Playwright Page Object Model (POM) Setup - Step-by-Step Guide

Step 1: Create Maven Project

- Set up a new Maven project using your IDE (e.g., IntelliJ, Eclipse).

- Use a meaningful name for the project (e.g., PlaywrightPOMFramework).

Step 2: Add Dependencies

- Add the required dependencies in the pom.xml file:

  - Playwright Java

  - TestNG

  - Apache POI (for Excel)

  - Extent Reports

  - Log4j

  - Gson / Jackson (if JSON parsing is needed)

Step 3: Project Structure - src/main/java

Create the following packages:

- base -> BasePage.java

- pages -> All your page classes (e.g., HomePage.java, LoginPage.java)

- utilities -> Utilities like TestConfig.java, MailSetup.java

- listeners -> Extent Report Listeners

- constants -> Constants.java for global/static values

Step 4: Project Resources - src/main/resources

- properties folder:

  - OR.properties -> Object Repository

  - log4j.properties -> Logging configuration

- Add log4j initialization code in your base/test classes.

Step 5: Test Code Structure - src/test/java

Create the following packages:

- base -> BaseTest.java
- testcases -> Test classes for all modules
- utilities -> Test utilities (e.g., DataProviderUtil.java)
- rough -> Temporary test/debug code

Step 6: Test Resources - src/test/resources

- logs -> Store generated logs
- properties -> test-config.properties etc.
- runner -> Store testng.xml
- testdata -> Store Excel files or JSON data

Step 7: Create BasePage.java

This is the core of the POM structure.

Example structure:

```
public class BasePage {
    private static FileInputStream fis;
    public BasePage() {
        // Load OR.properties here
    }
    // Setup all reusable keyword methods
}
```

Step 8: Create BaseTest.java

The heart of test execution.

Example structure:

```
public class BaseTest {
    // Playwright instance, Browser instance, Logger setup
    // Load config properties
    // @BeforeSuite -> Launch browser, config
    // @AfterSuite -> Tear down
}
```

Step 9: Global Constants

- Create Constants.java:
  - URLs, Excel paths, Sheet/Test names, Timeouts

Step 10: Page Classes

- Example: HomePage.java
  - Define locators and reusable methods

Step 11: Writing Test Cases

- Use TestNG in src/test/java/testcases
- Create page objects, call methods, use assertions

Step 12: Runner Setup

- Create testng.xml in src/test/resources/runner
- Include packages/classes to run
- Add ExtentListeners in <listeners> tag

Step 13: Data-Driven Testing Setup

- Use ExcelReader.java in utilities

- Create DataProviderUtil.java with @DataProvider

- Read test data using POI and pass to test cases