



FLICKY BLADE

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
3.2	CONFIG SHOCK WAVE EFFECT	7
3.3	CONFIG THE GRASS EFFECT	8
3.4	CHANGING SKYBOX COLOR	8
3.5	DAILY REWARD FEATURE	8
3.6	ADDING MORE KNIVES:	9
3.7	MOVING TARGET CONFIG	12
3.8	CUSTOMIZING UI	13
3.9	SOUNDS	13
4	ENABLING PREMIUM FEATURES	15
4.1	BEFORE YOU BEGIN	16
4.2	ADVERTISING	16
4.2.1	<i>Template-specific setup</i>	<i>16</i>
4.2.2	<i>Easy Mobile setup</i>	<i>17</i>
4.3	IN-APP PURCHASING	18
4.3.1	<i>Template-specific setup</i>	<i>18</i>
4.3.2	<i>Easy Mobile setup</i>	<i>21</i>
4.3.3	<i>Create the products for targeted stores</i>	<i>23</i>
4.4	GAME SERVICE	23
4.4.1	<i>Template-specific setup</i>	<i>23</i>
4.4.2	<i>Setup for your targeted stores</i>	<i>25</i>
4.4.3	<i>Easy Mobile setup</i>	<i>26</i>
4.5	NATIVE SHARING	27
4.6	RATING REQUEST	28
4.7	PUSH NOTIFICATION	29

1 INTRODUCTION



Show your blade flipping skill in this game **Flicky Blade** – a simple cool game that is suitable for people of all ages.

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

- Trending flip-something gameplay
- Daily reward system for better retention
- Tons of built-in cool blades for unlock.
- Free-to-use assets (fonts, sounds, music, model, etc.)
- Optimized for mobile

Most importantly, this template is pre-integrated with **Easy Mobile** plugin, making it a truly fully-featured game that is release-ready. Easy Mobile is a comprehensive, cross-platform package that provides most of desired features of mobile games:

- Support for AdColony, AdMob, Chartboost, Heyzap and UnityAds
- In-app purchasing
- Support for Game Center (iOS) and Google Play Games Services (Android) for leaderboards and achievements
- Recording gameplay and sharing animated GIF images or PNG screenshots to social networks
- Push notification using OneSignal service

- Native rating request popup (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.



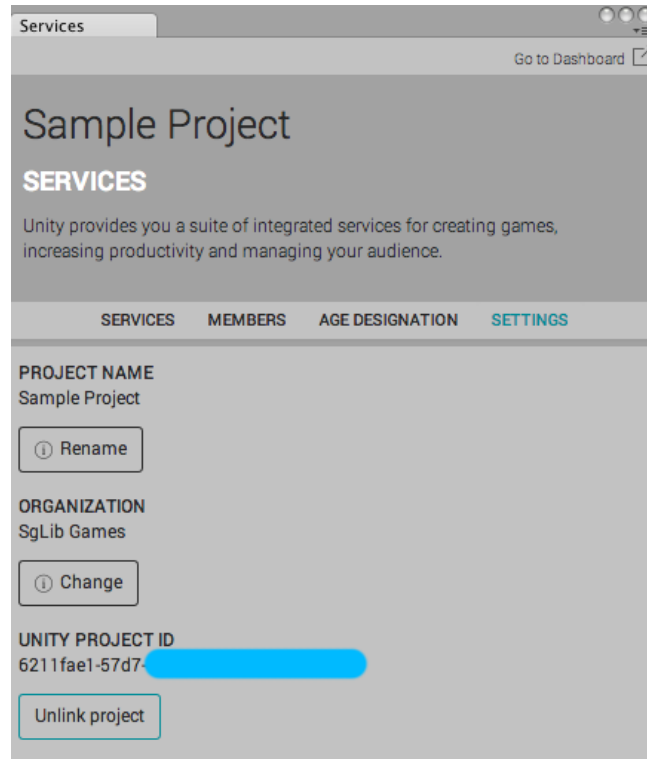
The screenshot shows the Unity Inspector window for a game object named 'AppInfo'. The script 'AppInfo' is selected, and the following fields are visible:

Field Name	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]

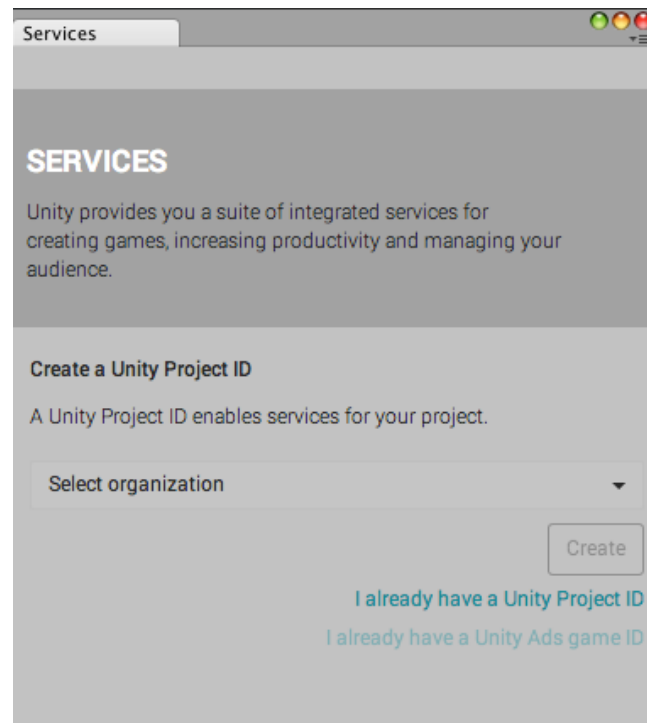
2.2 Link the game to your Unity project

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.

