CRASHY CHASY

USER GUIDE

We strive to provide the best service as we can, if you have any questions or suggestions, please contact us!
Thank you!

SgLib Games

Contents

1 INTRODUCTION

Tap left or right timely to control the car to drift around and avoid other cars, which will chase you and try to smash your car down and the deadly obstacles, which will fall down from the sky. Collect coins to unlock new characters by trying to smash two enemies at a time. **Crashy Chasy** is a simple yet exciting and addictive game that will keep the player entertained for hours.

This game is ready for release out-of-the-box. Everything just works. It is also flexible and customizable. Some highlights:

- Addictive one-touch gameplay
- Daily reward system for better retention
- 30 built-in unlock-able characters with cute blocky style
- Free-to-use assets (fonts, sounds, music, model, etc.)
- Optimized for mobile

Most importantly, when equipped with the **Easy Mobile** plugin, this template can form a truly full-featured game that is ready for release. Easy Mobile is a comprehensive, cross-platform package that provides most of desired features of mobile games:

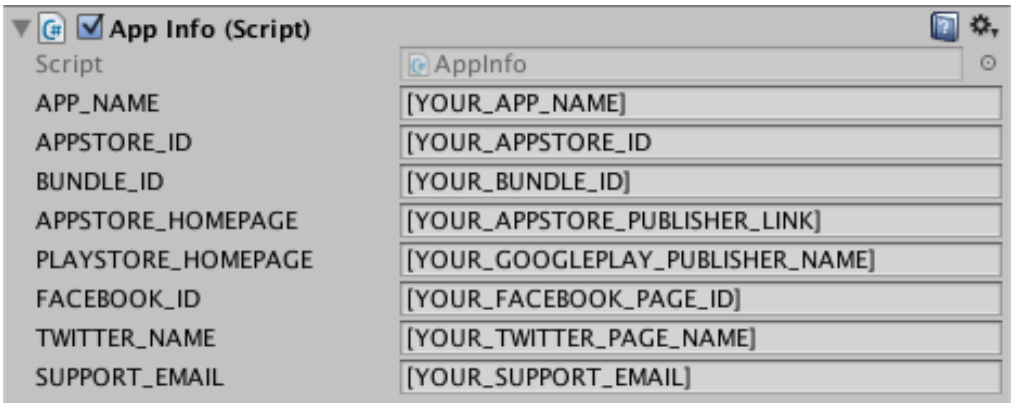
- Support for AdColony, AdMob, AppLovin, Chartboost, Facebook Audience Network, Heyzap, ironSource, MoPub, Tapjoy and UnityAds
- In-app purchasing
- Support for Game Center (iOS) and Google Play Games Services (Android)

for leaderboards and achievements
- Recording gameplay and exporting GIF images
- Sharing to social networks (PNG or GIF images)
- Push notification using OneSignal service
- Native rating request popup (rate my app)

2 GETTING STARTED

2.1 Enter app information

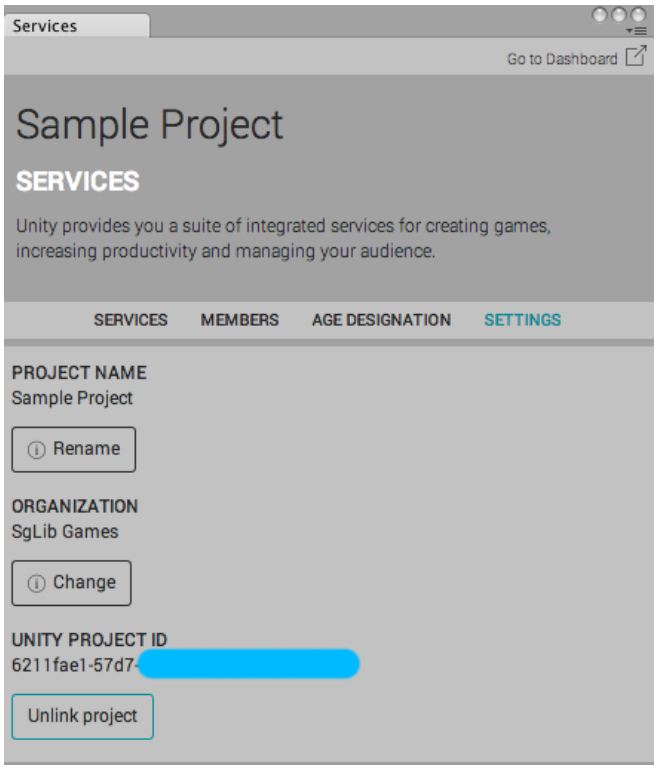
The project contains a game object called AppInfo where you can fill in important app-related metadata like AppStore Id and Bundle Id. These values will be used for features like Rate Us button and opening Facebook or Twitter page.



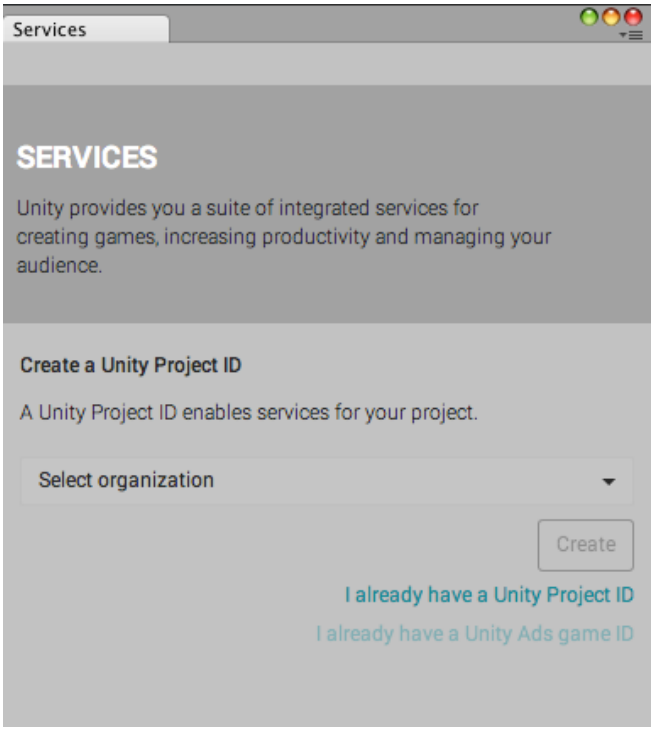
2.2 Link the game to your Unity project

When developing this template, we normally need to link it to our own Unity project for testing, therefore you may need to unlink it from our project and link it to your own one, if you're going to use Unity services (e.g. if you want to enable premium features of this template, you'll need to use Unity IAP service). To unlink the project:

- Select Window -> Unity Services
- Select SETTINGS tab
- Click Unlink Project button



Now you can create a new project for the game.



Now you game is linked to your own Unity project and is ready to use Unity.

services.

2.3 Change project settings

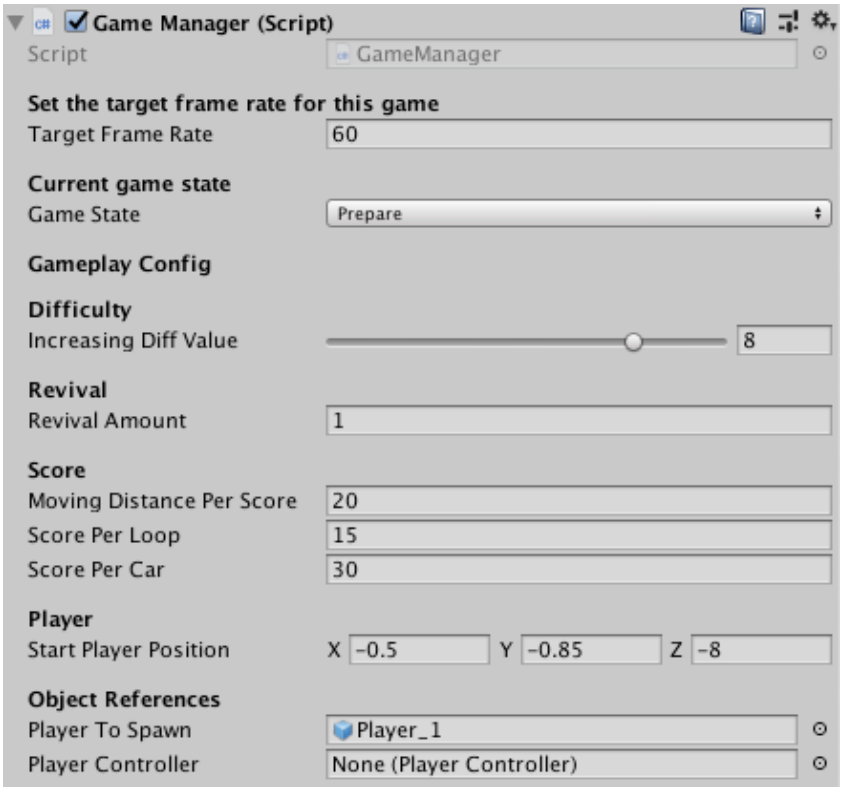
We also implemented a gravity for the player when the player blow up. So you need to look on your tool bar in unity and navigate to Edit/Project Settings/Physic and change The gravity in Y direction to -70.

3 TEMPLATE CUSTOMIZATION

3.1. Gameplay tweaking

3.1.1 GameManager

Most of important gameplay parameters can be configured within the GameManager component which is attached to a game object also named GameManager in the hierarchy.



You can tweak the gameplay by modifying following variables:

- *TargetFrameRate*: the target frame rate for the game, which should be at least 60fps for games requiring smooth, fast motion.
- *IncreasingDiffValue*: the increasing difficulty value of game. Increase this

value, if you want to game difficulty increases faster.

