第 1 章　开始启程，你的第一行 Android 代码

欢迎你来到 Android 世界！Android 系统是目前世界上市场占有率最高的移动操作系统，不管你在哪里，几乎都可以看到人人手中都会有一部 Android 手机。虽然今天的 Android 世界欣欣向荣，可是你知道它的过去是什么样的吗？我们一起来看一看它的发展史吧。

2003 年 10 月，Andy Rubin 等人一起创办了 Android 公司。2005 年 8 月谷歌收购了这家仅仅成立了 22 个月的公司，并让 Andy Rubin 继续负责 Android 项目。在经过了数年的研发之后，谷歌终于在 2008 年推出了 Android 系统的第一个版本。但自那之后，Android 的发展就一直受到重重阻挠。乔布斯自始至终认为 Android 是一个抄袭 iPhone 的产品，里面剽窃了诸多 iPhone 的创意，并声称一定要毁掉 Android。而本身就是基于 Linux 开发的 Android 操作系统，在 2010 年被 Linux 团队从 Linux 内核主线中除名。又由于 Android 中的应用程序都是使用 Java 开发的，甲骨文则针对 Android 侵犯 Java 知识产权一事对谷歌提起了诉讼……

可是，似乎再多的困难也阻挡不了 Android 快速前进的步伐。由于谷歌的开放政策，任何手机厂商和个人都能免费地获取到 Android 操作系统的源码，并且可以自由地使用和定制。三星、HTC、摩托罗拉、索爱等公司都推出了各自系列的 Android 手机，Android 市场上百花齐放。仅仅推出两年后，Android 就超过了已经霸占市场逾十年的诺基亚 Symbian，成为了全球第一大智能手机操作系统，并且每天都还会有数百万台新的 Android 设备被激活。目前 Android 已经占据了全球智能手机操作系统 70%以上的份额。

说了这些，想必你已经体会到 Android 系统炙手可热的程度，并且迫不及待地想要加入到 Android 开发者的行列当中了吧。试想一下，十个人中有七个人的手机都可以运行你编写的应用程序，还有什么能比这个更诱人的呢？那么从今天起，我就作为你 Android 旅途中的导师，一步步地引导你成为一名出色的 Android 开发者。

好了，现在我们就来一起初窥一下 Android 世界吧。

经验值：+5　　目前经验值：5
级别：萌级小菜鸟
捡到宝物：Android 前辈遗失的入门级通用修行卡一张。卡略有磨损，但仍可使用。

1.1 了解全貌，Android 王国简介

Android 从面世以来到现在已经发布了近二十个版本了。在这几年的发展过程中，谷歌为 Android 王国建立了一个完整的生态系统。手机厂商、开发者、用户之间相互依存，共同推进着 Android 的蓬勃发展。开发者在其中扮演着不可或缺的角色，因为再优秀的操作系统没有开发者来制作丰富的应用程序也是难以得到大众用户喜爱的，相信没有多少人能够忍受没有 QQ、微信的手机吧？而谷歌推出的 Google Play 更是给开发者带来了大量的机遇，只要你能制作出优秀的产品，在 Google Play 上获得了用户的认可，你就完全可以得到不错的经济回报，从而成为一名独立开发者，甚至是成功创业！

那我们现在就以一个开发者的角度，去了解一下这个操作系统吧。纯理论型的东西也比较无聊，怕你看睡着了，因此我只挑重点介绍，这些东西跟你以后的开发工作都是息息相关的。

1.1.1 Android 系统架构

为了让你能够更好地理解 Android 系统是怎么工作的，我们先来看一下它的系统架构。Android 大致可以分为四层架构，五块区域。

1. Linux 内核层

   Android 系统是基于 Linux 2.6 内核的，这一层为 Android 设备的各种硬件提供了底层的驱动，如显示驱动、音频驱动、照相机驱动、蓝牙驱动、Wi-Fi 驱动、电源管理等。

2. 系统运行库层

   这一层通过一些 C/C++库来为 Android 系统提供了主要的特性支持。如 SQLite 库提供了数据库的支持，OpenGL|ES 库提供了 3D 绘图的支持，Webkit 库提供了浏览器内核的支持等。

   同样在这一层还有 Android 运行时库，它主要提供了一些核心库，能够允许开发者使用 Java 语言来编写 Android 应用。另外 Android 运行时库中还包含了 Dalvik 虚拟机，它使得每一个 Android 应用都能运行在独立的进程当中，并且拥有一个自己的 Dalvik 虚拟机实例。相较于 Java 虚拟机，Dalvik 是专门为移动设备定制的，它针对手机内存、CPU 性能有限等情况做了优化处理。

3. 应用框架层

   这一层主要提供了构建应用程序时可能用到的各种 API，Android 自带的一些核心应用就是使用这些 API 完成的，开发者也可以通过使用这些 API 来构建自己的应用程序。

4. 应用层

   所有安装在手机上的应用程序都是属于这一层的，比如系统自带的联系人、短信等程序，或者是你从 Google Play 上下载的小游戏，当然还包括你自己开发的程序。

结合图 1.1 你将会理解得更加深刻，图片源自维基百科。

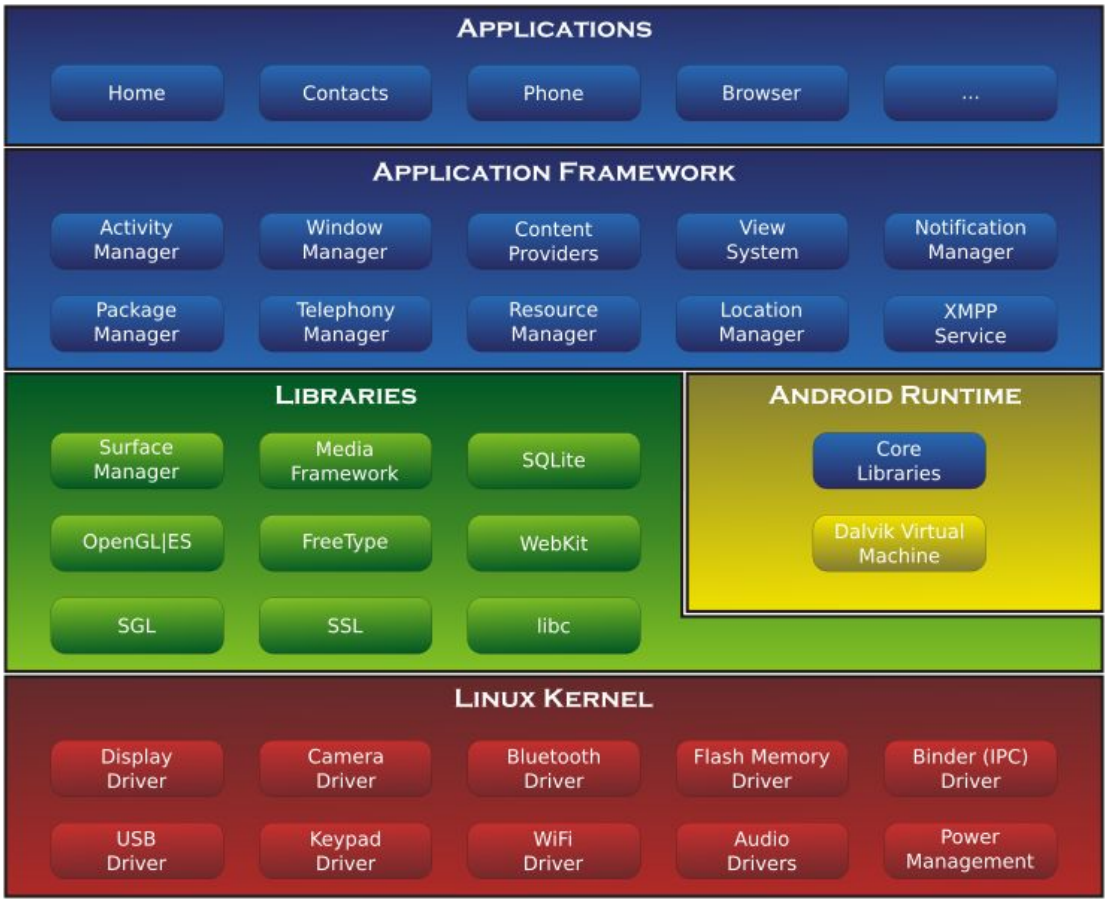

图　1.1

1.1.2　Android 已发布的版本

2008 年 9 月，谷歌正式发布了 Android 1.0 系统，这也是 Android 系统最早的版本。随后的几年，谷歌以惊人的速度不断地更新 Android 系统，2.1、2.2、2.3 系统的推出使 Android 占据了大量的市场。2011 年 2 月，谷歌发布了 Android 3.0 系统，这个系统版本是专门为平板电脑设计的，但也是 Android 为数不多比较失败的版本，推出之后一直不见什么起色，市场份额也少得可怜。不过很快，在同年的 10 月，谷歌又发布了 Android 4.0 系统，这个版本不再对手机和平板进行差异化区分，既可以应用在手机上也可以应用在平板上，除此之外还引入了不少新特性。目前最新的系统版本已经是 4.4 KitKat。

