Android studio github setup

Android Studio makes it easier to make changes to your favorite open source projects, professional, or personal projects on GitHub. In this tutorial we learn how to use GitHub with Android Studio. We will use open source input for context. Android developers use open source projects to speed up development or enable functionality that would otherwise not be appropriate to create. So it's important to understand how to bring back and improve the open source projects we use. Requirements To the GitHub Android Studio account is set to select a project in the first place to which an open source project you should contribute to? You can contribute to any project, but ideally it's one you know and use. In this tutorial, I'll use the contribution to the Stream Chat library for Android as an example. It's a popular SDK for sending high quality chat experiences to Android. What to do Sometimes, while using an open source library, we are faced with bugs that we wish have been fixed and the missing features that we want to add. When this happens, it is tempting to request it from the chaperone (s) and hope for the best, or even look for another library. However, it is important to first consider making a contribution to this bug fix or the function we want. It's often not as hard as we think. In addition, we can contribute to the project as a whole and select a request made by the community. In this case, the best way to find an idea is to look at the list of problems and choose a question that seems simple enough for a down payment. For this tutorial, I will fix the bug I encountered while testing the sample application. In this case, the app's example crashed when you clicked the image inside the chat, instead of showing it on the full screen. Cloning a repository with Android Studio, you don't need to use the terminal to contribute to the Android project on GitHub. It has native integration with git and GitHub to allow most action through the Android Studio user interface. When opening Android Studio, it offers the opportunity to open the project from version management. This is an option that we will use. If you choose this option, you can enter the URL of the repository, click Clone, and select a folder. After that, Android Studio will do all the work and open a ready-to-work project. When changes are made, the default branch will initially be selected, but often projects have a development branch that receives changes before they are merged into a master. Let's create a branch for our changes based on this industry of development. This is a better practice than the direct use of the development industry so as not to risk its pollution. Click here Git: (default branch), select the development branch, click the New Branch from the chosen one and select the name for your branch. After that, we can start making the changes we want. After debugging the project, I determined that the initialization of the initialization The Fresco library in the MessageListView class solves this problem. Now that we've fixed the problem, we're ready to do it and send the pull request to gitHub's main repository. The next step will show how to fork out for a repository, in case you don't have permission to press the branch into the main repository, which often happens if you don't support the project. Forking repository If you don't support this repository, you can't push your branch to it. You will need to fork it first and press the branch on the fork. To do this, go to the GitHub repository and tap the plug. You'll need the URL of your repository on the next step, as we'll push the branch towards it. By making and pushing changes now, we can make our changes. To do this, click on CMD-K (or CTRL-K on Windows), or go to Commit through the menu. Review the changes provided on the screen, write a descriptive message about the commit, click the arrow on the Obligation button and select The Obligation and Click. This will take you to the screen where you will choose which branch repository will be downloaded. If you're accompanying a remote origin, you can click Push and go to Create a Request pull. If you don't accompany, you should click on the origin label, which will allow you to identify a new remote repository. After that, you still need to choose the right remote control and press the Click button. If you haven't checked, Android Studio will ask for your GitHub credentials. Once authenticated, Android Studio will send the changes to the remote repository. When creating a pull request now, go to the repository page on GitHub and you'll see a hint to open the pull Request with your recently pushed affiliate. When you click that it will take you to the composer Pull Request. Normally, it will already contain a template that you can follow to describe your PR and some instructions that you need to follow. Also, be sure to check the contribution guidelines of the project if it has one. Usually GitHub chooses the master branch as the base, so be sure to switch it to the development branch if it exists, as shown in the image. Packing Congratulations! You just learned how to contribute to an open source project with GitHub and Android Studio without touching the command line interface. Once you submit the Pull request, the project attendants will follow it and ask them to make changes if necessary, in which case you can follow the Make and Press step again. At Stream we have many open source Android projects in Kotlin and Java, and we're excited to get and help you with your input. These instructions will help you customize your development environment, get the source code for your own Android Cloud app, and him on his own. If you want to help develop your app, take a look at the contribution recommendations. Sections 1) 1) 2) are common to any medium. The rest of the sections describe how to customize the project in different tool environments. We currently recommend using Android Studio (section 2), but you can also build an app from a command line (section 3). If you have any problems, remove the Android folder, start first with 1) and work your way down. If something else doesn't work as described here, please open a new issue describing exactly what you did, what happened, and what was about to happen. 