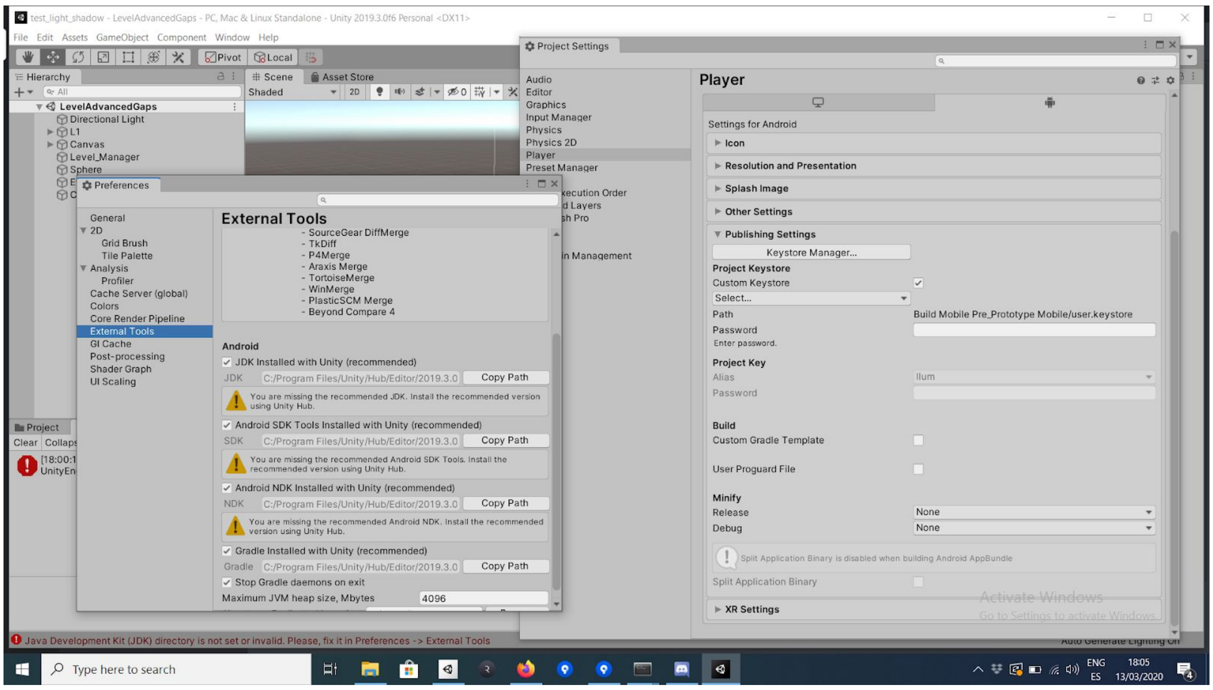
How to publish a mobile app created with Unity3D on the Google Play Console.

Here, using Unity3D 2019.3.x

1. Make sure you have the right tools and plugins

In the Unity menu, go to Edit > Preferences > External Tools

If you see the warnings below, it means you do not have them



External Tools

- SourceGear DiffMerge
- TkDiff
- P4Merge
- Araxis Merge
- TortoiseMerge
- WinMerge
- PlasticSCM Merge
- Beyond Compare 4

Android

✓ JDK Installed with Unity (recommended)
JDK C:/Program Files/Unity/Hub/Editor/2019.3.0
You are missing the recommended JDK. Install the recommended version using Unity Hub.

✓ Android SDK Tools Installed with Unity (recommended)
SDK C:/Program Files/Unity/Hub/Editor/2019.3.0
You are missing the recommended Android SDK Tools. Install the recommended version using Unity Hub.

✓ Android NDK Installed with Unity (recommended)
NDK C:/Program Files/Unity/Hub/Editor/2019.3.0
You are missing the recommended Android NDK. Install the recommended version using Unity Hub.

✓ Gradle Installed with Unity (recommended)
Gradle C:/Program Files/Unity/Hub/Editor/2019.3.0
✓ Stop Gradle daemons on exit
Maximum JVM heap size, Mbytes 4096

Player — Settings for Android — Publishing Settings

Project Keystore
Custom Keystore ✓
Path Build Mobile Pre_Prototype Mobile/user.keystore
Password
Enter password.

