

Selenium Advanced and POM

Page Object Model



SoftUni Team
Technical Trainers



SoftUni



Software University

<https://softuni.bg>

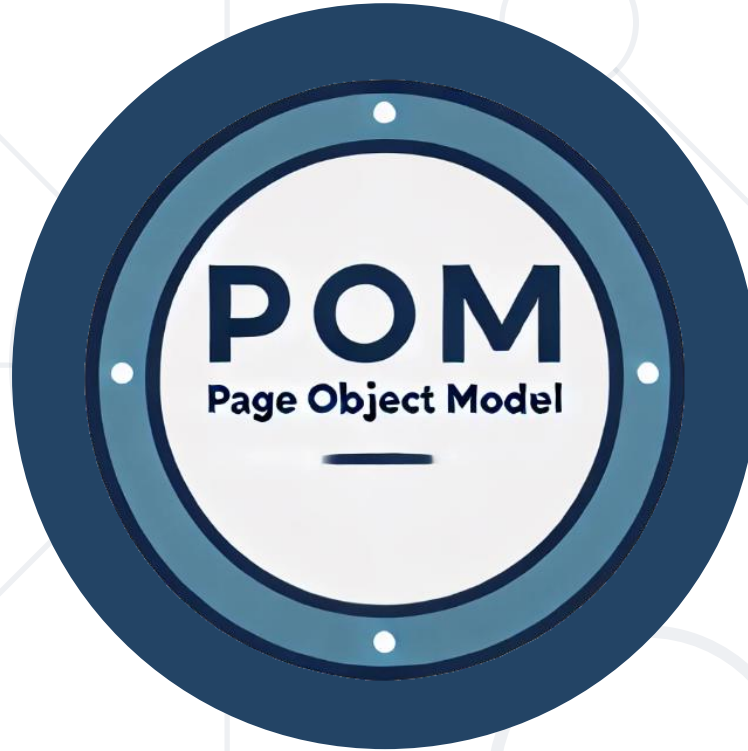
You Have Questions?

sli.do

#QA-FrontEnd

1. Page Object Model (POM) in Selenium
2. Creating Page Object Classes in C#
3. Unit Tests with Page Objects
4. Practice: Structuring Selenium Project with Page Objects





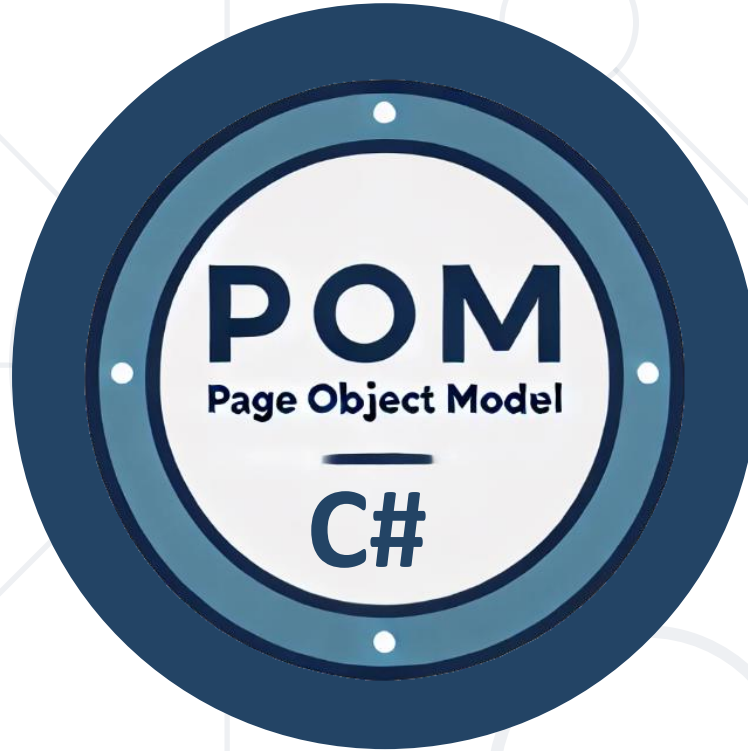
Page Object Model (POM)

