Unity 5 for Android Essentials

Unity is a very popular and effective technology for creating 2D and 3D games and applications. Unity 5 is a great starting point for game developers looking to develop stunning and robust games.

We begin by explaining how to set up the Android SDK on Windows and Mac OS X and configure Unity 5 settings for the Android platform. As you progress through the chapters, you will learn to implement new and improved features of the Unity 5 engine with the aid of real-world examples. You will explore how to render high-quality graphics with physically-based shaders and global illumination to enhance your project's performance. Building on this, you will then learn to transform your native C# and JavaScript code into Unity scripts. Best practices to improve your Android games will also be discussed to help you create games fast and efficiently. Finally, putting together all these concepts, you will learn to create your own Android game from scratch. This book will teach you how to harness the benefits of different tools to become proficient at game design and development processes.

Who this book is written for

This book is perfect for game developers who want to learn how to develop, optimize, and publish games for Android devices in a quick and easy manner using Unity 5. This book assumes a basic knowledge of game design concepts and/or some experience with other game technologies such as Unreal Engine 4, CryEngine, or GameMaker.
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 'Setting Up and Configuring an Android Platform'
- A synopsis of the book’s content
- More information on Unity 5 for Android Essentials
Valera Cogut is an independent professional software and video game developer who has worked in the game industry since 2008 and has more than 12 years of programming experience. He is a passionate software and game developer with different areas of expertise. Before diving into the game industry 6 years ago, Valera created websites and applications with PHP, the Yii framework, the Zend framework, Relational Database Management Systems, Apache and Nginx, C#, C++, C, Objective-C, Java, Python, UML, and many other technologies. Having a mathematical background (analytical geometry, linear algebra, logic, statistics and probability, differential equations, graph theory, and mathematical analysis), he finally realized that game development was his mission. Reusable designs, optimized algorithms, clean code, and elaborated workflows—these things make him happy.

Valera has had the opportunity to produce titles for multiple platforms, including Windows Phone, Android, iOS, PC, and Mac. Today, he continues to produce fun and original games, participate in game jams, and author books. He was a technical reviewer for Unity Android Game Development by Example Beginner's Guide, Packt Publishing.

Preface

Learn the Unity 5 engine to master designing and building awesome real-world games and applications for Android devices. Design beautiful effects, animations, physical behaviors, and other different real-world features and techniques for your Android games and applications. Optimize your project and any other real-world projects for Android devices. Know more in practice about accessing Android functionality, rendering high-end graphics, expanding your project using asset bundles, and of course learn about deploying on the Android platform and much more.

Unity is essentially a development environment for multiple platforms (Android, iOS, Blackberry, Windows Phone, Windows, Playstation, Xbox, Mac, Linux, and Web). Unity is a very popular and effective technology for creating 2D and 3D games and applications. This technology provides many useful and effective tools to solve various issues. The Unity rendering engine provides great real-time rendering of high-quality graphics without too much cost and effort.

The Android platform and game industry is developing like never before. Most programmers using Unity, especially those new to the technology, would like to learn to recreate different functional parts from the most popular real-world games and applications that will cause a great concern in the community.

Whether you are new to Unity 5 or an expert, this book will provide you with the required skills you need to successfully design, implement, build, and enhance the quality of your Android game or application. By sequentially working through the steps in each chapter, you will quickly master the key features of the Unity 5 engine to implement real-world Android games and application features in practice.

This book is aimed for competent Unity developers who want to learn to develop, optimize, and publish games for Android devices. Know more about the Unity 5 engine to build awesome real-world Android games.
Starting from the beginning, this book will explain the configuration of the Unity 5 settings for the Android platform. Implement the innovative and user-friendly features using real-world techniques and tips and tricks in practice. Explore how to improve your Android games and enhance its performance. Open a wonderful world of rendering high-quality graphics with physically-based shaders and global illumination. Discover Android features inside the Unity 5 engine and more about transformation from native C# and JavaScript codes into Unity scripts.

This book is aimed at covering the fundamentals relating to typical real-world games and applications and to explore a basic overview of the concepts but the focus is on providing the practical skills required to develop Android games and applications. *Unity 5 for Android Essentials* will teach you to use different tools provided by the Unity Technologies team in order to master your game's design and development processes. This book will be a practical guidebook that will help you to leverage the Unity 5 framework in order to build awesome games and apps for Android devices.

