



In-App-Purchase Simple & Easy Workflow

for Apple AppStore,
Google Play Store
and Amazon App Store

iap
Manager
DIGICRAFTS®



IAP Manager

Easy IAP Workflow

Document version 1.8.0

Support email: support@digicrafts.com.hk

A. Before Start

In order to use the **IAPManager**, the following steps should be done.

1. Add products for in-app-purchase in each store.

Follows the tutorial for each store in below links.

iTune Connect (IOS):

<https://www.raywenderlich.com/105365/in-app-purchases-tutorial-getting-started>

Play Store (Android):

<http://www.theappguruz.com/blog/implement-in-app-purchase-version-3> (See Part 4)

Amazon (Kindle):

<https://developer.amazon.com/public/apis/earn/in-app-purchasing/docs-v2/submitting-copy>

2. Setup Unity Purchasing Services.

You can follow the tutorial below to setup Unity Purchasing Services (The scripting part is not requested).

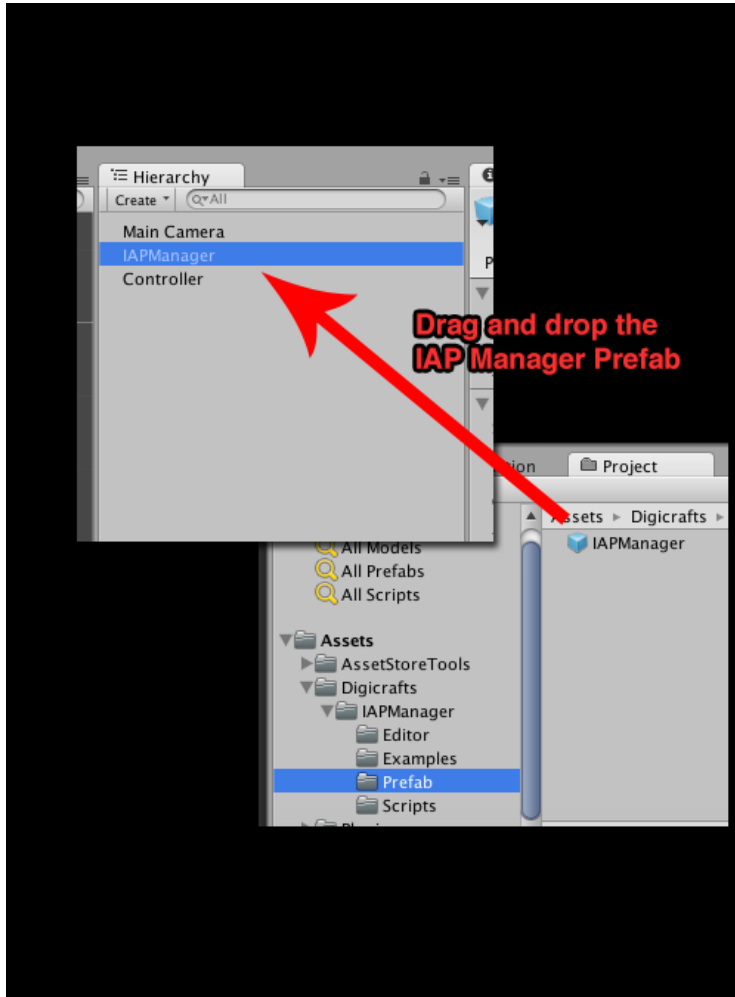
<http://docs.unity3d.com/Manual/UnityIAPSettingUp.html>

3. Download and import Digicrafts IAP Manager assets package from asset store.

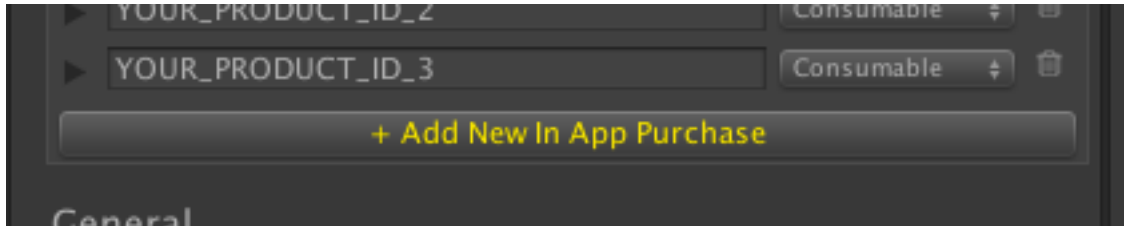
B. Quick start

**Please skip this section if you are using visual language editor such as Bolt (go to Section D) or Playmaker (go to Section E)*

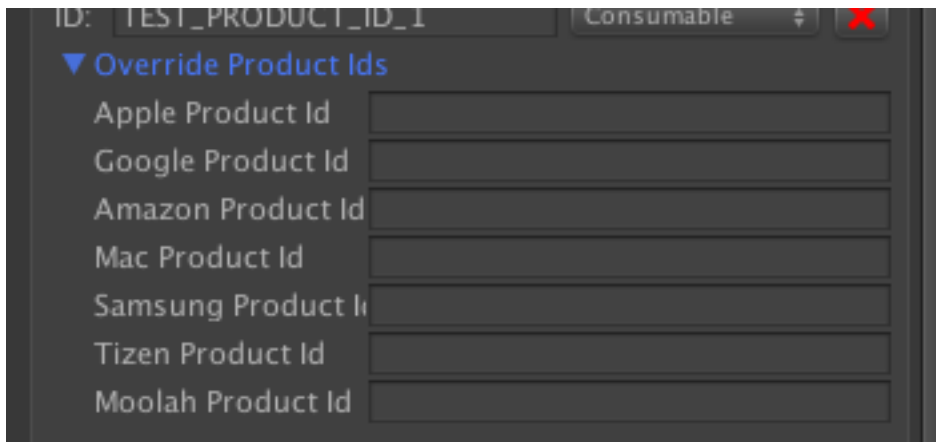
1. Drag and Drop the **IAP Manager** prefab into the scene. You can find the prefab from “Digicrafts/IAPManager/Prefab/IAPManager.prefab”.