Project Key
Alias Ilum
Password

Build
Custom Gradle Template ☐
User Proguard File ☐

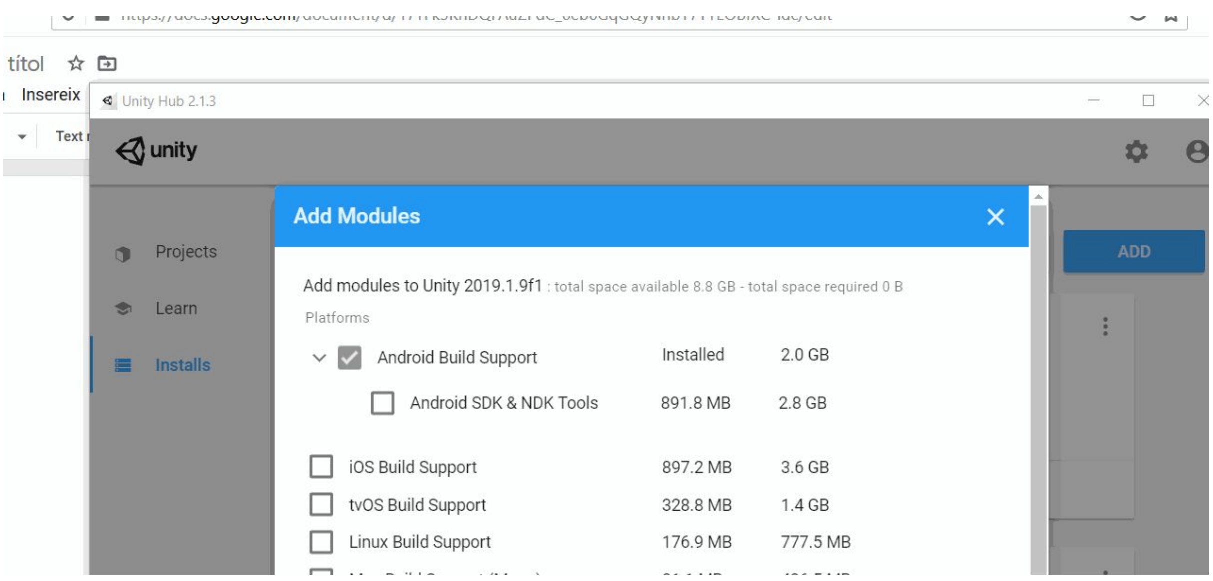
Minify
Release None
Debug None

Split Application Binary is disabled when building Android AppBundle
Split Application Binary ☐

Java Development Kit (JDK) directory is not set or invalid. Please, fix it in Preferences -> External Tools

If you do not have them, Go to Unity Hub (or install it if you haven't), go to the install that you are using, select Add Modules. Then, open the snippet Android Build Support, and ADD Android SDK Tools.

In 2019.1, it looks like this:



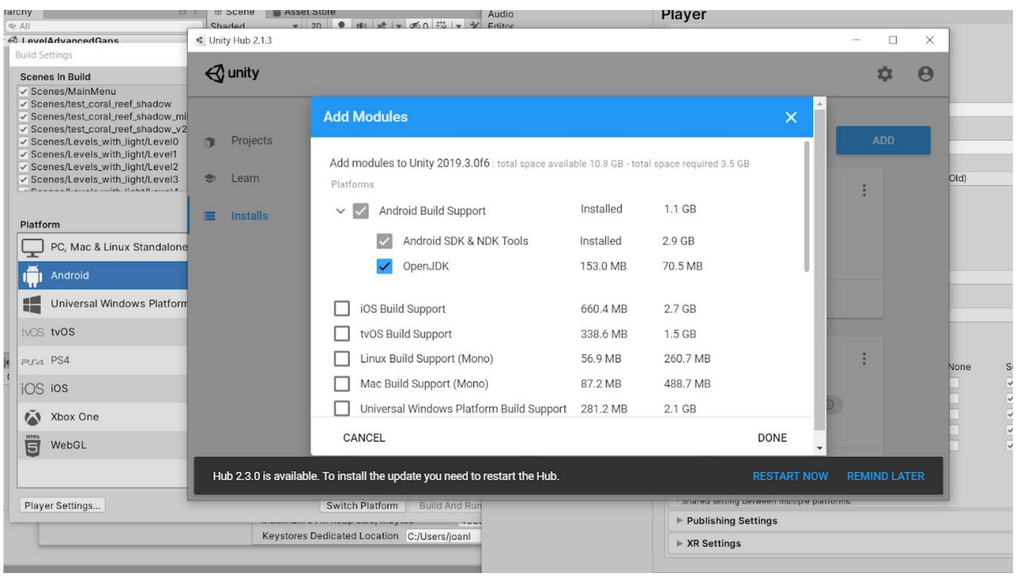
Add Modules

Add modules to Unity 2019.1.9f1 : total space available 8.8 GB - total space required 0 B

Platforms

✓ Android Build Support	Installed	2.0 GB
Android SDK & NDK Tools	891.8 MB	2.8 GB
iOS Build Support	897.2 MB	3.6 GB
tvOS Build Support	328.8 MB	1.4 GB
Linux Build Support	176.9 MB	777.5 MB

In 2019.3, it looks like this:


Build Settings
Scenes In Build
Scenes/MainMenu
Scenes/test_coral_reef_shadow
Scenes/Levels_with_light/Level0
Scenes/Levels_with_light/Level1
Scenes/Levels_with_light/Level2
Scenes/Levels_with_light/Level3
Platform
PC, Mac & Linux Standalone
Android
Universal Windows Platform
tvOS
PS4
iOS
Xbox One
WebGL
Player Settings...
Switch Platform
Build And Run
Keystores Dedicated Location
C:/Users/joanl
Player
Publishing Settings
XR Settings
Unity Hub 2.1.3
unity
Projects
Learn
Installs
ADD

Add Modules

Add modules to Unity 2019.3.0f6 : total space available 10.8 GB - total space required 3.5 GB

