手机平台应用开发实验报告

学号： 1433198 **班级 ：** 周五下午班

姓名： 黄建武 **实验名称：** Lab5

一． 参考资料

https://developer.android.com/guide/index.html Android 官方文档

http://blog.csdn.net/djcken/article/details/7801966 EditText 改变边框颜色

http://blog.csdn.net/silenceburn/article/details/6093074 widget 开发实例

二． 实验环境

系统： Windows 10

工具： Android Studio 2.1.3

JDK： 1.8

minSdk： 19

targetSdk：24

三． 实验目的

1. 掌握 AppWidget 编程基础

2. 掌握 Broadcast 编程基础

3. 掌握动态注册 Broadcast 和静态注册 Broadcast

四． 实验步骤

1. 阅读实验要求文档。

2. 运行 TA 的 Demo 程序，了解实验要求细节。

3. 复用 Lab4 的代码,在原先的基础上进行功能的添加。

   新建一个 widget.xml 的布局文件，添加一个 ImageView 和 TextView。然后完成该布局文件的 provider 文件,按照实验的细节要求设置字体大小等的细节要求,完成 Widget 的属相要求。

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:orientation="horizontal"
              android:gravity="center_vertical"
              android:layout_width="match_parent"
              android:layout_height="match_parent">

    <ImageView
        android:id="@+id/widget_img"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@mipmap/orange"
        android:layout_marginEnd="10dp"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ff0000"
        android:textSize="20dp"
        android:id="@+id/widget_text"
        android:text="Widget"/>

</LinearLayout>
```

```
<appwidget-provider
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:initialKeyguardLayout="@layout/widget"
    android:initialLayout="@layout/widget"
    android:minHeight="55dp"
    android:minWidth="200dp"
    android:previewImage="@drawable/example_appwidget_preview"
    android:resizeMode="horizontal|vertical"
    android:updatePeriodMillis="86400000"
    android:widgetCategory="home_screen">
</appwidget-provider>
```

4. 修改原来的 StaticRecevier.java,改名为 Widget.java,并且 Widget 继承 AppWidgetProvider,在这里需要注意因为 AppWidgetProvider 已经继承了 BroadcastReceiver,所以

Widget 就不需要再继承 BroadcastReceiver 了,一开始继承多个类出现了错误.然后实现 onUpdate 函数.

```
public void onUpdate(Context context, AppWidgetManager appWidgetManager, int[] appWidgetIds) {
  super.onUpdate(context, appWidgetManager, appWidgetIds);
  Intent intent = new Intent(context, MainActivity.class);
  PendingIntent pi = PendingIntent.getActivity(context, 0, intent, 0);
  RemoteViews rv = new RemoteViews(context.getPackageName(), R.layout.widget);
  rv.setOnClickPendingIntent(R.id.widget_img, pi);
  appWidgetManager.updateAppWidget(appWidgetIds, rv);
}
```

当点击 widget 的图片时跳转到主页面.因为点击水果列表后需要更新桌面的 widget 的图片和文字,所以需要为原先的 onReceive 增加以下内容,这简单的几句代码也出现了这次实验最大的 bug,在后面实验中遇到的问题会讲到.

```
RemoteViews rv = new RemoteViews(context.getPackageName(), R.layout.widget);
rv.setTextViewText(R.id.widget_text, fruit);
rv.setImageViewResource(R.id.widget_img, map.get(fruit));
AppWidgetManager am = AppWidgetManager.getInstance(context);
ComponentName cn = new ComponentName(context, Widget.class);
am.updateAppWidget(cn, rv);
```

5. 对静态注册功能进行调试.
6. 成功实现了静态注册的功能之后,动态注册也就差不多了,在上次的动态注册的 onReceive 的基础上再增加 Widget 部分的代码即可.
7. 对动态注册进行调试。
8. 完善样式，TA 的 demo 中动态注册页面的输入框获取到焦点后会有橙色的边框,上网查阅资料后也实现了这个样式。

   首先要自定义两个 drawable 的资源文件,分别设置输入框获取到焦点和失去焦点时的样式,然后再实现一个选择器文件,根据不同条件使用不同的样式.

失去焦点时的样式

```
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#FFFFFF" />
    <corners android:radius="3dp"/>
    <stroke
        android:width="3dp"
        android:color="#BDC7D8" />
</shape>
```

获取焦点时的样式

