2

Android 开发快速入门

学习目标：

对 Android 快速入门，能够开发运行简单 Android 应用，为以后深入学习打下坚实基础。

【知识目标】

- 理解 Android 相关的基本概念
- 熟练搭建 Android 开发运行环境
- 编写一个 Android 应用程序
- 了解 Android 应用四个主要组件

【技能目标】

- 能熟练搭建 Android 开发环境
- 编写运行一个 Android 应用

Android 是一个开放的手机操作系统平台，自 2005 年 8 月由 Google 收购注资至今，已经成为成熟的手机操作系统之一。为移动设备提供了一个包含操作系统、中间件及应用程序的软件叠层架构。2012 年的全球 Android 开发者数量已达 100 万，将来还会有越来越多的开发者加入进来。本章主要讲解如何配置 Android 开发环境，首先介绍 Android 开发所需要的开发包和工具，以及获得它们的方式；其次介绍如何正确安装和配置这些开发包；最后为了测试开发环境，创建一个 Android 项目——Hello Android，并在模拟器上运行和调试该程序。此外还简单介绍开发过程中可能会使用到的各种工具，来帮助大家进一步了解 Android。

2.1 开发环境的搭建

在开始 Android 开发之旅启动之前，首先要搭建环境，Android 开发环境的安装和配置是开发 Android 应用程序的第一步，也是深入 Android 平台的一个非常好的入口。

2.1.1 开发准备工作

配置 Android 开发环境之前，首先需要了解 Android 开发对操作系统的要求，Windows、Mac OS、Linux 操作系统上都可以搭建 Android 开发环境，本书使用 Windows 7 为例进行讲解。先前配置开发环境需下载的开发工具较多，配置也较为繁琐。Google 为使搭建 Android 开发环境变得更简单快捷，把 Eclipse、Android SDK、Android ADT 这三种工具整合为 Android Developer Tools（以下简称 ADT），并把它提供给开发者下载。Android 开发所需的软件版本及其下载地址见表 2-1。

表 2-1 Android 开发工具的版本及下载地址

软件名称	所用版本	下载地址
JDK	1.7	http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html
adt-bundle-windows-x86-20130514	20140514	本教材配套光盘中提供

2.1.2 开发工具的安装和使用

1. 安装 JDK

到上一节提供的下载地址将 JDK 进行下载后，开始安装。双击安装文件“jdk-7u75-windows-i586”开始安装。按照提示完成安装即可，安装后生成一个“jdk1.7”的文件夹，如图 2-1 所示。文件夹中的内容如图 2-2 所示。

图 2-1 JDK 安装文件及安装后文件夹

2. 配置 JDK 环境变量

（1）右键单击“计算机”，选择“属性”，在弹出“计算机基本信息”对话框的左侧，单击“高级系统设置”，如图 2-3 所示。

（2）弹出“系统属性”对话框，如图 2-4 所示。单击“高级”选项卡中的 环境变量(N)... 按钮，出现“环境变量”设置界面，如图 2-5 所示。

图 2-2　JDK 文件夹

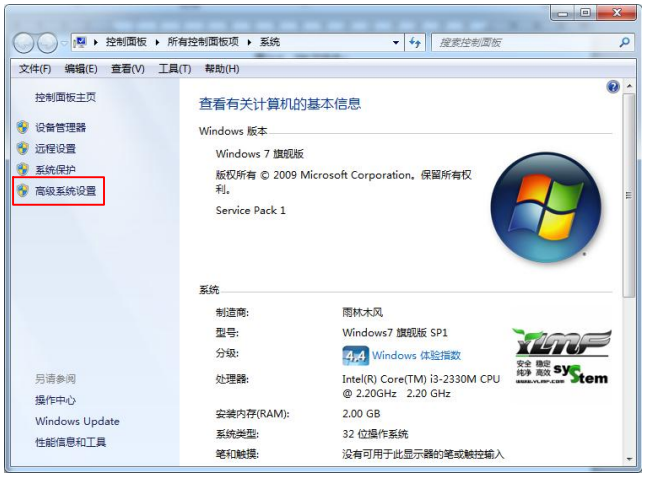

图 2-3　“计算机基本信息”对话框

图 2-4　“系统属性”对话框

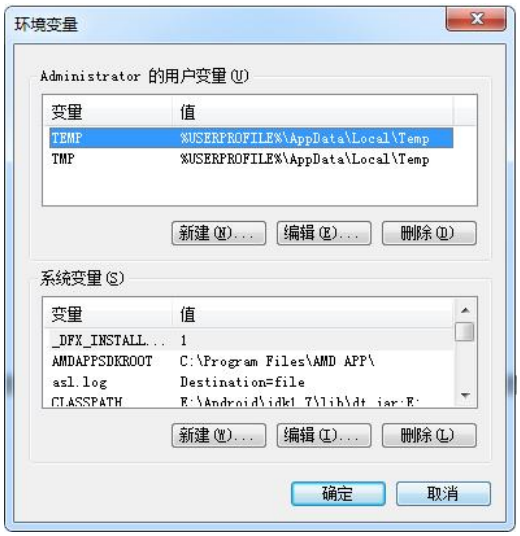

图 2-5 “环境变量”对话框

（3）配置 JAVA_HOME 变量。在下面的“系统变量”处选择新建，在变量名处输入“JAVA_HOME”，变量值处输入之前 JDK 的安装目录，如“E:\Android\jdk1.7”，如图 2-6 所示。

图 2-6 “新建 JAVA_HOME 变量”对话框

配置 JAVA_HOME 变量的目的是得到 JDK 的地址引用，避免每次都要输入很长的地址路径。

（4）配置 Path 路径。在“系统变量”列表中选中 Path 变量，单击编辑按钮，在原有 Path 路径值的末尾添加上一个分号以及安装 JDK 的 bin 文件夹目录，如图 2-7 所示。

图 2-7 “编辑 Path 变量”对话框

（5）配置 CLASSPATH 变量。在下面的“系统变量”处选择新建，在“变量名”处输入“JAVA_HOME”，在“变量值”处输入“.;%JAVA_HOME%/lib/rt.jar;%JAVA_HOME%/ lib/tools.jar”，如图 2-8 所示。注意：变量值前面的“.;”表示当前路径。

图 2-8 “新建 CLASSPATH 变量”对话框

配置 CLASSPATH 变量的目的是告诉 Java 编译到哪里发现标准类库。标准类库是已经完成好的可供编程人员利用的文件，以.jar 作为文件后缀名。

（6）最后在 DOS 命令状态下输入“java -version”，查看 JDK 的版本信息，如果显示如图 2-9 中所提示的信息，则说明 JDK 安装成功。

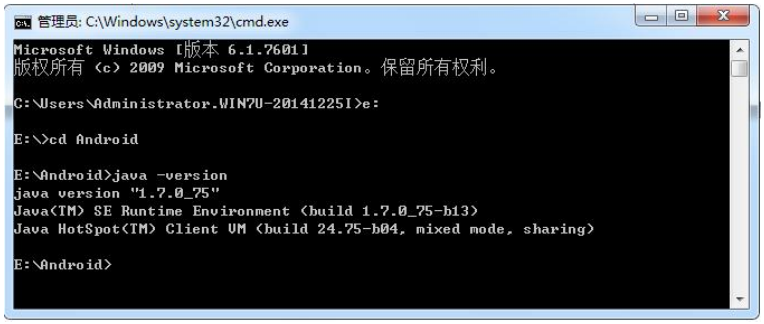

图 2-9 “测试 JDK”对话框

3. ADT 的安装

ADT 工具无需执行安装，在本教材提供的光盘素材中找到“adt-bundle-windows-x86-20130514”压缩文件，对其进行解压后就可以用，不过一定要安装好 JDK。

