Real Time Software Testing

Interview question & answers

1. Define Manual Testing?
2. Why is testing required?
3. What is static Testing?
4. What is dynamic testing?
5. What is test closure?
6. How will you determine when to stop testing?
7. Explain the STLC – Software Testing life cycle.
8. What are some of the bug or defect management tools?
9. Give an example of Low Priority-Low severity, Low Priority-High severity, High Priority-Low severity, High Priority-High severity defects.
10. what is webdriver
11. what is Actions class?
12. Difference between find element & find elements
13. what is return type of find element & elements
14. How to retrieve text from alert
15. What are the wait statement in selenium
16. Explain types of Assertions
17. Difference between getwindowHandle & handles
18. Difference between fluent wait & Explicitwait
19. What Is Cucumber?
20. What Does Feature File Consist Of ?
21. Explain the purpose of keywords that are used for writing a scenario in Cucumber.
22. Difference between Scenario & Scenario outline in feature file
23. Difference between throw & throws
24. Difference between final & finally
25. what is mean by runtime & compile time ploy
26. Difference between abstract & interface
27. what is mean by encapsulation
28. Explain java 8 new features
29. Write a program to count duplicate characters in string
30. What are the different types of object creation?

1. **Define Manual Testing?**

Answer: Software testing is a validation process that makes sure a system works as per the business requirements. It evaluates and qualifies a system on various aspects such as accuracy, completeness, usability, efficiency, and more.

2. **Why is testing required?**

We need software testing for the following reasons-
1. Testing provides an assurance to the stakeholders that the product works as intended.
2. Avoidable defects leaked to the end-user/customer without proper testing adds a bad reputation to the development company.
3. Defects detected earlier phase of SDLC results in lesser cost and resource utilization of correction.
4. Saves development time by detecting issues in an earlier phase of development.
5. The testing team adds another dimension to the software development by providing a different viewpoint to the product development process.

3. **What is static Testing?**

Ans. Static testing is a kind of testing for reviewing the work products or documentation that are being created throughout the entire project. It allows reviewing the specifications, business requirements, documentation, processes and functional requirements in the initial phase of testing.
So that the testers involved in it can understand the requirements in more detail before starting the testing lifecycle which intends to help in delivering the quality product.

4. **What is dynamic testing?**

Ans. Testing performed by executing or running the application under test either manually or using automation.

5. **What is test closure?**

Answer: It is a document that details the tests conducted during the entire SDLC, the analysis of the bugs and errors found and corrected, the density of defects, etc. It is a memo that indicates the formal completion of the testing procedure.

6. **How will you determine when to stop testing?**

Answer: Well, the most straightforward answer would be that when all the defects are not found in software, we can stop testing. However, it is not possible to have perfect software free of bugs. We can determine the exit criteria for testing based on the deadlines, budget, and the extent of testing performed. Usually, testers can find most of the major and critical bugs during the first and second weeks of testing. After the third and fourth weeks, even minor and cosmetic defects are taken care of, and the application moves into the regression testing phase. Once regression is completed, we can be assured that 99% of test scenarios have been covered, and software is ready to be rolled out.

7. **Explain the STLC – Software Testing life cycle.**

Ans. The software testing life cycle refers to all the activities performed during testing of a software product. The phases include-

Requirement analysis and validation – In this phase, the requirements documents are analyzed and validated and the scope of testing is defined.

Test planning – In this phase test plan strategy is defined, estimation of test effort is defined along with automation strategy and tool selection is done.

Test Design and Analysis – In this phase test cases are designed, test data is prepared and automation scripts are implemented.

Test environment setup – A test environment closely simulating the real-world environment is prepared.

Test execution – The test cases are prepared, bugs are reported and retested once resolved.

Test closure and reporting – A test closure report is prepared to have the final test results summary, learning, and test metrics.

8. **What are some of the bug or defect management tools?**

Ans. Some of the most widely used Defect Management tools are – Jira, Bugzilla, Redmine, Mantis, Quality Center, etc.

9. **Give an example of Low Priority-Low severity, Low Priority-High severity, High Priority-Low severity, High Priority-High severity defects.**

Ans. Below are the examples for different combinations of priority and severity-

Low priority-Low severity – A spelling mistake in a page not frequently navigated by users.

