• Introduction to Software Testing :-

14.05.25

Manual Testing modules —

1. Software Testing Concepts (What ?)
2. Software Testing Life cycle (STLC) (HOW ?)
3. Software Testing project & Templates
4. Agile Testing & Jira Tool
5. Usage of AI tools in Manual Testing

Module 1 Software Testing Concepts :-

A Software is a collection of computer programs that helps us to perform a task.

Types of Software :-

i) System Software , ex :- Device Drives, Operating systems etc

ii) Programming Software , ex :- Compilers, Debuggers, Interpreters.

iii) Application Software, ex :- Web Applications, Mobile apps.

Most of the time testing is done on Application Softwares.

What is Software Testing ?

→ Software Testing is a part of Software Development process. It is done to detect and identify the defects in the Software , the Objective of testing is to release quality product to the client.

Quality Software —

Bug free (100% bug free is not possible)

Delivered on time

Within budget

Meets client's requirements

Maintainable / user friendly

Why Software Testing —

- To ensure Software is Bug free.
- Ensure that Software meets customer requirements.
- Ensure that System meets end user expectations.
- Fixing the bugs identified after release is more expensive.

Project vs product

Project – Application or Software developed for specific customer requirements.

Product – Application or Software developed for multiple customers based on market requirements.

Error, Bug & Failure

Error – Human mistake that produces incorrect result. It can occur at Developer level.

Bug / Defect – Deviation from the expected to the actual result. Something which is not working according to customer's requirement in a Software. It occurs at Tester level.

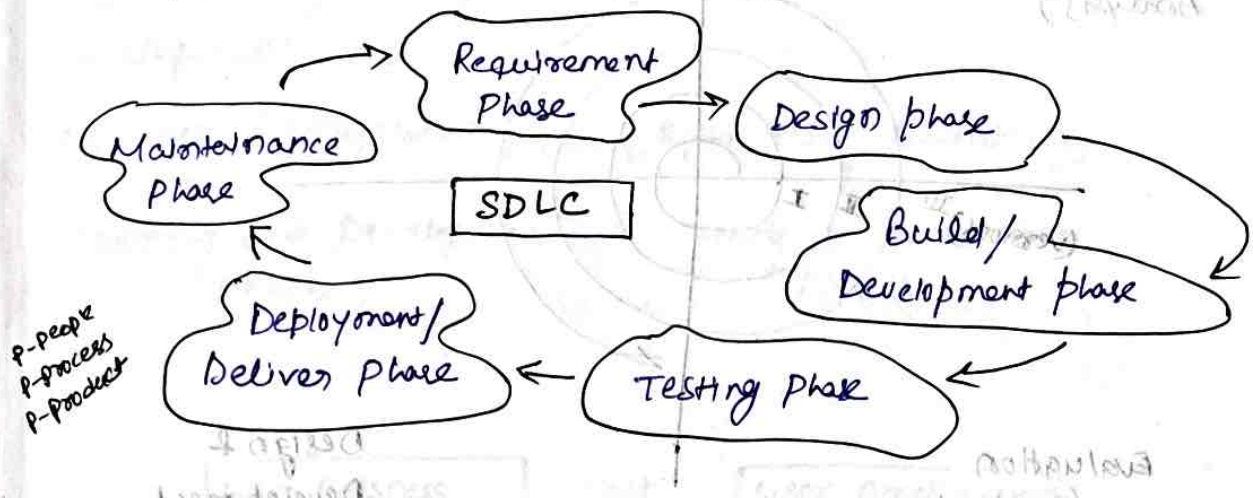
Failure – It occurs at customer level. The deviation or changes identified by the end users. It occurs after releasing the Software to the end users.

Why Software has Bugs?

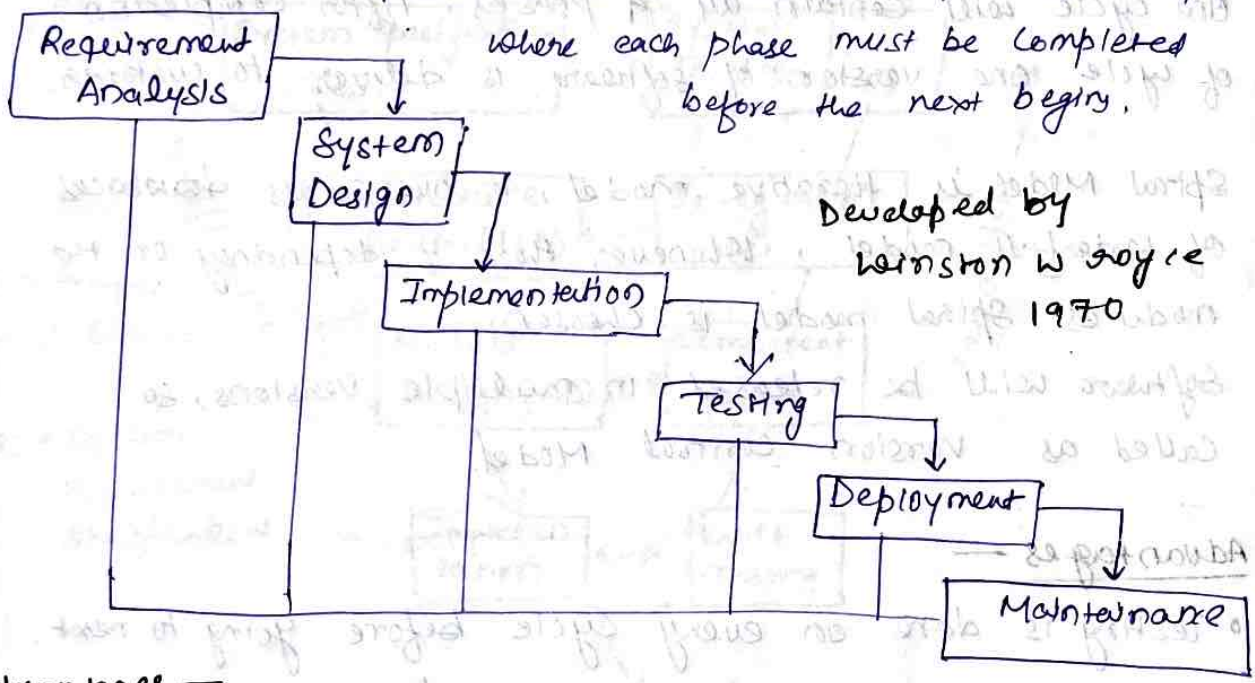
- Miscommunication b/w developers and Testers.
- Software Complexity
- Programming Errors
- Changing Requirements
- Lack of skilled Testers

SDLC (Software Development Life Cycle) Process –

It is the process used by Software Industry to Design, Develop and Test the Software before deploying.



Waterfall Model – Every phase is dependent on the other phase. It is a linear or sequential model where each phase must be completed before the next begin.



Developed by Winston W Royce 1970

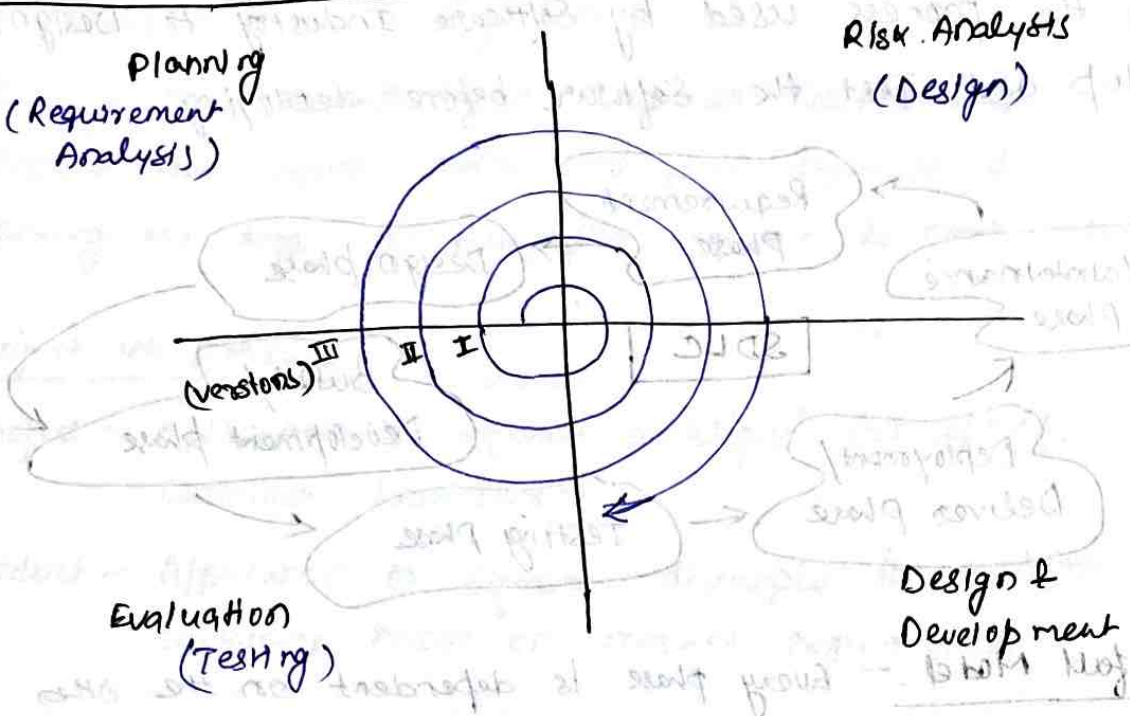
Advantages –

- Quality of the product will be good.
- Since Requirement changes are not allowed, Chances of finding bugs will be less.
- Preffered for small projects where requirements are freezed.