- *RevivalAmount*: amount of revival per game.
- *MovingDistancePerScore*: the moved distance of player to get a score.
- *ScorePerLoop*: the earned score for completing a drifting round.
- *ScorePerCar*: the earned score for killing a chasing car.
- *StartPlayerPosition*: The first position of the player when the game is play, the default value is center the screen and place on the ground

3.1.2 CarController:

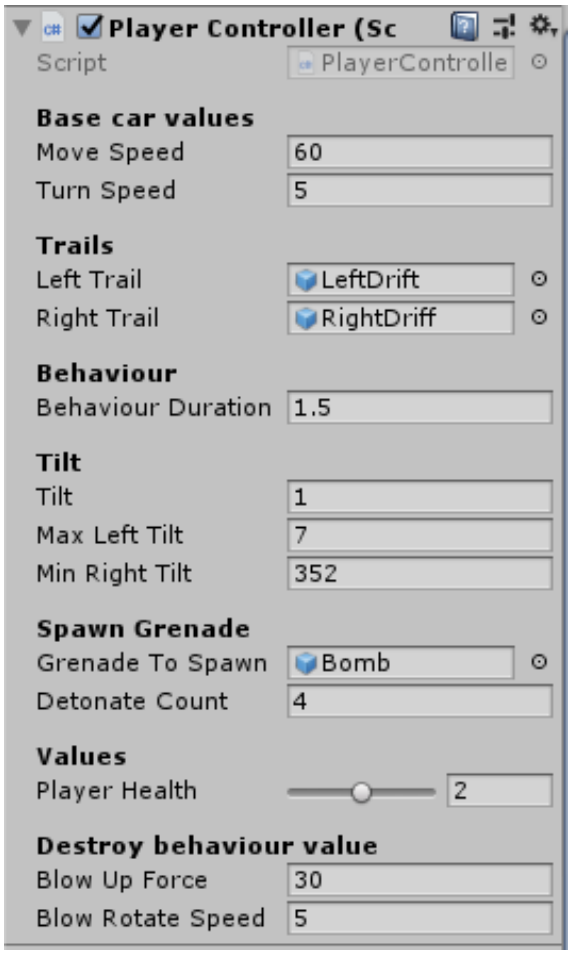
The base car scripts, control the car behavior.

➢ All cars have the same base values:

- *MoveSpeed*: the normal speed of car.
- *TurnSpeed*: the rotating speed of car.
- *LeftTrail*: the left car trail when the car is moving.
- *RightTrail*: the right car trail when the car is moving.
- Both left and right trail will be placed at the bottom of car wheels to create trail effect.
- *BehaviourDuration*: the survival time of car after dying.

3.1.3 PlayerController

The *Player prefab is an important object*, which holds the *PlayerController* component. It's placed at "Asset/_CrashyChasy/Prefabs/Game/Player".



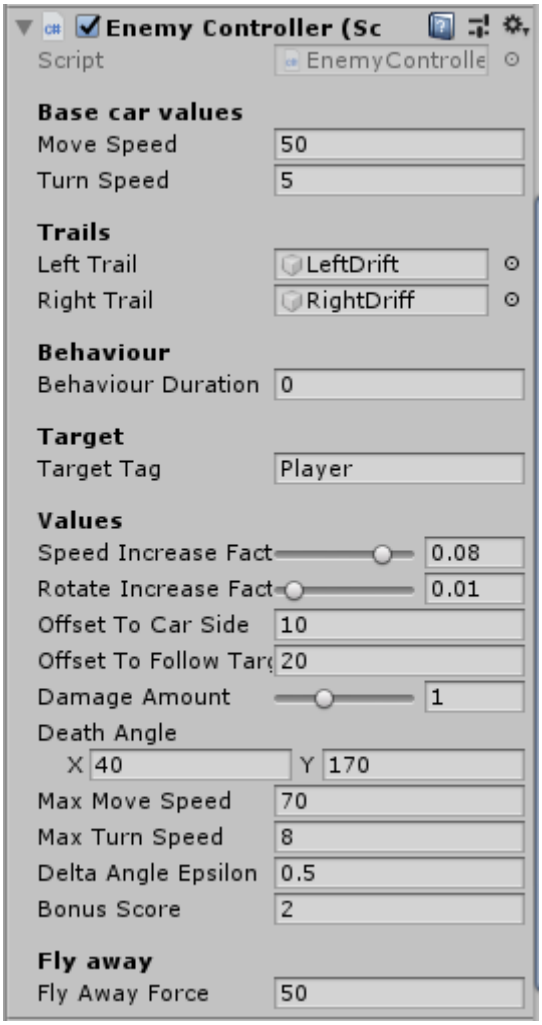
The component is inherited from the CarController. You can tweak the player behavior by modifying following variables:

- *Tilt*: The tilting speed of car
- *MaxLeftTilt*: The maximum angle when the car tilt to the left, the min left tilt is 0.
- *MinRightTilt*: the minimum angle when the car tilt to the right, the max left tilt is 360.
- *De*tonateCount: A bomb will be spawned in the center of rotating point,after player finish certain number of drifting round.
- *PlayerHealth*: The max health of player at start.
- *BlowUpForce*: How strong the car will be blown up.
- *BlowRotateSpeed*: the rotating speed of car after blowing up in air.

3.1.4 EnemyController

The EnemyController component is attached to the Enemy prefabs. They place at "Asset/_CrashyChasy/Prefabs/Game/Enemy".

The component is inherited from the CarController script.



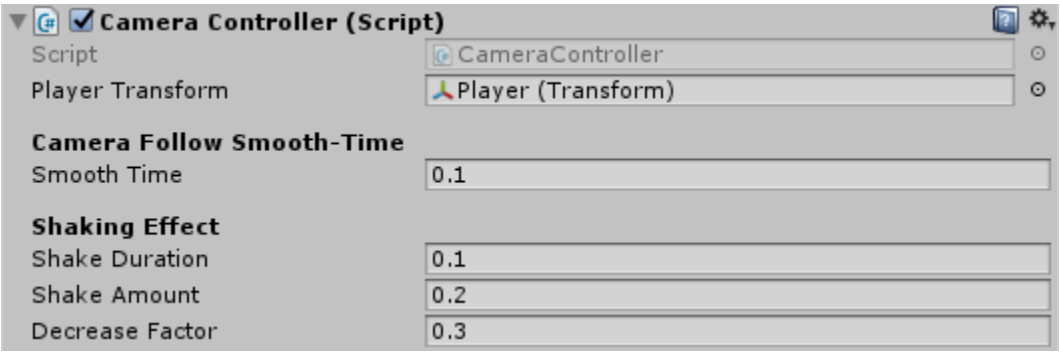
➢ To control behavior of enemy car, you need modify following variables:

- *TargetTag:* The prefab's target this enemy will chasing, the default tag is Player.
- *SpeedIncreaseFactor:* the increment value of moving speed over time.
- *RotateIncreaseFactor:* the increment value of rotating speed over time.
- *OffsetToCarSide:* the relative distance between the side of target and target.
- *OffsetToFollowTarget:* the enemy will chase the left or right side of target when distance between enemy and target is far, after this distance is equal or smaller than this value, enemy will chase directly to target.

- *DamageAmount:* given damage of enemy after smashing player.
- *DeathAngle:* When enemy and player collide with each other, we calculate the forward vector of two cars and get the angle between them. If this angle is in range of DeathAngle the enemy will die and conversely, the player will be damaged.
- *MaxMoveSpeed:* the maximum of moving speed of enemy.
- *MaxRotateSpeed:* the maximum of rotating speed of enemy.
- *DeltaAngleEpsilon:* The delta angle epsilon to compare with two velocity between two frames.
- *FlyawayForce:* the force impact on enemy after dying.

3.1.5 CameraController

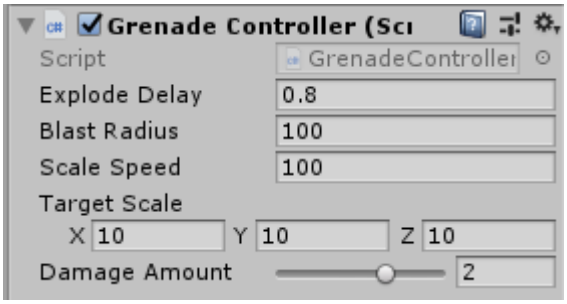
The CameraController component is attached to the Main Camera object in the hierarchy.



- *SmoothTime:* the smooth time of camera.
- *ShakeDuration:* how long the camera shaking.
- *ShakeAmount:* amplitude of the shake, a larger value shakes the camera harder.
- *DecreaseFactor:* the decrease value of shaking.

3.1.6 GrenadeController:

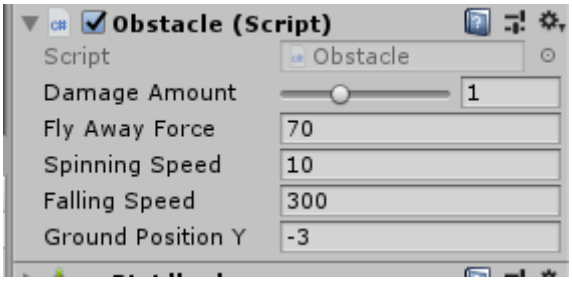
The GrenadeController component is attached to the Bomb prefabs.



- *ExplodeDelay:* the force impact on enemy after dying.
- *BlastRadius:* the radius of blast when exploding.
- *ScaleSpeed:* the scale speed of bomb.
- *TargetScale:* The bomb will be scaled from the original scale from the target scale
- *DamageAmount:* given damage of bomb to player.

3.1.7 Obstacle

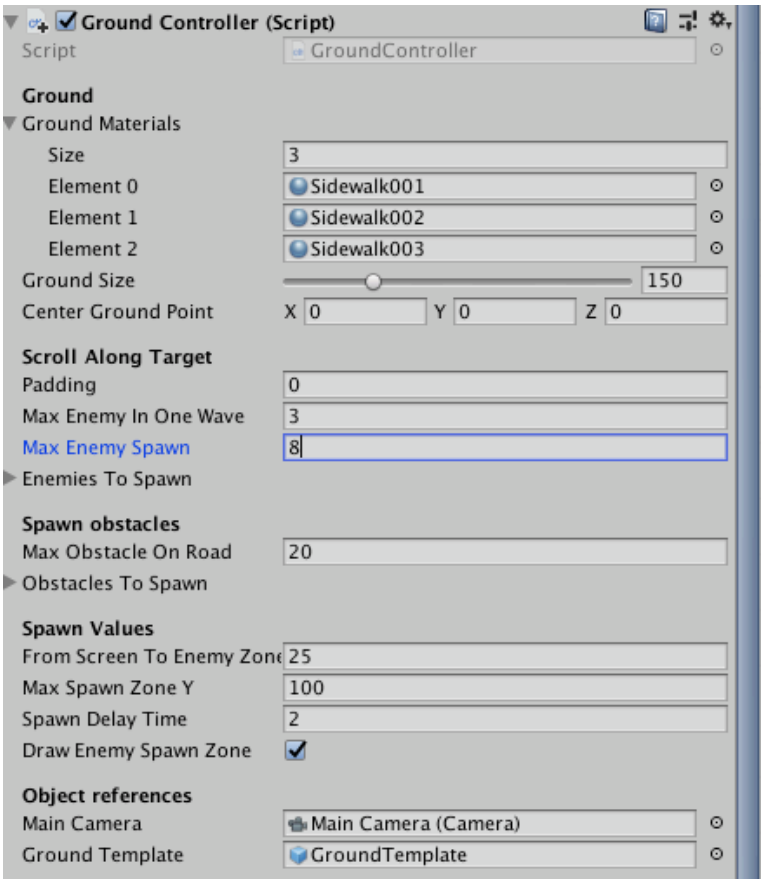
The Obstacle component is attached to the Obstacle prefabs



- *DamageAmount:* given damage of obstacle to player.
  - *FlyawayForce:* the force impact on enemy after collision.
  - *SpinningSpeed:* the spinning speed when the obstacle flies away in air.
  - *FallingSpeed:* falling speed of obstacle.
  - *GrounPostionY:* When obstacle reach this y position, it will stop falling.