**What this book covers**

*Chapter 1, Setting Up and Configuring an Android Platform*, explains how to configure Unity 5 for Android devices. Also in this chapter, we will explore about APK expansion files in Unity 5. At the end of the chapter, you will build a very simple and basic game example for Android devices.

*Chapter 2, Accessing Android Functionality*, covers how to create plugins for the Android platform using Java and C languages in Unity 5. The reader will learn in practice how to write simple plugins for the Android platform. Also, the reader will learn to make anti-piracy checks, detect screen orientation, handle vibration support, determine device generation, and do more useful things.

*Chapter 3, High-end Graphics for Android Devices*, primarily explores how to enhance the quality in games and applications using physically-based shaders. This chapter will also describe global illumination in Unity 5. At the end of the chapter, you will optimize a shader code.

*Chapter 4, Animation, Audio, Physics, and Particle Systems in Unity 5*, will cover new Mecanim animation features in Unity 5. After that, you will learn awesome new audio features in Unity 5. At the end of this chapter, you will explore physics and particle systems in Unity 5.
Chapter 5, Asset Bundles in Unity 5 Pro, includes an overview of asset bundles in Unity 5. You will learn to download new code and data in real time for Android devices. At the end of this chapter, you will discover the safety technique of the asset bundles in practice.

Chapter 6, Optimization and Transformation Techniques, introduces the usage of occlusion culling and optimization techniques for the level of detail. You will learn to optimize Unity C# and Unity JS code. Finally, you will see how to transform Unity C# code to Unity JavaScript code and vice versa.

Chapter 7, Troubleshooting and Best Practices, covers the optimization of any game for Android devices and teaches you to find any bottlenecks. In this chapter, you will discover some troubleshooting techniques for the Android platform. At the end of this chapter, you will learn the best practices that are used by many professionals from all over the world for their scripts and shaders.

The online chapter, Developing Glow Hockey from Scratch, shows you how easy it is to develop the most popular game in the Android Market (Glow Hockey has about 100,000,000–500,000,000 installs https://play.google.com/store/apps/details?id=com.natenai.glowhockey&hl=en) in Unity 5 from scratch. Also, you will learn to optimize your project and any other real-world projects for Android devices. Many more useful details and features will be covered in this chapter. You can find this chapter at https://www.packtpub.com/sites/default/files/downloads/9191OT_BonusChapter.pdf.
1

Setting Up and Configuring an Android Platform

This chapter will talk about installing the Android SDK on Windows and Mac OS X platforms. Further, the reader will find how to configure Unity 5 for Android devices. Also, in this chapter, we will explore the APK expansion files in Unity 5. In this chapter, the reader will build the Glow Hockey project (we will create this game in Unity 5 from scratch in the last chapter in this book) on an Android device. At the end of this chapter, the reader will explore the side-by-side comparisons of the Unity Pro and Unity Basic-specific features and rules.

The topics that will be covered in this chapter are as follows:

- Configuring Unity 5 for Android devices
- APK expansion files in Unity 5
- Building for Android devices
- Unity license comparison overview

Configuring Unity 5 for Android devices

Once you have installed the Android SDK and set up Unity, you must configure the correct settings for each of your Android project. We will begin our review by considering the Resolution And Presentation options as shown in the following screenshot. In order to access the Android platform settings in Unity, it is necessary to navigate to the Edit | Project Settings | Player menu and then click on the button with the Android icon. Also, the other way of getting to the Android platform settings is by navigating to File | Build Settings. After opening the window, you will need to click on the Player Settings button at the bottom.
The **Default Orientation** option is shared between multiple mobile platforms. This setting is necessary in order to indicate the orientation of the screen which is designed for your game or your application. The default is **Automatic Rotation** for all settings. If, for example, your project is designed only for the portrait orientation of the screen, then you need to select either the **Portrait** or **Portrait Upside Down** value:

![Settings for Android](image)

We can pick one of the following properties: **Portrait**, **Portrait Upside Down**, **Landscape Right**, **Landscape Left**, and **Auto Rotation**. They are pretty plain and speak for themselves. You just need to play a little with them to see their true purpose. The **Status Bar Hidden** checkbox needs no explanation because its meaning is obvious.

