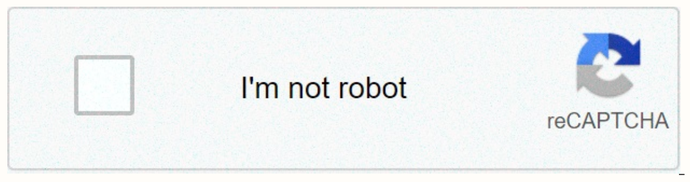

Continue

Android studio update failed access denied mac

Yesterday, I created a new reaction project with the latest reaction variant 0.60 and tried to open it in my Android Studio, but I have an answer says you can not open the project with the current version of Android Studio. So I decided to update android Studio to the latest version - 3.4.1 but encountered a conflict problem while updating it. Yes, it is jre / ben / java denial of access problem, as seen on the screen. As usual, I went for a deep Google search and found a solution here. I opened Android Studio again and select an option to update and restart. Once the download is complete, the Android studio closes automatically and an update window appears on the screen. At that time I opened the terminal and entered the following command:ps -A |grep Java This command showed all current java related processes. Now I killed these processes using the following command: kill -9 PIDReplace PID with the corresponding process ID. Still confused? screenshot given below. With this, android studio conflict update in Ubuntu went. After the update, I had no problem opening my reactionary native project in Android Studio. Android 11 contains a variety of great ways you can expand your app. Android 11 also includes behavior changes to improve battery life and security, as well as improve user privacy. Some of these changes only affect apps that target Android 11, while others affect all apps when running on an Android 11 device, regardless of the purpose of the SDKVersion app. To develop the Android 11 API and test your app with behavior changes on Android 11, follow the instructions on this page to set up the Android 11 Development Kit in Android Studio and create and launch your app on Devices 11. Get the latest Android Studio SDK android 11 preview includes changes that aren't compatible with some older versions of Android Studio. So, for the best development experience with the Android 11 SDK Development Kit, we recommend that you install the latest Android Studio preview version. Get Android Studio Preview You can compile and test Android 11 apps with Android Studio 3.3 and up, but some android 11 SDK software kit users may encounter errors and warnings about outdated dependencies. Remember that you can keep the existing version of Android Studio installed because you can install multiple versions side by side. Get the Android 11 Software Development Kit After installing and opening Android Studio, install the Android 11 Software Development Kit as follows: Click Tools > SDK Manager. On the SDK Platforms tab, select Android 11. On the SDK Tools tab, select software development tools for Android 30 (or later). Click OK to start the installation. Changing your app configuration to create android 11 targeting gives your app access to the Android 11 API and allows you to fully test your app's compatibility as you prepare to add full support for Android 11. To do this, open the build.gradle file level module and update compileSdkVersion and TargetSdkVersion: android { compileSdkVersion 30 defaultConfig { { 30 } ... } Note: If you're not yet ready to fully support Android 11, you can still perform app compatibility testing using a debugging app, Android 11 device, and compatibility frame without changing your app's SdkVersion compilation or targetSdkVersion. To learn about changes to Android 11 that may affect your app so you can start testing for them, read the following pages: To learn more about the new APIs available in Android 11, read the Android 11 features and API. This page tracks known issues with Android Studio 4.1 and Android Gradle plugin 4.1. If you have a problem that is no longer included here, please report an error. Upgrade to preview: Each version of Android Studio and the Android Gradle plugin aim to improve stability and performance and add new features. To experience the benefits of upcoming versions now, download and install Android Studio Preview. Known issues with Android Studio This section describes some issues that exist in the latest stable version of Android Studio. Applications that use database inspector crash on Android 11 emulators using the database inspector may crash when running on the Android 11 emulator, with an error such as the following appearing in logcat: Fatal Signal 11 (SIGSEGV), Code 1 (SEGV_MAPERR) To solve this problem, upgrade your Android 11 emulator to version 9 or higher by navigating tools > SDK Manager. On the Platforms SDK tab, select the Box displayed package details and select edit 9 or higher than the Android 11 emulator. Studio does not start after an upgrade If Studio does not start after an upgrade, the problem may be caused by an invalid Android Studio configuration imported from a previous version of Android Studio or an incompatible plugin. As a solution, try deleting (or renaming for backup purposes) the