Lab4 实验报告

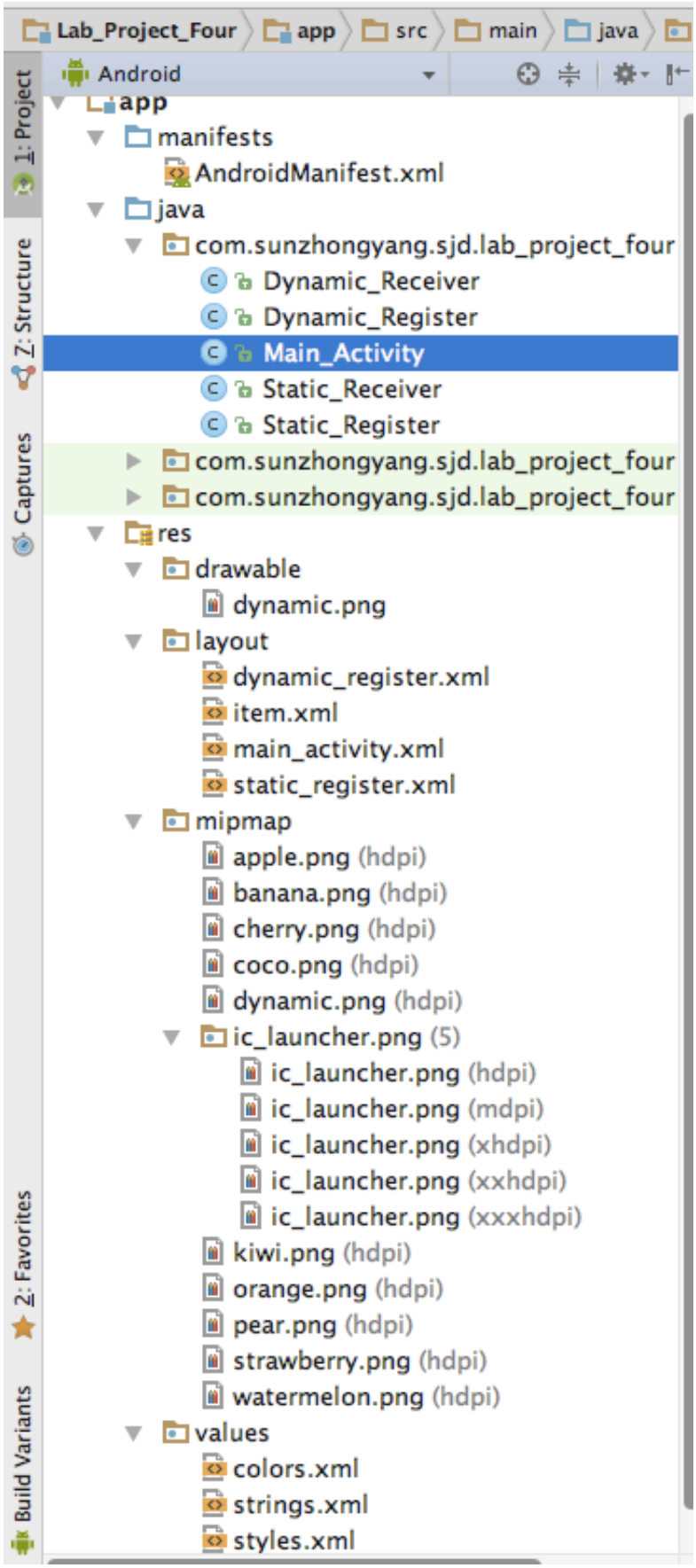
13331233 孙中阳

参考资料

《第一行代码 Android》 郭霖 著
《Lab4实验文档》

实验步骤

本次试验并不复杂，主要需要进一步熟悉 `Intent` 的使用并通过动态和静态的广播实现程序的动态响应，如发送通知等
图为项目结构

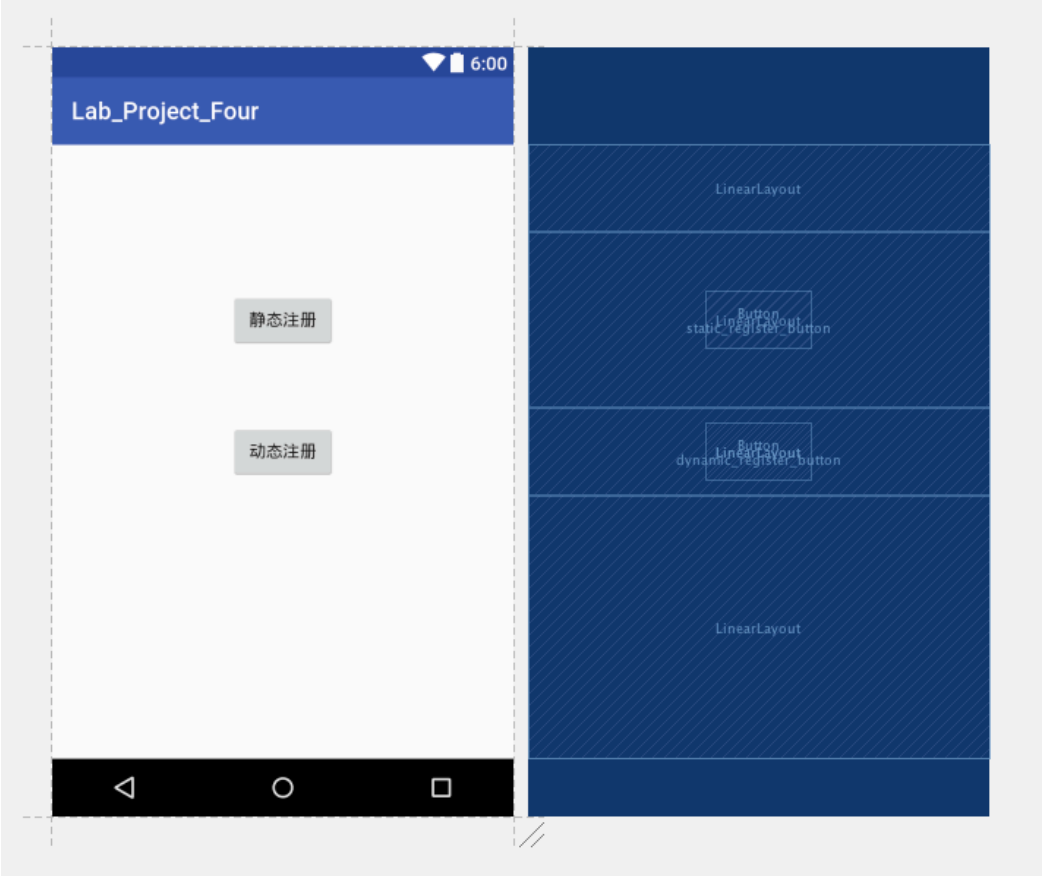

Lab_Project_Four
app
src
main
java
Android
1: Project
Z: Structure
Captures
2: Favorites
Build Variants
app
manifests
AndroidManifest.xml
java
com.sunzhongyang.sjd.lab_project_four
Dynamic_Receiver
Dynamic_Register
Main_Activity
Static_Receiver
Static_Register
com.sunzhongyang.sjd.lab_project_four
com.sunzhongyang.sjd.lab_project_four
res
drawable
dynamic.png
layout
dynamic_register.xml
item.xml
main_activity.xml
static_register.xml
mipmap
apple.png (hdpi)
banana.png (hdpi)
cherry.png (hdpi)
coco.png (hdpi)
dynamic.png (hdpi)
ic_launcher.png (5)
ic_launcher.png (hdpi)
ic_launcher.png (mdpi)
ic_launcher.png (xhdpi)
ic_launcher.png (xxhdpi)
ic_launcher.png (xxxhdpi)
kiwi.png (hdpi)
orange.png (hdpi)
pear.png (hdpi)
strawberry.png (hdpi)
watermelon.png (hdpi)
values
colors.xml
strings.xml
styles.xml

