React Native | Setting up the development environment

Development OS - Linux
Target OS - Android

Prerequisites

Before setting up the React Native development environment, ensure that the following prerequisites are met:

- Node.js and npm (Node Package Manager) installed on your machine.

If you haven't installed Node.js and npm, you can download them from the official Node.js website.

Step 1: Install Java Development Kit

React Native currently recommends version 17 of the Java SE Development Kit (JDK).

1. Download OpenJDK from AdoptOpenJDK.
2. Extract the downloaded JDK 17 archive file.
3. To configure the java environment:
    - Move the extracted JDK 17 folder to the “/opt” directory:

      ```
      $ sudo mv jdk-17 /opt/
      ```

    - Move the extracted JDK 17 folder to the “/opt” directory:

      ```
      $ sudo tee /etc/profile.d/jdk17.sh <<EOF
      ```

- Type below statements to add the variable path for JDK 17 on your Ubuntu 22.04 system:

```
> export JAVA_HOME=/opt/jdk-17
> export PATH=\$PATH:\$JAVA_HOME/bin
> EOF
```

- Write out the “source” command to execute the profile file “jdk14.sh”:

```
 $ source /etc/profile.d/jdk14.sh
```

4. Verify JDK version with the following command:

```
 $ java -version
```

Step 2: Download and Install Android studio

React Native allows you to build native mobile applications for both iOS and Android platforms. To do this, you will need to install the necessary development tools and SDKs for the platform you are working with.

1. **Install Android Studio**
    - Download Android Studio.
    - While on Android Studio installation wizard, make sure the boxes next to all of the following items are checked:
        - Android SDK
        - Android SDK Platform
        - Android Virtual Device
    - Click "Next" to install all of these components.

2. **Install the Android SDK**

   Android Studio installs the latest Android SDK by default. Building a React Native app with native code, however, requires the Android 14 (UpsideDownCake) SDK in particular. Additional Android SDKs can be installed through the SDK Manager in Android Studio.

   - Open Android Studio, Go to *settings*, under *Languages & Frameworks* select *Android SDK* from the left panel.
   - Select the "SDK Platforms" tab from within the SDK Manager, then check the box next to "Show Package Details" in the bottom right corner. Look for and expand the `Android 14 (UpsideDownCake)` entry, then make sure the following items are checked:
     - `Android SDK Platform 34`
     - `Intel x86 Atom_64 System Image` or `Google APIs Intel x86 Atom System Image`
   - Next, select the "SDK Tools" tab and check the box next to "Show Package Details" here as well. Look for and expand the "`Android SDK Build-Tools`" entry, then make sure that `34.0.0` is selected.
   - Finally, click "Apply" to download and install the Android SDK and related build tools.

3. **Configure the ANDROID_HOME environment variable**

   The React Native tools require some environment variables to be set up in order to build apps with native code.

   Add the following lines to your `.bash_profile`, `.bashrc`, or `.zshrc`, config file. (On the terminal,, run `nano .bashrc` to open the config file):

   ```
   export ANDROID_HOME=$HOME/Android/Sdk

   export PATH=$PATH:$ANDROID_HOME/emulator

   export PATH=$PATH:$ANDROID_HOME/platform-tools
   ```

   After saving the changes, restart your terminal or run `source ~/.bashrc` , `source ~/.zshrc` or `source ~/.bash_profile` to load the config into your current shell.

   Verify that ANDROID_HOME has been set by running `echo $ANDROID_HOME` and the appropriate directories have been added to your path by running `echo $PATH`

Step 3: Install Watchman

Watchman is a tool by Facebook for watching changes in the filesystem.

To install watchman, we will use Homebrew's Watchman package. If you do not have homebrew installed, you can install it from here.

We can now use homebrew to install watchman:

```
$ brew install watchman
```

Confirm successful installation by running:

```
$ watchman version
```

Step 4: Create and Run a React Native Application

1. Create a new application

   Now that you have the necessary tools installed, you can create a new React Native project using the following command:

   ```
   $ npx react-native@latest init MyReactNativeApp
   ```

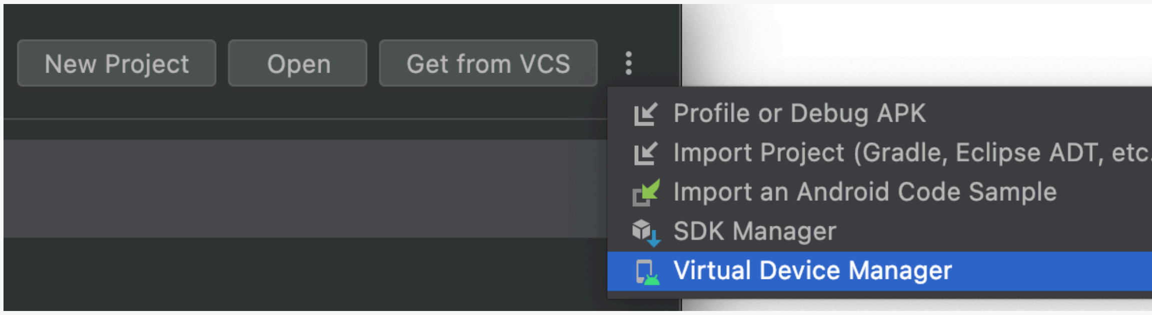
2. Prepare the Android device

   A. Using a physical device

   If you have a physical Android device, you can use it for development in place of an AVD by plugging it into your computer using a USB cable and following the instructions here.

   B. Using a virtual device

   - To create an Android Virtual Device, from the Android Studio Welcome screen, select **More Actions > Virtual Device Manager.**



- Select "Create Virtual Device...", then pick any Phone from the list and click "Next", then select the UpsideDownCake API Level 34 image.
- Click "Next" then "Finish" to create your AVD. At this point you should be able to click on the green triangle button next to your AVD to launch it, then proceed to the next step.

3. Running the application

Step 1: Start Metro

Metro is the JavaScript build tool for React Native. To start the Metro development server, run the following from your project folder:

```
$ npm start
```

Step 2: Start your application

Let Metro Bundler run in its own terminal. Open a new terminal inside your React Native project folder. Run the following:

```
$ npm run android
```

If everything is set up correctly, you should see your new app running in your Android emulator shortly.