3.1.8 GroundController:

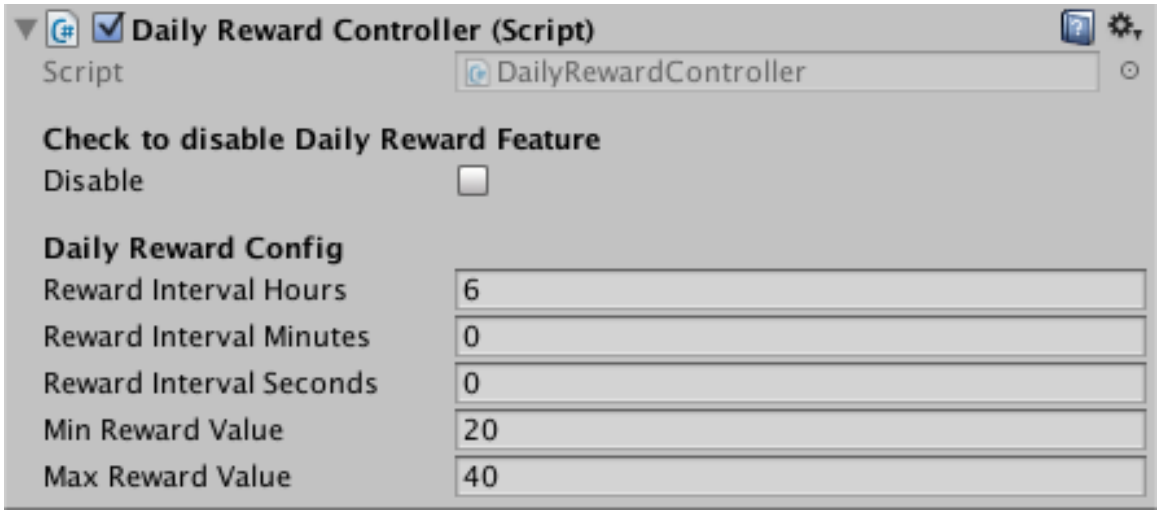
It's the important component of game. Attached to the GroundController gameobject in the hierarchy. It controls spawning enemies and obstacle.



- *GroundMaterials:* the list of ground materials to choose randomly.
- *ScrollAlongTarget:* The offset between the border of ground and the position which you want to update the position of ground when the car player reach.
- *EnemiesToSpawn*: The list of enemy prefabs to spawn randomly.
- *MaxEnemySpawnInOneWave:* The number of enemies can be spawned in per wave
- *MaxEnemySpawn:* the maximum number of enemies in same time.
- *MaxObstacleOnRoad:* the maximum number of existing obstacles in same time.
- *ObstaclesToSpawn:* The list of obstacle prefabs to spawn randomly.
- *FromScreenToSpawnEnemyZone*: The offset between the screen and the spawn enemy zone
- *FromScreenToSpawnObstacleZone:* The offset between the screen the spawn obstacle zone
- *SpawnDelayTime*: the delay time in each spawning turn.
- *DrawEnemySpawn*: enable to draw spawn zone.

3.2. Daily reward feature

This template has a built-in daily reward system in which the user will be rewarded with coins every predefined interval of time. This is an effective way to increase user engagement and retention for your game. You can configure this feature from the *DailyRewardController* object in the hierarchy.

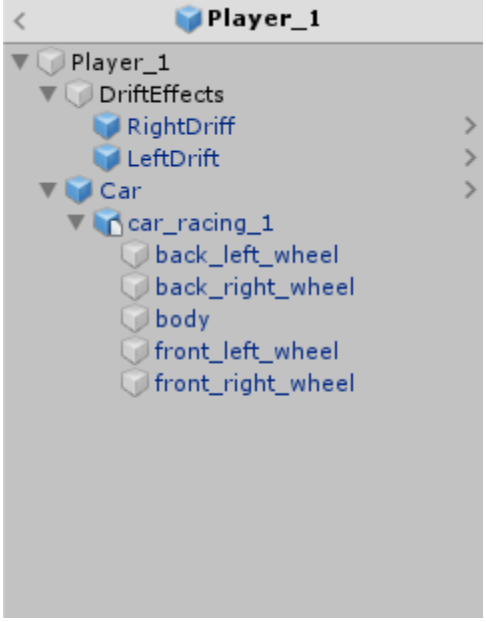


- *Disable*: check to disable this feature
- *Reward Interval Hours, Minutes and Seconds*: the amount of time until the next reward
- *Min Reward Value* & *Max Reward Value*: the actual rewarded coins will be randomized between these two values

3.2 Adding more characters

Out-of-the-box, this game is already packed with 30 characters, cute and ready to use! If you want to add more, follow these simple steps:

a. Create a character model or import your model from other tools with the pivot at the center at the path Assets/_CrashyChasy/Models/Cars. Make sure your model has **4 wheels** and each model name must contain **wheel.** Ex: back_left_wheel**.**



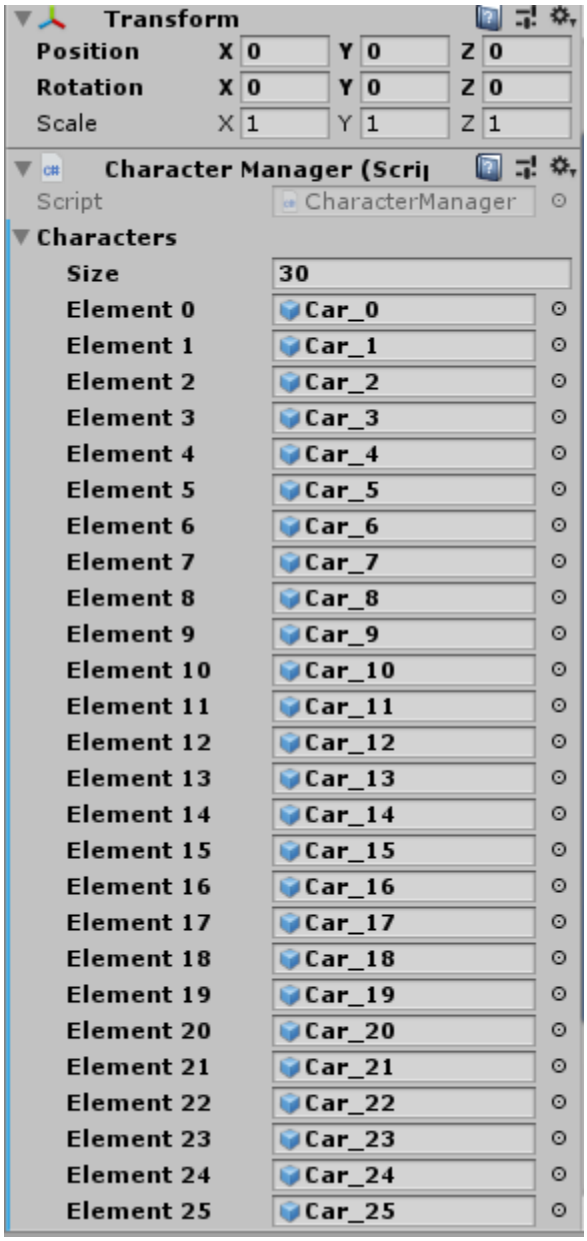
b. Navigate to *Assets/*_CrashyChasy */Prefabs/Game/Characters/SelectCharacters* and duplicate one of the available character prefabs.

c. Change the name of the prefab to a preferred one. Recommend <Car_yourmodelname>.

d. Drag your export model to make it become child of your <Car_yourmodelname>. Then save your <Car_yourmodelname> prefabs.

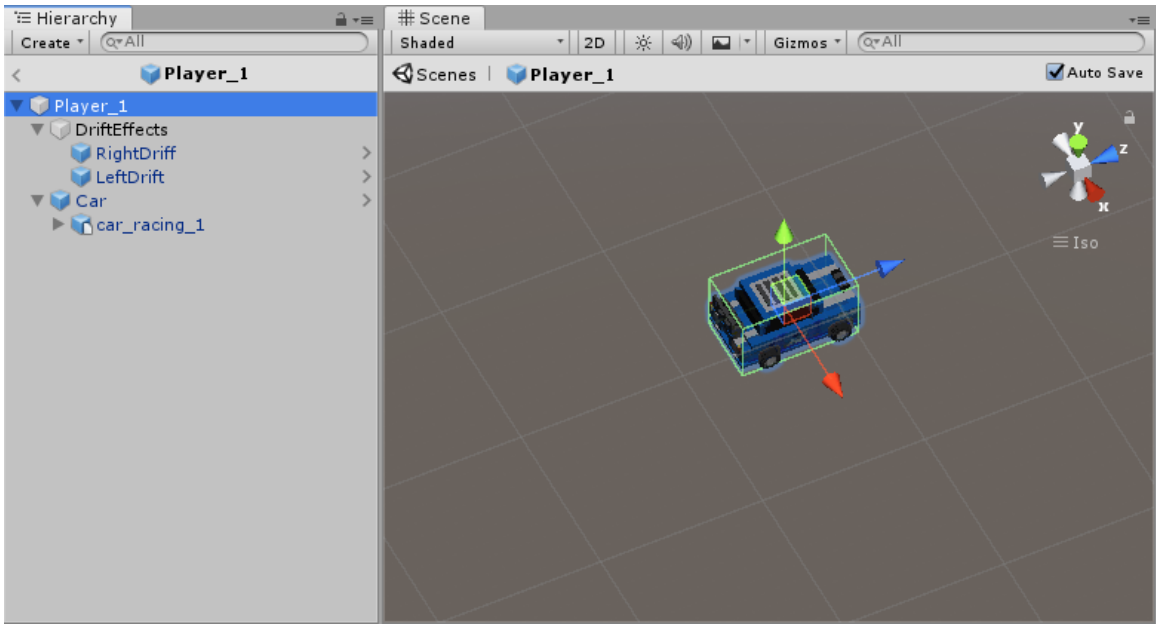
e. Enter the character name and price to the *Character* component. Check the *isFree* option if you want to give out this character for free (it will be automatically unlocked). ***Important:*** *the new character's name must not repeat any existing character name.*

f. Resize the character array in *CharacterManager* game object then drag the new character to it and hit Apply to save changes to its prefab.



* IMPORTANT: **COLLIDER, TRAIL AND MODEL MAY NOT HAVE A SAME POSITION**

This is the player prefabs.



