

Sprint 03

Assignment 02: Achievements

Step 01: Setup

- First download and import the “*GooglePlayGameService*” asset from the GitHub.
- Then, after importing the asset, change the GameBuild platform to “*Android*”.
- Then goto Windows > Google Play Games > Setup > Android Setup...
- Paste the XML Code in the Box, if it is not available already. And then click “*Setup*”.

Google Play Games - Android Configuration

To configure Google Play Games in this project, go to the Play Game console, then enter the information below and click on the Setup button.
[Open Play Games Console](#)

Constants class name
Enter the fully qualified name of the class to create containing the constants

Directory to save constant: Assets/ThirdParty/GooglePlayGames

Constants class name: GPGSIds

Resources Definition
Paste in the Android Resources from the Play Console

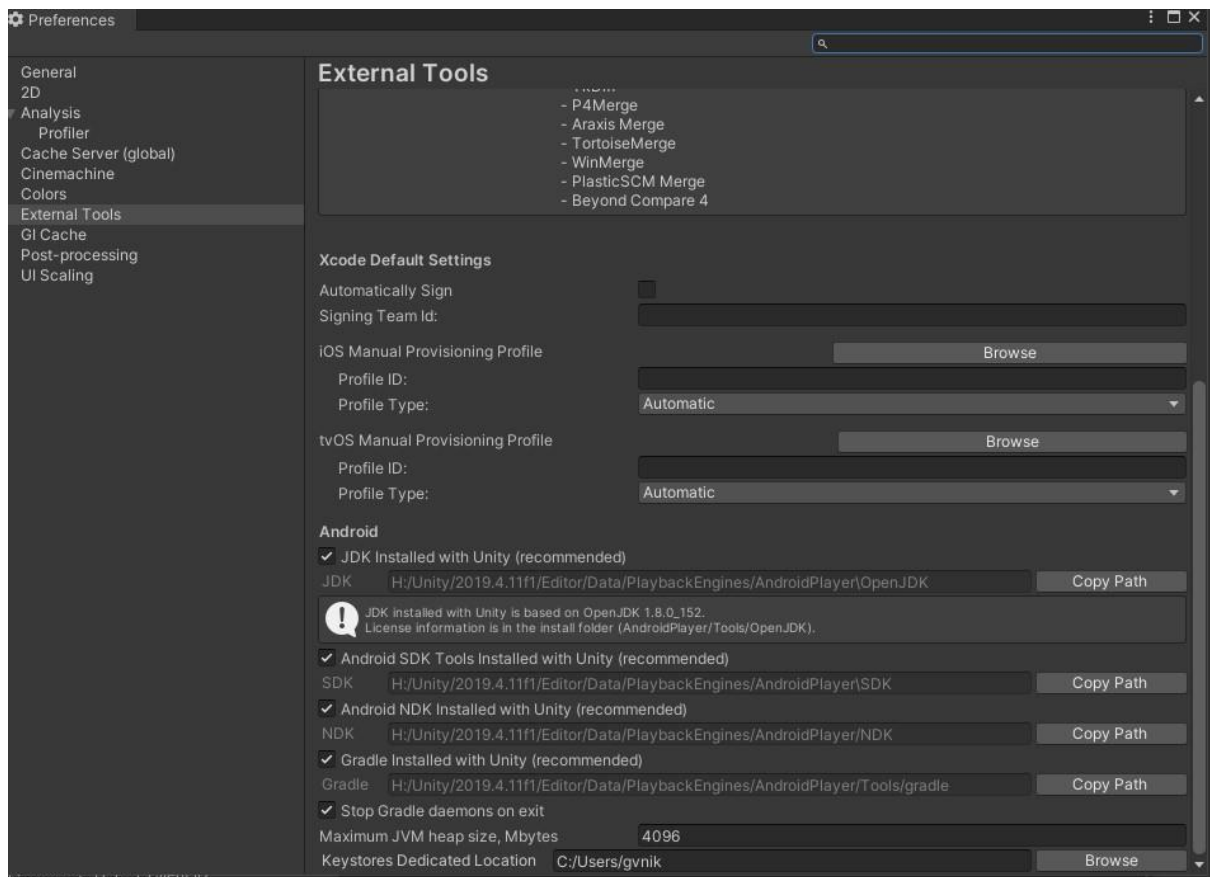
```
<?xml version="1.0" encoding="utf-8"?>
<!--Google Play game services IDs. Save this file as res/values/games-ids.xml in your project.-->
<resources>
  <!--app_id-->
  <string name="app_id" translatable="false">149052823565</string>
  <!--package_name-->
  <string name="package_name" translatable="false"></string>
  <!--achievement Is it all this easy?-->
  <string name="achievement_Is_it_all_this_easy" translatable="false">Cgkljbd2oasEEAIQAA</string>
</resources>
```

