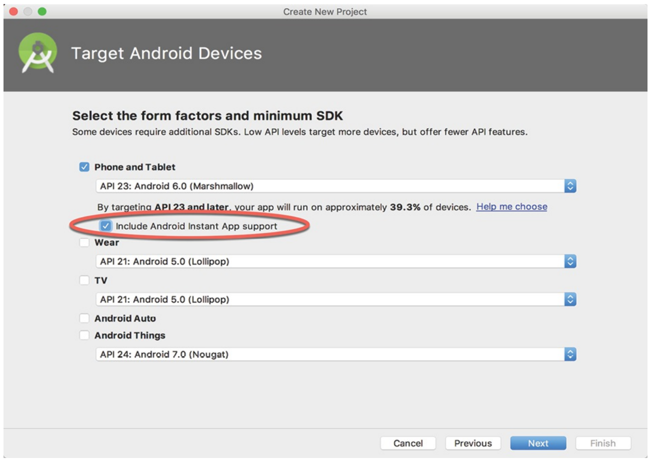
75. An Android Instant App Tutorial

The previous chapter has introduced Android Instant Apps and provided an overview of how these are structured and implemented. Instant Apps can be created as part of a new Android Studio project, or added retroactively to an existing project. This chapter will focus on including instant app support in a new project. The chapters that follow will outline how to add instant app support to an existing project.

75.1 Creating the Instant App Project

Launch Android Studio, select the option to create a new project and name the project *InstantAppDemo* before clicking on the *Next* button. On the subsequent screen, select the *Phone and Tablet* option and change the SDK setting to *API 23: Android 6.0 (Marshmallow)*. Before clicking the Next button, enable the *Include Android Instant App support* option as highlighted in Figure 75-1 below:



Create New Project

Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

☑ **Phone and Tablet**

API 23: Android 6.0 (Marshmallow)

By targeting **API 23 and later**, your app will run on approximately **39.3%** of devices. Help me choose

☑ Include Android Instant App support

□ **Wear**

API 21: Android 5.0 (Lollipop)

□ **TV**

API 21: Android 5.0 (Lollipop)

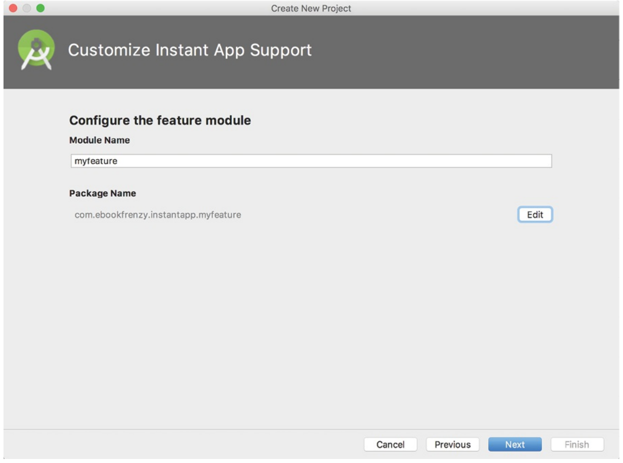
□ **Android Auto**

□ **Android Things**

API 24: Android 7.0 (Nougat)

Figure 75-1

Click *Next* and, on the instant app customization screen, name the feature *myfeature*:



Configure the feature module

Module Name: myfeature

Package Name: com.ebookfrenzy.instantapp.myfeature

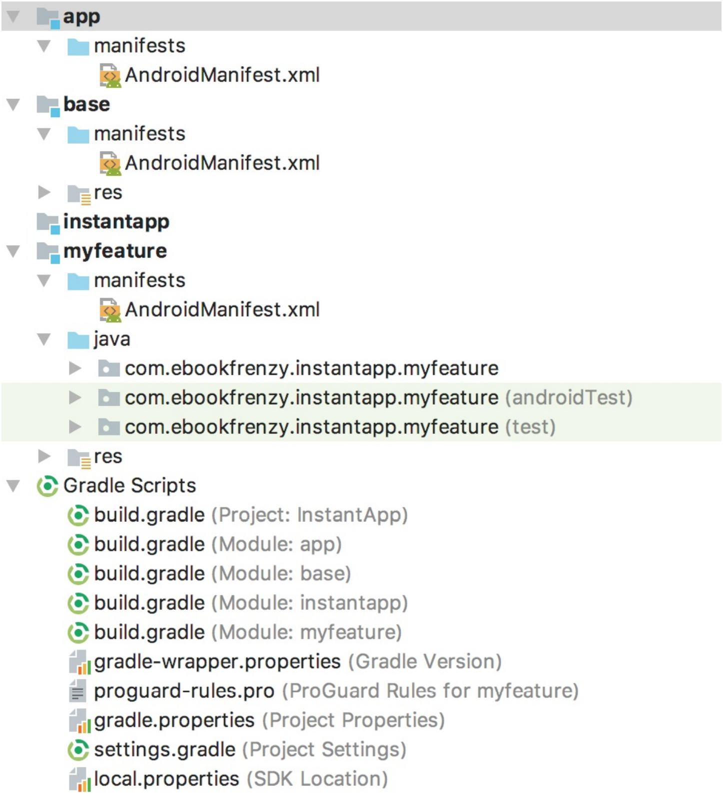
Figure 75-2

On the next screen, select the *Empty Activity* template before proceeding to the final screen. Since the Instant App option was selected, the activity configuration screen will provide fields within which to specify an app link for this activity. Specify *example.com* as the *Instant App URL Host*, select the *Path* option from the *Instant App URL Route Type* and enter */home* as the route URL. Name the activity *InstantAppActivity* and the layout *activity_instant_app* before clicking on the *Finish* button to create the new project.

75.2 Reviewing the Project

Based on the selections made, Android Studio has actually completed all of the work necessary to support both installed and instant app builds of the project. All that would be required to complete the app is to implement the functionality in the main activity and to add other feature modules if needed (the latter topic will be covered in the chapter entitled *"Creating Multi-Feature Instant Apps"*).

Before testing the app, it is worthwhile taking some time to review the way in which the project has been structured. At this point, the project structure within the Project Tool window should match that shown in Figure 75-3:



```
▼ app
    ▼ manifests
        AndroidManifest.xml
▼ base
    ▼ manifests
        AndroidManifest.xml
    ▶ res
  instantapp
▼ myfeature
    ▼ manifests
        AndroidManifest.xml
    ▼ java
        ▶ com.ebookfrenzy.instantapp.myfeature
        ▶ com.ebookfrenzy.instantapp.myfeature (androidTest)
        ▶ com.ebookfrenzy.instantapp.myfeature (test)
    ▶ res
▼ Gradle Scripts
    build.gradle (Project: InstantApp)
    build.gradle (Module: app)
    build.gradle (Module: base)
    build.gradle (Module: instantapp)
    build.gradle (Module: myfeature)
    gradle-wrapper.properties (Gradle Version)
    proguard-rules.pro (ProGuard Rules for myfeature)
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

Figure 75-3

The project now consists of an installed app module (*app*), and instant app module (*instantapp*), a base feature module (*base*) and an additional feature module (*myfeature*). Each of these modules has associated with it a *build.gradle* file that defines how the module is to be built and the other modules on which it is dependent. The *build.gradle (Module: app)* file, for example, uses the *com.android.application* plugin to build the installed app version of the project and declares both the *base* and *myfeature* modules as dependencies:

```
apply plugin: 'com.android.application'
.
.
.
dependencies {
    implementation project(':myfeature')
    implementation project(':base')
}
```

The Gradle build file for the *instantapp* module also declares the *base* and *myfeature* modules as dependencies, but this time the *com.android.instantapp* plugin is used to build the instant app version of the project:

```
apply plugin: 'com.android.instantapp'

dependencies {
    implementation project(':myfeature')
    implementation project(':base')
}
```

A review of the build file for the *base* module will reveal the use of the *com.android.feature* plugin, a declaration that this is the base class and the *app* and *myfeature* dependencies:

```
apply plugin: 'com.android.feature'