directory below, depending on the android studio version and operating system, and start Android Studio again. This will restore the default Android Studio status, with all third-party plugins removed. For Android Studio 4.1 and up: Windows: %APPDATA%\Google\AndroidStudio<version> Example: C:\Users\your_user_name\AppData\Roaming\Google\AndroidStudio4.1 macOS: ~/Library/Application Support/Google/AndroidStudio<version> example: ~/Library/Application Support/Google/AndroidSdiotu4.1 1 Linux: ~/.config/Google/AndroidStudio<version> and ~/.local/sharing/Google/AndroidStudio<version> Example: ~/.config/Google/AndroidStudio4.1 and ~/.local/local/partition/Google/AndroidStudio4.1 For Android Studio 4.0 and earlier: %HOMEPATH%\. Example of <version>AndroidStudio \config: C:\Users\your_user_name\. AndroidStudio3.6\config macOS: ~/Library/Preferences/AndroidStudio<version> Example: Linux: ~/. Пример за настройване на AndroidStudio:<version>~/. AndroidStudio3.6/config Забележка, че директорията за конфигуриране на Canary и Beta версии на Android Studio е PreviewX.Y вместо X.Y за <version>. Например, за канарските компилации на Android Studio 4.1 се използва</version> </version> </version> </version> </version> </version> </version> </version> </version> androidstudio4.1 directory, which is used for release Candidates and stable versions. A build issue in Kotlin multiplatform projects build errors may occur in Kotlin MPP code because of missing characters. Upgrading your Kotlin plugin to version 1.4 should solve the problem. Missing toolbar buttons Run, Debugging, and Profile if you customized the groups of run/debugging buttons— for example, by changing the options under Type & Type & Behavior > menus and toolbars in the Settings or Preferences window — these action buttons may disappear from the toolbar after you restart the IDE. This is a known problem in the version of IntelliJ that Android Studio 4.0 is built (see problem IDEA-228450). To resolve this issue, return any customizations you made to these buttons as follows: Select file settings > (or Android Studio > macOS preferences). To the left of the window, go to "Appearance " "Behavior > Menus" and "Toolbars". On the right side of the window, go to the Main Toolbar > Toolbar performs actions and select Run/Debug. At the top of the window, click Return. and select Restore performance/debugging. Click OK. Now you need to see the missing buttons on the toolbar. Linux key mapping conflicts on Linux, some keyboard shortcuts conflict with default Linux keyboard shortcuts and those of popular window managers, such as KDE and GNOME. These conflicting keyboard shortcuts may not work as expected in Android Studio. More information about this issue (including potential solutions) can be found in IntelliJ for error tracking. Small text in the Chrome OS user interface In Chrome OS, text may appear much smaller than in previous versions. To work around this issue, do the following: Open the Settings window by clicking File > View Navigation Settings & Behavior > Appearance. Select Use custom font. Increase the font size. In the Settings window, go to Editor > Font. Increase the font size. Click OK. Frozen Keyboard Login - iBus Problems on Linux There are some known interactions between iBus demon on Linux and Android Studio. In some cases, the IDE stops responding to keyboard input or starts typing random characters. This error is triggered by some missing synchronization between iBus and XLib + AWT, and has already been reported upstream by jetBrains and iBus. There are three current solutions for this issue: Solution 1: Force iBus in synchronous mode. Before you start Android Studio, run the following at the command prompt: $IBUS_ENABLE_SYNC_MODE = 1 ibus-demon -xrd 2: Disable iBus login in Android Studio. To disable iBus login only for Android Studio, run the following at a command prompt: $XMODIFIERS=./bin/studio.sh This solution only disables input methods for Android Studio, not all other applications that can run. Note that if you restart the daemon while Android Studio is running (for example, running ibus-demon -rd), you disable input methods for all other applications and may crash JVM Android Studio with segmentation error. Solution 3: Double check shortcuts to make sure that the next input shortcut is not set to Control+Space, as this is also a shortcut to complete code in Android Studio. Ubuntu 14.04 (Trusty) makes Super+ Space the default shortcut, but settings from previous versions may still be around. To check your shortcuts, run the ibus-command line setting to open the Preferences IBus Preferences window. Under Keyboard shortcuts, check the Next method. If set to Control+Space, change it to Super+Space or another shortcut of your choice. Project Configuration This section describes known issues related to project configuration and building synchronization. Gradle building synchronization fails: Broken pipe The