1. **What is Software Testing?**

**Ans:** is a process used to identify the correctness, completeness and the quality of the developed software.

1. **Why Software Testing Required?**

Ans: Quality Assurance, Error Free Code, Safety Assurance, Bug Free Application, Multiple OS support.

1. **What is Quality Control?**

**Ans:** is a product-oriented approach of running a program to determine if it has any defects, as well as making sure that the software meets all the requirements.

Example: functional testing and Non-functional Testing and compatibility performance.

1. **What is End to End Testing?**

Ans: **End-to-end testing** is a technique used to test whether the flow of an application right from start to finish is behaving as expected. The purpose of performing **end-to-end testing** is to identify system dependencies and to ensure that the data integrity is maintained between various system components and systems.

1. **Regression Testing?**

Ans: It is a type of Software testing executed to check whether a code change has not disturbed current features and functions of an application.

We will be testing only passed test cases in regression testing.

1. **Retesting?**

Ans: This testing is performed to ensure that the defects which were found and posted in the earlier

The retesting will be done on failed test cases only and it is planned testing

1. **Smoke Testing?**

Ans: **SMOKE TESTING**, also known as “Build Verification Testing”, is a type of software testing that comprises of a non-exhaustive set of tests that aim at ensuring that the most important functions work. The result of this testing is used to decide if a build is stable enough to proceed with further testing.

1. **Sanity Testing?**

Ans: The main aim of **Sanity testing** to check the planned functionality is working as expected. Instead of doing whole regression **testing** the **Sanity testing** is performing. **Sanity tests** helps to avoid wasting time and cost involved in **testing** if the build is failed. Tester should reject the build upon build failure.

1. **Accessibility Testing?**

Ans: **Accessibility Testing** is defined as a type of Software **Testing** performed to ensure that the application being tested is usable by people with disabilities like hearing, colour blindness, old age and other disadvantaged groups.

Examples:

* **Speech Recognition Software -** It will convert the spoken word to text, which serves as input to the computer.
* **Screen reader software** - Used to read out the text that is displayed on the screen
* **Screen Magnification Software**- Used to enlarge the monitor and make reading easy for vision-impaired users.
* **Special keyboard** made for the users for easy typing who have motor control difficulties

1. **Are mobile testing and device testing being same or not?**

Ans: **No Not same**

**Desktop application:**  
  
A native application that executes on a user's local machine.  This application may or may not have a network component, although most desktops have some kind of network component these days, even if it's just to update itself online.  To keep things simple, let's put it this way: if you need to update the application, an update needs to be downloaded locally.  
  
An example of a desktop app would be MS Word, Adobe Photoshop, a web browser.  
  
**Mobile application:**  
  
An application built to run natively on a mobile device.  The most common devices these days are either iOS or Android based, but there is a small population of Windows mobile users and a few folks still clinging to their Blackberries.  Mobile applications, similar to desktop apps may or may not have a network component -- but with the exception of the calculator on my phone, I can't think of any that don't.  Just like a desktop app, if you need to make an update to this app, something needs to be downloaded and installed.  
  
An example of a mobile app would be Angry Birds, Harvest for iOS (sorry, shameless plug) or the SMS app.

**Note: Include this they ask about common applications**  
**Web applications:**  
  
Applications that run 100% within a browser. This is where the waters get muddy, because there are a group of apps on both the desktop and on mobile that are just web apps, running within an app-specific browser.  An example of one of these would have been the early versions of the Facebook app -- those were just a bunch of web views running within a browser window.  Sometimes those are hard to identify, but a sure-fire tell is if your app updates without you needing to do anything from the App Store/Play Store.

1. **Software Defect Life Cycle:**

**Ans:** [Defect](https://www.guru99.com/the-unconventional-guide-to-defect-management.html)Life Cycle or Bug Life Cycle is the specific set of states that a Bug goes through from discovery to defect fixation.



