JAVA

1. Explain Static keyword

   Answer: **static keyword** is mainly used for memory management. It can be used with variables, methods, blocks and nested classes. It is a **keyword** which is used to share the same variable or method of a given class. Basically, **static** is used for a constant variable or a method that is same for every instance of a class.

2. Explain Final keyword

   Answer: The **final keyword** is used in several contexts to **define** an entity that can only be assigned once. Once a **final** variable has been assigned, it always contains the same value.

3. Explain this keyword.

   Answer: This **keyword** refers to the current object in a method or constructor. The most common use of this **keyword** is to eliminate the confusion between class attributes and parameters with the same name (because a class attribute is shadowed by a method or constructor parameter).

4. Explain super keyword.

   Answer: The **super keyword** refers to superclass (parent) objects. It is used to call superclass methods, and to access the superclass constructor. The most common use of the **super keyword** is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

5. Difference between this and super keyword.

   Answer: This vs **super keyword**. this **keyword** points to a reference of the current class, while **super keyword** points to a reference of the parent class. this can be used to access variables and methods of the current class, and **super** can be used to access variables and methods of the parent class from the subclass.

6. Difference between final and abstract class

   Answer:

   Final Class: A class which is declared with the “Final” keyword is known as the final class. The final keyword is used to finalize the implementations of the classes, the methods and the variables used in this class. The main purpose of using a final class is to prevent the class from being inherited (i.e.) if a class is marked as final, then no other class can inherit any properties or methods from the final class. If the final class is extended, Java gives a compile-time error.

   Abstract Class: A class that is declared using the “abstract” keyword is known as an abstract class. The main idea behind an abstract class is to implement the concept of Abstraction. An abstract class can have both abstract methods (methods without body) as well as the concrete methods (regular methods with the body). However, a normal class (non-abstract class) cannot have abstract methods.

7. Explain OOPS concept

   Answer: **Object-Oriented Programming** (**OOPs**) is a Programming **concept** that works on the principles of abstraction, encapsulation, inheritance, and polymorphism. The basic **concept** of **OOPs** is to create objects, re-use them throughout the program, and manipulate these objects to get results.

8. Explain the OOPS concept implemented in your project

   Answer:

   ABSTRACTION

   In Page Object Model design pattern, we write locators (such as id, name, xpath etc.,) in a Page Class. We utilize these locators in tests but we can’t see these locators in the tests. Literally we hide the locators from the tests.

Abstraction is the methodology of hiding the implementation of internal details and showing the functionality to the users.

INTERFACE

Basic statement we all know in Selenium is WebDriver driver = new FirefoxDriver ();

WebDriver itself is an Interface. So based on the above statement WebDriver driver = new FirefoxDriver (); we are initializing Firefox browser using Selenium WebDriver. It means we are creating a reference variable (driver) of the interface (WebDriver) and creating an Object. Here WebDriver is an Interface as mentioned earlier and FirefoxDriver is a class.

An interface in Java looks similar to a class but both the interface and class are two different concepts. An interface can have methods and variables just like the class but the methods declared in interface are by default abstract. We can achieve 100% abstraction and multiple inheritance in Java with Interface.

INHERITANCE

We create a Base Class in the Framework to initialize WebDriver interface, WebDriver waits, Property files, Excels, etc., in the Base Class.
We extend the Base Class in other classes such as Tests and Utility Class. Extending one class into other class is known as Inheritance.

POLYMORPHISM

Combination of overloading and overriding is known as Polymorphism. We will see both overloading and overriding below.
Polymorphism allows us to perform a task in multiple ways.

METHOD OVERLOADING

We use implicit wait in Selenium. Implicit wait is an example of overloading. In Implicit wait we use different time stamps such as SECONDS, MINUTES, HOURS etc., A class having multiple methods with same name but different parameters is called Method Overloading.

METHOD OVERRIDING

We use a method which was already implemented in another class by changing its parameters. To understand this, you need to understand Overriding in Java. Declaring a method in child class which is already present in the parent class is called Method Overriding. Examples are get and navigate methods of different drivers in Selenium.

ENCAPSULATION

All the classes in a framework are an example of Encapsulation. In POM classes, we declare the data members using @FindBy and initialization of data members will be done using Constructor to utilize those in methods.
Encapsulation is a mechanism of binding code and data together in a single unit.

9. Explain the difference between Array and Collection

Answer:

ARRAYS	COLLECTION
s are fixed in size that is once we create an array we ased or decreased based on our requirement.	ction are growable in nature that is based on our rement. We can increase or decrease of size.
respect to memory Arrays are not recommended to	respect to memory collection are recommended to

respect to performance Arrays are recommended to	respect to performance collection are not recomme e.
s can hold only homogeneous data types elements.	ction can hold both homogeneous and heterogeneo ents.
e is no underlying data structure for arrays and hence -made method support is not available.	collection class is implemented based on some star structure and hence for every requirement ready-m od support is available being a performance. we can method directly and We are not responsible to imp methods.
s can hold both object and primitive.	ction can hold only object types but primitive.

10. Explain the Collection architecture

Answer:

The **Collection** in Java is a framework that provides an **architecture** to store and manipulate the group of objects. Java **Collection** framework provides many interfaces (Set, List, Queue, Deque) and classes (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet).

11. Difference between list and set

Answer:

List is an **ordered sequence** of elements whereas Set is a **distinct list** of elements which is unordered.

List is a type of ordered collection that maintains the elements in **insertion order** while Set is a type of unordered collection so elements are not maintained any order.

List allows duplicates while Set doesn't allow **duplicate elements**. All the elements of a Set should be unique if you try to insert the duplicate element in Set it would replace the existing value.

List permits any number of **null values** in its collection while Set permits only one null value in its collection.

New methods are defined inside **List interface**. But, no new methods are defined inside Set interface, so we have to use Collection interface methods only with **Set subclasses**.

List can be inserted in in both **forward** direction and **backward** direction using Listiterator while Set can be traversed only in forward direction with the help of iterator.

12. Explain Multithreading

Answer: Multithreading is the ability of a central processing unit (CPU) (or a single core in a multi-core processor) to provide **multiple threads** of execution concurrently, supported by the operating system. This approach differs from multiprocessing.

13. Explain Synchronization

Answer: Synchronization is the precise coordination of multiple events or mechanical devices. In computing, it refers to the coordination of hardware devices, such that the data they contain or provide is made to be identical. The **synchronization** is usually done in a short time frame.

14. Explain Serialization

Answer: Serialization is the process of converting an object into a stream of bytes to store the object or transmit it to memory, a database, or a file. Its main purpose is to save the state of an object in order to be able to recreate it when needed. The reverse process is called deserialization.

15. Explain final, finally and finalize

Answer:

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

16. What is Thread in Java

Answer: Every **java** application has at least one **thread** – main **thread**. But from application point of view – main is the first **java thread** and we can create multiple **threads** from it. ... **Multithreading** refers to two or more **threads** executing concurrently in a single program.

17. Explain String

Answer: A **string** is a data type used in programming, such as an integer and floating point unit, but is used to represent text rather than numbers. It is comprised of a set of characters that can also contain spaces and numbers.

18. Difference between StringBuffer and StringBuilder.

Answer: String **is** immutable whereas **StringBuffer and StringBuilder** are mutable classes. **StringBuffer is** thread-safe and synchronized whereas **StringBuilder is** not. That's why **StringBuilder is** faster than **StringBuffer**. String concatenation operator (+) internally uses **StringBuffer** or **StringBuilder** class.

19. Exception Handling

Answer: An **exception** is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions. When an error occurs within a method, the method creates an object and hands it off to the runtime system. This block of code is called an **exception handler**.

20. Difference between throw and throws

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

21. Explain Reflection in java

Answer: **Reflection** is an API which is used to examine or modify the behaviour of methods, classes, interfaces at runtime. The required classes for **reflection** are provided under **java**. lang. **reflect** package. ... Through **reflection** we can invoke methods at runtime irrespective of the access specifier used with them.

22. What is Garbage collector

Answer: **Java** applications obtain objects in memory as needed. It is the task of **garbage collection** (**GC**) in the **Java** virtual machine (JVM) to automatically determine what memory is no longer being used by a **Java** application and to recycle this memory for other uses.

23. Local, Instance and Static variable

Answer:

Local Variables: A variable defined within a block or method or constructor is called local variable.

- These variable are created when the block in entered or the function is called and destroyed after exiting from the block or when the call returns from the function.
- The scope of these variables exists only within the block in which the variable is declared. i.e. we can access these variable only within that block.
- Initialization of Local Variable is Mandatory.

Instance Variables: Instance variables are non-static variables and are declared in a class outside any method, constructor or block.

- As instance variables are declared in a class, these variables are created when an object of the class is created and destroyed when the object is destroyed.
- Unlike local variables, we may use access specifiers for instance variables. If we do not specify any access specifier, then the default access specifier will be used.
- Initialization of Instance Variable is not Mandatory. Its default value is 0
- Instance Variable can be accessed only by creating objects.

Static Variables: Static variables are also known as Class variables.

These variables are declared similarly as instance variables; the difference is that static variables are declared using the static keyword within a class outside any method constructor or block.

Unlike instance variables, we can only have one copy of a static variable per class irrespective of how many objects we create.

Static variables are created at the start of program execution and destroyed automatically when execution ends.

Initialization of Static Variable is not Mandatory. Its default value is 0

If we access the static variable like Instance variable (through an object), the compiler will show the warning message and it won't halt the program. The compiler will replace the object name to class name automatically.

