Android app development in c language

I'm not robot

reCAPTCHA

Continue

I'm not robot

reCAPTCHA

So, have you decided that you want to learn how to develop Android apps? Well done! Unfortunately, the intentions can only carry you so far. Learning to code can be difficult. Sometimes it's not even clear where to start! So you probably have plenty of questions to answer before you even get started. What programming language should you learn? Where can you learn about your chosen language? Once you understand the basics, where do you even start typing code? In this post, we are going to try to answer this first question. To that end you will have to do some soul searching, how to decide which programming language to start with will depend entirely on what you hope to achieve. READ ALSO: The best manufacturers of Android apps to create apps and create them with zero code Take your choiceIf you want to develop android apps, the first step is to choose a language. Differences between different Android programming languages can be a bit complex and nuanced. But more important than the language itself, it is the tool to which it is attached, and the main features and purposes of this tool. Languages you might consider learning to develop Android include: Java - Java is the official language of Android development and is supported by Android Studio. It has been the official language longer than Kotlin, and it is also popular outside of Kotlin development for many other purposes. Java and Android Studio have a steep learning curve, however. Kotlin - Kotlin is another official Android language. It's similar to Java in many ways, but a little easier to get your head around. It is also currently Google's preferred language of choice, although it is not so widely used outside of Android Studio. This may make it a little less attractive for those hoping to work as developers through numerous projects. SH - Android Studio also supports THES using Java NDK. This allows for native coding applications that can be handy for things like games. The NHS is more complex though, and this option is basically just going to appeal to larger, professional teams. In addition, the NHS is supported by Unreal Engine.C. - C- C- is a more beginner-friendly alternative to C or C that confuses more code. It's also a little less complicated than Java, although the two languages are very similar. It's supported by some very handy tools, such as Unity and Xamarin, that are well served on game development and cross-platform development. C with Unity is the best option for many mobile game developers. LUA (Corona) - Another cross-platform instrument built on LUA. This makes it much easier to create applications while still being able to call home libraries. (PhoneGap) - If you already know how to create interactive web pages, then you can use this knowledge with PhoneGap to create a more basic cross-platform application. JavaThon it's time to develop apps for Android, Jave Jave one of two official options. This means that it has a lot of support from Google. Most non-themed applications have probably been built with either Java or Kotlin.The number one way to develop Android apps is to go ahead and download Android Studio. It's a piece of software called IDE, or Integrated Development Environment. It will come packed with Android SDK (a set of tools to facilitate Android development in particular) and it will give you everything you need in one place to get up and running. The official documentation from Google will refer to Android Studio and Java (or Kotlin) and you'll find a lot of support online. Java itself was released by Sun Microsystems back in 1995 and is used for a wide range of software applications. Although Google clearly prefers Kotlin, Java is so entrenched and familiar that many development teams have decided to stick to it. READ ALSO: Anatomy of the app: Introduction to the lifecycle of Activity Strengthened, Java is also complex and not a great first language. Things get more complicated once again you add Android SDK to the mix; for the first time a programmer can struggle to know what Java is and what Android is! Java is an object-oriented programming language with confusing themes such as designers, zero-pointer exceptions, proven exceptions, and more. It's not very readable and you'll use a lot of boiler code to do simple things. Developing using this route also requires a basic understanding of concepts such as Gradle, Android Manifest and XML marking language. This does not mean that Java is a bad language - far from it. It would be wrong to call any language bad, but it's also true that most of the Java inconveniences actually exist for our own good and encourage clean code. Many people love Java for this reason and it is also one of the most versatile and widely used. According to the PYPL (PopularitY of Programming Languages) table, Java is the most sought-after programming language among employers. Android Studio has also been going from strength to strength over the past few years. Features such as visual designer and offerings make the process smoother, while advanced, powerful features are added all the time to give developers access to things like easy-to-implement cloud storage. It's worth getting on board, even if this rapid progress makes it hard to keep up sometimes (especially if you're some poor guy who writes about this stuff for a living!). KotlinKotlin has been the official language for Android development for a while now, and Google has even gone so far as to make it the preferred option for Android development. However, many development teams are already deeply in Java, there are many who have decided not to make the transition. Like Java, Kotlin Works Java virtual machine. It is also fully compatible with Java and does not cause file sizes to slow or increase. The difference is that Kotlin requires less boiler code, which means it's a more streamlined and easy-to-read system. It also does away with errors like zero point exceptions and even excuses you from finishing each line with a comma. In short, it's great if you're just learning to develop apps for Android for