**[Android實戰技巧：組件間通信---Intent和IntentFilter](http://blog.csdn.net/hitlion2008/article/details/6966569)**

分類： [Android](http://blog.csdn.net/hitlion2008/article/category/664350) 2011-11-13 20:47 8682人閱讀 [評論](http://blog.csdn.net/hitlion2008/article/details/6966569#comments)(2) [收藏](javascript:void(0);) [舉報](http://blog.csdn.net/hitlion2008/article/details/6966569#report)

[android](http://www.csdn.net/tag/android)[action](http://www.csdn.net/tag/action)[scheme](http://www.csdn.net/tag/scheme)[service](http://www.csdn.net/tag/service)[file](http://www.csdn.net/tag/file)[string](http://www.csdn.net/tag/string)

目錄[(?)](http://blog.csdn.net/hitlion2008/article/details/6966569" \o "系统根据文章中H1到H6标签自动生成文章目录)[[+]](http://blog.csdn.net/hitlion2008/article/details/6966569)

**Understanding Intent and IntentFilter--理解Intent和IntentFilter**

Intent(意圖)在Android中是一個十分重要的元件，它是連接不同應用的橋樑和紐 帶，也是讓元件級複用(Activity和 Service)成為可能的一個重要原因。Intent的使用分為二個方面一個是發出Intent，另一個則是接收Intent用官方的說法就是 Intent Resolving。本主將對Intent和IntentFilter進行一些介紹。  
Intent和IntentFilter是Android和一種消息通信機制，可以讓系統的元件之間進行通信。資訊的載體就是Intent，它可以是一個 要完成的動作請求，也可以一般性的消息廣播，它可以由任意一個元件發出。消息，也就是Intent，最終也是要被元件來進行處理和消化。消息由元件發出， 通常在消息的裡面也會有會標記有目標群組件的相關資訊，另外目標群組件也需要告訴系統，哪些消息是它所感興趣的，它需要設置一些篩檢程式，以過濾掉無關的消息。  
其實這裡就好比學校裡的廣播，廣播有時會播放通知，但有時也會播放要執行的動作，比如打掃衛生。消息中通常都會說明消息的目標物件，可能是電腦學院，可 能是老師，也可能是英語系的人才需要關注。而每個人，或是學院組織，也只關心與他們有關的消息，當然這裡就要他們自己去判斷哪些是與他們有關的消息了。在 Android當中消息就是Intent，篩檢程式就是IntentFilter。發出消息的元件可以在消息中設置目標群組件的相關資訊，目標群組件也可以設置 篩檢程式，然後系統會進行過濾，只把元件所感興趣的消息，傳遞給元件。這裡假設讀者已經瞭解Intent和IntentFilter的基本使用方法，且並不 會進行全面的介紹，如果不瞭解，可以先讀讀[官方文檔](http://developer.android.com/guide/topics/intents/intents-filters.html)，這裡重點講講IntentFilter在使用時的一些注意事項。  
Intent消息機制通常有二種，一個是顯式Intent（Explicit Intent），另一個是隱式Intent（Implicit Intent）。

* 顯式Intent--需要在Intent中明確指定目標群組件，也就是在Intent 中明確寫明目標群組件的名稱（Component name），需要指定完整的包名和類名。因為對於本程式以外的其他應用程式，你很難知道它的元件名字，所以顯式的Intent通常用於應用程式內部通信， 更確切的說，顯示Intent是用於應用程式內部啟動元件，通常又是Activity或Service。還沒有見用顯式Intent來給 BroadcastReceiver發送廣播的。
* 隱式Intent--也就是不在Intent中指定目標群組件，在Intent中不能含有目標的名字。系統是根據其他的資訊，比如Data，Type和Category去尋找目標群組件。

隱式Intent通常用於與應用程式外部的元件進行通信。應用程式級別的元件複用也主要是靠 隱式Intent來完成的。而IntentFilter也是只有隱式Intent才用的著，顯式Intent都是直接把Intent傳遞給目標群組件，根本 不會理會組件的IntentFilter。

