第 1 章
Android 概述

【本章内容】

- ❒ Android 简介
- ❒ Android 平台架构
- ❒ Android 基本组件
- ❒ 搭建 Android Studio 开发环境
- ❒ 创建 Hello World 项目
- ❒ Android 项目目录结构

Android 是一种以 Linux 为基础的开放源代码的操作系统，主要使用于便携设备，是由 Google（谷歌）与开放手机联盟（Open Handset Alliance）共同提供的软件平台，为全球移动设备、智能设备带来了革命性的变化。据市场研究机构 IDC 统计，2019 年，Android 操作系统的智能手机市场份额从 2018 年的 85.1%上涨到 87%，市场占有率高居首位。随着 Android 智能设备的普及、5G 网络逐渐成熟并且市场化，Android 应用软件的需求势必会越来越大，这是一个潜力巨大的市场，吸引着广大的软件开发厂商和开发者投身其中。

1.1 Android 简介

Android 一词来源于法国作家利尔·亚当在 1886 年发表的科幻小说《未来的夏娃》，本意是“机器人”。虽然 Android 平台是由 Google 公司推出的，但更准确地说，Android 是开放手机联盟的产品。开放手机联盟是由 30 多家高科技公司和手机公司组成的，包括 Google、HTC（宏达电子）、T-Mobile、高通、摩托罗拉、三星、LG 以及中国移动等。开放手机联盟表示，Android 是本着成为第一个开放、完全免费、专门针对移动设备开发平台这一目标，完全从零开始创建的，因此 Android 是第一个完整、开放、免费的手机平台。

Android 系统具有以下特点。

（1）开放性。Google 通过与运营商、设备制造商、开发商等结成深层次的合作伙伴，通过建立标准化、开放式的移动电话软件平台，形成一个开放式的产业系统。

（2）平等性。在 Android 平台上，系统提供的软件和个人开发的应用程序是平等的，例如可以使用第三方开发的拨打电话程序来替代系统提供的拨打电话程序。

（3）应用程序之间的沟通很方便。在 Android 平台下开发的应用程序，可以很方便地

实现应用程序之间数据的共享，只需要进行简单的声明和操作，应用程序就可以访问或者调用其他应用程序的数据，或者将自己的数据提供给其他应用程序使用。例如，第三方的通讯录应用软件就可以访问手机自身的通讯录。

Note

2005 年，Google 收购了成立仅 22 个月的高科技企业 Android，2007 年正式向外界展示了 Android 操作系统，2008 年 9 月 23 日，Google 发布 Android 1.0，从此就有了今天风靡全球的 Android。

在发布 Android 1.5 的时候，Android 使用甜点名称作为系统版本代号。作为每个版本代号的甜点尺寸越变越大，然后按照 26 个字母数序：纸杯蛋糕（1.5），甜甜圈（1.6），松饼（2.1），冻酸奶（2.2），姜饼（2.3），蜂巢（3.0），冰激凌三明治（4.0），果冻豆（4.1），奇巧巧克力（4.4），棒棒糖（5.0），棉花糖（6.0），牛轧糖（7.0），奥利奥（8.0），派（9.0）。从 Android 10 开始，Google 宣布 Android 系统的重大改变，不仅换了全新的 logo，命名方式也变了，2019 年的 Android Q 的正式名称是 Android 10。在 2019 年 Android 开发峰会中，Google 官方首次提到了 Android 11。在 Android 开放源代码项目（AOSP）中，Google 已经启用了代号 Android R，按照 Android 命名规则，Android R 应该就是下一代 Android：Android 11。

1.2 Android 平台架构

在上节介绍了 Android 平台的发展历史及其特征，本节将对 Android 的内部系统框架进行介绍。Android 平台框架如图 1-1 所示，各组成部分介绍如下。

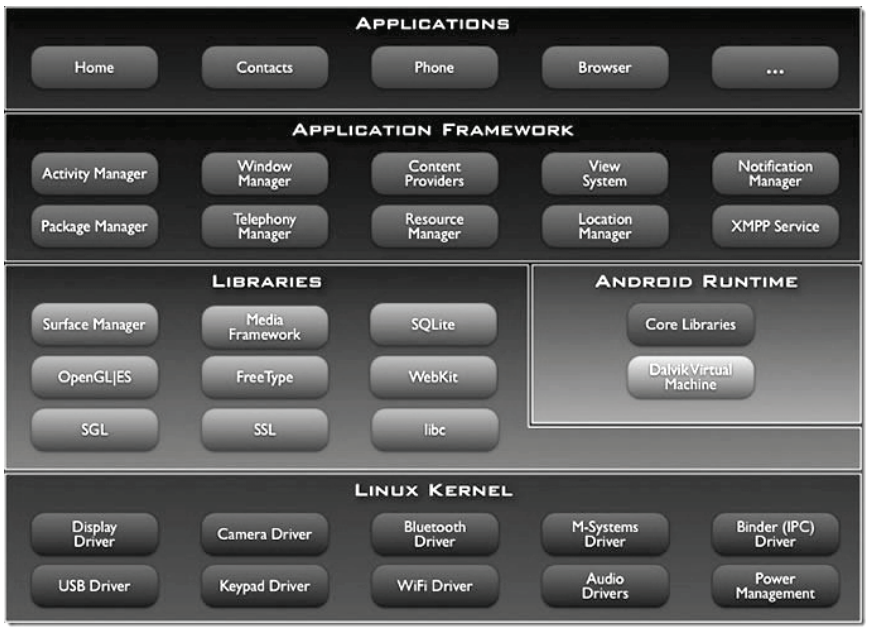

图 1-1 Android 平台应用程序框架图

Note

1．Linux Kernel（Linux 内核）

Android 基于 Linux 提供核心系统服务，例如安全、内存管理、进程管理、网络堆栈、驱动模型。Linux Kernel 作为硬件和软件之间的抽象层，隐藏了具体的硬件细节而为上层提供统一的服务。如果只是进行应用程序开发，则不需要深入了解 Linux Kernel 层。

2．Libraries（库）

Android 包含一个 C/C++库的集合，供 Android 系统的各个组件使用。这些功能通过 Android 的应用程序框架（Application Framework）展现给开发者。下面列出一些核心库。

