

setCTS.apk 使用及实现指南

如何快速配置 Android Device Configuration, 一个应用帮您解放双手--- setCTS. apk.

一:setCTS 功能实现如下:

- 1:Location 选项 -> on
- 2:WiFi 选项 -> on
- 3:USB Debugging -> on
- 4:Allow mock locations -> on(only in Android 5.x and 4.4.x)
- 5:Stay awake -> on
- 6:拷贝并安装 CtsDeviceAdmin. apk

二:实现方案如下:

方案一:setCTS APK 获得系统权限, 调用 Setting 接口;(本文档重点介绍)

方案二:Setting 中添加第三方应用接口, 使得可以修改 Setting 中设置选项, 实现方式:通过 Intent 发送消息, 与 Setting 讨论具体实现方案, 评估工作量及风险;(方案一失败则启用该方案)

方案三:脚本模拟用户点击操作, 触发设置中选项开关;(可后续通过该方案尝试实现跑 CTS 自动测试前“删除 Google 账户”功能)

方案四:将应用以“第三方应用”集成至当前软件版本系统.(该方案是方案二的补充)

三: 实现方案(一)背景介绍

背景:

将 APK 安装至测试设备中报错截屏如 图 1:

```
user@user-OptiPlex-7040:~$ adb install -r '/home/user/myself/setCTS_2016.12.13/setCTS/bin/setCTS.apk'
[100%] /data/local/tmp/setCTS.apk
  pkg: /data/local/tmp/setCTS.apk
Failure [INSTALL_FAILED_SHARED_USER_INCOMPATIBLE]
```

图 1

当前报错原因:当前 APK 没有签名或签名不符合系统签名.

解决方案:为 setCTS. apk 进行系统签名

方案 1: 在 Android 系统源码下用 make 进行编译, (适用于 setCTS. apk 作为第三方应用集成至软件系统, 简单了解即可, 签名方案 setCTS APK 采用方案 2)

3.1.1:在 AndroidManifest.xml 中的 manifest 节点加入属性 android:sharedUserId =
“android.uis.system”

3.1.2:修改 Android.mk 文件加入 LOCAL_CERTIFICATE:=platform(该处根据系统签名需要备选 media/shared)这个 APK 就拥有了和 system 相同的签名

3.1.3:使用“mm”加相对路径编译该 APK, 编译成功后 APK 文件带签名.

方案 2:不需要源码编译环境但需要得到软件系统签名文件及工具

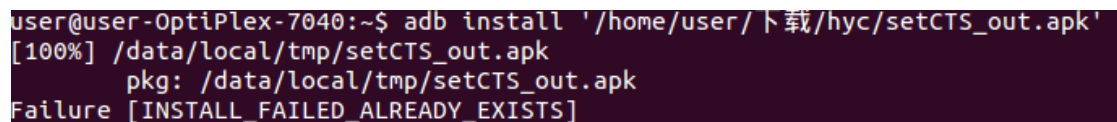
3.2.1:使用 eclipse 开发工具选中工程项目->右键->” Run as” ->” Android Application” -> 将 bin 文件夹根目录下生成的 APK(此时 APK 不带签名)拷贝至一个全新文件夹中(暂命名 “signature”)

3.2.2:从源码路径中找出签名工具(out/host/linux-x86/framework/signapk.jar)

- 3.2.3:从源码路径中找出签名文件(build/target/product/security)platform.pk8 和 platform.x509.pem(无盒子 M12 源码可从开发工程师处拷贝)
- 3.2.4:将被签名 APK(2.1 步骤中获得),两个签名文件,一个签名工具均放在 signature 文件夹中
- 3.2.5:执行命令#`java -jar signapk.jar platform.x509.pem platform.pk8 setCTS.apk setCTS_out.apk`(其中 setCTS_out.apk 为生成的签名文件)

注 1:

在安装过程中如遇到图 2 状况请使用#adb install -r 命令



```
user@user-OptiPlex-7040:~$ adb install '/home/user/下载/hyc/setCTS_out.apk'
[100%] /data/local/tmp/setCTS_out.apk
      pkg: /data/local/tmp/setCTS_out.apk
Failure [INSTALL_FAILED_ALREADY_EXISTS]
```

图 2

注 2:

报错 Installation error:INSTALL_FAILED_UPDATE_INCOMPATIBLE

报错原因:设备中重复安装同一应用.

