Android 6 Essentials

Android 6 is the latest Android version with some cutting-edge features you can harness to benefit your application.

This step-by-step guide will take you through the basics of the Android 6 permissions model and beyond into other crucial areas, such as the audio/video/camera API or Android for Work features.

With this book, you will learn how to create, deploy, and manage Android applications with with the Android 6 API and behavior changes.

The combination of instructions and real-world examples throughout this book will make your application deployment and testing easy.

Who this book is written for

This book is for Android developers looking to move their applications to the next Android version with ease. The reader should have a good understanding of Java and previous Android version APIs, and should be able to write applications with APIs older than Android 6.

What you will learn from this book

- Familiarize yourself with the features of Android 6
- Code with the new Android permissions model
- Use the auto back-up feature of apps
- Increase user engagement with apps with the use of an assistant
- Assess and handle proper API usage
- Work with the audio/video/camera API in Android 6
- Use the new features of Android for Work
- Understand and use Chrome custom tabs

Yossi Elkrief
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 5 *Audio, Video, and Camera Features*
- A synopsis of the book’s content
- More information on Android 6 Essentials
About the Author

Yossi Elkrief is an Android enthusiast with over 7 years of experience in the Android platform and is currently working as an Android architect and group leader at Tikal Knowledge.

Among his previous experiences, the noteworthy ones include ooVoo, Fiverr, Mobli, and Glide, reaching out to over 135 million users worldwide.

Yossi is a mentor at Google Launchpad, a lecturer on IoT and mobile development, and co-tech lead on the Madgera accelerator. He cofounded the GDG Be'er Sheva group and co-leads the group today, holding technology events for the technology community in Israel. He has a spouse and a daughter, and he lives in Be’er Sheva, Israel.

His main interests are Liverpool Football Club and his Android mini collectibles collection, with over 120 different pieces. You can find him on LinkedIn at https://il.linkedin.com/in/yossielkrief, on GitHub at MaTriXY, and on Google+ at Yossi.Elkrief.
Preface

Android 6 will primarily focus on improving the overall user experience, and it will bring in a few features, such as a redesigned permission model in which applications are no longer automatically granted all of their specified permissions at the time of installation, the Doze power scheme for extended battery life when a device is not manipulated by the user, and native support for fingerprint recognition.

If you're already an Android developer, you're only a few steps away from being able to use your existing development experience to reach your users wherever or whenever they want or need your app.

As a professional Android developer, you have to create production-ready apps for your users. This book will give you what it takes to ship polished apps as part of a development team at a company, an independent app developer, or just as a programmer using Android development best practices.

By the end of the book, you'll be able to identify critical areas for improvement in an app and implement the necessary changes and refinements to ensure it meets Android's Core App Guidelines prior to shipping.

What this book covers

Chapter 1, Android Marshmallow Permissions, discusses how the Android permission system and model are vast and have made a few changes that can help app developers and applications gain more traction, installations, and give users the ability to decide when your applications will be able to use each permission-dependent feature. Keep in mind, though, that this is just a starting point and Android Marshmallow still needs to gain market share and get adopted by OEMs, enabling users with the freedom of choice. You as an app developer must prepare in advance and make sure your application development is forward-facing, allowing new users to enjoy the latest updates as soon as possible while maintaining a high level of performance for your applications.
Chapter 2, App Links, talks about how app linking has become powerful in Android Marshmallow. This allows you, the app developers, help the system better decide how to act. Handling web URLs will give you wider exposure, a bigger funnel into your apps, and better experience, which you can provide to your users (sums up to better ratings and more downloads and vice versa).

App linking is simple to implement, easy to understand, and is a must-have feature in the mobile/web world today. While app linking enables better action handling for users using your applications, users can have multiple devices, expecting the same behavior on each device, and would be more engaged if their data and action handling is all around.

Chapter 3, Apps' Auto Backup, informs you that Android Marshmallow brings with it a great backup feature for apps, reducing friction for users migrating to new devices.

In a world full of such diverse apps, maximizing the benefits from automatic backups leads to excellent user experience. The goal of this feature is to unload the burden and shorten the time required to set up a new device with the user's favorite apps. Allowing the users to enter your app with just a password prompt, if required, after a new installation can be a great experience. Try it yourself!

Chapter 4, Changes Unfold, goes over a few of the changes in Android Marshmallow. All of these changes are important to follow and will help you in your app development cycles. A few more changes are discussed in future chapters with a more detailed approach.

Chapter 5, Audio, Video, and Camera Features, covers quite a few changes and additions to Android APIs. Android Marshmallow is more about helping us, the developers, achieve better media support and showcase our ideas when using audio, video, or camera APIs.