Web App Client ID (Optional)
The web app client ID is needed to access the user's ID token and call other APIs on behalf of the user. It is not required for Game Services.
Enter your oauth2 client ID below.
To obtain this ID, generate a web linked app in Developer Console. Example:
123456789012-abcdefghijklm.apps.googleusercontent.com

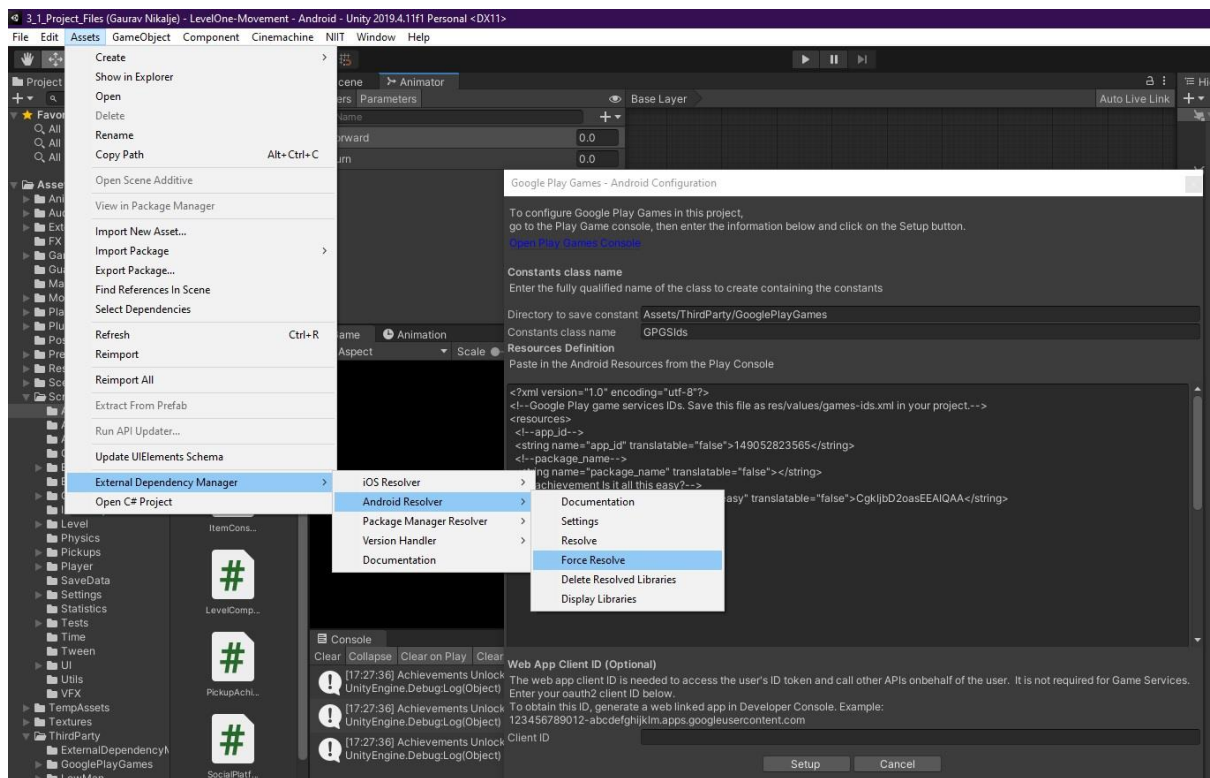
Client ID

Setup Cancel

- Then if we get some errors, to fix them, goto Edit > Preferences.
- And then toggle OFF and ON the “JDK, Android SDK & NDK”.



