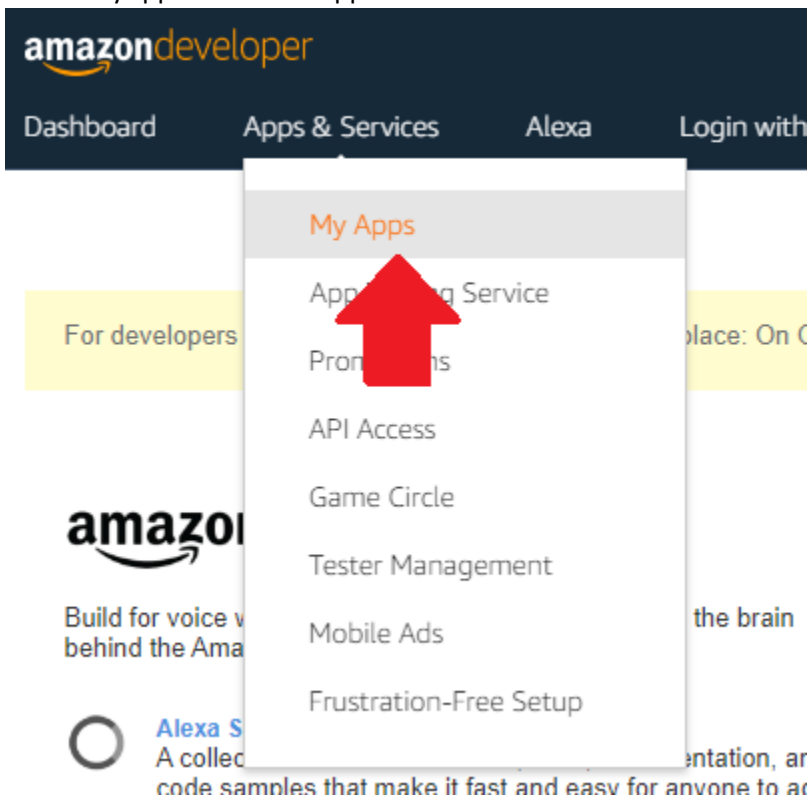


# Amazon IAP and BrainCloud Verification

1. Make an amazon developer account if you don't already have one here <https://developer.amazon.com/>
2. Go to your developer console



3. Under MyApps add a new app





In this case we are doing Android.

4. Fill in the app info  
**New App Submission**

App title\*

App SKU

App category\*

Customer support contact ☒ Use my default support information

Customer support email address\*

Customer support phone

Customer support website

5. Fill out your apps data and make sure everything is checked off

amazon developer

Dashboard Apps & Services Alexa Login with Amazon Dash Services Reporting Settings

< My apps

BrainCloudAmazonIAP

App version in progress 1/6 complete [App Submission Help](#)

☒ General Information ☒ Availability & Pricing ☒ Description ☒ Images & Multimedia ☒ Content Rating ☒ APK Files

App title\* BrainCloudAmazonIAP

App SKU bcAmazonIAP

App Submission API Keys

App ID   Release ID

App category\*

6. You'll notice in Availability and Pricing, that "If you app offers items for In-App purchases"... You will want to add your IAP here. We'll add a consumable for example.

General Information Availability & Pricing Description Images & Multimedia Content Rating APK Files

Where would you like this app to be available?\*

In all countries and regions where Amazon sells apps

Is your app free or paid?\*

Free

When would you like publishing to start?

Leave this field blank to start the publishing process as soon as your app has been approved. To defer the start of the publishing process to a future time and date, select it here. Your app will be available in the Appstore a few hours after the publishing process starts.

—

If your app offers items for In-App Purchase, click the button to get started.

Add IAP

Search IAPs

+ Add Single IAP Import Multiple IAPs using CSV Export Multiple IAPs

You have no in-app items. Add one today!

