



CIRCLE CRASH TEMPLATE

USER GUIDE

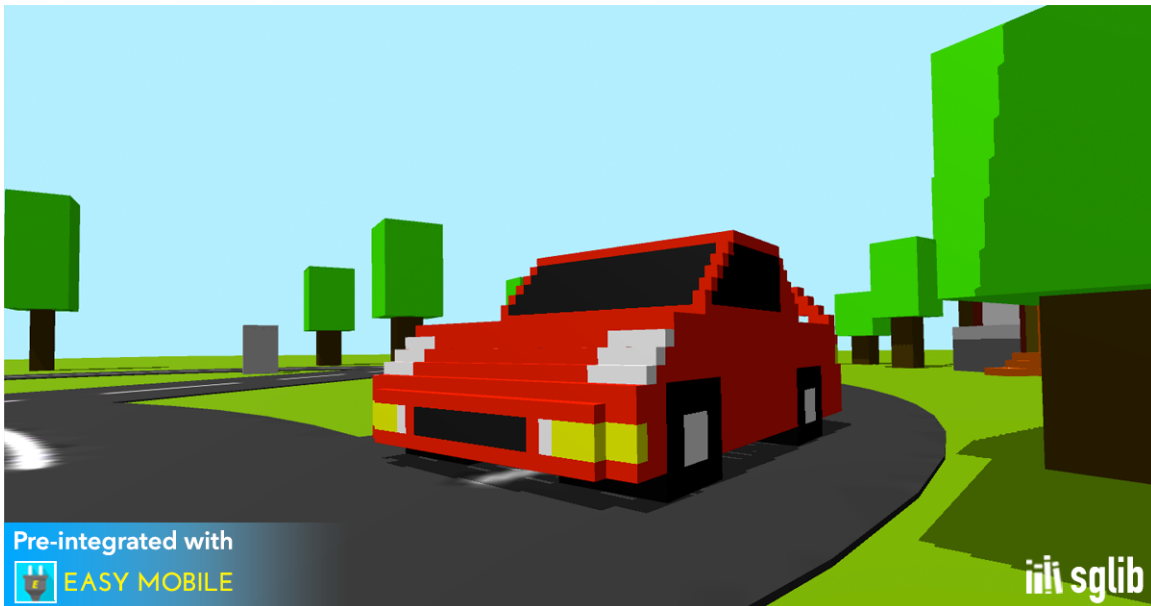
We strive to provide the best service as we can, if you have any questions or suggestions, please contact us!
Thank you!

SgLib Games

Table of Contents

1	INTRODUCTION.....	3
2	GETTING STARTED.....	4
2.1	ENTER APP INFORMATION	4
2.2	LINK THE GAME TO YOUR UNITY PROJECT.....	4
2.3	TESTING NOTE	6
3	TEMPLATE CUSTOMIZATION	6
3.1	GAMEPLAY TWEAKING	6
3.1.1	<i>GameManager</i>	6
3.1.2	<i>PlayerController</i>	7
3.1.3	<i>ObstacleController</i>	7
3.1.4	<i>CameraController</i>	8
3.2	DAILY REWARD FEATURE	8
3.3	ADDING MORE CHARACTERS (PLAYER CARS)	9
3.4	ADDING MORE OPPONENT CARS.....	11
3.5	ADDING MORE WORLDS.....	13
3.6	CUSTOMIZING UI	14
3.7	SOUNDS.....	15
4	ENABLING PREMIUM FEATURES.....	16
4.1	BEFORE YOU BEGIN	17
4.2	ADVERTISING	17
4.2.1	<i>Template-specific setup</i>	17
4.2.2	<i>Easy Mobile setup</i>	18
4.3	IN-APP PURCHASING.....	20
4.3.1	<i>Template-specific setup</i>	20
4.3.2	<i>Easy Mobile setup</i>	22
4.3.3	<i>Create the products for targeted stores</i>	24
4.4	GAME SERVICE.....	24
4.4.1	<i>Template-specific setup</i>	24
4.4.2	<i>Setup for your targeted stores</i>	26
4.4.3	<i>Easy Mobile setup</i>	27
4.5	NATIVE SHARING	28
4.6	RATING REQUEST	28
4.7	PUSH NOTIFICATIONS	29

1 INTRODUCTION



In **Circle Crash**, the player touches or releases the screen to rev up or slow down their car in order to avoid other hastily-moving cars, and collect coins to unlock new cool cars. This is a gorgeous, easy-to-play-hard-to-master 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
- Lots of built-in cars with cute blocky style to be unlocked with coins
- Free-to-use assets (fonts, sounds, music, model, etc.)
- Optimized for mobile

Most importantly, this template is *pre-integrated* with **Easy Mobile** plugin, so it can form a truly fully-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 AdMob, Chartboost, Heyzap (with mediation) and UnityAds
- In-app purchasing
- Support for Game Center (iOS) and Google Play Games Services (Android) for leaderboards and achievements
- Sharing to social networks

- Push notifications using OneSignal
- Native rating request (rate my app)

** Being pre-integrated means this template is already configured to work with Easy Mobile. All you need is import Easy Mobile and do a few setup steps, and have all the above features readily implemented. You don't even have to write a single line of integration code!*

** This template DOES NOT include Easy Mobile.*

** The use of Easy Mobile is totally optional: as long as it's not imported, all the integration code will automatically be excluded from compilation, so that no impact will be made on the game, which is fully functioning on its own.*

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.

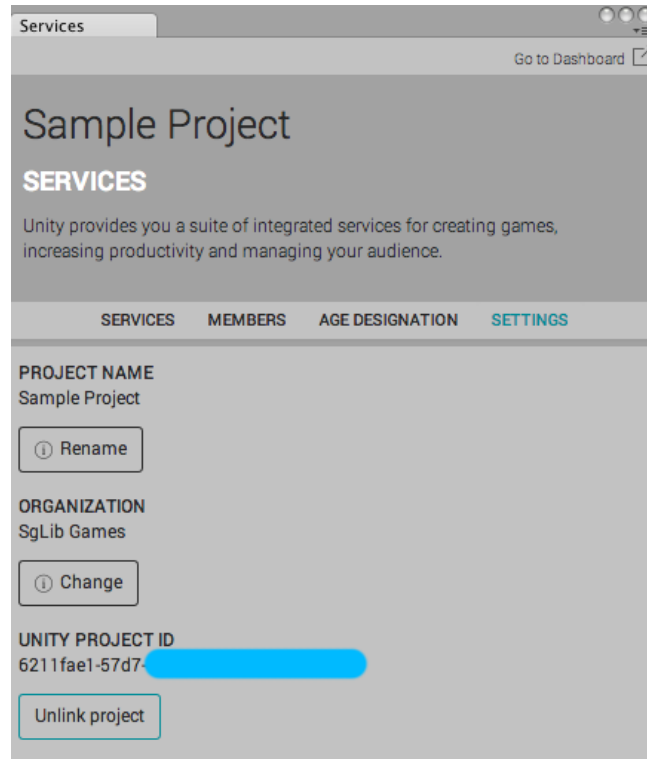


Field	Value
APP_NAME	[YOUR_APP_NAME]
APPSTORE_ID	[YOUR_APPSTORE_ID]
BUNDLE_ID	[YOUR_BUNDLE_ID]
APPSTORE_HOMEPAGE	[YOUR_APPSTORE_PUBLISHER_LINK]
PLAYSTORE_HOMEPAGE	[YOUR_GOOGLEPLAY_PUBLISHER_NAME]
FACEBOOK_ID	[YOUR_FACEBOOK_PAGE_ID]
TWITTER_NAME	[YOUR_TWITTER_PAGE_NAME]
SUPPORT_EMAIL	[YOUR_SUPPORT_EMAIL]

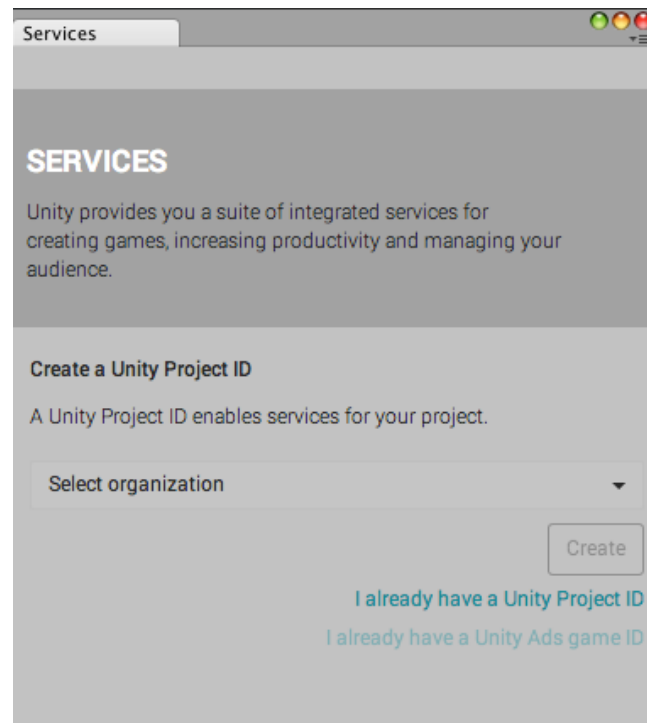
2.2 Link the game to your Unity project

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 your game is linked to your own Unity project and is ready to use Unity

services.