Chapter 6, Android for Work, covers how Android Marshmallow has brought in quite a few changes to the world of Android for Work. As developers, we need to always maintain a viable connection with the needs of an organization. Making sure that we go over and understand the Android for Work world with the changes in Marshmallow helps us build and target enterprise workflows with the added benefit of a simpler API.

Chapter 7, Chrome Custom Tabs, talks about the newly added feature, Chrome custom tabs, that allows us developers to embed web content into our application, modify the UI, and adjust it to our app's theme and colors and the look and feel. This helps us keep the user in our application and still provide a nice UI and overall feel.
Chapter 8, Authentication, discuss how Android Marshmallow gives us a new API to authenticate users with the fingerprint API. We can use the sensor and authenticate the user even within our application and save it for later usage if we want to save the need of user login using the Credentials grace period abilities Android Marshmallow has introduced. We also covered a way to make our application more secure using HTTPS only. The StrictMode policy, enforced with the help of the usesCleartextTraffic flag, allows us to make sure that all the nodes we connect to the outer world are examined to check if there's a need for a secure connection or not.
Audio, Video, and Camera Features

Android Marshmallow gives us good audio, video, and camera capabilities, and you can see that improvements have been made to enable and better support new or mint condition protocols or even change the behavior of some APIs, such as the camera service.

In this chapter, we will try and explain these changes with a proper discussion on their usage and benefits. Our journey in the upcoming pages will cover the following topics:

- Audio features
- Video features
- Camera features

Audio features

Android Marshmallow 6.0 adds some enrichments to the audio features that we will cover in the upcoming sections.

Support for the MIDI protocol

The android.media.midi package was added in Android 6.0 (API 23).

With the new midi APIs, you can now send and receive MIDI (short for Musical Instrument Digital Interface) events in a much simpler way than earlier.
Audio, Video, and Camera Features

The package was built to provide us with capabilities to do the following:

- Connect and use a MIDI keyboard
- Connect to other MIDI controllers
- Use external MIDI synthesizers, external peripherals, lights, show control, and so on
- Allow dynamic music generation from games or music-creation apps
- Allow the creation and passing of MIDI messages between apps
- Allow Android devices to act as multi-touch controllers when connected to a laptop

When dealing with MIDI, you must declare it in the manifest, as follows:

```xml
<uses-feature android:name="android.software.midi"
android:required="true"/>
```

Pay attention to the `required` part; in a manner similar to other features, setting it to `true` will make your app visible in the play store only if the device supports the MIDI API.

You can also check in runtime for MIDI support and then change the required part to `false`:

```java
PackageManager pkgMgr = context.getPackageManager();
if (pkgMgr.hasSystemFeature(PackageManager.FEATURE_MIDI)) {
    // we can use MIDI API here as we know the device supports the MIDI API.
}
```

**MidiManager**

A way to properly use the MIDI API is via the `MidiManager` class; obtain it via `context` and use it when required:

```java
MidiManager midiMgr = (MidiManager)context.getSystemService(Context.MIDI_SERVICE);
```

For more information, you can refer to:

Digital audio capture and playback
Two new classes have been added for digital audio capture and playback:

- `android.media.AudioRecord.Builder` - digital audio capture
- `android.media.AudioTrack.Builder` - digital audio playback

These will help configure the audio source and sink properties.

Audio and input devices
The new `hasMicrophone()` method has been added to the `InputDevice` class. This will report whether the device has a built-in microphone that developers can use.

Let's say you want to enable voice search from a controller connected to Android TV and you get an `onSearchRequested()` callback for the user's search. You can then verify that there's a microphone with the `inputDevice` object you get in the callback.

Information on audio devices
The new `AudioManager.getDevices(int flags)` method allows easy retrieval of all the audio devices currently connected to the system. If you want to be notified when there are audio device connections/disconnections, you can register your app to an `AudioDeviceCallback` callback via the `AudioManager.registerAudioDeviceCallback(AudioDeviceCallback callback, Handler handler)` method.

Changes in AudioManager
Some changes have been introduced in the `AudioManager` class, and they are as follows:

- Using `AudioManager` to set the volume directly is not supported.
- Using `AudioManager` to mute specific streams is not supported.
- The `AudioManager.setStreamSolo(int streamType, boolean state)` method is deprecated. If you need exclusive audio playback, use `AudioManager.requestAudioFocus(AudioManager.OnAudioFocusChangeListener l, int streamType, int durationHint)`.
- The `AudioManager.setStreamMute(int streamType, boolean state)` method is deprecated. If you need to use `AudioManager.adjustStreamVolume(int streamType, int direction, int flags)` for direction, you can use one of the newly added constants.
Audio, Video, and Camera Features

- **ADJUST_MUTE** will mute the volume. Note that it has no effect if the stream is already muted.
- **ADJUST_UNMUTE** will unmute the volume. Note that it has no effect if the stream is not muted.