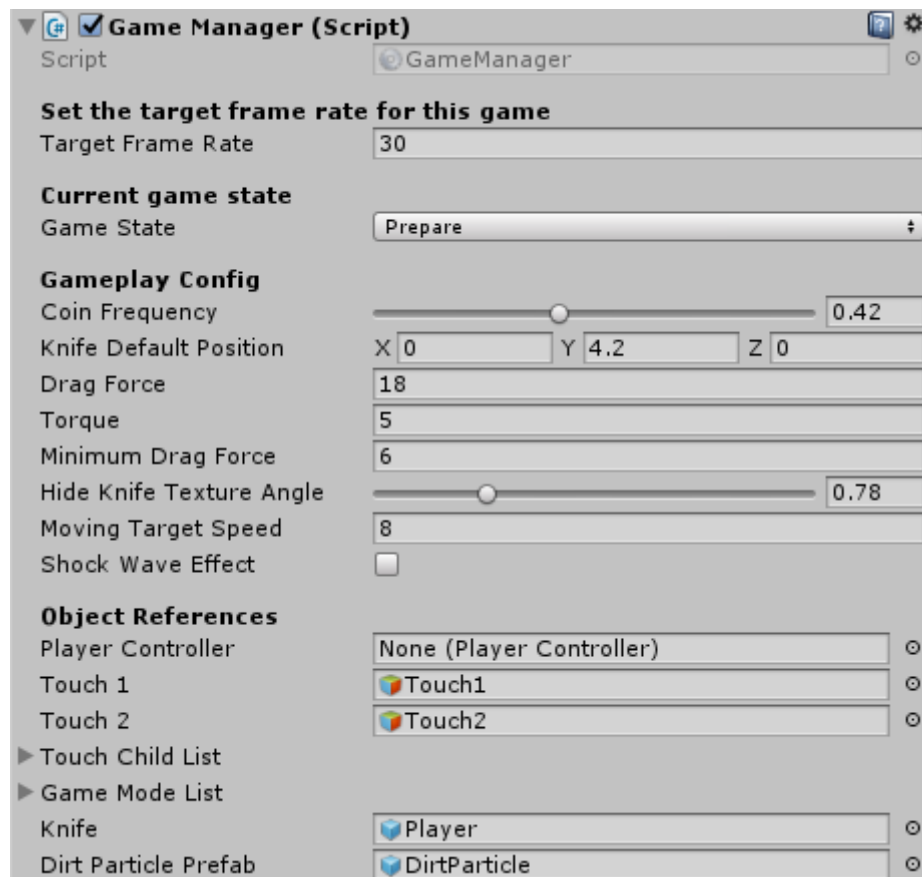
2.3 Testing Note

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

3 TEMPLATE CUSTOMIZATION

3.1 Gameplay tweaking

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.
- *GoldFrequency*: the appearance probability of “gold” (or coin).
- *Knife*: the default knife that will be spawned.

- *KnifeDefaultPosition*: the default position that knives will be spawned at.
- *MovingTargetSpeed*: the speed of moving target in game mode 2.
- *GameMode List*: list of game modes.
- *DragForce*: force to drag knife.
- *Torque*: rotating speed of knives.
- *MinimumDragForce*: The minimum drag force which will scale with drag Force.
- *ShockWaveEffect* : enable the shock wave effect when the knife hit the target and get point.

3.2 Config Shock Wave Effect

Shock wave effect is controlled by shock wave manager. You can find this script attached to camera of each game mode.



Layer Mask: This layer mask is used to get the shock wave normal. You don't need to change this.

Shock wave plane: This is the shock wave plane prefab which contains shock wave shader. You can find this prefab under Assets/_ProjectName/Prefabs/Game

Shock Wave Duration: How long the shock wave should last.

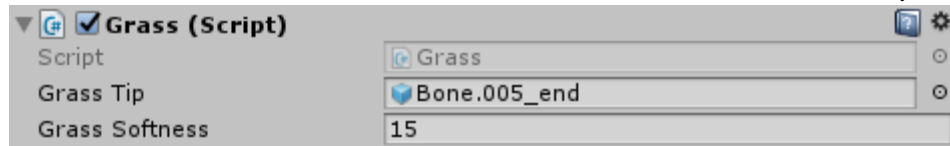
Max Shock Wave Radius: How big the shock wave would be at the end of shock wave effect.

Shock wave color tint: The color of the shock wave. Leave it black if you don't want your shock wave to have any color.

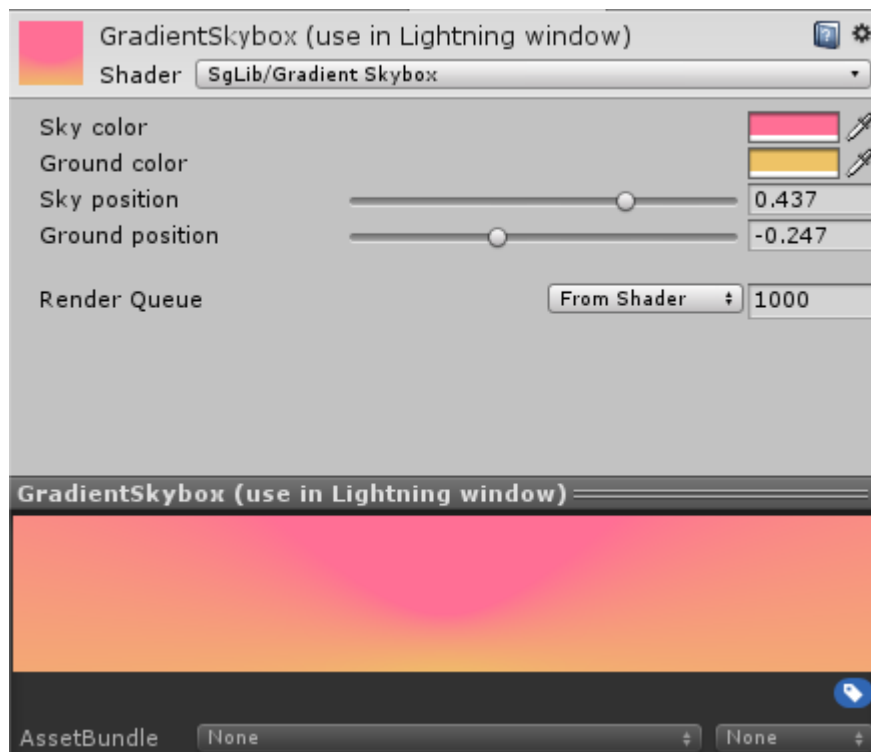
Shock wave offsets from surface amount: Sometimes when the knife sinks into a log or ground, the shock wave created after that will have the position inside the log or the ground. To prevent this offset the shock wave plane base on the ground's or log's normal by this amount.