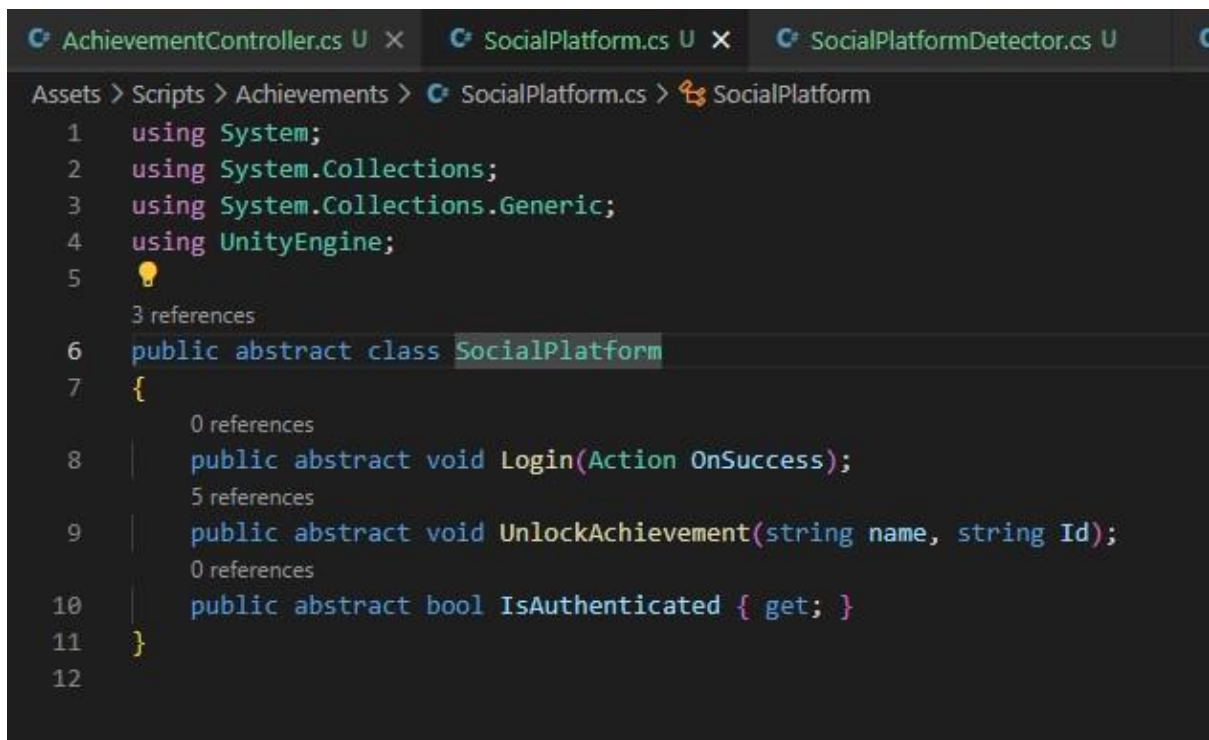
- Then next “Force Resolve” to get rid of all the errors.



Step 02: Script & Workflow

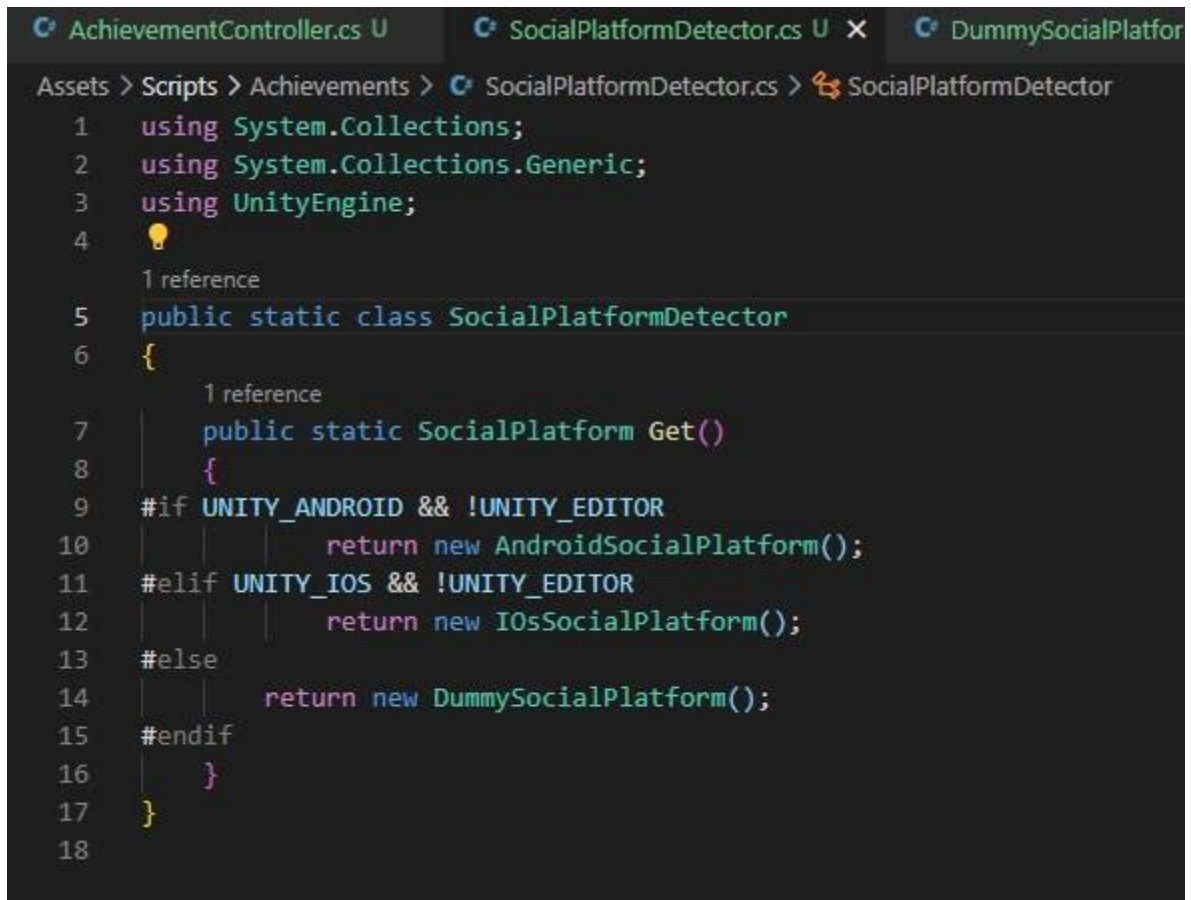
- The Scripts workflow is like this:
 - AchievementController > SocialPlatform, SocialPlatformDetector, LevelCompleteAchievement, ItemConsumedAchievement, EnemiesKilledAchievements, PickupAchievements & SpeedAchievements.
 - SocialPlatform > DummySocialPlatform & AndroidSocialPlatform.
 - AchievementController > LevelController & GameObject.
 - SocialPlatform & SocialPlatformDetector.


