チーム割り当て・同期システム（VRCJson U＃）
このシステムは、プレイヤーのチーム情報（赤・青・未所属）を管理し、すべての参加者間で正しく同期されるよう設計されています。
VRChatのplayerTagはネットワーク同期されない仕様であるため、チーム情報を一度DataListとして保持し、VRCJsonでJSON文字列に変換して同期する方式を採用しています。
データは [UdonSynced] 属性付きの文字列として同期され、OnPreSerializationでJSON化、OnDeserializationで逆変換しています。これにより、複数のプレイヤーが動的に参加・離脱しても、チーム状態を正確に共有することが可能です。
さらに、各プレイヤーのIDに対応したチーム情報に基づいて、ローカルでplayerTagに"Red"や"Blue"といったタグを再設定しています。これにより、同期はされないがローカルUI等で使用されるタグも常に最新状態に保たれます。
ユーザーがチームに参加・離脱する際には、自身をオブジェクトのオーナーに切り替えたうえで、RequestSerialization() を呼び出すことで権限と同期の整合性を担保しています。
この実装により、VRChatの制限を回避しつつ、安定したチーム管理とプレイヤー情報の可視化を実現しています。

ーーーーー
当初は int[] のような単純な配列でチーム情報を管理していましたが、Udonでは配列のサイズ変更や一部要素の削除・挿入といった動的操作ができない制約があるため、プレイヤー数の増減や不規則な参加・離脱に柔軟に対応できないという課題がありました。
そのため、より動的な構造を持つDataListと、ネットワーク同期が可能なVRCJsonを併用する設計へと移行しました。この変更により、データの拡張性と保守性が大きく向上し、プレイヤー数の変化にも柔軟に対応可能な仕組みを実現しています。

```
using UdonSharp;
using VRC.SDK3.Data;
using UnityEngine;
using VRC.SDKBase;
using VRC.Udon;
using VRC.Udon.Common;

Unity スクリプト (1 件のアセット参照)|2 個の参照
public class TeamManager : UdonSharpBehaviour
{
    [SerializeField] MenuBoardManager menuBoardManager;
    [UdonSynced] private string _teamJson;
    private DataList _teams = new DataList();
    1 個の参照
    public override void OnPreSerialization()
    {
        var token = new DataToken(_teams);
        if (VRCJson.TrySerializeToJson(token, JsonExportType.Minify, out var json))
        {
            _teamJson = json.String;
        }
    }
    1 個の参照
    public override void OnDeserialization()
    {
        if (VRCJson.TryDeserializeFromJson(_teamJson, out var token) && token.TokenType == TokenType.DataList)
        {
            _teams = token.DataList;
            menuBoardManager.SetTeamName(_teams);
        }
        UpdatePlayerTags();
        menuBoardManager.SetTeamCount(GetTeamCount("Red"),GetTeamCount("Blue"));
    }

    3 個の参照
    public void AssignTeam(int index, int teamId)
    {
        if (!Networking.IsOwner(Networking.LocalPlayer, gameObject))
        {
            Networking.SetOwner(Networking.LocalPlayer, gameObject);
        }

        while (_teams.Count <= index)
        {
```

```
GameSoundManager.cs  Podium.cs  SettingManager.cs  MessageData.c
TeamManager
    AssignTeam(Networking.LocalPlayer.playerId, 2);
}
0 個の参照
public void LeaveTeam()
{
    AssignTeam(Networking.LocalPlayer.playerId, 0);
}
2 個の参照
private void UpdatePlayerTags()
{
    int playerCount = VRCPlayerApi.GetPlayerCount();
    VRCPlayerApi[] players = new VRCPlayerApi[playerCount];
    VRCPlayerApi.GetPlayers(players);

    for (int i = 0; i < playerCount; i++)
    {
        VRCPlayerApi player = players[i];
        int id = player.playerId;

        if (id < _teams.Count)
        {
            int team = (int)_teams[id].Double;

            switch (team)
            {
                case 1:
                    player.SetPlayerTag("Team", "Red");
                    break;
                case 2:
                    player.SetPlayerTag("Team", "Blue");
                    break;
                default:
                    player.SetPlayerTag("Team", "None");
                    break;
            }
        }
    }
}
6 個の参照
public int GetTeamCount(string teamName)
{
    int count = 0;
    int playerCount = VRCPlayerApi.GetPlayerCount();
    VRCPlayerApi[] players = new VRCPlayerApi[playerCount];
    VRCPlayerApi.GetPlayers(players);
```

