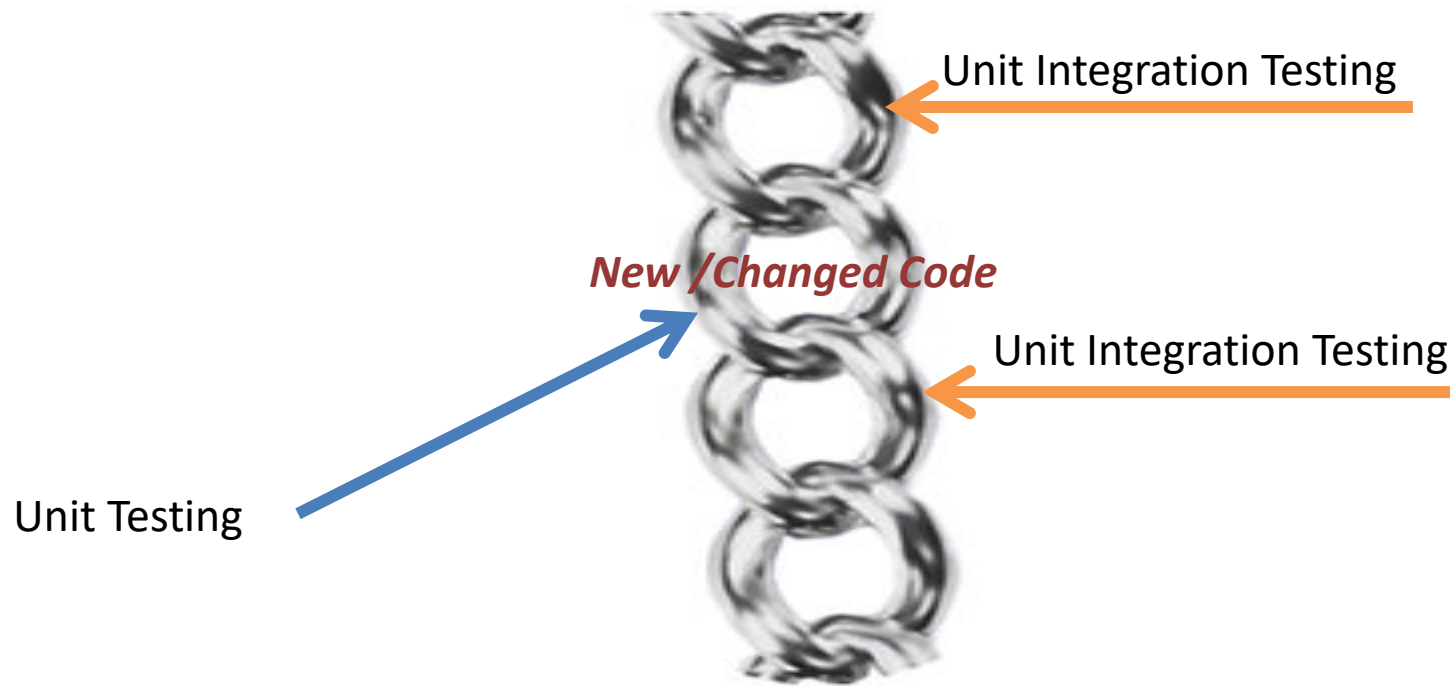


Unit Integration Testing

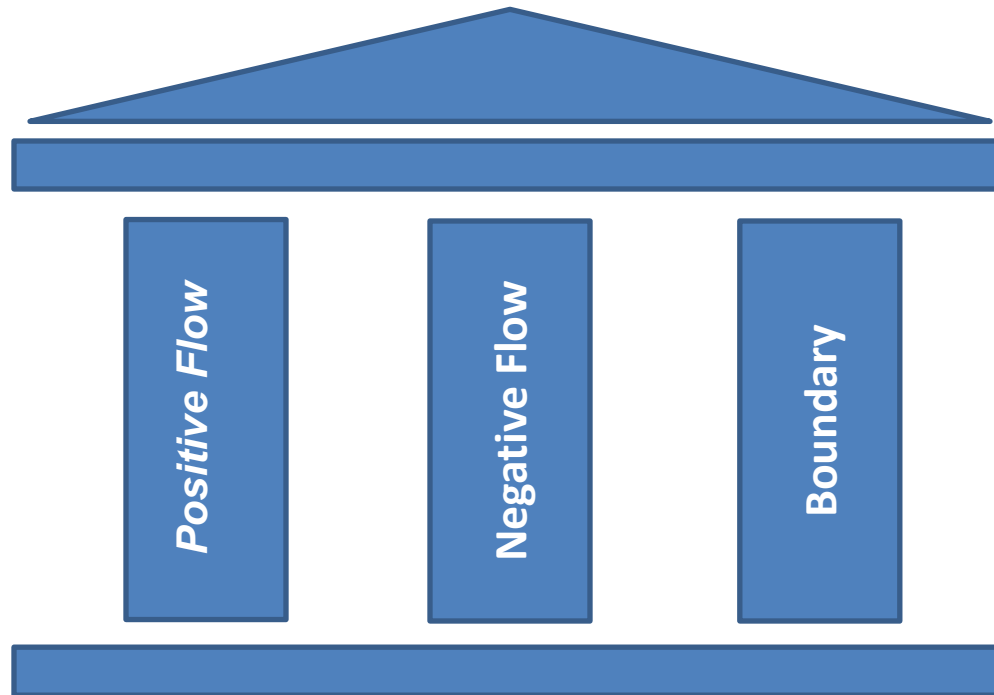
- Tests direct integration with the function, class, method, procedure, etc that is one layer up and one layer down from programmed code



Unit Testing vs. Unit Integration Testing



Three Pillars of Unit and Unit Integration



Positive Flow

- Ensure the unit of code works as expected under typical conditions.

Negative Flow

- Ensure the unit of code works can handle unexpected events or values. This includes error handling, exception handling, invalid data, etc.

Boundary

- Ensure data boundaries are working as expected

Levels of Competence in Skills for Unit Testing (QA)

1

Comprehending

2

**Defining Scenarios
for Development**

3

Authoring

Level **1** -Comprehending

- At this level tester learns to read unit tests to see what is tested.

Examples:

- Learn basic java
- Learn JUnit
- ✓ **Should be able to read methods and understand code and unit testing**

Level 2 -Defining Scenarios for Development

- At this level, QA should be able to identify unit testing scenarios for development and then Development will code the unit tests
 - ✓ Should be able to read methods, understand code, find scenarios and create unit testing matrix.



Level 3 -Authoring

- At this level, QA reads code and can author JUnit tests
 - *QA won't get to this level right away*
- ✓ Should be able to read methods, understand code, identify unit test scenarios and create Junit tests



Questions

