**STQA Mini Project No. 1**

# Title

Create a small application by selecting relevant system environment/ platform and programming languages. Narrate concise Test Plan consisting features to be tested and bug taxonomy. Prepare Test Cases inclusive of Test Procedures for identified Test Scenarios. Perform selective Black-box and White-box testing covering Unit and Integration test by using suitable Testing tools. Prepare Test Reports based on Test Pass/Fail Criteria and judge the acceptance of application developed.

## Problem Definition

Perform Desktop Application testing using Automation Tool like JUnit generate Test Report by Using tool like Apache Maven.

## Prerequisite

Knowledge of Core Java, Basic Concepts of Unit Testing, Test Cases Writing using Junit etc tool

## Software Requirements:

JDK 1.8, Eclipse java photon-R version, TestNG

## Learning Objectives:

We are going to learn how to Prepare Test Cases inclusive of Test Procedures for identified Test Scenarios. Perform selective Black-box and White-box testing covering Unit and Integration test by using suitable Testing tools. also Prepare Test Reports based on Test Pass/Fail Criteria

## Outcomes:

You are able to understand Unit and Integration testing with Tool with Test Report.

## Theory Concepts:

* + 1. **What is Unit Testing?**

Unit Testing of software applications is done during the development (coding) of an application.

The objective of Unit Testing is to isolate a section of code and verify its correctness. In procedural programming a unit may be an individual function or procedure

The goal of Unit Testing is to isolate each part of the program and show that the individual parts are correct. Unit Testing is usually performed by the developer.

## Unit Testing Tools

There are several automated tools available to assist with unit testing. We will provide a few examples below:

* + - 1. [Jtest](https://prsft.co/2n7GdAM): Parasoft Jtest is an IDE plugin that leverages open-source frameworks (Junit, Mockito, PowerMock, and Spring) with guided and easy one-click actions for creating, scaling, and maintaining unit tests.By automating the setime-consuming aspect so for unit testing, it frees the

developer to focus on business logic and create more meaningful test suites.

* + - 1. [Junit](https://www.guru99.com/junit-tutorial.html): Junit is a free to use testing tool used for Java programming language.It provides assertions to identify test method. This tool test data first and then inserted in the piece of code.
      2. [NUnit](http://nunit.org/): NUnit is widely used unit-testing framework use for all .net languages. It is open source tool which allows writing scripts manually.It supports data-driven tests which can run in parallel.
      3. [JMockit](http://jmockit.github.io/index.html): JMockit is open source Unit testing tool. It is code coverage tool with line and path metrics.It allows mocking API with recording and verification syntax. This tool offers Line coverage, Path Coverage, and Data Coverage.
      4. [EMMA](http://emma.sourceforge.net/): EMMA is an open-source toolkit for analyzing and reporting code written in Java language. Emma support coverage types like method, line, basic block. It is Java-based so it is without external library dependencies and can access to the source code.
      5. [PHPUnit](https://phpunit.de/): PHPUnit is a unit testing tool for PHP programmer. It takes small portions of code which is called units and test each of them separately. The tool also allows developers to use pre- define assertion methods to assert that system behave in a certain manner.

Those are just a few of the available unit testing tools. There are lots more, especially for C languages and Java, but you are sure to find a unit testing tool for your programming needs regardless of the language you use.

## Extreme Programming & Unit Testing

Unit testing in Extreme Programming involves the extensive use of testing frameworks. A unit test framework is used in order to create automated unit tests. Unit testing frameworks are not unique to extreme programming, but they are essential to it. Below we look at some of what extreme programming brings to the world of unit testing:

* Tests are written before the code
* Rely heavily on testing frameworks
* All classes in the applications are tested
* Quick and easy integration is made possible

## Bug taxonomy

Bug taxonomies help in providing fast and effective feedback so that they can easily identify possible reasons for failure of the software. Using bug taxonomy, a large number of potential bugs can be grouped into few categories.

Whenever a new bug is reported, using bug taxonomy, a tester can easily analyse and put that bug into any of the categories.

At the end of testing, Testers can understand the type of categories of bugs that frequently occurred and thereby in successive rounds of testing he can focus on writing more test cases that would help to detect such bugs. In addition, test leaders can guide their testers to focus on such frequently occurring bugs.

The summary of the Bug Taxonomy is given below,

* Requirements, Features, and FunctionalityBugs
* StructuralBugs
* DataBugs
* CodingBugs
* Interface, Integration, and SystemBugs
* Test and Test DesignBugs
* Testing and DesignStyle

## What is Integration Testing?

In integration Testing, individual software modules are integrated logically and tested as a group. A typical software project consists of multiple software modules, coded by different programmers. integration Testing focuses on checking data communication amongst these modules. Hence it is also termed as 'I & T' (Integration and Testing), 'String Testing' and sometimes 'Thread Testing **Integration Test Case:**

Integration [Test Case](https://www.guru99.com/test-case.html) differs from other test cases in the sense itfocuses mainly on the interfaces & flow of data/information between the modules. Here priority is to be given for the integrating links rather than the unit functions which are already tested.

Sample Integration Test Cases for the following scenario: Application has 3 modules say 'Login Page', 'Mail box' and 'Delete mails' and each of them are integrated logically.

Here do not concentrate much on the Login Page testing as it's already been done in [Unit Testing](https://www.guru99.com/unit-testing-guide.html). But check how it's linked to the Mail Box Page.

Similarly Mail Box: Check its integration to the Delete Mails Module.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test**  **Case ID** | **Test Case Objective** | **Test Case Description** | **Expected Result** |
| **1** | Check the interface link between the  Login and Mailbox module | Enter login credentials and click  on the Login button | To be directed to the  Mail Box |
| **2** | Check the interface link between the Mailbox and Delete Mails Module | From Mail box select the an email and click delete button | Selected email should appear in the  Deleted/Trash folder |

## Desktop Application Testing by Using Junit Tool What is Junit?

JUnit is a framework for implementing testing in Java.

It provides a simple way to explicitly test specific areas of a Java program, it is extensible and can be employed to test a hierarchy of program code either singularly or as multiple units. Why use a testing framework? Using a testing framework is beneficial because it forces you to explicitly declare the expected results of specific program execution routes. When debugging it is possible to write a

test which expresses the result you are trying to achieve and then debug until the test comes out positive. By having a set of tests that test all the core components of the project it is possible to modify specific areas of the project and immediately see the effect the modifications have on the other areas by the

results of the test, hence, side-effects can be quickly realized.

JUnit promotes the idea of first testing then coding, in that it is possible to setup test data for a unit which defines what the expected output is and then code until the tests pass. It is believed by some that this practice of "test a little, code a little, test a little, code a little..." increases programmer productivity and stability of program code whilst reducing programmer stress and the time spent debugging.

JUnit is a simple open source Java testing framework used to write and run repeatable automated tests.

It is an instance of the xUnit architecture for unit testing framework. Eclipse supports creating test cases and running test suites, so it is easy to use for your Java applications.

JUnit features include:

* Assertions for testing expectedresults
* Test fixtures for sharing common testdata
* Test suites for easily organizing and runningtests
* Graphical and textual test numbers.
  + 1. **Conclusion**

In this way using JUnit and Maven Automation tool we are Perform Unit Testing and Prepare Test Report of same.