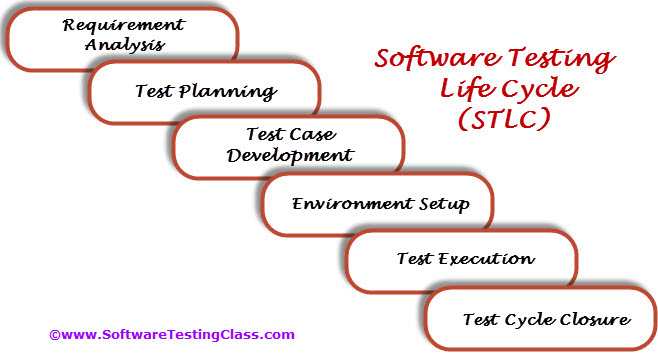
* **New:** When a new defect is logged and posted for the first time. It is assigned a status as NEW.
* **Assigned:** Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team
* **Open**: The developer starts analyzing and works on the defect fix
* **Fixed**: When a developer makes a necessary code change and verifies the change, he or she can make bug status as "Fixed."
* **Pending retest**: Once the defect is fixed the developer gives a particular code for retesting the code to the tester. Since the software testing remains pending from the testers end, the status assigned is "pending request."
* **Retest**: Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to "Re-test."
* **Verified**: The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is "verified."
* **Reopen**: If the bug persists even after the developer has fixed the bug, the tester changes the status to "reopened". Once again the bug goes through the life cycle.
* **Closed**: If the bug is no longer exists then tester assigns the status "Closed."
* **Duplicate**: If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to "duplicate."
* **Rejected**: If the developer feels the defect is not a genuine defect then it changes the defect to "rejected."
* **Deferred**: If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status "Deferred" is assigned to such bugs
* **Not a bug**:If it does not affect the functionality of the application then the status assigned to a bug is "Not a bug".

1. **Software Testing Life Cycle?**

**Ans: Software Testing Life Cycle (STLC)** is the testing process which is executed in systematic and planned manner. In STLC process, different activities are carried out to improve the quality of the product. Let’s quickly see what all stages are involved in typical Software Testing Life Cycle (STLC).

Steps involved:

* Requirement Analysis
* Test Planning
* Test Case Development
* Environment Setup
* Test Execution
* Test Cycle Closure



**Requirement Analysis:**

Requirement Analysis is the very first step in **Software Testing Life Cycle (STLC)**. In this step Quality Assurance (QA) team understands the requirement in terms of what we will testing & figure out the testable requirements. If any conflict, missing or not understood any requirement, then QA team follow up with the various stakeholders like Business Analyst, System Architecture, Client, Technical Manager/Lead etc to better understand the detail knowledge of requirement.

From very first step QA involved in the where STLC which helps to prevent the introducing defects into Software under test. The requirements can be either Functional or Non-Functional like Performance, Security testing. Also requirement and Automation feasibility of the project can be done in this stage (if applicable)

|  |  |  |
| --- | --- | --- |
| **Entry Criteria** | **Activities** | **Deliverable** |
| Following documents should be available:  – Requirements Specification.  – Application architectural  Along with above documents Acceptance criteria should be well defined. | Prepare the list of questions or queries and get resolved from Business Analyst, System Architecture, Client, Technical Manager/Lead etc.  Make out the list for what all Types of Tests performed like Functional, Security, and Performance etc.  Define the testing focus and priorities.  List down the Test environment details where testing activities will be carried out.  Checkout the Automation feasibility if required & prepare the Automation feasibility report. | List of questions with all answers to be resolved from business i.e. testable requirements  Automation feasibility report (if applicable) |

**Test Planning:**

Test Planning is most important phase of *Software testing life cycle* where all testing strategy is defined. This phase also called as **Test Strategy** phase. In this phase typically Test Manager (or Test Lead based on company to company) involved to determine the effort and cost estimates for entire project. This phase will be kicked off once the requirement gathering phase is completed & based on the requirement analysis, start preparing the Test Plan. The Result of Test Planning phase will be [Test Plan](https://www.softwaretestingclass.com/test-plan-template/)or Test strategy & Testing [Effort estimation](https://www.softwaretestingclass.com/software-estimation-techniques/) documents. Once test planning phase is completed the QA team can start with test cases development activity.