- Libc：标准 C 系统库的 BSD 衍生，并为基于嵌入式 Linux 设备进行了优化。
- Media Framework：基于 PacketVideo 的 OpenCORE，该库支持播放和录制许多流行的音频和视频格式，以及静态图像文件，包括 MPEG4、H.264、MP3、AAC、AMR、JPG、PNG 等。
- Surface Manager：管理显示子系统、无缝组合多个应用程序的二维和三维图形层。
- WebKit：嵌入式设备的 Web 浏览器引擎，驱动 Android 浏览器和内嵌的 Web 视图。
- SGL：基本的 2D 图形引擎。
- OpenGL ES：专门面向嵌入式系统的 OpenGL API 子集。
- FreeType：位图和矢量字体渲染。
- SQLite：所有应用程序都可以使用的轻量级关系数据库引擎。
- SSL：为网络通信提供安全及数据完整性的一种安全协议。

3．Android Runtime（Android 运行时）

Android 是包含一个核心库的集合，提供大部分在 Java 编程语言核心类库中可用的功能。每一个 Android 应用程序都在它自己的进程中运行，都拥有一个独立的 Dalvik 虚拟机实例。Dalvik 虚拟机依赖于 Linux 内核提供基本功能，来实现进程、内存和文件系统管理等各种服务，可以在一个设备中高效地运行多个虚拟机，可执行文件格式是.dex。.dex 格式是专为 Dalvik 设计的一种压缩格式，占用内存非常小，适合内存和处理器速度有限的系统。

Google 于 2014 年 10 月 15 日发布了 Android 5.0。Android 5.0 系统彻底从 Dalvik 转换到 ART，为开发者和用户带来了有史以来最流畅的系统。ART 的机制与 Dalvik 不同。在 Dalvik 下，应用每次运行的时候，字节码都需要通过即时编译器转换为机器码，这会降低应用的运行效率，而在 ART 环境中，应用在第一次安装的时候，字节码就会预先编译成机器码，使其成为真正的本地应用。这个过程叫作预编译（Ahead-Of-Time，AOT），它使应用的首次启动和执行都变得更加快速。当然，预编译也会带来一些缺点。一方面，机器码占用的存储空间更大。字节码变为机器码之后，可能会增加 10%～20%，不过在应用包中，可执行的代码常常只是一部分，比如最新的 Google+ APK 大小是 28.3MB，但是代码只有 6.9MB。另一方面，应用的安装时间会变长，至于延长的时间，取决于应用本身，一

Note

些复杂的应用（如 Facebook 和 Google+）需要更长的等待时间。

4．Application Framework（应用程序框架）

通过提供开放的开发平台，Android 使开发者能够编制极其丰富和新颖的应用程序，可以自由地利用设备的硬件优势、访问位置信息、运行后台服务、设置闹钟、向状态栏添加通知等。应用程序的体系结构简化了组件之间的重用，任何应用程序服从框架执行的安全限制，都能发布自己的功能。通过应用程序框架，开发人员可以自由地使用核心应用程序所使用的框架 API，实现自己程序的功能，替换系统应用程序。

所有的应用程序其实是一组服务和系统，主要包括如下内容。

- 视图系统（View System）：丰富的、可扩展的视图集合，可用于构建一个应用程序，包括列表、网格、文本框、按钮，甚至是内嵌的网页浏览器。
- 内容提供者（Content Providers）：使应用程序能访问其他应用程序（如通讯录）的数据，或向其他程序共享自己的数据。
- 资源管理器（Resource Manager）：提供访问非代码资源，如本地化字符串、图形和布局文件。
- 通知管理器（Notification Manager）：使所有的应用程序能够在状态栏显示自定义信息。
- 活动管理器（Activity Manager）：管理应用程序生命周期，提供通用的导航回退功能。

5．Application（应用程序）

Android 提供了一系列核心应用程序，包括电子邮件客户端、SMS 程序、拨打电话、日历、地图、浏览器、联系人和其他设置。这些应用程序都是用 Java 语言编写的，而开发人员可以开发出更有创意、功能更强大的应用程序。

1.3 Android 基本组件

Android 的一个主要特点是，一个应用程序可以利用其他应用程序的元素（假设这些应用程序允许）。相反，当需求产生时它只是启动其他应用的程序块。

对于这个工作，当应用程序的任何部分被请求时，系统必须能够启动一个应用程序的进程，并实例化该部分的 Java 对象。因此，不像其他大多数系统的应用程序，Android 应用程序没有一个单一的入口点（如没有 main()函数）。相反，系统能够实例化和运行需要几个必要的组件，主要有四种类型的组件。

- 活动（Activity）。
- 服务（Service）。
- 广播接收者（Broadcast Receiver）。
- 内容提供者（Content Provider）。

然而，并不是所有的应用程序都必须包含上面的四种组件，一种应用程序可以由上面的一种或几种组件来构成。本节将介绍 Android 平台的这四种基本组件。

Note

1．活动

活动是 Android 中最常用的组件，是应用程序的表示层，一般通过 View 来实现用户界面。一个活动表示一个可视化的用户界面，关注一个用户活动的事件。

一个应用程序可能只包含一个活动，也可能包含几个活动。这些活动是什么，以及有多少，取决于应用程序的设计。虽然它们一起工作形成一个整体的用户界面，但是每个活动是独立于其他活动的，每一个都是作为 Activity 类的一个子类。一般来讲，当应用程序被启动时，被标记为第一个的活动应该展示给用户，从一个活动移动到另一个活动由当前的活动完成。

窗口的可视内容由继承自 View 类的一个分层视图对象提供，每个视图控件是窗口内的一个特定的空间。同时，一个视图是活动与用户交互发生的地方。例如，一个视图可能显示一个小的图片或者当用户单击图片时发起一个行为。Android 提供了一些现成的视图可以使用，如按钮（Button）、文本域（TextView、EditText）、复选框（CheckBox）、列表视图（ListView）等。

2．服务

每个服务都继承自 Service 类，没有可视化用户界面，而可以在后台无期限地运行，如一个服务可能是播放背景音乐而用户做其他一些事情（如聊天、上网），或者从网络获取数据，或者计算一些东西并将结果提供给相应的活动。