3.3 Config the grass effect

You can see the grass in scene 1 react to the knife when it hit the target. If you want to control this effect, you can change grass softness in the grass script attached to each grass in the scene. If you want to see more grass in the scene, simply just duplicate the existed one change the configuration if needed the move it to wherever you like.



3.4 Changing skybox color



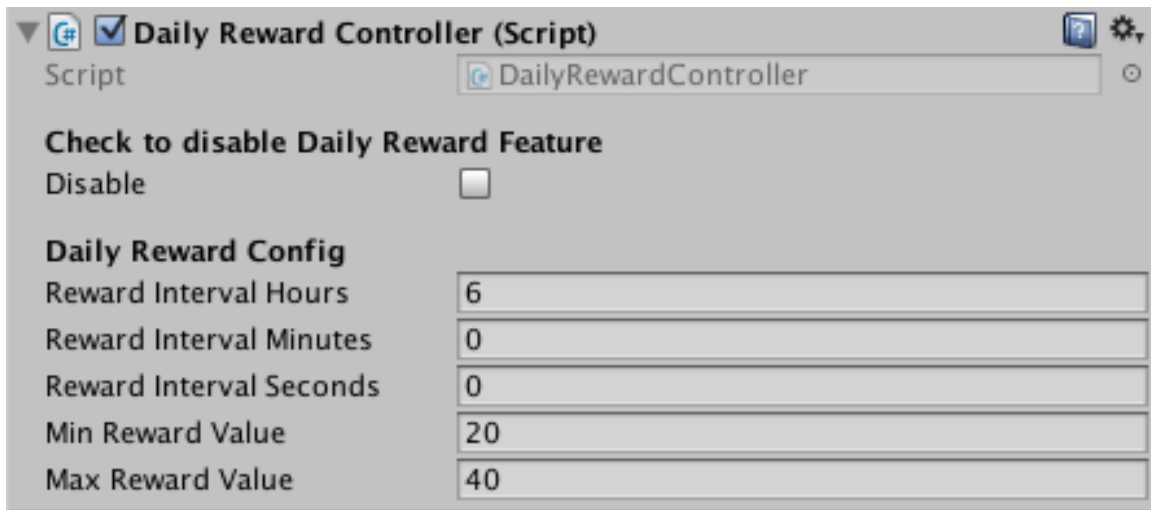
You can find GradientSkybox material under Assets/_ProjectName/Materials folder

Change the values in the inspector to change the skybox color.

3.5 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.

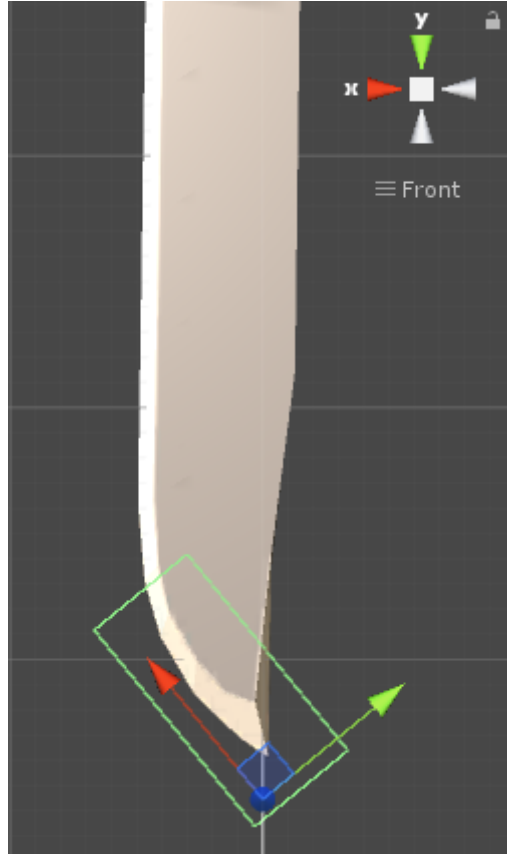


- *Disable*: check to disable this feature
- *Reward Interval Hours, Minutes and Seconds*: the amount of time until the next reward
- *Min Reward Value & Max Reward Value*: the actual rewarded coins will be randomized between these two values

3.6 Adding more knives:

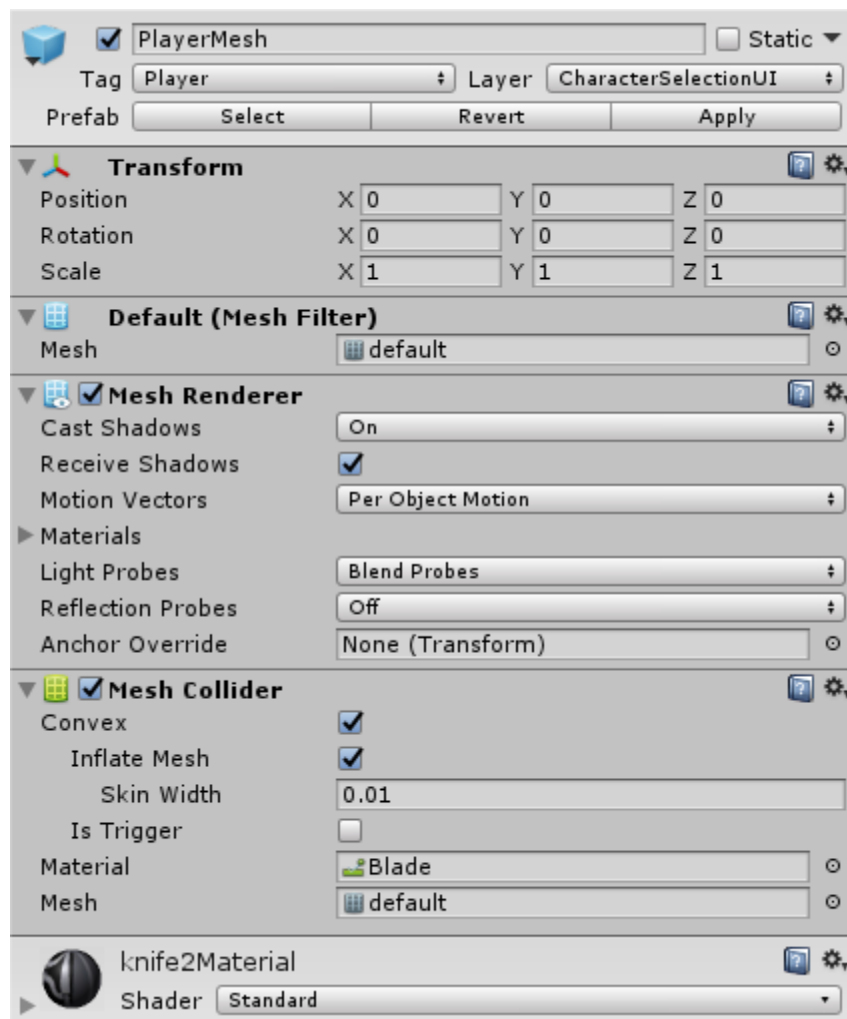
Out-of-the-box, this game is already packed with 30 knives and swords, ready to use! If you want to add more, follow these simple steps:

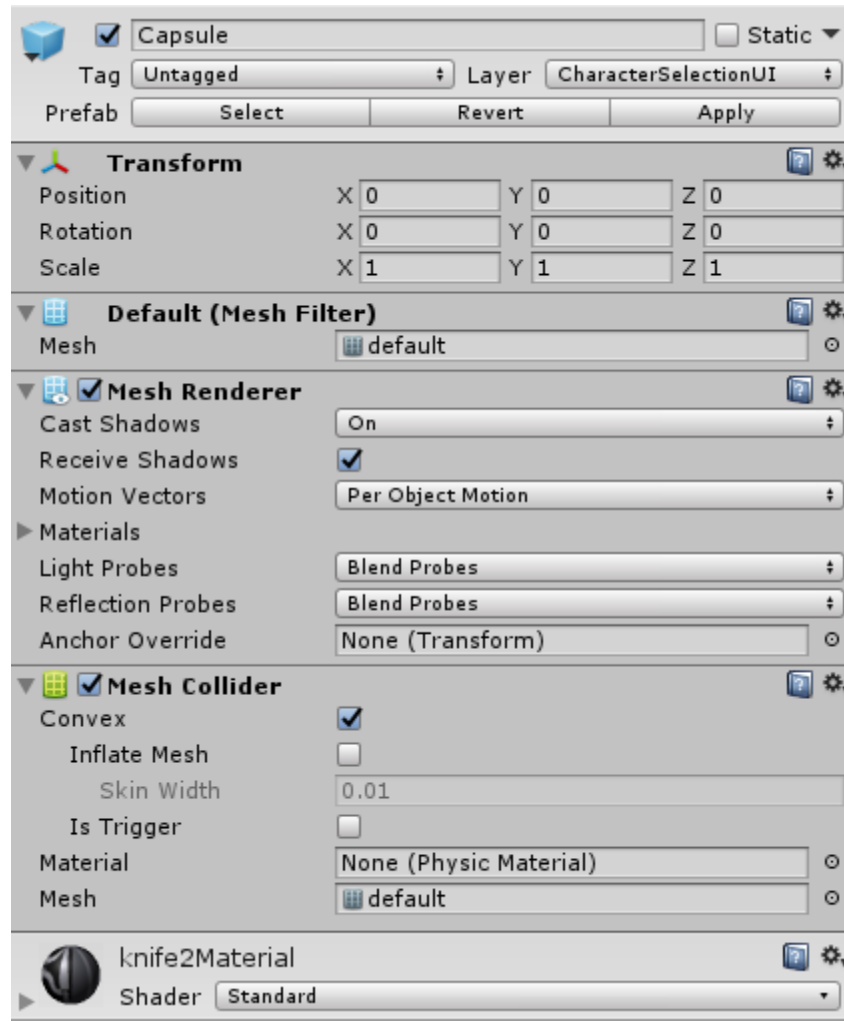
- a. Create a knife or sword model with the pivot at the bottom center with 2 separate mesh (knife blade and knife holder).
- b. Navigate to *Assets/Prefabs/Characters/CharacterPrefabs* and duplicate one of the available character prefabs.
- c. Change the name of the prefab to a preferred one.
- d. Replace the *Mesh* in the *MeshFilter* component with your new model mesh (Knife blade for playerMesh and knife holder for Capsule). Also, remember to replace Mesh colliders with the new meshes.
- e. Replace the *Material* in the *MeshRenderer* component with your new knife material.
- f. Adjust knife head box Collider. Make sure that it covers all of your new knife head.



- g. Enter the knife 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.
- h. Resize the character array in *CharacterManager* game object then drag the new knife to it and hit. Apply to save changes to its prefab.

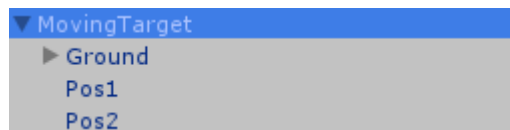






Now the new knife has been added and ready to use in game! You will see it listed in the *CharacterSelection* scene.

3.7 Moving target Config



You can find moving target under
Assets/_ProjectName/Prefabs/Game/Characters folder

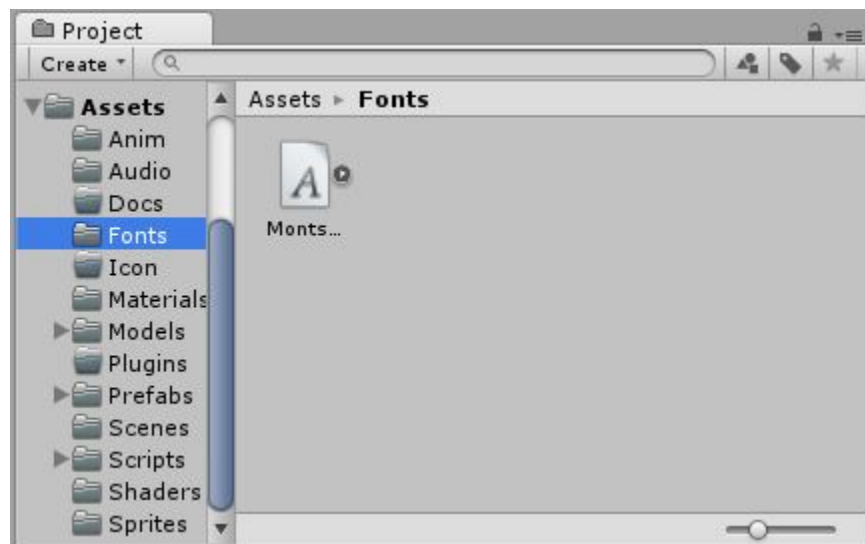
Moving target's position is random between pos1 and pos2 change the position of those two to change the random range of moving target's position

3.8 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.