解压完成后得到“eclipse”“sdk”文件夹和“SDK Manager.exe”可执行文件，如图 2-10 所示。

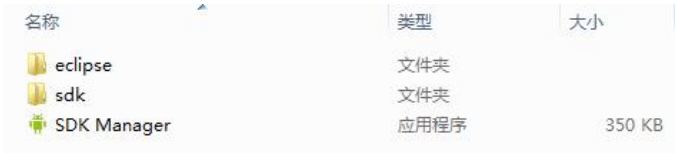

图 2-10 ADT 解压后文件夹

4. 启动 ADT 主界面

（1）打开“eclipse”文件夹，找到“eclipse.exe”可执行文件，双击打开，启动界面如图 2-11 所示。

图 2-11　Eclipse 启动界面

（2）当启动 ADT 时，会弹出如图 2-12 所示的对话框，在此单击 Browse... 可以设置项目的保存路径。下次打开 ADT 如不需再次弹出此对话框，可选中“Use this as the default and do not ask again”前的复选框。

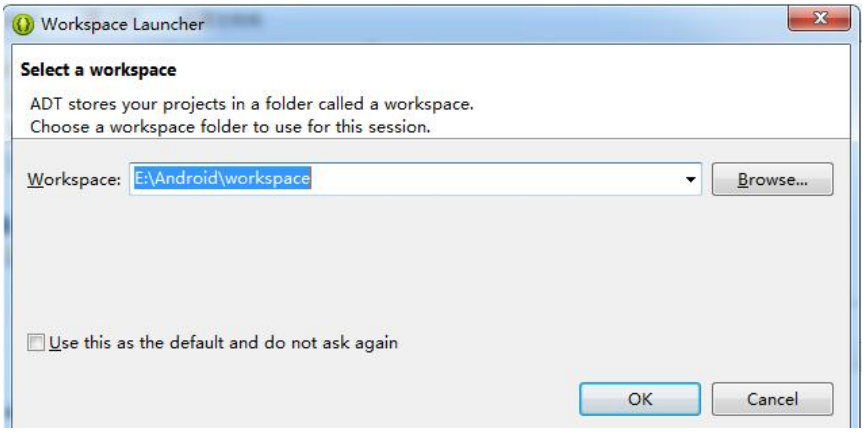

图 2-12　选择 Eclipse 的工作空间

（3）单击 OK 按钮，进入 ADT 的欢迎界面，如图 2-13 所示。

（4）单击“Android IDE”选项卡右边的“×”，关闭欢迎界面，将启动 ADT 的开发环境的主界面，如图 2-14 所示。

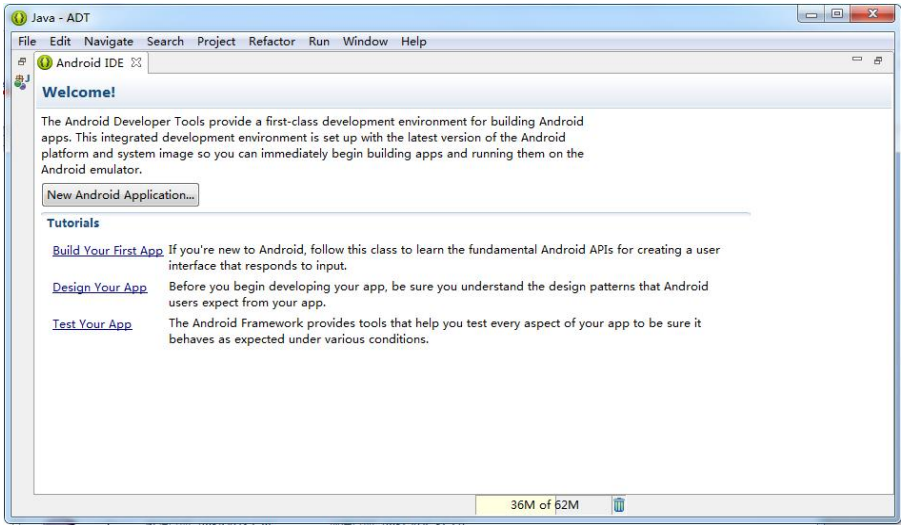

图 2-13　ADT 欢迎界面

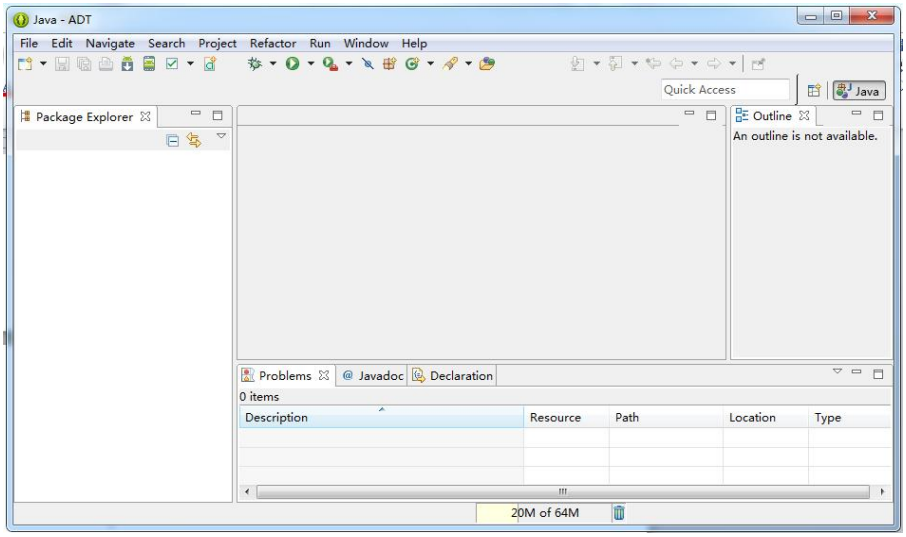

图 2-14　ADT 开发主界面

2.2　创建 AVD

当完成 Android 开发环境搭建后，就可以进行 Android 程序开发。在编写 Android 应用程序时，需要不断测试程序运行结果，为了方便设计人员测试开发程序，Android SDK 提供了 Android 虚拟设备模拟器（Android Virtual Devices），即 AVD，主要目的是方便程序开发人员

模拟在真实环境中运行程序一样。为使 Android 应用程序可以在模拟器上运行，在编写程序之前，必须先创建 AVD。

2.2.1 AVD 的操作简介

1. 利用 Eclipse 创建并启动 AVD

（1）启动 Eclipse，单击“Windows”，在下拉菜单中选择“Android Virtual Device Manager”，如图 2-15、图 2-16 所示。

