



Hiding Master

Thank you for purchasing the asset!

The asset is mainly made for publishing and learning purposes, but you can use it as you like.

Please contact me if you have any questions/issues!

If you like the asset, please leave a review!

If you don't like it, or you have any questions, don't hesitate to send me an email!

You can use anything from the asset for any purpose.

Email: ragendom@gmail.com

Facebook: <https://www.facebook.com/Ragendom-283436829168712/>

Please use Unity 2019.1.1f1 to avoid errors

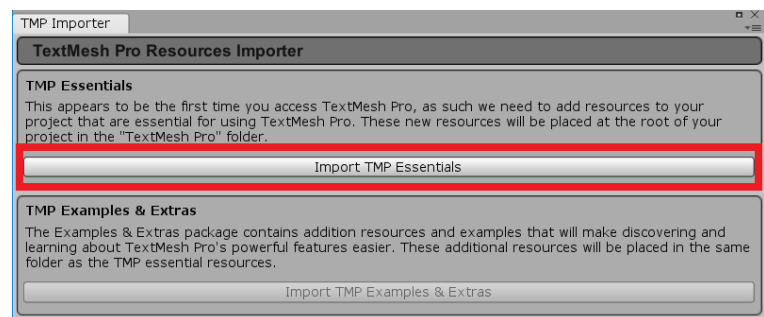
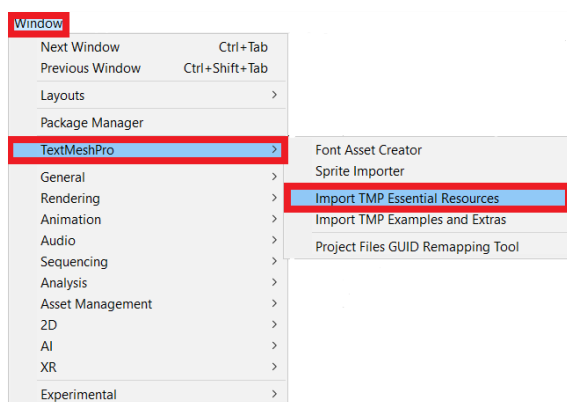
Credits:

Background music by Eric Matyas: <http://www.soundimage.org>

Sound effects: <https://www.noiseforfun.com>

IMPORTANT

1. In this asset I use **TextMesh Pro** for the text elements. First you have to import it.



SETTING UP THE PROJECT

[HOW TO CREATE NEW LEVELS](#)

[CLICK HERE TO WATCH A TUTORIAL VIDEO OF SETTING UP ONE OF MY GAMES](#)

1. Create a **new Unity project**
2. Double-click the **Unity package file** to import it to the project
3. Select All and **import** it
4. Double click one of the scenes inside the „**Scenes**” folder
5. **Import TextMesh Pro** as described previously
6. Go to „**File > Build Settings...**” and select **Andoid** or **iOS**, and click on „**Switch Platform**”
7. **Drag&drop the Scene(s)** from the „**Scenes**” folder below the „**Scenes In Build**” text at Build Settings

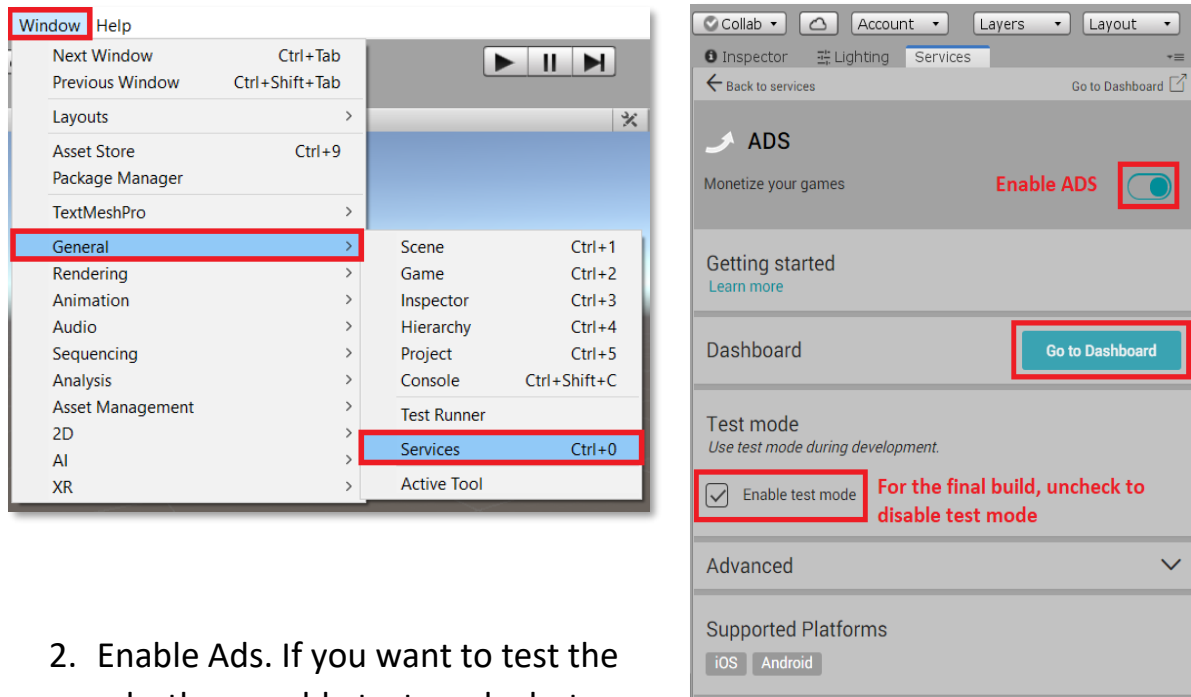
NOTE:

To test Admob ads, you only need to build it for your device, because Admob does not show up inside Unity. If you are ready to publish, then you need to change the Ad IDs and disable „Test Ads” variable as it is described in the next part of the documentation. The Ads will show up only after publishing.

IMPLEMENTING ADS

(Only do 1st, 2nd and 3rd steps if you want to use Unity Ads)

1. Open *Services*



2. Enable Ads. If you want to test the ads, then enable test mode, but **you will need to disable test mode for the final build.**

3. Close and reopen project!!!

4. Open StaticVariables.cs

You have to **change App ID, and AD IDs** for the final build.

You will need to disable test mode for the final build. You can select the AD network here.

FOR TESTING, LEAVE EVERYTHING AS IT IS

If you want to use Admob and you are ready to release, then TestAds = false, unityAds = false

If you are ready to release, then replace the IDs with your own Admob IDs

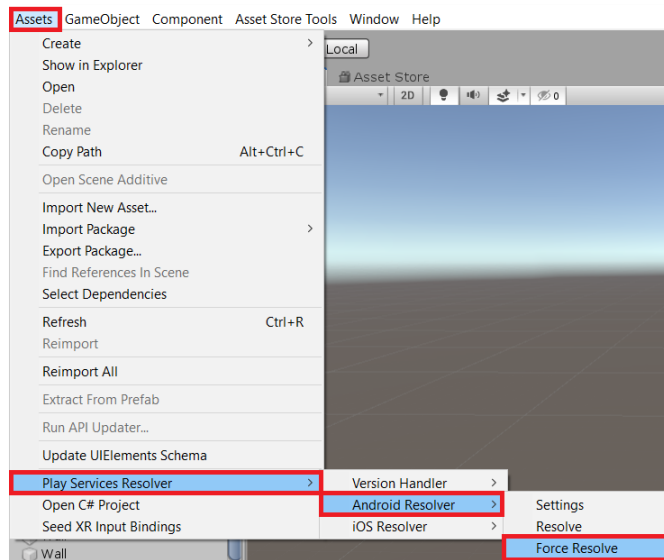
```
public class StaticVariables : MonoBehaviour
{
    public static int plusRewardCount = 10; //In this case, watching a rewarded
    public static int[] requiredTokensToUnlock = { 0, 15, 30}; //In this case
    costs 100. Change the values between the { } marks

    //Leave the following Admob IDs if you want to test the game
    public static bool TestAds = true; //This has to be true when testing, an
    public static bool unityAds = false; //Set this false if you want to use
    public static string appID = ""; //Paste your Admob appID between the "
    public static string bannerID = "ca-app-pub-3940256099942544/6300978111";
    public static string interstitialID = "ca-app-pub-3940256099942544/1033173712";
    the app
    public static string rewardVideoID = "ca-app-pub-3940256099942544/5224354917";
    app
}
```

5. Open **StaticVariables.cs**

You can change the amount of **tokens** the player gets after watching a **Reward Video Ad** here (*public static int **plusRewardCount***)

6. ***Resolve just before building the project.***



Calling ADS

Admob

```
FindObjectOfType<AdManager>().ShowAdmobBanner();  
FindObjectOfType<AdManager>().ShowAdmobInterstitial();  
FindObjectOfType<AdManager>().ShowAdmobRewardVideo();
```