ローカライズ

```
        positionObject.transform.localPosition = seting.GetDisplayPosition();

    }
    1 個の参照
    public void SetGameInfoMessage(int key, int score = 0)
    {
        string scoreText = (score == 0)? "" : score.ToString();
        switch (seting.GetCurrentLanguage())
        {
            case 0:
                infoText.text = messages_ja[key] + scoreText;
                break;
            case 1:
                infoText.text = messages_en[key] + scoreText;
                break;
            case 2:
                infoText.text = messages_ko[key] + scoreText;
                break;
        }
    }
    1 個の参照
    public void SetScore(int red,int blue)
    {
        redScoreText.text = red.ToString();
```

VRChatのUdonでは、Unity標準のローカライズツール（Localize String Table など）やScriptableObjectベースの言語管理が使用できない制約があります。そのため、配列とIDベースのシンプルな多言語切り替え処理を自作する形で実装しています。
また、各言語のテキストは対応するインデックス順に揃えることで、1つのキーで全言語のメッセージを取得可能な構造にしています。これにより、実行時の分岐や冗長なコードを最小限に抑えつつ、多言語展開にも対応しています。
目的と意図
この機能は、言語に依存せずに誰でも快適にプレイできる体験を提供するために導入したもので、特に国際プレイヤーの多いVRChatにおいて重要な要素と考えています。ゲーム内ガイドや勝敗結果など、ゲーム進行に直結する情報を正しく伝えることを目的としていっます。

TeamHaloManager.cs　GameSoundManager.cs

Assembly-CSharp

```
            13 個の参照
1       public enum GameInfoKey
2       {
3           RedTeamThrowJackBall,
4           BlueTeamThrowJackBall,
5           RedTeamThrowBall,
6           BlueTeamThrowBall,
7           ThrowJackBallAgain,
8           Aggregating,
9           RedGetScore,
10          BlueGetScore,
11          Draw,
12          RedWin,
13          BlueWin
14      }
15
```

TeamHaloManager.cs　GameSoundManager.cs　Podium.cs　SettingManager.cs　MessageData.cs　GameInfo.cs　BallLogic.cs

Assembly-CSharp　GameInfo

```
18      [SerializeField] SettingManager seting;
19
20      private string[] messages_ja =
21      {
22          "赤チームはジャックボールを投げてください",
23          "青チームはジャックボールを投げてください",
24          "赤チームはボールを投げてください",
25          "青チームはボールを投げてください",
26          "ジャックボールを再度投げてください",
27          "集計中...",
28          "赤チーム　得点:",
29          "青チーム　得点:",
30          "ゲーム終了　引き分け",
31          "ゲーム終了　赤チームの勝利",
32          "ゲーム終了　青チームの勝利"
33      };
34      private string[] messages_en =
35      {
36      "Red Team, please throw the jack ball.",
37      "Blue Team, please throw the jack ball.",
38      "Red Team, please throw your ball.",
39      "Blue Team, please throw your ball.",
40      "Please re-throw the jack ball.",
41      "Calculating scores...",
42      "Red Team Score:",
43      "Blue Team Score:",
44      "Game Over - Draw",
45      "Game Over - Red Team Wins",
46      "Game Over - Blue Team Wins"
47      };
48      private string[] messages_ko =
49      {
50      "레드팀, 잭볼을 던져주세요.",
51      "블루팀, 잭볼을 던져주세요.",
52      "레드팀, 공을 던져주세요.",
53      "블루팀, 공을 던져주세요.",
54      "잭볼을 다시 던져주세요.",
55      "점수 계산 중...",
56      "레드팀 점수:",
57      "블루팀 점수:",
58      "게임 종료 - 무승부",
59      "게임 종료 - 레드팀 승리",
60      "게임 종료 - 블루팀 승리"
61      };
62
```