❖ SocialPlatform.



```
Assets > Scripts > Achievements > SocialPlatform.cs > SocialPlatform
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using UnityEngine;
5  ⚡
   3 references
6  public abstract class SocialPlatform
7  {
   0 references
8      public abstract void Login(Action OnSuccess);
   5 references
9      public abstract void UnlockAchievement(string name, string Id);
   0 references
10     public abstract bool IsAuthenticated { get; }
11 }
12
```

❖ SocialPlatformDetector.



```
Assets > Scripts > Achievements > SocialPlatformDetector.cs > SocialPlatformDetector
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  
   1 reference
5  public static class SocialPlatformDetector
6  {
   1 reference
7      public static SocialPlatform Get()
8      {
9          #if UNITY_ANDROID && !UNITY_EDITOR
10             return new AndroidSocialPlatform();
11         #elif UNITY_IOS && !UNITY_EDITOR
12             return new IOSSocialPlatform();
13         #else
14             return new DummySocialPlatform();
15         #endif
16     }
17 }
18
```

❖ DummySocialPlatform.

```
AchievementController.cs U DummySocialPlatform.cs U X AndriodSocialPlatform.c
Assets > Scripts > Achievements > DummySocialPlatform.cs > DummySocialPlatform
1 using System;
2 using System.Collections;
3 using System.Collections.Generic;
4 using UnityEngine;
5 
6 1 reference
7 public class DummySocialPlatform : SocialPlatform
8 {
9     0 references
10     public override bool IsAuthenticated { get { return true; } }
11     0 references
12     public override void Login(Action OnSuccess)
13     {
14         Debug.Log("Login Attempted !!!");
15     }
16     5 references
17     public override void UnlockAchievement(string name, string Id)
18     {
19         Debug.Log("Achievements Unlocked : " + name + " " + Id);
20     }
21 }
```