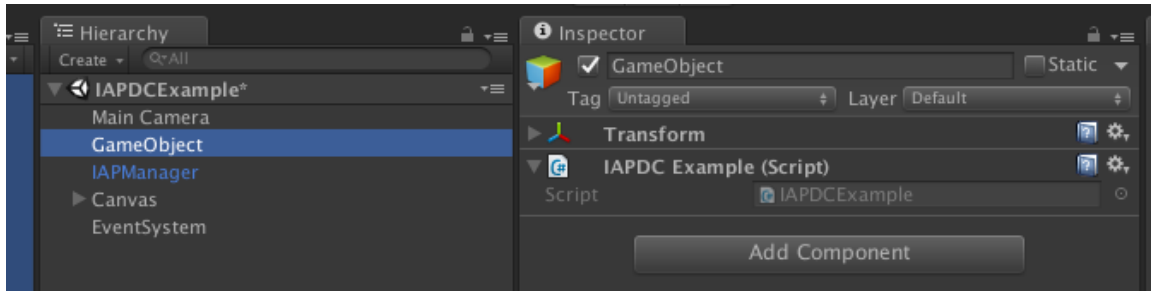
2. Select the **IAPManager** object and open inspector. Add a new product by pressing the “Add New In App Purchase” button. Fill in the product id which you have defined on app store and select the product type.



3. If you have a different product id in each store, you can fill in ids in the “Override product ids”. This will override the ID filled above.



4. Create an empty game object in your scene and add a new script with the following content.



```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;
using Digicrafts.IAP;

public class IAPDCEExample : MonoBehaviour, IIAPDelegate {

    //--- IIAPDelegate

    // Event when IAP initialized
    public void OnIAPInitialized(Dictionary<string, IAPProduct> products){

        // products contains a Dictionary whcih sotre product information
    }

    // Event when IAP initialized Fail
    public void OnIAPInitializeFailed(string error) {}

    // Event when a purchase started
    public void OnIAPPurchaseStart(IAPProduct product) {}

    // Event when when a purchase finished and success
    public void OnIAPProcessPurchase(IAPProduct product, string transactionID, string receipt) {

        // Do somthing after purchase finished
        // You can get the product information and receipt from here.
    }

    // Event when a purchase failed
    public void OnIAPPurchaseFailed(IAPProduct product, string failureReason){

    }

    // Event for deferred purchahse
    // On Apple platforms we need to handle deferred purchases caused by Apple's Ask to Buy
    // feature.
    // On non-Apple platforms this will have no effect; OnDeferred will never be called.
    public void OnIAPProcessDeferred(IAPProduct product) {

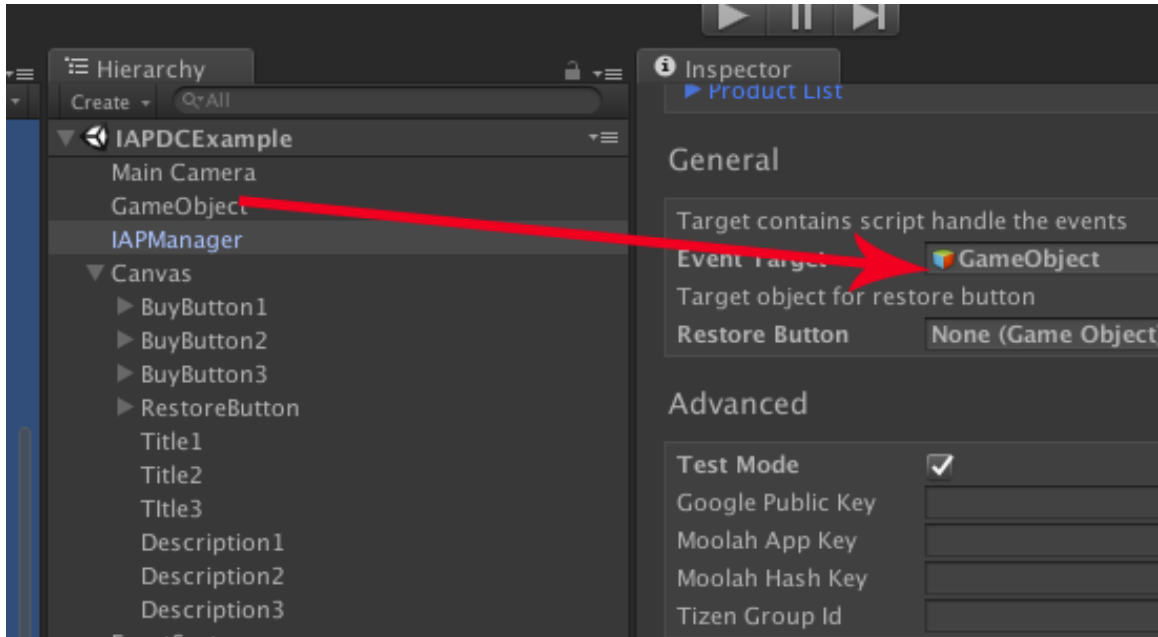
    }

    // Event for restore purchase
    // Success set to true if restore success
    public void OnIAPTransactionsRestored(bool success) {

    }

}
```

5. Connect the script object to IAPManager



Drag the game object with the script to the Event Target of **IAPManager**.

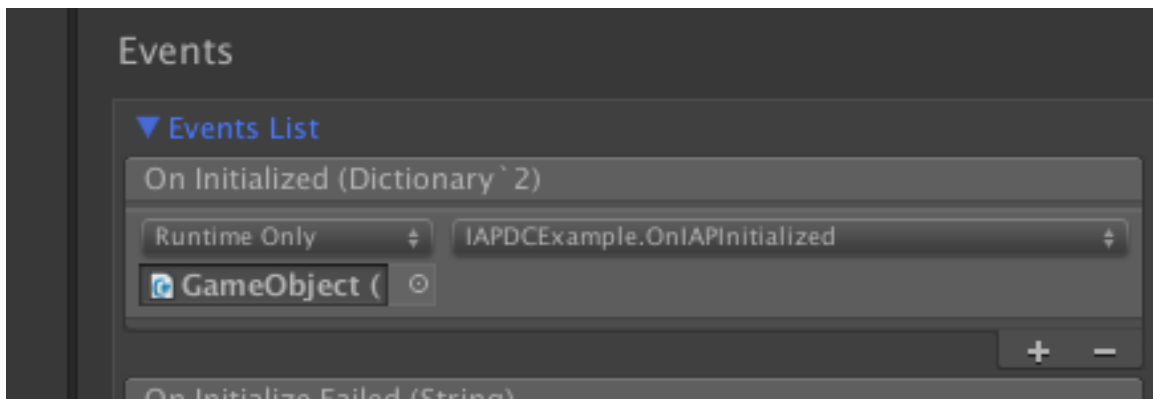
Basic setup is completed. You can test your script.

