2020年度
Unity講座(基礎編)

08回目

講師：幸田 将伍 (@MagurodonDev)

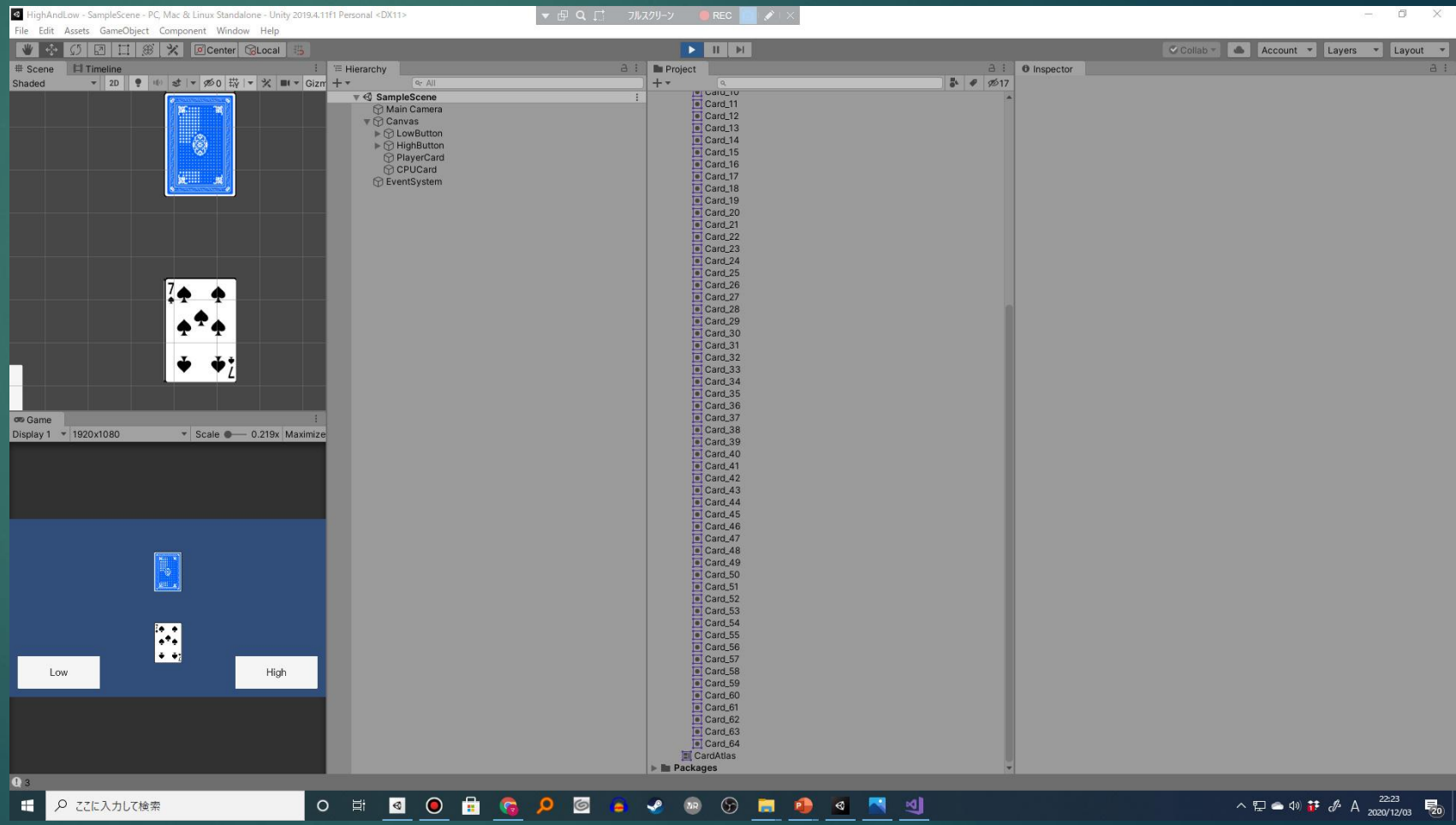
今回の講義の目的

- プログラムを自分で読めるようになる
- Unityを使って自分が実現したいことをできるようになる
- 自分一人でもゲームを作成できるレベルになる
- Unityの活用事例を学び、自分の進路に役立てる
- 実際のエンジニアがどういった仕事の進め方をしているかを知る
- ゲーム会社のクライアントエンジニアとして就職できるレベルになる

一緒にレベルアップして行きましょう！

Unity
ハイアンドロー

- 先週は、基本的なゲームの流れを実装することができたと思います
- そして、“Win”と”Lose”を表示するGameJudge.csを作成します
- そこから書いていきましょう
- もうすでに書いてある方は、次にすすんでいただいても大丈夫です
- 書き終わったら、HighAndLowSequence.csに追記していきましょう



HighAndLow - Microsoft Visual Studio

ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)

Debug Any CPU Unity にアタッチ

GameJudge.cs CardHelper.cs CardImageHelper.cs CPUCard.cs PlayerCard.cs Dealer.cs HighAndLowSequence.cs PlayerJudge.cs

Assembly-CSharp GameJudge JudgeText

4

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class GameJudge : MonoBehaviour
{
    public Text JudgeText;

    public void GameJudgeTextInit()
    {
        JudgeText.text = string.Empty;
    }

