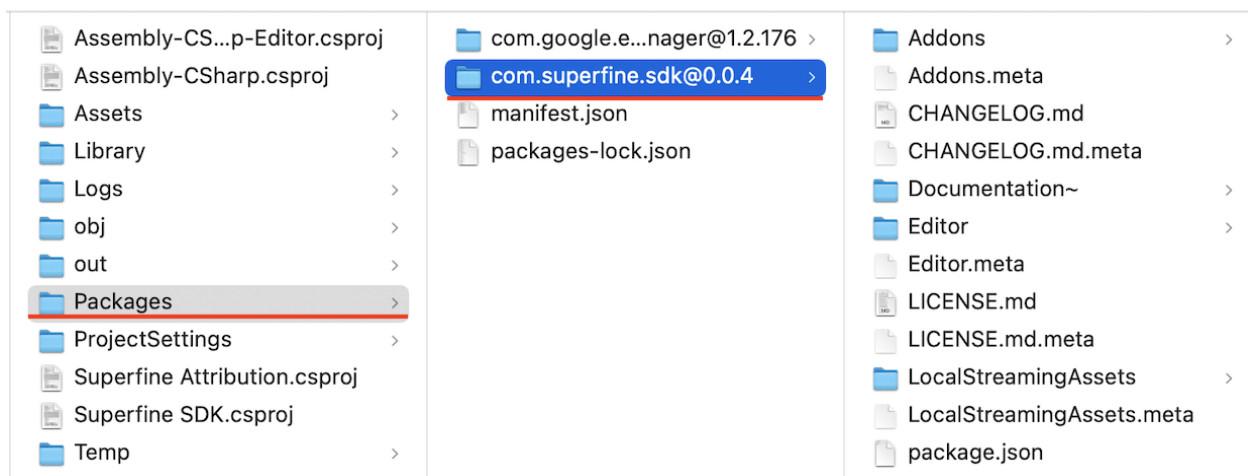# Superfine SDK - Unity

Version 0.0.4

1 Setup

1.1 Compatibility

Superfine SDK is compatible with Unity 2019.4.x and later.

1.2 Import Unity package

To begin using the Superfine SDK, follow these steps:

1. Download the SuperfineSDK ZIP file from [here](https://github.com/gamejamco/superfine-sdk-unity):
    https://github.com/gamejamco/superfine-sdk-unity
2. Unzip the package.
3. Copy the extracted files to your Unity project's Packages folder, located in the root directory of your project.
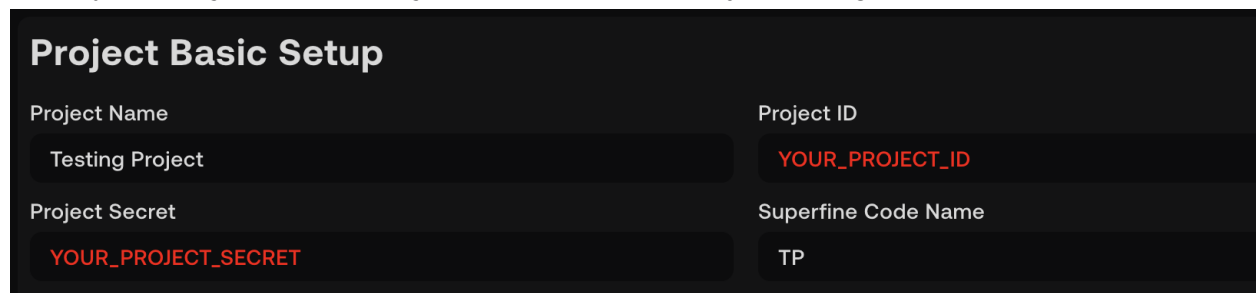
The Superfine SDK includes **JSON.NET** and **external-dependency-manager**, which are necessary for the SDK to function correctly. While we have included them within the SDK package, you have the flexibility to remove or modify them to match the version requirements of your application if needed.
You can access the Package Managers setting by navigating to Window > Package Managers. Alternatively, you can directly edit the package.json file located at Packages/com.superfine.attribution@0.0.4/package.json.

1.3 Get App Information

To integrate your app with the Superfine SDK, obtain the necessary information from the Superfine.org dashboard:

1. Go to the "Projects" section on the Superfine.org dashboard.
2. Select your project or app
3. Copy the **Project ID** and **Project Secret** from the project settings.