Platforms

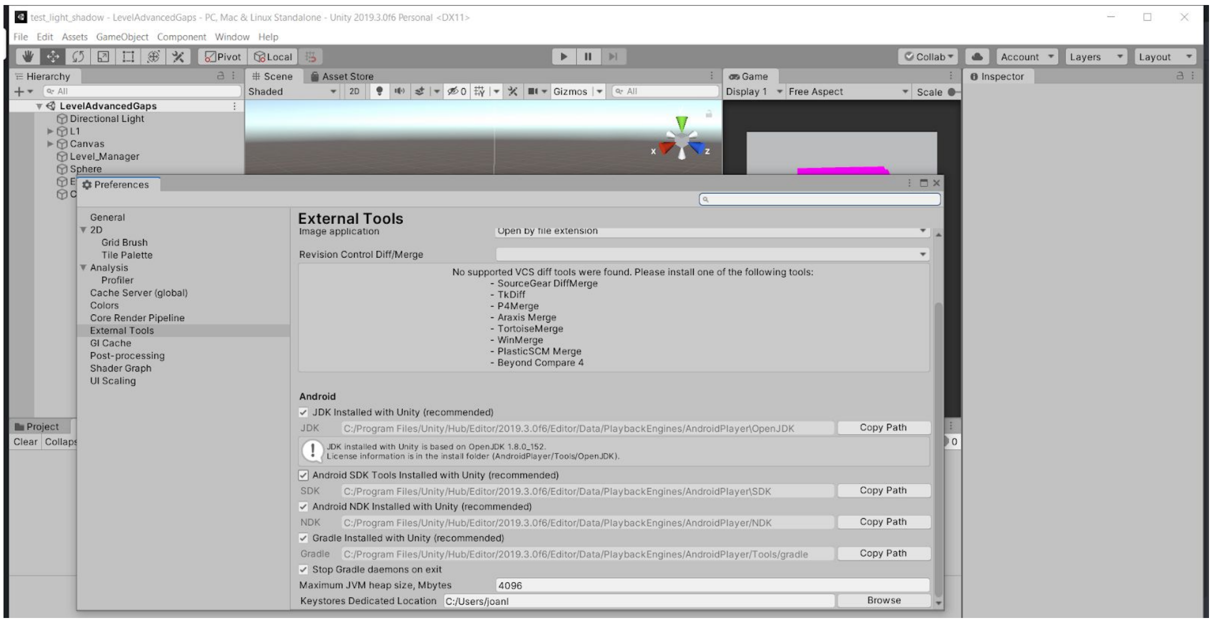
✓ Android Build Support	Installed	1.1 GB
✓ Android SDK & NDK Tools	Installed	2.9 GB
✓ OpenJDK	153.0 MB	70.5 MB
iOS Build Support	660.4 MB	2.7 GB
tvOS Build Support	338.6 MB	1.5 GB
Linux Build Support (Mono)	56.9 MB	260.7 MB
Mac Build Support (Mono)	87.2 MB	488.7 MB
Universal Windows Platform Build Support	281.2 MB	2.1 GB

CANCEL DONE

Hub 2.3.0 is available. To install the update you need to restart the Hub. RESTART NOW REMIND LATER

From now, we will be using 2019.3

Once Installed, your Unity window should look like this:



External Tools

Image application: Open by file extension

Revision Control Diff/Merge

```
No supported VCS diff tools were found. Please install one of the following tools:
- SourceGear DiffMerge
- TkDiff
- P4Merge
- Araxis Merge
- TortoiseMerge
- WinMerge
- PlasticSCM Merge
- Beyond Compare 4
```

Android

- ✓ JDK Installed with Unity (recommended)
  JDK C:/Program Files/Unity/Hub/Editor/2019.3.0f6/Editor/Data/PlaybackEngines/AndroidPlayer\OpenJDK
  JDK installed with Unity is based on OpenJDK 1.8.0_152.
  License information is in the install folder (AndroidPlayer/Tools/OpenJDK).
- ✓ Android SDK Tools Installed with Unity (recommended)
  SDK C:/Program Files/Unity/Hub/Editor/2019.3.0f6/Editor/Data/PlaybackEngines/AndroidPlayer\SDK
- ✓ Android NDK Installed with Unity (recommended)
  NDK C:/Program Files/Unity/Hub/Editor/2019.3.0f6/Editor/Data/PlaybackEngines/AndroidPlayer/NDK
- ✓ Gradle Installed with Unity (recommended)
  Gradle C:/Program Files/Unity/Hub/Editor/2019.3.0f6/Editor/Data/PlaybackEngines/AndroidPlayer/Tools/gradle
- ✓ Stop Gradle daemons on exit

