**The mobile developer role**

what kinds of things does a mobile developer do?

First, is designing, building and maintaining mobile applications.

This includes tasks like building a user interface or UI, based on design specifications. Writing code to support functionality and building animations and transitions to provide an engaging user experience.

Other common tasks include integrating third party application program interfaces or API's, to connect the app with other technologies or code.

Writing tests and releasing apps and updates to the Apple and Google stores.

Apart from the many coding tasks, developers also often work as part of a team which can include other developers, project managers, quality assurance or QA professionals, and designers.

Version control is a system that allows a development team or teams to track and manage changes made to the software code. Version control also allows your team or

teams to work simultaneously and collaboratively on projects by enabling access to the most recent version of the software code.

GitHub is one online platform that can assist you with version control.

Amongst other things, GitHub makes it easy for you to keep track of your and

others work and review code changes in collaborative projects.

It allows you to access and navigate between different versions of online files.

what other skills do they need to develop mobile apps?

Well, one of the most important is knowing the language that their apps

operating system or OS, software development kit is built upon.

In the case of android developers, this is Java or Kotlin.

iOS developers use objective C and Swift.

Other valuable skills for any mobile developer who builds the user interface or

UI for mobile apps, are design and user experience or UX as well as

knowledge of server technology and continuous integration or CI.

This is the practice of automating the integration of code changes from

multiple contributors into a single software project.

Mobile website and mobile app

A mobile website is a website designed, developed, and optimized for mobile devices only.

While mobile websites are designed to display and work on major browsers,

they still require an Internet connection and don't take advantage of the unique functionality of a mobile device.

A mobile app is installable software that runs on a smartphone device.

Mobile apps use the devices hardware, and software features and usually provide an efficient,

more intuitive, and seamless user experience.

mobile app or mobile website?

one way to decide is to ask yourself this question.

What specific actions do you expect your product to perform?

For example, if you don't need your users to perform any operations and your product

is mainly content-driven, then you may opt for a mobile website.

But if you want users to perform actions that will require the device's hardware,

you will need a mobile app.

The first advantage of mobile apps

is that they are faster than mobile websites.

They can also access the device or system resources,

such as the location or Bluetooth,

and can work without Internet access.

That said, mobile apps can be quite expensive to build.

and are more difficult to

maintain when compared to mobile websites.

**Popular mobile operating systems**

Two of the most popular mobile operating systems are Android and iOS.

**Android**

Android is a mobile OS that was released in 2008. You’ll likely find that it is the base OS of most phones and tablets worldwide.

Android OS is based on a modified version of the Linux kernel, which is the foundation of the Linux computer OS and other open-source software. This is software that anyone can use or modify. The Android OS is built primarily for smartphones, Chromebook, Android TV, Android Auto, as well as wearables such as smartwatches.

Android gives users a multitouch experience, which implies that all user input is via touch gesturing. Through several actions such as swiping, pulling and tapping, users can seamlessly interact with the screen. With built-in sensors such as gyroscopes and accelerometers, Android devices allow users to switch between the orientations of applications when tilting the devices.

**iOS**

iOS is Apple’s proprietary OS that runs on the iPhone, iPad and other mobile devices. iOS gives users a multitouch feel, meaning that all user input is via touch gesturing. Similar to Android, through several actions such as swiping, pulling and tapping, users can seamlessly interact with the screen. With built-in sensors such as gyroscopes and accelerometers, Apple devices allow users to switch between the orientations of applications when tilting the devices.

With the rise in popularity of iOS, developers now have more controls and access to its features. After the release of iPhone 2.0, a software development kit (SDK) was introduced. This is a set of tools that allows developers to build apps for the OS. Due to huge improvements with iPhone OS 3.0, developers were then given more power with core location and push notifications. Finally, multitasking was introduced in iOS 4.0.

Android Studio

Android Studio is Google's official Integrated Development Environment

or IDE for building apps that run on Android operating systems. It provides developers with features that not only improve the development of their apps,

but also the overall quality of work. This IDE supports various language for app development,

such as Kotlin, Java, C++, and more with extensions.

**Top Level Component**

The ability to connect to the internet, make calls, take pictures and much more is made possible in Android apps with the help of four top-level component classes: BroadcastReceiver, ContentProvider, Service and Activity. These are all accessible in the Android software development kit (SDK).

**Android Views**

In Android, Views occupy a rectangular area on the screen and are responsible for drawing and event handling. They can display images, text and more. A combination of all of these Views forms a design interface.