Overview

- A **web application** is composed of **various web pages**
- Each **web page** consists of **multiple web elements**
- In web apps development, the **UI undergoes** several **changes**:
 - Adding **new web elements**
 - **Removing existing** web elements
 - **Modifying** HTML attributes of web elements
- These **changes** can **directly affect** the **test automation scripts**

- Need for **Abstraction Layer**:
 - To minimize the impact of UI changes on test automation scripts, an abstraction layer is needed
 - It separates web element information from the test automation scripts, making them more resilient to changes
- **Page Object Model (POM)** is a design pattern widely used in Selenium to create the abstraction layer
- Involves **creating separate classes (Page Objects)** for each web page
- Each **Page Object contains** the **locators** and **methods** to **interact** with the web elements on that page

- Benefits of Using POM:
 - **Decoupling:**
 - Separates the testing logic from the UI interaction
 - **Maintainability:**
 - Code is more readable and maintainable
 - **Reusability:**
 - Avoids duplicating code by reusing page elements and methods



Page Object Class in C#

Page Object Model (POM) – Example

- "sum-num" app is part of your resources

Sum Two Numbers

Number 1:

Number 2:

Sum: 2

Web Page


```
+ C# SumNumberPage.cs
SumNumberPage
  driver : IWebDriver
  SumNumberPage(IWebDriver)
  PageUrl : string
  FieldNum1 : IWebElement
  FieldNum2 : IWebElement
  ButtonCalc : IWebElement
  ButtonReset : IWebElement
  ElementResult : IWebElement
  OpenPage() : void
  ResetForm() : void
  IsFormEmpty() : bool
  AddNumbers(string, string) : string
```

Page Object Class


Creating a Page Object Class in C#













■ Page Object class declaration and configuration

```
public class SumNumberPage
{
    private readonly IWebDriver driver;

    1 reference |  1/1 passing
    public SumNumberPage(IWebDriver driver)
    {
        this.driver = driver;
        driver.Manage().Timeouts().
            ImplicitWait = TimeSpan.FromSeconds(2);
    }

    const string PageUrl =
        "https://your-forked-sum-two-numbers-page.replit.dev/";
}
```

 SumNumberPage

-  driver : IWebDriver
-  SumNumberPage(IWebDriver)
-  PageUrl : string
-  FieldNum1 : IWebElement
-  FieldNum2 : IWebElement
-  ButtonCalc : IWebElement
-  ButtonReset : IWebElement
-  ElementResult : IWebElement
-  OpenPage() : void
-  ResetForm() : void
-  IsFormEmpty() : bool
-  AddNumbers(string, string) : string

Creating a Page Object Class in C#

- **UI elements** are mapped to C# properties

2 references

```
public IWebElement FieldNum1 =>  
    driver.FindElement(By.CssSelector("input#number1"));
```

2 references

```
public IWebElement FieldNum2 =>  
    driver.FindElement(By.CssSelector("input#number2"));
```

1 reference

```
public IWebElement ButtonCalc =>  
    driver.FindElement(By.CssSelector("#calcButton"));
```

1 reference

```
public IWebElement ButtonReset =>  
    driver.FindElement(By.CssSelector("#resetButton"));
```

2 references

```
public IWebElement ElementResult =>  
    driver.FindElement(By.CssSelector("#result"));
```

```
SumNumberPage  
  driver : IWebDriver  
  SumNumberPage(IWebDriver)  
  PageUrl : string  
  FieldNum1 : IWebElement  
  FieldNum2 : IWebElement  
  ButtonCalc : IWebElement  
  ButtonReset : IWebElement  
  ElementResult : IWebElement  
  OpenPage() : void  
  ResetForm() : void  
  IsFormEmpty() : bool  
  AddNumbers(string, string) : string
```

Creating a Page Object Class in C#


- **Page actions** are mapped to C# methods









1 reference |  1/1 passing





```
public void OpenPage()
{
    driver.Navigate().GoToUrl(PageUrl);
}
```

```
public void ResetForm()
{
    ButtonReset.Click();
}
```

```
public bool IsFormEmpty()
{
    return FieldNum1.Text + FieldNum2.Text +
        ElementResult.Text == "";
}
```

