**Blazor Important Interview Questions**

**1. Blazor(Component) Lifecycle Events?**

<https://www.c-sharpcorner.com/article/blazor-life-cycle-events-oversimplified/>

[https://www.c-sharpcorner.com/article/blazor-interview-questions/](11) Blazor Imp Interview Questions.docx)

[https://chatgpt.com/c/e8a88c9c-2c72-4643-aab9-5cdbe5a1b584](11) Blazor Imp Interview Questions.docx)

****1. What is Blazor?****

Blazor is a web framework developed by Microsoft that allows developers to build interactive web applications using C# and .NET instead of JavaScript. It supports both server-side (Blazor Server) and client-side (Blazor WebAssembly) hosting models.

1. ****What are the differences between Blazor Server and Blazor WebAssembly?****

****Blazor Server****: Runs on the server. The UI updates and event handling are performed over a SignalR connection. It offers faster load times but requires a constant connection to the server.

****Blazor WebAssembly****: Runs in the browser on WebAssembly. It loads the .NET runtime and application into the browser, enabling offline capabilities and less server dependency but with potentially longer load times.

**1** **What is Blazor?**

* **Answer:** Blazor is a framework for building interactive web UIs using C# instead of JavaScript. It enables full-stack web development with .NET by providing a client-side framework called Blazor WebAssembly and a server-side framework called Blazor Server.

**2** **Explain the difference between Blazor Server and Blazor WebAssembly.**

* **Answer:** Blazor Server runs on the server and handles UI updates over a SignalR connection, providing fast load times and access to server resources. Blazor WebAssembly runs client-side in the browser, using WebAssembly to execute .NET code directly in the browser, providing offline capabilities and reducing server load.

**3** **What is WebAssembly?**

* **Answer:** WebAssembly (Wasm) is a binary instruction format that allows code to run in web browsers at near-native speed. Blazor WebAssembly uses WebAssembly to run .NET code directly in the browser.

**4** **What is the role of SignalR in Blazor Server?**

* **Answer:** SignalR is used in Blazor Server to manage real-time communication between the client and the server. It enables the server to push updates to the client UI, making it possible to maintain an interactive application state.

**5** **How does Blazor handle dependency injection?**

* **Answer:** Blazor supports dependency injection, allowing services to be registered and injected into components and services. You can register services in the Startup.cs file using methods like AddSingleton, AddScoped, and AddTransient.

**6** **What are Blazor components, and how do you create one?**

* **Answer:** Blazor components are the building blocks of a Blazor application. They are reusable UI elements composed of C#, HTML, and CSS. You create a component by creating a .razor file and defining the component logic and markup within it.

**7** **How does Blazor handle state management?**

* **Answer:** Blazor provides several ways to manage state, including cascading parameters, dependency injection, local storage, and session storage. For more complex state management, you can use libraries like Fluxor or Redux.

**8** **What are the lifecycle methods of a Blazor component?**

* **Answer:** Blazor components have several lifecycle methods:
  + OnInitialized / OnInitializedAsync
  + OnParametersSet / OnParametersSetAsync
  + OnAfterRender / OnAfterRenderAsync
  + Dispose / DisposeAsync

**9** **How can you call JavaScript functions from Blazor?**

* **Answer:** You can call JavaScript functions from Blazor using JavaScript interop. Blazor provides the IJSRuntime interface for this purpose. You can use InvokeAsync to call JavaScript functions asynchronously.

**10** **What are RenderFragments and how are they used in Blazor?**

* **Answer:** RenderFragments are a way to define and pass a segment of UI content to a component. They are useful for creating templated components where the content can be defined by the parent component.

**11** **Explain how Blazor handles form validation.**

* **Answer:** Blazor provides built-in support for form validation using data annotations. You can decorate your model properties with validation attributes and use the EditForm component along with DataAnnotationsValidator to automatically validate form inputs.

**12** **How do you optimize the performance of a Blazor application?**

* **Answer:** To optimize Blazor application performance, you can:
  + Minimize JavaScript interop calls
  + Use asynchronous methods
  + Reduce the size of the Blazor WebAssembly app by trimming unused assemblies
  + Use Blazor Server for applications that require fast load times
  + Optimize component rendering and avoid unnecessary re-renders

**13** **How do you handle file uploads in Blazor?**

* **Answer:** File uploads in Blazor can be handled using the InputFile component. You can access the uploaded files through the files property and process them as needed.

**14** **Can Blazor be used with existing JavaScript libraries or frameworks?**

* **Answer:** Yes, Blazor can interoperate with existing JavaScript libraries or frameworks using JavaScript interop. This allows you to leverage existing JavaScript functionality while building your Blazor application.

**15** **How do you implement authentication and authorization in a Blazor application?**

* **Answer:** Authentication and authorization in Blazor can be implemented using ASP.NET Core Identity, JWT tokens, or third-party authentication providers. Blazor provides built-in support for authentication and authorization through components like AuthorizeView and services like AuthenticationStateProvider.