Unity Crossfire Plugin

Unity · https://namazustudios.com/docs/game-engine-client-support/unity/crossfire/

Github URL:
https://github.com/NamazuStudios/crossfire

A peer-to-peer networking plugin for Unity built on Unity Netcode for GameObjects and Unity WebRTC. It handles WebRTC connection negotiation, signaling, host election, and real-time connection monitoring through a single `NetworkSessionManager` component.

This plugin requires a running instance of Crossfire on Elements.

Requirements

- Unity 2022.3 or later
- Unity Netcode for GameObjects via Package Manager (`com.unity.netcode.gameobjects`)
- Unity WebRTC via Package Manager (`com.unity.webrtc`)
- Newtonsoft Json via Package Manager (`com.unity.nuget.newtonsoft-json`)
- WebSocket Sharp (pre-compiled, included in the Plugins folder)

Installation

1. Import the required packages via the Unity Package Manager.
2. Place the `NetworkSessionManager` prefab in your scene, or add a `NetworkSessionManager` component to a GameObject manually.
3. Assign a `NetworkManager` to the `NetworkSessionManager` in the Inspector.
4. Set the `serverHost` field in the `NetworkSessionConfig` to the base URL of your Crossfire server (for example, `ws://localhost:8080/app/ws/crossfire`).

> **Warning:** The `NetworkSessionManager` calls `DontDestroyOnLoad` on startup. To avoid duplicates when reloading scenes, either place it only in a one-time initialization scene, or check for an existing instance before creating one.

Quick Start

```
public class GameManager : MonoBehaviour
{
    [SerializeField] private NetworkSessionManager sessionManager;

    void Start()
    {
        sessionManager.OnPlayerJoined += player =>
            Debug.Log($"{player.profileId} joined");

        sessionManager.OnAllPlayersConnected += () =>
            Debug.Log("All players ready - start gameplay");

        sessionManager.OnConnectionError += error =>
            Debug.LogError($"Network error: {error}");

        // profileId and sessionToken come from your auth flow (e.g. Elements login)
        sessionManager.StartSession("player123", "auth-token");
        sessionManager.FindOrCreateMatch("default");
    }
}
```

A fuller example using the Elements Codegen client is in `Scripts/Test/NetworkTestViewController.cs`.

Documentation

- Events Reference - all events, parameter semantics, and usage examples
- Configuration - Inspector fields, CrossfireConstants, and tuning
- Architecture - state machine, component responsibilities, and local testing
- Extending the Plugin - custom transports, custom signaling, and best practices

Support

For questions or issues, open a GitHub issue with your Unity version, the relevant error logs, and a description of the steps to reproduce.