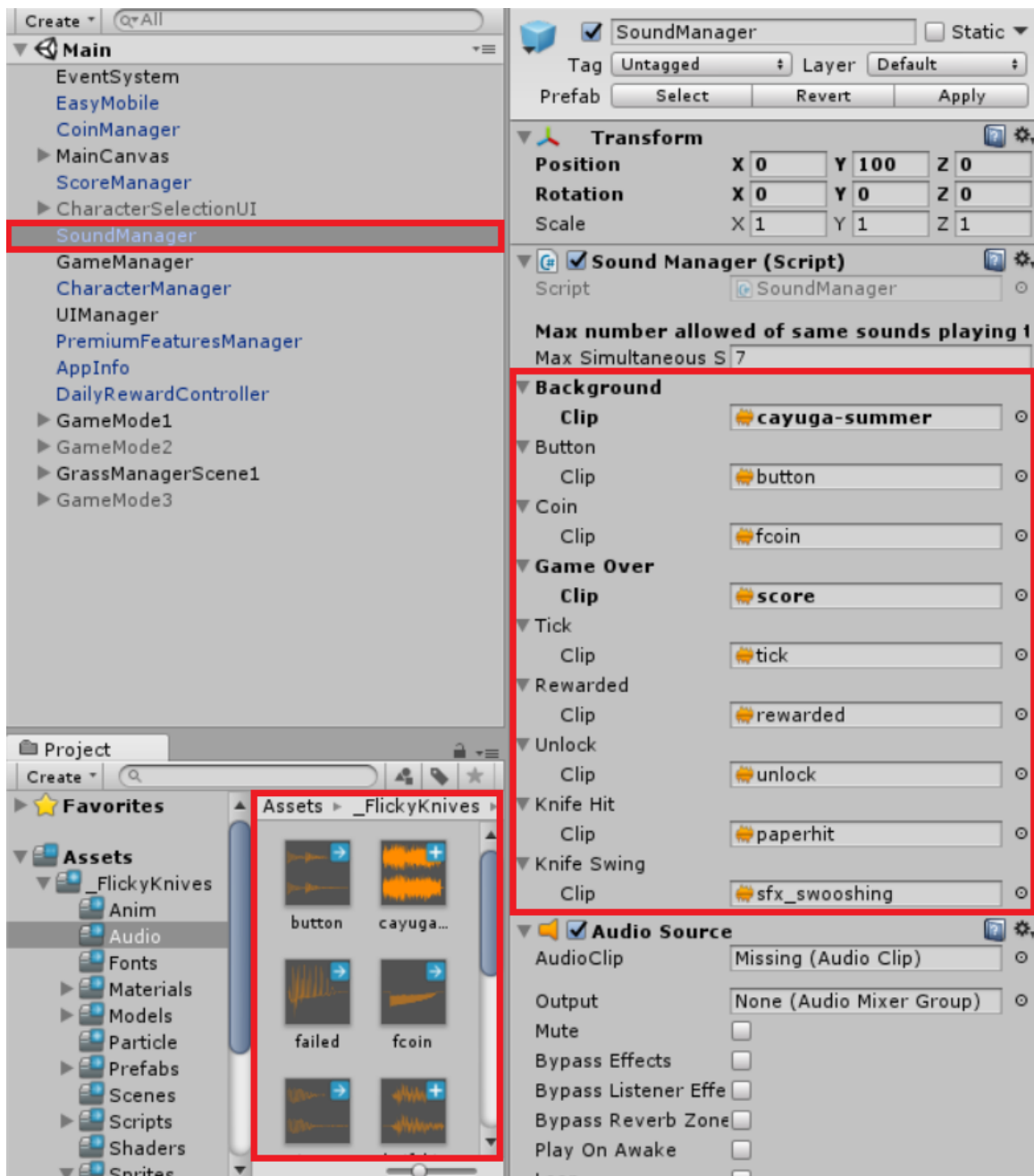
3.9 Sounds

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.



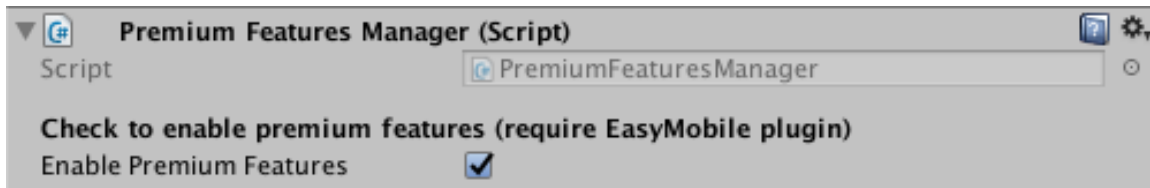
4 ENABLING PREMIUM FEATURES

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.

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.