A subsequent option that we will survey is known as **Use 32-bit Display Buffer**. You can decide if the **display buffer** should handle 16-bit color values (if not 32-bit enabled) or if it should handle 32-bit color values. Remember that you need to activate this setting only if you have some artifacts, because it greatly affects the performance. The **Show Loading Indicator** field ensures the ensuing behaviors for us: **Don't Show**, **Large**, **Inversed Large**, **Small**, and **Inversed Small**.
As shown in the preceding figure, there are a lot of settings; however, most of them can be left with default values except for a few that must be adjusted before making a build. You cannot build an APK file for your Android device without configuring the Bundle Identifier option, which is shared between mobile platforms. The Bundle Identifier string must match the provisioning profile of the game you are building. The fundamental structure of the identifier is `com.Company.ProductName`. The bundle version is responsible for holding the number that describes the release index. Moreover, Unity permits us to specify the minimum API level that will be supported by your Android project. Also, you can set the name and icon for your application. Other settings are obvious and do not require additional explanation. More detailed information can be found in the official documentation of Unity.
APK expansion files in Unity 5
Google Play requires that the size of your games and applications does not exceed 50 MB. For most applications and games, this size is enough. Alternatively, you may want to have fantastic graphics for your projects and other huge media files that can take up a lot of space. Google Play makes a developer's life simpler and easier by expanding an APK file by large expansion files. Extension files are kept in a shared folder storage of the device where your game has enough access rights.

Overview
Each file cannot be larger than 2 GB, and you can choose any format for it. Of course, the best way is to use only compressed files in order to conserve bandwidth while the downloading process is active. You can add only one or two expansion files to your APK. Behind each expansion file lies its meaning:

- The first expansion file is known as the main and should be used for extra resources needed in your game. That's why this expansion file is primary.
- The second expansion file is known as patch, and is required to update the main file. That's why it is optional.

You should know that the Developer Console does not allow you to update your existing APK file by uploading only a new expansion file.

Formats
For your expansion files, you can use any desired format, such as MP3, MP4, AVI, RAR, ZIP, DOC, and PDF. The JOBB tool can help you encapsulate and encrypt your resources and patches for them.

The updating process
In most cases, Google Play will do all jobs automatically for you. So, very often, you don't need to do anything while users will download or upload your expansion files on their devices. However, sometimes your game has to download these files itself by receiving a URL from Google Play's application licensing service.

The basic steps to download expansion files for your game are listed as follows:

1. On the game start event, you should look for the expansion files in the Android/obb/<package-name>/ directory.
2. In the first step, if you will find that your expansion files are already in that directory, then you can continue playing your game further.
3. In the event that the expansion files are not in that directory, you should perform the next two steps.

4. You have to receive URLs, names, and sizes for your game’s expansion files. Before downloading anything, you should know where and what to download.

5. After having all the required information to download the expansion files, you can get your files and put them into the Android/obb/<package-name>/ directory with the same name as Google Play told you.

The following notes that are listed are taken from the official Android Developers Documentation page at http://developer.android.com/google/play/expansion-files.html:

- The URL that Google Play provides for your expansion files is unique for every download, and each one expires shortly after it is given to your application.
- Whether your application is free or not, Google Play returns the expansion file URLs only if the user acquired your application from Google Play.
- A variety of errors may occur during the request and download that you must gracefully handle.
- Network connectivity can change during the download, so you should handle such changes. If interrupted, resume the download when possible.
- While the download occurs in the background, you should provide a notification that indicates the download progress, notifies the user when it’s done, and takes the user back to your application when selected.

Setting up expansion files in Unity 5

Navigate to the Player Settings | Publishing Settings menu, and at the bottom you will see an option called Split Application Binary. When this option is enabled, your project will be divided into the .apk file for code, and for all other assets and data, it will be the .obb file.

Let's look at a list of key aspects related to the loading of the expansion files (.obb):

- The expansion files do not need to be uploaded to the server Google Play.
- If you have decided to publish .apk and .obb files on Google Play, then you need to include the code to download the expansion files.
In the Unity Asset Store, you can find a great plugin for downloading and extracting your expansion files in the right location. The URL of this plugin is http://u3d.as/content/unity-technologies/google-play-obb-downloader/2Qq.

Before testing the .obb files, you need to be logged in your Google account.

