Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 5.258

IJCSMC, Vol. 5, Issue. 3, March 2016, pg.140 – 145

Production of Game Content for Mobile Phones based on Android Operating System

Omid Haghighatgoo[1]

[1]Department of Software Engineering, Payame Noor University, Iran

Abstract— Today, games have become an integral part of mobile phones. As it measure the power of a phone with its power to run games with high quality graphics. Most phones pose one of the Android, Windows Mobile or iOS systems. Among them, Android has attracted more programmers as it is open source and continuous improvement. As a result, more applications and games are produced for it, which leads to more popularity among buyers. One of the concerns of game developers has been selecting a powerful game engine in line with their needs. In this article we first introduce Android operating system and its features, we then examine the content of game. Necessity of using game engine is mentioned in this paper in next stage, and some game engine are studied in this paper as JMonkey, libgdx, AndEngine, in terms of characteristics, strengths and weaknesses, and performance of each one. Finally, a conclusion is presented for selecting appropriate Android game engine.

Keywords: Android operating system, game engine, java, mobile, Linux, middleware

I. INTRODUCTION

Since the production of the second generation of mobile phones around 1990 to 1995, when the phones were smaller and came with black and white screen, the games were inseparable elements of these devices. Nowadays, most mobile phones have one of these three operating systems: Android, iOS and Windows Mobile, which Android, because of being open source and its successive improvements, has become more powerful and attracted more programmers. As a result, more applications and games are being created for it, which leads to more popularity among buyers.

It's been always a concern for the game developers to choose a powerful game engine and in line with their needs. In this context, as a result of Android source being open, companies and various programming teams have produced game engines, which some of them are open source and some aren't free. Some of these engines are only capable of producing two-dimensional or three-dimensional games, but some can be used for both purposes. Each of them has its special characteristics with different priorities from game to game.

For example, some of the features that may be inconsequential for a game and important for another game are:

- Fire, smoke, rain, dust and lightning effects and etc.
- Dynamic spaces like ruins and deserts, making strange worlds, making the real worlds of objects, costumes, locations
- The artificial intelligence
- Various rendering systems

In this article we will briefly introduce Android operating system, then we will discuss about its architecture to become more familiar with its features, and at the third part we will talk about the game engine and explain the reasons of using these engines. In the fourth part, we will express the process of making games and the features we are after in game engines to know what we should consider for appropriate engine selection. Then we will introduce three Android game engine AndEngine, Libgdx, JMonkey and examine their characteristics. Finally, we summarize our article and present the most appropriate game engine based on our needs.

II. Introduction of Android

Android is a Linux-based operating system that is designed primarily for devices with touch screens. Android source code is available under the Apache license by Google. This license allows the operating system to be freely changed and redistributed by device manufacturers. Since 2012, Android became the most popular mobile operating system and was installed on many devices and is leading the market[1].

A. Structure of Android

Android interface is based on direct connection by using touch. Responding to input is designed to be prompt and smooth and sometimes notify the user to receive user commands with vibration response. In some cases, some hardware such as an accelerometer, gyroscope and proximity sensor is used to respond in some applications. Android devices Boot to the home screen, and main menu and side information are available by touch and are similar to the ones on the computer desktop screen. Programs that increase the device's ability, are often written by Java programming language developed by Android Software Development Kit (SDK). SDK includes various tools, including software libraries, debuggers, simulators based on QEMV, documentations, sample codes and some user manuals. QEMV which stands for Quick emulator is a free and open source host for hardware simulation. IDE is often eclipse that uses the ADT Plugin. Other development tools including a native software kit for apps, and extensions based on C and C ++ languages are available. The main hardware platform for Android is ARM 32-bit architecture. The AndroidX86 project provides the support for X86 architecture. Since 2013, the minimum of 512mb RAM for the new versions of Android was proposed.

B. Software Stack

On the Linux kernel, middleware, libraries and C language APIs that are software program frameworks which include Java-compatible libraries are implemented. Android uses a virtual machine called Dalvik. This machine uses compiling to run dex codes that are specific for Dalvik. Dalvik compatible dex codes are usually translated from Java bytecodes. The below figure shows the Android architecture.