一个典型的例子就是多媒体播放器播放列表中的歌曲，该播放器应用程序可能有一个或多个活动，允许用户选择歌曲和开始播放。然而，音乐播放本身不会被一个活动处理，因为当用户离开播放器时去做其他事情，希望保持音乐继续播放。为了保持音乐继续播放，媒体播放器活动可以启动一个服务在后台运行，甚至当媒体播放器离开屏幕时，系统将保持音乐播放服务继续运行。与活动和其他组件一样，服务运行在应用程序进程中的主线程中。因此，它们将不会阻止其他组件或用户界面，而是执行其他一些耗时的任务（如音乐播放）。

3．广播接收者

一个广播接收者接收广播公告时可以做出相应的反应。许多广播源自系统代码，如公告时区的改变、电池电量低、已采取图片、用户改变了语言设置。应用程序也可以发起广播，如通知其他程序数据已经下载到设备且可以使用这些数据。

一个应用程序可以有任意数量的广播接收者，用来处理它认为重要的任何公告。所有的接收者继承自 BroadcastReceiver 基类。广播接收者不显示一个用户界面，然而，它们可以启动一个活动去响应收到的信息，使用对话框或者 NotificationManager 通知用户。通知可以使用多种方式获得用户的注意，如闪烁的背光、振动设备、播放声音等，典型的方式

是放置一个持久的图标在状态栏，用户可以打开并获取信息。

4．内容提供者

Note

内容提供者可以将一个应用程序的指定数据集提供给其他应用程序使用，这些数据可以使用文件系统、SQLite 数据库或者其他任何合理的方式进行存储。内容提供者继承自 ContentProvider 类并实现一个标准的方法集合，使得其他应用程序可以检索和操作数据。

内容提供者是 Android 应用程序的主要组成部分之一，它们封装数据且通过 ContentResolver 接口提供给应用程序。只有需要在多个应用程序间共享数据时才使用内容提供者。例如，通讯录数据被多个应用程序使用，且必须存储在一个内容提供者中。如果不需要在多个应用程序间共享数据，可以直接使用 SQLite 数据库或者文件来保存数据。

1.4 搭建 Android 开发环境

Android 应用程序开发主要使用的语言是 Java。在进行 Android 应用程序开发时，除了 IDE 环境之外，还需要安装 Java 的 SDK。Android 开发常用的集成开发环境工具有 Eclipse 与 Android Studio。Android Studio 是 Google 官方推出的基于 IntelliJ IDEA 的 Android 应用程序集成开发环境，Android Studio 不仅是一个强大的代码编辑器和开发者工具，还具有更多可提高 Android 应用构建效率的功能，例如基于 Gradle 的灵活构建系统、快速且功能丰富的模拟器、可针对所有 Android 设备进行开发的统一环境、对各个版本的 SDK 具有良好的支持。鉴于 Android Studio 强大的功能，本书将以 Android Studio 为主，介绍 Android 应用程序开发环境的搭建过程。开发环境搭建步骤如下。

（1）安装 Java SDK。开发者可以在 Oracle 官网或者其他途径下载 Java SDK 的安装包。Java SDK 的安装非常简单，基本上就是默认选择“下一步”即可完成安装。在安装完毕后，需要设置系统的环境变量。

以 Windows 10 为例，假设 Java 的安装路径为 D:\Program Files\Java\jdk1.8.0_171\，设置系统环境变量的方法为：右击“计算机”桌面图标，选择“属性”命令，然后选择“高级”选项卡，单击“环境变量”按钮，弹出“环境变量”对话框。在环境变量中增加 CLASSPATH 变量，值为 D:\Program Files\Java\jdk1.8.0_171\lib；在 Path 变量的值后面增加 D:\Program Files\Java\jdk1.8.0_171\bin，如图 1-2 所示。

（2）安装 Android Studio。Android Studio 可以在官网（https://developer.android.google.cn/studio/）或者其他网站进行下载。开发者需要选择一个合适的版本（Windows 32 位版、64 位版，Mac 版，Linux 版）。Android Studio 的安装也很简单，基本上也是默认选择“下一步”即可完成安装。

（3）下载 Android SDK。Android SDK 即开发 Android 软件所要使用的工具包。在进行 Android 应用程序开发时，需要选择相应版本的 SDK。下载 SDK 的方法是：选择 Tools/SDK Manager 菜单命令或单击工具栏中相应的 SDK Manager 按钮。打开 Android SDK

界面后，首先选择 Android SDK 的存储位置，然后在 SDK Platforms 选项卡中选择相应的 SDK 版本，单击 OK 按钮，如图 1-3 所示。