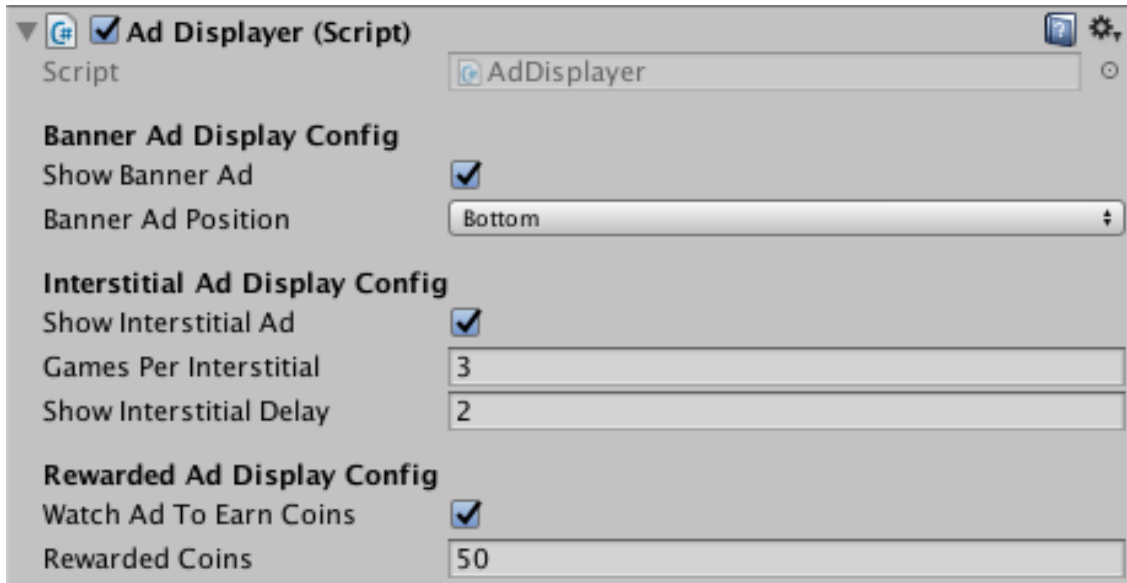


- Make sure to add the EasyMobile prefab to the Main scene, you can find the prefab at folder *Assets/EasyMobile*. It is necessary for the plugin to function properly.
- The settings interface of Easy Mobile can be opened via menu *Window > Easy Mobile > Settings*, this is the only place to go to configure 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 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.

ADVERTISING

ADCOLONY SETUP

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

Download AdColony Plugin

ADMOB SETUP

⚠ Google Mobile Ads (AdMob) plugin was imported.

Download Google Mobile Ads Plugin

AdMob IDs

▶ iOS

▶ Android

Ad Targeting

Gender: Unspecified

Tag For Child Directed Treatment: Unspecified

Extras: 0

Test Mode

Enable Test Mode: ☐

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.

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, including importing the required plugins, 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

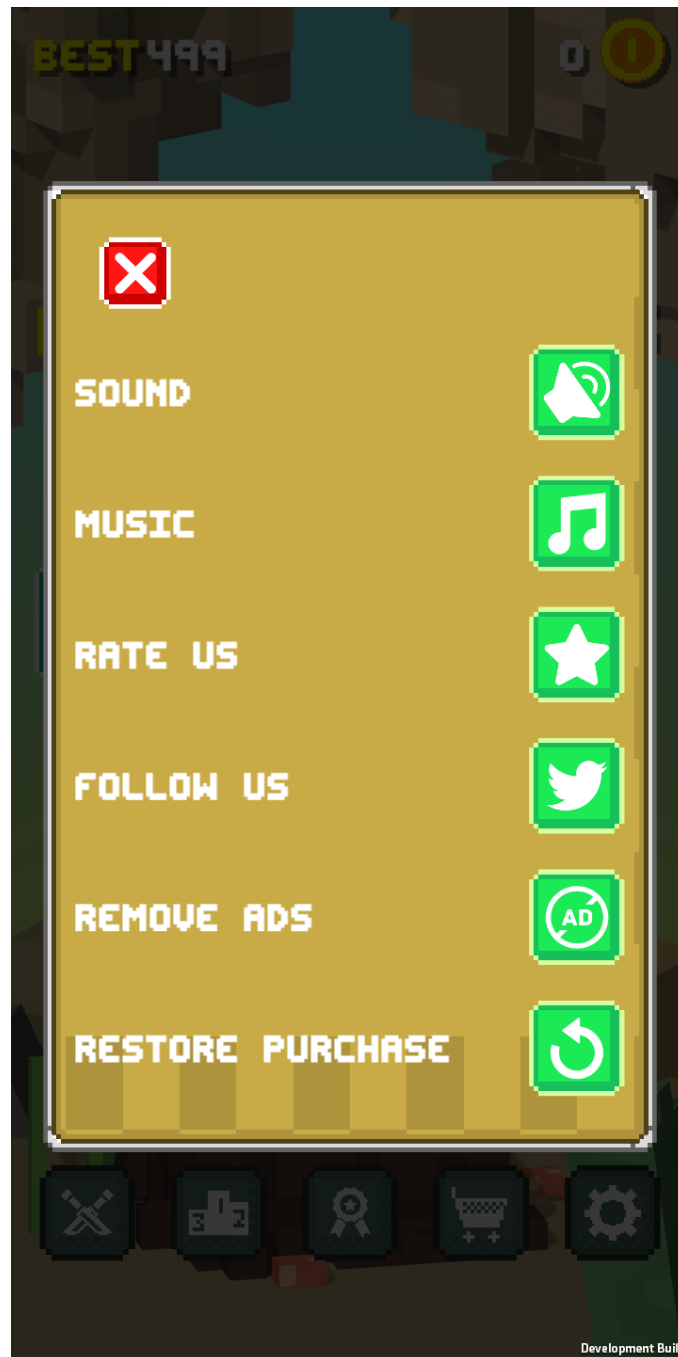
That’s it! Now your game is ready for showing ads!

