

使用Python语言实现 Appium自动化测试

APPIUM WITH PYTHON

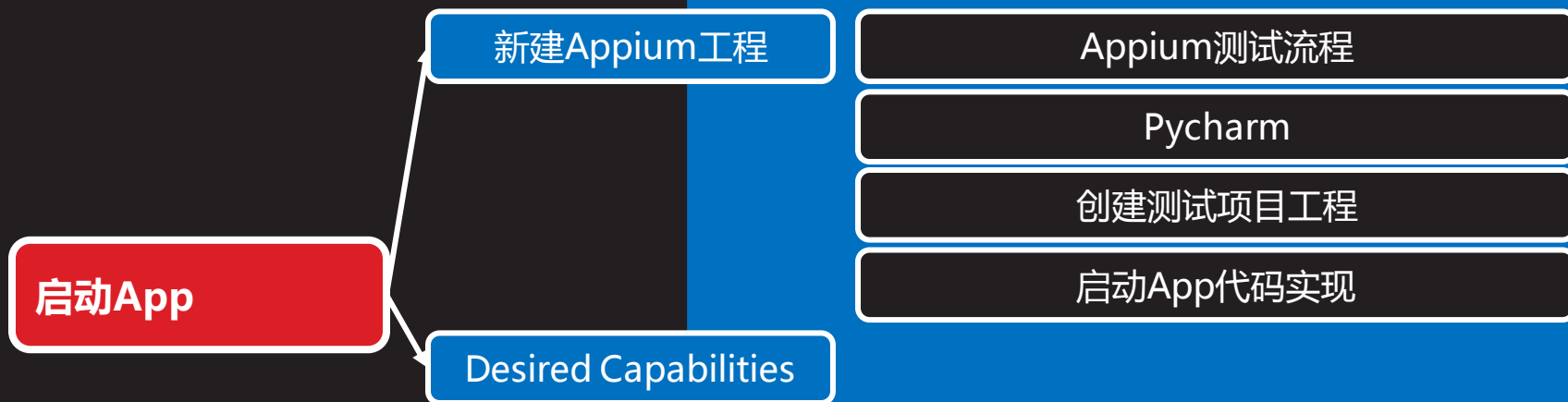
DAY03

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	启动App
	10:30 ~ 11:20	
	11:30 ~ 12:20	
下午	14:00 ~ 14:50	App启动参数
	15:00 ~ 15:50	
	16:00 ~ 16:50	
	17:00 ~ 17:30	总结和答疑



启动App



启动App (续1)

新建Appium工程

Desired Capabilities

启动App (续1)

Desired Capabilities介绍

automationName

platformName

platformVersion

deviceName

udid

app

browserName

autoWebview

appPackage

appActivity

unicodeKeyboard

resetKeyboard

新建Appium工程



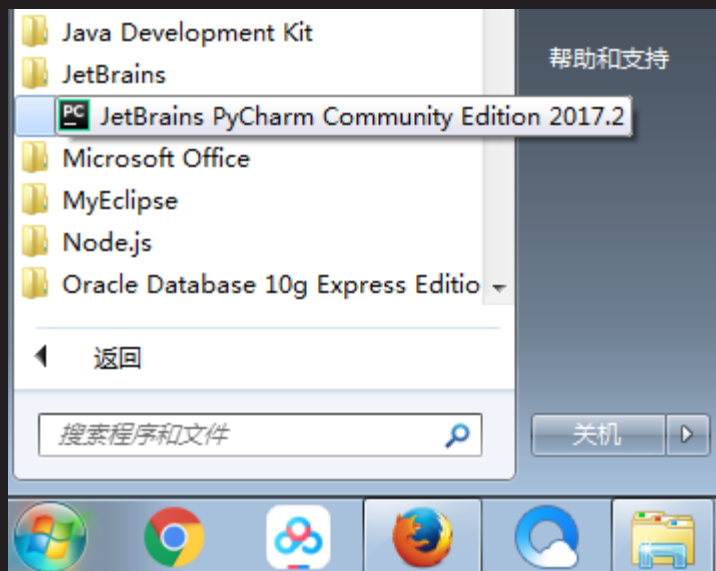
Appium测试流程

- 启动Pycharm等IDE
- 创建测试项目工程
- 启动指定设备上的指定App
- 调查App界面元素的定位方法
- 对App进行相关业务操作
- 检查结果是否符合测试需求