 SumNumberPage

-  driver : IWebDriver
-  SumNumberPage(IWebDriver)
-  PageUrl : string
-  FieldNum1 : IWebElement
-  FieldNum2 : IWebElement
-  ButtonCalc : IWebElement
-  ButtonReset : IWebElement
-  ElementResult : IWebElement

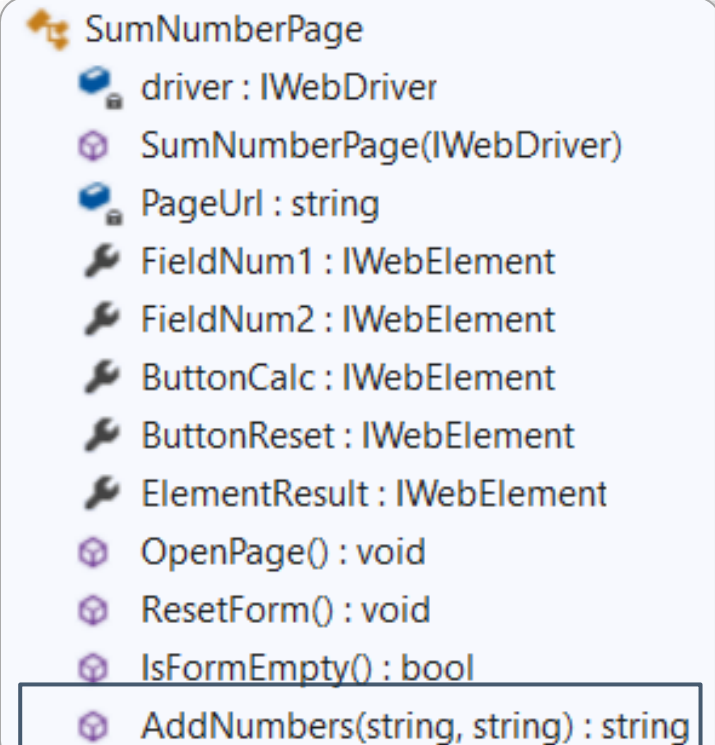
-  OpenPage() : void
-  ResetForm() : void
-  IsFormEmpty() : bool
-  AddNumbers(string, string) : string

Creating a Page Object Class in C#

- **Page actions** are mapped to C# methods

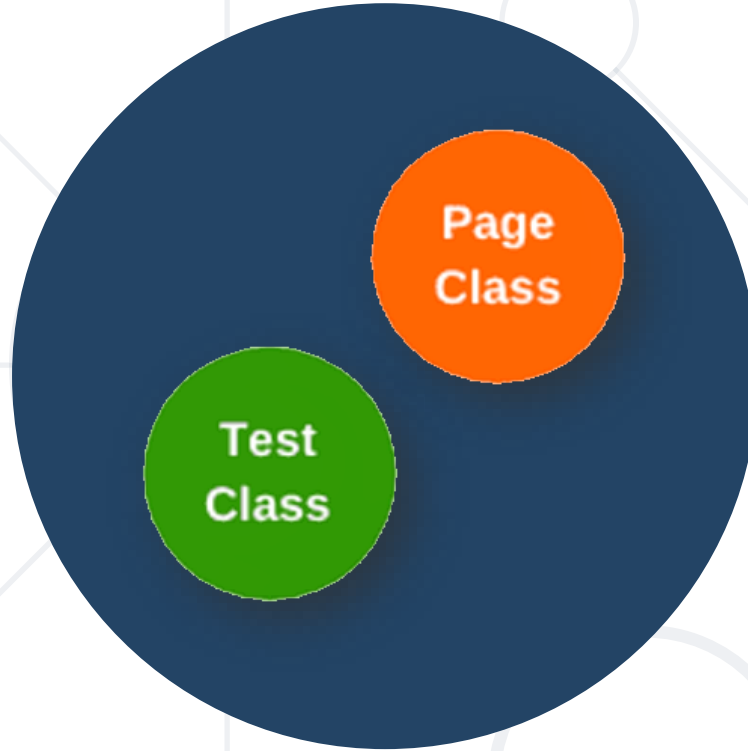
1 reference |  1/1 passing

```
public string AddNumbers(string num1, string num2)
{
    FieldNum1.SendKeys(num1);
    FieldNum2.SendKeys(num2);
    ButtonCalc.Click();
    string result = ElementResult.Text;
    return result;
}
```



