Page Object Model(POM)

Press Space for next page →

Agenda

1. Page Object Model(POM)
2. Overview of POM
3. Basic Structure of POM
4. BasePage Class
5. HomePage Class
6. BaseTest Class
7. HomePageTest Class

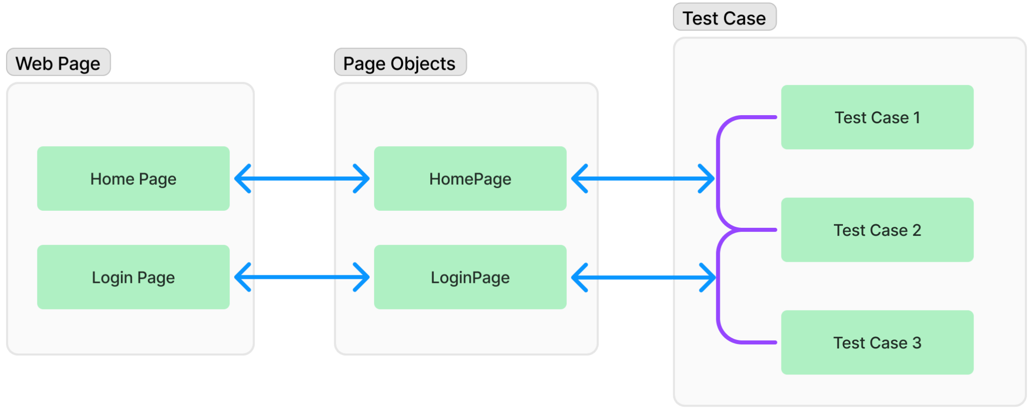
Page Object Model(POM)

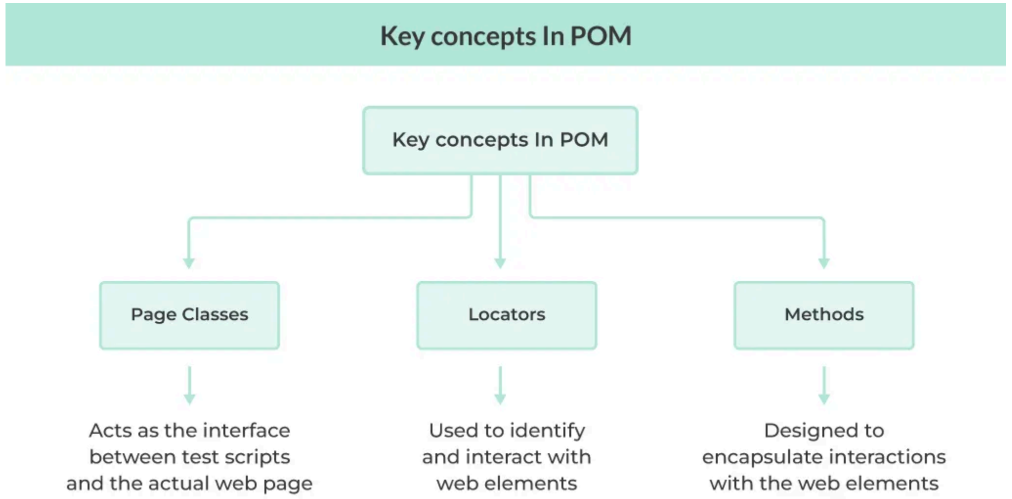
> Page Object Model is a design pattern to create Object Repository for web UI elements. Under this model, for each web page in the application, there should be corresponding page class. The page class will find the WebElements of that web page and also contains Page methods which perform operations on those WebElements.

Why POM?

- Maintainability
- Code Reusability
- Clarity and Readability
- Decoupling Test Logic from UI Logic

Overview of POM




Key concepts In POM
Key concepts In POM
Page Classes
Locators
Methods
Acts as the interface between test scripts and the actual web page
Used to identify and interact with web elements
Designed to encapsulate interactions with the web elements

Basic Structure of POM

```
src
├── main
│   └── java
│       └── pages
│           ├── BasePage.java
│           ├── HomePage.java
│           ├── LoginPage.java
│           └── ...
│       └── core
│           ├── Constants.java
│           ├── ResourceStrings.java
│           └── ...
│       └── utils
│           ├── ExcelReader.java
│           ├── ConfigReader.java
│           └── PDFReader.java
│
└── test
    └── java
        └── tests
            ├── BaseTest.java
            ├── LoginTest.java
            └── HomePageTest.java
            |── ...
```

BasePage Class

```java
class BasePage {
    protected WebDriver driver;
    private static final Logger logger = LogManager.getLogger(BasePage.class);

    public BasePage(WebDriver driver) {
        this.driver = driver;
    }

    public void sendKeys(WebElement element, String text) {
        element.sendKeys(text);
    }

    public void click(WebElement element) {
        element.click();
    }

    public String getText(WebElement element) {
        return element.getText();
    }

    public void clear(WebElement element) {
        element.clear();
    }
}
```

HomePage Class

```java
class HomePage extends BasePage {
    private static final Logger logger = LogManager.getLogger(HomePage.class);
    private final By accountButtonLocator = By.id("account");

    public HomePage(WebDriver driver) {
        super(driver);
    }

    public void clickAccountButton() {
        WebElement accountButton = driver.findElement(accountButtonLocator);
        logger.info("Clicking on Account Button");
        click(accountButton);
    }
}
```

BaseTest Class

```java
class BaseTest {
    protected WebDriver driver;
    private static final Logger logger = LogManager.getLogger(BaseTest.class);

    @BeforeMethod
    public void setUp() {
        driver = new ChromeDriver();
        logger.info("Driver is initialized");
        driver.manage().window().maximize();
        driver.get("https://example.com");
        logger.info("Navigated to the URL: https://example.com");
    }

    @AfterMethod
    public void tearDown() {
        driver.quit();
        logger.info("Driver is closed");
    }
}
```

HomePageTest Class

```java
class HomePageTest extends BaseTest {
    private static final Logger logger = LogManager.getLogger(HomePageTest.class);

    @Test
    public void testHomePage() {
        HomePage homePage = new HomePage(driver);
        homePage.clickAccountButton();
        logger.info("Clicked on Account Button");
    }
}
```

Thank you ❤️

qa-june-2024-automation-with-java-slides