    public void GameJudgeTextView(bool isWin)
    {
        if (isWin)
        {
            JudgeText.text = "Win";
        }
        else
        {
            JudgeText.text = "Lose";
        }
    }
}
```

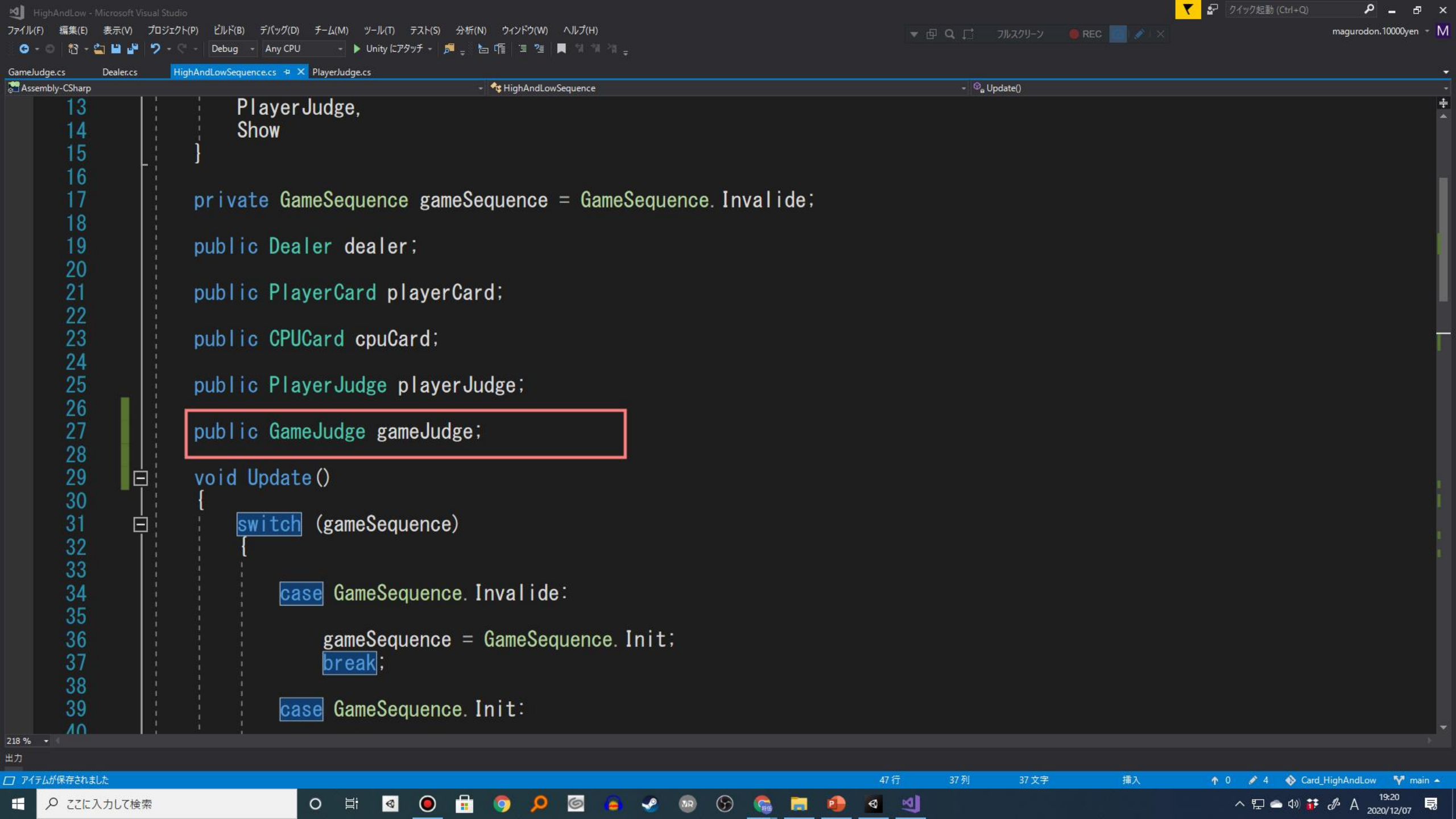
@MagurodonDev

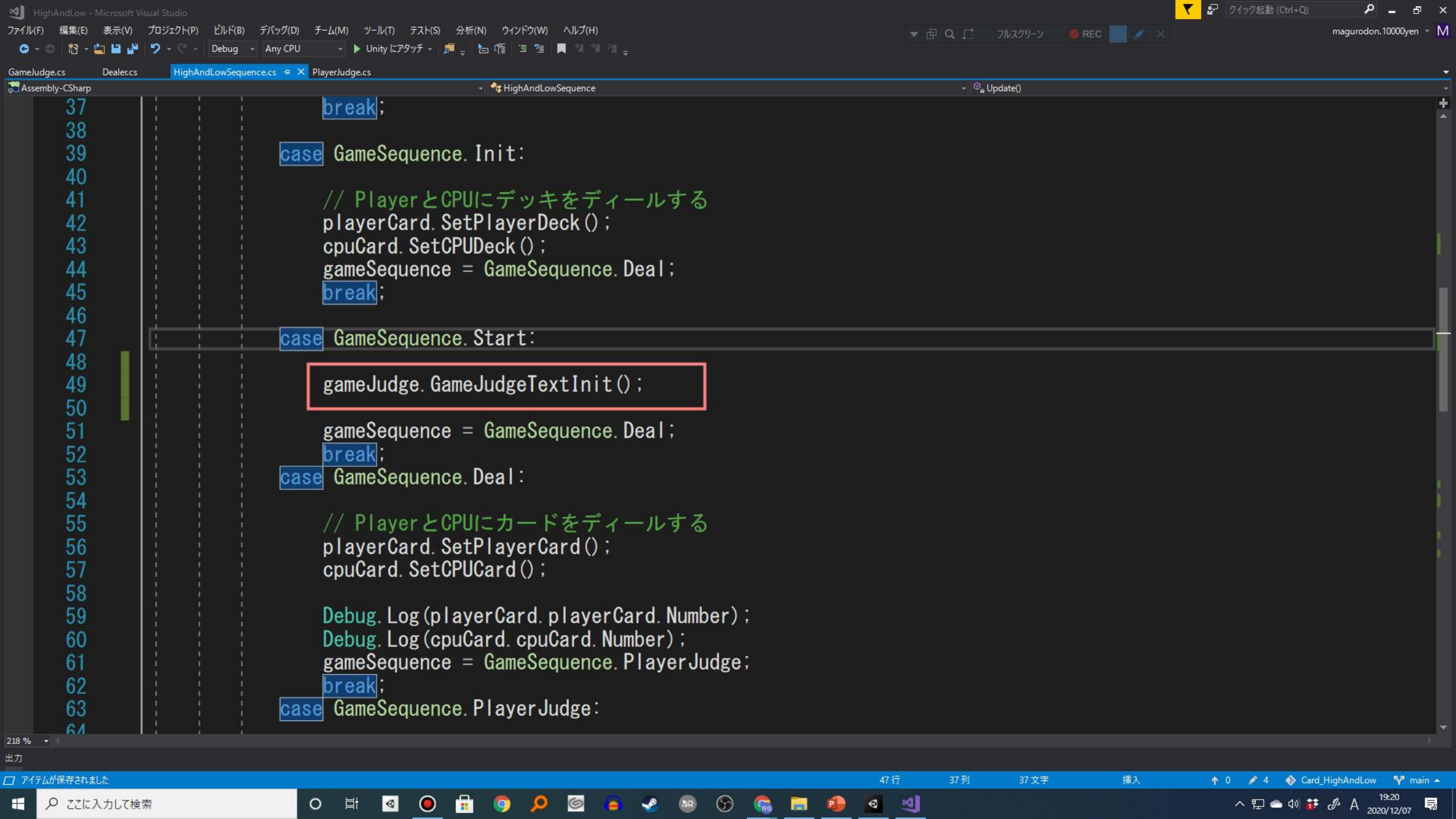
218 %

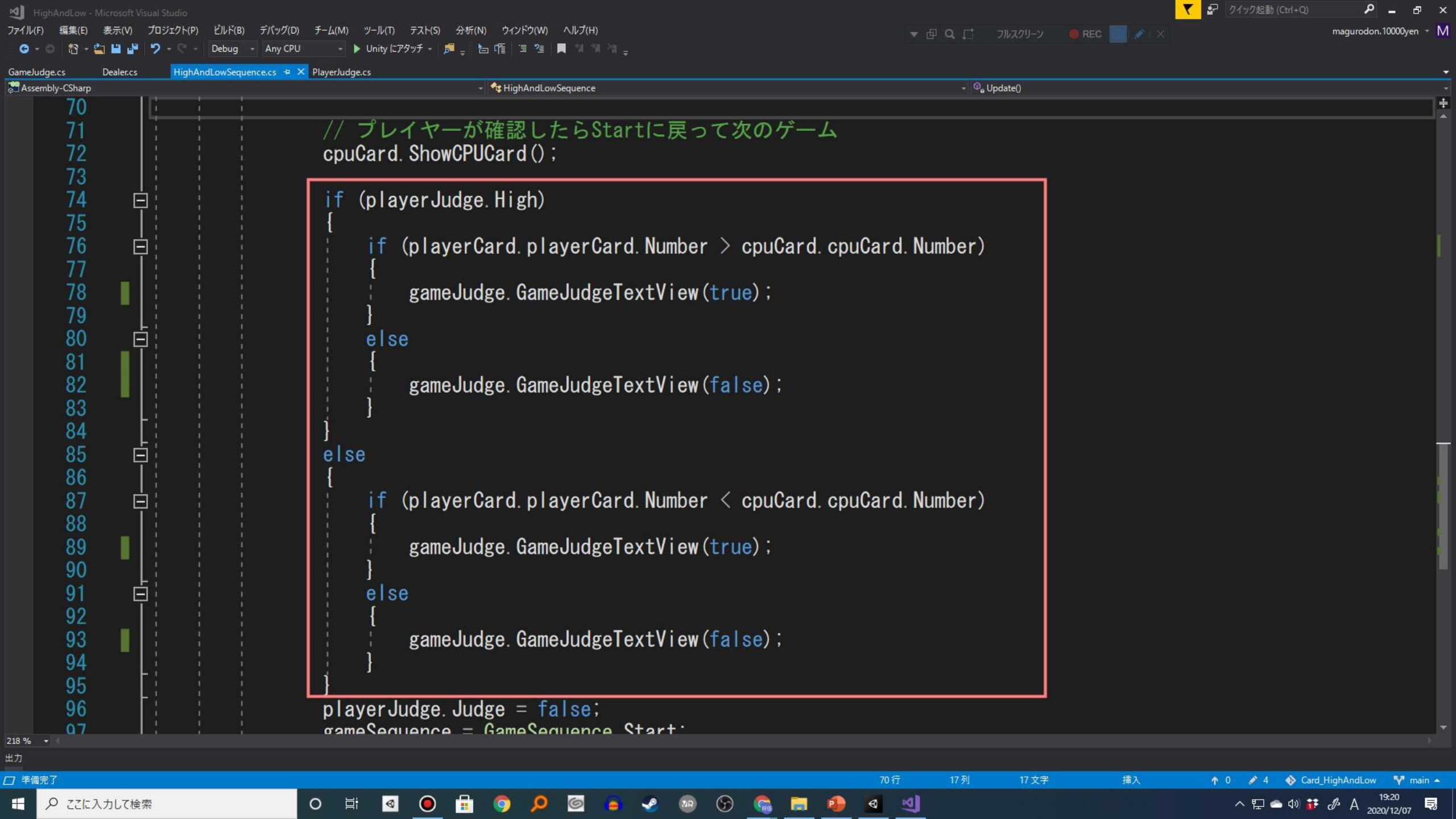
出力

アイテムが保存されました 27 行 1 列 1 文字 挿入 0 4 Card_HighAndLow main

ここに入力して検索 19:19 2020/12/07

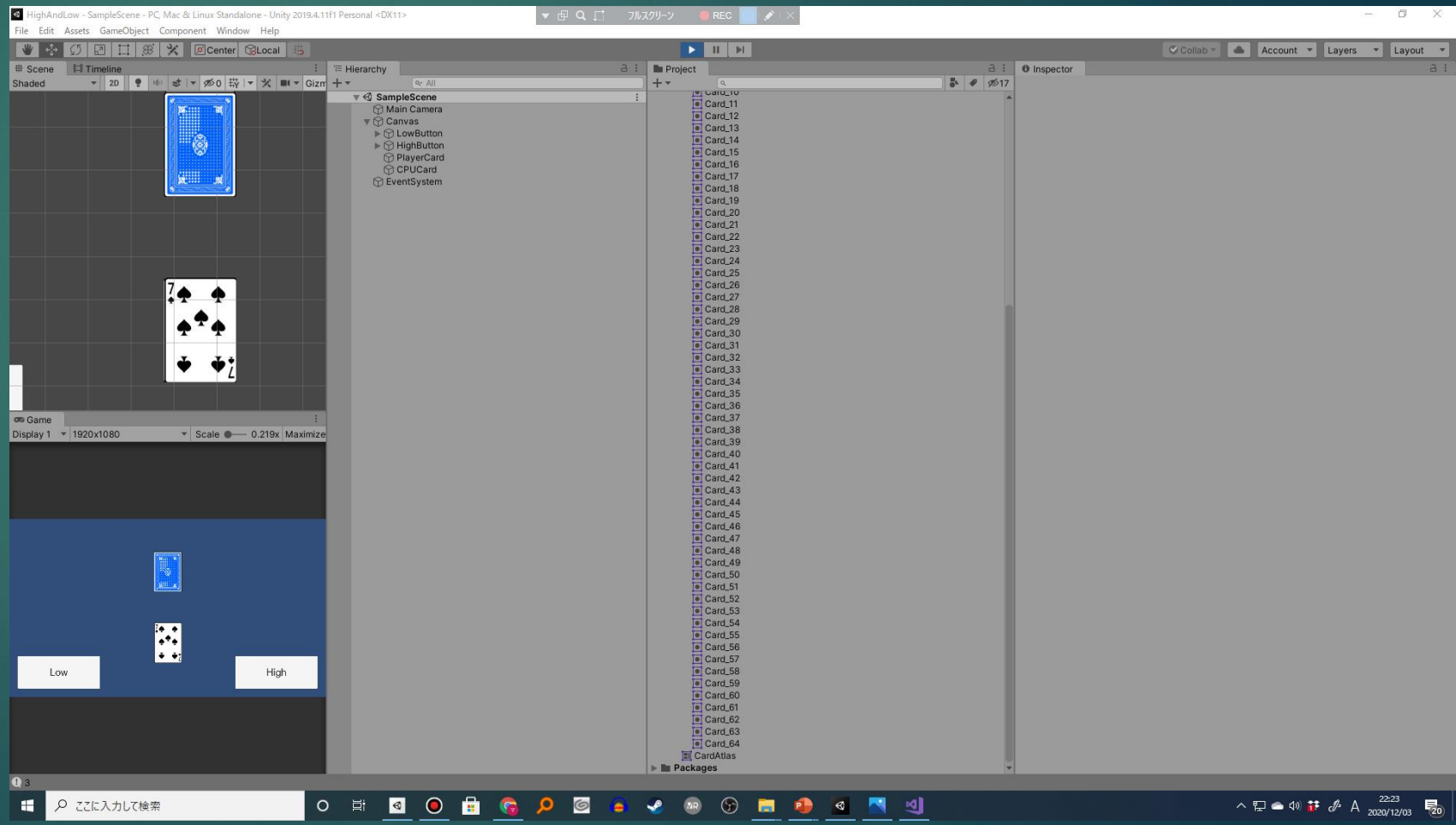

HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
Update()
PlayerJudge,
Show
}
private GameSequence gameSequence = GameSequence.Invalide;
public Dealer dealer;
public PlayerCard playerCard;
public CPUCard cpuCard;
public PlayerJudge playerJudge;
public GameJudge gameJudge;
void Update()
{
switch (gameSequence)
{
case GameSequence.Invalide:
gameSequence = GameSequence.Init;
break;
case GameSequence.Init:
218 %
出力
アイテムが保存されました
47 行
37 列
37 文字
挿入
Card_HighAndLow
main
ここに入力して検索
19:20
2020/12/07

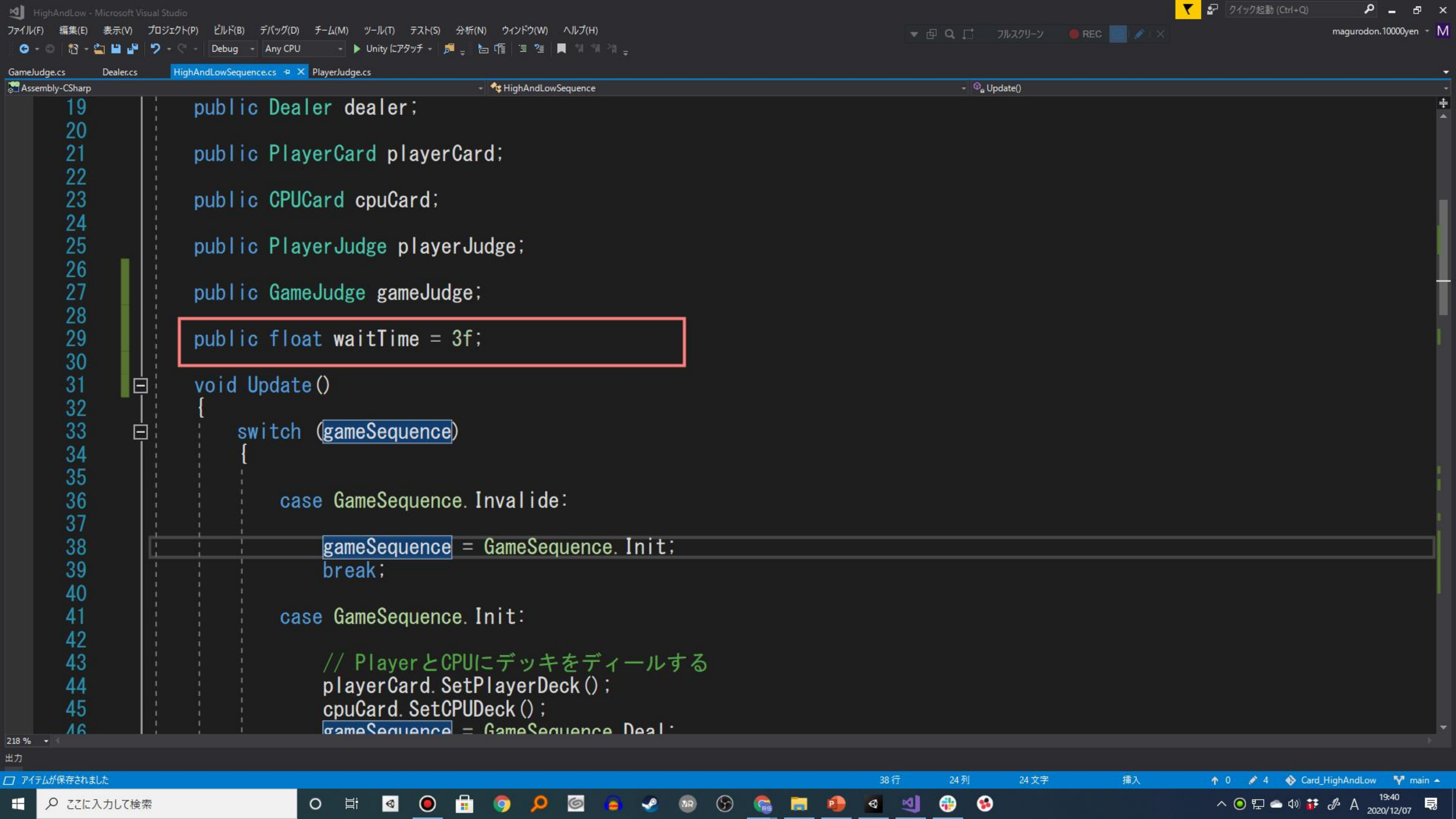

HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
Update()
37 break;
38
39 case GameSequence.Init:
40
41 // PlayerとCPUにデッキをディールする
42 playerCard.SetPlayerDeck();
43 cpuCard.SetCPUDeck();
44 gameSequence = GameSequence.Deal;
45 break;
46