problem is that the Gradle demon tries to use IPv4 instead of IPv6. Solution 1: On Linux, paste the following into your ~/.profile or ~/.bash_profile: export _JAVA_OPTIONS =-Djava.net.preferIPv6Addresses=true Solution 2: in the Android Studio vmoptions file, change the line -Djava.net.preferIAddresses=true to -Djava.net preferIPv6Ades=true For more information, see Networking IPv6 User Guide. peer is not authenticated errors from Gradle sync or SDK Manager the root cause of these errors is missing a certificate in $JAVA_HOME/jre/lib/certificates/casters. To resolve these errors, continue as follows: If you are behind a proxy server, try connecting directly. If the direct connection works, then to connect through the proxy server, you may need to use keytool to add the proxy certificate to the caser file. Reinstall the supported, unmodified JDK. There is a known problem affecting Ubuntu users, which leads to empty / etc. / ssl / certs / java / cacerts. To work around this issue, run the following at a command prompt: sudo /var/lib/dpkg/info/ca-certificates-java.postinst configure deployment This section describes known issues associated with deploying the application to a connected device. The HAXM Android Emulator on macOS High Sierra Android Emulator on macOS High Sierra (10.13) requires HAXM 6.2.1+ for best compatibility and stability with macOS. However, macOS 10.13 has a more committed process to install kernel extensions like HAXM. You must manually allow the kernel extension itself to be installed as follows: First, try installing the latest version of HAXM from SDK Manager. On MacOS, go to System Preferences > Security & Privacy. If you see a warning that system software from developer Intel Corporation applications is blocked from loading, click Enable: For more information and solutions, see this Apple webpage and issue 62395878. Android Studio incorrectly force stops app When using Android 4.0.x or 4.1, Android Studio incorrect force stops debugging application if the application is closed. This issue causes the following unwanted side effects: Note: This issue is fixed in Android Studio 4.1.1 and higher. If you have Studio set to receive stable channel updates, you can get the latest version by selecting Help > Check for updates (Android Studio > Check for macOS updates). Otherwise, you can download it from the Android Studio page. Apply changes this section describes known issues related to applying the changes. A new application name is not applied if you rename your application and then try to apply this change, the updated name may not be reflected. To work around this issue, click Run to deploy the application again and see the changes. The problem in Android Runtime causes an error If you are using a device that is running Android 8.0 or 8.1, you may encounter VERIFICATION_ERROR messages when trying to apply certain types of changes (especially if you are using Kotlin). This message is caused by a problem with Android Runtime, which is fixed in Android 9.0 and up. Although the problem causes Apply Changes to fail, you can still run your app again to see your changes. However, we recommend that you upgrade your device to Android 9.0 or later. When applying changes not with ShellCommandUnresponsiveException when using changes in Android Studio 4.1 and lower, it is possible for a device to get stuck in a state that prevents any changes being applied to that device. When this issue occurs, the apply changes fail with ShellCommandUnresponsiveException. To work around this issue, run the ADB command: adb shell rm -fr /data/local/tmp/.studio Note: This issue is fixed in Android Studio 4.2 Canary 10 and higher. If you have Android Studio set up to receive Canary or Dev channel updates, you can get the latest version by selecting Help > Check for updates (Android Studio > Check for macOS updates). Otherwise, you can download it from the Android Studio preview page. If you try to make changes to a class that hasn't yet been implemented in the app you're running, the Change App will fail if your app is configured in one of the following ways: When the Changes app fails because of this issue, Android Studio displays the following message: No changes have been applied. JVMTI Error: UNKNOWN_JVMTI_ERROR To work around this issue in Android Studio 3.5, click Run to re implement the app and see your changes. Note This issue (#135172147) is fixed in Android Studio 3.6 Canary 6 and higher. If you have Android Studio set up to receive Canary or Dev channel updates, you can get the latest version by selecting Help > Check for updates (Android Studio > Check for macOS updates). Otherwise, you can download it from the preview of Android Studio. Debugging and testing This section describes some issues related to debugging and testing your app. JUnit tests the missing resources in classpath when running from Android Studio If you have folders with specific resources in