Software Quality Assurance HNDIT-4022

Introduction to Software Testing

What is Software?

Software is;

- Computer programs (Code)
- Procedures
- Documentation
- Data necessary for operating the software system

What is Quality?

Quality means “Performance upon expectations” and “fit for functions.”

What is Software Quality?

IEEE Definition of "Software Quality"

1. The degree to which a system, component, or process meets specified requirements.

1. The degree to which a system, component, or process meets customer or user needs or expectations.

What is Software Quality Assurance?

IEEE Definition of "Software Quality Assurance"

1. A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements.

1. A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with quality control.

What is Software Testing?

1. Testing is the process of executing a program with the intent of finding errors.

The Art of Software Testing by Glenford Meyers

1. Testing is running tests: but testing is not just running tests.
2. Testing activities exist throughout the life cycle of the project.

Why Testing is Necessary?

Most of us have had an experience with software that did not work as expected. Software that does not work can have a large impact on an organization. It can lead to many problems including:

- Loss of money
- Loss of time
- Damage to business reputation
- Injury or death

Why Testing is Necessary?

- ❑ Because software is likely to have faults **YES**
- ❑ To learn about the reliability of the software **YES**
- ❑ To fill the time between delivery of the software and the release date **NO**
- ❑ To prove that the software has no faults **NO**
- ❑ Because testing is included in the project plan **NO**
- ❑ Because failures can be very expensive **YES**
- ❑ To avoid being sued by customers **YES**
- ❑ To stay in business **YES**
- ❑ Confidence in quality of software **YES**

Quality Control Vs SQA

Quality Assurance is process oriented and focuses on defect prevention, while **quality control** is product oriented and focuses on defect identification.

Quality Assurance (QA) is meant to minimize the costs of quality by introducing a variety of activities throughout the development process and maintenance process in order to prevent the causes of errors, detect them, and correct them in the early stages of the development. As a result, quality assurance substantially reduces the rate of non-qualifying products

Testing and Debugging

- Testing and debugging are different.
- Executing tests can show failures that are caused by defects in the software.
- Debugging is the development activity that finds, analyzes, and fixes such defects. Subsequent confirmation testing checks whether the fixes resolved the defects.

Errors, Faults and Failures

- A person can make an error (mistake), which can lead to the introduction of a defect (fault or bug) in the software code or in some other related work product.
- An error that leads to the introduction of a defect in one work product can trigger an error that leads to the introduction of a defect in a related work product
- If a defect in the code is executed, this may cause a failure, but not necessarily in all circumstances.

Errors, Faults and Failures…

- Errors may occur for many reasons, such as:
  - Time pressure
  - Human fallibility
  - Inexperienced or insufficiently skilled project participants
  - Miscommunication between project participants, including miscommunication about requirements and design
  - Complexity of the code, design, architecture, the underlying problem to be solved, and/or the technologies used
  - Misunderstandings about intra-system and inter-system interfaces, especially when such intrasystem and inter-system interactions are large in number
  - New, unfamiliar technologies

  In addition to failures caused due to defects in the code, failures can also be caused by environmental conditions.

Reliability of Software

- Reliability - the probability that software will not cause the failure of the system for a specified time under specified conditions.

Three Views of Quality

Product view: – functionality, correctness, performance, usability …
Process view: – efficiency, cost & schedule, failures (rework).
Business view: – timeliness, customer satisfaction – does it sell? – producing the right product at the right time for the right price?

Cost of Quality

Cost of quality includes all costs incurred in the pursuit of quality or perform quality related work
Quality cost includes:
- Prevention cost: - quality planning - formal technical reviews - testing equipment - training
- Appraisal cost: - in-process and inter-process inspection - testing
- Failure cost: - rework, repair, and failure mode analysis - complaint resolution - product return and replacement

Activity:

1. Define, What is software quality?
2. Explain the importance of software testing.
3. Describe, what are the Errors, Faults and Failures?

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Software Reviews

What is Software Review?

- "A **software review** is a process or meeting during which a software product is examined by a project personnel, users, computers, user representatives, or other interested parties for comment or approval."
- A review is a type of testing where a group of people discuss the product produced by the developer, detect errors, and try to correct them in a systematic manner.
- Reviews play a key role in the software testing process.

Types of Reviews;

- ❏ Informal Review
- ❏ Formal Reviews

Informal Reviews

- Informal reviews take place between two or three people. The review conference is scheduled at their convenience.

- ❏ This meeting is generally scheduled during the free time of the team members.
- ❏ There is no planning for the meeting.
- ❏ If any errors occur, they are not corrected in the informal reviews.
- ❏ There is no guidance from the team.
- ❏ This review is less effective compared to the formal review.

Formal Reviews

- Formal reviews take place among a team of three to five members. In the formal review, the members discuss the software model.

- ❏ This meeting is scheduled beforehand. This gives the team members time to prepare.
- ❏ This meeting consists of a professional team that identifies and corrects errors in the model.
- ❏ This meeting does not exceed two hours.

Review Meeting

- At the end of the meeting, there is an acceptance stage. It involves the following three rules:

❏ If the project meets the expectations, they accept the model.
❏ If the project has a significant amount of errors, they reject the model, ask to modify it, and have another review meeting.
❏ If the project has only a few errors, the team asks to correct the mistakes, and there are no further review meetings.

Review Meeting...

In addition, a formal technical review summary report is taken.

This report answers three questions:

❏ What was reviewed?
❏ Who reviewed it?
❏ What were the findings and conclusions?

Types of Review in Software Testing

Three Types of Review in Software Testing

❏ Software Peer Reviews
❏ Software Management Reviews
❏ Software Audit Reviews

1. Software Peer Reviews

This type of review is conducted by the main author of the software, or it can be between the colleagues so that the evaluation can be done of the technical content or quality of the work.

Different types of Software Peer Reviews;

❏ Code Review
❏ Pair Programming
❏ Inspection
❏ Walkthrough
❏ Technical Review

2. Software Management Reviews

The management representatives are responsible for this type of review. The status of the work is evaluated and the decision by the activities of the software. This review is very important in making a decision regarding software.

3. Software Audit Reviews

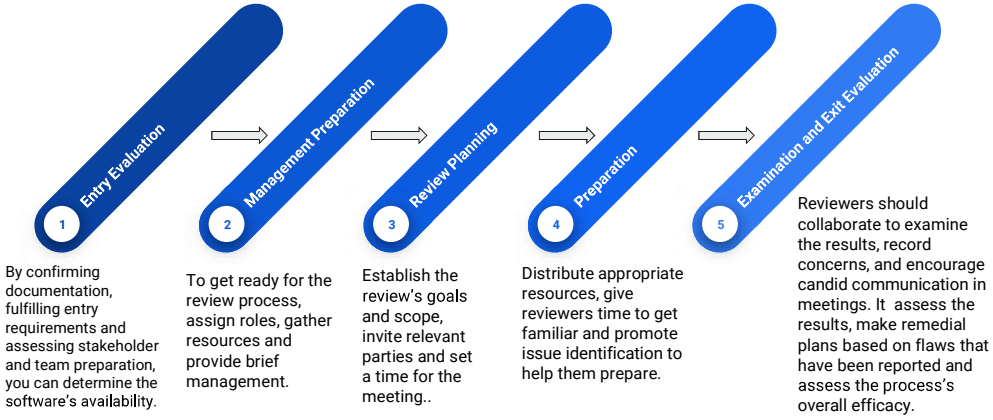
These are conducted by the personnel outside of the software project. They evaluate the software with specifications, standards, and other criteria.

Objective of Software Review

The objective of the software review is;

1. To improve the productivity of the development team.
2. To make the testing process time and cost-effective.
3. To make the final software with less defects.
4. To eliminate the inadequacies.

Process of Software Review



Advantages of Software Review

- ❏ Defects can be identified earlier stage of development (especially in formal review).
- ❏ Earlier inspection also reduces the maintenance cost of software.
- ❏ It can be used to train technical authors.
- ❏ Reviews can improve software development productivity and reduce development timescales.
- ❏ They can also reduce testing time and cost.
- ❏ Reviews generally reduce fault levels and lead to increased quality. This can also result in improved customer relations.
- ❏ Reviews are cost-effective.

Activity:

1. What is software review?
2. State different types of Software Peer review and describe them.
3. Describe the difference in between the Informal review and formal review.

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Static Verification Techniques

Software Verification Vs Validation

Verification is

Verification typically involves reviews and meetings to evaluate documents, plans, code, requirements, and specifications. This can be done with checklists, issues lists, walkthroughs, and inspection meetings.

Two Types of Verification;

- ❏ Dynamic verification, also known as Test or Experimentation - This is good for finding bugs
- ❏ Static verification, also known as Analysis - This is useful for proving correctness of a program although it may result in false positives

Software Verification Vs Validation

Validation is

- Validation typically involves actual testing and takes place after verifications are completed.

Software Verification Vs Validation...

- Software verification asks the question, "Are we building the product right?"; that is, does the software conform to its specification.

- Software validation asks the question, "Are we building the right product?"; that is, is the software doing what the user really requires.

Software Verification Vs Validation...

Lifecycle phase	Dynamic testing	Static testing
Requirements analysis and specification		x
Top-level design		x
Detailed design		x
Implementation	x	x
Acceptance testing	x	

What is Static Testing?

- Static Testing is a software testing technique that involves reviewing and analyzing:- software documentation – design, or code without executing the software. It is a way of checking for errors and defects in the software by examining it in a “static” state rather than actively running it.
- Static Testing aims to identify issues early in software development when they are easier and less costly to fix.
- By catching defects early, Static Testing can help improve the software’s overall quality and reliability.
- Also, it enhances maintainability and ultimately saves time and money in the long run.

What is Static Testing?...

There are two basic types of static testing.

1. People-based - People-based techniques are generally known as “reviews" There are different ways in which reviews can be performed.

2. Tool-based -The tool-based techniques examine source code and are known as "static analysis"

Why Static Testing?

- Static Testing is a valuable technique because it helps detect errors and defects in software before it is tested.
- For example, You review software requirements as part of Static Testing. You may identify potential conflicts that could be difficult to address later in the development cycle. Similarly, if you review code during Static Testing, you may spot issues like

  coding errors,

  inconsistent naming conventions

When to Perform Static Testing?

- Static testing is recommended to be done during the design, documentation, and development phases before dynamic testing commences.

Objectives of Static Testing

- Ensure all programming conventions, standards, and guidelines are followed.
- Verify that the source code is complete and can be compiled & built.
- Ensure that all design and implementation specifications are met.
- Identify areas of the software that may have logical, structural, formatting, or syntactic errors.