47 case GameSequence.Start:
48
49 gameJudge.GameJudgeTextInit();
50
51 gameSequence = GameSequence.Deal;
52 break;
53 case GameSequence.Deal:
54
55 // PlayerとCPUにカードをディールする
56 playerCard.SetPlayerCard();
57 cpuCard.SetCPUCard();
58
59 Debug.Log(playerCard.playerCard.Number);
60 Debug.Log(cpuCard.cpuCard.Number);
61 gameSequence = GameSequence.PlayerJudge;
62 break;
63 case GameSequence.PlayerJudge:
218 %
出力
アイテムが保存されました
47 行
37 列
37 文字
挿入
Card_HighAndLow
main
ここに入力して検索
19:20
2020/12/07


HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
Update()
// プレイヤーが確認したらStartに戻って次のゲーム
cpuCard.ShowCPUCard();
if (playerJudge.High)
{
if (playerCard.playerCard.Number > cpuCard.cpuCard.Number)
{
gameJudge.GameJudgeTextView(true);
}
else
{
gameJudge.GameJudgeTextView(false);
}
}
else
{
if (playerCard.playerCard.Number < cpuCard.cpuCard.Number)
{
gameJudge.GameJudgeTextView(true);
}
else
{
gameJudge.GameJudgeTextView(false);
}
}
playerJudge.Judge = false;
218 %
出力
準備完了
70 行
17 列
17 文字
挿入
Card_HighAndLow
main
ここに入力して検索
19:20
2020/12/07

Unity
ハイアンドロー

- Unityを設定をして、実行してみましょう
- 表示はできたと思うのですが、一瞬で消えてしまっていると思います
- ここでは処理を待ちたいですよね。
- 何パターンか書き方がありますので、覚えていきましょう
- HighAndLowSequence.csにまず、待つ時間の変数を作成します

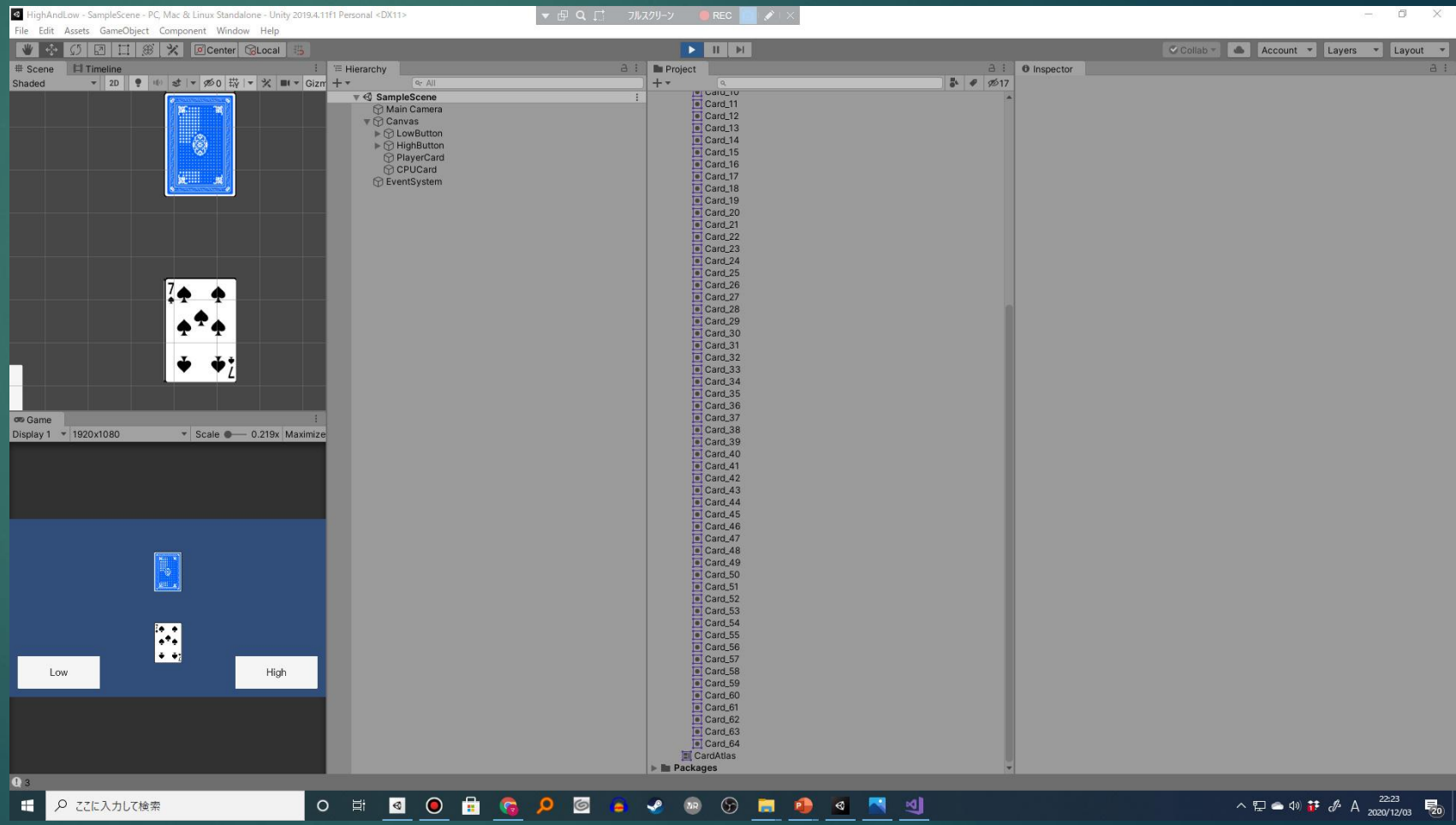


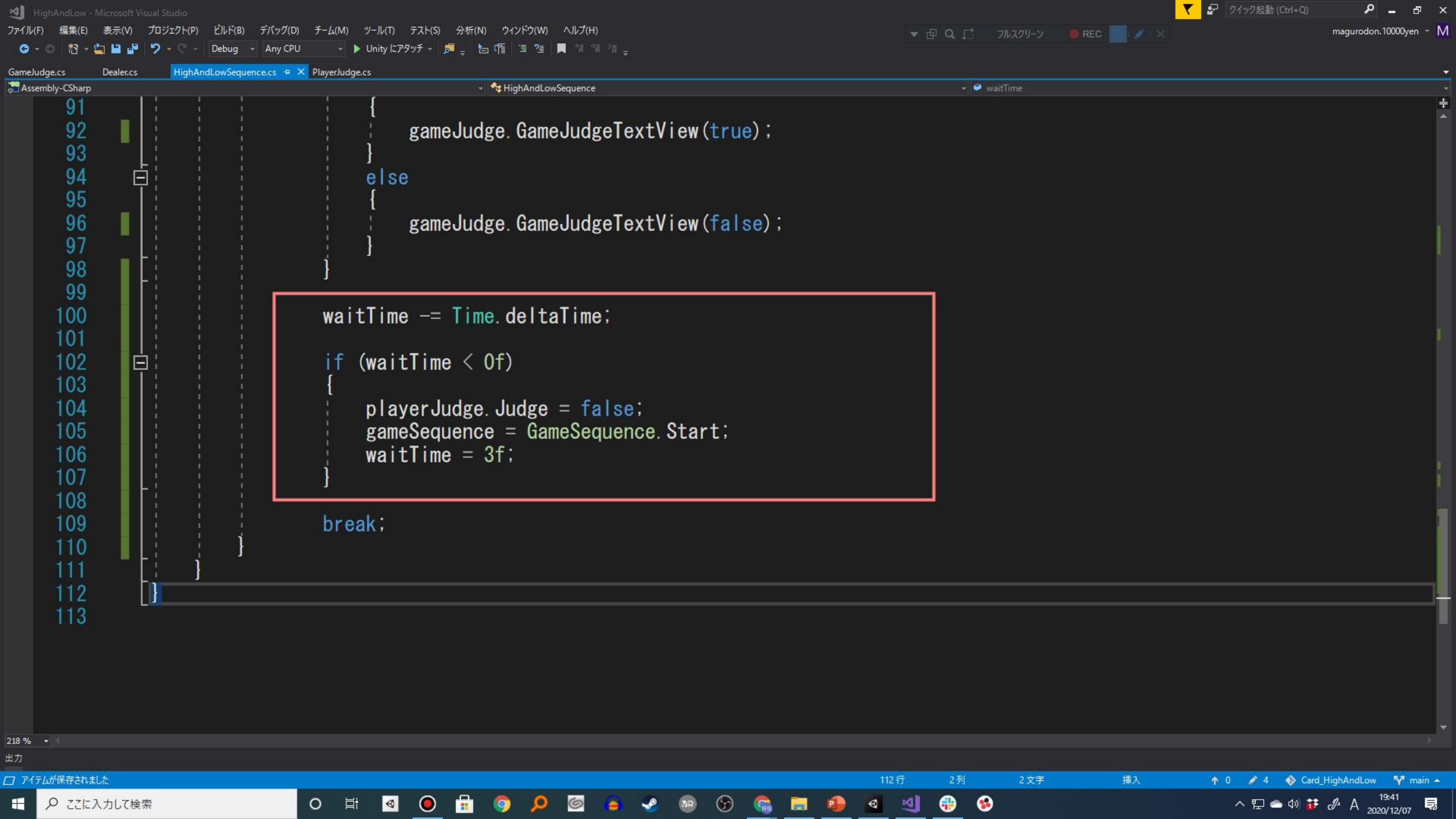

HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
Update()
public Dealer dealer;
public PlayerCard playerCard;
public CPUCard cpuCard;
public PlayerJudge playerJudge;
public GameJudge gameJudge;
public float waitTime = 3f;
void Update()
{
switch (gameSequence)
{
case GameSequence.Invalide:
gameSequence = GameSequence.Init;
break;
case GameSequence.Init:
// PlayerとCPUにデッキをディールする
playerCard.SetPlayerDeck();
cpuCard.SetCPUDeck();
218 %
出力
アイテムが保存されました
38 行
24 列
24 文字
挿入
Card_HighAndLow
main
ここに入力して検索
19:40
2020/12/07

Unity

ハイアンドロー

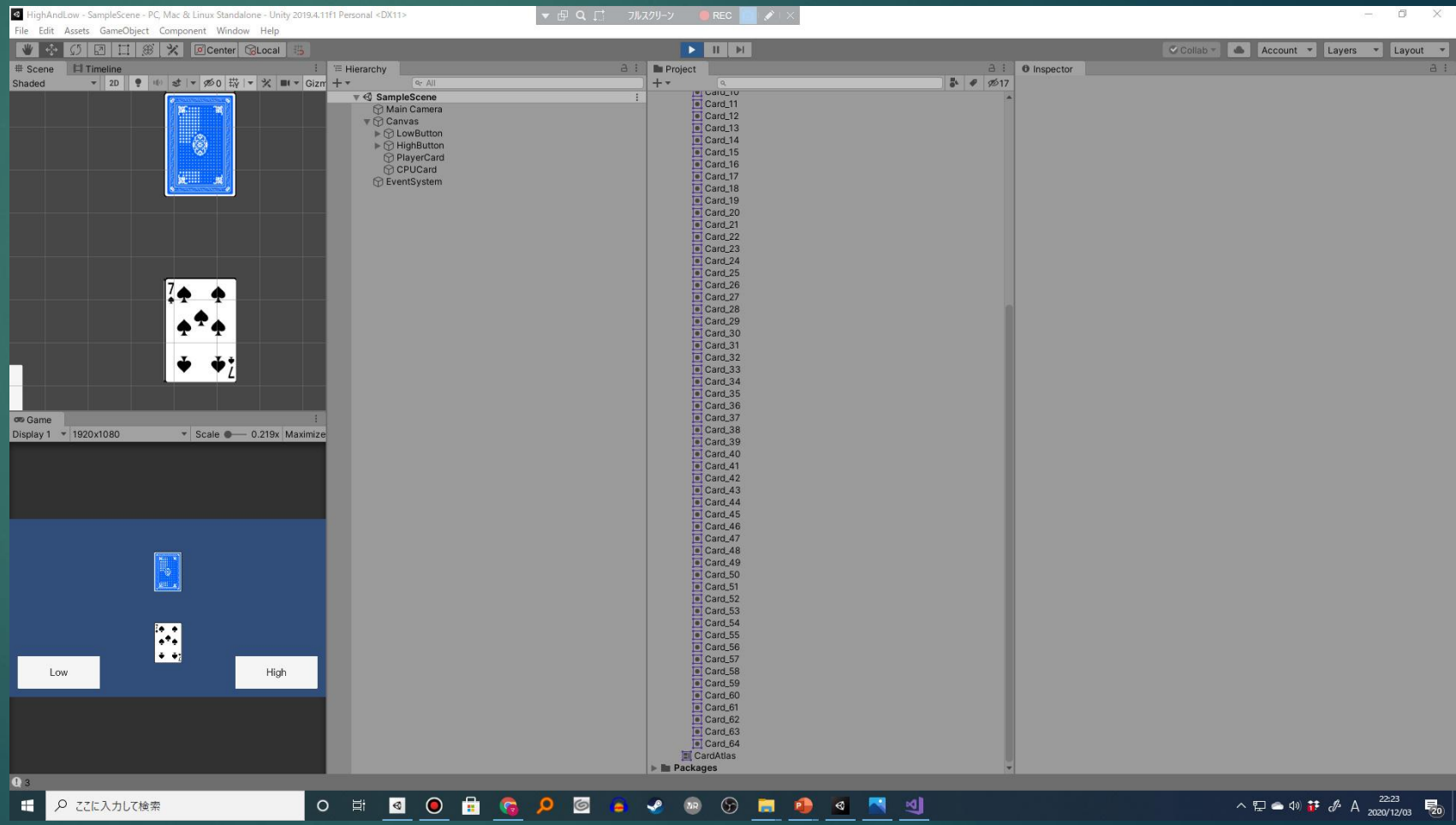
- 一つ目の方法はRollABallの時にもやったと思いますが、Time.deltaTimeでフレームとフレームの間の時間を減算していく方法があります
- スタンダードな方法です
- 時間等で待ちたいときはこの書き方をすればよいでしょう

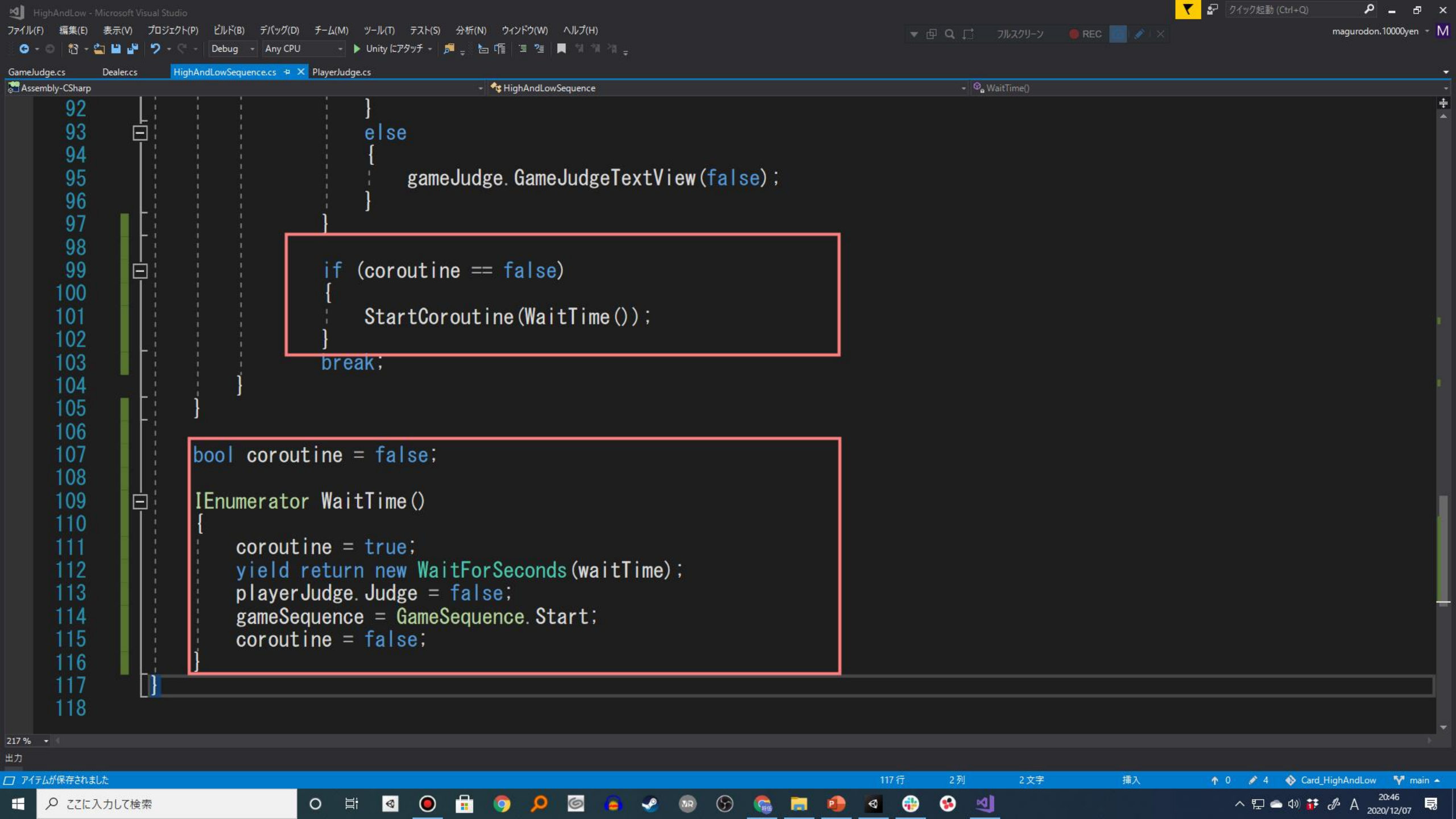



HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
waitTime
91 {
92 gameJudge.GameJudgeTextView(true);
93 }
94 else
95 {
96 gameJudge.GameJudgeTextView(false);
97 }
98 }
99
100 waitTime -= Time.deltaTime;
101
102 if (waitTime < 0f)
103 {
104 playerJudge.Judge = false;
105 gameSequence = GameSequence.Start;
106 waitTime = 3f;
107 }
108
109 break;
110 }
111 }
112 }
113
218 %
出力
アイテムが保存されました
112 行
2 列
2 文字
挿入
Card_HighAndLow
main
ここに入力して検索
19:41
2020/12/07

Unity
ハイアンドロー

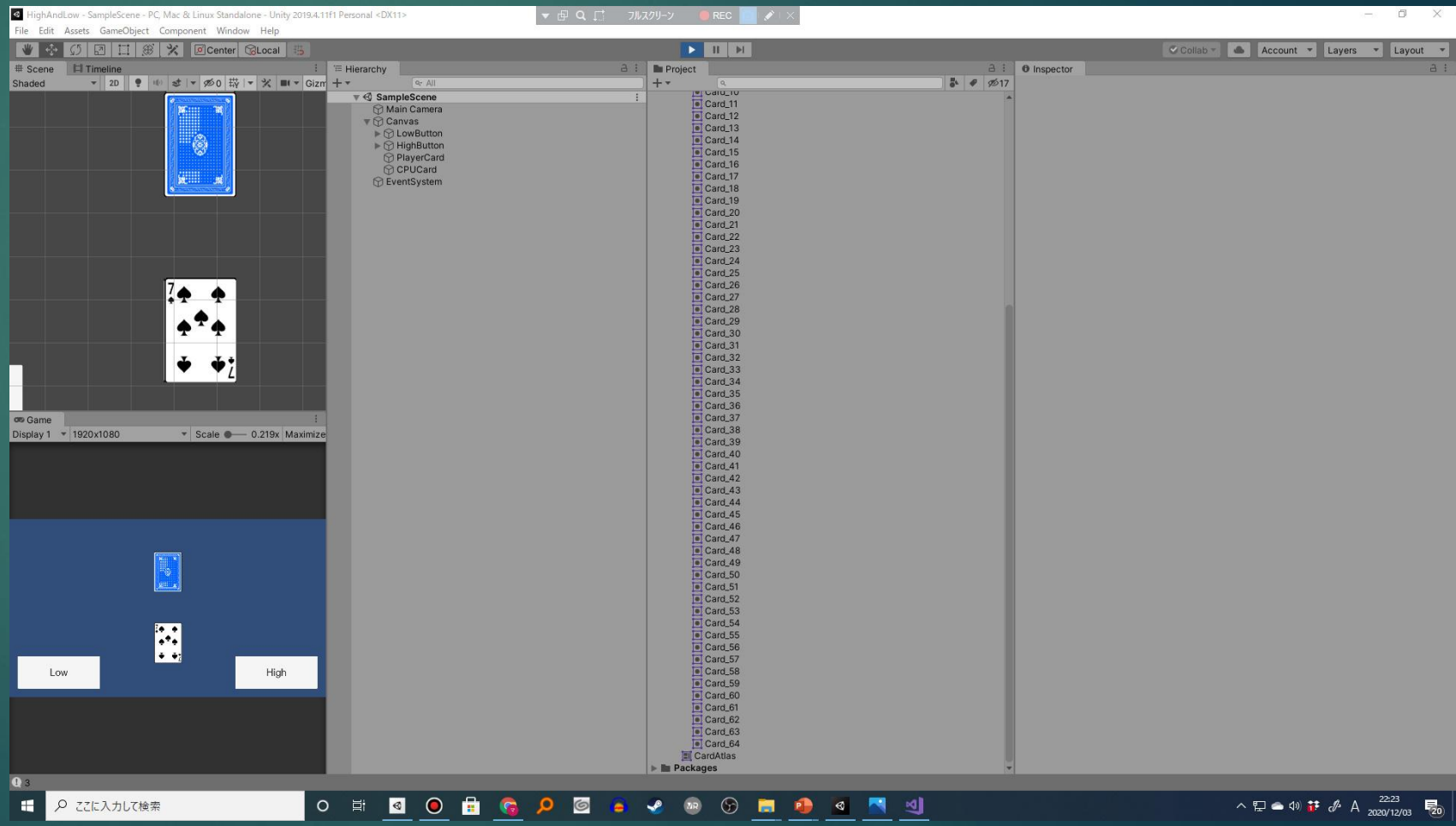