SumNumberPage

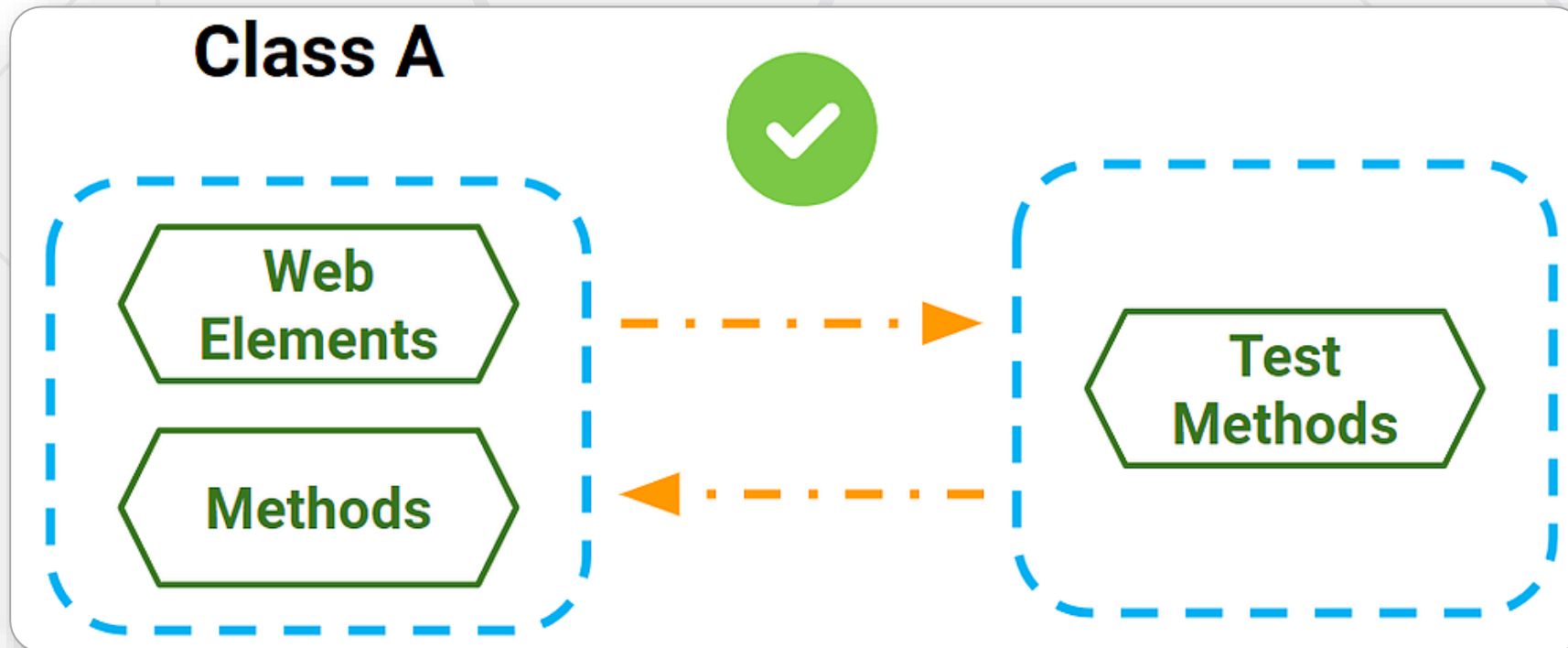
- driver : IWebDriver
- SumNumberPage(IWebDriver)
- PageUrl : string
- FieldNum1 : IWebElement
- FieldNum2 : IWebElement
- ButtonCalc : IWebElement
- ButtonReset : IWebElement
- ElementResult : IWebElement
- OpenPage() : void
- ResetForm() : void
- IsFormEmpty() : bool
- AddNumbers(string, string) : string



