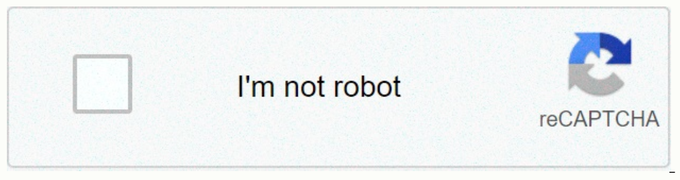

Continue

Emulateur android sur kali linux

Android is an operating system for mobile devices developed by Google, built on the Linux kernel. Android competes with Apple's iOS (for iPhone/iPad), RIM's BLACKBERRY, Microsoft's Windows Phone, Symbian OS and many other proprietary mobile OSes.The other. The latest Android supports Phones/Tablets, TVs, Wear (watches and glass), Cars and the Internet of things. Android is based on Linux with a set of root core C/C++ libraries. Android apps are written in Java. However, they run on Android's Java Virtual Machine, called Dalvik Virtual Machine (DVM) (instead of JDK's JVM) optimized to work on small and mobile devices. In May 2017, Google announced support for Android app development in the Kotlin programming language, supported in Android Studio 3.0. Kotlin will not be discussed in this article. Follow these steps to install Android Studio on Chrome OS: If you haven't already, install Linux for Chrome OS. Open the File app and locate the DEB package you downloaded in the Downloads folder under My Files. Right-click the DEB package and choose Install using Linux (Beta). January 30, 2019 Run Android apps on Linux with these add-ons. Well those are the best Android models for Linux that we think you can check out. Whether you want to test an app you're developing, or you just want to try a new Android app on your Linux PC, these simulators must meet your expectations. The parent site for Android is . For programmeers and developers, you website to download SDKs, Android Training, API Guides, and API documentation. Installing Android software is probably the most difficult part of this project - for those who are unlucky. It takes time - from 30 minutes to n hours to forever - depending on your luck (in fact, your IT knowledge) and your computer. You may need a decent computer (with 8GB of RAM) and 10GB of free disk space to run the Android simulator!!! Running on actual Android devices (phones, tablets) requires much less resources. Before installing android SDK, you need to install Java Development Kit (JDK). Read 'How to install JDK'. Make sure that your JDK is equal to or higher than 1.8. You can check your version of JDK with the command 'javac -version'. Uninstall the older version(s) of 'Android Studio' and 'Android SDK', if available. The installation and multiple operations take a long time to complete. DO NOT stare at your screen or at the ceiling. Browse 'Android Developer' @ . For developers, check out 'Developer Guide'. We need to install: Android Studio, which is an integrated development environment (IDE) based on IntelliJ (a popular Java IDE); andAndroid Software Development Kit (SDK) to develop Android. Reference: 'Install Android Studio' @ the JAVA_HOME is set to the JDK installation folder via the 'set JAVA_HOME' command. If not, follow the steps HERE. Check out the requirements for Android Studio/SDK @ example: For Windows 10, recommend 8GB of RAM, 4GB of disk space, and a screen resolution of at least 1280x800. Goto 'Android Studio' @ ⇒ Click 'Download Android Studio 3.3.x for Windows 64 bit (948MB)', for example android-studio-ide-182.xxxxxxx-windows.exe.Run downloader ⇒ You can watch a short video @ ⇒ In 'Select Ingredients', select 'Android Studio' and 'Android Virtual Device' ⇒ In 'Android Studio Location Settings', accept the default 'C:Program FilesAndroidAndroid Studio' ⇒ Follow the on-screen instructions and accept the default to complete the installation. You need about 3-4GB of free disk space! Take note (and take pictures) on the installation locations of 'Android Studio' (by default @ 'C:Program FilesAndroidAndroid Studio') and 'Android SDK' (by default @ c:UsersusernameAppDataLocalAndroidSdk). Check system requirements @ for example: For Mac OS X 10.10 to 10.13, recommend 8GB of RAM, 4GB of disk space, and a minimum screen resolution of 1280x800. Goto 'Android Developer' @ ⇒ Select 'Download Android Studio' ⇒ 'Download Android Studio 3.3.x for Mac (997MB)', for example, android-studio-ide-182.xxxxxxx-mac.dmg.Launch the downloaded '.dmg' installation file ⇒ You can watch the short video @ and drop Android Studio into the Apps folder. The installation will continue to step 2 below. Note: If you see a warning that the package is damaged and will be moved to the recycle bin, visit ⇒ Security & Privacy system options ⇒ under Allow apps to be downloaded from ⇒ select Anywhere. Then run again. 