

2.5.6 ViewFlipper(翻转视图)的基本使用

分类 [Android 基础入门教程](#)

Android 基础入门教程(Q群号 : 153836263)

[1.0 Android基础入门教程](#)

[1.0.1 2015年最新Android基...](#)

[1.1 背景相关与系统架构分析](#)

[1.2 开发环境搭建](#)

[1.2.1 使用Eclipse + ADT + S...](#)

[1.2.2 使用Android Studio开...](#)

[1.3 SDK更新不了问题解决](#)

[1.4 Genymotion模拟器安装](#)

[1.5.1 Git使用教程之本地仓...](#)

[1.5.2 Git之使用GitHub搭建...](#)

[1.6 .9\(九妹\)图片怎么玩](#)

[1.7 界面原型设计](#)

[1.8 工程相关解析\(各种文件...](#)

[1.9 Android程序签名打包](#)

[1.11 反编译APK获取代码&...](#)

[2.1 View与ViewGroup的概念](#)

[2.2.1 LinearLayout\(线性布局\)](#)

[2.2.2 RelativeLayout\(相对布...](#)

[2.2.3 TableLayout\(表格布局\)](#)

[2.2.4 FrameLayout\(帧布局\)](#)

[2.2.5 GridLayout\(网格布局\)](#)

[2.2.6 AbsoluteLayout\(绝对...](#)

[2.3.1 TextView\(文本框\)详解](#)

[2.3.2 EditText\(输入框\)详解](#)

[2.3.3 Button\(按钮\)与ImageB...](#)

[2.3.4 ImageView\(图像视图\)](#)

[2.3.5.RadioButton\(单选按钮...](#)

[2.3.6 开关按钮ToggleButton...](#)

[2.3.7 ProgressBar\(进度条\)](#)

[2.3.8 SeekBar\(拖动条\)](#)

[2.3.9 RatingBar\(星级评分条\)](#)

[2.4.1 ScrollView\(滚动条\)](#)

本节引言：

本节给大家带的是ViewFlipper，它是Android自带的一个多页面管理控件，且可以自动播放！和ViewPager不同，ViewPager是一页一页的，而ViewFlipper则是一层层的，和ViewPager一样，很多时候，用来实现进入应用后的引导页，或者用于图片轮播，本节我们就使用ViewFlipper写一个简单的图片轮播的例子吧~官方

API：[ViewFlipper](#)

1.为ViewFlipper加入View的两种方法

1) 静态导入

所谓的静态导入就是像图中这样，把个个页面添加到ViewFlipper的中间！

```
<ViewFlipper
    android:id="@+id/vflp_help"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <include layout="@layout/page_help_one" />

    <include layout="@layout/page_help_two" />

    <include layout="@layout/page_help_three" />

    <include layout="@layout/page_help_four" />

</ViewFlipper>
```

2) 动态导入

通过addView方法填充View

```
vflp_help = (ViewPager) findViewById(R.id.vflp_help);  
vflp_help.addView(v1);  
vflp_help.addView(v2);  
vflp_help.addView(v3);  
vflp_help.addView(v4);
```

2.常用的一些方法

setInAnimation : 设置View进入屏幕时使用的动画

setOutAnimation : 设置View退出屏幕时使用的动画

showNext : 调用该方法来显示ViewPager里的下一个View

showPrevious : 调用该方法来显示ViewPager的上一个View

setFlipInterval : 设置View之间切换的时间间隔

setFlipping : 使用上面设置的时间间隔来开始切换所有的View，切换会循环进行

stopFlipping : 停止View切换

3.使用实例

1) 示例1：使用ViewPager实现图片轮播(静态导入)

实现效果图：



实现代码：

每个页面的布局都是一个简单的ImageView，这里就不贴了~先贴下两个进入以及离开的动画：

right_in.xml：

```
<?xml version="1.0" encoding="utf-8"?>
```

2.4.2 Date & Time组件(上)

2.4.3 Date & Time组件(下)

2.4.4 Adapter基础讲解

2.4.5 ListView简单实用

2.4.6 BaseAdapter优化

2.4.7 ListView的焦点问题

2.4.8 ListView之checkbox错...

2.4.9 ListView的数据更新问题

2.5.0 构建一个可复用的自定...