4.3 In-App Purchasing

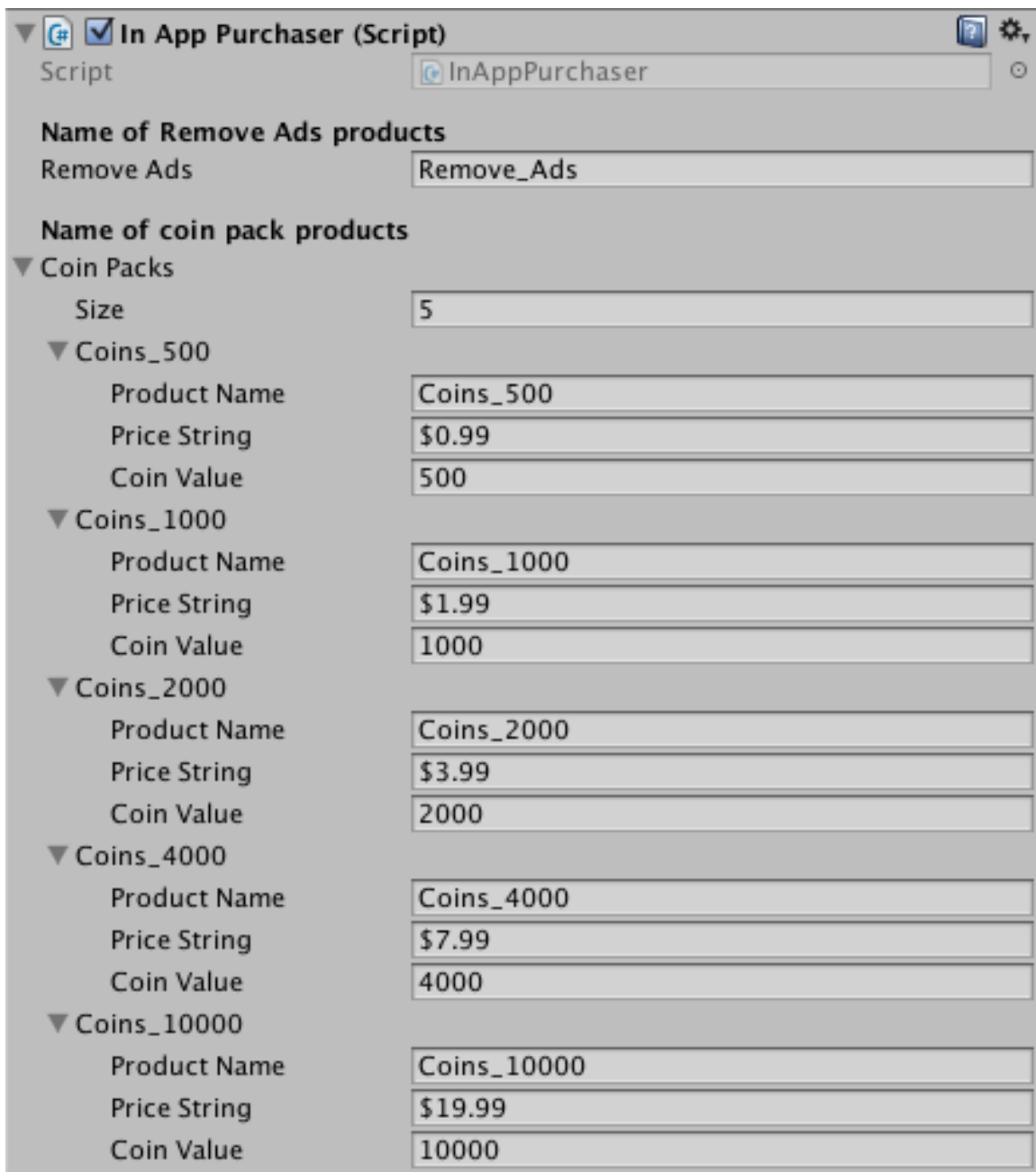
4.3.1 Template-specific setup

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



In App Purchaser (Script)

Script

Name of Remove Ads products

Remove Ads

Name of coin pack products

▼ Coin Packs

Size

▼ Coins_500

Product Name

Price String

Coin Value

▼ Coins_1000

Product Name

Price String

Coin Value

▼ Coins_2000

Product Name

Price String

Coin Value

▼ Coins_4000

Product Name

Price String

Coin Value

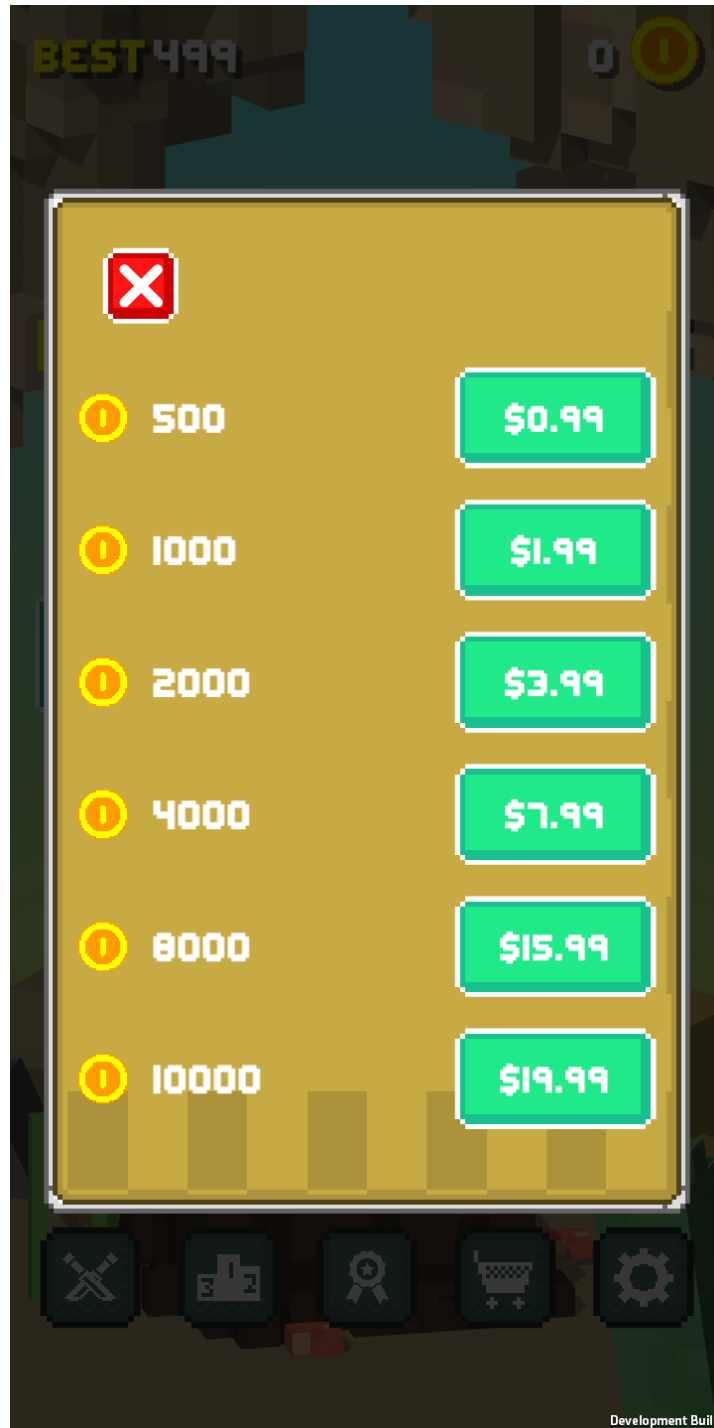
▼ Coins_10000

Product Name

Price String

Coin Value

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.

