Android中Fragment数据保存和恢复

作者 MeloDev (/u/f5909165c1e8) + 关注
2016.05.16 23:25* 字数 1467 阅读 4725 评论 12 喜欢 79 赞赏 1
(/u/f5909165c1e8)

Android中Fragment数据保存和恢复

> 本文独家授权给stormzhang运营的公众号AndroidDeveloper，拒绝其他任何形式的转载。

写在前面：

上周我们总结了Activity中数据的保存和恢复，我们花两分钟来回顾一下：

Android中Activity数据的保存和恢复 (http://www.jianshu.com/p/6622434511f7)

一句话总结：

- 临时数据
  对于**临时数据**，我们使用**onSaveInstanceState**方法进行保存，并且在**onCreate**方法中恢复。

- 永久数据
  对于**持久性数据**，我们要在onPause方法中进行存储，但是要注意，onPause方法中不能进行大量操作，会影响其他Activity进入任务栈栈顶。

ps：在Activity中弹出一个当前Activity的**Dialog**并不会有任何生命周期方法调用（以前我曾以为会调用onPause方法）。因为Dialog作为一个View本身就是属于当前Activity的，Activity并没有**失去焦点**。

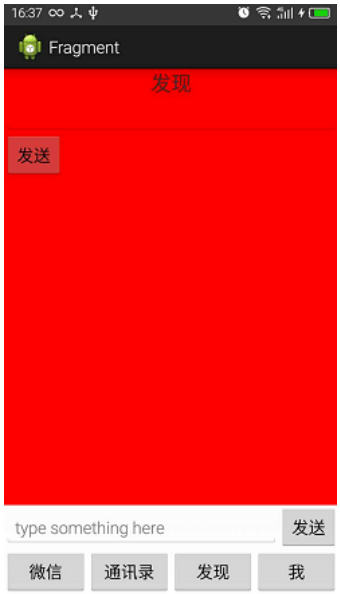
ok，完成了回顾，下面来开始本篇博客：

Fragment在我们的项目中真的太实用和常见了，它的使用频率和数量甚至超过了Activity，所以本文目的是探究Fragment的数据保存和恢复。

在开始讲解之前，你应该对Fragment的**生命周期方法**有一定了解，推荐给大家一篇博客，我认为不错：

Fragment生命周期方法详解
(http://blog.csdn.net/wanghao200906/article/details/45561385)

准备工作做了这么多，下面我们正式开始吧！



测试App截图

本文直接选用了《第一行代码》中Fragment模块的讲解例子，点击下面的按钮**分别跳转**这四个Fragment。为了方便观察，我重写了Fragment所有**生命周期方法**和**onSaveInstanceState**方法，并打印了Log。

我们目的是探究Fragment数据的保存和恢复，在这里我把它分为两大类的情况：

1. 单个Fragment遭遇一些突发情况

2. Fragment之间相互的切换或覆盖

在此之前，先引入一个**返回栈**的概念。
我想你应该知道返回栈是什么，并且你以前接触的应该是保存Activity的返回栈，类比Activity，Fragment返回栈其实是保存Fragment的栈结构。区别在于：**Fragment的返回栈由Activity管理；而Activity的返回栈由系统管理。**

在未修改之前，本文添加并切换Fragment的方式都是在返回栈中**仅有一个** fragment：

```
public void weiXin(View v){
    FragmentManager fm = getFragmentManager();
    FragmentTransaction ft = fm.beginTransaction();
    ft.replace(R.id.fm, fg1);
    ft.commit();
}
```