'Android SDK' will be installed in '~/Library/Android/sdk' by default, which ~ shows your main folder. Note: Adding too many SDK packages, especially so-called system images to simulate different devices (e.g. different phones/tablets), will take a very long time, especially if people are downloading and jamming the network. The system image also takes up a lot of disk space – a few GBytes per API level!!! For our erthe er000 project, we only need a small set of SDK packages. We have to go. Check if it is possible to copy the SDK instead of downloading 1GB during installation? Android Studio launch ⇒ It will run the 'setup' wizard for the first boot ⇒ not enter previous settings ⇒ In 'Welcome', select ⇒ 'next' In 'Installation type', select 'Standard' (default) ⇒ In 'Select UI Theme', select 'IntelliJ' (default) ⇒ Be aware of the SDK folder, by default @ c:UsersusernameAppDataLocalAndroidSdk ⇒ Finish. This step will download another 1GB SDK package and take a long time to complete. Note: In Windows, 'AppData' is a hidden folder. You need to choose to ⇒ 'Display hidden items' to view this folder. (Optional) You can check package installed by selecting 'Configuration' (at the end of Android Studio) ⇒ 'SDK Manager' ⇒ 'Android SDK' (side bar): In the tab 'SDK Platform': Android 8.1 (Oreo) (API Level 27)In the tab 'SDK Tools': Android SDK 29-rc1Android Emulator (27.1.12)Android SDK Platform-Tools (28.0.2)Android SDK Tools (26.1.1)Intel x86 Emulator Accelerator (HAXM installer) (7.3.2) Support Repository (ConstraintLayout for Android, Solver for ConstraintLayout, Android support, Google Repository ⇒ It will run the 'setup' wizard for the first launch ⇒ do not import previous settings ⇒ In 'Welcome', Google Repository launch Android Studio ⇒ It will run the 'setup' wizard for the first launch ⇒ do import previous settings ⇒ In 'Welcome' , Google Repository)Launch Android Studio ⇒ It will run the 'setup' wizard for the first launch ⇒ do import previous settings ⇒ In 'Welcome', Google Repository)select ⇒ 'next' In 'Installation type', select 'Standard' (default) ⇒ Note SDK folder, by default @ '/User/username/Library/Android/sdk' (also known as '~/Library/Android/sdk') ⇒ End. This step will download another 1GB SDK package and take quite a while to complete. (Optional) You can check the SDK packages installed by selecting 'Configuration' (at the end of Android Studio) ⇒ 'SDK Manager' ⇒ ⇒ 'Android SDK' (side bar): In the tab 'SDK Platform': Android 8.1 (Oreo) (API Level 27)Under the tab 'SDK Tools': Android SDK 29-rc1Android Emulator (27.1.12)Android SDK Platform-Tools (28.0.2)Android SDK Tools (26.1.1)Intel x86 Emulator Accelerator (HAXM installer) (7.3.2)Android Support Repository (ConstraintLayout for Android, Solver for ConstraintLayout, Android support, The Google Repository)Android app is written in Java and uses XML extensively. I would assume that you have basic knowledge of Java and XML.Take note that Android is slow - VERY SLOW!!! Be patient!!! Goto 'Android Guide' @ . Read Build your first app. Launch Android Studio. Select Start a new Android Studio project. Under 'Choose your project', select 'Phone and tablet' tab ⇒ 'Empty activity' ⇒ Next.In 'Configure your project' ⇒ Set 'Name' to 'Hello Android' (this will be 'Title' in your phone's app menu) ⇒ 'Package name' and 'Save location' will be updated automatically ⇒ In 'Language', select 'Java' ⇒ Leave 'Minimum API Level' and the rest to default ⇒ Finish. Be patient! It may take a few minutes to set up your first app. See the 'progress bar' in the bottom status bar and Zzzzzzzzz..... After the progress bar says complete, a hello-world app is created by default. To run your Android app under the simulator, you first need to create an Android Virtual Device (AVD). AVD models a specific device (e.g. your jPone or Taimi). You can create AVD to simulate different Android devices (e.g. phones/tablets, Android version, screen size, etc.). In android studio, select 'Tools' ⇒ Android ⇒ AVD Manager. See 'Common errors' below if you can't find 'AVD Manager'. Click 'Create virtual device'. Under 'Choose device definition' ⇒ In 'Category', select 'Phone' ⇒ In 'Name', select '2.7 QVGA' (the smallest device available - you can try a larger device later) ⇒ Next.In Next.In Photo: ⇒ ⇒ Select the version with the highest API level ⇒ Click 'Download' ⇒ Next.In 'AVD Name', type '2.7 QVGA API 27' (default) ⇒ Finish.If you see 'VT-x disabled in BIOS': Check your BIOS settings to make sure 'Virtualization Technology' is turned on. Shut down and restart your computer to enter BIOS setup. This is a dependency machine. Google 'Your-PC-brand-and-model enter BIOS setup'. For example, for my HP computer ⇒ Boot ⇒ 'ESC' to enter the BIOS ⇒ Advanced ⇒ System Options ⇒ Check 'Virtualization Technology (VTx)' ⇒ Save ⇒ Exit.Select the 'Run' menu ⇒ 'Run app' ⇒ Under 'Available Virtual Devices', select '2.7 QVGA API 27⇒ OK. Follow the instructions to install HAXM. Be patient! It may take a few minutes to activate the app on the simulator. First you'll see a Google ⇒ that's 'Android' ⇒ then a 'wallpaper' ⇒ followed by the message 'Hello, world!'