24. Explain Java Generics

Answer: The **Java Generics** allows us to create a single class, interface, and method that can be used with different types of data (objects). This helps us to reuse our code. Note: **Generics** does not work with primitive types (int , float , char , etc).

25. Explain about Constructor

Answer: A constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

It is a special type of method which is used to initialize the object.

Every time an object is created using the new() keyword, at least one constructor is called.

It calls a default constructor if there is no constructor available in the class. In such case, Java compiler provides a default constructor by default.

26. Difference between Constructor and Methods

Answer: Constructor is used to initialize an object whereas **method is** used to exhibits functionality of an object. **Constructors** are invoked implicitly whereas **methods** are invoked explicitly. **Constructor** does not return any value where the **method** may/may not return a value.

27. Explain about Access Modifier

Answer: Access modifiers (or **access** specifiers) are keywords in object-oriented languages that set the accessibility of classes, methods, and other members. ... When the class is declared as public, it is accessible to other classes **defined** in the same package as well as those **defined** in other packages.

28. Explain about Interface

Answer: Like a class, an interface can have methods and variables, but the methods declared in an interface are by default abstract (only method signature, no body).

- Interfaces specify what a class must do and not how. It is the blueprint of the class.
- An Interface is about capabilities like a Player may be an interface and any class implementing Player must be able to (or must implement) move (). So it specifies a set of methods that the class has to implement.
- If a class implements an interface and does not provide method bodies for all functions specified in the interface, then the class must be declared abstract.

29. What is tagged interface

Answer: An **interface** with no methods in it is referred to as a **tagging interface**. A class that implements a **tagging interface** does not need to define any methods (since the **interface** does not have any), but the class becomes an **interface** type through polymorphism

30. Difference between Interface and Abstract class

Answer: Abstraction: Hiding the internal implementation of the feature and only showing the functionality to the users. i.e. what it works (showing), how it works (hiding). Both abstract class and interface are used for abstraction.

Abstract class vs Interface

1. **Type of methods:** Interface can have only abstract methods. Abstract class can have abstract and non-abstract methods. From Java 8, it can have default and static methods also.
2. **Final Variables:** Variables declared in a Java interface are by default final. An abstract class may contain non-final variables.
3. **Type of variables:** Abstract class can have final, non-final, static and non-static variables. Interface has only static and final variables.
4. **Implementation:** Abstract class can provide the implementation of interface. Interface can't provide the implementation of abstract class.
5. **Inheritance vs Abstraction:** A Java interface can be implemented using keyword "implements" and abstract class can be extended using keyword "extends".
6. **Multiple implementation:** An interface can extend another Java interface only, an abstract class can extend another Java class and implement multiple Java interfaces.
7. **Accessibility of Data Members:** Members of a Java interface are public by default. A Java abstract class can have class members like private, protected, etc.

31. Difference between Method Overriding and Method Overloading

Answer: Overloading occurs when two or more **methods** in one class have the same **method** name but different parameters. **Overriding** means having two **methods** with the same **method** name and parameters (i.e., **method** signature). One of the **methods is in the** parent class and the other **is in the** child class.

32. Can we overload java main method?

Answer: Yes, **we can overload** the **main method** in **Java**, but When **we** execute the class JVM starts execution with public static void **main**(String [] args) **method.**

33. Can we override java main method?

Answer: No, **we** cannot **override main method** of **java** because a static **method** cannot **be overridden**. The static **method** in **java** is associated with class whereas the non-static **method** is associated with an object. ... Therefore, it is not possible to **override** the **main method** in **java**.

34. Can we override static method?
Answer: No, **we** cannot **override static methods** because **method overriding** is based on dynamic binding at runtime and the **static methods** are bonded using **static** binding at compile time. ... If **we** call a **static method** by using the child **class** object, the **static method** of the child **class** will **be** called.

35. Explain about Run time polymorphism
Answer: Run time polymorphism where at run time we came to know which method is going to invoke, it can be achieved through dynamic binding, Inheritance is involved, Method overriding is an example of runtime polymorphism.

36. Explain about Compile time polymorphism
Answer: Compile time polymorphism means binding is occurring at compile time, It can be achieved through static binding, Inheritance is not involved, Method overloading is an example of compile time polymorphism.

37. Difference between Parent objp = new Child (); | Child objc = new Child ();
Answer: Object 'objc' has access to child's properties as well as its parent class properties. Hence accessing objc.name and objc.age was possible. By using child reference we can call both parent and child class (state)Fields and (behaviour)methods.

Parent objp = new Child ();
It is called as up-casting. Object 'objp' has limited access. It can only access parent class properties, because p is a reference to parent class. Hence accessing 'objp.age' was not possible even though objp is a child object. So by using parent reference we can call only methods available in parent class and child specific methods are not accessible.
It is nothing but polymorphism. If we don't know exact runtime type of object, then we should use this approach(polymorphism).

38. Difference between == operator and. equals () method
Answer: **==** checks if both objects point to the same memory location whereas. equals **()** evaluates to the **comparison** of values **in the** objects.

39. Difference between length variable and length () method
Answer: **length () method** returns the number **of** characters presents **in the** string. The **length variable is** applicable to array but not **for** string objects whereas the **length () method is** applicable **for** string objects but not **for** arrays. To directly accesses a field member **of** array we can use.

40. Explain Daemon Thread
Answer: **Daemon thread** is a low priority **thread** that runs in background to perform tasks such as garbage collection. Properties: They cannot prevent the JVM from exiting when all the user **threads** finish their execution. If JVM finds running **daemon thread**, it terminates the **thread** and after that shutdown itself.

41. What is toString () method
Answer: A **toString ()** is an in-built **method** in Java that returns the value given to it in string format. Hence, any object that this **method** is applied on, will then be returned as a string object.

42. Explain java main method

Answer: The **main** () **method** is the entry point into the application. The signature of the **method** is always: public static void **main** (String [] args) Command-line arguments are passed through the args parameter, which is an array of String s.

43. Explain System.out.println();

Answer: println () is used to print an argument that is passed to it. The statement can be broken into 3 parts which can be understood separately as: **System**: It is a final class **defined** in the java. **out**: This is an instance of PrintStream type, which is a public and static member field of the **System** class.

44. How will you create object for singleton class?

Answer: You **can**not **make objects of singleton class** outside the **class** because the constructor is private. Constructor instantiate the **object**. if the constructor itself is private then the **object** will never be instantiated outside the **class** rather you **can** use getInstance method.

45. JDBC connectivity

Answer: The **JDBC Connection** class, java. sql. **Connection**, represents a database **connection** to a relational database. Before you can read or write data from and to a database via **JDBC**, you need to open a **connection** to the database.

Code :

```
      public class DataBaseConnectionExample {
public static void main (String [] args) throws SQLException,
ClassNotFoundException {
                Class.forName("com.mysql.jdbc.Driver");
Connection connection =
DriverManager.getConnection("jdbc:mysql://root@127.0.0.1/demo");
                Statement statement = connection.createStatement();
ResultSet result = statement.executeQuery("Select * from student");

                        while(result.next()) {
                              System.out.println(result.getString(2));
                        }
          }
}
```

46. what is the return type of constructor?

Answer: No, **constructor** does not have any **return type** in Java. ... It does not have a **return type** and its name is same as the class name. Mostly it is used to instantiate the instance variables of a class. If the programmer doesn't **write** a **constructor** the compiler writes a **constructor** on his behalf.

47. Enum Keyword?

Answer: The enum keyword declares an enumerated (unchangeable) type.
An enum is a special "class" that represents a group of constants (unchangeable variables, like final variables).

To create an enum, use the enum keyword (instead of class or interface), and separate the constants with a comma.

Selenium

1. Explain the Selenium components

   Answer: Selenium is an automation testing tool used to test web-based applications. Selenium is not a single tool but a suite of tools. There are four components of Selenium – Selenium IDE, RC, WebDriver, and Grid. Last two being the most famous one.

2. Explain the Selenium WebDriver Architecture

   Answer: Selenium WebDriver API enables interaction between browsers and browser drivers. This architecture consists of four layers namely the Selenium Client Library, JSON Wire Protocol, Browser Drivers and Browsers.

- Selenium Client Library consists of languages like Java, Ruby, Python, C# and so on. After the test cases are triggered, entire Selenium code will be converted to Json format.
- JSON stands for JavaScript Object Notation. It takes up the task of transferring information from the server to the client. JSON Wire Protocol is primarily responsible for transfer of data between HTTP servers. Generated Json is made available to browser drivers through http Protocol.
- Each browser has a specific browser driver. Browser drivers interact with its respective browsers and execute the commands by interpreting Json which they received from the browser. As soon as the browser driver gets any instructions, they run them on the browser. Then the response is given back in the form of HTTP response.

3. Explain the locators in Selenium

   Answer: Locators in Selenium are one of the most powerful commands
   ID, Name, Class, Xpath, LinkText, Partial LinkText, Css Selector, Tag Name.

4. What will you prefer Xpath or ID

   Answer: By.**ID** () is the faster technique because at its root, the call goes down to document. getElementById (), which is optimized by most browsers. But, finding elements using **XPath** is better for locating elements having complex selectors, and is no doubt the most flexible selection strategy.

5. Explain css selector

   Answer: CSS selectors are used *to select the content you want to style*. Selectors are the part of CSS rule set. CSS selectors select HTML elements according to its id, class, type, attribute etc.