android {
    compileSdkVersion 26
    baseFeature true
.
.
.
}

dependencies {
    application project(':app')
    feature project(':myfeature')
    api 'com.android.support:appcompat-v7:26.0.0'
    api 'com.android.support.constraint:constraint-layout:1.0.2'
}
```

The *myfeature* module contains both the layout and class file for the main activity. Although not necessary for the purposes of this tutorial, any change to the activity would be made within these module files.

In addition to the build files and module structure, Android Studio has also placed the appropriate intent filter for the app link to the *AndroidManifest.xml* file belonging to the *instantapp* module.

75.3 Testing the Installable App

Test the installable app by selecting the *app* entry in the toolbar run configuration selection menu as shown in Figure 75-4 and then clicking on the run button:

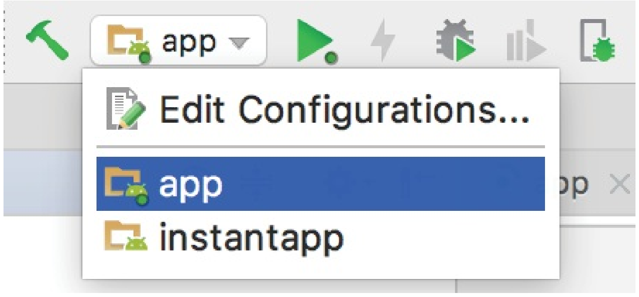


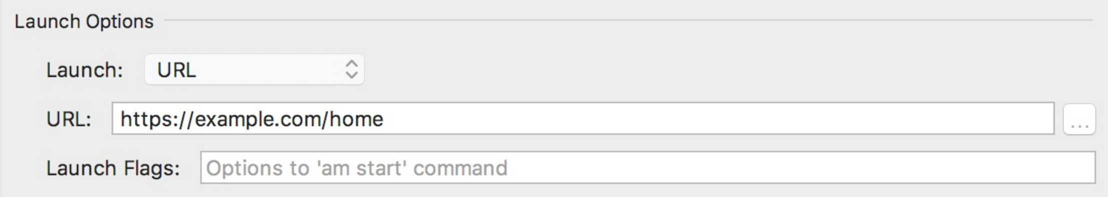
Figure 75-4

Select a suitable deployment target and verify that the APK installs and the app launches.

75.4 Testing the Instant App

Before the instant app can be tested, the installed app must first be removed from the device or emulator being used for testing. Launch the Settings app and navigate to the *Apps & notifications* screen. Click on the *App info* link, locate and select the *InstantAppDemo* app then click on the *Uninstall* button.

Once the installed app has been removed, return to Android Studio and select the *instantapp* module in the run configuration menu. Before running the app, open the menu once again and select the *Edit configurations…* option. In the *Run/Debug Configurations* dialog, note that Android Studio has automatically configured the module to launch using the previously declared app link URL:



Launch Options

Launch: URL

URL: https://example.com/home

Launch Flags: Options to 'am start' command

Figure 75-5

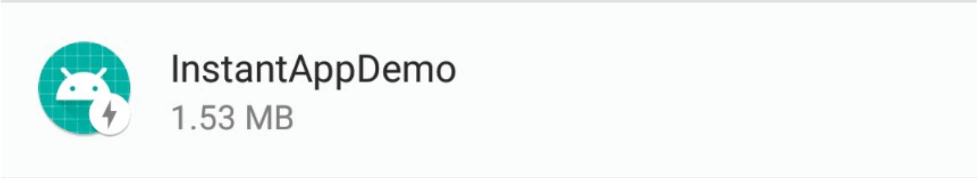
Close the configuration dialog and click on the run button to launch the instant app. As the app is launching the following output will appear in the Run Tool window confirming the instant app is being launched:

```
07/19 10:38:27: Launching instantapp
Side loading instant app.
Launching deeplink: https://example.com/home.

