[Android] 使用Include布局+Fragment滑动切换屏幕

原创 Eastmount 最后发布于2014-11-10 00:52:44 阅读数 12505 ☆ 收藏 展开

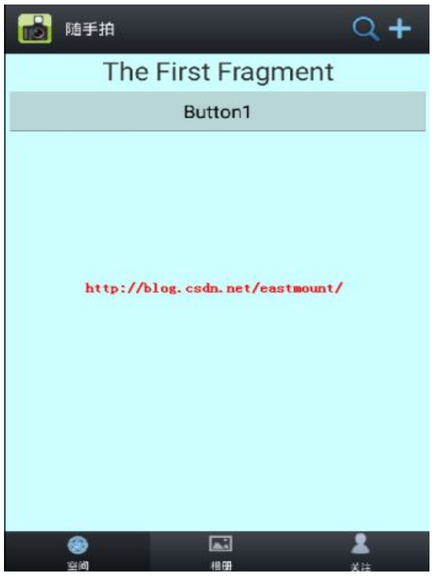
前面的文章已经讲述了"随手拍"项目图像处理的技术部分,该篇文章主要是主界面的布局及屏幕滑动切换,并结合鸿洋大神的视频和郭神的第一行代码(强推两人Android博客),完成了下面的内容:

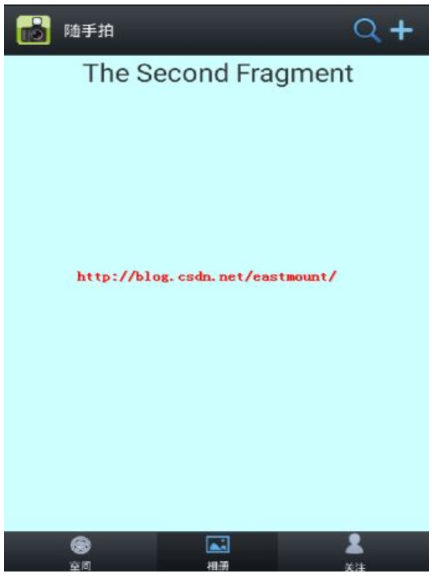
(1).学习使用Include布局XML
(2).通过添加适配器加载fragment
(3).实现滑动触摸切换屏幕ViewPager
(4).改变图标及背景,并响应fragment中控件及传递参数

参考资料:
郭霖大神的《Android第一行代码》
鸿洋大神的微信界面 http://www.imooc.com/learn/198

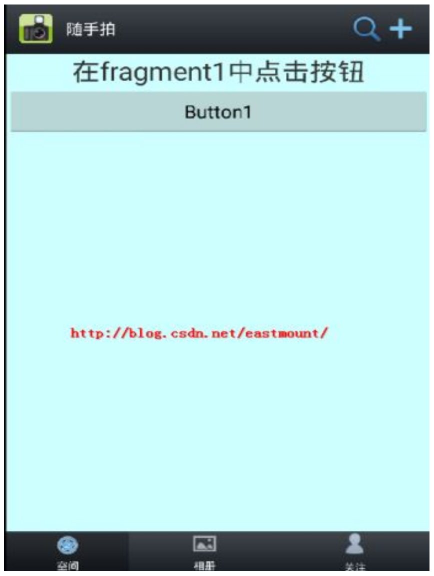
一. 运行效果

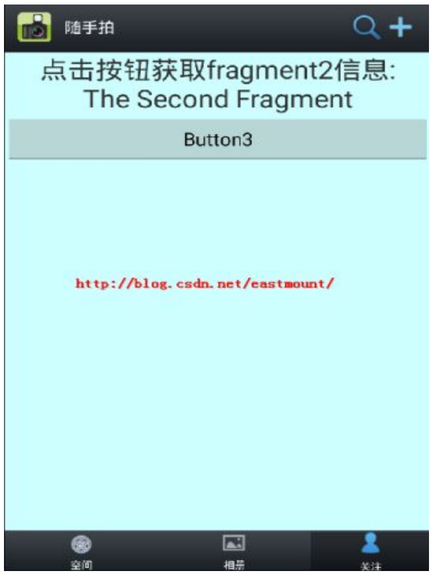
如下图所示,滑动屏幕可以切换布局"空间"、"相册"、"关注".同时会有图标颜色变蓝,背景颜色加深的效果.



