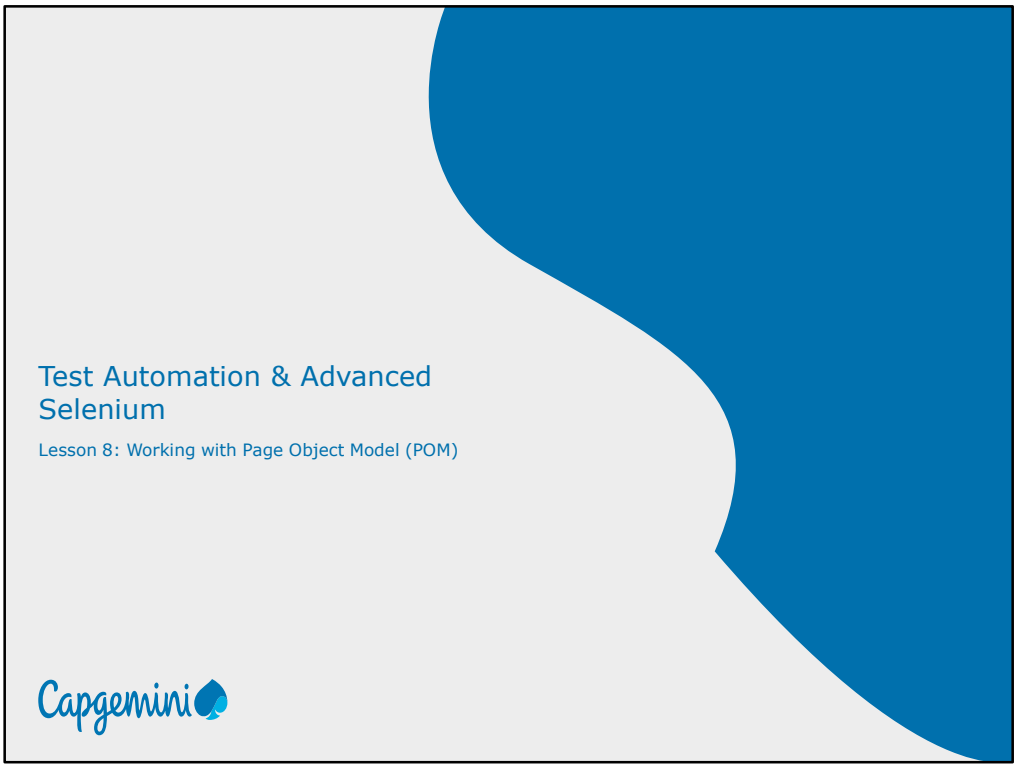
Test Automation & Advanced Selenium
Lesson 8: Working with Page Object Model (POM)
Capgemini

Lesson Objectives

- Why Page Object Model (POM)?
- What is Page Object Model (POM)?
- Page Object Model (POM) Architecture
- Advantages of Page Object Model (POM)
- Overview of Selenium Design Patterns
- Importance of Design Patterns in Selenium Automation Testing

8.1: Getting Started with Page Object Model

Why Page Object Model (POM)?

Problem:

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. Most of the times this code does the job of fetching the elements, verify some state of the elements through various assertions and move on to fetch the next element.

Consider the following simple example: Figure 1.1

```
List<WebElement> countryCodes =
driver.findElements(By.id("countryCodes"));
for (WebElement countryCode : countryCodes)
{
     if (countryCode.getText().equals("1043"))
     {
        countryCode.click();
        break;
     }
}
```

8.1: Getting Started with Page Object Model

Why Page Object Model (POM)? (Cont.)

Following are some of the challenges posed by traditional automation methods as given in Figure 1.1

- Difficult to maintain test scripts: The test script maintenance becomes difficult with time as the test suit grows. If 10 different scripts are using the same page element, with any change in that element, you need to change all 10 scripts. This is time consuming and error prone.
- Test cases are difficult to read: Even with a simple test as this, readability is very poor. There is a lot of WebDriver code, that obscures the purpose of the test, making it slow and difficult to digest.
- Duplication of selectors both inside and across tests: Duplicity of locators makes the test code inefficient.
- Changes in the UI breaks multiple tests often in several places: With any interface, and web interface, it is common that both minor and major changes to the UI is implemented frequently. This could be a new design, restructuring of fields and buttons, and this will likely impact your tests. So, your test fails, and you need to update your selectors.