下表中列出了目前市场上主要的一些 Android 系统版本及其详细信息。你看到这张表格时，数据很可能已经发生了变化，查看最新的数据可以访问 http://developer.android.com/about/dashboards/。

版本号	系统代号	API	市场占有率
2.2	Froyo	8	1.2%
2.3.3 – 2.3.7	Gingerbread	10	19.0%
3.2	Honeycomb	13	0.1%
4.0.3 – 4.0.4	Ice Cream Sandwich	15	15.2%
4.1.x	Jelly Bean	16	35.3%
4.2.x		17	17.1%
4.3		18	9.6%
4.4	KitKat	19	2.5%

从上表中可以看出，目前 4.0 以上的系统已经占据了 80%左右的 Android 市场份额，而且以后这个数字还会不断增加，因此我们本书中开发的程序也是主要面向 4.0 以上的系统，2.x 的系统就不再去兼容了。

1.1.3 Android 应用开发特色

预告一下，你马上就要开始真正的 Android 开发旅程了。不过先别急，在开始之前我们再来一起看一看，Android 系统到底提供了哪些东西，供我们可以开发出优秀的应用程序。

1. 四大组件

Android 系统四大组件分别是活动（Activity）、服务（Service）、广播接收器（Broadcast Receiver）和内容提供器（Content Provider）。其中活动是所有 Android 应用程序的门面，凡是在应用中你看得到的东西，都是放在活动中的。而服务就比较低调了，你无法看到它，但它会一直在后台默默地运行，即使用户退出了应用，服务仍然是可以继续运行的。广播接收器可以允许你的应用接收来自各处的广播消息，比如电话、短信等，当然你的应用同样也可以向外发出广播消息。内容提供器则为应用程序之间共享数据提供了可能，比如你想要读取系统电话簿中的联系人，就需要通过内容提供器来实现。

2. 丰富的系统控件

Android 系统为开发者提供了丰富的系统控件，使得我们可以很轻松地编写出漂亮的界面。当然如果你品味比较高，不满足于系统自带的控件效果，也完全可以定制属于自己的控件。

3. SQLite 数据库

Android 系统还自带了这种轻量级、运算速度极快的嵌入式关系型数据库。它不仅

支持标准的 SQL 语法，还可以通过 Android 封装好的 API 进行操作，让存储和读取数据变得非常方便。

4. 地理位置定位

移动设备和 PC 相比起来，地理位置定位功能应该可以算是很大的一个亮点。现在的 Android 手机都内置有 GPS，走到哪儿都可以定位到自己的位置，发挥你的想象就可以做出创意十足的应用，如果再结合上功能强大的地图功能，LBS 这一领域潜力无限。

5. 强大的多媒体

Android 系统还提供了丰富的多媒体服务，如音乐、视频、录音、拍照、闹铃等等，这一切你都可以在程序中通过代码进行控制，让你的应用变得更加丰富多彩。

6. 传感器

Android 手机中都会内置多种传感器，如加速度传感器、方向传感器等，这也算是移动设备的一大特点。通过灵活地使用这些传感器，你可以做出很多在 PC 上根本无法实现的应用。

既然有 Android 这样出色的系统给我们提供了这么丰富的工具，你还用担心做不出优秀的应用吗？好了，纯理论的东西也就介绍到这里，我知道你已经迫不及待想要开始真正的开发之旅了，那我们就开始启程吧！

1.2　手把手带你搭建开发环境

俗话说得好，工欲善其事，必先利其器，开着记事本就想去开发 Android 程序显然不是明智之举，选择一个好的 IDE 可以极大幅度地提高你的开发效率，因此本节我就将手把手带着你把开发环境搭建起来。

1.2.1　准备所需要的软件

我现在对你了解还并不多，但我希望你已经是一个颇有经验的 Java 程序员，这样你理解本书的内容时将会轻而易举，因为 Android 程序都是使用 Java 语言编写的。如果你对 Java 只是略有了解，那阅读本书应该会有一点困难，不过一边阅读一边补充 Java 知识也是可以的。但如果你对 Java 完全没有了解，那么我建议你可以暂时将本书放下，先买本介绍 Java 基础知识的书学上两个星期，把 Java 的基本语法和特性都学会了，再来继续阅读本书。

好了，既然你已经阅读到这里，说明你已经掌握 Java 的基本用法了，那么开发 Java 程序时必备的 JDK 你一定已经安装好了。下面我们再来看一看开发 Android 程序除了 JDK 外，还需要哪些工具。

1. Android SDK

Android SDK 是谷歌提供的 Android 开发工具包，在开发 Android 程序时，我们需

要通过引入该工具包，来使用 Android 相关的 API。

2. Eclipse

相信所有 Java 开发者都一定会对这个工具非常地熟悉，它是 Java 开发神器，最好用的 IDE 工具之一。Eclipse 是开源的，这使得有很多基于 Eclipse 制作的优秀 IDE 得以问世，如 MyEclipse、Aptana 等。但我觉得它最吸引人的地方并不在这儿，而是它超强的插件功能。Eclipse 支持极多的插件工具，使得它不仅仅可以用来开发 Java，还可以很轻松地支持几乎所有主流语言的开发，当然也非常适合 Android 开发。

除了 Eclipse 外，同样适合开发 Android 程序的 IDE 还有 IntelliJ IDEA、Android Studio 等。其中 Android Studio 是谷歌官方近期推出的新 IDE，由于是专门为开发 Android 程序定制的，在 Android 领域大有要取代 Eclipse 的势头。不过本书中还是决定暂时继续使用 Eclipse，因为 Android Studio 才推出不久，恐怕还不够稳定。另外你将来的同事大多数应该还是用的 Eclipse，如果跟他们选择不同的 IDE，在工作效率上可能要打点折扣了。

3. ADT

ADT 全称 Android Development Tools，是谷歌提供的一个 Eclipse 插件，用于在 Eclipse 中提供一个强大的、高度集成的 Android 开发环境。安装了 ADT，你不仅可以联机调试，而且还能够模拟各种手机事件、分析你的程序性能等等。由于是 Eclipse 的插件，你不需要进行下载，在 Eclipse 中在线安装就可以了。

1.2.2 搭建开发环境

你可以将上述的软件全部都准备好，然后一个个安装完成（我当年就是这么干的），不过这已经是老方法了。谷歌现在提供了一种简便方式，在 Android 官网可以下载到一个绑定好的 SDK 工具包，你所需要用到的 Android SDK、Eclipse、ADT 插件全都包含在里面了，这样可以省去很多费时的安装操作。下载地址是：http://developer.android.com/sdk/。

你下载下来的将是一个压缩包，解压该压缩包之后的目录结构如图 1.2 所示。

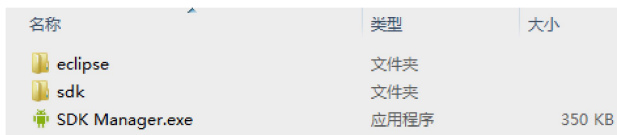

图 1.2

其中 SDK Manager 就是我们 Android SDK 的管理器，双击打开它可以看到所有可下载的 Android SDK 版本。由于 Android 版本已经非常多了，全部都下载会很耗时，并且前面我也说过，我们开发的程序主要面向 Android 4.0 以后的系统，因此这里我只勾选 API 14 以上的 SDK 版本，如图 1.3 所示。当然如果你带宽和硬盘都十分充足，也可以全部勾选。