C. Additional Setup

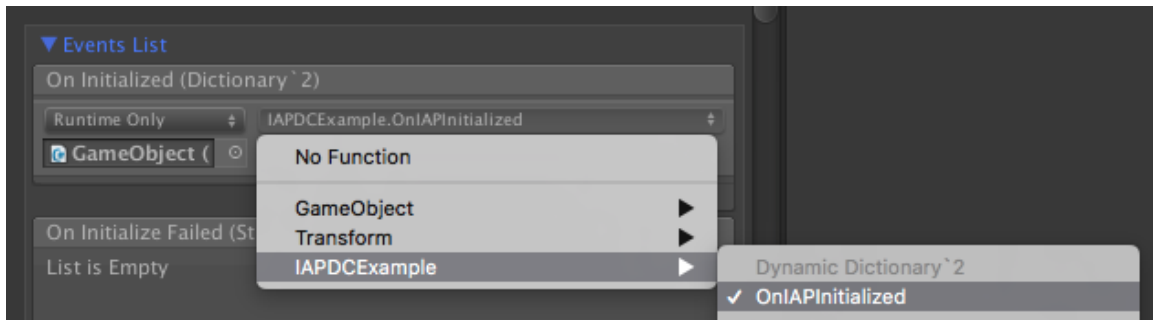
Connecting Events

IAP Manager inspector provides a list of event connector. You can connect the event to your own script function.

Open up the events section in the IAP Manager inspector. Click the “+” button to create a new event entry.

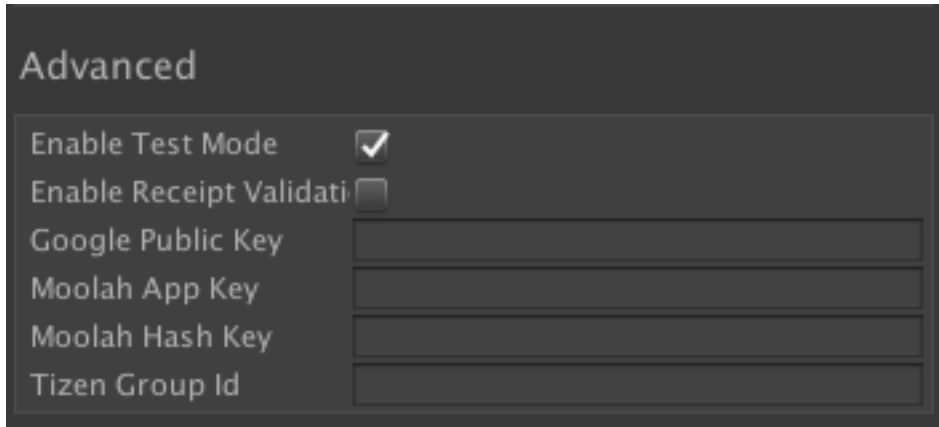


Select the game object/script to connect and specify the function to call.



Enabled Receipt Validation

You can enable/disable client side receipt validation from the IAP Manager inspector.

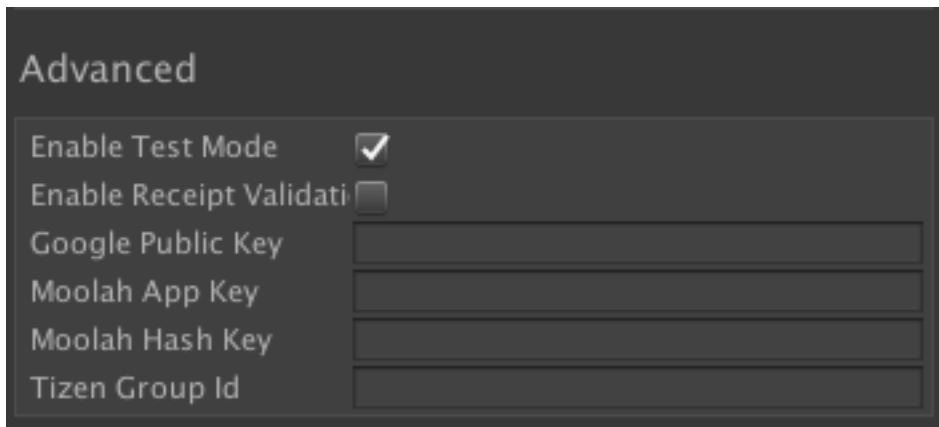


Advanced

Enable Test Mode	<input checked="" type="checkbox"/>
Enable Receipt Validation	<input type="checkbox"/>
Google Public Key	<input type="text"/>
Moolah App Key	<input type="text"/>
Moolah Hash Key	<input type="text"/>
Tizen Group Id	<input type="text"/>

Enabled Test Mode

You can enable/disable test mode from the inspector of IAP Manager. This will enable IAP Manager to use test data when testing on device. **Remember to disable it when you submit to app store.**



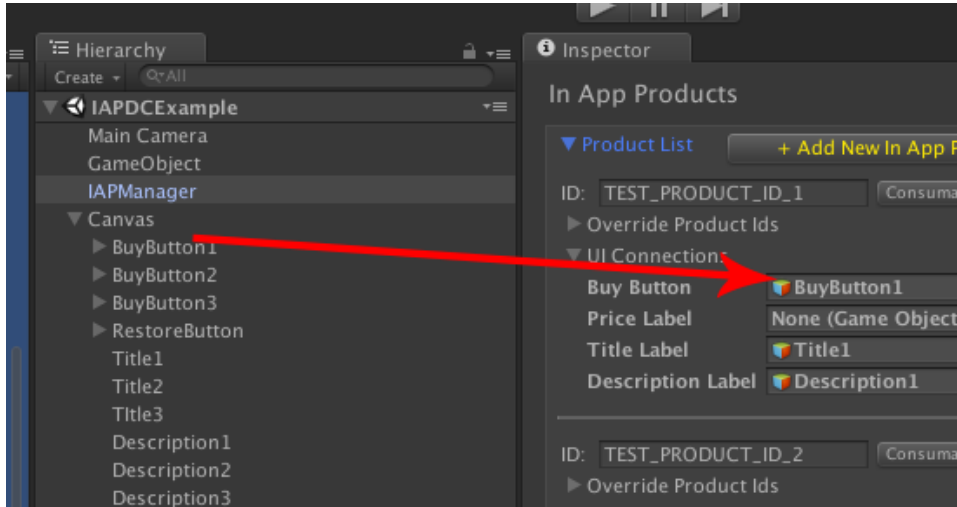
Advanced

Enable Test Mode	<input checked="" type="checkbox"/>
Enable Receipt Validation	<input type="checkbox"/>
Google Public Key	<input type="text"/>
Moolah App Key	<input type="text"/>
Moolah Hash Key	<input type="text"/>
Tizen Group Id	<input type="text"/>

Add a “Buy” buttons

IAPManager can help you setup buy buttons automatically. First, create a UI button in your scene. Then drag the button from hierarchy to **Buy Button** field of corresponding product in **IAPManager** inspector.