1.4 Update Superfine Settings

Configure the Superfine SDK settings in your Unity project:

1. From the Unity menu, go to Superfine > Edit Settings.
2. Update the following settings:
   a. Project ID: Paste the copied Project ID from the Superfine.org dashboard.
   b. Superfine App Secret: Paste the copied Superfine App Secret from the Superfine.org dashboard.
   c. TenjinApiKey for iOS and Android: Specify the Tenjin API Key for iOS and Android platforms if applicable.
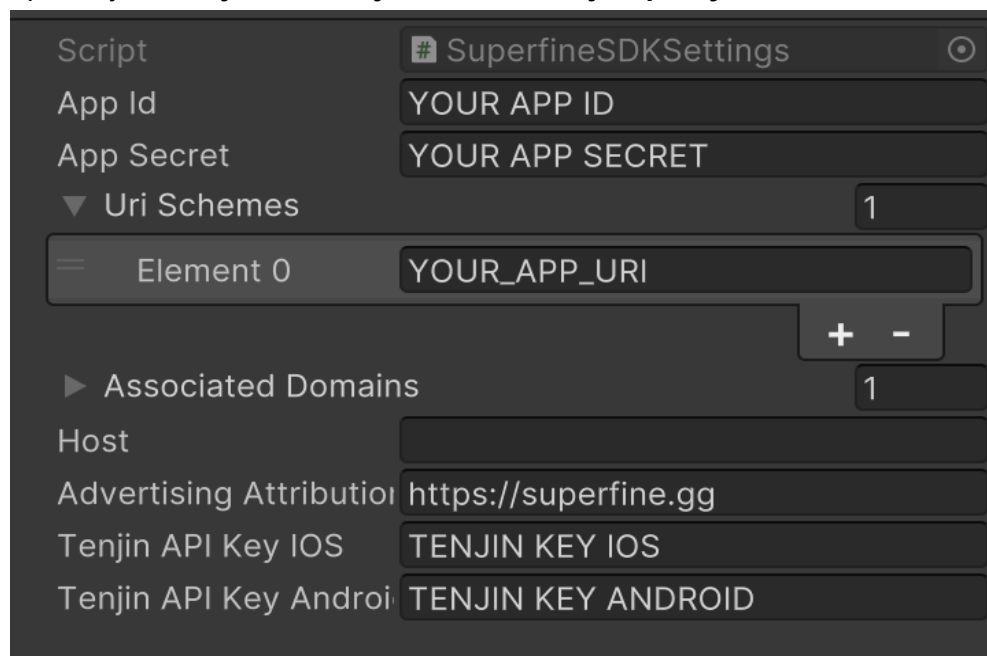
Update your **Project ID**, **Project Secret**, **TenjinApiKey** for **iOS** and **Android**

1.5 Initialize SDK

To initialize the Superfine SDK, add the following code to the Awake function of a new component:

```
void Awake()
{
    SuperfineSDKInitOptions options = new SuperfineSDKInitOptions();

#if !UNITY_EDITOR
#if UNITY_ANDROID
    // Enable VERBOSE (or INFO, DEBUG) logging for Android.
    options.logLevel = LogLevel.VERBOSE;
#elif UNITY_IOS
    // Enable debug mode for iOS
    options.debug = true;
    // Enable capturing in-app purchases for iOS
    options.captureInAppPurchases = true;
#endif
#endif
        // Create an instance of SuperfineSDK
    SuperfineSDK.CreateInstance(options);
}
```

SuperfineSDKInitOptions: Can't be null. Contains:
- flushInterval (**Long**): Time interval (milliseconds) for data flush to the server.
- flushQueueSize (**Integer**): Maximum number of stored events before server flush.
- customUserId (**Boolean**): Flag for using a custom user ID.
- userId (**String**): Custom user identifier to associate events with a user
- waitConfigId (**Boolean**): Flag to wait for the configuration ID before starting.
- autoStart (**Boolean**). Flag to automatically start the SDK on initialization.
- storeType (**StoreType**): Can be UNKNOWN, GOOGLE_PLAY, APP_STORE
- logLevel (**LogLevel**): for Android only. Can be VERBOSE, INFO, DEBUG)
- debug (**Boolean**): for iOS only. If set to true, will display the debug log
- captureInAppPurchases (**Boolean**): for iOS only. Enable capturing in-app purchases for iOS

2 Send Events

2.1 Wallet Events

LogWalletLink

void LogWalletLink(string wallet, string type = "ethereum");

Call this method when you want to link the user's wallet address.