Disadvantages –

- Requirement changes not allowed.
- Defect in Requirement continues in later phases.
- Testing will start only after coding.

Spiral Model / Iterator Model



One cycle will contain all 4 phases. After completion of cycle one version of software is deliver to customer.

Spiral Model is iterative model, it overcomes drawbacks of waterfall model. Whenever there is dependency on the modules Spiral model is choosen.
Software will be released in multiple versions, so called as version control Model.

Advantages —

- Testing is done on every cycle before going to next.
- Customer will get to use software for every module.
- Requirement changes are allowed after every cycle before going to next.

Disadvantages

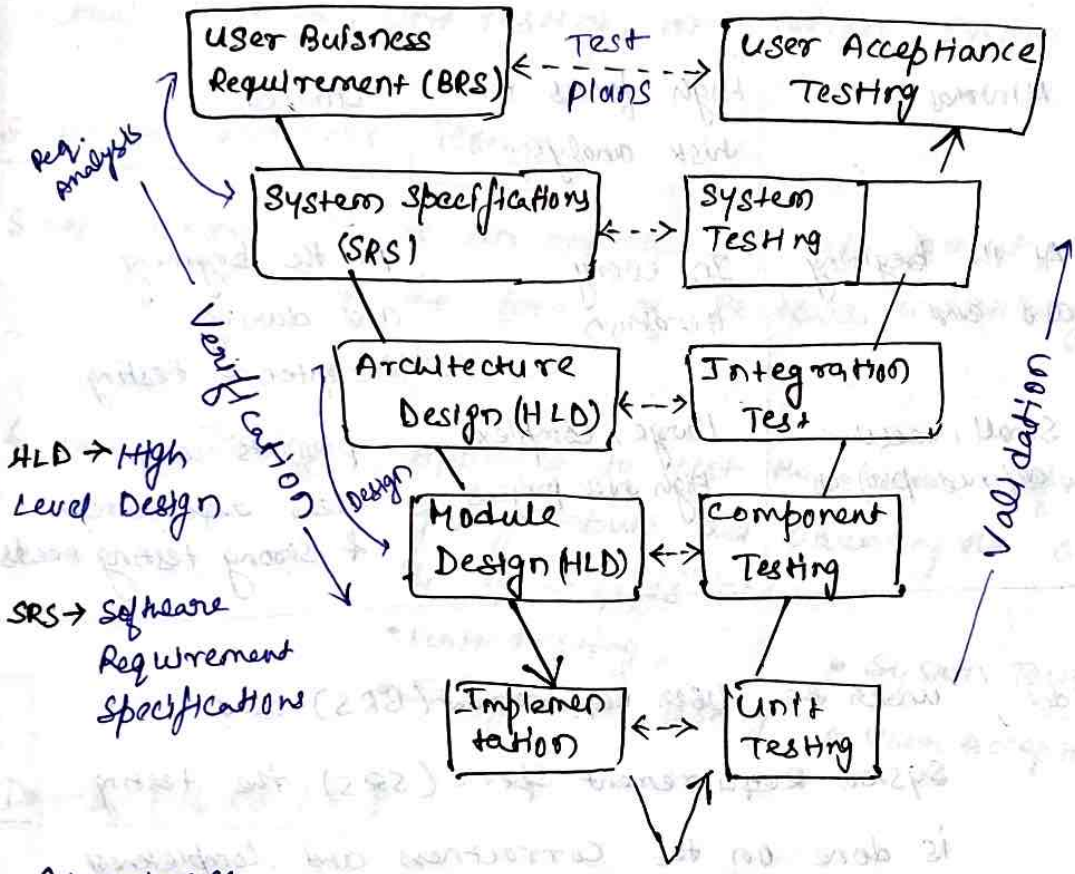
- Requirement changes not allowed in between.
- Every cycle of Spiral model looks like waterfall model.
- There is no testing in requirement and design phase.

V-Model of SDLC :- Known as verification & validation Model.

It extends the features of traditional waterfall model by allowing testing & validation in each phase of development stage.

Left Side (verification) → Planning and Development phases

Right Side (validation) → Testing & validation phases



Advantages

- Testing is involved in each & every phase.
- Defects are found early.
- Emphasizes early test planning.

Disadvantages

- Not suitable for complex or iterative projects.
- Documentation & Initial investment is more
- Not flexible to changes.

Difference Between Waterfall, Spiral and V-model

Feature	Waterfall Model	Spiral Model	V-Model (verification & validation)
Approach	Linear and Sequential	Iterative and risk driven	Sequential with corresponding testing phase
Flexibility	Rigid, difficult to go back	Flexible, allows changes in each iteration.	Rigid, but testing is integrated early
Risk Management	Minimal	High focus on risk analysis	Limited
customer Involvement	At the begining and end	In every Iteration	At the begining and during acceptance testing
Best for	Small, well defined projects	Large, complex, High risk projects	Projects with Clear requirement & strong testing need

In V-Model, under the User Requirement (BRS) and System Requirement Spec (SRS) the testing is done on the correctness and completeness of the documents, called as static testing. All the testing are done on the documents as softwares are not ready, during Requirement, Design phase.

- The Correctness and Completeness of the documents are done under —
  - Review, walkthrough, Inspection and these are static testing techniques.

Verification vs Validation VVI

Verification checks whether we are building the right product. It focuses on documentation. It involves

- Reviews
- Walkthroughs
- Inspections

* verification comes under Static Testing
* Validation is under Dynamic Testing

Validation checks whether we are building the product right or not. It takes place after verification, it involves actual testing, unit testing, integration, system testing.

Static vs Dynamic Testing

Static Testing :- It is an approach to test project documents in the form of Reviews, walkthroughs and Inspections.

Dynamic Testing :- Approach to test the actual software by giving inputs and observing the output.
The techniques are :-

- unit Testing,
- Integration Testing
- System Testing
- User Acceptance Testing

Day-3 (16.05.25)

QA vs QC (QA focus on Process, QC focus on Product)

Quality Assurance (QA) :- Happens before product is made, it make sure the process used to make the product is good so that the final product is good to.
e.g> Making a checklist of how software to be tested

Quality Control (QC) :- It happens after product is made it checks the final product to find and fix any defect or mistake. e.g> Testing software to find bugs.

SDET → Software Development Engineering In Testing

Testing Methods

1. White Box Testing
2. Black Box Testing
3. Gray Box Testing

"White Box Testing conducts on internal logic of the programs. Testers can look inside the system and test its internal structure, logic or code"
eg → Testing a car engine, open hood → look engine parts → test each part

To do the White Box testing we should have the programming knowledge.

White Box Testing → Unit testing, Integration Testing

"Black Box Testing is conducted on the functionality of the software or application not the code and check wether it is working according to customer needs or not.

Ex → System Testing & User Acceptance Testing

Testing functionality of a Calculator is Black Box Test..

"Grey Box Testing is the combination of both White Box as well as Black Box Testing.

Ex → Database Testing i.e having knowledge of the internal structure (like database schema, tables etc) and testing from outside (like entering data through UI and check wether correct)

Black Box	+	White Box	=	Grey Box
(unknown Internal code Structure)		(known Internal code Structure)		(Internal code Structure partially known)

Levels of Testing in V-Model

i) Unit Testing ii) Integration Testing iii) System Testing iv) UAT (User Acceptance Testing)

ii) → White Box Testing

iii), iv) → Black Box testing

Unit Testing or Module Testing or Component Testing is testing of individual component of same application.

for ex:- In a Banking service, testing the features like Add balance, Withdraw etc are Unit Testing.

ii) **Integration Testing** :- Testing different modules after combining and checking the results after combining. or Integrating. (B/w two or more modules)

iii) **System Testing** :- Testing the entire system or functionality after software is made.