Unit Tests with Page Objects

Unit Tests with Page Objects

- Test methods interact with web elements through the methods defined in the Page Object class, providing a clear abstraction layer between the test logic and UI details



Example of a Test Method Using Page Object

[Test]

✓ | 0 references

```
public void Test_Valid_Numbers()
{
    var sumpage = new SumNumberPage(driver);
    sumpage.OpenPage();
    var result = sumpage.AddNumbers("5", "6");
    Assert.That(result, Is.EqualTo("Sum: 11"));
}
```

[Test]

✓ | 0 references

```
public void Test_AddTwoNumbers_Invalid()
{
    var sumpage = new SumNumberPage(driver);
    sumpage.OpenPage();
    string resultText = sumpage.AddNumbers("hello", "world");
    Assert.That(resultText, Is.EqualTo("Sum: invalid input"));
}
```

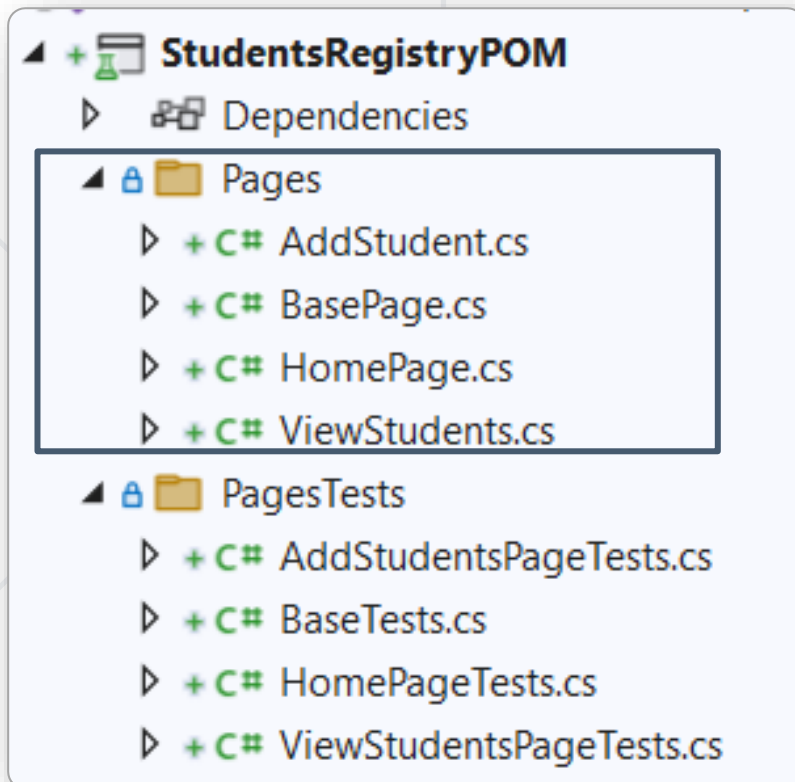



Practice

Structuring Selenium Project with Page Objects

Structuring Selenium Project with Page Objects

- Create a **base page class and page classes** for each page



[Home](#) | [View Students](#) | [Add Student](#)

Students Registry

Registered students: 3

[Home](#) | [View Students](#) | [Add Student](#)

Registered Students

- Marry (marry@gmail.com)
- Steve (steve@yahoo.com)
- Teddy (teddy@mail.ru)

[Home](#) | [View Students](#) | [Add Student](#)