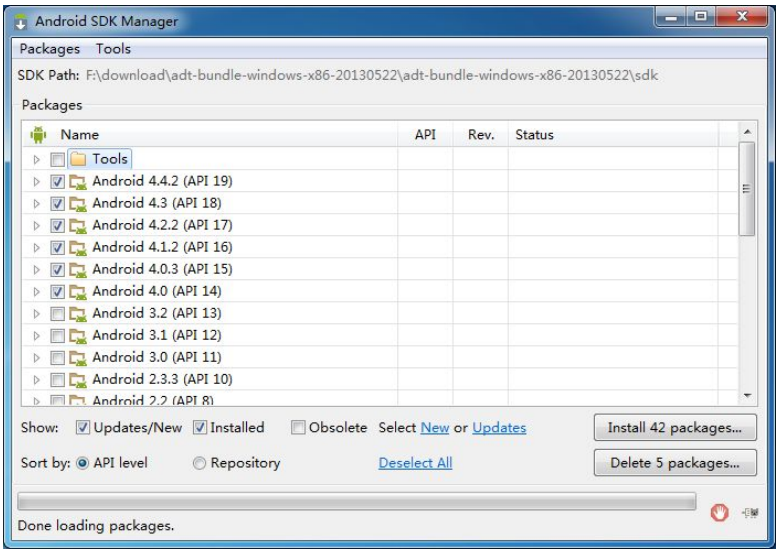

图　1.3

勾选完后点击右下角的 Install 42 packages，然后会进入到一个确认安装界面，如图 1.4 所示。

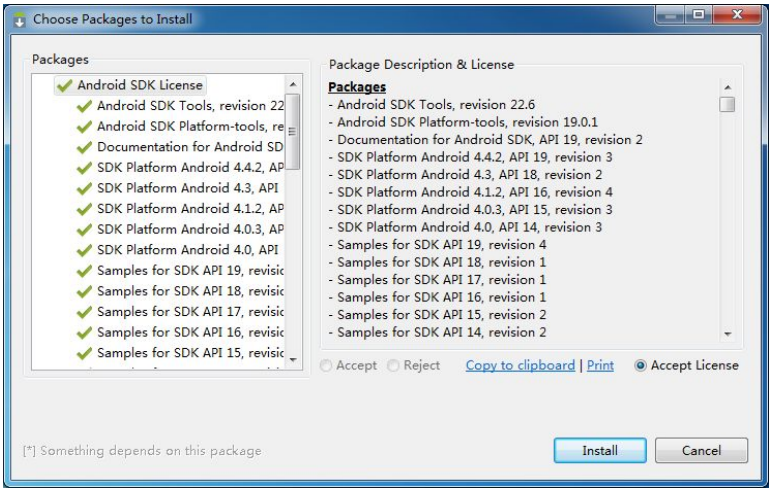

图　1.4

选中右下角的 Accept License，然后点击 Install，就进入了一个漫长的等待过程。这个时候也是你最轻松的时候了，因为你没什么事情要干，只需要等待就好。现在你可以喝杯茶，休息一会，如果你勾选的 SDK 比较多的话，干脆先去睡个觉吧！

经过漫长的等待之后，SDK 终于是下载完成了。所有下载好的内容都放在了 sdk 这个目录下，除了开发工具包外，里面还包含文档、源码、示例等等。具体的东西等你用到的时候我再做介绍，目前你不用太过关心 sdk 这个目录下的内容，里面的东西过多，现在容易让你头晕眼花。

好了，sdk 这个目录就先不管它了，是时候来看下 eclipse 这个目录了。其实这个目录也没什么好说的，就是进入 eclipse 目录，双击 eclipse.exe 来启动 Eclipse 就完了。这个 Eclipse 是安装好 ADT 插件的，因此你已经可以直接在这个 Eclipse 上开发 Android 程序了，那还不快点对着启动图标点右键，发送到桌面快捷方式！

Eclipse 的界面你应该是比较熟悉了，不过安装过 ADT 插件的 Eclipse 会多出一些东西来，比如你会在 Eclipse 的工具栏中找到图 1.5 所示的几个图标。

图 1.5

这几个图标你应该是没有见过的，我来简单为你介绍下。最左边的图标其实你已经比较熟悉了，就是你睡觉前使用过的 Android SDK 管理器，点击它和点击 SDK Manager 效果是一样的。中间的图标是用来开启 Android 模拟器的，如果你还没有 Android 手机的话，开发时就必须使用模拟器了。最右边的图标是用来进行代码检查的，你暂时还用不到它。

那我们现在就来启动一个模拟器看看效果吧，点击中间的图标会弹出如图 1.6 所示的窗口。

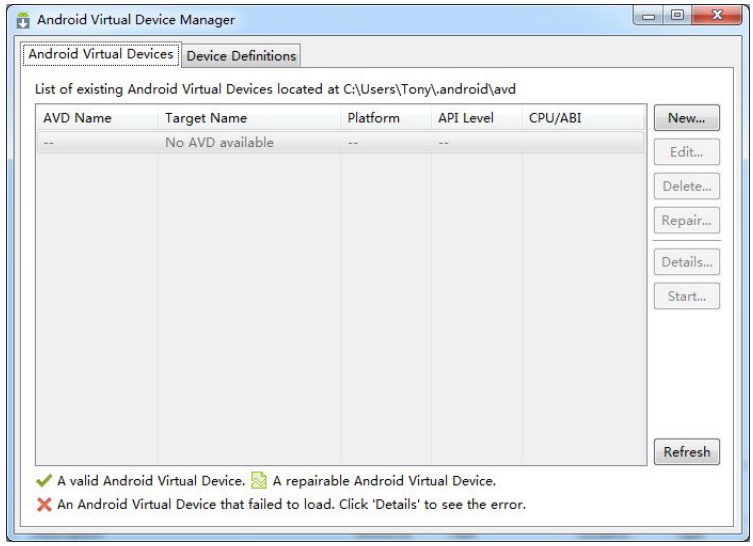

图 1.6

然后点击右侧的 New 来创建一个新的模拟器。这里我们准备创建一个 Android 4.0 系统的模拟器，因此模拟器名就叫 4.0 好了，设备这里我选择了一个 3.2 英寸屏幕的手机，目标指定为 Android 4.0，然后再稍微分配一下手机内存和 SD 卡大小，就可以点击 OK 了，如图 1.7 所示。

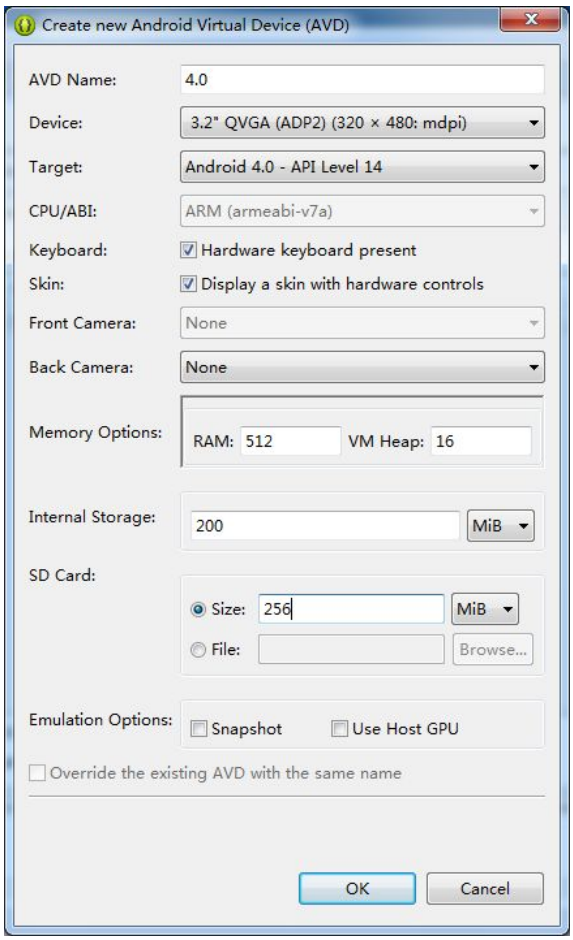

图　1.7

创建完成后，我们选中刚刚创建的模拟器，然后点击 Start，在弹出窗口中点击 Launch，就可以启动模拟器了。模拟器会像手机一样，有一个开机过程，启动完成之后的界面如图 1.8 所示。