CREATE NEW CONSUMABLE

Consumable Title\*



Enter your consumable title

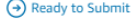

Consumable SKU\*

Enter your consumable SKU

Cancel Add Consumable

To complete your consumable, click on the newly created consumable in the list and fill in the details and pricing details and submit the IAP. For this example we are making it 0\$ to purchase.

Title	SKU	Price	Type	Status	▼ Last Submitted Date
 orb1 	bcOrb1	USD 0	Consumable	Ready to Submit	N/A

**orb1**  

 **Submit IAP**

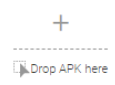
- Now that you've added something a player can buy, keep filling out the rest of the app checklist (Description, Images and Multimedia, content rating and apk files)

☒ General Information
 ☒ Availability & Pricing
 ☒ Description
 ☒ Images & Multimedia
 ☒ Content Rating
 ☒ APK Files

**Add APK** [Appstore Certificate Hashes](#)

Apply Amazon DRM? <sup>\*</sup> ☐ Yes ☒ No

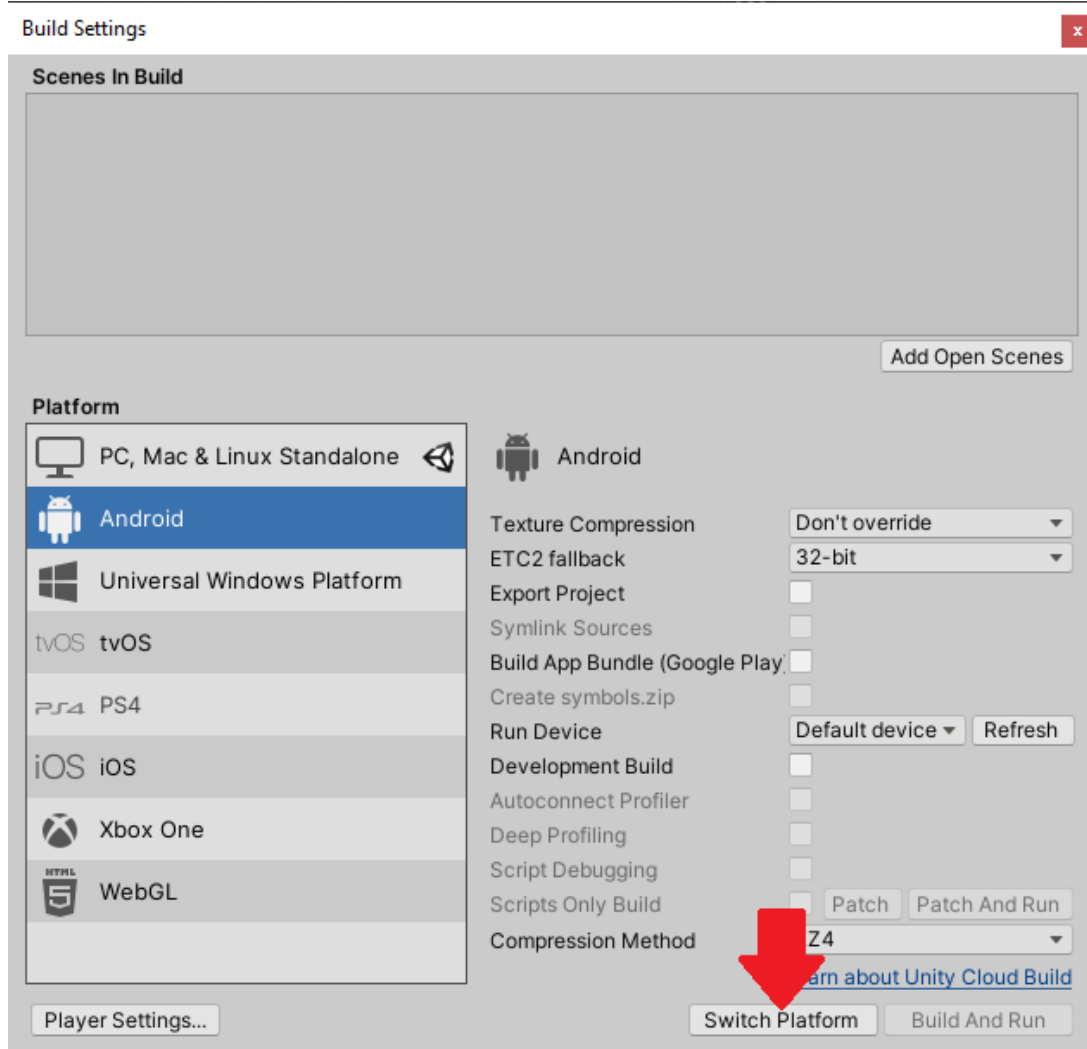
**APK File <sup>\*</sup>**  
 Upload your APK files one by one.  
 As part of the ingestion process, Amazon removes your developer signature and applies an Amazon signature. This signature is unique to you, does not change, and is the same for all apps in your account.