Why Page Object Model (POM)?

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. Duplicated code could be caused by duplicated functionality and this will result in duplicated usage of locators. If you need to use same web element at another place, you need to locate it again. It is developing same code again and again. Your code is not reusable. It produces duplicate code. Waste of effort and time. This leads to the project which is less maintainable. Suppose you have used same locator in multiple script files. After some deployment, there is change in locator. You will be fed up by updating locator in all script files. Imagine if there are multiple locators are changed. If some locator will change, you must walk through the whole test code to adjust locators where necessary.

8.1: Getting Started with Page Object Model

Why Page Object Model (POM)? (Cont.)

Solution:

- 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. You create an object that represents the UI you want to test, which could be a whole page or a significant part of it. The responsibility of this object is to wrap HTML elements and encapsulate interactions with the UI, meaning that this is where all calls to WebDriver will go. This is where most web elements are. And this is the only place you need to modify when the UI changes. The Page Object Model (POM) is a new test automation buzz word. It was brought to address the multiple bottlenecks in web application testing like given above.

8.1: Getting Started with Page Object Model

What is Page Object Model (POM)?

- A Page Object Model is a design pattern that can be implemented using selenium WebDriver
- A Design pattern is a generic solution to a common software design/architecture problem
- Implementation of these design patterns leads to inclusion of best practices and best solution, evolved over the time by others while working with similar problems
- Page Object Model is a design pattern to create object repository for web UI elements
- It essentially models the pages/screen of the application as objects called Page Objects, all the functions that can be performed in the specific page are encapsulated in the page object of that screen
- In this way any change made in the UI will only affect that screens page object class thus abstracting the changes from the test classes

What is Page Object Model (POM)?

Page Object Model is a Design Pattern which has become popular in Selenium Test Automation. It is widely used design pattern in Selenium for enhancing test maintenance and reducing code duplication. 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.

PageObjects 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.

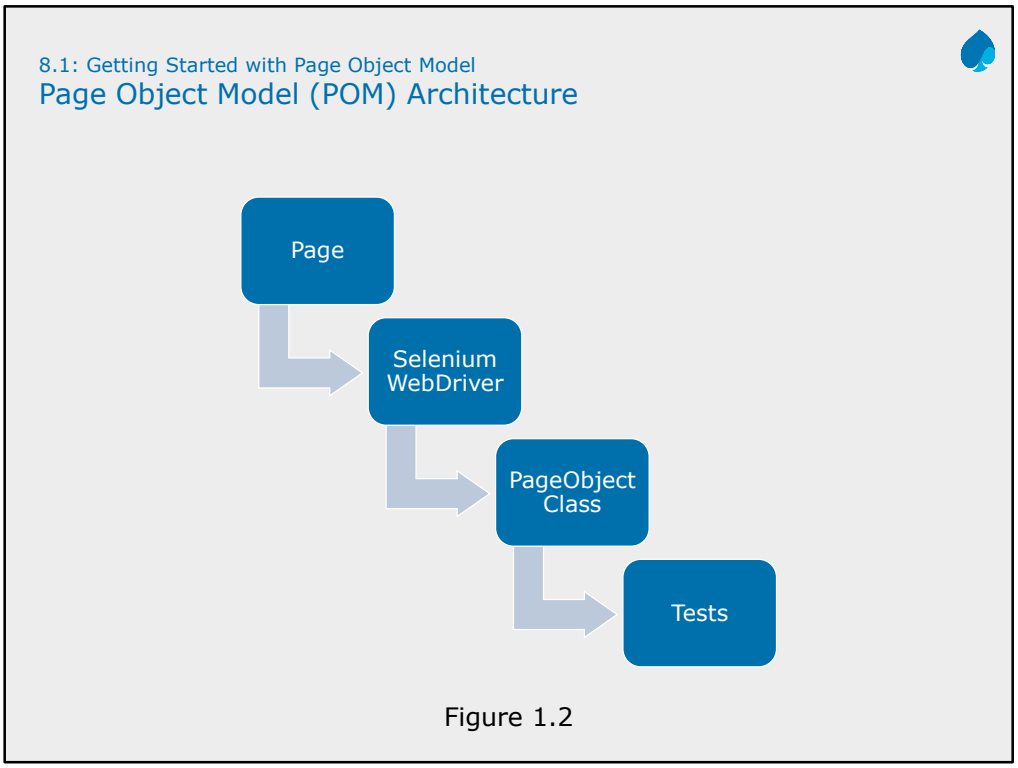

Figure 1.2

Page Object Model (POM) Architecture

In Figure 1.2, a clear flow is seen 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.