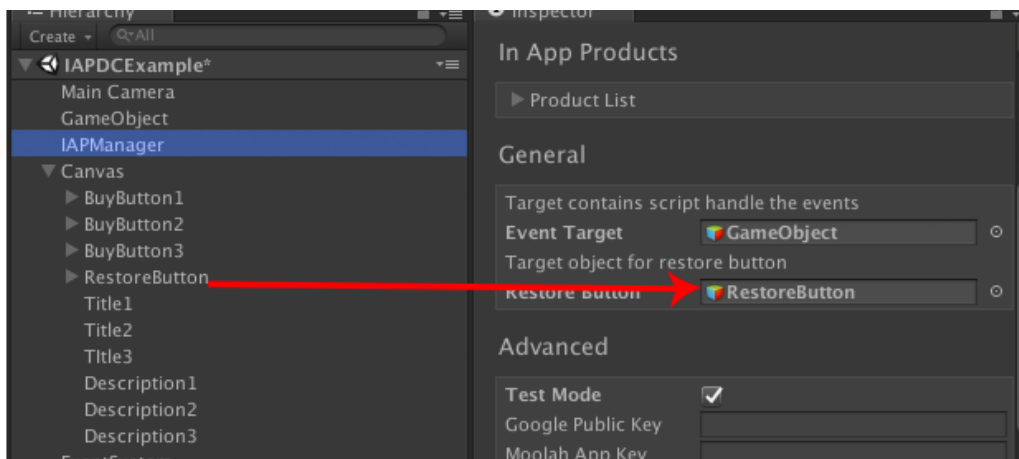
When run, the UI Button text will fill with price of the product automatically. If player click on the button, the purchase process will start.



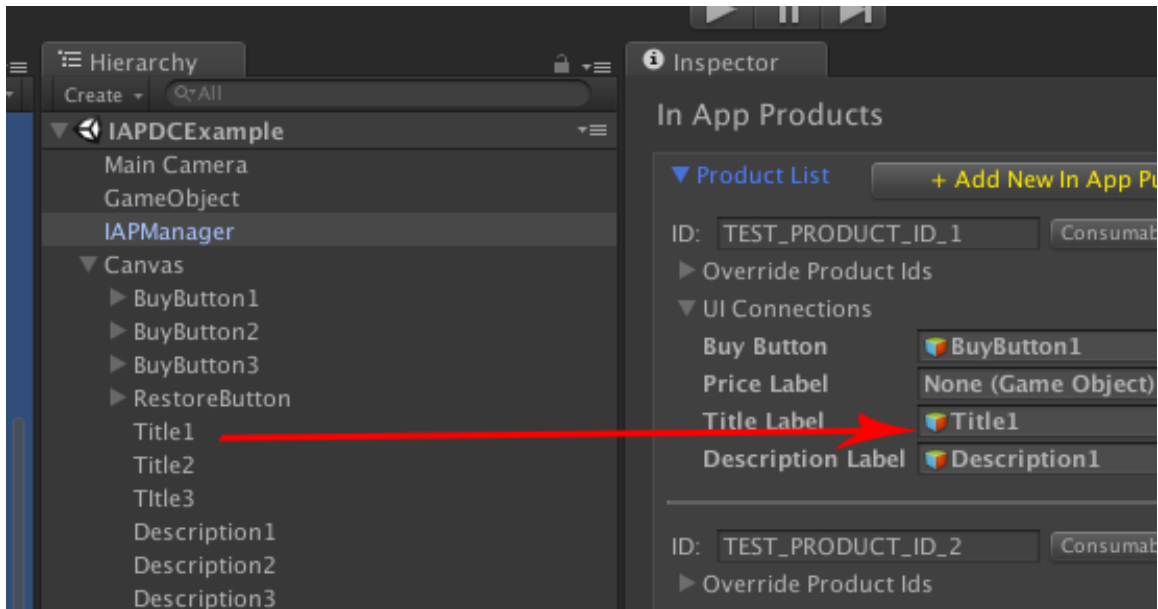
Add a “Restore button” for non-consumable product

IAPManager can help you setup restore button automatically. First, create a UI button in your scene. Select the **IAPManager** and open the inspector window. Drag the button from hierarchy to **Restore Button** field.

If you follow the quick start and setup the purchase process, no additional coding is needed.



Add product information text



IAPManager can display product information to Text UI. The Text UI will fill with product information automatically. The information included products's price, title and description.

D. Quick Start for Bolt

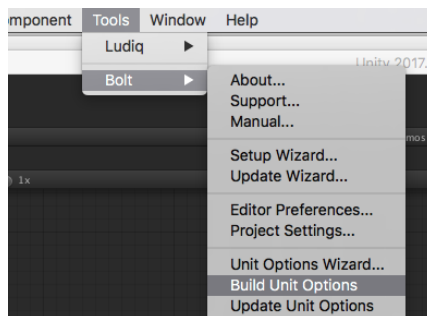


Pre-request

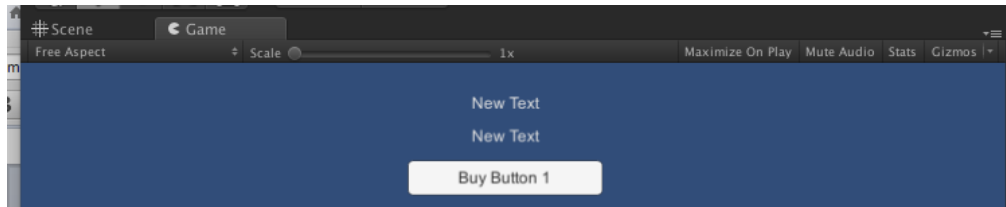
1. Installing Bolt. <https://ludiq.io/bolt>
2. Install IAPManager Bolt support plugin from menu Tools>Digicrafts>IAP>Install Bolt Plugin



3. *Rebuild the Bolt Unit Options by selecting the menu Tools>Bolt>Build Unit Options*