**Building for Android devices**

After creating a new project in Unity, it is a good idea to adjust the global quality settings as shown in the following figure. Most of them affect your game performance. Let's look deeper at *Quality Settings*, as it is a must before building applications for Android devices:

Unity allows you to create a template for your quality settings; you can also select one of the templates available in Unity by default. These settings greatly affect the performance of your application and the quality of your graphics. It is especially important for mobile platforms, where resources are very limited. You will need to play around with the settings on your target platforms to find the best template settings for your quality and performance. To access these settings, you need to navigate to *Edit | Project Settings | Quality*. You can select different templates separately for each platform supported by Unity. This setup window is divided into two main parts. The upper part, as shown in the preceding figure, is for managing templates, while the lower part, as shown in the following figure, is responsible for the settings themselves.
Each pattern (even Unity built-in templates) can be named as you wish. For all platforms supported by Unity, you can choose several accessible settings templates and also a default one. The default settings template is highlighted in green. Your settings should be as simple as possible, especially for mobile platforms. Unity allows you to create new settings templates and delete them by clicking on the icon with the basket.

The first part we intend to cover is Rendering as shown in the preceding screenshot. It contains the Pixel Light Count option that affects the upper limit pixel light only within Forward Rendering mode. The Forward Rendering path renders each object in one or more passes, depending on lights that affect the object. Lights themselves are also treated differently by Forward Rendering, depending on their settings and intensity.
The **Rendering** part contains the **Texture Quality** property with four existing options: **Full Res**, **Half Res**, **Quarter Res**, and **Eighth Res**. This lets you choose whether to display textures at maximum resolution or at a fraction of this (lower resolution has less processing overhead). Always remember, in any project you need to find a golden balance between the next two characteristics: quality and performance. The following property is named **Anisotropic Textures**, and it allows you to select just three values: **Disabled**, **Per Texture**, and **Forced On**. This describes if and how anisotropic textures will be used. On Wikipedia ([http://en.wikipedia.org/wiki/Anisotropic_filtering](http://en.wikipedia.org/wiki/Anisotropic_filtering)), we can read next about anisotropic filtering: "In 3D computer graphics, **anisotropic filtering** (abbreviated AF) is a method of enhancing the image quality of textures on surfaces of computer graphics that are at oblique viewing angles with respect to the camera where the projection of the texture (not the polygon or other primitive on which it is rendered) appears to be non-orthogonal (thus the origin of the word: "an" for not, "iso" for same, and "tropic" from tropism, relating to direction; anisotropic filtering does not filter the same in every direction)".

The next property you learn about is **anti-aliasing**. Anti-aliasing can be turned off by selecting the **Disabled** option or it can be turned on by selecting **2x**, **4x**, and **8x Multi Sampling** options. The next setting is just to toggle soft blending for particles, and its name is **Soft Particles**. This was the last option for the **Rendering** part.

The next part is **Shadows**, and the name absolutely describes itself and its purpose. We can choose one value from three open values: **Hard and Soft Shadows**, **Hard Shadows Only**, and **Disable Shadows**. A big processing overhead can result if you choose the highest resolution for the **Shadow Resolution** option. Possible settings are the following: **Low**, **Medium**, **High**, and **Very High**. There are two separate routines for anticipating shadows from a directional light within the Shadow Projection option. If we choose **Close Fit**, then it renders higher resolution shadows, which can in some cases wobble marginally if the camera moves. The next option, and the last one for **Shadow Projection**, is the **Stable Fit** value, which is the opposite of **Close Fit**. This means that **Stable Fit** renders a lower resolution but without any artifacts when the camera moves. Next is the **Shadow Cascades** setting, which has an effect on processing overhead. A higher cycle of cascades can cope with more processing overhead. Do not forget about the golden balance. Obtainable options for cascades are next: **No Cascades**, **Two Cascades**, and **Four Cascades**.

On mobile platforms, real-time shadows for directional lights always use one shadow cascade and are hard shadows. Directional lights are mostly used as a key light—sunlight or moonlight—in an outdoor game. Viewing distances can be huge, especially in first and third person games, and shadows often require some tuning to get the best quality versus performance balance for your situation.

The value of Shadow Distance is responsible for how far we can see shadows. Shadows beyond this length are visible, others are not.