图 1.8

很清新的 Android 界面出来了！看上去还挺不错吧，你几乎可以像使用手机一样使用它，Android 模拟器对手机的模仿度非常高，快去体验一下吧。

模拟器的右侧是一块键盘区域，其中中间的四个按键非常重要，从左到右依次是 Home 键、Menu 键、Back 键和 Search 键。Home 键让你在任何时候都可以回到桌面，Menu 键用于在程序界面中显示菜单，Back 键用于返回到上一个界面，Search 键让你可以更加轻松地使用谷歌搜索功能。

目前为止，Android 开发环境就已经全部搭建完成了。那现在应该做什么？当然是写下你的第一行 Android 代码了，让我们快点开始吧。

经验值：+100　　目前经验值：105

级别：萌级小菜鸟

赢得宝物：战胜开发环境搭建外围守卫者。拾取守卫者掉落的宝物，小屏幕二手 Android 手机一部、全新 Android 模拟器一个、九成新粗布 Android 战袍一套、微型信心增强大力丸一颗。穿戴好战袍，服下大力丸。继续前进。

1.3　创建你的第一个 Android 项目

任何一个编程语言写出的第一个程序毫无疑问都会是 Hello World，这已经是自 20 世纪 70 年代一直流传下来的传统，在编程界已成为永恒的经典，那我们当然也不会搞例外了。

1.3.1　创建 HelloWorld 项目

在 Eclipse 的导航栏中点击 File→New→Android Application Project，此时会弹出创建 Android 项目的对话框。其中 Application Name 代表应用名称，此应用安装到手机之后会在手机上显示该名称，这里我们填入 Hello World。Project Name 代表项目名称，在项目创建完成后该名称会显示在 Eclipse 中，这里我们填入 HelloWorld（项目名通常不加空格）。接着 Package Name 代表项目的包名，Android 系统就是通过包名来区分不同应用程序的，因此包名一定要有唯一性，这里我们填入 com.test.helloworld。

接下来是几个下拉选择框，Minimum Required SDK 是指程序最低兼容的版本，这里我们选择 Android 4.0。Target SDK 是指你在该目标版本上已经做过了充分的测试，系统不会再帮你在这个版本上做向前兼容的操作了，这里我们选择最高版本 Android 4.4。Compile With 是指程序将使用哪个版本的 SDK 进行编译，这里我们同样选择 Android 4.0。最后一个 Theme 是指程序 UI 所使用的主题，我个人比较喜欢选择 None。全部都选择好的界面如图 1.9 所示。