Maximum JVM heap size, Mbytes: 4096

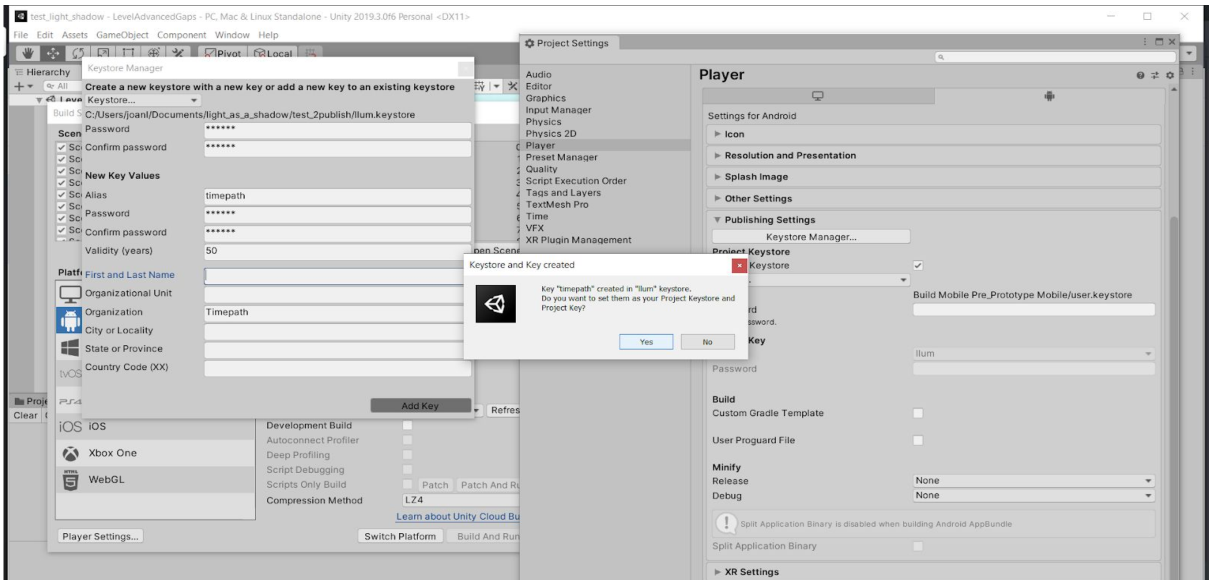
Keystores Dedicated Location: C:/Users/joanl

Notice the absence of warnings.

2. Inside Unity3D, you go to File > Build Settings, and select   Android
3. Inside the Build Settings of Unity3D, you go to Player Settings > Player > Publish Settings

4. At this stage you need to create a new Keystore. This is what you will need to sign the app in order to publish it.

When you create a keystore, you will need to provide a master password, plus a user, with another password for the user.   It will look like this:



Keystore Manager

Create a new keystore with a new key or add a new key to an existing keystore

Keystore...

C:/Users/joanl/Documents/light_as_a_shadow/test_2publish/llum.keystore

Password	******
Confirm password	******

New Key Values

Alias	timepath
Password	******
Confirm password	******
Validity (years)	50
First and Last Name	
Organizational Unit	
Organization	Timepath
City or Locality	
State or Province	
Country Code (XX)	

Add Key

Keystore and Key created

Key "timepath" created in "llum" keystore.
Do you want to set them as your Project Keystore and Project Key?

Yes No

Build Mobile Pre_Prototype Mobile/user.keystore

Split Application Binary is disabled when building Android AppBundle

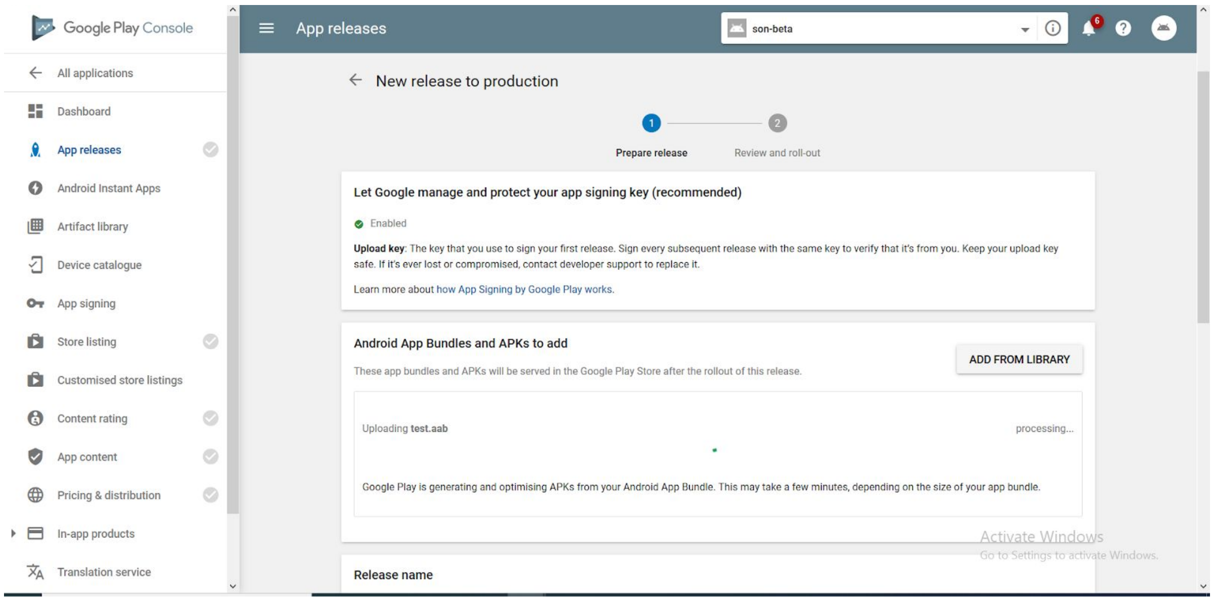
Now, the main challenge for Unity3D developers is find out how to add the certificate that is provided by Android Play.