In the following text, we are going to explore what is known as Other part. It contains five options for tuning any project. Let’s start with the first option—Blend Weights. We can choose just three values which are very important for performance within this setting. The lower the value, the higher the performance. This setting tells the number of bones that can affect a given vertex during an animation. We can select 1 Bone, 2 Bones, or 4 Bones; not more and not less. The next feature highly affects the performance and is not the first priority question about quality because of its hardly visible artifact. The name of this setting is VSync Count and the name of this artifact is known as tearing. If we want to avoid such artifacts then we need to synchronize rendering with the refresh rate of the display device, but do not forget about your performance. Synchronization can reduce a lot of your performance, so you should be ready for such kind of conditions. There are just three existing options for the VSync Count parameter: the first option is for synchronization with every vertical blank (VBlank), the second value is for synchronization with every second vertical blank, and the third option allows us to disable all synchronizations, thereby speeding up your application. The next setting in our research is LOD Bias. This value does its work only when Unity needs to decide which LOD level to choose. For example, when there is a choice between two LOD levels, LOD Bias comes to help by selecting only one value. This is set in a range from zero to one as a fraction. The closer to zero, the less detailed a level will be chosen and vice versa. There are two remaining options that we will now consider. The first is Maximum LOD Level, and its purpose is to remember the number for the highest LOD level that you can use in your project. The second is Particle RayCast Budget, which needs particle system collisions with Low and Medium qualities, and this number describes the highest value of ray cast for physics approximation.
Setting Up and Configuring an Android Platform

As for the **Maximum LOD level** parameter, all models in which the value will be less than this number will not be included in the build; Unity will ignore them, which can significantly reduce the amount of spending memory for your application or your game. The initial default value for this parameter is zero, which means that every model will be included in your build regardless of its level of detail. For each platform, depending on its configuration, Unity will use the smallest possible LOD level.

If the Android SDK installation and Unity setup are successful, you can safely create a build of your project. To do this, you will need to navigate to **File | Build Settings**, and in the opened window, you can create a build for various supported platforms as shown in the following figure. If you have properly installed the Android SDK and configured Android and have quality and player settings in Unity, you can safely click on the **Build** button or on the **Build And Run** button (if your Android device is properly configured and connected via USB) in the lower-right corner of the window.

Now, it's time to build Glow Hockey (we will create this game in Unity 5 from scratch in the last chapter in this book). First of all, you should create a new project in the Unity Editor. You can name it as you wish. To create a new project, you should click on the **Create Project** button in the bottom-right corner of the window.

The Unity Editor will be shown after creating the new project.

Glow Hockey is a very good example for the deployment of the project on the various platforms supported by Unity. In this game, there are many different effects, animations, sound effects, physics, and many other aspects from Unity. After deploying this project on an Android platform, you can test the various functions that are supported by Unity.

After opening the main **Glow Hockey** scene, you can make any changes or perform experiments as you wish. However, in this chapter, our goal is just to build this game on an Android device. We will not make any changes in the project in this chapter.
After opening the **Build Settings** window, you should select the **Android** platform, following which you can create an .apk file by clicking on the **Build** button in the bottom-right corner in order to share it with your friends, for example. Also, you can press the **Build And Run** button to export an .apk file as in the first case and to deploy this project right onto a connected device via the USB cable at the same time.
Now, let's discover more about Unity license comparison in the next section.

Unity License Comparison Overview

This section is based on the Unity License Comparison Overview. The link http://unity3d.com/unity/licenses will show you the side-by-side comparisons of the Unity Pro and Unity Basic's specific features and rules.

**NavMeshes, pathfinding, and crowd simulation**

Pathfinding is a built-in feature of Unity. This system allows you to easily find the right way from the beginning to the end, avoiding all the obstacles encountered on that way. All you have to do before you use this functionality is bake your navigation data in the Unity Editor. There, you must specify which floor objects or grounds can be walked on, and which objects are obstacles—all other problems will be solved by Unity without any effort after calling the pathfinding function with the start and end points as parameters. With a strong desire, you can create your own pathfinding system and crowd simulation. The pathfinding system in Unity is available for Basic and Pro licenses.
Level of Detail (LOD) support

Level Of Detail (LOD) allows you to optimize your productivity very well due to the fact that there are several different levels of quality for your mesh, each of which will be displayed in the camera view, depending on the distance to the camera. That is, when the camera is too far away, it is not optimal to display the most complex and detailed mesh, since all the power of all these details on the mesh will not be visible at all, which wastes precious resources. This is not very good for your performance as a whole. Detail mesh should be displayed only when the camera is close enough for you to see all these details on your mesh. LOD is supported only by the Unity Pro license. If you have only the Unity Basic license, then you can create your own LOD system very easily. The same idea for optimization can easily be implemented by yourself. The main thing is to change meshes in real time (more or less detailed) for your rendering, depending on their distance from the camera, which in turn will reduce unnecessary costs for the hardware.