Get the [**Sample Test Plan Template**](https://www.softwaretestingclass.com/test-plan-template/) here.

|  |  |  |
| --- | --- | --- |
| **Entry Criteria** | **Activities** | **Deliverable** |
| Requirements Documents (Updated version of unclear or missing requirement).  Automation feasibility report. | Define Objective & scope of the project.  List down the testing types involved in the STLC.  Test effort estimation and resource planning.  Selection of testing tool if required.  Define the testing process overview.  Define the test environment required for entire project.  Prepare the test schedules.  Define the control procedures.  Determining roles and responsibilities.  List down the testing deliverable.  Define the entry criteria, suspension criteria, resumption criteria and exit criteria.  Define the risk involved if any. | [Test Plan](https://www.softwaretestingclass.com/test-plan-template/) or Test strategy document.  Testing estimation document. |

**Test Case Development:**

The test case development activity is started once the test planning activity is finished. This is the phase of STLC where testing team write down the detailed test cases. Along with test cases testing team also prepare the test data if any required for testing. Once the test cases are ready then these test cases are reviewed by peer members or QA lead.

Also the Requirement Traceability Matrix (RTM) is prepared. The Requirement Traceability Matrix is an industry-accepted format for tracking requirements where each test case is mapped with the requirement. Using this RTM, we can track backward & forward traceability.

|  |  |  |
| --- | --- | --- |
| **Entry Criteria** | **Activities** | **Deliverable** |
| Requirements Documents (Updated version of unclear or missing requirement).  Automation feasibility report. | Preparation of test cases.  Preparation of test automation scripts (if required).  Re-requisite test data preparation for executing test cases. | Test cases.  Test data.  Test Automation Scripts (if required). |

**Test Environment Setup:**

Setting up the test environment is vital part of the STLC. Basically test environment decides on which conditions software is tested. This is independent activity and can be started parallel with Test Case Development. In process of setting up testing environment test team is not involved in it. Based on company to company may be developer or customer creates the testing environment. Meanwhile testing team should prepare the smoke test cases to check the readiness of the test environment setup.

|  |  |  |
| --- | --- | --- |
| **Entry Criteria** | **Activities** | **Deliverable** |
| Test Plan is available.  Smoke Test cases are available.  Test data is available. | Analyze the requirements and prepare the list of Software & hardware required to set up test environment.  Setup the test environment.  Once the Test Environment is setup execute the Smoke test cases to check the readiness of the test environment. | Test Environment will be ready with test data.  Result of Smoke Test cases. |

**Test Execution:**

Once the preparation of Test Case Development and Test Environment setup is completed then test execution phase can be kicked off. In this phase testing team start executing test cases based on prepared test planning & prepared test cases in the prior step.

Once the test case is passed then same can be marked as Passed. If any test case is failed, then corresponding defect can be reported to developer team via bug tracking system & bug can be linked for corresponding test case for further analysis. Ideally every failed test case should be associated with at least single bug. Using this linking we can get the failed test case with bug associated with it. Once the bug fixed by development team then same test case can be executed based on your test planning.

If any of the test cases are blocked due to any defect then such test cases can be marked as Blocked, so we can get the report based on how many test cases passed, failed, blocked or not run etc. Once the defects are fixed, same Failed or Blocked test cases can be executed again to retest the functionality.

|  |  |  |
| --- | --- | --- |
| **Entry Criteria** | **Activities** | **Deliverable** |
| [Test Plan](https://www.softwaretestingclass.com/test-plan-template/)or Test strategy document.  Test cases.  Test data. | Based on test planning execute the test cases.  Mark status of test cases like Passed, Failed, Blocked, Not Run etc.  Assign Bug Id for all Failed and Blocked test cases.  Do Retesting once the defects are fixed.  Track the defects to closure. | Test case execution report.  Defect report. |

