.NET 8 Blazor New Rendering & Interactivity Modes

Imran Younas
Consultant | Solution Architect | Developer
4 articles + Follow

October 23, 2023

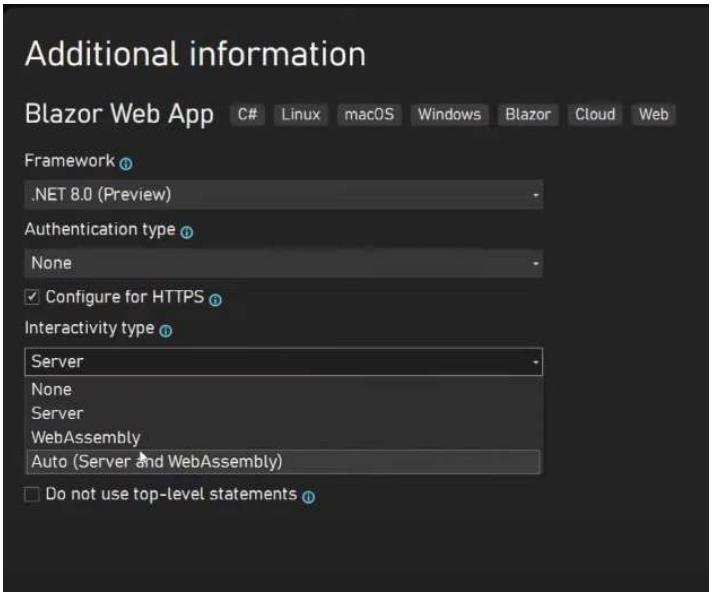
Blazor is a web framework that allows developers to build interactive web apps using C#.NET instead of JavaScript. What differentiates Blazor from MVC Razor pages is, its consistent (& shareable) component based programming model with ability to make selectively some components statically rendered and some with dynamically interactive in a single page. Thus it makes a truly modern web & mobile apps (with .NET MAUI Blazor Hybrid) development platform.

Microsoft has added some new features to Blazor with .NET 8 release, like Static Server Rendering , Progressive Enhancements (e.g. Enhanced Navigation & Form Handling) Streaming Rendering and ability to add interactivity per component basis, using Blazor Server or WebAssembly.

And to provide these all new features, Visual Studio has now new project template called **Balzor Web App**

This new Blazor Web App project template in Visual Studio provides four types of Blazor **Interactivity Types**:

1. None
2. Server
3. WebAssembly
4. Auto



.NET 8 Blazor Project - Interactivity Types

Here's a brief overview of each type:

1. **None**: This option creates a basic Blazor app with no client-side interactivity and is called 'Static Server Side Rendering'. It's suitable for apps that deal with a lot of static data.

2. **Server**: This option creates a Blazor app that uses 'Server Side Rendering'. Server side rendered page components in Blazor are going to be processed and rendered on the web server, once all operations to collect data and execute logic have completed, then the HTML produced will be sent to the browser to be rendered. For any interactivity SignalR will be used to communicate with Server and resultant JSON will be automatically rendered.

3. **WebAssembly**: This option creates a Blazor app that uses client-side rendering using WebAssembly. In this mode, the entire application is downloaded to the client's browser and executed there. This makes it an excellent choice for apps that require high interactivity and responsiveness.

4. **Auto**: This option creates a Blazor app that uses both server-side and client-side rendering using WebAssembly, and ability to switch them at runtime too. Initially, the app is rendered on the server, but subsequent visits after the Blazor bundle is downloaded will use client-side rendering using WebAssembly

With the above interactivity types we also need to select '**Interactivity Location**' , and here we have two options:

1. Per page/component
2. Global

So for example if we select interactivity type 'Server ' & interactivity location as 'Global', then Blazor app will run as Blazor Server App (like we had earlier in .NET 7), providing interactivity using WebSockets/SignalR.

And similarly if we select interactivity type 'WebAssembly ' & interactivity location as 'Global' then Blazor app will run as ASP.NET Core hosted Blazor WebAssembly app.

I hope this would help you get started with new .NET Blazor project template. Happy coding !

Published by

Imran Younas
Consultant | Solution Architect | Developer
Published • 3mo

Microsoft with release of .NET 8 (currently RC2) has added some new rendering & interactivity modes to Blazor. Here is a brief introduction to these new modes.
#blazor #microsoft #dotnet #webapp

Imran Younas
Consultant | Solution Architect | Developer

+ Follow

More from Imran Younas

DTO / JSON Payload ExpressJS Validation Middleware

Imran Younas on LinkedIn

Blazor WebAssembly is here !

Imran Younas on LinkedIn

Setting up & Publishing C#.NET Core Web Application on Linux

Imran Younas on LinkedIn