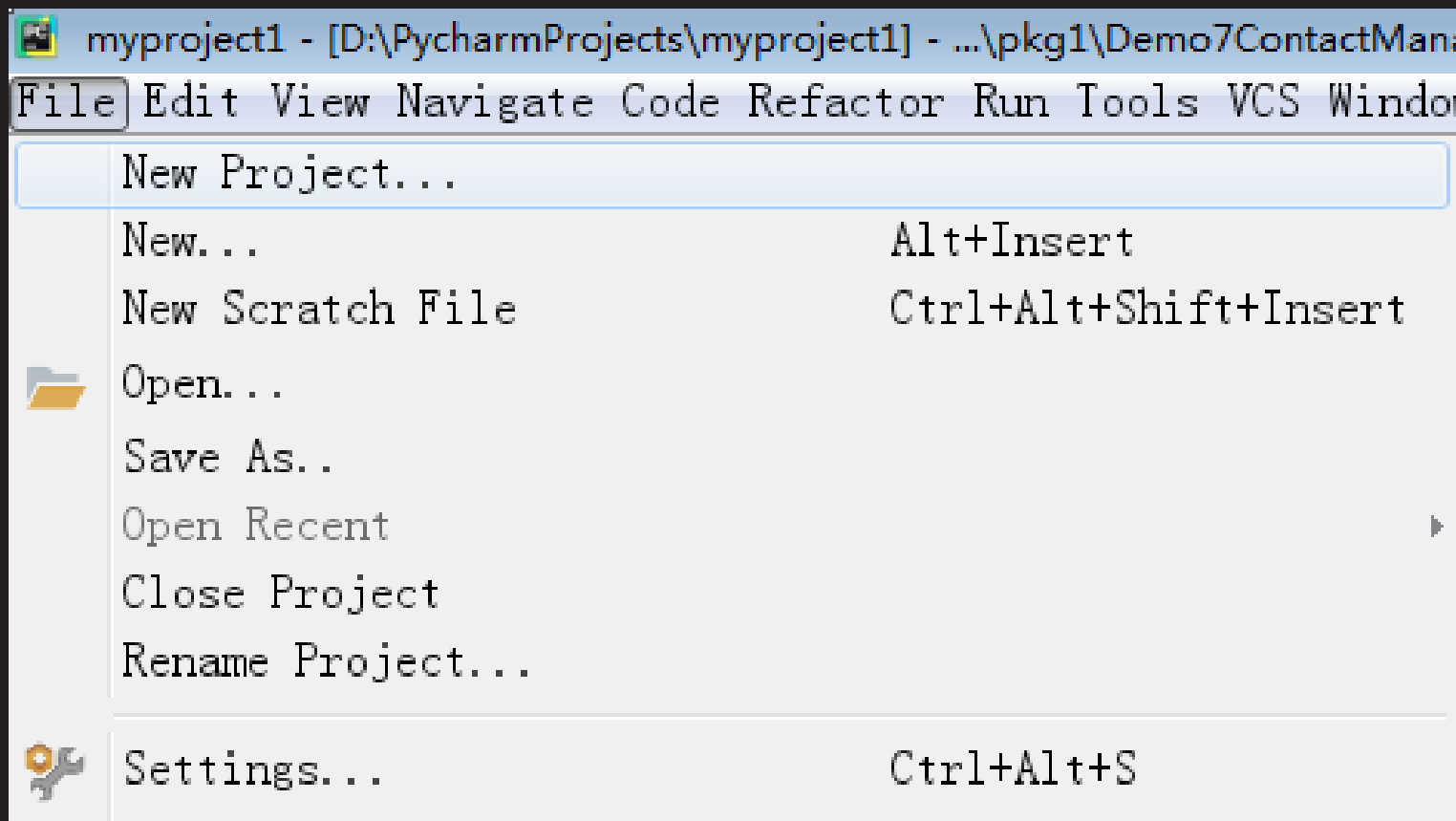
Pycharm

- PyCharm是由JetBrains打造的一款Python IDE，带有一整套可以帮助用户在使用Python语言开发时提高其效率的工具，比如调试、语法高亮、Project管理、代码跳转、智能提示、自动完成、单元测试、版本控制。
- 此外，该IDE提供了一些高级功能，以用于支持Django框架下的专业Web开发。