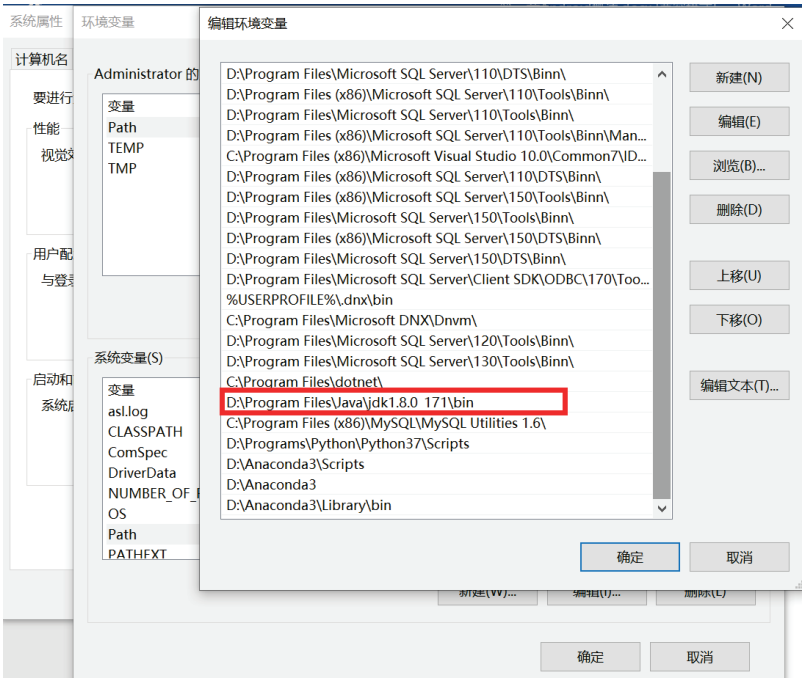

图 1-2　设置系统环境变量

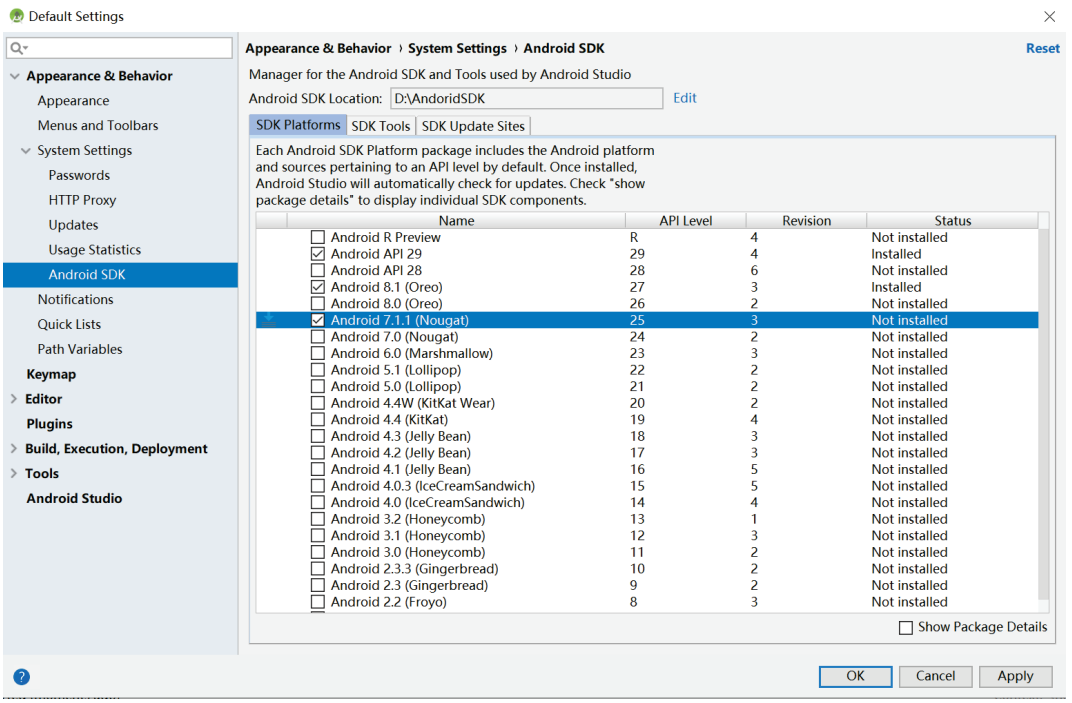

图 1-3　Android SDK 界面

Note

（4）创建虚拟设备。在编写完代码后，需要在模拟器或者手机中运行调试。如果要在虚拟设备中调试程序，则需要创建虚拟设备。创建虚拟设备的方法如下。

① 选择 Tools/AVD Manager 菜单命令或者单击工具栏中的 AVD Manager 按钮。

② 在弹出的窗口中，可以查看到已经创建的虚拟设备，也可以创建新的虚拟设备，如图 1-4 所示。

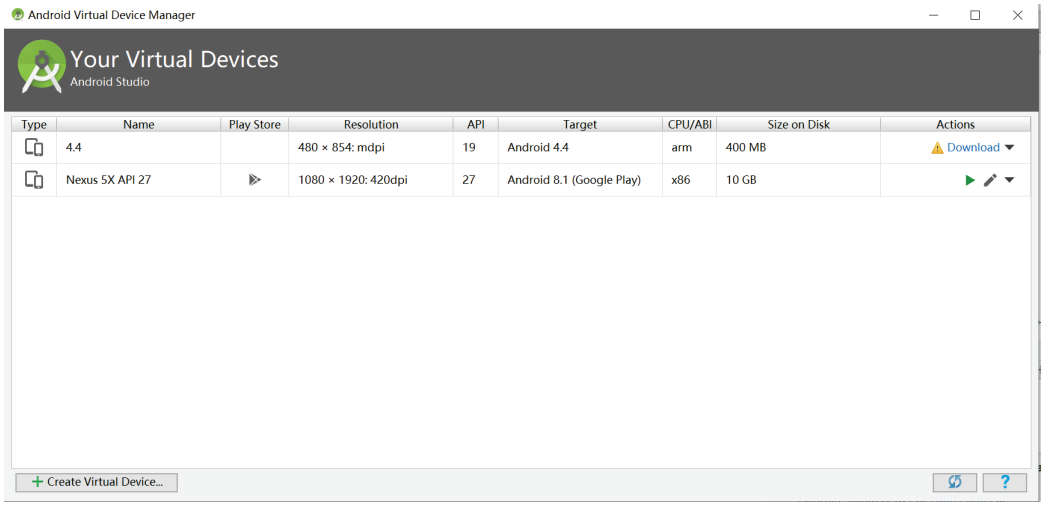

图 1-4　Android Virtual Device Manager 界面

③ 单击图 1-4 中的 Create Virtual Device 按钮，在 Virtual Device Configuration/Choose a device definition 中选择设备类型及模拟器的设备，然后单击 Next 按钮，如图 1-5 所示。