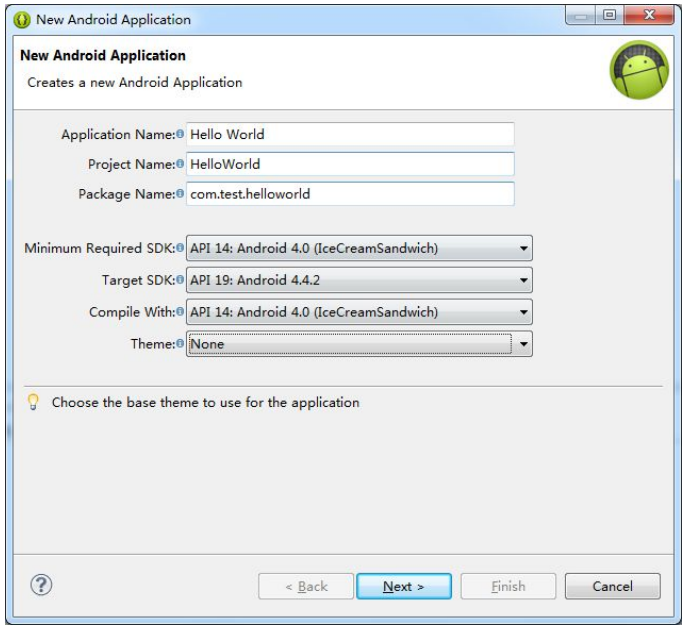

图　1.9

现在我们可以点击 Next 了，下一个界面是创建项目的一些配置，全部保持默认配置就好，如图 1.10 所示。

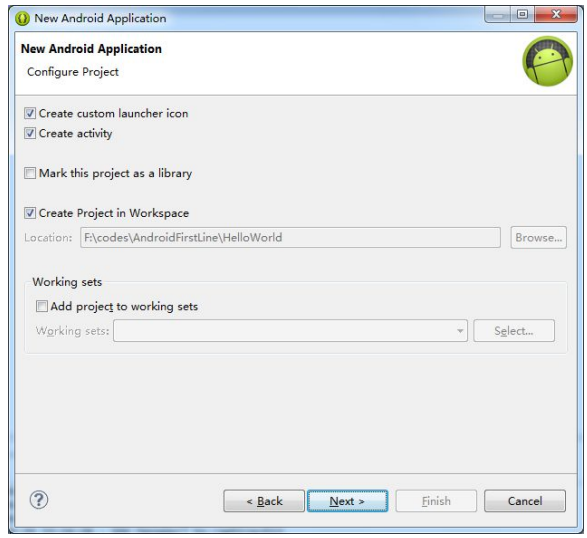

图　1.10

直接点击 Next 进入到启动图标的配置界面，在这里配置的图标就会是你的应用程序安装到手机之后显示的图标，如图 1.11 所示。

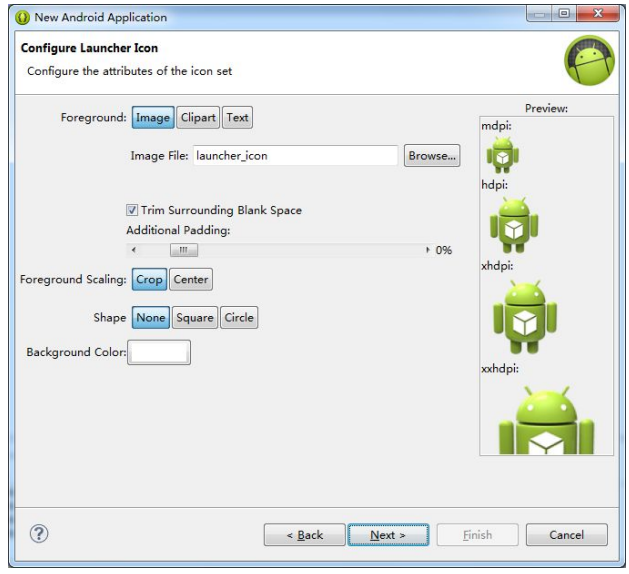

图　1.11

如果你程序的 Logo 还没设计好，别着急，在项目里面也是可以配置启动图标的，这里我们就先不配置，直接点击 Next。

然后跳转到的是创建活动界面，在这个界面你可以选择一个你想创建的活动类型，这里我们就选择 Blank Activity 了，如图 1.12 所示。

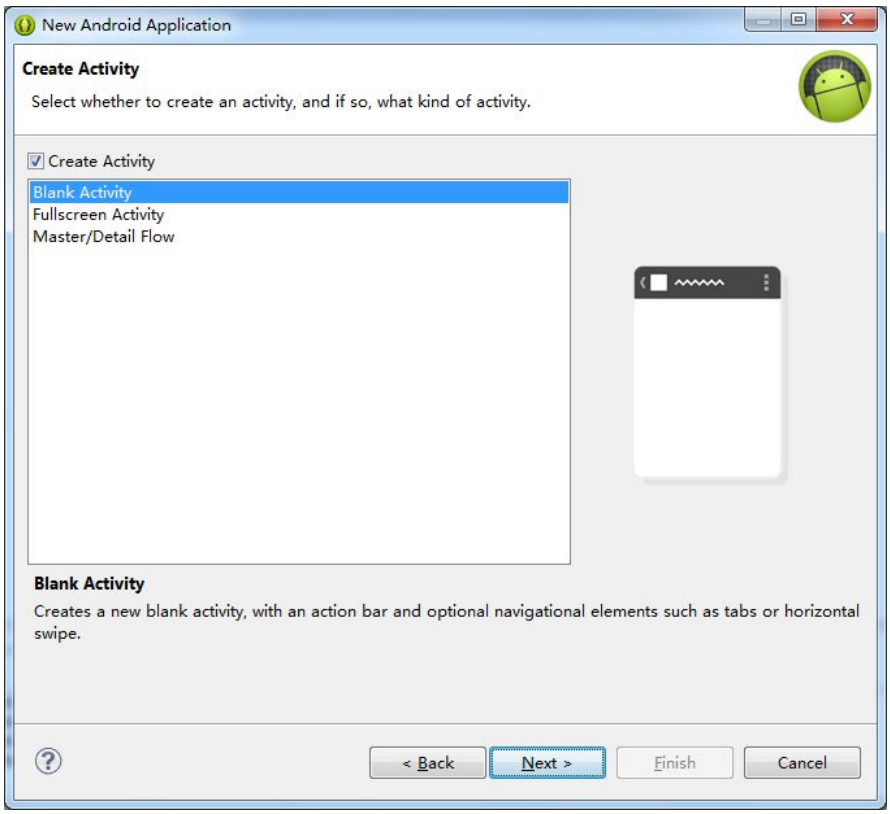

图　1.12

继续点击 Next 后，我们需要给刚刚选择的 Blank Activity 起一个名字，然后给这个活动的布局也起一个名字。Activity Name 就填入 HelloWorldActivity，Layout Name 就填入 hello_world_layout 吧，如图 1.13 所示。

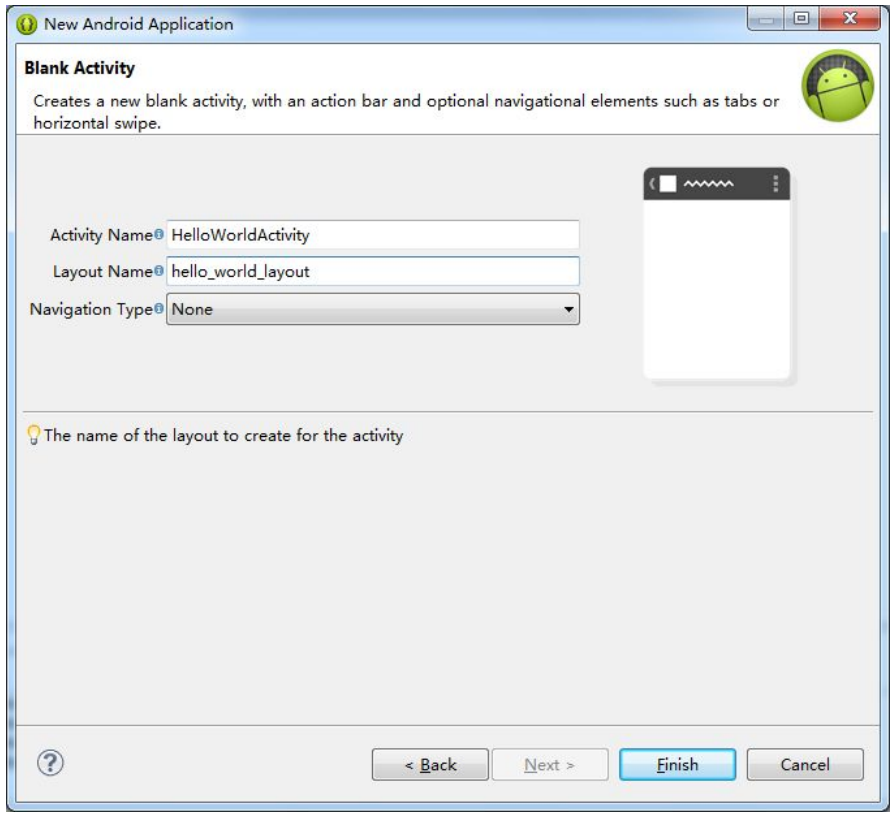

图 1.13

然后点击 Finish，项目终于创建完成了！

1.3.2 运行 HelloWorld

这个时候你的 Eclipse 中应该会显示出刚刚创建的 HelloWorld 项目，由于 ADT 已经自动为我们生成了很多东西，你现在不需要写任何代码，HelloWorld 项目就已经可以运行了。不过在运行之前，让我们先检查一下刚才的模拟器是不是还在线。

点击 Eclipse 导航栏中的 Window→Open Perspective→DDMS，这时你会进入到 DDMS 的视图中去。DDMS 中提供了很多我们开发 Android 程序时需要用到的工具，不过目前你只需要关注 Devices 窗口中有没有 Online 的设备就行了。如果你的 Devices 窗口中有一个设备显示是 Online 的，那就说明目前一切正常，你的模拟器是在线的。如果 Devices 窗口中没有设备，可能是你已经把模拟器关掉了，没关系，按照前面的步骤重新打开一次就行了。如果你的 Devices 窗口中虽然有设备，但是显示 Offline，说明你的模拟器掉线了，这种情况概率不高，但是如果出现了，你只需要点击 Reset adb 就好了，如图 1.14 所示。

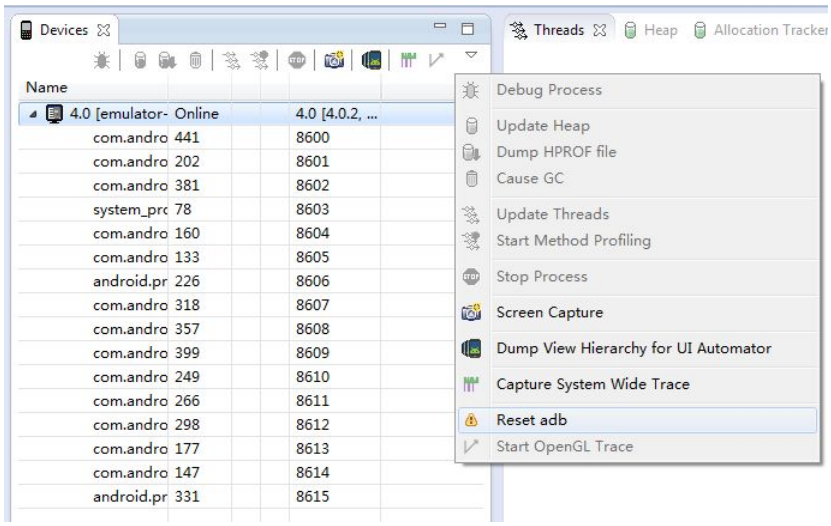

图　1.14

好了，确认完模拟器在线后，点击 Eclipse 工具栏右侧的 Java 选项，回到之前的视图，然后我们来运行一下项目吧。右击 HelloWorld 项目→Run As→Android Application。等待大约几秒钟的时间，你的项目就会运行起来了。现在快去看看你的模拟器吧，结果应该和图 1.15 中显示的是一样的。