Parameters	
wallet	**String**: Can't be null. The wallet address that you want to log the linking event for
type	**String**: Default is ethereum. The chain of the wallet address

Example:

```
//Link wallet address "ronin:50460c4cd74094cd591455cad457e99c4ab8be0" in
the "ronin" chain
SuperfineSDK.LogWalletLink("ronin:50460c4cd74094cd591455cad457e99c4ab8be0",
"ronin");
```

LogWalletUnlink

void LogWalletUnlink(string wallet, string type = "ethereum");

Call this method when you want to unlink the user's wallet address.

Parameters	
wallet	**String**: Can't be null. The wallet address that you want to log the linking event for
type	**String**: Default is ethereum. The chain of the wallet address

Example:

```
//Unlink wallet address "ronin:50460c4cd74094cd591455cad457e99c4ab8be0" in
"ronin" chain
SuperfineSDK.LogWalletUnlink("ronin:50460c4cd74094cd591455cad457e99c4ab8be0
", "ronin");
```

2.2 Game Level Events

LogLevelStart

void LogLevelStart(int id, string name);

Call this method when starting a level.

Parameters	
id	**Integer**: Can't be null. The level id that you want to log.
name	**String**: Can't be null. The name of the level that you want to log.

Example:

```
// Log start level ID 10 with the name "level_10"
SuperfineSDK.LogLevelStart(10,"level_10");
```

LogLevelEnd

void LogLevelEnd(int id, string name, bool isSuccess);

Call this method when completing a level:

Parameters	
id	**Integer**: Can't be null. The level id that you want to log.
name	**String**: Can't be null. The name of the level that you want to log.
isSuccess	**Boolean**: Can't be null. True if you pass the level. False if you can't pass the level.

Example:

```
//Log you completed level ID 10 with the name "level_10" and you won.
SuperfineSDK.LogLevelEnd(10, "level_10", true);
```

2.3 Ad Events

These events are used to track ads from your app. You can use the Superfine dashboard later to check ad performance based on these events.

LogAdLoad

void LogAdLoad(string adUnit, AdPlacementType adPlacementType, AdPlacement adPlacement = AdPlacement.UNKNOWN);

Call this method when an ad placement is loaded.

Parameters	
adUnit	**String**: Can't be null. The ad unit that you want to log.
adPlacementType	**enum AdPlacementType**: Can't be null. Can be: - BANNER = 0 - INTERSTITIAL = 1 - REWARDED_VIDEO = 2
adPlacement	**enum AdPlacement**: Can't be null. Can be: - UNKNOWN = -1 - BOTTOM = 0 - TOP = 1 - LEFT = 2 - RIGHT = 3 - FULL_SCREEN = 4

Example:

```
//Log ad unit "ad_unit_test" with ad placement type is "INTERSTITIAL" at
placement "FULL_SCREEN" loaded
SuperfineSDK.LogAdLoad("ad_unit_test", AdPlacementType.INTERSTITIAL,
AdPlacement.FULL_SCREEN);
```

LogAdClosed

void LogAdClosed(string adUnit, AdPlacementType adPlacementType, AdPlacement adPlacement = AdPlacement.UNKNOWN);

Call this method when an ad is closed.

Parameters	
adUnit	**String**: Can't be null. The ad unit that you want to log.
adPlacementType	**enum AdPlacementType**: Can't be null. Can be: - BANNER = 0 - INTERSTITIAL = 1 - REWARDED_VIDEO = 2
adPlacement	**enum AdPlacement**: Can't be null. Can be: - UNKNOWN = -1 - BOTTOM = 0 - TOP = 1 - LEFT = 2 - RIGHT = 3 - FULL_SCREEN = 4

Example:

```
//Log ad unit "ad_unit_test" with ad placement type is "INTERSTITIAL" at
placement "FULL_SCREEN" closed
SuperfineSDK.LogAdClosed("ad_unit_test", AdPlacementType.INTERSTITIAL,
AdPlacement.FULL_SCREEN);
```

LogAdImpression

void LogAdImpression(string adUnit, AdPlacementType adPlacementType, AdPlacement adPlacement = AdPlacement.UNKNOWN);

Call this method when the ad impression is displayed.

Parameters	
adUnit	**String**: Can't be null. The ad unit that you want to log.
adPlacementType	**enum AdPlacementType**: Can't be null. Can be: - BANNER = 0 - INTERSTITIAL = 1 - REWARDED_VIDEO = 2
adPlacement	**enum AdPlacement**: Can't be null. Can be: - UNKNOWN = -1 - BOTTOM = 0 - TOP = 1 - LEFT = 2 - RIGHT = 3 - FULL_SCREEN = 4

Example:

```
//Log ad unit "ad_unit_test" with ad placement type is "INTERSTITIAL" at
placement "FULL_SCREEN" Impression
SuperfineSDK.LogAdImpression("ad_unit_test", AdPlacementType.INTERSTITIAL,
AdPlacement.FULL_SCREEN);
```

LogAdClick

void LogAdClick(string adUnit, AdPlacementType adPlacementType, AdPlacement adPlacement = AdPlacement.UNKNOWN);

Call this method when the user clicks on an ad.