Audio filters

Audio filters allow you to programmatically create different effects with the sound in real time. Imagine a situation where, in the game, you have to play a sound while your character is walking on sand. However, if the player suddenly gets into a tunnel, then the sound should be different from walking on sand. To solve this problem, you can choose one of the possible scenarios. The first solution to this problem lies in the fact that you can create or already use a ready-made sound for walking in tunnels and so on. For each situation, you will have a variety of ready-made sounds. Alternatively, this approach requires a lot of memory if you have too many different situations in the game where the game should play different sounds. This approach is not flexible relative to the second solution in the case of audio filters provided by Unity software in real time. Audio filters are supported only by the Unity Pro license.

Video playback and streaming

Today, many applications and games need to play different videos. A video may take a lot of memory, and this especially is an acute problem for mobile devices. In order to reduce the additional cost of memory for your video content, Unity allows you to broadcast via internet video streaming. This feature is only available on the Unity Pro license.
Full-fledged streaming with asset bundles
Asset bundles are supported only by the Unity Pro license. This feature greatly helps in optimizing the way of creating a quality game or application. This functionality allows developers to stream the content via the Internet, for example, adding new characters, new buildings, new weapons, new textures, and much more into the game.

Hundred thousand dollar turnover
This item is not a functionality in Unity, but it should rather be seen as a condition or requirement of Unity. This condition states that if in the last fiscal year, you (personally) or your organization, have earned more than $100,000 (inclusive), then you are required to use a Unity Pro license; that is, you have no right to use the Unity Basic license at all. This can be considered quite a logical and reasonable condition by Unity. After all, if you or your organization has earned more than $100,000 (or exactly $100,000), then you or your organization can buy a Unity Pro license without any problem.

Mecanim – IK Rigs
The new animation system in Unity, which is known as Mecanim, allows you to use a variety of different and useful functionalities, but a special opportunity and a key feature is the Inverse Kinematics (IK) Rigs. IK Rigs is supported in Mecanim for only humanoid characters with a correctly configured Avatar. The meaning of this feature is that you can call a function and pass a final point where the leg should be placed for hitting the ball (as one possible example, if you are creating a football game); after that, IK Rigs system will make all the rest automatically for you. For example, your character must take a cup on the table, but before that he will need to get up from the chair and go to the table on which the desired cup is, and only after that will your character be able to take a cup in his hand. These movements will play animations. All this hard work will completely rely on the IK Rigs system, you only need to specify the endpoint. IK Rigs is supported only by the Unity Pro license.

Mecanim – sync layers and additional curves
Mecanim also allows you to use different animation states simultaneously; for example, a character with full health will walk normally, but every time the health is decreased by about 20 percent, the character will begin to become worse and go slower, and then starts to limp at poor health. When the character’s health becomes very poor, he begins to creep along the ground. This approach uses the sync layers option for grouping different animation states. This greatly simplifies the creation of a variety of conditions by the reusability of sync layers for different situations.
It is possible to modify sync layers dynamically in real time to reuse your state machine many times with different animations, but with the same conditions. Thus, developers do not need to create so many different state machines for all animations, but only a few, and reuse them while playing different animations. This feature is supported only by the Unity Pro license.

Additional curves allow you to add new curves to your animations in order to control different animation parameters. It's easy and very convenient to manage your animation curves in Unity Editor. This feature is supported only by the Unity Pro license.

**Custom splash screen**
This feature has the following meaning: while using the Unity Basic license, every time your application boots, your user sees the Unity logo image. If you want to replace that logo with your image, then you will need to purchase a Unity Pro license.

**Build size stripping**
This is a very important Unity feature, especially for mobile devices. With this feature, Unity allows you to remove all the excess out of your build. Unity helps you greatly in it, because it includes only those assets that are used in your game in your final build. Also, this feature allows you to include only those parts of the Unity engine that are utilized in your game in the final build. This feature is supported only by the Unity Pro license.

**Lightmapping with global illumination and area lights**
All Unity licenses support lightmapping. Unity allows you to bake lights and shadows for static objects. You can add more realism to your game by adding the global illumination and area lights provided by this feature, supported only by the Unity Pro license.