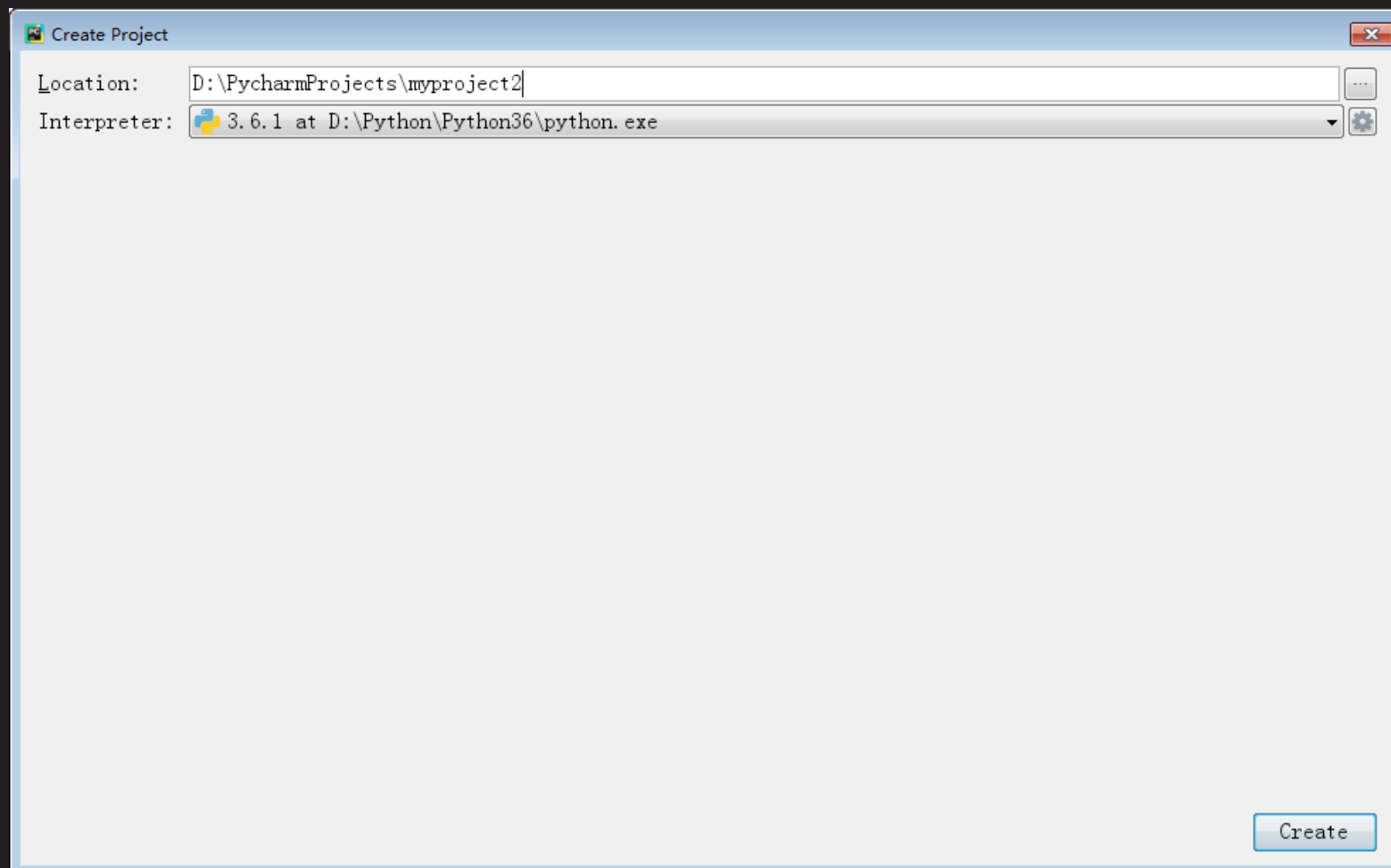
创建测试项目工程

- File=>New=>New Project



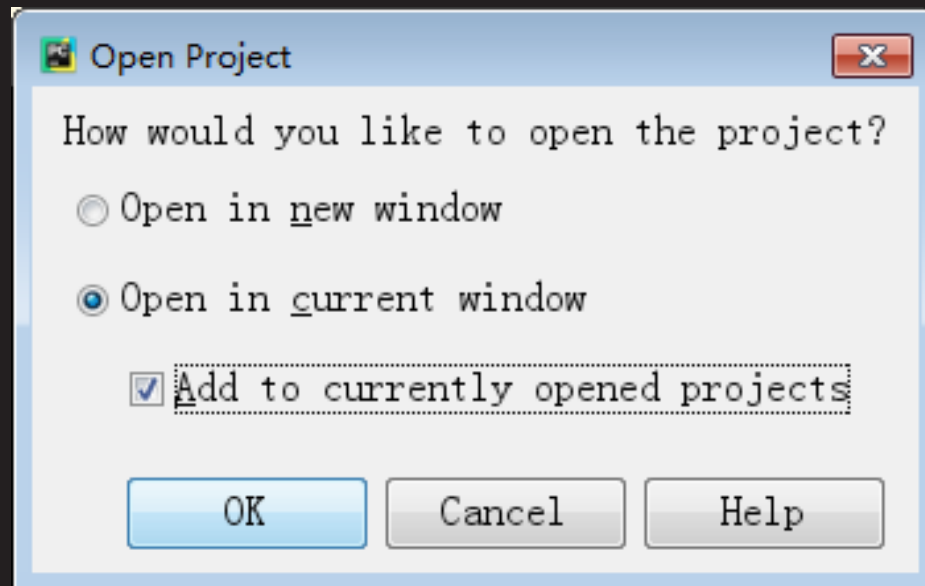
创建测试项目工程（续1）

- 输入Location，选择Interpreter为python.exe



创建测试项目工程（续2）

- 选择在哪个窗口中显示工程
 - Open in new window在新窗口中显示工程
 - Open in current window在当前窗口中显示工程
 - Add to currently opened projects添加到当前已打开工程中



启动App代码实现

- 启动App步骤：
 - 新建Desired Capabilities对象
 - 调用Remote构造方法启动App
- Desired Capabilities信息主要包括：
 - platformName：这里是android
 - deviceName：手机设备名称，通过adb devices查看
 - platformVersion：android系统的版本号
与你的模拟器或者模拟器的版本是一一对应的。
 - app：app本地路径
你的需要测试的应用的绝对路径。
 - appPackage：apk包名
 - appActivity：apk的launcherActivity



启动App代码实现（续1）

- 启动App示例：

- 启动ContactManager

```
# ! /usr/bin/env python
# -*- coding:utf-8 -*-
from appium import webdriver
desired_caps={
    "platformName": "Android",
    "platformVersion": "6.0",
    "deviceName": "Custom Phone",
    "appPackage": "com.example.android.contactmanager",
    "appActivity":
    "com.example.android.contactmanager.ContactManager"
}
driver=webdriver.Remote("http://127.0.0.1:4723/wd/hub",
desired_caps)
```



Desired Capabilities



Desired Capabilities介绍

- Desired capabilities是一些键值对（key-value）的JSON对象，其中存储初始化参数，Python里面采用字典的语法格式来实现。
- 客户端将这些键值对发给服务端，服务端接收到这些JSON信息，获知基本参数数据。
 - 比如：我们可以把platformName的capability设置为Android，就是告诉Appium服务端，我们想要一个Android的会话（session），而不是一个IOS的。
- Desired Capabilities 在启动会话（session）的时候是必须提供的。



Desired Capabilities介绍 (续1)

- 创建DesiredCapabilities两种方式：

- 方式一：新建字典时给定各个键值对

```
desired_caps={
    "platformName": "Android",
    "platformVersion": "6.0",
    "deviceName": "Android Emulator",
    "appPackage": "com.example.android.contactmanager",
    "appActivity":
    "com.example.android.contactmanager.ContactManager"
}
```



Desired Capabilities介绍（续2）

- 创建DesiredCapabilities两种方式：
 - 方式二：新建空字典，然后再添加字典中的元素

```
desired_caps={}
desired_caps["platformName"]="Android"
desired_caps["platformVersion"]="6.0"
desired_caps["deviceName"]="Android Emulator"
desired_caps["appPackage"]="com.example.android.contactma
nager"
desired_caps["appActivity"]="com.example.android.contactman
ager.ContactManager"
```



Desired Capabilities介绍 (续3)

- Desired Capabilities分为：
 - 通用Capabilities
 - Android独有Capabilities
 - IOS独有Capabilities
- 参考官方说明文档：
 - <https://github.com/appium/appium/blob/master/docs/en/writing-running-appium/caps.md>



automationName

- automationName :
 - 这个capability主要是定义自动化测试引擎。
 - 当你在安卓平台上进行测试的时候，你需要确认你使用的 android sdk版本，如果是小于17的话，你需要指定测试引擎为：Selendroid。如果大于等于17，你需要使用的引擎是：Appium，默认就是 Appium测试引擎。
 - iOS无需进行这个设置，默认就是Appium引擎。
 - 设置代码：

```
desired_caps["automationName"]="Selendroid"
```

