Lab – bUnit

Übungsdauer: 30 Minuten

Overview

In dieser Übung wird eine neue Blazor Web App erstellt und entsprechende bUnit Tests geschrieben.

Ziel

- bUnit Tests für Blazor Web App schreiben

Schritte

1. **bUnit Projekt und Test Projekt erstellen**
   a) Erstellt ein neues Blazor Web App Projekt in .NET 9
      a. Stellt den „Interactive render mode“ auf „Auto“
      b. Stellt die „Interactivity location“ auf „Per Page“
   b) Erstellt ein neues MS Test Projekt und fügt es der Solution hinzu

2. **Test: Counter Komponente MatchMarkup**
   c) Erstellt eine Test Klasse für die Counter Komponente
   d) Schreibt einen Testcase, die die „IncrementCount“ Methode der Counter Komponente testet
   e) In der Testmethode muss als erstes der Bunit TestContext erstellt werden

```
var context = new Bunit.TestContext();
```

   f) Dann muss die Komponente gerendert werden

```
var component = context.RenderComponent<Counter>();
```

   g) Der Button Click soll simuliert werden

```
component.Find("button").Click();
```

   h) Und eine Behauptung wird aufgestellt und überprüft

```
component.Find("p").MarkupMatches("<p> Current count: 1 </p>");
```

3. **Test: Home Komponente HasComponent**
   i) Erstellt eine Test Klasse für die Home Komponente
   j) Bindet in der Home Komponente die Counter Komponente ein
   k) Testet mithilfe von „component.HasComponent“, dass die Counter Komponente in der Home Komponente gerendert wird

4. **Test: Injection und FindComponents.Count**
   l) Erstellt eine neue Komponente, Modell Klasse, ein Service und eine Service Schnittstelle, die in der Counter Komponente injected wird

a. z.B. IPersonService, PersonService, Person und PersonComponent
b. Die PersonComponent soll nur den Namen der Person anzeigen

```
@using BUnitTest.Client.Models
<h3>PersonComponent</h3>
<p>@($"{Person.FirstName} {Person.LastName}")</p>


@code {
    [Parameter]
    public Person Person { get; set; }
}
```

m) Definiert eine Methode GetPeople(), die eine Liste von Personen zurückgibt
n) In der Counter Komponente soll eine OnInitialized() Methode implementiert werden, die GetPeople() aus dem Service aufruft
o) Für jede geladene Person soll eine PersonComponent gerendert werden
p) Benutzt Moq, um den PersonService zu mocken
    a. Konfiguriert mit Setup eine Rückgabe für die GetPeople() Methode
q) Überprüft, dass die Counter Komponente abhängig vom gemockten Service die richtige Anzahl an PersonComponents rendert

```
Assert.AreEqual(2, component.FindComponents<PersonComponent>().Count);
```