

日専祭 SDK v2022.1.4

クイックスタート ガイドブック

2022 年 10 月 11 日
システム制作セクション

目次

内容

日専祭 SDK v2022.1.4	0
クイックスタート ガイドブック	0
目次	1
1. 内容物	2
SystemStatusManager.prefab	2
Nissensai.cs	2
Sample	2
2. 実装方法	3
① 「SystemStatusManager.prefab」をタイトルシーンに入れてください。	3
② Inspector Window で基本設定を調整します。	3
③ ゲーム開始とゲーム終了時の Callback メソッドを用意する	3
④ Callback メソッドの設定をする	5
⑤ 実装完了！	6
3. Nissensai.cs の基礎	7
• public static Player CurrentPlayer	7
• public static void SendResult(ResultRank rank)	7
• public static void GiveUp()	7
• public static void ShowQrCode()	7
• public static void HideQrCode()	7
• public static string GetPlayerName()	7
4. 上級な使い方	8
① QR コードを手動で表示/非表示する	8
② コンソールウィンドウについて	8
③ コンソールにログを出力したい	9
④ コンソールにコマンドを追加したい	9
5. 問い合わせ先	9

1. 内容物

- ・「NissensaiSDK 2022.1.4.unitypackage」

- ・ Nissensai2022

- ・ Internal

内部スクリプト

- ・ Nissensai.cs

実装に使うメソッド(関数)

- ・ Sample

使い方のサンプルデータ

- ・ SystemStatusManager.prefab

実装に使うプレハブ

- ・ 「NissensaiSDK 2022.1.4 Quick Start Guidebook.pdf」 (本書)

- ・ Plugins

QR コードを生成するためのライブラリ

※皆さんは赤色で表記されている部分以外は触らなくて大丈夫です。

SystemStatusManager.prefab

使い方は後ほど説明しますが、これはサーバー通信機能、システム状態管理やプレイヤー情報の管理などをまとめたものです。

Nissensai.cs

実装に使えるメソッド(関数)をまとめたスクリプトです。中身については後ほど解説します。

Sample

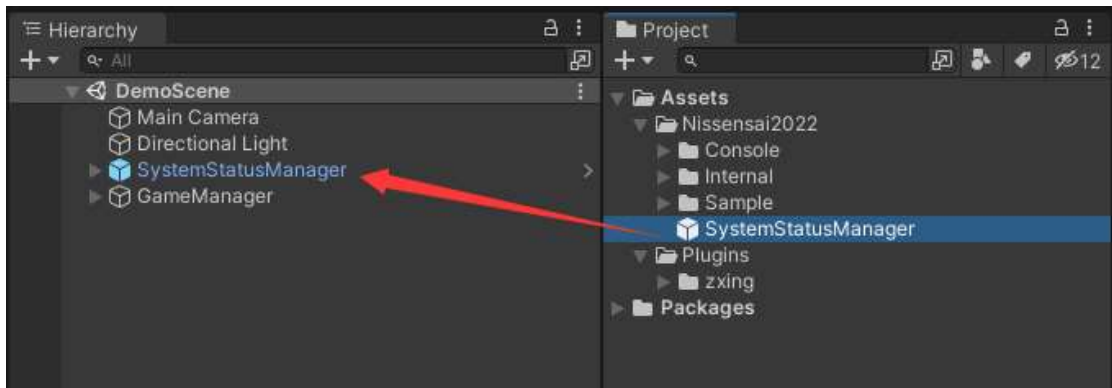
このフォルダはサンプルが入っています。分からなかったらこれを参考に実装してみてください。