A game object named **Car** is your character. If you want to know whether your character, your collider and your drifting effects are the same position, delete the default Car gameObject and drag your Character to the Player_1 prefab.

If they don't have the same position or the same scale for the collider and car, follow this step:

- Scale your **Car** gameObject to it fits the collider size.
- Find your child game object of **Car.** For example, image is **car_racing_1.** This is your imported or created model when you create a new character.
- Adjust your **car_racing_1** position to it stay in the collider.

NOTE:

- Don't try to scale your collider or change the offset of the collider, or change the DriftEffects position. This is the default transform of all your characters.

3.3 Adding more enemies

There are 4 available enemies in this game. If you want to add more, follow those steps:

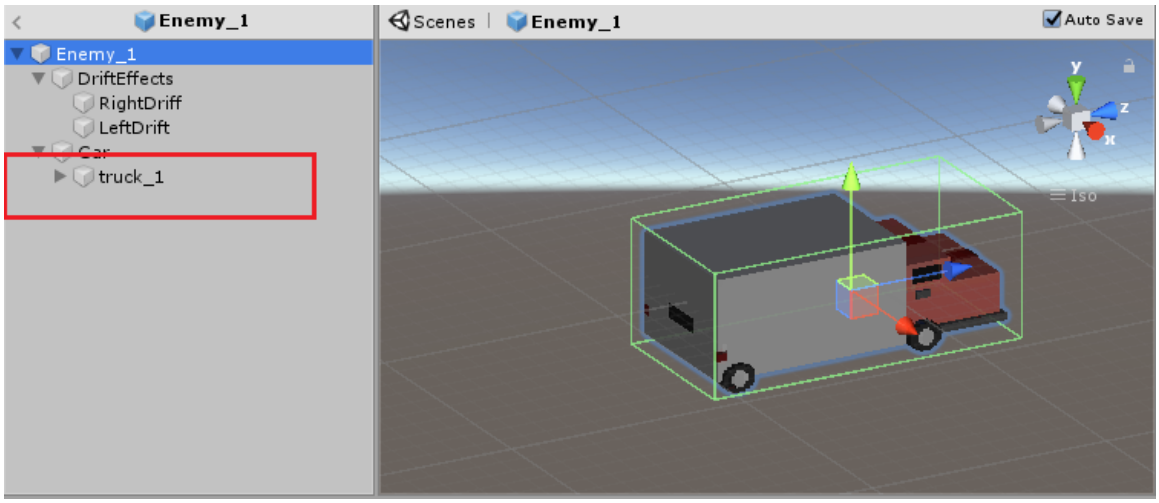
a) Create a character model or import your model from other tools with the pivot at the center at the path Assets/_DrityKing/Models/Cars.Make sure your model has **four 4 wheels** and each model name must contain **wheel.** Ex:

back_left_wheel

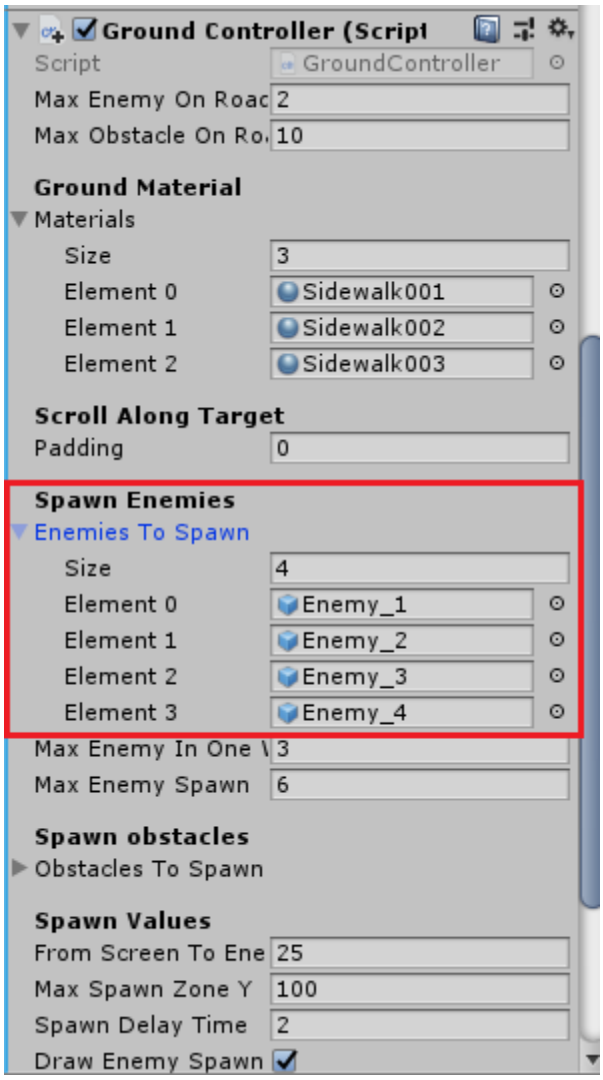
b) Navigate to the Assets/_DrityKing/Prefabs/Game/Characters/Enemy. And duplicate an enemy prefab.

c) Change the name of the prefab to a preferred one. Recommend <Enemy_yournemyname>

d) On your new enemy prefabs, find the child of Car gameobject, replace with your model and save the prefabs.



e) On the hierarchy, find the **Ground** GameObject and drag your new enemy prefabs to the Enemies To Spawn field.



* IMPORTANT: **COLLIDER, TRAIL AND MODEL MAY NOT HAVE A SAME POSITION**

If your model, collider and trail effect don't have the same position, follow this step

- Click on the Enemy prefabs and change the offset of your collider until it fit the model
- Change the size of the collider to it fits the model.
- Adjust the trail effects position. If you want to see the trail. Find the **RightDriff** and **LeftDriff** child and set the emitting to true in the Trail Renderer component. Don't forget to turn it off when finish.

NOTE:

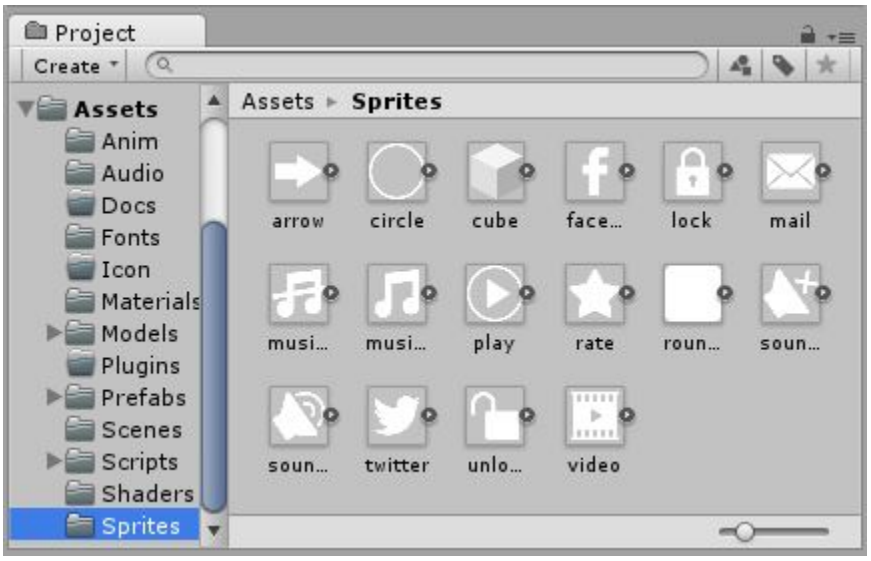
- This is different from the player solution. So you should not change the

transform of the **Car** gameObject or the model.

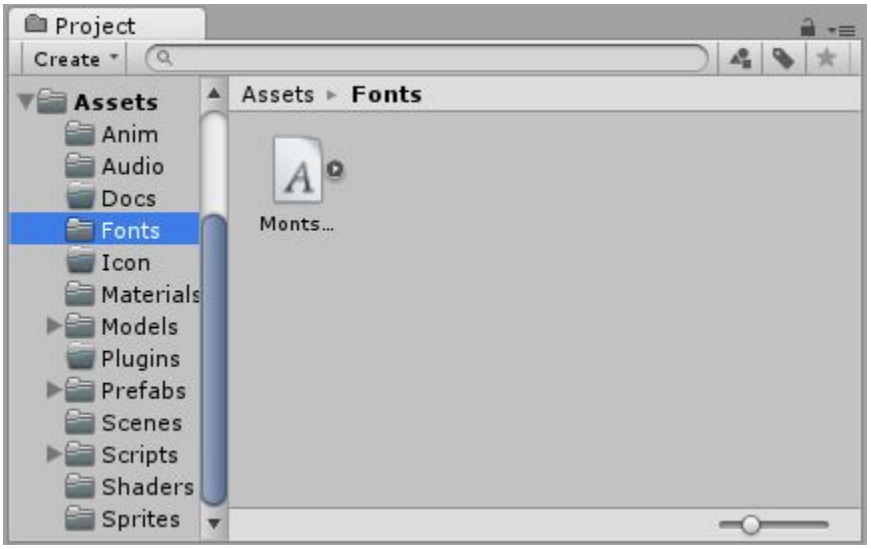
- You can also add your own obstacles, bomb with the similar steps as adding enemies.

3.4 Customizing UI

All sprites used in this game (for buttons and other UI components) are located under the *Sprites* folder. You can replace them with your own sprites to modify the UI as you like.

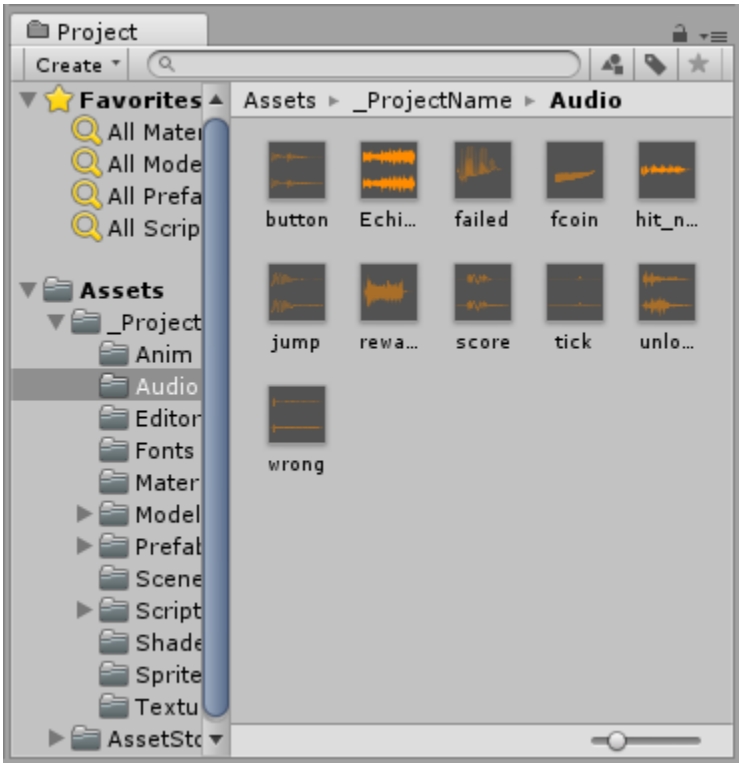


All fonts used in this game are free-to-use in commercial projects. Fonts are located under the *Fonts* folder together with appropriate license files.