**顯式Intent（Explicit Intent)**

顯示Intent使用起來比較簡單，只需要在Intent中指定目標群組件的名字即可，也就是 通過Intent的方法設置Component屬性。如前所述，顯式Intent通常用於應用程式內部啟動Activity或Service。事實上，並 不完全局限在應用程式內部，對於外部應用的Activity和Service，也可以用顯式Intent來啟動，但你必須知道它們的名字。  
設置元件的名字的方法有：

**[java]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. public Intent setComponent(ComponentName component)；
2. public Intent setClass(Context packageContext, Class<?> cls)；
3. public Intent setClassName (Context packageContext, String className)；
4. public Intent setClassName (String packageName, String className)；

事 實上雖然設置的方法有這麼多，但Intent內部標識目標群組件的屬性只有一個Component，所以這麼設置方法的目的也只是設置目標群組件的 Component，事實上有這麼多的方法原因在於ComponentName的構造是多重載了的。在解析Intent時，系統也是取得這個 Component屬性，然後去啟動它。  
ComponentName Intent.getComponent();  
對於應用程式內部啟動Activity通常是這樣子設置Intent：

**[java]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. Intent i = new Intent();
2. // Select one of them
3. i.setComponent(new ComponentName(getApplication(), ViewStubDemoActivity.class));
4. i.setComponent(new ComponentName(getApplication(), ViewStubDemoActivity.class.getName()));
5. i.setComponent(new ComponentName(getApplication().getPackageName(), ViewStubDemoActivity.class.getName()));
6. i.setClass(getApplication(), ViewStubDemoActivity.class);
7. i.setClassName(getApplication(), ViewStubDemoActivity.class.getName());
8. i.setClassName(getApplication().getPackageName(), ViewStubDemoActivity.class.getName());
9. startActivity(i);

因為應用程式內部的元件類，都是可以訪問到的，所以要盡可能少寫字串常量，以減少拼寫錯誤，如果一定要使用包名和類名，也要注意，類名必須是全稱，也就是從包名開始，如“com.hilton.networks.WifiManagerActivity"。  
但是對於外部應用程式的Activity，通常只能通過以下方法：

**[java]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. Intent i = new Intent();
2. // select one of them
3. i.setComponent(new ComponentName("com.hilton.networks", "com.hilton.networks.WifiManagerActivity"));
4. i.setClassName("com.hilton.networks", "com.hilton.networks.WifiManagerActivity");
5. startActivity(i);

首 先，帶有Context為參數的是不能夠用的，因為通常你無法拿到其他應用程式的Context，你只能拿到你所在應用程式的Context，所以用你所 在的應用程式的Context去啟動外部的Activity肯定會報錯的。其次，不參再像上面那樣通過Class.getName()去指定類名，你為你 無法導入外部的類，會有編譯錯誤的。另外，千萬要注意不要拼錯，否則會有RuntimeException拋出的。  
對於Service元件，也是一樣，Intent的寫法與Activity元件一致，但是對於BroadcastReceiver元件通常都用顯式Intent。

**隱式Intent的消息篩檢程式--IntentFilter**

IntentFilter是用來解析隱式Intent(Implicit Intent)的，也就是說告訴系統你的元件(Activity, Service, BroadcastReceiver)能夠處理哪些隱式的Intent。在使用的時候我們通常是這樣子的：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <manifest ...>
2. <receiver ...>
3. <intent-filter>
4. <action android:name="android.appwidget.action.APPWIDGET\_UPDATE" />
5. <action android:name="android.appwidget.action.APPWIDGET\_ENABLED" />
6. <action android:name="android.appwidget.action.APPWIDGET\_DISABLED" />
7. <action android:name="android.appwidget.action.APPWIDGET\_DELETED" />
8. </intent-filter>
9. <intent-filter>
10. <action android:name="android.intent.action.MEDIA\_MOUNTED"/>
11. <action android:name="android.intent.action.MEDIA\_UNMOUNTED"/>
12. <action android:name="android.intent.action.MEDIA\_SHARED"/>
13. <action android:name="android.intent.action.MEDIA\_REMOVED"/>
14. <action android:name="android.intent.action.MEDIA\_EJECT"/>
15. <data android:scheme="file" />
16. </intent-filter>
17. <intent-filter>
18. <action android:name="android.intent.action.PACKAGE\_ADDED"/>
19. <action android:name="android.intent.action.PACKAGE\_REMOVED"/>
20. <action android:name="android.intent.action.PACKAGE\_DATA\_CLEARED"/>
21. <data android:scheme="package" />
22. </intent-filter>
23. </receiver>
24. </manifest>

