第3章 Android应用程序



 \square

[]

掌握使用Android Studio开发Android应用程序的方法

⑦ 了解AndroidManifest.xml文件的用途;

了解Android的程序结构。



- •创建Android工程
 - 启动Android Studio,显示的集成开发环境如下图所示,点击Create New Project,跳转到选择模板页面。



- •创建Android工程
 - •可以根据需要选择创建适用手机或平板电脑的Android工程,同时还提供创建适用于电视、可穿戴设备、车载应用、智能家居的Android工程。为了简化开发工作,选择Empty Activity建立一个空白的Activity即可,如下图所示



- •创建Android工程
 - •填写应用程序名称与工程名称
 - •进入工程配置页面后,需要在 Name栏中输入应用程序名称
 - Package name栏中输入包名称
 - Save location栏中选择项目路径
 - Language单选框选择Java
 - •在Minimum SDK栏中选择能运行的 最低版本的SDK,推荐使用API16, 可以支持目前大部分的设备



- •创建Android工程
 - •填写应用程序名称与工程名称
 - •包名称(Package Name)是包的命名空间,需要遵循Java包的命名方法。
 - •包名称由两个或多个标识符组成,中间用点隔开,例如 edu.hrbeu.HelloAndroid。
 - 使用包主要为了避免命名冲突,因此可以使用反写电子邮件地址的方式 保证命名的唯一性,例如笔者电子邮件地址是 wangxianghui@hrbeu.edu.cn,则可以将包名称命名为 cn.edu.hrbeu.wangxianghui。
 - •为了保证代码的简洁,第一个Android程序的包名称使用 ice.hrbeu.helloandroid。

- •创建Android工程
 - •填写应用程序名称与工程名称
 - SDK最低版本(Minimum SDK)是指的是Android程序能够运行的最低的 API等级
 - •如果手机中的Android系统的API等级低于程序的SDK最低版本,则程序不能够在该Android系统中运行
 - •选择低版本的API可以提高程序的兼容性,但是由于为了兼容低版本API, 在工程中就无法使用新版本API中加入的新功能
 - •笔者一般选择使用API16,可以在兼容绝大部分手机的情况下,使用一些流行的新功能

- •创建Android工程
 - •填写应用程序名称与工程名称
 - API等级是Android系统中用来标识API框架版本的一个整数,用来识别 Android程序的可运行性
 - •如果Android程序标识的API等级高于Android系统所支持的API等级,程序则无法在该Android系统中运行
 - API等级与系统版本之间的对照关系可参考下表(API等级对照表)

- •创建Android工程
 - •填写应用程序名称与工程名称

| 系统版本 | API等级 | 版本代号 | 支持设备类型 |
|---------------|-------|-----------------|----------------------|
| Android 11 | 30 | R | 智能手机 平板电脑 智能家居 |
| Android 10.0 | 29 | Q | 智能手机 平板电脑 |
| Android 9.0 | 28 | Pie(P) | 智能手机 平板电脑 |
| Android 8.1 | 27 | 0reo(0) | 智能手机 平板电脑 |
| Android 8.0 | 26 | 0reo(0) | 智能手机 平板电脑 |
| Android 7.1 | 25 | Nouget(N) | 智能手机 平板电脑 |
| Android 7.0 | 24 | Nouget(N) | 智能手机 平板电脑 |
| Android 6.0 | 23 | Marshmallow(M) | 智能手机 平板电脑 |
| Android 5.1 | 22 | Lollipop MR1(L) | 智能手机 平板电脑 |
| Android 5.0.1 | 21 | Lollipop | 智能手机 平板电脑 |

