

5.1 Fragment基本概述

分类 [Android 基础入门教程](#)

本节引言

好的，在上一章中我们把Android的四大组件Activity，Service，BroadcastReceiver，ContentProvider 以及他们之间的纽带：Intent，都撸了一遍，而本章节给大家带来的是一个Fragment(碎片)的东西，本节我们就来介绍这个Fragment的一些基本概念以及用法！官方文档：[Fragment](#)

1.基本概念

1) 它是什么鬼，有什么用？

答：Fragment是Android3.0后引入的一个新的API，他出现的初衷是为了适应大屏幕的平板电脑，当然现在他仍然是平板APP UI设计的宠儿，而且我们普通手机开发也会加入这个Fragment，我们可以把他看成一个小型的Activity，又称Activity片段！想想，如果一个很大的界面，我们就一个布局，写起界面来会有多麻烦，而且如果组件多的话是管理起来也很麻烦！而使用Fragment 我们可以把屏幕划分成几块，然后进行分组，进行一个模块化的管理！从而可以更加方便的在运行过程中动态地更新Activity的用户界面！另外Fragment并不能单独使用，他需要嵌套在Activity 中使用，尽管他拥有自己的生命周期，但是还是会受到宿主Activity的生命周期的影响，比如Activity 被destory销毁了，他也会跟着销毁！

下图是文档中给出的一个Fragment分别对应手机与平板间不同情况的处理图：



反馈

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)



Fragment的生命周期图

```
graph TD
    Start([添加Fragment]) --> Inflate[onInflate()]
    Inflate --> Attach[onAttach()]
    Attach --> Create[onCreate()]
    Create --> CreateView[onCreateView()]
    CreateView --> ActivityCreate[onActivityCreated()]
    ActivityCreate --> Start[onStart()]
    Start --> Resume[onResume()]
    Resume --> Running([运行状态])
    Running --> Pause[onPause()]
    Pause --> Paused([暂停状态])
    Paused --> Stop[onStop()]
    Stop --> Stopped([停止状态])
    Stopped --> DestroyView[onDestroyView()]
    DestroyView --> Destroy[onDestroy()]
    Destroy --> Detach[onDetach()]
    Detach --> Destroyed([销毁状态])
    DestroyView -- "该Fragment从Back栈返回界面" --> CreateView
```

ps:开发时可以按需要覆盖对应的回调方法,至少写: onCreateView() 返回view

该方法只在我们直接用标签在布局文件中定义的时候才会被调用

当该Fragment被添加到Activity中会回调,只会被调用一次

创建Fragment时回调,只会被调用一次

每次创建,绘制该Fragment的View组件时回调,会将显示的View返回

当Fragment所在的Activity启动完成后回调

启动Fragment时被回调

恢复Fragment时被回调, onStart()方法后一定回调onResume()方法
onStart可见, onResume后才能交互

①该Activity转到后台,或者Fragment被删除/替换
②该Fragment被添加到Back栈

暂停Fragment时被回调

停止Fragment时被回调

销毁该Fragment所包含的View组件时使用

销毁Fragment时被毁掉

将该Fragment从Activity被删除/替换完成后回调该方法; onDestroy()方法后一定会回调该方法.该方法只调用一次

销毁状态

- 2/10

3) 核心要点：

下面说下使用Fragment的一些要点：

3.0版本后引入,即minSdk要大于11

Fragment需要嵌套在Activity中使用,当然也可以嵌套到另外一个Fragment中,但这个被嵌套的Fragment也是需要嵌套在Activity中的,间接地说,Fragment还是需要嵌套在Activity中!! 受寄主Activity的生命周期影响,当然他也有自己的生命周期!另外不建议在Fragment里面 嵌套Fragment因为嵌套在里面的Fragment生命周期不可控!!!

官方文档说创建Fragment时至少需要实现三个方法: onCreate(), onCreateView(), onPause(); 不过貌似只写一个onCreateView也是可以的...