Low priority-High severity – Application crashing in some very corner case.

High priority-Low severity – Slight change in logo color or spelling mistake in the company name.

High priority-High severity – Issue with login functionality.

10. **what is webdriver**

Answer: WebDriver is defined as one interface is Selenium which can be used to perform below steps:

- To automate webbased applications

- Webdriver supports multiple programming language like java, c#, python, php, Perl, ruby.
- Webdriver supports for multiple programming language

11. what is Actions class?

Answer: Actions class is an ability provided by Selenium for handling keyboard and mouse events. In Selenium WebDriver, handling these events includes operations such as drag and drop, clicking on multiple elements with the control key, among others. These operations are performed using the advanced user interactions API

12. Difference between find element & find elements

Answer: find element:

A command used to uniquely identify a web element within the web page.
Returns the first matching web element if multiple web elements are discovered by the locator-it Throws **NoSuchElementException** if the element is not found
it Detects a unique web element

findElements:

A command used to identify a list of web elements within the web page. Returns a list of multiple matching web elements, Returns an empty list if no matching element is found. It Returns a collection of matching elements

13. what is return type of find element & elements

Answer: **findElement(By by)** method finds and returns the first matching element within the current context by the given mechanism.
findElements(By by) finds and returns all matching elements within the current context by the given mechanism.

14. How to retrieve text from alert

Answer: driver.switchTo().alert().getText(); **[or]**
There is a method in Alert interface which gives you the text of the alert box message. As below:
Alert alert = driver.switchTo().alert();
alert.getText();

15. What are the wait statement in selenium?

Answers: There is 2 different way to handle waits:
1. Implicit Waits
2. Explicit Waits

Implicit Waits: WebDriver waits for an element if they are not immediately available. So, WebDriver does not throw NoSuchElementException immediately. This is known as implicitlyWait(). This can be achieved using:
driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);
Note: If the DOM is get loaded within the wait time it should not wait for the remaining time it go for the next step,
for an example: Here we wait for 20 seconds, after that it gives NoSuchElementException. If the element present in 5 second then it should not wait for another 15 seconds.
Disadvantages:
(i) In any case, it blindly wait for given seconds.
(ii) Once set, the implicit wait is set for the life of the WebDriver object instance.
Explicit Waits
A. **Thread.sleep()** : This is to wait the running program for sometime, this can be done using: Ex.- Thread.sleep(3600).
Disadvantages:
(i) In this case, it again blindly wait for given seconds. It is not a better way because if the the element is present within this given wait time, program does not move further until the wait time finished.
(ii) Some computers are slow, an element may not show up in a slow computer within the given wait time.
3.It is sleep time for script, not good way to use in script as it sleeps without condition.
Expected Condition
An explicit wait is code you define to wait for a certain condition to occur before proceeding further in the code. There are some convenient methods provided that help you to write code that will wait only as long as required. WebDriverWait in combination with ExpectedCondition is one way this can be accomplished.
Ex.- WebDriverWait wait = new WebDriverWait(driver, 15);
WebElement element =
wait.until(ExpectedConditions.presenceOfElementLocated(By.id()));
This waits up to 15 seconds before throwing a TimeoutException or if it finds the element will return it in 0 – 15 seconds. WebDriverWait by default calls the ExpectedCondition every 500 milliseconds until it returns successfully. A successful return is for ExpectedCondition type is Boolean return true or not null return value for all other ExpectedCondition types.

There are some common conditions that are frequently come across when automating web browsers. Listed below are Implementations of each. Java happens to have convienence methods so you don’t have to code an ExpectedCondition class yourself or create your own utility package for them.
– Element is Clickable – it is Displayed and Enabled.
Ex.- WebDriverWait wait = new WebDriverWait(driver, 15);
    WebElement element = wait.until
(ExpectedConditions.elementToBeClickable(By.id()));
3. **Fluent Wait Command:** Each FluentWait instance defines the maximum amount of time to wait for a condition, as well as the frequency with which to check the condition. Furthermore, the user may configure the wait to ignore specific types of exceptions whilst waiting, such as ‘NoSuchElementExceptions’ when searching for an element on the page.
Ex.- Wait wait = new FluentWait(driver)
.withTimeout(30, SECONDS)
.pollingEvery(5, SECONDS)
.ignoring(NoSuchElementException.class);
wait.until(ExpectedConditions.elementToBeClickable(By.id()));