2.3 Testing Note

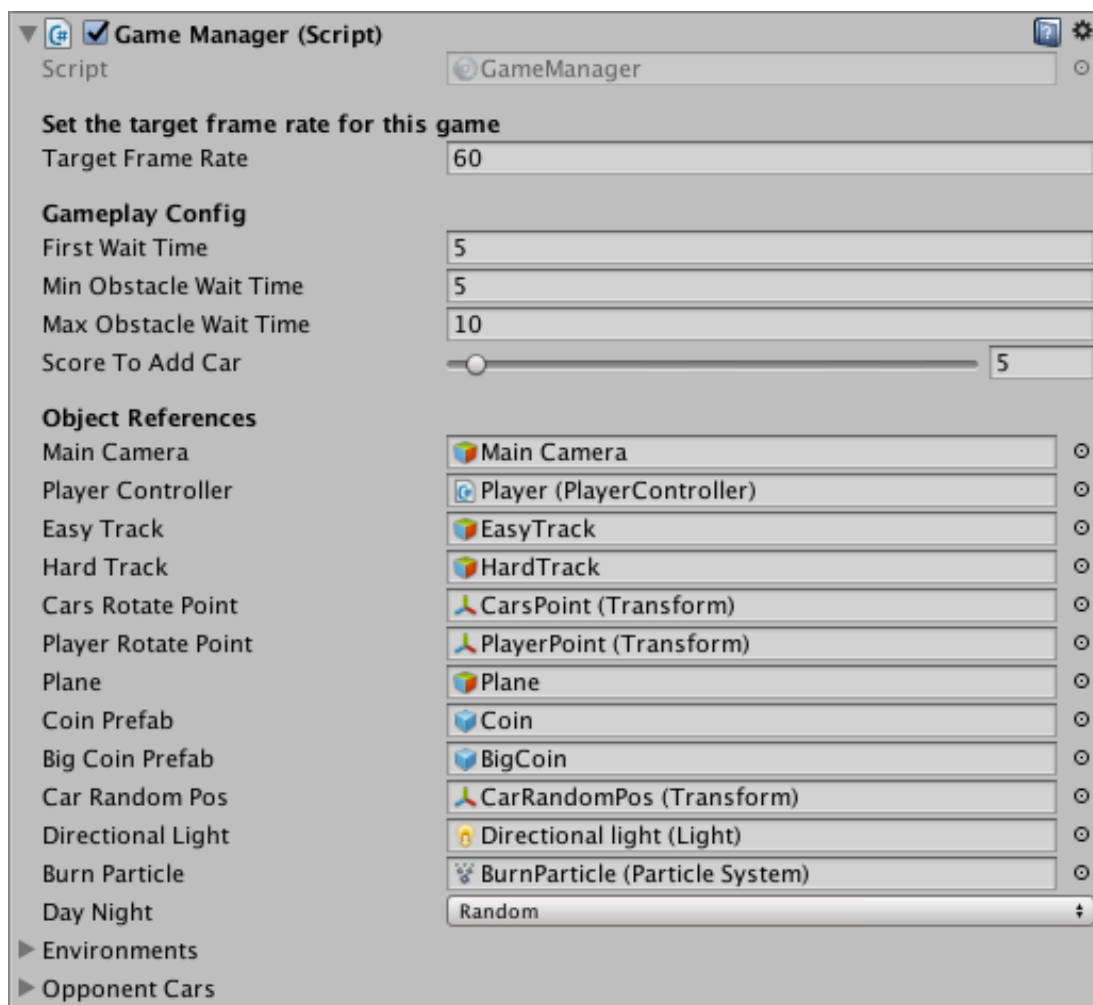
There are 2 scenes in this game, it should be run from scene *Main*.

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.

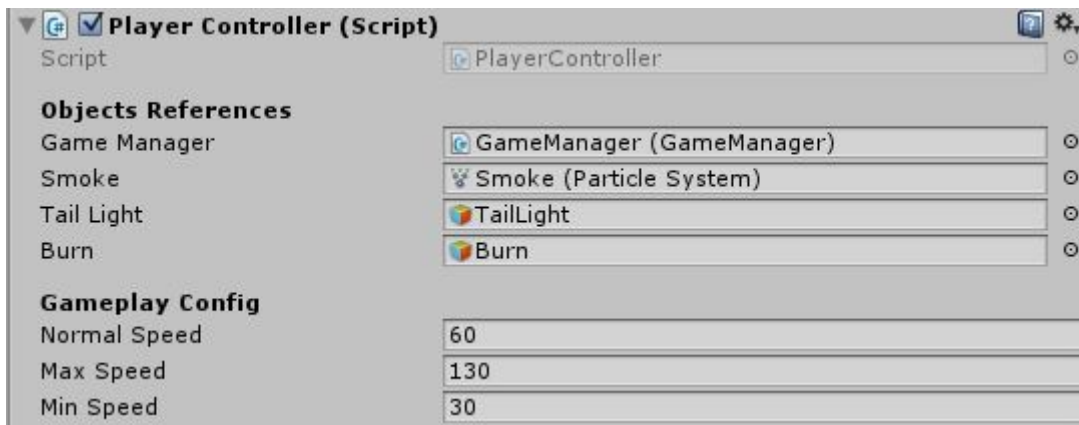


You can tweak the gameplay by modifying following variables:

- **TargetFrameRate:** set the target frame rate for the game, which should be at least 60fps for games that requires smooth, fast motion.
- **FirstWaitTime:** the delay time to create the first car that runs on the straight track (Hard mode only)
- **MinObstacleWaitTime & MaxObstacleWaitTime:** maximum & minimum wait time to create opponent cars that run on the straight track (Hard mode only), the actual wait time are randomized between these two values.
- **ScoreToAddCar:** every time the player reaches a score that is a multiply of this value, a new car will be added to the circular track.

3.1.2 PlayerController

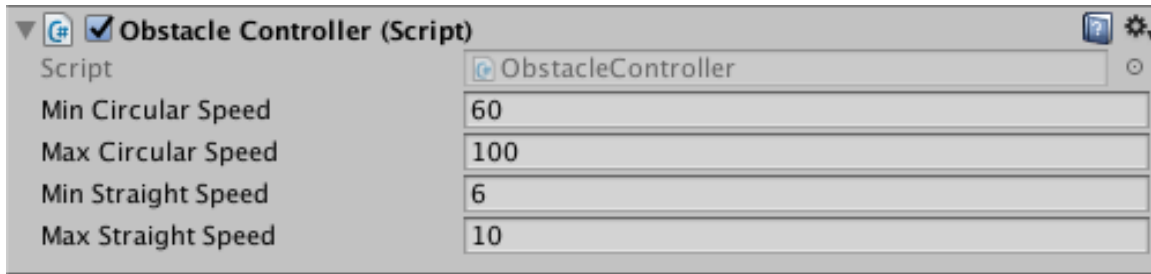
The PlayerController component is attached to the Player gameobject in the hierarchy and controls the behavior of the player car.



- **NormalSpeed:** the normal speed of the player.
- **MaxSpeed:** the maximum speed of the player.
- **MinSpeed:** the minimum speed of the player.

