





```
<h1>Counter</h1>
```

```
<p>Current count: @_currentCounter</p>
```

```
<button id="increment-button"  
        class="btn btn-primary"  
        @onclick="IncrementCount">Click me</button>
```

```
@code {  
    private int _currentCounter;  
  
    private void IncrementCount()  
    {  
        _currentCounter++;  
    }  
}
```

```
@using AngleSharp.Dom
```

```
@using Bunit
```

```
@using Xunit
```

```
@inherits Bunit.TestContext
```

```
@code {
```

```
    [Fact]
```

```
    public void IncrementCounterByOneWhenButtonClicked()
```

```
    {
```

```
        // Render our component including all lifecycle events
```

```
        var cut = Render(@<Counter />);
```

*// Find the button via CSS selector and click it*

```
WebElement button = cut.Find("button");
```

```
button.Click();
```

*// Assert the markup*

**var** pElement = cut.Find("p");

```
    pElement.MarkupMatches ( @<p>Current count: 1</p> );  
}  
}
```























































































































```
Element.TextContent.Should().Be("Current count: 1");
```

# Counter - Test

```
@using AngleSharp.Dom
@using Bunit
@using Xunit

@inherits Bunit.TestContext

@code {
    [Fact]
    public void IncrementCounterByOneWhenButtonClicked()
    {
        // Render our component including all lifecycle events
        var cut = Render(@<Counter />);

        // Find the button via CSS selector and click it
        IElement button = cut.Find("button");
        button.Click();

        // Assert the markup
        var pElement = cut.Find("p");
        pElement.TextContent.Should().Be("Current count: 1");
        pElement.MarkupMatches(@<p>Current count: 1</p>);
    }
}
```

```
<h1>Counter</h1>

<p>Current count: @_currentCounter</p>

<button id="increment-button"
        class="btn btn-primary"
        @onclick="IncrementCount">Click me</button>

@code {
    private int _currentCounter;

    private void IncrementCount()
    {
        _currentCounter++;
    }
}
```

➤ Using AngleSharp we can directly get the content of the node

# The beauty of razor syntax

Setting up parameter