图　1.15

HelloWorld 项目运行成功！并且你会发现，你的模拟器上已经安装上 Hello World 这个应用了。打开启动器列表，如图 1.16 所示。

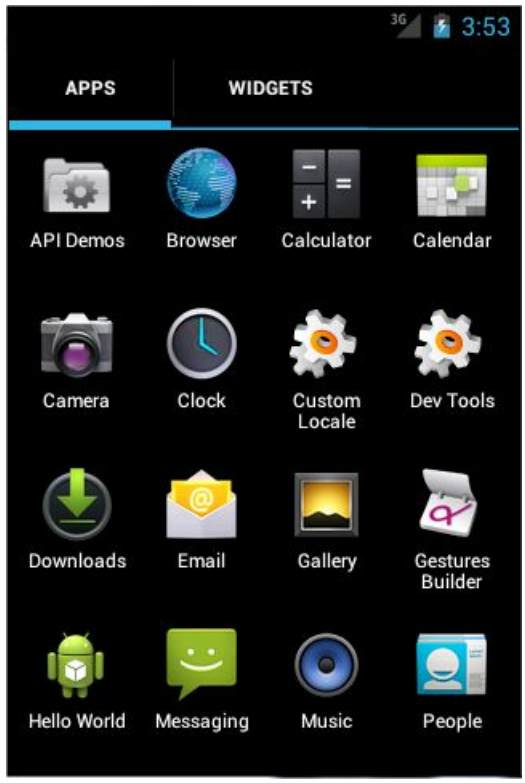

图　1.16

这个时候你可能会说我坑你了，说好的第一行代码呢？怎么一行还没写，项目就已经运行起来了？这个只能说是因为 ADT 太智能了，已经帮我们把一些简单内容都自动生成了。你也别心急，后面写代码的机会多着呢，我们先来分析一下 HelloWorld 这个项目吧。

1.3.3　分析你的第一个 Android 程序

还是回到 Eclipse 中，首先展开 HelloWorld 项目，你会看到如图 1.17 所示的目录结构。

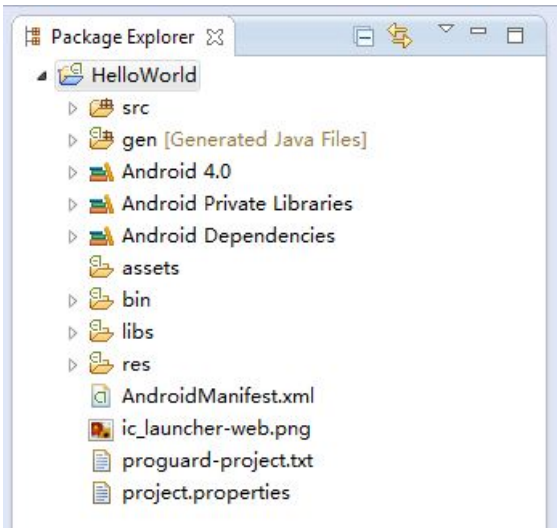

图　1.17

一开始看到这么多陌生的东西，你一定会感到有点头晕吧。别担心，我现在就对上图中的内容一一讲解，你很快再看这张图就不会感到那么吃力了。

1. src

毫无疑问，src 目录是放置我们所有 Java 代码的地方，它在这里的含义和普通 Java 项目下的 src 目录是完全一样的，展开之后你将看到我们刚才创建的 HelloWorldActivity 文件就在里面。

2. gen

这个目录里的内容都是自动生成的，主要有一个 R.java 文件，你在项目中添加的任何资源都会在其中生成一个相应的资源 id。这个文件永远不要手动去修改它。

3. assets

这个目录用得不多，主要可以存放一些随程序打包的文件，在你的程序运行时可以动态读取到这些文件的内容。另外，如果你的程序中使用到了 WebView 加载本地网页的功能，所有网页相关的文件也都存放在这个目录下。

4. bin

这个目录你也不需要过多关注，它主要包含了一些在编译时自动产生的文件。其中会有一个你当前项目编译好的安装包，展开 bin 目录你会看到 HelloWorld.apk，把这个文件拷到手机上就可以直接安装了。

5. libs

如果你的项目中使用到了第三方 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里去。你可以展开上图中 Android 4.0、

Android Private Libraries、Android Dependencies 这些库，其中显示的 Jar 包都是已经被添加到构建路径里的。

6. res

这个目录下的内容就有点多了，简单点说，就是你在项目中使用到的所有图片、布局、字符串等资源都要存放在这个目录下，前面提到的 R.java 中的内容也是根据这个目录下的文件自动生成的。当然这个目录下还有很多的子目录，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，所以你不用担心会把整个 res 目录弄得乱糟糟的。

7. AndroidManifest.xml

这是你整个 Android 项目的配置文件，你在程序中定义的所有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，也可以重新指定你创建项目时指定的程序最低兼容版本和目标版本。由于这个文件以后会经常用到，我们用到的时候再做详细说明。

8. project.properties

这个文件非常地简单，就是通过一行代码指定了编译程序时所使用的 SDK 版本。我们的 HelloWorld 项目使用的是 API 14，你也可以在这里改成其他版本试一试。

这样整个项目的目录结构就都介绍完了，如果你还不能完全理解的话也很正常，毕竟里面有太多的东西你都还没接触过。不用担心，这并不会影响到你后面的学习。相反，等你学完整本书后再回来看这个目录结构图时，你会觉得特别地清晰和简单。

接下来我们一起分析一下 HelloWorld 项目究竟是怎么运行起来的吧。首先打开 AndroidManifest.xml 文件，从中可以找到如下代码：

```
<activity
    android:name="com.test.helloworld.HelloWorldActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

这段代码表示对 HelloWorldActivity 这个活动进行注册，没有在 AndroidManifest.xml 里注册的活动是不能使用的。其中 intent-filter 里的两行代码非常重要，<action android:name="android.intent.action.MAIN" />和<category android:name="android.intent.category.LAUNCHER" />表示 HelloWorldActivity 是这个项目的主活动，在手机上点击应用图标，首先启动的就是这个活动。

那 HelloWorldActivity 具体又有什么作用呢？我在介绍 Android 四大组件的时候说过，

活动是 Android 应用程序的门面，凡是在应用中你看得到的东西，都是放在活动中的。因此你在图 1.15 中看到的界面，其实就是 HelloWorldActivity 这个活动。那我们快去看一下它的代码吧，打开 HelloWorldActivity，代码如下所示：

```
public class HelloWorldActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.hello_world_layout);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.hello_world, menu);
        return true;
    }

}
```

首先我们可以看到，HelloWorldActivity 是继承自 Activity 的。Activity 是 Android 系统提供的一个活动基类，我们项目中所有的活动都必须要继承它才能拥有活动的特性。然后可以看到 HelloWorldActivity 中有两个方法，onCreateOptionsMenu()这个方法是用于创建菜单的，我们可以先无视它，主要看下 onCreate()方法。onCreate()方法是一个活动被创建时必定要执行的方法，其中只有两行代码，并且没有 Hello world!的字样。那么图 1.15 中显示的 Hello world!是在哪里定义的呢？

其实 Android 程序的设计讲究逻辑和视图分离，因此是不推荐在活动中直接编写界面的，更加通用的一种做法是，在布局文件中编写界面，然后在活动中引入进来。你可以看到，在 onCreate()方法的第二行调用了 setContentView()方法，就是这个方法给当前的活动引入了一个 hello_world_layout 布局，那 Hello world!一定就是在这里定义的了！我们快打开这个文件看一看。

布局文件都是定义在 res/layout 目录下的，当你展开 layout 目录，你会看到 hello_world_layout.xml 这个文件。打开之后代码如下所示：

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".HelloWorldActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

</RelativeLayout>
```

现在还看不懂？没关系，后面我会对布局进行详细讲解的，你现在只需要看到上面代码中有一个 TextView，这是 Android 系统提供的一个控件，用于在布局中显示文字的。然后你终于在 TextView 中看到了 hello world 的字样，哈哈终于找到了，原来就是通过 android:text="@string/hello_world"这句代码定义的！咦？感觉不对劲啊，好像图 1.15 中显示的是 Hello world!，这感叹号怎么没了，大小写也不太一样。

其实你还是被欺骗了，真正的 Hello world!字符串也不是在布局文件中定义的。Android 不推荐在程序中对字符串进行硬编码，更好的做法一般是把字符串定义在 res/values/strings.xml 里，然后可以在布局文件或代码中引用。那我们现在打开 strings.xml 看一下，里面的内容如下：

```
<resources>
    <string name="app_name">Hello World</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>
</resources>
```