What is Subject to Static Testing?

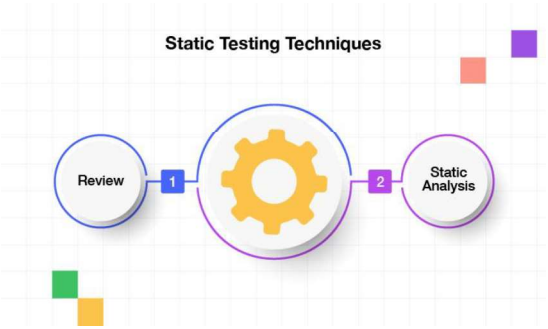
Here are some major features that are subject to static testing,

- Requirements & Specifications Review
- Structural Design Analysis
- Code Review
- Database Analysis
- Documentation Review
- User Manual Review

Static Testing Techniques

Static Testing is a necessary software testing technique comprising two approaches;

- ❏ Review
- ❏ Static Analysis



Static Testing Techniques...

- Review

  Reviews are a necessary feature of Static Testing. It enables testers to identify defects and issues in documentation, such as requirements and design. The importance of reviews lies in detecting the sources of failure at the earliest stage.

- Static Analysis

  Reviews are a necessary feature of Static Testing. It enables testers to identify defects and issues in documentation, such as requirements and design. The importance of reviews lies in detecting the sources of failure at the earliest stage.

Benefits of Static Testing

Static Testing provides several benefits to software development projects.

- Early defect detection
- Improved code quality
- Reduced costs and time
- Prevention of common issues
- Improved collaboration

Type of Static Testing

Static Testing can be covered under two different types:

- Manual Static Testing
- Automated Static Testing

Manual Method of Static Testing

Manual static testing methods are techniques used to identify errors and defects in software code and documentation without actually executing the code

- Inspections
- Walkthroughs
- Informal reviews
- Technical reviews
- Audits

Automation Method of Static Testing

This method leverages specialized tools to analyze source code, documentation, and other artifacts without execution.

Static Analysis : Tools dissect the code's structure, logic, and flow, identifying potential errors like:

- Syntactic errors
- Logic errors
- Security vulnerabilities
- Code quality issues
- Coding standard violations
- Code Reviews & Lint Checks
- Formal Methods

Automation Method of Static Testing...

- Code Reviews & Lint Checks: Tools scan the code for adherence to defined coding standards and best practices, highlighting areas for improvement.

- Formal Methods: Utilize mathematical techniques to prove the correctness of code properties and ensure they adhere to specific specifications.

Tools Used for Static Testing

Here are some tools;

- ❏ Checkstyle
- ❏ SourceMeter
- ❏ Soot
- ❏ Lint
- ❏ SonarQube
- ❏ PDM
- ❏ Findings

What is Tested in Static Testing?

Key areas of Static Testing;

- ❏ Code Quality
  - Syntax errors
  - Coding standards
  - Potential bugs
  - Performance bottlenecks
  - Security vulnerabilities

- ❏ Documentation Quality
  - Requirements completeness and consistency
  - Design document correctness
  - Test plan completeness and feasibility

How Static Testing is Performed?

How static testing is performed;

- ❏ The first step in static testing is planning.
- ❏ prepare the necessary artifacts, such as source codes, design documents, requirements documents, and test cases.
- ❏ Static analysis is the core of this testing.
- ❏ Code reviews are an essential component of the static testing process.
- ❏ Small team of experts systematically reviews the code and finds potential errors using various methods.
- ❏ Any issues or bugs during the static testing process are reported and documented.
- ❏ The results collected during the static testing process should be analyzed to determine the quality of the software product. This is the final step in static testing.

Advantages of Software Review

- ❏ Defects can be identified earlier stage of development (especially in formal review).
- ❏ Earlier inspection also reduces the maintenance cost of software.
- ❏ It can be used to train technical authors.
- ❏ Reviews can improve software development productivity and reduce development timescales.
- ❏ They can also reduce testing time and cost.
- ❏ Reviews generally reduce fault levels and lead to increased quality. This can also result in improved customer relations.
- ❏ Reviews are cost-effective.

Activity:

1. What is Static Testing?
2. What is tested in static testing?
3. Who inspects the documents in static testing?
4. State the different between Static testing and Dynamic testing.

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Software Testing Techniques

Software Testing Techniques

Testing Techniques:

Testing techniques are the means used by testers to accomplish their test objectives.

Software Testing Techniques help you design better test cases. Since exhaustive testing is not possible;

Manual Testing Techniques help reduce the number of test cases to be executed while increasing test coverage.

They help identify test conditions that are otherwise difficult to recognize.

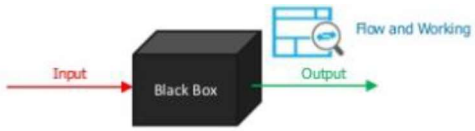
Type of Software Testing Technique

- Black Box Testing
- White Box Testing
- Gray Box Testing

Black Box Testing

Black Box Testing : tester tests the application.software by applying different inputs and comparing the output with expected results.

It is also known as Behavioural Testing.



Techniques in Black Box Testing

- Boundary Value Analysis
- Equivalence class partitioning
- Decision Table based testing
- State Transition
- Error guessing

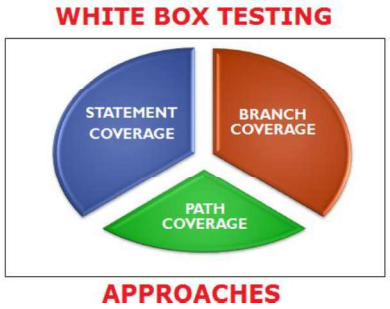
White Box Testing

White box testing is a verification technique where software engineers can use to examine if the code works as expected or not.

It is the strategy testing is based on

- Software's internal paths
- Design and coding
- Logics
- Structures
- Implementation of software under test

Techniques in White Box Testing



Gray Box Testing

It is a software testing methodology to test the software with partial knowledge of the code in the application—a combination Black and White box testing.

Techniques in Gray Box Testing

- Matrix Testing
- Regression Testing
- Orthogonal Array Testing
- Pattern Testing

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Test Case Design & Test Cases

What's a 'Test Case'?

Test Case is;

A test case is a set of actions performed on a system to determine if it satisfies software requirements and functions correctly.

A test case describes an input, action, or event and an expected response, to determine if a feature of a software application is working correctly.

Test Case-Definitions:

- **IEEE Standard 610 (1990) defines test case as follows:**
"A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement."

- **IEEE Standard 829 (1983) defines test case as follows:**
"Documentation specifying inputs, predicted results, and a set of execution conditions for a test item."

- **According to Ron Patton (2001):**
"Test cases are the specific inputs that you'll try and the procedures that you'll follow when you test the software."

- **Boris Beizer (1995):**
"A sequence of one or more subtests executed as a sequence because the outcome and/or final state of one subtest is the input and/or initial state of the next. The word 'test' is used to include subtests, tests proper, and test suites."

Why we need test cases?

We will write the test for the following reasons:

- To require consistency in the test case execution
- To make sure a better test coverage
- It depends on the process rather than on a person
- To avoid training for every new test engineer on the product

Types of Test Cases

- Positive Test Cases
- Negative Test Cases

Elements of a Test Case

Followings are the elements of test cases:

- Test Case ID
- Test Scenario
- Test Case Description
- Test Steps
- Prerequisite
- Test Data
- Expected Result
- Actual Result
- Environment Information
- Comments

Attributes of a Good Test Case

Attributes of a good test case are as follows:

- Accurate
- Economical
- Repeatable, self-standing
- Appropriate
- Traceable

Test Case Design Techniques

What is a Test Design Technique?

Test case design technique is a procedure for selecting or designing tests based on a structural or functional model of the software. It is successful at finding faults and deriving good test cases.

Categories of Test Design Techniques

Test design techniques can be widely categorized into three different areas as follows:

1. Specification-Based or Black Box Design Techniques
2. Structure based or White Box Design Techniques
3. Experienced Based Testing Techniques

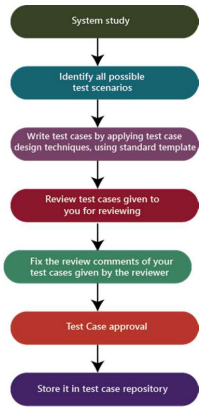
Advantages of Test Design Techniques

- Test Design finds Faults in early stage.
- Faults found early are cheaper to fix.
- Early test design technique can prevent faults from entering into the system.
- Change requests can be reduced considerably through test design.
- Early test design helps to build quality and stops fault multiplication.

How to Choose a Test Design Technique?

- Following are the list of factors that influence the selection of the appropriate test design technique for a particular kind of problem:
- The type of system,
- Risk identified,
- Customer requirements,
- Models for use case modeling,
- Requirements models or tester knowledge.

The Process To Write Test Cases

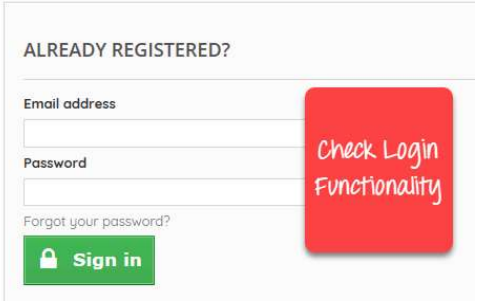


Best Practice for writing good Test Case.

1. Test Cases need to be simple and transparent
2. Create Test Case with End User in Mind
3. Avoid test case repetition
4. Do not Assume
5. Ensure 100% Coverage
6. Test Cases must be identifiable
7. Implement Testing Techniques
8. Self-cleaning
9. Repeatable and self-standing
10. Peer Review

The Process To Write Test Cases

Example : Create a Test Case for the scenario: Check Login Functionality.

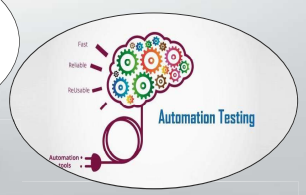


Q&A's ?

Thank you !

Software Quality Assurance
HNDIT-4022

AUTOMATION TESTING



CONTENT

01.What is automation testing?

- Software testing techniques.
- Used special automation tool.
- Practice of running test automatically.
- The execution of tests and then compares actual test results with predicted or expected results.
- This is done automatically with little or no intervention from the test engineer.

