

Communications en temps interactif



Objectifs

- Communications en temps interactif (temps réel)
- SignalR

Communications en temps interactif (temps réel)

Facilite l'ajout de fonctionnalité temps réel en permettant au serveur d'envoyer du contenu aux clients

Cas d'utilisation :

- Besoin de haute fréquence de mise à jour : jeu, GPS, vote, réseaux sociaux
- Tableau de bord et surveillance applicative : suivi des ventes, de production
- Collaboration : tableau blanc, outils de réunion
- Besoin de notifications : réseaux sociaux, courriels, chat, jeux, etc.

SignalR

- SignalR permet d'effectuer des appels RPC entre le client et le serveur dans les deux sens
- SignalR :
 - Gère la connexion automatiquement (WebSockets, Server-Sent Events, Long polling)
 - Peut effectuer le même appel RPC sur tous les clients
 - Peut cibler un client en particulier ou un groupe
 - Est une solution dimensionnable
- Package : Microsoft.AspNetCore.SignalR.Core

SignalR – Hub (Serveur)

- Configurer les services et les middlewares

```
public void ConfigureServices(IServiceCollection services)
{
    ...
    services.AddSignalR();
}

public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    ...
    app.UseEndpoints(endpoints =>
    {
        endpoints.MapRazorPages();
        endpoints.MapHub<TableauBlancHub>("/TableauBlancHub ");
    });
}
```

SignalR - Hub

```
public class TableauBlancHub : Hub
{
    public static List<Ligne> _dessin = new List<Ligne>();

    public async Task DessinerLigne(Ligne ligne)
    {
        _dessin.Add(ligne);

        await Clients.All.SendAsync("DessinerLigne", ligne);
    }

    public async Task EffacerTableau()
    {
        _dessin.Clear();

        await Clients.All.SendAsync("EffacerTableau");
    }

    public async override Task OnConnectedAsync()
    {
        await Clients.Caller.SendAsync("DemarrageTableau", _dessin);

        await base.OnConnectedAsync();
    }
}
```

SignalR - Client

- Dans l'explorateur de solution > Ajouter > Bibliothèque côté client : tapez : « @microsoft/signalr@latest » (fournisseur « unpkg »)
- Construire la connexion avec le serveur :

```
var connexion = new
signalR.HubConnectionBuilder().withUrl("/tableauBlancHub").build();

connexion.start().catch(function (err) {
    return console.error(err.toString());
});
```

SignalR - Client

- Répondre aux appels du serveur :

```
connexion.on("DessinerLigne", function (ligne) {  
    ...  
});  
  
connexion.on("EffacerTableau", function () {  
    ...  
});
```

- L'appeler

```
connexion.invoke(  
    "DessinerLigne",  
    { ... }).catch(function (err) {  
        console.error(err.toString());  
    });
```

Références

- <https://dotnet.microsoft.com/apps/aspnet/signalr> : Page de vente de SignalR
- <https://docs.microsoft.com/fr-ca/aspnet/core/tutorials/signalr> : Documentation SignalR
- <https://docs.microsoft.com/en-us/aspnet/signalr/overview/guide-to-the-api/working-with-groups> : travailler avec les groupes
- <https://docs.microsoft.com/en-us/aspnet/core/signalr/hubcontext> : envoyer des messages en dehors du hub, par exemple à partir d'un contrôleur