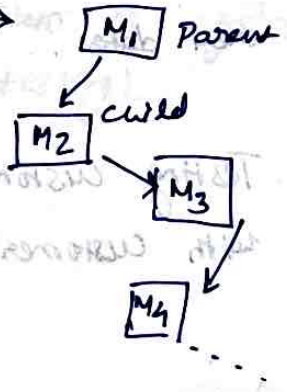
iv) **User Acceptance Testing** :- It is the testing done by end users to test the final System meets the user requirement.

Integration Testing two types —

i) Incremental Integration Testing :- Incrementally adding the modules and testing the data b/w the modules.

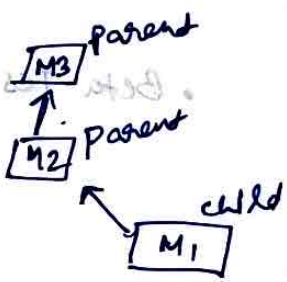
There are three approaches —

- Top Down
- Bottom up
- Hybrid



Hybrid = Top Down + Bottom Up

Bottom up



ii) Non - Incremental Integration :- Not to add modules incrementally, all modules will be added together and test wether system works or not.

NOTE → Difficult to find issue, as all the modules will be together, hence we prefer Incremental Integration Testing.

System Testing (Details one)

Testing over all functionality of system with respect to client requirements.
It's a black box technique, and conducted by Testing team.

System testing focuses on –

- User Interface Testing (GUI)
- Functional Testing
- Non Functional Testing
- Usability Testing

UAT Testing (User Acceptance Testing)

After completion of system testing UAT team conducts acceptance testing in two levels.

- Alpha Testing – Testing application in the development environment, in own company with real data.
- Beta Testing – Testing customer's application with customer data.

System Testing Types

i) GUI Testing → Testing the UI of the application

ii) Usability Testing

iii) Functional Testing

iv) Non-Functional Testing

GUI Testing :- Makes sure all buttons, icons and screens in the software look and work as they should.

e.g → Checking the layout is correct or not by clicking buttons.

Usability Testing :- Tests for if the software is easy for people to use and understand.

e.g → Checking if the menus are clear and tasks are straightforward

Functional Testing :- Verifies that each part of the software performs its jobs correctly.

e.g → Testing if login, search and other functions work as intended.

Non-Functional Testing :- Looks at aspects beyond specific functions like performance, security and how easy it is to recover from errors.

e.g → Checking how many users the system can handle at once (performance testing), or testing how secure the system is against unauthorised access (security testing).

Functional Testing

It is done to ensure whether the software is working according to the customes needs or not.

Types -

i) Object properties Testing :-

Checks the properties of object present on the application.

Ex- Enable, disable, visible, focus...

ii) Database Testing :- (Grey Box Testing)

It ensures that the application interacts correctly with it's database. It checks if data is stored, retrieved and manpulated accurately within the database.

ex → Confirming that users details entered through a registration form are correctly saved and can be retrieved when logging in.

iii) Error Handling Testing :-

Testor verify the error message while performing incorrect actions on the application, Error messages should be clear and easy to understand.

e.g → Testing the system's response when a user tries to submit a form with a missing information, ensuring it displays a helpful Error message.

iv) Calculations & Manipulation Testing

It ensures that calculations give correct results, and the calculation is working or not.

e.g → Calculating total price of card in e-commerce platform.

v) Links Testing

Internal Link → Target content will navigate to same webpage
External Link → Will navigate to different webpage
Broken Link → 404 Not found, Forbidden
Target pages are not present in server

It verifies that the hyperlinks attached within the application work as expected, and user is directed to the correct page.

e.g → Clicking on menu links of website and accessing different menus

vi) Cookies & Session Testing :-

It evaluates how well the application manages user-specific information. It checks if session related functionalities such as login persistance works as usual and expected.

e.g → Logging into an account, closing the browser and reopening it, and confirming that the user remains logged in due to correct handling of cookies or session data.

Cookies are the temporary files created by the Browsers.

Sessions are the time slots created by the server. Sessions will expire after some time if you are idle for some time.

Non-Functional Testing

1. Performance Testing
   - Load Testing
   - Stress Testing
   - Endurance Testing
   - Spike Testing
   - Volume Testing
2. Security Testing
3. Recovery Testing
4. Compatibility Testing
5. Configuration Testing
6. Installation Testing
7. Sanitation / Garbage Testing

"Non functional Testing focuses on the non-functional aspects of a system i.e how well the system performs under certain conditions rather than specific behaviours or functions". It assures that software meets criteria such as performance, usability, reliability, security.

Performance Testing :- Measures the system responsiveness and stability under load.

Load Testing :- It evaluates the performance of the system under the expected workload. Load test includes determining the response time, throughput, error rate etc.

eg → For a newly developed application with an expected load of around 1000 concurrent users. We create load test script with 1000 virtual users and determine how the application behaves.
Tools are → Load Runner, Jmeter

Stress Testing :- Test the performance of the system with unexpected or beyond the expected load. It is done to determine the break point of the application.

e.g → For an application with expected load of 1000 users we run the stress testing with 1200 users and check whether the application is robust to not crash.

Endurance Testing / Soak Testing

Endurance Testing is also known as Soak Testing. It is done to check for how much time duration the system can handle the expected load.

It checks whether the system can sustain the expected load for a long duration.

Focus on stability of the application for long time with expected load.

e.g → An application is used for a long duration continuously by different users, the testing is done for 24 hrs to 2 days to monitor the memory utilization and stability.

Spike Testing :- It analysis the behaviour of the system after sudden and rapid increase of the number of users. It checks whether the application is able to recover after the sudden burst of users beyond expected load.

e.g → In e-commerce website, a sale is in limited duration, In that time the no. of customers can increase suddenly, hence Spike testing is done to analyse the scenario.

Volume Testing :- Testing the application how much volumes of data is able to handle.

The application is tested by feeding or inserting large amount of data in the database or by providing a large file to the application.

e.g → In a newly developed e-commerce website we can perform volume testing by inserting millions of rows in the database and carry out or check performance.

2. Security Testing :- It checks if the Software is secure and protect sensitive information. It finds and fix the vulnerabilities that could be exploited by hackers.

It focuses on — User Authentication, Authorization, access control, Data Encryption, Network security, Client side application security and Server side application security.

Ex:- Testing an online banking application to make sure user account information is safe and protected from unauthorized access.

3. Recovery Testing :- It checks how well a system can recover after failure or crash, and checks whether it works or resume operation without loosing data.

Involves — power supply failure,
External Server unreachable,
Wireless network signal loss
Database Server down
API's response failed etc.

4. Compatibility Testing :- It ensures that Software works well on different devices, browsers and operating systems, and on different environment.

e.g → Testing a mobile app on different Smartphone and ensuring it functions correctly on different screen sizes.

5. Configuration Testing :- It checks if the Software works correctly with different configuration or settings, and adapts well accordingly.

e.g → Testing video game on different computers with various hardware configurations to ensure smooth gameplay.

6. Installation Testing :- It is the testing associated with the installing and uninstalling of softwares and ensures whether it is smooth or not without causing issues.

e.x → Installing a new application and checking that it is not conflicting with existing installations.

7. Sanitation / Garbage Testing :- It tests for the unnecessary and leftover data in the system and ensures software cleans up itself and doesn't leave unused or garbage data behind.

e.g → Testing a messaging app that deleted messages are permanently removed and not present in the background.

Functional Testing vs Non-Functional Testing

Functional Testing	Non-Functional Testing
It ensures that the software functions as expected.	It evaluates how well the software performs under certain conditions.
It checks specific features, functions, actions and behaviour.	It checks performance, security, durability, endurance.
It checks WHAT the software does, by ensuring that individual features works as intended.	It checks HOW WELL the software performs by checking aspects like speed security and overall user experience.

Testing Terminology

Regression Testing :- Regression Testing is the process of testing a software application to ensure that new changes (like updates, bug fixes or new features) do not negatively impact the existing functionality of the software.

e.g> Imagine a mobile app that allows users to login, view their profile, send messages. If the Developer adds new feature, like uploading profile picture, regressing testing ensures that after this change, users can still log in, view their profiles and send messages.

Unit Regression Testing :- Only testing the specific changes or updates made by the developer.

Regional Regression Testing :- Testing is done on the specific changes along with the connected parts. Meeting is conducted b/w developer and testing team to figure out which parts will be affected by the changes.

Full Regression Testing :- Testing is done on the main part that was changed and also the rest of the software.

e.g> If developer made changes in many areas instead of checking each one seperately we test everything together in one full round.

Re-Testing:- Retesting is the testing of a bug found fixed by developers again and again.

Testers close the bug if it is fixed otherwise again send the bug to the developers.

ex:- Build 1.0 was ~~Teste~~ released. Test team found some defects (Defect ID 1.0.1, 1.0.2) and posted.

Build 1.1 was released, now testing the defects 1.0.1 and 1.0.2 in this build is retesting.

It is done to ensure that the defects which were found and posted in the earlier build were fixed or not in the current build.

