Building for Android quick tutorial: https://youtu.be/Ska81xpB-Po

Unity android documentation link.

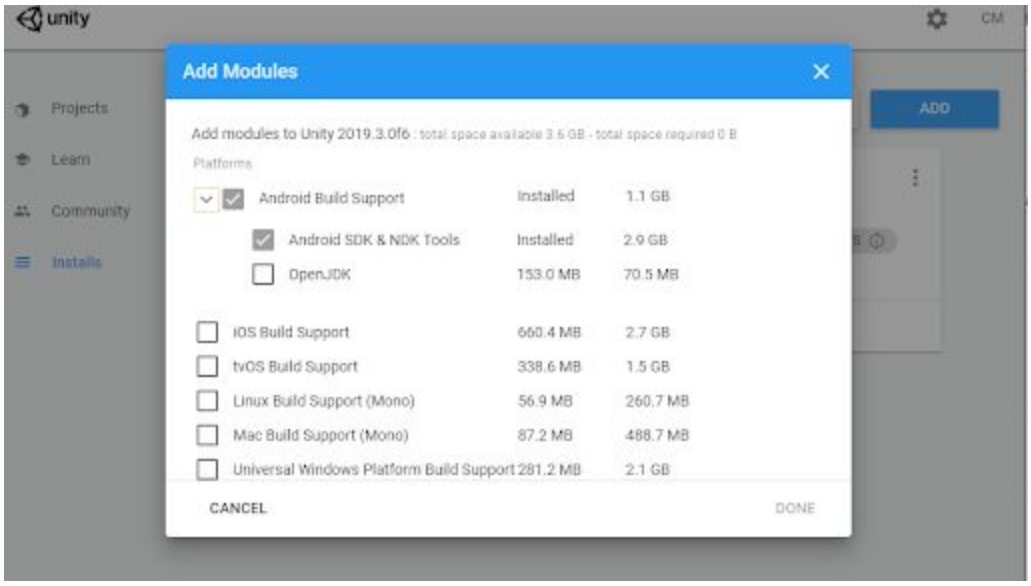
Android environment setup

To build and run for Android, you must install the Unity **Android Build Support** platform module. You also need to install the Android Software Development Kit (SDK) and the Native Development Kit (NDK) to build and run any code on your Android device.

Note: Unity supports Android 4.4 “KitKat” and above. See AndroidSdkVersions for details.

Install Android Build Support and the Android SDK & NDK tools

Use the Unity Hub to install **Android Build Support**, make sure to also include **Android SDK & NDK Tools**



You can install **Android Build Support** when you install the Unity Editor, or at a later time.

For information on adding the Android modules:

- At install time, see Installing Unity.

- To an existing installation, see Adding modules to the Editor.

If you are using a 2018 version of Unity, see the Unity 2018.4 documentation for information on manually installing these dependencies.

Enable USB debugging on your device

To enable USB debugging, you must enable Developer options on your device. To do this, find the build number in your device's **Settings** menu. The location of the build number varies between devices; for stock Android, it's usually **Settings > About phone > Build number**. For specific information on your device and Android version, refer to your hardware manufacturer.

After you navigate to the build number using the instructions above, tap on the build number seven times. A pop-up notification saying "You are now X steps away from being a developer" appears, with "X" being a number that counts down with every additional tap. On the seventh tap, Developer options are unlocked.

Note: On Android versions prior to 4.2 (Jelly Bean), the Developer options are enabled by default.

Go to **Settings** > **Developer options** (or, if this does not work, on some devices the path is **Settings** > **System** > **Developer options**), and check the **USB debugging** checkbox. Android now enters debug mode when it is connected to a computer via USB.