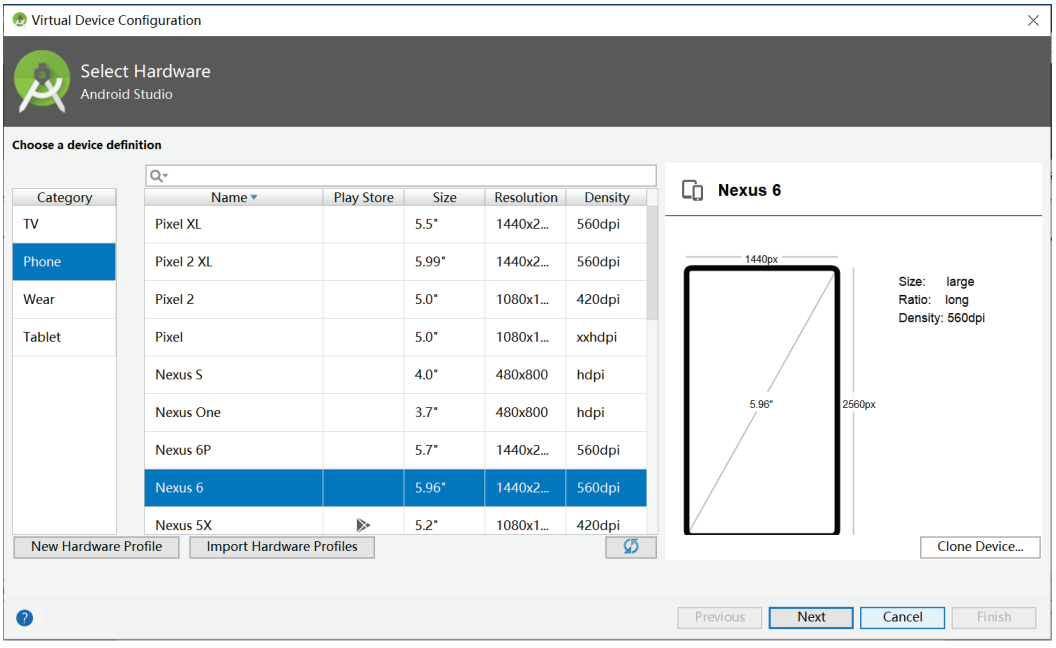

图 1-5　Choose a device definition 界面

④ 在 Virtual Device Configuration/Select a system image 中选择相应的镜像文件，然后单击 Download 进行下载，下载完毕后，单击 Next 按钮，如图 1-6 所示。

⑤ 在 Verify Configuration 界面中输入虚拟设备的名字，单击 Show Advanced Settings 可以对虚拟设备进行高级设置，例如手机方向、摄像头、网络、运行内存、机身内存、SD 卡等，如图 1-7 所示。

Note

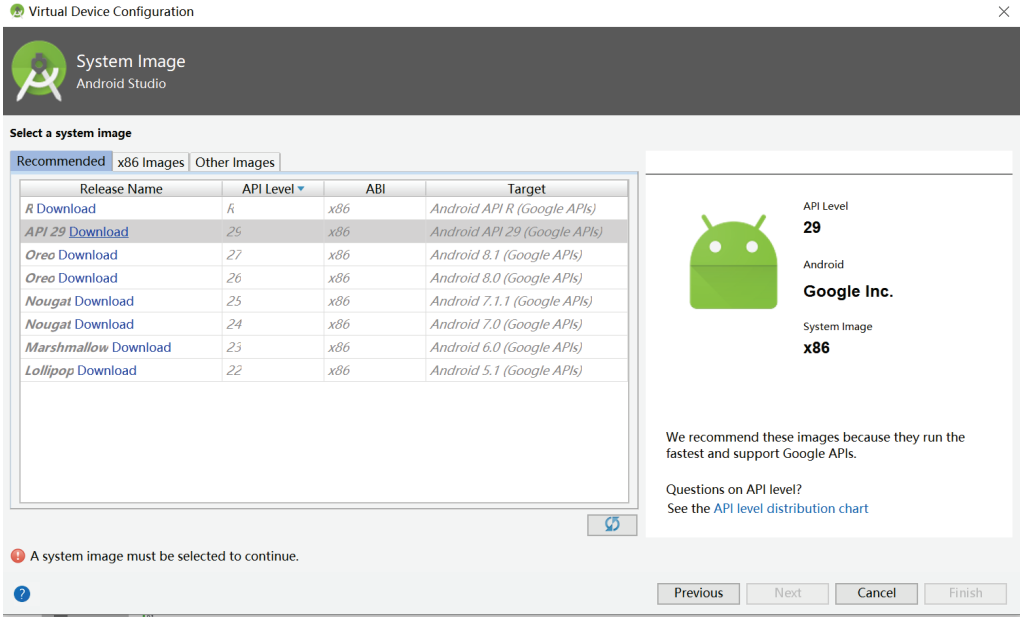

图 1-6　Select a system image 界面

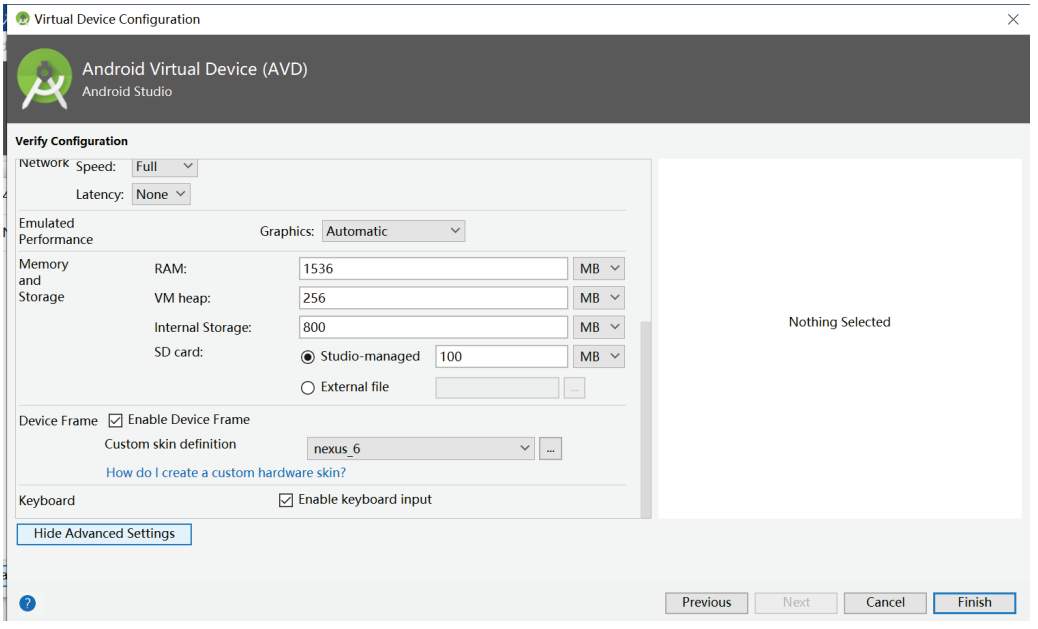

图 1-7　Verify Configuration 界面

1.5 创建 HelloWorld 项目

Note

完成 Android 开发环境的部署之后，就可以开始 Android 应用程序的开发之旅了。在本节，将创建第一个 Android 项目：Hello World。通过创建这个项目，主要介绍创建 Android 项目的过程。创建 Hello World 应用程序的步骤如下。