- •创建Android工程
 - •填写应用程序名称与工程名称

| 系统版本 | API等级 | 版本代号 | 支持设备类型 |
|-----------------------|-------|--------------------|----------|
| Android 4.4W | 20 | KitKat Wear | 可穿戴设备 |
| Android 4.4 | 19 | KitKat | 智能手机平板电脑 |
| Android 4.3 | 18 | Jelly Bean | 智能手机平板电脑 |
| Android 4.2 | 17 | Jelly Bean | 智能手机平板电脑 |
| Android 4.1 | 16 | Jelly Bean | 智能手机平板电脑 |
| Android 4.0.3 - 4.0.4 | 15 | Ice Cream Sandwich | 智能手机平板电脑 |
| Android 4.0 | 14 | Ice Cream Sandwich | 智能手机平板电脑 |
| Android 3.2 | 13 | HONEYCOMB_MR2 | 平板电脑 |
| Android 3.1.x | 12 | HONEYCOMB_MR1 | 平板电脑 |
| Android 3.0.x | 11 | HONEYCOMB | 平板电脑 |
| Android 2.3.4 | 10 | GINGERBREAD_MR | <u> </u> |
| Android 2.3.3 | 10 | 1 | 習能于机 |
| Android 2.3.2 | | | |
| Android 2.3.1 | 9 | GINGERBREAD | 智能手机 |
| Android 2.3 | | | |
| Android 2.2.x | 8 | FROYO | 智能手机 |
| Android 2.1.x | 7 | ECLAIR_MR1 | 智能手机 |
| Android 2.0.1 | 6 | ECLAIR_0_1 | 智能手机 |
| Android 2.0 | 5 | ECLAIR | 智能手机 |
| Android 1.6 | 4 | DONUT | 智能手机 |
| Android 1.5 | 3 | CUPCAKE | 智能手机 |
| Android 1.1 | 2 | BASE_1_1 | 智能手机 |
| Android 1.0 | 1 | BASE | 智能手机 |

- •完成创建
 - •最后单击Finish按钮,工程向导会根据用户所填写的Android工程信息,自动 在后台创建Android工程所需要的基础文件和目录结构。当创建过程结束, 用户将看到下图的内容





•用户无须在HelloAndroid工程中添加任何代码,即可运行HelloAndroid程序。但为了让 Android程序能够正常运行,必须首先建立Android虚拟设备(Android Virtual Device, AVD)

/ 建立Android虚拟设备

- •Android虚拟设备(AVD)
 - •为了让Android程序能够正常运行,必须首先建立Android虚拟设备(Android Virtual Device, AVD)
 - 配置AVD最简单的方式是通过Android Studio的Tools→AVD Manager启动AVD管理器,也可以 点击上方工具栏中机器人与手机的图标打开AVD管理器

| 👗 Android Virtual Device Manager | - | × |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|---|
| Your Virtual Devices | | |
| | | |
| Virtual devices allow you to test your application without having to own the physical devices. | | |
| + Create Virtual Device | | |
| To prioritize which devices to test your application on, visit the Android Dashboards, where you can get up-to-date information on which devices are active in the Android and Google Play ecosystem. | | |

/ 建立Android虚拟设备

•Android虚拟设备(AVD)

• 在AVD管理器中单击Create Virtual Device按钮,打开AVD创建界面,如下图所示。选择其中一个分辨率合适的设备,点击next。选择一个Android sdk的版本点击Download下载

| 👗 Virtual Device C | onfiguration | | | | | | | × |
|--------------------|------------------|-------------------|-------|------------|---------|-------------|-----------------|---------------|
| 🔊 Se | ect Hardwa | are | | | | | | |
| Choose a de | vice definition | | | | | | | |
| | Q. | | | | | D Pixel 2 | | |
| Category | Name 🔻 | Play Store | Size | Resolution | Density | | | |
| TV | Pixel 3 | ⊳ | 5.46" | 1080x2160 | 440dpi | - 1080px | | |
| Phone | Pixel 2 XL | | 5.99" | 1440x2880 | 560dpi | | Size: large | |
| Wear OS | Pixel 2 | ⊳ | 5.0" | 1080x1920 | 420dpi | 5.0" 1920px | Density: 420dpi | |
| Tablet | Pixel | | 5.0" | 1080x1920 | 420dpi | | | |
| Automotive | Nexus S | | 4.0" | 480x800 | hdpi | | | |
| | Nexus One | | 3.7* | 480x800 | hdpi | | | |
| | Nevus 6P | | 5 7* | 1440v2560 | 560dni | | | |
| New Hardware | e Profile Import | Hardware Profiles | | | G | | | Clone Device |
| | | | | | | | | |
| ? | | | | | | Previous | Next | Cancel Finish |