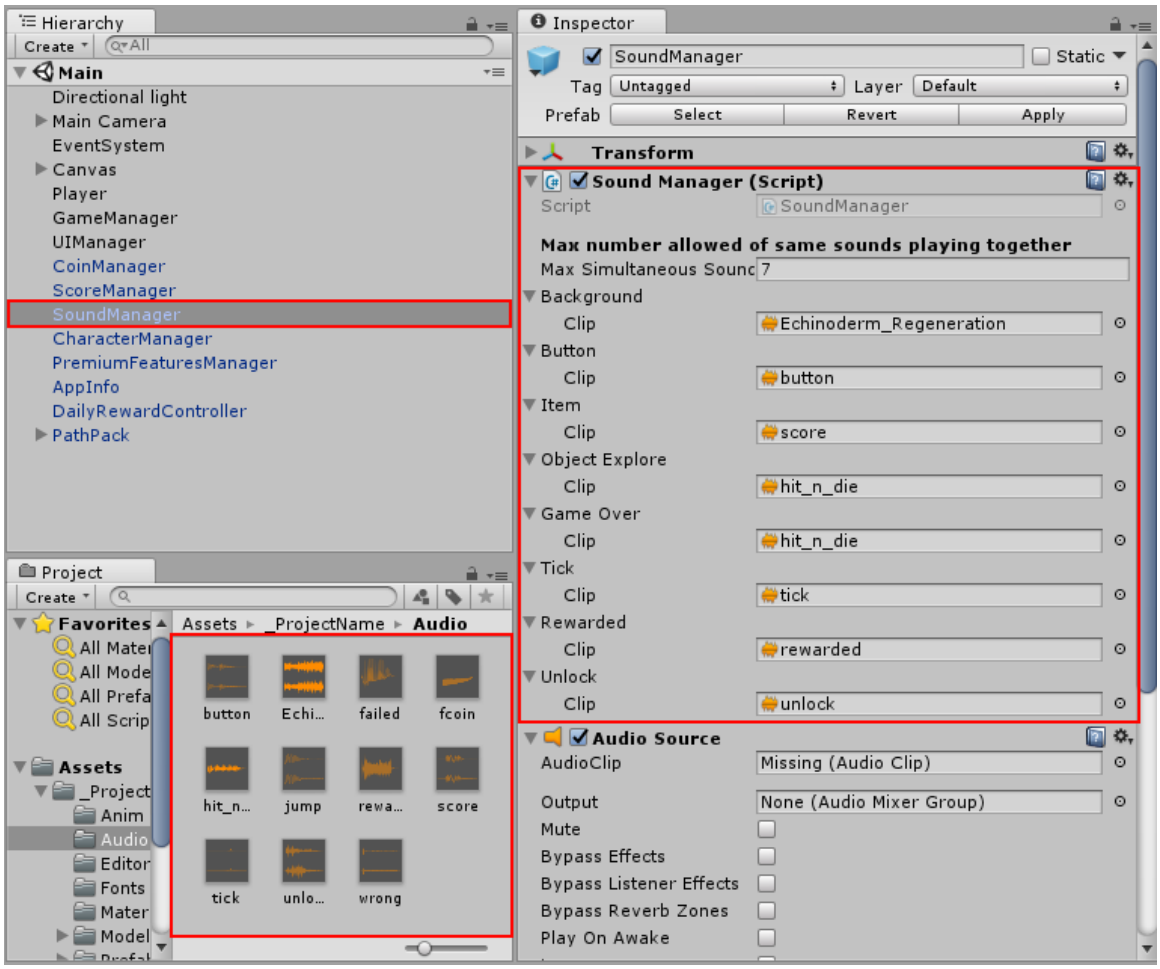


3.5 Sounds

All sounds included in this game are free-to-use in commercial projects and are located under the *Audio* folder.



This game features a *SoundManager* class to manage activities in game like playing music or mute/unmute sounds. If you want to replace sounds in this game, simply drag and drop new sounds to appropriate slots in the *SoundManager* component.

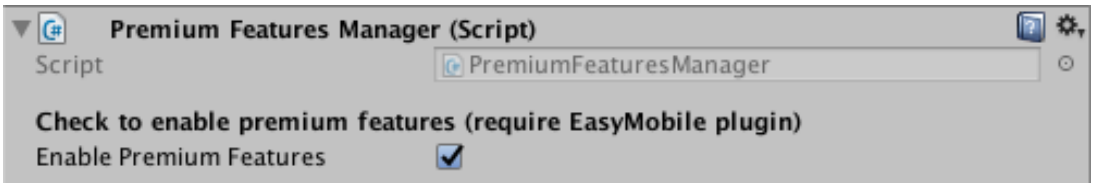


4 ENABLING PREMIUM FEATURES

This section provides a guide on configuring premium features for your game. As these features are implemented using Easy Mobile plugin, this guide is intended to be used alongside Easy Mobile's official user guide. Therefore, it is strongly recommended that you also read through that guide to familiarize yourself with the plugin. You can access the online user guide from menu *Window > Easy Mobile > Online Documentation* (after Easy Mobile is imported to the project).

4.1 Before You Begin

- In the Main scene's hierarchy, there's an object named *PremiumFeaturesManager* which contains all the relevant components from which you can configure how premium features behave in your game.
- Make sure the *EnablePremiumFeatures* option in the *PremiumFeaturesController* object is checked.

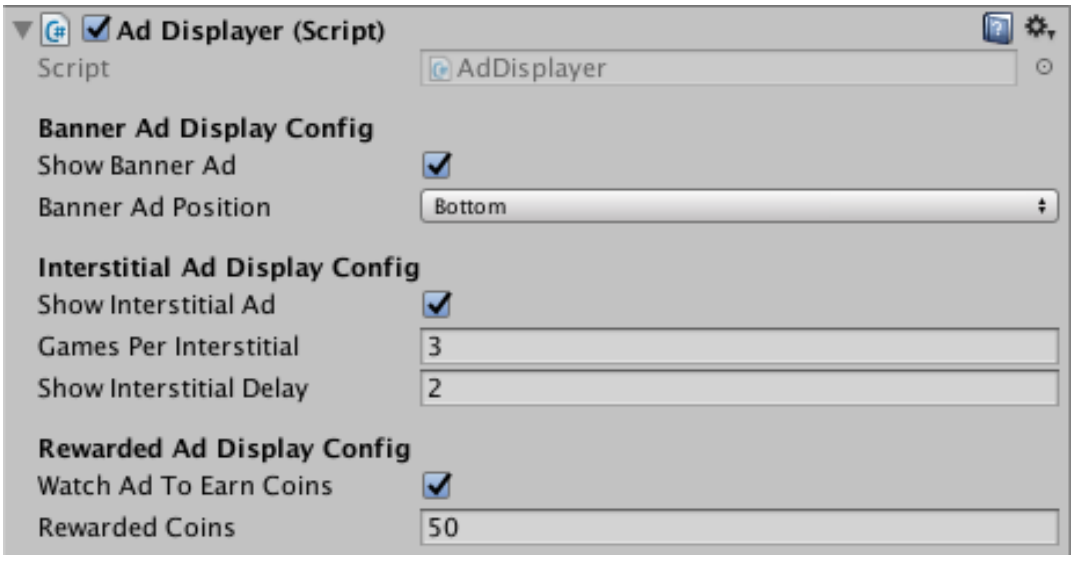


- The settings interface of Easy Mobile can be opened via menu *Window > Easy Mobile > Settings*, this is the only place to go to configure this plugin.
- Note that you won't need to write a single line of integration code for Easy Mobile to work, as the integration was done beforehand, you only need to configure the plugin in the editor (that means you can ignore all the Scripting sections in Easy Mobile user guide if you wish to).

4.2 Advertising

4.2.1 Template-specific setup

The PremiumFeatureManager object contains a component named *AdDisplayer* which is responsible for all ads displaying activities in the game. There you can configure how ads should be served in your game.



Banner ads are configured in the **Banner Ad Display Config** section.

- *Show Banner Ad*: whether to show a banner ad in game
- *Banner Ad Position*: which position the banner should be placed

Interstitial ads are configured in the **Interstitial Ad Display Config** section.

- *Show interstitial ad*: whether to show interstitial ads when game over
- *Games Per Interstitial*: how many games to be played before showing ad

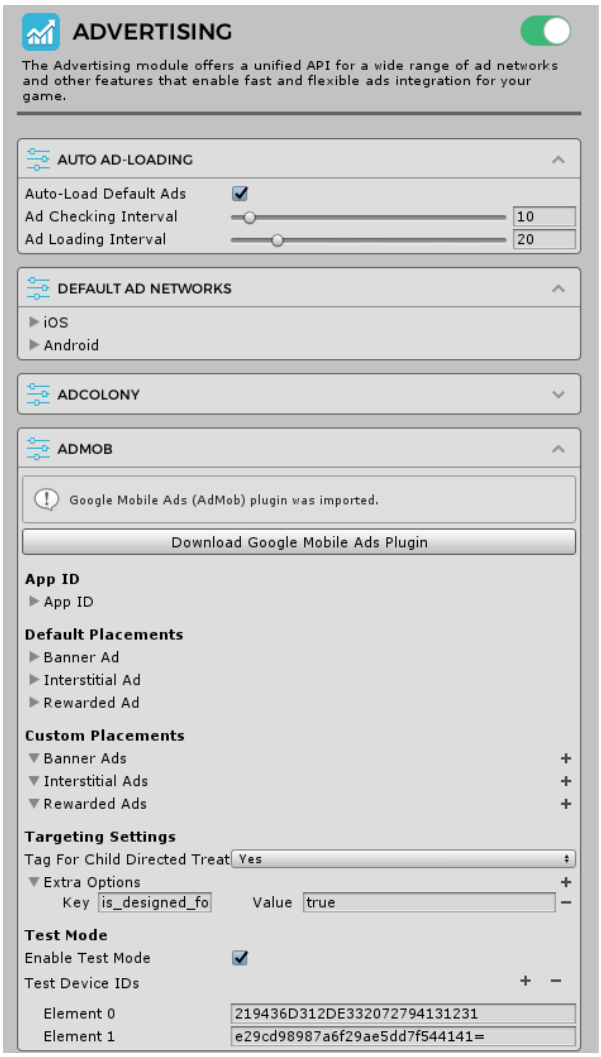
- *Show Interstitial Delay*: how many seconds after game over that ad is shown

Rewarded ads are configured in the **Rewarded Ad Display Config** section.

