



**EASY IAP (IN APP PURCHASE)**



## 1. WHY DO YOU NEED TO USE THIS PLUGIN

---

- ✓ Make in app purchases with minimal setup and very little programming knowledge.
- ✓ Same code for all supported platforms.
- ✓ Just import Easy IAP and add your product ID's and all is done.
- ✓ Can be tested using Unity Editor.
- ✓ Support for Consumable. Non-Consumable and Subscriptions.
- ✓ Works with Unity 5.3 and above and with Free, Plus or Pro versions of Unity.



## **2. CURRENTLY SUPPORTED PLATFORMS**

---

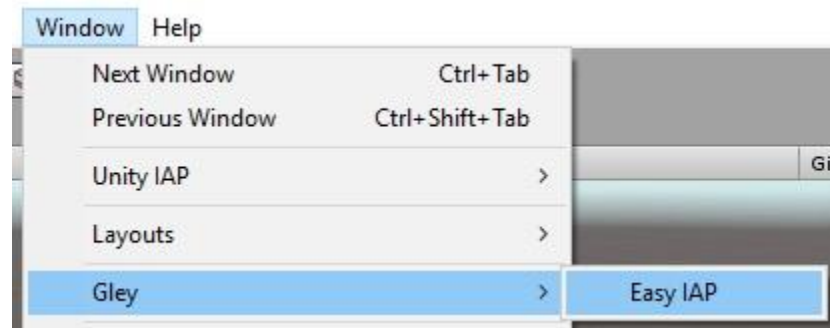
- **Android**
- **iOS**
- **More to come very soon**



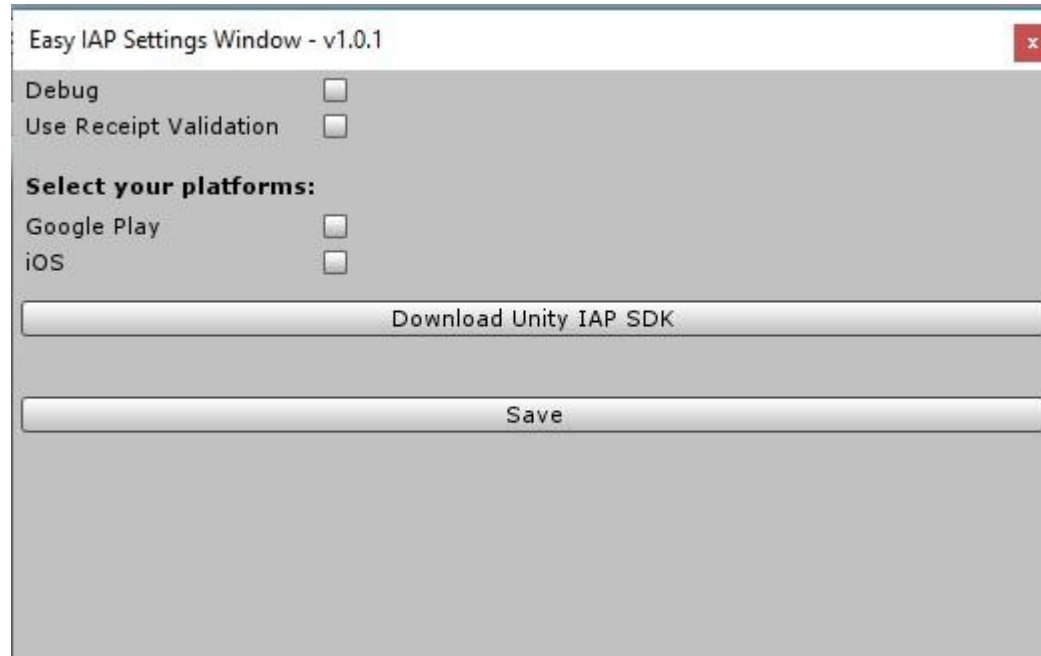
### 3. SETUP GUIDE

---

- Import **Gley Easy IAP Plugin** into Unity.
- Go to **Window->Gley->Easy IAP** to open the plugin settings window.