On the Android Play console there are several ways described. This can be done with Android Studio, with a tool called PEPK, or others. See, for example:
https://forum.unity.com/threads/android-app-bundle-google-play-app-signing-what-option-do-we-use-with-unity.604723/
https://www.youtube.com/watch?v=Ks1U0mT0WH0

I did not manage to use this method. The simplest way I found is detailed here:
https://answers.unity.com/questions/1372982/how-i-can-sign-my-apk-with-an-google-plays-upload.html

I now explain in further detail how to do this:

First, in google console, when asked about the way you do the signing key, use the default method, letting google manage and protect your app signing key (see screenshot)



Let Google manage and protect your app signing key (recommended)

Enabled

Upload key: The key that you use to sign your first release. Sign every subsequent release with the same key to verify that it's from you. Keep your upload key safe. If it's ever lost or compromised, contact developer support to replace it.

Learn more about how App Signing by Google Play works.

Android App Bundles and APKs to add

These app bundles and APKs will be served in the Google Play Store after the rollout of this release.

Uploading test.aab processing...

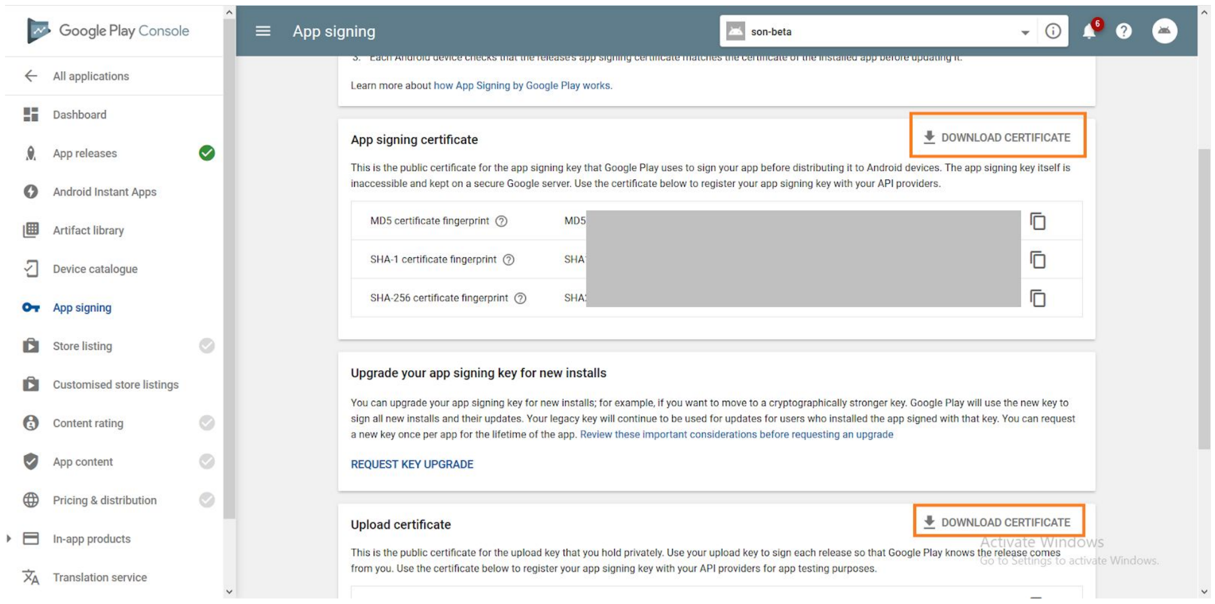
Google Play is generating and optimising APKs from your Android App Bundle. This may take a few minutes, depending on the size of your app bundle.

Release name

Once this is done, you will need to do this:

5. You need to download your certificate from your google play console.

In your google play console, in the App signing section, you can download two certificates, the signing one, and the upload one (see below).


Google Play Console

All applications

Dashboard

App releases

Android Instant Apps

Artifact library

Device catalogue

App signing

Store listing

Customised store listings

Content rating

App content

Pricing & distribution

In-app products

Translation service

App signing

son-beta

3. Each Android device checks that the release's app signing certificate matches the certificate of the installed app before updating it.

Learn more about how App Signing by Google Play works.

App signing certificate

DOWNLOAD CERTIFICATE

This is the public certificate for the app signing key that Google Play uses to sign your app before distributing it to Android devices. The app signing key itself is inaccessible and kept on a secure Google server. Use the certificate below to register your app signing key with your API providers.

