

Design and Implementation of a Calculator App on Android

Jnanesh Satisha Karmar, Vivek K Kumar

Department of Electrical Engineering

IIT Hyderabad

Email: ee25btech11029@iith.ac.in, ee25btech11062@iith.ac.in

I. MOTIVATION

This project goes over the possibility of development of Android apps on Android devices through Termux and Android SDK. The objective is to create an android application containing a webview which can load HTML strings. A wide variety of UI and functionalities can be implemented through this method. We specifically implement a full-stack calculator app in this case, with HTML, CSS and bootstrap, vanilla Javascript running on a webview in an android app on the front-end entirely within the limits of an android device. We also have a local server running entirely within the device, accepting requests in the backend. In this report we emphasise on the instructions to get such a calculator app running on an android device.

II. PREREQUISITES

Set the Termux application up on your android device. The instructions to do so can be found here - <https://github.com/gadepall/fwc-1>

Inside debian, you will need to install a jdk (Java Development Kit) and gcc (a C Compiler).

```
1 apt update
2 apt install openjdk-17
3 apt-get install gcc
```

III. SETTING UP ANDROID SDK

Download the SDK from the official Android Developers website

<https://developer.android.com/studio#~:text=Command%20line%20tools%20>.

Copy the link for linux platform and download it inside a folder named android, preferably inside home directory in debian using the following commands

```
1 cd ~
2 mkdir android
3 cd android
4 wget <your zip link>
```

Inside the android folder, unzip the file you just downloaded

```
1 mv ~/commandlinetools-XXX_latest.zip ./
2 unzip commandlinetools-XXX_latest.zip
3 rm commandlinetools-XXX_latest.zip
```

Now, we need to slightly reorganize the folder structure. Inside the android folder, perform the following changes.

```
1 cd cmdline-tools
2 mkdir tools
3 mv * tools
```

Edit the .bashrc file to include the \$ANDROID_HOME environment variable using the text editor of your choice.

```
1 nvim ~/.bashrc
```

Add these lines to the .bashrc file

```
1 export ANDROID_HOME=$HOME/android
2 export PATH=$ANDROID_HOME/cmdline-tools/tools/bin
  /:$PATH
3 export PATH=$ANDROID_HOME/emulator/:$PATH
4 export PATH=$ANDROID_HOME/platform-tools/:$PATH
```

Close the text editor, exit the Debian environment, and log back into Debian.

```
1 exit
2 proot-distro login debian
```

The following command should execute successfully.

```
1 sdkmanager
```

If the command was not found, then try repeating the above steps again.

Now accept the license agreement for using sdkmanager

```
1 yes | sdkmanager --licenses
```

You have to install a few tools using sdkmanager to entirely be able to start developing apps.

```
1 sdkmanager --install "platform-tools" "platforms;
  android-34" "build-tools;35.0.0"
```

We now need to install aapt2 for termux (android asset packaging tool). It is important to note that we need to do this outside of proot and directly on termux.

```
1 exit
2 pkg install aapt2
```

We can continue the rest of the installation inside of proot now. We now need to install gradle to be able to build the apk later.

```
1 proot-distro login debian
2 apt install gradle
```

IV. CREATING THE ANDROID APP

Create the android app by running the following script.

```
1 cd ~
2 wget https://raw.githubusercontent.com/Vivek-Kumar-1907/ee1003/refs/heads/main/calculator/androidapp/init.sh
3 chmod +x init.sh
4 ./init.sh
```

The script is interactive, and you have to type the details of the app like name and package name.

Run the following command to generate the APK. Building using Gradle Wrapper (gradlew) may take some time during the first run (approximately 10 minutes). Subsequent builds will be significantly faster.

```
1 ./gradlew assembleDebug
```

The apk will be generated. This can be executed by simply opening the apk in the file manager of your choice. To copy the apk into the sdcard the following command can be used.

```
1 cp app/build/outputs/apk/debug/app-debug.apk /sdcard/
```

The script in the beginning of this section is for making a general app using Webviews.

For making a calculator app (without the backend), copy the code in the following link and paste it into the index.html file generated using the script.

```
1 cd app/src/main/assets
2 wget https://raw.githubusercontent.com/Vivek-Kumar-1907/ee1003/refs/heads/main/calculator/androidapp/index.html
```

You will have to go back to the original directory for running gradlew again. A clean-up might be required.

The HTML file represents the front-end layer of the application and is implemented using Bootstrap and vanilla JavaScript. It is rendered through the WebView component embedded in the native Android application.

V. RUNNING THE SERVER

The code for the server is available in the repository

<https://github.com/Vivek-Kumar-1907/ee1003/calculator/>

Just compile and execute the C code which will host the server locally on port 8080.

```
1 cd backend
2 gcc main.c -o main
3 ./main
```

It is worth noting that the server needs to constantly be running in the background. Make sure you don't stop the execution of the code. You might also need to disable third-party network security applications(not necessary in most phones) to allow the server to receive requests from external sources.

You can now return back into the calculator app and use it.

VI. ACKNOWLEDGEMENTS

We thank Dr. GVV Sharma for inspiring and truly supporting us to pursue this project and making it possible.

The steps required to install and set up Android SDK on termux, was a modified version of the steps required to set up Android SDK in general without the use of Android Studio which we found here: <https://proandroiddev.com/how-to-setup-android-sdk-without-android-studio-6d60d0f2812a>

The script for creating an app with a web-view on termux was inspired by u/AddressSpiritual9574's script for generating boilerplate code for a basic android app without the use of Android Studio.

We also would like to thank Subhodeep Chakraborty for his true support.