这下没有什么再能逃出你的法眼了，Hello world!字符串就是定义在这个文件里的。并且字符串的定义都是使用键值对的形式，Hello world!值对应了一个叫做 hello_world 的键，因此在 hello_world_layout.xml 布局文件中就是通过引用了 hello_world 这个键，才找到了相应的值。

这个时候我无意中瞄到了这个文件中还有一个叫做 app_name 的键。你猜对了，我们还可以在这里通过修改 app_name 对应的值，来改变此应用程序的名称。那到底是哪里引用了 app_name 这个键呢？打开 AndroidManifest.xml 文件自己找找去吧！

1.3.4 详解项目中的资源

如果你展开 res 目录看一下，其实里面的东西还是挺多的，很容易让人看得眼花缭乱，如图 1.18 所示。

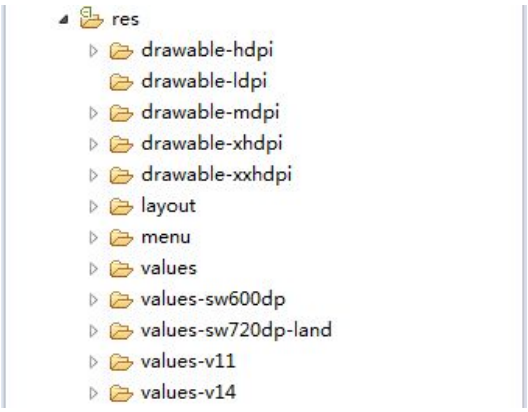

图　1.18

看到这么多的文件夹不用害怕，其实归纳一下，res 目录就变得非常简单了。所有以 drawable 开头的文件夹都是用来放图片的，所有以 values 开头的文件夹都是用来放字符串的，layout 文件夹是用来放布局文件的，menu 文件夹是用来放菜单文件的。怎么样，是不是突然感觉清晰了很多？之所以有这么多 drawable 开头的文件夹，其实主要是为了让程序能够兼容更多的设备。在制作程序的时候最好能够给同一张图片提供几个不同分辨率的副本，分别放在这些文件夹下，然后当程序运行的时候会自动根据当前运行设备分辨率的高低选择加载哪个文件夹下的图片。当然这只是理想情况，更多的时候美工只会提供给我们一份图片，这时你就把所有图片都放在 drawable-hdpi 文件夹下就好了。

知道了 res 目录下每个文件夹的含义，我们再来看一下如何去使用这些资源吧。比如刚刚在 strings.xml 中找到的 Hello world!字符串，我们有两种方式可以引用它：

1. 在代码中通过 R.string.hello_world 可以获得该字符串的引用；
2. 在 XML 中通过@string/hello_world 可以获得该字符串的引用。

基本的语法就是上面两种方式，其中 string 部分是可以替换的，如果是引用的图片资源就可以替换成 drawable，如果是引用的布局文件就可以替换成 layout，以此类推。这里就不再给出具体的例子了，因为后面你会在项目中大量地使用到各种资源，到时候例子多得是呢。另外跟你小透漏一下，HelloWorld 项目的图标就是在 AndroidManifest.xml 中通过 android:icon="@drawable/ic_launcher"来指定的，ic_launcher 这张图片就在 drawable 文件夹下，如果想要修改项目的图标应该知道怎么办了吧？

经验值：+200　　目前经验值：305

级别：萌级小菜鸟

赢得宝物：战胜资深 HelloWorld 程序撰写者（外围守卫者）。拾取守卫者掉落的宝物，大容量移动电源一个、修罗界移动开发者大会纪念品双肩包一个（印有“Android 开发小能

手”字样）、八成新棉麻混纺 Android 战袍一套、微型信心增强大力丸 3 颗。换上新战袍，服下 3 颗大力丸，将其余物资放入双肩包。旁边有一只神秘的松鼠在对我点头。微微向它颔首致意。继续前进。

1.4 前行必备，掌握日志工具的使用

通过上一节的学习，你已经成功创建了你的第一个 Android 程序，并且对 Android 项目的目录结构和运行流程都有了一定的了解。现在本应该是你继续前行的时候，不过我想在这里给你穿插一点内容，讲解一下 Android 中日志工具的使用方法，这对你以后的 Android 开发之旅会有极大的帮助。

1.4.1 添加 LogCat 到你的 Eclipse

日志在任何项目的开发过程中都会起到非常重要的作用，在 Android 项目中如果你想要查看日志则必须要使用 LogCat 工具。当你第一次在 Eclipse 中运行 Android 项目的时候，Eclipse 会提醒你一次是否要添加 LogCat 这个工具。如果你现在还没有添加上的话，我这里教你一下如何手动添加 LogCat 到你的 Eclipse 中。

点击 Eclipse 导航栏中的 Window→Show View→Other，会弹出一个 Show View 对话框。你在 Show View 对话框中展开 Android 目录，会看到有一个 LogCat 的子项，如图 1.19 所示。

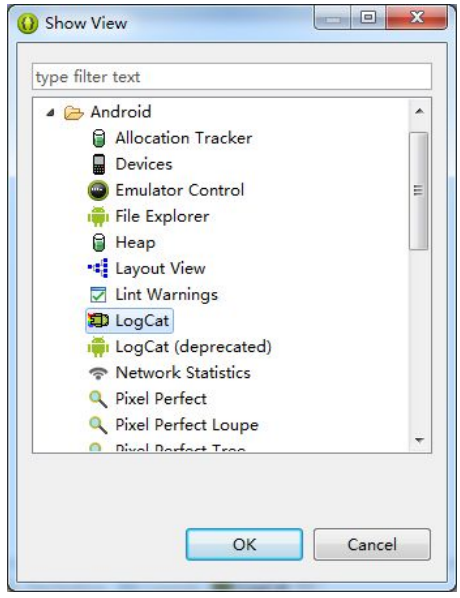

图 1.19

然后选中 LogCat，点击 OK，这样你就成功将 LogCat 添加到 Eclipse 中了。

1.4.2　使用 Android 的日志工具 Log

既然 LogCat 已经添加完成，我们来学习一下如何使用 Android 的日志工具吧。Android 中的日志工具类是 Log（android.util.Log），这个类中提供了如下几个方法来供我们打印日志。

1. Log.v()

   这个方法用于打印那些最为琐碎的，意义最小的日志信息。对应级别 verbose，是 Android 日志里面级别最低的一种。

2. Log.d()

   这个方法用于打印一些调试信息，这些信息对你调试程序和分析问题应该是有帮助的。对应级别 debug，比 verbose 高一级。

3. Log.i()

   这个方法用于打印一些比较重要的数据，这些数据应该是你非常想看到的，可以帮你分析用户行为的那种。对应级别 info，比 debug 高一级。

4. Log.w()

   这个方法用于打印一些警告信息，提示程序在这个地方可能会有潜在的风险，最好去修复一下这些出现警告的地方。对应级别 warn，比 info 高一级。

5. Log.e()

   这个方法用于打印程序中的错误信息，比如程序进入到了 catch 语句当中。当有错误信息打印出来的时候，一般都代表你的程序出现严重问题了，必须尽快修复。对应级别 error，比 warn 高一级。

其实很简单，一共就五个方法，当然每个方法还会有不同的重载，但那对你来说肯定不是什么难理解的地方了。我们现在就在 HelloWorld 项目中试一试日志工具好不好用吧。