**HDR and tone mapping**
*High Dynamic Range (HDR)* and tone mapping functionality are very useful for improving the quality of the images in your game, but it requires a significant investment of resources. You must be very careful to use such an expensive operation, as well as many other expensive features in Unity. This feature allows you to use more colors than usual, which allows you to create, for example, morning light in the room. This feature is supported only by the Unity Pro license.
Occlusion culling
This feature is very useful for optimization. Unity excludes all unnecessary objects for rendering, such as those that are behind the wall or far from the camera. Otherwise, objects that are hiding will waste processor time and memory. You can easily create a system with the same idea for your specific tasks. This feature is supported only by the Unity Pro license.

Light probes
This functionality is used to supplement the lightmapping optimization method or the so-called light baking, which is used only for static objects, while dynamic objects look much worse. Light probes solve this problem for dynamic objects, but they must be used very carefully and gently so as not to harm the performance of your application or your game. This feature is supported only by the Unity Pro license.

Static batching
This functionality may optimize the rendering process in your game scene by reducing a large number of draw calls for static objects. This feature allows us to reduce many unnecessary draw calls. It works only for static objects, and is supported only by the Unity Pro license.

Render-to-texture effects
This Unity functionality is very interesting and often useful. This feature is useful when you want to directly render your camera not to the screen, but to your image. After that, you can do what you want with that image; for example, you can create a TV box in your game. Also, you can perform postprocessing effects with that image, and much more. However, this feature is very expensive, so use it carefully. This feature is supported only by the Unity Pro license.

Fullscreen postprocessing effects
This feature can also create very interesting effects. Alternatively, this functionality should be used very carefully, especially for mobile platforms as it can take a lot of resources for execution. While optimizing, you should not forget about its high price. For example, you can create effects such as a motion blur for a Formula 1 game, where cars go at a very high speed. Also, you can create bloom effects with this functionality, which makes objects glow like neon. This feature is supported only by the Unity Pro license.
NavMesh – dynamic obstacles and priority
While searching for the right way with a pathfinding system, there may be dynamic obstacles that your character should avoid. You can programmatically set objects as obstacles in your code for a certain time. The ability to manage priorities affects searching for the right path. This feature is supported only by the Unity Pro license.

.NET socket support
The ability to use the .NET sockets allows you to create a variety of network games, as well as connect directly to a device without a server. This feature is supported by both Unity Basic and Unity Pro licenses.

Profiler and GPU profiling
This is very useful for profiling your projects. Optimization should begin with finding bottlenecks in your application or in your game. To be more effective, while searching bottlenecks in your project, you should have good tools. You can create such tools yourself, or you can use ready-made solutions. One of the solutions provided by Unity is the profiler tool. This feature is supported only by the Unity Pro license. Those who have only a Unity Basic license have to create their own profiler tool. That's why at the end of the book, we will develop a very simple code profiler tool.

Real-time directional shadows
Lighting and shadows are key aspects in most games. Many developers from around the world create their games trying to achieve the most realistic lighting. No shadows in a world with realistic lighting is much worse than having shadows. Alternatively, to create such a realistic world requires a lot of resources, such as time and memory. This is especially important for mobile devices. You will need to find a balance between quality and performance. This feature is supported by both the Unity Basic and Unity Pro licenses.
Script access to asset pipeline

This feature is also very useful. With this functionality, you can automate the processing of large amounts of assets or builds. For example, imagine that you need to put a watermark on each of your textures. If it is only a few textures, then it can be done manually, but if there are too many textures, hundreds for example, then the automation of the processing of each picture will be very useful. For more information, you can look in the official documentation of Unity. Unity provides a variety of functions for the convenient handling of your assets and builds. This feature is supported only by the Unity Pro license.

Summary

In this chapter, we looked at how to install the Android SDK on Windows and Mac OS X. We also covered the various Unity settings before making the first build. After that, we looked at the APK expansion files in Unity for Android devices. Then, we talked about the build settings for Android. We created a very simple and small game build for Android platforms. At the end of the chapter, we considered step-by-step key points and the difference between the Unity Pro and Unity Basic licenses.

The next chapter has many more interesting details about the Android platform. You will learn how to create plugins within Unity for the Android platform. You will also find out how to make anti-piracy checks, detect screen orientation, handle vibration support, determine device generation, and many more useful things. Let’s move on!
Where to buy this book

You can buy Unity 5 for Android Essentials from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.