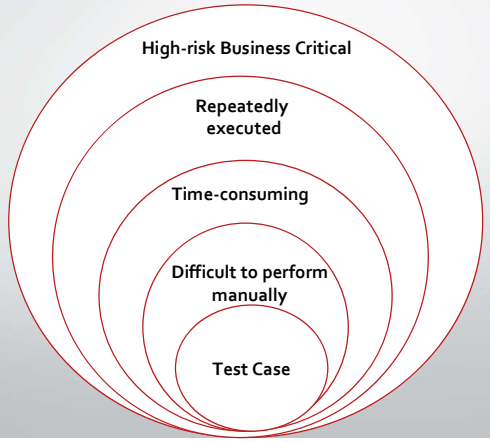
02.Why automation testing?

- Automated testing saves money and time.
- Vastly increase your test coverage.
- Testing improves accuracy.
- Automated test helps developer and testers.
- Automation test does what manual test cannot.
- Speed of your test execution.

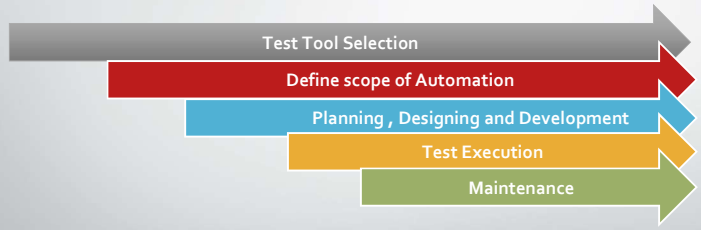
03.Manual testing VS automation

- ***Manual testing*** is done by QA tester and ***Automation testing*** is done automatically using automation tools and script.
- ***Manual testing*** is time consuming with less efficient and in ***Automation*** more testing less time and greater efficiency.
- ***Manual testing*** is less reliable and ***Automation testing*** more reliable.
- ***Manual testing*** needs have a more straightforward test execution setup and ***Automation testing*** requires less complex test execution set up.
- ***Manual testing*** No need for programming and ***Automation testing*** Programming knowledge is a must.

04.Test case types to automate



05.Automation Testing Process

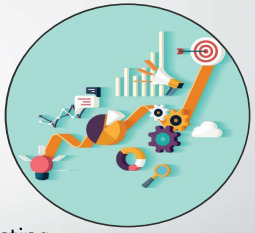


05.Automation Testing Process (cotd.)

1.Test Tool Selection – criteria

- What is the budget of your organization.
- What is the actual price of your selected tool.
- Does tool support Operating System /Device which your application runs.
- Does the tool support the technologies and third party controls.
- Does th tool support connecting to different data source.
- How the report mechanism of testing tool.
- Can the tool be integrated with test case and bug management repositories?

2.Define Scope Of Automation



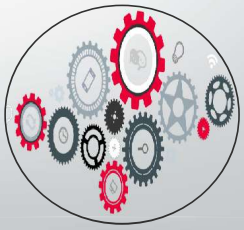
- Business Scenarios.
- Description of manual test cases.
- Common Functionalities related to applications.
- Technical Feasibility.
- Test Case complexity.
- Ability to use the same test cases for cross-browser testing.
- Reused business components.
- End – to – End flows.
- Validation.

3.Planing , Designing and Development

- Automation tool selection.
- Framework designing with features.
- Items categorization with in-scope and out-scope.
- Automation testbed preparation.
- Schedule and Timeline of scripting and execution
- Deliverables of Automation Testing

4.Test Execution

- Test script are run in this phase.
- Ready with input test data.
- Use automation tool or Test Management Tool (**Eg:-Quality center**)
- Generated report with related detail after executed.



5.Maintenance

- Check whether the software is working with added new features or not.
- This phase work when new automation test script are added.
- Therefore need for reviewed and maintained.
- To improve the effectiveness of automation scripts with each successive release cycle.

06.Framework for automation

- *1.Data Driven Automation Framework*
- *2. Keyword Driven Automation Framework*
- *3.Modular Automation Framework*
- *4. Hybrid Automation Framework*

7.Different type of software test that can be automated

- Unit Testing.
- Integration Testing.
- Functional Testing.
- Keyword Testing.
- Regression Testing.
- Data – Driven Testing.
- Black-box testing.
- Smoke Testing.

8.Automation Testing Tools.

1.Commercial Automation Tools

- IBM Rational Functional Tester
- Katalon studio
- Ranorex
- Sahi pro
- Apache JMeter
- Appium
- RedHoodHQ
- Cypress

8.Automation Testing Tools (cotd).

2.Open Source DevOps Automation Testing Tools

- Citrus
- Galen
- Karate – DSL
- Robot Framework
- Selenium
- Watir
- Gauge

9.The Dark side of automation testing

- Automating the wrong things.
- The tools you picked can't solve your problems.
- The tools you selected aren't a good fit for your testers.
- You failed to add up the total cost of tool ownership.
- You chose a tool just because it's open source.
- You didn't foster a testing culture that embraces automation.

Software Quality Assurance
HNDIT-4022

SELENIUM – TEST AUTOMATION TOOL

CONTENT

- Selenium is.
- Features of selenium.
- Advantages of selenium.
- Components of selenium.
- References.

1.SELENIUM IS

- Open source License.
- Browser based web application testing tool
- Support for multiple browsers.
- Extensible framework.
- Selenium Software is not just a single tool but a suite of software.
- Support mobile testing (Android,OS)
- Write test cases using language of choice java,c#.
- Support Native Browser Testing.

2.FEATURES OF SELENIUM.

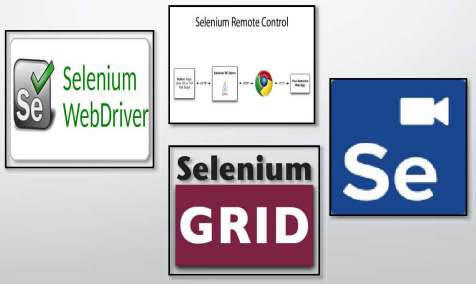
- Selenium is functional automation tool for web application.
- Selenium is an open source tool.
- Selenium supports the language like HTML,Java,PHP,Perl,Python,Ruby, and C#.
- It supports the browser like IE,Mozilla Firefox,Safari,Google Chrome and Opera.
- It supports the operating systems like Windows,Linux,Mac.
- It is very flexible.

3.ADVANTAGES OF SELENIUM.

- Free of cost for company because of open source software.
- Customize according to the requirements.
- Ajax based system.
- Support native testing for all major browsers.
- It can be integrated with tools such as **TestNG** & **JUnit** for managing test cases and generating reports.
- Support for any operating system.

4.COMPONENTS OF SELENIUM

- Selenium IDE.
- Selenium RC.
- Selenium Web Driver.
- Selenium Grid.

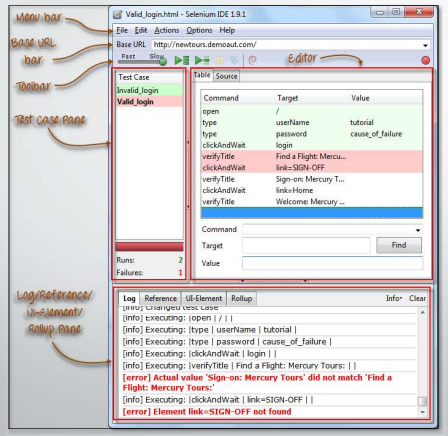


4.COMPONENTS OF SELENIUM (COTD.)

1.Selenium IDE.

- IDE stand for **Integrated Development Environment**.
- It was a Firefox plugin for faster creation of test cases.
- Which is use for Record and Play back the script.
- Selenium IDE is a accountable for user action.
- We can run the recorded scripts against other browsers by using selenium RC.
- Tests recorded via the plugin can be exported in different programming languages like: Java, Ruby, Python etc.

1.Selenium IDE(contd.) - Features



1.Selenium IDE(contd.) – Pros & Cons

Pros

- Very easy to use and install.
- No programming experience is required, through knowledge of HTML and DOM are needed.
- Can export tests to formats unable in Selenium RC and WebDriver.
- Has built-in help and test results reporting module.
- Provide supports for extensions.

Cons

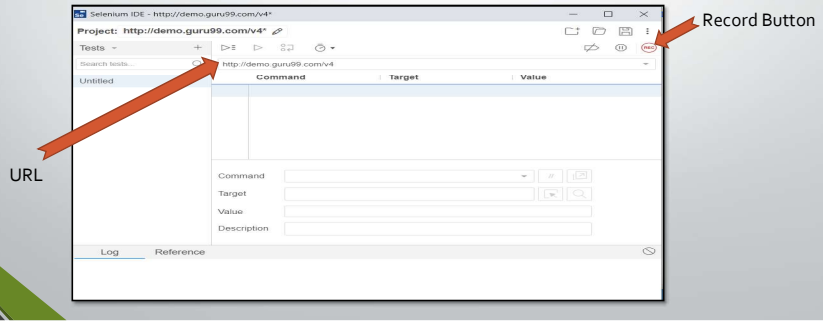
- Available only in Firefox.
- Designed only to create prototypes of tests.
- No support for iteration and confidential operations.
- Test execution is slow compared to that of selenium RC and WebDriver.

Selenium IDE(contd.) – Demo

1.Record and playback selenium IDE Test Case

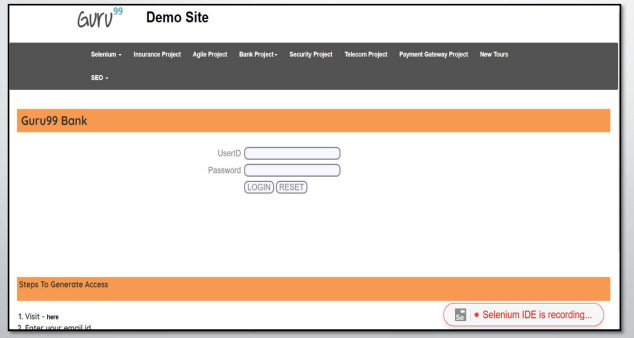
Steps to Record Selenium IDE Test Script

1. Launch Firefox and Launch Selenium IDE by Clicking on ***Tools* => *Selenium IDE***.
2. Type the Base URL value: ***http://demo.guru99.com/v4***
3. Click on the ***Record*** button to start the recording of the Test Case.