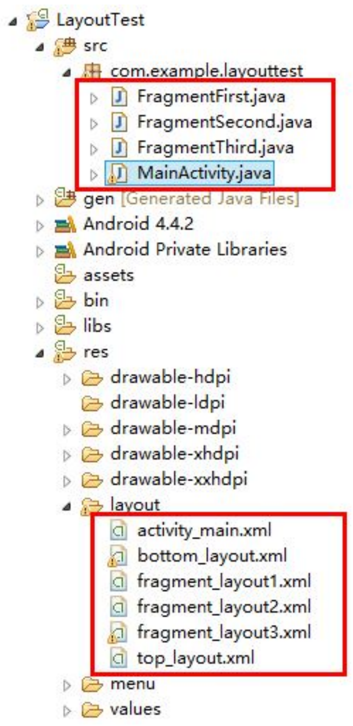


同时添加了按钮事件,在fragment1中点击按钮显示内容,在fragment3中点击按钮获取第二个布局内容并显示.





二. 项目工程结构



```
LayoutTest
  src
    com.example.layouttest
      FragmentFirst.java
      FragmentSecond.java
      FragmentThird.java
      MainActivity.java
  gen [Generated Java Files]
  Android 4.4.2
  Android Private Libraries
  assets
  bin
  libs
  res
    drawable-hdpi
    drawable-ldpi
    drawable-mdpi
    drawable-xhdpi
    drawable-xxhdpi
    layout
      activity_main.xml
      bottom_layout.xml
      fragment_layout1.xml
      fragment_layout2.xml
      fragment_layout3.xml
      top_layout.xml
    menu
    values
```

三. Include布局XML文件

首先添加头部布局top_layout.xml,采用相对布局,右边两图标:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:paddingLeft="12dp"
    android:paddingRight="12dp"
    android:background="@drawable/image_toolbar_bg" >
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
                                                     android:layout_gravity="center"
        android:orientation="horizontal" >
        <ImageView
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:src="@drawable/icon_suishoupai" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="12dp"
            android:text="随手拍"
            android:textSize="15sp"
            android:layout_gravity="center"
            android:textColor="#ffffff" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_gravity="center"
        android:layout_alignParentRight="true"
        android:orientation="horizontal" >
        <ImageView
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:src="@drawable/image_top_watch" />
        <ImageView
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:src="@drawable/image_top_add" />
    </LinearLayout>
</RelativeLayout>
```

然后添加底部布局bottom_layout.xml，由3个LinearLayout水平布局组成，其中每个LinearLayout有ImageView和TextView组成：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="40dp"
    android:background="@drawable/image_toolbar_bg"
    android:orientation="horizontal" >
        <LinearLayout
            android:id="@+id/bottomLayout1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:gravity="center"
        android:background="@drawable/image_toolbar_bg_sel"
                android:orientation="vertical" >
        <ImageView
            android:id="@+id/image1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="1dp"
            android:src="@drawable/image_bottom_effect" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="15dp"
            android:text="空间"
                                             android:textColor="#ffffff"
            android:textSize="10dp" />
     </LinearLayout>
     <LinearLayout
        android:id="@+id/bottomLayout2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:gravity="center"
                android:orientation="vertical" >
        <ImageView
            android:id="@+id/image2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="1dp"
            android:src="@drawable/image_bottom_frame_no" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="15dp"
            android:text="相册"
            android:textColor="#ffffff"
            android:textSize="10dp" />
     </LinearLayout>
     <LinearLayout
        android:id="@+id/bottomLayout3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:gravity="center"
                android:orientation="vertical" >
        <ImageView
            android:id="@+id/image3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="1dp"
            android:src="@drawable/image_bottom_person_no" />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="15dp"
            android:text="关注"
            android:textColor="#ffffff"
            android:textSize="10dp" />
     </LinearLayout>
</LinearLayout>
```

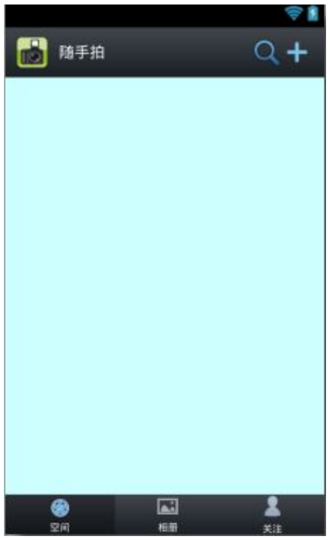
最后在activity_main.xml中调用Include布局,ViewPager用于加载不同的fragment,并实现触屏切换在该控件上:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
        android:orientation="vertical">

    <include layout="@layout/top_layout"/>
    <android.support.v4.view.ViewPager
        android:id="@+id/viewpager1"
        android:layout_width="match_parent"
        android:layout_height="0dp"          android:background="#ccffff"
        android:layout_weight="1" />
    <include layout="@layout/bottom_layout"/>