Parameters	
adUnit	**String**: Can't be null. The ad unit that you want to log.

adPlacementType	**enum AdPlacementType**: Can't be null. Can be:
	  - BANNER = 0
	  - INTERSTITIAL = 1
	  - REWARDED_VIDEO = 2

adPlacement	**enum AdPlacement**: Can't be null. Can be:
	  - UNKNOWN = -1
	  - BOTTOM = 0
	  - TOP = 1
	  - LEFT = 2
	  - RIGHT = 3
	  - FULL_SCREEN = 4

Example:

```
//Log ad unit "ad_unit_test" with ad placement type is "INTERSTITIAL" at
placement "FULL_SCREEN" clicked.
SuperfineSDK.LogAdClick("ad_unit_test", AdPlacementType.INTERSTITIAL,
AdPlacement.FULL_SCREEN);
```

LogAdRevenue

void LogAdRevenue(string network, double revenue, string currency, string mediation = "", SimpleJSON.JSONObject networkData = null)

Call this method to record revenue obtained from an advertisement.

Parameters	
network	**String**: Can't be null. The ad-network that generated the revenue.
revenue	**double**: Can't be null. The amount of revenue obtained from the ad-network.
currency	**String**: The currency code (e.g., "USD", "EUR") corresponding to the revenue.
mediation	**String:** Can be empty. The mediation platform used (e.g., Direct, Max, LevelPlay, etc.).
networkData	**SimpleJSON.JSONObject:** Can be null. Additional information about the ad-network.

Example:

```
SuperfineSDK.LogAdRevenue(testAdNetworkId, 1.0, "USD", "DIRECT", new
Superfine.Unity.SimpleJSON.JSONObject
        {
            { "param1", "abc" },
            { "param2", 100 },
            { "param3", true }
        }
);
```

2.4 IAP Events

These events are used to track in-app purchases from your app.

LogIAPBuyStart

void LogIAPBuyStart(string pack, float price, int amount, string currency);

Call this method when the user attempts to buy an IAP item.

Parameters	
pack	**String**: Can't be null. The unique identifier of the purchased item or pack.
price	**Float**: Can't be null. The price of the item or pack.
amount	**Integer**: Can't be null. The quantity of the purchased item or pack.
currency	**String**: Can't be null. The currency code (e.g., "USD", "EUR") for the price.

Example:

```
//Log when the user attempts to purchase a package with ID "test_pack" for
0.99 USD, getting 150 units.
SuperfineSDK.LogIAPBuyStart("test_pack", 0.99, 150, "USD");
```

LogIAPBuyEnd

void LogIAPBuyEnd(string pack, float price, int amount, string currency);

Call this method when the IAP purchase process is completed.

Parameters	
pack	**String**: Can't be null. The unique identifier of the purchased item or pack.
price	**Float**: Can't be null. The price of the item or pack.
amount	**Integer**: Can't be null. The quantity of the purchased item or pack.
currency	**String**: Can't be null. The currency code (e.g., "USD", "EUR") for the price.

Example:

```
//Log when the user completed to purchase a package with ID "test_pack" for
0.99 USD, getting 150 units.
SuperfineSDK.LogIAPBuyEnd("test_pack", 0.99, 150, "USD");
```

LogAPRestorePurchase

void LogIAPRestorePurchase();

Call this method when restoring a purchase for the iOS devices.

Example:

```
SuperfineSDK.LogIAPRestorePurchase();
```

2.5 Custom Events

You can define any custom event to fit your needs.

Log

void Log(string eventName, int data);

Call this method to log a custom event with integer data.

Parameters	
eventName	**String**: Can't be null. The name of your custom event.
data	**Int**: Can't be null. The integer data associated with the event.

void Log(string eventName, string data);

Call this method to log a custom event with string data.

Parameters	
eventName	**String**: Can't be null. The name of your custom event.
data	**String**: Can't be null. The string data associated with the event.

void Log(string eventName, Dictionary<string, string> data = null);

Call this method to log a custom event with dictionary data.

Parameters	
eventName	**String**: Can't be null. The name of your custom event.
data	**Dictionary<string, string>:** Default is null. The dictionary data associated with the event.

void Log(string eventName, SimpleJSON.JSONObject data = null);

Call this method to log a custom event with JSON object data.