```
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#FFFFFF" />
    <size android:height="40dp" />
    <corners android:radius="3dip"/>
    <stroke android:width="2dp"
        android:color="#f09814" />
</shape>
```

选择器文件

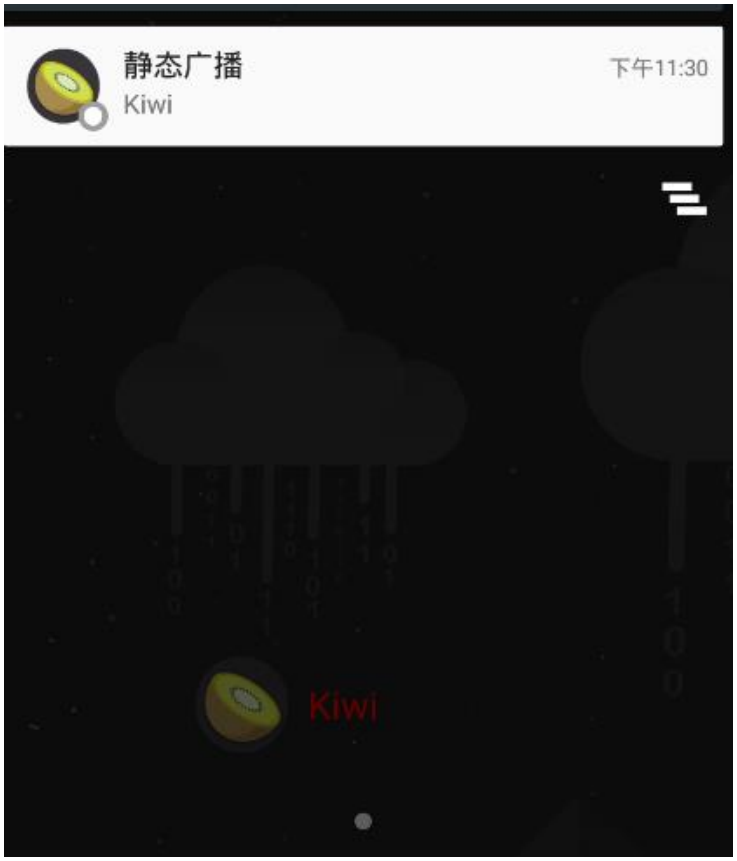
```
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_focused="true"
 android:drawable="@drawable/edittext_focused" />
    <item android:state_window_focused="false"
 android:drawable="@drawable/edittext_normal" />
</selector>
```

最后只需要设置文本框的 background 为选择器就可以根据不同的状态使用不同的样式。

五． 实验结果截图

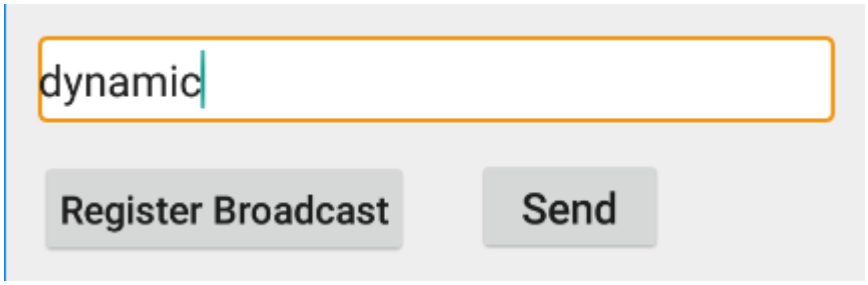
桌面默认的 widget

点击静态注册页面的水果,.出现对应通知并且 widget 相应改变

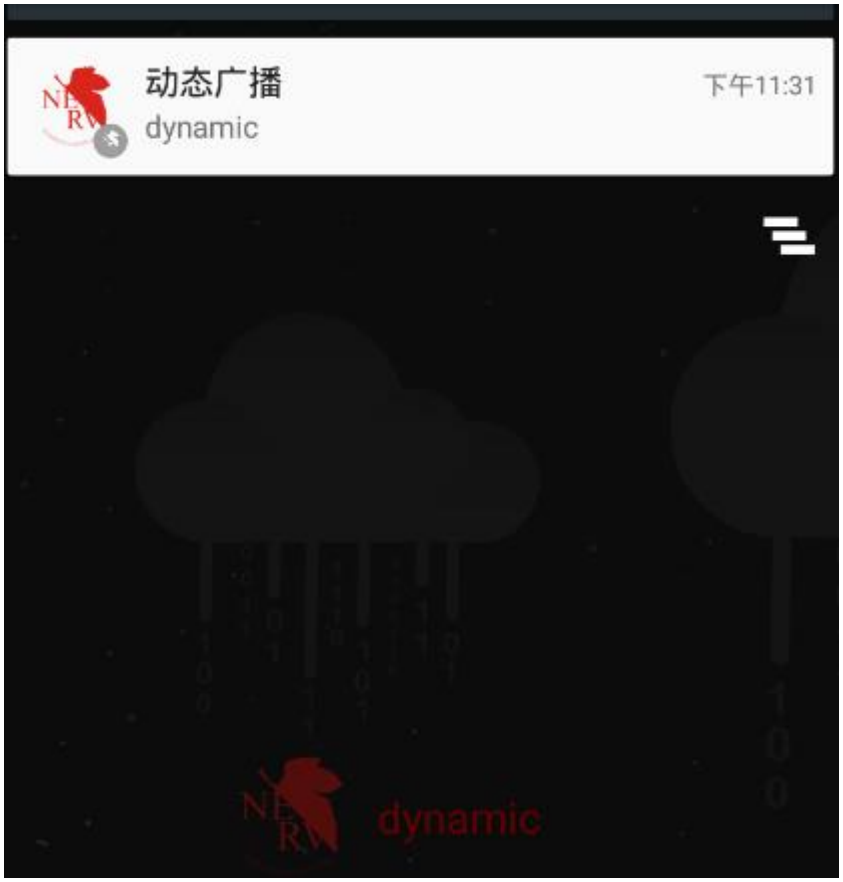


点击后发送广播，点击通知回到主界面

动态注册页面,设置了输入框样式



注册广播后点击 send 按钮



Widget 相应发生变化

点击 widget 回到主页面

六． 实验过程遇到的问题

1. Xml 位置错误

   一开始创建 widget_info.xml 文件时,惯性思维,看到是 xml 后缀的就直接在 layout 文件夹下新建了,结果写完 widget_info.xml 后按照实验文档里的在注册文件中添加了相应的内容后找不到文件,仔细一看才发现 widget_info.xml 的路径是 xml 文件夹下,新建了一个 xml 文件夹后修改相应的代码解决了这个问题.