添加切换fragment

不要心急，过一会再说怎么去在返回栈中压入多个fragment，我们先来处理**只有一个**的情况

1. 单个Fragment遭遇突发情况

   仍然是用以下**突发情况**进行测试：

   - 点击back键
   - 点击锁屏键
   - 点击home键
   - 其他APP进入前台

- 启动了另一个Activity
- 屏幕方向旋转
- APP被Kill

不过与上篇博客不同的是，我们在清单文件中，给Activity做了如下配置：

```
<activity
    android:name=".MainActivity"
    android:configChanges="keyboardHidden|orientation|screenSize"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

配置configChange

这么做的目的是当屏幕方向发生改变的时候，fragment所依附的Activity并不会重新销毁再创建，让**情况相对简单一点。**

测试结果：

当一个fragment孤零零地呆在返回栈时，它所处的情况与Activity如出一辙。类比Activity对数据的保存和恢复，我们可以对此得出结论：

- 临时数据 对于**临时数据**，我们使用**onSaveInstanceState**方法进行保存，并且在**onCreateView**方法中恢复（请注意是onCreateView）。
- 永久数据 对于**持久性数据**，我们要在onPause方法中进行存储。

2. Fragment之间的相互切换或覆盖
   当返回栈中保证**只有一个Fragment**，相互切换时，生命周期方法的调用是怎样的呢？例如本例中，从fragment03切换到fragment04：

L...	Time	PID	TID	Application	Tag	Text
E	05-16 21:25:54.929	8249	8249	com.lessismore.f...	Fragment03	onAttach
E	05-16 21:25:54.929	8249	8249	com.lessismore.f...	Fragment03	onCreate
E	05-16 21:25:54.929	8249	8249	com.lessismore.f...	Fragment03	onCreateView
E	05-16 21:25:54.933	8249	8249	com.lessismore.f...	Fragment03	onActivityCreated
E	05-16 21:25:54.933	8249	8249	com.lessismore.f...	Fragment03	onStart
E	05-16 21:25:54.934	8249	8249	com.lessismore.f...	Fragment03	onResume
E	05-16 21:25:57.430	8249	8249	com.lessismore.f...	Fragment03	onPause
E	05-16 21:25:57.430	8249	8249	com.lessismore.f...	Fragment03	onStop
E	05-16 21:25:57.431	8249	8249	com.lessismore.f...	Fragment03	onDestroyView
E	05-16 21:25:57.432	8249	8249	com.lessismore.f...	Fragment03	onDestroy
E	05-16 21:25:57.433	8249	8249	com.lessismore.f...	Fragment03	onDetach
E	05-16 21:25:57.433	8249	8249	com.lessismore.f...	Fragment04	onAttach
E	05-16 21:25:57.433	8249	8249	com.lessismore.f...	Fragment04	onCreate
E	05-16 21:25:57.433	8249	8249	com.lessismore.f...	Fragment04	onCreateView
E	05-16 21:25:57.435	8249	8249	com.lessismore.f...	Fragment04	onActivityCreated
E	05-16 21:25:57.435	8249	8249	com.lessismore.f...	Fragment04	onStart
E	05-16 21:25:57.435	8249	8249	com.lessismore.f...	Fragment04	onResume

fragment03切换fragment04

可以看到，上述的这种情况，两个fragment从创建到销毁，经历了所有的生命周期方法。

如果返回栈中fragment的数量为多个呢？首先在切换时，加上以下代码，保证将fragment放入返回栈中：

```
public void faXian(View v) {
    FragmentManager fm = getFragmentManager();
    FragmentTransaction ft = fm.beginTransaction();
    ft.replace(R.id.fm, fg3);
    ft.addToBackStack(null);
    ft.commit();
}