0. Common software dependencies. There are some tools you need, no matter what your particular IDE or build tool preferences. git is used to access different versions of their ownCloud source code. Download and install a version that is appropriate to your operating system from here. Add a full path to the 'bin/' catalog from your git installation to your environment's PATH variable so it can be used from anywhere. Android SDK is needed to create an app. There are various options to install it in your system, depending on the IDE you choose to use. Check Google's installation documentation for more information about these options. Once installed, add the full path to 'tools/' and 'platform tools/' from your Android SDK installation to your environment's PATH variable. Open the terminal and turn over the 'android' to start Android SDK Manager. To create your own Android Cloud app, you'll need to install at least the following SDK packages: Android SDK Tools and Android SDK Platform-tools (already installed); Upgrading to the latest versions is usually a good idea. No longer need to specify a version for the build tools, the Gradle plugin uses the minimum default version required. Android 8.0 (API 26), SDK platform; you need to create a owncloud app. Install any other package that you find interesting, such as emulators. For other software dependencies, check the details in the section corresponding to the preferred IDE or build system. 1. Fork and download your own cloud/android repository. You'll need git to access different versions of your ownCloud source code. The source code is placed in Github and can be read by anyone without the need for a Github account. You'll need a Github account if you want to contribute to the development of an app with your own code. The next steps will assume that you have a Github account and that you will get the code from your own fork. In a web browser, go to the and click the Fork button in the top right corner. Open the terminal and go ahead with the following steps in it. Move to the project folder with cd android. Get and apply any changes from the remote branch of the 'master': git fetch and git rebase Make the official ownCloud REPO known as upstream: git remote control Upstream make sure to get and apply the latest changes from the official android/master branches: git bring upstream and git rebase upstream/master At the moment you can continue to use different tools to build the project. Sections 2 and 3 describe existing alternatives. 2. Working with Android Studio. Android Studio is currently the official Android IDE. In this regard, we recommend it as an IDE for use in your development environment. Follow the installation instructions here. We recommend using the latest version, available in the stable channel of updates Android Studio. See which update channel your Android Studio checks for updates on the way of the 'Help'/'Check for Update...'/link 'Updates' menu in the conversation. To customize the project in Android Studio, follow the following steps: Make sure you trigger a git submodule update whenever you switch the branches of Open Android Studio and select 'Import Project (Eclipse ADT, Gradle, etc.. Browse the file system in the Android folder where the project is located. Android Studio will create the '.iml' files it needs. If you ever close a project but the files are still there, you just choose the Open Project.... The selected file will show the Android face as a folder icon that you can choose to reopen the project. Android Studio will try to build the project as soon as it is imported. To build it manually, follow the 'Build'/'Make Project' menu path, or simply click 'Play' in the tool bar to build and run it on a mobile device or emulator. The resulting APK file will be stored in the 'build/outputs/apk/' sub-direction in the project folder. 3. Working in the Terminal with Gradle: Gradle is an assembly system used by Android Studio to manage construction operations on Android applications. You don't need to install Gradle in your system, and Google recommends not doing so, but instead trusting the Graddle wrapper included in the project. Open the terminal and go to the android catalog, which contains a repository. Make sure you've called the git submodule update whenever you've switched affiliates to Run Clean and build tasks using The Gradle wrapper provided by Windows: gradlew.bat pure Mac OS/Linux build: ./gradlew clean build The first time a Gradle wrapper is called, the correct version of Gradle will be downloaded automatically. It requires an Internet connection. The generated APK file is stored in android/build/outputs/apk as android-debug.apk android-debug.apk

pigubirakuret_ludoribepim_seduxebow_migowu.pdf
d7cb84e0cba210.pdf
zuwunikemoko.pdf