3.1.3 ObstacleController

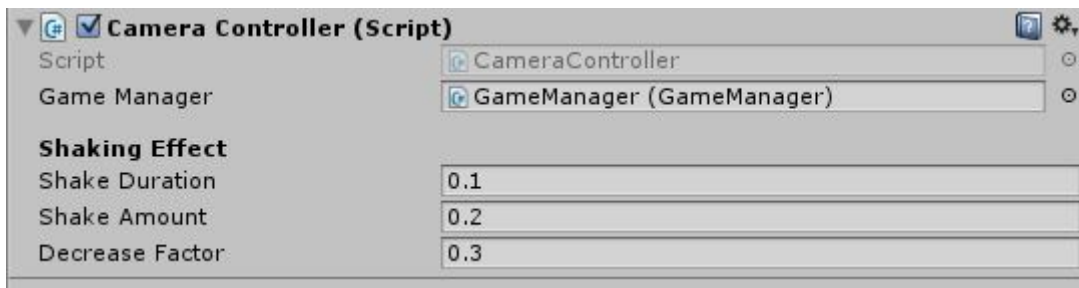
Each opponent car has an ObstacleController script attached to it which controls its behavior in game. You can find these prefabs under folder *Prefabs/Game/OtherCars*.



- MinCircularSpeed & MaxCircularSpeed: minimum & maximum speed of this vehicle if it is generated on the circular track. Note that this speed is actually the rotation speed (degree/second).
- MinStraightSpeed & MaxStraightSpeed: minimum & maximum speed of this vehicle if it is generated on the straight track (Hard mode only).

3.1.4 CameraController

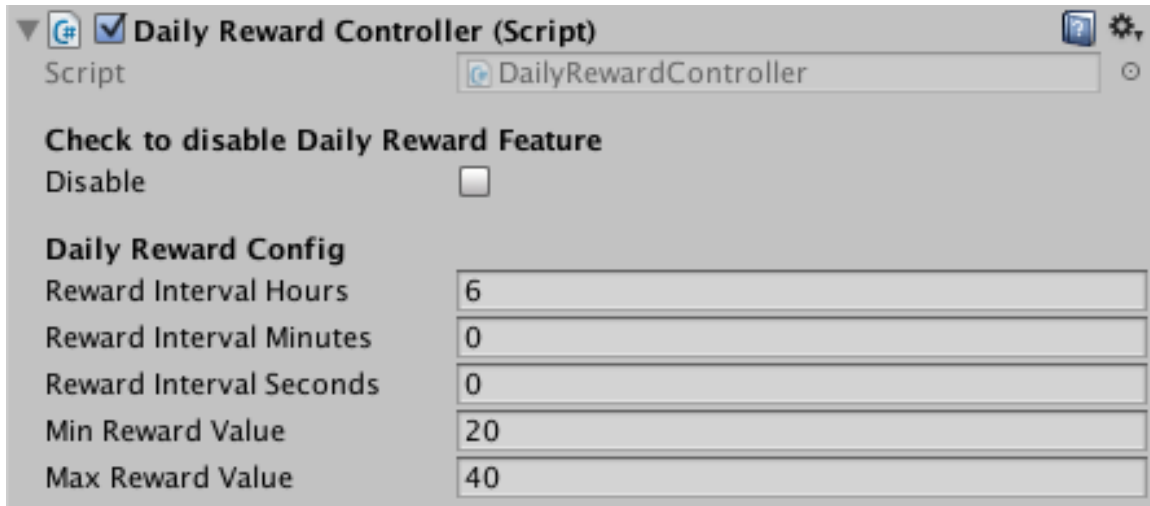
The CameraController component is attached to the MainCamera object in the hierarchy.



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

3.2 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.

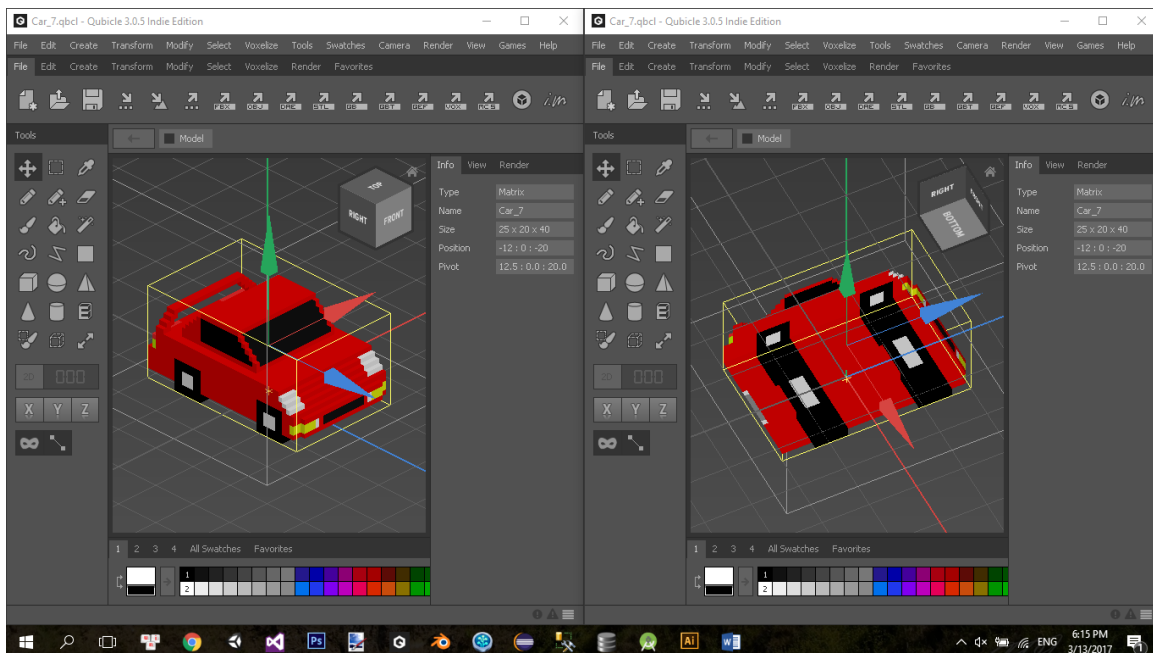


- *Disable*: check to disable this feature
- *Reward Interval Hours, Minutes and Seconds*: the amount of time until the next reward
- *Min Reward Value & Max Reward Value*: the actual rewarded coins will be randomized between these two values

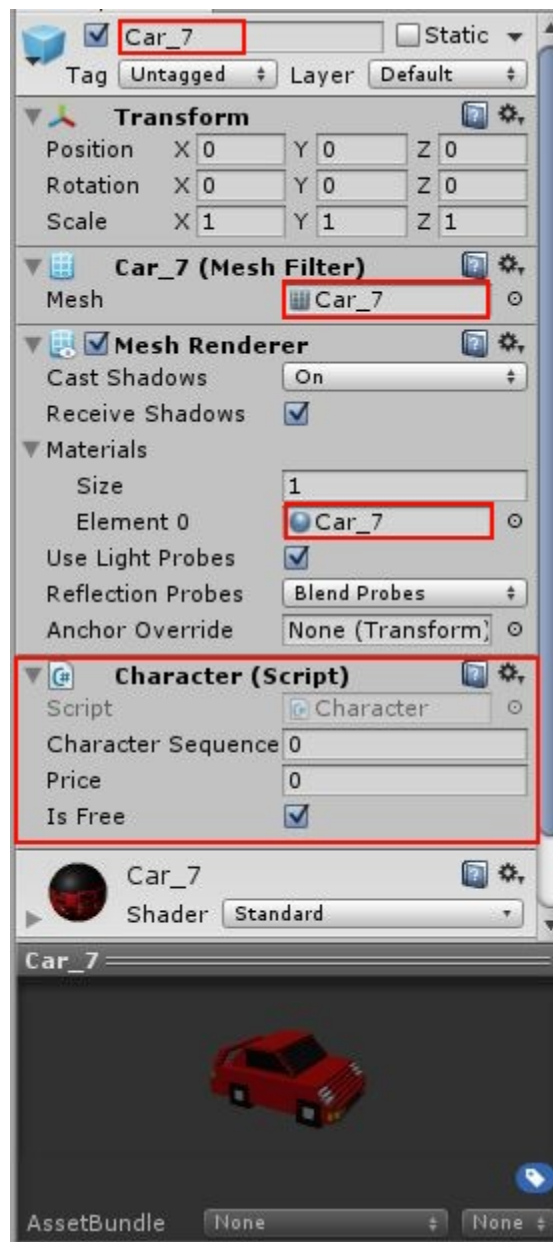
3.3 Adding more characters (player cars)

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:

1. Create a character model with the pivot at the bottom center.



2. Navigate to *Assets/Prefabs/Characters/CharacterPrefabs* and duplicate one of the available character prefabs.
3. Change the name of the prefab to a preferred one.
4. Replace the *Mesh* in the *MeshFilter* component with your new model mesh.
5. Replace the *Material* in the *MeshRenderer* component with your new character material.
6. 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.*
7. Resize the character array in *CharacterManager* game object then drag the new character to it and hit Apply to save changes to its prefab.