| elect a system image commended <u>x86 image</u> Other Images Release Name API Level × ABI Target 5 Download S x86_64 Android API S (Google Play) 6 Download 3C x86_64 Android 11.0 (Google Play) 7 Download 3C x86_64 Android 11.0 (Google Play) 8 Download 3C x86_64 Android 11.0 (Google Play) 8 Download 3C x86_64 Android 11.0 (Google Play) 9 Download 29 x86_64 Android 10.0 (Google Play) | Syst | em Image | 9 | | |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|--------------------------|--------|-----------------------------|------------------------------------------------------------------|
| Release Name API Level × ABI Target \$ Download \$ x86_64 Android API \$ (Google Play) \$ Download \$ x86_64 Android API \$ (Google APIs) \$ Download \$ x86_64 Android 11.0 (Google APIs) \$ Download \$ x86_64 Android 11.0 (Google APIs) \$ Download \$ x86_64 Android 11.0 (Google APIs) \$ Download \$ \$ X86_64 \$ Download \$ \$ \$ \$ Download \$ \$ \$ | elect a system | image Images Other Im | nages | | R |
| s Download S X86_64 Android API S (Google Play) s Download S X86_64 Android API S (Google APIs) k Dewnload 3C X86_64 Android 11.0 (Google APIs) k Download 3C X86 Android 11.0 (Google APIs) k Download 3C X86 Android 11.0 (Google APIs) k Download 3C X86_64 Android 11.0 (Google APIs) g Download 29 X86_64 Android 10.0 (Google Play) | Release Name | API Level 🔻 | ABI | Target | |
| S Download S x86_64 Android API S (Google APIs) 30 A Dewnload 3C x86_64 Android 11.0 (Google Play) Android Android K Download 3C x86 Android 11.0 (Google APIs) Android 11.0 K Download 3C x86_64 Android 11.0 (Google APIs) Google Inc. Google Inc. | 5 Download | 5 | x86_64 | Android API S (Google Play) | API Level |
| R Dexmload 30 x86_64 Android 11.0 (Google Play) R Download 30 x86 Android 11.0 (Google APls) Android R Download 30 x86_64 Android 11.0 (Google APls) Google Inc. Q Download 29 x86_64 Android 10.0 (Google Play) Google Inc. | 5 Download | S | x86_64 | Android API S (Google APIs) | 30 |
| R Download 3L x86 Android 11.0 (Google APIs) 11.0 R Download 3L x86_64 Android 11.0 (Google APis) 11.0 Q Download 29 x86_64 Android 10.0 (Google APis) Google Inc. | R Download | 36 | x86_64 | Android 11.0 (Google Play) | |
| R Download 3L x86_64 Android 11.0 (Google APIs) Google Inc. Q Download 29 x86_64 Android 10.0 (Google Play) Google Inc. | R Download | 30 | x86 | Android 11.0 (Google APIs) | 11 0 |
| Q Download 29 x86_64 Android 10.0 (Google Play) | R Download | 36 | x86_64 | Android 11.0 (Google APIs) | Google Inc |
| | Q Download | 29 | x86_64 | Android 10.0 (Google Play) | Google Inc. |
| Q Download 29 x86 Android 10.0 (Google APIs) System Image | Q Download | 29 | x86 | Android 10.0 (Google APIs) | System Image |
| Q Download 29 x86_64 Android 10.0 (Google APIs) x86_64 | Q Download | 29 | x86_64 | Android 10.0 (Google APIs) | x86_64 |
| Questions on API level? See the API level distribution chart | - | | | 0 | Questions on API level / See the API level distribution chart |

/ 建立Android虚拟设备

•Android虚拟设备(AVD)