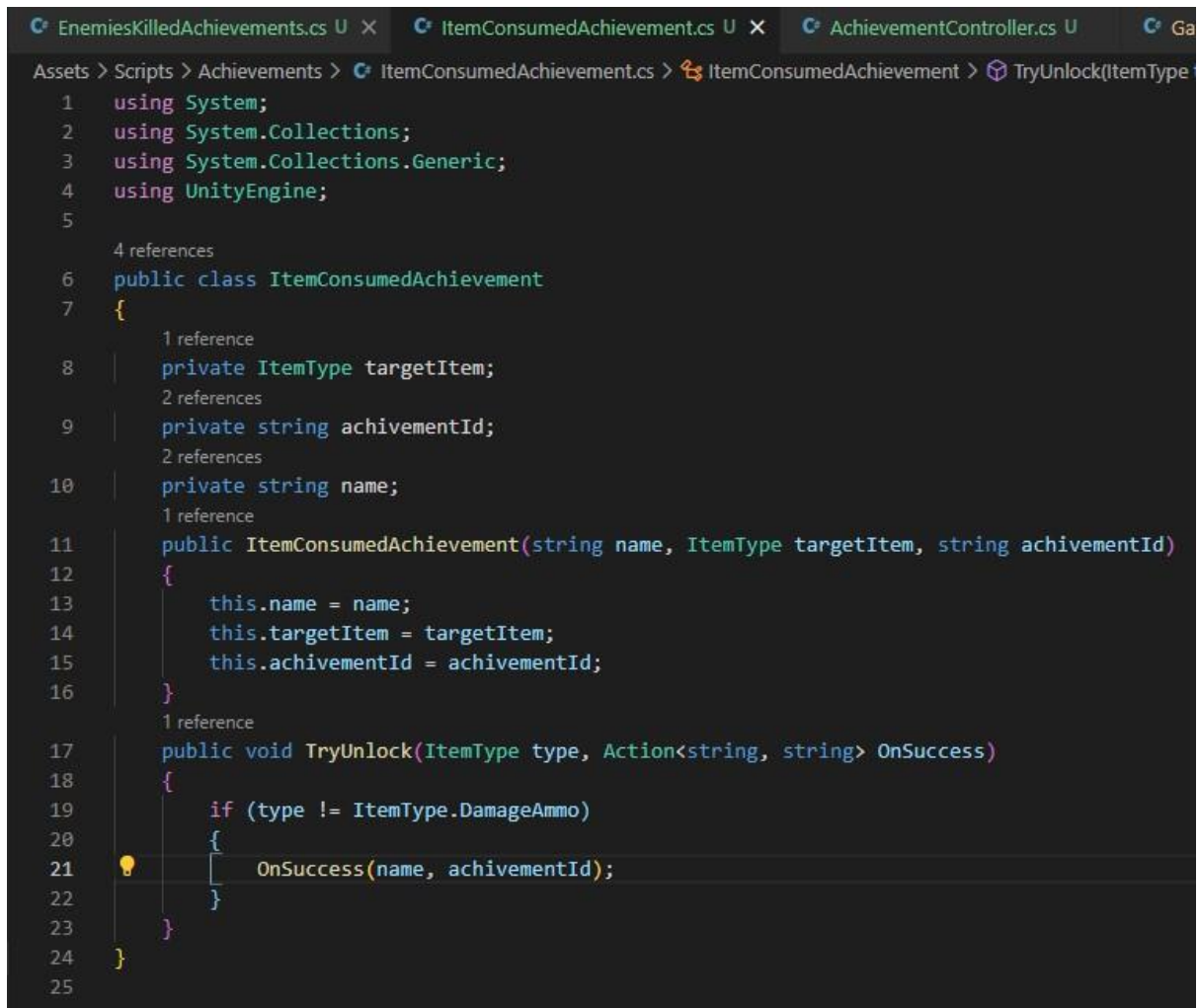
❖ AndroidSocialPlatform.

```
AchievementController.cs U  AndriodSocialPlatform.cs U X  GameController.cs M  GuardManager.cs M
Assets > Scripts > Achievements > AndriodSocialPlatform.cs
1  #if UNITY_ANDROID && !UNITY_EDITOR
2  using System;
3  using System.Collections;
4  using System.Collections.Generic;
5  using UnityEngine;
6  using GooglePlayServices;
7  using GooglePlayGames.BasicApi;
8  using GooglePlayGames;
9
10 public class AndroidSocialPlatform : SocialPlatform
11 {
12     public override bool IsAuthenticated
13     {
14         get { return Social.localUser.authenticated; }
15     }
16
17     public override void Login(Action OnSuccess)
18     {
19         PlayGamesClientConfiguration config = new PlayGamesClientConfiguration.Builder().Build();
20         PlayGamesPlatform.InitializeInstance(config);
21         PlayGamesPlatform.DebugLogEnabled = true;
22         PlayGamesPlatform.Activate();
23         Social.localUser.Authenticate((success) =>
24         {
25             if(success)
26             {
27                 OnSuccess();
28             }
29         });
30     }
31
32     public override void UnlockAchievement(string name, string Id)
33     {
34         if (IsAuthenticated)
35         {
36             Social.ReportProgress(Id, 100, (success) => { });
37         }
38     }
39 }
40 #endif
41
```


❖ EnemiesKilledAchievement.

```
Assets > Scripts > Achievements > EnemiesKilledAchievements.cs > EnemiesKilledAchievements > OnGuardsSetup(int totalGuards)
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using UnityEngine;
5
6  7 references
7  public class EnemiesKilledAchievements
8  {
9      3 references
10     private string achivementId;
11     3 references
12     private string name;
13     4 references
14     private int guardsKilled;
15     3 references
16     private int killTarget;
17     2 references
18     private int pacifistMinimum;
19     2 references
20     private int guardsInLevel;
21     1 reference
22     public void Reset()
23     {
24         guardsKilled = 0;
25     }
26     2 references
27     public EnemiesKilledAchievements(string name, int target, int pacifistMinimum, string achivementId)
28     {
29         this.name = name;
30         this.killTarget = target;
31         this.achivementId = achivementId;
32         this.pacifistMinimum = pacifistMinimum;
33     }
34     1 reference
35     public void OnGuardKilled()
36     {
37         guardsKilled++;
38     }
39     1 reference
40     public void OnGuardsSetup(int totalGuards)
41     {
42         guardsInLevel = totalGuards;
43     }
44
45     1 reference
46     public void TryUnlock(Action<string, string> OnSuccess)
47     {
48         if(guardsKilled == 0 && killTarget == 0 && guardsInLevel >= pacifistMinimum)
49         {
50             OnSuccess(name, achivementId);
51         }
52         else if (guardsKilled >= killTarget)
53         {
54             OnSuccess(name, achivementId);
55         }
56     }
57 }
```