Fragment的生命周期和Activity有点类似:三种状态:

Resumed:在允许中的Fragment可见

Paused:所在Activity可见,但是得不到焦点

Stope: ①调用addBackStack(), Fragment被添加到Bcak栈 ②该Activity转向后台,或者该Fragment被替换/删除

ps:停止状态的fragment仍然活着(所有状态和成员信息被系统保持着),然而,它对用户 不再可见,并且如果activity被干掉,他也会被干掉.

4) Fragment的几个子类：

ps:很多时候我们都是直接重写Fragment,inflate加载布局完成相应业务了,子类用的不多,等需要的时候在深入研究!

对话框:DialogFragment

列表:ListFragment

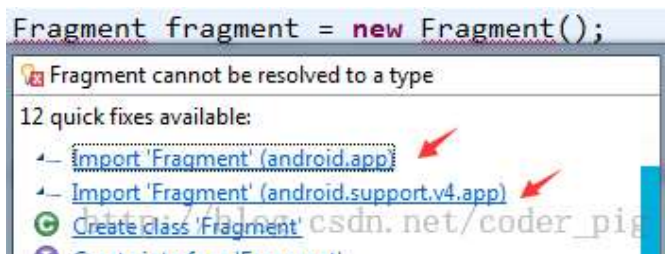
选项设置:PreferenceFragment

WebView界面:WebViewFragment

5) 是用App包下的Fragment还是v4包下的：

问题概述：

相信很多朋友在使用Fragment的时候都会遇到下面这种情况：



那么我们到底是使用android.app下的Fragment还是用的android.support.v4.app包下的Fragment呢？

答：其实都可以，前面说过Fragment是Android 3.0(API 11)后引入的，那么如果开发的app需要在3.0以下的版本运行呢？比如还有一点市场份额的2.3！于是乎，v4包就这样应运而生了，而最低可以兼容到

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实例精讲一...

5.2.2 Fragment实例精讲一...

5.2.3 Fragment实例精讲一...

5.2.4 Fragment实例精讲一...

5.2.5 Fragment实例精讲一...

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JavaScip...

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

1.6版本！至于使用哪个包看你的需求了,现在3.0下手机市场份额其实已经不多,随街都是4.0以上的,6.0十月份都出了,你说呢...所以这个时候,你可以直接使用app包下的Fragment 然后调用相关的方法,通常都是不会有什么问题的;如果你Fragment用了app包的,FragmentManager和FragmentTransaction都需要是app包的！要么用全部用app,要么全部用v4,不然可是会报错的哦!当然如果你要自己的app对于低版本的手机也兼容的话,那么就可以选择用v4包！

使用v4包下Fragment要注意的地方：

①如果你使用了v4包下的Fragment,那么所在的那个Activity就要继承FragmentActivity哦! 案例:今天在xml文件中静态地载入fragment,然后重写了Fragment,但是在加载Activity的时候就报错了, 大概的提示就是Fragment错误还是找不到什么的,name属性改了几次还是错!最后才发现是用了 v4的包的缘故,只需让自己的Activity改成FragmentActivity即可!

②之前写了下面这段代码, 然后报错：

```
getFragmentManager().beginTransaction().replace(R.id.fragment1, fragment);
```

The method replace(int, Fragment) in the type FragmentTransaction is not applicable for the arguments (int, Fragment)

1 quick fix available:

Change type of 'fragment' to 'FragmentManager'

http://blog.csdn.net/coder_pig

有点莫名其妙啊,Fragment,FragmentManager,FragmentTransaction都是用的v4包啊, Activity也是继承FragmentActivity的啊?都改成app包就可以了,但是这不和我们用v4包的 前提冲突了么?其实也是有解决方法的哈?