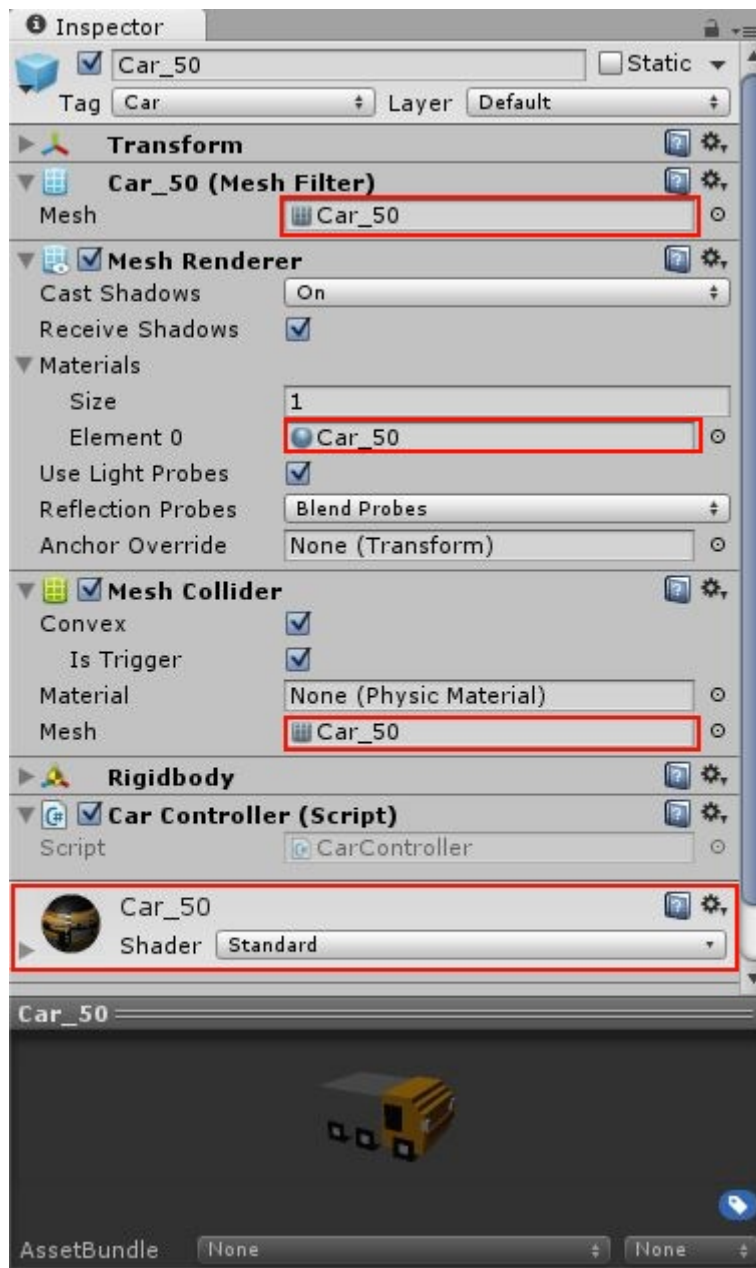


Now the new character has been added and ready to use in game! You will see it listed in the *CharacterSelection* scene.

3.4 Adding more opponent cars

1. Create a character model with the pivot at the bottom center.
2. Navigate to *Prefabs/Game/OtherCars* and duplicate one of the available character prefabs.
3. Change the name of the prefab to a preferred one.

4. Replace the *Mesh* in the *MeshFilter* component with your new model mesh.
5. Replace the *Material* in the *MeshRenderer* component with your new car material.
6. Replace the *Mesh* in the *MeshCollider* component with your new model mesh, set *IsTrigger* to *True*.
7. Add *Rigidbody* and set *IsKinematic* to *True*.
8. Add *ObstacleController* script to the prefab.
9. Set the prefab tag as “Obstacle”.
10. Resize the *OpponentCars* array in *GameManager* game object then drag the new character to it and hit Apply to save changes to its prefab.

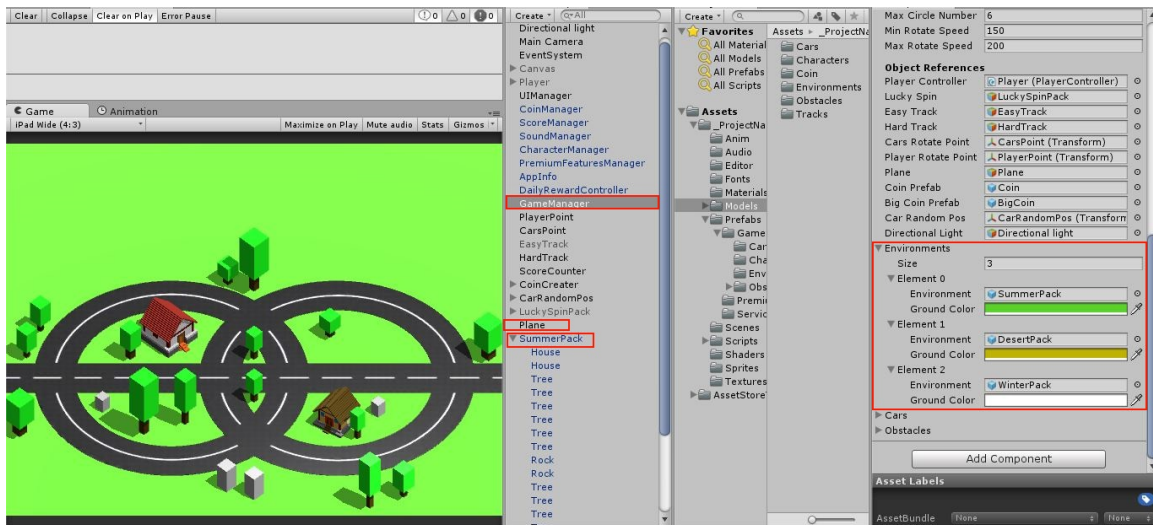


3.5 Adding more worlds

1. Create an empty object in Scene view.
2. Go to *Assets/Models/Environment* and drag those models into Scene view, as children of the newly created object, to form the world you want.
3. Add *MeshCollider* to these environment models.
4. Find the object named "Plane" and change the color of its material to fit

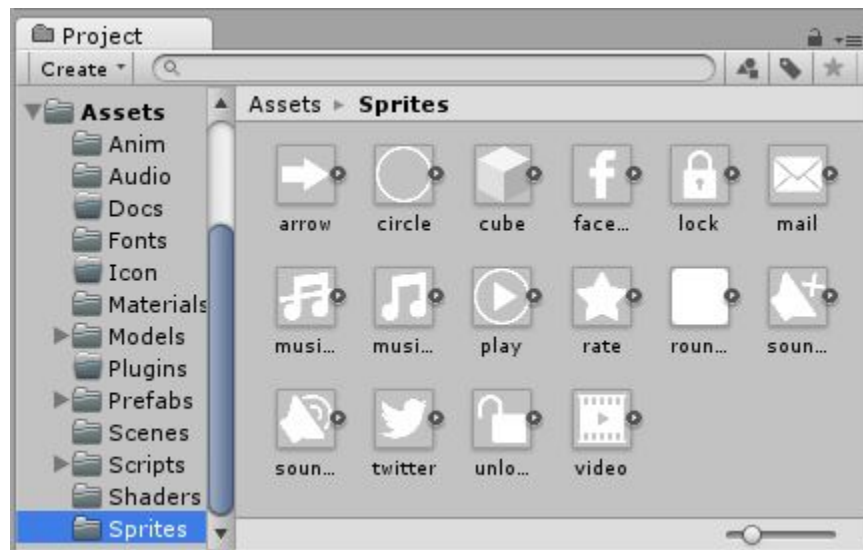
the newly created world.

5. Drag the parent object (the empty object created in the first step) into Assets/Prefabs/Game/Environments to make it a prefab.
6. Enlarge the *Environments* array in *GameManager* game object then drag the new prefab to it.
7. Set the GroundColor to the current color of the plane.