- プレイしてみると3秒待たれていることが分かると思います
- ２つ目の方法はコルーチンを使うという方法です
- これもスタンダードな方法です
- 時間だけではなく、フラグでも待ちたい時に推奨される書き方です
- (今回はあまり良い例ではないのですが、力技でコルーチンを動かしてみます。またどこかで良い例が出てくるので、待ち方だけ覚えておいてください)




HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
WaitTime()
92 }
93 else
94 {
95 gameJudge.GameJudgeTextView(false);
96 }
97 }
98
99 if (coroutine == false)
100 {
101 StartCoroutine(WaitTime());
102 }
103 break;
104 }
105 }
106
107 bool coroutine = false;
108
109 IEnumerator WaitTime()
110 {
111 coroutine = true;
112 yield return new WaitForSeconds(waitTime);
113 playerJudge.Judge = false;
114 gameSequence = GameSequence.Start;
115 coroutine = false;
116 }
117 }
118
217 %
出力
アイテムが保存されました
117 行
2 列
2 文字
挿入
Card_HighAndLow
main
ここに入力して検索
20:46
2020/12/07

Unity
ハイアンドロー

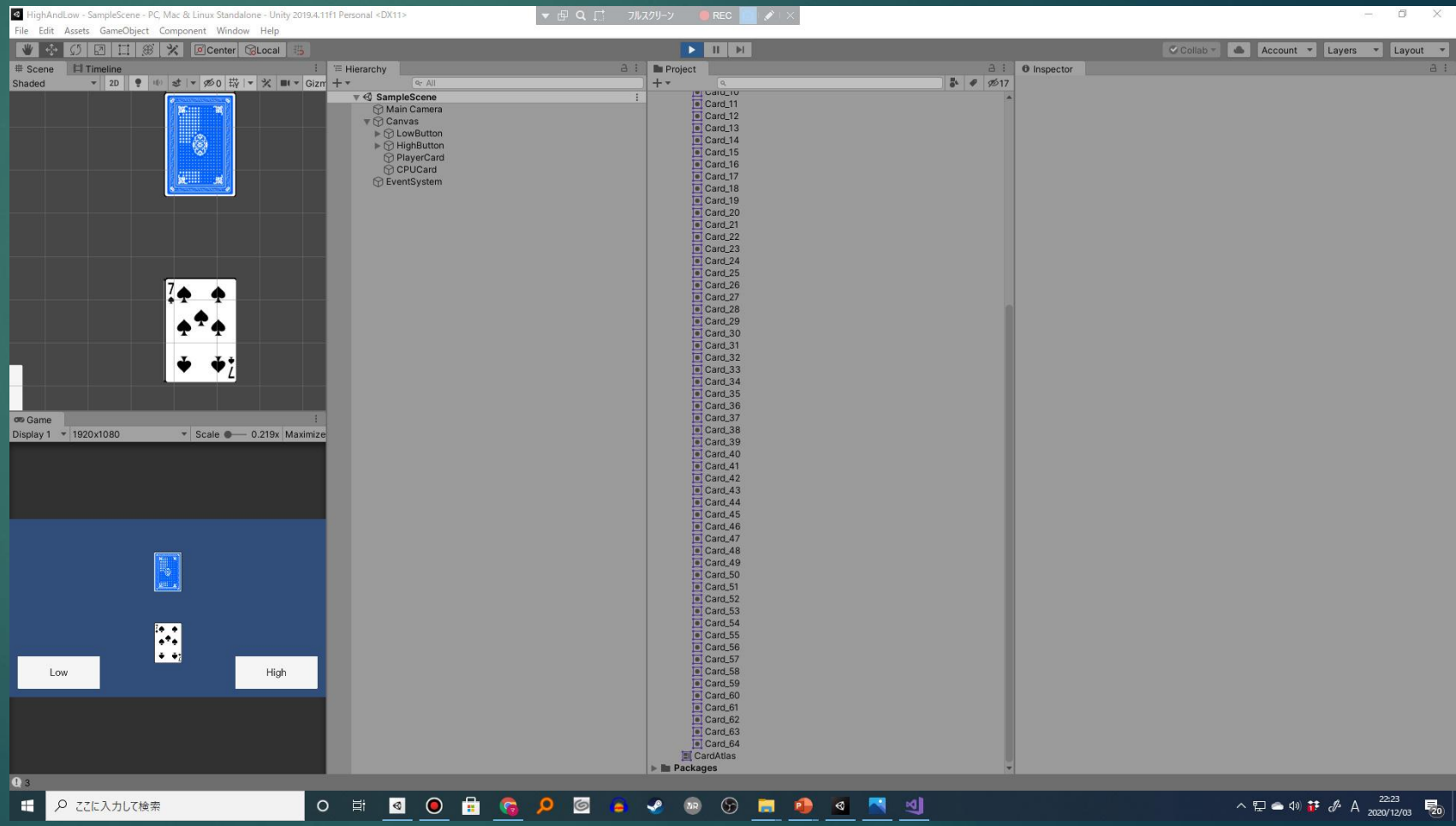
- プレイしてみると3秒待たれていることが分かると思います
- ※Yield Return New hogehoge
- Hogehogeの部分には、Coroutineが持っている様々な待ち方が入ります
- 実際のゲーム業界の現場では通信待ち等に使われています
- ではtime.deltaTimeを使う待ちに戻しておきます



Unity
ハイアンドロー

- では、スコアを表示していきましょう
- 自力で頑張ってみてください
- Winになった時に2枚加算されるといった感じです
- 負ければ、CPU側に2枚加算されます

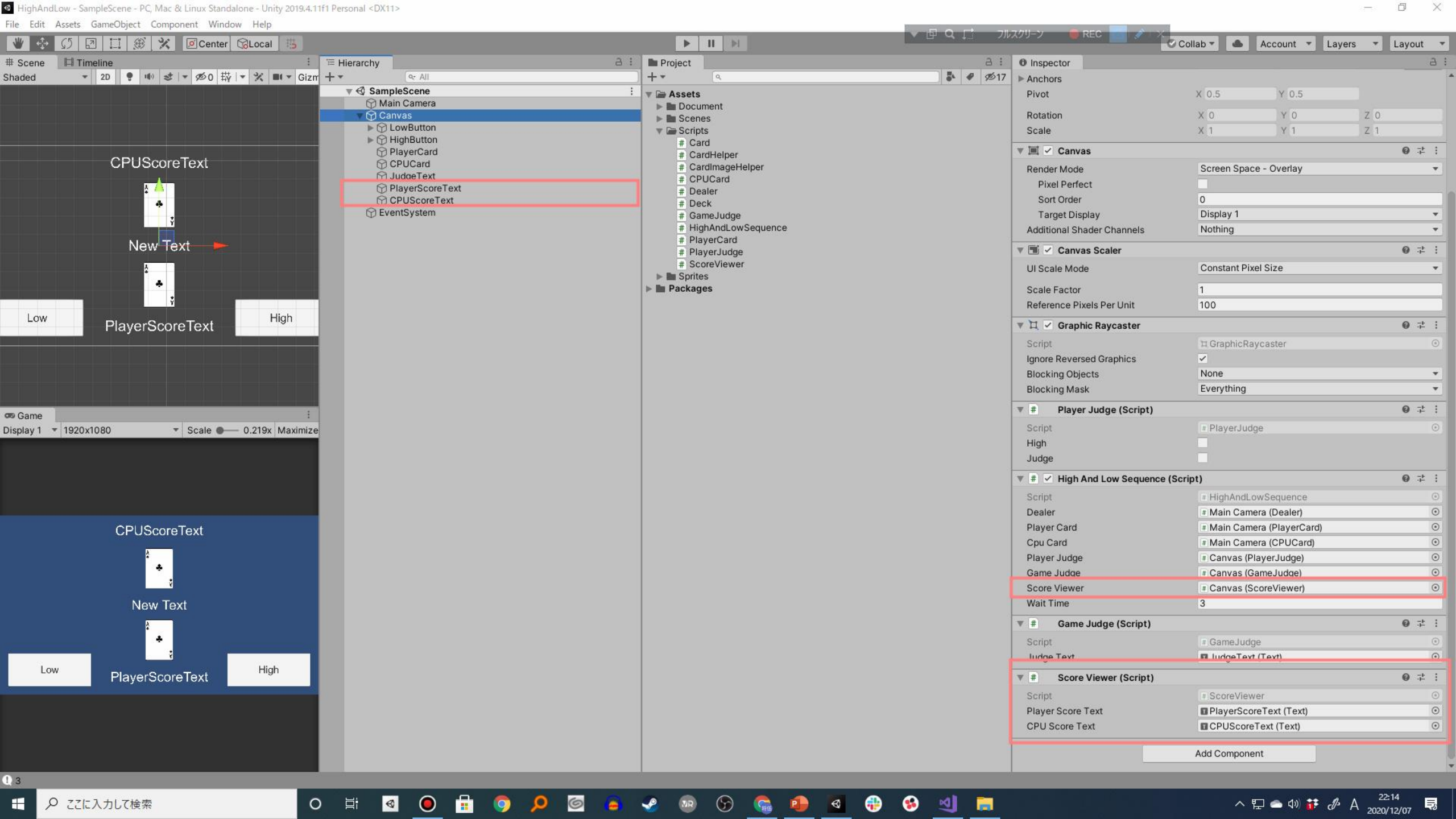
- それをUIに表示するところまでやってみましょう
- ①ScoreViewer.csを作成し、WinならばPlayer側のTextに2枚づつ加算されるメソッドを作ります
- ② ScoreViewer.cs内に、LoseならばCPU側のTextに2枚づつ加算されるメソッドを作成します
- 最後にUnityで設定してみましょう