这次的文件多一些

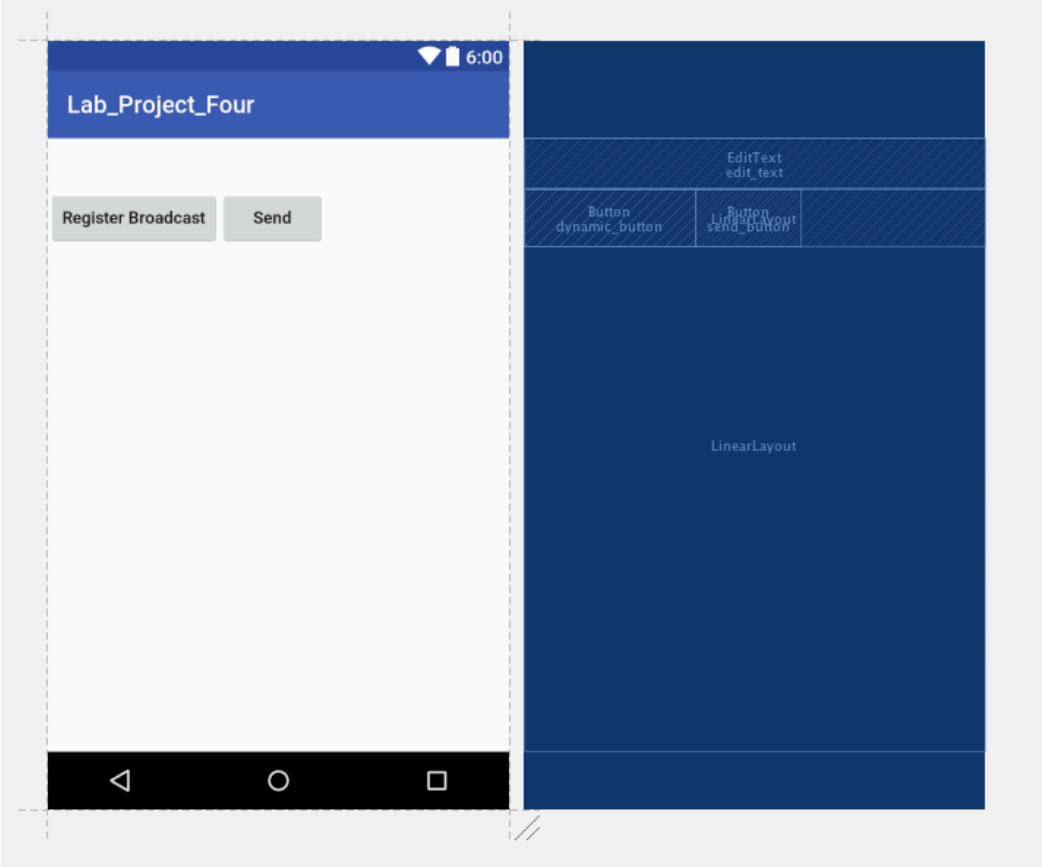
从实验文档看，需要实现的目标主要有

1.动态注册 `Broadcast`
2.静态注册 `Broadcast`
3.掌握 `Notification` 编程基础

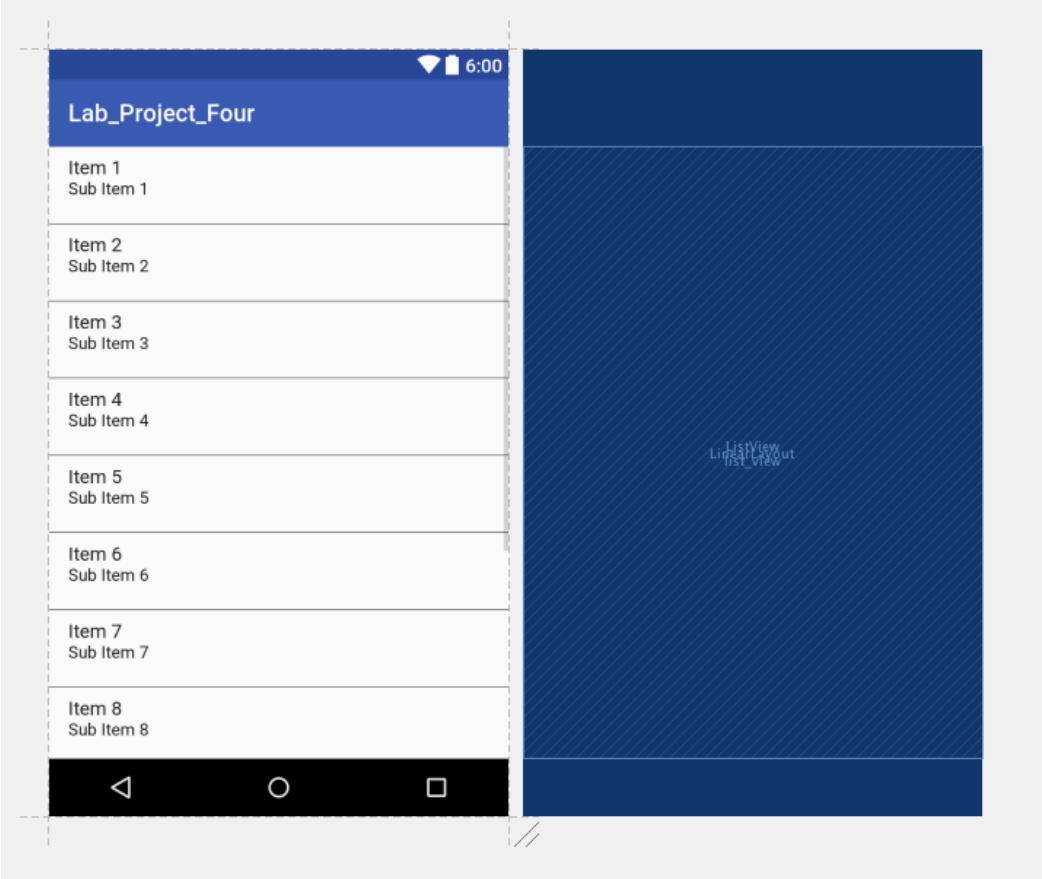
图为主页面布局：



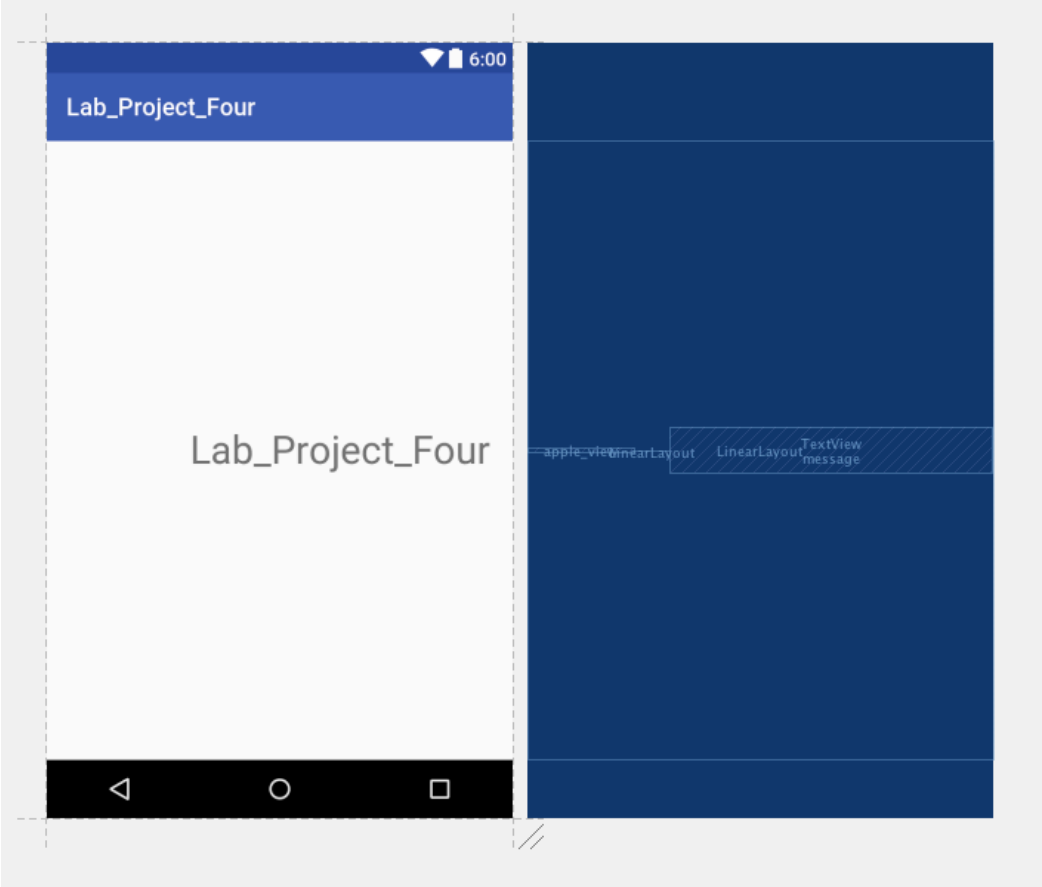
图为动态注册页面布局：


6:00
Lab_Project_Four
Register Broadcast
Send
EditText
edit_text
Button
dynamic_button
Button
send_button
LinearLayout

图为静态注册页面布局：



图为静态注册ListView布局：



接下来考虑实现
本次开发主要涉及两个方面：

1.设置动态注册
主页面的点击按钮和按钮的跳转不多赘述，完成上述工作后，需要为动态注册创建一个 `Activity` ，然后为其创建一个用于相应的 `Receiver` ，这个 `Receiver` 不需要在 `AndroidManifest.xml` 中注册