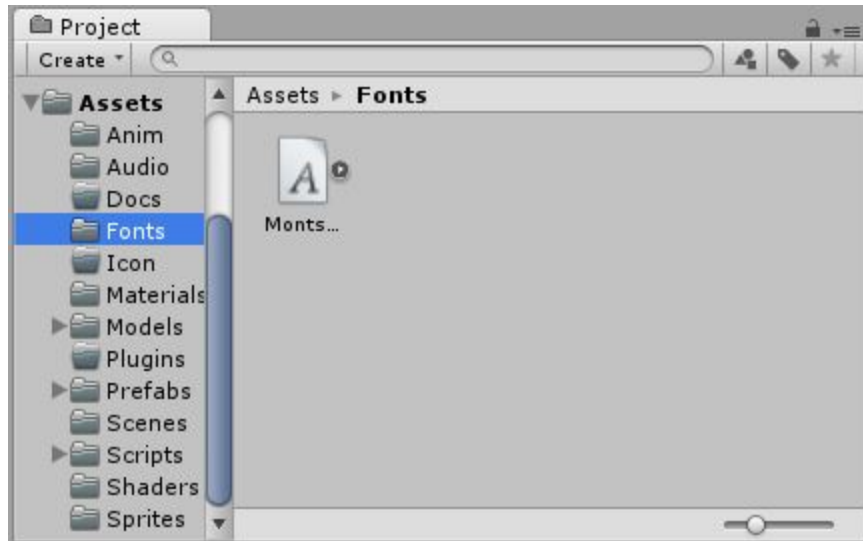


3.6 Customizing UI

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.

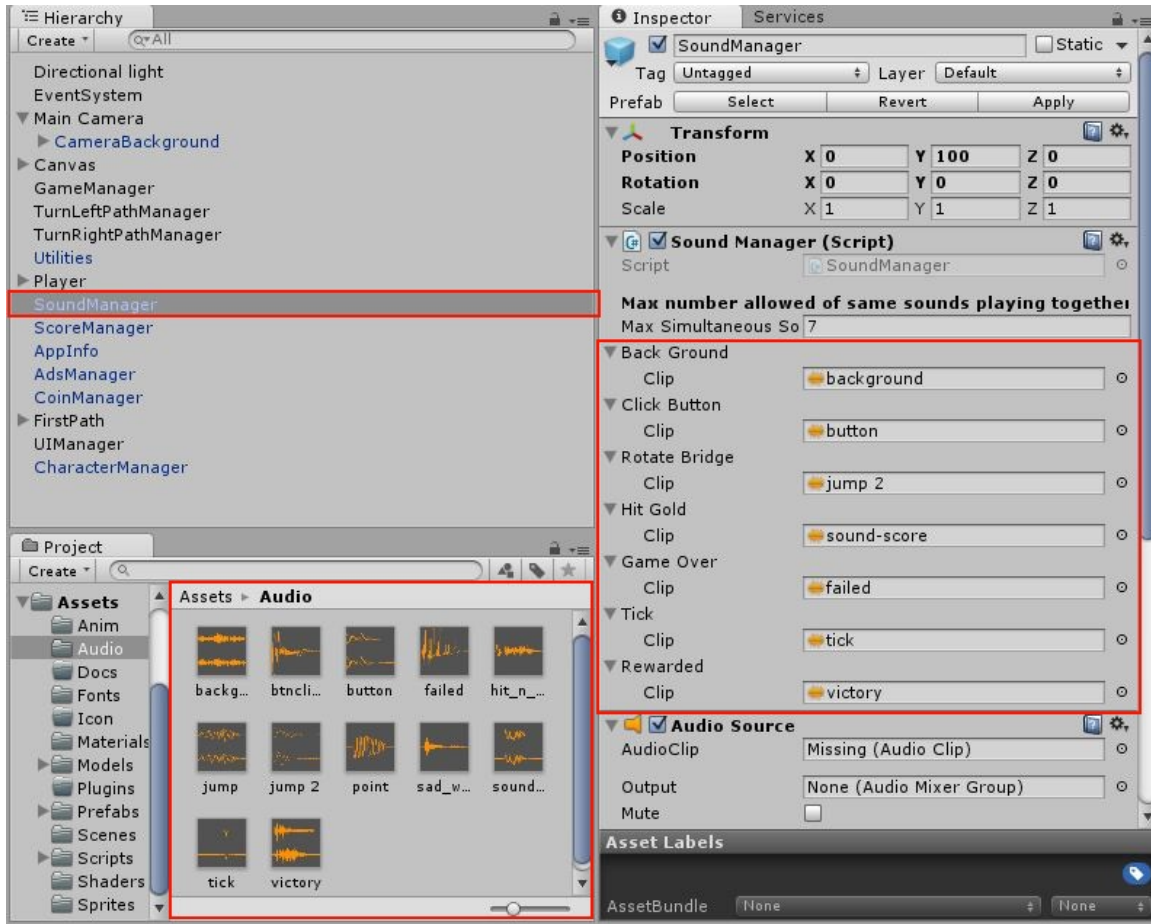


3.7 Sounds

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

Premium features include:

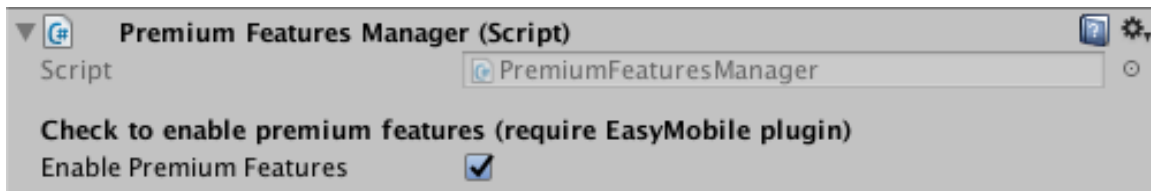
- Advertising
- In-app purchasing
- Leaderboards & achievements
- Sharing
- Rating Request
- Push notifications

To enable premium features of this template, you need to download and import Easy Mobile plugin from <http://u3d.as/Dd2>.

This section provides a guide on configuring each feature for your game. If you're not familiar with using Easy Mobile, it is strongly recommended that you read through its user guide to familiarize yourself with the plugin <https://sglibgames.gitbooks.io/easy-mobile-user-guide/content/>.

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 *PremiumFeaturesManager* object is checked.

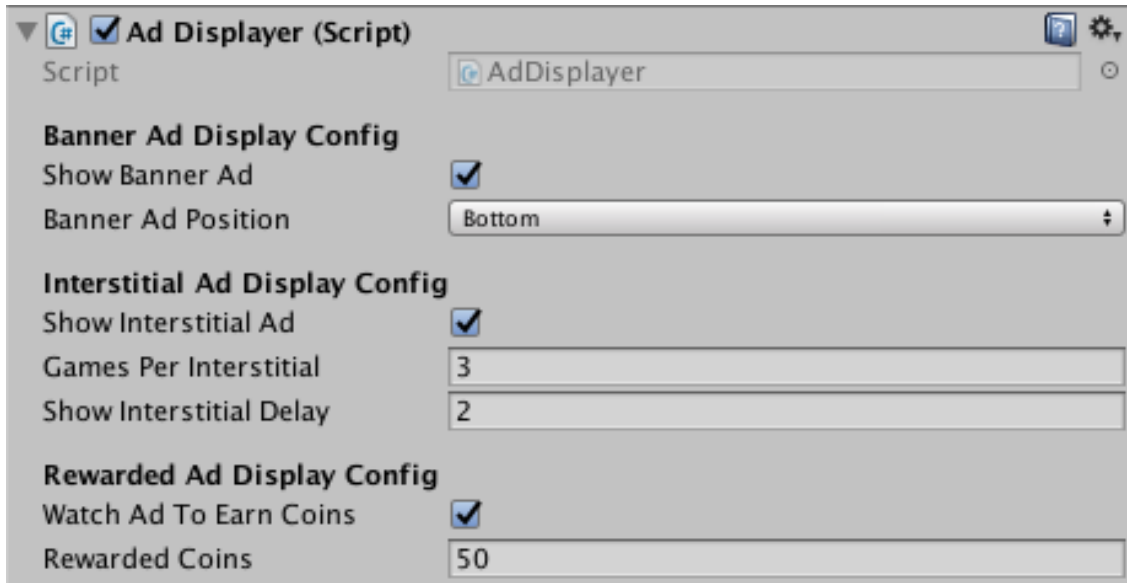


- Make sure there's an EasyMobile object in the Main scene's hierarchy. You can add it by dragging the EasyMobile prefab at folder *Assets/EasyMobile* to the scene hierarchy. It is necessary for the plugin to function properly.
- The settings interface of Easy Mobile can be opened via menu *Window > 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 safely ignore all the Scripting sections in Easy Mobile user guide).