```
public class ScoreViewer : MonoBehaviour
{
    public Text PlayerScoreText;
    public Text CPUScoreText;
    public int playerScore = 0;
    public int cpuScore = 0;

    public void ScoreTextInit()
    {
        PlayerScoreText.text = string.Empty;
        CPUScoreText.text = string.Empty;
    }

    public void AddScoreViewer(bool isWin)
    {
        if (isWin)
        {
            playerScore += 2;
            PlayerScoreText.text = $"{playerScore}";
        }
        else
        {
            cpuScore += 2;
            CPUScoreText.text = $"{cpuScore}";
        }
    }
}
```

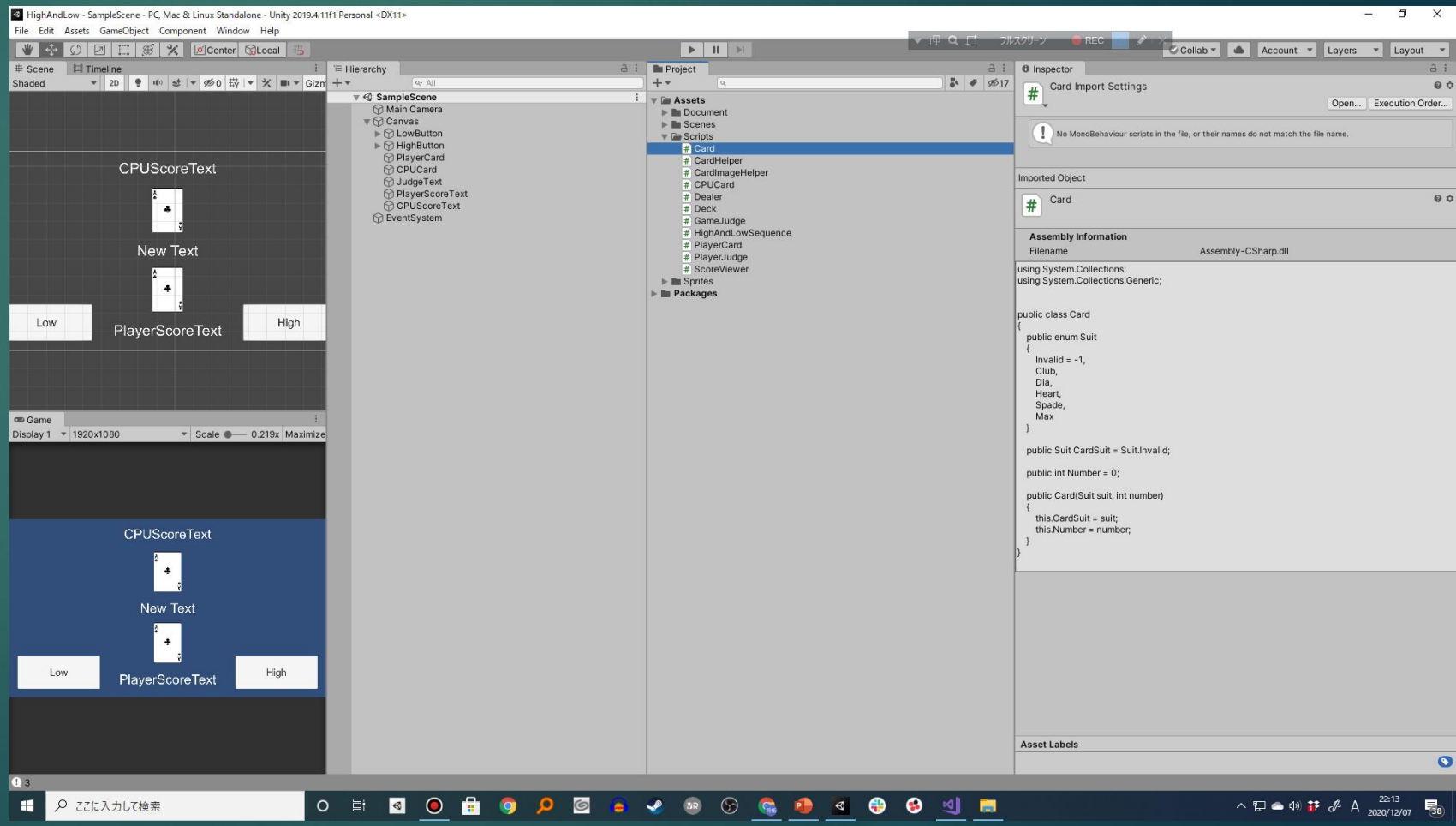

HighAndLow - SampleScene - PC, Mac & Linux Standalone - Unity 2019.4.11f1 Personal <DX11>
File Edit Assets GameObject Component Window Help
Center Local
Collab Account Layers Layout
Scene Timeline
Shaded 2D
CPUScoreText
New Text
Low
PlayerScoreText
High
Game
Display 1 1920x1080 Scale 0.219x Maximize
CPUScoreText
New Text
Low
PlayerScoreText
High
Hierarchy
SampleScene
Main Camera
Canvas
LowButton
HighButton
PlayerCard
CPUCard
JudgeText
PlayerScoreText
CPUScoreText
EventSystem
Project
Assets
Document
Scenes
Scripts
Card
CardHelper
CardImageHelper
CPUCard
Dealer
Deck
GameJudge
HighAndLowSequence
PlayerCard
PlayerJudge
ScoreViewer
Sprites
Packages
Inspector
Anchors
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Canvas
Render Mode Screen Space - Overlay
Pixel Perfect
Sort Order 0
Target Display Display 1
Additional Shader Channels Nothing
Canvas Scaler
UI Scale Mode Constant Pixel Size
Scale Factor 1
Reference Pixels Per Unit 100
Graphic Raycaster
Script GraphicRaycaster
Ignore Reversed Graphics
Blocking Objects None
Blocking Mask Everything
Player Judge (Script)
Script PlayerJudge
High
Judge
High And Low Sequence (Script)
Script HighAndLowSequence
Dealer Main Camera (Dealer)
Player Card Main Camera (PlayerCard)
Cpu Card Main Camera (CPUCard)
Player Judge Canvas (PlayerJudge)
Game Judge Canvas (GameJudge)
Score Viewer Canvas (ScoreViewer)
Wait Time 3
Game Judge (Script)
Script GameJudge
Judge Text JudgeText (Text)
Score Viewer (Script)
Script ScoreViewer
Player Score Text PlayerScoreText (Text)
CPU Score Text CPUScoreText (Text)
Add Component

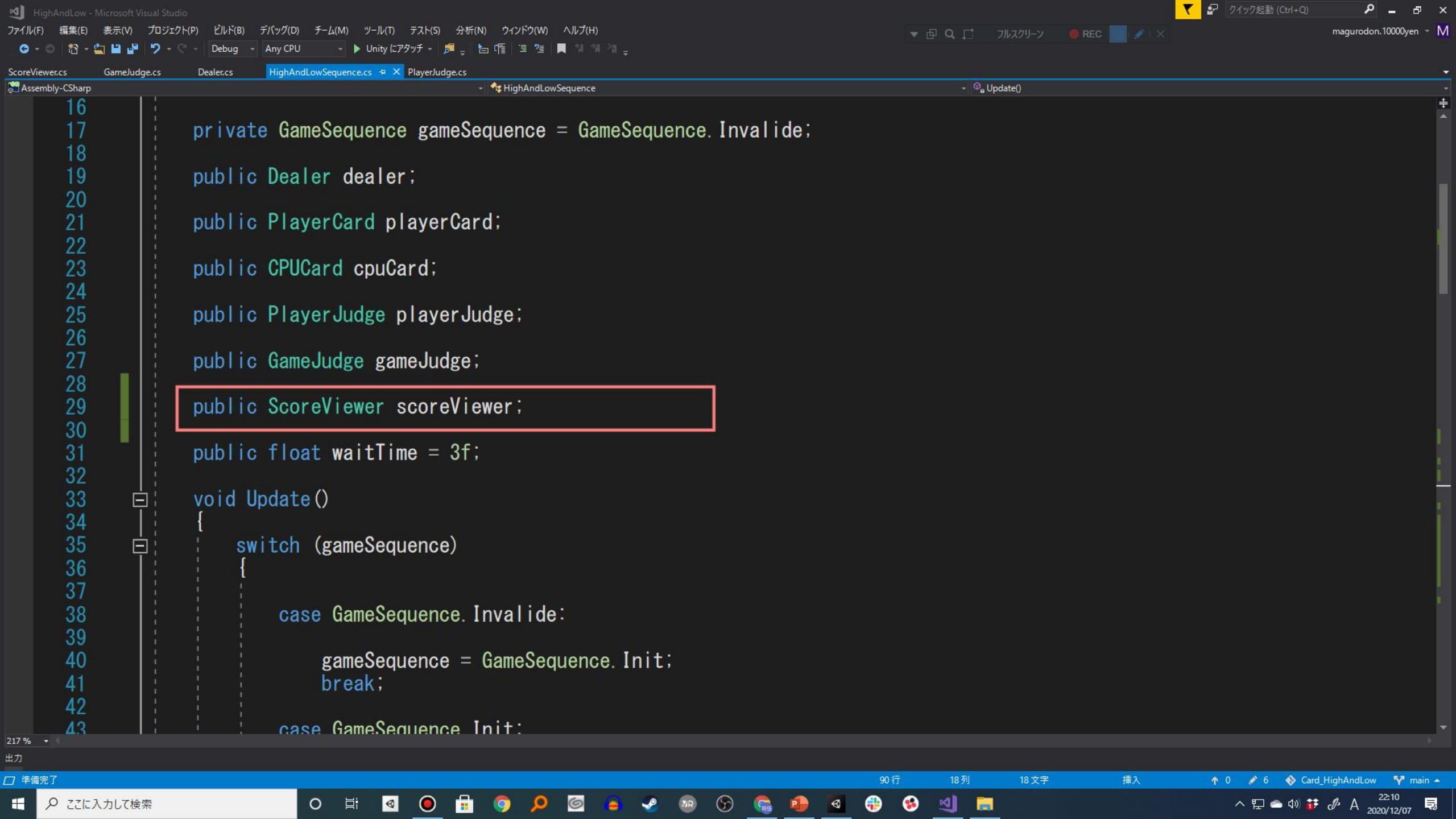
Unity

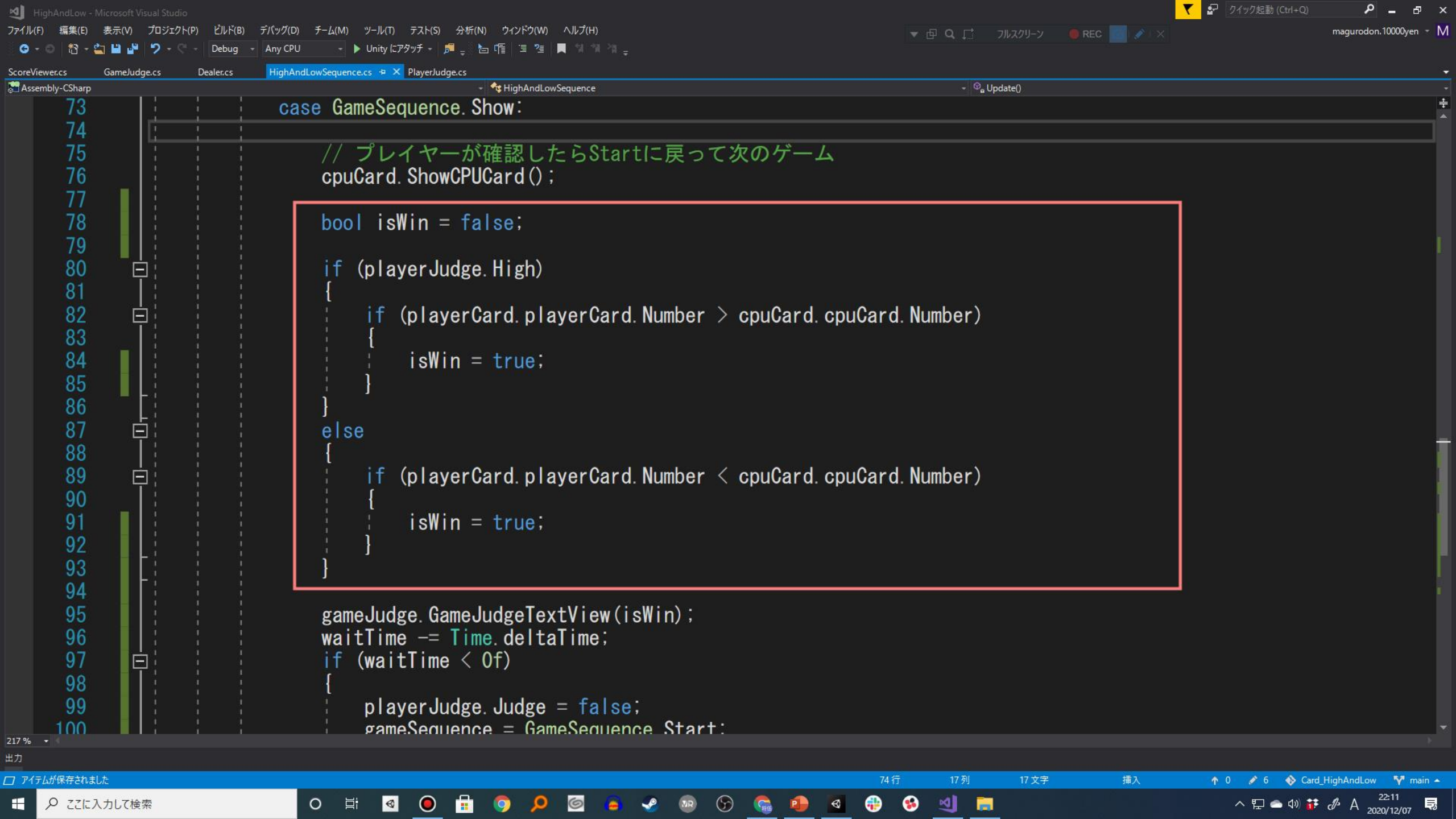
ハイアンドロー

- 今のままだと、なんらかの不具合やバグっぽい挙動が出ると思いますので、このあたりでリファクタリングを行います

- ※リファクタリングとは
- 記述したコードを読みやすく、また、保守しやすく書き換える行為です
- 基本的には同じような処理を二回以上書いている場合、リファクタリングを試みてください




HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
ScoreViewer.cs
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
Update()
16
17 private GameSequence gameSequence = GameSequence.Invalide;
18
19 public Dealer dealer;
20
21 public PlayerCard playerCard;
22
23 public CPUCard cpuCard;
24
25 public PlayerJudge playerJudge;
26
27 public GameJudge gameJudge;
28
29 public ScoreViewer scoreViewer;
30
31 public float waitTime = 3f;
32
33 void Update()
34 {
35 switch (gameSequence)
36 {
37
38 case GameSequence.Invalide:
39
40 gameSequence = GameSequence.Init;
41 break;
42
43 case GameSequence.Init;
217 %
出力
準備完了
90 行
18 列
18 文字
挿入
Card_HighAndLow
main
ここに入力して検索
2020/12/07


HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
ScoreViewer.cs
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
HighAndLowSequence
Update()
case GameSequence.Show:
// プレイヤーが確認したらStartに戻って次のゲーム
cpuCard.ShowCPUCard();
bool isWin = false;
if (playerJudge.High)
{
if (playerCard.playerCard.Number > cpuCard.cpuCard.Number)
{
isWin = true;
}
}
else
{
if (playerCard.playerCard.Number < cpuCard.cpuCard.Number)
{
isWin = true;
}
}
gameJudge.GameJudgeTextView(isWin);
waitTime -= Time.deltaTime;
if (waitTime < 0f)
{
playerJudge.Judge = false;
gameSequence = GameSequence.Start;
217 %
出力
アイテムが保存されました
74 行
17 列
17 文字
挿入
Card_HighAndLow
main
ここに入力して検索
2020/12/07