在Manifest中使用IntentFilter時要注意以下三點：  
1. 千萬注意拼寫錯誤  
這裡有一個需要十分小心和注意的地方那就是對於IntentFilter裡面的Action和Data字串常量不要寫錯，因為這個在編譯時是不會被檢查， 在運行時又不會拋出異常，如果你拼寫錯了，比如大小寫拼錯了，在編譯時和運行時都不會有錯誤，但是你的程式卻不能正常工作，你的程式無法收到相應的 Intent。曾有一個同事在IntentFilter中寫了一些Action，但把其中一個的大小寫拼錯了，結果花了他一個下午的時間來調試，最後還是 另外一個同事到他那聊天才發現了是大小寫的拼寫錯誤。  
這裡也可以發現Android在Manifest檔中的IntentFilter這塊的封裝性很差。如果，僅僅是如果，這些Action常量也可以通過 引用的方式來寫，就可以在編譯時做些檢查和匹配，可以大大的減少出錯的機率，同時也加強了封裝和資訊隱藏。比如，對於上面的可以寫成這樣：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <manifest ...>
2. <receiver ...>
3. <intent-filter>
4. <action android:name="@android:action/AppWidgetManager.APPWIDGET\_UPDATE" />
5. <action android:name="@android:action/AppWidgetManager.APPWIDGET\_ENABLED" />
6. <action android:name="@android:action/AppWidgetManager.APPWIDGET\_DISABLED" />
7. <action android:name="@android:action/AppWidgetManager.APPWIDGET\_DELETED" />
8. </intent-filter>
9. </receiver>
10. </manifest>

雖 然這種拼寫錯誤很低級，但是因為它低級所以當程式不能正常工作時沒有人會想到是因為拼寫錯誤，所以這種拼寫錯誤通常會耗費不少的調試時間。另外一種避免此 種錯誤的方法就是在代碼中通過Context.registerReceiver(BroadcastReceiver,IntentFilter)來注 冊BroadcastReceiver，就可以直接寫入常量，而非具體字串。但這只能是接收Broadcast的時候，對於那些想作為公開介面的元件，還 是需要在Manifest裡面聲明，比如Email，它要能處理Intent.ACTION\_SEND\_TO，就需要在Manifest中聲明。2. 要注意Data欄位除了上面討論的之外，對於IntentFilter還有另外的一點需要注意，就是對於某些Action是需要加上Data欄位資訊，否 則有可能接收不到。比如：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <manifest ...>
2. <receiver ...>
3. <intent-filter>
4. <action android:name="android.intent.action.MEDIA\_MOUNTED"/>
5. <action android:name="android.intent.action.MEDIA\_UNMOUNTED"/>
6. <action android:name="android.intent.action.MEDIA\_SHARED"/>
7. <action android:name="android.intent.action.MEDIA\_REMOVED"/>
8. <action android:name="android.intent.action.MEDIA\_EJECT"/>
9. <data android:scheme="file" />
10. </intent-filter>
11. <intent-filter>
12. <action android:name="android.intent.action.PACKAGE\_ADDED"/>
13. <action android:name="android.intent.action.PACKAGE\_REMOVED"/>
14. <action android:name="android.intent.action.PACKAGE\_DATA\_CLEARED"/>
15. <data android:scheme="package" />
16. </intent-filter>
17. </receiver>
18. </manifest>