   There are several different types of selectors in CSS.

1. CSS Element Selector
2. CSS Id Selector
3. CSS Class Selector
4. CSS Universal Selector
5. CSS Group Selector

6. When Using findElement and findElements but element no found, what exception will throw.

   Answer:
   ByfindElement - Throws **NoSuchElementException** if the element is not found
   ByfindElements - Returns an empty list if no matching element is found

7. WebDriver is class or interface

   Answer: Selenium WebDriver is an **interface** that defines a set of methods. However, implementation is provided by the browser specific **classes**. Some of the implementation **classes** are AndroidDriver, ChromeDriver FirefoxDriver, InternetExplorerDriver SafariDriver etc

8. ChromeDriver is class or interface

Answer: The major implementation **classes** of WebDriver **interface** are **ChromeDriver**, EdgeDriver, FirefoxDriver, InternetExplorerDriver etc. Each driver **class** corresponds to a browser. We simply create the object of the driver **classes** and work with them. It helps you to execute **Selenium** Scripts on Chrome browser.

9. Explain about Remote WebDriver

Answer: Remote WebDriver consists of a server and a client.

The server is a component that listens on a port for various requests from a Remote WebDriver client. Once the request is received, it forwards the request to the browser driver: FirefoxDriver, IDE Driver and ChromeDriver.

The client libraries serve as a Remote WebDriver client. The client translates test script requests to JSON payload and sends it across to the Remote WebDriver server using the JSON wire protocol. The diagram below depicts the remote webdriver architecture.

10. Explain webdriver commands.

Answer: we can generally classify Webdriver commands as:

- Browser commands,
- Get commands,
- Navigation commands,
- Webelement commands,
- Action commands and
- Result commands.

List of 25 More Popular WebDriver Commands & Examples

1. Get ()
2. GetCurrentUrl ()
3. GetTitle ()
4. GetText ()
5. FindElement ()
6. FindElements ()
7. isEnabled ()
8. SendKeys ()
9. Submit ()
10. Size ()
11. ImplicityWait ()
12. Untill ()
13. Navigate ()
14. MoveToElement ()
15. Click ()
16. Close ()
17. SwtichTo ()
18. Accept ()
19. Dismiss ()
20. GetWindowHandel ()
21. GetWindowHandels ()
22. GetConnection ()
23. Quit ()

11. How will you handle dropdown?

Answer:

1. selectByIndex - It is used to select an option based on its index, beginning with 0. **dropdown**. SelectByIndex (5);
2. selectByValue - It is used to select an option based on its 'value' attribute. **dropdown**. selectByValue("Database");
3. selectByVisibleText - It is used to select an option based on the text over the option.

12. How will you handle drop down without using Select class.

Answer:

If you want to **select an** option directly **without using the Select class**, you can achieve it by **using** Xpath [get **the** tag '**select**' and then it's child tag 'option'], as.

13. How will you handle pop-ups and alert in selenium?

Answer:

Handling alerts manually is a tedious task. To reduce human intervention and ease this task, Selenium provides a wide range of functionalities and methods to handle alerts.

The following methods are useful to handle alerts in selenium:

1. **Void dismiss ():** This method is used when 'Cancel' button is clicked in the alert box.
   Syntax: driver (). SwtichTo (). Alert (). Dismiss ();
2. **Void accept ():** This method is used to click on the 'OK' button of the alert.
   Syntax: driver (). SwtichTo (). Alert (). accept ();
3. **String GetText ():** This method is used to capture the alert message.
   Syntax: driver (). SwtichTo (). Alert (). getText ();
4. **Void SendKeys (String stringToSend):** This method is used to send data to the alert box.
   Syntax: driver (). SwtichTo (). Alert (). SendKeys ("Text");

14. How will you handle window/desktop pop up?

Answer:

1. Driver. getWindowHandles (); In order to **handle** the opened **windows** by Selenium webdriver, you can use Driver. getWindowHandles () to switch between the **windows**.
2. Driver. getWindowHandle (); When the webpage is loaded, you can **handle** the main **window** by using driver. getWindowHandle ().

15. How will you handle frames?

Answer:

Select a **frame** by its (zero-based) index. That is, if a page has **multiple frames** (more than 1), the first **frame** would be at index "0", the second at index "1" and so on. Once the **frame** is selected or navigated, all subsequent calls on the **WebDriver** interface are made to that **frame**.

Syntax: driver (). switchTo (). frame (0); / driver (). switchTo (). frame ("frame1");

16. Explain about 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.

17. Explain the types of xpath?

Answer:

Two Types of xpath.

1. Absolute xpath
2. Relative xpath.

18. Explain about HTML DOM structure

Answer: The **DOM** represents the document as nodes and objects so that programs can change the document structure, style, and content. In that hat way, programming languages connect to the webpage.

19. Explain about parent child/siblings/ansector-descendant/contains in xpath?

Answer:

1. **Contains () in Selenium** is a function within Xpath expression which is used to search for the web elements that contain a particular text. We can extract all the elements that match the given text value using the contains () function throughout the webpage. It has the ability to find the element with partial text.
   Syntax: "//h4/a[contains(text(),'SAP M')]"
2. A **Sibling in Selenium Webdriver** is a function used to fetch a web element which is a sibling to the parent element. If the parent element is known, then the web element can be easily found or located that can use the sibling attribute of the Xpath expression in selenium webdriver.
   Syntax: "//di[@class='canvas']/following-sibling::h4"
3. **Ancestor**: To find an element on the basis of the parent element we can use ancestor attribute of XPath.
4. **Parent in Selenium** is a method used to retrieve the parent node of the current node selected in the web page. It is very useful in the situation when you select an element and need to get the parent element using Xpath. This method is also used to get the parent's parent.

20. Difference between findelement and findElements

Answer: findElement: A command used to uniquely identify a web element within the web page. **findElements**: A command used to identify a list of web elements within the web page.

21. Explain about selenium Synchronization?

Answer: Synchronization meaning: when two or more components involved to perform any action, we expect these components to work together with the same pace. The co-ordination between these components to run paralelly is called **Synchronization**. **Synchronization** (Wait) in **Selenium** has a great significant value.

22. Difference between implicit and explicit wait?

Answer: Implicit waits are used to provide a default **waiting** time **between** each consecutive test step/command across the entire test script. ... **Explicit waits** are used to halt the execution until the time a particular condition is met or the maximum time has elapsed.

23. What class is used for explicit wait?

Answer: WebDriverWait Class

In order to declare explicit wait, one has to use “Expected Conditions”. The following Expected Conditions can be used in Explicit Wait. To use Explicit Wait in test scripts, import the following packages into the script. Then, Initialize A Wait **Object** using WebDriverWait Class.

24. What is fluent wait?

Answer: The **Fluent Wait** command defines the maximum amount of time for Selenium WebDriver to **wait** for a certain condition to appear. To put it simply, **Fluent Wait** looks for a web element repeatedly at regular intervals until timeout happens or until the object is found.

25. What is Wait? Class or Interface?

Answer: Wait is the generic interface, which makes selenium to wait for a particular time with the associated event. Events like elements to be click-able, element to present.

FluentWait, and WebDriverWait implements Wait interface.

26. What is different method to launch the web browser and explain?

27. Write a code for Screenshot

Answer: We have two type in TakesScreenshot.

```
        public class TakeScreenShotExample {
public static void main (String [] args) throws IOException {
                    WebDriver driver = new ChromeDriver ();
                    driver.get("https://www.hilton.com/en/");

TakesScreenshot screenshot = (TakesScreenshot) driver;
File srcfile = screenshot.getScreenshotAs(OutputType.FILE);
                    File file = new File("D://hiltonhome.png");
                    FileUtils.copyFile(srcfile, file);
              }
        }

        Another Way:
        public class TakeScreenShotExample {
public static void main (String [] args) throws AWTException {
                    Robot robot = new Robot ();
Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize ();
                    Rectangle rectangle = new Rectangle(screenSize);
BufferedImage soruce = robot. createScreenCapture (rectangle);
                    File desfile = new File("D://hiltonhomerobot.png");
                    ImageIO.write(soruce, "png", desfile);
      }
}
```

28. Difference between getWindowHandle and getWindowHandles?

Answer: **getWindowHandles** () – It stores the set **of** handles **for** all the pages opened simultaneously. driver. **getWindowHandle** () – It fetches the handle **of** the web page which **is** in focus. It gets the address **of** the active browser and it has a return type **of** String.

29. How will you handle browser window?

Answer: Get the handles of all the **windows** that are currently open using the command: Set<String> allWindowHandles = driver. getWindowHandles (); which returns the set of handles. Use the SwitchTo command to switch to the desired **window** and also pass the URL of the web page.