At the last step you'll notice you'll need an apk to finalize your app. So we'll save that for later.

- Time to open Unity. Make a new Unity project
- Use unity plugin here <https://developer.amazon.com/docs/apps-and-games/sdk-downloads.html> and import it into Unity.

10. In File>Build Settings, change the platform to Android



11. In File>Build Settings>Player Settings, in the Player tab go to Other Settings and under Identification you will find Package name. This can be changed by changing your Company Name and Product Name at the top of the Player tab. Change this now, because once you've uploaded an apk to Amazon it will always need to be the same package name for the app you upload it to.
12. Build the apk
13. Now Navigate to your project folder and go to Temp>StagingArea and open the UnityManifest.xml
14. Copy all the code in the UnityManifest.xml then add it to the AmazonIapV2SampleAndroidManifest.xml in Assets>Plugins>Android. This would have been added when you imported the plugin. This is mentioned here <https://developer.amazon.com/de/docs/cross-platform-plugins/cpp-add-plugin-to-unity-project.html>. ONLY DO THIS IF YOU DON'T ALREADY HAVE YOUR OWN ANDROID MANIFEST FILE, otherwise ADD THIS CODE TO YOUR ANDROID MANIFEST FILE!

Add this to the application part of your .xml

```
<receiver android:name = "com.amazon.device.iap.ResponseReceiver"
```

```

    android:permission = "com.amazon.inapp.purchasing.Permission.NOTIFY" >

<intent-filter>

    <action android:name = "com.amazon.inapp.purchasing.NOTIFY" />

</intent-filter>




</receiver>