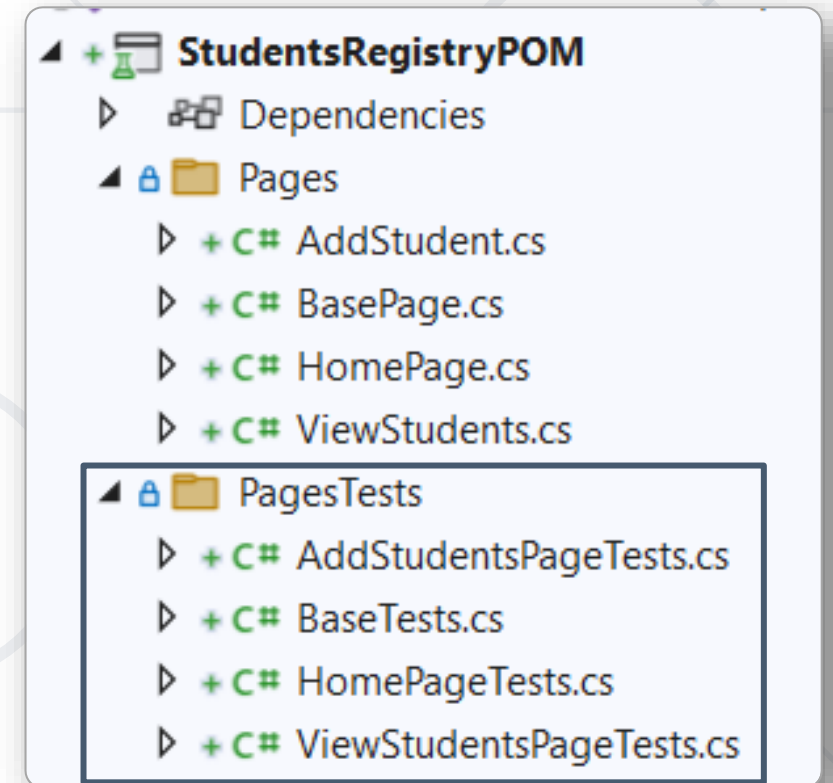
Register New Student

Name:

Email:

Structuring Selenium Project with Page Objects

- Create also a **test classes** for each page
 - BaseTests;
 - HomePageTests;
 - ViewStudentsPageTests;
 - AddStudentsPageTests;



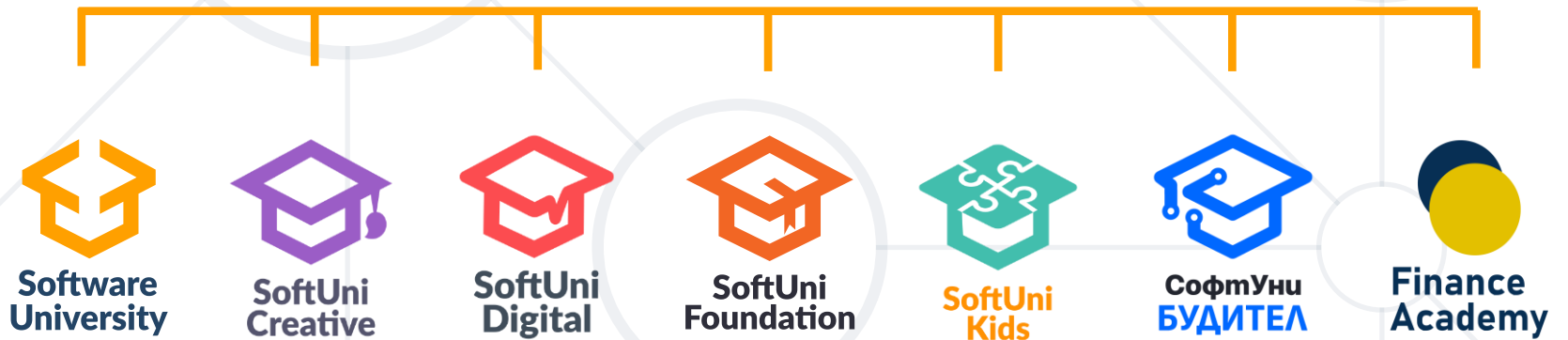
- **POM** - separating the web element information from test scripts
- **Implementing POM** involves creating:
 - **Page Classes** for each web page
 - **Using these classes in Test Classes**



Questions?



SoftUni



Diamond Partners



THE CROWN IS YOURS



- Software University – High-Quality Education, Profession and Job for Software Developers

- softuni.bg, about.softuni.bg

- Software University Foundation

- softuni.foundation

- Software University @ Facebook

- facebook.com/SoftwareUniversity



- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – <https://about.softuni.bg/>
- © Software University – <https://softuni.bg>