❖ ItemsConsumedAchievement.



```
1 using System;
2 using System.Collections;
3 using System.Collections.Generic;
4 using UnityEngine;
5
6 4 references
7 public class ItemConsumedAchievement
8 {
9     1 reference
10     private ItemType targetItem;
11     2 references
12     private string achievementId;
13     2 references
14     private string name;
15     1 reference
16     public ItemConsumedAchievement(string name, ItemType targetItem, string achievementId)
17     {
18         this.name = name;
19         this.targetItem = targetItem;
20         this.achievementId = achievementId;
21     }
22     1 reference
23     public void TryUnlock(ItemType type, Action<string, string> OnSuccess)
24     {
25         if (type != ItemType.DamageAmmo)
26         {
27             OnSuccess(name, achievementId);
28         }
29     }
30 }
```

❖ LevelCompleteAchievement.

```
Assets > Scripts > Achievements > LevelCompleteAchievement.cs > ...
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using UnityEngine;
5
6  7 references
   public class LevelCompleteAchievement
7  {
8      2 references
       private int targetLevel;
9      2 references
       private string achivementId;
10     2 references
       private string name;
11     4 references
       public LevelCompleteAchievement(string name, int targetLevel, string achivementId)
12     {
13         this.name = name;
14         this.targetLevel = targetLevel;
15         this.achivementId = achivementId;
16     }
17     1 reference
       public void TryUnlock(int completedLevelId, Action<string, string> OnSuccess)
18     {
19         if(completedLevelId == targetLevel)
20         {
21             OnSuccess(name, achivementId);
22         }
23     }
24 }
25
```

❖ PickupAchievement.

```
Assets > Scripts > Achievements > PickupAchievement.cs > PickupAchievement > OnPickupCollected(Action<string, string> onSuccess)

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  5 references
8  public class PickupAchievement
9  {
10     3 references
11     private int pickupsCollected;
12     2 references
13     private string name;
14     2 references
15     private int target;
16     2 references
17     private string achievementID;
18     1 reference
19     public PickupAchievement(string name, int target, string achievementID)
20     {
21         this.name = name;
22         this.target = target;
23         this.achievementID = achievementID;
24     }
25     1 reference
26     public void OnPickupCollected(Action<string, string> onSuccess)
27     {
28         pickupsCollected++;
29
30         if (pickupsCollected >= target)
31         {
32             onSuccess(name, achievementID);
33         }
34     }
35     1 reference
36     public void OnLevelComplete()
37     {
38         Reset();
39     }
40     1 reference
41     private void Reset()
42     {
43         pickupsCollected = 0;
44     }
45 }
```

❖ SpeedAchievement.

```
Assets > Scripts > Achievements > SpeedAchievement.cs > SpeedAchievement

1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using UnityEngine;
5
6  8 references
7  public class SpeedAchievement
8  {
9      4 references
10     private int targetTime;
11     3 references
12     private string achivementId;
13     3 references
14     private string name;
15     3 references
16     private SpeedAchievementType achievementType;
17     7 references
18     private List<SaveData.Level> levels;
19     2 references
20     public SpeedAchievement(string name, int targetTime, SpeedAchievementType achievementType, string achivementId)
21     {
22         this.name = name;
23         this.targetTime = targetTime;
24         this.achivementId = achivementId;
25         this.achievementType = achievementType;
26     }
27
28     1 reference
29     public void TryUnlock(float completedTime, int completedLevelId, Action<string, string> OnSuccess)
30     {
31         if (achievementType == SpeedAchievementType.Any && completedTime <= targetTime)
32         {
33             OnSuccess(name, achivementId);
34         }
35         else if (achievementType == SpeedAchievementType.All )
36         {
37             if(levels.Count > 1)
38             {
39                 return;
40             }
41             if(levels[0].ID == completedLevelId)
42             {
43                 if(completedTime<targetTime)
44                 {
45                     OnSuccess(name, achivementId);
46                 }
47             }
48         }
49     }
50 }
```

```

37         OnSuccess(name, achivementId);
38     }
39 }
40 }
41 }
42
43     6 references
44     public enum SpeedAchievementType
45     {
46         2 references
47         All,
48         2 references
49         Any
50     }
51
52     1 reference
53     public void Load(SaveData saveData)
54     {
55         int totalLevels = Levels.ALL.Count;
56         levels = saveData.levels;
57         for (int i = levels.Count - 1; i >= 0; i--)
58         {
59             if(levels[i].score <= targetTime)
60             {
61                 levels.Remove(levels[i]);
62             }
63         }
64         Debug.Log("Levels Setup !");
65     }
66 }
67

