

## 5. Creating an Android Virtual Device (AVD) in Android Studio

In the course of developing Android apps in Android Studio it will be necessary to compile and run an application multiple times. An Android application may be tested by installing and running it either on a physical device or in an *Android Virtual Device (AVD)* emulator environment. Before an AVD can be used, it must first be created and configured to match the specifications of a particular device model. The goal of this chapter, therefore, is to work through the steps involved in creating such a virtual device using the Nexus 5X phone as a reference example.

### 5.1 About Android Virtual Devices

AVDs are essentially emulators that allow Android applications to be tested without the necessity to install the application on a physical Android based device. An AVD may be configured to emulate a variety of hardware features including options such as screen size, memory capacity and the presence or otherwise of features such as a camera, GPS navigation support or an accelerometer. As part of the standard Android Studio installation, a number of emulator templates are installed allowing AVDs to be configured for a range of different devices. Additional templates may be loaded or custom configurations created to match any physical Android device by specifying properties such as processor type, memory capacity and the size and pixel density of the screen. Check the online developer documentation for your device to find out if emulator definitions are available for download and installation into the AVD environment.

When launched, an AVD will appear as a window containing an emulated Android device environment. [Figure 5-1](#), for example, shows an AVD session configured to emulate the Google Nexus 5X model.

New AVDs are created and managed using the Android Virtual Device Manager, which may be used either in command-line mode or with a more user-friendly graphical user interface.



Figure 5-1

## 5.2 Creating a New AVD

In order to test the behavior of an application in the absence of a physical device, it will be necessary to create an AVD for a specific Android device configuration.

To create a new AVD, the first step is to launch the AVD Manager. This can be achieved from within the Android Studio environment by selecting the *Tools -> Android -> AVD Manager* menu option from within the main window.

Once launched, the tool will appear as outlined in [Figure 5-2](#) if existing AVD instances have been created:

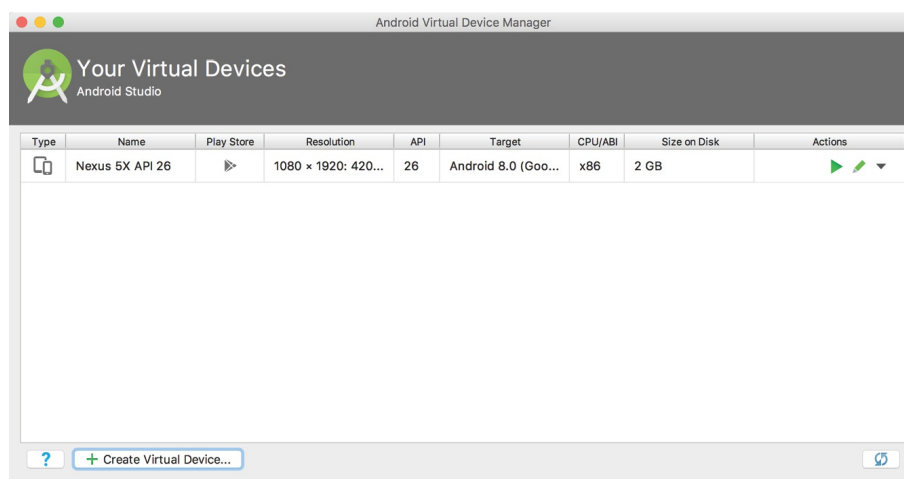


Figure 5-2

To add an additional AVD, begin by clicking on the *Create Virtual Device*

button in order to invoke the *Virtual Device Configuration* dialog:

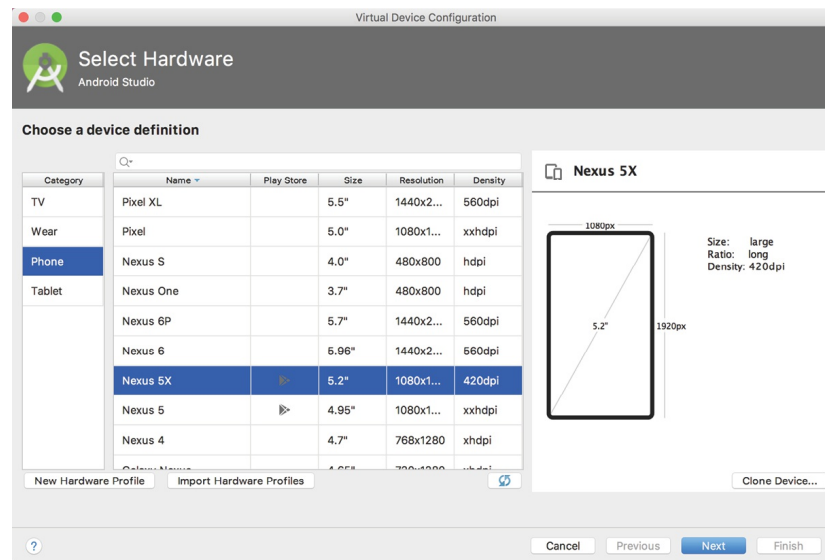


Figure 5-3

Within the dialog, perform the following steps to create a Nexus 5X compatible emulator:

1. From the *Category* panel, select the *Phone* option to display the list of available Android tablet AVD templates.
2. Select the *Nexus 5X* device option and click *Next*.
3. On the System Image screen, select the latest version of Android (at time of writing this is Oreo, API level 26, Android 8.0 with Google Play) for the *x86* ABI. Note that if the system image has not yet been installed a *Download* link will be provided next to the Release Name. Click this link to download and install the system image before selecting it. If the image you need is not listed, click on the *x86 images* and *Other images* tabs to view alternative lists.
4. Click *Next* to proceed and enter a descriptive name (for example *Nexus 5X API 26*) into the name field or simply accept the default name.
5. Click *Finish* to create the AVD.
6. With the AVD created, the AVD Manager may now be closed. If future modifications to the AVD are necessary, simply re-open the AVD Manager, select the AVD from the list and click on the pencil icon in the *Actions* column of the device row in the AVD Manager.

## 5.3 Starting the Emulator

To perform a test run of the newly created AVD emulator, simply select the emulator from the AVD Manager and click on the launch button (the green triangle in the Actions column). The emulator will appear in a new window and begin the startup process. The amount of time it takes for the emulator to start will depend on the configuration of both the AVD and the system on which it is running. In the event that the startup time on your system is considerable, do not hesitate to leave the emulator running. The system will detect that it is already running and attach to it when applications are launched, thereby saving considerable amounts of startup time.

The emulator probably defaulted to appearing in portrait orientation. It is useful to be aware that this and other default options can be changed. Within the AVD Manager, select the new Nexus 5X entry and click on the pencil icon in the *Actions* column of the device row. In the configuration screen locate the *Startup and orientation* section and change the orientation setting. Exit and restart the emulator session to see this change take effect. More details on the emulator are covered in the next chapter ([“Using and Configuring the Android Studio AVD Emulator”](#)).

To save time in the next section of this chapter, leave the emulator running before proceeding.

## 5.4 Running the Application in the AVD

With an AVD emulator configured, the example AndroidSample application created in the earlier chapter now can be compiled and run. With the AndroidSample project loaded into Android Studio, simply click on the run button represented by a green triangle located in the Android Studio toolbar as shown in [Figure 5-4](#) below, select the *Run -> Run ‘app’* menu option or use the Ctrl-R keyboard shortcut:

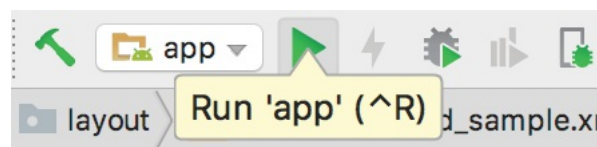


Figure 5-4

By default, Android Studio will respond to the run request by displaying the *Select Deployment Target* dialog. This provides the option to execute the application on an AVD instance that is already running, or to launch a new AVD session specifically for this application. [Figure 5-5](#) lists the previously

created Nexus 5X AVD as a running device as a result of the steps performed in the preceding section. With this device selected in the dialog, click on *OK* to install and run the application on the emulator.

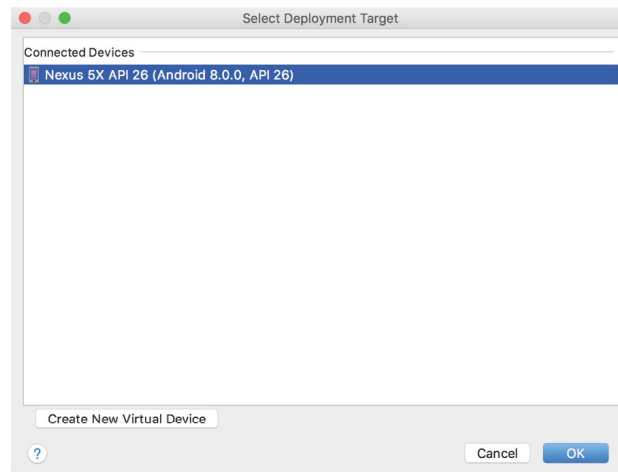


Figure 5-5

Once the application is installed and running, the user interface for the `AndroidSampleActivity` class will appear within the emulator:

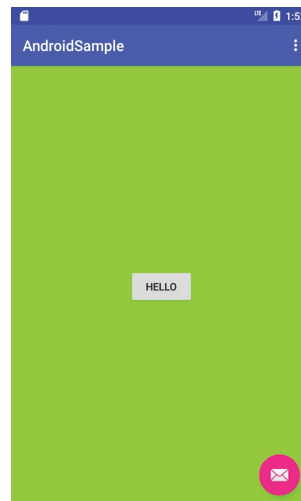


Figure 5-6

In the event that the activity does not automatically launch, check to see if the launch icon has appeared among the apps on the emulator. If it has, simply click on it to launch the application. Once the run process begins, the Run and Logcat tool windows will become available. The Run tool window will display diagnostic information as the application package is installed and launched. [Figure 5-7](#) shows the Run tool window output from a successful application launch:



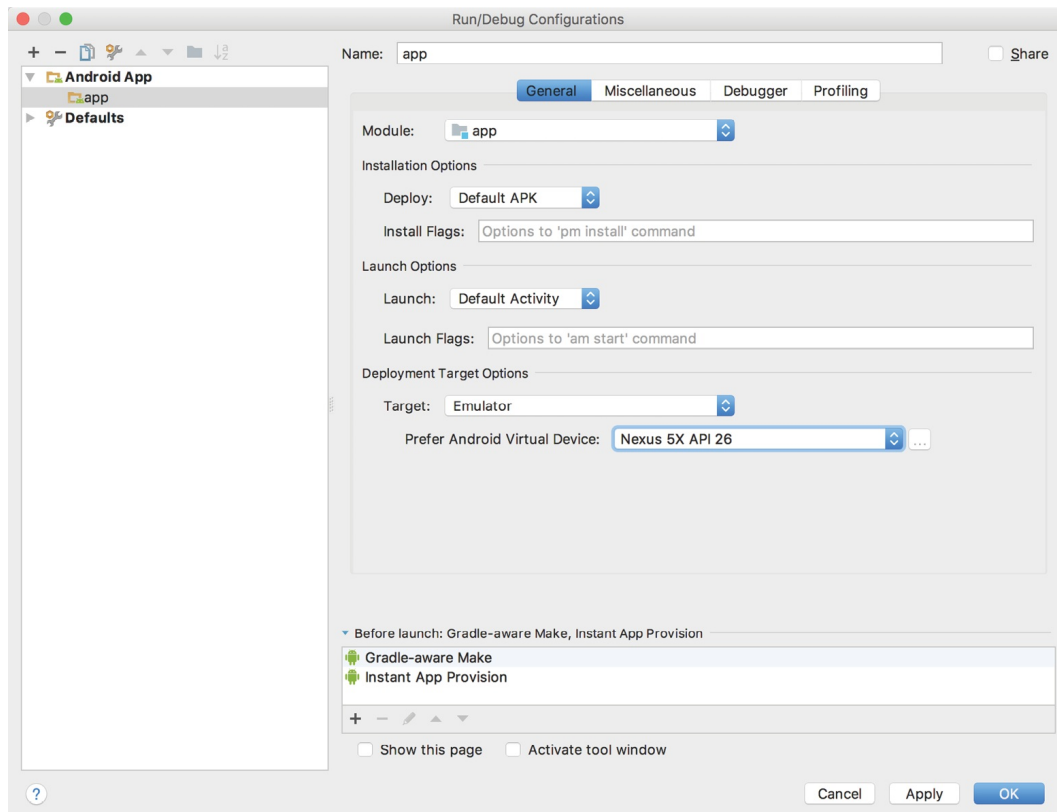


Figure 5-9

Be sure to switch the Target menu setting back to “Open Select Deployment Target Dialog” mode before moving on to the next chapter of the book.

## 5.6 Stopping a Running Application

To stop a running application, simply click on the stop button located in the main toolbar as shown in [Figure 5-10](#):

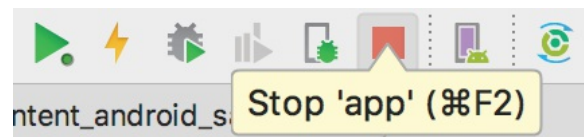


Figure 5-10

An app may also be terminated using the Logcat tool window. Begin by displaying the *Logcat* tool window either using the window bar button, or via the quick access menu (invoked by moving the mouse pointer over the button in the left-hand corner of the status bar as shown in [Figure 5-11](#)).