**Android Layout Files**

In Android, each layout is represented by an XML file. These plain text files serve as blueprints for the interface that your application presents to the user. In addition to the XML approach, there are other ways to create a user interface. For example, interfaces can be created with Android Views entirely in code using Kotlin or Java. Also, Google has created a completely new way of creating Android user interfaces - Jetpack Compose. With this library, interfaces are created entirely in Kotlin code, without XML.

**Project Files**

Android Project Files belong to one of three main categories: configuration, code and resource. Configuration files define the project structure, code files provide the logic and resource files provide essentially everything else.

In Android development, using Extensible Markup Language (XML) is one way you can develop your app's user interface (UI). Read on to find out more about XML, its functions and structure.

**What is XML?**

XML (Extensible Markup Language) is used in the presentation of different kinds of data. This is a flexible and descriptive way to create well-structured data and electronically distribute it across the public internet, as well as via corporate networks.

Its main function is to create data formats that are used to encode information for documentation, records, transactions and several other data formats. Data presented using XML may be used for creating different data types that are generated by building dissimilar types of content, including web and mobile content.

To use the Android emulator, you will need to download it first. This reading will guide you through setting up emulator software in Android Studio.

Although the set-up process has been captured with a Mac device, the same instructions apply to Windows users.

You can download the emulator from the SDK manager located in the top right corner of the Android Studio welcome screen.

**Mobile CPU architecture**

In order to develop and deploy apps for different mobile devices, the Central Processing Unit (CPU) architecture must be considered.

**What is a CPU?**

A CPU is like a translator between the software and the hardware of a device. It can take high-level software instructions and translate them into native machine language that a mobile phone can understand and use to perform specific operations. Ideally, you want a CPU that combines efficiency and power, without requiring excess resources, something that will be determined by its architecture. A CPU with great architecture provides mobile users with a seamless user experience without consuming significant battery resources.

**What’s ARM, ARM64 and x86?**

As of now, there are three main CPU architectures used in most smartphones – ARM, ARM64 and x86. CPU types include:

* ARM: ARMv7 or armeabi
* ARM64: AArch64 or arm64
* x86: x86 or x86abi

Of these three, ARM is the most common as it is properly optimized for battery use.

You now know that Android operating system (OS) images are versions of Android that you can run on a computer. For example, you could download the system image of the latest Android OS and run it on the emulator to see what the OS is like and test the latest features. Or, if you wanted to test your app and how it runs on an earlier version of Android, you could download an older system image like Android Lollipop and run that.

**Why use libraries?**

Libraries extend the capabilities of the Android software development kit (SDK), allowing you to use code written by other developers. These open source libraries are hosted on an external server and are downloaded by the build system, Gradle, when you are building a project.

The best libraries provide entirely new functionalities and give you access to awesome functions with lesser code as a developer. Unlike copying and pasting code, libraries are entirely portable and easy to plug in too. This makes it easy to access advanced features with minimal work or confusion.

**Layouts**

Part of setting up the screen of your app in Android Studio involves layouts. Read on to learn more about the function and types of layouts in Android Studio.

**What is a layout?**

In Android, layout defines the user interface (UI) for an app or activity and holds the UI elements that will appear to the user.

Related to layouts are **View** and **ViewGroup**.

**View**

A **View** is defined as the UI which is used to create interactive UI components such as **TextView**, **ImageView**, **Label**, **RadioButton** and so on. It is in charge of event handling and drawing. They are generally referred to as "widgets".

**ViewGroup**

A **ViewGroup** serves as a parent class for layouts and layout parameters that hold other Views or ViewGroups and define the layout properties. They are generally referred to as "layouts".

**Types of Android Layouts**

**LinearLayout**

**LinearLayout** is a ViewGroup subclass used to render child View elements one after the other either horizontally or vertically based on the orientation property specified.

**ConstraintLayout**

**ConstraintLayout** is a ViewGroup subclass used to specify the position of layout constraints for every child View relative to other views on the screen.

**Frame Layout**

**FrameLayout** is a ViewGroup subclass used to specify the position of View elements it contains on the top of each other to display only a single View inside the **FrameLayout**.

**Table Layout**

**TableLayout** is a ViewGroup subclass used to display the child View elements in rows and columns.

**WebView**

**WebView** is a browser that is used to display the web pages in your activity layout.

**RecyclerView**

You can use **RecyclerView** with LinearLayoutManager to display scrollable lists of items in a single column. You can also use **RecyclerView** with GridLayoutManager to display a scrollable list of items in a grid view of rows and columns.