</LinearLayout>
```

在MainActivity.java中onCreate函数设置无标题requestWindowFeature(Window.FEATURE_NO_TITLE),在xml文件中可设置Frame预览效果无标题,显示布局如下图所示：



四. 实现触屏切换fragment

首先设置Fragment的布局XML文件,fragment_layout1.xml如下,其他类似:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="25sp"
        android:gravity="center"
        android:text="The First Fragment" />
    <Button
        android:id="@+id/button1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Button1" />
</LinearLayout>
```

然后添加FragmentFirst.java、FragmentSecond.java和FragmentThird,其中FragmentSecond.java如下,其他类似:

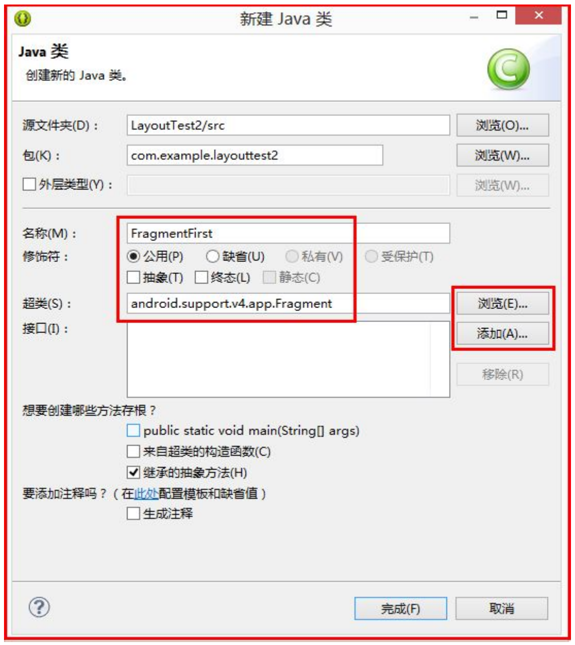
```
package com.example.layouttest;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class FragmentSecond extends Fragment {

        @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_layout2, container, false);
    }
}
```

PS:由于刚学习Android一个月,所以文章很基础,在新建类中可以点击"浏览"自定义添加继承超类或点击"添加"增加接口,此处继承Fragment.注意"import android.support.v4.app.Fragment;"所有的需要一致.



新建 Java 类

Java 类
创建新的 Java 类。

源文件夹(D)： LayoutTest2/src 浏览(O)...
包(K)： com.example.layouttest2 浏览(W)...
☐ 外层类型(Y)： 浏览(W)...

名称(M)： FragmentFirst
修饰符： [✓ 公用(P)] 缺省(U) 私有(V) 受保护(T)
☐ 抽象(T) ☐ 终态(L) ☐ 静态(C)
超类(S)： android.support.v4.app.Fragment 浏览(E)...
接口(I)： 添加(A)... 移除(R)

想要创建哪些方法存根？
☐ public static void main(String[] args)
☐ 来自超类的构造函数(C)
☑ 继承的抽象方法(H)
要添加注释吗？（在此处配置模板和缺省值）
☐ 生成注释

完成(F) 取消

然后设置MainActivity.java,代码如下：

```
package com.example.layouttest;

import java.util.ArrayList;
import java.util.List;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentActivity;
import android.support.v4.app.FragmentPagerAdapter;
import android.support.v4.view.ViewPager;
import android.view.Window;

public class MainActivity extends FragmentActivity {

        //注意:导入时均为support.v4.app/view 保持一致
        private ViewPager viewPager1;
        private FragmentPagerAdapter fpAdapter;
        private List<Fragment> listData;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //注意:设置无标题需要在setContentView前调用 否则会崩溃
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        setContentView(R.layout.activity_main);
        //初始化设置ViewPager         setViewPager();
    }
        private void setViewPager() {
                //初始化数据
                viewPager1 = (ViewPager) findViewById(R.id.viewpager1);
                listData = new ArrayList<Fragment>();
                FragmentFirst fragmentFirst = new FragmentFirst();
                FragmentSecond fragmentSecond = new FragmentSecond();
                FragmentThird fragmentThird = new FragmentThird();
                //三个布局加入列表
                listData.add(fragmentFirst);
                listData.add(fragmentSecond);
                listData.add(fragmentThird);
                //ViewPager相当于一组件容器 实现页面切换
                fpAdapter =new FragmentPagerAdapter(getSupportFragmentManager())
                {
                        @Override
                        public int getCount()
                        {
                                return listData.size();
                        }
                        @Override
                        public Fragment getItem(int arg0)
                        {
                                return listData.get(arg0);
                        }
                };
                //设置适配器
                viewPager1.setAdapter(fpAdapter);
        }
}
```

此时即可实现触屏切换效果,但同时需要注意:

(1).需要把MainActivity继承从Activity改为FragmentActivity.