30. Write the code for Excel read and write?

Answer:

Read Excel

```
        public class ReadExcelExample {
public static void main (String [] args) throws IOException {
FileInputStream fis = new FileInputStream("D:\\Test.xlsx");
                    XSSFWorkbook wb = new XSSFWorkbook(fis);
                    XSSFSheet sh = wb. getSheetAt (0);
                    for (int i=0; i<=2; i++) {
                          System.out.println(sh. getRow(i). getCell(0));
                    }
                    wb.close ();
             }
}
```

Write Excel

```
        public class WriteExcelExample {
public static void main (String [] args) throws IOException {
                    File file = new File("D:\\Test.xlsx");
                    FileInputStream fis = new FileInputStream(file);
                    XSSFWorkbook wb = new XSSFWorkbook(fis);
      XSSFSheet sh = wb.getSheetAt(1);
        for(int i=0; i<=10; i++) {
                          XSSFRow row = sh.createRow(i);
WebElement alltitle = driver.findElement(By.xpath("//*[@class ='_3LU4EM']/div"));
                          String name = alltitle.getText();
                          sh.getRow(i).createCell(0).setCellValue(name);
                    }
                    FileOutputStream fos = new FileOutputStream(file);
                    wb.write(fos);
      wb.close();
        }
        }
```

31. How will you handle the dynamic web table?

32. How will you perform scroll down?

Answer:

```
        public class JavaScriptScrollDown {
              public static void main (String [] args) {
        WebDriver driver = new ChromeDriver ();
                          driver.get ("http://www.leafground.com/pages/Edit.html");

        JavascriptExecutor e = (JavascriptExecutor) driver;
              e.executeScript ("window.scroll(0,750)", "");
      }
}
```

33. What all are the exceptions in selenium?

Answer: there are many Exception classes under WebDriverException, we commonly see the below ones.

- NoSuchElementException
- NoSuchWindowException
- NoSuchFrameException
- NoAlertPresentException
- InvalidSelectorException
- ElementNotVisibleException
- ElementNotSelectableException

- TimeoutException
- NoSuchSessionException
- StaleElementReferenceException

34. Explain about StaleElementException?
    Answer: Stale Element means an old element or no longer available element. Assume there is an element that is found on a web page referenced as a WebElement in WebDriver. If the DOM changes then the WebElement goes stale. If we try to interact with an element which is staled then the **StaleElementReferenceException** is thrown.

35. How will you handle stale element exception?
    Answer: We can **handle Stale Element** Reference **Exception** by using POM. We could avoid StaleElementException using POM. In POM, we use initElements () method which loads the **element** but it won't initialize **elements**. initElements () takes latest address.

36. What exception we will get for timeout?
    Answer: Thrown when a response is not received within a specified time period. And **TimeoutException**. **Exception** thrown when a blocking operation times out. Blocking operations for which a **timeout** is specified need a means to indicate that the **timeout** has occurred.

37. Difference between quit and close?
    Answer: quit() is used to exit the browser, end the session, tabs, pop-ups etc. But the when your driver. **close** (), only the window that has focus is **closed**.

38. Difference between isDisplayed, isEnabled and isSelected?
    Answer:
    isDisplayed () is the method used to verify a presence of a web element within the webpage. **isEnabled** () is the method used to verify if the web element is **enabled** or disabled within the webpage. **isEnabled** () is primarily used with buttons.
    isSelected () is the method used to verify if the web element is selected or not.

39. Explain about properties file?
    Answer: properties files are mainly used in **Java** programs to maintain project configuration data, database config or project settings, etc. Each parameter in **properties file** is stored as a pair of strings, in key-value pair format, where each key is on one line.

40. How will you handle time outs in selenium?
    Answer: This is used to set the amount of time the **WebDriver** must wait for an asynchronous script to finish execution before throwing an error. If the **timeout** is negative, then the script will be allowed to run indefinitely.
    Types:

1. implicitlyWait ()
2. setScriptTimeout ()
3. pageLoadTimeout ()
4. Thread. Sleep ()

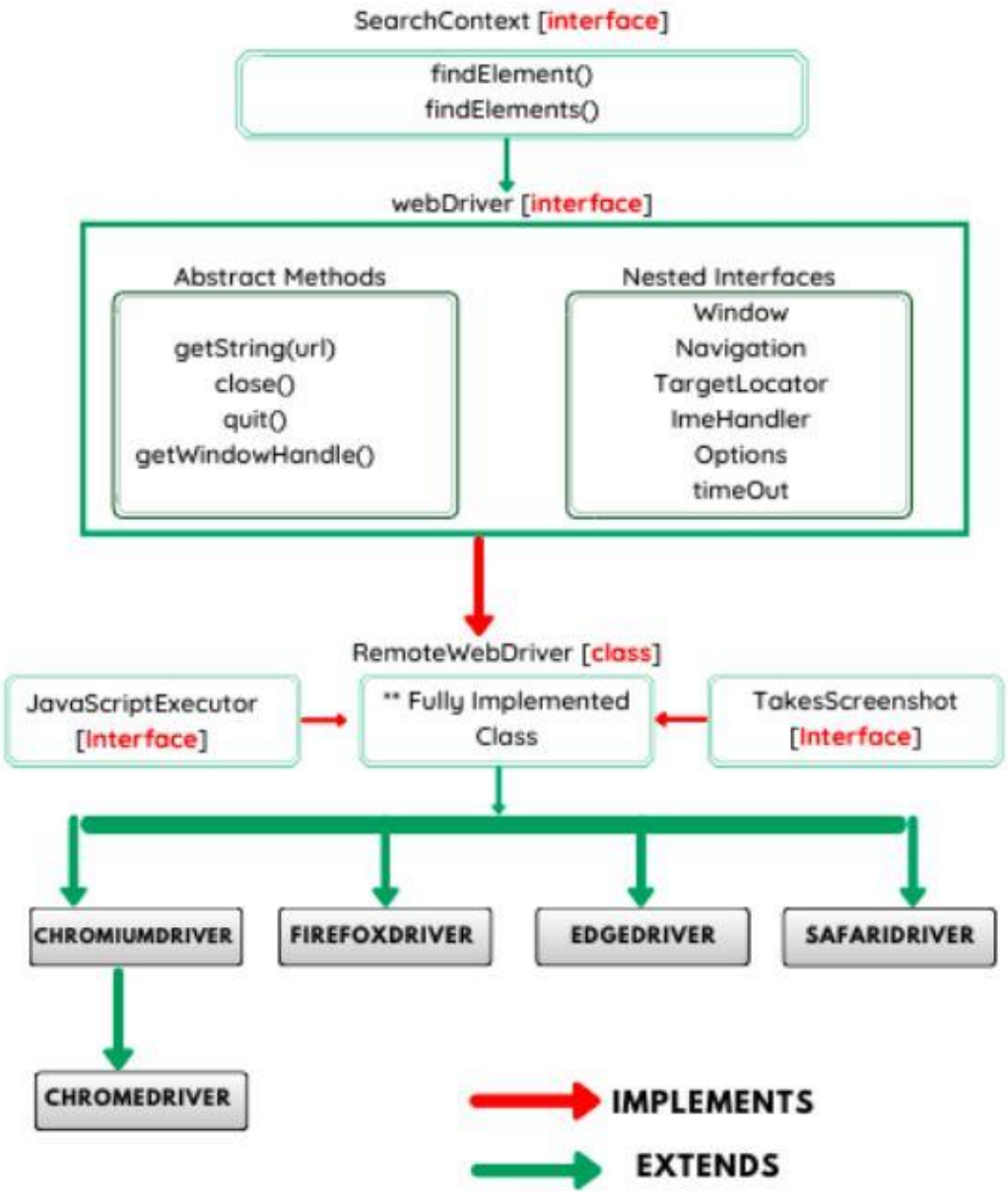
41. How will you create generic methods for object creation?

Answer: To **create** objects of **generic** class, we use following syntax. // To **create** an **instance** of **generic** class BaseType <Type> obj = new BaseType <Type>() Note: In Parameter type we can not use primitives like 'int','char' or 'double'. We can also pass multiple Type parameters in **Generic** classes.

42. Different types of interface in Selenium?

Answer:

Selenium Webdriver Interface Hierarchy



TestNG

1. What is TestNG

   Answer: **TestNG** is an automation testing framework in which NG stands for "Next Generation". **TestNG** is inspired from JUnit which uses the annotations (@). **TestNG** overcomes the disadvantages of JUnit and is designed to make end-to-end testing easy. The **TestNG in Selenium** provides an option, i.e., **TestNG**-failed.

2. Advantages of TestNG

   Answer:

- It gives the ability to produce HTML Reports of execution.
- Annotations made testers life easy.
- Test cases can be Grouped & Prioritized more easily.
- Parallel **testing** is possible.
- Generates Logs.
- Data Parameterization is possible.

3. Explain the TestNG annotation in sequence

   Answer:

   @BeforeSuite
   @BeforeTest
   @BeforeClass
   @BeforeMethod
   @Test
   @AfterMethod
   @AfterClass
   @AfterTest
   @AfterSuite

4. Explain the Attributes of TestNG (Groups, dependOnMethod, priority, invocationcount, enabled, invocationTimeout, Excepted Exception)

   Answer:

   Groups - The 'groups' attribute is used to group the different test cases that belong to the same functionality.

   Syntax: In Java File - @Test(groups= {"software company"})

   In Xml File

```
<test>
    <groups>
    <run>
        <include name = “software company”/>
    </run>
    </groups>
</test>
```

   dependOnMethod - When the second test method wants to be dependent on the first test method, then this could be possible by the use of "**dependOnMethods**" attribute. If the first test method fails, then the dependent method on the first test method, i.e., the second test method will not run.

   Syntax: @Test(dependsOnMethods= {"WebStudentLogin"})

5. Explain about @Factory annotation?

Answer: TestNG @**Factory annotation** is used to specify a method as a **factory** for providing objects to be used by TestNG for test classes. The method marked with @**Factory annotation** should return Object array.

6. What is Assertion in selenium?

Answer:

In **Selenium**, **Asserts** are validations or checkpoints for an application. **Assertions** state confidently that application behaviour is working as expected. One can say that **Asserts in Selenium** are used to validate the test cases. They help testers understand if tests have passed or failed.

