**­POM and Page Factory**

|  |  |
| --- | --- |
| **Q. 1** | **Why Page Object Model (POM)?** |
| Ans. | * When you are writing functional tests using Selenium, a major part of your code will consist of **interactions with the web interface** you are testing through the WebDriver API. * Developing test scripts using Selenium can result into an unmaintainable project. One of the primary reasons behind this is that too many duplicated code is used and this will result in **duplicated usage of locators**. * The test script maintenance becomes difficult with time as the **test suit grows** * Even with a simple test as this, **readability is very poor**. There is a lot of WebDriver code, which obscures the purpose of the test, making it slow and difficult to digest. * **Changes in the UI** breaks multiple tests often in several places and makes your test script fail. * So, instead of having each test fetch elements directly and being fragile towards UI changes, the **Page Object Model** introduces what is basically a **decoupling layer,** which separates the test script which interacts with the web page and the object repository for locating the web elements. |
| **Q. 2** | **What is Page Object Model (POM)?** |
| Ans. | * Page Object Model is a Design Pattern which has become popular in Selenium Test Automation. * A Page Object represents different pages/sections of a website as objects within the test script. Page object model (POM) can be used in any kind of framework such as modular, data-driven, keyword driven, hybrid framework etc. * Page Objects introduces an abstraction layer within your Selenium tests and it provides a programmatic API to drive and interact with a UI. It makes automation easily readable and maintainable. Each page of your AUT (Application Under Test) is mapped to a class file in your code and each method within the class file can be treated as a service offered by the PageObject. * The benefit is that if the UI changes for the page, the tests themselves don’t need to change, only the code within the page object needs to change. Subsequently, all changes to support that new UI is in one place. * This model helps in enhancing the tests, makes them highly customizable, reduces the code duplication, builds a layer of abstraction and finally hides the inner implementation from tests |
| **Q. 3** | **Explain Page Object Model (POM) Architecture** |
| Ans. | * The below figure shows a clear flow from the test cases to the webpage. * Initially, the test scripts interact with the page objects. * These page objects interact with the Selenium WebDriver which contains selenium functions that finally hit the webpage and perform actions on it. |
| **Q. 4** | **Summarize the advantages of Page Object Model (POM)** |
| Ans. | * **Increases Code Reusability** – We could achieve code reusability by writing the code once and use it in different tests. * **Improves Code Maintainability** – POM can separate the page operations from complicated business logic in the test case. When the change happens in page, you only need to update the page object rather than tens or hundreds of test cases. * **Independent Object Repository** – Object repository is made independent of any test scripts. Whatever objects you need to develop the test scripts, you can call from a specific object repository. * **Readability** – Improves readability due to clean separation between test code and page specific code. * **Supports Selenium Frameworks** – POM can be used with any type of selenium framework like Keyword, Data driver, hybrid etc. |
| **Q. 5** | **Give an overview of Selenium Design Patterns** |
| Ans. | * A design pattern is an approach to implement a general reusable solution to a commonly occurring problems in software design * A design pattern is not a finished design or a solution that can be readily transformed into the code * In short, they are templates or description of how to solve a problem that can be used in many different situations * Although design patterns are not reserved only for software development they seem not to be widely discussed in software automation * There are sophisticated design patterns used to solve complex issues in software development * But also, there does exist the easy to understand and implement design patterns in automation testing that can significantly improve readability and maintainability of our test automation code   **Advantages of Design Patterns in Selenium Automation Testing**   * + Eliminates Duplicate Code   + Introduces Locator Strategy   + Design for Scalability |
| **Q. 6** | **What is Page Factory Design Pattern ?** |
| Ans. | **Page Factory Design Pattern :**   * Page Factory pattern is an optimized version of the Page Object Model (POM) Design Pattern * Page Factory is an inbuilt page object model concept for Selenium WebDriver, but it is very optimized * Page Factory can be used in any kind of framework such as Data Driven, Modular or Keyword Driven * Page Factory gives more focus on how the code is being structured to get the best benefit out of it * By integrating POM and Page Factory with the Test Case Model, you receive more focus on how the code is being structured to get the most benefits * PageFactory is a class provided by Selenium WebDriver to support the Page Object design pattern. * It makes handling "Page Objects" easier and optimized by providing the annotations and methods |
| **Q. 7** | **Give the advantages of Page Factory Design Pattern** |