2.5.1 ListView Item多布局的...

2.5.2 GridView(网格视图)的...

2.5.3 Spinner(列表选项框)...

2.5.4 AutoCompleteTextVie...

2.5.5 ExpandableListView(...

2.5.6 ViewPager(翻转视图)...

2.5.7 Toast(吐司)的基本使用

2.5.8 Notification(状态栏通...

2.5.9 AlertDialog(对话框)详解

2.6.0 其他几种常用对话框基...

2.6.1 PopupWindow(悬浮框...

2.6.2 菜单(Menu)

2.6.3 ViewPager的简单使用

2.6.4 DrawerLayout(官方侧...

3.1.1 基于监听的事件处理机制

3.2 基于回调的事件处理机制

3.3 Handler消息传递机制浅析

3.4 TouchListener PK OnTo...

3.5 监听EditText的内容变化

3.6 响应系统设置的事件(Co...

3.7 AsyncTask异步任务

3.8 Gestures(手势)

4.1.1 Activity初学乍练

4.1.2 Activity初窥门径

4.1.3 Activity登堂入室

4.2.1 Service初涉

4.2.2 Service进阶

4.2.3 Service精通

4.3.1 BroadcastReceiver牛...

4.3.2 BroadcastReceiver庖...

4.4.1 ContentProvider初探



反馈

```
<set xmlns:android="http://schemas.android.com/apk/res/android"
>

    <translate
        android:duration="2000"
        android:fromXDelta="100%p"
        android:toXDelta="0" />

</set>
```

right_out.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
>

    <translate
        android:duration="2000"
        android:fromXDelta="0"
        android:toXDelta="-100%p" />

</set>
```

然后是activity_main.xml布局文件：

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ViewFlipper
        android:id="@+id/vflp_help"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:inAnimation="@anim/right_in"
        android:outAnimation="@anim/right_out"
        android:flipInterval="3000">

        <include layout="@layout/page_help_one" />

        <include layout="@layout/page_help_two" />

        <include layout="@layout/page_help_three" />

        <include layout="@layout/page_help_four" />

    </ViewFlipper>

</RelativeLayout>
```

4.4.2 ContentProvider再探...

4.5.1 Intent的基本使用

4.5.2 Intent之复杂数据的传递

5.1 Fragment基本概述

5.2.1 Fragment实例精讲—...

5.2.2 Fragment实例精讲—...

5.2.3 Fragment实例精讲—...

5.2.4 Fragment实例精讲—...

5.2.5 Fragment实例精讲—...

6.1 数据存储与访问之——文...

6.2 数据存储与访问之——S...

6.3.1 数据存储与访问之——...

6.3.2 数据存储与访问之——...

7.1.1 Android网络编程要学...

7.1.2 Android Http请求头与...

7.1.3 Android HTTP请求方...

7.1.4 Android HTTP请求方...

7.2.1 Android XML数据解析

7.2.2 Android JSON数据解析

7.3.1 Android 文件上传

7.3.2 Android 文件下载 (1)

7.3.3 Android 文件下载 (2)

7.4 Android 调用 WebService

7.5.1 WebView(网页视图)基...

7.5.2 WebView和JavaScrip...

7.5.3 Android 4.4后WebVie...

7.5.4 WebView文件下载

7.5.5 WebView缓存问题

7.5.6 WebView处理网页返...

7.6.1 Socket学习网络基础准备

7.6.2 基于TCP协议的Socket...

7.6.3 基于TCP协议的Socket...

7.6.4 基于UDP协议的Socke...