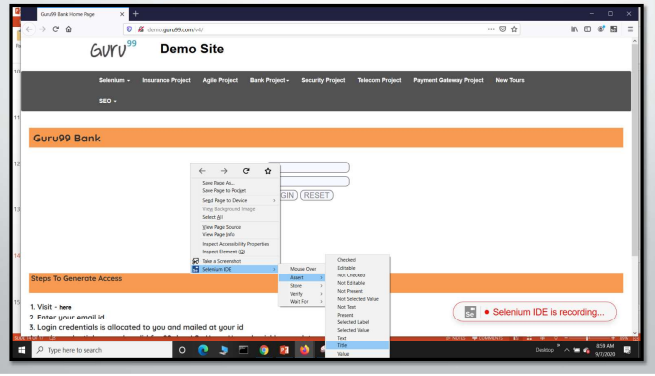


Steps to Record Selenium IDE Test Script (contd.)

4.In Firefox, navigate to ***http://demo.guru99.com/v4*** Firefox should take you to the page similar to the one shown below.

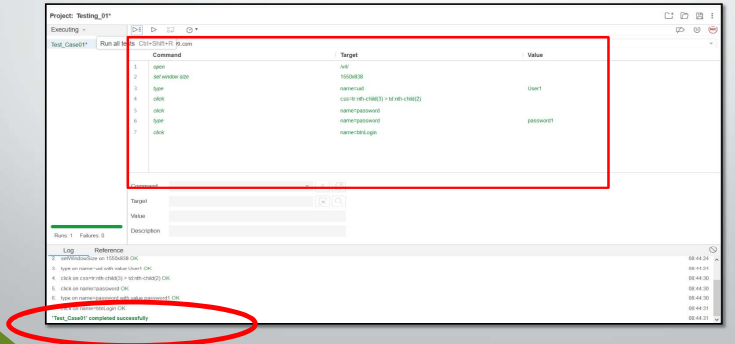


5.Right-click on any blank space within the page, this will bring up the Selenium IDE context menu. Select the ***Show IDE*** option. Then, select ***assert*** then, select ***Title*** This is a command that makes sure that the page title is correct.



6.Enter Username ***User1*** in the first text box and Password ***Password1*** in the second text box. Then click on ***Login*** button to submit.

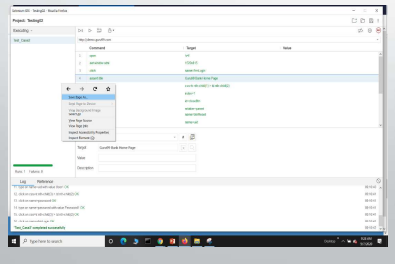
7. Stop the recording by clicking on the ***Record*** button. In the ***Table*** tab of ***Test Step Pane*** you can see the commands recorded by selenium IDE.



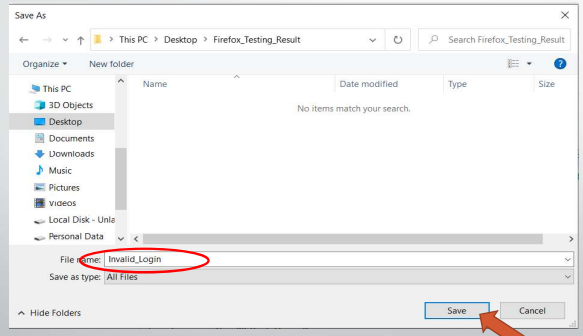
2.Steps to Save Selenium IDE Test Script

1. Save the test case for future use by clicking on ***Save project icon*** or right click and ***click save as***

Save project icon

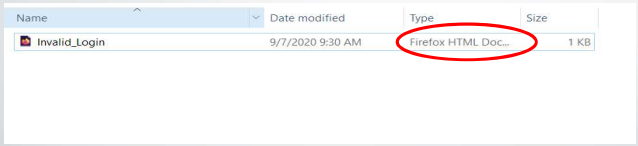


2. Choose your location, and then name the test as ***Invalid Login***. Click the ***Save*** button.



Save

3. The file was saved as HTML.



3.Steps to PlayBack Selenium IDE Test Script

Run all tests

1.Selenium IDE(contd.) - Selenium IDE Commands (Selenese)

1.Actions

Generally manipulate the state of the application.

Eg:-type this box, click this link or select option

If an Action fails current test is stopped.

Many Actions can be called with the **AndWait** suffix, e.g. **clickAndWait**, **typeAndWait**.

2.Accessors

These commands examine the state of the application and store the result in variables.

Eg:-**storeTitle**

They are also used to automatically generate Assertions.

3.Assertions

These commands are like ***accessors***, but they verify that the state of the application conforms to what is expected.

Selenium Assertions can also be categorized into three categories. **Eg;-Assert, Verify, WaitFor**

1.Selenium IDE(contd.) - Commonly used Selenium IDE commands

Additional Reading :- https://ui.vision/rpa/docs/selenium-ide

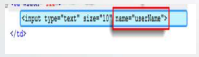
Command	Description
1.type	*Sets the value of an input field, as though you typed it in*
2.open	*Opens a page using a URL.*
3.click	*Clicks on a link, button, checkbox or radio button.*
4.assertTitle, VerifyTitle	*Verifies an expected page title*
5. assertElementPresent, verifyElementPresent	*Verify / Asserts the presence of an element on a web page.*
6. assertTextPresent, verifyTextPresent	Verify / Asserts the presence of a text within the web page.
7.type, typeKeys, sendKeys	*Enters a value (String) in the specified web element.*

Command	Description
8. Click, clickAt, clickAndWait	*Clicks on a specified web element within a web page.*
9. waitForPageToLoad	*Sleeps the execution and waits until the page is loaded completely.*
10. waitForElement Present	*Sleeps the execution and waits until the specified element is present*
11.chooseOkOnNext Confirmation, chooseCancelOn NextConfirmation	*Click on "OK" or "Cancel" button when next confirmation box appears.*
12. highlight	*Briefly changes the background Color of the specified element yellow. Useful for debugging.*
13. echo	*Prints the specified message into the third table cell in your Selenese tables. Useful for debugging.*
14.pause	*Wait for the specified amount of time (in milliseconds).*

1.Selenium IDE(contd.) - Locators

- These are the command which are like GUI elements.
- Identification of correct GUI elements is a prerequisite to creating an automation script.
- But accurate identification of GUI elements is more difficult than it sounds.
- Since Selenium provides locators to precisely locate a GUI element.

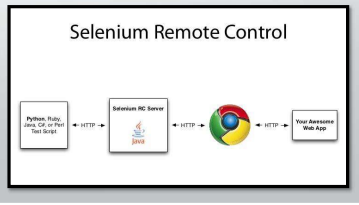
1.Selenium IDE(contd.) - Locators

Locator	Define	Description
1. Locating by ID	id=*id of the element* 	Way of locating elements since ID's
2. Locating by name	name=*name of the element* 	Locating elements by name.
3.Locating by Link Text	*link=link_text* 	This type of locator applies only to hyperlink texts.
4.Locating by CSS Selector	*css=tag#id*	CSS Selectors are string patterns used to identify an element based on a combination of HTML tag, id, class, and attributes.

4.COMPONENTS OF SELENIUM (COTD.)

2.Selenium RC

- **Flagship Testing Framework** of the whole Selenium project for a long time.
- **Allow users to use a programming language they prefer.**
- **RC** version 2.25.0 support for C#,PHP,and Java.
- Make request using **GET/POST method** and capable of setting **HTTP requests.**
- Automate web application.



2.Selenium RC(contd.) – Pros & Cons

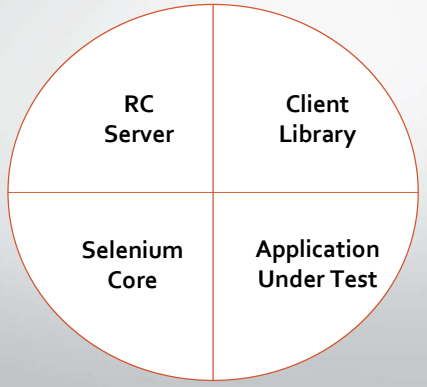
Pros

- Cross browser and platform.
- Can perform looping and conditional operations.
- Support for data-driven testing.
- Has matured and complete API.
- Support for new browsers.
- Faster execution when comparing with IDE.

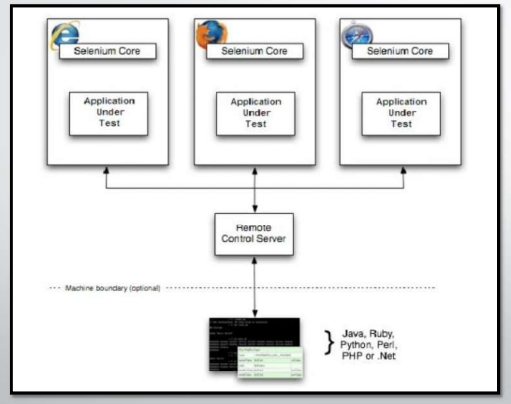
Cons

- Installation is not easy when comparing with IDE.
- User need technical knowledge regarding programming language.
- Need Selenium RC server to execute.
- API commands are more confusing.
- Less realistic browser interaction.

2.Selenium RC(contd.) – Main components



2.Selenium RC(contd.) – Overview of Selenium RC



2.Selenium RC(contd.) – Installation

- https://www.protechtraining.com/content/selenium_tutorial-selenium_remote_control

4.COMPONENTS OF SELENIUM (COTD.)

Selenium WebDriver

3.Selenium WebDriver

- Better than both **Selenium IDE and Selenium RC.**
- Does not rely on JavaScript for Automation.
- It controls the browser by directly communicating with it.
- Support language ;- Java, C#, PHP, Python, Perl, Ruby.
- It mainly supports browsers like Firefox, Chrome, Safari and Internet Explorer.
- You can now use **conditional operations** like ***if-then-else or switch-case***. You can also perform looping like ***do-while***.

3.Selenium WebDriver(contd.) – Pros & Cons

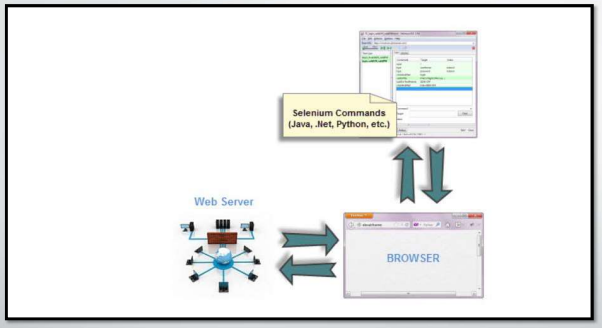
Pros

- Easy to install comparing with RC.
- Directly connect with browser.
- Support for data-driven testing.
- Browser interaction is more realistic.
- Execution time is fast than RC and IDE.