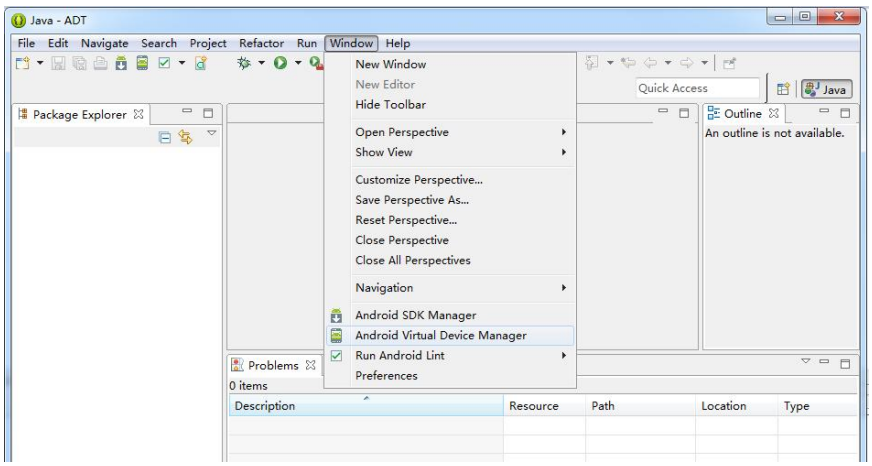

图 2-15　创建 AVD 的 Windows 下拉菜单

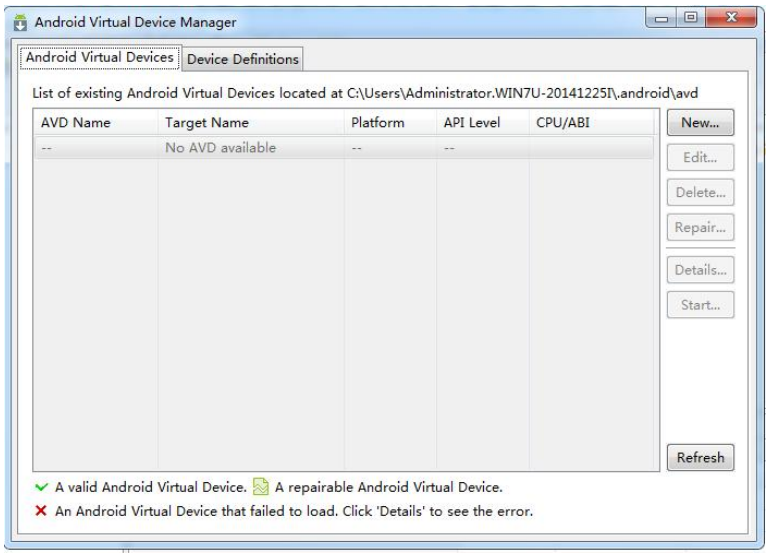

图 2-16　创建 AVD 界面

（2）在默认“Android Virtual Device”选项卡右侧单击 New... 按钮，创建模拟器，如图 2-17 所示。

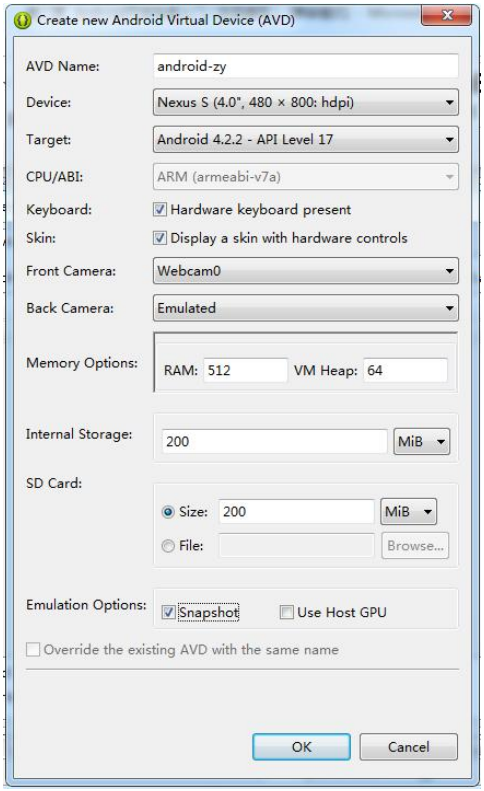

图 2-17　创建 AVD 参数设置

AVD 参数说明：

- AVD Name：即模拟器名称，此项为必填项，支持大小写英文字母、数字、下划线，不能与之前新建的 AVD 名称相同。
- Device：选择适合自己的屏幕大小、分辨率。
- Target：模拟器的 Android 系统版本，此项需要在 SDK Manager 中下载对应系统版本的平台才可以选。
- CPU/ABI：此项是根据下载的系统镜像文件来的，一般是 arm。
- Keyboard：直接勾选即可。
- Skin：直接勾选即可。
- Front/Back Camera：选择前后摄像头设备，可以任意选择，也可以不选。
- Memory Options（存储选项）：即模拟器的运行内存大小，类似电脑内存，可在设置→应用程序中，查看正在运行标签页下显示的具体值。

- Internal Storage（内部存储）：即手机自带存储大小，是模拟器内置存储空间大小，用于存放安装程序和数据的，可在设置→应用程序中，查看其他标签页下显示的具体值。“VM heap”是设置 VM 缓存堆栈的大小，一般使用默认值就可以。
- SD Card：即 SD 存储卡大小，可以选择右侧的下拉选项以改变数值的存储单位，还可以从已有的文件中选择 SD 卡。
- Emulation Options：其他选项可以保持默认，勾选“Snapshot”表示开启快照功能，勾选“Use Host GPU”即表示使用主机的 GPU。