SMOKE Testing:- Also called BVT (Build Verification Test) Focus on the stability of the build and we check whether it is ready for further testing or not.

The term 'smoke' comes from the idea that if there's a major issue, it would generate enough smoke to stop further testing.

During smoke test we check build installation, appearance of basic screens and navigations.

ex → Application is able to install or not
once installed, all the screens are
working properly or not.

Smoke testing check whether an application is ready for further testing or not as it checks the stability.

It is conducted in the initial cycle, after the build is stable then further testing process is carried out.

Sanity Testing :- After Smoke Testing Sanity testing is done, it is also called Basic Functional Testing.

During Sanity testing we check basic functanilities like login, users registration and logout etc.

ex→ Sign up option is available on login page
clicking Sign up redirects to proper Signup form
Sign up submission is successful without crash
User Signed up is able to login.

Smoke testing is done on the initial level of builds, after the build is stable then we do Sanity testing.

Exploratory testing vs Adhoc Testing vs Monkey Testing

Exploratory Testing :- It is done by understanding and exploring the application completely, identify all possible scenarios, document it then use it for testing. It is done when application is ready but there is no requirement.

Adhoc Testing :- Testing application randomly without any test cases or any business requirement document It is an Informal testing type with an aim to break the system. Tester should have prior or previous knowledge of application even though he doesn't have requirements/test Cases.
No Documentation, No Test Design, No Test Case.

Monkey Testing :-
Testing application randomly without having any test Cases or any business requirement in an intention to break the system. Typically used for gaming applications.

Positive Testing :- Testing application by providing valid inputs It is done to confirm the software works correctly under normal or expected conditions.

ex- Login test (entering valid details for login)

Adding two numbers in Calculator

Negative Testing :- Testing application whether it is failing or not after providing invalid data.

e.g > uploading other extension files than requirement.

End to End Testing :- Testing entire application from start to finish.

It involves testing the various components and systems to ensures they work seamlessly and correctly together.

e.g : Making purchase on e-commerce platform.

Testing is done on each and every process the user goes through, from login, to adding items to cart to processing to checkout and then payment and then payment confirmation.

Globalization Testing :- It checks an application can function seamlessly across different regions, cultures, accomodate various languages, date formats, currencies.

Localization Testing :- It ensures an application is good fit for specific culture & linguistic need of a particular region. It checks whether application can adapt specific local or target audience by verifying language, cultural preference and regional requirements.

STLC - Software Testing Life cycle

STLC is a systematic process followed during Software testing to ensure the quality and effectiveness of the Software product.

The different phases of STLC are:-

1. Requirement Analysis :- The goal is to understand what needs to be tested.
2. Test planning :- It defines the strategy and scope of testing. By defining test objectives and identifying tools and training needs.
3. Test case Development :- Creates Test case and test data, detailed test cases are written and reviewed.
4. Test Environment Setup :- Prepare the environment where testing will be executed
5. Test Execution :- To execute test cases and log defects
6. Test cycle Closure :- Evaluating the testing process & outcomes.

Test plan Content

- → Overview
- → Scope of testing
- → Features to be tested
- → Features not to be tested
- → Test Environment
- → Test Strategy
- → Defect Reporting procedure
- → Roles & Responsibility
- → Test Schedule
- → Test Deliverables
- → Entry and Exit Criteria
- → Suspension and Resumption Criteria
- → Tools
- → Risks and Mitigations
- → Approvals

Use Case, Test Scenario & Test Case

Use Case :- It describes the requirement, contains three key elements :-

Actor :- Represents the users, either an Individual or a group interacting with a process.

Action :- Specifies the steps or activities taken to achieve the desired outcome.

Goal :- Defines the successful result of the user's interaction with the system.

Test Scenario :- Possible area to be tested.

Test Case :- Step by step actions to be performed to validate functionality of how to test. It contains test steps, expected result & actual result.

e.g

Use Case :- Buy a product

Actor :- Customer

Goal :- purchase a specific product on the website

Steps :- Browse the product catalog
Select the product and add to Cart
Proceed to checkout
Enter shipping and payment information
Place the order
Receive confirmation and order tracking info

Test Scenario :- Successful purchase with valid credit card

This scenario covers a happy path where the customer completes the purchase without any issues.

Test Case : TC01 - purchase with valid Credit Card

Pre-conditions :- Customer must have valid account and product is added to the cart.

Steps - Enter valid shipping address
Enter valid Credit Card details
Click "Place Order"

Expected Result :- Order Confirmation message
Credit Card is charged
Order status reflects "processing"

TC02 :- purchase with invalid Credit Card

TC03 :- purchase with insufficient funds

TC04 :- Purchase with missing address

Test Case Contents :-

Test Case ID, Title, Description, Pre-Condition, priority (P0, P1, P2, P3) - Order, Requirement ID, Steps, Action, Expected Result, Actual Result

P0 → Smoke Sanity | P1 - Regression Test Case | P2 → Functional Test Case | P3 → UI Test Case

Test case have only priority

But Defects have priority and severity

Requirement Traceability Matrix (RTM)

It contains a document that establishes a mapping between the requirements and test cases. It is to ensure that all the test cases are covered or not.

Key components of RTM are -

- Requirement ID → Unique identifier assigned to each requirement, making it easy to track
- Req description → A detailed description of each requirement, outlining what functionality is expected.
- Test case ID's :- A list of test cases associated with each requirement.

ex→

Requirement ID	Requirement Description	Test Case ID's
REQ 001	User should be able to login.	TC001, TC002
REQ 002	System should display product.	TC003, TC004
REQ 003	Users can add items to cart	TC005, TC006

Test Environment :- It is a platform specially built for test case execution on the software. Another name is Test Bed.

Defect Reporting and Tracking

Any mismatch functionality found in an application is called Defect / bug / issue.

During Test execution Test Engineers report mismatches as defects to developers.

Defect Reporting tools are -

Jira, Quality Centre, Bug zilla etc

Defect Category

Severity	Priority
(How much impact on application due to defect)	(The importance of defect and how soon it needed to be fixed)
• Blocker	• High (P1)
• Critical	• Medium (P2)
• Major	• Low P(3)
• Minor	

Tester decides the severity, it decides the seriousness of defect and how much impact on business workflow.

Blocker → This indicates nothing can proceed further.

Ex → Application crashed.

Critical → The main functionality is not working. Customer business workflow is broken

Ex → Fund transfer not working

Major → It calls some undesirable behaviour, but the feature is still functional.

Ex → After mail no confirm message

Minor ⇒ It won't cause any major break down of the system.

ex ⇒ spellings, allignments.

Defect priority :- priority means importance of the defect how soon it should be fixed.
Testers decides the priority and Developers or product managers can change the priority later if needed.
Defects will be fixed by developers on priority.

P1(High) :- Must be resolved immediately, as system cannot be used until fixed.

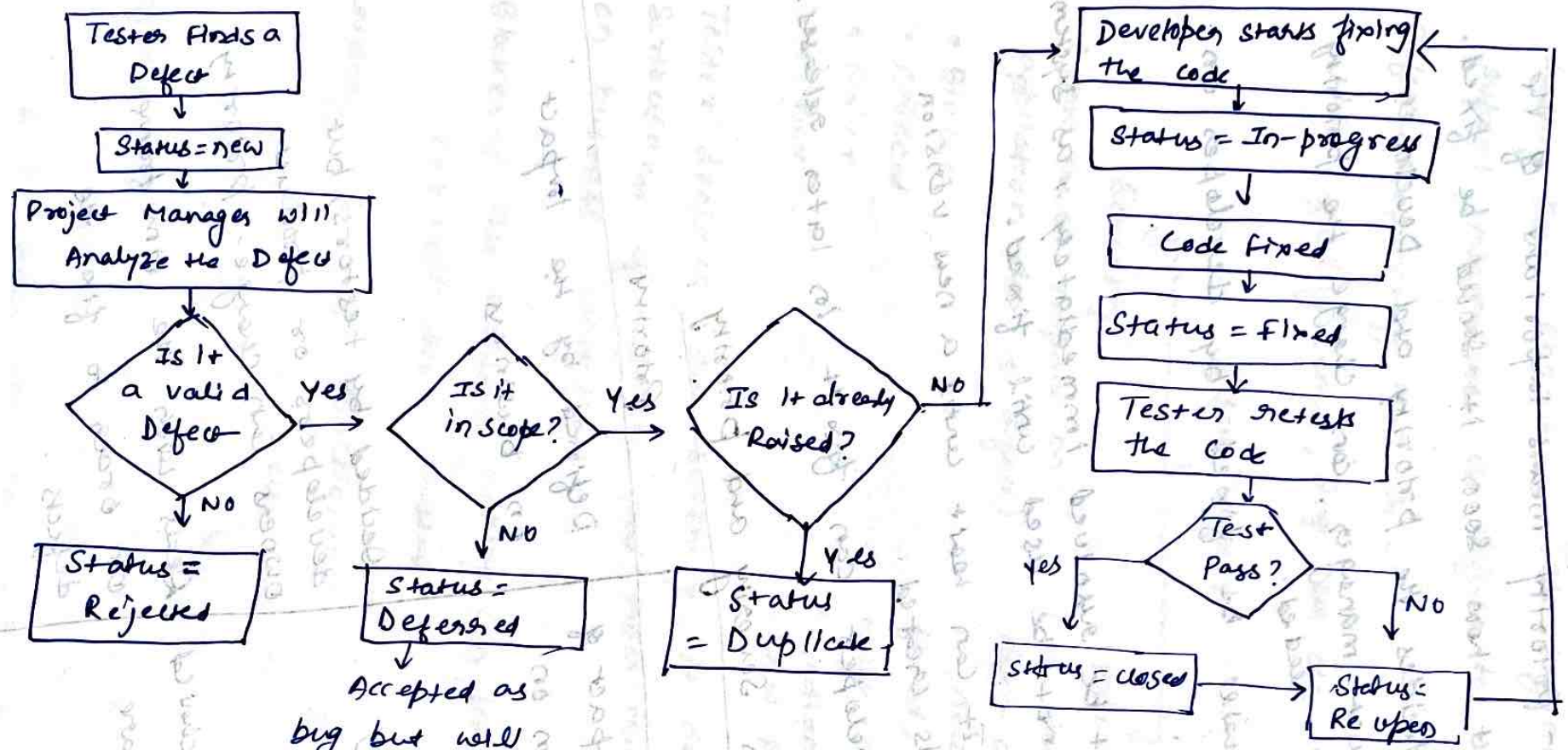
P2(Medium) :- It can wait until a new version is created.