public void wo(View v) {
    Fragment04 fg4 = new Fragment04();
    FragmentManager fm = getFragmentManager();
    FragmentTransaction ft = fm.beginTransaction();
    ft.replace(R.id.fm, fg4);
    ft.addToBackStack(null);
    ft.commit();
}
```

addToBackStack

使用addToBackStack方法，就能将fragment放入相应的返回栈中去了，从表象上来看区别在于进入其他fragment时，点击back键时，可以返回上一个fragment。这时候切换时，生命周期方法就是如何调用的呢？

Level	Time	PID	TID	Application	Tag	Text
E	05-16 21:58:47.568	18497	18497	com.lessismore.f...	Fragment03	onAttach
E	05-16 21:58:47.568	18497	18497	com.lessismore.f...	Fragment03	onCreate
E	05-16 21:58:47.568	18497	18497	com.lessismore.f...	Fragment03	onCreateView
E	05-16 21:58:47.578	18497	18497	com.lessismore.f...	Fragment03	onActivityCreated
E	05-16 21:58:47.579	18497	18497	com.lessismore.f...	Fragment03	onStart
E	05-16 21:58:47.583	18497	18497	com.lessismore.f...	Fragment03	onResume
E	05-16 21:58:50.964	18497	18497	com.lessismore.f...	Fragment03	onPause
E	05-16 21:58:50.964	18497	18497	com.lessismore.f...	Fragment03	onStop
E	05-16 21:58:50.964	18497	18497	com.lessismore.f...	Fragment03	onDestroyView
E	05-16 21:58:50.965	18497	18497	com.lessismore.f...	Fragment04	onAttach
E	05-16 21:58:50.965	18497	18497	com.lessismore.f...	Fragment04	onCreate
E	05-16 21:58:50.965	18497	18497	com.lessismore.f...	Fragment04	onCreateView
E	05-16 21:58:50.967	18497	18497	com.lessismore.f...	Fragment04	onActivityCreated
E	05-16 21:58:50.967	18497	18497	com.lessismore.f...	Fragment04	onStart
E	05-16 21:58:50.967	18497	18497	com.lessismore.f...	Fragment04	onResume

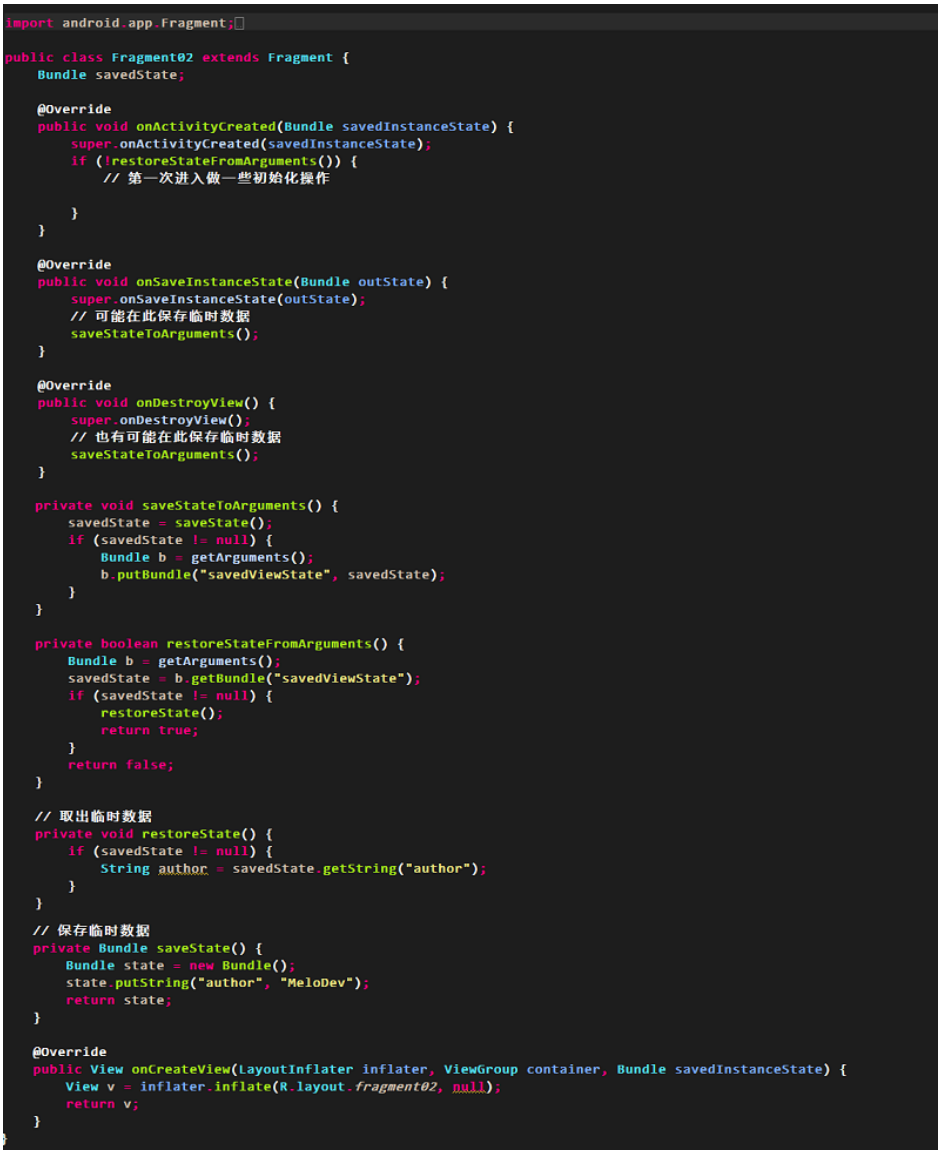
返回栈有多个fragment切换

对比这两张生命周期方法的图，能得出两个结论。

1.无论任务栈中fragment数量为多少，onSaveInstanceState方法都没有调用

2.当fragment任务栈中有多个fragment时，进入下一个fragment时，并不会销毁**fragment实例**，而是仅仅**销毁视图**，最终调用的方法为onDestoryView。

所以此时我们要去保存临时数据，并不能仅保存在onSaveInstanceState中（因为它可能不会调用），还应该在onDestoryView方法中进行保存临时数据的操作，源码如下：