對 於手機外部存儲卡的狀態變化的Broadcast，在註冊監聽器的時候就需要加上DataScheme，否則就會接收不到。這個也花費了我幾個小時的調試 時間，改在代碼中用Context.registerReceiver(BroadcastReceiver,IntentFilter)註冊也不行，最 後參考了Music中的做法，加上了DataScheme才能在onReceive()中接收到Intent。同樣對於後面的Package相關的 Broadcast，也是要加上DataScheme否則也是接收不到Broadcast。可悲的是對於像這樣的系統公共的Broadcast  
Intent，在Intent的文檔中並沒有說明如何使用，如果沒有參考事例，相信需要一定的時間才能夠找出為什麼接收不到Intent。  
除了DataScheme還有一個是MimeType，這個對於系統公共介面是必須加上的，比如Email要處理Intent.ACTION\_SENTTO，就需要這樣聲明：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <manifest ...>
2. <activity android:name="EmailComposer">
3. <intent-filter>
4. <action android:name="android.intent.action.SEND"/>
5. <data android:mimeType="image/\*" />
6. </intent-filter>
7. </activity>
8. </manifest>

3. 同時也要注意Category欄位  
如果沒有對IntentFilter寫正確的Category欄位，也是收不到Intent。比如：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <manifest ...>
2. <receiver ...>
3. <intent-filter>
4. <action android:name="com.hilton.controlpanel.action.BUTTON\_ACTION" />
5. <category android:name="com.hilton.controlpanel.category.SELF\_MAINTAIN" />
6. </intent-filter>
7. </receiver>
8. </manifest>

如果把Category去掉，死活也接收不到Intent，當然這要取決於Intent是如何發出的，如果Intent發出時沒有加Category，那就沒有必須在IntentFilter加上Category。  
總之，對於Intent，要保證發出和接收完全一致，否則系統就無法找到相應的匹配，程式也就無法接收Intent。  
有關於 DEFAULT category，也要注意，如果是針對Activity的Implicit Intent隱式Intent，如果在沒有其他Category的情況下，一定要加上DEFAULT Category。因為系統會在Context.startActivity(Intent)和 Context.startActivityForResult(  
Intent)時給Intent加上DEFAULT category。而對於 Context.sendBroadcast(Intent)，Context.sendOrderedBroadcast(Intent)，Contxt.sendStickyBroadcast(Intent) 和Context.bindService(Intent)Context.startService(Intent)就不會加DEFAULT Category。  
另外要注意，儘量把Action進行合併寫進一個IntentFilter中。因為對於每個IntentFilter標籤都會創建一個 IntentFilter物件，所以如果寫幾個就會有幾個物件在那，不但耗費資源而且在匹配的時候也會耗費更多的時間，因為在查詢匹配的時候是要一個 IntentFilter物件接著一個IntentFilter物件進行檢查的。直到找到最佳匹配或是到所有的IntentFilter都檢查完為止。

**IntentFilter的匹配規則**

1. 通過Action欄位來匹配這個是Intent中比較基本的一個欄位，也比較簡單，就是一個字串，如果相等就匹配成功，否則證明還沒找到目標。但要注意， 如果IntentFilter沒有指定Action，那麼它不會匹配到任何的隱式Intent，它只能被顯式的Intent匹配上。反過來，如果 Intent自己沒有指定Action，那麼它能匹配上含有任何Action的IntentFilter，但不能匹配上沒有指定Action的 IntentFilter。對於Action，平時要注意拼寫錯誤，因為在AndroidManifest檔中聲明Action都是字串，並且在編譯時 不會做檢查，運行時，如果Action拼錯了導致匹配不上，要麼是程式不能正常工作，要麼會有異常拋出。

2. 通過Category欄位來匹配對於Activity來講，如果想處理隱式Intent，並且除了Intent.ACTION\_MAIN以外，必須指定 Category為DEFAULT，否則不會被匹配到。因為Context.startActivity()和 Context.startActivityForResult()會自動加上DEFAULT Category。其他情況，Service和BroadcastReceiver則不會，對於Service和BroadcastReceiver，如 果Intent中沒有指定Category，那麼在其IntentFilter中也不必指定。

3. 通過Data欄位來匹配這個相對來講比較複雜，通常Data包含uri, scheme(content, file, http)和type(mimeType)對於Intent來講有二個方法：

**[java]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. Intent.setData(Uri); //一個Uri，Scheme包含在其中
2. Intent.setType(String); //指定MimeType，比如'image/jpeg', 'audio/mpeg'等
3. Intent.setDataAndType(Uri, String); //上面二個方法的簡便調用方式，一起搞進去