Parameters	
eventName	**String**: Can't be null. The name of your custom event.
data	**SimpleJSON.JSONObject:** Default is null. The JSON object data associated with the event.

Example:

```
// Log a custom event with JSON object data
SimpleJSON.JSONObject eventData = new SimpleJSON.JSONObject();
eventData["score"] = 1000;
eventData["level"] = "beginner";
//Log your event
SuperfineSDK.Log("My_Custom_Event_Name", eventData);
```

2.6 Addons

Ads reporting helper class

We offer ad revenue reporting support through our addon classes. Automatically receive detailed reports by implementing the appropriate class based on your chosen mediation platform and registering for events. Currently, we provide support for the following mediations: Max Mediation (AppLovin), IronSource Mediation, and Google AdMob Mediation.

Applovin Addons Helper Class

Integration: Add the AppLovin Helper Addons to your project by going to the Superfine > Copy AppLovin Addon.
Event Registration: Begin logging revenue and impressions by calling `SuperfineSDKApplovin.RegisterPaidEvent()`. When you're done, turn it off with `SuperfineSDKApplovin.UnregisterPaidEvent()` or when your manager class is destroyed.

IronSource Addons Helper Class

Integration: Add the Ironsource Helper Addons to your project by going to the Superfine > Copy IronSource Addon.
Event Registration: Begin logging revenue and impressions with `SuperfineSDKIronSource.RegisterPaidEvent()`. Turn it off with `SuperfineSDKIronSource.UnregisterPaidEvent()` when done or when your manager class is removed.

Google AdMob Addon Helper Class

Integration: Add the Admob Helper Addons to your project by going to the Superfine > Copy Admob Addon.
Event Registration: For Google Admob you have to register events for all placement that you have.
   - For Banner: `SuperfineSDKAdMob.RegisterBannerViewPaidEvent(bannerView, adUnitId)` and `SuperfineSDKAdMob.UnregisterBannerViewPaidEvent(bannerView, adUnitId)`.
   - For Interstitial: `SuperfineSDKAdMob.RegisterInterstitialAdPaidEvent(interstitialAd, adUnitId)` and `SuperfineSDKAdMob.UnregisterInterstitialAdPaidEvent(interstitialAd, adUnitId)`.
   - For Rewarded video ad: `SuperfineSDKAdMob.RegisterRewardedAdPaidEvent(rewardedAd, adUnitId)` and `SuperfineSDKAdMob.UnregisterRewardedAdPaidEvent(rewardedAd, adUnitId)`.
   - For Rewarded Interstitial Ads `SuperfineSDKAdMob.RegisterRewardedInterstitialAdPaidEvent(rewardedInterstitialAd, adUnitId)` and `SuperfineSDKAdMob.UnregisterRewardedInterstitialAdPaidEvent(rewardedInterstitialAd, adUnitId)`.

- For App `Open SuperfineSDKAdMob.RegisterAppOpenAdPaidEvent(appOpenAd, adUnitId)` and `SuperfineSDKAdMob.UnregisterAppOpenAdPaidEvent(appOpenAd, adUnitId)`.
- For the Ad revenue `SuperfineSDK.LogAdRevenue()`.

Facebook Events Helper Class

The SuperfineSDKFacebook class simplifies sending events to Facebook for marketing purposes. To smoothly integrate this feature, follow these steps:

Integration: Add the Facebook Helper Addons to your project by going to the Superfine > Copy Facebook Addon.
Event Registration: Begin logging events with `SuperfineSDKFacebook.RegisterSendEvent()`. When done, turn it off using `SuperfineSDKFacebook.UnregisterSendEvent()` or when your manager class is removed.
By following these instructions, you can effectively utilize the SuperfineSDKFacebook class to transmit custom events to Facebook, enhancing marketing insights and decision-making for your app.

2.7 Postback Conversion Value for iOS

The method allows you to update both the conversion value and coarse conversion values, and it provides the option to send the postback before the conversion window ends. Additionally, it allows you to specify a completion handler to handle situations where the update fails.

```csharp
using Superfine.Tracking.Unity;

// Example 1: Basic usage without coarseValue or lockWindow
SuperfineSDK.UpdatePostbackConversionValue(10);

// Example 2: Usage with coarseValue
SuperfineSDK.UpdatePostbackConversionValue(8, "low");

// Example 3: Usage with lockWindow
SuperfineSDK.UpdatePostbackConversionValue(15, "medium", true);
```