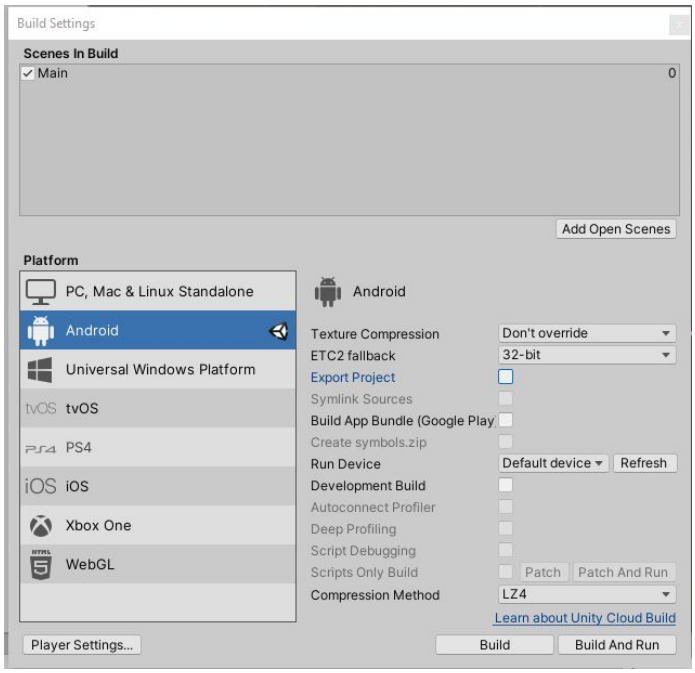
Connect your device to your computer using a USB cable. If you are developing on a Windows computer, you might need to install a device-specific USB driver. See the manufacturer website for your device for additional information.

Building apps for Android

There are two locations to configure settings that affect how your app is built:

- Player settings - Allows you to configure runtime settings for the app. For more information on Player settings, see Player settings for the Android platform.

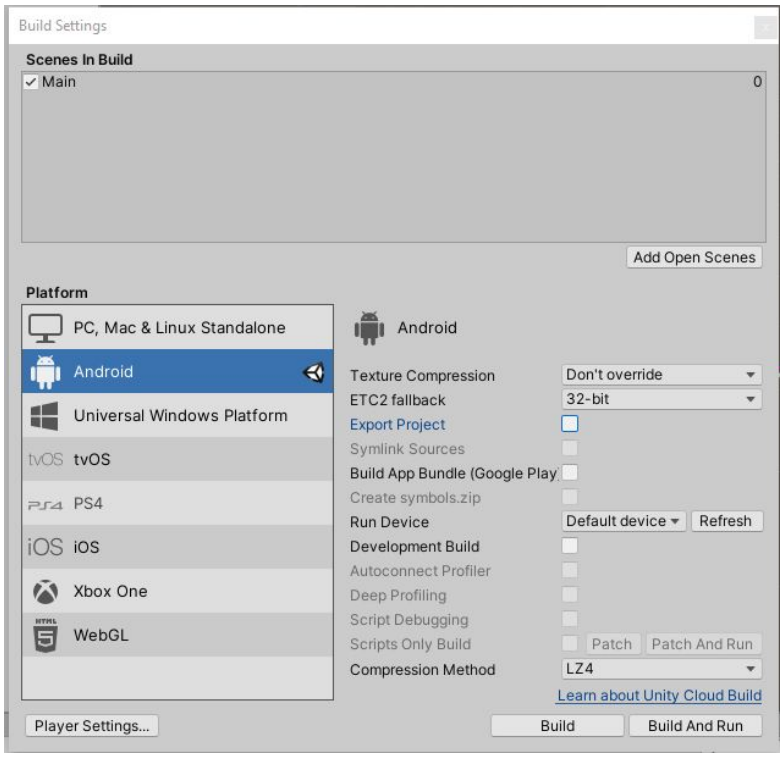
- Build settings - Allows you to configure build system parameters and build the app.



The output package includes an **APK**, and an **APK** expansion file (OBB) if the **Split Application Binary** option is selected in the Player settings. For more information on OBB files, see OBB Support.

Configuring Build Settings

To configure and build apps for Android, access the Build Settings window (**File** > **Build Settings**). In **Platform**, select **Android**.