点击注册按钮后，如果之前没有注册过，则动态生成一个过滤器，否则取消这个过滤器

```
//当点击注册广播按钮
dynamicButton.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        //如果之前没有注册,则注册
        if(dynamicButton.getText().equals("Register Broadca
st"))
        {
            //生成一个过滤器并注册该过滤器
            IntentFilter dynamic_filter = new IntentFilter(
);
            dynamic_filter.addAction(DYNAMICACTION);
            registerReceiver(dynamicReceiver, dynamic_filte
r);

            //更改按钮的内容为取消注册
            dynamicButton.setText(R.string.dynamic_unregist
er_button);
        }
        //如果之前注册过,则取消注册
        else
        {
            unregisterReceiver(dynamicReceiver);

            //更改按钮的内容为注册
            dynamicButton.setText(R.string.dynamic_register
_button);
        }
    }
});
```

然后设置 `Send` 按钮，点击按钮则将输入框中的内容附加到一个广播中并发送该广播

```
sendButton.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        //获取输入框的文本
        String text = editText.getText().toString();

        //将要传递的通知文本加入到bundle中
        Bundle bundle = new Bundle();
        bundle.putString("Content", text);

        //发送广播
        Intent intent = new Intent(DYNAMICACTION);
        intent.putExtras(bundle);
        sendBroadcast(intent);
    }
});
```

在 `Receiver` 中，收到一个广播后首先获取广播中的附加信息，然后使用此附加信息发送一个通知

```
@Override
public void onReceive(Context context, Intent intent)
{
    //获取bundle及其中的通知文本
    Bundle bundle = intent.getExtras();
    String text = bundle.getString("Content");

    Notification.Builder builder = new Notification.Builder
(context);

    //设置并显示一个通知
    builder.setContentTitle("动态广播")
            .setContentText(text)
            .setTicker("您有一条新消息")
            .setLargeIcon(BitmapFactory.decodeResource(cont
ext.getResources(), R.drawable.dynamic))
            .setSmallIcon(R.mipmap.dynamic)
            .setAutoCancel(true);

    NotificationManager manager = (NotificationManager) con
text.getSystemService(Context.NOTIFICATION_SERVICE);

    //点击通知则跳转到主界面
    Intent mintent = new Intent(context, Main_Activity.clas
s);
    PendingIntent pendingIntent = PendingIntent.getActivity
(context, 0, mintent, 0);
    builder.setContentIntent(pendingIntent);

    manager.notify(0, builder.build());
}
```

2.设置静态注册

静态注册的部分复杂一些，主要是因为涉及到一个 `ListView` 这个 `ListView` 中有图片，所以 `SimpleAdapter` 貌似用不了，于是参考网上的方法实现了一个自定义的版本，核心思想是按顺序为每个 `list item` 进行初始化

```
//一个自定义的ListViewAdapter
public class ListViewAdapter extends BaseAdapter
{
```

```
    //用于存放所有list item的view
    View[] itemViews;

    public ListViewAdapter(String[] itemTexts, int[] itemIm
ageRes)
    {
        itemViews = new View[itemTexts.length];

        //按次序初始化每个list item
        for (int i = 0; i < itemViews.length; ++i) {
            itemViews[i] = makeItemView(itemTexts[i],
                    itemImageRes[i]);
        }
    }

    private View makeItemView(String strText, int resId)
    {
        LayoutInflater inflater = (LayoutInflater)Static_Re
gister.this.getSystemService(Context.LAYOUT_INFLATER_SERVIC
E);

        // 使用View的对象itemView与R.layout.item关联
        View itemView = inflater.inflate(R.layout.item, nul
l);

        //初始化某个list item的描述信息
        TextView text = (TextView)itemView.findViewById(R.i
d.message);
        text.setText(strText);
        //初始化某个list item的图像
        ImageView image = (ImageView)itemView.findViewById(
R.id.apple_view);
        image.setImageResource(resId);

        return itemView;
    }

    //返回当前ListView中item的数目
    public int getCount() {
        return itemViews.length;
    }

    //获取某个位置的list item
```

```
    public View getItem(int position) {
        return itemViews[position];
    }

    //获取某一个位置的list item的id
    public long getItemId(int position) {
        return position;
    }

    //获取某一个位置的list item的view
    public View getView(int position, View convertView, Vie
wGroup parent)
    {
        return itemViews[position];
    }
}
```

初始化 `ListView` 的方法和前一实验类似，这里着重说一下 `list item` 的点击事件

```
//为ListView设置点击事件
list.setOnItemClickListener(new AdapterView.OnItemClickList
ener()
{
    @Override
    public void onItemClick(AdapterView<?> adapter, View vi
ew, int i, long l)
    {
        //设置一个bundle用于传送list item的文本和图像id
        Bundle bundle = new Bundle();
        bundle.putString("Content", fruits[i]);
        bundle.putInt("PictureId", resIds[i]);

        //发送广播
        Intent intent = new Intent(STATICACTION);
        intent.putExtras(bundle);
        sendBroadcast(intent);
    }
});
```

