


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

```
<p
```

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

```
@code {  
    private int _currentCounter { get; set; }  
  
    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.Click();
```

```
// Assert the markup
```

```
var pElement = cut.Find(“
```

```
pElement.MarkupMatches( @<p
```

```
”
```

```
);
```

>CurrentCount: @currentCounter</p>



































































































































>currentCount:1</p>);



























style

class = 'society'



"#incident-but-on";





style: ignore

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("#increment-button");
        button.Click();

        // Assert the markup
        var pElement = cut.Find(".some-text");
        pElement.MarkupMatches(@<p style:ignore>Current count: 1</p>);
    }
}
```

```
<h1>Counter</h1>

<p class="some-text">Current count: @_currentCounter</p>

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

@code {
    private int _currentCounter { get; set; }

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

➤ We can use any css selector - making our test more stable

➤ style:ignore helps us to ignore the “styling” aspect

I don't care about the markup?

Using Anglesharp