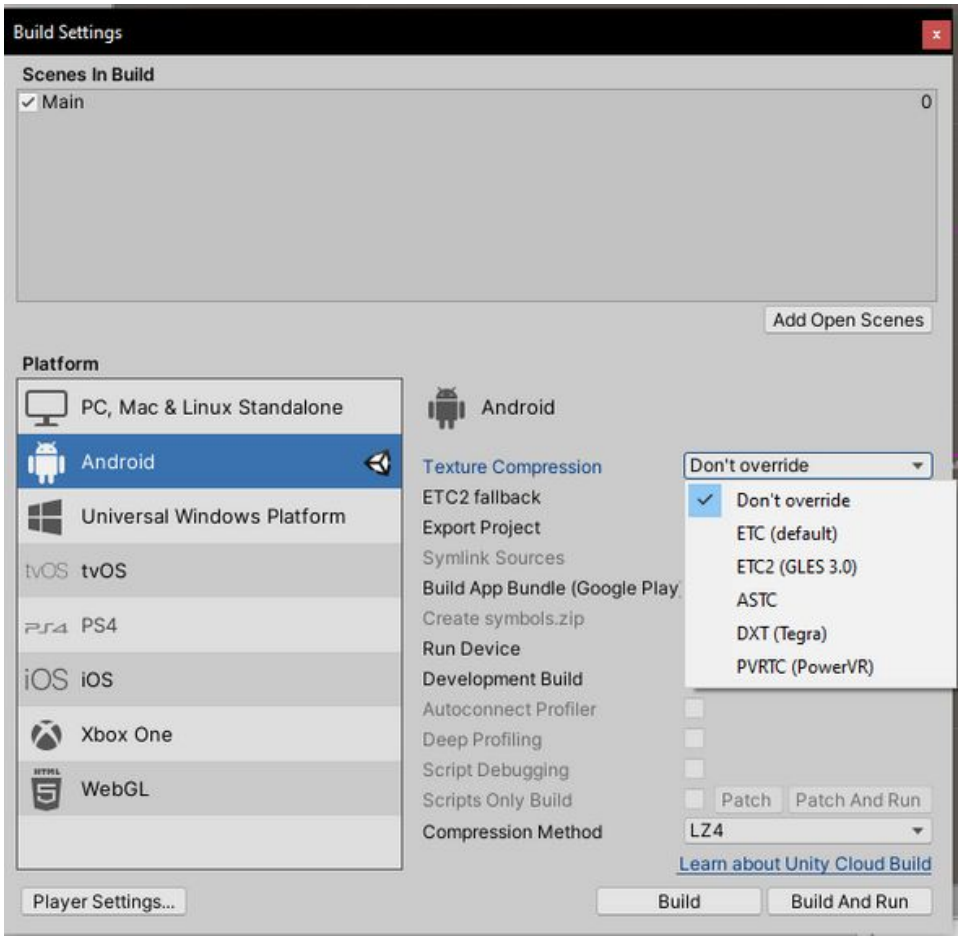


To set Android as your default build platform, click the **Switch Platform** button.

After you specify your build settings, click the **Build** button to create your build. To build the app, click **Build And Run** to create and run your build on the platform you have specified.

Texture compression

Unity uses the Ericsson Texture Compression (ETC) format for textures that don't have individual **texture format** overrides. When building an APK to target specific hardware, use the **Texture Compression** option to override this default behavior. **Texture Compression** is a global setting for the Project. If a texture has a specific override on it, that texture is not affected by the **Texture Compression** setting. For additional information, see Textures