**Test Cycle Closure:**

Call out the testing team member meeting & evaluate cycle completion criteria based on Test coverage, Quality, Cost, Time, Critical Business Objectives, and Software. Discuss what all went good, which area needs to be improve & taking the lessons from current STLC as input to upcoming test cycles, which will help to improve bottleneck in the STLC process. Test case & bug report will analyze to find out the defect distribution by type and severity. Once complete the test cycle then test closure report & Test metrics will be prepared. Test result analysis to find out the defect distribution by type and severity.

|  |  |  |
| --- | --- | --- |
| **Entry Criteria** | **Activities** | **Deliverable** |
| Test case execution is completed  Test case Execution report  Defect report | Evaluate cycle completion criteria based on Test coverage, Quality, Cost, Time, Critical Business Objectives, and Software Prepare test metrics based on the above parameters.  Prepare Test closure report  Share best practices for any similar projects in future | Test Closure report  Test metrics |

1. **What is RTM (Requirement Traceability matrix)?**

**Ans:** The relationship between test cases and requirements is shown with the help of a document. This document is known as traceability matrix.

1. **What is Exploratory Testing?**

**Ans:**

* It is also known as Ad hoc testing.
* It is a simultaneous learning, test planning, analysis, test design and test execution all done together instantly.
* This testing is about exploring the system and encouraging real-time and parallel thinking of a tester.

1. **What is User Acceptance testing?**

**Ans:** User Acceptance is defined as a type of testing performed by the Client to certify the system with respect to the **requirements** that was agreed upon. This testing happens in the final phase of testing before moving the software application to the Market or **Production** environment.

1. **Alpha Testing:** Simulated (or) actual operational testing performed by end user with in a company but outside the development group
2. **Beta Testing:** Simulated (or) actual operational testing performed by Sub-set of actual customers outside the company
3. **Integration Testing?**

**Ans: INTEGRATION TESTING** is a level of software **testing** where individual units are combined and **tested** as a group. The purpose of this level of **testing** is to expose faults in the interaction between **integrated** units. **Test** drivers and **test** stubs are used to assist in **Integration Testing**.

1. **What is Data Driven Testing (DDT)?**

**Ans:**In data driven testing process, application is tested with multiple test data. Application is tested with different set of values.

1. **When a defect found in production from your testing?**

**Ans:** First we need to perform Root Cause analysis

should note it down so that I will not repeat it

out of box scenarios and should learnt to deal with such scenarios.

1. **What is Test case?**

**Ans:** It is a document that describes an input action (or) event and an expected response to determine whether a feature of an application is working fine or not.

1. **What is Test Scenario?**

**Ans:** A Test Scenario is any functionality that can be tested. It is also called Test Condition or Test Possibility.

1. **How will u identify regression scenarios?**

**Ans:**

1. Include the test cases that have frequently yielded bugs: Some areas in the application are so error-prone that they usually fail following a small coding change. We can keep track of these failing test cases throughout the product cycle and cover them in the regression test suite.

2. Include the test cases that verify core features of the application: Prior to designing the test cases, identify all the core features of the application. Ensure that test cases cover all functionality mentioned in the requirements document. One can make use of a traceability matrix to make sure that no requirement is left untested.

3. Include the test cases for functionalities that have undergone recent changes: Maintain the history of functionality changes for test case documentation in order to identify the test cases to include in the regression suite.

4. Include all the integration test cases: Even if integration testing is a separate part of the software testing cycle, its test cases should be included in the regression test suite. A last-minute fix, an already-tested application can break the integrity between two different modules. For example, data might get lost across an interface, messages might not get passed properly, or interfaces might not be implemented as specified.

5. Include all complex test cases: Some system functionality may only be accomplished by following a complex sequence of graphic user interface (GUI) events. To open a file, a user may have to click on the “File” menu and then select “Open,” use a dialog box to specify the file name, and then focus the application on the newly opened window. Obviously, increasing the number of possible operations exponentially augments the sequencing problem. This can become a serious issue; in some scenarios, the whole system’s functionality comes to a halt. As a result, all complex test cases should be part of the regression test suite.