「GameManger.cs」は日専祭システムを使って簡単な連打ゲームを実現しています。

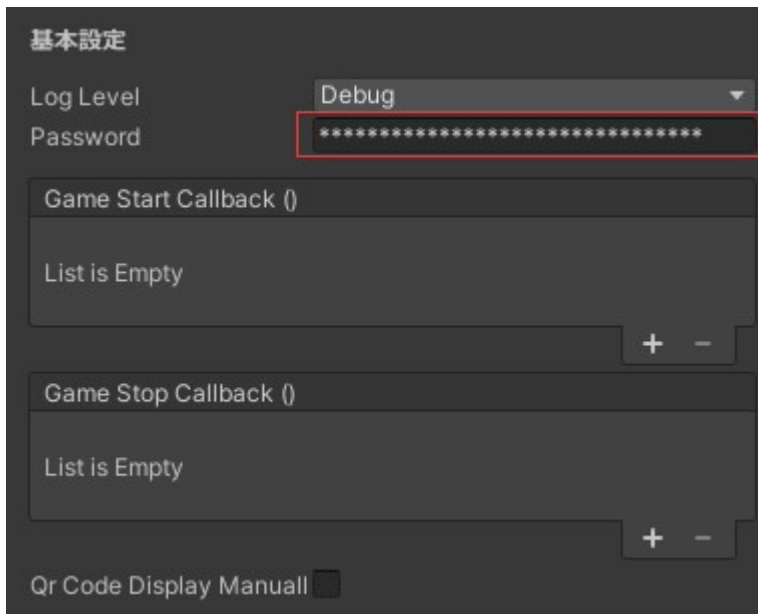
「DemoScene」でやってみましょう！

2. 実装方法

- ① 「SystemStatusManager.prefab」をタイトルシーンに入れてください。



- ② Inspector Window で基本設定を調整します。



各コンパニーの代表者にお渡しする**パスワード**をここに**設定**してください。

- ③ ゲーム開始とゲーム終了時の Callback メソッドを用意する

✓ Callback メソッドとは？

何かが発生した時点で呼ばれる(実行される)メソッドのことを Callback メソッドと呼びます。

この GameStartCallback()はプレイヤーが QR コードを読み取って、又は勇者番号を入力して、ゲームがスタートする時に呼ばれるメソッドです。

同様、GameStopCallback()はリザルトがサーバーに送信された後、或いはプレイヤーはゲームを中断した時に呼ばれるメソッドです。

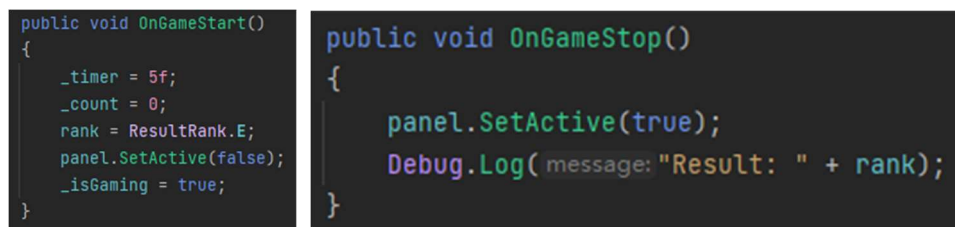
✓ Callback メソッドの作り方

基本的に MonoBehaviour を継承したクラスの public メソッドであればどれも Callback として扱うことができます。

分かりやすく言うと、Unity で新規作成したスクリプトの中に public で作った関数ならどれでも大丈夫です。

例えば「Sample」フォルダの「GameManager.cs」の 20 行から 27 行まではサンプルの GameStartCallback メソッドになります。

その下、32 行から 36 行の間は GameStopCallback メソッドになります。



```
public void OnGameStart()
{
    _timer = 5f;
    _count = 0;
    rank = ResultRank.E;
    panel.SetActive(false);
    _isGaming = true;
}

public void OnGameStop()
{
    panel.SetActive(true);
    Debug.Log(message: "Result: " + rank);
}
```

※今後は Callback メソッドが入っているスクリプトのことを **Callback スクリプト**と言います。Callback スクリプトがアタッチされているゲームオブジェクトのことを **Callback オブジェクト**と言います。