| Ans. | **Separation of Concerns** :   * Page Factory Model (Extended POM) provides clear separation of HTML Attributes, Methods to access UI Elements, Test Logic and Assert statements. * It further helps us to separate Test data and Configuration from the test logic using TestNG, Maven/Ant tools. * This approach enables us to build a robust test automation framework that can be sustainable with easy maintenance.     **Avoids code Duplication** :   * Page Factory Model facilitates us to define all the UI Elements and the methods to access the UI Elements inside a single Web Page class and access them by declaring the object of the Web Page class inside the test method. * Let’s assume there are 10 different test methods that refers to the UI Elements on Flipkart home page, with Page Factory we just have to initialize the page object of the Flipkart home page. * This significantly reduces the code duplication   **Easy Maintenance** :   * Business requirements frequently change for web applications as compare to any other application. * In most of the cases, these changes in requirements brings in the changes in the UI of the web application. * Using Page Factory Model (Extended POM) we can easily enhance the test scripts where the test logic/ workflow remains the same but the UI Element locators have changed. |
| **Q. 8** | **Explain how to implement Page Factory Design Pattern** |
| Ans. | 1. **Initialization of Page Objects Using Page Factory :**   **Use of initElements() :** We should initialize page objects using initElements() method from PageFactory Class. Once we call initElements() method, all elements will get initialized. PageFactory.initElements() static method takes the driver instance of the given class and the class type, and returns a Page Object with its fields fully initialized.  There are 3 ways of initializing this. We should preferably use a constructor which takes a WebDriver instance as its only argument. An exception will be thrown if the class cannot be instantiated.       1. **Use Annotations of PageFactory Class :**   In Page Factory, annotations are used to give descriptive names for WebElements to improve code readability.   1. **@FindBy Annotation**  * @FindBy can accept TagName, PartialLinkText, Name, LinkText, Id, Css, ClassName, XPath as attributes. * This is an alternative mechanism for locating the element or a list of elements. This allows users to quickly and easily create PageObjects. * Example: The below given two annotations are pointing to the same UI Element      1. **@CacheLookup:**  * Every time when a method is called on a WebElement, the driver will first find it on the current page and then simulate the action on the WebElement. * There are cases where we will be working with a basic page, and we know that we will find the element on the page every time we look for it, In such cases we can use this annotation   ‘@CacheLookup‘ which is another annotation in page factory.   * But whenever we use @CacheLookup annotation, we will be losing one of the page factory benefit as it will find the element once and then keep a reference to it, hence, we are more likely to see StaleElementExceptions.      1. **AjaxElementLocatorFactory:**  * AjaxElementLocatorFactory is a lazy load concept in Page Factory pattern to identify WebElements only when they are used in any operation i.e. a timeOut for a WebElement can be assigned to the Object page class with the help of AjaxElementLocatorFactory. * The below code will wait for maximum of 30 seconds until the elements specified by annotations is loaded. If the element is not found in the given time interval, it will throw NoSuchElementException exception. |
| **Q. 9** | **Give the comparison Between Page Object Model (POM) and Page Factory** |
| Ans. | * A Page Object Model (POM) is a test design pattern which works on the principle of organizing the page objects in such a manner that script and page objects can be separated easily * A Page Factory is one way of implementing PageObject Model which is inbuilt in selenium * Page Object Model is a design approach whereas the PageFactory class from the package ‘org.openqa.selenium’ is used to provide additional support for the Page Object pattern * In plain POM, you define locators using ‘By’ while in Page Factory, you use FindBy annotation to define page objects * Plain Page Object Model (POM) is not optimal as it does not provide lazy initialization while Page Factory provides lazy initialization * Plain Page Object Model (POM) will not help in StaleElementReferecneException while Page Factory takes care of this exception by relocating web element every time whenever it is used. * In plain Page Object Model (POM), you need to initialize every page object individually otherwise you will encounter NullPointerException while in PageFactory all page objects are initialized (Lazily) by using initElements() method. |
| **Q. 10** | **Why we have to use External Object Repository and how it can be implemented.** |
| Ans. | * One of the main burdens of automated GUI test script maintainability is the amount of maintenance needed when object properties change within the application under test. * A very common way of minimizing the time it takes to update your automated test scripts is the use of a central object repository. * An object repository is a common storage location for all objects * With reference to Selenium WebDriver, objects would typically be the locators used to uniquely identify web elements * The important advantage of implementing Object Repository is the separation of objects from test cases * You are required to make changes only in the object repository when any of the WebElement’s locator value is changes rather than making changes in all the test cases in which the locator has been used * Maintaining an object repository increases the modularity of framework implementation and hence external object repository is very important.   **Implementing external object repository:**   * A basic object repository can be implemented as a collection of key-value pairs, with the key being a logical name identifying the object and the value containing unique objects properties used to identify the object on a screen. It can be implemented in following ways : * Object Repository Using Properties file * Object Repository Using XML file |