解决方案:在 data/app 目录、system/app 路径、data/data 路径/data/system/packages.xml 文件中将 该 APK 相关项删除,然后再重新安装。

注 3:

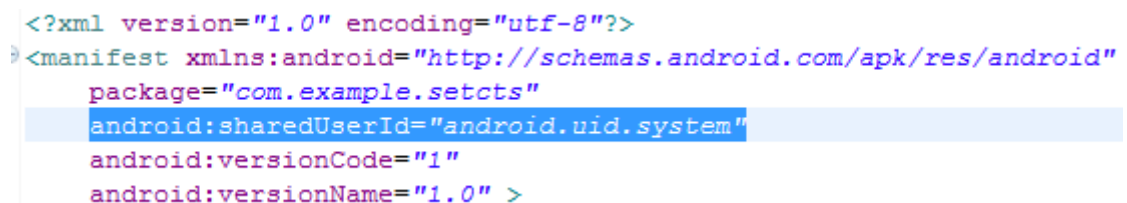
如果签名失败建议将 APK 解压后的 CETR.SF 和 CERT.RSA 文件删除再重新签名。

注 4:

综上,为什么要为 APK 进行系统签名?不同的应用程序间如果需要共享数据或共享代码(例如我们需要调用 Google 系统 Setting 应用方法),就需要保证这两个 APK 运行在同一进程中,并且他们拥有同一签名证书。

注 5:

结合下文“AndroidManifest 部分”分析 setCTS apk AndroidManifest.xml



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.setcts"
    android:sharedUserId="android.uid.system"
    android:versionCode="1"
    android:versionName="1.0" >
```

AndroidManifest 添加签名必须属性

添加 sharedUserId 可以将不同的程序加入到同一进程,方便数据共享,当然,前提是共享 Id 的两个程序必须使用相同签名;定义“uid.system”意味将 setCTS 应用加入到系统进程中,也就可以实现对 Setting 方法的调用。

四:API Level 概念及使用

API 级别是一个对 Android 平台版本提供的框架 API 修订版进行唯一标识的整数值;每个 Android 平台版本都将其 API 级别标识符存储在 Android 系统自身内部,图 3 列举出各 Android 平台版本支持的 API 级别,开发中作为参考。

平台版本	API 级别	VERSION_CODE
Android 7.0	24	N
Android 6.0	23	M
Android 5.1	22	LOLLIPOP_MR1
Android 5.0	21	LOLLIPOP
Android 4.4W	20	KITKAT_WATCH
Android 4.4	19	KITKAT
Android 4.3	18	JELLY_BEAN_MR2
Android 4.2、4.2.2	17	JELLY_BEAN_MR1
Android 4.1、4.1.1	16	JELLY_BEAN
Android 4.0.3、4.0.4	15	ICE_CREAM_SANDWICH_MR1
Android 4.0、4.0.1、4.0.2	14	ICE_CREAM_SANDWICH
Android 3.2	13	HONEYCOMB_MR2
Android 3.1.x	12	HONEYCOMB_MR1
Android 3.0.x	11	HONEYCOMB
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD

图 3

五：下载、安装配置及使用 Eclipse 工具

5.1 ADT(Android Development Tools,Android 开发工具)，ADT 可以帮助我们在 Eclipse 开发环境中快速创建 Android 应用程序，自动生成一些简单代码.Android 提供了针对 Eclipse 的开发插件，ADT 可以调用 SDK 中的工具，具体哪些工具见下.

5.2 SDK(software development kit)软件开发工具，SDK 中提供了一系列工具：模拟硬件设备的模拟器(Emulator)、AAPT、DDMS(Dalvik Debug Monitor Service)常用功能有：为测试设备截屏，查看特定行程中正在运行的线程以及堆内存使用情况信息等；

