***DEVAN SODHI***

**TOPIC A:**

* Java is the official language for Android App Development and consequently, it is the most used language as well. Many of the apps in the Play Store are built with Java and it is also the most supported language by Google. In addition to all this, Java has a great online community for support in case of any problems.
* A mobile application, also referred to as a mobile app or simply an app, is a computer program or software application designed to run on a mobile device such as a phone, tablet, or watch. Apps were originally intended for productivity assistance such as email, calendar, and contact databases, but the public demand for apps caused rapid expansion into other areas such as mobile games, factory automation, GPS and location-based services, order-tracking, and ticket purchases, so that there are now millions of apps available.
* Every Android app has a main thread which is in charge of handling UI (including measuring and drawing views), coordinating user interactions, and receiving lifecycle events. If there is too much work happening on this thread, the app appears to hang or slow down, leading to an undesirable user experience. Any long-running computations and operations such as decoding a bitmap, accessing the disk, or performing network requests should be done on a separate background thread. In general, anything that takes more than a few milliseconds should be delegated to a background thread. Some of these tasks may be required to be performed while the user is actively interacting with the app. To learn how you can run tasks on background threads and off the main UI thread while the app is actively being used, please take a look at the threading solutions guide.
* Applications may also require some tasks to run even when the user is not actively using the app such as syncing periodically with a backend server or fetching new content within an app on a periodic basis. Applications may also require services to run immediately to completion even after the user has completed interacting with the app. This guide will help you learn which solution best meets your needs for these use cases.

**TOPIC B:**

* The absolute minimum requirements for Android were originally a 200 MHz processor, 32 MB of RAM, and 32 MB of storage.
* Out of the box, Android is incompatible with ARMv4 or lower; ARMv5 or higher is needed to run native code without modifications.
* Android 4.4+ requires an ARMv7 processor. Custom versions have been made for ARMv6 however.
* The requirements in these documents must be met for a device to be "Google Approved" and ship with the official Google apps such as the Play Store and Google Talk. However, they are not necessarily hard requirements. Since Android is open-source it can be modified to run on lesser hardware, and the opposite is possible

as well — modifications necessary to run the OS on a device may make the firmware image too large to fit on it, just for example.

* Android is a Linux based operating system it is designed primarily for touchscreen mobile devices such as smartphones and tablet computers.
* The hardware that supports android software is based on ARM architecture platform.
* HTC, Samsung, LG, Huawei, Lenovo, and Sony are just a few of the manufacturers that run Android on the devices they manufacture. Android is now running on one billion devices.

**TOPIC C:**

* The user interface (UI), in the industrial design field of human–computer interaction, is the space where interaction between humans and machines occurs. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls, and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology.
* User. Each user is intended to be used by a different physical person. Each user has distinct application data and some unique settings, as well as a user interface to explicitly switch between users. A user can run in the background when another user is active; the system manages shutting down users to conserve resources when appropriate. Secondary users can be created either directly via the primary user interface or from a Device Administration application.
* It seems almost impossible to restart the phone when the power button is not working. But is it actually impossible to restart the device when the power button is not working. Obviously not; there definitely is a way to restart the device when the power button is not working. If the device is already on, then restarting the phone is not too much of a bother. So, there are 2 cases here. One is when the phone is switched off and the other one being the Android device in switched on state.

**TOPIC D:**

* Input device configuration files are typically not necessary for standard peripherals such as HID keyboards and mice since the default system behavior usually ensures that they will work out of the box. On the other hand, built-in embedded devices, particularly touch screens, almost always require input device configuration files to specify their behavior.
* Users expect to be able to control the volume of an audio app. Standard behavior includes the ability to use the volume controls (either buttons or knobs on the device or sliders in the UI), and to avoid suddenly playing out loud if a peripheral like headphones is disconnected while in use.
* We may not come face to face with our phones' file systems too often, but it's valuable to know they're there — and to know how they can work for us when the need arises.

**TOPIC E:**

* They release a major OS update once a year.
* Security starts at the application layer with our built-in malware defence. Backed by Google’s machine learning, it’s always adapting and improving. Every day, it automatically scans all of the apps on Android phones and works to prevent harmful apps from ever reaching them, making it the most widely deployed mobile threat protection service in the world. And with Find My Device built in, devices stay protected, even when they’re lost.
* Android incorporates industry-leading security features and works with developers and device implementers to keep the Android platform and ecosystem safe. A robust security model is essential to enable a vigorous ecosystem of apps and devices built on and around the Android platform and supported by cloud services. As a result, through its entire development lifecycle, Android has been subject to a rigorous security program.

**TOPIC F:**

* This lesson describes how to write applications that have fine-grained control over their usage of network resources. If your application performs a lot of network operations, you should provide user settings that allow users to control your app’s data habits, such as how often your app syncs data, whether to perform uploads/downloads only when on Wi-Fi, whether to use data while roaming, and so on. With these controls available to them, users are much less likely to disable your app’s access to background data when they approach their limits, because they can instead precisely control how much data your app uses.

Class that answers queries about the state of network connectivity. It also notifies applications when network connectivity changes.

The primary responsibilities of this class are to:

1. Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)
2. Send broadcast intents when network connectivity changes
3. Attempt to "fail over" to another network when connectivity to a network is lost
4. Provide an API that allows applications to query the coarse-grained or fine-grained state of the available networks
5. Provide an API that allows applications to request and select networks for their data traffic