✓ GameStartCallback メソッドでやるべきこと

画面のトランジションエフェクト(フェードアウトとか)の開始やシーンの遷移、インゲーム変数の初期化などを行ってください。

基本的タイトルシーンからインゲームに遷移するのと変わりはありません。

✓ GameStopCallback メソッドでやるべきこと

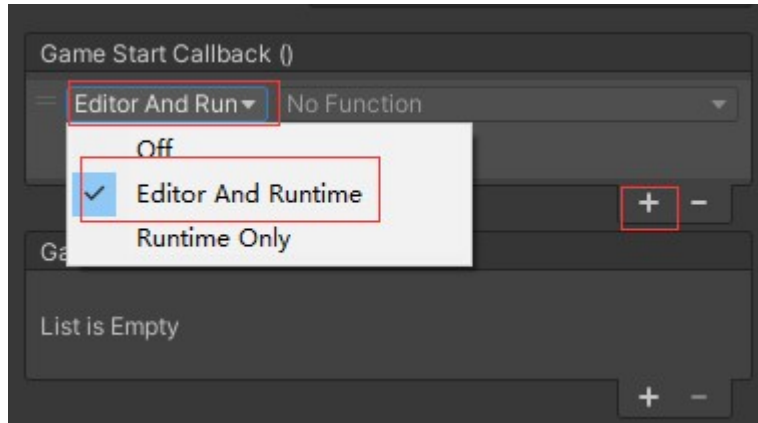
リザルト画面への遷移や演出の再生など、ワンプレイが終わった時点で必要な処理を行ってください。

要注意！この**ゲーム終了**の判定は後ほど説明する **SendResult メソッド**が実

行され、サーバーにリザルトデータが送信されたタイミングになります。

④ Callback メソッドの設定をする

「SystemStatusManager」の Inspector Window で設定を行います。

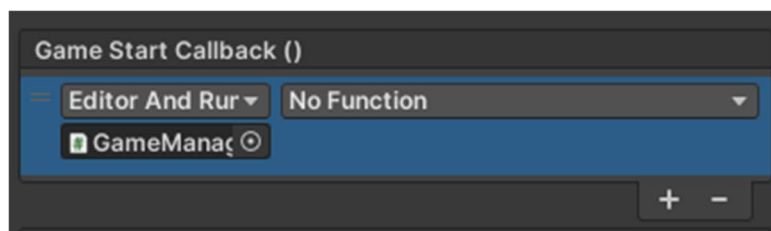


「+」マークを押して GameStartCallback を 1 つ追加します。(そうです！複数追加は可能です！)

画像通りに選択リストから「Editor And Runtime」を選択してください。

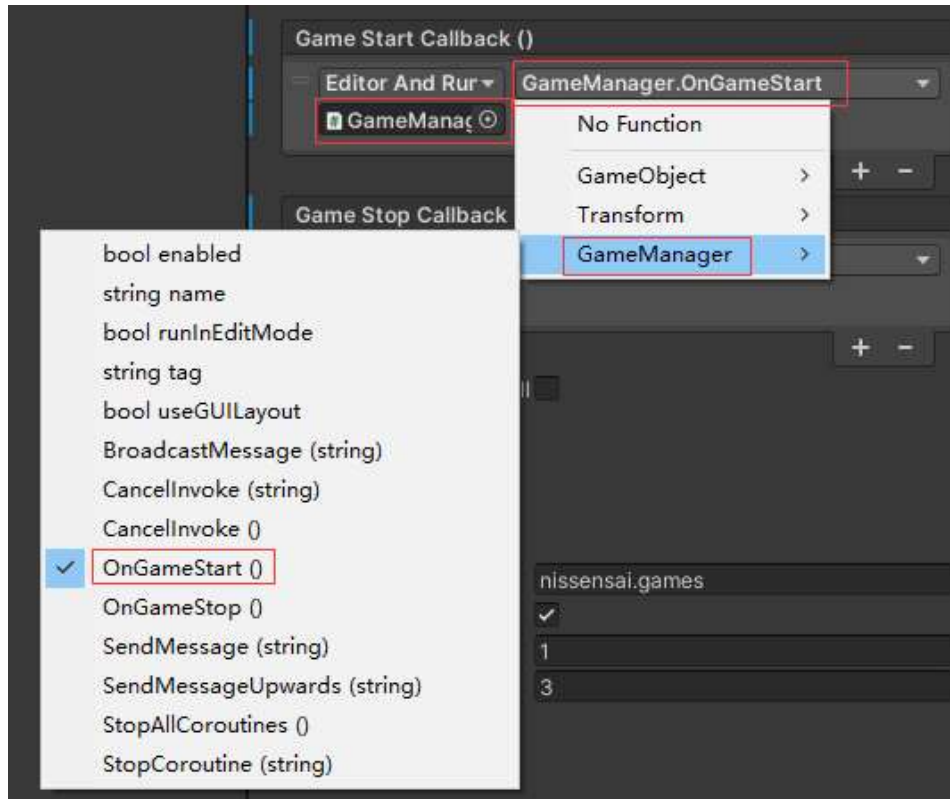
Callback オブジェクトを「None(Object)」のところにドラッグアンドドロップしてください。