- *Watch Ad To Earn Coins*: whether to allow the user to watch an ad to earn extra coins
- *Rewarded Coins*: how many coins should be awarded after watching an ad

4.2.2 Easy Mobile setup

Open Easy Mobile's settings interface to start configuring its Advertising module (see its user guide for more information). With Easy Mobile you'll have support for AdColony, AdMob, Chartboost, Heyzap (with mediation) and Unity Ads. You can use multiple ad networks at once and have different configurations for iOS and Android. Below is the settings interface of the Advertising module.



You can setup the module in just a few steps as below. Please see the Advertising section in Easy Mobile's user guide for detailed instructions on each step.

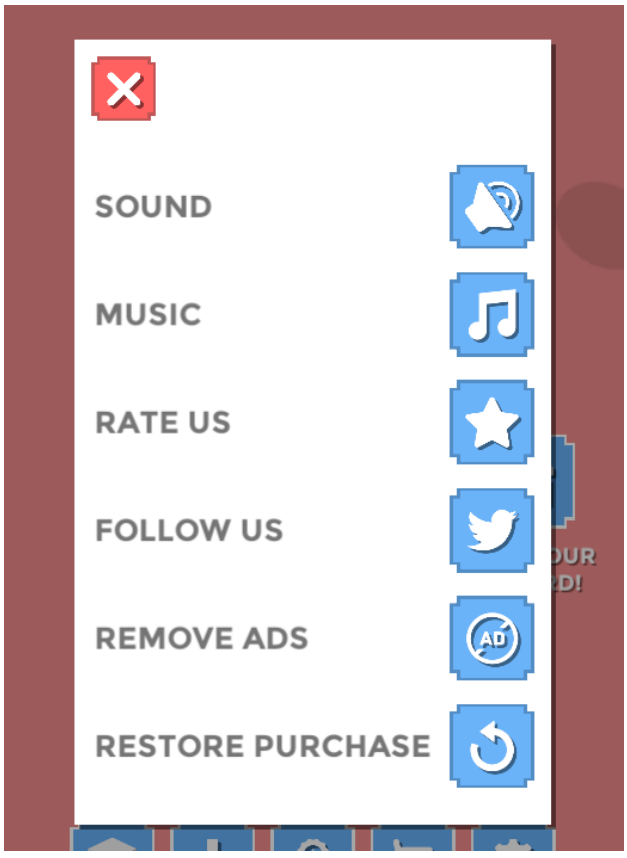
a. Setup the ad networks you want to use, including importing the required plugins, please see Easy Mobile user guide for more information
b. Enable auto ad-loading feature: simply leave the *Auto-Load Default Ads* option as checked and other parameters as default, the plugin will automatically load ads in the background
c. Select default ad networks for each platform: choose your preferred network for each type of ad on each platform

That's it! Now your game is ready for showing ads!

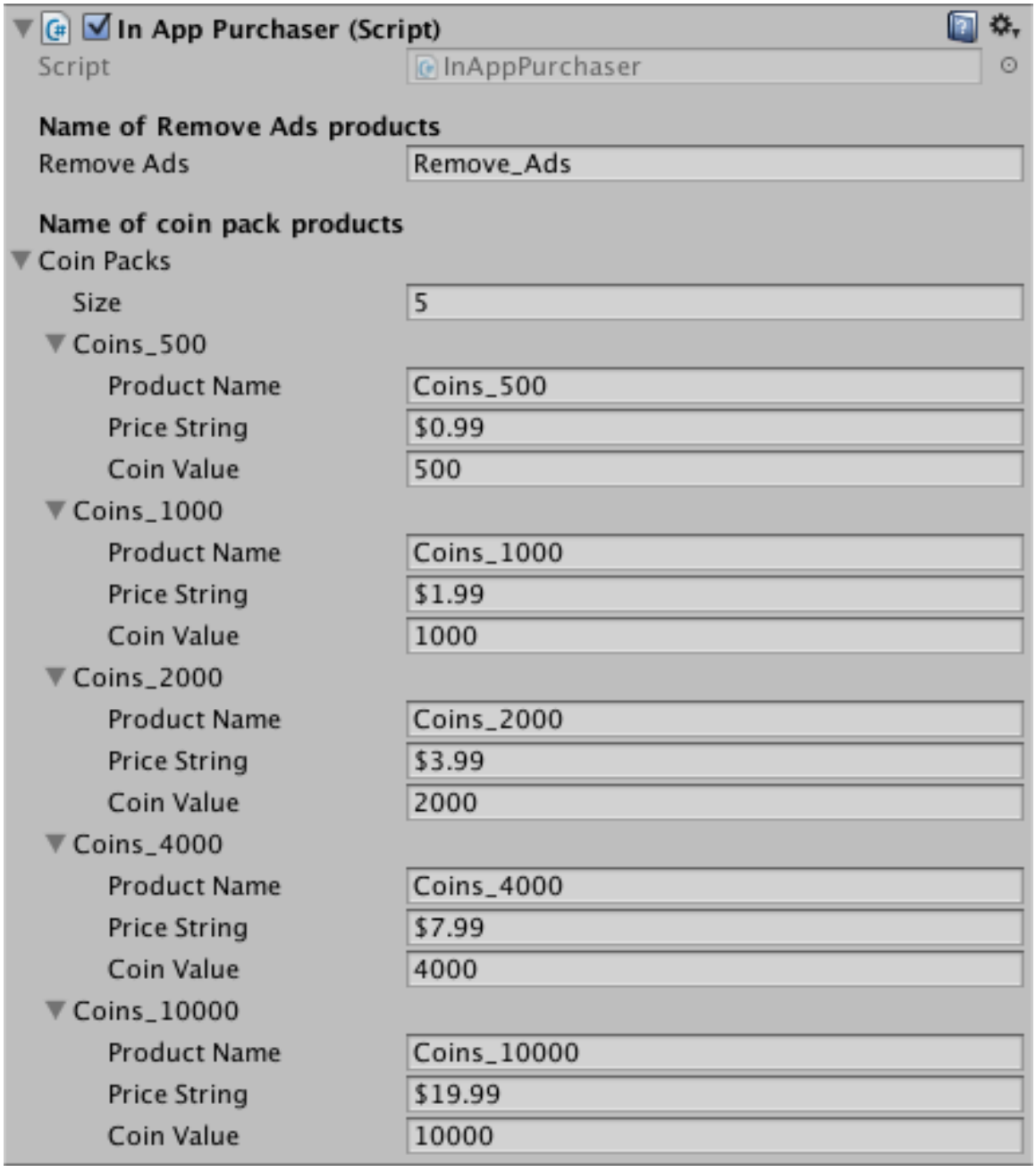
4.3 In-App Purchasing

4.3.1 Template-specific setup

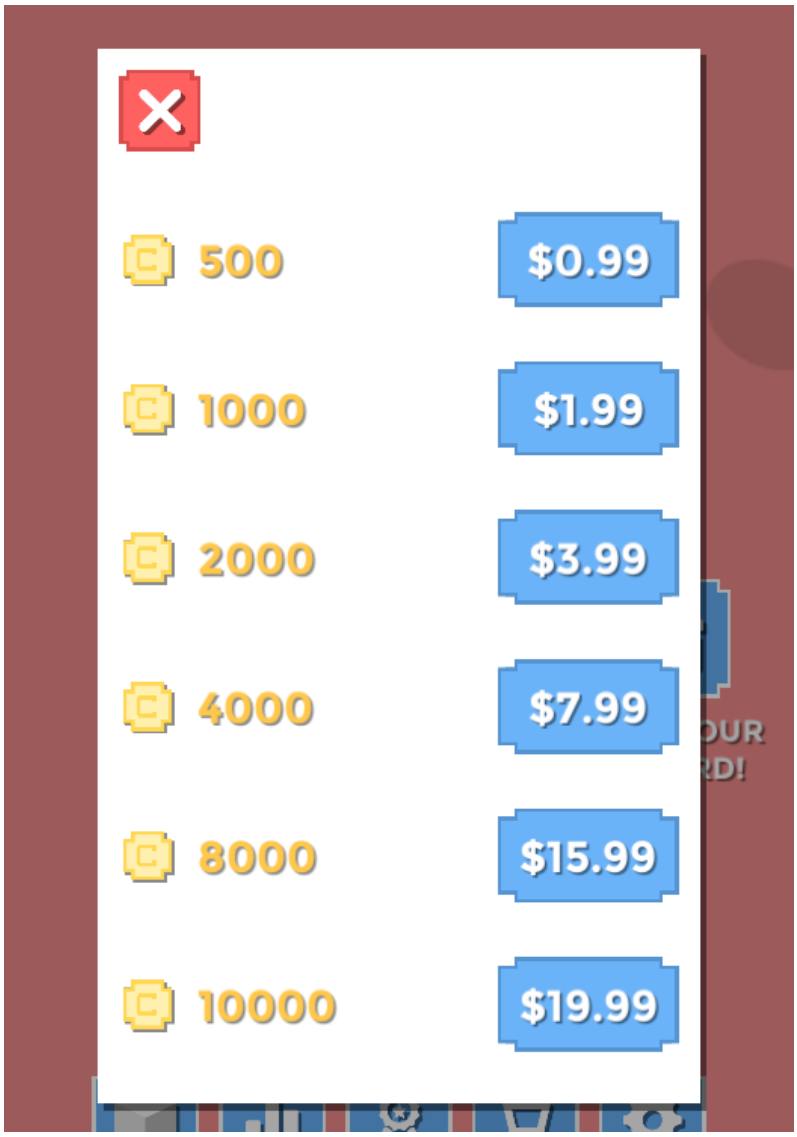
The built-in in-app purchases of this template include a *Remove Ads* button, and several coin packs. You can modify existing products and add more coin packs if you like. There's also one *Restore Purchase* button as required on iOS.



The PremiumFeaturesManager object contains a component named *InAppPurchaser* which manages all the in-app purchasing activities in this game.



Here you can modify the product definitions including the displayed name, price or coin value of the coin packs. To add more coin packs, simply increase the *CoinPacks* array size and enter necessary information for your new packs. The built-in store UI will automatically update to your changes in the product list without you having to do anything.



