# **[Bundle对象的使用](https://www.cnblogs.com/mrray/p/6178978.html)**

在Android开发中，如果要通过一个Activity启动另外一个Activity,需要调用startActivity()函数，这个函数的参数是一个Intent对象，这个对象通常的初始化方式如下：

Intent intent = new Intent();

intent.setClass(this,SecondActivity.class);

startActivity(intent);

这样就完成了一个新的Activity的启动，但是这种启动方式两个Activity之间不会有任何的数据传递，很多情况下，我们遇到的往往是前一个Activity要把数据传递给新启动的Activity,这就要用到Bundle对象了。  
比如在第一个Activity中，我们获取了身高和性别两种数据，需要传递给新启动的Activity,那么就要把这些数据封装进Bundle对象里面，再把Bundle对象assign给Intent，作为starActivity()函数的参数。  
实现代码如下：

Intent intent = new Intent();

intent.setClass(this,SecondActivity.class);

//封装Bundle对象

Bundle bundle = new Bundle();

bundle.putDouble("height",height);//height为double型变量

bundle.putString("sex",sex);//sex为string型变量

//把Bundle对象assign给Intent

intent.putExtras(bundle);

startActivity(intent);

第二个Activity相应的也要接收数据，方法也很简单，先从Intent对象中分离Bundle,再按照相同方法提取数据。  
实现代码如下：

Bundle bundle1=this.getIntent().getExtras();

String sex=bundle1.getString("sex");

double height=bundle1.getDouble("height");

值得注意的是，如果程序中有多个Activity，要在AndroidManifest.xml中声明，声明一个Activity格式如下：

<activity android:name="SecondActivity"></activity>

当存在多个Activity时，必须指定一个最先启动的Activity,也是在AndroidManifest.xml中声明，声明方式如下：

<activity android:name="SecondActivity">

<intent-filter>

<action android:name="android.intent.action.MAIN"/>

<category android:name="android.intent.category.LAUNCHER"/>

</intent-filter>

</activity>

Bundle对象还有其他很多种对不同数据类型的操作方法,比如getBoolean等，具体可以到Android官网去参考。

# Android接收短信-createFromPdu

2016年02月20日 16:40:12 [小小工匠](https://me.csdn.net/yangshangwei" \t "https://blog.csdn.net/yangshangwei/article/details/_blank) 阅读数 9939更多

分类专栏： [【移动开发-Android杂货箱】](https://blog.csdn.net/yangshangwei/article/category/5969777" \t "https://blog.csdn.net/yangshangwei/article/details/_blank)

版权声明：本文为博主原创文章，遵循[CC 4.0 BY-SA](http://creativecommons.org/licenses/by-sa/4.0/" \t "https://blog.csdn.net/yangshangwei/article/details/_blank)版权协议，转载请附上原文出处链接和本声明。

本文链接：<https://blog.csdn.net/yangshangwei/article/details/50705196>

# **createFromPdu(byte[]） method was deprecated in API level 23. Use createFromPdu(byte[], String) instead.**

Android收到短信的事件是由系统发一个有序广播的，所以这里需要一个BroadcastReceiver。receiver收到的Intent里面并不是直接存储的短信内容，而是短信的原始数据。所以我们需要自己解码。

# **获取短信的原始数据**

获取短信的原始数据： 原始数据被叫做PDU，一个PDU就是一个数据段，如果短信比较长的话可能是由几个PDU组成的。

@Overridepublic void onReceive(Context context, Intent intent) {

Bundle smsBundle = intent.getExtras();

if (smsBundle != null) {

Object[] pdus = (Object[]) smsBundle.get("pdus");

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8

上面的smsBundle.get("pdus")返回的实际类型是byte[][]，二维数组的每一个子数组就是一个pdu。

# **解码短信内容**

解码短信内容： 这里主要使用到的是SmsMessage.createFromPdu方法。

private String parseMessageFromRawData(Object[] pdus) {

if (pdus == null) return null;

try {

StringBuilder message = new StringBuilder();

for (Object pdu : pdus) {

SmsMessage smsMessage = SmsMessage.createFromPdu((byte[]) pdu);

if (smsMessage == null) continue;

message.append(smsMessage.getDisplayMessageBody());

}

return message.toString();

} catch (Exception e) {

Log.e(TAG, "SMSBroadcastReceiver read sms failed", e);

} catch (OutOfMemoryError oom) {

Log.e(TAG, "SMSBroadcastReceiver caused OOM =\_=!", oom);

//为了避免后续操作出现问题，gc一下

System.gc();

System.gc();

}

return null;

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21

上面的重点就是SmsMessage smsMessage = SmsMessage.createFromPdu((byte[]) pdu);这一句了。需要注意的是这个操作容易导致OOM（已经有好多Crash上报了。。。），所以要特别处理一下。

