



河北师范大学软件学院
Software College of Hebei Normal University



Android 基础开发

第五章 第二讲 Intent启动内置程序



Java与移动智能设备开发



教学目标

- 掌握常见Intent启动系统组件



目录



1

使用Intent启动内置应用

2

PendingIntent



使用Intent启动内置应用

- 借助Intent对象的Action属性和Data属性、Type属性，可以启动内置的Android应用程序（如打电话、发短信、打开浏览器等等）。
- 启动内置应用程序需要使用Android提供的标准Action属性：
 - 标准Action属性请参考：

<http://developer.android.com/reference/android/content/Intent.html#constants>



使用Intent启动内置应用

- 启动内置应用程序的基本流程：
 1. 创建Intent对象；
 2. 设置Intent对象的属性（ Action、 Data、 Category 等 ）；
 3. 在AndroidManifest.xml文件中申请内置应用程序启动权限；
 4. 启动内置应用程序。



使用Intent启动内置应用

- 拨打电话：

```
Intent i = new Intent();  
i.setAction(Intent.ACTION_DIAL);  
i.setData(Uri.parse("tel:10086"));  
startActivity(i);
```

- **Intent.ACTION_DIAL**为拨打电话应用程序所匹配的动作 Action，表示打开拨打电话窗口，但还未拨出电话。
- 可以使用简写形式：

```
Intent i = new Intent(Intent.ACTION_DIAL,  
                      Uri.parse("tel:10086"));  
startActivity(i);
```



使用Intent启动内置应用

- 在使用内置拨打电话程序时，必须在AndroidManifest.xml文件中申请权限：

```
<uses-permission  
android:name="android.permission.CALL_PHONE"/>
```

- **注意**：模拟器测试需要在设置中给相应APP设置允许拨号的权限。
 1. 打开安卓模拟器的Setting；
 2. 点击Apps；
 3. 找到要运行的APP；
 4. 进去点Permissions；
 5. 把需要的权限打开。



使用Intent启动内置应用

- 发送短信：

```
Intent i = new Intent();  
i.setAction(Intent.ACTION_SENDTO);  
i.setData(Uri.parse("smsto:10086"));  
i.putExtra("sms_body", "短信内容");  
startActivity( i );
```

- 在使用发送短信程序时，必须在AndroidManifest.xml文件中申请权限：

```
<uses-permission  
android:name="android.permission.SEND_SMS"/>
```




使用Intent启动内置应用

- 打开地图：

```
Intent i = new Intent();  
i.setAction(Intent.ACTION_VIEW);  
i.setData(Uri.parse("geo:39.89,116.3"));  
startActivity(i);
```

- geo：后的参数代表当前地理位置的经度和纬度值。
- 该段代码会打开本机默认的地图软件。



使用Intent启动内置应用

- 打开Web浏览器：

```
Intent i = new Intent();  
i.setAction(Intent.ACTION_VIEW);  
i.setData(Uri.parse("http://www.baidu.com"));  
startActivity( i );
```

- 打开默认手机浏览器，以显示结果页。



使用Intent启动内置应用

- Intent可以启动的其它内置应用，请参考：

<https://www.cnblogs.com/JLZT1223/p/6796413.html>



目录



- 1 使用Intent启动内置应用
- 2 PendingIntent



PendingIntent简介

- PendingIntent，潜在意图，是在将来某一时刻由外部应用程序触发当前应用程序中Intent的特殊对象。
- PendingIntent可以看作是对Intent的包装，供当前APP之外的其它APP调用。
 - 当前APP不能使用PendingIntent启动它所包裹的Intent；
 - 外部APP使用PendingIntent，间接地调用其所包裹的Intent。
- PendingIntent主要包装了Intent对象和当前APP的上下文对象(Context)，正因为如此，外部APP才可以借助当前APP的Context来调用其内所定义的Intent对象。



PendingIntent使用方法

- 创建PendingIntent对象的主要方法有：
 - PendingIntent. **getActivity**() : 获取到用于启动Activity的PendingIntent对象；
 - PendingIntent. **getService**() : 获取到用于启动Service的PendingIntent对象；
 - PendingIntent. **getBroadcast**() : 获取到用户启动Broadcast的PendingIntent对象。



PendingIntent的应用场合

- PendingIntent的主要使用场合：
 - 发送Notification状态栏通知时，使用PendingIntent绑定用户单击状态栏通知时的激活Intent动作。
 - 使用SmsManager发送短信时，使用PendingIntent启动一个广播服务。
 - 在AlarmManager定时提醒应用中，使用PendingIntent绑定用户的行为。



内容回顾

- 1 使用Intent启动内置应用
- 2 PendingIntent



Thank you!

