

Section 1: What is Software Testing?

- **Definitions and goals**

- An indication of the quality of a system in terms of defects found.
- A confirmation that a system works as expected
 - Secure
 - Usable
 - Performs well under load
 - Can recover after a crash
- An affirmation that an application was delivered according to what was required.
- A tool used for decision-making.



- What is Software Testing?

- **WHY?**

- To get rid of defects which can expose the company to fraud.
- Protect the Brand from ruin.
- Ensure we do not displease our customers.
- Reduce operation overhead because we are a great team!
- Ensure the system is easy to maintain and configure.
- 'Cos we are all rockstars. Rockstars test applications.

- **WHEN?**

- When requirements are presented or discussed. Requirement elicitation is important.
- Before code is written (TDD).
- While code is written.
- After code is written.

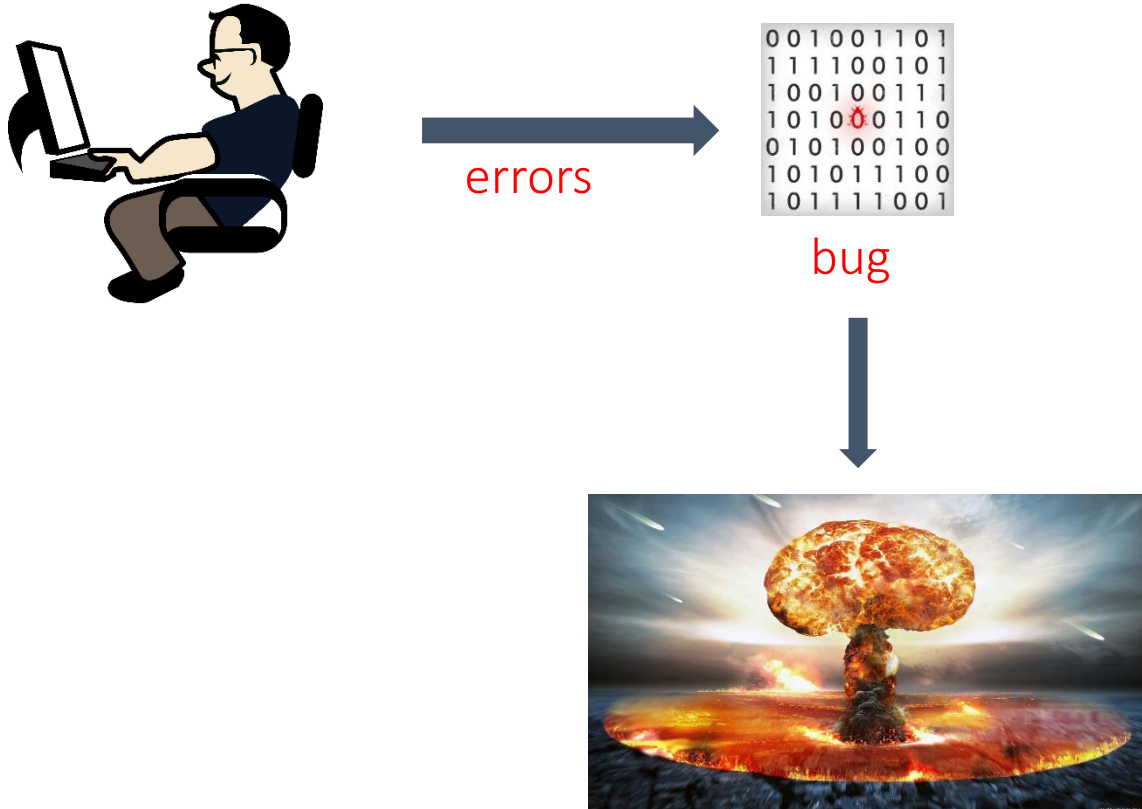
- **HOW MUCH IS ENOUGH?**

- Level of Risk
 - Financial
 - Brand Image/User experience
 - Sector-specific (Health, Aviation etc)
- Time constraint
- Budget constraint

Objectives of Testing

- To improve the quality and functionality of a system
- To prevent against hacking and fraud caused by system misuse
- To satisfy customer needs
- To make things easy for the end-users
- To ensure the company's reputation is protected and upheld.
- To reduce bugs... of course.
- To provide sufficient information for Stakeholders to make decisions about developed software.
- Reduce Operational/Monitoring overhead.