另外： 上面的SmsMessage.createFromPdu方法已经被官方文档说明即将被废弃（deprecated）原因是为了同时支持3GPP和3GPP2，他们是移动系统通信标准的拟定组织分别拟定了GSM/UMTS/LTE标准和CDMA/LTE标准。因此推荐是用的方法是createFromPdu(byte[] pdu, String format)其中fotmat可以是SmsConstants.FORMAT\_3GPP或者SmsConstants.FORMAT\_3GPP2。

出于好奇我看了一眼SmsMessage.createFromPdu的代码实现，发现他本身就做了制式判断的（=\_=!），代码写的还是不错的，粘出来。万一以后这个方法被废弃了，还有个参考。

public static SmsMessage createFromPdu(byte[] pdu) {

SmsMessage message = null;

// cdma(3gpp2) vs gsm(3gpp) format info was not given,

// guess from active voice phone type

int activePhone = TelephonyManager.getDefault().getCurrentPhoneType();

String format = (PHONE\_TYPE\_CDMA == activePhone) ?

SmsConstants.FORMAT\_3GPP2 : SmsConstants.FORMAT\_3GPP;

message = createFromPdu(pdu, format);

if (null == message || null == message.mWrappedSmsMessage) {

// decoding pdu failed based on activePhone type, must be other format

format = (PHONE\_TYPE\_CDMA == activePhone) ?

SmsConstants.FORMAT\_3GPP : SmsConstants.FORMAT\_3GPP2;

message = createFromPdu(pdu, format);

}

return message;

}

# **Android cursor query方法详解**

0.1032017.04.08 12:39:53字数 537阅读 2446

1.cursor query 方法入参

public final Cursor query (Uri uri, String[] projection,String selection,String[] selectionArgs, String sortOrder)

第一个参数，uri，rui是什么呢？好吧，上面我们提到了Android提供内容的叫Provider，那么在Android中怎么区分各个Provider？有提供联系人的，有提供图片的等等。所以就需要有一个唯一的标识来标识这个Provider，Uri就是这个标识，android.provider.ContactsContract.Contacts.CONTENT\_URI就是提供联系人的内容提供者，可惜这个内容提供者提供的数据很少。

第二个参数，projection，真不知道为什么要用这个单词，这个参数告诉Provider要返回的内容（列Column），比如Contacts Provider提供了联系人的ID和联系人的NAME等内容，如果我们只需要NAME，那么我们就应该使用：

Cursor cursor = contentResolver.query(android.provider.ContactsContract.Contacts.CONTENT\_URI,

new String[]{android.provider.ContactsContract.Contacts.DISPLAY\_NAME}, null, null, null);

当然，下面打印的你就只能显示NAME了，因为你返回的结果不包含ID。用null表示返回Provider的所有内容（列Column）。

第三个参数，selection，设置条件，相当于SQL语句中的where。null表示不进行筛选。如果我们只想返回名称为张三的数据，第三个参数应该设置为：

Cursor cursor = contentResolver.query(android.provider.ContactsContract.Contacts.CONTENT\_URI,

new String[]{android.provider.ContactsContract.Contacts.DISPLAY\_NAME},

android.provider.ContactsContract.Contacts.DISPLAY\_NAME + "='张三'", null, null);

结果：

11-05 15:30:32.188: I/System.out(10271): 张三

第四个参数，selectionArgs，这个参数是要配合第三个参数使用的，如果你在第三个参数里面有？，那么你在selectionArgs写的数据就会替换掉？，

Cursor cursor = contentResolver.query(android.provider.ContactsContract.Contacts.CONTENT\_URI,

new String[]{android.provider.ContactsContract.Contacts.DISPLAY\_NAME},

android.provider.ContactsContract.Contacts.DISPLAY\_NAME + "=?",

new String[]{"张三"}, null);

效果和上面一句的效果一样。

第五个参数，sortOrder，按照什么进行排序，相当于SQL语句中的Order by。如果想要结果按照ID的降序排列：

2人点赞

[android](https://www.jianshu.com/nb/10769623" \t "https://www.jianshu.com/p/_blank)

getContentResolver()怎么用？

因为在android中，每个应用程序是可以实现数据共享的，对于每一个应用程序程序都拥有一个contentprovider实例进行存储，

而contentresolver则是用于管理所有程序的contentprovider实例，通过contentrescolver可以获得数据，

插入数据等……至于getcontentrescolver()就是获取实例。

另：

用ContentProvider时会用，比如访问系统内置的一些数据库，或者自己建一个库，提供他其他应用程序调用时会用到。

Cursor c = getContentResolver().query(Phones.CONTENT\_URI, null, null, null, null);

其实应该是Cursor c =context.getContentResolver().query(Phones.CONTENT\_URI, null, null, null, null);

在Activity里用当然没问题，要是在自己定义的一个公用的类里用，就需要传Context对象了。