私は「GameManager.cs」というスクリプトがアタッチされている「GameManager」をドラッグアンドドロップしたので、ここに「GameManager」が表示されています。(表示するのはオブジェクトの名前です。)

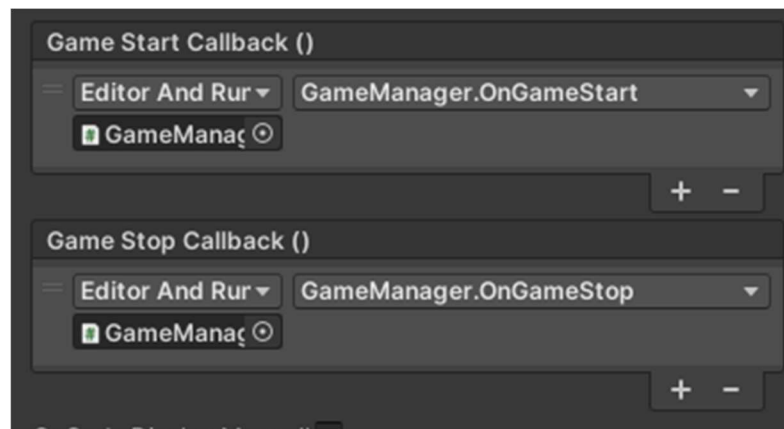


そしたら右側の「No Function」をクリックして、GameStartCallback メソッドとして指定したいメソッドを選びます。(リストに表示されない場合はそのメソッドが public になっているかどうかをご確認ください。)

※次のページに参考画像があります。



私の場合は「GameManager.cs」の「OnGameStart」というメソッドにしたいので、「GameManager」→「OnGameStart」の順番で設定します。



同様に、GameStopCallback メソッドも設定しておきます。

⑤ 実装完了！

ここまでやっていたらもう実装ができました！起動して確認しましょう！

インゲームがまだ完成できていない場合は適当にログとか、何キーを押したら

ゲームクリアとかで対応して実行できるかどうかを確認してみましょう！

3. Nissensai.cs の基礎

使う前にスクリプトの一番上に using を追加しましょう。

```
using Nissensai2022.Runtime;
```

- ・ public static Player CurrentPlayer

現在のプレイヤー情報を返してきます。Player クラスについては後ほど説明します。

- ・ public static void SendResult(ResultRank rank)

リザルトをサーバーに送信します。

例：「Nissensai.SendResult(ResultRank.A);」は「今プレイしているプレイヤーのリザルトランクは A ですよ」というのをサーバーに送信します。

もしサーバーが受付しましたら GameStopCallback メソッドが実行されます。

- ・ public static void GiveUp()

プレイヤーはクリアせず、ゲームをやめた場合に使いましょう。これを使わないと次の人がプレイできなくなるので気をつけてください。

- ・ public static void ShowQrCode()

後ほど「上級な使い方」で紹介しますが、手動で QR コードを表示させます。

- ・ public static void HideQrCode()

後ほど「上級な使い方」で紹介しますが、手動で QR コードを非表示させます。

- ・ public static string GetPlayerName()

