ASP.NET Core 2.1

Version

2.2 Preview 2

2.1

2.0

1.1

1.0

Introduction to ASP.NET Core SignalR

04/25/2018 2 minutes to read Contributors

In this article

What is SignalR?

ASP.NET Core SignalR is an open-source library that simplifies adding real-time web functionality to apps. Real-time web functionality enables server-side code to push content to clients instantly.

Good candidates for SignalR:

- Apps that require high frequency updates from the server. Examples are gaming, social networks, voting, auction, maps, and GPS apps.
- Dashboards and monitoring apps. Examples include company dashboards, instant sales updates, or travel alerts.
- Collaborative apps. Whiteboard apps and team meeting software are examples of collaborative apps.
- Apps that require notifications. Social networks, email, chat, games, travel alerts, and many other apps use notifications.

SignalR provides an API for creating server-to-client remote procedure calls (RPC). The RPCs call JavaScript functions on clients from server-side .NET Core code.

Here are some features of SignalR for ASP.NET Core:

- Handles connection management automatically.
- Sends messages to all connected clients simultaneously. For example, a chat room.
- Sends messages to specific clients or groups of clients.
- Scales to handle increasing traffic.

The source is hosted in a SignalR repository on GitHub.

Transports

SignalR supports several techniques for handling real-time communications:

- WebSockets
- Server-Sent Events
- Long Polling

SignalR automatically chooses the best transport method that is within the capabilities of the server and client.

Hubs

SignalR uses *hubs* to communicate between clients and servers.

A hub is a high-level pipeline that allows a client and server to call methods on each other. SignalR handles the dispatching across machine boundaries automatically, allowing clients to call methods on the server and vice versa. You can pass strongly-typed parameters to methods, which enables model binding. SignalR provides two built-in hub protocols: a text protocol based on JSON and a binary protocol based on MessagePack. MessagePack generally creates smaller messages compared to JSON. Older browsers must support XHR level 2 to provide MessagePack protocol support.

Hubs call client-side code by sending messages that contain the name and parameters of the client-side method. Objects sent as method parameters are deserialized using the configured protocol. The client tries to match the name to a method in the client-side code. When the client finds a match, it calls the method and passes to it the deserialized parameter data.

Additional resources

- Get started with SignalR for ASP.NET Core
- Supported Platforms
- Hubs
- JavaScript client