（1）启动 Android Studio，依次选择 File/New/New Project，将弹出创建新项目界面，输入 Application name（应用程序的名字）、Project location（项目保存位置），也可以对程序的包名进行编辑（或者采用默认），选择是否包括 C++、Kotlin 支持之后，单击 Next 按钮，如图 1-8 所示。

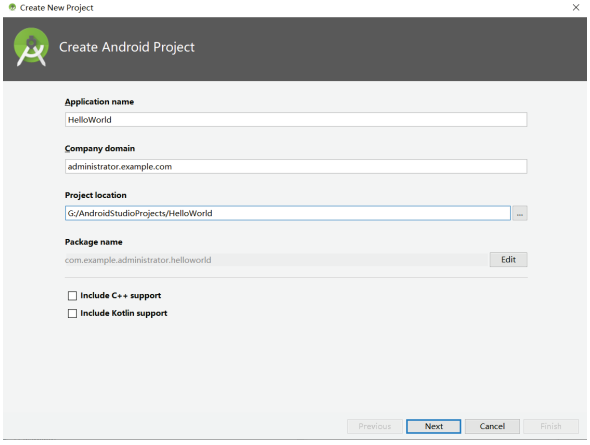

图 1-8　创建 Android 应用程序步骤 1

（2）选择目标设备及最低的 SDK 要求。设备类型包括手机与平板、穿戴设备、电视、Android 汽车设备、Android 物联网设备等，如图 1-9 所示。

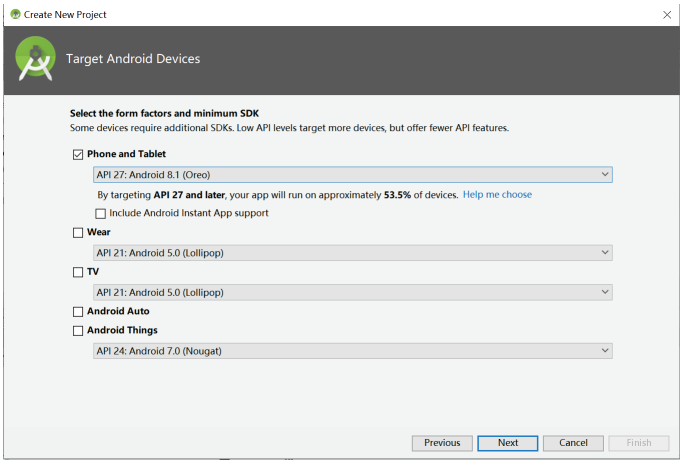

图 1-9　创建 Android 应用程序步骤 2

Note

（3）为应用程序选择 Activity 类型，默认的为空 Activity，如图 1-10 所示。

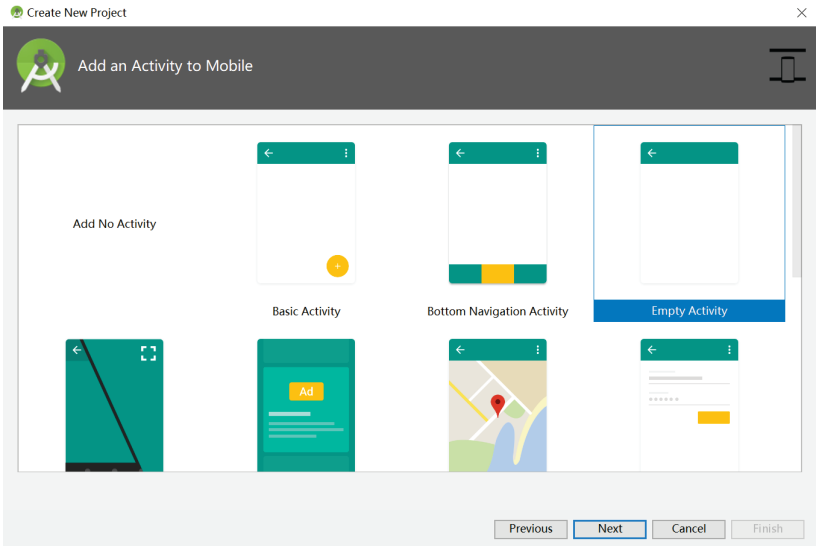

图 1-10　创建 Android 应用程序步骤 3

（4）输入默认启动 Activity 的类名及布局文件名，如图 1-11 所示。注意，如果选中 Backwards Compatibility(AppCompat)，则在应用程序中配置会增加相应的依赖，有可能会引起版本冲突。如果引起冲突，需要进行相应的配置，配置方法为：打开 build.gradle 文件，找到 dependencies 节点，将 com.android.support 后的版本号修改为合适的版本号（需要根据具体的情况来确定）即可，代码如下所示。

```
dependencies {
    implementation 'com.android.support:appcompat-v7:27.1.1+'
}
```