對於IntentFilter來講，需要設置的是Scheme和Type，Scheme是對Uri的限制，目標需要限制Scheme是因為Scheme能告訴目錄能從哪裡拿到Uri所指向的檔，Type是MimeType對類型的限制。

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <intent-filter>
2. <action android:name="android.intent.action.SEND" />
3. <category android:name="android.intent.category.DEFAULT" />
4. <data android:scheme="content" android:mimeType="image/\*" />
5. </intent-filter>

Data匹配時的規則一共有四條：

a.如果Intent沒有指定Data相關的欄位，只能匹配上沒有 指定Data的IntentFilter。也就是說如果一個Intent沒有指定任何的Data（Uri和Type），它只能匹配到沒有指定任何 Data（Scheme和Type）的IntentFilter。

b.如果一個Intent只指定了Uri但是沒有Type（並且 Type也不能夠從Uri中分析出）只能匹配到僅指定了相應Scheme且沒有指定Type的IntentFilter。實際的例子有如果一個 Intent是想要發郵件，或是打電話，它們的Intent是類似這樣的："mailto:someone@sb.com" 和"tel:1234567"。換句話說，這些Uri本身就是資料，而不再是一個指向資料的位址。比如：Phone中的Dialer就有如下的 IntentFilter：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <intent-filter>
2. <action android:name="android.intent.action.CALL" />
3. <category android:name="android.intent.category.DEFAULT" />
4. <data android:scheme="tel" />
5. </intent-filter>

再如，要處理SD狀態變化的IntentFilter：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <intent-filter>
2. <action android:name="android.intent.action.MEDIA\_MOUNTED"/>
3. <action android:name="android.intent.action.MEDIA\_UNMOUNTED"/>
4. <action android:name="android.intent.action.MEDIA\_SHARED"/>
5. <action android:name="android.intent.action.MEDIA\_REMOVED"/>
6. <action android:name="android.intent.action.MEDIA\_EJECT"/>
7. <category android:name="android.intent.category.DEFAULT" />
8. <data android:scheme="file" />
9. </intent-filter>

再如，要處理Package狀態變化的IntentFilter：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <intent-filter>
2. <action android:name="android.intent.action.PACKAGE\_ADDED"/>
3. <action android:name="android.intent.action.PACKAGE\_REMOVED"/>
4. <action android:name="android.intent.action.PACKAGE\_DATA\_CLEARED"/>
5. <category android:name="android.intent.category.DEFAULT" />
6. <data android:scheme="package" />
7. <intent-filter>

但是注意，對於想對資料進行操作的Intent，最好不要只指定Uri，而不指定類型。因為如果這樣做通常會匹配到一大堆

c. 如果一個Intent只指定了Type，但是沒有指定Uri，它只能匹配到只指定了相應Type且沒有指定Scheme的IntentFitler

d. 如果一個Intent即有Uri又有Type，那麼它會匹配上：1).Uri和Type都匹配的IntentFilter；2).首先Type要匹配，另 外如果Intent的Uri是content:或file:，且IntentFilter沒有指定Scheme的IntentFilter。因為對於 Android來講content和file這二種Scheme是系統最常見也是用的最多的，所以就當成缺省值來對待。

另外需要注意，Type，因為是MimeType，所以是允許使用萬用字元的，比如'image/\*'，能匹配上所有以'image'為開頭的類型，也說是說能匹配上所有的圖像。

**根據Data匹配的例子**

假如系統中有四個Activity，A的IntentFilter是這樣子的：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <activity ...>
2. <intent-filter>
3. <action android:name="android.intent.action.SEND" />
4. <category android:name="android.intent.category.DEFAULT" />
5. <data android:scheme="content" android:mimeType="image/\*" />
6. </intent-filter>
7. </activity>

這表明A可以發送一切圖片類型，並且內容必須是由ContentProvider提供的，也就是Uri必須是以"content://"開頭的  
而另外一個Activity B是這樣子聲明的：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <activity ...>
2. <intent-filter>
3. <action android:name="android.intent.action.SEND" />
4. <category android:name="android.intent.category.DEFAULT" />
5. <data android:scheme="file" android:mimeType="image/\*" />
6. </intent-filter>
7. </activity>