6. Prioritize the test cases for regression testing: Prioritize the test cases as they relate to business impact and critical and frequently used functionalities. It is always helpful if an analysis is completed to determine which test cases are relevant. One idea is to classify the test cases into various priorities based on importance and customer use. Here, it is suggested that test cases be sorted into three categories:

* Priority 0: Sanity test cases check for basic functionality (as per the SRS of the application) and are run to verify pre-system acceptance and ensure functionality after an application under test goes through a major change. These test cases deliver high project value.
* Priority 1: This includes the test cases that test the essential functionalities for delivering high project value.
* Priority 2: These are executed as a part of the system test cycle and are selected for regression testing as-needed. These test cases deliver moderate project value.

The selection of test cases based on priority will greatly reduce efforts spent on regression testing.

1. **Entry Criteria?**

**Ans:** Entry Criteria for STLC phases can be defined as specific conditions; or, all those documents which are required to start a particular phase of STLC should be present before entering any of the STLC phase.

1. **Exit Criteria?**

**Ans:** Exit Criteria for STLC phases can be defined as items/documents/actions/tasks that must be completed before concluding the current phase and moving on to the next phase.

1. **Difference between Quality Control and Quality Assurance?**

**Ans:**

|  |  |
| --- | --- |
| Quality Control | Quality Assurance |
| Is a set of activities for ensuring quality in the products | Is a set of activities for ensuring the quality in the process by which products are developed |
| Aims to identify the defected in completed product | Aims to prevent the defects with a focus on process used to make the product |
| Is a product oriented approach | Is process oriented approach |

1. **Test Bed?**

**Ans:** is an Environment used for testing an application, including the hardware and software needed to run the program to be tested

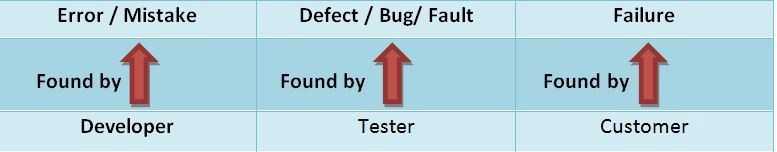
1. **Verification and validation?**

**Ans:**

|  |  |
| --- | --- |
| Verification | Validation |
| It is a static analysis technique. Testing is done without executing the code | It is a dynamic analysis technique where testing is done executing the code. |
| The verifying process includes checking documents, design, code, and program | validating the actual product |
| Verification uses methods like reviews, walkthroughs, inspections, and desk- checking etc. | It uses methods like Black Box Testing, [White Box Testing](https://www.guru99.com/white-box-testing.html), and non-functional testing |
| Example: **A *clickable* button *with name Submet***  Verification would check the design doc and correcting the spelling mistake.  Otherwise, the development team will create a button like | Example: **A *clickable* button *with name Submet***  Validation will check button is working or not. |

1. **Difference between Bug, Defect and Error?**

**Ans:**



**Defect**: The variation between the actual results and expected results is known as defect. If a developer finds an issue and corrects it by himself in the development phase then it’s called a defect.

**Bug**: If testers find any mismatch in the application/system in testing phase then they call it as Bug. As I mentioned earlier, there is a contradiction in the usage of Bug and Defect. People widely say the bug is an informal name for the defect.

**Error**: We can’t compile or run a program due to coding mistake in a program. If a developer unable to successfully compile or run a program, then they call it as an **error**.

**Failure**: Once the product is deployed and customers find any issues then they call the product as a failure product. After release, if an end user finds an issue then that particular issue is called as **failure.**

1. **Difference between functional and Non-functional Testing:**

**Ans:**

* Functional testing verifies each function/feature of the software whereas Non Functional testing verifies non-functional aspects like performance, usability, reliability, etc.
* Functional testing can be done manually whereas Non Functional testing is hard to perform manually.
* Functional testing is based on customer’s requirements whereas Non Functional testing is based on customer’s expectations.
* Functional testing has a goal to validate software actions whereas Non Functional testing has a goal to validate the performance of the software.
* A Functional Testing example is to check the login functionality whereas Non-Functional testing example is to check the dashboard should load in 2 seconds.
* Functional describes what the product does whereas Non-Functional describes how the product works.
* Functional testing is performed before the non-functional testing.