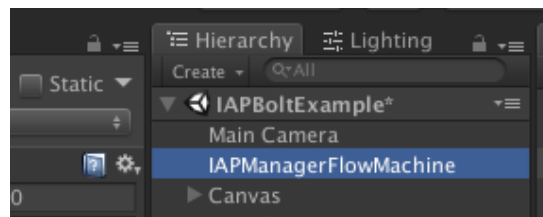


4. A scene, which setup with UI.

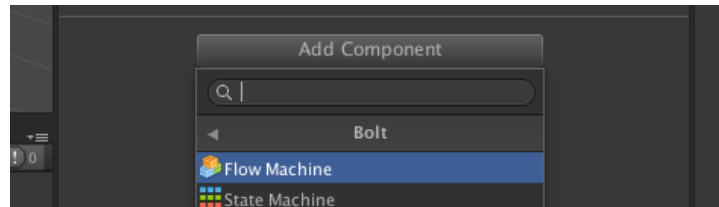


Setup IAP Manager flow graph

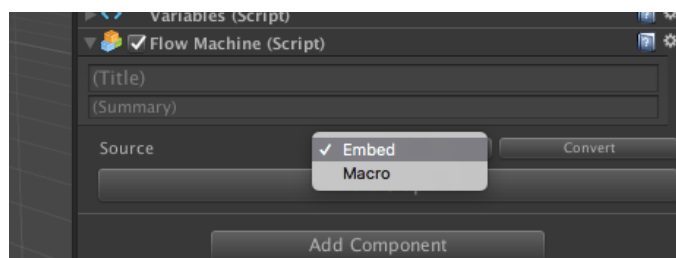
1. Create an empty *GameObject* and named “IAPManagerFlowMachine”.



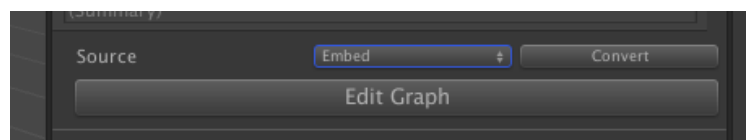
2. Add a **Flow Machine** by click the “Add Component” button and select *Bolt>Flow Machine*.



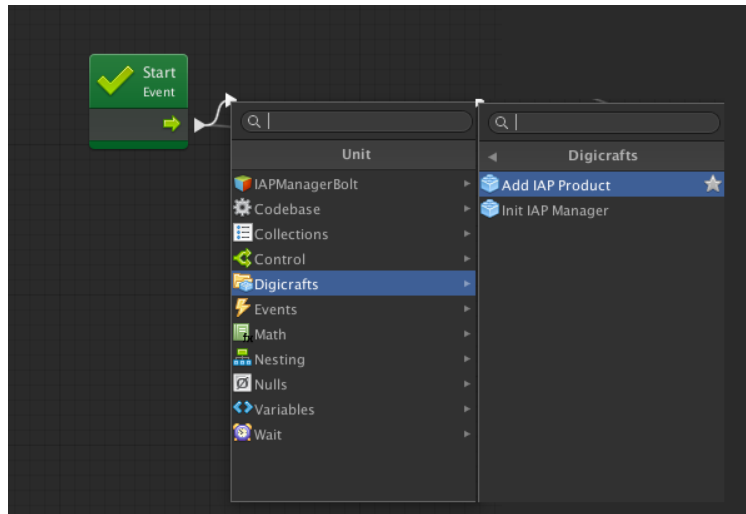
3. Select “*Embed*” from the source drop down.



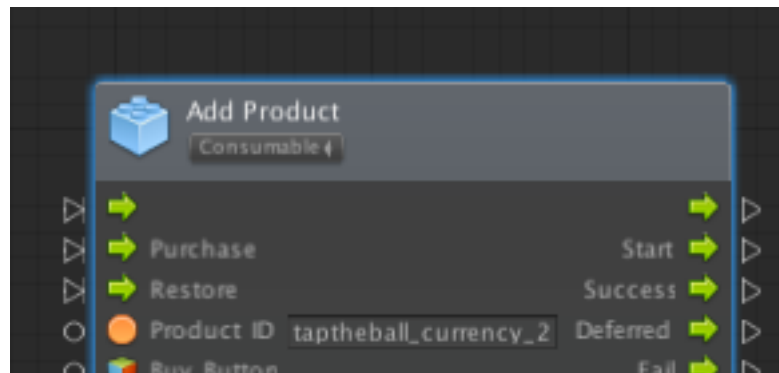
4. Open the **Graph Editor** by click on the Edit Graph button.



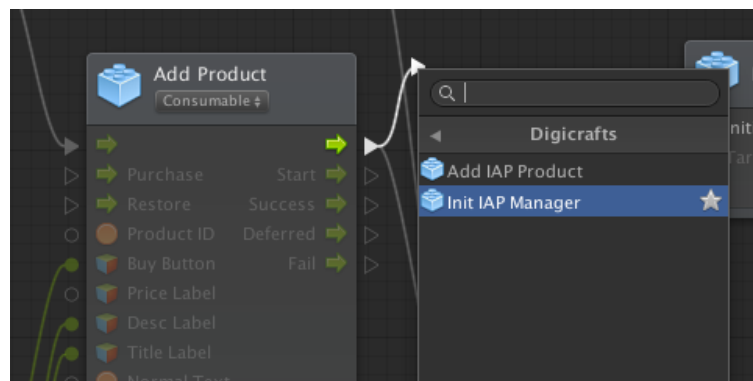
5. In the **Graph Editor**, click on the output port of Start Unit. Drag and drop to create an “**Add IAP Product**” unit from selecting *Digicrafts>Add IAP Product*.



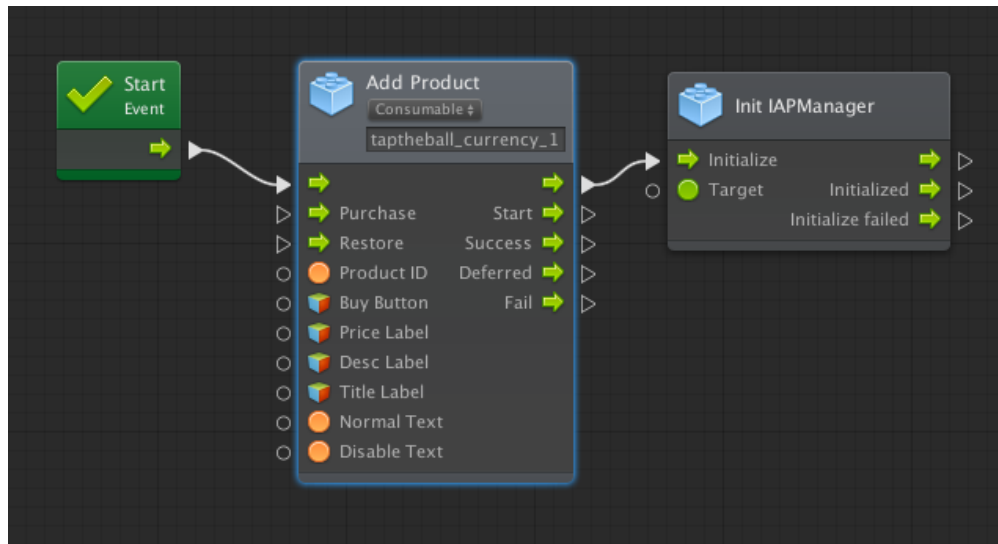
6. Select the **Add IAP Product** unit. Open the **Graph Inspector** and fill in the *Product ID* and select the type of the product.