(2).可能会遇到错误"类型对于参数(FragmentFirst)不适用",你需要把导入修改"import android.support.v4.app.Fragment;"同时注意support.v4.app/view 保持一致.

五. 实现滑屏变换图标

此时设置底部滑动切换的图标时需要添加自定义变量:

```
//底部图标
private ImageView image1;
private ImageView image2;
private ImageView image3;
private LinearLayout layout1;
private LinearLayout layout2;
private LinearLayout layout3;
```

然后,在setViewPager()函数中"viewPager1.setAdapter(fpAdapter)"后添加如下代码即可实现,其中switch中0、1、2对应listData中装入的三个布局:

```
//初始化图标
image1 = (ImageView) findViewById(R.id.image1);
image2 = (ImageView) findViewById(R.id.image2);
image3 = (ImageView) findViewById(R.id.image3);
layout1 = (LinearLayout) findViewById(R.id.bottomLayout1);
```

```
layout2 = (LinearLayout) findViewById(R.id.bottomLayout2);
layout3 = (LinearLayout) findViewById(R.id.bottomLayout3);   //滑屏变换图标
viewPager1.setOnPageChangeListener(new OnPageChangeListener() {
        @Override
        public void onPageSelected(int arg0)
        {
                switch(arg0)
                {
                case 0:
                        //图片切换
                        image1.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_effect));
                        image2.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_frame_no))
                        image3.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_person_no)
                        //背景加深
                        layout1.setBackgroundResource(R.drawable.image_toolbar_bg_sel);
                        layout2.setBackgroundResource(R.drawable.image_toolbar_bg);
                        layout3.setBackgroundResource(R.drawable.image_toolbar_bg);
                        break;
                case 1:
                        //图片切换
                        image1.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_effect_no)
                        image2.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_frame));
                        image3.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_person_no)
                        //背景加深
                        layout1.setBackgroundResource(R.drawable.image_toolbar_bg);
                        layout2.setBackgroundResource(R.drawable.image_toolbar_bg_sel);
                        layout3.setBackgroundResource(R.drawable.image_toolbar_bg);
                        break;
                case 2:
                        //图片切换
                        image1.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_effect_no)
                        image2.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_frame_no))
                        image3.setImageDrawable(getResources().getDrawable(R.drawable.image_bottom_person));
                        //背景加深
                        layout1.setBackgroundResource(R.drawable.image_toolbar_bg);
                        layout2.setBackgroundResource(R.drawable.image_toolbar_bg);
                        layout3.setBackgroundResource(R.drawable.image_toolbar_bg_sel);
                        break;
                }
        }
        @Override
        public void onPageScrolled(int arg0, float arg1, int arg2)
        {

        }
        @Override
        public void onPageScrollStateChanged(int arg0)
        {

        }
});
```

六. 调用Fragment中按钮及传递参数

设置FragmentFirst.java文件,通过onActivityCreated函数实现点击按钮事件:

```
public class FragmentFirst extends Fragment {

        @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_layout1, container, false);
                                                                                   }

        @Override
        public void onActivityCreated(Bundle savedInstanceState) {
                super.onActivityCreated(savedInstanceState);
                //添加Fragment1的响应事件
                Button button1 = (Button) getActivity().findViewById(R.id.button1);
                button1.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                TextView textView1 = (TextView) getActivity().findViewById(R.id.textView1);
                textView1.setText("在fragment1中点击按钮");
            }
        });
        }
}
```

FragmentThird.java实现点击Fragment3中按钮获取Fragment2中数据:

```
public class FragmentThird extends Fragment {

        @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_layout3, container, false);
    }

        @Override
        public void onActivityCreated(Bundle savedInstanceState) {
                super.onActivityCreated(savedInstanceState);
                //添加Fragment3的响应事件
                Button button3 = (Button) getActivity().findViewById(R.id.button3);
                button3.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                TextView textView1 = (TextView) getActivity().findViewById(R.id.textView2);
                TextView textView3 = (TextView) getActivity().findViewById(R.id.textView3);
                textView3.setText("点击按钮获取fragment2信息:\n"+textView1.getText());
            }
        });
        }
}
```

PS:是否Fragment的XML文件TextView需要设置不同的id,如果Fragment1与Fragment2设置相同的textView1程序没有响应.

本文主要讲述使用Include布局、Fragment切屏和ViewPager滑动效果.最后希望文章对大家有所帮助,尤其是对Android初学者,文章中有错误或不足之处,请包涵.

下载地址: http://download.csdn.net/detail/eastmount/8139915

(By:Eastmount 2014年11月10日夜1点 http://blog.csdn.net/eastmount/)

点赞 5 收藏 分享 ···

Eastmount 博客专家

发布了445 篇原创文章 · 获赞 5981 · 访问量 487万+

他的留言板 关注