（3）单击 OK 按钮即可完成创建 AVD，在弹出的窗口中就会显示刚刚新建的模拟器，如图 2-18 所示。

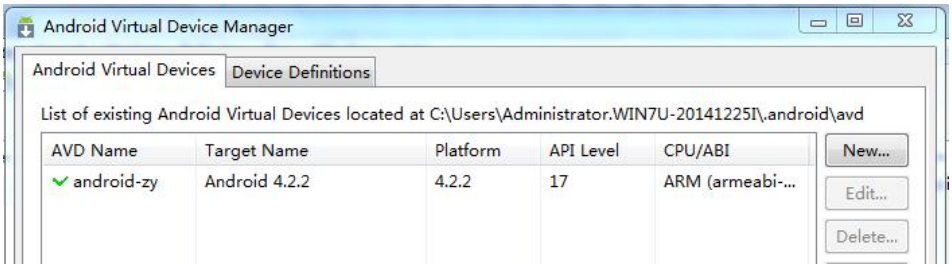

图 2-18　显示新建 AVD

（4）选中该模拟器，单击右侧的 Start... 按钮，可以启动模拟器，如图 2-19 所示，接下来就可以进行 Android 程序开发了。

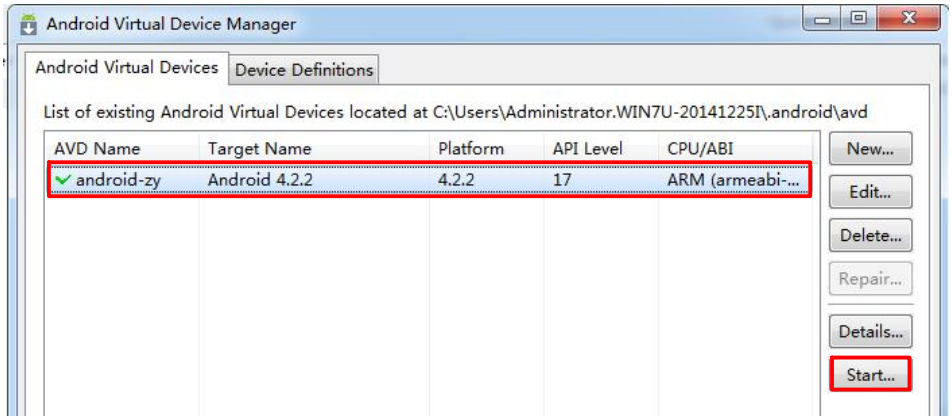

图 2-19　启动 AVD

2. AVD 主界面基本操作

（1）显示主界面。AVD 启动后，会自动显示其主界面，如图 2-20 所示。

（2）查看应用程序。单击屏幕下方的，可显示当前所有的应用程序，如图 2-21 所示。

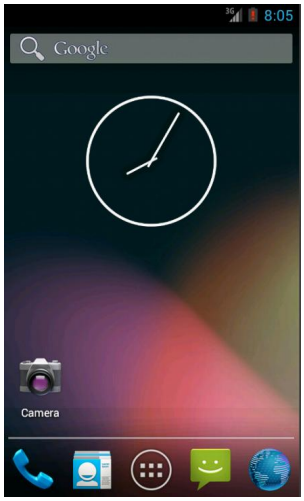

图 2-20　AVD 主界面

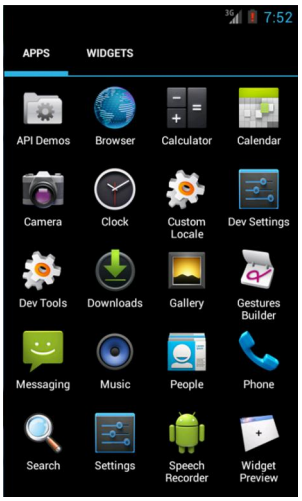

图 2-21　AVD 应用程序界面

（3）查看菜单。单击如图 2-21 所示界面右侧的图标，可显示菜单项 Manage apps 和 System settings，分别选择菜单，可进入对应界面，如图 2-22、图 2-23 所示，进行各种管理与设置。

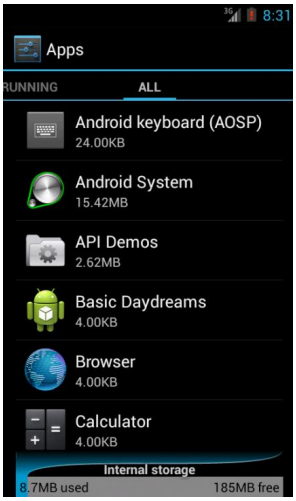

图 2-22　管理应用界面

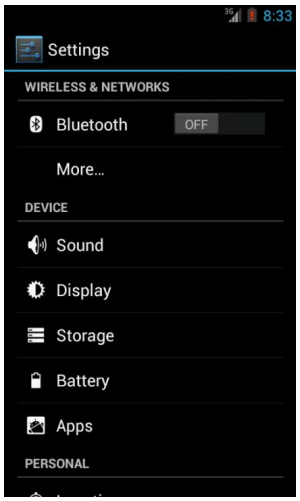

图 2-23　系统设置界面

2.2.2　adb shell 命令的使用

1. adb 介绍

SDK 的 platform-tools 文件夹下包含着 Android 模拟器操作的重要命令 adb，adb 的全称为 Android Debug Bridge，也就是调试桥的意思。通过 adb 可以在 Eclipse 中用 DDMS 来调试 Android 程序。借助这个工具，可以管理设备或手机模拟器的状态。还可以进行以下的操作：

- 快速更新设备或手机模拟器中的代码，如应用或 Android 系统升级；
- 在设备上运行 shell 命令；
- 管理设备或手机模拟器上的预定端口；
- 在设备或手机模拟器上复制或粘贴文件。

2. adb 常用命令

（1）命令格式：android list targets。

含义：显示系统中全部 Android 平台。

（2）命令格式：android list avd。

含义：显示系统中全部 AVD。

（3）命令格式：android create avd --name 名称 --target 平台编号。

含义：创建 AVD。

（4）命令格式：emulator -avd 名称 -sdcard ~/名称.img (-skin 1280x800)。

含义：启动 AVD。

（5）命令格式：android delete avd --name 名称。

含义：删除 AVD。

（6）命令格式：adb devices。

含义：显示当前运行的全部模拟器。

（7）命令格式：adb -s 模拟器编号 命令。

含义：对某一模拟器执行命令。

（8）命令格式：adb install -r 应用程序.apk。

含义：安装应用程序。

（9）命令格式：adb help。

含义：查看 adb 命令帮助信息。

2.3 第一个 Android 程序

ADT 提供了简单的生成 Android 应用框架的功能，本节使用 ADT 通过 Eclipse 创建第一个 Android 项目。在 Eclipse IDE 开发环境中建立一个 Android 应用程序之前，首先要创建一个 Android 项目工程，并且建立一个启动配置，建立项目工程的目的就是为开发的应用程序搭建好运行环境需要的支持。然后才可以编写、调试和运行应用程序。

2.3.1 创建 Android 项目

创建 Android 项目的步骤如下：

（1）启动 Eclipse 开发工具，新建一个项目，在弹出的“New Project”对话框的列表中展开“Android”项，然后选择“Android Application Project”子项，如图 2-24 所示。

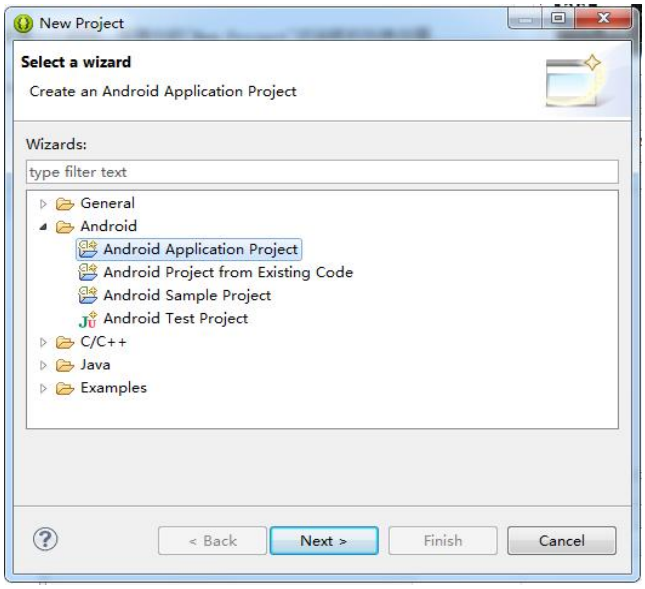

图 2-24　新建一个 Android 工程

（2）单击“Next”按钮，在“Application name”文本框中输入这个应用程序的名字，如 HelloAndroid，在“Project name”文本框中输入工程名称，如 helloandroid，在“Package name”文本框中输入应用程序包的名字，如 com.app.hello，如图 2-25 所示。