- Settings Window will open

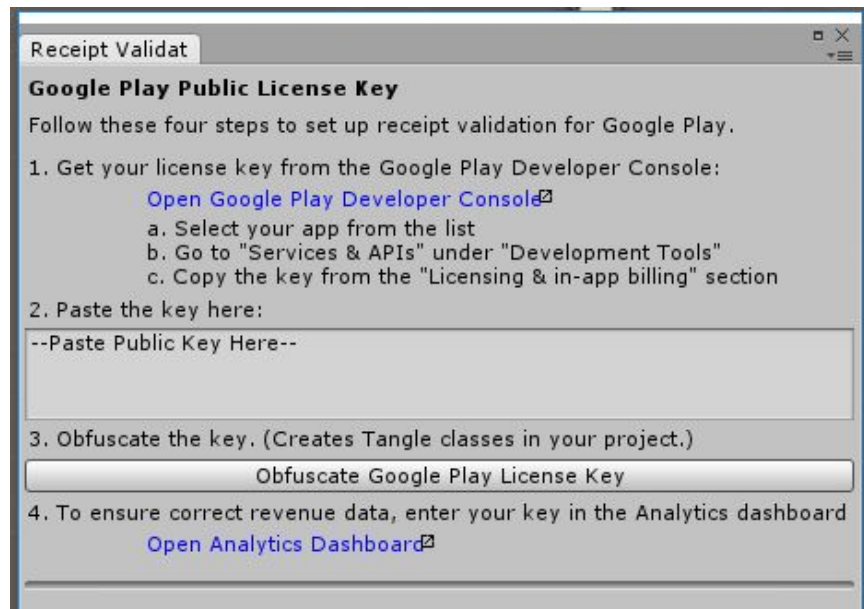


- Enable **Debug** to see debug messages on your device.



## Receipt Validation

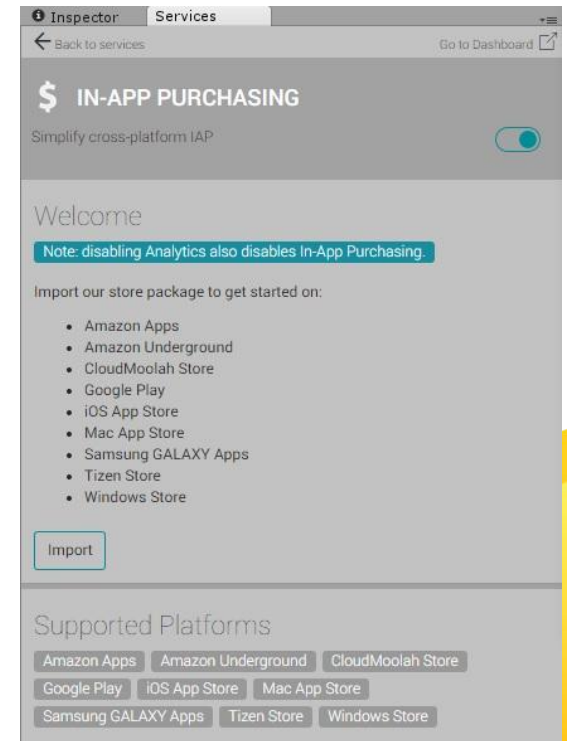
1. Receipt validation helps you prevent users from accessing content they have not purchased. To enable it check the **Use Receipt Validation** checkbox.
2. To setup validation key:
  - a. Go to **Window > Unity IAP > IAP Receipt Validation Obfuscator**
  - b. Paste your GooglePlay public key (from the application's [Google Play Developer Console's Services & APIs](#) page).
  - c. Click **Obfuscate Google Play Licence Key**.