What is a Software Bug?



Application Blunders!

- St. Mary's Mercy Medical Center Kills Its Patients, On Paper: mapping error.
- A Stockbroking Firm Loses Nine Figures in 30 Minutes: bought high and sold low.
- World War III Narrowly Averted – a nuclear early warning system malfunctioned.
- ???
- ???
- ???
- ???

<https://www.intertech.com/Blog/15-worst-computer-software-blunders/>

How customers feel...

When software goes awry the resulting poor UX creates brand hostility. But, what constitutes 'going awry'? Awry could be anything from OS version incompatibility to input field glitches. (Let's not even talk about slow-load times on restricted networks or downtime due to code errors!)

- 88% of users form a negative opinion of a brand if it has a poorly performing app
- 79% of users are less likely to continue purchasing from a poorly performing platform
- 48% of users think poor software performance means the authoring company doesn't care about their users
- 34% of users will move to a competitor if they're not happy with your software.

<https://www.qualitylogic.com/2018/06/15/software-bug-brand-reputation/>

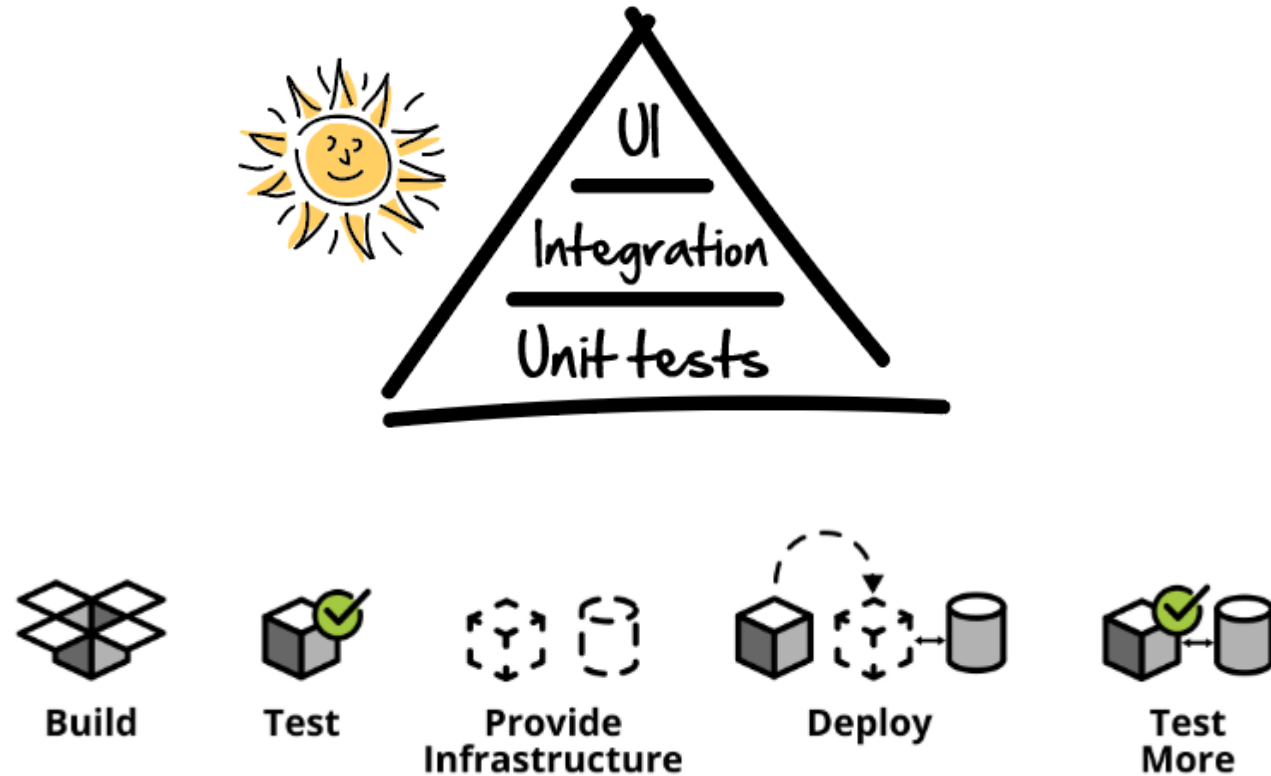
Who is a Software Tester?