```
82                  if (playerCard.playerCard.Number > cpuCard.cpuCard.Number)
83                  {
84                      isWin = true;
85                  }
86              }
87              else
88              {
89                  if (playerCard.playerCard.Number < cpuCard.cpuCard.Number)
90                  {
91                      isWin = true;
92                  }
93              }
94
95              gameJudge.GameJudgeTextView(isWin);
96              waitTime -= Time.deltaTime;
97              if (waitTime < 0f)
98              {
99                  playerJudge.Judge = false;
100                 gameSequence = GameSequence.Start;
101                 scoreViewer.AddScoreViewer(isWin);
102                 waitTime = 3f;
103             }
104
105             break;
106         }
107     }
108
109 }
```

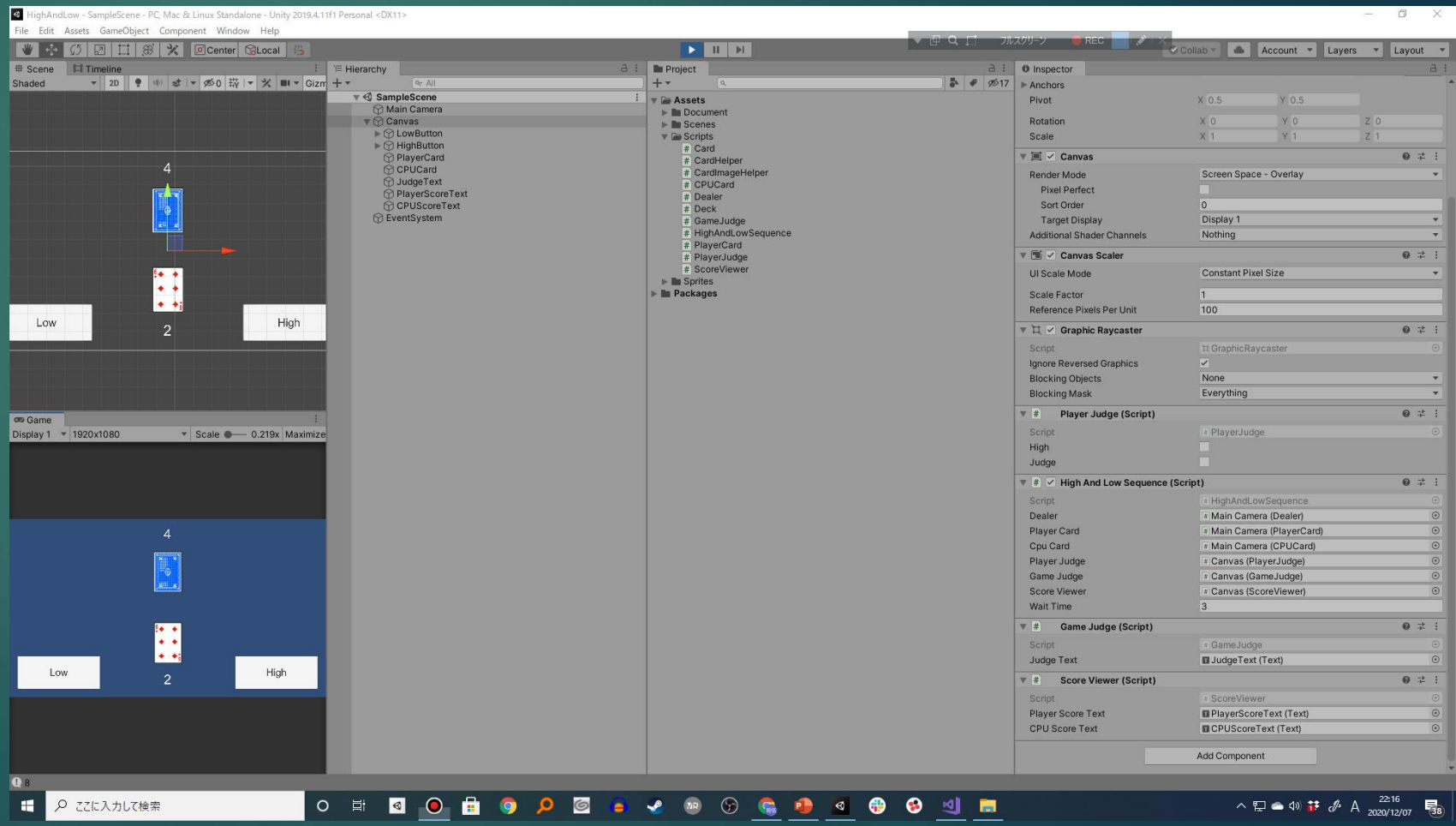
Unity
ハイアンドロー

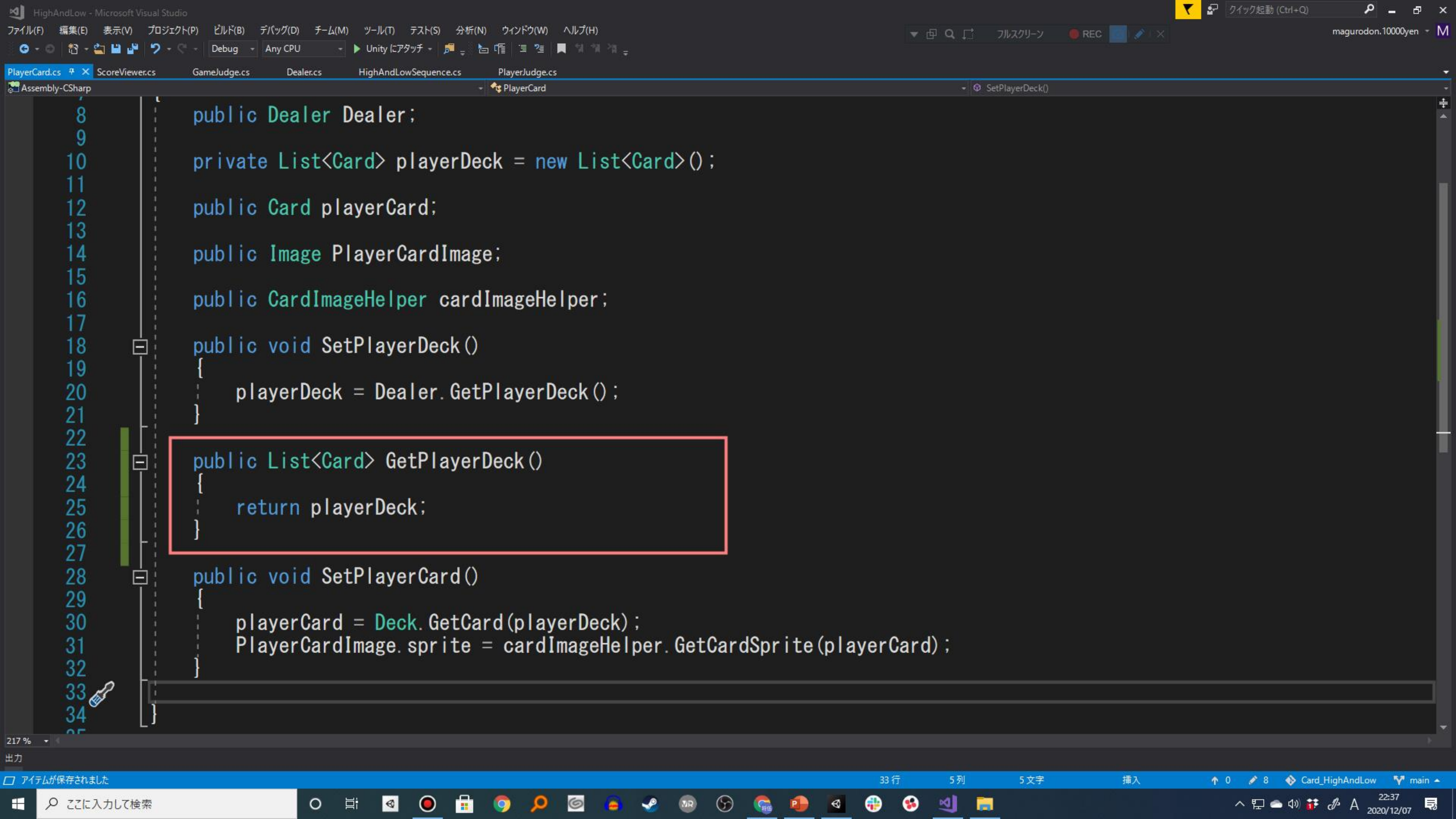
- ここまでできれば、あとはゲームの終了条件を整えてあげれば、ゲームのシーケンスとして完璧に回ることになります

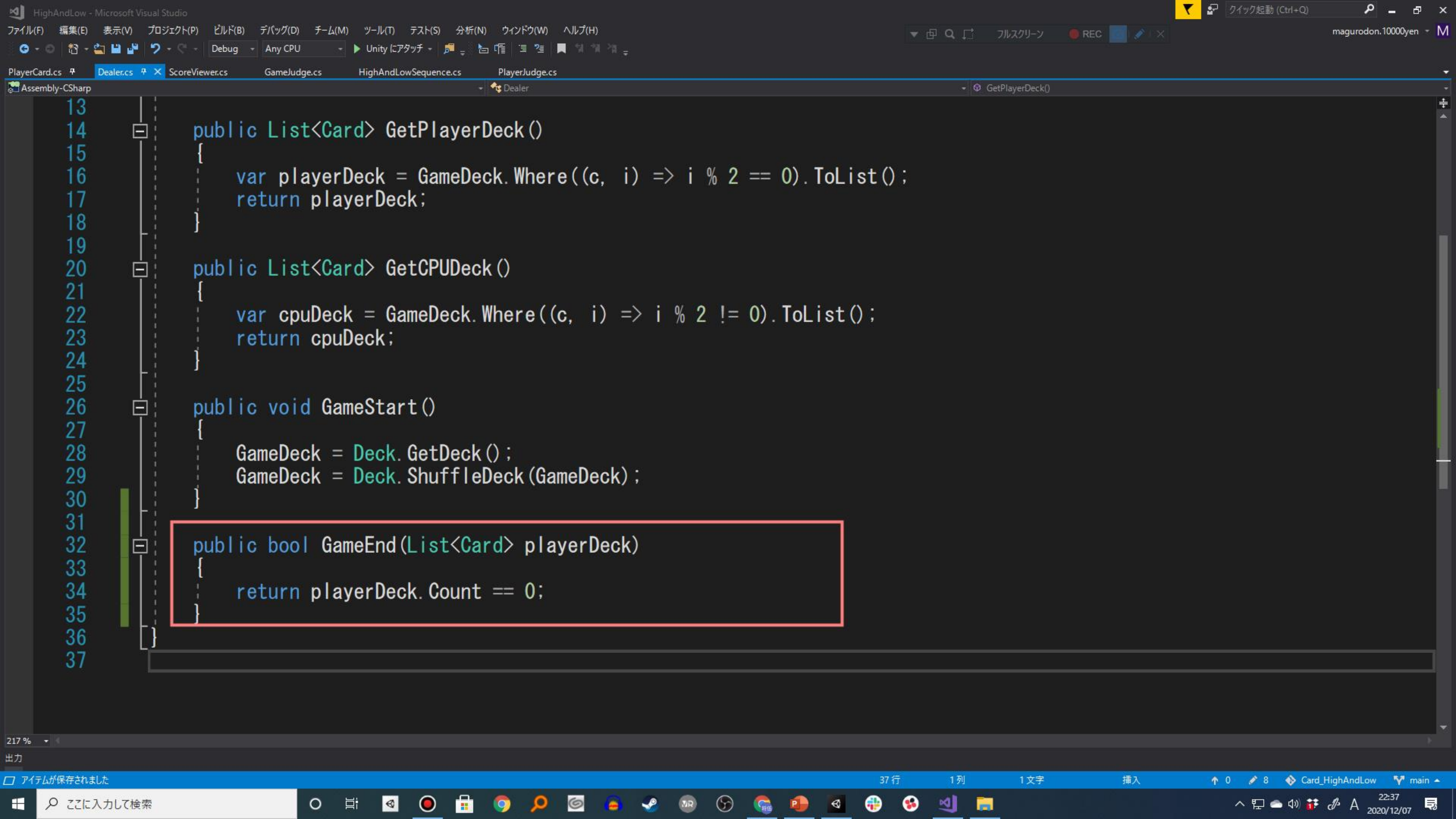
- 終了条件は1デッキ52枚全て使いきった場合、最終結果をJudgeTextに表示します

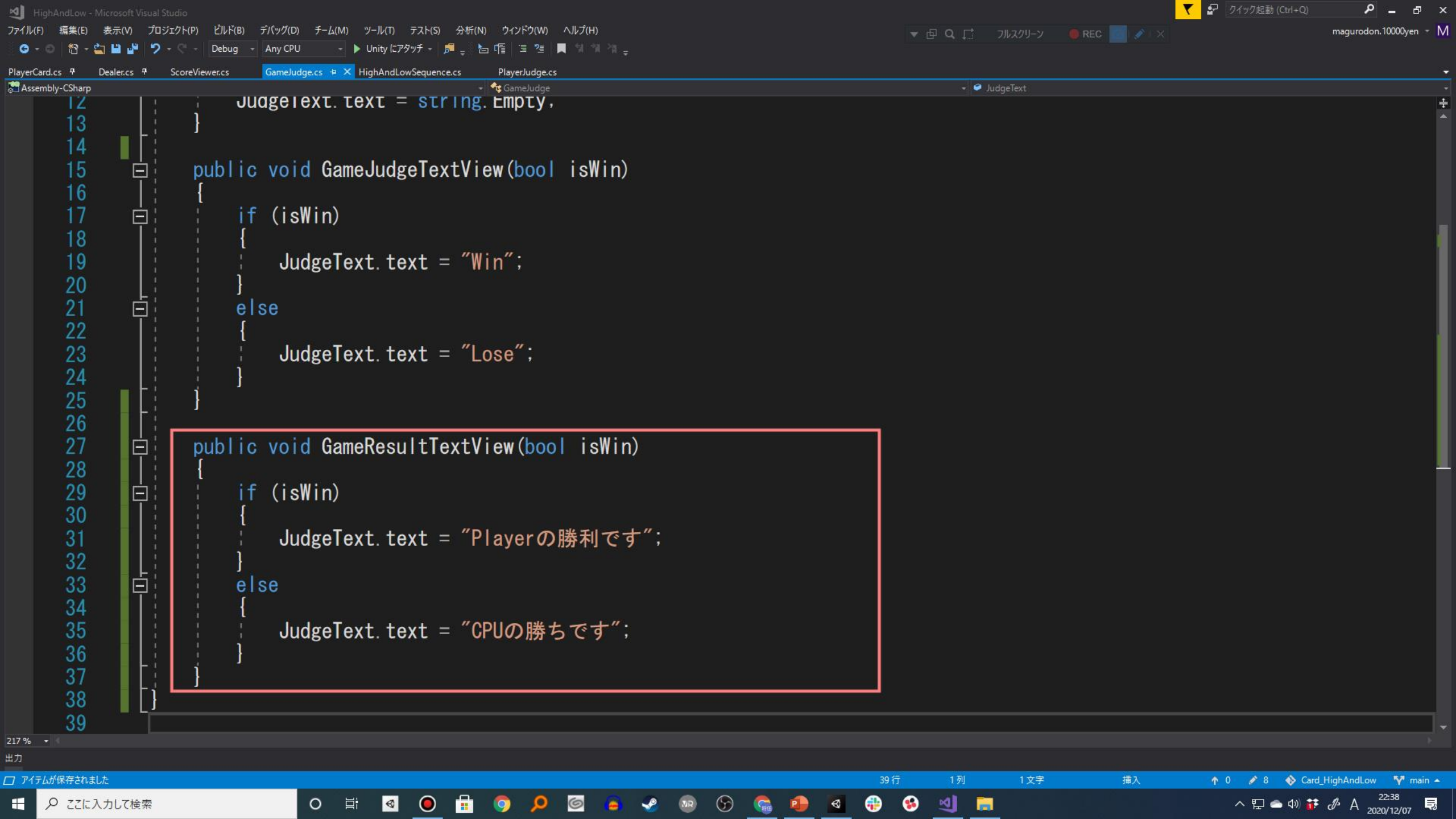
- PlayerCard.cs
- Dealer.cs
- GameJudge.cs
- HighAndLowSequence.cs

- 上記スクリプトに変更を加えていきます




HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
PlayerCard.cs
ScoreViewer.cs
GameJudge.cs
Dealer.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
PlayerCard
SetPlayerDeck()
public Dealer Dealer;
private List<Card> playerDeck = new List<Card>();
public Card playerCard;
public Image PlayerCardImage;
public CardImageHelper cardImageHelper;
public void SetPlayerDeck()
{
playerDeck = Dealer.GetPlayerDeck();
}
public List<Card> GetPlayerDeck()
{
return playerDeck;
}
public void SetPlayerCard()
{
playerCard = Deck.GetCard(playerDeck);
PlayerCardImage.sprite = cardImageHelper.GetCardSprite(playerCard);
}
}
217 %
出力
アイテムが保存されました
33 行
5 列
5 文字
挿入
Card_HighAndLow
main
ここに入力して検索
22:37
2020/12/07


HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
PlayerCard.cs
Dealer.cs
ScoreViewer.cs
GameJudge.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
Dealer
GetPlayerDeck()
13
14 public List<Card> GetPlayerDeck()
15 {
16 var playerDeck = GameDeck.Where((c, i) => i % 2 == 0).ToList();
17 return playerDeck;
18 }
19
20 public List<Card> GetCPUDeck()
21 {
22 var cpuDeck = GameDeck.Where((c, i) => i % 2 != 0).ToList();
23 return cpuDeck;
24 }
25
26 public void GameStart()
27 {
28 GameDeck = Deck.GetDeck();
29 GameDeck = Deck.ShuffleDeck(GameDeck);
30 }
31
32 public bool GameEnd(List<Card> playerDeck)
33 {
34 return playerDeck.Count == 0;
35 }
36 }
37
217 %
出力
アイテムが保存されました
37 行
1 列
1 文字
挿入
Card_HighAndLow
main
ここに入力して検索
22:37
2020/12/07


HighAndLow - Microsoft Visual Studio
ファイル(F) 編集(E) 表示(V) プロジェクト(P) ビルド(B) デバッグ(D) チーム(M) ツール(T) テスト(S) 分析(N) ウィンドウ(W) ヘルプ(H)
Debug
Any CPU
Unity にアタッチ
PlayerCard.cs
Dealer.cs
ScoreViewer.cs
GameJudge.cs
HighAndLowSequence.cs
PlayerJudge.cs
Assembly-CSharp
GameJudge
JudgeText
JudgeText.text = string.Empty;
}
public void GameJudgeTextView(bool isWin)
{
if (isWin)
{
JudgeText.text = "Win";
}
else
{
JudgeText.text = "Lose";
}
}
public void GameResultTextView(bool isWin)
{
if (isWin)
{
JudgeText.text = "Playerの勝利です";
}
else
{
JudgeText.text = "CPUの勝ちです";
}
}
}
217 %
出力
アイテムが保存されました
39 行
1 列
1 文字
挿入
Card_HighAndLow
main
ここに入力して検索
22:38
2020/12/07