```

❖ GameController.

```

GameController.cs M  GuardManager.cs M  LevelController.cs M
Assets > Scripts > GameController.cs > GameController
1  using System.Linq;
2  using UnityEngine;
3  using UnityEngine.SceneManagement;
4
5  1 reference
6  public class GameController : MonoBehaviour
7  {
8
9
10
11
12
13
14
15
16
17
18
19  private AchievementController achievementController; //S3 - Assignment 02
20

```



```

117 1 reference
118 private void OnLevelLoaded(Scene scene)
119 {
120     LevelController levelController = FindObjectOfType<LevelController>();
121     if (levelController != null)
122     {
123         levelController.OnLevelComplete += (score) =>
124         {
125             saveDataController.UpdateScore(saveData, levelController.levelID, score);
126             saveDataController.Save(saveData);
127             achievementController.OnLevelComplete(levelController.levelID, score); //S3 - Assignment 02
128         };
129
130         //S3 - Assignment 02
131         levelController.OnItemConsumed += (item) => achievementController.OnItemConsumed(item);
132         levelController.OnGuardKilled += () => achievementController.OnGuardKilled();
133         levelController.OnGuardsSetup += achievementController.OnGuardsSetup;
134         levelController.OnPickupCollected += () => achievementController.OnPickupCollected();
135
136         levelController.OnLevelLoadRequest += (levelID) => OnSceneLoadRequested(levelID);
137         levelController.OnExitRequest += () => OnLevelSelectLoadRequested();
138
139         globalAudio.OnLevelLoaded(scene, levelController);
140     }
141 }
142

```

❖ LevelController.

```

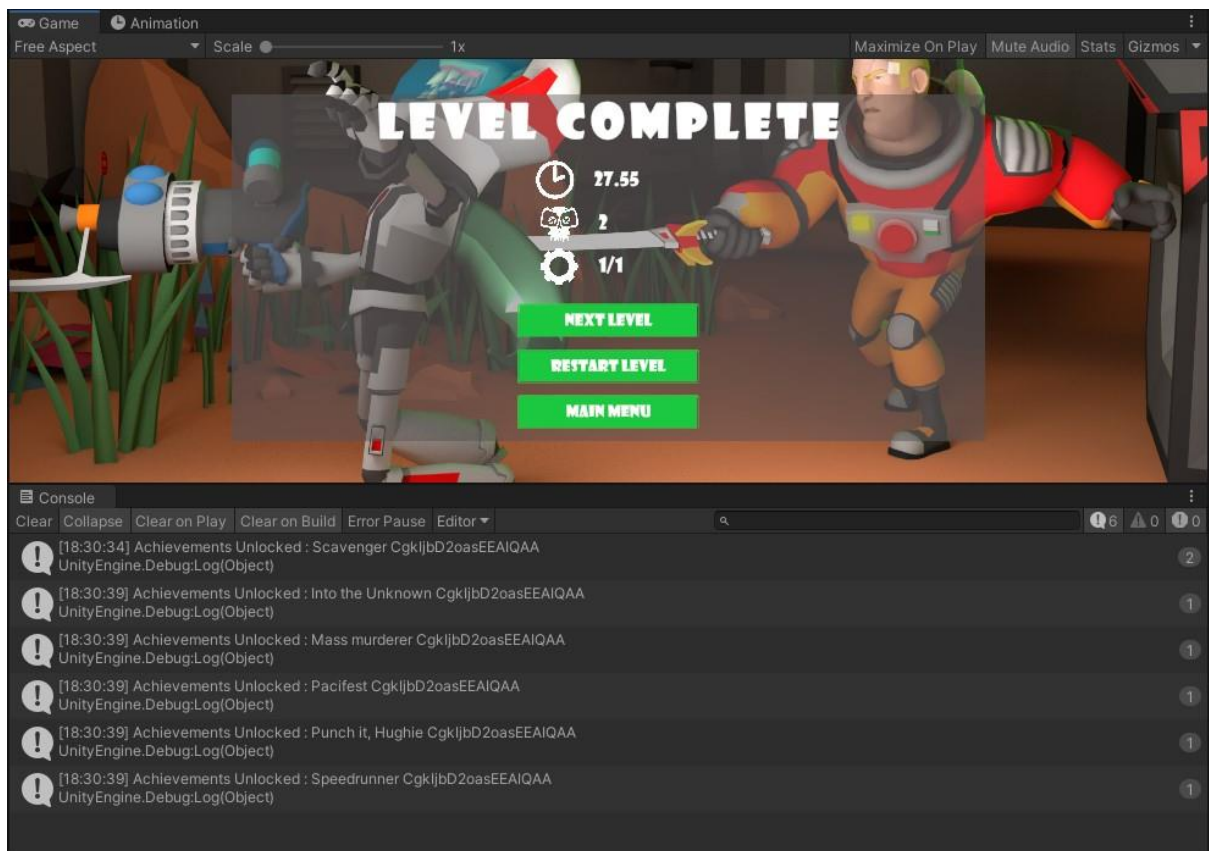
28 2 references
29 private ProjectileManager projectileManager;
30
31 2 references
32 public Action<int> OnGuardsSetup = delegate { }; //S3 - Assignment 02
33
34 0 references
35 public void Start()
36 {
37     LevelDependencies dependencies = GetComponentInChildren<LevelDependencies>();
38     if(dependencies == null)
39     {
40         Debug.LogError("Unable to find LevelDependencies. Cannot play level.");
41     }
42
43     TimeController timeController = new TimeController();
44
45     GuardEvents guardEvents = new GuardEvents();
46
47     GameObject playerObj = CreatePlayerObject(dependencies.player);
48
49     PlayerSettings playerSettings = playerObj.GetComponent<PlayerSettingsHolder>().playerSettings;
50
51     ProjectilePool projectilePool = new ProjectilePool(dependencies.projectileLibrary, new ProjectileFactory(playerSettings));
52     projectileManager = new ProjectileManager(projectilePool);
53
54     guardManager = new GuardManager(guardEvents, projectilePool);
55     guardManager.OnGuardsSetup += OnGuardsSetup; //S3 - Assignment 02
56
57     PickupEvents pickupEvents = new PickupEvents();
58