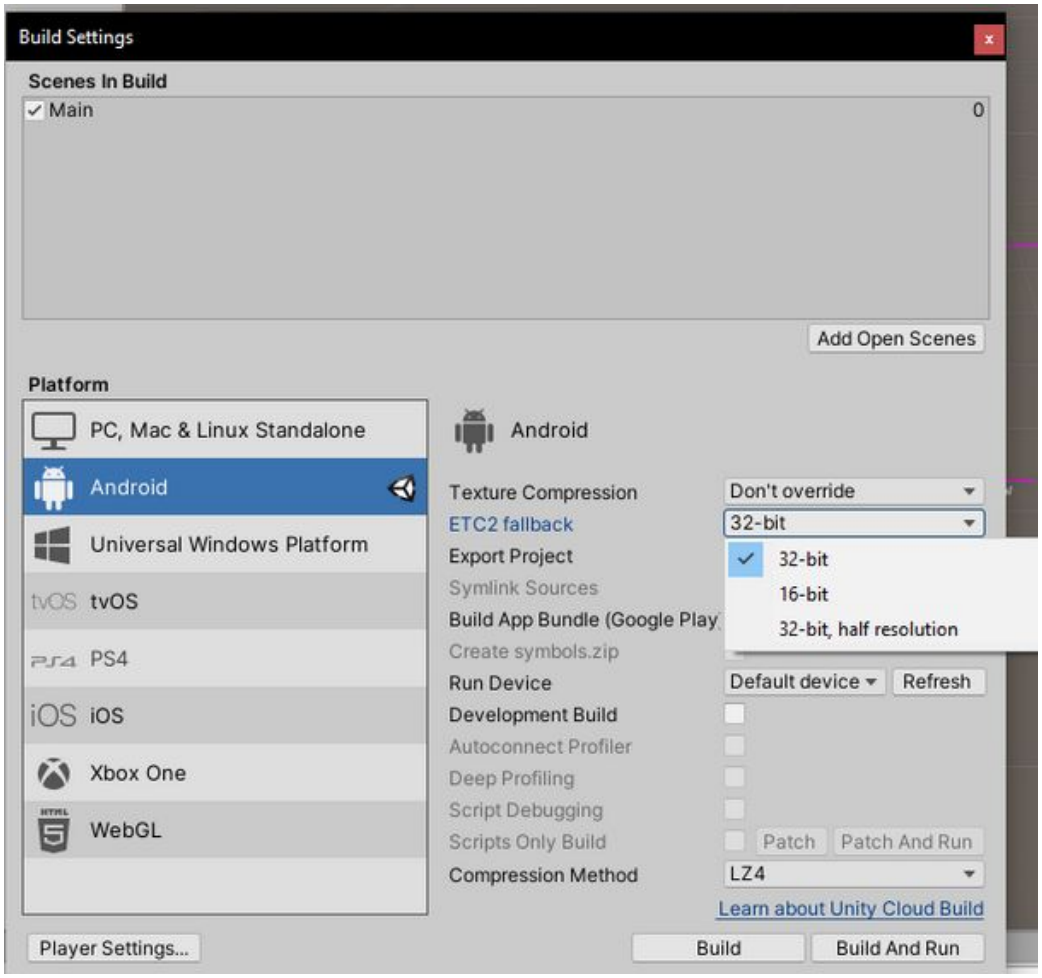


For additional information on textures and texture compression, see the Mobile and WebGL specific formats and Notes on Android sections on the Texture compression formats for platform-specific overrides page.

Note: **Texture Compression** is a global setting. Individual textures override the global setting.

ETC2 fallback

For Android devices that don't support ETC2 (which don't support GL ES3), you can override the default ETC2 texture decompression by choosing from 32-bit, 16-bit, or 32-bit with half the resolution formats.



This option allows you to choose between the uncompressed image quality and the amount of memory the uncompressed texture occupies. 32-bit RGBA texture is the highest quality format, and takes twice the required disk space as the 16-bit format, but a texture in 16-bit might lose some valuable color information. 32-bit half-resolution reduces the memory requirement further, but the texture is likely to become blurry.

Build system

Unity supports two Android build systems: **Gradle**

and **Internal**.

The steps involved with building for Android are:

- Preparing and building the Unity Assets.
- Compiling scripts.
- Processing the **plug-ins**

  .

- Splitting the resources into the parts that go to the APK and the OBB, if **Split Application Binary** is selected.
- Building the Android resources using the AAPT utility (internal build only.)
- Generating the Android manifest.
- Merging the library manifests into the Android manifest (internal build only.)
- Compiling the Java code into the Dalvik Executable format (DEX) (internal build only.)
- Building the IL2CPP library, if **IL2CPP Scripting Backend** is selected.
- Building and optimizing the APK and OBB packages.