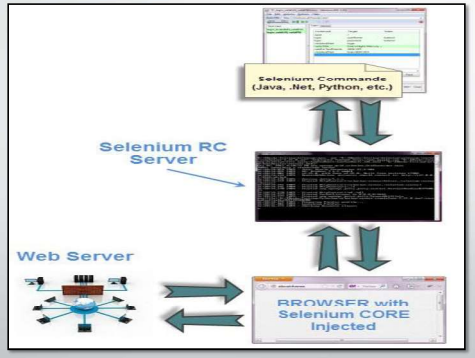
Cons

- But installation is complicated when comparing with selenium IDE.
- Need programming language.
- Not support readily for new browsers.
- API commands are more confusing.
- There was no any built-in mechanism to generate result.

3.Selenium WebDriver(contd.) – Architecture

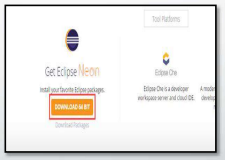


3.Selenium WebDriver(contd.) – Architecture with RC



3.Selenium WebDriver(contd.) – Installation

- Install Java SDK
- Install Eclipse
- Install Selenium Driver Files



3.Selenium WebDriver(contd.) – Install Selenium Driver Files

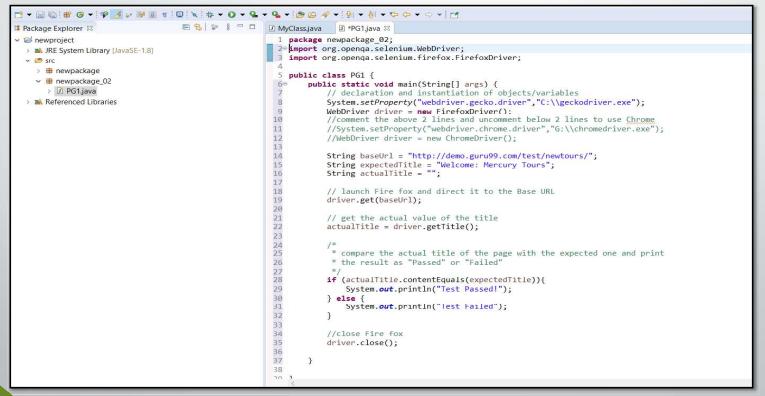
1. Download the **Selenium Java Client Driver.**

2. Configure Eclipse IDE with WebDriver

https://www.guru99.com/installing-selenium-webdriver.html

First Selenium Web driver Script

3. First Selenium Web driver Script



1.Instantiating objects and variables

The default Firefox profile will be launched by our Java program

```
WebDriver driver = new ChromeDriver();
```

2.Launching a Browser Session

get() method is used to launch a new browser session and directs it to the URL

```
String baseUrl = "http://demo.guru99.com/test/newtours/";

driver.get(baseUrl);
```

3. Get the Actual Page Title

getTitle() method that is always used to obtain the page title of the currently loaded page.

```
actualTitle = driver.getTitle();
```

4. Compare the Expected and Actual Values

simply uses a basic Java if-else structure to compare the actual title with the expected one..

```
if (actualTitle.contentEquals(expectedTitle)){
    System.out.println("Test Passed!");
} else {
    System.out.println("Test Failed");
}
```

5. Terminating a Browser Session

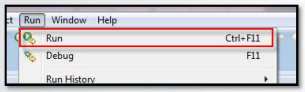
simply uses to close the browser window...

```
//close Fire fox
driver.close();
```

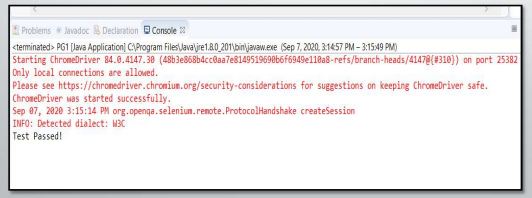
4. Running the Test

On Eclipse's menu bar, click **Run > Run.**

Press **Ctrl+F11** to run the entire code.



5. Final Result – Console View

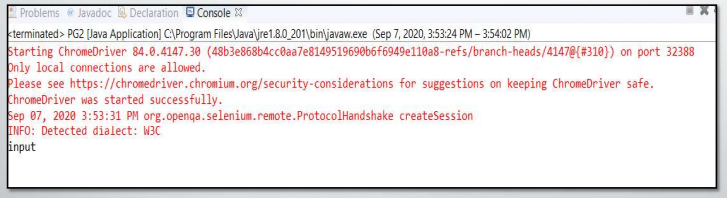


Locating GUI Elements

Variation	Description	Sample
By.**class** Name	finds elements based on the value of the "class" attribute	findElement(By.className("someClassName"))
By.**cssSe** lector	finds elements based on the driver's underlying CSS Selector engine	findElement(By.cssSelector("input#email"))
By.**id**	locates elements by the value of their "id" attribute	findElement(By.id("someId"))
By.**linkTe** xt	finds a link element by the exact text it displays	findElement(By.linkText("REGISTRATION"))
By.**name**	locates elements by the value of the "name" attribute	findElement(By.name("someName"))
By.**partia** lLinkText	locates elements that contain the given link text	findElement(By.partialLinkText("REG"))
By.**tagNa** me	locates elements by their tag name	findElement(By.tagName("div"))
By.**xpath**	locates elements via XPath	findElement(By.xpath("//html/body/div/table/tbody/tr/td[2]/table/tbody/tr[4]/td/table/tbody/tr/td[2]/table/tbody/tr[2]/td[3]/ form/table/tbody/tr[5]"))

```
1  package newpackage_03;
2  import org.openqa.selenium.By;
3  import org.openqa.selenium.WebDriver;
4  import org.openqa.selenium.chrome.ChromeDriver;
5
6  public class PG2 {
7      public static void main(String[] args) {
8          System.setProperty("webdriver.chrome.driver","C:\\Users\\Asus\\Pictures\\Sele\\chromedriver.exe");
9          WebDriver driver = new ChromeDriver();
10         String baseUrl = "http://www.facebook.com";
11         String tagName = "";
12
13         driver.get(baseUrl);
14         tagName = driver.findElement(By.id("email")).getTagName();
15         System.out.println(tagName);
16         driver.close();
17         System.exit(0);
18 }
19
20 }
21
```

We used the **getTagName()** method to extract the tag name of that particular element whose id is "email". When run, this code should be able to correctly identify the tag name "input" and will print it out on Eclipse's Console window.



Closing and Quitting Browser Windows

quit(), all windows will be closed - not just the parent one. Try running the code below and you will notice that the two pop-ups above will automatically be closed as well.

that only the parent browser window was closed and not the two pop-up windows.

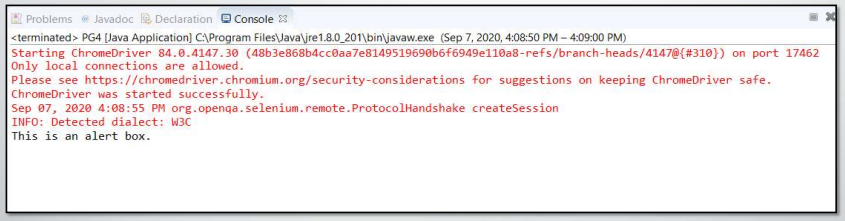
```
package newpackage_04;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class PG3 {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Asus\\Pictures\\Sele\\chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("http://www.popuptest.com/popuptest2.html");
        driver.quit();  // using QUIT all windows will close
    }

}
```

Switching Between Pop-up Windows

WebDriver allows pop-up windows like alerts to be displayed, unlike in Selenium IDE. To access the elements within the alert (such as the message it contains), we must use the **"switchTo().alert()"** method.

```
package newpackage_05;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class PG4 {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Asus\\Pictures\\Sele\\chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        String alertMessage = "";

        driver.get("http://jsbin.com/usidix/1");
        driver.findElement(By.cssSelector("input[value=\"Go!\"]")).click();
        alertMessage = driver.switchTo().alert().getText();
        driver.switchTo().alert().accept();

        System.out.println(alertMessage);
        driver.quit();

    }

}
```

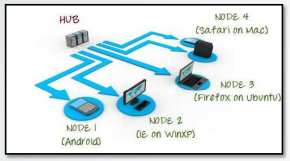
On the Eclipse console, notice that the printed alert message is:



4.COMPONENTS OF SELENIUM (COTD.)

- **4.Selenium Grid.**
- Used together with Selenium RC to run parallel tests (running multiple tests at once)
- Across different machines and different browsers all at the same time.
- Since Saves time enormously.
- Utilizes the **hub-and-nodes** concept(The hub acts as a **central source of Selenium commands** to each node connected to it)
- To run a huge test suite, that needs to complete in the soonest time possible.

4.Selenium Grid (cotd.) - hub-and-nodes concept



Hub	Node
1.Central point where you load your tests into	1.Selenium instances that will execute the tests that you loaded on the hub
2. Only be one hub	2.Can be one or more nodes
3. Launched only on a single machine	3. Can be launched on multiple machines with different platforms and browsers.
4. The machine containing the hub is where the tests will be run	4.The machines running the nodes need not be the same platform as that of the hub.

5.REFERENCES

- https://www.guru99.com/introduction-selenuim-ide.html
- https://www.guru99.com/first-webdriver-script.html
- https://www.guru99.com/introduction-to-selenium-grid.html#:~:text=Selenium%20Grid%20is%20a%20part,server%20acts%20as%20a%20hub.
- https://www.edureka.co/blog/what-is-selenium/
- https://www.toolsqa.com/selenium-ide/locators-in-selenium-ide/

Software Quality Assurance
HNDIT-4022

Performance testing and reporting with -

Apache JMeter

Agenda

- Introduction
- History
- Protocols
- Features
- How it works?
- Installation
- Thread Group
- Controller
- Execution Order
- Demo

Introduction

JMeter is a software that can perform load test, performance test, regression test ,stress test etc.. On the different protocols and the technology,

- JMeter is an Open Source testing software.
- It is 100% pure Java application for load and performance testing.
- JMeter is designed to cover categories of tests like load, functional, performance, regression, etc.

History

- **Stefano Mazzocchi** of the Apache Software Foundation was the original developer of JMeter. He wrote it primarily to test the performance of Apache JServ (now called as Apache Tomcat project). Apache later redesigned JMeter to enhance the GUI and to add functional testing capabilities.