```java
import android.app.Fragment;

public class Fragment02 extends Fragment {
    Bundle savedState;

    @Override
    public void onActivityCreated(Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);
        if (!restoreStateFromArguments()) {
            // 第一次进入做一些初始化操作

        }
    }

    @Override
    public void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        // 可能在此保存临时数据
        saveStateToArguments();
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        // 也有可能在此保存临时数据
        saveStateToArguments();
    }

    private void saveStateToArguments() {
        savedState = saveState();
        if (savedState != null) {
            Bundle b = getArguments();
            b.putBundle("savedViewState", savedState);
        }
    }

    private boolean restoreStateFromArguments() {
        Bundle b = getArguments();
        savedState = b.getBundle("savedViewState");
        if (savedState != null) {
            restoreState();
            return true;
        }
        return false;
    }

    // 取出临时数据
    private void restoreState() {
        if (savedState != null) {
            String author = savedState.getString("author");
        }
    }
    // 保存临时数据
    private Bundle saveState() {
        Bundle state = new Bundle();
        state.putString("author", "MeloDev");
        return state;
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment02, null);
        return v;
    }
}
```

代码截图

因为没有了系统提供的bundle参数，我们选择把数据保存在Arguments中，代码就不带着大家一步一步的看了，因为逻辑并不复杂，挺好理解的。通过这种方式，我们就挺容易的将**临时数据和fragment的一些状态**保存进bundle中并在需要时恢复了。

不知不觉本篇文章就要结束了，感兴趣的可以尝试当调用**ft.add()**方式去添加fragment时，生命周期方法又是怎样调用的呢？

结束之前我们来一句话总结下本文：
Fragment对**临时数据**的保存，仅仅依靠**onSaveInstanceState**方法是不行的，还需要在**onDestoryView**中进行相应操作，具体参考上面的代码。

Fragment中对于一些持久性的数据，仍应在**onPause**中保存。

感觉各位已经看到这里，不求赞赏，只求喜欢关注~

赞赏支持


(/u/2fb6988652a3)

♡ 喜欢 | 79


更多分享

(http://cwb.assets.jianshu.io/notes/images/398727


写下你的评论...

12条评论 只看作者 按喜欢排序 按时间正序 按时间倒序


戢鳞 (/u/520256894235)
2楼 · 2016.05.18 08:20
(/u/520256894235)
写的很好，思路清晰，Mark啦

赞 回复

> MeloDev (/u/f5909165c1e8)： @戢鳞 (/users/520256894235) 喜欢就好~可以照着例子敲敲，印象会深刻一点
> 2016.05.18 08:41 回复
>
> 添加新评论


Shaun白一辰 (/u/b7e9763ffd86)
3楼 · 2016.05.18 08:53
(/u/b7e9763ffd86)
mark, 有所收获，不过好像saveStateToArguments方法中不用空判断了，你的写法不太可能出现空在这里～

赞 回复


脱氧核糖核苷酸 (/u/5e02dfbe21ba)
4楼 · 2016.05.23 11:28
(/u/5e02dfbe21ba)
博主，我这边想请教个问题，我看这里关键恢复是利用的getArgument()，但是如果开始跳转的frament可能不需要携带参数过来，你这边不会出现报空指针的问题吗？

赞 回复

> MeloDev (/u/f5909165c1e8)： @脱氧核糖核苷酸 (/users/5e02dfbe21ba) 确实应该加一下非空判断
> 2016.05.23 12:11 回复
>
> 脱氧核糖核苷酸 (/u/5e02dfbe21ba)： @MeloDev (/users/f5909165c1e8) 博主，如果做了判空，但是getArgument()为空值，那还能通过什么做fragment的数组恢复呢