MD5 certificate fingerprint	MD5 [redacted]
SHA-1 certificate fingerprint	SHA
SHA-256 certificate fingerprint	SHA

Upgrade your app signing key for new installs

You can upgrade your app signing key for new installs; for example, if you want to move to a cryptographically stronger key. Google Play will use the new key to sign all new installs and their updates. Your legacy key will continue to be used for updates for users who installed the app signed with that key. You can request a new key once per app for the lifetime of the app. Review these important considerations before requesting an upgrade

REQUEST KEY UPGRADE

Upload certificate

DOWNLOAD CERTIFICATE

This is the public certificate for the upload key that you hold privately. Use your upload key to sign each release so that Google Play knows the release comes from you. Use the certificate below to register your app signing key with your API providers for app testing purposes.

Activate Windows
Go to Settings to activate Windows.

NOTE: it gives errors, the only way is to close and reopen the application, after setting up the keystore configuration.

I downloaded both, but only used the one called “deployment_cert.der”

6. Once you have your certification, you need to import it into the keystore you used the first time you exported from Unity. You probably saved that somewhere important.

7. For this, you need to use a program called keytool.exe. The path to that program probably looks something like:
   C:\Program Files\Java\jdk1.8.0_192\bin
   or
   C:\Program Files\Java\jre1.8.0_241\bin

   If you do not have such files, it means you need to install Java.
   https://www.java.com/en/download/

8. If you have those folders, you can go there in windows explorer and type "cmd" into the address bar, it will open that location in the terminal.

9. You then need to use the following code in the terminal (make sure you are at the location of keystore.exe):

```
keytool.exe -importcert -file upload_cert.der -keystore <keystorefile>
```

To do this, you need to:

a. point that code to your certificate and to the keystore you're importing the ceritificate into. your final instruction will look something like the following:

b. Write in the command line the following instruction:

```
keytool.exe -importcert -file
"C:\Users\MyAccountName\DesktopOrWherever\deployment_cert.der"
-keystore C:\Users\MyAccountName\DesktopOrWherever\user.keystore
```

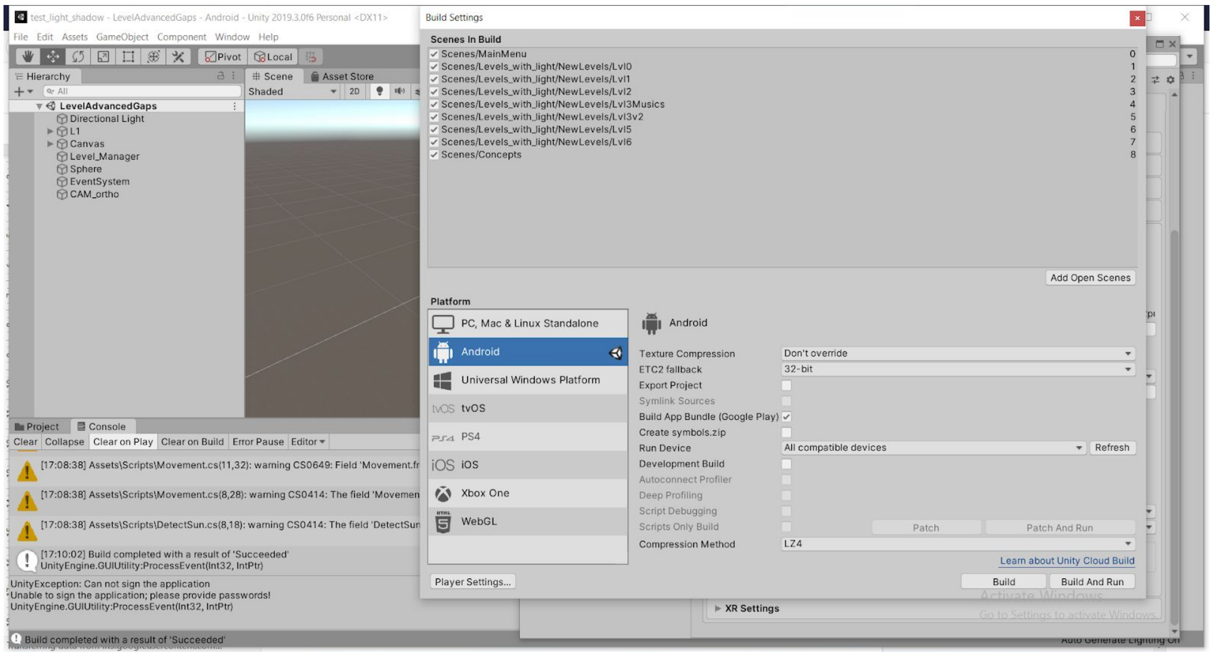
In the example of the screenshots, this is how it looks like:

```
C:\Program Files\Java\jre1.8.0_241\bin>keytool.exe -importcert -file
"C:\Users\joanl\Documents\light_as_a_shadow\test_2publish\upload_cert.der"  -keystore
C:\Users\joanl\Documents\light_as_a_shadow\test_2publish\son.keystore
```

(Notice that in command line ctrl+v does not paste, if you want to copy and paste then to use paste you can use a mouse button, it works with simply a right click)

10. keystore.exe will ask you for the password you used and whether you trust the app. Then it will update the keystore