Video features
In Android Marshmallow, the video processing API has been upgraded with new capabilities. Some new methods and even a new class has been added just for developers.

**android.media.MediaSync**
The all new MediaSync class has been designed to help us with synchronous audio and video streams’ rendering. You can also use it to play audio- or video-only streams. You can use the dynamic playback rate and feed the buffers in a nonblocking action with a callback return. For more information on the proper usage, read:

https://developer.android.com/reference/android/media/MediaSync.html

**MediaCodecInfo.CodecCapabilities.getMaxSupportedInstances**
Now, we have a MediaCodecInfo.CodecCapabilities getMaxSupportedInstances helper method to get the maximum number of supported concurrent codec instances. However, we must consider this only an upper bound. The actual number of concurrent instances can be lower depending on the device and the amount of available resources at the time of usage.

Why do we need to know this?
Let's think of a case where we have a media-playing application and we want to add effects between the movies played. We will need to use more than one video codec, decode two videos, and encode one video stream back to be displayed on screen. Checking with this API will allow you to add more features that rely upon multiple instances of codecs.
MediaPlayer.setPlaybackParams

We can now set the media playback rate for fast or slow motion playback. This will give us the chance to create a funny video app where we slow down parts or play them fast, creating a new video while playing. Audio playing is synced accordingly, so you might hear a person talking slowly or even fast, for that matter.

Camera features

In Android Lollipop, there was the new Camera2 API, and now, in Android Marshmallow, there are a few more updates to the camera, flashlight, and image reprocessing features.

The flashlight API

Almost every device today has a camera, and almost every camera device has a flash unit. The setTorchMode() method has been added to control the flash torch mode.

The setTorchMode() method is used in the following manner:

CamerasManager.setTorchMode (String cameraId, boolean enabled)

The cameraId element is the unique ID for the flash unit camera with which you want to change the torch mode. You can use getCameraIdList() to get the list of cameras and then use getCameraCharacteristics(String cameraId) to check whether flash is supported in that camera. The setTorchMode() method allows you to turn it on or off without opening the camera device and without requesting permission from the camera. The torch mode will be switched off as soon as the camera device becomes unavailable or when other camera resources that have the torch on become unavailable. Other apps can use the flash unit as well, so you need to check the mode when required or register a callback via the registerTorchCallback() method.

Refer to the sample app, Torchi, to see the entire code at:

https://github.com/MaTriXy/Torchi

Turning on the torch mode may fail if the camera or other camera resources are in use.
The reprocessing API
As mentioned earlier, the Camera2 API was given a few boosts to allow added support for YUV and private opaque format image reprocessing. Before using this API, we need to check whether these capabilities are available. This is why we use the `getCameraCharacteristics(String cameraId)` method and check for the `REPROCESS_MAX_CAPTURESTALL` key.

**android.media.ImageWriter**
This is a new class that's been added to Android 6.0.
It allows us to create an image and feed it into a surface and then back to CameraDevice. Usually, ImageWriter is used along with ImageReader.

**android.media.ImageReader**
This is a new class that's been added to Android 6.0.
It allows us direct access to the image data rendered in a surface. ImageReader, along with ImageWriter, allows our app to create an image feed from the camera to the surface and back to the camera for reprocessing.

Changes in the camera service
Android Marshmallow has made a change to the first come, first serve access model; now, the service access model has favorites processes—ones that are marked as high-priority. This change results in some more logic-related work for us developers. We need to make sure that we take into account a situation where we get bumped up (higher priority) or debunked (lower priority due to a change in our application).

Let's try and explain this in a few simple bullets:

- When you want to access camera resources or open and configure a camera device, your access is verified according to the priority of your application process. An application process with foreground activities (visible user) is normally given a higher priority, which in turn allows a better chance to get the desired access when needed.

- On the other side of the priority scale, you can find low-priority apps that can and will be tossed aside (revoked from access) when a high-priority application attempts to use the camera. For example, when using the Camera API, you will get the `onError()` call when evicted, and when using the Camera2 API, you will get the `onDisconnected()` call when evicted.
• Some devices out in the wild can allow separate applications to open and use separate camera devices simultaneously. The camera service now detects and disallows performance issues that are caused due to multiprocess usage. When the service detects such an issue, it will evict low-priority apps even if only one app is using that camera device.

• In a multiuser environment, when switching users, all active apps using the camera in the previous user profile will be evicted in order to allow proper usage and access to apps for the current user. This means that switching users will stop the camera-using apps from using the camera for sure.

Summary

In this chapter, we covered quite a few changes in and additions to the Android APIs. Android Marshmallow is more about helping us, the developers, achieve better media support and showcase our ideas when using the audio, video, or camera APIs.

In the next chapter, we will go over some of the Android features to understand the features, additions, and changes made.
Where to buy this book

You can buy Android 6 Essentials from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.