the first time. So, Kotlin is definitely an easier starting point for beginners, and the fact that you can still use Android Studio is a big plus. However, it's still a complex language in its own right, and you still have to figure out a lot of extra stuff to build an Android app that way. There's also a slight downside to Kotlin less widely used outside of Android development. Find out why you should try Cotlin here. It's fair to say that most people reading this shouldn't choose this route to develop Android apps. Android Studio offers support for C/C' code with Android NDK (Native Development Kit). This means that you will write code that does not work on Java Virtual Machine, but works on the device in your native language and gives you more control over things like memory distribution. For intense apps such as 3D games, this can allow you to squeeze extra performance out of your Android device. This also means that you will be able to use libraries written on C or C. However, it also tends to be much harder to set up, it introduces more bugs, and it is less flexible. And if you want to create a computer game, you'll probably be better off using a ready-made game engine like Unity.C was developed by Microsoft to combine C power and visual basic simplicity. It's very similar to Java, and if you know one of these languages, it will be relatively easy to swap for another. Like Java, the NHS is going to litter, which means you don't have to worry about things like leaking memory and freeing up memory yourself. At the same time, though, the NHS is more modern than Java with cleaner syntax - although it may just be my own bias through. The best language for developing Android apps often just comes down to taste! If you want a particularly easy and welcoming introduction to Android app development, I recommend the combination of C and Unity. Unity is a game engine (meaning things like physics calculations and 3D graphics visualization) and IDE (such as Android Studio). It's a free tool that makes it incredibly easy to create your own games - in just a few lines of code you can have a basic platform game created in less than an hour. Without And it's totally powerful too, being the tool used by most gaming studios on the Google Play Store. As a cross-platform solution, Unity will also allow you to port your games to other other systems such as iOS and Windows. You can even make console games! In addition, development thus provides a very practical way to study object-oriented coding (because objects in this case are actually objects most of the time!). For those hoping to start a career in game development, learning Unity is a great first step. Limit? Unity is useful for creating games, but sub-par to create standard Android apps, especially if you want to match the language of material design of Google. Not interested in unity? Then you could consider Unreal instead (better graphics, less suitable for mobile devices) or simplified game-makers like GameMaker Studio.Also read: What's better? The Unity vs Unreal game engine for Android app development can also be used using Xamarin through Visual Studio. It's more like the traditional Development of Android with the advantage of cross-platform (one codebase for Android and iOS). For a complete beginner, this route is again a bit of a blunt entry point in the Development of Android. But for a small company wanting to create an app for iOS and Android, it makes sense and there is a lot of support and information out there to help you. LUA (Corona)Corona offers another much simpler option for developing android apps, while at the same time giving you a fair amount of energy and control. You'll encode in LUA, which is already much easier than Java. And Corona SDK (Software Development Kit) will make things even easier. It supports all native libraries and allows you to publish on multiple platforms. It is heavily used to create games, but can be used in a variety of other ways too. You will need to use a text editor like Notepad to enter your code and you can run said code on the emulator without even needing to compile in the first place. When you're ready to create ANK and deploy, you'll be able to do so with an online tool. This requires basic coding skills, but offers a pleasant and gentle introduction to the world of code. At the same time though, it's certainly limited in what it can achieve, and it's only a few steps from getting into the builder's application territory. This is more useful for those who want to create something relatively simple and are not so concerned about developing their coding skills or becoming a professional. If you want to use features such as in-app shopping, you will need to pay a fee. The same goes for using the native Android APIs.HTML/CSS/JavaScript (PhoneGap)PhoneGap is powered by Apache Cordova and essendially allows you to create applications using the same code that is commonly used to create a website: HTML, CSS and JavaScript. This is then displayed through A widget that displays the website in the app. PhoneGap acts as a bridge, allowing developers to access some of the phone's main native functions, such as an accelerometer or or or isn't really a true Android development though, and the only real programming would be JavaScript. For many basic tasks, it will do the job, but if you want to be able to claim a true Android app developer (this is the thing), then you have to brave one of the other options on this list. Conclusion So take your choice! Tere has many options for developing Android apps: from Java and Kotlin, to C, C and JavaScript! The right choice will depend on your feelings and your goals, but whatever you decide, you will find that learning to code is a fantastically rewarding experience and one that opens a ton of doors for you. And learning to code with Android is the perfect place to start. Hopefully now you're at least for now a little more on how to develop Android apps, but if you have any questions, shout them out in the comments and our team - and our readers - will do their best to answer them. Good luck! READ ALSO: Introduction to Java syntax to develop Android