Protocols

The following protocols are supported by JMeter,

- Web - HTTP, HTTPS
- Web services - SOAP / REST
- FTP services
- Database via JDBC drivers
- Directory - LDAP
- Message-oriented middleware (MOM) via JMS
- Mail - SMTP(S), POP3(S) and IMAP(S)
- Native commands or shell scripts
- TCP

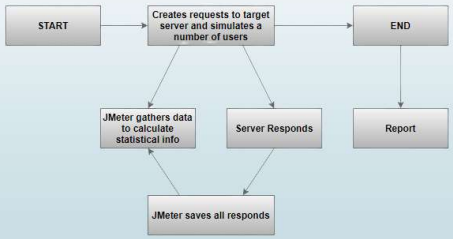
Features

Followings are some of the feature of JMeter;

- Multithreading Framework
- Compatible with TCP
- 100% Java scripted
- Compatible with LDAP
- Data Analysis and Visualization
- Pluggable Samplers
- Compatible with Database via JDBC
- GUI Design and Interface
- Dynamic Input
- Compatible with SOAP / REST
- Result Analysis and Caches
- Compatible with Native Commands
- Highly Extensible Core
- Compatible with FTP
- Compatible with Web – HTTP, HTTPS
- Scriptable Samplers
- Compatible with Mail – SMTP(S)
- Compatible with POP3(S) and IMAP(S)
- Compatible with Message-oriented middleware via JMS

How it works?

- Take a look at the following figure that describe how JMeter works,



What we can do with JMeter

- Using command line parameters in JMeter for load testing.
- Database load testing with JMeter
- Load test TCP protocol services with JMeter
- Run a stress test in JMeter
- Performance and load testing with JMeter
- Test SOAP services with JMeter
- Generate Random variables in JMeter
- Apache Kafka load testing using JMeter
- Build a distributed load testing infrastructure with AWS, Docker and JMeter
- Load testing video streaming with JMeter

Installation

- JMeter Environment Setup – Java JDK 1.6 or above required
- Set JAVA_HOME
- Download JMeter - http://jmeter.apache.org/download_jmeter.cgi
- Set JMETER_HOME (export PATH=$PATH:$JMETER_HOME/bin)
- Run JMeter via command line or GUI

Thread Group

Thread Group elements are the beginning points of your test plan.

- Set the number of threads
- Set the ramp-up period
- Set the number of times to execute the test

Controller

JMeter has two types of Controllers:

- Samplers

Samplers allow JMeter to send specific types of requests to a server. They simulate a user request for a page from the target server.

- Logic Controllers

Logic Controllers control the order of processing of Samplers in a Thread. Logic controllers can change the order of a request coming from any of their child elements.

Execution Order

- Configuration elements
- Pre-Processors
- Timers
- Sampler
- Post-Processors
- Assertions
- Listeners

Demo - Http Request Demo with Recording

- Configuration of JMeter
- Create Sample project for selected website to check record (HTTP test script Recorder)
- User Defined Variable.
- Report Generation - Summary report generation
- Summary Report - report visualization on the Dashboard

Thank You !

Software Quality Assurance HNDIT-4022

White Box Testing

White Box Testing

White box testing techniques analyze the internal structures the used data structures, internal design, code structure, and the working of the software rather than just the functionality as in black box testing. It is also called glass box testing or clear box testing or structural testing. White Box Testing is also known as transparent testing or open box testing.

White Box Testing

White box testing is a verification technique where software engineers can use to examine if the code works as expected or not.

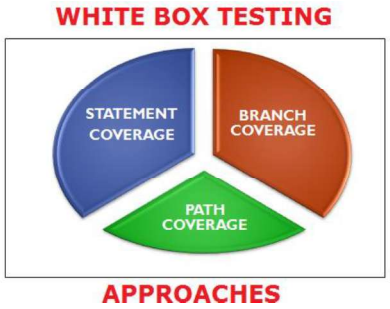
It is the strategy testing is based on

- Software's internal paths
- Design and coding
- Logics
- Structures
- Implementation of software under test

Process of White Box Testing

1. Input: Requirements, Functional specifications, design documents, source code.
2. Processing: Performing risk analysis to guide through the entire process.
3. Proper test planning: Designing test cases to cover the entire code. Execute rinse-repeat until error-free software is reached. Also, the results are communicated.
4. Output: Preparing final report of the entire testing process.

Techniques in White Box Testing

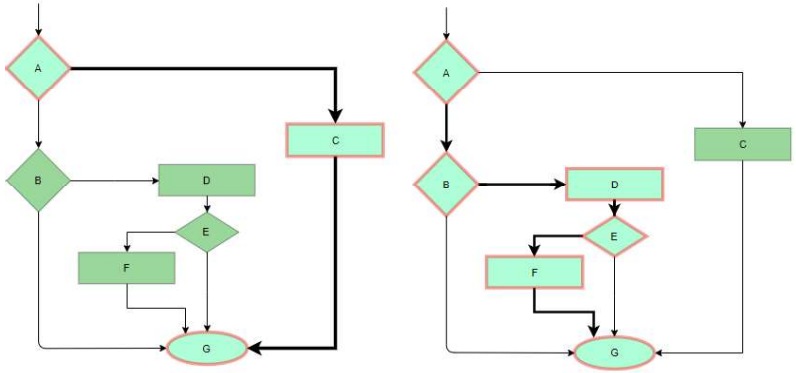


Testing Techniques...

1. Statement Coverage

In this technique, the aim is to traverse all statements at least once. Hence, each line of code is tested. In the case of a flowchart, every node must be traversed at least once. Since all lines of code are covered, it helps in pointing out faulty code.

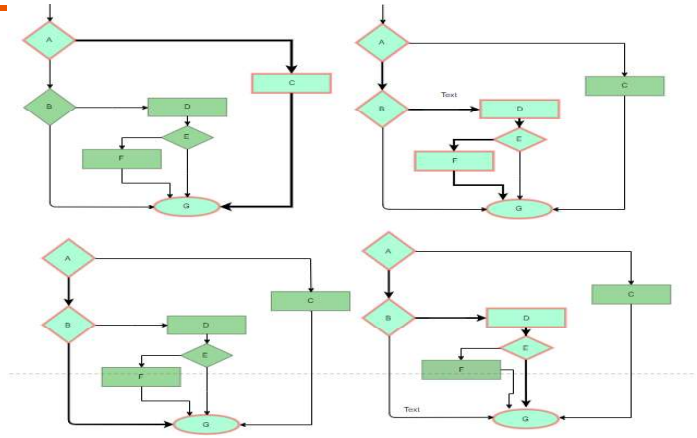
1. Statement Coverage...



2. Branch Coverage:

In this technique, test cases are designed so that each branch from all decision points is traversed at least once. In a flowchart, all edges must be traversed at least once.

2. Branch Coverage…



3. Condition Coverage

In this technique, all individual conditions must be covered as shown in the following example:

- READ X, Y
- IF(X == 0 || Y == 0)
- PRINT '0'
- #TC1 – X = 0, Y = 55
- #TC2 – X = 5, Y = 0

Other White Box Testing Techniques

4. Multiple Condition Coverage
5. Basic Path Testing
6. Loop Testing
    - ❏ Simple loops
    - ❏ Nested loops
    - ❏ Concatenated loops

White Testing is performed in 2 Steps;

1. Tester should understand the code well
2. Tester should write some code for test cases and execute them

Tools required for White box testing:

- PyUnit
- Sqlmap
- Nmap
- Parasoft Jtest
- Nunit
- VeraUnit
- CppUnit
- Bugzilla
- Fiddler
- JSUnit.net
- OpenGrok
- Wireshark
- HP Fortify
- CSUnit

Features of White box Testing:

- Code coverage analysis
- Access to the source code
- Knowledge of programming languages
- Identifying logical errors
- Integration testing
- Unit testing
- Optimization of code
- Security testing
- Verification of Design
- Check for Accurate Code
- Identifying Coding Mistakes
- Path Examination
- Determining the Dead Code

Advantages of Whitebox Testing:

- Thorough Testing:
- Code Optimization:
- Early Detection of Defects:
- Integration with SDLC:
- Detection of Complex Defects:
- Comprehensive Test Cases:
- Testers can ensure that the code meets coding standards and is optimized for performance.

Disadvantages of White box Testing:

- Programming Knowledge and Source Code Access:
- Overemphasis on Internal Workings:
- Bias in Testing:
- Test Case Overhead:
- Dependency on Tester Expertise:
- Inability to Detect Missing Functionalities:
- Increased Production Errors:

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Black Box Testing

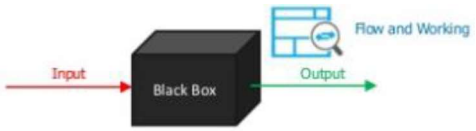
What Is Black Box Testing

Black-box testing is a type of software testing in which the tester is not concerned with the internal knowledge or implementation details of the software but rather focuses on validating the functionality based on the provided specifications or requirements.

Black Box Testing

Black Box Testing : tester tests the application.software by applying different inputs and comparing the output with expected results.

It is also known as Behavioural Testing.



Types Of Black Box Testing

The following are the several categories of black box testing:

- Functional Testing
- Regression Testing
- Nonfunctional Testing (NFT)

What can be identified by Black Box Testing

- Discovers missing functions, incorrect function & interface errors
- Discover the errors faced in accessing the database
- Discovers the errors that occur while initiating & terminating any functions.
- Discovers the errors in performance or behaviour of software.

Techniques in Black Box Testing

- Boundary Value Analysis
- Equivalence class partitioning
- Decision Table based testing
- State Transition
- Error guessing

Advantages of Black Box Testing

- The tester does not need to have more functional knowledge or programming skills to implement the Black Box Testing.
- It is efficient for implementing the tests in the larger system.
- Tests are executed from the user's or client's point of view.
- Test cases are easily reproducible.
- It is used to find the ambiguity and contradictions in the functional specifications.

Disadvantages of Black Box Testing

- There is a possibility of repeating the same tests while implementing the testing process.
- Without clear functional specifications, test cases are difficult to implement.
- It is difficult to execute the test cases because of complex inputs at different stages of testing.
- Sometimes, the reason for the test failure cannot be detected.
- Some programs in the application are not tested.
- It does not reveal the errors in the control structure.
- Working with a large sample space of inputs can be exhaustive and consumes a lot of time.

Features of black box testing

- Independent testing
- Testing from a user's perspective
- No knowledge of internal code
- Requirements-based testing
- Different testing techniques
- Easy to automate
- Scalability
- Limited knowledge of application

Tools Used for Black Box Testing:

- Appium
- Selenium
- Microsoft Coded UI
- Applitools
- HP QTP.