1. **When to Stop testing?**

**Ans:**

1. When the all the requirements are tested, included example of Encapsulation
2. When there are no open issues (if open, an() =re not critical or high severity)
3. When all the entry and exit criteria are met
4. **Static and Dynamic Test Tools?**

**Ans:**

**Static Test Tools:** These tools do not involve actual input and output. Rather, they take a symbolic approach to testing, i.e. they do not test the actual execution of the software. These tools include the following.

1**) Flow analyzers**: They ensure consistency in data flow from input to output.

2) **Path tests:** They find unused code and code with contradictions.

3) **Coverage analyzers**: It ensures that all logic paths are tested.

4) **Interface analyzers**: It examines the effects of passing variables and data between modules.

**Dynamic Test Tools**: These tools test the software system with ‘live’ data. Dynamic test tools include the following

1**) Test driver**: It inputs data into a module-under-test (MUT).

2**) Test beds**: It simultaneously displays source code along with the program under execution.

3) **Emulators**: The response facilities are used to emulate parts of the system not yet developed.

4) **Mutation analyzers**: The errors are deliberately ‘fed’ into the code in order to test fault tolerance of the system.

1. **Boundary Value Analysis?**

**Ans:** Boundary testing is the process of testing between extreme ends or boundaries between partitions of the input values.

* So these extreme ends like Start- End, Lower- Upper, Maximum-Minimum, Just Inside-Just Outside values are called boundary values and the testing is called "boundary testing".
* The basic idea in boundary value testing is to select input variable values at their Minimum, Just above the minimum, A nominal value, Just below the maximum, Maximum.

Example: Pizza values 1 to 10 is considered valid boundary values 0,1,2,9,10,11

1. **Equivalence Partitioning?**

**Ans:** Equivalent Class Partitioning is a black box technique (code is not visible to tester) which can be applied to all levels of testing like unit, integration, system, etc. In this technique, you divide the set of test condition into a partition that can be considered the same.

Example: Following password field accepts minimum 6 characters and maximum 10 characters

0-5 and 6-10 and 11-14.

**AUTOMATION QUESTIONS**

1. **Difference between Selenium and UFT?**

**Ans:**

|  |  |
| --- | --- |
| **HP UFT (QTP)** | **Selenium** |
| It is commercial tool by Micro Focus and hence it requires a license and is expensive | Its an open source testing tool hence it does not require license and is free |
| It is used for testing client-server applications. It can test web- based as well as desktop applications | Using Selenium only web applications can be automated |
| QTP  tests can only be developed in[QTP](https://www.guru99.com/quick-test-professional-qtp-tutorial.html)IDE | Selenium has the option of using wide range of IDEs like Visual Studio, Eclipse, Netbeans |
| HP UFT only supports VB script | Selenium supports JAVA, .NET, Ruby, Perl, PHP, and many other programming languages |
| HP UFT comes with built in object repository.  Object repository development and maintenance is quite easy in HP ALM | Selenium does not  have such built in object repository, but object can be managed by using UI element user extension |

1. **Syntax for handling Frames in Selenium:**

**Ans:**

Drivername.findElements(By. tag Name (“Name of the frame”)). size ();

Drivername.switchTo(). frame (“name of the frame (or) Id of the frame”);

Note: If many frames are there we should use index number

1. **Locators in Selenium:**

**Ans:** Selenium Locators to find match the elements of a webpage page that it needs to interact with.

Selenium has 8 locators

1. Identifier
2. Id
3. Name
4. Link
5. DOM
6. Xpath
7. CSS
8. UI elements
9. **What is Xpath:**

Ans: XML path is a syntax or language for finding any element on the webpage using xml path expression.

Syntax: //tagname [@attribute = ‘value’].