P3(Low) :- Developer can fix it in later releases.

Difference between Severity and priority

Severity	Priority
Defined by the impact of a specific problem on an application.	Defined by the impact on business
category decides by testers.	Decided by testors, but developers or product owners can change priority.
Deals with the technical aspects of the software application.	Deals with the timeframe or order to fix the defects.

Defect (Bug) life cycle



Defect Resolution :- It is the developer's opinion on the defect which has been reported.

After recieving defect report from testing team, development team conduct a review meeting to fix defects. Then they send resolution type to the testing team.

Resolution types are :-

Accept, reject, Duplicate, Enhancement, Fixed etc

The product manager removes the conflict b/w developer and testers.

Defect Report Contents

Defect ID - Unique identification number for the defect

Defect Description - Detail of the defect and where it is found.

Version - version of the application where it is found

Steps

Date Raised → Date when defect was raised

Reference → Reference to documents like requirement, design, architecture or screenshot of error.

Detected by → Name/ID of testers who raised the defect

Status → Status of the defect

Fixed by → Name/ID of the developer who fixed it

Date closed

Severity → Impact of the defect on the application

Priority → Defect fixing urgency.

Test Metrics

→ % of Test cases Executed = $\frac{\text{No. of test cases Executed}}{\text{Total No. of test cases}} \times 100$

→ % of Test Cases Not Executed = $\frac{\text{No. of test cases not Executed}}{\text{Total no. of test cases}} \times 100$

→ % Test Cases Passed = $\frac{\text{No. of test cases passed}}{\text{Total Test cases executed}} \times 100$

Defect Density = Number of defects identified per requirement

Defect Removal Efficiency (DRE) = $\left(\frac{A}{A+B}\right) \times 100$

A → Fixed Defects

B → Missed Defects

Defect Leakage = $\frac{\text{No. of defects found in UAT}}{\text{No. of defects found in testing}} \times 100$

Defect Rejection Ratio = $\frac{\text{No. of defect Rejected}}{\text{Total no. of defects raised}} \times 100$

Defect Age = Fixed date Reported date

Customer Satisfaction = No. of complains per period of time

Manual Testing Project

- Project Introduction
- understanding and Explore the functionality (FRS)
- Test plan
- Writing Test scenarios
- Writing Test cases & reviews
- Environment setup & Build deployment
- Test Execution
- Bug reporting & tracking
- Saniy Testing, Re-Testing & Regression Testing
- Test sign off

agile Methodology

User story → product owner will prepare
Sprint → short span of time → 7, 15, 21 days
Sprint backlog
Sprint retrospection →

Scrum ~~Methodology~~ meeting → standup Meeting

Scrum Master
Product Manager
team

Selenium

Introduction to Web Drivers & Setting up Web Drivers

Challenges in manual Testing — Time Consuming, Tedious, mistakes

How does automation tools work?

Manual Test Case → Manual Execution → AUT

Manual to Automation Conversion → Automation Script → Execute Automation Tests

Selenium :- Selenium is a free (open source) automated testing suite for web applications.

Selenium is not just a single tool but a suite of softwares.

It has 4 components —

- Selenium Integrated Development Environment (IDE)
- Selenium Remote Control
- Web Drivers
- Selenium Grid

Advantages of Selenium → Open source tool, It supports variety of languages that include Java, Perl, Python, C#, Ruby, JavaScript etc.

It supports many operating systems

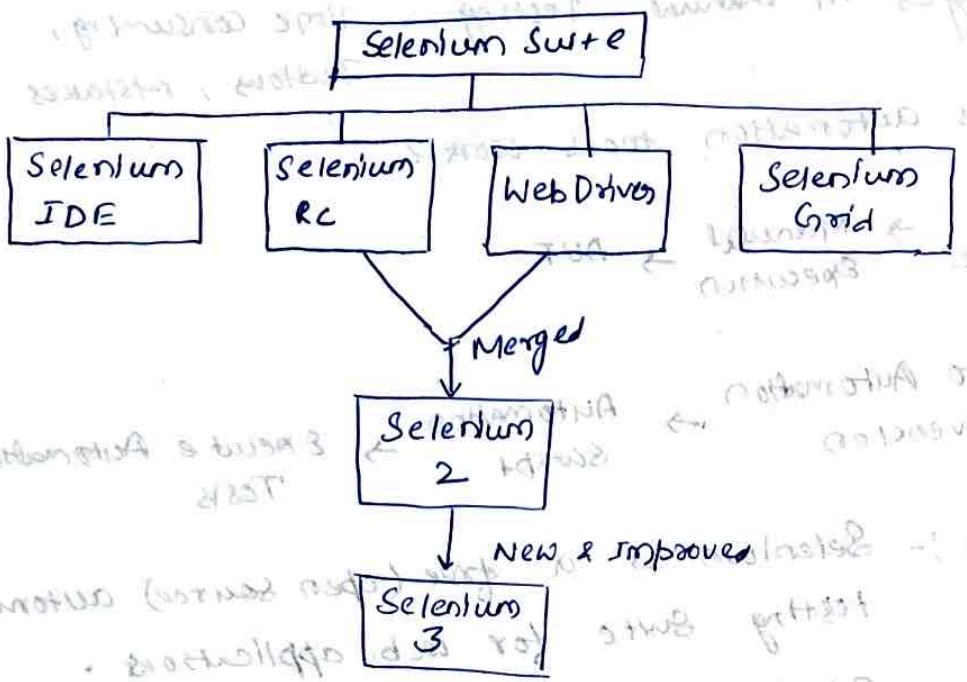
Supports many browsers

Can be integrated with Jenkins for Continuous Integration

Disadvantages → Used to test only web applications, not desktop application.

For image based testing, we need to integrate Sikuli.

- No - native reporting facility, needs to integrate with TestNG.



- Selenium IDE → A firefox and Chrome add on that you can only use in creating relatively simple test cases and test suits.

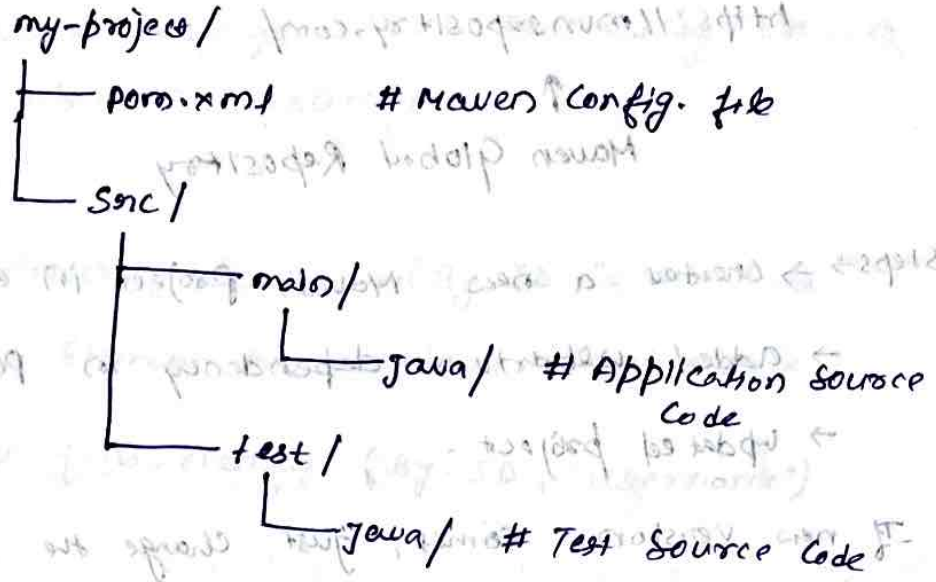
- Selenium Remote Control → Also known as Selenium 1, It allows users to use programming languages in creating complex tests.