16.Explain types of Assertions

Answers: **Hard Assertion :** A hard assert throw AssertException immediately after a test fails and the test is marked as failed.
assertEquals,assertNotEquals,assertTrue,assertFalse
assertNull,assertNotNull
Soft Assertion : now the failed assertions will be reported in the testNg report and not making the test to abort anywhere.
we have to include it’s corresponding class (as SoftAssert()) in the script.

17.Difference between getwindowHandle & handles

Ans: getWindowHandle() returns the window handle of currently focused window/tab. **getWindowHandles**() returns all windows/tabs handles launched/opened by same driver instance including all parent and child window.
Return type of getWindowHandle() is String while return type of getWindowHandles() is Set<String>. The return type is Set as window handle is always unique.
In chrome and Firefox , Each tab in a window will have unique window handles. So getWindowHandles() will return handles for all tabs of a window. For example:- If there are four tabs in a window is opened, getWindowHandles()

method will give “four” for chrome and firefox browsers. I am not sure about IE and EDGE. I will bring this in a new post.
getWindowHandles() internally uses LinkedHashSet. So whatever Set it returns, it will give window handles in order it is opened.

18. Difference between fluent wait & Explicitwait

Ans: WebDriverWait is applied on certain element with defined expected condition and time. This wait is only applied to the specified element. This wait can also throw exception when element is not found. Webdriver doesn't perform pooling for this wait scenario.

WebDriverWait wait = new WebDriverWait (driver, 20);
wait.until(ExpectedConditions.VisibilityofElementLocated(By.xpath("//button[@value='Save Changes']")));

Fluent wait is another type of Explicit wait and you can define polling and ignore the exception to continue with script execution in case element is not found. Here, we can set pooling time, which isn't possible in Webdriver wait.

new FluentWait<WebDriver>(driver).withTimeout(30, TimeUnit.SECONDS).pollin"

19. What Is Cucumber?

Answer: Cucumber is a Behavior Driven Development (BDD) tool. Cucumber is a tool that executes plain-text functional descriptions as automated tests. The language that Cucumber understands is called Gherkin.
In BDD, users (business analysts, product owners) first write scenarios or acceptance tests that describes the behavior of the system from the customer’s perspective, for review and sign-off by the product owners before developers write their codes.

20. What Does Feature File Consist Of ?

Answer: Feature file in cucumber consist of parameters or conditions required for executing code, they are:

·Feature
·Scenario
·Scenario Outline
·Given
·When
·Then

21. **Explain the purpose of keywords that are used for writing a scenario in Cucumber.**

Answer: “Given” keyword is used to specify a precondition for the scenario. When a keyword is used to specify an operation to be performed. The keyword is used to specify the expected result of a performed action. “And” keyword is used to join one or more statements together into a single statement.

22. **Difference between Scenario & Scenario outline in feature file**

Answer: In a single execution, Scenario is executed only once.

```
Scenario: Eat 5 out of 12
  Given there are 12 cucumbers
  When I eat 5 cucumbers
  Then I should have 7 cucumbers

Scenario: Eat 5 out of 20
  Given there are 20 cucumbers
  When I eat 5 cucumbers
  Then I should have 15 cucumbers
```

whereas **Scenario outline** (For similar data trace) can be executed multiple times depending upon the data provided as Example.

```
Scenario Outline: Eating
  Given there are <start> cucumbers
  When I eat <eat> cucumbers
  Then I should have <left> cucumbers
Examples:
    | start | eat | left |
    |  12   |  5  |  7   |
    |  20   |  5  |  15  |
```

23. **Difference between throw & throws**

Ans: Throw is a keyword which is used to throw an exception explicitly in the program inside a function or inside a block of code.

throw only single exception at a time i.e we cannot throw multiple exception with throw keyword. Syntax wise throw keyword is followed by the instance variable. keyword is used within the method.