```
using System.Collections.Generic;
using UnityEngine;

public class HighAndLowSequence : MonoBehaviour
{
    private enum GameSequence
    {
        Invalide,
        Init,
        Start,
        Deal,
        PlayerJudge,
        Show,
        Result
    }

    private GameSequence gameSequence = GameSequence.Invalide;

    public Dealer dealer;

    public PlayerCard playerCard;

    public CPUCard cpuCard;

    public PlayerJudge playerJudge;

    public GameJudge gameJudge;
```

```
92              }
93
94              gameJudge.GameJudgeTextView(isWin);
95              waitTime -= Time.deltaTime;
96              if (waitTime < 0f)
97              {
98                  playerJudge.Judge = false;
99
100                 if (dealer.GameEnd(playerCard.GetPlayerDeck()))
101                 {
102                     gameSequence = GameSequence.Result;
103                 }
104                 else
105                 {
106                     gameSequence = GameSequence.Start;
107                 }
108
109                 scoreViewer.AddScoreViewer(isWin);
110                 waitTime = 1f;
111             }
112
113             break;
114
115         case GameSequence.Result:
116
117             bool isResultWin = false;
118
119             if (scoreViewer.playerScore > scoreViewer.cpuScore)
```

```
                }

                scoreViewer.AddScoreViewer(isWin);
                waitTime = 1f;
            }

            break;

        case GameSequence.Result:

            bool isResultWin = false;

            if (scoreViewer.playerScore > scoreViewer.cpuScore)
            {
                isResultWin = true;
            }
            gameJudge.GameResultTextView(isResultWin);
            break;

        }
    }
}
```

Unity

ハイアンドロー

- ここでScenesフォルダ内にGameLaunchシーンとGameResultシーンを作成します
- あとはRollABallの時を思い出していただきます
- GameLaunchシーンでゲーム本編へ
- ゲーム本編が終われば GameResultシーンへ行ってください
- 実習です