這表明B可以發送一切圖片，但內容必須是單獨的一個檔，也就是Uri必須是由"file://"開頭的  
還有一個C是這樣子聲明的：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <activity ...>
2. <intent-filter>
3. <action android:name="android.intent.action.SEND" />
4. <category android:name="android.intent.category.DEFAULT" />
5. </intent-filter>
6. lt;/activity>

這表明C只能接收那些沒有指定任何Uri和Type的Action是SEND的Intent。  
而D是這樣子聲明的：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <activity ...>
2. <intent-filter>
3. <action android:name="android.intent.action.SEND" />
4. <category android:name="android.intent.category.DEFAULT" />
5. <data android:mimeType="image/\*" />
6. </intent-filter>
7. </activity>

這表明D可以發送一切圖片，無論是資料庫內的(content)，還是單獨的檔(file)。  
如果一個Intent是這樣寫的：

**[java]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. Intent share = new Intent(Intent.ACTION\_SEND);
2. startActivity(share);

那麼它只能匹配C，因為C沒有指定資料和類型，Action是SEND，根據規則a，它只能匹配Activity A。但如果給Intent加上額外的條件

**[java]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. share.setDataAndType(uri,"image/jpeg");

那麼如果uri是資料庫內容，它會匹配到A，如果它是一個檔，會匹配到B。但無論是content還是file都會匹配到D，因為它能處理以任何形式存儲的圖片。但始終不會匹配到C，因為C沒有聲明Data欄位，所以不會匹配上。  
所以，通常想把元件作為系統公用介面時都是這樣子來寫：

**[html]** [view plaincopyprint?](http://blog.csdn.net/hitlion2008/article/details/6966569)

1. <activity ...>
2. <intent-filter>
3. <!-- implement public actions such as View, Edit, Pick or Send -->
4. <action android:name="android.intent.action.SEND" />
5. <!-- never forget default category, otherwise your activity never receives intents -->
6. <category android:name="android.intent.category.DEFAULT" />
7. <!-- specify mimeType to constrain data type, receive data from both content provider and file -->
8. <data android:mimeType="image/\*" />
9. <!-- specify scheme to constrain data source, if necessary -->
10. <data android:shceme="http" />
11. </intent-filter>
12. lt;/activity>

Intent 和IntentFilter對於元件Activity來講注意事項比較多，但是對於Service和BroadcastReceiver來說就沒有那麼多 的注意事項了，因為對於Service和BroadcastReceiver通常都不用設置Category和Data。但也有例外，比如前面所講到的 SD相關廣播和應用程式安裝相關廣播。  
另外要注意，如果使用Context.startActivity()或 Context.startActivityForResult()，Context.bindService()和 Context.startService()，如果系統沒有為Intent匹配到目標Activity和Service那麼會有 RuntimeException(ActivityNotFoundException)拋出;如果有多個目標同時匹配，會以清單的方式來讓使用者選擇使 用哪個。

**使用IntentFilter匹配來進行查詢可用的元件**

Intent和IntentFilter不但可以用來進行元件複用，還可以用於查詢系統內都 有哪裡元件能做哪些事情。比如Launcher上面會列出很多的應用，其實這種說法不準確，應該是上面列出了所有的能啟動一個應用的元件（比 如，Dialer和Contacts同屬於一個應用程式Contacts中，但是在Launcher裡面卻有二個，一個是Dialer一個是 Contacts。那麼Launcher是如何做到的呢？它不可能是去檢查系統檔，看看哪些應用程式檔案存在，然後再列出來。它是通過查詢Intent 的方式，把所有含有"android.intent.action.MAIN"和"android.intent.category. LAUNCHER"的Activity的相關資訊都取出來，然後列出它們的名稱和Icon。同樣，我們也可這樣來獲得具體相應特徵的元件，具體的請參考 SDK中的一篇[文章](http://developer.android.com/resources/articles/can-i-use-this-intent.html)(Resources->Articles->Can I Use this Intent?)，講的很詳細，且有Sample Code。