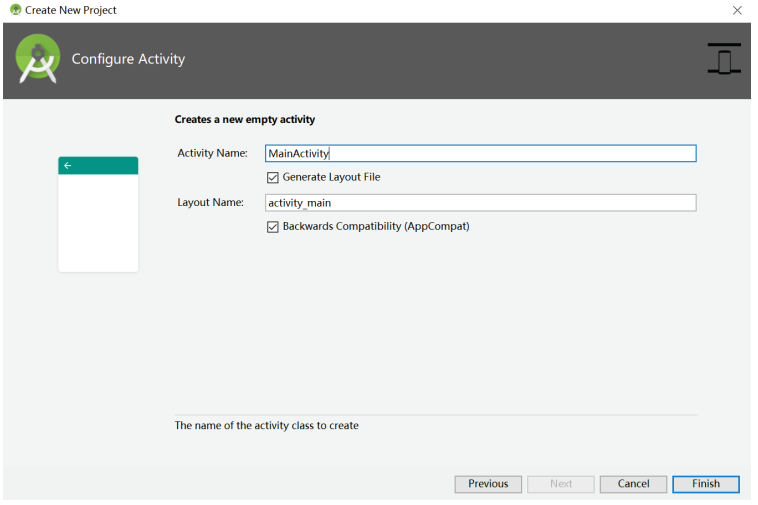

图 1-11　创建 Android 应用程序步骤 4

（5）创建完毕后，选择菜单栏中的 Run/Run App 命令或者单击工具栏中的绿色三角按钮，然后选择相应的运行设备，即可运行创建好的 Android 程序，如图 1-12 所示。

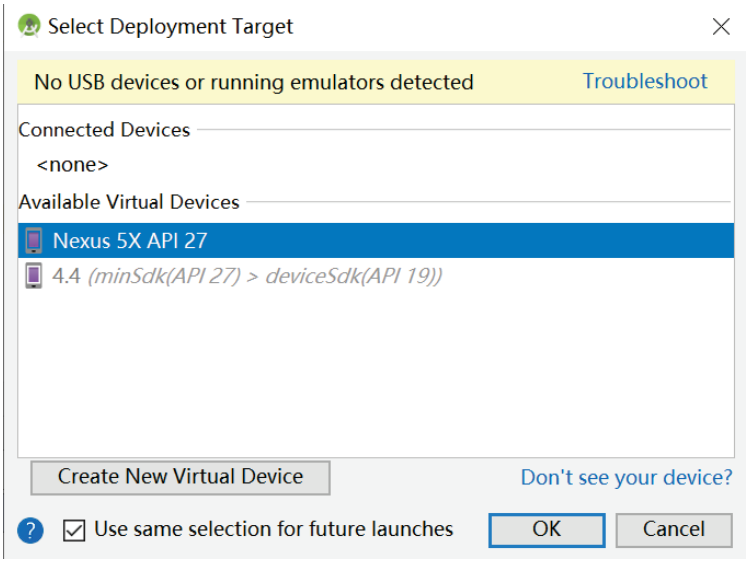

图 1-12　选择运行设备

1.6　Android 项目目录结构

在进行 Android 应用程序开发时，开发者需要了解 Android 项目的目录结构，这样才能明白在开发程序时需要在哪里编写代码、需要做什么工作、相应的文件需要存放在什么位置。下面以 1.5 节所创建的 HelloWorld 项目为例，介绍 Android 项目的目录结构。

在 Android Studio 中，提供了多种项目结构类型，但是最常用的有两种：Android 结构类型与 Project 结构类型，两者之间的区别在于查看项目结构的角度不同，切换两种结构类型的方法如图 1-13 所示，下面分别进行介绍。

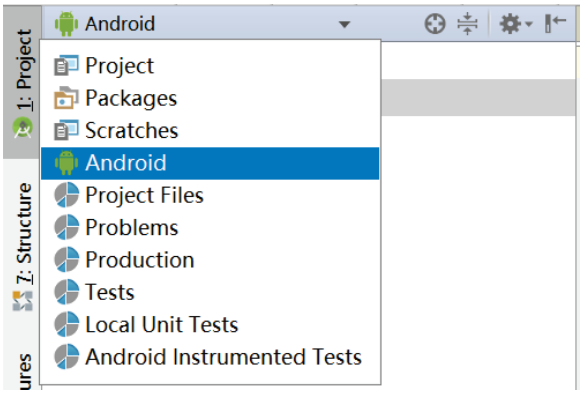

图 1-13　切换项目结构类型

1.6.1 Android 结构类型

Note

Android 结构类型是最常用的一种结构类型，目录结构如图 1-14 所示。下面对 Android 目录结构视图的组成进行介绍。

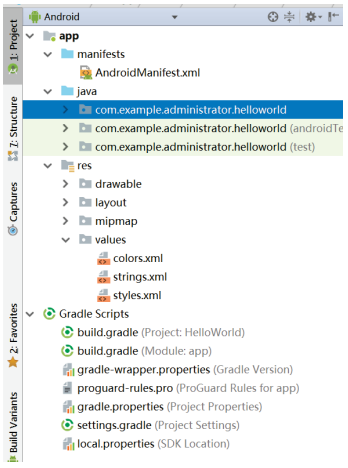

图 1-14　Android 目录结构视图

（1）app/manifests/AndroidManifest.xml：对应用程序进行配置，如声明应用程序所包含的 Activity 类、广播接收者、权限等。

（2）app/java: 主要为应用程序的源代码和测试代码。在 Java 下面可以建立多个 Package（包），在一个 Package 中可以建立多个 Java 类，主要用于实现应用程序的功能。在 Java 中包括三个包，即 com.example.administrator.helloworld 为创建的项目 HelloWorld 的包；androidTest、test 为用来编写 Android Test 测试用例的包，可以对项目进行一些自动化测试。

（3）app/res: 主要是资源目录包，存储项目使用的所有资源。drawable 用于存储一些图片以及关于图片形状的.xml 文件，*dpi 表示存储分辨率的图片，用于适配不同的屏幕；layout 用于存储布局文件；mipmap 用于存储 app 的图标资源；values 用于存储 app 引用的一些公共值，其中 colors.xml 存储了一些 color 的样式；strings.xml 存储了引用的 string 值；styles.xml 存储了 app 需要用到的一些样式。

（4）Gradle Scripts/build.gradle：项目的 gradle 配置文件。注意，在一个项目中有两个 build.gradle 文件，一个位于应用程序的目录下，作为 gradle 项目自动编译的配置文件；一个位于应用程序目录/app 文件夹下，作为 app 模块的 gradle 编译文件。

（5）Gradle Scripts/gradle-wrapper.properties：存储一些项目对 gradle 的配置信息，声明了 gradle 的目录与下载路径以及当前项目使用的 gradle 版本。默认的路径一般不会更改。