8.1.1 Android中的13种Draw...

8.1.2 Android中的13种Draw...

8.1.3 Android中的13种Draw...

8.2.1 Bitmap(位图)全解析 P...

8.2.2 Bitmap引起的OOM问题

8.3.1 三个绘图工具类详解

8.3.2 绘图类实战示例

这里我们设置了flipInterval = 3000，即每隔3000ms切还一个~ 最后我们只需在MainActivity.java中调用ViewFlipper的startFlipping()方法开始滑动！

MainActivity.java：

```
public class MainActivity extends AppCompatActivity {

    private ViewFlipper vflp_help;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        vflp_help = (ViewFlipper) findViewById(R.id.vflp_help);
        vflp_help.startFlipping();
    }
}
```

2) 示例2：支持手势滑动的ViewFlipper(动态导入)

实现效果图：



代码实现：

因为我们分为进入上一页，进入下一页，所以除了上面的两个动画外，我们再添加两个动画：

left_in.xml：

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    >

    <translate
        android:duration="500"
        android:fromXDelta="-100%p"
```

8.3.3 Paint API之—— Mask...

8.3.4 Paint API之—— Xferm...

8.3.5 Paint API之—— Xferm...

8.3.6 Paint API之—— Xferm...

8.3.7 Paint API之—— Xferm...

8.3.8 Paint API之—— Xferm...

8.3.9 Paint API之—— Color...

8.3.10 Paint API之—— Colo...

8.3.11 Paint API之—— Colo...

8.3.12 Paint API之—— Path...

8.3.13 Paint API之—— Sha...

8.3.14 Paint几个枚举/常量值...

8.3.15 Paint API之——Type...

8.3.16 Canvas API详解(Part 1)

8.3.17 Canvas API详解(Part...

8.3.18 Canvas API详解(Part...

8.4.1 Android动画合集之帧...

8.4.2 Android动画合集之补...

8.4.3 Android动画合集之属...

8.4.4 Android动画合集之属...

9.1 使用SoundPool播放音...

9.2 MediaPlayer播放音频与...

9.3 使用Camera拍照

9.4 使用MediaRecord录音

10.1 TelephonyManager(电...

10.2 SmsManager(短信管理...

10.3 AudioManager(音频管...

10.4 Vibrator(振动器)

10.5 AlarmManager(闹钟服务)

10.6 PowerManager(电源服...

10.7 WindowManager(窗口...

10.8 LayoutInflater(布局服务)

10.9 WallpaperManager(壁...

10.10 传感器专题(1)——相...

10.11 传感器专题(2)——方...

10.12 传感器专题(3)——加...

10.12 传感器专题(4)——其...

10.14 Android GPS初涉

11.0 《2015最新Android基...

```

        android:toXDelta="0" />

</set>

```

left_out.xml :

```

<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
>

    <translate
        android:duration="500"
        android:fromXDelta="0"
        android:toXDelta="100%p" />

</set>

```

MainActivity.java :

```

public class MainActivity extends AppCompatActivity {

    private Context mContext;
    private ViewFlipper vflp_help;
    private int[] resId = {R.mipmap.ic_help_view_1,R.mipmap.ic_
help_view_2,
        R.mipmap.ic_help_view_3,R.mipmap.ic_help_view_4};

    private final static int MIN_MOVE = 200;    //最小距离
    private MyGestureListener mListener;
    private GestureDetector mDetector;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mContext = MainActivity.this;
        //实例化SimpleOnGestureListener与GestureDetector对象
        mListener = new MyGestureListener();
        mDetector = new GestureDetector(this, mListener);
        vflp_help = (ViewFlipper) findViewById(R.id.vflp_help);
        //动态导入添加子View
        for(int i = 0;i < resId.length;i++){
            vflp_help.addView(getImageView(resId[i]));
        }

    }

    //重写onTouchEvent触发MyGestureListener里的方法
    @Override
    public boolean onTouchEvent(MotionEvent event) {
        return mDetector.onTouchEvent(event);
    }
}

```