Expectations...

Testing vs Testers

- Testers exercise and critique the implemented code
 - Testers are accountable for software quality
 - They are at a higher level of test independence
 - Testers analyze the functionality across systems
-
- ❖ Testing is everyone's responsibility
 - ❖ Testing is not performed only when code is completed. It should be done right from when requirements are provided.
 - ❖ Testing is not just about confirming software functionality. It is about breaking software.
 - ❖ Testing is on the side of user experience. It ensures the software is easy to use, not as the development team intends for it to be used.

Testing Pyramid and Continuous Testing



Some Principles of Testing

- Testing shows presence of defects, it does not prove the absence of them: even if there are no bugs found, it does not mean the application is bug-free.
- Exhaustive testing is impossible: all permutations of data cannot be used during testing activities.
- Early testing is smart: defects found earlier in the software development process are cheaper to fix.
- Pesticide paradox: the same set of tests will not catch new bugs. Old tests have to be updated or new tests written to exercise more parts of the system.
- Testing is context-dependent: testing methods differ with the type of application/system.

Section 2: How do we test Software?

Test Planning and Control

This is the process of specifying what to test and how testing is going to be done. Questions to be answered here are:

- Which features should be tested?
- What test tool/method is most suitable?
- How would test documentation be done?
- Which environment will the test run on?

Test control is an on-going process. Throughout the testing cycle, we check to see the outcomes are as earlier specified in the plan, regularize the deviation and ensure continuous reporting.

Test Analysis and Design

In this phase, the testing objectives are translated into test cases.

- Requirements are reviewed.
- System/architecture is analyzed, failure points are noted.
- Test scenarios are identified and documented.
- Test data are created/gathered.

The test environment is prepared here. Any additional tool/infrastructure is identified and steps taken to procure them.

Test Implementation and Execution

This is the stage in which test cases are run.

- Finalizing, implementing and prioritizing test cases.
- Developing and prioritizing test procedures, creating test data and writing automated test scripts
- Creating test suites for efficient test execution
- Executing tests either manually or by using test execution tools, according to the planned sequence
- Comparing actual and expected results and logging the results of tests.
- Re-testing and regression testing.

Exit criteria and test closure activities

- Checking test logs against the exit criteria specified in test planning
- Assessing if more tests are needed or if the exit criteria specified should be changed
- Writing a test summary report for stakeholders
- Closing incident reports or raising change records for any that remain open
- Learn lessons from testing to improve testing maturity and knowledge.

Test Independence

- Developers don't want to break their software, often because of the time it will take them to fix it. A builder of a house wants to complete it, an owner wants to find flaws.
- Bias from being too close to the work. Sometimes, errors are not seen.
- A bit of 'distance' from the code helps ensure more bugs are found and standards are followed.

Test Independence... cont'd

A certain degree of independence (avoiding the author bias) often makes the tester more effective at finding defects and failures. Independence is not, however, a replacement for familiarity, and developers can efficiently find many defects in their own code. Several levels of independence can be defined as shown here from low to high:

- Tests designed by the person(s) who wrote the software under test (low level of independence)
- Tests designed by another person(s) (e.g., from the development team)
- Tests designed by a person(s) from a different organizational group (e.g., an independent test team) or test specialists (e.g., usability or performance test specialists)
- Tests designed by a person(s) from a different organization or company (i.e., outsourcing or certification by an external body)

Login


Email or verified phone number

Password

☐ Remember me [Forgot password](#)


Login

[New User? Register here](#)




Adubiaro, Deborah
₦500

DA




Card



QR

Select or input your card number ▼



MM / YY

What's this?

CVV

☐ Add Card to Wallet

Pay

right app for your phone

Account Information

Please fill in the fields below.

Card Type

VISA *

Payment Amount

30.00

Card Number (No dashes or spaces)

*

Expiration

Select month Select year *

Cardholder Name

First Name

*

MI

Last Name

*

Credit Card Billing Street Address

*

City

*

State or Province

Select a state *

Postal Code (no dashes):

*

* indicates required information