- •在AVD Name输入框中输入AVD名称后,点击Finish按钮保存AVD的配置信息。
- •然后在AVD管理器中单击Start按钮启动Android模拟器

| Kirtual Device Configuration | × | 👗 And | roid Virtual Device Manager | | | | | | | - | | × |
|--------------------------------------------------------|-----------------------------|-------|----------------------------------|------------|---------------------|-----|----------------------|---------|--------------|---------|---|---|
| Android Virtual Device (AVD) | | | Your Virtual [Android Studio | Device | 5 | | | | | | | |
| Verify Configuration | | Туре | Name | Play Store | Resolution | API | Target | CPU/ABI | Size on Disk | Actions | | |
| AVD Name Pixel 2 API 30 2 | | Co | Pixel 2 API 30 2 | ⊳ | 1080 × 1920: 420dpi | 30 | Android 11.0 (Google | x86 | 9.3 GB | • | / | |
| Divisition 5.0 1080x1920 xxhdpi Change | | | | | | | | | | | | |
| R Android 11.0 x86_64 Change | | | | | | | | | | | | |
| Startup orientation Portrait Landscape | Nothing Selected | | | | | | | | | | | |
| Emulated Graphics: Automatic v | | | | | | | | | | | | |
| Device Frame 🗹 Enable Device Frame | | | | | | | | | | | | |
| Show Advanced Settings | | | | | | | | | | | | |
| | | + | Create Virtual Device | | | | | | | G | | ? |
| ? | Previous Next Cancel Finish | | | | | | | | | | | |

建立Android虚拟设备

- •创建运行/调试配置信息
 - 在运行、调试或测试Android 程序代码之前,需要创建运行 /调试配置信息
 - 首先选择工具栏Add Configuration... 下拉列表
 - 然后点击Run/Debug Configurations对 话框左上方的"+"号按钮
 - 选择Android App模板创建运行配置



建立Android虚拟设备

- •使用Android Studio运行Android程序非 常简单
 - •只要从Run→Run'app' 便可运行Android程序
 - Android Studio会自动完成Android程序编译、 打包和上传等过程,并将程序的运行结果 显示在模拟器中
- •HelloAndroid程序的运行结果



Hello World!



- •在Android Studio中一个程序项目有三种视图
 - Android视图
 - Project视图

• Packages视图

3.3 Android程序结构

- Android视图是通过类型来组织项目的资产文件
 - •例如,AndroidManifest文件和XML文件在manifests文件中,所有的Java类都在java文件夹里面,所有的资源文件都在res文件夹。Android视图如右图所示



• 在默认的Android视图结构中,不能反映项目在磁盘上的实际物理组织。要查看项目的实际结构,就要切换到Project视图结构,如下图所示。



 Package视图与Project视图相比,最大的区别就是隐藏了相关的配置文件,属性 文件和系统自身的目录,只显示当前的Module列表和Module下面的目录和文件, 如下图所示



•建立HelloAndroid程序

- 在建立HelloAndroid程序的 过程中, Android Studio会 自动建立一些目录和文件
- •这些目录和文件有其固定的作用,有的允许修改,有的则不能进行修改。

| - | - | Halls Andreid Dy Andreid Ctudio Markelace \ Halle Andreid |
|---|---|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v | | |
| | | |
| | 5 | lidea |
| | | app |
| | | |
| | | |
| | | src |
| | | android lest |
| | | w main |
| | | Java |
| | | Ice.nrbeu.nelloandroid Main A stiniture |
| | | |
| | | les les |
| | | |
| | | |
| | | minman anudni v26 |
| | | mipmap-anyopi-vzo |
| | | mipmap-ndpi |
| | | mipmap-mapi |
| | | mipmap-xndpi |
| | | mipmap-xxhdpi |
| | | |
| | | Values-night |
| | | AndroidManifest xml |
| | | ▶ test |
| | | aitignore |
| | | Support of the second seco |
| | | proguard-rules.pro |
| | | aradle |
| | | V wrapper |
| | | gradle-wrapper.jar |
| | | a gradle-wrapper.properties |
| | | gitignore |
| | | a build.gradle |
| | | gradle.properties |
| | | I gradlew |
| | | a gradlew.bat |
| | | local.properties |
| | | R settings.gradle |
| ► | | External Libraries |
| | | Scratches and Consoles |