1. **Types of Xpath:**

**Ans:** Two types of Xpaths**:**

1. Absolute Xpath: Direct way to find the element but the disadvantage is if there are any changes made in the path of element (or code) then Xpath will get failed

Example: Html/Body/Head/Div[1]/Div[1]/B

1. Relative Xpath:

Example: //\*[@class=’featured-box’]//\*[text()=’Testing’]

1. **Xpath axes methods**

**Ans:** These are used to find dynamic elements, which otherwise not possible to find by normal Xpath method.

Different Xpath axes methods are

1. Following
2. Ancestor
3. Child
4. Preceding
5. Sibling
6. Parent
7. Self
8. Descendent
9. **Selenium Components:**

**Ans:**

|  |  |  |
| --- | --- | --- |
| **Selenium IDE** | **Selenium RC** | **Selenium Web Driver** |
| Works only in Mozilla Firefox | Supports all browsers | Supports all browsers |
| Support Record & Play back | Doesn’t Support Record & Play back | Doesn’t Support Record & Play back |
| Doesn’t require any server to start before executing the test script | require RC server to start before executing the test script | Doesn’t require any server to start before executing the test script |
| It is a GUI plugin | It is a standalone java program which allows you to run HTML test suites | Actual core API which has binding in range of languages |
| Core engine is Java Script based | Core engine is Java Script based | Interacts natively with browser application |
| Not object oriented | API’s are less object oriented | API’s are entirely object oriented |
| Doesn’t Support moving of mouse cursors and listeners | Doesn’t Support moving of mouse cursors and listeners | Support moving of mouse cursors and listeners |
| Doesn’t support to test on IPhone/android application | Doesn’t support to test on IPhone/android application | It supports to test on IPhone/android application |

**Selenium Grid:**

It is a tool used together with selenium RC to run parallel tests across different Machines and different browsers all at the same time

**Basic terminology of selenium grid: -**

**Hub:** is a central point to the entire GRID Architecture which receives all requests. There is only one hub in the selenium grid. Hub distributes the test cases across each node.

**Node:** There can be multiple nodes in Grid each node communicates with the hub and performs teste assigned to it

1. **Where we use OOPs concept in selenium?**

**Ans:**

1. Abstraction: use of Locators (id, name, Xpath).
2. Interface: Web Driver itself is an interface for which reference variables are created
3. Polymorphism:

* Method overloading: - Time Stamps such as Seconds, minutes, hours, etc.
* Method overriding: - get and navigator methods

1. Encapsulation: all classes in a framework are an example of Encapsulation
2. **Excel sheet syntax:**

**Ans:** reading data from Excel Sheet

FIleInputStream E1=new **FIleInputStream** (path);

Workbook w=workbook. getWorkbook(“E1”); (Note: XSSFWorkbook :XLSX and HSSFWorkbook:XLS)

Sheet s = w. getSheet (“index (or) name of the sheet”);

Row count:s.getLastRowNum();

Cellcount:s.getRow(i).getLastCellNum();

Writing Data into Excel Sheet:

FileOutputStream outputStream = new FileOutputStream(file);

w. write(outputStream);

outputStream.close();

w.close()

1. **Actions?**

Ans: Actions are used to handle Keyboard and mouse Events

Syntax: Action a1=new Action (name of the driver);

Action mouseoverHome = a1.moveToElement(link\_home). build ();

mouseoverHome.perform();

1. **Drag and drop Syntax using Actions?**

**Ans:** Action a1=new Action (name of the driver);

A1.dragAndDrop(Source locator, Destination Locator).

1. **List few Exceptions u faced in selenium?**

**Ans:**

1. ConnectionClosedException: This Exception takes place when there is a disconnection in the driver.
2. ErrorHandlerUnknownServerException: used as a place holder if server returns an error without a stack trace.
3. InvalidArguementException: This is thrown if argument does not belong to expected type
4. NoSuchElementException: occurs if an element could not be found
5. TimeOutException: Thrown when there is not enough time for a command to be completed
6. **Syntax for screen shot in selenium:**

**Ans:** File screenshotFile = ((TakesScreenshot)driver). getScreenshotAs (OutputType.FILE);

FileUtils.copyFile(screenshotFile, new File("filename\_with\_path"));

1. **Difference between get() and navigate()?**

**Ans: get()**method is used to open an URL and it will wait till the whole page gets loaded.

***navigate.to()***method navigates to an URL and It will not wait till the whole page gets loaded. It maintains the browser history and cookies, so we **can use forward and backward button to navigate between the pages** during the coding of Testcase.

