DEVware
buero fuer software-entwicklung

Lab Blazor CRUD

Übungsdauer : 60 Minuten

Overview

In dieser Übung wird eine CRUD Situation für eine ToDo -Entität erstellt. Der REST Service-Endpunkt ist in der Solution bereits enthalten, sowie einige Hilfs-Konstrukte im Blazor Client. Es wird die Auflistung der ToDo´s implementiert als Blazor Komponente mit einer weiteren ToDo-Detail-Komponente.

Ziel

- Blazor Komponente
- DataBinding

Schritte

- **Öffnen sie die Solution**
  DemoBlazorCRUD\Before\DemoToDoApp.sln

1. Navigations Punkt anlegen
   - Öffnen sie die Datei „NavMenu.cshtml“ in dem Ordner „Shared“
   - Suchen sie die Stelle der <li> Auflistung um ein weiteren Menüpunkt einzufügen:

```
<li class="nav-item px-3">
</li>
```

   - In demli-Tag wird nun ein NavLink mit der zukünftigen URL Adresse der To Do Auflistung eingefügt:

```
<NavLink class="nav-link" href="/todoes">
  <span class="oi oi-list-rich" aria-hidden="true"></span> ToDoes
</NavLink>
```

2. ToDo-ViewModel erstellen
   - Fügen sie in dem Ordner Pages eine C#-Klassen Datei mit dem Namen ToDoesViewModel.cs ein
   - Leiten sie die Klasse von BlazorComponent ab

```
public class ToDoesViewModel : BlazorComponent
{
}
```

   - Für das DateBinding im View wird eine Auflistungs-Eigenschaft aller ToDoes benötigt. Fügen sie dazu eine Eigenschaft für die ToDoes ein:

```
public ToDo[] todoes { get; set; }
```

- Außerdem soll die Komponente mit der vorbereiteten ToDoClientRepository Serviceklasse die ToDoes laden. Dazu wird die Service Komponente mit Property Injection bereitgestellt:

```
[Inject]
protected IToDoClientRepository repository { get; set; }
```

- Nun fügen sie eine Überladung der Methode „OnInitAsync“ ein, um in der Initialisierung der Komponente die ToDos zu laden:

```
protected override async Task OnInitAsync()
{
}
```

- Fügen sie noch den Aufruf zu dem Repository ein, um die ToDo´s zu laden und in der eigenen Eigenschaft zu halten:

```
await repository.RefreshDataAsync();
todoes = repository.Data.ToArray();
```

3. ToDo-View erstellen
   - Fügen sie in dem Ordner Pages eine Razor-View Datei mit dem Namen ToDoes.cshtml ein
   - Zuerst geben sie am Anfang der die Direktiven für das Page-Routing und die Angabe der ViewModel Klasse an:

```
@using DemoToDoApp.Shared
@page "/todoes"
@inherits ToDoesViewModel
```

   - Nun bauen sie dem Rumpf zu anzeige der Liste mit einer Verzweigung, um das laden der ToDoes zu signalisieren:

```
<h1>ToDo-List</h1>

@if (todoes == null)
{
    <p><em>Loading...</em></p>
}
else
{
}
```

   - In dem Else-Zweig fügen sie nun eine Zeile ein, um den Create Knopf anzubieten:

```
<div class="row mb-3">
        <a href="/CreateToDo" class="btn btn-primary btn-sm">Neu</a>
</div>
```

   - Darunter fügen sie eine weitere Zeile ein, um die Liste der ToDoes mit einer ToDo-Detail-Komponente aufzulisten:

```
<div class="row">
   @foreach (ToDo myItem in todoes)
   {
      <ToDoItem CurrentToDo="@myItem" />
   }
</div>
```

4. ToDo-View erstellen
   - Fügen sie in dem Ordner Components eine Razor-View Datei mit dem Namen ToDoItem.cshtml ein
   - Fügen sie zuerst ein using für die Entität ein:

```
@using DemoToDoApp.Shared
```

   - Ein ToDo soll als Bootstrap 4 Karte dargestellt werden, fügen sie dazu einen Html – Rumpf wie folgt ein:

```
<div class="card bg-light m-3" style="width: 18rem;">
    <div class="card-header">
    </div>
    <div class="card-body">
    </div>
</div>
```

   - Fügen sie einen C# Function Block in die Datei ein, um ein ToDo-Objekt als Parameter anzuzeigen:

```
@functions {

[Parameter]
private ToDo CurrentToDo { get; set; }

}
```

   - Fügen sie nun unter Verwendung der Eigenschaft CurrentToDo den Header eines ToDoes ein:

```
@CurrentToDo.Date.ToString("dd.MM.yyyy")
```

   - Fügen sie nun den Body eines ToDoes ein:

```
<h5 class="card-title">@CurrentToDo.ToString()</h5>
<p class="card-text">@CurrentToDo.Description</p>
```

   - Fügen sie noch für die Bearbeitung und das Löschen eines ToDo-Elements die Knöpfe mit den entsprechenden Links ein:

```
<a href="/EditToDo/@CurrentToDo.ID" class="btn btn-primary btn-
sm">Edit</a>
<a href="/DeleteToDo/@CurrentToDo.ID" class="btn btn-primary btn-
sm">Delete</a>
```

   - Testen sie die Applikation