- •说明
 - 在Project视图下, Android Studio以工程名称HelloAndroid和External Libraries作为根目录,将所有自动生成的和非自动生成的文件都保存在这两个根目录下
- •子目录、库和工程文件
 - •根目录下包含4个子目录.gradle、.idea、app和gradle
 - External Libraries根目录存放项目所依赖的所有类库
 - •7个工程文件.gitignore、build.gradle、gradle.properties、gradlew、gradlew.bat、local.properties和settings.gradle

•.grandle目录和.idea目录

•.grandle目录和.idea目录是存放Android Studio自动编译工具生成的文件和开发工 具产生的文件

•app目录

- app目录用来存放程序中的代码资源文件。
- •其中,build目录在编译时生成,主要包含 编译时自动生成的内容

| ∇ | R, | HelloAndroid D:\AndroidStudioWorkplace\HelloAndroid |
|----------|----|-----------------------------------------------------|
| | ► | 🖿 .gradle |
| | ► | 🖿 .idea |
| | ▼ | app |
| | | 🕨 🔚 build |
| | | 🖿 libs |
| | | V src |
| | | androidTest |
| | | 🔻 🖿 main |
| | | 🔻 🖿 java |
| | | 🔻 🖿 ice.hrbeu.helloandroid |
| | | C MainActivity |
| | | 🔻 🖿 res |
| | | drawable |
| | | drawable-v24 |
| | | layout |
| | | 🕨 🖿 mipmap-anydpi-v26 |
| | | 🕨 🖿 mipmap-hdpi |
| | | 🕨 🖿 mipmap-mdpi |
| | | 🕨 🖿 mipmap-xhdpi |
| | | 🕨 🖿 mipmap-xxhdpi |
| | | 🕨 🖿 mipmap-xxxhdpi |
| | | values |
| | | values-night |
| | | 🛲 AndroidManifest.xml |
| | | test |
| | | 🜏 .gitignore |
| | | 🗬 build.gradle |
| | | 붙 proguard-rules.pro |
| | ۳ | In gradle |
| | | V wrapper |
| | | gradle-wrapper.jar |
| | | gradle-wrapper.properties |
| | | 🐻 .gitignore |
| | | build.gradle |
| | | radle.properties |
| | | I gradlew |
| | | gradlew.bat |
| | | in local.properties |
| | | w settings.gradle |
| ► | | External Libraries |
| | | Scratches and Consoles |

•app目录

- outputs 目录下存放打包好的apk文件,libs 目录存放第三方jar包,然后 jar包会被自动添加到构建路径
 - (如集成地图sdk,把jar包放到libs目录,可以在build.gradle文件中查看 当前项目依赖)
- src目录是源码目录
 - androidTest目录是用来编写android test测试用例的,可以对项目进行自动化测试
 - main目录下的java目录是存放Java代码的地方,res是存放资源的目录。
 Android程序所有的图像、颜色、风格、主题、界面布局和字符串等资源都保存在res目录下的几个子目录中。

•res目录

- drawable 目录用来保存图像文件
- layout 目录用来保存与用户界面相关的布局文件
- •mipmap-hdpi、mipmap-mdpi、mipmap-xhdpi、mipmap-xxhdpi和 mipmap-xxxhdpi目录用来保存同一个程序中针对不同屏幕尺寸需要显 示的不同大小的图标文件,引导页的图片也建议放在这里
- values 目录保存颜色、风格、主题和字符串等资源
- Android Manifest.xml是整个项目的配置文件,四大组件都需要在这里注册 才能正常的运行

•src目录

- •test目录用来编写Unit Test测试用例的目录,是对项目进行自动化测试的另一种方式
- •.gitignore文件是为git源码管理的配置文件
- build.gradle文件是Android项目的Gradle构建脚本文件,用于配置 Android构建过程所需要的参数和引用依赖

•gradle目录

- gradle目录下包含了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,可以点击 Android Studio导航栏 --> File --> Settings --> Build,Execution,Deployment --> Gradle 进行配置更改。

- •.gitignore(外层)文件
 - •.gitignore(外层)文件用于将指定的目录或文件排除在版本控制之外, 作用和内层的.gitignore文件类似
- •build.gradle(外层)文件
 - build.gradle(外层)文件是项目全局编译环境配置