说明：automationName一般不用设置，保持默认即可



platformName

- platformName :
 - 定义测试平台的名称，通常用于移动设备。
 - 值有：Android、iOS和FirefoxOS。
 - 在使用的过程中，请按照实际平台来填写即可。
 - 设置代码示例：


```
desired_caps["platformName"]="Android"
```



platformVersion

- platformVersion :
 - 测试平台版本，移动设备固件的版本号。
 - 比如：iOS的7.1.1，9.3等，Android的4.4.2、5.1.1、6.0等。
 - 设置代码示例：


```
desired_caps["platformVersion"]="6.0"
```



deviceName

- deviceName :
 - 移动设备的名字 , 比如iPhone 5s、 Google Nexus等
 - 设置代码示例 :

```
desired_caps["deviceName"]="Android Emulator"
```



udid

- udid :
 - 连接真机的唯一设备号
 - 设置代码示例：

```
desired_caps["udid"]="1ae203187fc012g"
```



app

- app :
 - 苹果app或者安卓app的路径，可以是本地的绝对路径，也可以是远程网络路径，只要有访问权限即可。
 - 根据这个app capabilities，Appium会在启动测试之前安装好app到设备。
 - 在测试安卓的时候，appPackage和appActivity也需要设置，和app搭配使用。
 - 本地安装apk包的文件路径设置，不需要安装就不用设置。
 - 设置代码示例：

```
desired_caps["app"]="D:\WorkspaceForAppium\ContactManager.apk"
```



browserName

- browserName :
 - 如果测试wap网站，需要设置。
 - 如果测试非web app，不用设置。
 - 对于Android来说，你可能会使用chrome浏览器或Android系统自带的浏览器。
 - 对于iOS来说，你可能要定义Safari浏览器。
 - 设置代码示例1：chrome浏览器


```
desired_caps["browserName"]="Chrome "
```
 - 设置代码示例2：Android系统自带的浏览器


```
desired_caps["browserName"]="Browser"
```



autoWebview

- autoWebview :
 - 直接进入WebView上下文。
 - 默认false。
 - 如果测试的是混合应用并且想直接进入WebView内容中，那么需要设置这个capability的值为true。
 - 一般来说这个不用设置，保持默认即可。
 - 设置代码示例：

```
desired_caps["autoWebview"]=True
```



appPackage

- appPackage :
 - 这是Android独有Capability。
 - 设置安卓app包名的capability，指定运行的Android应用的Java程序包名。
 - Android App应用这个必须配置。
 - 设置代码示例：


```
desired_caps["appPackage"]="com.example.android.contactmanager"
```



appActivity

- appActivity :
 - 这是Android独有Capability。
 - 设置要运行的app的activity（相当于一个界面或者理解成网页），比如：LoginActivity，登录的activity，可以理解为登录界面。
 - Android App应用这个必须配置。
 - 设置代码示例：


```
desired_caps["appActivity"]="com.example.android.contactmanager.ContactManager"
```



unicodeKeyboard

- unicodeKeyboard :
 - 这是Android独有Capability。
 - 是否使用unicode键盘输入
 - 如果设置为true，那么可以输入中文和特殊字符。
 - 很常用，默认 false，一般设置为true。
 - 设置代码示例：


```
desired_caps["unicodeKeyboard"]=True
```



resetKeyboard

- resetKeyboard :
 - 这是Android独有Capability。
 - 在使用了unicode输入法测试结束后，重置输入法到原有状态，也就是将键盘隐藏起来。
 - 如果单独使用，将会被忽略。
 - 默认值 false。
 - 设置代码示例：


```
desired_caps["resetKeyboard"]=True
```



启动APP

- 启动APP，详见【COOKBOOK】



App启动参数

App启动参数

获取设备标识

获取appPackage和
appActivity

获取设备标识

方法一

方法二

方法三

方法四

方法五

方法六

方法七

获取设备标识



获取设备标识

- 通过Appium测试真机，首先要确保测试机已经和电脑正确连接。
- 请确定手机与电脑通过USB数据线连接时，USB的连接方式要选择“设备文件管理选项”。



获取设备标识（续1）

- 真机测试命令行中使用adb devices命令来查看udid（设备标识），如果出现图中的信息，就说明电脑和手机已经正确连接。
- 注意：如果报5307端口被占用的错误信息，找出占用端口的程序，卸载就可以了。

C:\windows\system32\cmd.exe

Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。

C:\Users\ [redacted] > adb devices

List of devices attached

6207febc device

C:\Users\ [redacted] >



获取appPackage和 appActivity

获取appPackage和appActivity

- appPackage和appActivity 进行appium自动化测试非常重要的两个参数，我们所测试的APP不同，这两个参数肯定也是不一样的。
- 那如何快速的获取这APP的这两个参数呢？
- 例：
 - `desired_caps["appPackage"]="com.example.android.contactmanager"`
 - `desired_caps["appActivity"]="com.example.android.contactmanager.ContactManager"`