## Download Unity IAP sdk

- The Unity IAP sdk can be downloaded using one of the options listed below :
  - a. Press **Download Unity IAP SDK** button from Settings Window
  - b. From Asset Store following this link:  
<https://assetstore.unity.com/packages/add-ons/services/billing/unity-iap-68207>
  - c. Enable Unity IAP from Unity Services (Recommended)  
<https://docs.unity3d.com/Manual/UnityIAPSettingUp.html>





## Add in app products

- Select platforms to use
- Press **Add New Product** button to add a product
- Press **Remove Product** to remove a product

Easy IAP Settings Window - v1.0.1

Debug ☒  
Use Receipt Validation ☐

**Select your platforms:**  
Google Play ☒  
iOS ☒

Download Unity IAP SDK

**In App Products Setup**

Consumable1

Product Name: Consumable1  
Product Type: Consumable  
Reward Value: 100  
Google Play ID: com.gley.chickenjump.consumable1  
App Store (iOS) ID: com.gley.chickenjump.consumable1

Remove Product

Add new product

Save





## Product details

---

- **Product Name**
  - Name of your product, it will be used in your code to make a purchase.
  - Name cannot contain white spaces, or special characters and cannot start with number.
- **Product Type**
  - Consumable - this product will consume after it is bought (ex: pack of in game currency).
  - Non Consumable - this product is available for lifetime (ex: remove ads, unlock a level, a skin for player, etc).
  - Subscription - usually is auto renewable and it will be recharged until user cancels it. Used for premium content (ex: access multiplayer section, get access to special level, etc).
- **Reward value**
  - For consumable - value of the reward given to the user(ex: number of in game coins).
  - For non-consumable and subscription - does not have any meaning, should be set to 0.



## Product Details

- **Google Play ID**
  - The id of the product from Google Play developer console.

MANAGED PRODUCTS ?		SUBSCRIPTIONS ?	
4 active, 0 inactive		1 active, 0 inactive	
<input type="text" value="Search by product name or ID"/>		Import / Export ▼	CREATE MANAGED PRODUCT
Name & ID	Price	Last updated	Status
Consumable 1 (com.gley.chickenjump.consumable1)	RON 3.50	Jun 5, 2018	Active
Consumable 2 (com.gley.chickenjump.consumable2)	RON 3.60	Jun 10, 2018	Active
Unlock Level 1 (com.gley.chickenjump.nonconsumable1)	RON 3.70	Jun 10, 2018	Active
Unlock Level 2 (com.gley.chickenjump.nonconsumable2)	RON 3.80	Jun 10, 2018	Active



## Product Details

- **App Store (iOS) ID**
  - The id of the product from iTunes Connect console.

In-App Purchases (5) [+](#)

[Q Search](#)

Reference Name ^	Type	Product ID
<a href="#">Consumable1</a>	Consumable	com.gley.chickenjump.consumable1
<a href="#">Consumable2</a>	Consumable	com.gley.chickenjump.consumable2
<a href="#">Subscription</a>	Non-Renewing Subscription	com.gley.chickenjump.subscription1
<a href="#">UnlockLevel1</a>	Non-Consumable	com.gley.chickenjump.nonconsumable1
<a href="#">UnlockLevel2</a>	Non-Consumable	com.gley.chickenjump.nonconsumable2



## 4. USER GUIDE

---

### Initialize Plugin

//call this method only **once** at the beginning of the game to initialize product information.

**IAPManager.Instance.InitializeIAPManager(InitializeResultCallback);**

**InitializeResultCallback** -> callback method described below



## Initialize Callback example method

---

//this method will be called after initialization process is done

```
private void InitializeResultCallback(IAPOperationStatus status, string message, List<StoreProduct>  
shopProducts)
```

```
{
```

```
    if (status == IAPOperationStatus.Success)
```

```
    {
```

```
        //IAP was successfully initialized
```

```
        //loop through all products
```

```
        for (int i = 0; i < shopProducts.Count; i++)
```

```
        {
```

```
            if (shopProducts[i].productName == "YourProductName")
```

```
            {
```

```
                //if active variable is true, means that user had bought that product
```

```
                //so enable access
```

```
                if (shopProducts[i].active)
```

```
                {
```

```
                    yourBoolVariable = true;
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