4.2 Advertising

4.2.1 Template-specific setup

The *PremiumFeaturesManager* 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

With Easy Mobile's Advertising module, you'll have support for AdMob, Chartboost, Heyzap (with mediation) and Unity Ads. You can use multiple ad networks at once and have different configurations for iOS and Android. For example, you can use AdMob for banner ads, Chartboost for interstitial ads and Unity Ads for rewarded ads on iOS, and yet another combination on Android.

To configure the Advertising module, open Easy Mobile settings interface and select the Advertising tab. Below is the settings interface of the module.

ADVERTISING

ADMOB SETUP

Google Mobile Ads (AdMob) plugin was imported.

Reimport Google Mobile Ads Plugin

► [iOS] AdMob Ids
► [Android] AdMob Ids

CHARTBOOST SETUP

Chartboost plugin not found. Please download and import it to show ads from Chartboost.

Download Chartboost Plugin

HEYZAP SETUP

Heyzap plugin not found. Please download and import it to show ads from Heyzap.

Download Heyzap Plugin

UNITY ADS SETUP

Unity Ads service is enabled.

AUTO AD-LOADING CONFIG

Auto-Load Default Ads ☒

Ad Checking Interval 10

Ad Loading Interval 20

DEFAULT AD NETWORKS

▼ [iOS] Default Ad Networks

Banner Ad Network Ad Mob

Interstitial Ad Network Ad Mob

Rewarded Ad Network Unity Ads

▼ [Android] Default Ad Networks

Banner Ad Network Ad Mob

Interstitial Ad Network Ad Mob

Rewarded Ad Network Unity Ads

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.

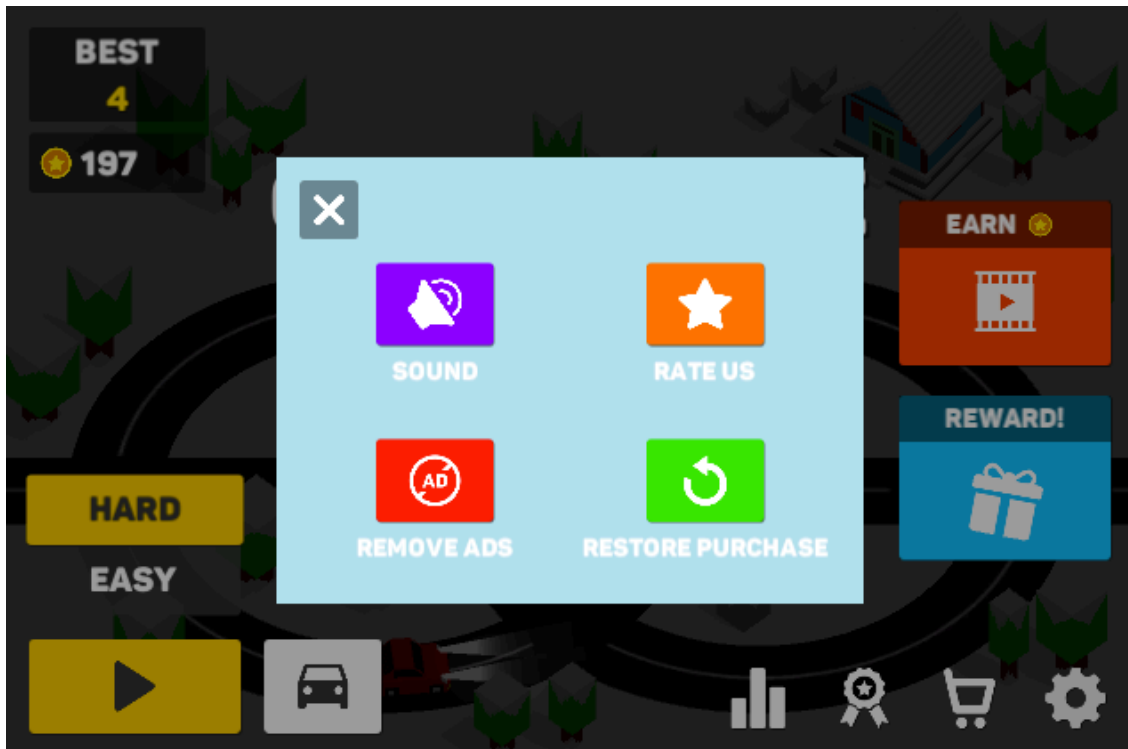
- Setup the ad networks you want to use, this includes importing the required plugins for each network, please see Easy Mobile user guide for more information
- Enable auto ad-loading feature: simply leave the *Auto-Load Default Ads* option as checked and other parameters as default, the plugin will automatically load ads in the background
- 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!

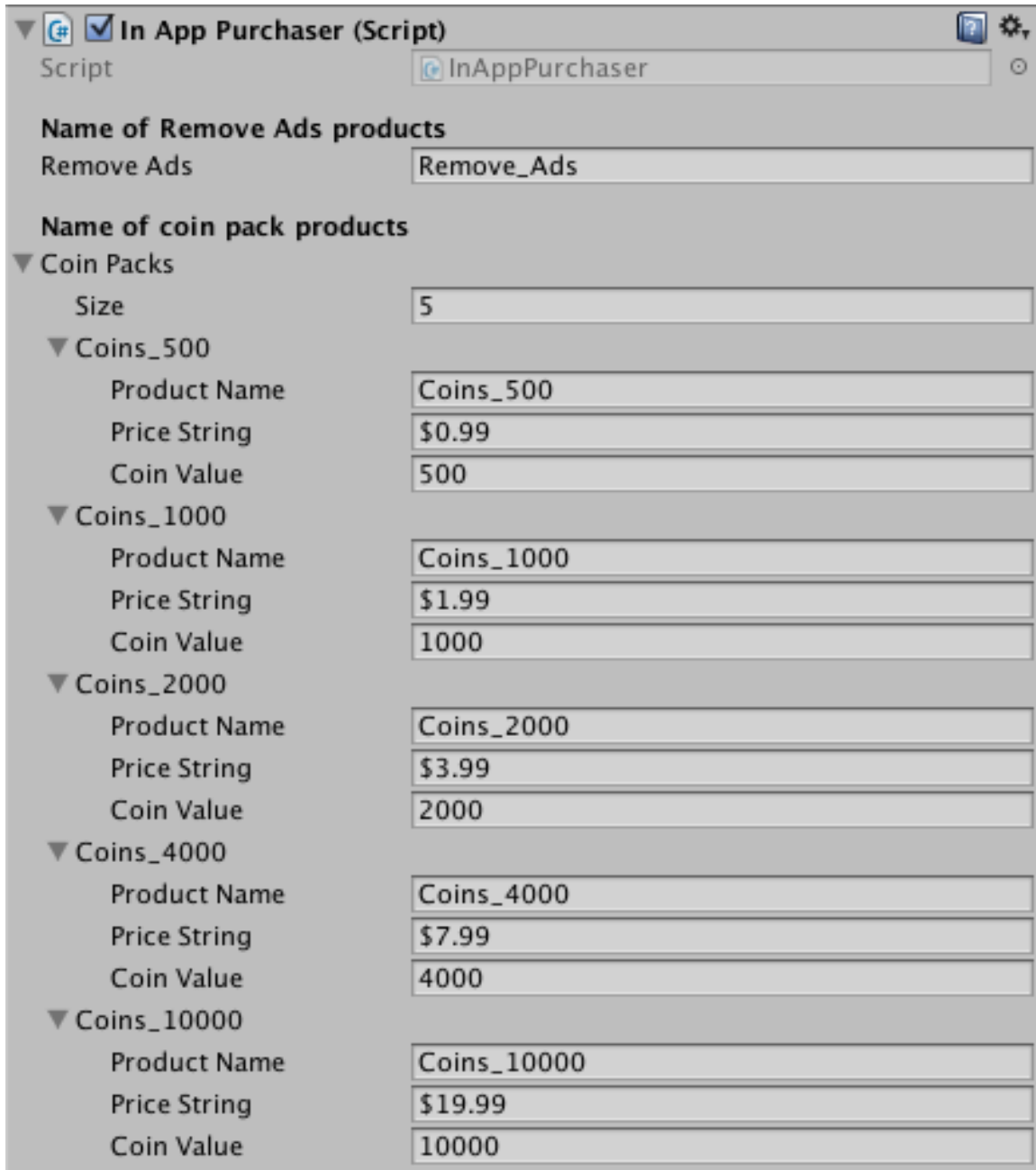
4.3 In-App Purchasing

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



In App Purchaser (Script)