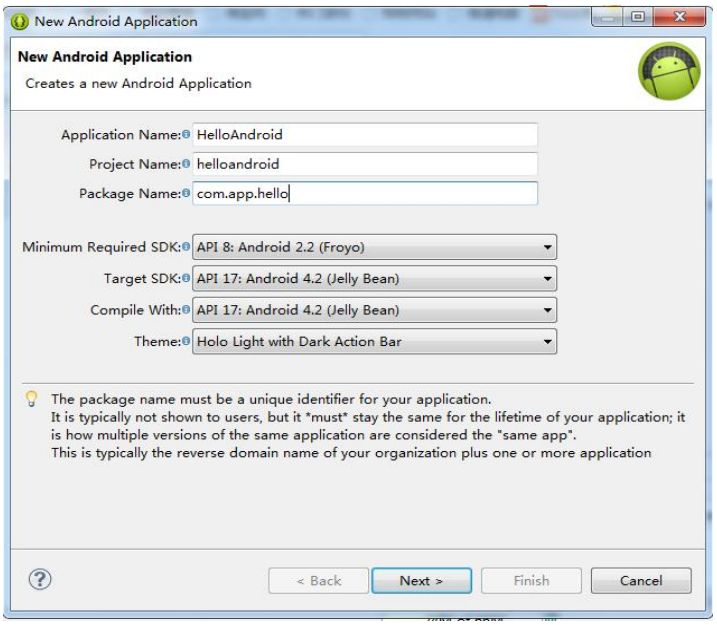

图 2-25　新建 helloandroid 工程

注：Application name 即应用程序呈现给用户的名称。Package name 即 Activity 类的包名称。

（3）连续单击“Next”按钮，对新弹出的对话框均不做改动，选取默认值，最后单击“Finish”按钮，完成 Android 项目的创建，这时 Eclipse 开发平台左边的“Package Explorer”窗口中会显示新创建的项目"helloandroid"，如图 2-26 所示。

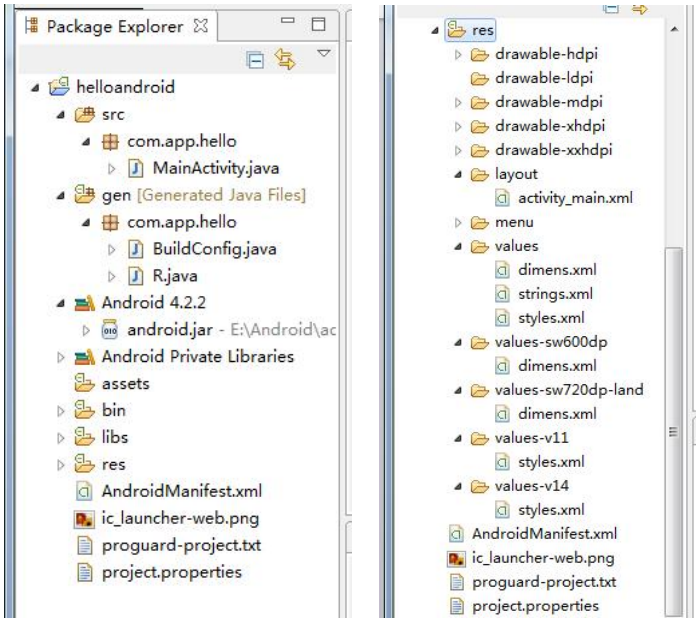

图 2-26　显示项目管理器

此时“helloandroid”项目已经创建好，而且这个项目是由 ADT 插件自动生成，所以不用编写代码，即可运行。

2.3.2　项目框架解析

1. 项目结构

在图 2-26 中，可以看到一个 Android 项目的基本目录结构，下面来讲解一下 Android 项目中常见的目录结构。

src/——源代码存放目录，存放项目中的 Java 源程序文件。用于管理由 ADT 自动生成的 Activity 框架代码以及用户自己创建的代码，允许用户修改的 Java 文件都存放在这里。其中 com.app.hello/为程序包目录，这里根据前面新建项目的时候填写的“Package Name”选项来生成应用程序包的 Namespace 命名空间。

gen/——自动生成目录，存放所有由 Android 开发工具自动生成的文件。目录中最重要的就是 R.java 文件，这个文件由 Android 开发工具自动生成，用于引用资源文件夹中的资源文件。ADT 会自动根据放入 res 目录中的 xml 界面配置文件、图片以及一些文本等资源文件同步更

新修改 R.java 文件，所以不要随意手动编辑该文件。

assets/——资源目录，存放应用程序中用到的较大的文件，如视频文件、MP3 等一些媒体文件。注意：assets 目录下的资源文件不会在 R.java 自动生成 ID，所以读取 assets 目录下的文件必须指定文件的路径。

res/——资源文件夹，存放应用中用到的文字、图片以及布局等资源，包含程序中的所有文件。其中：

- res/drawable 用于存放图片文件资源。hdpi、mdpi、ldpi、xhdpi 和 xxhdpi 分别表示不同分辨率的图片。
- res/layout 用于存放显示界面（布局）的 xml 配置文件。
- res/menu 用于存放菜单设计 xml 文件。
- res/values 用于存放一些常量信息，如：strings.xml 用于定义字符串和数值、array.xml 用于定义数组信息、colors.xml 用于定义颜色、dimens.xml 用于定义尺寸数据、styles.xml 用于定义样式等。

android.jar——Android 程序应用的库文件，Android 支持的 API 都包含在这个文件夹里。

AndroidManifest.xml——项目清单文件，应用程序中的所有功能都在此列出。包括项目中的 Activity、Services、Broadcast Receiver 等信息，以及一些用户权限信息等。它是每个 Android 项目都必需的基础配置文件。

project.properties——项目环境信息，存放了项目的环境配置信息，一般不用编辑。

2. 几个重要的项目文件解析

（1）Java 源代码文件——MainActivity.java。当一个项目被创建后，会在 src 文件夹中生成一个源代码文件 MainActivity.java，其代码如下：

```
1   package com.app.hello;
2   Import android.os.Bundle;
3   Import android.app.Activity;
4   Import android.view.Menu;
5   Public class mainactivity extends Activity {
6       @Override
7       Protected void oncreate(Bundle savedinstancestate) {
8           Super.oncreate(savedinstancestate);
9           Setcontentview(R.layout.activity_main);
10      }
11      @Override
12      Public boolean oncreateoptionsmenu(Menu menu) {
13          Getmenuinflater().inflate(R.menu.main, menu);
14          Return true;
15      }
16  }
```

MainActivity 类继承自 Activity，Android 中所有的用户界面展示的类都直接或间接继承自 Activity。onCreate()是一个重要的方法，在这里，它重载 Activity 类中的 onCreate()方法，每个

Activity 类的子类都要重载该方法来初始化界面。R.layout.activity_main 是一个资源的常量，这个资源是对 activity_main.xml 的一个间接引用，当程序启动时将 activity_main.xml 文件中的内容展示给用户。

（2）Java 源代码文件——R.java。Android 应用程序 gen 目录下保存的是项目的所有包及源文件（.java），gen 目录下包含了项目中的所有资源。.java 文件是在建立项目时自动生成的，为只读模式，不能更改。R.java 文件是定义项目所有资源的索引文件。其部分代码如下：

```
1   package com.app.hello;
2   public final class R {
3       public static final class attr {
4       }
5       public static final class dimen {
6           public static final int activity_horizontal_margin
7   =0x7f040000;
8           public static final int activity_vertical_margin
9   =0x7f040001;
10      }
11      public static final class drawable {
12          public static final int ic_launcher=0x7f020000;
13      }
14      public static final class id {
15          public static final int action_settings=0x7f080000;
16      }
17      public static final class layout {
18          public static final int activity_main=0x7f030000;
19      }
20      ...
21      }
22  }
```

以上代码定义了很多常量，它们的常量名都与 res 文件夹中的文件名相对应。有了这个文件，在程序中使用资源将更加方便，能很快地找到要使用的资源。当项目加入了新资源后，只需刷新一下该项目，.java 文件便会自动更新。