7. Difference between Soft Assert and Hard Assert?

Answer:

Soft Assert: **Soft Assert** collects errors during @Test. **Soft Assert** does not throw an exception when an **assert** fails and would continue with the next step after the **assert** statement.

Hard Assert: **Hard Assert** throws an AssertExceptionimmediately when an **assert** statement fails and test suite continues with next @Test.

8. Explain @Parameters in TestNG?

Answer:

One of the important features of **TestNG** is parameterization. This feature allows user to pass **parameters** to tests as **arguments**. This is supported by using the **testng** @**Parameters** annotation. There are mainly two ways through which we can provide **parameter values** to **testng** tests.

1. Xml File – parameter tags.
2. @parameters annotation.

9. Explain about DataProvider in TestNG?

Answer:

An important features provided by **TestNG** is the **testng DataProvider** feature. It helps you to write data-driven tests which essentially means that same test method can be run multiple times with different data-sets. It helps in providing complex parameters to the test methods as it is not possible to do this from XML.

10. Explain about TestNG Listeners?

Answer:

TestNG provides the @**Listeners** annotation which listens to every event that occurs in a selenium code. **Listeners** are activated either before the test or after the test case. It is an interface that modifies the **TestNG** behaviour.

11. How will you retry the failed test cases in TestNG?

Answer:

1. After the first run of an automated **test** run. Right click on Project – Click on Refresh.
2. A folder will be generated named "**test**-output" folder. Inside "**test**-output" folder, you could find "**testng**-**failed**. xml"
3. Run "**testng**-**failed**. xml" to execute the **failed test cases** again.

12. Explain about TestNG.xml file?

Answer:

TestNG. **xml file** is a configuration **file** that helps in organizing our tests. It allows testers to create and handle multiple test classes, **define** test suites and tests. It makes a tester's job easier by controlling the execution of tests by putting all the test cases together and run it under one **XML file**.

13. How will you do parallel execution in TestNG.xml file?

Answer:

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="Test Suite" verbose="2" parallel="methods" thread-count="2">
```

s ıe test cases inside <test> tag of Testing xml file will run parallel.

ɜeɜe test cases inside a Java class will run parallel

ıooɑe methods with @Test annotation will execute parallel.

ınc cases in same instance will execute parallel but two methods of two different instanc
n different thread.

14. Advantage of TestNG over Junit?

Answer:

There are three major **advantages of TestNG over JUnit**:

1. Annotations are easier to understand.
2. Test cases can be grouped more easily.
3. Parallel testing is possible.

15. Explain about @cacheLookup?

Answer:

PageFactory annotation @**CacheLookup** is used to mark the WebElements once located so that the same instance in the DOM can always be used. **CacheLookup** attribute can be used to instruct the InitElements () method to cache the element once its located and so that it will not be searched over and over again.

16. Explain about ExtentReport?

Answer:

ExtentReport is an open-source reporting library for test automation which can be easily integrated with all test-frameworks such as TestNG, JUnit, NUnit etc. These reports are high rich HTML reports. It provides results in the form of PIE charts. Using Extent Reports we could generate custom logs, add snapshots. We could use external XML file to provide extra information.

Cucumber

1. Explain about BDD framework.

Answer: **Cucumber** is a testing approach which supports Behaviour Driven Development (**BDD**). It **explains** the behaviour of the application in a simple English text using Gherkin language.

2. Explain about Cucumber

Answer: A **cucumber** is a tool based on Behaviour Driven Development (BDD) framework which is used to write acceptance tests for the web application. It allows automation of functional validation in easily readable and understandable format (like plain English) to Business Analysts, Developers, Testers, etc.

3. Difference between TDD and BDD framework

Answer:

TDD - TDD stands for Test Driven Development. In this software development technique, we create the test cases first and then write the code underlying those test cases. Although TDD is a development technique, it can also be used for automation testing development.

The teams that implement TDD, take more time for development however, they tend to find very few defects. TDD results in improved quality of code and the code that is more reusable and flexible.

TDD also helps in achieving high test coverage of about 90-100%. The most challenging thing for developers following TDD is to write their test cases before writing the code.

BDD - BDD stands for Behaviour Driven Development. BDD is an extension to TDD where instead of writing the test cases, we start by writing a behaviour. Later, we develop the code which is required for our application to perform the behaviour.

The scenario defined in the BDD approach makes it easy for the developers, testers and business users to collaborate.

BDD is considered a best practice when it comes to automated testing as it focuses on the behaviour of the application and not on thinking about the implementation of the code.

The behaviour of the application is the center of focus in BDD and it forces the developers and testers to walk-in the customer's shoes.

4. How does Cucumber works

Answer:

http://www.seleniumframework.com/cucumber-2/make-a-case/how-cucumber-works-2/

5. Explain about Features file, step definition file and cucumber runner class?

Answer:

Feature File:

Features file contain high level description of the Test Scenario in simple language. It is known as Gherkin. Gherkin is a plain English text language.

Step Definition:

Step definition maps the Test Case Steps in the feature files (introduced by Given/When/Then) to code. It which executes the steps on Application Under Test and checks the outcomes against expected results. For a step definition to be executed, it must match the given component in a feature. Step definition is defined in java files under "features/step_definitions/*_steps.java".

Runner Class:

The test **runner class** also acts as an interlink between feature files and step definition **classes**. It is in test **runner class**, that you provide the path for both feature file and step definition **class**.

6. Explain about Gherkin syntax

Answer: Gherkin is the **format** for cucumber specifications. It is a domain specific language which helps you to describe business behavior without the need to go into detail of implementation. This text acts as documentation and skeleton of your automated tests.

- **Feature**: A feature would describe the current test script which has to be executed.
- **Scenario**: Scenario describes the steps and expected outcome for a particular test case.
- **Scenario Outline**: Same scenario can be executed for multiple sets of data using scenario outline. The data is provided by a tabular structure separated by (I I).
- **Given**: It specifies the context of the text to be executed. By using data tables "Given", step can also be parameterized.
- **When**: "When" specifies the test action that has to performed
- **Then**: The expected outcome of the test can be represented by "Then"

7. Advantage of BDD framework?

Answer:

1. Coverage of User Stories
2. Clarity of Scenarios
3. Automation of Test Scenarios
4. Code Reuse in Framework
5. Parameterization in Feature File
6. Continuous Integration – Easy to integrate

8. Keywords in cucumber?

Answer:

Primary Keywords are:

Feature, Rule, Example, Scenario, Given, When, Then, And, But, Background, Scenario Outline, Scenario Template and Examples

Secondary Keywords are:

(“”) Doc String, (||) Data Tables, (@) Tags, (#) Comments

9. Explain about Scenario Outline?

Answer: Scenario outline basically replaces variable/keywords with the value from the table. Each row in the table is considered to be a **scenario**.

10. Explain about cucumber runner class?

Answer: Cucumber test **runner class** is one of the many mechanisms using which you can **run Cucumber** feature file. ... In addition to **running** a **cucumber** feature file, the test **runner class** also acts as an interlink between feature files and step definition **classes**.

11. Explain about cucumber options

Answer:

Following Main Options are available in Cucumber:

Options Type	Purpose	Default Value
dryRun	true: Checks if all the Steps have the Step Definition	false
features	set: The paths of the feature files	{}
glue	set: The paths of the step definition files	{}
tags	instruct: What tags in the features files should be executed	{}
monochrome	true: Display the console Output in much readable way	false
format	set: What all report formaters to use	false
strict	true: Will fail execution if there are undefined or pending steps	false

12. Explain about tags in cucumber?

Answer: When we just have one, two, or maybe five scenarios in a feature file. For this, **Cucumber** has already provided a way to organize your scenario execution by using **tags** in feature file. We can **define** each scenario with a useful **tag**.

13. Different ways to achieve data driven in cucumber framework?

Answer:

- Parameterization using Excel Files.
- Parameterization using Json.
- Parameterization using XML.

14. What is the return type of raw()
15. Explain about hooks and tagged hooks

POM

1. What is POM

Answer: **Page Object model** is an **object** design **pattern** in Selenium, where web **pages** are represented as classes, and the various elements on the **page** are defined as variables on the class. ... In this case we will use **Page** Factory to initialize web elements that are defined in web **page** classes or **Page Objects**.

2. What is page object

Answer: **Page Object** Model, also known as POM is a design pattern in Selenium that creates an **object** repository for storing all web elements. It is useful in reducing code duplication and improves test case maintenance. In **Page Object** Model, consider each web **page** of an application as a class file.

3. What is page factory.

Answer: **Page Factory** is a class provided by Selenium WebDriver to support **Page** Object Design patterns. In **Page Factory**, testers use @FindBy annotation. The initElements method is used to initialize web elements. @FindBy: An annotation used in **Page Factory** to locate and declare web elements using different locators.

4. Advantage of POM

Answer: **Page Object Model** (**POM**) is a design **pattern**, popularly used in test automation that creates **Object** Repository for web UI elements. The **advantage** of the **model** is that it reduces code duplication and improves test maintenance

Maven

1. What is Maven

Answer:

Maven is an automation and management tool developed by Apache Software Foundation. It was initially released on 13 July 2004. In Yiddish language the meaning of Maven is "accumulator of knowledge".

It is written in Java Language and used to build and manage projects written in C#, Ruby, Scala, and other languages. It allows the developer to create projects using Project Object Model and plugins.

It helps to build projects, dependency, and documentation. Its development process is very similar to ANT. However, it is much advanced than ANT.

Maven is also able to build any number of projects into desired output such as jar, war, metadata.

2. Advantage of Maven

Answer:

Maven can add all the dependencies required for the project automatically by reading pom file. One can easily build their project to jar, war etc. **Maven** makes easy to start project in different environments and one doesn't needs to handle the dependencies injection, builds, processing, etc.

3. Explain about POM.xml

Answer:

Project Object Model (POM) Files are XML file that contains information related to the project and configuration information such as dependencies, source directory, plugin, and goals etc. used by Maven to build the project. When you should execute a maven command you give maven a POM file to execute the commands. Maven reads pom.xml file to accomplish its configuration and operations.

4. What is Maven Artifact
5. Maven Build life cycle
6. What is group id, artifact id, version
7. Explain about maven repository and types
8. Explain about super POM
9. Explain about build plugins in maven
10. Explain about surfire and complier plugin
11. Maven command to execute the testcase in cmd

JIRA

1. What is JIRA?

Answer: **Jira** Software is part of a family of products designed to help teams of all types manage work. Originally, **Jira** was designed as a bug and issue tracker. But today, **Jira** has evolved into a powerful work management tool for all kinds of use cases, from requirements and test case management to agile software development.

2. How will you create issue in JIRA?

Answer:

1. Select the project.
2. Choose the **issue** type.
3. Add a summary.
4. Add optional details: a description, due date, sprint, priority, assignee, etc.

3. Explain JIRA workflow
4. Issue types in JIRA

Answer:

Jira Core – Task and Sub Task

Task: A task represents work that needs to be done.

Sub Task: A subtask is a piece of work that is required to complete a task.

Jira Software - Epic, Bug, Story, Task and Sub Task.

Epic: A big user story that needs to be broken down. Epics group together bugs, stories, and tasks to show the progress of a larger initiative. In agile development, epics usually represent a significant deliverable, such as a new feature or experience in the software your team develops.

Bug: A bug is a problem which impairs or prevents the functions of a product.

Story: A user story is the smallest unit of work that needs to be done.

Task: A task represents work that needs to be done.

Sub Task: A subtask is a piece of work that is required to complete a task. Subtasks issues can be used to break down any of your standard issues in Jira (bugs, stories or tasks).

Jira Service Management – Change, IT help, Incident, New feature, Problem, Service request, Service request with approval, Support.

Change: Requesting a change in the current IT profile.

IT help: Requesting help for an IT related problem.

Incident: Reporting an incident or IT service outage.

New feature: Requesting new capability or software feature.

Problem: Investigating and reporting the root cause of multiple incidents.

Service request: Requesting help from an internal or customer service team.

Service request with approval: Requesting help that requires a manager or board approval.

Support: Requesting help for customer support issues.

5. Colour code in JIRA?

Answer:

Blue – To Do
Orange – In Progress
Green – Done
Red - Bug

6. Clone in JIRA?

Answer: Cloning means to copy. To clone an issue means to create a duplicate issue within the same project. A cloned issue can be treated as a new issue and edited like other issues.

We should consider the following points while cloning an issue.

- A cloned issue is completely a separate issue from the original issue.
- Any action or operation taken on the original issue does not have any impact on the clone issue and vice-versa.
- The only connection between the original and clone is the link that is created.

Jenkins

1. What is Jenkins?

   Answer: Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. **Jenkins** is **used** to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build.

2. What is CI and CD

   Answer: Continuous integration (**CI**) and continuous delivery (**CD**) embody a culture, set of operating principles, and collection of practices that enable application development teams to deliver code changes more frequently and reliably. The implementation is also known as the **CI**/**CD** pipeline.

3. Difference between continuous delivery and continuous deployment?

   Answer: Continuous Delivery (CD) **is** a practice of automating the entire software release process. ... **Continuous Deployment is** a step up from **Continuous Delivery** in which every change **in the** source code **is deployed** to production automatically, without explicit approval from a developer.