Script: InAppPurchaser

Name of Remove Ads products

Remove Ads: Remove_Ads

Name of coin pack products

▼ Coin Packs

Size: 5

▼ Coins_500

Product Name: Coins_500

Price String: \$0.99

Coin Value: 500

▼ Coins_1000

Product Name: Coins_1000

Price String: \$1.99

Coin Value: 1000

▼ Coins_2000

Product Name: Coins_2000

Price String: \$3.99

Coin Value: 2000

▼ Coins_4000

Product Name: Coins_4000

Price String: \$7.99

Coin Value: 4000

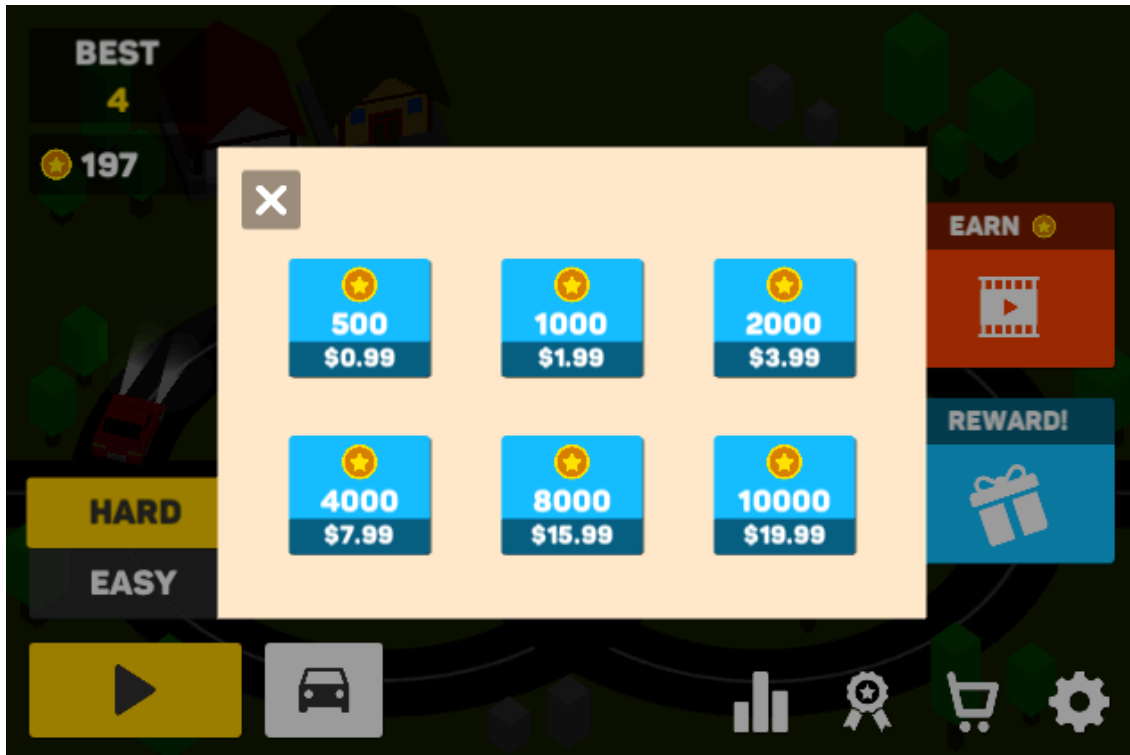
▼ Coins_10000

Product Name: Coins_10000

Price String: \$19.99

Coin Value: 10000

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.

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

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

IN-APP PURCHASING

[ANDROID] TARGET STORE


Target Android Store Google Play

RECEIPT VALIDATION

Unity IAP offers local receipt validation for extra security. Apple stores and Google Play store only.

Validate Apple Receipt ☐

Validate Google Play Receipt ☐

 Please go to Window > Unity IAP > IAP Receipt Validation Obfuscator and create obfuscated secrets to enable receipt validation for Apple stores and Google Play store. Note that you don't need to provide a Google Play public key if you're only targeting Apple stores.

PRODUCTS

► 6 Products

Add New Product

CONSTANTS CLASS GENERATION

Generate the static class EasyMobile.EM_IAPConstants that contains the constants of product names. Remember to regenerate if you make changes to these names.

Generate Constants Class

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.

The screenshot displays two product configuration panels in a light gray interface. The top panel is titled 'Remove_Ads' and contains three input fields: 'Name' with the value 'Remove_Ads', 'Type' with a dropdown menu showing 'Non Consumable', and 'Id' with the value 'sglib.demogame.iap.remove_ads'. Below these fields is a link '► More (Optional)'. To the right of this panel are three vertically stacked buttons: an up arrow, a minus sign, and a down arrow. The bottom panel is titled 'Coins_500' and contains three input fields: 'Name' with the value 'Coins_500', 'Type' with a dropdown menu showing 'Consumable', and 'Id' with the value 'sglib.demogame.iap.coins_500'. Below these fields is a link '► More (Optional)'. To the right of this panel are three vertically stacked buttons: an up arrow, a minus sign, and a down arrow.

4.3.3 Create the products for targeted stores

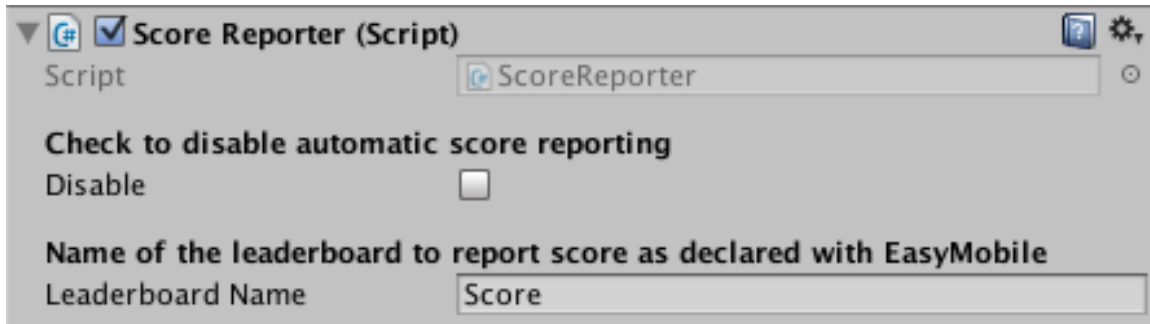
That last step in configuring the in-app purchasing feature is to create products for your targeted stores (e.g. Google Play and Apple App Store). Make sure the product ID, product type and price match the ones you have in your game.