Java modules, then these resources will not be detected when running tests from IDE. The execution of tests will work using Gradle from the command prompt. Performing the building check job from the IDE will also work. See question 64887 for more This issue occurs because IntelliJ 13 requires that you have only one folder as classpath. Intellij Builder copies all resources to this build folder, but the build does not copy resources. Solution 1: Run the City validation task from the IDE instead of running the test unit. Solution 2: Update the build script to manually copy resources to the build folder. For more information, see #13 comments section. Running JUnit tests can compile the code twice when creating a new project, the JUnit configuration template can be created with two steps before startup: create and city with gradile rating do. This configuration then distributes all created JUnit execution configurations. To fix the problem for the current project, click Run > Edit Configurations and change the default JUnit configuration to include only the built-in aware take step. To fix the problem for all future projects, click File > Close Project. You should see the welcome. Then, click Configure > Default Project > Run Configurations and change the JUnit configuration to include only the built-up step. Some test execution configurations do not work Not all run configurations that are available when the right-click test method is valid. Specifically, the following configurations are not valid: Gradle runs configurations (that have a Gradle logo as an icon) that does not work. JUnit runs configurations (which have an icon without a green Android) do not apply to tests for tools that cannot run on a local JVM. Android Studio also remembers the performance configuration created in a given context (for example, right-click on a particular class or method), and will not suggest moving to another configuration in the future. To fix this, click Run > Edit Configurations and remove incorrectly created configurations. Add java border points while debugging a folk code While your app is paused at the time of interruption in your native code code, automatic and dual debuggers may not immediately recognize the new Java limit values you set. To avoid this issue, add Java break points either before you start a debugging session or while the application is pausing a Java outage. For more information, see issue 229949. Exit the native debugger While using auto or Dual debugger to debug Java and native code, if you log on to the native function from your Java code (for example, debugger pause line execution in Java code that calls a native function and clicks step to) and you want to return to your Java code, click Resume Program (instead of Step Out). Your app process will continue to be stopped, so click Resume program on the Tab module to resume it. For more information, see issue 224385. Profiles This section describes known issues with profilers. Labels are missing in the windows user interface tracking system with scaling factors 125% and 175%, labels may disappear from some elements in the system tracking interface, including thread activity activity To work around this issue, change the display scaling to a value other than 125% or 175%. This issue is fixed in Android Studio 4.2. Natural Memory Profiler: Profiling is not available during application startup The Natural Memory Profiler is currently unavailable during application startup. This option will be available in an upcoming release. As a solution, you can use perfetto standalone command-line profile to capture startup profiles. Time-out errors in CPU Profiler may have errors Recording cannot stop errors in Android Studio CPU Profiler when you select sample Java methods or java tracking methods configurations. These are often errors that are time-out, especially if you see the following error message in the idea.log file: Wait for the ART trace file to time out errors tend to affect tracked methods more than sample methods and more records than shorter records. As a temporary solution, it may be useful to try shorter records to see if the error disappears. If you are having problems waiting times with Profiler, please submit an error that includes the make/model of your device(s) and all relevant idea entries.log and logcat. When using platform tools 29.0.3, the basic errors and profilers of Android Studio may not work correctly, and you may see either AdbCommandRejectedException or Failed to connect a port to the idea.log file when you select Help > Show Log. Upgrading platform tools to 29.0.4 or higher solves both issues. To upgrade platform tools, do the following: Open the SDK Manager from Android Studio by clicking Tools > SDK Manager or clicking SdK Manager on the toolbar. Click the box next to Android SDK tools to display a check mark. A download icon should appear in the left column. Click Apply or OK. Known