```


❖ GuardManager.

```
GuardManager.cs M X
Assets > Scripts > Guards > GuardManager.cs > GuardManager > OnLevelLoaded(Transform levelObjects)
13 private List<GuardSpawnerController> spawners;
14 public Action<int> OnGuardsSetup = delegate { }; //S3 - Assignment 02
15
16 public GuardManager(GuardEvents guardEvents, ProjectilePool projectilePool)
17 {
18     this.guardEvents = guardEvents;
19     this.projectilePool = projectilePool;
20
21     guards = new Dictionary<Guard, GuardController>();
22     spawners = new List<GuardSpawnerController>();
23 }
24
25 public void OnLevelLoaded(Transform levelObjects)
26 {
27     player = levelObjects.Find("Player").gameObject;
28     playerObjectData = player.GetComponent<PlayerObjectData>();
29
30     Guard[] guards = levelObjects.GetComponentsInChildren<Guard>(); //S3 - Assignment 02
31     GuardSpawner[] spawners = levelObjects.GetComponentsInChildren<GuardSpawner>(); //S3 - Assignment 02
32
33     int totalGuards = guards.Length + spawners.Length; //S3 - Assignment 02
34
35     foreach (Guard guard in guards) //levelObjects.GetComponentsInChildren<Guard>() //S3 - Assignment 02
36     {
37         SpawnGuard(guard);
38     }
39
40     foreach(GuardSpawner spawner in spawners) //levelObjects.GetComponentsInChildren<GuardSpawner>() //S3 - Assignment 02
41     {
42         SetUpSpawner(spawner);
43     }
44 }
45
```

❖ Final Result.



Step 03: What I Learnt

- **When would the Hidden property be set in the configuration of an achievement?**
 - Hidden Property helps the Player to understand how to achieve certain Achievements and Rewards, in a game.
 - It is kind of a guide or walkthrough for the Player to get the Reward or Achievement.

- But we should be aware while using this property, as excess use of it, may cause the much spoilers to the game, and the Player will feel bored after a time period.
 - As the Level Progresses, we can give the Player bigger achievements, so that the Player is attracted towards the Game.
-
- **Why might it not be the best idea to design an achievement set so that every achievement unlocks during the first play through?**
 - Because then the Player would think the Game is too easy, and may feel bored after completing the Level.
 - And also, if we give all the achievements since the First Level only, then there would be less achievements to receive in Next Levels.
 - So, the best thing to do is to first give the Player some achievements which the Player feels happy after receiving, and gets attracted towards the game.
 - Then, from the next levels onwards, we could increase the difficulty levels and give greater rewards in return.

-----**THE END**-----