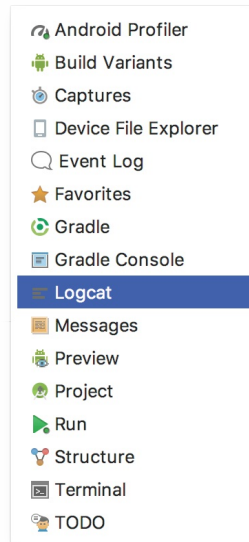


Figure 5-11

Once the Logcat tool window appears, select the *androidsample* app menu highlighted in [Figure 5-12](#) below:

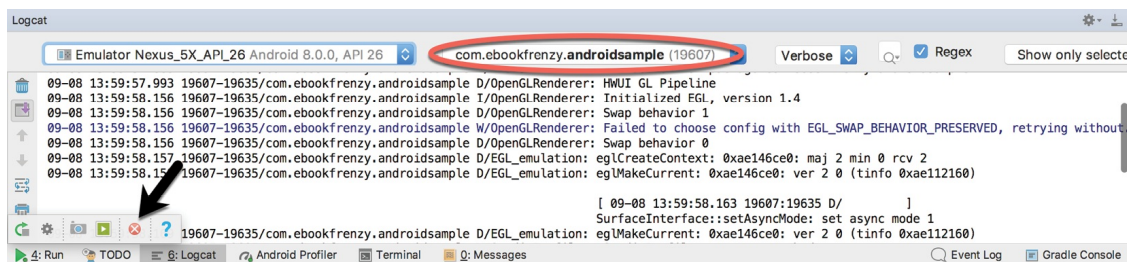


Figure 5-12

With the process selected, stop it by clicking on the red *Terminate Application* button in the toolbar to the left of the process list indicated by the arrow in the above figure.

An alternative to using the Android tool window is to open the Android Device Monitor. This can be launched via the *Tools -> Android -> Android Device Monitor* menu option. Once launched, the process may be selected from the list ([Figure 5-13](#)) and terminated by clicking on the red *Stop* button located in the toolbar above the list.