Unity ADS

```
FindObjectOfType<AdManager>().ShowUnityVideoAd();  
FindObjectOfType<AdManager>().ShowUnityRewardVideoAd();
```

If you want to use Unity Ads, then after you have done the 1, 2, 3 steps, you have to **uncomment** some lines in the following scripts:

GameManager.cs **Uncomment lines: 21-23, 32-37**

```
19 void Start () {
20     //UNCOMMENT THE FOLLOWING LINES IF YOU ENABLED UNITY ADS AT UNITY SERVICES AND REOPENED THE PROJECT!
21     //if (FindObjectOfType<AdManager>().unityAds)
22     //    CallUnityAds();    //Calls Unity Ads
23     //else
24     //    CallAdmobAds();    //Calls Admob Ads
25
26     StartPanelActivation();
27     HighScoreCheck();
28     AudioCheck();
29 }
30
31 //UNCOMMENT THE FOLLOWING LINES IF YOU ENABLED UNITY ADS AT UNITY SERVICES AND REOPENED THE PROJECT!
32 //public void CallUnityAds()
33 //{
34 //    if (Time.time != Time.timeSinceLevelLoad)
35 //        FindObjectOfType<AdManager>().ShowUnityVideoAd();    //Shows Interstitial Ad when game starts (except for the first time)
36 //    FindObjectOfType<AdManager>().HideAdmobBanner();
37 //}
```

RewardButton.cs **Uncomment lines: 23-25**

```
20 public void ShowRewardVideo()
21 {
22     //UNCOMMENT THE FOLLOWING LINES IF YOU ENABLED UNITY ADS AT UNITY SERVICES AND REOPENED THE PROJECT!
23     //if (FindObjectOfType<AdManager>().unityAds)
24     //    FindObjectOfType<AdManager>().ShowUnityRewardVideoAd();    //Shows Unity Reward Video ad
25     //else
26     //    FindObjectOfType<AdManager>().ShowAdmobRewardVideo();    //Shows Admob Reward Video ad
```

AdManager.cs **Uncomment lines: 253-259, 262-288**

```
251 /*
252 //Call this to show video ad
253 public void ShowUnityVideoAd()
254 {
255     Debug.Log("ShowUnityVideoAd");
256
257     if (Advertisement.IsReady("video"))
258         Advertisement.Show("video");
259 }
260
261 //Call this to show reward video ad
262 public void ShowUnityRewardVideoAd()
263 {
264     Debug.Log("ShowUnityRewardVideoAd");
265     if (Advertisement.IsReady("rewardedVideo"))
266     {
267         Debug.Log("Showing Advertisement");
268         var options = new ShowOptions { resultCallback = HandleShowResult };
269         Advertisement.Show("rewardedVideo", options);
270     }
271 }
272
273 private void HandleShowResult(ShowResult result)
274 {
275     switch (result)
276     {
277         case ShowResult.Finished:
278             Debug.Log("The Unity Reward ad was shown successfully");
279             FindObjectOfType<RewardButton>().AddReward();    //Adds reward to the player when rewardVideo is finished successfully
280             break;
281         case ShowResult.Skipped:
282             Debug.Log("Ad was skipped");
283             break;
284         case ShowResult.Failed:
285             Debug.LogError("The ad failed to be shown");
286             break;
287     }
288 }
289 */
```

ABOUT THE GAME

How to play

On mobile : Move your finger on the screen while holding it down to control the player.

On desktop/web : Move the mouse on the screen while holding down the left mouse button to control the player.

Goal

Your goal is to find all of the hiders or to hide from the seeker.

FOLDER STRUCTURE

- **Assets**

Contains all of the used assets for the game

- ***Animations***

Contains all of the used animations

- ***Audio***

Contains all of the used sound effects and background music

- ***Fonts***

Contains every used fonts

- ***Materials***

Contains every used material

- ***Meshes***

Contains every used mesh

- ***Particles***

Contains the used particle systems

- ***Prefabs***

Contains the used prefabs (for example obstacles)

- ***Scripts***

Contains C# scripts

- ***Sprites***

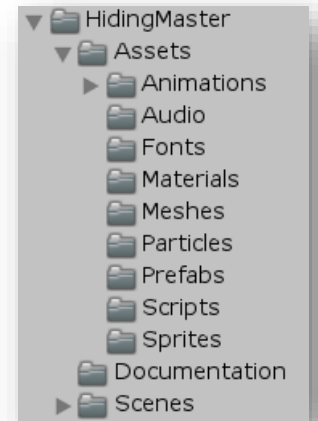
Contains all of the used 2D sprites (for example UI elements)