```

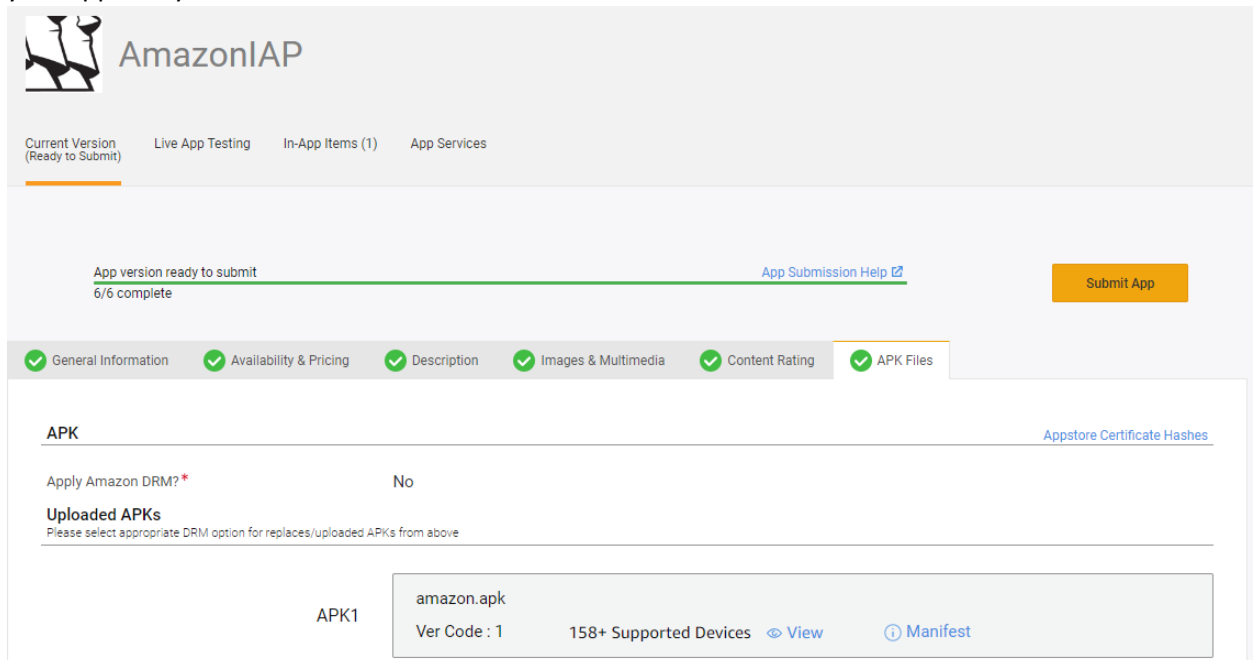
15. Rename AmazonIapV2SampleAndroidManifest.xml to AndroidManifest.xml, IF YOU DON'T ALREADY HAVE YOUR OWN ANDROID MANIFEST FILE.
16. Change the package name of your AndroidManifest to the package name matching your project. To confirm the package name matches go to Player Settings, and in the Player tab go to Other Settings and under Identification you will find Package name.
17. Build the apk again and check that the receiver was properly added. Drag your apk into Andorid studio and look at the AndroidManifest.xml and confirm the receiver is there and the package name is as desired.
18. This link gives a good idea of what you'll need to do in order to get an in-app purchase going in code. Refer to their example if you're trying to set up a simple purchase with amazon. You can also refer to our code example. Basically you simply need to make a Purchase call through the amazon instance, and listen for the purchase callback to get the userId and receiptId you need. <https://developer.amazon.com/de/docs/cross-platform-plugins/cpp-use-the-iap-plugin-for-unity.html>

**\*NOTE\*** One of the most key parts of getting in-app purchases to work without headaches is patience. The IAP you added to your Amazon Console has to first be approved and go LIVE before you can make any purchases with it. So make sure to submit it while you're working away.

#### In App Purchasing

Search IAPs				<a href="#">Add Single IAP</a> <a href="#">Import multiple IAPs using CSV</a> <a href="#">Export Multiple IAPs</a>	
Title	SKU	Price	Type	Status	▼ Last Submitted Date
 orb2	test_bitheads_orb2	CAD 0	Consumable	Live	Jun 17, 2020
 orb1	test_bitheads_orb1	CAD 0	Consumable	Live	Jun 17, 2020
 test_power	test_bitheads_power	CAD 0	Entitlement	Live	Jun 17, 2020

19. You can test your app by setting up Live-App Testing. To do this, first upload your APK and get your app ready to submit.



AmazonIAP

Current Version (Ready to Submit) Live App Testing In-App Items (1) App Services

App version ready to submit 6/6 complete [App Submission Help](#)

[Submit App](#)

General Information Availability & Pricing Description Images & Multimedia Content Rating **APK Files**

APK [Appstore Certificate Hashes](#)


Apply Amazon DRM? \* No

**Uploaded APKs**  
Please select appropriate DRM option for replaces/uploaded APKs from above

APK1

amazon.apk  
Ver Code : 1 158+ Supported Devices [View](#) [Manifest](#)

20. Then go to the Live-App testing tab, create a new test, set up testers with emails, then from your device, accept the invitation through the testers email, and download the app onto your device to test it. Make sure you are using the proper amazon localization for your app. For example if you're in Canada, you would visit it on amazon.ca, respectively. Make sure your test says "Test In Progress" or else it will not be the proper test with the latest apk that you've uploaded. It could take more than 10 minutes for a test to get processed and say its in progress.