- Web Drives → The newer breakthrough that allows you test scripts to communicate directly to the browsers. It is a software component that allows to programmatically control a web Browsers.
  - → It opens a Browsers
  - → Navigate to web pages
  - → Click Buttons, files, fill forms, scroll pages
  - → Extract data from web page

- Selenium Grid → It is a tool that allows to execute parallel tests across different browsers and operating systems remotely.

Maven → It is a build automation and project management tool primarily used for Java projects.

Structure →



Steps to create Maven project –

→ Create a Maven project

→ Create a simple project ☑ → Next

→ Group Id (project Name)
Artifact ID (project Name)

→ Finish

☐ seleniumwebdrivers
- → src/main/java
- → src/main/resources] used for development
- → src/test/java
- → src/test/resources] used for testing

JRE System Library

src

target

pom.xml → Jar files are included automatically

Pom.xml → Dependencies (something we need to add in existing project)

Go to mvnrepository

https://mvnrepository.com/

↑

Maven Global Repository

Steps → → Created a new Maven project in eclipse

→ added webdriver dependency in pom.xml

→ updated project.

If new version comes, just change the version and update project, all dependencies get automatically updated

First Automation program

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

public class Main
{
    public static void main(String[] args)
    {
        WebDriver driver = new ChromeDriver();
```

To open URL `driver.get("url");`

To get title `String actual_title = driver.getTitle();`

`System.out.println(actual_title);`

Selenium Locators - ID, Name, Link Text, Partial Link Text, Tag Name

Selenium Locators are used to find and interact with web elements on a webpage when automating browser actions using Selenium WebDrivers.

1. ID Locator :-

Syntax :- driver.find_element(By.ID, "element_id")

Use when :- Element has a unique id attribute

ex :- driver.find_element(By.ID, "username")

2. Name Locator

Syntax :- driver.find_element(By.NAME, "element_name")

Use when :- The element has name attribute

ex :- driver.find_element(By.NAME, "email")

3. Class Name Locator

Syntax :- driver.find_element(By.CLASS_NAME, "class_name")

Use when :- The element has class attribute

ex :- driver.find_element(By.CLASS_NAME, "btn-primary")

4. Tag Name Locator

Syntax :- driver.find_element(By.TAG_NAME, "tag_name")

Use when :- you want to find element by HTML Tag

ex :- driver.find_element(By.TAG_NAME, "input")

5. Link Text Locator

Syntax :- driver.find_element(By.LINK_TEXT, "link text")

Use when :- you want to find a hyper link by it's exact visible text

eg :- driver.find_element(By.LINK_TEXT, "Forget password")

6. Partial Link Text Locator

Syntax: drives.find_element(BY.PARTIAL_LINK_TEXT, "Forgot")

Use when - you want to match part of the link text

7. Css selector → customize locator

Syntax:- drives.find_element(BY.CSS_SELECTOR, "input[type='text']")

ease when :- you want to use CSS style selector

8. X Path → customize locator

Syntax :- drives.find_element(BY.XPATH, "xpath_expression")

use when → you need a powerful and flexible way to locate element

e.g → drives.find_element(BY.XPATH, "//input[@id='username']");

input → tagname

type, name, value, placeholder → Attributes

findElement(By.name("search")) → WebElement

WebElement searchBox = drives.findElement(By.name("search"));

[Ctr + shift + O → To add import after hovering to WebDriver]

To maximize of full screen window

↓

drives.manage().window().maximized();

driver.get("https://demo.nopcommerce.com/");

driver.manage().window().maximize();

driver.findElement(By.className(" ")).sendKeys("Macbook");

use to set a value

isDisplayed() → method to return an object is present

if you want to perform more operations onto an element then we should store it into variables, and perform the operations.

If there is multiple links with same link text then don't prefer LinkText or PartialLinkText, prefer xpath

TagName and ClassName is used whenever we want to locate a group of webelement or multiple webelement

i.e, <img>, <a>

↓

To locate multiple elements we use

driver.findElements( )

The multiple webElement will be stored in variable of returntype List.

List<WebElement>

List<WebElement> links = driver.findElements(By.tagName("a"));

System.out.println(links.size());

↳ prints total link present in webpage in number

findElement() vs findElements()

findElement() → will throw error

findElements() → will print zero if element not present

CSS selectors

tag id — tag#id

tag class — tag.classname

tag attribute — tag[attribute="value"]

tag class attribute — tag.classname[attribute="value"]

tag id → driver.findElement(By.cssSelector("input#searchbar")).sendKeys("clothes");

tag class → driver.findElement(By.cssSelector("input.searchbar")).sendKeys("iphone");

tag attribute → driver.findElement(By.cssSelector("input[placeholder='Search Store']")).sendKeys("mobiles");

tag class attribute

driver.findElement(By.cssSelector("input.search-box[name='q']")).sendKeys("Hii");

X-path :- It is the address of the element, we can capture the x-path from the DOM.

Two types of x-path :-

i) Absolute x-path (full x-path)

ii) Relative x-path (partial x-path)

Absolute → /html/body/header/div/div/div[2]/div/input

Relative → //*[@id="search"]/input

Which x-path should be preffered?

→ Relative x-path, as Absolute x-path contains the whole hierarchy and any changes in any part of the hierarchy will impact the x-path

But Relative x-path focuses on the attribute of the element i.e., //*[@name="search"]

Hence, changes in the webpage will not impact.

Difference Between Absolute and Relative x-path :-

ABSOLUTE	RELATIVE
→ Starts with single /	→ Starts with double //
→ Absolute x-path do not use attributes	→ Relative x-path uses attributes
→ Absolute x-path traverse through each node till it finds element.	→ Relative x-path directly jump and find the element by using attribute.

Relative xpath / partial x-path

Can generate —

i> Automatically (using Devtools, selectors)

ii> Manually (own x-path)

↓

Syntax :- //tagname [@attribute = 'value']

OR

//*[@attribute = 'value']

↓

for putting double quotes we need to use Regular expression like

//*[@title = /"MacBook"/]

If we use multiple attributes in x-path, then all the attributes should be correct, even one attribute is not correct element will not be located it will show exception.

eg > By.xpath("//input [@name = 'search'] [@placeholder = 'search-bar'])

X-path with 'and' 'OR' operator

//input [@name = 'search' and @placeholder = 'search-bar'])

Both must be correct

//input [@name = 'search' OR @placeholder = 'search-bar']

Any one is true

than element can be located

x-path with inner text() method

```
//a[text()='Desktops']
```

```
driver.findElement(By.xpath("//*[text()='MacBook']")).click();
```

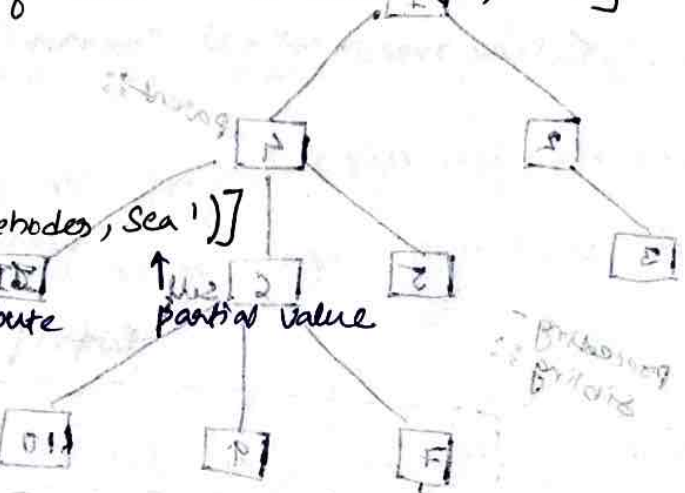
How to capture a label just a text →

```
//h3[text()='Featured']
```

```
String value = driver.findElement(By.xpath("//h3[text()='Featured']")).getText();
System.out.(value);
```

we can use . instead of text `//[contains(.,'')]`

x-path with Contains



x-path with start with