- **Documentation**

Contains documentation

- **Scenes**

Contains complete level(s) and lightmap/navmesh of the level(s)

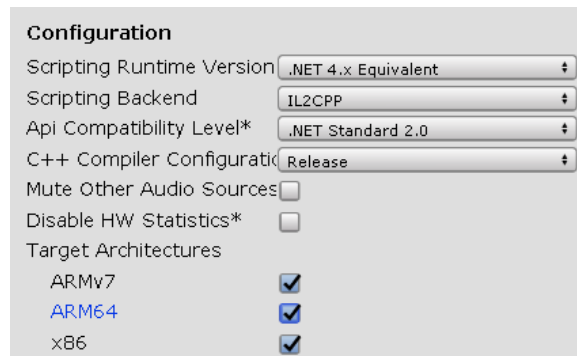


HOW TO BUILD FOR ANDROID 64 BIT?

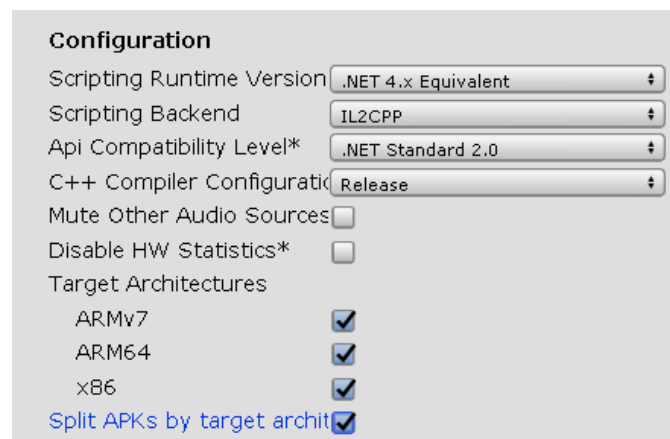
Credits: <https://stackoverflow.com/questions/55974331/unity-export-android-64-bit>

Unity supports x64 since 2017 LTS. To enable it you can go to File > Build Settings > Player Settings.

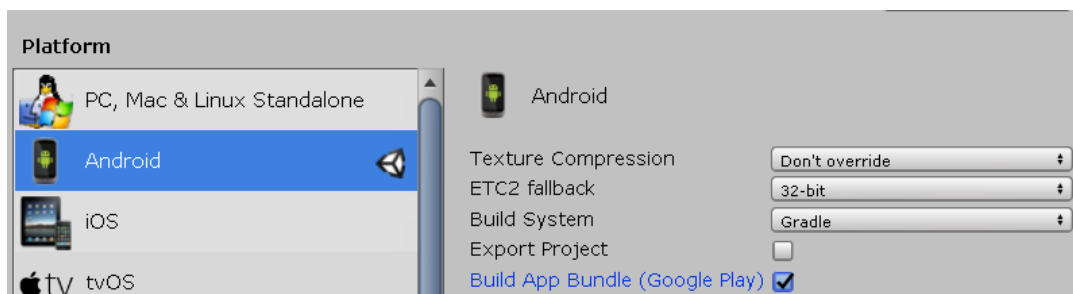
There under "Other Settings" you have to change your Script Backend to IL2CPP, and you will have the ARM64 checkbox active. Check it.



And then you can either select to generate Split APKs by Target Architecture



or in your Build Settings dialog choose to Build App Bundle (Google Play).

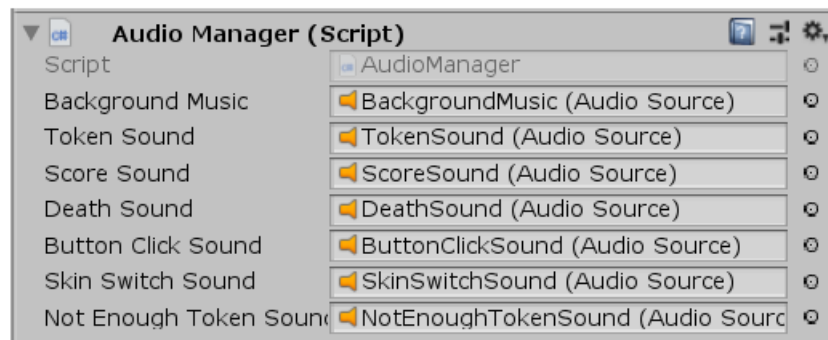


Hope it helps, you will need to have the Android NDK for this.