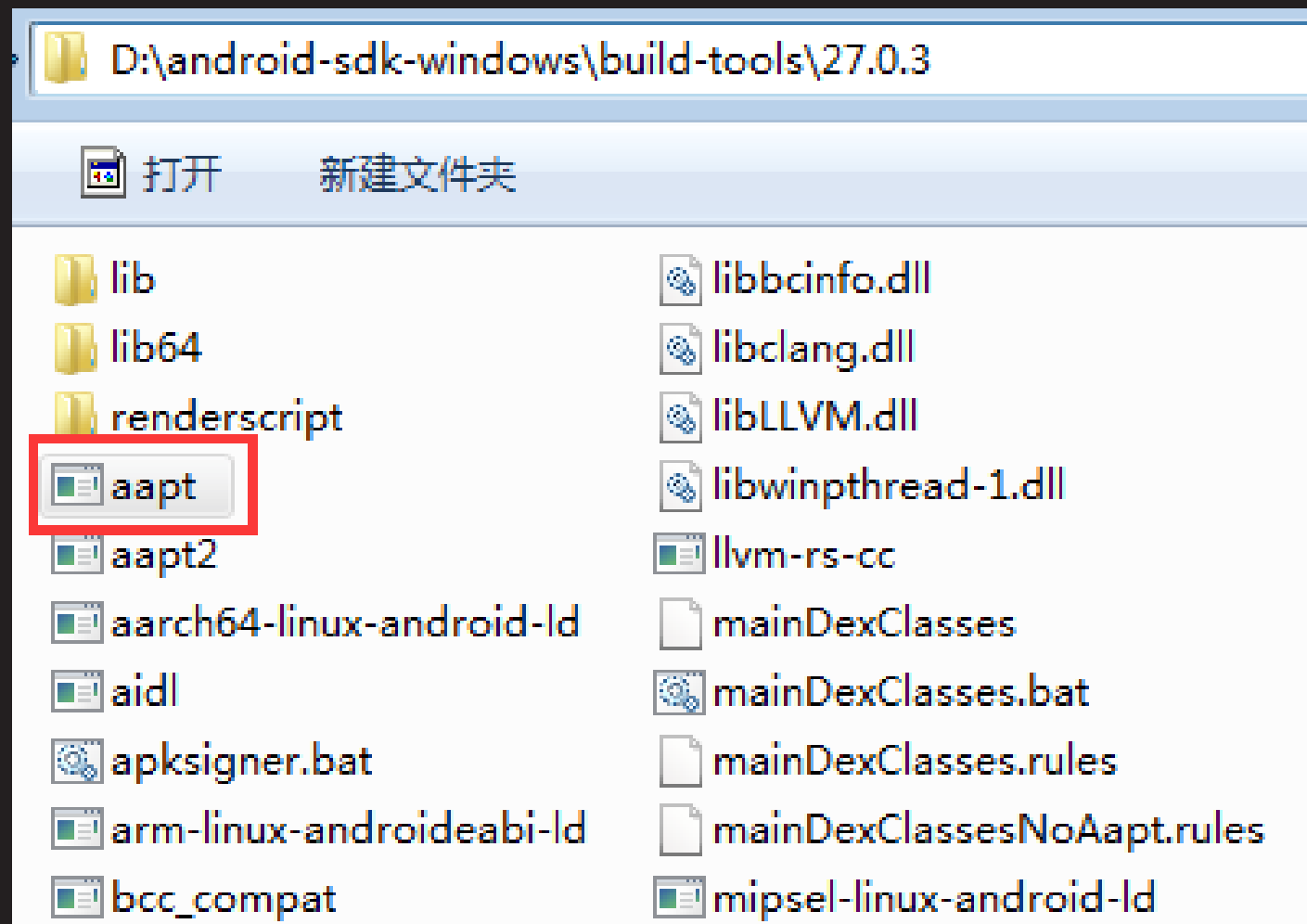
方法一

- 方法一：通过AAPT反编译来获取包名及入口
- AAPT：
 - aapt即Android Asset Packaging Tool，是Android资源打包工具。
 - 该工具在SDK的build-tools某个Android版本同名的目录下。
 - 可应用于查看apk包名、主activity、版本等很多信息，打包apk文件构成一个Android 应用程序。
- 注意：
 - 在使用aapt之前需要在环境变量里面配置SDK-tools路径，或者是路径+aapt的方式进入aapt。



方法一（续1）

- 查看aapt所在路径



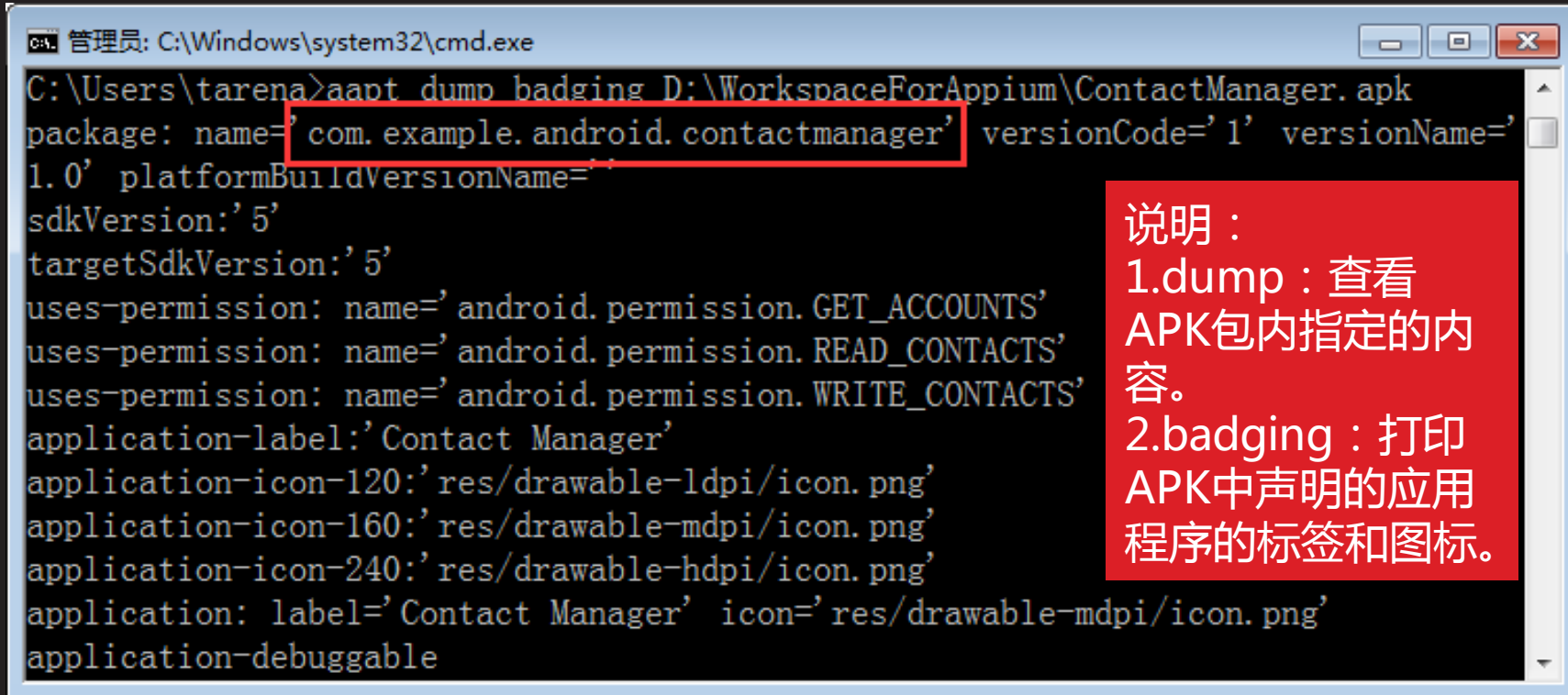
方法一（续2）

- 配置到环境变量Path中：`%ANDROID_HOME%\build-tools\27.0.3;`



方法一（续3）

- 命令行中输入aapt dump badging XXX.apk
- 例如：
 - aapt dump badging D:\ContactManager.apk