```
("//input[starts-with(@placeholder,'Sea')]")
```

Contains and startwith are used to handle dynamic attributes.

eg→ if id keeps changing as start & stop

xpath = `//*[@id='start' or @id='stop']`

`//*[contains @id,'st']`

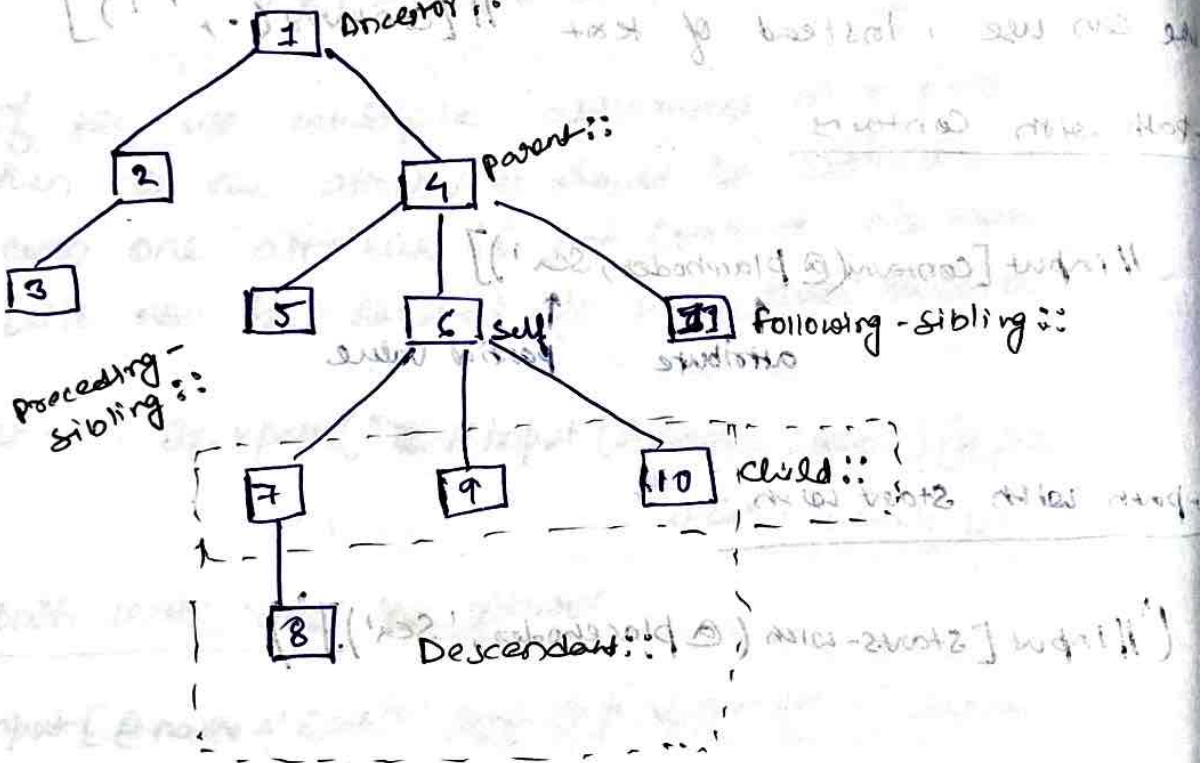
`//*[starts-with @id,'st']`

Chained x-path → If you dont have attributes or inner text of the element

//div[@id='logo']/a/img

Write x-path for parent then navigate to child
It is the combination of Relative and Absolute x-path.

X-path Axis → More efficient way of writing locators of complicated elements



Locating a parent element

e.g → <form id = "frmLogin" method = 'post' action = " ">
<input name = "txtUsername" id = "txtUsername" type = "text

ex → //input[@id='txtUsername']/parent::form

x-path of self Element — tag name of parent element

Locating a child element from parent

```
<div id = "divUsername" class = "textInputContainer">
    <input name = "txtusername" id = "txtUsername" type = "text">
</div>
```

x-path → //div [@id = 'divusername'] / child :: input

Locating Grand Children

```
<form id = "fromLogin" method = "post" action = "/index.php">
    <div id = "divUsername" class = "textInput">
    <input name = "txtusername" id = "txtUsername">
</div>
```

x-path → //form /*/input

//form/div/input

Locating Ancestors (parent of parent)

//input [@id='txtusername']/ancestor::form

Locating Descendant (child of child)

//form[@id='fromLogin']/descendant::input

How to handle web elements

i) get methods
ii) Conditional methods
iii) Browser Methods
iv) Navigational Methods
v) Wait Methods

i) get methods → We can access these methods through WebDriver instance

get(url) → Open the url on the browser
getTitle() → Returns title of the page
getCurrentUrl() → Returns url of the page
getPageSource() → Returns the source code of page
getWindowHandle() → Return ID of the single Browser window
getWindowHandles() → return IDs of the multiple browser windows

ii) Conditional Methods → We can access through the (WebElement not WebDriver)

Returns Boolean values

isDisplayed() → for images and for all elements
isEnabled() → for input boxes
isSelected() → used for Radio buttons/checkboxes

iii) Browser Methods

close() → close single Browser
quit() → Closes multiple Browsers

Web Drivers Methods – Wait Method

Synchronization :- During running automation script the elements and webpage are not available at the same time, hence exception occurs.

Cause → Execution of automation script is faster than Application response.

When page takes more time to load, then we use Thread.sleep("5000");

↑ It is a java Method

Selenium provides two wait commands :-

i) Implicit wait

ii) Explicit wait / fluent wait

NoSuchElementException :- Element is not present on webpage, due to synchronization

ElementNotFoundException :- Locator is incorrect

Disadvantages of Thread.sleep()

i) If the time is not sufficient then also exception occurs.

ii) If the webpage is loaded before the maximum timeout it will reduce the performance

iii) We need to write multiple times.

Implicity wait

Should be put as a one single statement at the begining after the drives instance is created.

```
drives.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));
```

Disadvantage of Implicit wait → If an element is taking more than the defined time than it will give exception.

Standard time is 10 sec (min)

Explicit / Fluent wait (Based on condition)

Two steps → i) Declaration

ii) Usage

Add special class →

```
WebDriverWait mywait = new WebDriverWait();
```

Takes two parameters

```
(drives, Duration.ofSeconds(10));
```

To apply →

```
mywait.until(ExpectedConditions.visibilityOfElementLocated
(Locator));
```

Navigational Commands

navigate().to(Url) → Same as get(url) command

navigate().back() → To go back to previous page

navigate().forward() → To go to forward

navigate().refresh()

In navigate().to() method we can pass URL as string as well as an object.

i.e, URL myurl = new URL("https://demo...");

driver.navigate().to(myurl);

↑

object of URL

But driver.get() → Accepts Url in string format only

These commands are used so that we dont need to give the full Url each time to go and back.

Use of getWindowHandle() & getWindowHandles()

↓ Return id of single Browser window

↓ Return id's of multiple Browser window

Switch Command

How to capture id's of diffent windows / tabs

driver.get("https://opensource-demo...");

driver.findElement(By.xpath("//a[@id='btn-1']").click();

Parent window → Child window

driver se windowHandles kaise lete hai? →

```
Set<String> windowIDs = driver.getWindowHandles();

//Approach 1 (using List → It contains get() method
                only used for less no. of tabs)

List<String> windowList = new ArrayList(windowIDs);
String parentID = windowList.get(0);
String childID = windowList.get(1);

//Switch to child window
driver.switchTo().window(childID);
Syso(driver.getTitle());

//Switch to parent window
driver.switchTo().window(parentID);
Syso(driver.getTitle());
```

↓ ↓

In order to switch to other windows we need to capture their id's.

Command →

```
driver.switchTo().window(windowID);
```

Handle Check Boxes & Radio Buttons

```
List<WebElement> CheckBoxes = driver.findElement(By.xpath(" "));

for(WebElement chk : CheckBoxes)
{
    chk.click();
}
```

Using normal for loop →

```
for(int i=0 ; i<CheckBoxes.size() ; i++ )
{
    CheckBoxes.get(i).click();
}
```

Alert / Popups

There are 3 alerts in selenium

- Normal JS Alert
- Confirm Alert
- JS prompt alert

Alerts or popups are not web elements we can't inspect it

i> Handling normal Alert with OK Button

To handle alert and popups we need to switch our driver from webpage to the alert window by,

```
driver.switchTo().alert().accept();
```

To perform operation in the alert window store it in a variable.