•gradle.properties文件

- •gradle.properties是全局的gradle配置文件。这里配置的属性将会影响到项目中所有的gradle编译脚本
- •gradlew和gradlew.bat文件
 - •gradlew和gradlew.bat用来在命令行界面执行gradle命令,其中gradlew是在Linux或Mac系统中使用,gradlew.bat是在Windows系统中使用

- •. local.properties文件
 - •.local.properties配置文件用来指定本机中的Android SDK的路径,一般 是自动生成,除非电脑SDK位置发生变化,否则无需修改该文件的路 径
- •setting.gradle文件
 - setting.gradle用于指定项目中所有引入的模块。由于项目中就只有一个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,需要手动去修改这个文件的场景比较少

- •. local.properties文件
 - •.local.properties配置文件用来指定本机中的Android SDK的路径,一般 是自动生成,除非电脑SDK位置发生变化,否则无需修改该文件的路 径
- •setting.gradle文件
 - setting.gradle用于指定项目中所有引入的模块。由于项目中就只有一个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,需要手动去修改这个文件的场景比较少

- AndroidManifest.xml
 - •AndroidManifest.xml是XML格式的Android程序声明文件
 - 包含了Android系统运行Android程序前所必须掌握的重要信息,这些信息包括应用程序名称、图标、包名称、模块组成、授权和SDK最低版本等

AndroidManifest.xml文件的代码如下:

- 1. <?xml version="1.0" encoding="utf-8"?>
- 2. <manifest xmlns:android="http://schemas.android.com/apk/res/android" package="ice.hrbeu.helloandroid"
- 3. android:versionCode="1"
- 4. android:versionName="1.0">
- 5. <uses-sdk
- 6. android:minSdkVersion="16"
- 7. android:maxSdkVersion="30"/>
- 8. <application
- 9. android:allowBackup="true"
- 10. android:icon="@mipmap/ic_launcher"
- 11. android:label="@string/app_name"
- 12. android:roundIcon="@mipmap/ic_launcher_round"
- 13. android:supportsRtl="true"
- 14. android:theme="@style/Theme.HelloAndroid">
- 15. <a ctivity
- 16. android:name=".MainActivity"
- 17. android:label="@string/app_name">
- 18. <intent-filter>
- 19. <action android:name="android.intent.action.MAIN" />
- 20. <category android:name="android.intent.category.LAUNCHER" />
- 21. </intent-filter>
- 22. </activity>
- 23. </application>
- 24. </manifest>

- •AndroidManifest.xml文件
 - manifest元素仅能包含一个application元素,application元素中能够声明 Android程序中最重要的四个组成部分
 - •包括Activity、Service、BroadcastReceiver和ContentProvider,所定义的属性将影响所有组成部分

•activity元素

- •是对Activity子类的声明,不在AndroidManifest.xml文件中声明的 Activity将不能够在用户界面中显示
- intent-filter
 - intent-filter中声明了两个子元素action和category

•引用资源

资源引用有两种情况:一种是在代码中引用资源;另一种是在资源中引用资源

- 代码中引用资源,需要使用资源的ID,可以通过 [R.resource_type.resource_name]或 [android.R.resource_type.resource_name]获取资源ID
 - resource_type代表资源类型,也就是R类中的内部类名称
 - resource_name代表资源名称,对应资源的文件名或在XML文件中定义的资源名称属性
- •资源中引用资源,引用格式: @ [package:]type:name
 - @表示对资源的引用
 - package是包名称,如果在相同的包,package则可以省略

• activity_main.xml文件

• activity_main.xml文件是界面布局文件,利用XML语言描述的用户界面,界面布局的相关 内容将在第5章用户界面设计中进行详细介绍

• activity_main.xml文件的代码如下:

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android" xmlns:tools="http://schemas.android.com/tools" 2. 3. android:layout width="match parent" android:layout_height="match_parent" 4. 5. tools:context="ice.hrbeu.helloandroid.MainActivity" > 6. 7. <TextView android:layout_width="wrap_content" 8. android:layout height="wrap content" 9. 10. android:text="@string/hello_world" /> 11. </RelativeLayout> 12.