```
    else
```

```
    {
```

```
        Debug.Log("Error occurred " + message);
```

```
    }
```

```
}
```



## Buy Product

---

//call this method to make a purchase

**IAPManager.Instance.BuyProduct(ShopProductNames.YourProductName, ProductBoughtCallback);**

// automatically called after one product is bought

// this is an example of product bought callback

**private void ProductBoughtCallback(IAPOperationStatus status, string message, StoreProduct product)**

**{**

**if (status == IAPOperationStatus.Success)**

**{**

//each consumable gives coins in this example

**if (product.productType == ProductType.Consumable)**

**coins += product.value;**

//non consumable Unlock Level 1 -> unlocks level 1 so we set the corresponding bool to true

**if (product.productName == "UnlockLevel1")**

**unlockLevel1 = true;**

//subscription has been bought so we set our subscription variable to true

**if (product.productName == "Subscription")**

**subscription = true;**

**}else**

**{**

//an error occurred in the buy process, log the message for more details

**Debug.Log("Buy product failed: " + message);**

**}**

**}**



## Useful methods

---

//Check if plugin is initialized

**public bool IsInitialized()**

//Get the amount of in game currency received for a product (use this to display the reward)

**public int GetValue(ShopProductNames product)**

// Get the price and currency code of the product as a string (use this to display the price)

**public string GetLocalizedPriceString(ShopProductNames product)**

//Get product price denominated in the local currency

**public int GetPrice(ShopProductNames product)**

// Get product currency in ISO 4217 format; e.g. GBP or USD.

**public string GetIsoCurrencyCode(ShopProductNames product)**

// Get description from the store

**public string GetLocalizedDescription(ShopProductNames product)**

// Get title from the store

**public string GetLocalizedTitle(ShopProductNames product)**

// Gets the status of the product -> if returns true the product was bought by user

**public bool IsActive(ShopProductNames product)**



## Restore Purchases

---

//only required for iOS App Store

//restores previously bought products

//this is also done automatically every time at initialize

**IAPManager.Instance.RestorePurchases(ProductRestoredCallback);**

// automatically called after one product is restore, is the same with Buy Product callback

// this is an example of product restored callback