点击之后同样是附加一些信息，包括点击的水果的种类和水果图片在资源文件中的id.然后会发送一个内容为 `STATICACTION` 的广播，这个广播的

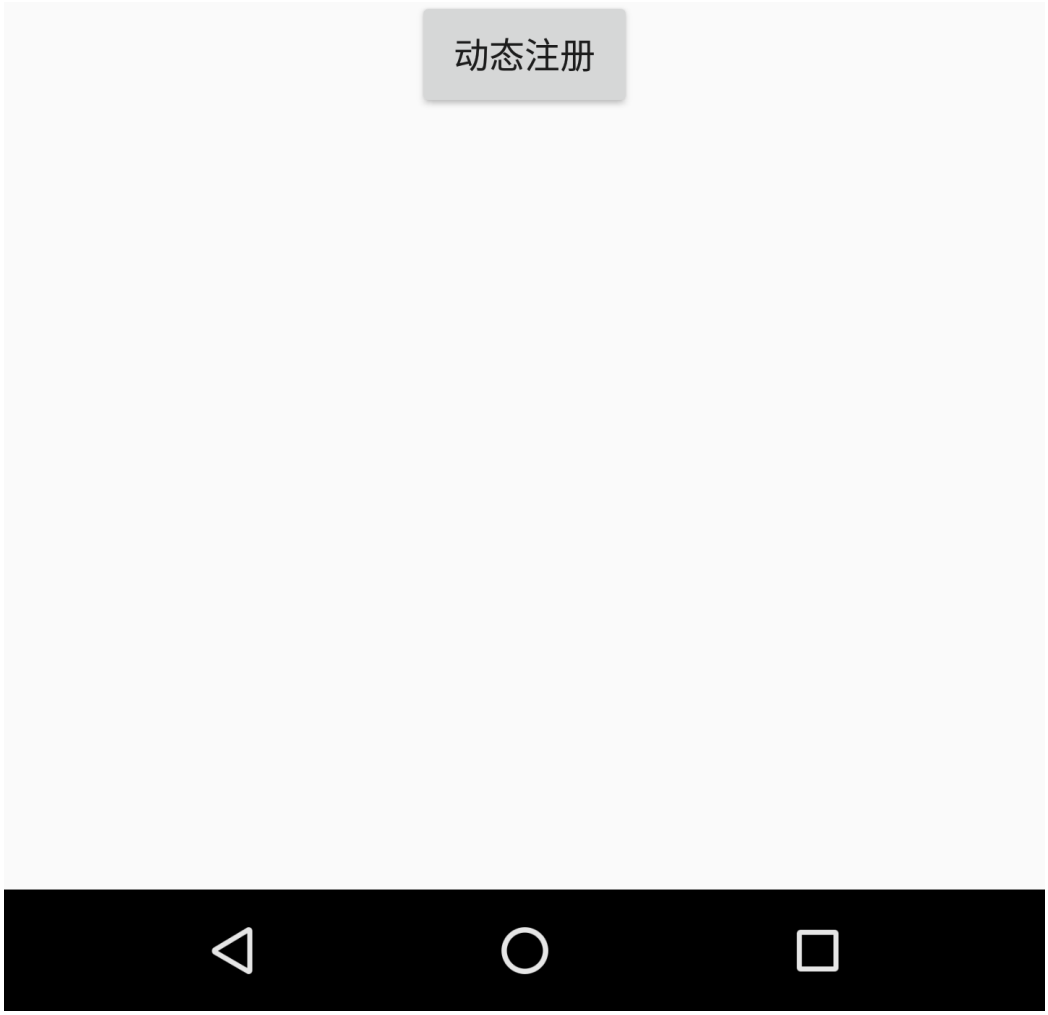
`Receiver` 需要在 `AndroidManifest.xml` 中注册

```
<receiver android:name=".Static_Receiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name="com.sunzhongyang.sjd.lab_proj
ect_four.staticcreceiver" />
    </intent-filter>
</receiver>
```

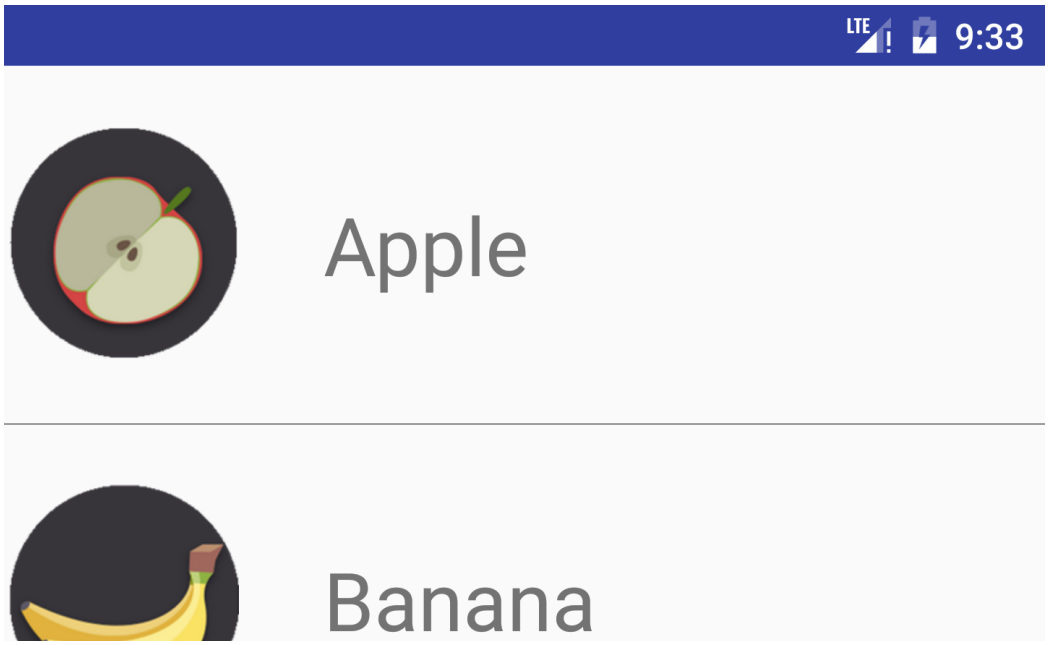
然后，`Receiver` 便可接受相应的广播
收到后，同样是发送一个通知，不同的是，这次 `largeIcon` 的图标由传入水果图片 `Id` 关联水果图片经 `Bitmap` 工厂生成的 `Bitmap` 初始化