（3）资源文件——activity_main.xml。Android 中的 activity_main.xml 文件内容主要是有关用户界面布局和设计的，利用 XML 语言描述用户界面，为应用程序中使用的组件进行定义、设置界面属性值，以及组件使用的资源参考设置等。在 res/layout 目录下，双击 activity_main.xml 文件，其代码如下：

```
1   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2       xmlns:tools="http://schemas.android.com/tools"
3       android:layout_width="match_parent"
4       android:layout_height="match_parent"
5       android:paddingBottom="@dimen/activity_vertical_margin"
6       android:paddingLeft="@dimen/activity_horizontal_margin"
7       android:paddingRight="@dimen/activity_horizontal_margin"
8       android:paddingTop="@dimen/activity_vertical_margin"
```

```
9           tools:context=".MainActivity" >
10          <TextView
11              android:layout_width="wrap_content"
12              android:layout_height="wrap_content"
13              android:text="@string/hello_world" />
14      </RelativeLayout>
```

以上代码中大部分都是为组件设置布局参数，其格式为 android：属性=“属性值”。

（4）资源文件——strings.xml。Android 应用程序的 res 目录下有一个 value 子目录，其中的 strings.xml 文件是用来存放所有文本信息和数值。使用该文件有两个目的：一是为了程序国际化，可以将屏幕中可能出现的文字信息都集中存储在 strings.xml 文件中，当不同国家的用户使用时，只需修改 strings.xml 文件内容，而不用修改主程序。二是为了减少文字的重复使用，当一个提示信息需要在程序中使用多次时，就可以放在 strings.xml 文件中，需要时引用一下就可以。

strings.xml 文件代码如下：

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <resources>
3       <string name="app_name">HelloAndroid</string>
4       <string name="action_settings">Settings</string>
5       <string name="hello_world">Hello world!</string>
6   </resources>
```

代码中每个 string 标签声明了一个字符串，name 指定其引用名。value 文件中还有几个代表不同类别的 xml 文件，如 dimens.xml、styles.xml 它们的根元素都是<resources>，这样才可以识别调用资源。

（5）系统控制文件——AndroidManifest.xml。Android 包含四大组件，分别是：Activity、BroadCast receiver、service、Content Provider，当 Android 启动一个应用程序组件之前，必须知道哪些组件是存在的，所以开发人员在开发过程中，如果用到了这些组件，一定要在 AndroidManifest.xml 文件中声明，否则 Android 应用程序在运行时会报错。

这个文件以 XML 作为结构格式，而且对于所有应用程序，都叫做 AndroidManifest.xml。为声明一个应用程序组件，它还会做很多额外工作，比如指明应用程序所需链接到的库的名称（除了默认的 Android 库之外）以及声明应用程序期望获得的各种权限。当然 manifest 文件的主要功能还是向 Android 声明应用程序的组件。

以下是 AndroidManifest.xml 的代码实例：

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3                                       //定义 Android 的命名空间
4       package="com.app.hello"       //定义应用程序的包名称
5       android:versionCode="1"
6       android:versionName="1.0" >
7       <uses-sdk
8           android:minSdkVersion="8"
```

```
9               android:targetSdkVersion="17" />
10          <application
11              android:allowBackup="true"
12              android:icon="@drawable/ic_launcher"
13              //定义应用程序的图标，资源类型为图像，名称为 ic_launcher
14              android:label="@string/app_name" //定义应用程序的标签
15              android:theme="@style/AppTheme"> //定义应用程序的主题？
16              <activity                   //定义活动的内容
17                  android:name="com.app.hello.MainActivity"
18  //定义活动的名称
19                  android:label="@string/app_name" >
20                                          //定义 Android 应用程序的标签名称
21                  <intent-filter> //描述此 Activity 启动的位置和时间
22                    <actionandroid:name=
23  "android.intent.action.MAIN" />
24                      <category android:name=
25  "android.intent.category.LAUNCHER" />
26                  </intent-filter>
27              </activity>
28          </application>
29  </manifest>
```

2.3.3 运行项目

上面我们已经利用 ADT 插件通过 Eclipse 创建好了第一个 Android 项目，而且没有编写任何代码，下面将其在模拟器上运行。

右击“HelloAndroid”项目名称，在菜单中选择“Run As”项目下的“Android Application”，便可以运行 helloandroid 项目了，不过 Android 模拟器启动非常慢，需要等待一会儿。启动后如图 2-27 所示。

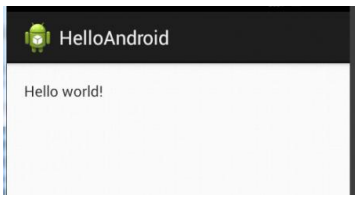

图 2-27 启动模拟器运行 HelloAndroid 项目界面

2.4 DDMS 应用

在完成了第一个“HelloAndroid”项目的创建之后，大家可以体会到在 ADT 开发环境中进行 Android 开发是一件很方便的事。实际上，ADT 还为用户提供了一个非常方便的调试工具，那就是 DDMS。使用这个工具可以将代码调试工作也变简单。DDMS 的全称是 Dalvik Debug Monitor Service，它为用户提供的服务包括：为测试设备截屏，针对特定的进程查看正在运行的线程以及堆信息、Logcat（用于得到程序 log 信息的命令）、广播状态信息、模拟电话呼叫、接收 SMS、虚拟地理坐标等。DDMS 为 Android 集成环境、模拟器及真正的 Android 设备架起了一座桥梁。

单击 Eclipse 界面 Window 下拉菜单中的“Open Perspective”，选择“DDMS”，切换到 DDMS 界面，如图 2-28、图 2-29 所示。单击图 2-29 右上角红色方框中的按钮，也可以切换界面。