4.3.2 Easy Mobile setup

Setting up the In-App Purchasing module of Easy Mobile includes the following steps. Please see the In-App Purchasing section in Easy Mobile's user guide for detailed instructions on each step.

a. Enable Unity In-App Purchasing service
b. Select target store if you're on Android
c. Enable receipt validation if you wish
d. Declare the products

Below is the settings interface of the In-App Purchasing module of Easy Mobile.

Note that the products declared with Easy Mobile must have names that match with the ones you have in the aforementioned *InAppPurchaser* component. Also note that *Remove Ads* is a non-consumable product, while the coin packs must be consumable.

4.3.3 Create the products for targeted stores

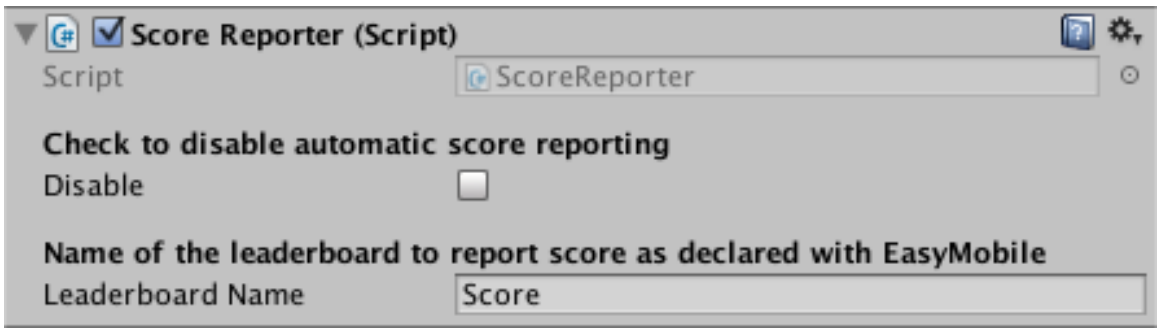
That last step in configuring the in-app purchasing feature is to create products for your targeted stores (e.g. Google Play and Apple App Store). Make sure the product ID, product type and price match the ones you have in your game.

4.4 Game Service

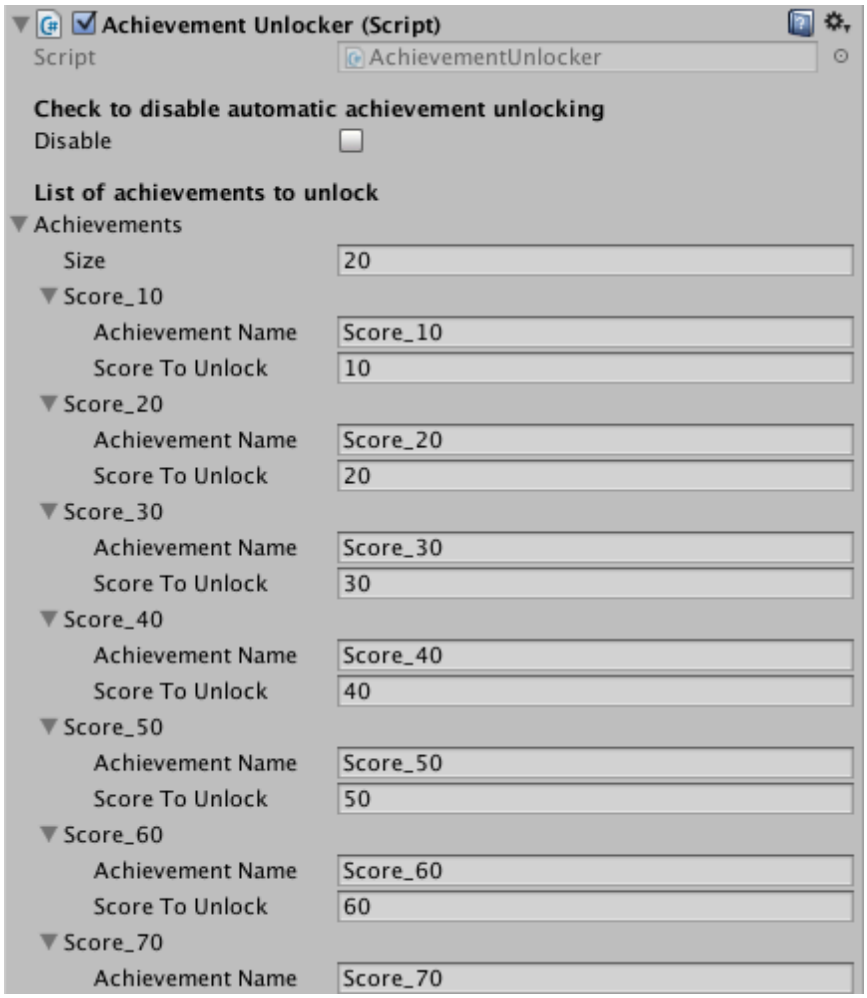
4.4.1 Template-specific setup

This template has a built-in leaderboard for ranking users' scores, and many achievements. It works with Game Center (iOS) and Google Play Game Services (Android).

User's score will be submitted automatically when game over by a component named *ScoreReporter*, which is also attached to PremiumFeaturesManager object. There you can change the leaderboard name or even disable automatic score reporting altogether.



Achievements will be unlocked automatically when the user reaches a certain score. The achievement unlocking is handled by the component named *AutoAchievementUnlocker*. In this component, you can modify existing achievements and add or remove achievements. You can also disable the automatic achievement unlocking feature if you wish.



4.4.2 Setup for your targeted stores

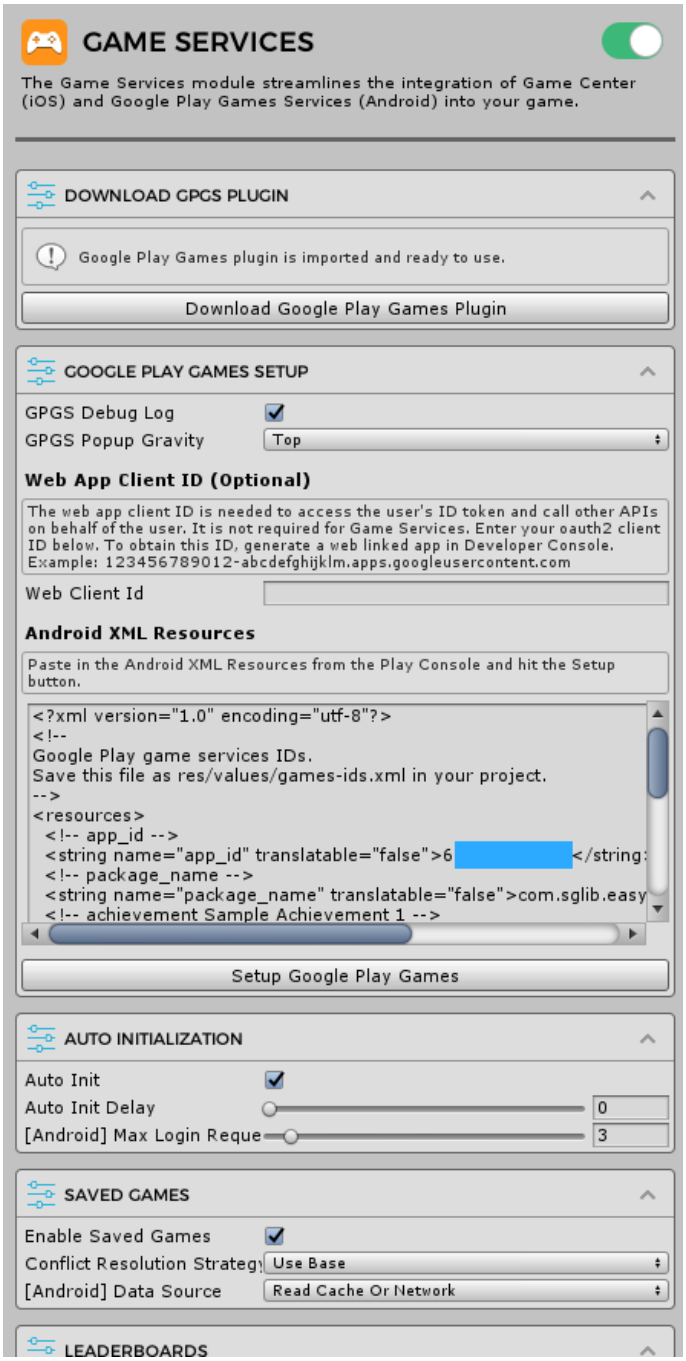
The next step is to create the required leaderboard and achievements for your targeted stores (i.e. in iTunes Connect for App Store and the Developer Console for Google Play). Take note of their IDs for use in the next step.

4.4.3 Easy Mobile setup

Setting up the Game Service module of Easy Mobile includes the following steps. Please see the Game Service section in Easy Mobile's user guide for detailed instructions on each step.

a. Import Google Play Games plugin for Unity and setup it if you're targeting Android
b. Enable the automatic initialization feature: just leave everything under the **AUTO-INIT CONFIG** section as default
c. Declare the leaderboards and achievements

Below is the settings interface of the Game Service module of Easy Mobile.