```
//自定义一个GestureListener,这个是View类下的，别写错哦！！

private class MyGestureListener extends GestureDetector.SimpleOnGestureListener {
    @Override
    public boolean onFling(MotionEvent e1, MotionEvent e2,
float v, float v1) {
        if(e1.getX() - e2.getX() > MIN_MOVE){
            vflp_help.setInAnimation(mContext,R.anim.right_in);
            vflp_help.setOutAnimation(mContext, R.anim.right_out);
            vflp_help.showNext();
        }else if(e2.getX() - e1.getX() > MIN_MOVE){
            vflp_help.setInAnimation(mContext,R.anim.left_in);
            vflp_help.setOutAnimation(mContext, R.anim.left_out);
            vflp_help.showPrevious();
        }
        return true;
    }
}

private ImageView getImageView(int resId){
    ImageView img = new ImageView(this);
    img.setBackgroundResource(resId);
    return img;
}
}
```

代码要点解析：

1.这里我们通过动态的方法添加View，这里只是简单的ImageView，可根据自己需求进行扩展！2.相信细心的你发现了，这里我们的手势用的不是通过onTouchEvent直接判断的，然后重写 onTouchEvent事件，对Action进行判断，然后如果是MotionEvent.ACTION_MOVE的话，就执行下述代码：

```
if(event.getX() > startX){
    vflp_help.setInAnimation(this,R.anim.left_in);
    vflp_help.setOutAnimation(this, R.anim.left_out);
    vflp_help.showPrevious();
}else if(startX > event.getX()){
    vflp_help.setInAnimation(this,R.anim.right_in);
    vflp_help.setOutAnimation(this, R.anim.right_out);
    vflp_help.showNext();
}
```

后来发现，模拟器上因为是鼠标的关系，并不会频繁的抖动，而真机上，因为手指一直是颤抖的 所以ACTION_MOVE会一直触发，然后View一直切换，后来考虑了Berial(B神)的建议，加入了一个值来进行判断，就是添加一个标志：


```
switch (event.getAction()){
    case MotionEvent.ACTION_DOWN:
        startX = event.getX();
        isChange = true;
        break;
    case MotionEvent.ACTION_MOVE:
        if(isChange){
            //向右滑动, 看前一页
            if(event.getX() > startX){
                vflp_help.setInAnimation(this,R.anim.left_in);
                vflp_help.setOutAnimation(this, R.anim.left_out);
                vflp_help.showPrevious();
            }else if(startX > event.getX()){
                vflp_help.setInAnimation(this,R.anim.right_in);
                vflp_help.setOutAnimation(this, R.anim.right_out);
                vflp_help.showNext();
            }
            isChange = false;
        }
        break;
    case MotionEvent.ACTION_UP:
```

可以是可以，不过感觉还是有点不流畅，怪怪的，后来想想还是用手势类，直接在onFling处理 就好，于是就有了上面的代码，运行起来杠杠滴~ 当然，如果你对Gesture手势不熟悉的话，可以参见之前写过的一篇文章：[Android基础入门教程——3.8 Gesture\(手势\)](#)

4.代码示例下载

- [ViewPagerDemo.zip](#)
- [ViewPagerDemo2.zip](#)

本节小结：

好的，本节给大家讲解了ViewPager(翻转视图)的基本使用，以后做图片轮播和引导页，你就多了一个选择了~嗯，就说这么多，谢谢~

| | | | |
|---|---|--|---|
| 在线实例 | 字符集&工具 | 最新更新 | 站点信息 |
| <ul style="list-style-type: none">· HTML 实例· CSS 实例· JavaScript 实例· Ajax 实例· jQuery 实例· XML 实例 | <ul style="list-style-type: none">· HTML 字符集设置· HTML ASCII 字符集· HTML ISO-8859-1· HTML 实体符号 | <ul style="list-style-type: none">· Docker 清理命令· 一些docker的技...· PHP接收 json，并...· Foundation CSS ... | <ul style="list-style-type: none">· 意见反馈· 免责声明· 关于我们· 文章归档 |

· Java 实例

· HTML 拾色器

· Foundation CSS ...

· JSON 格式化工具

· Foundation 图标...

· Foundation 网格...

关注微信



Copyright © 2013-2015 菜鸟教程 **runoob.com** All Rights Reserved. 备案号：闽ICP备15012807号-1