> 2016.05.25 13:51 回复
>
> MeloDev (/u/f5909165c1e8)： @脱氧核糖核苷酸 (/users/5e02dfbe21ba) 是这样的，在相互切换之前我给fragment人为的设置setArguments，保正它不为空就行了
> 2016.05.25 13:57 回复
>
> 添加新评论 | 还有2条评论， 展开查看

fumier (/u/d725f231c254)
5楼 · 2016.07.27 07:27
(/u/d725f231c254)
博主，Fragment之间的相互切换或覆盖，当返回栈中保证只有一个Fragment，相互切换时。。这个例子，是返回栈没有fragment的情况吧。可以通过以下代码监听返回栈的fragment的个数。
getSupportFragmentManager().addOnBackStackChangedListener(new FragmentManager.OnBackStackChangedListener() {
@Override (/users/d55323762b08)
public void onBackStackChanged() {
Log.d("backstack", getSupportFragmentManager().getBackStackEntryCount() + "");
}
});

赞 回复

> MeloDev (/u/f5909165c1e8)： @fumier (/users/d725f231c254) 感谢补充哇，啊哈哈
> 2016.07.27 09:58 回复

添加新评论

落雨收柴 (/u/c3c56175fd3a)
6楼 · 2017.07.04 22:00
(/u/c3c56175fd3a)
赞

赞 回复

被以下专题收入，发现更多相似内容

＋ 收入我的专题


技术 (/c/a9c9d2d89ae9?utm_source=desktop&utm_medium=notes-included-collection)


Android开发 (/c/d1591c322c89?utm_source=desktop&utm_medium=notes-included-collection)


程序员 (/c/NEt52a?utm_source=desktop&utm_medium=notes-included-collection)


首页投稿 (/c/bDHhpK?utm_source=desktop&utm_medium=notes-included-collection)


技术干货 (/c/38d96caffb2f?utm_source=desktop&utm_medium=notes-included-collection)


Android知识 (/c/3fde3b545a35?utm_source=desktop&utm_medium=notes-included-collection)


安卓资源收集 (/c/c2955a70fdd6?utm_source=desktop&utm_medium=notes-included-collection)

展开更多

推荐阅读 更多精彩内容 (/)

Android微信自动回复功能 (/p/f67e950d84f7?utm_ca…

Android微信自动回复功能本文原创，转载请经过本人准许。写在前面：最近接到老大的一个需求，要求在手机端拦截微信的通知（Notification），从而获得…

(/p/f67e950d84f7?utm_campaign=maleskine&utm_content=note&utm

MeloDev (/u/f5909165c1e8?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

开发一个Android应用之前，应该考虑点什么？ (/p/549c2fc9adfe?utm_ca…

开发一个Android应用之前，应该考虑点什么？本文原创，转载请注明出处。欢迎关注我的简书，关注我的专题AndroidClass我会长期坚持为大家收录简书上高质量的Android相关博文。开心一刻写在前面：昨天参加…

MeloDev (/u/f5909165c1e8?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

【真实故事】十年，从一穷二白到有房有车再到抑郁症… (/p/307b618e1a36?utm_campaign=maleskine&utm_content=note&utm

我是一个抑郁症患者，昨天确认了。 我一个人开车去的医院，家离医院也就十几分钟的车程，我竟然在医院附近转了近一个小时也没找到停车位，后来我…

谢晨晨 (/u/fa41b1cdd105?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

我就是那个一直单身，却被误以为有对象的人 (/p/6085… (/p/6085c7ad96cc?utm_campaign=maleskine&utm_content=note&utm

前几天，简书首页推了一篇文章，题目叫做《有对象还装单身，你的爱是假的吧？》，觉得有点意思，便点了进去。大致讲的就是真正爱对方的人总会想…

包子抹茶味 (/u/0769c2e4fc0b?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

你喜欢自己现在的状态吗？ (/p/6ffea3fc20b6?utm_ca… (/p/6ffea3fc20b6?utm_campaign=maleskine&utm_content=note&utm

1 今天是我44岁的生日。 人到中年，多多少少会有点危机感。所以，从40岁开始，我和家人已形成默契，每年生日蛋糕上插的蜡烛就定格在“28”岁上。 有…

Jk不二子 (/u/a04ea30a1097?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)