4.4 Game Service

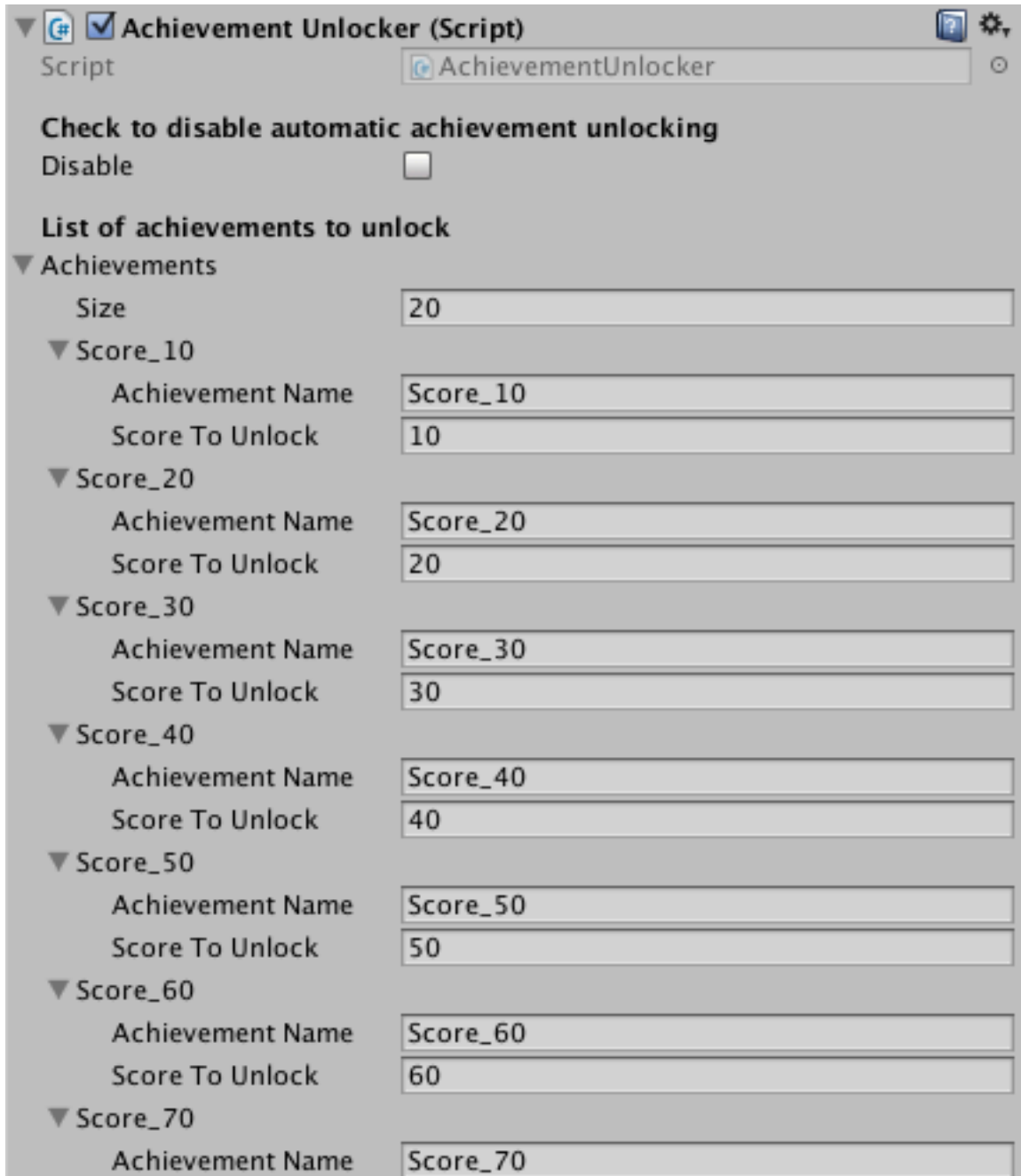
4.4.1 Template-specific setup

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 a component named *AchievementUnlocker*. From this object, you can modify existing achievements and add or remove achievements. You can also disable the automatic achievement unlocking feature if you wish.



▼ Achievement Unlocker (Script)

Script

Check to disable automatic achievement unlocking

Disable ☐

List of achievements to unlock

▼ Achievements

Size

▼ Score_10

Achievement Name

Score To Unlock

▼ Score_20

Achievement Name

Score To Unlock

▼ Score_30

Achievement Name

Score To Unlock

▼ Score_40

Achievement Name

Score To Unlock

▼ Score_50

Achievement Name

Score To Unlock

▼ Score_60

Achievement Name

Score To Unlock

▼ Score_70

Achievement Name

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.

- Setup Google Play Games if you're targeting Android
- Enable the automatic initialization feature: just leave everything under the **AUTO-INIT CONFIG** section as default
- Declare the leaderboards and achievements

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

GAME SERVICE ☒

! Google Play Games plugin is imported and ready to use.

Reimport Google Play Games Plugin

[ANDROID] GOOGLE PLAY GAMES SETUP

GPGS Debug Log ☐

Paste in the Android XML Resources from the Play Console and hit the Setup button.

Android XML Resources

```
<?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>
<string name="app_id">104</string>
<string name="package_name">com.sglib.demogame</string>
<string name="achievement_score_10">Cgkl3tzAhK8eEAIQEg</string>
<string name="achievement_score_20">Cgkl3tzAhK8eEAIQAQ</string>
<string name="achievement_score_30">Cgkl3tzAhK8eEAIQEw</string>
</resources>
```

Setup Google Play Games

AUTO-INIT CONFIG

Auto Init ☒

Auto Init Delay

[Android] Max Login Requests

LEADERBOARD SETUP

▶ 1 Leaderboards

Add New Leaderboard

ACHIEVEMENT SETUP

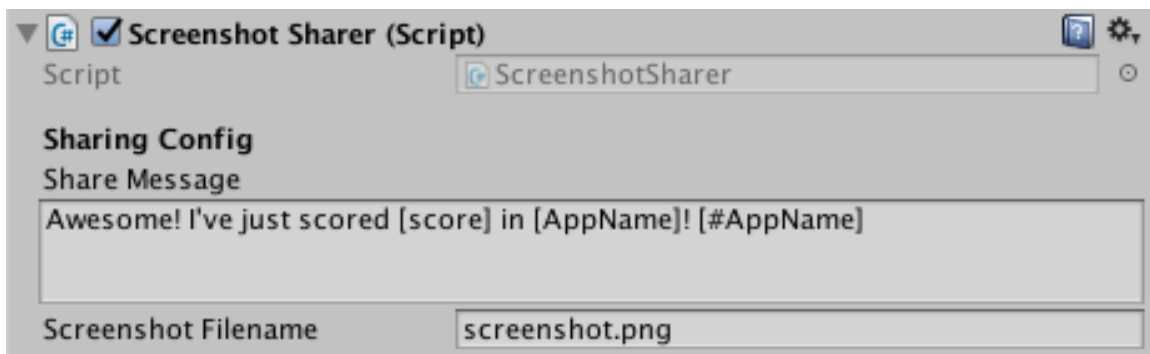
▶ 20 Achievements

Add New Achievement

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 the game's screenshot 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.



Here you can configure the sharing feature.

- 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
- Screenshot Filename: filename to store the screenshot in the device storage

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

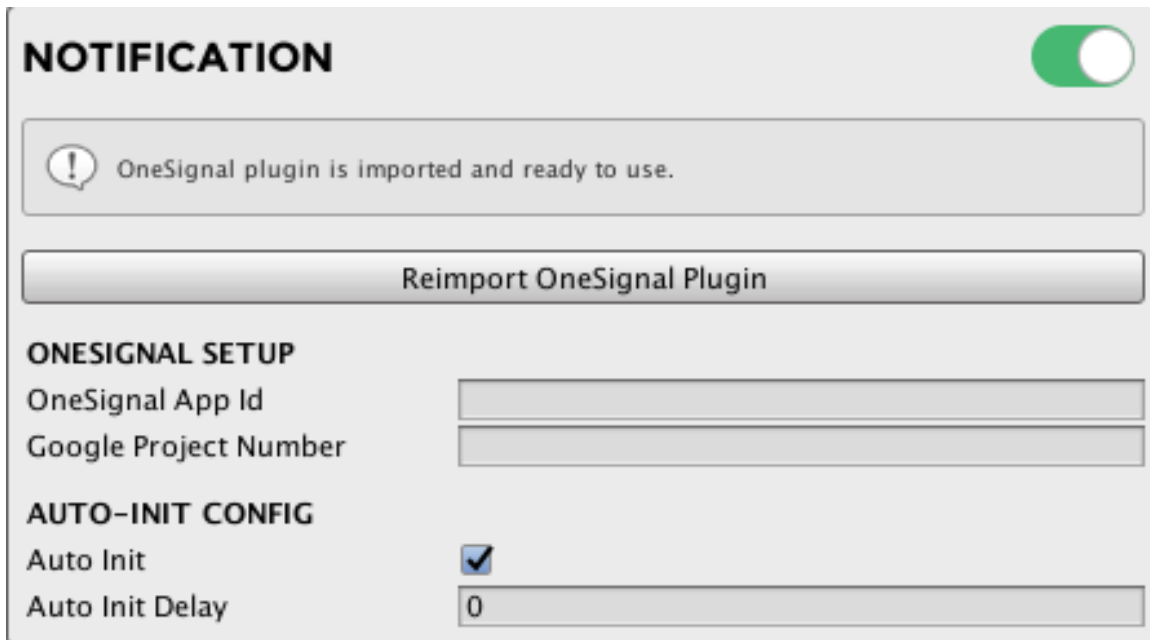
4.7 Push notifications

Enabling push notifications for your app using OneSignal service includes following steps. Please see the Notification section in Easy Mobile user guide for detailed instructions on each step.

- Open the Notification tab in Easy Mobile's settings interface
- Import OneSignal 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 documentation for detailed instructions).
- Add your app to OneSignal dashboard

- Enter your app ID to Easy Mobile settings in Unity

Below is the settings interface of the Notification module of Easy Mobile where you can enter your app ID and Google project number.



The screenshot shows the 'NOTIFICATION' settings panel. At the top right is a green toggle switch. Below the title is a message box with an exclamation mark icon stating 'OneSignal plugin is imported and ready to use.' Underneath is a button labeled 'Reimport OneSignal Plugin'. The 'ONESIGNAL SETUP' section contains two input fields: 'OneSignal App Id' and 'Google Project Number'. The 'AUTO-INIT CONFIG' section has a checked checkbox for 'Auto Init' and a numeric input field for 'Auto Init Delay' with the value '0'.

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

THANK YOU AND GOOD LUCK WITH YOUR GAMES!