11. Now you can go back to Unity, select again the Keystore to create a bundle (.aab). Notice that for quick testing of your app it is easier to create an android build (.apk), but for android play the bundle is simpler.



```
[17:08:38] Assets\Scripts\Movement.cs(11,32): warning CS0649: Field 'Movement.fr
[17:08:38] Assets\Scripts\Movement.cs(8,28): warning CS0414: The field 'Movemen
[17:08:38] Assets\Scripts\DetectSun.cs(8,18): warning CS0414: The field 'DetectSun
[17:10:02] Build completed with a result of 'Succeeded'
UnityEngine.GUIUtility:ProcessEvent(Int32, IntPtr)
UnityException: Can not sign the application
Unable to sign the application; please provide passwords!
UnityEngine.GUIUtility:ProcessEvent(Int32, IntPtr)
Build completed with a result of 'Succeeded'
```

Scenes In Build

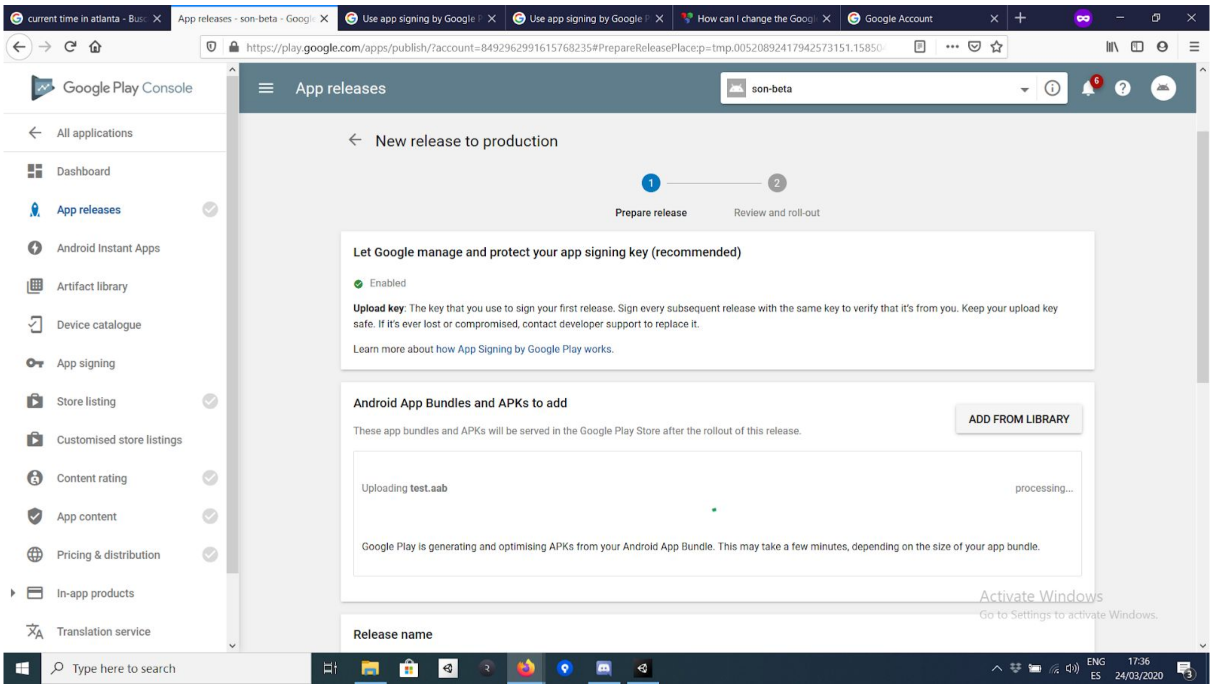
✓ Scenes/MainMenu	0
✓ Scenes/Levels_with_light/NewLevels/Lvl0	1
✓ Scenes/Levels_with_light/NewLevels/Lvl1	2
✓ Scenes/Levels_with_light/NewLevels/Lvl2	3
✓ Scenes/Levels_with_light/NewLevels/Lvl3Musics	4
✓ Scenes/Levels_with_light/NewLevels/Lvl3v2	5
✓ Scenes/Levels_with_light/NewLevels/Lvl5	6
✓ Scenes/Levels_with_light/NewLevels/Lvl6	7
✓ Scenes/Concepts	8

Android

Texture Compression	Don't override
ETC2 fallback	32-bit
Export Project	☐
Symlink Sources	☐
Build App Bundle (Google Play)	✓
Create symbols.zip	☐
Run Device	All compatible devices
Development Build	☐
Autoconnect Profiler	☐
Deep Profiling	☐
Script Debugging	☐
Scripts Only Build	☐
Compression Method	LZ4

Notice that it is very important that you use the keystore with the certificate added when you build your app, otherwise it will not be accepted by the Android apple store.

Then, you upload to the .aab, fill in all the information until the checks in the different snippets found on the left are in green (by default they will appear in grey), and then you will be able to publish.



https://play.google.com/apps/publish/?account=8492962991615768235#PrepareReleasePlace:p=tmp.00520892417942573151.15850

New release to production

1 Prepare release — 2 Review and roll-out

Let Google manage and protect your app signing key (recommended)

Enabled

Upload key: The key that you use to sign your first release. Sign every subsequent release with the same key to verify that it's from you. Keep your upload key safe. If it's ever lost or compromised, contact developer support to replace it.