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 the following 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 the following 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

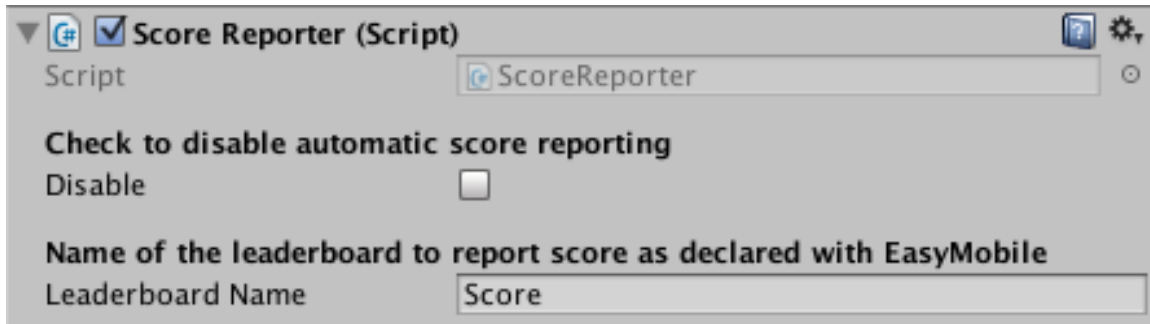
That last step in configuring the in-app purchasing feature is to create products for your targeted stores (e.g. Google Play and Apple App Store). Make sure the product ID, product type and price match the ones you have in your game.

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 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.

▼ ☒ 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.

- Import Google Play Games plugin for Unity and setup it 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[REDACTED]</string>
  <string name="package_name">com.sglib.demogame</string>
  <string name="achievement_score_10">CgkI3tzAhK8eEAIQEg</string>
  <string name="achievement_score_20">CgkI3tzAhK8eEAIQAQ</string>
  <string name="achievement_score_30">CgkI3tzAhK8eEAIQEW</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 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.

The image shows a configuration window for a script named "Screenshot Sharer (Script)". The window has a title bar with a script icon and a settings icon. Below the title bar, there is a "Script" section with a dropdown menu showing "ScreenshotSharer".

The main configuration area is divided into several sections:

- Check to disable sharing:** A checkbox labeled "Disable Sharing" is currently unchecked.
- Sharing Config:**
 - Shared Image Format:** A dropdown menu set to "Both".
 - Share Message:** A text field containing the message: "Awesome! I've just scored [score] in [AppName]! [#AppName]".
- Gif Filename:** A text field containing "animated_screenshot".
- Png Filename:** A text field containing "screenshot".
- GIF Settings:**
 - Gif Auto Height:** A checkbox that is checked.
 - Gif Width:** A text field containing "320".
 - Gif Height:** A text field containing "480".
 - Gif Fps:** A slider control ranging from 0 to 15, currently set at 15.
 - Gif Length:** A slider control ranging from 0 to 3, currently set at 3.
 - Gif Loop:** A text field containing "0".
 - Gif Quality:** A slider control ranging from 0 to 80, currently set at 80.
 - Gif Thread Priority:** A dropdown menu set to "Below Normal".
- Giphy Credentials – leave both empty to use Giphy Beta key:**
 - Giphy Username:** An empty text field.
 - Giphy Api Key:** An empty text field.
 - Giphy Upload Tags:** A text field containing "dashy, worm".

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

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

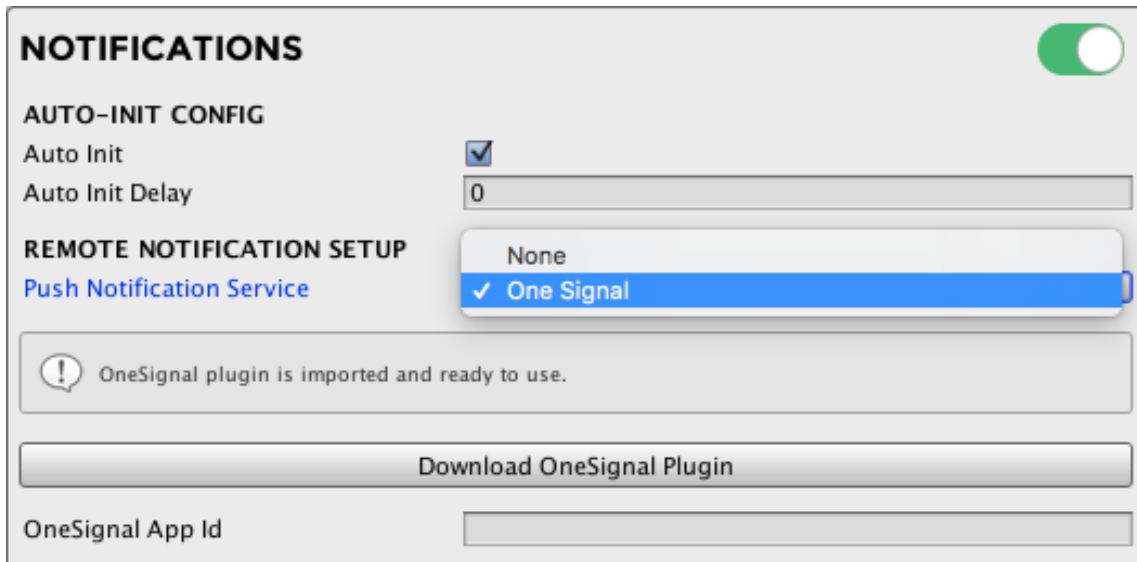
4.7 Push notification

Enabling push notification for your app using OneSignal 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 after importing OneSignal plugin and you can enter your app ID.



The screenshot shows the 'NOTIFICATIONS' settings panel in the Easy Mobile interface. At the top right, there is a green toggle switch that is turned on. Below the title, the 'AUTO-INIT CONFIG' section includes 'Auto Init' with a checked checkbox and 'Auto Init Delay' with a numeric input field set to '0'. The 'REMOTE NOTIFICATION SETUP' section has a 'Push Notification Service' dropdown menu currently open, showing 'None' and 'One Signal' (which is selected with a blue highlight and a checkmark). Below the dropdown is a message box with an exclamation mark icon stating 'OneSignal plugin is imported and ready to use.' Underneath this is a button labeled 'Download OneSignal Plugin'. At the bottom, there is a label 'OneSignal App Id' followed by an empty text input field.

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

THANK YOU AND GOOD LUCK WITH YOUR GAMES!