Black Box Vs White Box Testing

Area	Black Box Testing	White Box Testing
Testing objectives	mainly focused on testing the functionality of the software, ensuring that it meets the requirements and specifications.	mainly focused on ensuring that the internal code of the software is correct and efficient.
Knowledge level	does not require any knowledge of the internal workings of the software, and can be performed by testers who are not familiar with programming languages.	White box testing requires knowledge of programming languages, software architecture and design patterns.
Testing methods	uses methods like equivalence partitioning, boundary value analysis, and error guessing to create test cases.	uses methods like control flow testing, data flow testing and statement coverage.
Scope	generally used for testing the software at the functional level.	used for testing the software at the unit level, integration level and system level.

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

System Integration Testing (SIT)

What is System Integration Testing (SIT)?

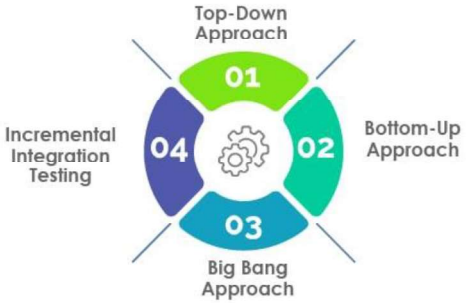
- System Integration Testing, or SIT, is a QA process that is leveraged to ensure the compatibility of two or more systems. It helps to ensure that the systems are working together correctly and that any interactions are appropriate and safe. This process can involve the testing of software, systems, or networks. It can also involve testing of the system's performance, integrity, and compatibility.

Importance of System Integration Testing

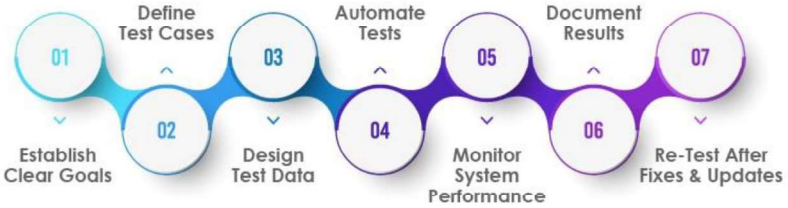
- System integration testing is an essential part of many business operations.
- It allows for the identification and cleaning up of systems, data, and processes.
- The importance of system integration testing is evident as it verifies that all components of a system interact with each other correctly.
- This type of testing ensures that any changes made to one component do not have an adverse impact on the other components or the overall system.
- SIT also helps identify and resolve any compatibility issues between different systems, which are necessary for successful integration.

Benefits of System Integration Testing

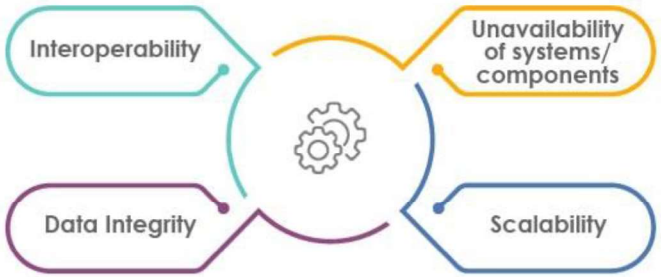
System Integration Testing Techniques



System Integration Testing: Best Practices



Factors to Consider While Integration Testing



Objective of System Integration Testing

- ❏ To meet software with user requirements.
- ❏ To maximize the memory usage.
- ❏ To control data flow in within the system.
- ❏ To test the control flow of the system.
- ❏ To find the errors and bugs in the system.
- ❏ To minimize time consumption for testing process.

Major States of SIT

There are three major states of SIT.

- ❏ Data state within the integration layer
- ❏ Data state within the database layer
- ❏ Data state within the application layer

Q&A's ?

Thank you !

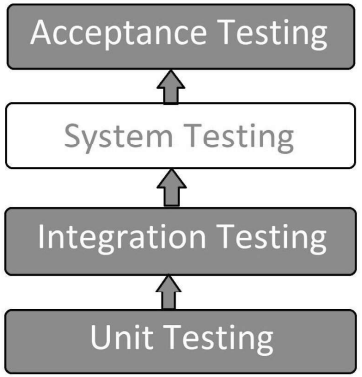
Software Quality Assurance HNDIT-4022

System Testing

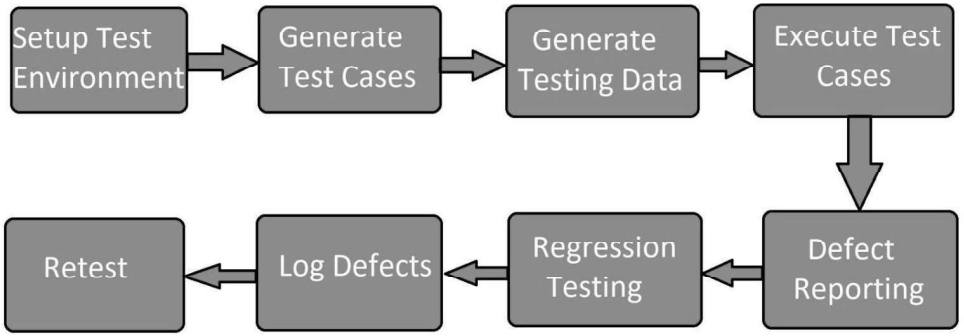
What is System Testing ?

- System testing is a type of software testing that evaluates the overall functionality and performance of a complete and fully integrated software solution. It tests if the system meets the specified requirements and if it is suitable for delivery to the end-users.
- This type of testing is performed after the integration testing and before the acceptance testing.

What is System Testing ?



System Testing Process



Objectives of System Testing

- Objectives of system testing include:

  • Reducing risk

  • Verifying whether the functional and non-functional behaviors of the system are as designed and specified

  • Validating that the system is complete and will work as expected

  • Building confidence in the quality of the system as a whole

  • Finding defects

  • Preventing defects from escaping to higher test levels or production

Types of System Testing

System testing is divided into more than 50 types, but software testing companies typically uses some of them. These are listed below:

- Regression Testing
- Load Testing
- Functional Testing
- Recovery Testing
- Migration Testing
- Usability Testing
- Software and Hardware Testing
- Performance Testing:
- Load Testing
- Stress Testing
- Scalability Testing

Why is System Testing Important?

- System Testing gives hundred percent assurance of system performance as it covers end to end function of the system.
- It includes testing of System software architecture and business requirements.
- It helps in mitigating live issues and bugs even after production.
- System testing uses both existing system and a new system to feed same data in both and then compare the differences in functionalities of added and existing functions so, the user can understand benefits of new added functions of the system.

Tools used for System Testing

- JMeter
- Gallen Framework
- Selenium
- HP Quality Center/ALM
- IBM Rational Quality Manager
- Selenium
- Appium
- LoadRunner
- Apache JServ
- SoapUI

Advantages of System Testing

The testers do not require more knowledge of programming to carry out this testing.

It will test the entire product or software so that we will easily detect the errors or defects which cannot be identified during the unit testing and integration testing.

The testing environment is similar to that of the real time production or business environment.

It checks the entire functionality of the system with different test scripts and also it covers the technical and business requirements of clients.

Disadvantages of System Testing

- ❏ This testing is time consuming process than another testing techniques since it checks the entire product or software.
- ❏ The cost for the testing will be high since it covers the testing of entire software.
- ❏ It needs good debugging tool otherwise the hidden errors will not be found.
- ❏ Can be time-consuming and expensive.
- ❏ Requires adequate resources and infrastructure.
- ❏ Can be complex and challenging, especially for large and complex systems.
- ❏ Dependent on the quality of requirements and design documents.

Disadvantages of System Testing…

- ❏ Limited visibility into the internal workings of the system.
- ❏ Can be impacted by external factors like hardware and network configurations.
- ❏ Requires proper planning, coordination, and execution.
- ❏ Can be impacted by changes made during development.
- ❏ Requires specialized skills and expertise.
- ❏ May require multiple test cycles to achieve desired results.

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Testing & Risk

Risk Definition and Risk Attributes

Risk is a potential event, hazard, threat, or situation whose occurrence causes an adverse effect.

A risk can be characterized by two factors:

- ❏ Risk likelihood
- ❏ Risk impact (harm)

Type of Risks

In software testing one is generally concerned with two types of risks;

- ❏ Project risks
- ❏ Product risks.

Project Risks

Project risks are related to the management and control of the project. Project risks include:

- ❏ Organizational issues (e.g., delays in work products deliveries, inaccurate estimates, cost-cutting)
- ❏ People issues (e.g., insufficient skills, conflicts, communication problems, shortage of staff)
- ❏ Technical issues (e.g., scope creep, poor tool support)
- ❏ Supplier issues (e.g., third-party delivery failure, bankruptcy of the supporting company)

Product Risks

Product risks are related to the product quality characteristics (e.g., described in the ISO 25010 quality model).

Examples of product risks include;

- ❏ Missing or wrong functionality
- ❏ incorrect calculations,
- ❏ Runtime errors
- ❏ Poor architecture
- ❏ Inefficient algorithms
- ❏ Response time
- ❏ Poor user experience
- ❏ Security

Product Risks...

Product risks, when they occur, may result in various negative consequences, like:

- ❏ User dissatisfaction
- ❏ Loss of revenue, trust, reputation
- ❏ Damage to third parties
- ❏ High maintenance costs, overload of the helpdesk
- ❏ Criminal penalties
- ❏ In extreme cases, physical damage, injuries or even death

Product Risk Analysis

From a testing perspective, the goal of product risk analysis is to provide an awareness of product risk in order to focus the testing effort in a way that minimizes the residual level of product risk. Ideally, product risk analysis begins early in the SDLC.

Product risk analysis consists of Risk Identification and Risk Assessment.

Risk identification is about generating a comprehensive list of risks. Stakeholders can identify risks by using various techniques and tools,

E.g:

Brainstorming, workshops, interviews, or cause-effect diagrams.

Product Risk Analysis...

Risk assessment can use a quantitative or qualitative approach, or a mix of them. In the quantitative approach the risk level is calculated as the multiplication of risk likelihood and risk impact. In the qualitative approach the risk level can be determined using a risk matrix.

Risk assessment involves:

- ❏ Categorization of identified risks,
- ❏ determining their risk likelihood,
- ❏ risk impact and level,
- ❏ prioritizing, and proposing ways to handle them.

Categorization helps in assigning mitigation actions, because usually risks falling into the same category can be mitigated using a similar approach.

Product Risk Analysis...