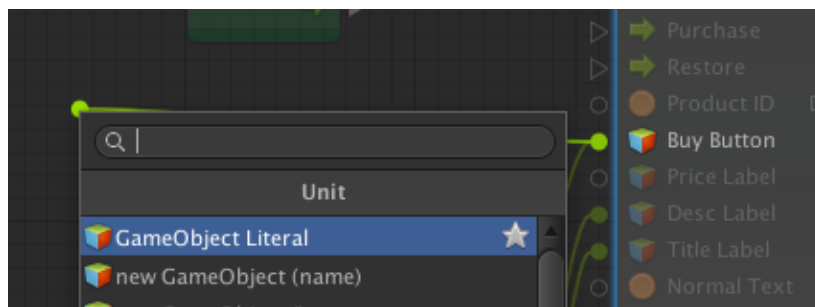
7. Click on the output port of the “**Add IAP Product**” unit. Drag and drop to create an “**Init IAP Manager**” unit from selecting *Digicrafts>Init IAP Maanger*.



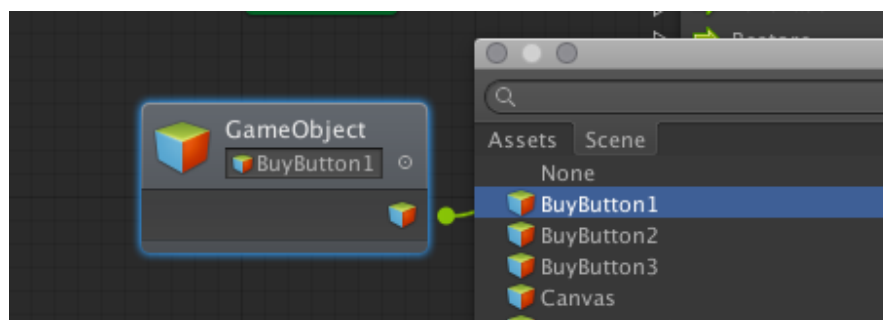
8. The basic flow graph is ready. We are going to add interactive.



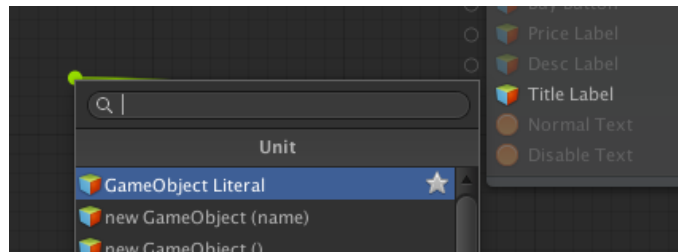
9. Select the *Add IAP Product* unit. Drag and drop from the “Buy Button” port and create a “*GameObject Literal*” unit.



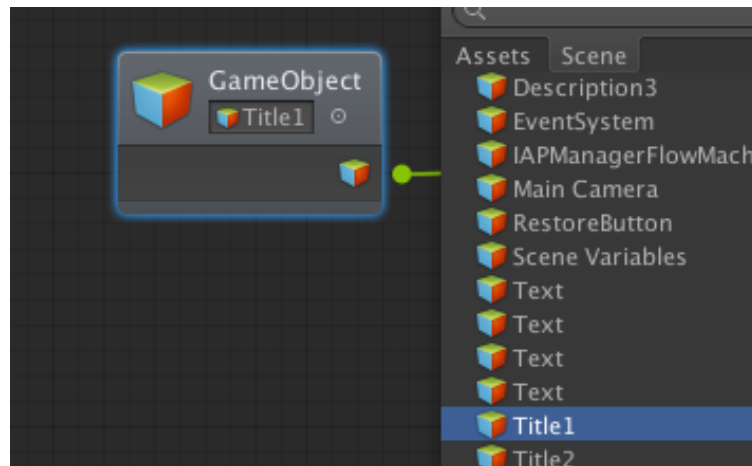
10. From the *GameObject* unit, select the buy button UIButton from your scene.



11. Drag and drop from the “*Title Label*” port and create a “*GameObject Literal*” unit.



12. Select the UILabel object which use to display the product title

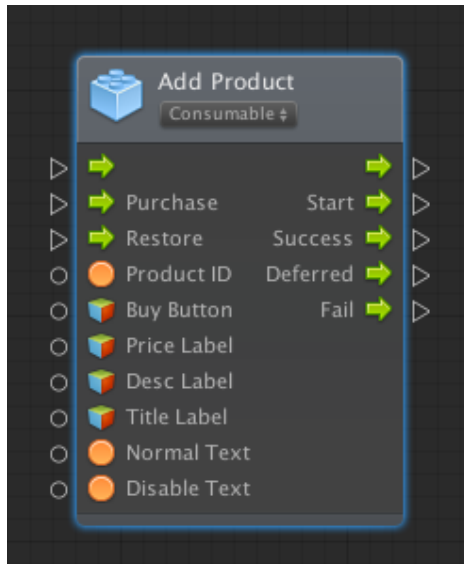


13. The basic setup is finish. Run the scene and you should see the buy button and product title.



IAP Manager Bolt Units

Add Product



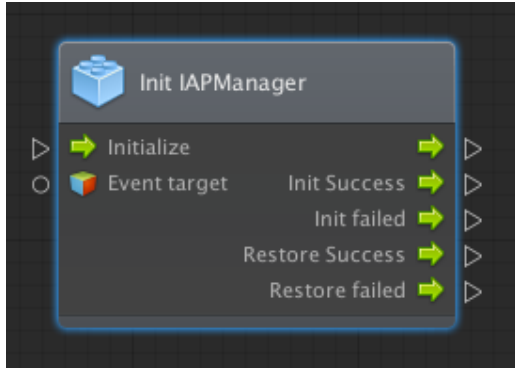
Inputs

Purchase	Trigger for purchasing this product
Restore	(IOS only) Trigger for restore this product
Product ID	String for setting product id.
Buy Button	UIButton reference. Button will perform buy action when click. And fill the label with product price
Price Label	UILabel reference. Auto fill with product price.
Desc Label	UILabel reference. Auto fill with product description.
Title Label	UILabel. Auto fill with product title.
Normal Text	String value for buy button when enabled
Disable Text	String value for buy button when disabled

Outputs

Start	Event when start purchasing.
Success	Event when purchase success. This event also fire if a product is restore successfully.
Deferred	Event when purchase deferred.
Fail	Event when purchase failed.