4. How will you create job in Jenkins?
5. What is cron pattern
6. What is automated deployment process
7. What is pipeline
8. When will you go for nightly execution and day execution
9. What is your contribution towards Jenkins
10. How will you automate the defect life cycle
11. What is upstream and downstream job
12. Explain about pre-build and post-build actions

GITHUB

1. What is git.

   Answer: Git is the most commonly **used** version control system. **Git** tracks the changes you make to files, so you have a record of what has been done, and you can revert to specific versions should you ever need to. **Git** also makes collaboration easier, allowing changes by multiple people to all be merged into one source.

2. What is GitHub.

   Answer: GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.
   GitHub essentials like repositories, branches, commits, and Pull Requests.

3. Difference between git and GitHub?

   Answer: Git is a version control system that lets you manage and keep track of your source code history. **GitHub is** a cloud-based hosting service that lets you manage **Git** repositories. If you have open-source projects that use **Git**, then **GitHub is** designed to help you better manage them.

4. Explain the GitHub workflow

   Answer: GitHub flow is a lightweight, branch-based **workflow** that supports teams and projects where deployments are made regularly.

5. Difference between git pull, git fork and git clone?

Answer:

git clone means you are making a copy of the repository in your system. **git** fork means you are copying the repository to your **Github** account. **git pull** means you are fetching the last modified repository. ... **Clone is** generally used to get remote repo copy.

6. Git push, git fetch, git conflict, git stash, git revert, git reset, git clean, git squash, git merge, working tree

Answer:

Git revert - The **git revert** command is a forward-moving undo operation that offers a safe method of undoing changes. Instead of deleting or orphaning commits in the commit history, a **revert** will create a new commit that inverses the changes specified. **Git revert** is a safer alternative to **git** reset in regards to losing work.

Git Merge - The **git merge** command lets you take the independent lines of development created by **git** branch and integrate them into a single branch. Note that all of the commands presented below **merge** into the current branch.

Git Clean - To recap, **git clean** is a convenience method for deleting untracked files in a repo's working directory. Untracked files are those that are in the repo's directory but have not yet been added to the repo's index with **git** add.

Git Reset - git reset is a powerful command that is used to undo local changes to the state of a **Git** repo. **Git reset** operates on "The Three Trees of **Git**". These trees are the Commit History (HEAD), the Staging Index, and the Working Directory. There are three command line options that correspond to the three trees.

Git Push - The **git push** command is used to upload local repository content to a remote repository. **Pushing** is how you transfer commits from your local repository to a remote repo.

Git Fetch - The **git fetch** command downloads commits, files, and refs from a remote repository into your local repo. **Fetching** is what you do when you want to see what everybody else has been working on. When downloading content from a remote repo, **git** pull and **git fetch** commands are available to accomplish the task.

Git Conflict – Git can handle most merges on its own with automatic merging features. A **conflict** arises when two separate branches have made edits to the same line in a file, or when a file has been deleted in one branch but edited in the other. **Conflicts** will most likely happen when working in a team environment.

Git stash - git stash temporarily shelves (or stashes) changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on.

7. What is staging area

Answer: Staging area is files that are going to be a part of the next commit, which lets **git** know what changes in the file are going to occur for the next commit.

8. What is origin in “git push **origin** master”?

Answer: With **git push origin master** you tell git to push all of the commits in the currently checked out local branch (i.e. from your file system) to the remote repo identified by the name **origin** on its remote branch named master.

9. How will you check the version id?

10. What is Head?

Answer: When working with **Git**, only one branch can be checked out at a time - and this is what's called the "**HEAD**" branch. Often, this is also referred to as the "active" or "current" branch. **Git** makes note of this current branch in a file located inside the **Git** repository,

11. What is SSH key?

Answer: An **SSH key** is an access credential for the **SSH** (secure shell) network protocol. This authenticated and encrypted secure network protocol is used for remote communication between machines on an unsecured open network. **SSH** is used for remote file transfer, network management, and remote operating system access.

12. What is git restore?

Answer: The "restore" command helps to unstage or even discard uncommitted local changes.

One the one hand, the command can be used to undo the effects of git add and unstage changes you have previously added to the Staging Area.

On the other hand, the restore command can also be used to discard local changes in a file, thereby restoring its last committed state.

Manual

1. Explain about SDLC process
2. Explain about STLC process
3. Principle of Software Testing
4. What is regression, integration, system, module, user acceptance testing
5. What is exploratory, adhoc, buddy, pair testing
6. What is smoke and sanity testing
7. What is alpha and beta testing
8. Explain about defect life cycle
9. What is latent bug
10. What is show stopper bug
11. What is entry and exit criteria
12. Difference between QA and QC
13. Difference between verification and validation
14. What is RTM
15. What is test plan
16. What is test strategy
17. Explain about test execution report and test summary report

General Questions

1. Self Intro

Answer:

I am Raj Mohammed, currently work as software test analysis in Accelerated Development Machine, I got nearly 4.5 years of experience in testing, where as one year in manual testing and three years in automation testing using selenium web driver in java.

I graduated with master of computer application in 2013 and was offered as data analyst position in data matrix work till 2015 and the started my carrier as tester in Accelerate development machine. During this time, I completed my ISTQB certification too.

2. Explain about project

Answer:

Gird Collection – Path Plastics, south Africa (Scrum Call – 12.30pm ITS)

The purpose of the project is a widely implemented strategy for managing a company's interactions with vendor and purchase prospects.

Establish and maintain strong vendor relationship while reducing inquires to finance department with an integrated vendor center that provides vendors with self-service access to purchase orders, accounts payable data and other key documents.

We worked as 7 members team in both manual and automation testing, we followed the Data driven and Behaviour data driven Framework approach to test an entire application.

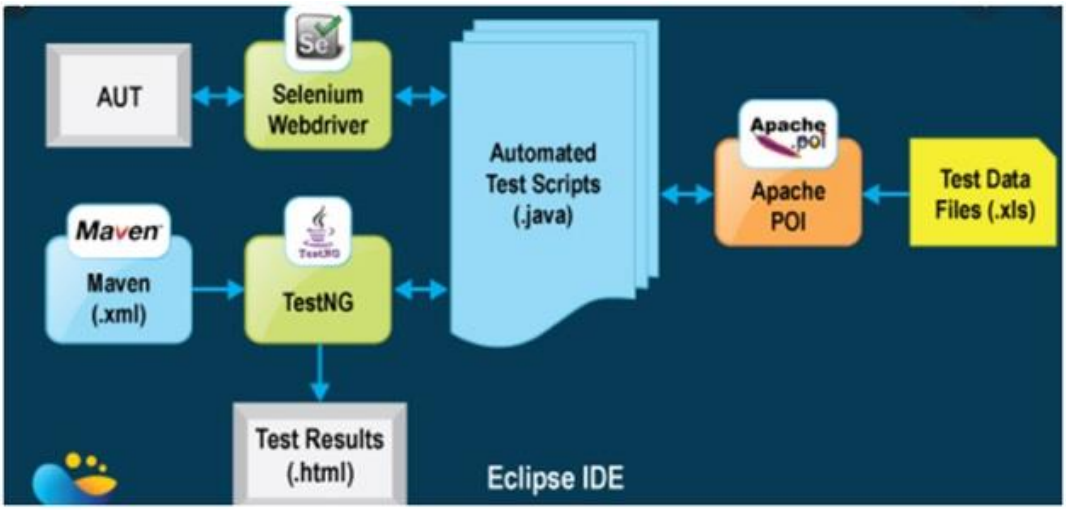
We followed the industry standard agile principles and ceremonies like daily stand-ups, monthly retrospection and completion of sprint review meeting.

Programming language we used is Java and Testing Tools like Selenium, TestNG, Data Driven/Cucumber with Selenium.

3. Explain about the current project framework architecture.

Answer:

Framework as a Set of Rules or Best Practices which we can follow in a systematic way to achieve the desired Goals.



Programming Language: Java
Framework: Data-Driven

Structure of Framework:

Test-Base Class: Every framework has a Base Class; we do initialize WebDriver and implicit waits and bloggers, reports, etc in the test base class.

Page Object Model: Do your tests independent of element locators. You have to specify where you store all your element your locaters such as ID XPath CSS whatever, if you are using a Page Object model to separate element locators with tests then clearly explain that. Nowadays everyone is using the Page Object design model to do this. We can also keep Element locators in property files.

Functions Library: Every project has some Functions Library but in most of the cases, we do use separate Functions as Generic Functions and Project-Specific functions also called

Business functions. Sometimes asked in an interview, what is the difference between Generic and Business Function?
Don't get confuse and reply Generic Functions are related to Operations we do perform like Click, select etc., of an Element and Business Functions are more related for B2B Business to Business applications nature. Example Transfer Funds Functionality is more related to Banking Application, whereas Order Booking Functionality more over related to Retail Business applications like Amazon and Flipkart. If you call Test data from the excel sheet comes under Generic Function and more technically we call it as Utility functions property.

Property Files: Details such as browser type URL could be placed in property files and also specify what are the Test NG Annotations that you have used in your Framework.

Test NG: In case if you are using Test NG Annotations you can explain how you do use Test NG Annotations into your Framework.

@BeforeSuite
@BeforeTest
@BeforeClass
@BeforeMethod
@Test
@AfterMethod
@AfterClass
@AfterTest
@AfterSuite

Parameterization: You can explain how you do Parameterize Test cases using Excel files or Test NG Annotations.
Screenshots: Explain how you captured Error screenshots image into Screenshot folder for the Pass or Failed Test Cases.
Test Reports: Explain how you Generate Reports like HTML, Extent Reports etc. in your Test Reports Folder and how you send those Reports to your peers like sending that email to your leads or related Peers.
Version Control: Mention what is the Version Control you have used as a GIT Repository or SVN.
Continuous Integration Tool: Mention continuous-integration tools like Jenkins if you use the same into your Project.

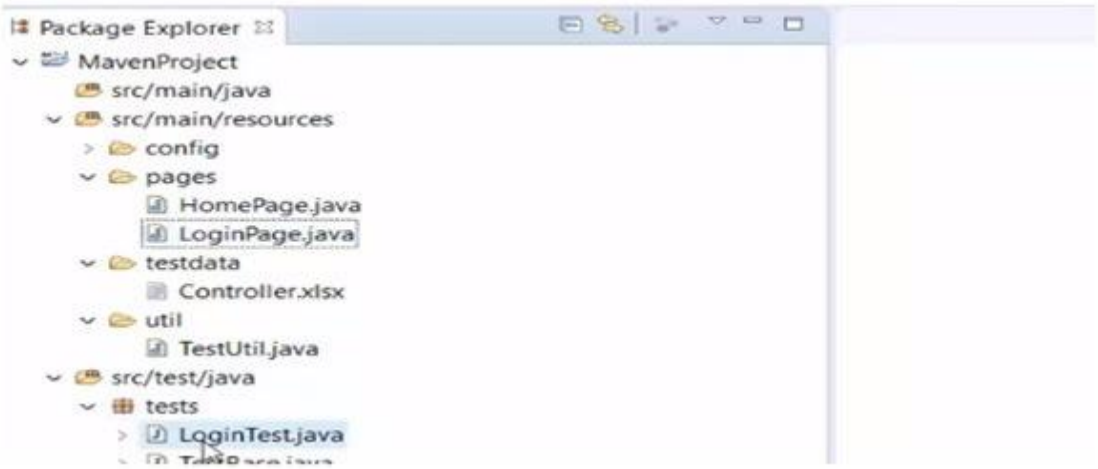
Sample:

Let's get started with the language you can tell that in our selenium project. Like we are using Java language in our project because most of the Automation developers know selenium with Java type of framework or any language which you use Python, C# etc.
Types of Framework: In our project, we are using a Data-driven or Keyword-driven or Hybrid Framework by using the Page Object Model design pattern with Page Factory.

To continue in detail related Page Object Model you can mention for the Page Object Model. Under POM we have maintained a Class for Every Web Page separately under each Web Page has a separate Class and that Class holds the functionality and members of that Particular Web Page. We have separate classes for every individual Test case.

Packages: we have separate Packages for pages and Test scripts. All the web page related Classes comes under Pages Package and all the Tests related Classes comes under Tests Package. And all the Tests are kept in the SRC tests Java. Remaining files such as
Config. properties Element Locators Utility Files Test data, etc. are kept in SRC may in Java.

For Example: Homepage and login page of the Application have a separate class to store element locators for the login Tests.



There would be a separate class that calls the method from the Homepage class and login page.

To continue about Test base class, Test base class deals with all the common functions used by all the pages. This class is responsible for loading the configurations from property files, Initializing the WebDriver implicit waits, External reports and to create the Object of a file Input stream. Which is responsible for pointing towards the file from which the data should be read?

To continue about Functions Library, Utility file class - which we also call it as Functions class it stores and handles the Functions. A code that is repetitive such as waits, actions or capturing screenshots or accessing Excel sheets and sending emails these are the repetitive Tasks. These repetitive tasks can be commonly used across the entire framework. These repetitive tasks we make it as a function and we use the entire framework. The main reason behind creating utility classes is to achieve with reusability of code. Like we create Login and Logout Functions and add in our Function's Library we can call the same and use for every Test case into our Project Application Framework. This class extends the Test Base class to inherit the properties of Tests Based on Test Util.

To be continue related Properties File - This file stores the information that remains static throughout the framework such as Browser-specific information, Application URL and Screenshots path etc. All the details which change as per the Environment and Authorization such as your Login credentials are kept in config.Property files keeping these details in a separate file makes easy to maintain.

Screenshots: will be captured and stored in a Separate Folder also the Screenshots of the Failed Test cases will be added in the Extent Reports.

Test Data, All the historical test data will be kept in the excel sheet by using this particular Excel Sheet we pass this data and handle Data-driven testing. We use Apache to point to handle Excel sheets

Test NG, we use Test NG annotations for Assertions, Grouping and parallel Execution.

Maven, we are using Maven for build execution and dependency purpose.