Test	ATS Results	Start Date	End Date	Status	Actions
Test #14	0 issues	Fri Jun 19 2020	-	 Test In Progress	<a href="#">View</a>

### IMPORTANT TIP WHEN SETTING UP TESTS

Make sure to **delete** your apk from both the current version of your app and the live app testing version of your app every time you want to set up a new test with a new apk. Delete, then drag and drop the new apk, this may avoid some headaches where only older versions of your app are being downloaded through testing.

Also, it is a good idea to have a version number of some sort to see that you are actually working with your latest versions when testing,

## Now we're ready to hook up **BrainCloud**!

21. Start by creating a new app on brainCloud portal or working off an already existing app  
<https://portal.braincloudservers.com/loginfailed>
22. Under Core App Info > Platforms, be sure to check off Amazon

23. Under Core App Info > Application Ids go to Configure Platforms, select amazon, and in Shared Secret, you will need the Shared secret from your amazon developer console. That can be found under Settings > Identity in your developer console.

The screenshot shows the Amazon Developer console interface. At the top, the navigation bar includes links for Dashboard, Apps & Services, Alexa, Login with Amazon, Amazon Dash Replenishment, Reporting, and Settings. Below this, a secondary bar contains links for My Account, Company Profile, Payment Information, Tax Identity, User Permissions, Mobile Ads, Identity, and Security. Red arrows point to the 'Identity' and 'Settings' links in the top bar. The main content area is divided into two sections. The first section, titled 'Shared Key', displays a long black bar with a red highlight and a red arrow pointing to it. Below this bar is a 'Copy' button and a note: 'The shared key is unique to your developer account. Utilize it to validate transactions outside of your app. NOTE: This is a dynamically generated key and for security reasons not stored in our servers'. The second section, titled 'Configure Platforms', shows a row of platform icons: Apple, Facebook, Google, Steam, Twitter, and Amazon. A red arrow points to the Amazon icon. Below the icons is a 'Shared Secret' field, which is a long black bar with a red highlight.

24. The next thing to do is to make braincloud products that match your amazon IAPs. To do this, go to MarketPlace>Products and add a Product.

The screenshot shows the BrainCloud MarketPlace>Products page. On the left, there is a table with the following columns: Item ID, Category, Title, and Price. The table contains one record: 'test\_bitheads\_orb1', 'powerup', 'orb1', and 'Not for sale'. Below the table is a '1 records' indicator. To the right of the table is an 'Upload Image' button. On the right side of the page is a form for adding a product. The form includes fields for 'Item id \*' (with the value 'test\_bitheads\_orb1'), 'Title \*' (with the value 'orb1'), and 'Category' (with the value 'powerup'). Below these fields is a 'Description' field with the value 'an orb for powering up!'. The 'Product Type' is set to 'Consumable'. There is a 'Virtual Currencies' section with a '+' icon. Below this is a 'Prices' section with a '+' icon, containing a radio button for 'Not for sale (Default)'. At the bottom of the form is an 'Extra Data' section with a '+' icon. At the very bottom of the form are three buttons: 'Close', 'Delete', and 'Save'.

25. Save it then you can edit it and add a price to it, enable it for Amazon. Make sure the Amazon Product Id matches the SKU of the product and the price matches the price you have on Amazon. For proper practice I would also make the brainCloud Item ID match the SKU of the Amazon product you are linking with brainCloud.



26. Grab our plugin here and import it into Unity <https://github.com/getbraincloud/braincloud-csharp/releases>
27. Sign into your braincloud account and app though our select settings. You will want to set up a test similar to this example, where you Authenticate with brainCloud, Check the sales inventory for the list of products, then make an amazon purchase, and in the purchase response you will want to isolate the userId and receiptId so that you can verify the purchase with brainCloud!

And that's it! Happy Coding!