```
Alert myalert=driver.switchTo().alert();

System.out.println( myalert.getText());

myalert.accept();
```

ii) Handling Confirmation Alert (OK & confirmation)

```
driver.switchTo().alert().accept();
```

↑

closes the alert using OK button

```
driver.switchTo().alert().dismiss();
```

↓

Closes the alert using Cancel button

iii) Handling prompt Alerts

passing text to the prompt

```
Alert myalert = driver.switchTo().alert();
myalert.sendKeys("Welcome");
myalert.dismiss();
```

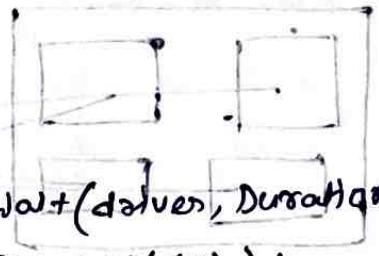
or

```
myalert.accept();
```

```
driver.switchTo().alert().sendKeys(text).accept();
```

Handle Alerts using Explicit wait

Without using switchTo().alert() we can handle Alerts by explicit wait.



```
WebDriverWait mywait = new WebDriverWait(driver, Duration.
                                ofSeconds(10));
driver.get("https://-- --");
driver.findElement(By.xpath("//input[@id='alertexample']")).click();
```

↑ Clicking on Alert

```
Thread.sleep(3000);
Alert myalert = mywait.until(ExpectedConditions.alertIsPresent());
myalert.getText();
myalert.accept();
```

Authenticated Alerts (popup asking for username, password)

We will bypass the username & password along with the URL called as injection process.

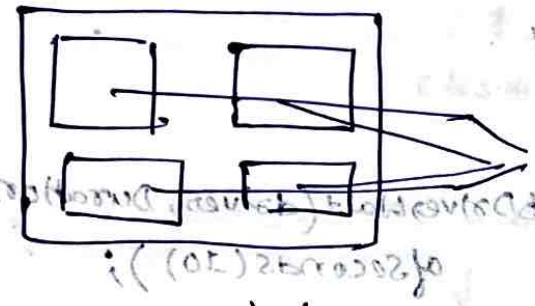
Don't pass direct URL, pass the username and password along with URL

Syntax → driver.get("http://username:password@the-internet.herokuapp.")

```
driver.get("http://admin:admin@url");
```

Frames

Frame is a subset of webpage which contains elements



Iframe → Using iframe we can embed one webpage into another webpage

Command → switchTo().frame();

↳ id, name, locator

driver.switchTo().frame(name)

- frame(id)
- frame(WebElement)
- frame(Index) ↘ use only when there is only one frame as Index = 0

We need to first switch to the frame then only the locators can be used

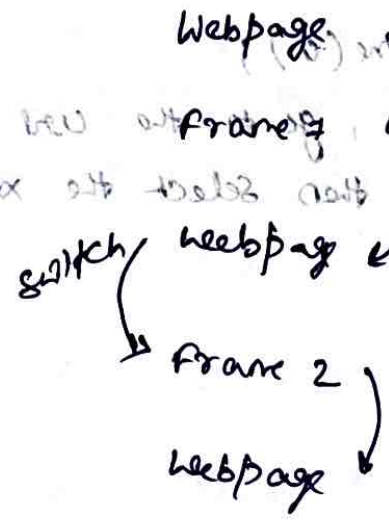
```
//Frame 1
WebElement frame1 = driver.findElement(By.xpath("//frame[@src='frame1.html']"));
driver.switchTo.frame("frame1");

driver.findElement(By.xpath("//input[@name='mytext1']")).
                                        sendKeys("welcome");
```

```
//Frame 2
WebElement frame2 = driver.findElement(By.xpath("//frame
```

Before switching to another frame we need to come out to the page then switch to another frame

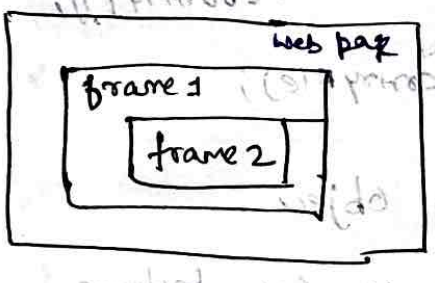
We can't switch from one frame to another frame directly.

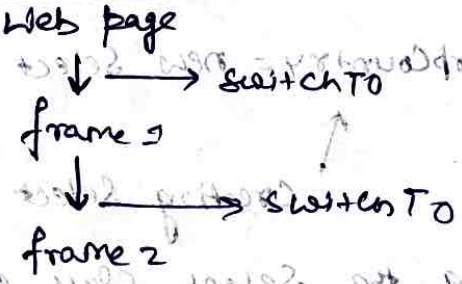


To switch Back to webpage →

driver.switchTo().defaultContent();

Inner frames :- When frame contains another frame or nested frames





In this case no need to go back to the Webpage

Bcoz Frame 2 is part of frame 1

Here, Reverse option is not available

(frame 2 → frame 1) is not possible

So, driver.switchTo().defaultContent() will get back to main page

Inner frame to outer frame is not possible

To interact with elements of iframe or embeded webpage.

We need to switchTo.frame(0);

Then to locate the xpath, go to the url of the embeded page and then select the xpath.

Handle Dropdown Box

1) Select dropdown
2) Bootstrap dropdown
3) Hidden drop down

1) Select Dropdown

```
WebElement dropCountryEle = driver.findElement(By.xpath("//select[@id='country']"));
```

```
Select dropCountry = new Select(dropCountryEle);
```

↑ Creating Select Class object

By using the Select class object we can perform operations in the dropdown

// To select options from Dropdown

i) selectByVisibleText()

ii) selectByValue()

iii) selectByIndex()

```
dropCountry.selectByVisibleText("France");
```

OR

```
dropCountry.selectByValue("China");
```

value → Inner text from inspecting

OR

```
dropCountry.selectByIndex(5);
```

~~ii) Bootstrap Dropdown~~

```
drpCountry.getOptions();
```

↑

This will return the ~~list~~ count no of options

```
List<WebElement> options = drpCountry.getOptions();
System.out.println(options.size());
```

Printing the options

```
for(int i=0; i<options.size(); i++)
{
Syso( options.get(i).getText());
}
```

```
for(WebElement ~~select~~ op : options)
{
Syso(~~Webelement~~ op.getText());
}
```

2. Bootstrap Down → We have to handle using x-path only

3. Hidden Dropdown → Some dropdowns are not visible on the HTML skeleton, they get loaded during the runtime

We should use Selector Hub's Debugger to freeze and then inspect the web element

~~33. Handle Static Web Table~~

36. Handling Mouse Events

Actions → Mouse hover → MoveToElement()
right click
~~left~~ double click
drag & drop

Actions → predefined class provided by selenium

i) Mouse Hover Action → MoveToElement()

```
Actions act = new Actions(driver);
```
→ pass driver as parameter

Package → import org.openqa.selenium.interactions.Actions;

act.moveToElement(desktops); → This method is used for Mouse Hover

act.moveToElement(desktops).moveToElement(mac);

But to reflect on the UI we must add

build().perform() after every action method

build() → create action
perform() → complete/performs action

act.moveToElement(desktops).moveToElement(mac).build().perform();

We can also click() but still need to end with build().perform()

build() → Creates an Action

Perform() → complete the action as well as can build an action

* We can skip the build() as perform() will do the same alone

Right click action → contextClick()

```
driver.get("https://swisnl.github.io/jQuery-contextMenu/demo.html");
WebElement btn = driver.findElement(By.xpath(" "));

Actions act = new Actions(driver);

act.contextClick(btn).build().perform();
```

contextClick() → This method is used for Right click

Double Click Action → doubleClick()

```
act.doubleClick(button).build().perform()
```

getText() vs getAttribute()

↓ gives the inner text

↘ gives the attribute

```
<input id = "xyz"> Welcome </input>
```

↑ This is inner text

We can used getText() method it will give, welcome

getAttribute("id") → it will give xyz

Attribute can be id, text, value etc

We use getAttribute when innerText is not present in HTML

drag and drop

↓

files within the Application only is possible

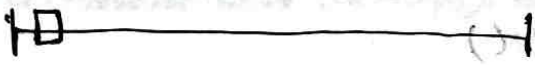
Can't drag & drop from Desktop it is called uploading

Need to Capture Source and target elements

```
Actions act = new Actions(drivers);
WebElement rome = drivers.findElement(By.xpath(" "));
WebElement italy = drivers.findElement(By.xpath(" "));
.act.dragAndDrope(rome, italy).perform();
                  ↓      ↓
                Source  target
```

Day-36

Handling Slides



In case of horizontal Slides

Change the position of x-Co-ordinate

(x, y)

↓

will change

In Case of vertical Slides

the position of

y-Co-ordinate will change

(x, y)

↑

this value will change

WebElement min-slider = driver.findElement(By.xpath("//span[3]"));

min-slider.getLocation(); (59,251)

↓

This method gives the Co-Ordinate

min-slider.getLocation().getX();

↓

This gives the X-Cordinate only

To drag the Coordinate we use

Actions act = new Actions(driver);

act.dragAndDropBy(min-slider, x-axis, Y-axis);

- act.dragAndDropBy(min-slider, 100, 251).perform();

↓

59+100 = 159