5.3 综上,ADT 作为 Eclipse 开发工具的插件调用 SDK 中的工具进行可视化开发,提高开发效率,同时使用 Eclipse 的又需要 JDK 的支持，环境配置准备工作是这样的：下载安装 JDK、下载 Eclipse、下载 SDK、下载安装 ADT

5.3.1 下载安装 Eclipse

网址: <https://www.eclipse.org/>

下载见图 4:

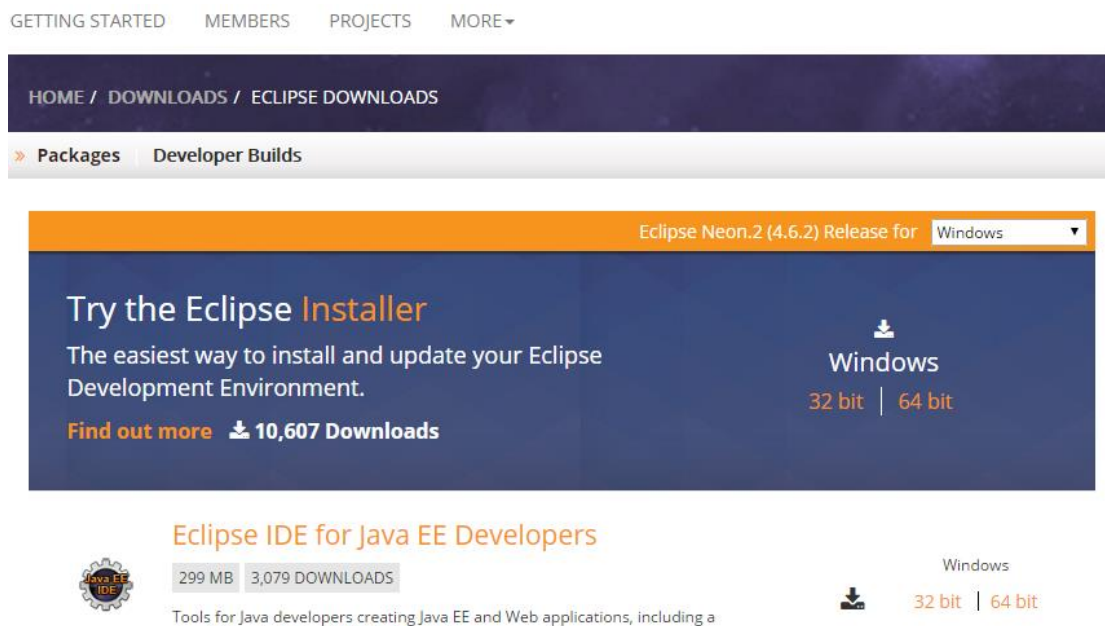


图 4

注：

下载时注意系统区分 OS(Windows 86bit/64bit、Linux)

下载完成后，点击执行文件打开 Eclipse 见图 5：

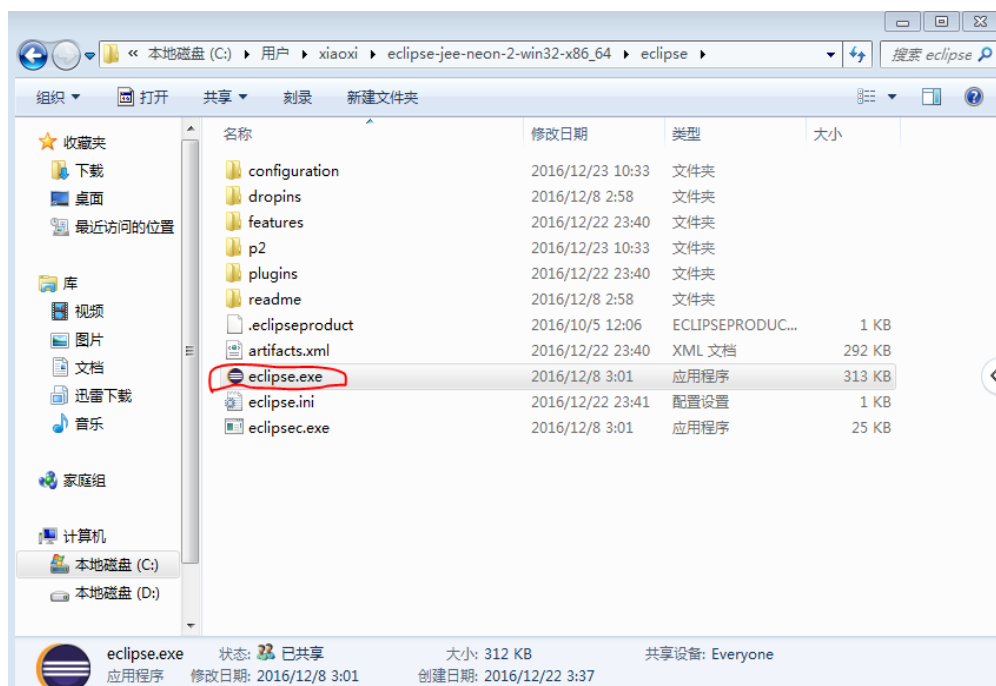


图 5

开启后报错提示如下，见图 6

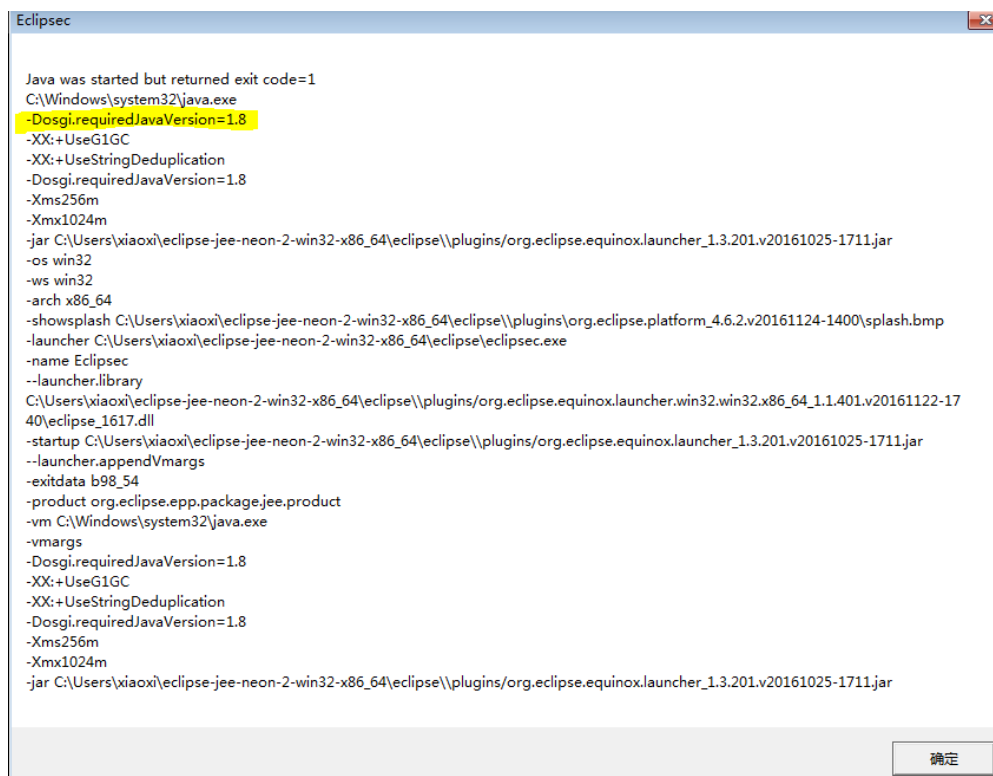


图 6

报错原因:

猜测报错原因 1: 下载 Eclipse 版本跟当前操作系统版本不相关

猜测报错原因 2: 当前 Eclipse 版本需要 JDK1.8, 根据提示, 安装 JDK1.8 如下

5.3.2 JDK1.8 官网下载见图 7

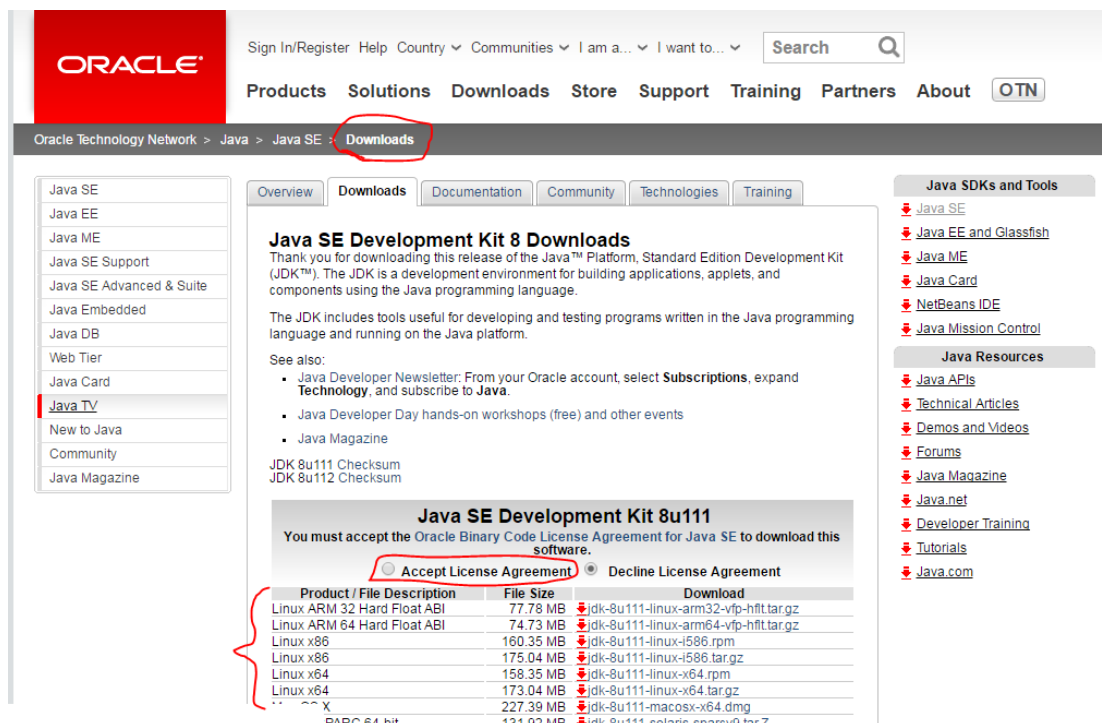


图 7

JDK1.8 安装流程见图 8、图 9



图 8



图 9

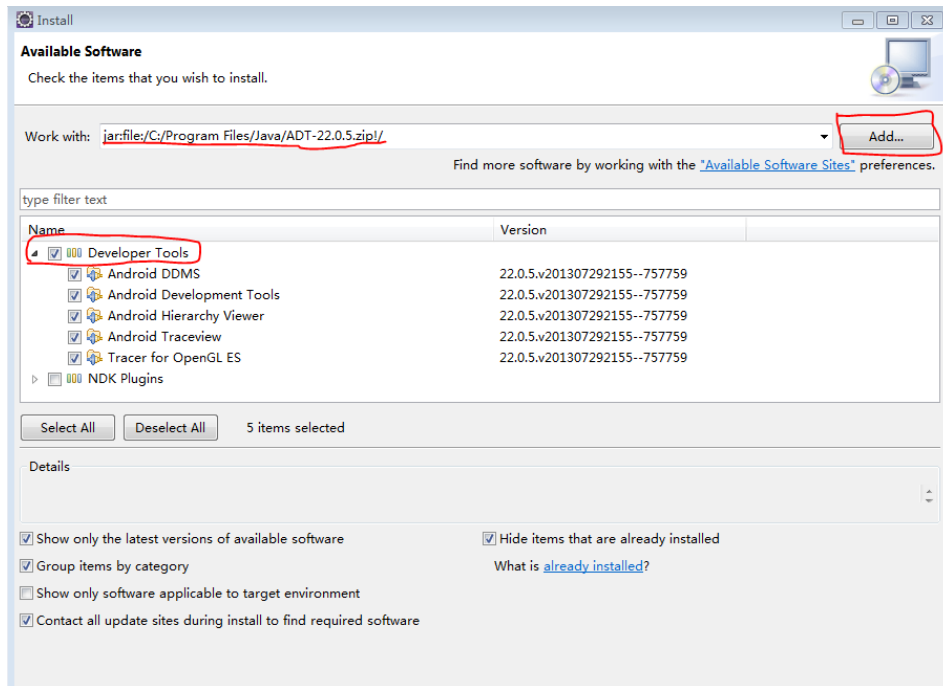
安装成功后查看当前 Java 版本，操作命令见图 10