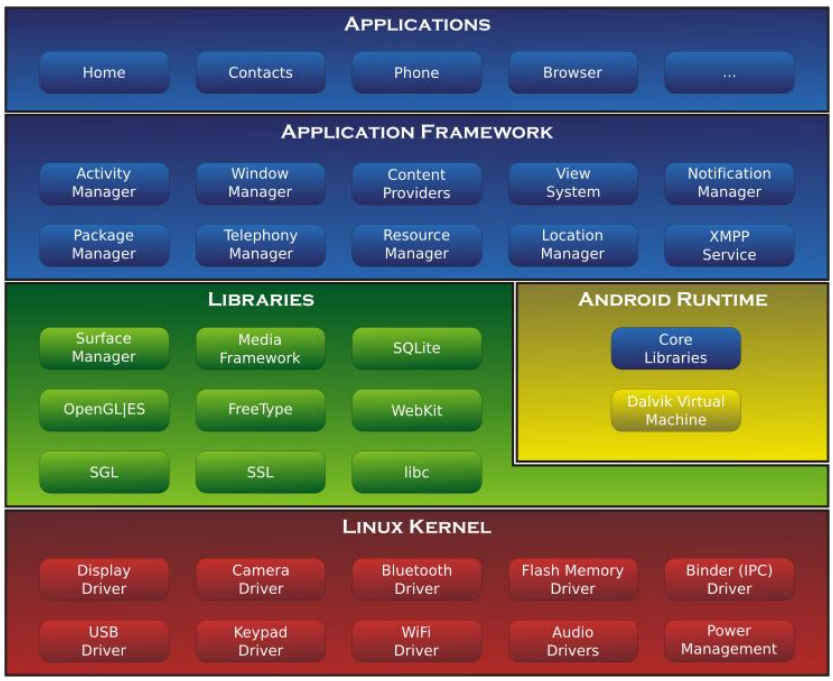

Fig. 1 The architecture of the Android operating system

III. Game Engines

Game engine, is a system designed to build and develop video games[2]. Game engines provides a software framework for developers to be able to create games for consoles, mobile devices and computers. The tasks of a game engine can be briefly paraphrased as:

- Rendering engine for 2D and 3D graphics, Animation and sound
- Diagnose and fix collision
- Providing a script language
- Artificial intelligence
- Memory, network and multi-threading management
- Localization Support (determining the location of factor and its distance within the environment)
- Scene graph (a data structure for storing and manipulating graphical information based on vectors)

Production process becomes more economical by reusing or accordance of a game engine to produce different games. These game engines also help to prepare games for different platforms. In addition to provide the ability of reusing, most of these engines provide graphical tools to expand the games. Game engines are also called firmware, because they provide the required tools, reusability, flexibility, and the main required factors in a separate package, to produce games with less cost in time and money. Some game engines only provide three-dimensional real-time processing capabilities instead of different required capabilities for games. These engines are relying on the game developers and entrusted the rest of the tasks to them. These motors are often called graphical engines, processing engines, or three-dimensional engines, instead of game engines. Most of three-dimensional processing systems in game engines are made on the basis of APIs. For example, Direct3D or OpenGL which provide abstract applications of the video card or graphics processing unit (GPU). Low-level libraries such as DirectX, Open GL or Simple Direct Media Layer are sometimes used independently associated with the computer hardware such as input devices (keyboard, mouse, joystick) network card and sound card. Game engines can be written in various languages including C ++, C and Java, in which case each has its own performance and capabilities.

We expect Android game engines the same from the other engines that we have already talked about them, the only difference is that these games are supposed to run on devices with Android OS; these devices generally have weaker hardware than home computers or game consoles. The other feature is the input device which is a 3 to 10 inches touch screen. However, one of the most important decisions which should be made before any action, is the type of the game engine, but the process of producing the games contains some steps that must be regarded in almost all games. Knowing this process will help us in choosing the right engine. We will compare the solutions and services provided by different engines for the steps to find out which engine is better and has more tools, so that we can reach our goal more quickly and be able to make a perfect game to match to the devices you're considering it to run at.

A. The main processes of making a game

Selecting the type of the engine is one of the basic tasks that must be done according to the requirements[3].

The first type is the Object Engine. This object is not generally suitable, because you do not have any restrictions for fps so it is possible that the video ends faster in one device than another.

The second type is Fixed Step Engine object. In this type, by determining the fps, you can see the same video on different devices.

The third type is Object Limit FPS Engine. This type uses a series of internal calculations to compare the input fps to the appropriate amount and if any defects be detected, it will choose the right state.

