# Modul Blazor

# Lab: Page Title Render Componente

Ziel Komponente erstellen die den Page Titel ändert. Dazu wird eine ViewModel Klasse erstellt, die per Event im UI Kontext Statehaschanged aufruft. Für den reinen HTML Part wir eine Klasse mit Renderer genutzt

Zeitdauer 20 Minuten.

Erstellen Sie im Visual Studio Projekt eine Klasse als Page Titel Mini View Model mit dem Namen pagetitlevm.cs

Tippen sie propfull [TAB] [TAB]

Navigieren Sie mit der Tab Taste durch die drei hervorgehobenen Platzhalter und schreiben sie string, \_title und Titel.

public class PageTitelVM

{

private string \_title;

public string Titel

{

get { return \_title; }

set { \_title = value;

}

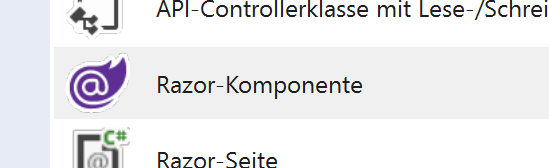
}

}

In Startup.cs wird die Klasse im DI Service Collection per Addscoped hinzugefügt. Methode ConfigureServices. Ergänzen sie ggf den nötigen Namespace per using wenn PageTitelVM nicht erkannt wird.

services.AddScoped<PageTitelVM>();

Erstellen sie im Pages Verzeichnis eine Rlazor Komponente und damit eigentlich eine Blazor Page.



Fügen sie die Page Route hinzu z.B. für /t und referenzieren per DI Injection die VM Klasse. Die Eigenschaft Titel kann zwei Werte annehmen. Einmal direkt nach dem Laden und das andere Mal wenn der Benutzer den Button klickt und damit das Change Event auslöst. Zur Kontrolle wird die Titel Eigenschaft per @ im Browser ausgegeben.

@page "/t"

@using BlazorSchulungApril.Data

@inject PageTitelVM pagetitel

<button **@onclick**="change">titel</button>

@pagetitel.Titel

@code {

protected override async Task OnInitializedAsync()

{

pagetitel.Titel = "aaaaaa";

}

void change()

{

pagetitel.Titel = "bbbbb";

}

}

Starten Sie die Anwendung in Visual Studio mit F5 und verifizieren Sie die Funktion im Browser.

Als nächstes wird eine Custom Komponent hinzugefügt. Erstellen sei eine Klasse TitelRender.cs. Implementieren sie das Interface ComponentBase.

Nutzen sie in Visual Studio das Schraubenzieher Symbol links um die Überschreibung für die BuildRenderTree zu generieren



Um den Wert des Titels aus dem Viewmodel Objekt zu erhalten, wird die Referenz inijiziert.

Ergänzen Sie den Code im Rendertree so das zur Laufzeit <titel>wert</titel> gerendert wird.

public class TitelRender:ComponentBase

{

[Inject] public PageTitelVM vm { get; set; }

protected override void BuildRenderTree(RenderTreeBuilder builder)

{

base.BuildRenderTree(builder);

builder.OpenElement(0, "title");

builder.AddContent(1, vm.Titel);

builder.CloseElement();

}

}

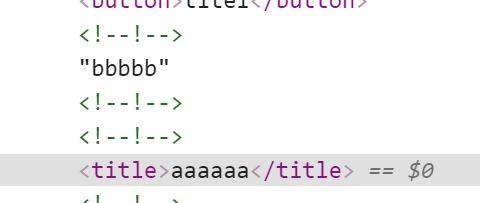
Fügen sie die Titel Componente in ihre Page unterhalb des Buttons aber vor @ code in der Form <TitelRender></TitleRender> ein. Visual Studio Editor erkennt dies, schlägt per Intellisense vor und färbt das Element lila



Starten Sie die Anwendung mit F5 und öffnen per F12 die Browser Developer Tools. Beachten Sie die Ausgabe im DOM nach dem Button



Drücken Sie den Button und beachten den Unterschied von Anzeige und dem rendering der Title Komponente



Um der Komponente mitzuteilen, das sich die gebunden Daten geändert haben und damit ein rerendering auszulösen muss ein Event implementiert werden. Ergänzen sie in der Viewmodel Klasse ein Event UpdateUI. Das Event wird im Setter des Title Propertys ausgelöst per Invoke.

UpdateUI?.Invoke(this, EventArgs.Empty);

}

}

public event EventHandler UpdateUI;

Das Event wird in der Component TitleRenderer.cs abonniert. Die Datenklasse (VM) wirft das Event, das im UI Renderer sozusagen gefangen wird und dort wiederum StateHasChanged aufruft. Um dies im Renderer Context zu erledigen, innerhalb einer InvokeAsync Dispatcher Methode von componentbase.

protected override async Task OnInitializedAsync()

{

vm.UpdateUI += (\_, \_\_) => InvokeAsync(StateHasChanged);

}

Die Syntax mit den beiden unbenannten Parametern dient nur der Erheiterung. Auch \_ und \_\_ sind Namen.

Die Render Komponente wird als nächstes im HTML Urprung der Web Anwendung eingebunden. Editieren sie im Pages Verzeichnis die Datei \_hosts.cshtml

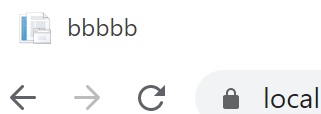
<meta name="viewport" content="width=device-width, initial-scale=1.0" />

@(await Html.RenderComponentAsync<TitelRender>( RenderMode.ServerPrerendered))

<base href="~/" />

Achten Sie auf den Unterschied zur eigentlich App mit dem Component HTML Tag Helper im Vergleich zum verwendeten HTML Helper. Beides ist funktionell ident.

Starten Sie die Anwendung mit F5 und verifzieren sie die Funkion speziell im Hinblick auf den Page Title im Browser



Hannes Preishuber 22.4.2020