CUSTOMIZATION

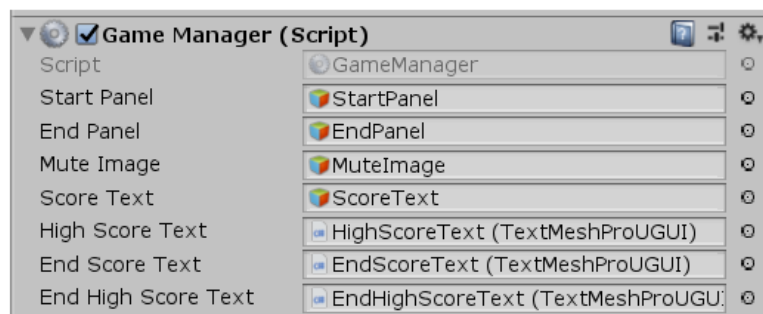
AudioManager:

You can change the music, sound effects here and under the AudioManager gameObject



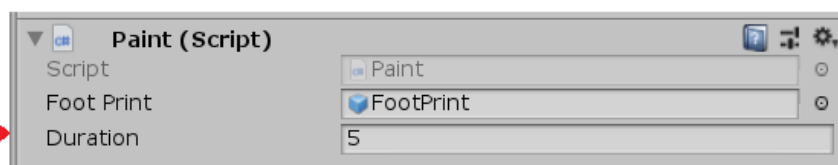
GameManager:

GameManager script contains the main functions of the game. It manages scenes (starts, restarts the level), enables/disables scripts and gameObjects, switches between panels



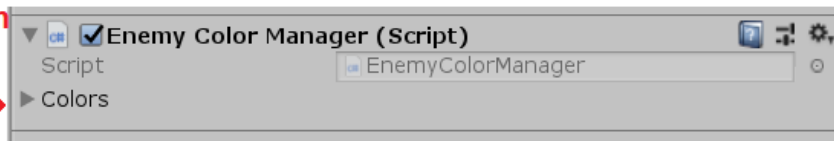
Paint:

This is how long the paint drops will be visible if the player touches the gameobject ➡



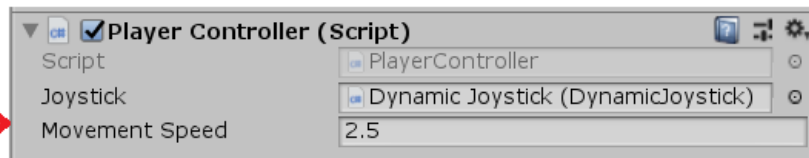
EnemyColorManager:

You can add new random colors for the enemies here



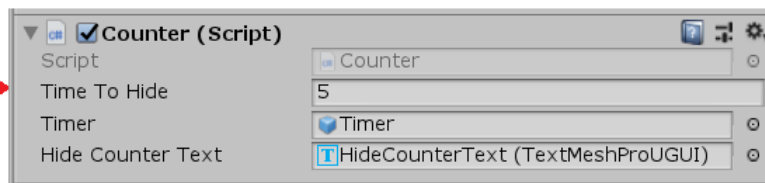
Player:

This is the movement speed of the player



Counter:

There is X seconds to hide



StaticVariables.cs

FOR TESTING, LEAVE EVERYTHING AS IT IS

If you want to use Admob and you are ready to release, then TestAds = false, unityAds = false

If you are ready to release, then replace the IDs with your own Admob IDs

```
public class StaticVariables : MonoBehaviour
{
    public static int plusRewardCount = 10; //In this case, watching a rewarded
    public static int[] requiredTokensToUnlock = { 0, 15, 30}; //In this case
    costs 100. Change the values between the { } marks

    //Leave the following Admob IDs if you want to test the game
    public static bool TestAds = true; //This has to be true when testing, and
    public static bool unityAds = false; //Set this false if you want to use
    public static string appID = ""; //Paste your Admob appID between the ""
    public static string bannerID = "ca-app-pub-3940256099942544/6300978111";
    public static string interstitialID = "ca-app-pub-3940256099942544/1033173712";
    the app
    public static string rewardVideoID = "ca-app-pub-3940256099942544/5224354917";
    app
}
```

If the gameobject has a prefab, then change the script values there, instead of in the Hierarchy -> gameobject itself.

Best regards, Ragendom