**private void ProductRestoredCallback(IAPOperationStatus status, string message, StoreProduct product)**

**{**

**if (status == IAPOperationStatus.Success)**

**{**

//consumable products are not restored

//non consumable Unlock Level 1 -> unlocks level 1 so we set the corresponding bool to true

**if (product.productName == "UnlockLevel1")**

**unlockLevel1 = true;**

//subscription has been bought so we set our subscription variable to true

**if (product.productName == "Subscription")**

**subscription = true;**

**}else**

**{**

//an error occurred in the buy process, log the message for more details

**Debug.Log("Buy product failed: " + message);**

**}**

**}**

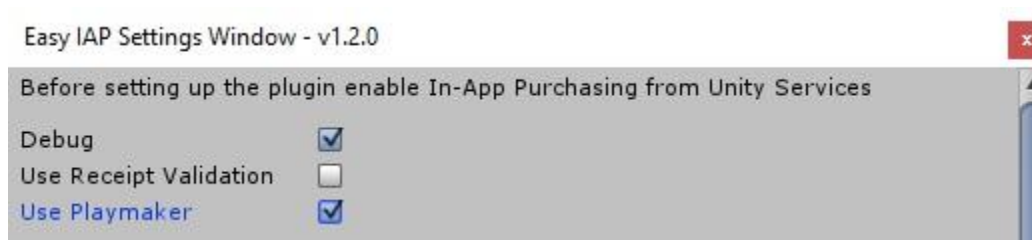




## Playmaker Support

---

- To be able to use Playmaker actions, **Use Playmaker** must be enabled from Settings Window:



- Playmaker supports the following actions:
  - **Initialize IAP**
  - **Buy Product**
  - **Check If Bought**
  - **Get Product Value**
  - **Get Store Price**
  - **Restore Purchases**
- A step by step integration tutorial is available on our Youtube channel:  
<https://youtu.be/sP8kIDq6Ogk>



## 5. EXAMPLE

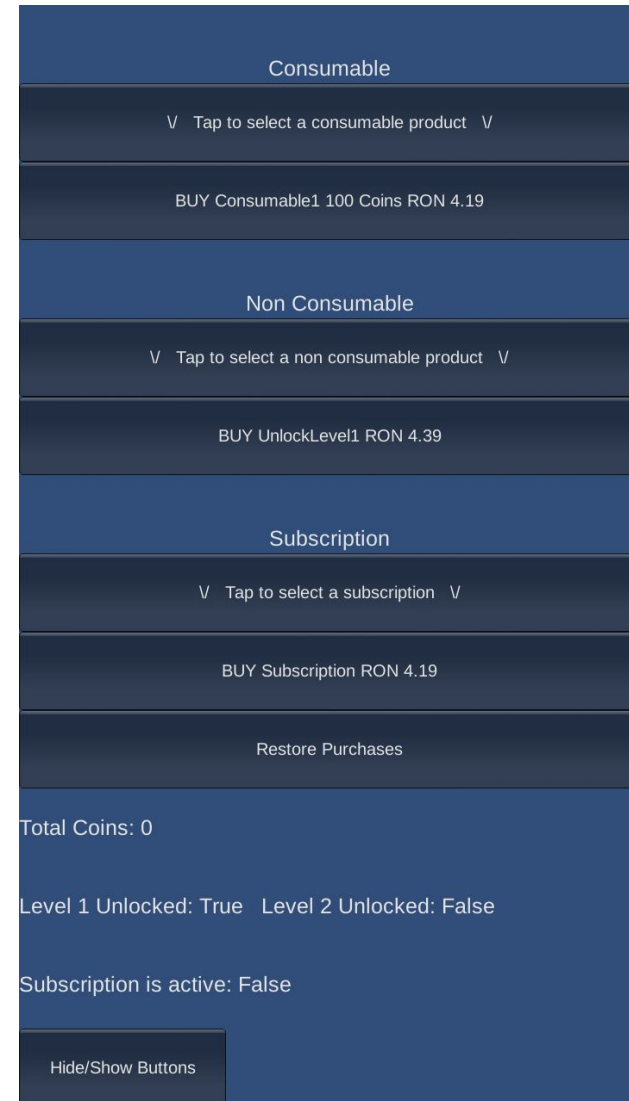
You can find the example test scene here:

**Assets/GleyPlugins/EasyIAP/Example/TestIAP.unity**

### How to use the scene:

- At first only an **Initialize** button is displayed.
- After initialization is complete, the UI from the right image will appear.
- Your store products are divided into 3 categories: Consumable, Non Consumable and Subscription
- Tap to select a product from each category and then you can buy it using the **Buy Button**
- If debug mode is enabled you will see debug messages on your device after each action you perform.
- Press **Show/Hide Buttons** to hide all buttons and read more easily the debug messages.

**To use the test scene first you have to add your products in the settings window as described in Setup Guide.**





## 6. TROUBLESHOOTING

---

- **Easy IAP should work in Unity Editor without charging you.**
- **To make sure all works from the first time on your device please check that:**
  - internet connection is on.
  - device date is correct.
  - your IDs from settings window does not contain any illegal characters at the beginning or the end.
  - you are using your test user for unpublished apps on IOS or Google Play.
  - your app is published in alpha or beta for Google Play.
  - your app has the same bundle id as your app created on Google Play or iTunes Connect console.
  - your app is signed with the release key used to upload it on Google Play console.



Version 1.2.0 / 2018