答:只需要把getFragmentManager()改成getSupportFragmentManager()就可以了

2.创建一个Fragment

1) 静态加载Fragment

实现流程：

静态加载Fragment的流程



示例代码：

Step 1:定义Fragment的布局，就是fragment显示内容的

Step 2:自定义一个Fragment类,需要继承Fragment或者他的子类,重写onCreateView()方法 在该方法中调用inflater.inflate()方法加载Fragment的布局文件,接着返回加载的view对象

```
public class Fragmentone extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
    container,
```

- 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...

```
        Bundle savedInstanceState) {  
            View view = inflater.inflate(R.layout.fragment1, container, false);  
            return view;  
        }  
    }  
}
```

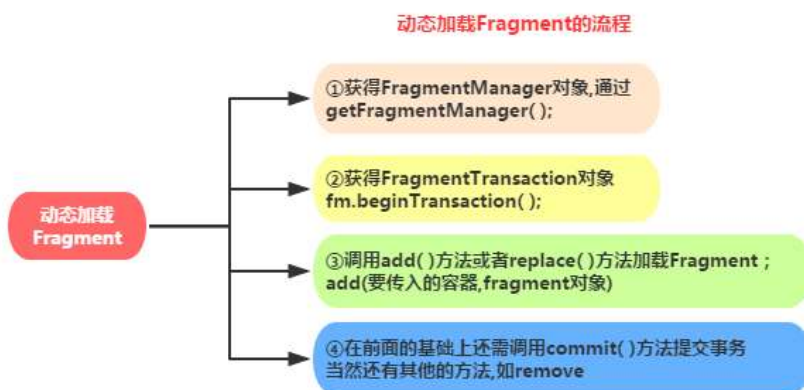
Step 3:在需要加载Fragment的Activity对应的布局文件中添加fragment的标签，记住，name属性是全限定类名哦，就是要包含Fragment的包名，如：

```
<fragment  
    android:id="@+id/fragment1"  
    android:name="com.jay.example.fragmentdemo.Fragmentone"  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1" />
```

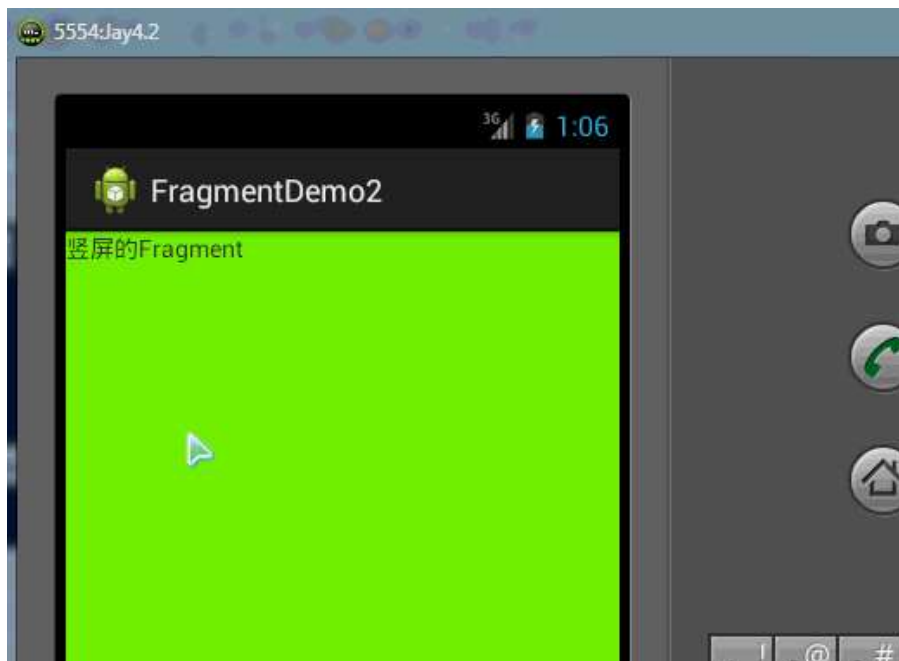
Step 4: Activity在onCreate()方法中调用setContentView()加载布局文件即可！