```
D:\>java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
```

图 10

5.3.3 下载安装 Eclipse 插件 - ADT

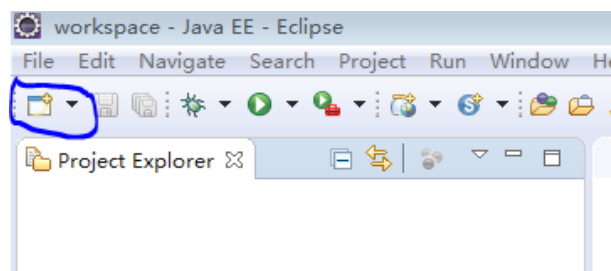
Eclipse 操作界面 “Help 选项” -> “Install New Software”，见下图 “ADT 安装界面”



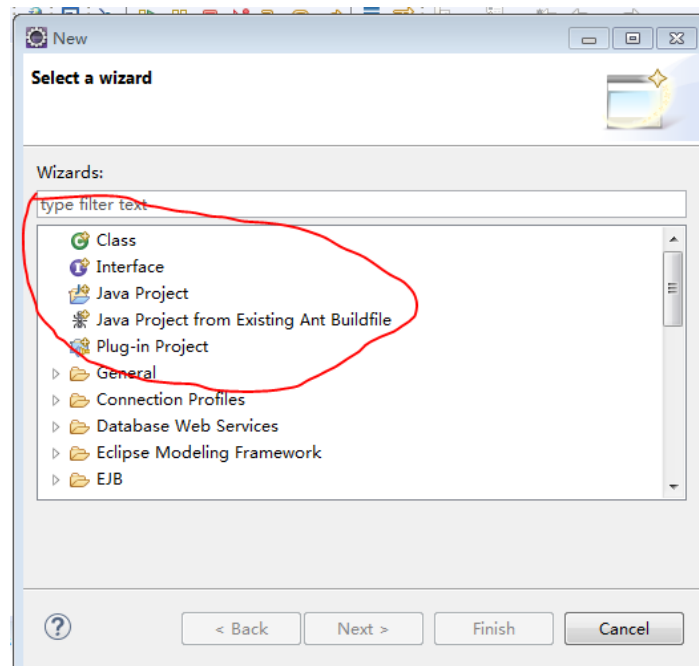
图：ADT 安装界面

5.3.4 下载安装 SDK、Eclipse 打开 setCTS 工程

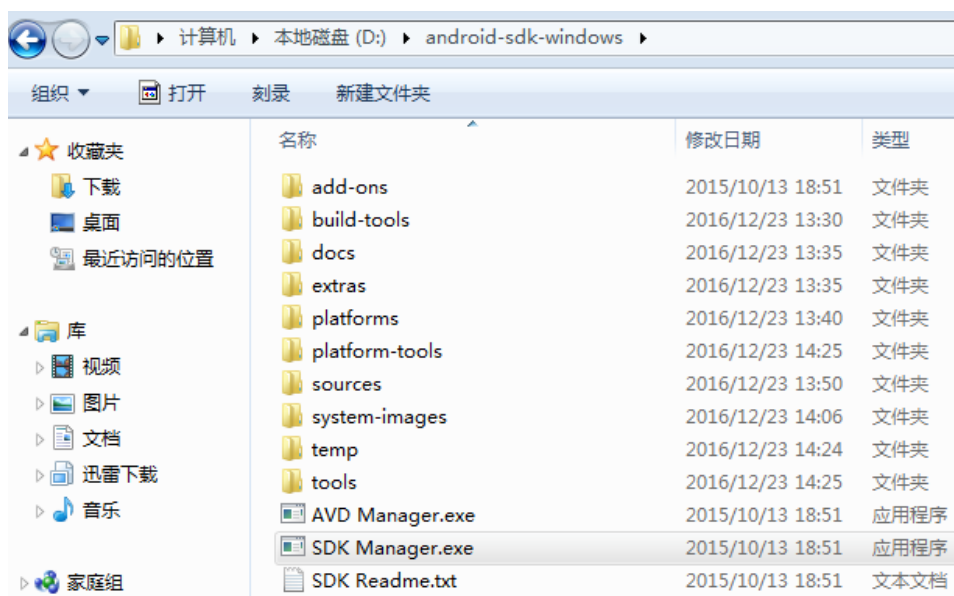
当我们使用 Eclipse 打开 setCTS 应用工程时发现找不到 Android 选项，我们需要安装 SDK，见下图“打开导入 Android 工程”，“无 Android 导入选项”



图：打开导入 Android 工程

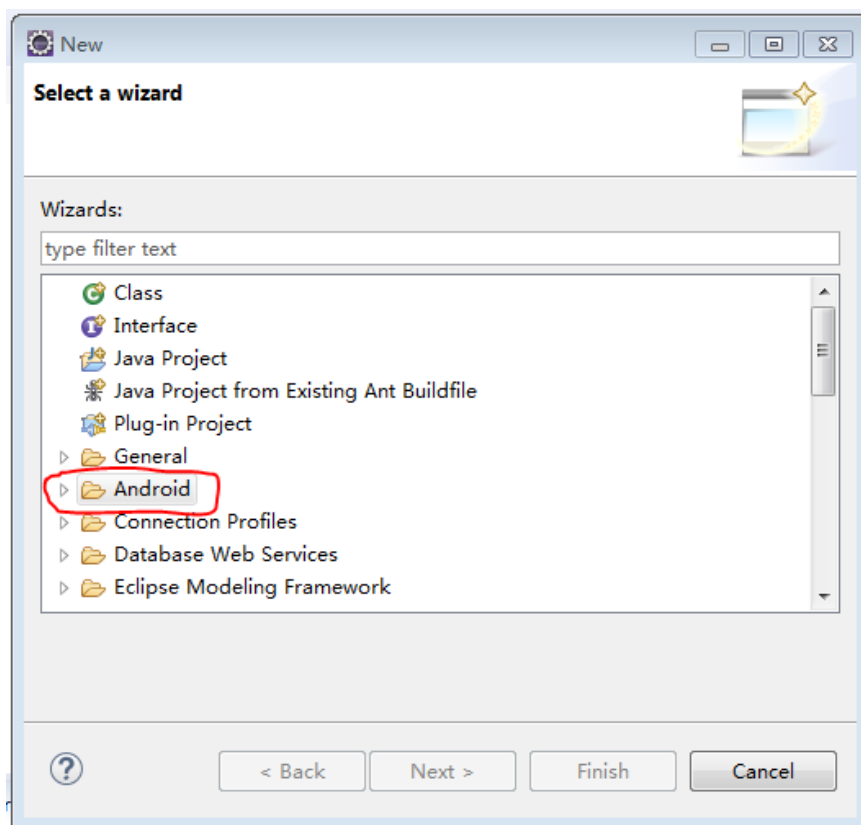


图：无 Android 导入选项



图：打开安装 SDK

安装 SDK 成功, Eclipse 打开 setCTS 源工程, 这时选择 “Android” -> “Android Project from Existing Code”, 见下图 “打开 setCTS 工程”



图：打开 setCTS 工程

六:APK 结构组成、res 目录与 assets 目录区别和使用

6.1 assets 目录下的文件不会编译成二进制, 而是直接打包到 APK 中; 同时 assets 文件中可以自行创建子目录。

6.2 编译应用时, aapt (Android Asset Packing Tool, Android 资源打包工具) 会生成 R 类, 其中包含当前工程 res/ 中所有资源 ID, res/drawable 目录下文件可以通过 R.drawable 获取当前路径下的资源文件; assets 路径下资源文件读取时必须指定文件路径, 可通过 AssetManager 类来访问这些文件。

6.3 setCTS apk 中 assets 目录存放需要拷贝并安装的 APK 文件 “CtsDeviceAdmin.apk”, 见图 “Assets 目录”。功能实现整体思路: 将 assets 目录中 apk 取出并拷贝到设备 sdcard 目录, 再进行本地安装。