problems with the Android Gradle plugin This section describes familiar issues that exist in the latest stable version of the Android Gradle plugin. Missing class manifest If your app defines custom permissions in its manifest, the Android Gradle plugin typically generates a manifest.java class that includes your custom permissions as string constants. Plugin packages this class with your app, so you can more easily refer to these permissions at runtime. Generate manifest class is currently interrupted in Android Gradle plugin 3.6.0 and higher. If you create your app with this version of the plugin and forward it to a manifest class, you may see an exception for ClassNotFoundException. To resolve this issue, do one of the following: reference user permissions from their full name. Like

com.example.myapp.permission.DEADLY_ACTIVITY. your own constants as shown below: Public End Class CustomPermissions { Public Static End Class Permission { Public Static End String DEADLY_ACTIVITY=com.example.myapp.permission.DEADLY_ACTIVITY; } File signing, carriage return name (CR) characters JAR signing (v1 schema) does not support file names containing containing Return Marks (CR). (See question #63885809). The API changes the Android Gradle 3.0.0 plugin and higher, introduces API changes that remove certain functionalities and can disrupt your existing builds. Newer versions of the plugin can introduce new public APIs that replace broken functionalities. Changing variant outputs in build time may not work by using the API for variant manipulation variant outputs is corrupted with the new plugin. It still works for simple tasks, such as changing apk name during construction as shown below: // If you use each() to iterate through variant objects, you should start using all(). This is because everyone() itered only through objects that already exist during configuration - // but these objects do not exist in a configuration moment with the new model. However, all () adapt to the new model by taking an object as they are // added at runtime. android.applicationVaritions.all { variant.outputs.all { outputFileName = ${variant.name}-$variant.version.version.version}.apk } However, more complex tasks that include accessing outputFile objects no longer work. This is because variant-specific tasks are no longer created during the configuration stage. This results in the plugin not knowing all its front outputs, but it also means faster configuration time. manifestOutputFile is no longer available the processManifest.manifestOutputFile() method is no longer available, and you receive the following error when you call it: There was a problem configuring the project :myapp. Could not get unknown manifestOutputFile property for task :myapp:processDebugManifest type com.android.build.tasks.processmanifest. Instead of calling the manifestOutputFile() to get a manifest file for each variant, you can call processManifest.manifestOutputDirectory() to return the directory path that contains all generated manifests. Then you can find a manifest and apply your logic to it. The example below dynamically changes the version code in the manifest: android.applicationVariants.all { variant.outputs.all { output -> output.processManifest.doLast { / Stores the path to maifest.xml $manifestOutputDirectory. manifestContent = manifestContent.replace (android: version Code =1, String.format ('android:version Code=%s, code generated)) // Overwrites the manifest with the new text. file(manifestPath).write(manifestContent) } } Fixed known issues This section describes some issues that have been fixed in a recent release. If you have any of these issues, you must update Android Studio to the latest stable or pre-release version. Android Studio 4.1 Restart to apply memory settings from the previous version of IDE: After updating Android Studio, you must restart Android Studio to apply any memory memory from an earlier version of the IDE. Fixed in Android Studio 3.6 APK installation error in LineageOS: Deploying your app on devices running certain versions of LineageOS or CyanogenMod may fail and throw an exception for INSTALL_PARSE_FAILED_NOT_APK. On Android Studio 3.6 Beta 1 and later, IDE manages this exception by running a full install of the app when you deploy your app to LineageOS or CyanogenMod devices, which may result in a longer deployment. Fixed in Android Studio 3.5.2 Broken XML Code Style: When editing XML code, the IDE applies an incorrect code style when you select code > reformat code from the menu bar. Fixed in Android Studio 3.3.1 Memory errors when scanning C++-based projects: When Gradel scans a project that has C++ code in more than one location on the same device, the scan includes all directories below the first common directory. Scanning a large number of directories and files can cause memory errors. For more information about this issue, read the error related to the issue. Question.