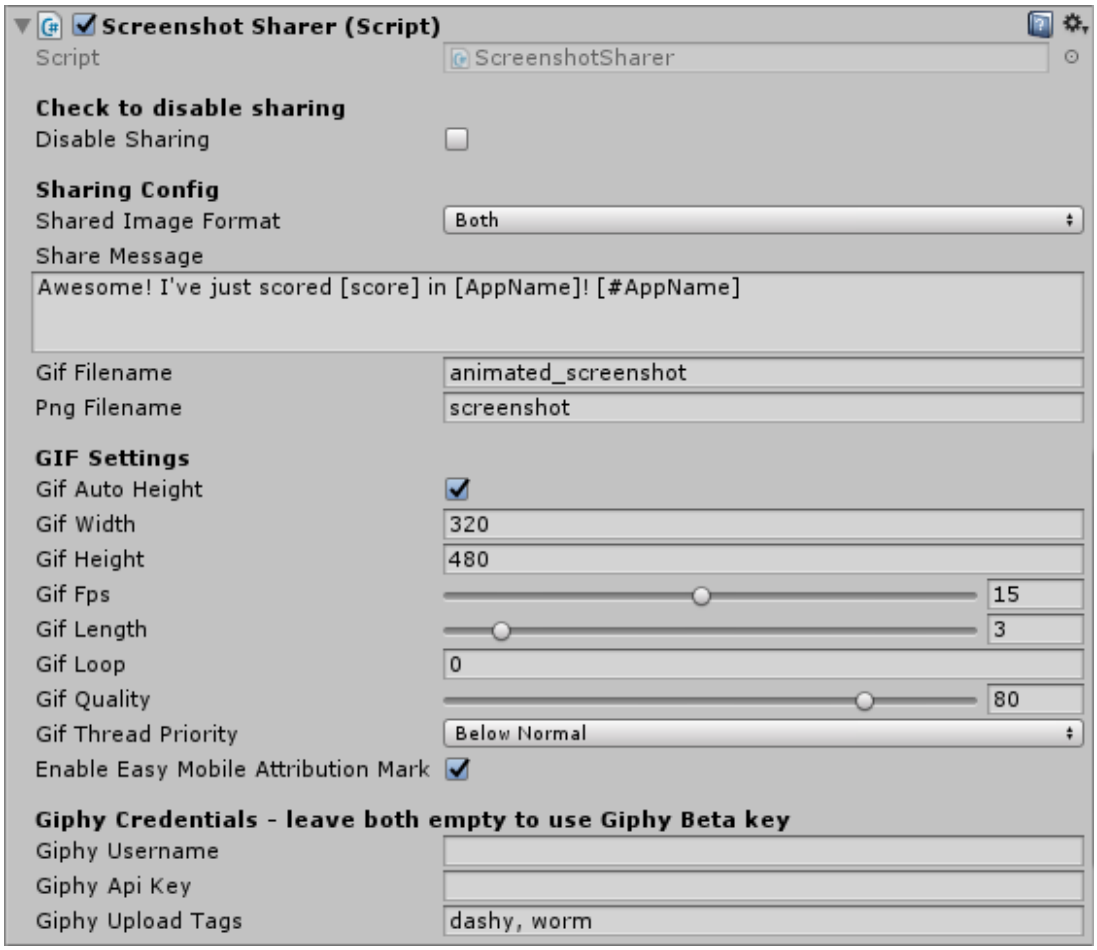


Note that you must declare the leaderboard and achievements with the same

names as the ones you have in the *AutoScoreReporter* and *AutoAchievementUnlocker* components. Also their IDs must match the ones you created in iTunes Connect and Google Play Developer Console.

4.5 Native Sharing

This template has a Share button that allows the user to share a screenshot of the gameplay (in animated GIF or static PNG format) to social networks using the native sharing functionality. This activity is managed by a component named *ScreenshotSharer*, which is also attached to the PremiumFeaturesManager object. When a new game starts, this component will setup and start a recorder to record the screen content, and stop the recorder when the game ends. The recorder automatically stores a few last seconds of the recording, and discards the rest. The recorded clip will then be exported to a GIF image, then uploaded to Giphy, and finally its URL can be shared and played automatically on major social networks including Facebook and Twitter.



Here you can configure the sharing feature.

- *Disable Sharing*: disable this feature
- *Share Image Format*: you can share GIF or PNG image, or both
- *Share Message*: the default sharing message, note that [score] will be automatically replaced by actual score, and [AppName] will be replaced by the app name declared in AppInfo
- *Gif Filename*: filename to store the generated GIF image
- *Png Filename*: filename to store the captured PNG image

In the **GIF Settings** section, you can configure the generation of the GIF image.

- *Gif Auto Height*: automatically calculate the image height based on the specified width and the screen aspect ratio
- *Gif Width*: the image width
- *Gif Height*: the image height, will be overwritten if *GifAutoHeight* is enabled
- *Gif Fps*: frame per second of the GIF image
- *Gif Length*: the length of the GIF in seconds, as mentioned earlier, the recorder only keeps this many seconds of the recording, and discards old content
- *Gif Loop*: looping mode of the GIF; 0 means loop indefinitely, -1 means no loop, > 0 means loop a set number of times
- *Gif Quality*: quality setting value in range [1,100], bigger values mean better quality but slightly longer generation time; 80 is generally a good value in terms of quality-time balance
- *Gif Thread Priority*: the priority of the GIF generation thread
- *Enable Easy Mobile Attribution Mark*: whether to display Easy Mobile attribution when generating GIF images

You can also control the Giphy uploading activity with the following parameters:

- *Giphy Username & Giphy Api Key*: provide these values if you want to upload the GIF image to your own Giphy channel; otherwise leave them empty to use the Giphy beta key
- *Giphy Upload Tags*: comma-delimited list of tags of the uploaded image

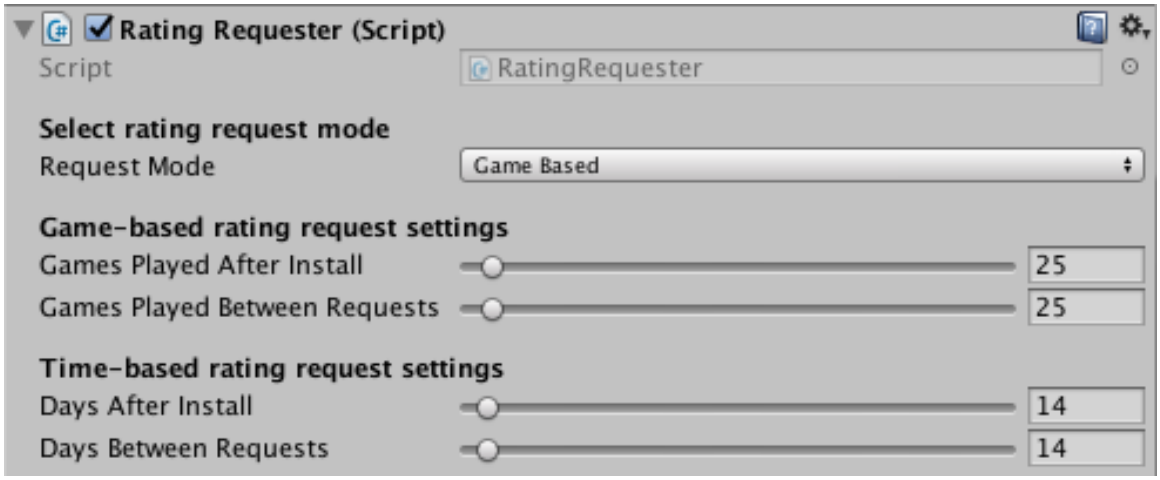
Note that you need to enable the *External Write Permission* for this feature to function properly on Android. Please see the Native Sharing section in Easy Mobile user guide for detailed instructions on doing that.

4.6 Rating Request

This template employs the Rating Request feature of Easy Mobile, to show a rate-my-app popup when game over, if some certain conditions are met. The Rating

Request feature of Easy Mobile allows us to show the built-in rating prompt on iOS (10.3+) and a native rating popup on Android. Please see the Rating Request section in Easy Mobile user guide for instructions on configuring the appearance and behavior of this popup.

You can set the conditions to show this rating popup using the *RatingRequester* component of the PremiumFeaturesManager object.



- *Request Mode*: whether to show the rating popup based on the number of games played (Game Based mode), or based on the time since the installation of the app (Time Based mode)

If you select *Game Based* mode, pay attention to these two variables:

- *Games Played After Install*: how many games should be played since the installation before a rating popup is shown
- *Game Played Between Requests*: how many games should be played since the last time a rating popup is shown (in case it was dismissed by the user) that a new popup can be shown

If you select *Time Based* mode, adjust these two variables:

- *Days After Install*: how many days after the installation that a rating popup is shown
- *Days Between Requests*: how many days since the last time a rating popup is shown that a new one can be shown

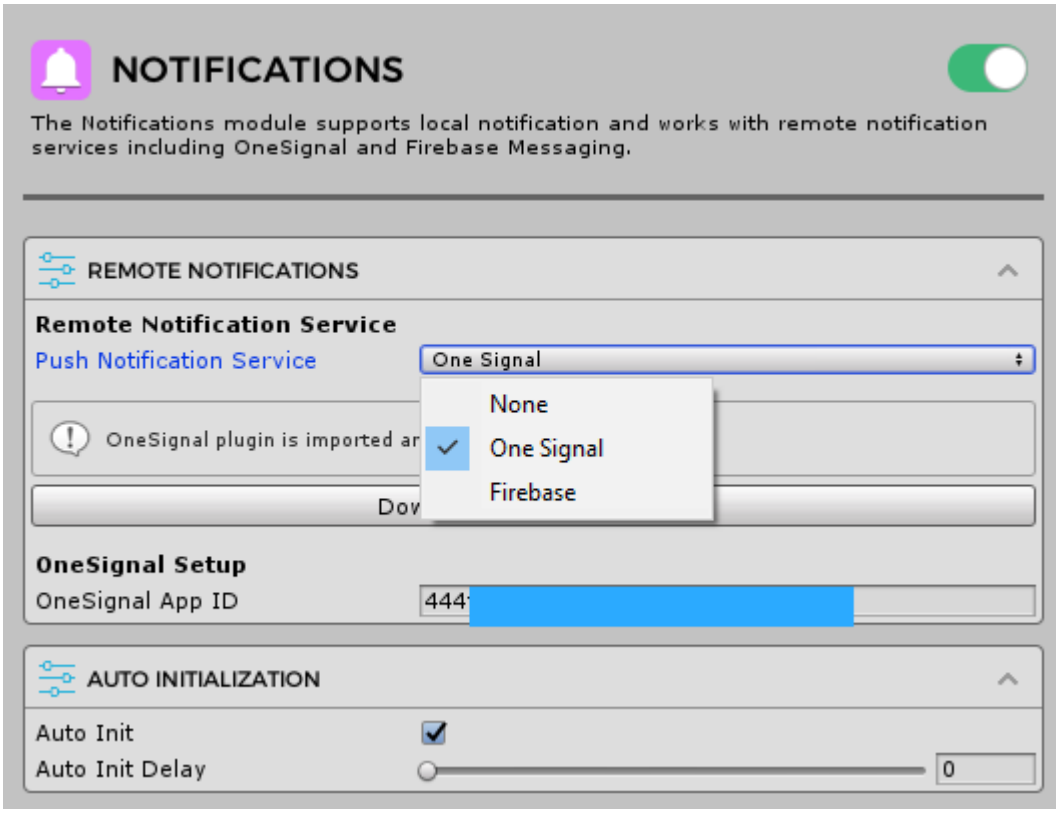
4.7 Push notification

Enabling push notification for your app using OneSignal or Firebase service includes following steps. Please see the Notifications section in Easy Mobile user

guide for detailed instructions on each step.

- Open the Notification tab in Easy Mobile's settings interface
- Import OneSignal or Firebase plugin
- Prepare your app for push notifications, e.g. enable the Push Notification capability for the provisioning profile on iOS (please see Easy Mobile user guide as well as OneSignal/Firebase documentation for detailed instructions).
- Add your app to OneSignal/Firebase dashboard
- Enter your app ID to Easy Mobile settings in Unity

Below is the settings interface of the Notification module of Easy Mobile after importing OneSignal/Firebase plugin and you can enter your app ID.



That's it! You've just finished implemented premium features for your game!

THANK YOU AND GOOD LUCK WITH YOUR GAMES!