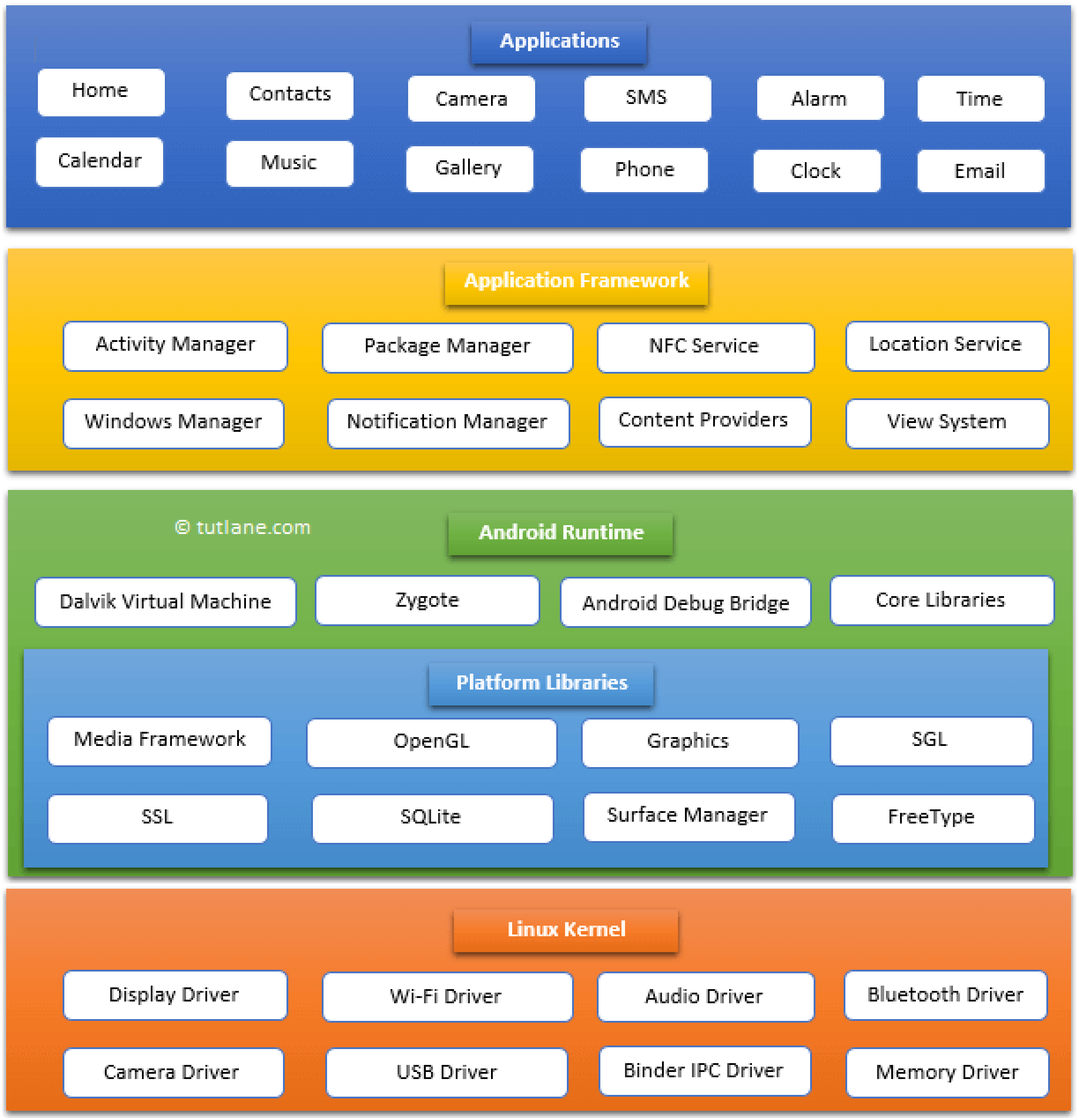
Soriano, Trixie L.
CEIT-37-601P

ANATOMY OF ANDROID

Main Components of Android architecture:



1. **Application** - Applications that we design, both native and third-party, such as contacts, email, music, galleries, clocks, games, etc., will only be installed on this layer.

2. **Android Framework** - provides the classes necessary to build Android apps. Additionally, it controls the user interface and application resources and offers a general abstraction for hardware access. In essence, it offers the services needed to develop a specific class and make it useful for building applications.

3. **Android Runtime environment** - The Dalvik virtual machine and core libraries are both included in the Android Runtime Environment, which is an essential rather than internal component of Android. The applications are powered by the Android run time, which also serves as the foundation for the application framework, together with the libraries.

4. **Platform Libraries** - includes various C/C++ core libraries and Java-based libraries such as SSL, libc, Graphics, SQLite, Webkit, Media, Surface Manger, OpenGL, etc. to provide support for Android development.

   The following are the summary details of some core android libraries available for android development.

   - Media library for playing and recording audio and video formats
   - The Surface manager library to provide a display management
   - SGL and OpenGL Graphics libraries for 2D and 3D graphics
   - SQLite is for database support and FreeType for font support
   - Web-Kit for web browser support and SSL for Internet security.

5. **Linux Kernel** - The foundation and brain of the Android architecture. It oversees all of the drivers, including those for the Android device's display, camera, Bluetooth, audio, memory, and other components that are frequently needed during operation. Between the hardware of the device and the remaining portions of the stack, the Linux Kernel will offer an abstraction layer. Memory management, power management, device management, resource access, etc. are all under its control.

ANATOMY OF ANDROID APPLICATION

4 Building blocks to an Android Application

- **Activity** - Typically, an activity in application consists of just one screen. The implementation of each activity consists of a single class that extends the Activity base class. The class will respond to events and show a user interface made up of Views.

- **Intent Receiver** - To transition between screens on Android, a unique class called **Intent** is used. Applications' intentions specify what they want done. The action and the data to act upon are the two components of the intent data structure that are most crucial. The MAIN (the application's main entrance), VIEW, PICK, EDIT, etc., are typical examples of action values. Using a uniform resource indicator, the information is expressed (URI). Then when you want some parts of your application to run in response to a specific external event, you can use an **intent receiver**. Intent receivers don't provide a user interface (UI), but they might show notifications to let the user know if something noteworthy has happened. Intent receivers can be registered from code using Context.registerReceiver, in addition to being registered in AndroidManifest.xml ().

- **Service** - Long-lived code that functions without a user interface is known as a service. A media player playing music from a play list is a nice illustration of this. There would most likely be one or more activities in a media player program that let the user select tracks and begin playing them. The user will expect the music to continue playing even after switching to a new screen, hence the music playback itself shouldn't be managed by an activity. In this situation, the media player activity might launch a background service using Context.startService() to continue playing music. After then, the system will continue to play music until it is finished.

- **Content Provider** - Applications have the option of storing their data in files, a SQLite database, preferences, or any other logical storage method. However, a content provider is helpful if you want the data from your application to be shared with other applications. A content provider is a class that implements a common set of methods to enable other applications to store and retrieve the kinds of data that are handled by that content provider.

Reference:

- https://www.tutlane.com/tutorial/android/android-architecture#:~:text=Android%20architecture%20is%20a%20software,services%2C%20runtime%2C%20and%20application.
- https://www.oodlestechnologies.com/blogs/Anatomy-of-Android-Application/#:~:text=Android%20uses%20a%20special%20class,%2C%20PICK%2C%20EDIT%2C%20etc.