Init IAPManager



Inputs

Initialize	Trigger for initialization. Start initializing the IAPManager and fetch data from store.
-------------------	--

Event Target	Event target for events. Target will receive events from IAPManager
---------------------	---

Outputs

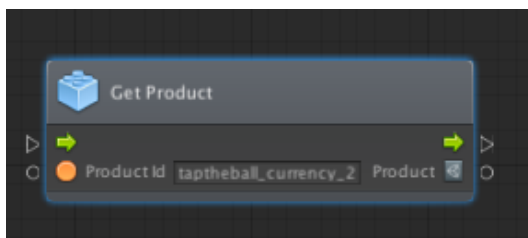
Init Success	Event when initialization finishes.
---------------------	-------------------------------------

Init failed	Event when initialization failed.
--------------------	-----------------------------------

Restore Success	(IOS only) Event fire after performs restore action success.
------------------------	--

Restore Success	(IOS only) Event fire after performs restore action fail.
------------------------	---

Get Product



Inputs

Start	Start event to get the product
--------------	--------------------------------

Product ID	The product id to identify the product.
-------------------	---

Outputs

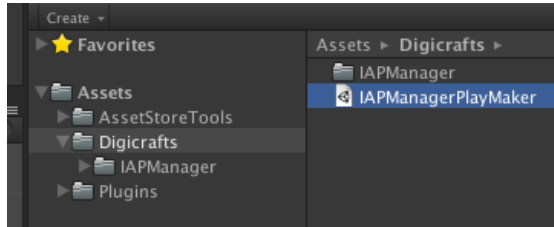
Finish	Event when action finishes.
---------------	-----------------------------

Product	IAPProduct Object output
----------------	--------------------------

E. Quick Start for PlayMaker

Pre-request

- PlayMaker 1.8+, IAP Manager 1.6+ and Unity IAP 1.92+ installed
- Familiarize working with PlayMaker.



Setup Play Maker Action

1. Install PlayerMaker support plugin from menu *Tools>Digicrafts>IAP>Install Playmaker Plugin*

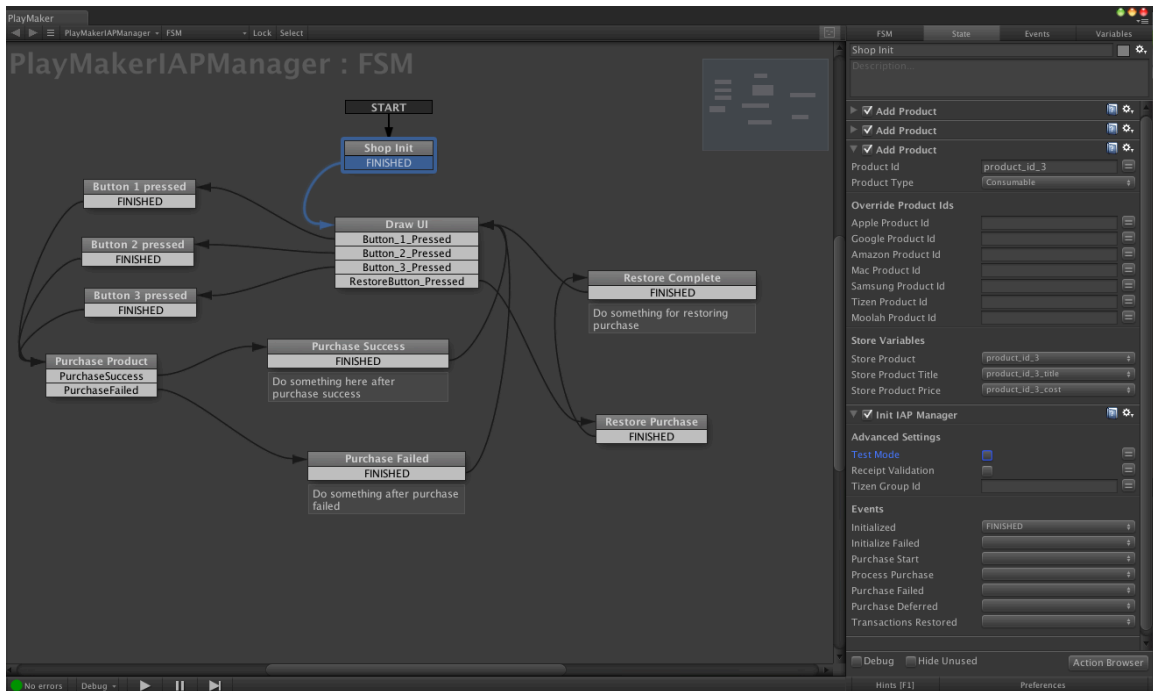


2. Create a new Play Maker FSM or using your existing FSM.
3. Create the variables to hold the product info. Here we have created “[product_id_x](#)” for IAPProduct instance, “[product_id_x_title](#)” for the product name and “[product_id_x_cost](#)” for the product cost.