$ adb shell setprop log.tag.AppIndexApi VERBOSE
$ adb shell am start -a android.intent.action.VIEW -
c android.intent.category.BROWSABLE -d https://example.com/home
```

Aside from the different output, the instant app feature should launch just as it did for the installable app.

A review of the installed app on the device within the Settings app will now display the app icon with a lightning bolt to indicate that this is an instant app:

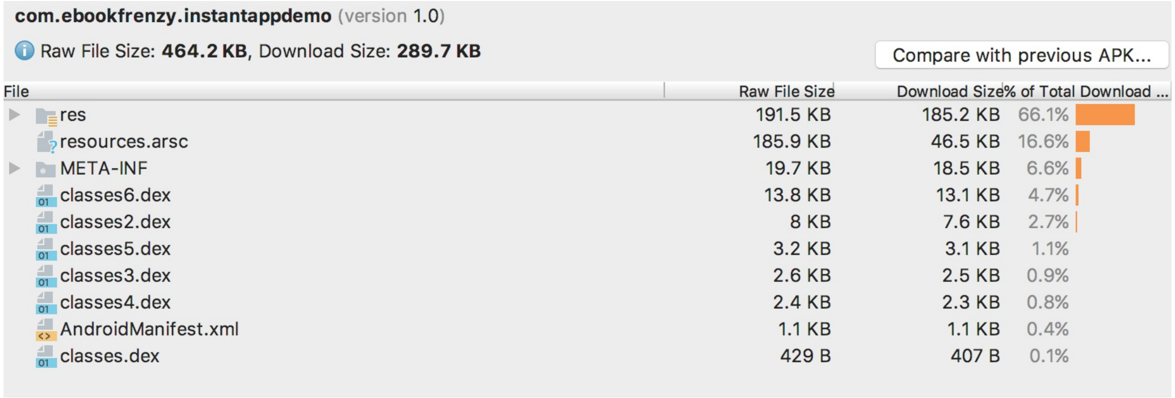


InstantAppDemo
1.53 MB

Figure 75-6

75.5 Reviewing the Instant App APK Files

The previous chapter explained that the installable app APK file contains the basic components that make up an app. To see this in practical terms, select the Android Studio *Build -> Analyze APK…* menu option and navigate to, select and open the *InstantApp -> app -> build -> outputs -> apk -> debug -> app-debug.apk* file. Once selected, the APK Analyzer panel will open and display the content of the APK file. As shown in Figure 75-7 below, this file contains the class dex files and the associated resources for the entire app:



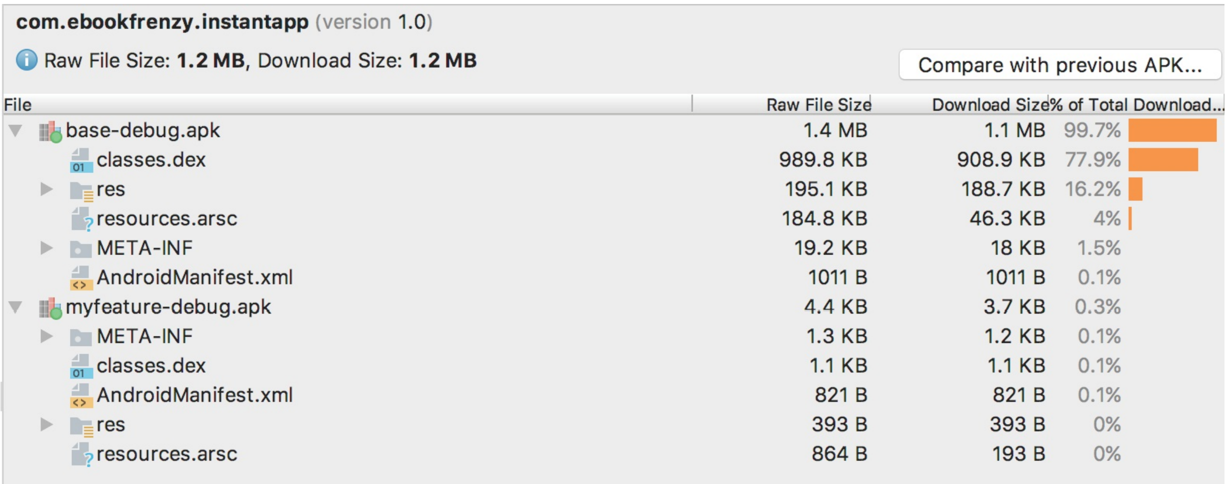
com.ebookfrenzy.instantappdemo (version 1.0)

Raw File Size: **464.2 KB**, Download Size: **289.7 KB**

File	Raw File Size	Download Size	% of Total Download ...
res	191.5 KB	185.2 KB	66.1%
resources.arsc	185.9 KB	46.5 KB	16.6%
META-INF	19.7 KB	18.5 KB	6.6%
classes6.dex	13.8 KB	13.1 KB	4.7%
classes2.dex	8 KB	7.6 KB	2.7%
classes5.dex	3.2 KB	3.1 KB	1.1%
classes3.dex	2.6 KB	2.5 KB	0.9%
classes4.dex	2.4 KB	2.3 KB	0.8%
AndroidManifest.xml	1.1 KB	1.1 KB	0.4%
classes.dex	429 B	407 B	0.1%

Figure 75-7

Repeat this step, this time navigating to the *InstantApp -> instantapp -> build -> outputs -> apk -> debug -> instantapp-debug.zip* file. Note that this file contains two APK files, one for the base module and the other for the myfeature module, each containing its own dex and resource files:



com.ebookfrenzy.instantapp (version 1.0)

Raw File Size: **1.2 MB**, Download Size: **1.2 MB**

File	Raw File Size	Download Size	% of Total Download...
▼ base-debug.apk	1.4 MB	1.1 MB	99.7%
classes.dex	989.8 KB	908.9 KB	77.9%
▶ res	195.1 KB	188.7 KB	16.2%
resources.arsc	184.8 KB	46.3 KB	4%
▶ META-INF	19.2 KB	18 KB	1.5%
AndroidManifest.xml	1011 B	1011 B	0.1%
▼ myfeature-debug.apk	4.4 KB	3.7 KB	0.3%
▶ META-INF	1.3 KB	1.2 KB	0.1%
classes.dex	1.1 KB	1.1 KB	0.1%
AndroidManifest.xml	821 B	821 B	0.1%
▶ res	393 B	393 B	0%
resources.arsc	864 B	193 B	0%

Figure 75-8

75.6 Summary

This chapter has outlined the steps to creating a new Android Studio project including Android Instant App support. As revealed in this chapter, much of the work involved in structuring the project to build both installable and instant apps is performed automatically by Android Studio. Having created the example app, this chapter also outlined how to test instant apps and demonstrated the use of the APK Analyzer to review the difference between the APK files for installable and instant app projects.