（6）Gradle Scripts/proguard-rules.pro：指定项目代码的混淆规则，当代码开发完成后打成安装包文件，如果不希望代码被别人破解，通常会将代码混淆，从而让破解者难以阅读。

Note

（7）Gradle Scripts/gradle.properties：配置 gradle 运行环境的文件，如配置 gradle 运行模式，运行时 jvm 虚拟机的大小。

（8）Gradle Scripts/settings.gradle：指定项目中所有引入的模块。由于 HelloWorld 项目中就只有一个 app 模块，因此该文件中也就只引入了 app 这一个模块。通常情况下模块的引入都是自动完成的，需要开发者手动去修改这个文件的场景可能比较少。

（9）Gradle Scripts/local.properties：指定本机中的 Android SDK 路径，通常内容都是自动生成的，并不需要修改。除非本机中的 Android SDK 位置发生了变化，那么就将这个文件中的路径改成新的位置即可。

1.6.2 Project 结构类型

Android 项目另外一种常用结构类型就是 Project，目录结构如图 1-15 所示。下面对 Project 目录结构视图的组成进行介绍。

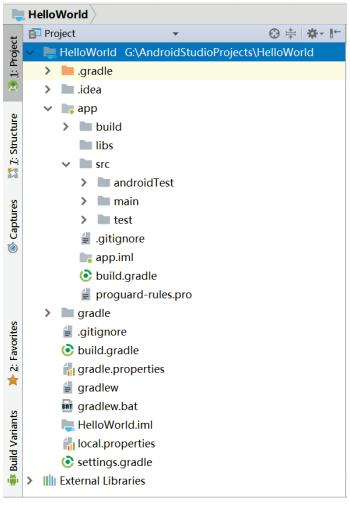

图 1-15 Project 目录结构视图

（1）.gradle 和.idea：这两个目录下放置的都是 Android Studio 自动生成的一些文件，开发者无须关心，也不要手动编辑。

（2）app：项目中的代码、资源等内容几乎都是放置在这个目录下，Android 应用程序的开发工作也基本都是在这个目录下进行的，因为这个目录比较重要，下面对这个目录单独展开进行讲解。

① build：这个目录和外层的 build 目录类似，主要包含了一些在编译时自动生成的文件，且它里面的内容会更多、更杂，开发者不需要过多关心。

② libs：如果项目中使用到第三方 jar 包，就需要把这些 jar 包都放在 libs 目录下，放在这个目录下的 jar 包都会被自动添加到构建路径。

③ src/androidTest：此处是用来编写 Android Test 测试用例的，可以对项目进行一些自动化测试。

Note

④ src/main/java：毫无疑问，java 目录是放置开发者所有 java 代码的地方，展开该目录，可看到 HelloWorld 项目的 MainActivity 文件。

⑤ src/main/res：这个目录下的内容比较多，在项目中使用的所有图片、布局、字符串等资源都要存放在这个目录的子目录下，如图片放在 drawable 目录下，布局放在 layout 目录下，字符串放在 values 目录下。

⑥ src/main/AndroidManifest.xml：整个 Android 项目的配置文件，在程序中定义的四大组件都需要在这个文件里注册，另外还可以在这个文件中给应用程序添加权限声明。

⑦ src/test：此处是用来编写 Unit Test 测试用例的，是对项目进行自动化测试的另一种方式。

⑧ .gitignore：将 app 模块内指定的目录或文件排除在版本控制之外，作用和外层的.gitignore 文件类似。

⑨ app.iml：IntelliJ IDEA 项目自动生成的文件，开发者不需要关心或修改这个文件中的内容。

⑩ build.gradle：这是 app 模块的 gradle 构建脚本，这个文件中会指定很多项目构建相关的配置。

⑪ proguard-rules.pro：指定项目代码的混淆规则，当代码开发完成后打成安装包文件，如果不希望代码被别人破解，通常会将代码混淆，从而让破解者难以阅读。

（3）gradle：包含了 gradle wrapper 的配置文件，使用 gradle wrapper 的方式不需要提前将 gradle 下载好，而是会自动根据本地的缓存情况决定是否需要联网下载 gradle。Android Studio 默认没有启动 gradle wrapper 的方式，如果需要打开，可以选择 Android Studio 导航栏中的 File /Settings /Build/Execution/Deployment/Gradle，进行配置。

（4）.gitignore：将指定的目录或文件排除在版本控制之外。

（5）build.gradle：项目全局的 gradle 构建脚本，通常这个文件的内容是不需要修改的。

（6）gradle.properties：这个文件是全局的 gradle 配置文件，在这里配置的属性将会影响到项目中所有的 gradle 编译脚本。

（7）gradlew 和 gradlew.bat：这两个文件用来在命令行界面中执行 gradle 命令，其中 gradlew 适用于 Linux 或 Mac 系统，gradlew.bat 适用于 Windows 系统。

（8）HelloWorld.iml：.iml 文件是所有 IntelliJ IDEA 项目都会自动生成的一个文件（Android Studio 是基于 IntelliJ IDEA 开发的），用于标识这是一个 IntelliJ IDEA 项目，开发者不需要修改这个文件中的任何内容。

（9）local.properties：指定本机中的 Android SDK 路径，通常内容都是自动生成的，开发者并不需要修改。除非 Android SDK 位置发生了变化，才将这个文件中的路径改成新的位置。

（10）settings.gradle：指定项目中所有引入的模块。由于 HelloWorld 项目中就只有一个 app 模块，因此该文件中也就只引入了 app 这一个模块。通常情况下模块的引入都是自动完成的，需要开发者手动修改这个文件的场景可能比较少。

1.7 习 题

Note

1．Android 系统的内核基础系统是（ ）。

A．UNIX B．Windows C．Linux D．Dos

2．下面的选项中，不属于 Android 应用程序组件的是（ ）。

A．Activity B．Intent C．Service D．ContentProvider

3．在 Android 中，使用的内置数据库是（ ）。

A．SQL Server B．SQLite C．MySQL D．NoSQL

4．简述 Android 平台的特点。

5．简述 Android 的四种基本组件及其作用。

6．简述 Android 平台架构的各组成部分及其作用。

7．简述 Android 项目中各组成文件的作用。