Gradle build system

The Gradle build system uses Gradle to build an APK or export a Project in Gradle format, which can then be imported to Android Studio. When you select this build system, Unity goes through the same steps as the **Internal** build system excluding resource compilation with AAPT, merging manifests, and running DEX. Unity then generates the *build.gradle* file (along with the other required configuration files) and invokes the Gradle executable, passing it the task name and the working directory. Finally, the APK is built by Gradle.

For more details, see Gradle for Android.

Internal build system

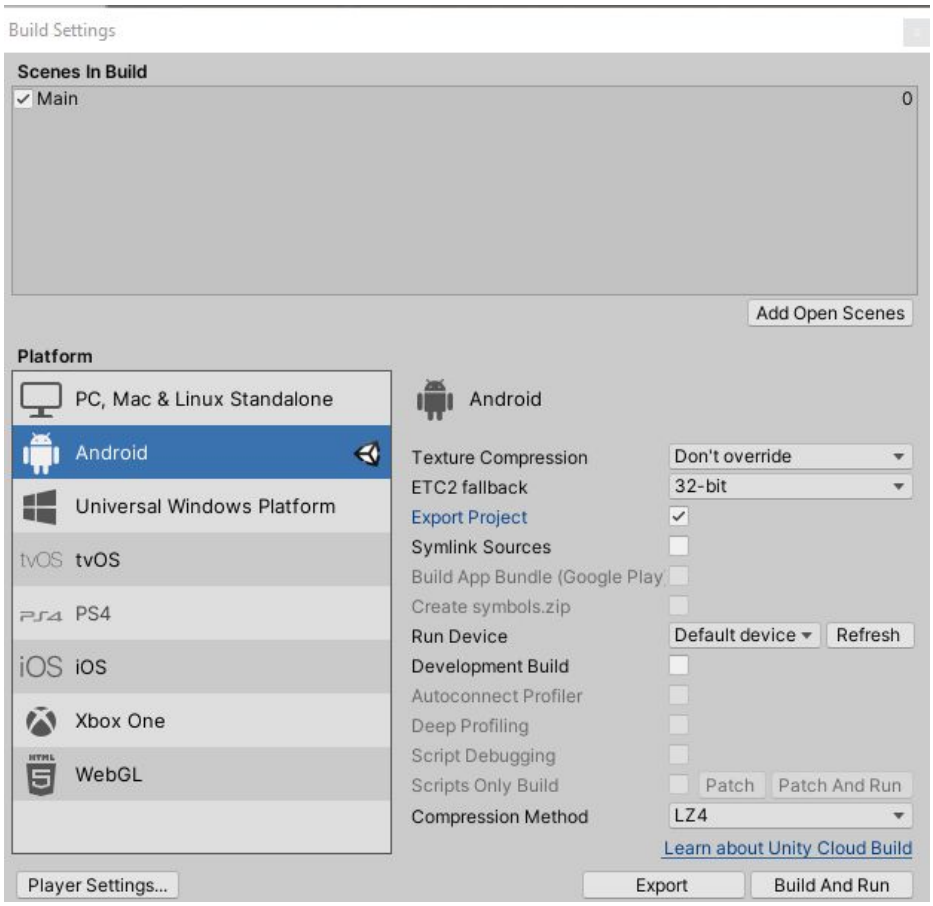
The **Internal** build system creates an APK using the Android SDK utilities to build and optimize the APK and OBB packages. For more information about OBB files, see OBB Support.

Exporting the Project

If you need more control over the build pipeline, or to make changes that Unity does not normally allow (for example, fine tuning the manifest files that are automatically generated by Unity), you can export your Project and import it into Android Studio. Exporting a Project is only available when you have selected **Gradle** as your **Build System**.