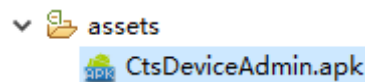
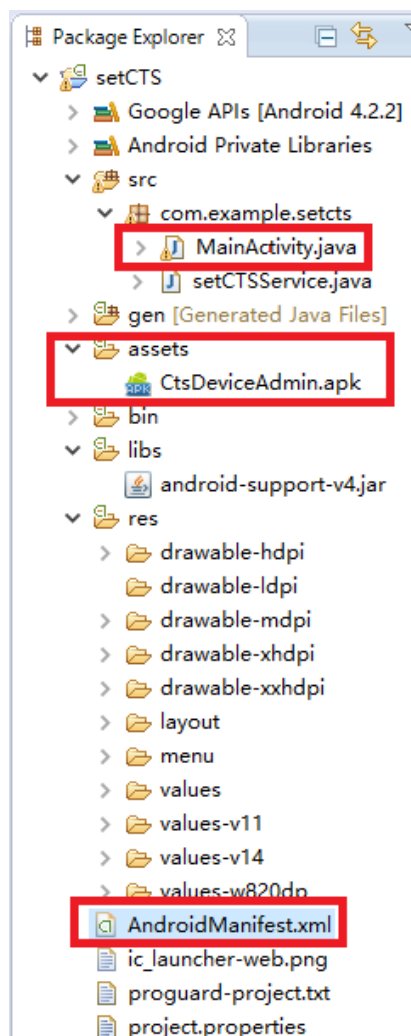


图: assets 目录文件

6.4 setCTS APK 目录结构组成



七:setCTS APK 中配置文件需声明的权限

7.1 AndroidManifest.xml 文件：所有项目中该文件名称不变，它是 Android 工程的全局配置文件，我们开发过程中用到的四大组件(Activity、Service、ContentProvider 和 Receiver)都要在该文件中声明，同时该文件还可以声明一些权限以及 SDK 的最低版本等。

7.2 setCTS apk 配置文件需要配置项如下图“配置文件声明权限”，它描述了应用程序和其他应用程序交互所需的权限，参考“注释”。

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.setcts"
    android:sharedUserId="android.uid.system"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="17"
        android:targetSdkVersion="19" />
    <!-- 允许程序读取或写入系统设置 -->
    <uses-permission android:name="android.permission.WRITE_SETTINGS" />
    <!-- 允许程序读写系统安全敏感的设置项 -->
    <uses-permission android:name="android.permission.WRITE_SECURE_SETTINGS" />
    <!-- 允许程序修改当前应用配置如本地化、开启关闭蓝牙wifi等 -->
    <uses-permission android:name="android.permission.CHANGE_CONFIGURATION" />
    <!-- 允许程序写入外部存储，如SD CARD -->
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <!-- 允许程序安装应用 -->
    <uses-permission android:name="android.permission.INSTALL_PACKAGES" />
```