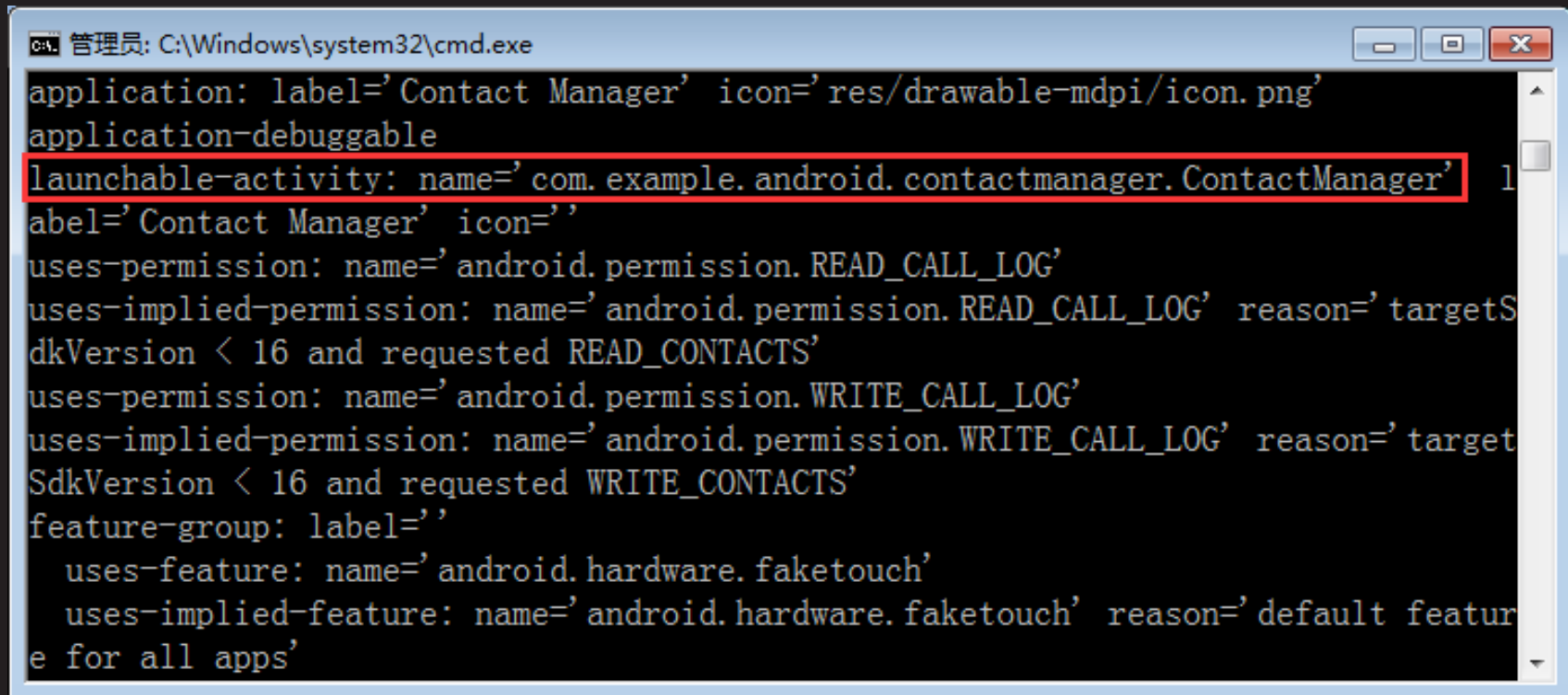
```
管理员: C:\Windows\system32\cmd.exe
C:\Users\tarena>aapt dump badging D:\WorkspaceForAppium\ContactManager.apk
package: name='com.example.android.contactmanager' versionCode='1' versionName='
1.0' platformBuildVersionName='
sdkVersion:'5'
targetSdkVersion:'5'
uses-permission: name='android.permission.GET_ACCOUNTS'
uses-permission: name='android.permission.READ_CONTACTS'
uses-permission: name='android.permission.WRITE_CONTACTS'
application-label:'Contact Manager'
application-icon-120:'res/drawable-ldpi/icon.png'
application-icon-160:'res/drawable-mdpi/icon.png'
application-icon-240:'res/drawable-hdpi/icon.png'
application: label='Contact Manager' icon='res/drawable-mdpi/icon.png'
application-debuggable
```