Product risk analysis may influence the thoroughness and scope of testing. Its results are used to:

- ❏ Determine the scope of testing to be carried out
- ❏ Determine the particular test levels and propose test types to be performed
- ❏ Determine the test techniques to be employed and the coverage to be achieved
- ❏ Estimate the test effort required for each task
- ❏ Prioritize testing in an attempt to find the critical defects as early as possible
- ❏ Determine whether any activities in addition to testing could be employed to reduce risk

Product Risk Control

Product risk control consists of **risk mitigation** and **risk monitoring**.

Risk mitigation involves implementing the actions proposed in risk assessment to reduce the risk level.

The aim of risk monitoring is to ensure that the mitigation actions are effective, to obtain further information to improve risk assessment, and to identify emerging risks.

With respect to product risk control, once a risk has been analyzed, several response options to risk are possible,

E.g. Risk mitigation by testing, Risk acceptance, Risk transfer, or contingency plan.

Product Risk Control...

Actions that can be taken to mitigate the product risks by testing are as follows:

- ❏ Select the testers with the right level of experience and skills, suitable for a given risk type
- ❏ Apply an appropriate level of independence of testing
- ❏ Conduct reviews and perform static analysis
- ❏ Apply the appropriate test techniques and coverage levels
- ❏ Apply the appropriate test types addressing the affected quality characteristics
- ❏ Perform dynamic testing, including regression testing

What is Risk Management?

Risk Management is a systematic process of recognizing, evaluating, and handling threats or risks that have an effect on the finances, capital, and overall operations of an organization.

The main goal of risk management is to predict possible risks and find solutions to deal with them successfully.

Why is risk management important?

Risk management is important because it helps organizations to prepare for unexpected circumstances that can vary from small issues to major crises.

By actively understanding, evaluating, and planning for potential risks, organizations can protect their financial health, continued operation, and overall survival.

Risk Management Process

Risk management is a sequence of steps that help a software team to understand, analyze, and manage uncertainty. Risk management process consists of

- ❏ Risks Identification.
- ❏ Risk Assessment.
- ❏ Risks Planning.
- ❏ Risk Monitoring.

Risk Management Standards and Frameworks

Risk management standards and frameworks give organizations guidelines on how to find, evaluate, and handle risks effectively. They provide a structured way to manage risks, making sure that everyone follows consistent and reliable practices.

Well-known risk management standards and frameworks:

- ❏ **COSO ERM Framework:**
- ❏ **ISO 31000**:
- ❏ **BS 31100**:

Benefits of Risk Management

- ❏ Here are some benefits of risk management:
- ❏ Helps protect against potential losses.
- ❏ Improves decision-making by considering risks.
- ❏ Reduces unexpected expenses.
- ❏ Ensures adherence to laws and regulations.
- ❏ Builds resilience against unexpected challenges.
- ❏ Safeguards company reputation.

Limitation of Risk Management

- Too much focus on risk can lead to missed opportunities.
- Implementing risk management can be expensive.
- Risk models can be overly complex and hard to understand.
- Having risk controls might make people feel too safe.
- Relies on accurate human judgment and can be prone to mistakes.
- Some risks are hard to predict or quantify.
- Managing risks can take a lot of time and resources.

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Software Testing & Quality Management

What is Software Testing?

Testing is the primary activity of detecting and solving technical issues in the software source code and assessing the overall product usability, performance, security, and compatibility.

What is Software Quality?

Software quality refers to the degree to which a software product or system satisfies the stated and implied needs of its customers and users. It also relates to how the software product meets the quality standards set by the industry and how it compares with other similar software products.

What is Quality Assurance?

Quality assurance is a proactive process that starts from the very beginning of the software development process. The goal of quality assurance is to prevent defects by focusing on the process used to create the product quality of the software. It includes activities like defining and setting quality standards, process standards, quality planning, and continuous improvement.

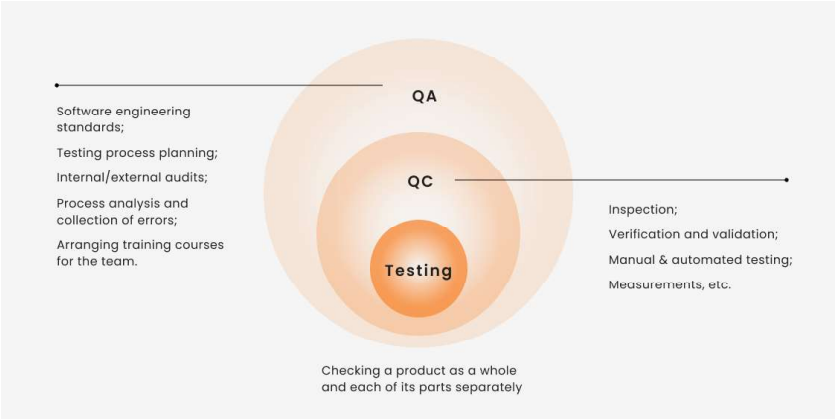
What is Quality Control?

In the software development lifecycle, quality control is the stage where the software product is tested and reviewed.

Quality control helps to ensure that the software product meets the quality standards that were set during quality planning.

It is a critical aspect of software quality management and involves the development team, testing team, and sometimes even software users in a structured and organised set.

Software Quality Assurance vs Quality Control vs Testing



Software Quality Assurance vs Quality Control vs Testing

	QA	QC	Testing
Approach	Proactive	Reactive	Reactive
Focus	Process-oriented	Product-oriented	Focused on finding mistakes and bugs
Techniques	Developing strategies, establishing standards, creating test management plans	Verifying the quality of software using testing, code reviews, continuous	Conducting all types of testing through manual or automated tools
Timeframe	Throughout the development process	Before the product release	Along the development process
Executor	The entire team, including stakeholders	A small, dedicated team	Test engineers, developers

What is Software Quality Management?

Software quality management is a systematic process of ensuring that software meets the highest standards in terms of functionality, reliability, and user satisfaction.

Software Quality Management (SQM) is the process of constant improvement of product quality, integrated into the development process until the expectations are achieved.

Why Software Quality Management is Important?

It helps to identify and correct errors in software, which is essential for the effective implementation of a project.

The quality of the software is simply the measurement of how well the product has been able to fulfill customer needs.

It is measured by how often errors are found and repaired, how well the software works under different conditions, and how easily software can be maintained.

Software quality is determined by comparing actual results against the expected results.Software quality can include a large variety of factors such as degree of completeness, efficiency, usability, correctness, and reliability

What are Software Quality Management Metrics?

Software metrics can be classified into three categories

- Product metrics - Mean Time to Failure/ Defect Density /Customer Problems /Customer Satisfaction
- Process metrics - Defect density during machine testing/ Defect arrival pattern during machine testing/ Phase-based defect removal pattern/ Defect removal effectiveness
- Project metrics - Fix backlog and backlog management index/ Fix response time and fix responsiveness/ Percent delinquent fixes/ Fix quality

Benefits of Software Quality Management

- Improves product quality.
- Adds to customer satisfaction.
- The software meets user needs and expectations and generates more value, leading to increased user adoption and potentially higher revenue.
- Helps bring down the cost of rework as defects are detected early in SDLC.
- Assists in making credible and reliable product development decisions and test management strategies.
- Increases productivity.

Q&A's ?

Thank you !

Software Quality Assurance HNDIT-4022

Software Testing Environment

What is a Test Environment?

The test environment is the hardware and software set up for the testing teams to run test cases.

This test environment setup varies from product to product and its configuration completely depends on the application under test requirement.

Types of Test Environment

- Integration Test Environment:
- Performance Test Environment:
- Security Test Environment:
- Chaos Test Environment:

Key Areas to Set up Test Environment

- ❑ Database:
- ❑ Operating System
- ❑ Network protocol
- ❑ Test Data
- ❑ Manual testers
- ❑ Automation testers
- ❑ Documentation

Process for Setup of Test Environment

System admins, developers, and testers are some of the people that are involved in the testing of the application. The test environment involves setting up different areas like:

- ❑ Test Server
- ❑ Network
- ❑ PC setup
- ❑ Bug Reporting
- ❑ Test Tool
- ❑ Test Data

Overview of roles in testing

Typically, career in software testing field consists of 7 major roles, based on knowledge and experience :

- Junior Software Tester / QA Engineer.
- Senior Software Tester / Senior QA Engineer.
- Test Architect.
- QA Lead / Test Lead.
- QA Manager / Test Manager.
- Quality Head.
- Delivery Head.

Overview of roles in testing

- ❑ **QA Engineer**
  The Test Engineer role is the principal role in testing and is responsible for the core activities of the test effort that involves designing, implementing, executing and evaluating the necessary tests and logging the outcome of that test.

- ❑ **Test Lead**
  The Test Lead role is primarily responsible for test planning, identifying and defining the required tests, monitoring detailed testing progress, and evaluating the Test Results.

Overview of roles in testing

QA teams help meet clients' demands and expectations most fully, and this is the reason there are a lot of companies providing QA testing services nowadays.
The basic QA levels all companies would have are :

A. QA Tester:
QA testers are of two types :

- Manual QA Engineer :
- Manual QA specialists find bugs in a system by relying on human observation. They make sure that the tests behave exactly as the expectations. Testers are provided with a test plan according to which they write test cases describing an input action and an expected response. QA engineers compare the actual result with the expected result and log a defect if they don't match. The found defects are then reported via any bug tracking and management tool, e.g., Jira.

Overview of roles in testing

Automation QA Engineer
Automation QA engineers have more advanced coding skills than manual QA engineers. With the help of their programming skills, they write scripts to run repetitive tests. They are often called developers in testing. Different automation tools are used with the help of which these QA engineers design and deploy test automation scripts. The ultimate goal is to automate as many of the test cases as possible so that the manual testers do not have to put the effort into running the same test cases on daily basis.

Overview of roles in testing

B. QA Lead
QA lead is someone who supervises the work of manual and automation QA engineers. The day-to-day responsibilities of a QA lead are :

Supervising team members,
Administering outside resources, and
Designing test programs.
Prepare Test strategy
Controlling and planning the testing process.
Representing QA team at meetings across departments

Overview of roles in testing

C. QA Manager
Quality assurance managers work with other departments as well to establish quality standards. Managers also monitor that the standards are followed against agreed targets.
Some responsibilities of QA managers are:

- Establish Quality Assurance Procedures
- Ensuring that the customer requirements are met.
- Analyze Production and Quality Control Reports
- Investigating and setting standards for quality, health, and safety
- Ensuring that processes comply with standards at both national and international level
- Writing management and technical reports.
- Determining training needs
- Recording and analyzing statistical information
- Monitoring engineer's performance.

Q&A's ?

Thank you !