2) 动态加载Fragment

实现流程：



示例代码： 这里演示的是，当横竖屏切换的时候地切换Fragment：



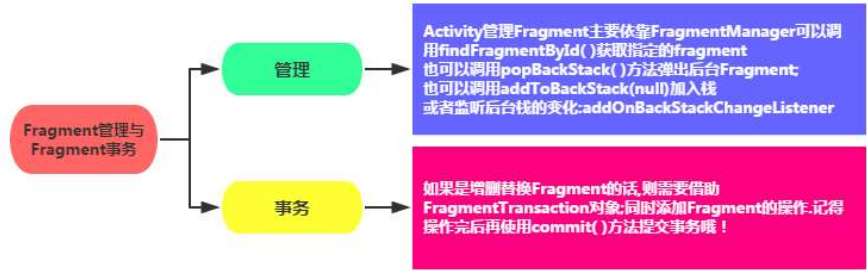
Fragment以及布局代码就不贴出来了，直接贴MainActivity的关键代码：

```
public class MainActivity extends Activity {

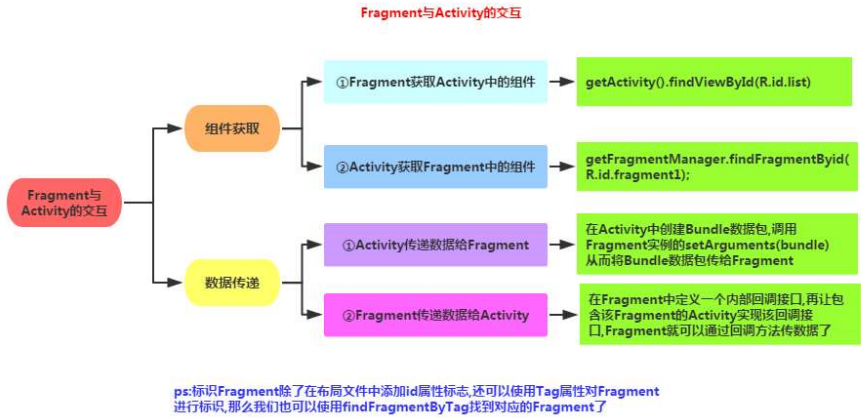
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Display dis = getWindowManager().getDefaultDisplay();
        if(dis.getWidth() > dis.getHeight())
        {
            Fragment1 f1 = new Fragment1();
            getFragmentManager().beginTransaction().replace(R.id.LinearLayout1, f1).commit();
        }

        else
        {
            Fragment2 f2 = new Fragment2();
            getFragmentManager().beginTransaction().replace(R.id.LinearLayout1, f2).commit();
        }
    }
}
```

3.Fragment管理与Fragment事务



4.Fragment与Activity的交互



可能有的朋友不喜欢看图，接下来用文字介绍下吧：

1) 组件获取

Fragment获得Activity中的组件: `getActivity().findViewById(R.id.list)` ;
Activity获得Fragment中的组件(根据id和tag都可
以) : `getFragmentManager.findFragmentById(R.id.fragment1)`;

2) 数据传递

①Activit传递数据给Fragment:

在Activity中创建Bundle数据包,调用Fragment实例的 `setArguments(bundle)` 从而将Bundle数据包传给Fragment,然后Fragment中调用 `getArguments` 获得 Bundle对象,然后进行解析就可以了

②Fragment传递数据给Activity

在Fragment中定义一个内部回调接口,再让包含该Fragment的Activity实现该回调接口, Fragment就可以通过回调接口传数据了,回调,相信很多人都知道是什么玩意,但是 写不出来啊,网上的一百度"fragment传数据给Activity",全是李刚老师的那个代码,真心无语算了,这里就写下局部代码吧,相信读者一看就懂的了:

Step 1:定义一个回调接口:(Fragment中)

```
/*接口*/
public interface CallBack{
```

```

    /*定义一个获取信息的方法*/
    public void getResult (String result);
}

```

Step 2 : 接口回调 (Fragment中)

```

/*接口回调*/
public void getData (CallBack callBack) {
    /*获取文本框的信息,当然你也可以传其他类型的参数,看需求咯*/
    String msg = editText.getText().toString();
    callBack.getResult(msg);
}

```

Step 3:使用接口回调方法读数据(Activity中)

```

/* 使用接口回调的方法获取数据 */
leftFragment.getData (new CallBack() {
    @Override
    public void getResult (String result) {
        /*打印信息*/
        Toast.makeText (MainActivity.this, "-->" + result,
1).show();
    }
});

```

总结下方法： ->在Fragment定义一个接口,接口中定义抽象方法,你要传什么类型的数据参数就设置为什么类型;

->接着还有写一个调用接口中的抽象方法,把要传递的数据传过去

->再接着就是Activity了,调用Fragment提供的那个方法,然后重写抽象方法的时候进行数据 的读取就可以了!!!