说明：
1.dump：查看
APK包内指定的内
容。
2.badging：打印
APK中声明的应用
程序的标签和图标。



方法一（续4）

- 查看package : name就是app包名。
- 查看Launchable-activity : name , 就是app的主Activity名称。



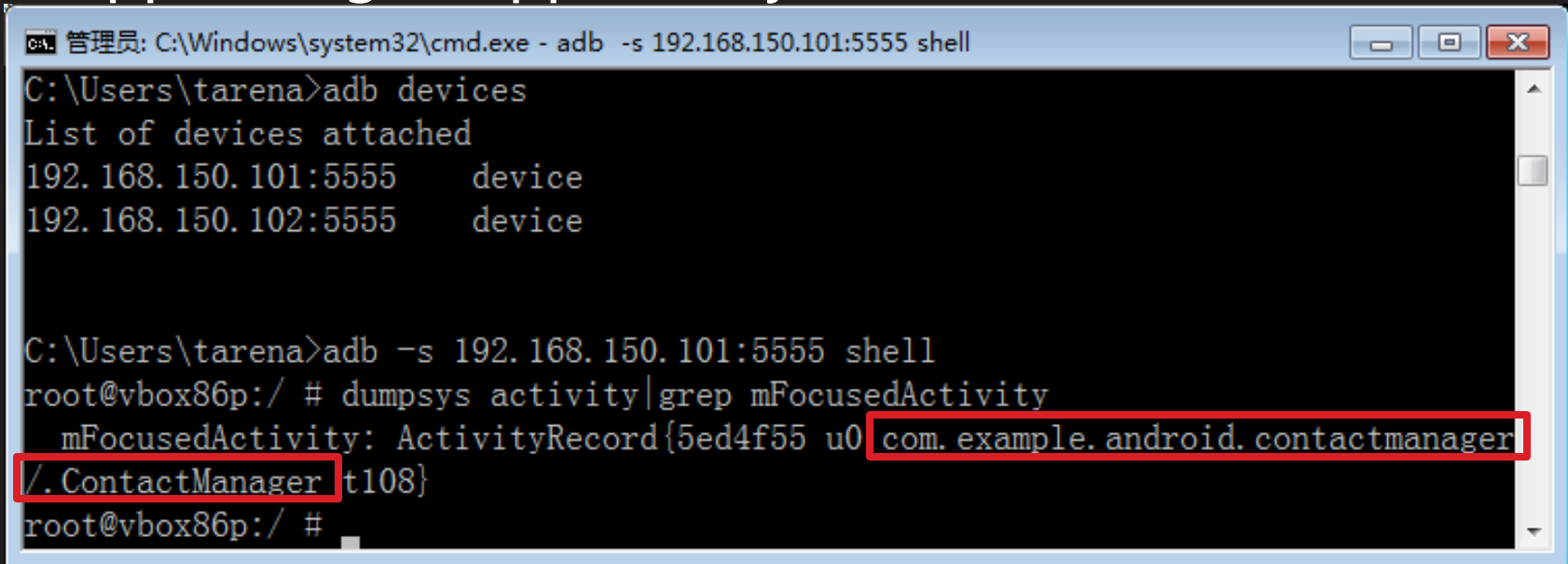
```

管理员: C:\Windows\system32\cmd.exe
application: label='Contact Manager' icon='res/drawable-mdpi/icon.png'
application-debuggable
launchable-activity: name='com.example.android.contactmanager.ContactManager'
label='Contact Manager' icon=''
uses-permission: name='android.permission.READ_CALL_LOG'
uses-implicit-permission: name='android.permission.READ_CALL_LOG' reason='targetSdkVersion < 16 and requested READ_CONTACTS'
uses-permission: name='android.permission.WRITE_CALL_LOG'
uses-implicit-permission: name='android.permission.WRITE_CALL_LOG' reason='targetSdkVersion < 16 and requested WRITE_CONTACTS'
feature-group: label=''
  uses-feature: name='android.hardware.faketouch'
  uses-implicit-feature: name='android.hardware.faketouch' reason='default feature for all apps'
  
```



方法二

- 方法二：在设备中打开App后，在命令行输入adb shell进入唯一的设备，或者加-s参数指定设备，再使用dumpsys activity | grep mFocusedActivity命令来获得appPackage和appActivity名。



```

C:\Users\tarena>adb devices
List of devices attached
192.168.150.101:5555    device
192.168.150.102:5555    device

C:\Users\tarena>adb -s 192.168.150.101:5555 shell
root@vbox86p:/ # dumpsys activity|grep mFocusedActivity
mFocusedActivity: ActivityRecord{5ed4f55 u0 com.example.android.contactmanager
/.ContactManager t108}
root@vbox86p:/ #
  
```



方法三

- 方法三：查看配置文件
 - 将XXX.apk重命名为XXX.zip或XXX.rar
 - 解压缩为一个文件夹
 - 使用记事本或写字板打开该文件夹内找到AndroidManifest.xml
 - 在里面搜索：找到**manifest** 对应的就是**appPackage** , 搜索：**activity**对应的就是**appActivity**。（ activity关键字很多，你要注意辨别。 ）



ContactManager



AndroidManifest.xml - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```

? ? ? ? ? X0 ? ? ? ? ? 4 R v ? ? ? ? ? "0 40 H0 ? ? ? ? ? V0
? ? ? ? ? ? ? R0 f0 ? ? versionCode ? versionName minSdkVersion
? targetSdkVersion ? name ? label ? icon debuggable ? andr
oid *http://schemas.android.com/apk/res/android ? pac
kage ? manifest "com.example.android.contactmanager" ? 1.
0 ? uses-sdk ? uses-permission android.permission.GET_AC
COUNTS android.permission.READ_CONTACTS ? android perm
ission.WRITE_CONTACTS ? application ? activity ? .Contact
Manager intent-filter ? action ? android.intent.action.M
AIN ? category android.intent.category.LAUNCHER ? Conta
ctAdder €00 ( 0000000000p0000 000 000 000 00 00 ? ? ? ? ?
? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
L ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?

```



方法四

- 方法四：可以适用于无apk的情况，比如Android原生自带的APP（计算器、通讯录、短信...），可以通过adb命令。
 - 打开APP
 - 命令行执行> adb logcat> D:/log.txt
 - 对APP做任意一些操作
 - Ctrl+c 结束adb命令
 - 打开log.txt文件，搜索：Displayed



方法四（续1）

管理员: C:\Windows\system32\cmd.exe

```
C:\Users\tarena>adb devices
```

```
List of devices attached
```

```
192.168.150.101:5555    device
```

```
192.168.150.102:5555    device
```

```
C:\Users\tarena>adb -s 192.168.150.101:5555 logcat>D:\log.txt
```

```
^C
```

```
C:\Users\tarena>
```



方法四（续2）

D:\log.txt

开文件

log.txt x

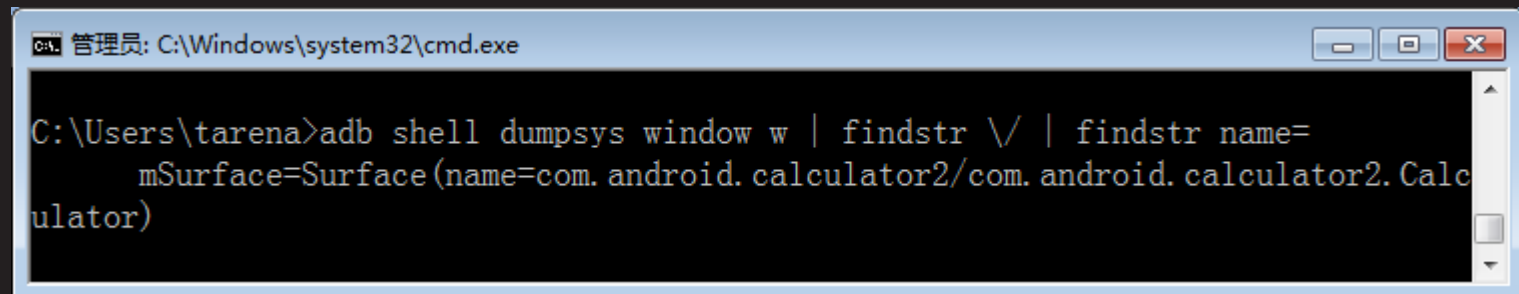
```

