

Blazor Tips & Tricks

Dal RenderFragment ai Generics, tips for Blazor developers

Andrea Dottor

Microsoft MVP Developer Technologies

andrea@dottor.net | @dottor



Blazor
Conference
2021

RenderFragment

"Represents a segment of UI content, implemented as a delegate that writes the content to a RenderTreeBuilder."

Delegato che permette di renderizzare una porzione di UI

Utilizzato per far passare al componente un template da utilizzare

RenderTreeBuilder

"Provides methods for building a collection of RenderTreeFrame entries."

Fornisce metodi per la gestione dell'output dei componenti blazor.

Non si tratta di qualcosa di nuovo: i file RazorComponent (*.razor) vengono trasformati in classi che usano il RenderTreeBuilder

- obj/Debug/net5.0/Razor/..

```

...[Microsoft.AspNetCore.Components.RouteAttribute("/Sample1")]
...public partial class Sample1 : Microsoft.AspNetCore.Components.ComponentBase
...{
...    #pragma warning disable 1998
...    protected override void BuildRenderTree(Microsoft.AspNetCore.Components.Rendering.RenderTreeBuilder __builder)
...    {
...        __builder.AddMarkupContent(0, "<h3>HTML dinamico con RenderFragment</h3>\r\n\r\n");
...        __builder.OpenElement(1, "div");
...        __builder.AddAttribute(2, "class", "card");
...        __builder.OpenElement(3, "div");
...        __builder.AddAttribute(4, "class", "card-body");
...        __builder.AddContent(5,
#nullable restore
#line 7 "C:\Users\andre\Source\Repos\XE.Dottor.Demo.BlazorWebApp\BlazorConference2021\Dottor.BlazorTips\Pages\Sample1.razor"
...        CreateTitle()
...
#line default
#line hidden
#nullable disable
...    );
...    __builder.AddMarkupContent(6, "\r\n");
...    __builder.AddMarkupContent(7, "<h6 class='card-subtitle mb-2 text-muted'>Andrea Dottor</h6>\r\n");
...    __builder.AddMarkupContent(8, "<p class='card-text'>Dal RenderFragment ai Generics, tips for Blazor");
...    __builder.OpenElement(9, "button");
...    __builder.AddAttribute(10, "class", "btn btn-primary");
...    __builder.AddAttribute(11, "onclick", Microsoft.AspNetCore.Components.EventCallback.Factory.Create<Mic
#nullable restore

```

```

    ...}
    ...);
    ...}
    ...#pragma warning restore 1998
    ...[global::Microsoft.AspNetCore.Components.InjectAttribute] private DashboardService DashboardService { get;
    ...}
}

namespace __Blazor.Dottor.BlazorTips.Pages.Sample4
{
    ...#line hidden
    ...internal static class TypeInference
    ...{
    ...    public static void CreateGridRepeater_0<TItem>(global::Microsoft.AspNetCore.Components.Rendering.RenderTree
    ...    {
    ...        __builder.OpenComponent<global::Dottor.BlazorTips.Components.Sample4.GridRepeater<TItem>>(seq);
    ...        __builder.AddAttribute(__seq0, "Data", __arg0);
    ...        __builder.AddAttribute(__seq1, "Columns", __arg1);
    ...        __builder.AddAttribute(__seq2, "ChildContent", __arg2);
    ...        __builder.CloseComponent();
    ...    }
    ...}
}

#pragma warning restore 1591

```

Creazione di codice dinamico

RenderTreeBuilder.**OpenElement**

- Apertura di un tag html

RenderTreeBuilder.**OpenComponent**

- Apertura di un tag (che punta ad uno specifico componente Blazor)

RenderTreeBuilder.**AddAttribute**

- Aggiunta di un attributo all'elemento precedentemente aperto

Alcune direttive da conoscere

Generics:

- Utilizzare la direttiva **@typeparam** per specificare il tipo, esattamente come nei *Generics* di C#

Ereditarietà:

- Utilizzare la direttiva **@inherits** per specificare la classe da cui ereditare
 - La classe base deve ereditare a sua volta da *ComponentBase*
- Permette di centralizzare la logica usata da più componenti

Demo

Blazor
Conference
2021

DynamicComponent

Introdotta con le novità di ASP.NET Core in .NET 6 Preview 1

Possibilità di includere dinamicamente un componente specificando il tipo e gli eventuali parametri

```
<DynamicComponent Type="@someType" Parameters="@myDictionaryOfParameters" />
```

```
@code
```

```
{
```

```
    Type someType = ...
```

```
    IDictionary<string, object> myDictionaryOfParameters = ...
```

```
}
```

RenderTreeBuilder - Sequence Number

Why sequence numbers should relate to code line numbers, not execution order

- <https://gist.github.com/SteveSandersonMS/ec232992c2446ab9a0059dd0fbc5d0c3>

ASP.NET Core Blazor advanced scenarios

- <https://docs.microsoft.com/en-us/aspnet/core/blazor/advanced-scenarios?view=aspnetcore-5.0>

Citazioni di Steve Sanderson in merito al *sequence*:

*A key example of this are sequence numbers. These indicate to the runtime which outputs came from which distinct and ordered lines of code. **The runtime uses this information to generate efficient tree diffs in linear time, which is far faster than is normally possible for a general tree diff algorithm.***

*Unlike .jsx files, .razor/.cshtml files are always compiled. This is potentially a great advantage for .razor, because **we can use the compile step to inject information that makes things better or faster at runtime.***

Grazie!

.NET in pillole

|  DEV.is.it



	IdentityServer4, non tutti sanno che...	04 Apr	10:36
	Scopriamo le micro/minimal API di ASP.NET 6	28 Mar	10:32
	Novità per ASP.NET Core in .NET 6 (preview 2)	21 Mar	12:43
	Architettura a Microservizi...non per tutti	14 Mar	11:23
	CQRS, CQS - facciamo chiarezza e sfatiamo alcuni ...	07 Mar	14:08
	Clean Architecture Solution Template	28 Feb	13:02
			

@dottor
andrea@dottor.net
www.dottor.net



Slide e materiale su
<https://aspit.co/BlazorConf-21>

Blazor
Conference
2021