Demo – Implementing Page Object Model (POM)

- Simple POM Implementation

8.1: Getting Started with Page Object Model

Advantages of Page Object Model (POM)

- Increases Code Reusability
- Improves Code Maintainability
- Independent Object Repository
- Readability
- Supports Selenium Frameworks

Advantages of Page Object model (POM)
Page Object can be considered as offering the “services” which a page offers rather than exposing the details and UI structure of the page. It 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.

Increases Code Reusability: We could achieve code reusability by writing the code once and use it in different tests.

Improves Code Maintainability: POM facilitates clean separation between test code and page specific code such as UI element locators and layout which becomes very easy to maintain code. Change in code will take place only in Page Object Classes when a UI change occurs. It effectively enhances test maintenance and reduces code duplication.

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.

8.2: Overview of Design Patterns in Selenium Automation Testing

Overview of Selenium Design Patterns

- In software engineering, 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

8.2: Overview of Design Patterns in Selenium Automation Testing

Importance of Design Patterns in Selenium Automation Testing

- Implementing design patterns in test automation is neither mandatory nor necessary but as a seasoned automation test engineer one should understand the importance and advantages of implementing design patterns while automating tests.
- Advantages of Design Patterns in Selenium Automation Testing
  - Eliminates Duplicate Code
  - Introduces Locator Strategy
  - Design for Scalability

Importance of Design Patterns in Selenium Automation Testing
Implementing design patterns in test automation is neither mandatory nor necessary but as a seasoned automation test engineer one should understand the importance and advantages of implementing design patterns while automating tests.

Advantages of Design Patterns in Selenium Automation Testing
Eliminates Duplicate Code: One of the major advantages of design patterns in automation code is that it helps in eliminating duplication of code. The duplication of code is a common problem among software and automation development. Duplicate code tends to break the test. You need to update your code, then you need to find everywhere else it was copied and update it. Using Page Object Model (POM) design pattern you can refactor all the duplicate code.
Introduces Locator Strategy: UI element locators plays a very important role in Selenium tests. The Selenium web driver uses locators to find the elements on web pages. The most efficient and preferred way to locate an element on a web page is to use unique IDs. When there is no ID to use, the next preferred choices are NAME and CSS locators. The key is to have a clear locator strategy. Page Object Model (POM) introduces Object Repository to address the issue at a greater extent.
Design for Scalability: Selenium tests should be built to scale. Growing from small to an abundance of tests happens so quickly. The execution time increases and starts to hold up releases until all the tests complete. Executing hundreds of tests requires a bit more planning and design from the beginning.

Three Important Points for Scaling Test Automation:
1. It starts by developing automated tests to be small and modular (independent), which allows for faster feedback.
2. Continuous Integration.
3. Ability to run hundreds of tests in parallel.

Summary

In this lesson you have learnt:

- Challenges of traditional automation testing methods
- Approach/Solution to resolve challenges posed by traditional automation testing methods
- Need for Page Object Model (POM)
- Introduction to Page Object Model (POM)
- Overview of Selenium Design Patterns
- Importance of Design Patterns in Selenium

Review Question

Question 1:

- _____ is an optimized version of the Page Object Model Design pattern.

Question 2:
Page Object model is _____

- Design Pattern
- Unit level Framework
- Pattern Matching

Question 3:
What are the advantages of Page Object Model?

- Increase Code Reusability
- Improved code Maintainability
- Independent Object Repository
- All of the above

Review Question

Question 4:

- Design Patterns are _____ or _____ of how to solve a problem that can be used in many different situations.

Question 5:
Advantages of Design Pattern:

- Eliminate Duplicate Code
- Introduces Locator strategy
- Design for Scalability
- None of the above