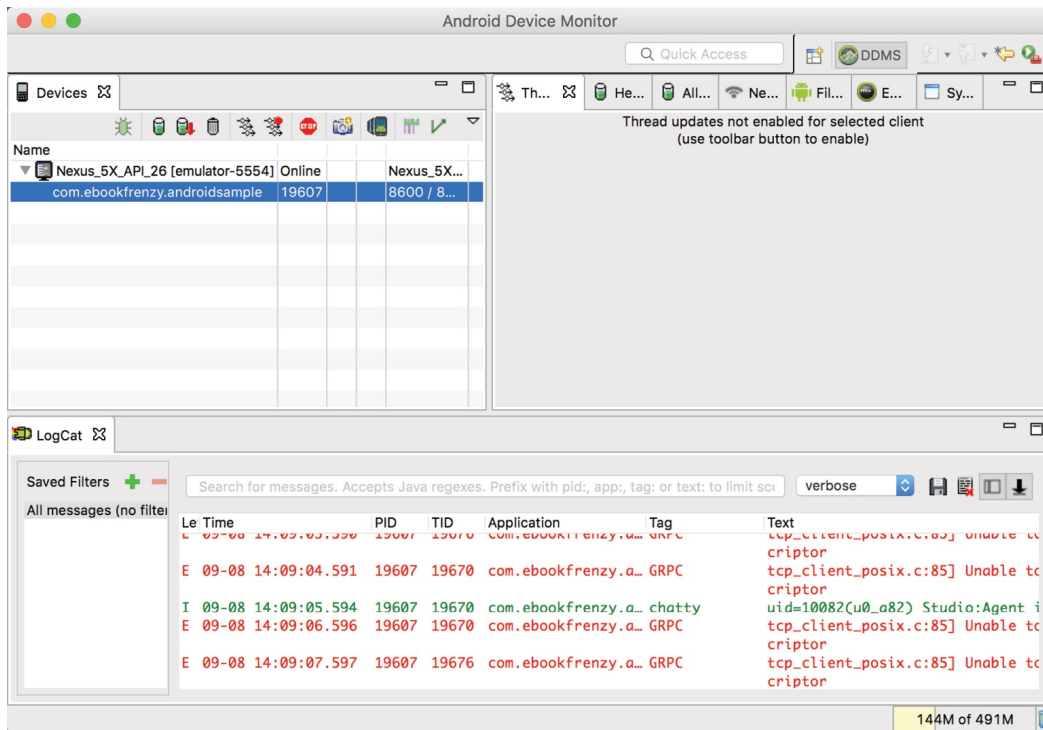


Figure 5-13

## 5.7 AVD Command-line Creation

As previously discussed, in addition to the graphical user interface it is also possible to create a new AVD directly from the command-line. This is achieved using the *avdmanager* tool in conjunction with some command-line options. Once initiated, the tool will prompt for additional information before creating the new AVD.

Assuming that the system has been configured such that the Android SDK *tools* directory is included in the PATH environment variable, a list of available targets for the new AVD may be obtained by issuing the following command in a terminal or command window:

```
avdmanager list targets
```

The resulting output from the above command will contain a list of Android SDK versions that are available on the system. For example:

```
Available Android targets:
```

```
-----
```

```
id: 1 or "android-25"
    Name: Android API 25
    Type: Platform
    API level: 25
```

```
Revision: 3
-----
id: 2 or "android-26"
Name: Android API 26
Type: Platform
API level: 26
Revision: 1
```

The `avdmanager` tool also allows new AVD instances to be created from the command line. For example, to create a new AVD named *Nexus9* using the target ID for the Android API level 26 device using the x86 ABI, the following command may be used:

```
avdmanager create avd -n Nexus9 -k "system-images;android-26;google_apis;x86"
```

The `android` tool will create the new AVD to the specifications required for a basic Android 8 device, also providing the option to create a custom configuration to match the specification of a specific device if required. Once a new AVD has been created from the command line, it may not show up in the Android Device Manager tool until the *Refresh* button is clicked.

In addition to the creation of new AVDs, a number of other tasks may be performed from the command line. For example, a list of currently available AVDs may be obtained using the *list avd* command line arguments:

```
avdmanager list avd
```

Available Android Virtual Devices:

```
Name: Nexus_5X_API_26
Device: Nexus 5X (Google)
Path: /Users/neilsmyth/.android/avd/Nexus_5X_API_26.avd
Target: Google Play (Google Inc.)
Based on: Android 8.0 (Oreo) Tag/ABI: google_apis_playstore,
Skin: nexus_5x
Sdcard: 100M
```

Similarly, to delete an existing AVD, simply use the *delete* option as follows:

```
avdmanager delete avd -n <avd name>
```

## 5.8 Android Virtual Device Configuration Files

By default, the files associated with an AVD are stored in the *.android/avd* sub-directory of the user's home directory, the structure of which is as follows (where *<avd name>* is replaced by the name assigned to the AVD):

```
<avd name>.avd/config.ini
```

```
<avd name>.avd/userdata.img  
<avd name>.ini
```

The *config.ini* file contains the device configuration settings such as display dimensions and memory specified during the AVD creation process. These settings may be changed directly within the configuration file and will be adopted by the AVD when it is next invoked.

The *<avd name>.ini* file contains a reference to the target Android SDK and the path to the AVD files. Note that a change to the *image.sysdir* value in the *config.ini* file will also need to be reflected in the *target* value of this file.

## 5.9 Moving and Renaming an Android Virtual Device

The current name or the location of the AVD files may be altered from the command line using the *avdmanager* tool's *move avd* argument. For example, to rename an AVD named Nexus9 to Nexus9B, the following command may be executed:

```
avdmanager move avd -n Nexus9 -r Nexus9B
```

To physically relocate the files associated with the AVD, the following command syntax should be used:

```
avdmanager move avd -n <avd name> -p <path to new location>
```

For example, to move an AVD from its current file system location to */tmp/Nexus9Test*:

```
avdmanager move avd -n Nexus9 -p /tmp/Nexus9Test
```

Note that the destination directory must not already exist prior to executing the command to move an AVD.

## 5.10 Summary

A typical application development process follows a cycle of coding, compiling and running in a test environment. Android applications may be tested on either a physical Android device or using an Android Virtual Device (AVD) emulator. AVDs are created and managed using the Android AVD Manager tool which may be used either as a command line tool or using a graphical user interface. When creating an AVD to simulate a specific Android device model it is important that the virtual device be configured with a hardware specification that matches that of the physical device.