图：配置文件声明权限

八:setCTS.apk 对 Android 原生系统 Setting 属性接口的调用

8.1 查看 Google Android 源码网站：<http://androidxref.com/>。

8.2 setCTS 中调用/frameworks/base/core/java/android/provider/Settings.java 的 putInt 方法修改数据库，Setting 监听数据库的改变而改变选项状态，同时其他应用监听数据库修改设置选项

AndroidXRef KitKat 4.4.2_r1

xref: /frameworks/base/core/java/android/provider/Settings.java

Home | History | Annotate | Line# | Navigate | Download Search

```
1247         } catch (NumberFormatException e) {
1248             throw new SettingNotFoundException(name);
1249         }
1250     }
1251
1252     /**
1253      * Convenience function for updating a single settings value as an
1254      * integer. This will either create a new entry in the table if the
1255      * given name does not exist, or modify the value of the existing row
1256      * with that name. Note that internally setting values are always
1257      * stored as strings, so this function converts the given value to a
1258      * string before storing it.
1259      *
1260      * @param cr The ContentResolver to access.
1261      * @param name The name of the setting to modify.
1262      * @param value The new value for the setting.
1263      * @return true if the value was set, false on database errors
1264      */
1265     public static boolean putInt(ContentResolver cr, String name, int value) {
1266         return putIntForUser(cr, name, value, UserHandle.myUserId());
1267     }
1268
1269     /** @hide */
1270     public static boolean putIntForUser(ContentResolver cr, String name, int value,
1271         int userHandle) {
1272         return putStringForUser(cr, name, Integer.toString(value), userHandle);
1273     }
```

8.3 源码实现: Allow mock locations -> on(only in Android 5.x and 4.4.x)

```
// 3: 允许模拟位置加API LEVEL 判断
// 将String 类型转化为int类型
int api_level = Integer.valueOf(API_LEVEL_NUMBER).intValue();
Log.d(TAG, "api_level:" + api_level);
if (api_level >= 19 && api_level <= 22) {
    Log.d(TAG, "level 大于19小于22");
    Settings.Secure.putInt(getContentResolver(),
        Settings.Secure.ALLOW MOCK_LOCATION, 1);
} else {
    Toast.makeText(mContext, "当前android 版本 不需要 '开启允许模拟位置' ", Toast.LENGTH_LONG).show();
    Log.d(TAG, "当前android 版本 不需要 开启允许模拟位置");
}

// 获取当前系统的版本号
Log.d(TAG, "Product Model: " + android.os.Build.MODEL
    + ", VERSION.SDK: " + android.os.Build.VERSION.SDK
    + ", Build.VERSION.RELEASE"
    + android.os.Build.VERSION.RELEASE);
API_LEVEL_NUMBER = android.os.Build.VERSION.SDK;
Log.d(TAG, "API_LEVEL_NUMBER:" + API_LEVEL_NUMBER);
```

8.4 源码实现: Stay awake -> on

```
// 4:stay awak
Settings.Global.putInt(getContentResolver(),Settings.Global.STAY_ON_WHILE_PLUGGED_IN,
    BatteryManager.BATTERY_PLUGGED_AC | BatteryManager.BATTERY_PLUGGED_USB);
```

8.5 源码实现: USB Debugging -> on

```
// 2: 打开usb调试
Settings.Global.putInt(getContentResolver(),Settings.Global.ADB_ENABLED, 1);
```

8.6 源码实现: Location 选项 -> on

```
// 打开Location
Settings.Secure.setLocationProviderEnabled(getContentResolver(),LocationManager.GPS_PROVIDER, true);
```

8.7 源码实现: WiFi 选项 -> on

```
// 1: wifi开关
WifiManager mWifiManager = (WifiManager) getSystemService(Context.WIFI_SERVICE);
mWifiManager.setWifiEnabled(true);
```

九:后续可优化项

9.1: 轻量级应用, 采用 Service 实现方式在后台运行, 手动开启和关闭, “静默”配置 CTS 自动测试环境, 或配置 **CTS 环境后自动卸载该应用**。

9.2: 针对不同 Android 软件版本设置相同 Setting 项, 添加判断当前系统版本开关, 提高 APP 健壮性。

9.3: 在优化方向 2 基础上设置判断当前测试设备是“小米盒子”or“小米电视”, 据不同产品在同一 APP 中做不同配置, 提高 APP 扩展性。

9.4: 友好界面优化

十：源码获取路径

链接：<http://pan.baidu.com/s/1geFCC5l> 密码：p3jv