. If you have problems running on the fake, I suggest you try running on an actual Android device (phone/pad) if you have one. Goto the next step. DON'T PLAY THE I.A., as it really takes a long time to get started. You can always run the app again (or run a new one) on the same simulator. Try running the app again by selecting the 'Run' menu ⇒ 'Run the app'. Common error:If things don't work out, select the 'Files' menu ⇒ 'Disable Cache / Restart...' ⇒ and waiting ... 'AVD management' can't be found in the 'Tools' menu: You may not have enough packages needed for the project. You'll have a window with Gradle alerts with a link that you can click on and you'll see a window with reminders to download missing packages. When all download tools 'ADV manager' will be activated. If you receive an error message 'Target could not be found using the 'android-26' hash string.' Click the 'Install missing platform(s) link and sync project' to install API-26 (download another GB!); or In 'Gradle Scripts' ⇒ Open 'build.gradle (Module: app)' ⇒ Change 'compileSdkVersion' and 'targetSdkVersion' from 26 to 27 (we installed API-27) and 'com.android.support:appcompat-v7:26.x.x' to '27.0.0'. If you receive an error message 'Download re-dependency and project sync (network request)', click the link to download it. If the message appears again: (Windows) Goto 'C:Usersusername.gradlewrapperdists' and delete 'gradle-x.x-all'. Note '.gradle' is a

hidden folder and you need to allow viewing of hidden folders. (Mac) Goto '~.gradlewrapperdists' and delete 'gradle-x.x-all'. Note '.gradle' is a hidden folder and you need to allow viewing of hidden folders. Restart Android Studio. This error is caused by poor network conditions, resulting in a corrupted download. If you get the error 'Emulation: ERROR: x86 emulation is currently required hardware acceleration', Read ' , (6) and If you get the error 'HAX doesn't work...' ⇒ 'SDK' 'SDK' ⇒ SDK Tools ⇒ Check that 'Intel x86 Emulator Accelerator (HAXM Installer)' is installed ⇒ Goto SDK Location (by default, 'C:Usersyour-usernameAppDataLocalAndroidSdkextrasintelHardware_Accelerated_Execution_Manager' for Windows or '~/Library/Android/sdk/extras/intel/Hardware_Accelerated_Execution_Manager' for Mac OS X) ⇒ runs 'intelhaxm-android.exe' to install HAXM ⇒ Be patient! The installer may take a while to ⇒ according to the screen instructions to complete setup. Note that: (a) In Windows, the 'AppData' folder is hidden. You'll need to unside through 'Control Panel' ⇒ 'Folder Options' (or 'File Explorer Options' in Windows 10) ⇒ Check 'Show hidden files, folders, and drives'. (b) In Mac OS X, the 'Library' folder is hidden. You can unside through the 'Finder' ⇒ Go ⇒ Home ⇒ Settings ⇒ Show View Option. If the problem persists, remove and then reinstall. If you get an error 'Intel Virtualization Technology (VT-x) isn't turned on' ⇒ Check your BIOS settings to make sure 'Intel Virtualization Technology' is turned on. Shut down and restart your computer to enter BIOS setup. This is a dependency machine. Google 'Your-PC-brand-and-model enter BIOS setup'. If 'Intel Virtualization Technology' has been activated, this error may have been caused by your antivirus software. Disable your antivirus for this session and run again. See you're having trouble creating AVD through 'AVD Manager' (Mac OS X encounters an 'Unexpected Exit Studio' error), opens the AVD manager via the command line as follows: Reference: 'Run your app', 'Run on real device' @ runs Android app on REAL device (Android phone or Tablet):Connect real device to your computer. Make sure you have 'USB Drivers' installed for your device on your computer. Otherwise, goto INSTALL OEM USB Drivers. If you device is not certified there, good luck! It took me hours to find a compatible driver for my cheap un-brand tablet. Turn on 'USB Debugging' mode on your real device: (On Android 4.2/5.0 or later) 'Developer options' should be turned on through 'Settings' ⇒ About ⇒ Scroll to the bottom and tap 'Build number' seven (7) times until 'Developer Mode' is displayed. Go back to the previous screen to find 'Developer options' ⇒ Open 'Developer Options' ⇒ Turn on 'USB debugging'. (On Android 4.0) From 'Settings' ⇒ 'Developer Options' check ⇒ 'USB debugging'. (On Android 3.2 and above) From 'Settings' ⇒ 'Apps' ⇒ 'Development' ⇒ 'USB Debugging' test. This allows android SDK to transfer data between your computer and your device. Also allow 'Unknown source' from 'Application'. This allows the installation of applications from unknown sources on the device. You'll see the message 'USB Debugging when you plug a USB cable into