打开 HelloWorldActivity，在 onCreate()方法中添加一行打印日志的语句，如下所示：

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.hello_world_layout);
    Log.d("HelloWorldActivity", "onCreate execute");
}
```

Log.d 方法中传入了两个参数，第一个参数是 tag，一般传入当前的类名就好，主要用于对打印信息进行过滤。第二个参数是 msg，即想要打印的具体的内容。

现在可以重新运行一下 HelloWorld 这个项目了，仍然是右击 HelloWorld 项目→Run As →Android Application。等程序运行完毕，可以看到 LogCat 中打印信息如图 1.20 所示。

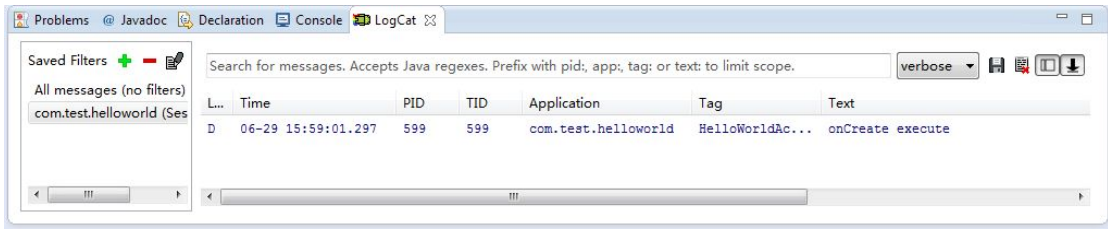

图 1.20

其中你不仅可以看到打印日志的内容和 Tag 名，就连程序的包名、打印的时间以及应用程序的进程号都可以看到。如果你的 LogCat 中并没有打印出任何信息，有可能是因为你当前的设备失去焦点了。这时你只需要进入到 DDMS 视图，在 Devices 窗口中点击一下你当前的设备，打印信息就会出来了。

另外不知道你有没有注意到，你的第一行代码已经在不知不觉中写出来了，我也总算是交差了。

1.4.3 为什么使用 Log 而不使用 System.out

我相信很多的 Java 新手都非常喜欢使用 System.out.println()方法来打印日志，不知道你是不是也喜欢这么做。不过在真正的项目开发中，是极度不建议使用 System.out.println()方法的！如果你在公司的项目中经常使用这个方法，就很有可能要挨骂了。

为什么 System.out.println()方法会这么遭大家唾弃呢？经过我仔细分析之后，发现这个方法除了使用方便一点之外，其他就一无是处了。方便在哪儿呢？在 Eclipse 中你只需要输入 syso，然后按下代码提示键，这个方法就会自动出来了，相信这也是很多 Java 新手对它钟情的原因。那缺点又在哪儿了呢？这个就太多了，比如日志打印不可控制、打印时间无法确定、不能添加过滤器、日志没有级别区分……

听我说了这些，你可能已经不太想用 System.out.println()方法了，那么 Log 就把上面所说的缺点全部都做好了吗？虽然谈不上全部，但我觉得 Log 已经做得相当不错了。我现在就来带你看看 Log 和 LogCat 配合的强大之处。

首先在 LogCat 中是可以很轻松地添加过滤器的，你可以在图 1.21 中看到我们目前所有的过滤器。

图　1.21

目前只有两个过滤器，All messages 过滤器也就相当于没有过滤器，会把所有的日志都显示出来。com.test.helloworld 过滤器是我们运行 HelloWorld 项目时自动创建的，点击这个过滤器就可以只看到 HelloWorld 程序中打印的日志。那可不可以自定义过滤器呢？当前可以，我们现在就来添加一个过滤器试试。

点击图 1.21 中的加号，会弹出一个过滤器配置界面。我们给过滤器起名叫 data，并且让它对名为 data 的 Tag 进行过滤，如图 1.22 所示。

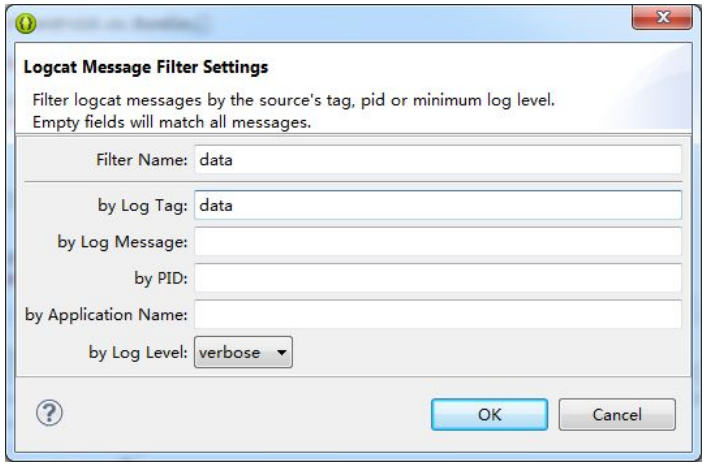

图　1.22

点击 OK，你就会发现你已经多出了一个 data 过滤器，当你点击这个过滤器的时候，你会发现刚才在 onCreate()方法里打印的日志没了，这是因为 data 这个过滤器只会显示 Tag 名称为 data 的日志。你可以尝试在 onCreate()方法中把打印日志的语句改成 Log.d("data",

"onCreate execute")，然后再次运行程序，你就会在 data 过滤器下看到这行日志了。

不知道你有没有体会到使用过滤器的好处，可能现在还没有吧。不过当你的程序打印出成百上千行日志的时候，你就会迫切地需要过滤器了。

看完了过滤器，再来看一下 LogCat 中的日志级别控制吧。LogCat 中主要有 5 个级别，分别对应着我在上一节介绍的 5 个方法，如图 1.23 所示。

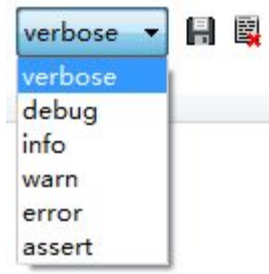

图 1.23

当前我们选中的级别是 verbose，也就是最低等级。这意味着不管我们使用哪一个方法打印日志，这条日志都一定会显示出来。而如果我们将级别选中为 debug，这时只有我们使用 debug 及以上级别方法打印的日志才会显示出来，以此类推。你可以做下试验，如果你把 LogCat 中的级别选中为 info、warn 或者 error 时，我们在 onCreate()方法中打印的语句是不会显示的，因为我们打印日志时使用的是 Log.d()方法。

日志级别控制的好处就是，你可以很快地找到你所关心的那些日志。相信如果让你从上千行日志中查找一条崩溃信息，你一定会抓狂的吧。而现在你只需要将日志级别选中为 error，那些不相干的琐碎信息就不会再干扰你的视线了。

关于 Android 中日志工具的使用我就准备讲到这里，LogCat 中其他的一些使用技巧就要靠你自己去摸索了。今天你已经学到了足够多的东西，我们来总结和梳理一下吧。

1.5 小结与点评

你现在一定会觉得很充实，甚至有点沾沾自喜。确实应该如此，因为你已经成为一名真正的 Android 开发者了。通过本章的学习，你首先对 Android 系统有了更加充足的认识，然后成功将 Android 开发环境搭建了起来，接着创建了你自己的第一个 Android 项目，并对 Android 项目的目录结构和运行流程有了一定的认识，在本章的最后还学习了 Android 日志工具的使用，这难道还不够充实吗？

不过你也别太过于满足，相信你很清楚 Android 开发者和出色的 Android 开发者还是有很大的区别的，你还需要付出更多的努力才行。即使你目前在 Java 领域已经有了不错的成

绩，我也希望在 Android 的世界你可以放下身段，以一只萌级小菜鸟的身份起飞，在后面的旅途中你会不断地成长。

现在你可以非常安心地休息一段时间，因为今天你已经做得非常不错了。储备好能量，准备进入到下一章的旅程当中。

经验值：+200　　升级！（由萌级小菜鸟升级至小菜鸟）　　目前经验值：505

级别：小菜鸟

捡到宝物：在一棵粗大的二叉树下露营时，在钉帐篷时，发现地下埋藏的一本上古时期的算法孤本《算法本源》，内容艰深，眼下还读不懂。作者署名是 TC。作者介绍中提到其在神界的职位是一位乡村教师，喜欢在河边教小天使们唱歌，尽管他声称研究算法只是他的业余爱好，但细心的朋友会发现，他养得最多的植物是瑞亚树（一种以时光女神瑞亚的名字命名的二叉橡皮树）。此人在人界也有兼职，但字迹模糊，已无法辨认。书的前言中还提到，当阅读者的编程级别提升至某个层次时，将更容易看懂这本书，但具体是什么级别，书中没有说，只说“造化弄人，因人而异”。至于为什么一本上古的书会埋得这么浅，不得而知。装好书。继续前进。希望有一天能读懂它。