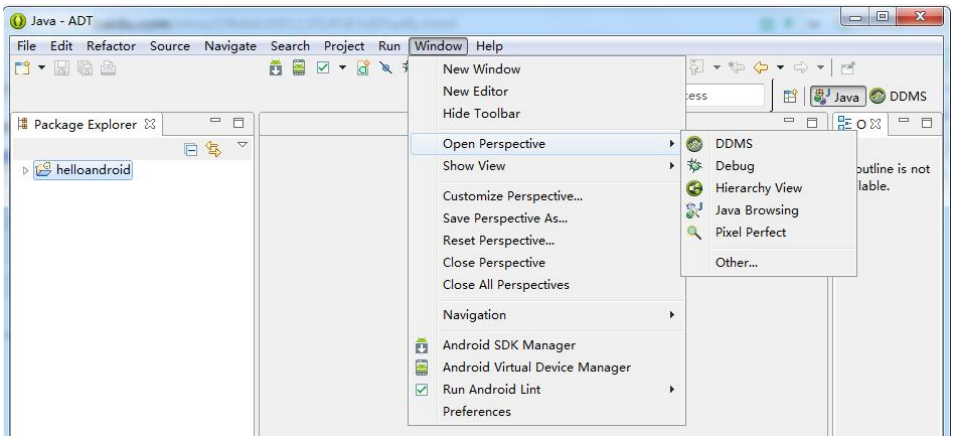

图 2-28　启动 DDMS

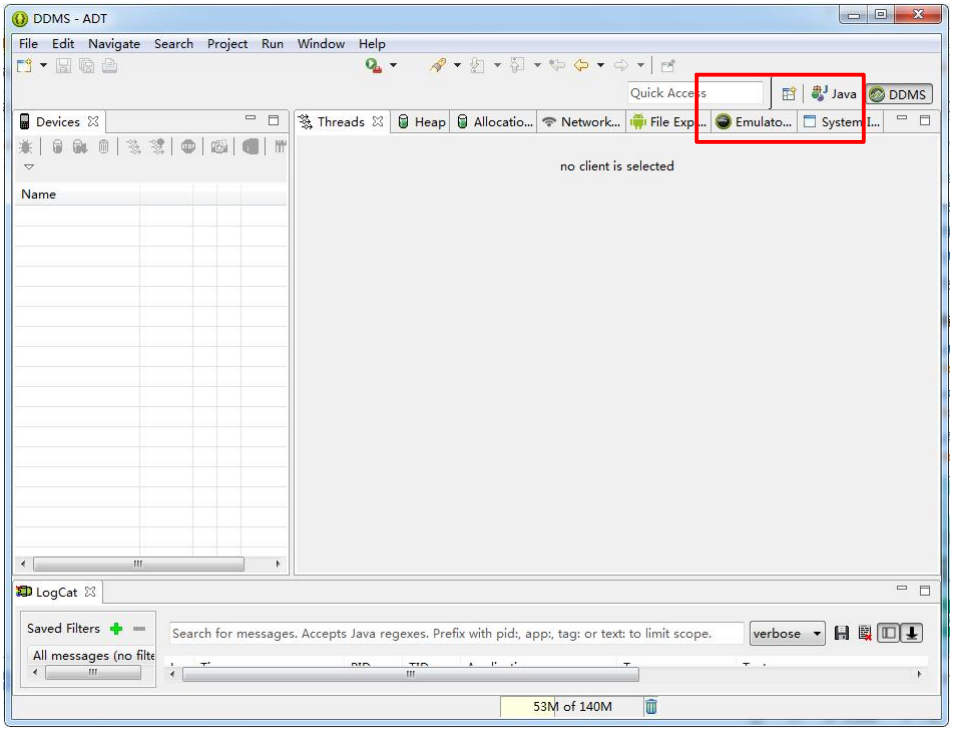

图 2-29　DDMS 界面

DDMS 布局中包含了四个主要部分：

（1）Devices。这个视图列出了当前连接到 Eclipse 的所有目标设备以及运行在目标设备上的进程。如果目标设备为手机模拟器，还会列出模拟器的端口号。Devices 视图的工具栏上还包含了一些按钮。比如屏幕截取按钮，它的外观是一个很小的 Android 手机屏幕，单击这个按钮以后，手机当前的屏幕就会被截取下来，开发者还可以将截取到的屏幕保存成 PNG 格式的图片以备它用。

（2）Threads/ Heap/ File Explorer。这三个视图可以让程序开发人员知道系统内部的运行状况。Threads 视图用来显示当前进程的所有活动线程。首先在 Devices 视图中选择想要查看的进程，然后单击 Devices 工具栏上的 Update Threads 按钮，该进程的所有活动线程信息就会显示在 Threads 视图中。如图 2-30 所示。

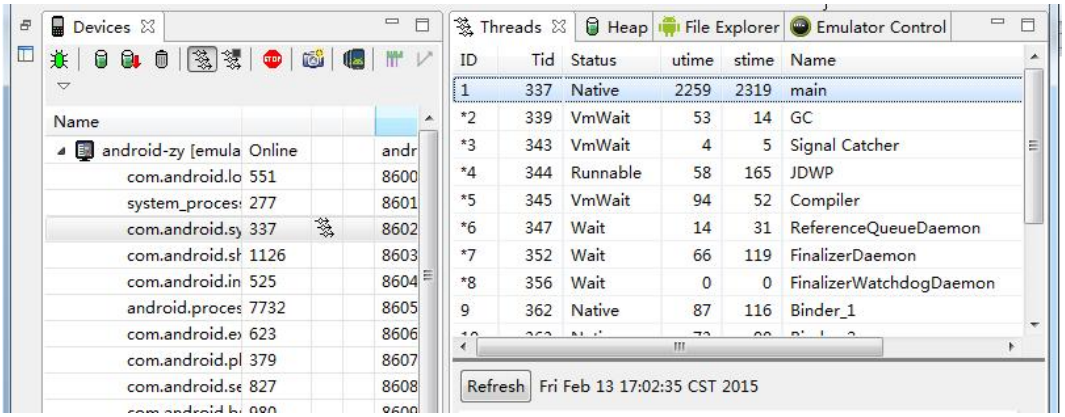

图 2-30　Thread 视图

Heap 视图用来显示 Dalvik 虚拟机中堆内存的使用情况，该视图的内容会在每次执行完垃圾回收操作之后进行更新。想要激活 Heap 视图必须单击 Devices 视图工具栏上的 Update Heap 按钮。系统的垃圾回收操作是不定时的，如果想要强制进行垃圾回收操作可以单击 Heap 视图中的 Cause GC 按钮。

File Explorer 视图是 Android 系统的文件浏览器，在这里可以浏览设备里面的文件目录，比如之前讲 Android 数据存储时提到过可以使用 DDMS 来浏览对应的存储文件，就是这个视图的功能。

（3）Emulator Control。这个视图仅适用于 Android 模拟器，用于模拟电话功能和定位功能：

1）模拟电话功能：用于模拟各种网络状态（离线、服务区内、漫游和搜寻网络等）和宽带下的语音通话、数据服务和短信息服务。这个功能对于测试应用程序的鲁棒性非常有用。

2）模拟来电或 SMS 文本消息：在“来电号码”字段中输入一个数字，然后单击呼叫，模拟

呼叫发送到模拟器或手机。单击挂断键终止通话。在消息字段中输入文字，然后单击发送按钮发送消息。

3）模拟定位功能：用于设定手机模拟器的虚拟位置信息，输入模拟的经纬度。还可以指定一个 GPX 或者 KML 轨迹文件作为输入，用于模拟手机的移动状态，如图 2-31 所示。

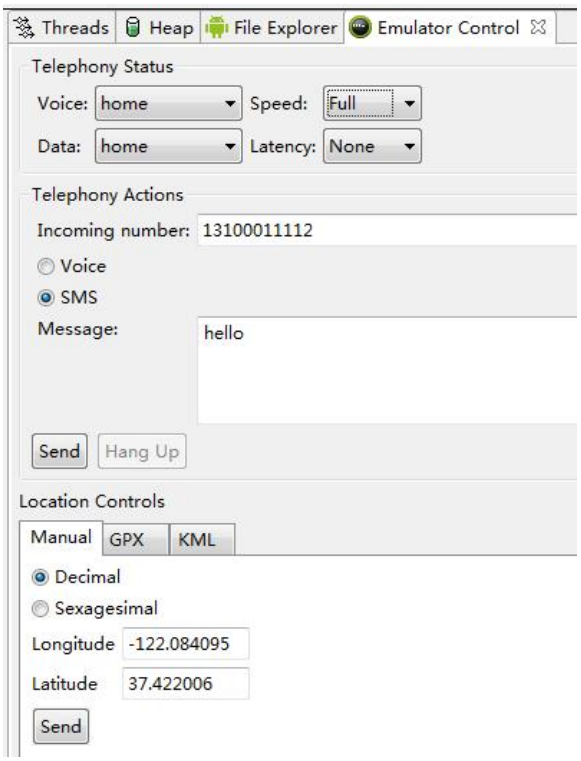

图 2-31 “Emulator Control”视图

（4）LogCat。这个视图用于打印设备的调试信息，在开发过程中最常用。这里的信息分为五级，分别对应上面的 V（VERBOSE）、D（DEBUG）、I（INFO）、W（WARN）、E（ERROR）五个圆形的按钮。此外，还可以通过单击这些按钮来过滤相应的调试信息。