your computer. From Android Studio, Studio, The 'Run' ⇒ 'Run apps' ⇒ Your device will be listed under Select a running device ⇒ Choose a ⇒ OK. To delete a project, select 'File' ⇒ 'Close Project' ⇒ On 'Recent Project' ⇒ Hover over project ⇒ Press the 'Delete' key on the project to remove the project from Android Studio ⇒ Then you can delete the project folder from the file system. There are two ways to create a user interface (UI) on Android: (1) Write Java code; (2) Layout via XML description and let the system create Java code for you. Let's start by writing Java code (because I suppose to teach you programming). We'll continue from the previously created 'Hello Android' project. Expand the 'app' button. Expand the 'java' button. Extend the 'com.example.helloandroid' package button. Open 'MainActivity.java' (actually opened). REPLACE the onCreate() method as follows and add the import report. Don't touch the rest of the code. Run the app ('Run' ⇒ 'Run the app'). You'll see the message 'Hello, from my Java code!' displayed. An Android app that can have one or more Activity.An activities, usually with a screen, is a unique, focused thing that users can interact with (hereby called activity). MainActivity expands the android.app.Activity class (or android.support.v7.app.AppCompatActivity in a newer version) and overrides the onCreate(). OnCreate() is a recall method, recalled by the Android system when the operation is launched. A view is a user interface component (or utility or control). We build a TextView (which is a subclass view to display a text message), and set its text. Then we set the content view of the MainActivity screen to this TextView.Each Android app has a manifest file called AndroidManifest.xml under 'app' ⇒ 'manifests'. It describes the Android app. For example, our 'Hello Android' app, with an activity called MainActivity, has the following expression (generated automatically by Android Studio when the project was built):P sy part <manifest>zi specify the package name. Contains <manifest>part <application>death. <application>zi specify the icon, label (title of the app) and its theme. It contains a prime ore. This app has an operation. The <activity>factor claims its program name ('MainActivity' in the current package '.'). It may contain <intent-filter>. The <intent-filter>claims that this activity is the entry point (android.intent.action.MAIN) of the app. This activity will be added to the app launcher (android.intent.category.LAUNCHER). Instead of writing Java code to create a user interface (UI) (as in the example above using a TextView component). It is more flexible and therefore recommended to layout your UI components through a described XML layout file. This way, you don't need to hardcode the views, and you can easily modify the cůa</intent-filter> </intent-filter> </activity> </activity> </application> </application> </manifest> </manifest> </manifest> by editing the XML marker. Therefore, Java codes can focus on business logic. Let's reww our hello-world to use the XML layout. CLOSE the previous project, via 'File' ⇒ 'Close the project' (Always close the previous project before starting a new project).' Start a new Android Studio project' ⇒ 'Phones and Tablets' ⇒ 'Empty Activity' ⇒ 'Name' to 'Hello Android XML'. Expand the 'apps' button, 'res (resources)', 'layout'. Open 'activity_main.xml' (actually opened). Android Studio offers two views for this XML file: 'Design (or Graphics)' and 'Text (or XML)' - which can be selected at the bottom of the dashboard. Select 'Text' mode and study the code:It claims a TextView (text field) contains a text string 'Hello World!'. The TextView component has a width and height large enough to hold its contents ('wrap_content'). Instead of encoding the Hello-World string directly inside TextView (as in the XML file above), we'll use a string reference (similar to a variable) for more flexibility. Expand the res/values button. Open the string.xml and ADD a red line:This 'string.xml' defines 2 references/values:A reference string (variable) 'app_name' contains the name of the application, which you entered when you created the project. The string reference (variable) 'hello' contains the value 'Hello world from XML!'. Now, modify 'activity_main.xml' to use the string reference 'hello', in the '@string/hello' format, as follows:Next, check 'MainActitivy.java' (under app/java/com.example.helloandroidxml), as follows:'MainActivity' sets the content view to 'R.layout.activity_main', mapped to the XML 'reslayoutactivity_main.xml' layout file that we modified earlier. Run the app. You'll see the new string 'Hello, from XML!' displayed. REFERENCES & RESOURCESAndroid Developer @ Android Guides' @ 'API Reference' @ . .