**You can name the variables as your need. No restriction on the naming.*

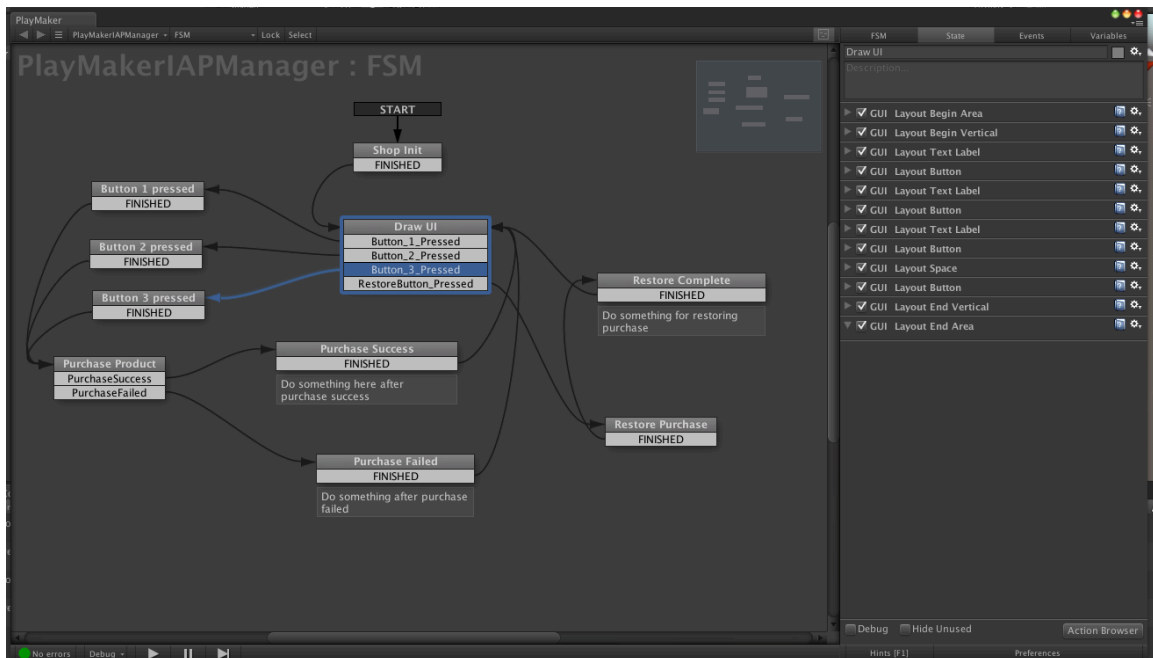
Name	Used	Type
button_event	0	String
product_id_1	1	Object
product_id_1_cost	2	String
product_id_1_title	2	String
product_id_2	1	Object
product_id_2_cost	2	String
product_id_2_title	2	String
product_id_3	1	Object
product_id_3_cost	2	String
product_id_3_title	2	String
product_to_buy	4	String

3. Create a state for initialing the IAPManager. Add the *"Add Product"* action from Action Browser.
4. In the *"Add Product"* action fill in the product id (i.e, the unique product id create from iTunes Connect or Google App portal). And assign the variables created from step 2.
5. Create action for each product.
6. Add the *"Init IAP Manager"* action at last.



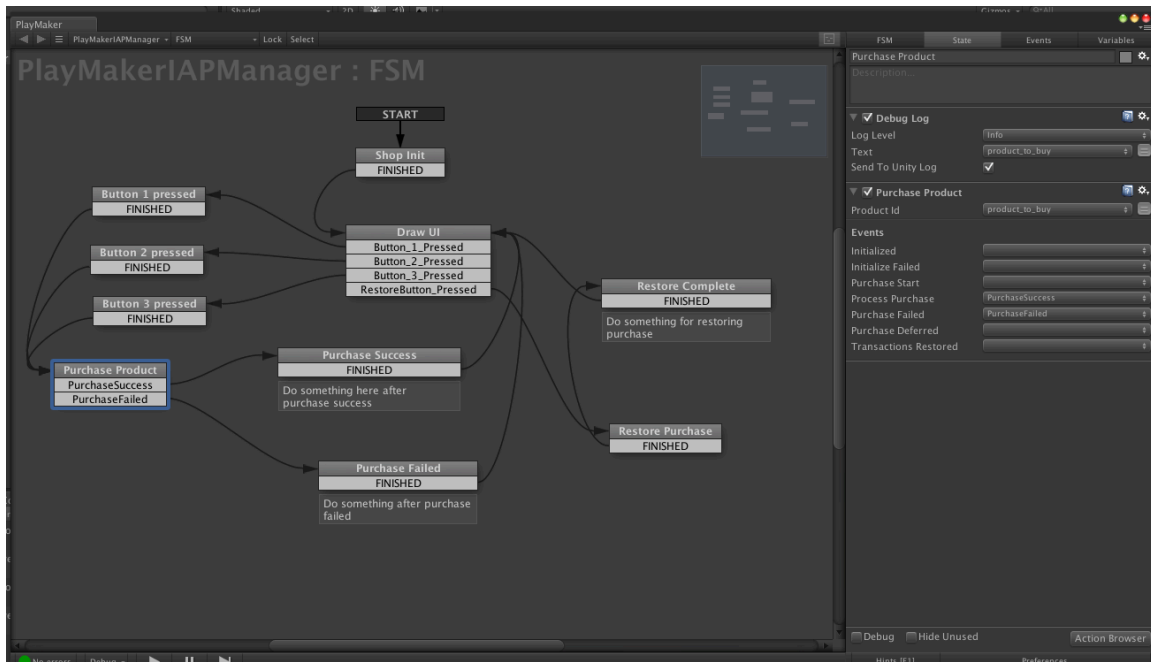
7. Connect the “*Initialized*” event to “*FINISH*”. The finish event should transit to a state, which display the buy button.

8. Create you own interface state for buy buttons. We create 3 buy buttons with Layout Button. And set the “*product_to_buy*” string according to button pressed.



9. Create the “Purchase Product” state. In the state, we add the “*Purchase Product*” action.

10. Fill the product id with the product you want this action to purchase. Here we use a variable *“product_to_buy”* to store the product id after press the buy button.



11. Connect the *“Process Purchase”* event to a state, which do something after purchase success.

12. Connect the *“Purchase failed”* event to a state, which do something after purchase failed.

IAP Manager Play Maker Action

The IAP Manager Action is organizing in the “IAP Manager” section. It contains following actions.

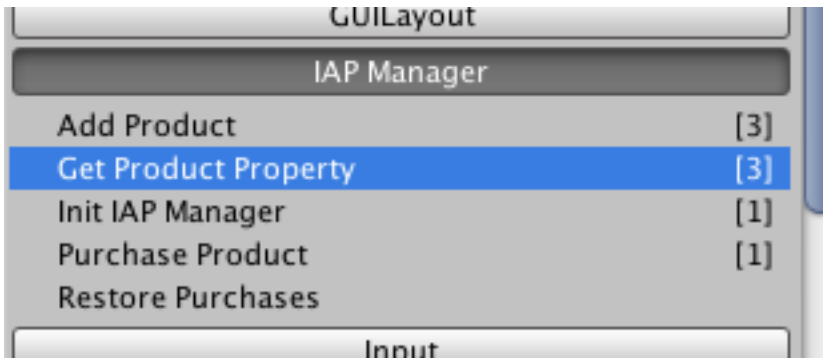
Add Product – add a product to IAPManager

Get Product Property – get the property of an IAPProduct variable

Init IAP Manager – initialize IAPManager

Purchase Product – purchasing a product with product id

Restore Purchases – invoke restore purchase process



E. Prepare for Amazon Store Submit

Amazon kindle use Android as platform. You need to package the apps with Android Apps.

Before build to Amazon, please select Window>Unity IAP>Android>Target Amazon. Then, start build your Android apk.