349 iumUnicodeIME: onStartInput
350 ivityManager: Displayed com.example.android.contactmanager/.ContactManager: +82
351 linkSocketObserver: NeighborEvent{elapsedMs=19238559, 10.0.3.2, [525400123502],
352 linkSocketObserver: NeighborEvent{elapsedMs=19259277, 10.0.3.2, [525400123502],
353 linkSocketObserver: NeighborEvent{elapsedMs=19259277, 10.0.3.2, [525400123502],
354 linkSocketObserver: NeighborEvent{elapsedMs=19278943, 10.0.3.2, [525400123502],
355 /dService: unknown message set_device_clipboard
356 linkSocketObserver: NeighborEvent{elapsedMs=19373916, 10.0.3.2, [525400123502],
357 linkSocketObserver: NeighborEvent{elapsedMs=19388573, 10.0.3.2, [525400123502],
358 ty : uid=10061 com.sankuai.meituan:dppushservice expire 423 lines
  
```



方法五

- 方法五：可以适用于无apk的情况，比如Android原生自带的APP（计算器、通讯录、短信...），可以通过adb命令。
 - 打开APP
 - 命令行执行>adb shell dumpsys window w | findstr \ / | findstr name=



```

管理员: C:\Windows\system32\cmd.exe

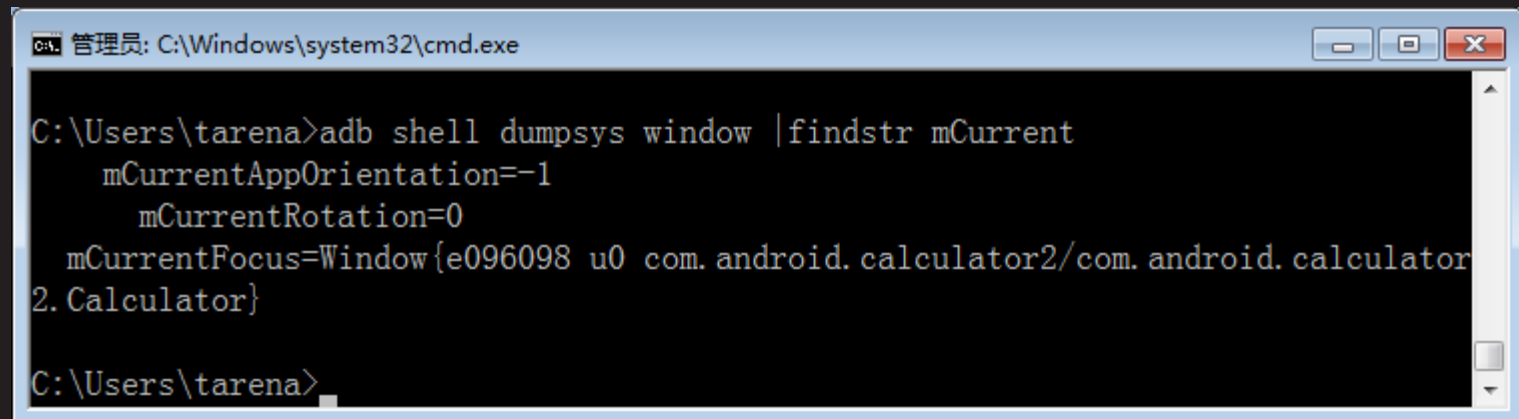
C:\Users\tarena>adb shell dumpsys window w | findstr \ / | findstr name=
        mSurface=Surface(name=com.android.calculator2/com.android.calculator2.Calculator)
    
```



方法六

- 方法六：可以适用于无apk的情况，比如Android原生自带的APP（计算器、通讯录、短信...），可以通过adb命令。

- 打开APP
- 命令行执行> adb shell dumpsys window |findstr mCurrent或者adb shell dumpsys window | findstr mCurrentFocus



```

C:\Windows\system32\cmd.exe

C:\Users\tarena>adb shell dumpsys window |findstr mCurrent
    mCurrentAppOrientation=-1
    mCurrentRotation=0
    mCurrentFocus=Window{e096098 u0 com.android.calculator2/com.android.calculator
2. Calculator}

C:\Users\tarena>
  
```



方法七

- 方法七：可以适用于无apk的情况，比如Android原生自带的APP（计算器、通讯录、短信...），可以通过adb命令获取当前所有的活动，从中挑选当前使用的活动。
 - 打开APP
 - 命令行执行> adb shell dumpsys activity activities
 - 查看realActivity



方法七（续1）

```

C:\Windows\system32\cmd.exe
C:\Users\tarena>adb shell dumpsys activity activities
ACTIVITY MANAGER ACTIVITIES (dumpsys activity activities)
Display #0 (activities from top to bottom):
  Stack #27:
    Task id #351
      * TaskRecord{97e9478 #351 A=com.android.calculator2 U=0 sz=1}
        userId=0 effectiveUid=u0a23 mCallingUid=u0a8 mCallingPackage=com.android.l
launcher3
        affinity=com.android.calculator2
        intent={act=android.intent.action.MAIN cat=[android.intent.category.LAUNCH
ER] flg=0x10200000 cmp=com.android.calculator2/.Calculator bnds=[60, 504][180, 624
]}
        realActivity=com.android.calculator2/.Calculator
        autoRemoveRecents=false isPersistable=true numFullscreen=1 taskType=0 mTask
kToReturnTo=0
        rootWasReset=true mNeverRelinquishIdentity=true mReuseTask=false mLockTask
Auth=LOCK_TASK_AUTH_PINNABLE
        Activities=[ActivityRecord{95ede05 u0 com.android.calculator2/.Calculator
t351}]
        askedCompatMode=false inRecents=true isAvailable=true
        lastThumbnail=android.graphics.Bitmap@8f1f751 lastThumbnailFile=/data/syste
m/recent_images/351_task_thumbnail.png
        stackId=27
  
```



查看指定APPappPackage和appActivity

- 练习使用AAPT来查看APP包名（ package name ）和主活动（ main Activity ），详见【COOKBOOK】



总结和答疑