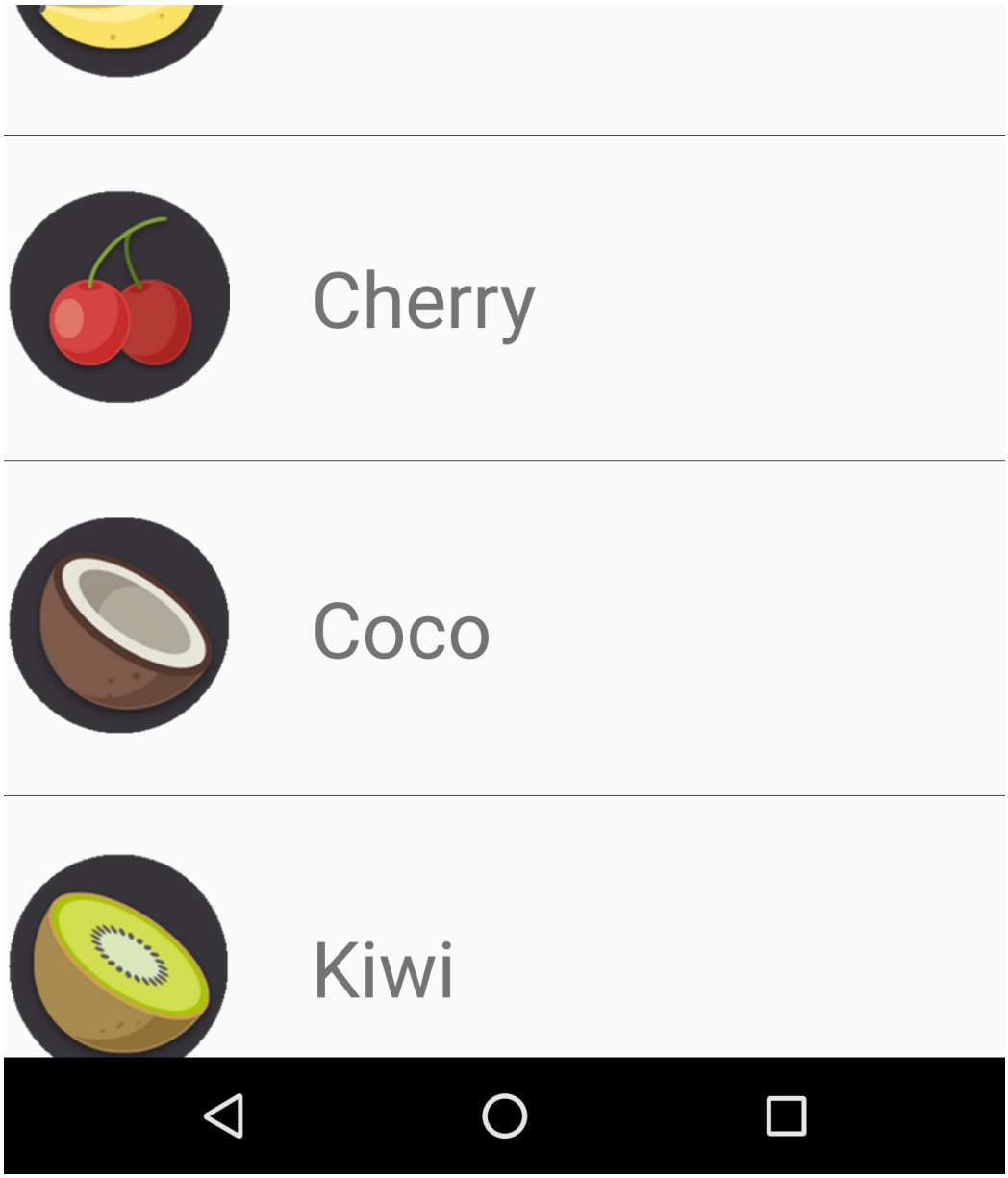
实验成果截图

主页面

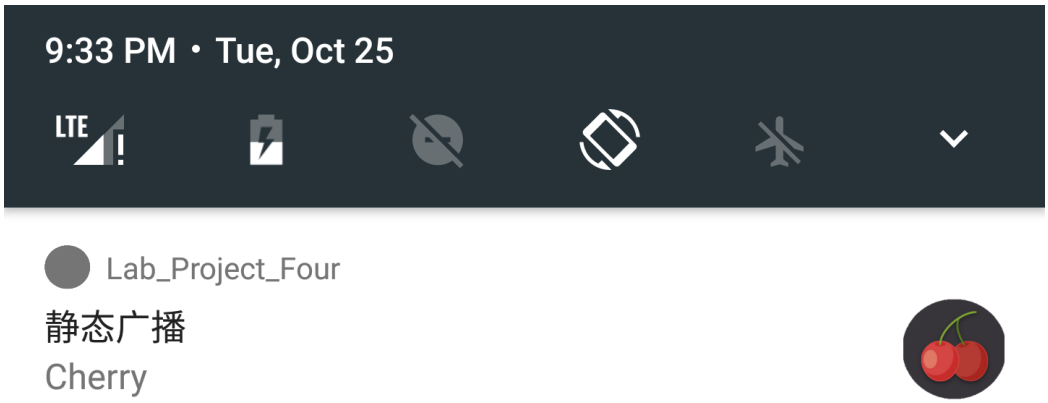


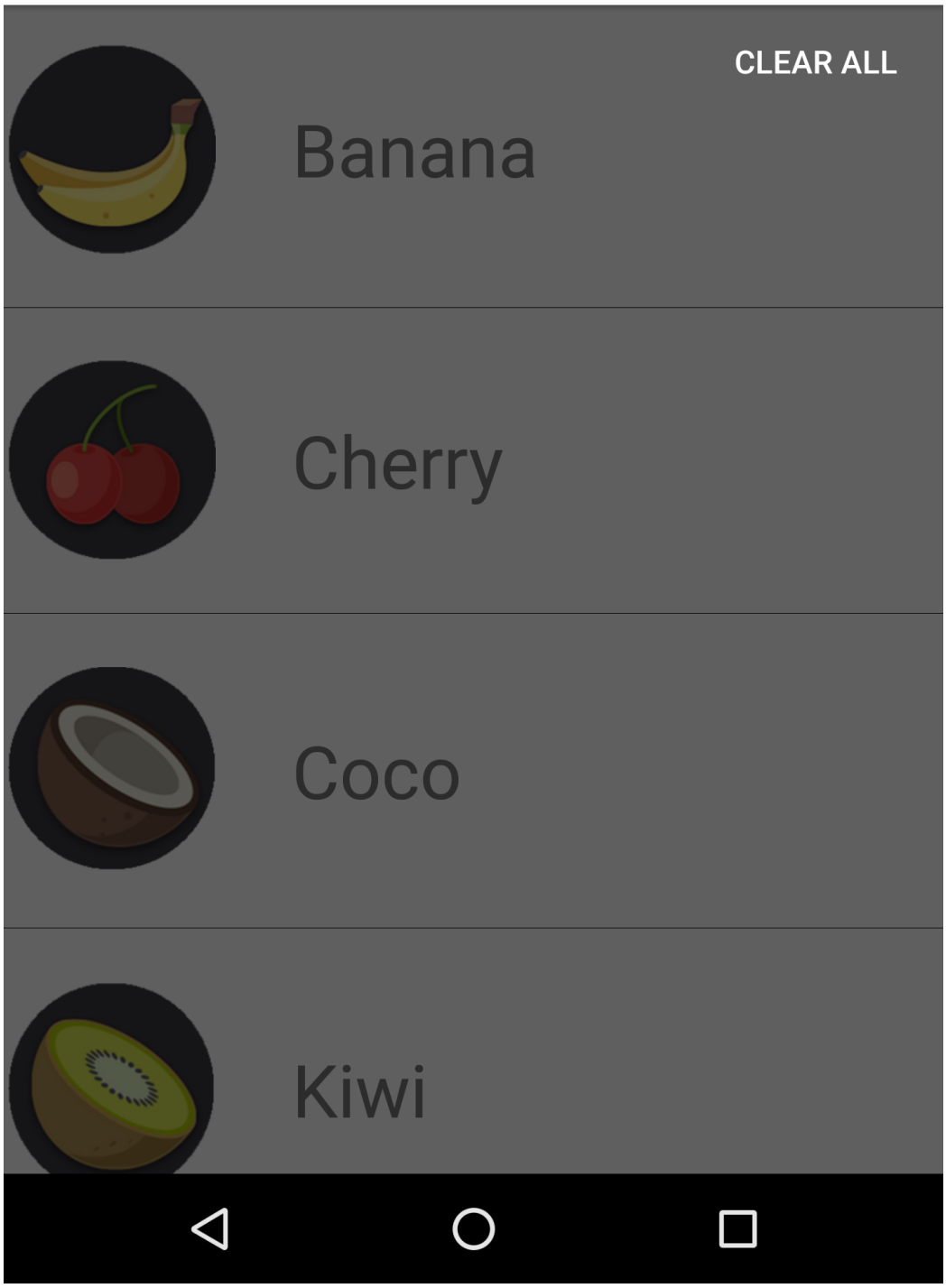
动态注册页面



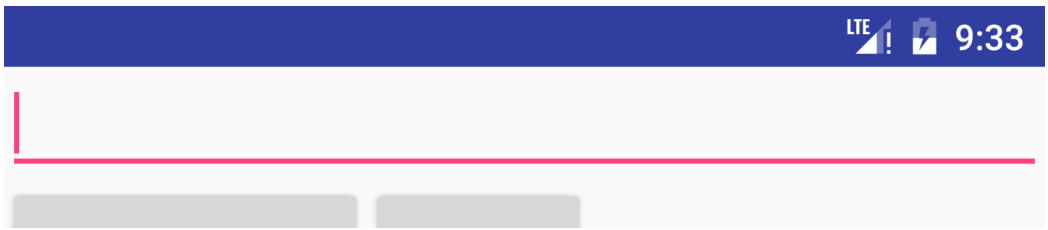

Cherry
Coco
Kiwi

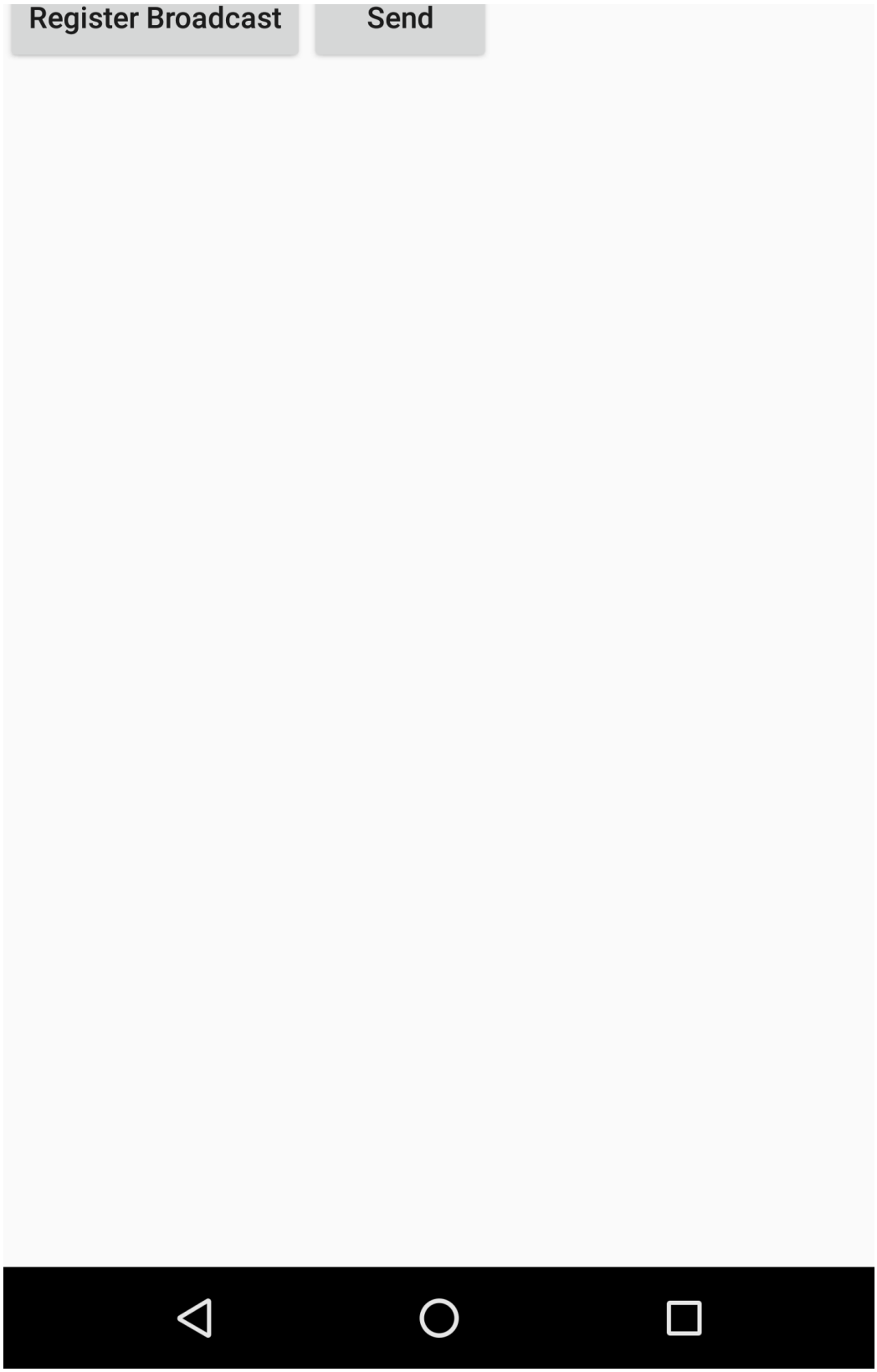
动态注册通知


9:33 PM • Tue, Oct 25
LTE
Lab_Project_Four
静态广播
Cherry