2.5 Activity 介绍

2.5.1 Activity 的简介

Activity（活动）是 Android 组件中最基本也是最常用的一种组件。在 Android 系统中，Activity 是应用程序和用户交互的窗口，一个 Activity 通常就是一个单独的屏幕。Activity 是

用户唯一可以看得到的东西。几乎所有的 Activity 都与用户进行交互，所以 Activity 主要负责的就是创建显示窗口，以便在其中存放各种显示控件，如菜单、文本输入等。Activity 展现在用户面前的经常是全屏窗口，也可以将 Activity 作为浮动窗口来使用，或者嵌入到其他的 Activity 中。一个 Android 程序由多个 Activity 程序组成，第一个呈现给用户的 Activity 称为 main Activity（主活动）。

通常称用户从一个屏幕界面转移到另一个屏幕界面的过程为 Activity 之间的切换。Activity 切换活动有两种类型：“独立”Activity 与“相依”Activity。它们的主要区别为是否与其他 Activity 交换信息。独立的 Activity 只是单纯从一个屏幕界面跳到下个界面，不涉及信息的交换。相依 Activity 又可分为单向与双向：从一个屏幕跳到下一个屏幕时，会把一些参数传给下一个 Activity 使用，这就是单向相依 Activity；要在两个屏幕之间切换，屏幕上的信息会因另一个屏幕的操作而改变的，就是双向相依 Activity。

2.5.2 创建 Activity

Activity 提供了和用户交互的可视化界面。将界面上的内容称为控件或者 View，用户通过对这些控件的操作，实现需求。Activity 提供了很多可直接使用的 View 控件，如：buttons（按钮）、menu items（菜单选项）、text fields（文本输入）、check boxes（选项）等。

1. 创建 Activity 的步骤

（1）一个 Activity 其实就是一个子类，这个类继承于 Activity 或其子类。

（2）覆盖 onCreate()方法，该方法在 Activity 第一次运行时，Activity 框架会调用这个方法。

（3）由于 Activity 是 Android 应用程序的一个组件，所以每一个 Activity 都需要在配置文件 AndroidManifest.xml 中进行配置。

（4）为 Activity 添加必要的控件。在 layout 文件夹中创建一个声明 xml 格式的布局文件，然后再在这个布局文件中对 Activity 的布局以及不同的控件进行设置。

（5）再在第一步定义的 Activity 子类中通过 findViewById(R 中对应的 id 类中控件的 id)方法来获取布局文件中声明的控件，前提是布局文件 R 中必须声明这些控件的 id。

2. 创建实例

（1）右击 com.example.lp 包→【New】，选择新建一个 Class 类文件，如图 2-32 所示。

（2）创建一个名为 firstActivity 的类，单击【Finish】完成创建，如图 2-33 所示。

（3）重写 Activity 的 onCreate()方法，代码如下：

```
1   public class firstActivity extends Activity {
2           @Override
3       protected void onCreate(Bundle savedInstanceState) {
4           super.onCreate(savedInstanceState);//调用了父类的 onCreate()方法
5       }
6   }
```

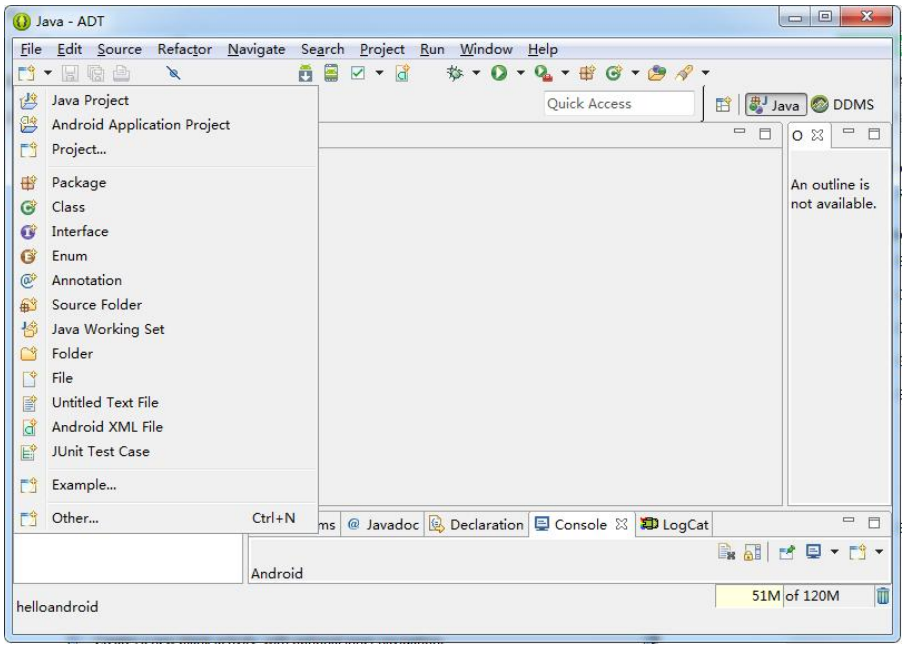

图 2-32　新建一个 Class

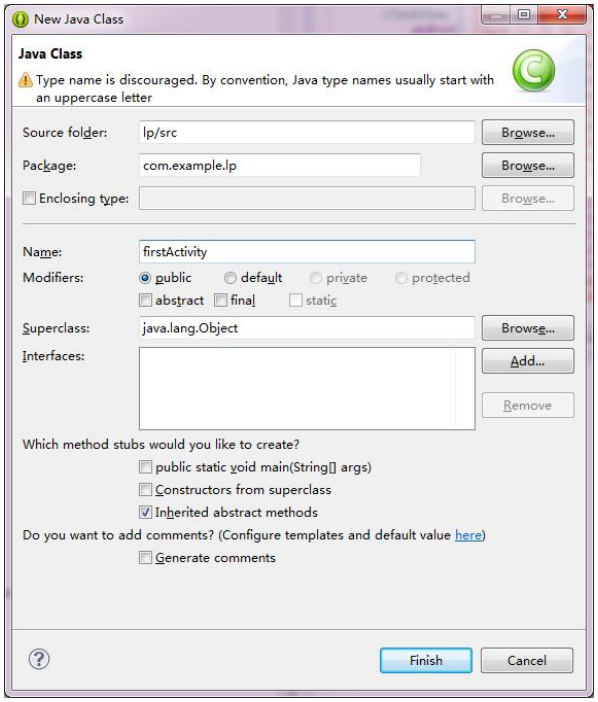

图 2-33　新建一个 firstActivity 类

2.6 本章小结

本章首先讲解了 Android 开发环境的搭建。要进行 Android 程序开发，需要下载安装 JDK 以及 ADT，并完成环境变量的配置即可。然后，介绍了 Android 开发工具的界面及创建启动 AVD 的过程。接着，讲解了在 Android 系统中建立并测试一个程序项目的过程，分析了应用程序项目的架构和重要文件。此外，还简单介绍了 ADT 为用户提供的便捷调试工具 DDMS 的使用方法。最后，介绍了 Android 系统四大组件之一的 Activity 的主要知识和创建方法。通过本章的学习，对 Android 程序设计流程有一个全面的认识。

2.7 本章习题

1．请简要说明 Android 应用中有哪些主要组件，并描述其作用。
2．创建 Android 应用，显示“Android ACTIVITY 组件”文本信息。