We integrate the Test NG dependency in POM.XML file and running this POM.XML file using Jenkins

Version Control Tool - In our project, we use GIT as a Repository to store our Test scripts.

By using Jenkins's continuous-integration tool, we execute Test cases daily basis also for nightly execution based on the scheduled. Test Result will send to the Peers using Jenkins.

Extent Reports, For the Reporting purpose, we are using Extent Reports it generates beautiful HTML reports. We use the Extent Reports for maintaining logs also to include the screenshots of Failed Test cases in the External reports.

4. Roles and responsibilities.

Answer:

- Created automation framework in Selenium WebDriver using behaviour driven approach like Cucumber and also Data Driven.
- Developed automation test scripts
- Designed & developed Hybrid framework using Cucumber, TestNG and Apache POI
- Involved in the preparing & Maintenance of Planned schedule and Defect log & Test Results review
- Status update on daily basis through e-mails and discussing the same in the Daily Scrum calls.
- Co-ordination with Functional & Technical teams and defining the scope of Testing for each phase.
- Developed **Maven** Build scripts to run Smoke and Regression Test scripts developed in **Selenium WebDriver**
- Involved in the Continuous Integration of the automation framework with **Jenkins**
- Performed Defect Tracking & Management in **JIRA**

5. Agile Methodology.

Answer:

Agile methodologies are approaches to product development that are aligned with the values and principles described in the Agile Manifesto for software development. Agile methodologies aim to deliver the right product, with incremental and frequent delivery of small chunks of functionality, through small cross-functional self-organizing teams, enabling frequent customer feedback and course correction as needed.

In doing so, Agile aims to right the challenges faced by the traditional “waterfall” approaches of delivering large products in long periods of time, during which customer requirements frequently changed, resulting in the wrong products being delivered.

Application of Agile Methodology

Through most of its brief history (since 1999-2000), “Agile” has been predominantly an approach to software development and IT application development projects. Since then, however, it now extends to other fields, too, especially in the knowledge and services industries.

Agile is about being responsive to the market and to the customer by responding quickly to their needs and demands and being able to change direction as the situation demands. Be it IT or software development or any other field where there is a flow of work and delivery of work products, Agile methods are applicable. Agile methods attempt to maximize the delivery of value to the customer and minimize the risk of building products that do not – or no longer – meet market or customer needs.

They do this by breaking up the traditionally long delivery cycle (typical of the legacy “waterfall methods”) into shorter periods, called sprints or iterations. The iteration provides the cadence for delivering a working product to the customer, getting feedback and making changes based on the feedback.

Thus, Agile methods have sought to reduce delivery times (delivering early, delivering often) to ensure that smaller vertical chunks of the product get to the market, enabling customers to provide feedback early and ensure that the product they finally get meets their needs.

Agile has become an umbrella term for a variety of planning, management and technical methods and processes for managing projects, developing software and other products and services in an iterative manner. These methods include Scrum, by far the most prevalent and popular method for software, XP (eXtreme Programming or Paired Programming), and more lately Kanban.

Agile methods also include technical practices – most of which fall under the umbrella term DevOps – that enable Test Automation, Continuous Integration/ Continuous Delivery/ Deployment (CI/ CD) and overall, an ever-shrinking delivery cycle for software and other products and services.

The use of Agile as an approach to project management has increased dramatically in recent years. Gartner predicts that agile development methods will soon be used in 80% of all software development projects.

What is the Agile Manifesto?

The Agile Manifesto is a statement of core values and principles for software development. The Agile Manifesto for software development was set up in 2001 and it is a declaration of 4 vital rules and 12principles that serve as a guide for people in agile software development. It was created by 17 professionals who already practiced agile methods such as XP, DSDM, SCRUM, FDD, etc, gathered in the snowy mountains of the US state of Utah, convened by Kent Beck.

4 Core values of Agile Manifesto

- **Individuals and interactions over processes and tools** – The first value emphasizes teamwork and communication. We must understand that software development is a human activity and that the quality of interaction between people is vital. Tools are an important part of software development, but making great software depends much more on teamwork, regardless of the tools team may use.
- **Working software over comprehensive documentation** – Documentation has its place and can be a great resource or reference for users and co-workers alike. The main goal of software development, however, is to develop software that offers business benefits rather than extensive documentation.
- **Customer collaboration over contract negotiation** – Development teams must work closely and communicate with their customers frequently. By listening to and getting feedback, teams will understand what all stakeholders really want.

- **Responding to change over following a plan** – Changes are a reality in Software development, a reality that your Software process should reflect. A project plan must be flexible enough to change, as the situation demands.

12 Principles of the Agile Manifesto

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Businesspeople and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity–the art of maximizing the amount of work not done–is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Key Agile Methodologies

Agile is an umbrella term for several methods and practices. Let's look at some of the popular methodologies:

- Scrum
- Extreme Programming (XP)
- Adaptive Software Development (ASD)
- Dynamic Software Development Method (DSDM)
- Feature Driven Development (FDD)
- Kanban
- Behaviour Driven Development (BDD)

6. Explain Scrum Process?

Answer:

Scrum is an agile way to manage a project, usually software development. Agile software development with Scrum is often perceived as a methodology; but rather than viewing Scrum as methodology, think of it as a framework for managing a process.

What is Scrum?

All the foundational knowledge of Scrum including: the framework, values, different roles, meetings, backlogs, and improving efficiency & quality.

In the agile Scrum world, instead of providing complete, detailed descriptions of how everything is to be done on a project, much of it is left up to the Scrum software development team. This is because the team will know best how to solve the problem they are presented.

This is why in Scrum development, for example, a sprint planning meeting is described in terms of the desired outcome (a commitment to a set of features to be developed in the next sprint) instead of a set of Entry criteria, Task definitions, Validation criteria, Exit criteria (ETVX) and so on, as would be provided in most methodologies.

Scrum relies on a self-organizing, cross-functional team. The scrum team is self-organizing in that there is no overall team leader who decides which person will do which task or how a problem will be solved. Those are issues that are decided by the team as a whole.

And in Scrum, a team is cross functional, meaning everyone is needed to take a feature from idea to implementation.

Within agile development, Scrum teams are supported by two specific roles. The first is a Scrum Master, who can be thought of as a coach for the team, helping team members use the Scrum process to perform at the highest level.

The product owner (PO) is the other role, and in Scrum software development, represents the business, customers or users, and guides the team toward building the right product.

Scrum Development: What's Involved?

The Scrum model suggests that projects progress via a series of sprints. In keeping with an agile methodology, sprints are time boxed to no more than a month long, most commonly two weeks.

Scrum methodology advocates for a planning meeting at the start of the sprint, where team members figure out how many items they can commit to, and then create a sprint backlog – a list of the tasks to perform during the sprint.

During an agile Scrum sprint, the Scrum team takes a small set of features from idea to coded and tested functionality. At the end, these features are done, meaning coded, tested and integrated into the evolving product or system.

On each day of the sprint, all team members should attend a daily Scrum meeting, including the Scrum Master and the product owner. This meeting is time boxed to no more than 15 minutes. During that time, team members share what they worked on the prior day, will work on that day, and identify any impediments to progress.

The Scrum model sees daily scrums as a way to synchronize the work of team members as they discuss the work of the sprint.

At the end of a sprint, the team conducts a sprint review during which the team demonstrates the new functionality to the PO or any other stakeholder who wishes to provide feedback that could influence the next sprint.

This feedback loop within Scrum software development may result in changes to the freshly delivered functionality, but it may just as likely result in revising or adding items to the product backlog.

Another activity in Scrum project management is the sprint retrospective at the end of each sprint. The whole team participates in this meeting, including the Scrum Master and PO. The meeting is an opportunity to reflect on the sprint that has ended, and identify opportunities to improve.

Scrum Process: The Main Artifacts

The primary artifact in Scrum development is, of course, the product itself. The Scrum model expects the team to bring the product or system to a potentially shippable state at the end of each Scrum sprint.

The product backlog is another artifact of Scrum. This is the complete list of the functionality that remains to be added to the product. The product owner prioritizes the backlog so the team always works on the most valuable features first.

The most popular and successful way to create a product backlog using Scrum methodology is to populate it with user stories, which are short descriptions of functionality described from the perspective of a user or customer.

In Scrum project management, on the first day of a sprint and during the planning meeting, team members create the sprint backlog. The sprint backlog can be thought of as the team's to-do list for the sprint, whereas a product backlog is a list of features to be built (written in the form of user stories).

The sprint backlog is the list of tasks the team needs to perform in order to deliver the functionality it committed to deliver during the sprint.

Additional artifacts resulting from the Scrum agile methodology is the sprint burn-down chart and release burn-down chart. Burn-down charts show the amount of work remaining either in a sprint or a release, and are an effective tool in Scrum software development to determine whether a sprint or release is on schedule to have all planned work finished by the desired date.

The Agile Scrum Project: Main Roles

Even if you are new to Scrum, you may have heard of a role called the Scrum Master. The Scrum Master is the team's coach, and helps Scrum practitioners achieve their highest level of performance.

In the Scrum process, a Scrum Master differs from a traditional project manager in many ways, including that this role does not provide day-to-day direction to the team and does not assign tasks to individuals.

A good Scrum Master shelters the team from outside distractions, allowing team members to focus maniacally during the sprint on the goal they have selected.

While the Scrum Master focuses on helping the team be the best that it can be, the product owner works to direct the team to the right goal. The product owner does this by creating a compelling vision of the product, and then conveying that vision to the team through the product backlog.

The product owner is responsible for prioritizing the backlog during Scrum development, to ensure it’s up to par as more is learned about the system being built, its users, the team and so on.

The third and final role in Scrum project management is the Scrum team itself. Although individuals may join the team with various job titles, in Scrum, those titles are insignificant. Scrum methodology states that each person contributes in whatever way they can to complete the work of each sprint.

This does not mean that a tester will be expected to re-architect the system; individuals will spend most (and sometimes all) of their time working in whatever discipline they worked before adopting the agile Scrum model. But with Scrum, individuals are expected to work beyond their preferred disciplines whenever doing so would be for the good of the team.

One way to think of the interlocking nature of these three roles in this agile methodology is as a race car.

The Scrum team is the car itself, ready to speed along in whatever direction it is pointed. The product owner is the driver, making sure that the car is always going in the right direction. And the Scrum Master is the chief mechanic, keeping the car well-tuned and performing at its best.