Learn more about how App Signing by Google Play works.

Android App Bundles and APKs to add

These app bundles and APKs will be served in the Google Play Store after the rollout of this release.

ADD FROM LIBRARY

Uploading **test.aab** processing...

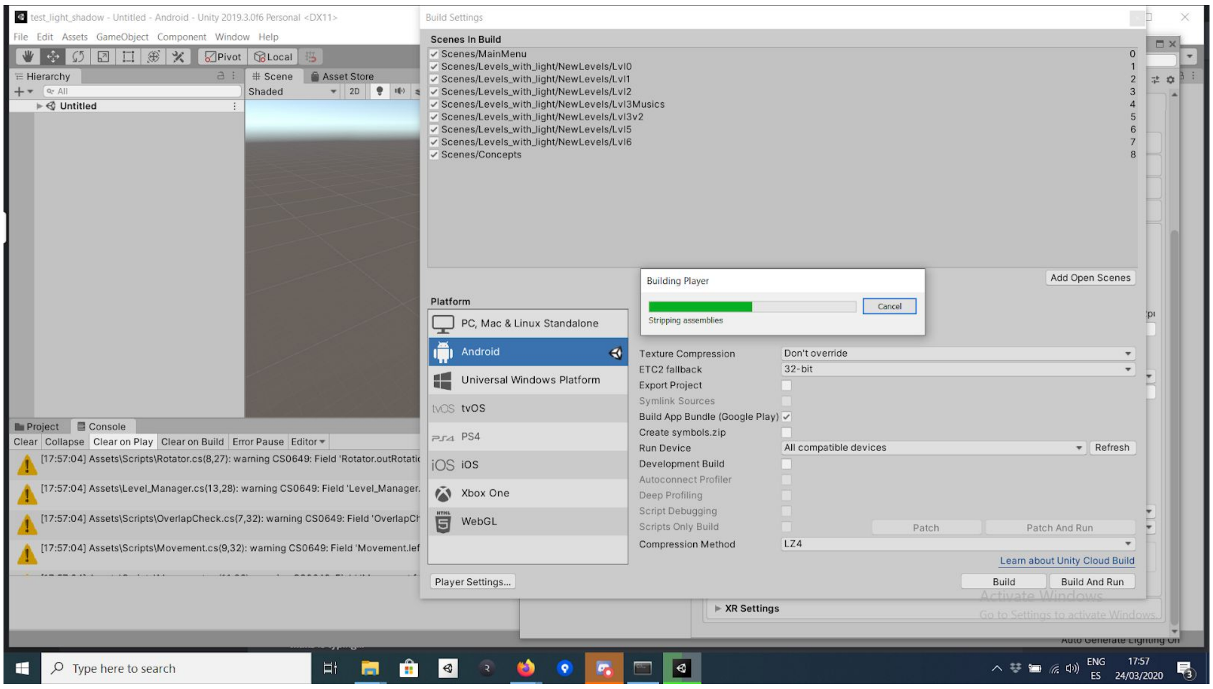
Google Play is generating and optimising APKs from your Android App Bundle. This may take a few minutes, depending on the size of your app bundle.

Activate Windows
Go to Settings to activate Windows.

Release name

changelog:

certificates:



test_light_shadow - Untitled - Android - Unity 2019.3.0f6 Personal <DX11>

Build Settings

Scenes In Build

```
✓ Scenes/MainMenu                                  0
✓ Scenes/Levels_with_light/NewLevels/Lvl0          1
✓ Scenes/Levels_with_light/NewLevels/Lvl1          2
✓ Scenes/Levels_with_light/NewLevels/Lvl2          3
✓ Scenes/Levels_with_light/NewLevels/Lvl3Musics    4
✓ Scenes/Levels_with_light/NewLevels/Lvl3v2        5
✓ Scenes/Levels_with_light/NewLevels/Lvl5          6
✓ Scenes/Levels_with_light/NewLevels/Lvl6          7
✓ Scenes/Concepts                                  8
```

Building Player

Stripping assemblies

Platform: Android

Texture Compression	Don't override
ETC2 fallback	32-bit
Export Project	☐
Symlink Sources	☐
Build App Bundle (Google Play)	✓
Create symbols.zip	☐
Run Device	All compatible devices
Development Build	☐
Autoconnect Profiler	☐
Deep Profiling	☐
Script Debugging	☐
Scripts Only Build	☐
Compression Method	LZ4

```
[17:57:04] Assets\Scripts\Rotator.cs(8,27): warning CS0649: Field 'Rotator.outRotati
[17:57:04] Assets\Level_Manager.cs(13,28): warning CS0649: Field 'Level_Manager.
[17:57:04] Assets\Scripts\OverlapCheck.cs(7,32): warning CS0649: Field 'OverlapCh
[17:57:04] Assets\Scripts\Movement.cs(9,32): warning CS0649: Field 'Movement.lef
```