Out of two types of Single Scene Split Screen Engine and double Scene Split Screen Engine, we choose one due to the game type to be single player or multi player.

Then, we should select the appropriate image resolution. Given that the devices may vary between 3 to 10 inches, suitable image resolution should be selected for all devices to be displayed in an acceptable quality. The next step is making the game management which its job is to keep the game data such as scores, game information etc. After this step, is selecting the right voice. We must now create the textures and choose or design the desired music and fonts. By making the resource management, we will have access to the arbitrary methods. This includes loading graphic screens, selecting the appropriate sounds in different parts of the game as well as the rules and procedures of the game, saving facilities and reloading from the last part of the game which was played. We should consider these tools and store the required information.

B. AndEngine

AndEngine is an open source game engine based on OpenGL. It covers the most important features of OpenGL[4]. This means that if you do not have any experiences of working with OpenGL, you can still use it. The most important features of this engine are:

- Scalability on many devices (with different resolutions)
- Multi-touch support
- Physical development or 2D kit (used in the famous AngryBirds game)
- Support for tiling in TMX format

The benefits of this game engine are as follows:

- Free and open source
- Most important aspect of game producing is provided by the engine.
- No need to learn OpenGL

The disadvantages of this engine are as follows:

- There is no proper documentation for the codes.
- Sometimes it is slower than the other engines.

C. Libgdx game framework

This framework is built for Windows, Linux and Android[5]. It uses the Java programming language to implement C++ codes associated with the physical engine or processing the sounds and images. As a programmer, you should focus just on its Java areas; this framework also provides the local codes.

Libgdx tries to provide an integrated framework to run games on all platforms the same way. Unfortunately achieving this goal is a little difficult, because the Android devices are usually slower than desktop computers, and this makes the games run faster on computers. But there are some measures that have been taken in this framework which can solve this problem. The framework consists of all required modules including modules for graphics, sound and music, input and output control, reading and writing data to files, etc. For each module, it has one or several Java Interfaces for each operational platform. This is called back-end implementations. Three types of back-end are:

1. Desktop back-end
2. Applet back-end
3. Android back-end

Programmers don't need to know the full information about the back-end, and by dealing with just one general interface, the framework will do the rest.

D. JMonkey game engine

This free and open-source game engine is based on Java programming language. The engine which is intended for Java programmers is dedicated to the production of 3D games with the latest technologies[6]. According to its creators, though this engine is written by Java, it can be very fast and easy to work with, than C++. The engine with its provided tools, has made programmers carefree of concerning about compiling problems. JME3 SDK comes with an auto-update feature to overcome a lot of programming requirements. The SDK is provided with netbeans IDE. The advantages of this engine are as follows:

- Beautiful and strong light and shadow effects.
- Objects and materials System that are entirely based on color shades
- Smoke, water and mirror effects and filters
- Automatic LOD generator
- Multi-threading support

Weaknesses of this game engine include:

- It is only capable of producing 3D games
- To use it, you must be fluent in Java programming language
- IDE is netbeans and eclipse is not supported
- Its documentations are not enough

IV. Conclusion

One of the most popular and revenue generating industries of this era is game production. At the beginning, these games were only applicable with consoles and computers. However, after the third generation of mobile phones introduced, games also became one of the key elements of this industry. Making games for the Android OS can offer the financial, advertising, cultural and educational aspects to have a special place. So, in this article we discussed about the Android structure and then presented the game production levels. To reach that, we examined three game engines AndEngine, Libgdx, Jmonkey to know the ability of each in order to understand which one of them has the fit tools for our purpose. Of course, these are not the only available engines. For example playcanvas is based on the

JavaScript programming language. Or C # programming language enthusiasts can take advantage of MataliPhysics engine.

Generally, by knowing the game production process, we can use various tools available to produce the desired games. It is important to remember that the game engines are only tools to accelerate and simplify the production process and they won't stop us from reaching our goal or limit our framework.

REFERENCES

[1] http://en.wikipedia.org/wiki/Android_(operating_system)

[2] http://en.wikipedia.org/wiki/Game_engine

[3] Jayme Schroeder, Brian Broyles *AndEngine for android game development Cookbook* , packtpub , 2013.

[4] Mateusz mysliwiec http://www.matim-dev.com/introduction-to-the-andengine.html , 2013.

[5] http://libgdx.badlogicgames.com

[6] http://hub.jmonkeyengine.org