2. 老师上课讲了获取 Map 之类的 value 的时候很容易出问题,因为有可能键不存在获取到的为 null,所以要设置一个默认值.本次实验中想到这点而且看到 AS 自动补全了有一个函数

getorDefault 就使用了,结果编译的时候报错,尴尬,要求 API 要 24 才能使用这个函数.

3. 一开始完成实验的时候看着实验要求文档打的代码,结果打完静态注册之后过了编译,可是当把 widget 拉到桌面能出现 widget 但程序就崩了,logcat 显示空指针异常,但是出错的那些文件都是系统的 java 文件,完全找不到我哪里空指针了,再仔细看了一遍实验文档发现 setTextViewText、setImageViewResource,之后使用 AppWidgetManager 类对 Widget 进行更新。而我的代码里就直接设置了文本和图片之后就完了,完全没有出现 AppWidgetManager,看了老师的课件后发现,设置了这些之后 widget 还不会更新,要手动更新.

```
AppWidgetManager am = AppWidgetManager.getInstance(context);
ComponentName cn = new ComponentName(context, Widget.class);
am.updateAppWidget(cn, rv);
```

七． 思考与总结

这次实验利用上次实验的代码基础其实难度并不大，只是自己还没弄清楚 widget 就开始实验，而且阅读实验文档也不够仔细,一味的复制代码,没有看清楚文档和老师的课件，出现了一些不该犯的错误,发了很多时间找bug,给了自己一个教训,以后还是先了解实验的原理后再开始实验,磨刀不误砍柴工,还能事半功倍。通过这次实验，学习到了 Widget 编程的基础，平常使用的天气,时间等 widget 原来原理是这样的,以前一直都去注意这些小部件。