1. **Drop Down in Selenium?**

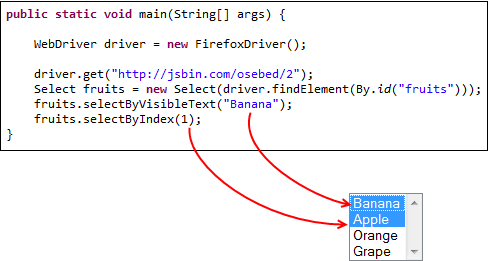
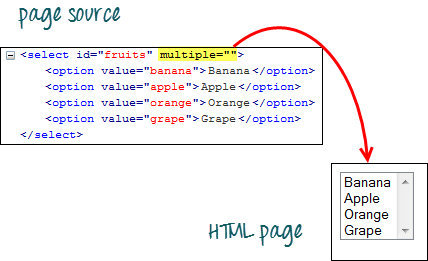
Ans: selectByVisibleText() and deselectByVisibleText()

selectByValue() and deselectByValue()

selectByIndex() and deselectByIndex()

isMultiple(): Returns TRUE if the drop-down element allows multiple selections at a time; FALSE if otherwise.

deselectAll(): for clearing all selections



1. **Implicit and explicitWait?**

Ans: implicit:

Driver.manage().timeouts().implicitlyWait(DELAY, TimeUnit.Seconds);

Explicit:

WebDriverWait wait =new WebDriverWait(driver,20);

**UFT Questions:**

1. **What are the types object Repositories in Micro Focus UFT?**

**Ans:** QTP Supports 2 types of Object Repository

1) Shared Object Repository (also called Global)

2) Per-Action Object Repository, (also called Local)

1. **Can we call QTP test from another test using scripting? Suppose there are 4 tests and I want to call these tests in a main script. Is this possible in QTP?**

**Ans:** yes.  You can call 4 or even more scripts in your tests. For this, first you will need to make the Actions in the corresponding scripts re-usable. Then from the destination script you can make calls to these re-usable actions.

1. **How many types of recording modes in QTP? Which will be used when?**

**Ans:** QTP supports 3 types of recording modes

1. Normal mode also called Contextual

2. Low-level recording mode

3.Analog mode

1. **What is Reporter.ReportEvent?**

**Ans:** Reporter.ReportEvent is standard method provided by QTP to send custom messages to the test results window.

Syntax

Reporter.ReportEvent EventStatus, ReportStepName, Details [, ImageFilePath]

1. **What is GetRoProperty?**

**Ans:** GetRoProperty is a standard method provided by QTP to fetch property values of a run -time object.

1. **Give the syntax to import/export xls into QTP?**

**Ans:** DataTable.ImportSheet "..\..\TestData\Input.xls",1, dtGlobalSheet

DataTable.ExportSheet "..\..\Results\Output.xls","Global"

1. **What is descriptive programming in UFT means?**

**Ans:** Descriptive programming includes property name and property value.  Whenever UFT is facing difficulty in identifying objects from object repository, and instead the object is directly identified from the script is known as descriptive programming.

**Short description about ECM:**

As part of Enterprise content management we will be handling communications that will be sent to the customer.

For this we will be using CCM (Customer communication management) tools such as Doc1 and Engage One through which we will create relevant, accurate, and highly personalized content, delivered in both paged (e.g., Print or PDF) and digital formats (email, HTML, XML, Bank mail, Push Notifications, Alerts).

Once we develop the communications using Doc1 Designer we will deploy them to Engage One repository. Based on the customer preferences we will configure the communications in Admin.

**JAVA**