- •activity_main.xml文件
 - •代码的第7行说明在界面中使用TextView控件,TextView 控件主要用来显示字符串文本
 - •代码第10行说明TextView控件需要显示的字符串,非常 明显,@string/hello_world是对资源的引用

•Strings.xml文件的代码

1. <?xml version="1.0" encoding="utf-8"?>

```
2. <resources>
```

- 4. <string name="app_name">HelloAndroid</string>
- 5. <string name="hello_world">Hello world!</string>
- 6. <string name="action_settings">Settings</string>

```
7.
```

3.

```
8. </resources>
```

- •通过strings.xml文件的第5行代码分析,在TextView控件中显示的字符串应是"Hello World, HelloAndroidActivity!"
- •如果读者修改strings.xml文件的第5行代码的内容,重新编译、运行后,模拟器中显示的结果也应该随之更改

MainActivity.java

- MainActivity.java是Android工程向导根据Activity名称创建的java文件
- •这个文件完全可以手工修改
- •为了在Android系统上显示图形界面,需要使用代码继承Activity类
- •并在onCreate()函数中声明需要显示的内容

•MainActivity.java文件的代码如下:

| 1 | package edu.hrbeu.helloandroid; | | | | | | | | | |
|------------|--------------------------------------------------------------------|--|--|--|--|--|--|--|--|--|
| 2 | import android app Activity: | | | | | | | | | |
| 1 | import and old app. Activity, | | | | | | | | | |
| 4 | import android.os.Bundle; | | | | | | | | | |
| 5 | import android.view.Menu; | | | | | | | | | |
| 6 | import android.view.MenuItem; | | | | | | | | | |
| 7 | | | | | | | | | | |
| 8 | public class MainActivity extends Activity { | | | | | | | | | |
| 9 | | | | | | | | | | |
| 1 |) @Override | | | | | | | | | |
| 1 | nrotected void on Create (Bundle savedInstance State) | | | | | | | | | |
| 1 | 1. protected void officieate(Dufful SavedifistanceState) { | | | | | | | | | |
| 1 | 2. super.onCreate(savedInstanceState); | | | | | | | | | |
| 1 | 3. setContentView(R.layout.activity_main); | | | | | | | | | |
| 1 | 4. } | | | | | | | | | |
| 1 | 5. | | | | | | | | | |
| 1 | 6. @Override | | | | | | | | | |
| 1 | 7. public boolean onCreateOptionsMenu(Menu menu) { | | | | | | | | | |
| 1 | 3. // Inflate the menu; this adds items to the action bar if it is | | | | | | | | | |
| | present. | | | | | | | | | |
| 1 | 9. getMenuInflater().inflate(R.menu.main, menu); | | | | | | | | | |
| 2 | D. return true; | | | | | | | | | |
| 2 | 1. } | | | | | | | | | |
| 2 | 2. @Override | | | | | | | | | |
| : - | | | | | | | | | | |

•MainActivity.java文件的代码如下:



•代码的第3行和第4行,通过android.jar从Android SDK中引入了Activity和Bundle两个重要的包,用以子类继承和信息传递

•MainActivity.java文件

- •第8行声明MainActivity类继承Activity类
- 第10行表明需要重写onCreate()函数
- •第11行的onCreate()会在Activity首次启动时会被调用,为了便于理解,可以认为onCreate()是HelloAndroid程序的主入口函数
- •第12行调用父类的onCreate()函数,并将savedInstanceState传递给父类, savedInstanceState是Activity的状态信息;
- •第13行声明了需要显示的用户界面,此界面是用XML语言描述的界面 布局,保存在scr/layout/activity_main.xml资源文件中



•到这里分析了Android程序的目录结构和文件的用途,对 AndroidManifest.xml文件、Java代码文件、资源引用等内 容有了初步的了解



- •1.简述AndroidManefiest.xml文件的用途。
- 2.简述res目录下的各种资源类型。
- 3.使用Android Studio建立名为MyAndroidStudio的工程, 包名称为edu.hrbeu.MyAndroidStudio,程序运行时显示 Hello MyAndroidStudio。