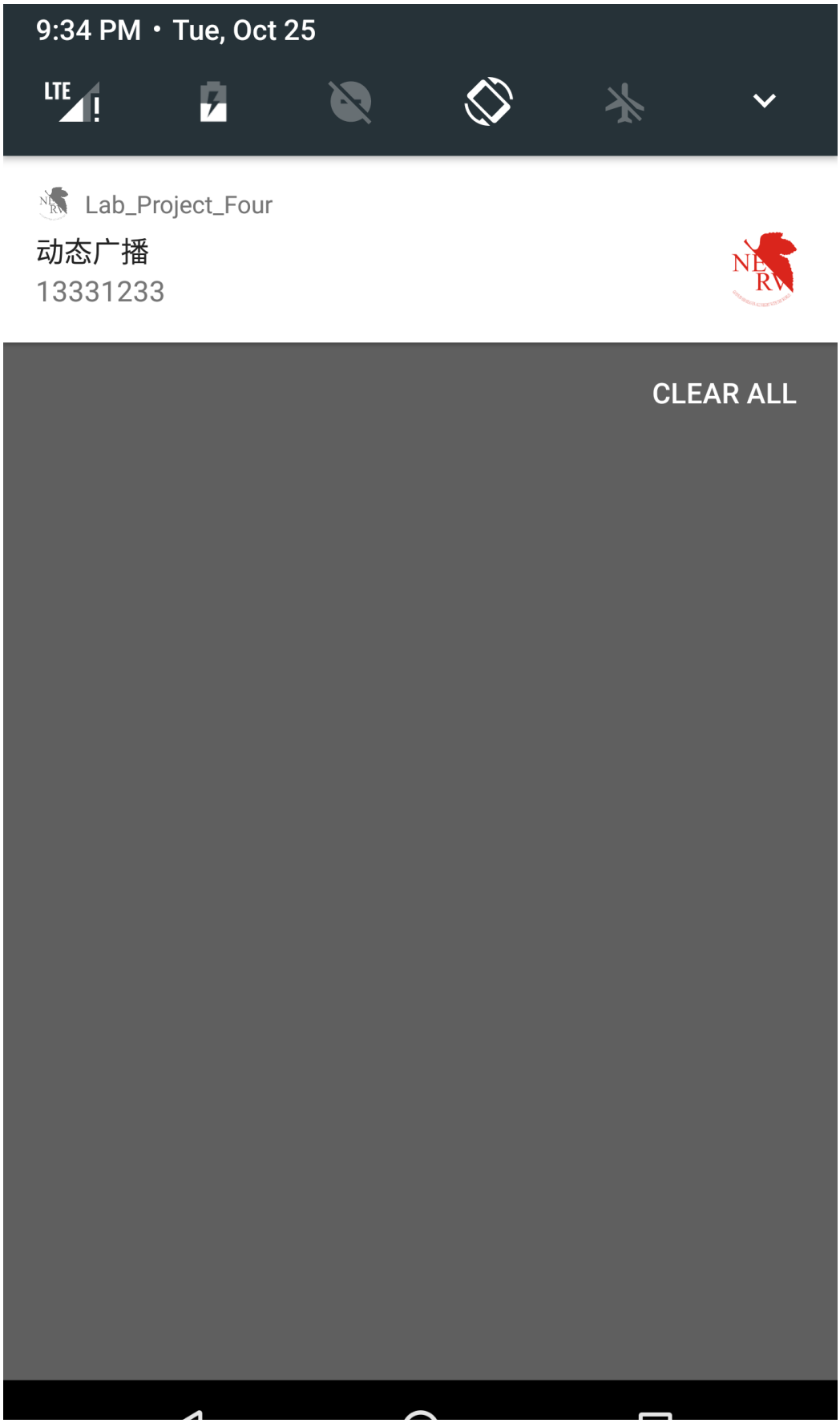


静态注册页面





静态注册通知


9:34 PM • Tue, Oct 25
Lab_Project_Four
动态广播
13331233
CLEAR ALL

实验过程遇到的问题

这次同样遇到一些问题

1.在动态注册页面中，滑动 `ListView` 后 `ListView` 的排序会发生混乱
经查询，发现这一问题是由于Android不会在初始化时初始化所有 `list item` ，动态加载的过程中如果时间过快则会加载错误位置的图片和文本。为解决这一问题，只需强制初始化时加载所有 `list item` 即可

```
public View getView(int position, View convertView, ViewGro
up parent)
{
    return itemViews[position];
}
```

直接返回一个 `view` ，之前并不检查当前 `list item` 是否在加载的范围中

2.小米Note无法加载 `largeIcon`
安装一个Android模拟器后，使用模拟器运行程序发现问题解决

思考与总结

本次试验深入了解了Android开发的有关知识，对Android Studio的使用也有了更深刻的把握，尤其是和通知以及广播有关的内容。大部分内容可以通过查阅PPT或作业说明得到解决，只有少部分需要百度。Android较UWP应用范围更广，网上的资料也更多，问题也解决的比较顺利。总的来说很有收获。