現在プレイしているプレイヤーの名前を取得します。

※どんなコンパニーでも絶対は使うのは「SendResult」のみです。

4. 上級な使い方

① QR コードを手動で表示/非表示する

演出面でカスタマイズしたいコンパニー向けの内容です。

「SystemStatusManager」の基本設定で「Qr Code Display Manually」のチェックを入れてください。

チェックを入れたら、QR コードの On/Off は自動で行わなくなります。

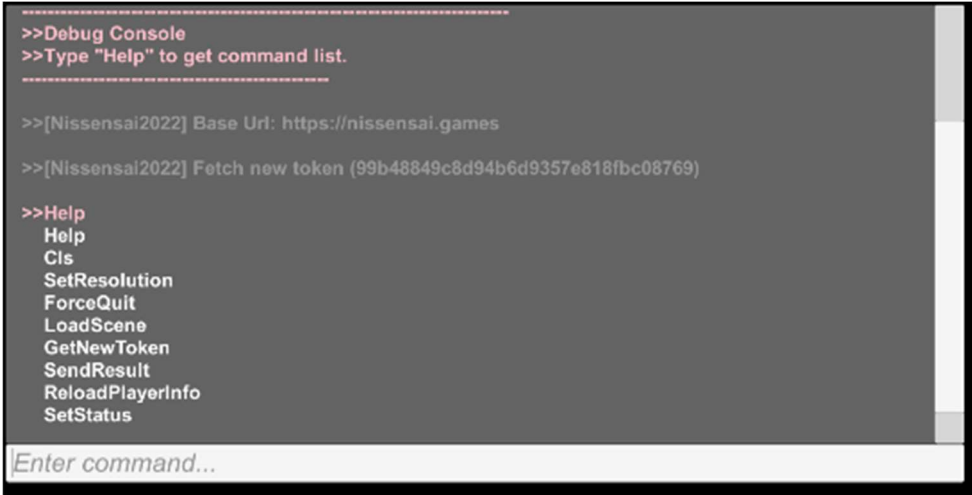
前述の「**ShowQrCode**」と「**HideQrCode**」のメソッドで制御してください。

② コンソールウィンドウについて

システムにコンソールウィンドウが内蔵されています。

ゲーム起動時にログを確認したり、チートコマンド/裏コマンドを打ち込んだりできます。

F1 キーで表示/非表示できます。Help と打って Enter を押したらコマンド一覧が見られます。(大文字小文字は気をつけてください)



```
>>Debug Console
>>Type "Help" to get command list.

>>[Nissensai2022] Base Url: https://nissensai.games
>>[Nissensai2022] Fetch new token (99b48849c8d94b6d9357e818fbc08769)

>>Help
Help
Cls
SetResolution
ForceQuit
LoadScene
GetNewToken
SendResult
ReloadPlayerInfo
SetStatus

Enter command...
```

③ コンソールにログを出力したい

まずは using を追加しましょう。

```
using Logger = Nissensai2022.Runtime.Logger;
```

そしたら「Logger.Log("");」、「Logger.Warn("");」、「Logger.Error("");」のように使いましょう。

要注意！Error で記録した場合はコンソールウィンドウが強制表示されてしまうので、重大なエラーでない限りは Log か Warn を使いましょう。

④ コンソールにコマンドを追加したい

「Nissensai.AddConsoleMethod(string command, Func<string> method)」を使って追加しましょう。

先ずコマンドを実行する static メソッドを用意します。

例：private static string Test() { return “Test”; }

※必ず string を返すこと。これは結果としてコンソールに表示されます。

```
Nissensai.AddConsoleMethod(“Test”, Test);
```

でコマンドを追加します。

Test と打ったら Test メソッドが実行されるイメージです。

更に上級なカスタマイズは「Internal」→「SystemStatusManager.cs」の 116 行から 119 行までを参考してください。

5. 問い合わせ先

本書を読んでもわからなかった場合は楊までお問い合わせください。

学祭 Discord の「システム制作」というチャンネルに投稿してください。

Discord の参加はこちらです。 <https://discord.gg/maVzjHUnaS>