③Fragment与Fragment之间的数据互传

其实这很简单,找到要接受数据的fragment对象,直接调用 setArguments传数据进去就可以了 通常的话是replace时,即 fragment跳转的时候传数据的,那么只需要在初始化要跳转的 Fragment 后调用他的setArguments方法传入数据即可! 如果是两个Fragment需要即时传数据,而非跳转的话,就需要先在 Activity获得f1传过来的数据, 再传到f2了,就是以Activity为媒介~

示例代码如下：

```

FragmentManager fManager = getSupportFragmentManager();
FragmentTransaction fTransaction = fManager.beginTransaction();
Fragmentthree t1 = new Fragmentthree();
Fragmenttwo t2 = new Fragmenttwo();
Bundle bundle = new Bundle();
bundle.putString("key",id);
t2.setArguments(bundle);

```



```
fTransaction.add(R.id.fragmentRoot, t2, "~~~");
fTransaction.addToBackStack(t1);
fTransaction.commit();
```

5.走一次生命周期图：

思前想后还是决定要带大家简单的走一趟生命周期图，加深大家对Fragment生命周期的理解：

- ①Activity加载Fragment的时候,依次调用下面的方法: **onAttach -> onCreate -> onCreateView -> onActivityCreated -> onStart -> onResume**
- ②当我们弄出一个悬浮的对话框风格的Activity,或者其他,就是让Fragment所在的Activity可见,但不获得焦点 **onPause**
- ③当对话框关闭,Activity又获得了焦点: **onResume**
- ④当我们替换Fragment,并调用addToBackStack()将他添加到Back栈中 **onPause -> onStop -> onDestroyView** !! 注意,此时的Fragment还没有被销毁哦!!!
- ⑤当我们按下键盘的回退键, Fragment会再次显示出来:
onCreateView -> onActivityCreated -> onStart -> onResume
- ⑥如果我们替换后,在事务commit之前**没有调用addToBackStack()**方法将 Fragment添加到back栈中的话;又或者退出了Activity的话,那么Fragment将会被完全结束, Fragment会进入销毁状态 **onPause -> onStop -> onDestroyView -> onDestroy -> onDetach**

本节小结：

本节跟大家讲解了以下Fragment一些基本的概念以及简单的用法，相信大家会慢慢喜欢上 Fragment的，因为篇幅的关系，本节就写这么多，下一节我们带大家来写一些关于Fragment 的常用实例，敬请期待，谢谢~

← 4.5.2 Intent之复杂数据的传递

5.2.1 Fragment实例精讲——底部导航栏的实现(方法1) →

在线实例

- HTML 实例
- CSS 实例
- JavaScript 实

字符集&工具

- HTML 字符集设置
- HTML ASCII 字符集

最新更新

- Swift 正式开源
- PHP 7 正式发布

站点信息

- 意见反馈
- 免责声明
- 关于我们

	<div>例</div> <div><div>· Ajax 实例</div><div>· jQuery 实例</div><div>· XML 实例</div><div>· Java 实例</div></div>	<div><div>· HTML ISO-8859-1</div><div>· HTML 实体符号</div><div>· HTML 拾色器</div><div>· JSON 格式化工具</div></div>	<div><div>· Shell 编程快速入门</div><div>· Shell 文件包含</div><div>· Shell 输入/输出...</div><div>· Shell printf 命令</div><div>· Shell 基本运算符</div></div>	<div><div>· 文章归档</div></div> <div><div>关注微信</div><div></div><div><div>Copyright © 2013-2015 菜鸟教程 runoob.com All Rights Reserved. 备案号：闽ICP备15012807号-1</div></div></div>
--	---	---	--	---