Throws is a keyword used in the method signature used to declare an exception which might get thrown by the function while executing the code.

we can declare multiple exceptions with throws, On other hand syntax wise throws keyword is followed by exception class names. used with the method signature."

24.Difference between final , finally & finalize

Ans: **Final** is used to apply restrictions on class, method and variable. Final class can't be inherited, final method can't be overridden and final variable value can't be changed. Final is a keyword.
Finally is used to place important code, it will be executed whether exception is handled or not. Finally is a block.
Finalize is used to perform clean up processing just before object is garbage collected. Finalize is a method.

25.what is mean by runtime polymorphism & compile time polymorphism

Ans: There are two types of polymorphism in java:
1) Static Polymorphism also known as compile time polymorphism
2) Dynamic Polymorphism also known as runtime polymorphism
Compile time Polymorphism (or Static polymorphism)
Polymorphism that is resolved during compiler time is known as static polymorphism. Method overloading is an example of compile time polymorphism.
Method Overloading: This allows us to have more than one method having the same name, if the parameters of methods are different in number, sequence and data types of parameters.
Runtime Polymorphism (or Dynamic polymorphism)
It is also known as Dynamic Method Dispatch. Dynamic polymorphism is a process in which a call to an overridden method is resolved at runtime, thats why it is called runtime polymorphism.

26.Difference between abstract & interface

Ans: **interface**:Interface can have only abstract methods. Since Java 8, it can have default and static methods also.
Variables declared in a Java interface is by default final
Members of a Java interface are public by default
interface should be implemented using keyword “implements”
An interface can extend another Java interface only
Abstract : Abstract class can have abstract and non-abstract methods.
An abstract class may contain non-final variables.
A Java abstract class can have the usual flavors of class members like private, protected, etc..
A Java abstract class should be extended using keyword “extends”. an abstract class can extend another Java class and implement multiple Java interfaces."

27.what is mean by encapsulation

Ans: Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as data hiding.

28.Explain java 8 new features

Ans :

- foreach() method in Iterable interface
- default and static methods in Interfaces
- Functional Interfaces and Lambda Expressions
- Java Stream API for Bulk Data Operations on Collections
- Java Time API
- Collection API improvements
- Concurrency API improvements
- Java IO improvements
- Miscellaneous Core API improvements"

29.Write a program to count duplicate characters in string

Ans: Steps to write an program:

1. Define a string.
2. Two loops will be used to find the duplicate characters. Outer loop will be used to select a character and initialize variable count by 1.
3. Inner loop will compare the selected character with rest of the characters present in the string.
4. If a match found, it increases the count by 1 and set the duplicates of selected character by '0' to mark them as visited.
5. After inner loop, if count of character is greater than 1, then it has duplicates in the string.

```
public class DuplicateCharacters {
    public static void main(String[] args) {
        String string1 = "Great responsibility";
        int count;
        //Converts given string into character array
        char string[] = string1.toCharArray();
        System.out.println("Duplicate characters in a given string: ");
        //Counts each character present in the string
        for(int i = 0; i <string.length; i++) {
            count = 1;
            for(int j = i+1; j <string.length; j++) {
                if(string[i] == string[j] && string[i] != ' ') {
                    count++;
                    //Set string[j] to 0 to avoid printing visited character
                    string[j] = '0';
                }
            }
            //A character is considered as duplicate if count is greater than 1
            if(count > 1 && string[i] != '0')
                System.out.println(string[i]);
        }
    }
}
```

30.What are the different types of object creation?

Ans: **Using the new keyword**-The constructor gets called
Employee emp1 = new Employee();
Using newInstance() method of Class class-The constructor gets called
Employee emp2 = Employee.class.newInstance();
Using newInstance() method of Constructor class-The constructor gets called
Constructor<Employee> constructor = Employee.class.getConstructor();
Employee emp3 = constructor.newInstance();
Using clone() method-No constructor call
Employee emp4 = (Employee) emp3.clone();
Using deserialization-No constructor call
ObjectInputStream in = new ObjectInputStream(new
FileInputStream(""data.obj""));
Employee emp5 = (Employee) in.readObject();"