To export the Project:

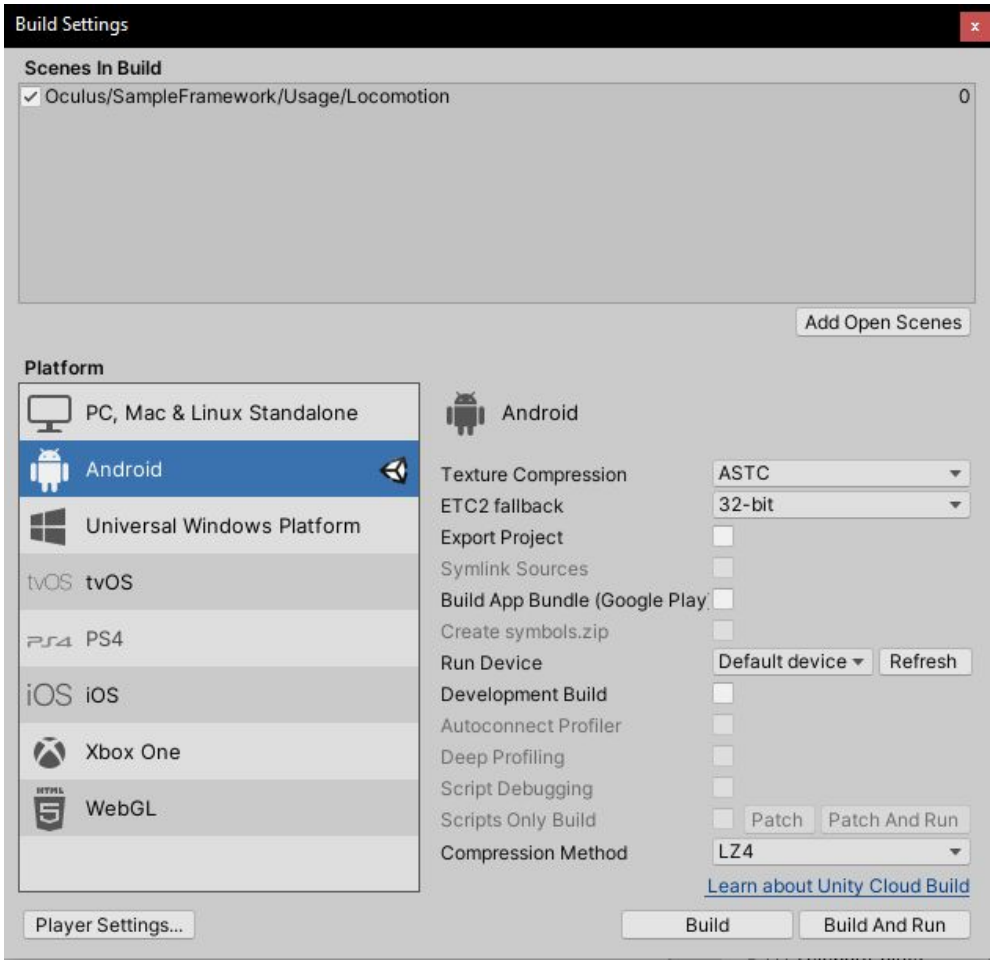
1. From the **Build System** drop-down menu, select **Gradle**.
2. Check the **Export Project** checkbox. When **Export Project** is checked, the **Build** button is relabeled **Export** and the **Build And Run** button is disabled.
3. Click the **Export** button and select a destination folder for the project.



When the export finishes, open Android Studio and import your project. For more information on importing projects to Android Studio, see the Migrate to Android Studio section of the Android Developer documentation.

Build or Build and Run

The **Build Settings** window offers two options: **Build** and **Build and Run**. Using either option saves the output packages (APK and OBB, if enabled) to the path that you select. Selecting **Build and Run** saves the output packages to the file path you specify, while also installing your app on the Android device connected to your computer.



Tip: Specify the output path for the packages and then use the **Ctrl+B** (Windows) or **Cmd+B** (macOS) keyboard shortcut to **Build and Run** using the saved output path.