Android：我为何要封装DialogFragment？（上）

2016-03-02 安卓应用频道

(点击上方公众号，可快速关注)

> 来源：牛晓伟
> 链接：http://www.jianshu.com/p/af6499abd5c2

我为何要封装DialogFragment

最近在重构项目代码，项目中创建对话框用的是Dialog，AlertDialog。但是官方推出了DialogFragment来代替Dialog。那我就去认真的了解下DialogFragment。

DialogFragment

DialogFragment是在Android3.0的时候被引入的，从其名字可以很直观的看出它是一种基于Fragment的Dialog，可以用来创建对话框，它是用来替代Dialog的。一个新事物的出现是为了解决旧事物存在的问题，那不建议使用的Dialog存在什么问题呢？下面简单的说下。

Dialog存在问题：

- 在手机配置发生变化后（比如：旋屏后），变化之前显示的Dialog，变化之后不会显示，更别提Dialog状态的恢复了。
- 管理自定义的Dialog和系统原生的Dialog麻烦

DialogFragment怎么解决Dialog存在的问题：

- DialogFragment说到底还是一个Fragment，因此它继承了Fragment的所有特性。同理FragmentManager会管理DialogFragment。在手机配置发生变化的时候，FragmentManager可以负责现场的恢复工作。调用DialogFragment的setArguments(bundle)方法进行数据的设置，可以保证DialogFragment的数据也能恢复。
- DialogFragment里的onCreateView和onCreateDIalog 2个方法，onCreateView可以用来创建自定义Dialog，onCreateDIalog 可以用Dialog来创建系统原生Dialog。可以在一个类中管理2种不同的dialog。

用DialogFragment替代Dialog

既然DialogFragment有这些好处，那我就毅然决然的对项目中的Dialog用DialogFragment来进行替代。

重构的思路是这样的：

- 首先先创建一个ConfirmDialogFragment类（该类是用来创建确认对话框的），ConfirmDialogFragment类继承了DialogFragment。
- 其次在创建一个ProgressDialogFragment类（该类是用来创建进度对话框），同时它也继承了DialogFragment。
- 其他类型的Dialog就不举例了。
- 最后在BaseActivity（项目中所有Activity的基类）添加显示Dialog的方法，供BaseActivity的子类、Fragment、还有非Activity和非Fragment的类来调用。

我们先看下关键代码片段：

ConfirmDialogFragment中的代码片段：

```
    /**
      *用来创建确认对话框
    * Created by niuxiaowei on 2015/10/16.
 */
 public class ConfirmDialogFragment extends DialogFragment{            private ConfirmDialogListener mLis
tener;            //对外开放的接口
            public static interface ConfirmDialogListener extends DialogInterface.OnClickListener{
            }            /**
             * @param title
            * @param message
             * @param cancelable
             * @return
             */
            public static ConfirmDialogFragment newInstance(String title, String message,boolean cancelab
le){
                    ConfirmDialogFragment instance = new ConfirmDialogFragment();
                    Bundle args = new Bundle();
                    args.putString("title",title);
                    args.putString("message",message);
                    args.putBoolean("cancelable",cancelable);
                    instance.setArguments(args);                    return instance;
```

```
            }            @NonNull
            @Override
            public Dialog onCreateDialog(Bundle savedInstanceState) {
                  创建ConfirmDialog核心代码，可以下载源代码查看.....
            }            @Override
            public void onAttach(Activity activity) {                  super.onAttach(activity);
        if (getActivity() instanceof ConfirmDialogListener ) {
                        mListener= (ConfirmDialogListener ) getActivity();
                  }
            }
            ......
    }
```

ConfirmDialogFragment很关键的一点，ConfirmDialogFragment中的mListener属性的值是通过

```
    @Override
            public void onAttach(Activity activity) {                  super.onAttach(activity);
        if (getActivity() instanceof ConfirmDialogListener ) {
                        mListener= (ConfirmDialogListener ) getActivity();
                  }
            }
```

方式获取的。

BaseActivity中代码片段：

```
public class BaseActivity extends FragmentActivity   {               /*
                *显示确认对话框方法
             */
            public void showConfirmDialog(...){
                        ......
             }               /*
                *显示进度条对话框方法
             */
            public void showProgressDialog(...){
                        ......
            }
}
```

那我们就重构BaseActivity的子类显示Dialog的代码：

我拿MainActivity来举例子：

MainActivity的关键重构代码：

```
        /*
      *实现确认对话框的ConfirmDialogListener 接口
       *created by niuxiaowei
    */
    public class MainActivity extends BaseActivity implements ConfirmDialogListener {


                @Override
               public void onClick(DialogInterface dialogInterface, int i) {                    Toast.ma
keText(this,"点击了MainActivity 调起的确认对话框 i="+i,Toast.LENGTH_LONG).show();
                }             //调用显示ConfirmDialog代码
             showConfirmDialog(...);            //调用显示ProgressDialog代码
             showProgressDialog(...);
 }
```

现在MainActivity里的代码运行起来完全没问题，因为MainActivity里面包含了3个Fragment，每个Fragment里面都有显示ConfirmDialog和ProgressDialog的代码，所以开始重构这3个Fragment：

重构思路：

- 每个Fragment里都可以获取到相对应的Activity的实例，只要获取到实例就可以调用显示对话框的方法来显示对话框了。
- 对话框中的事件怎么传递给Fragment问题？Activity可以获取到Fragment的实例，对话框可以把事件传递给Activity，因此Activity顺理成章的可以把事件传递给对应的Fragment。
- 一个Activity有多个Fragment调用显示对话框的方法，在Activity的实现了对话框接口的方法里怎样区分不同的Fragment调用者？可以在BaseActivity显示对话框的方法里加个id参数，用id来区分不同的Fragment调用者。

那就上关键代码片段：

MainActivity中的AFragment代码片段：

```
public AFragment extends Fragment implements ConfirmDialogListener{          @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                  Toast.makeText(this,"点击了AFragment 调起的确认对话框 i="+i,Toast.LENGTH_LONG).show(
);
            }          //调用显示ConfirmDialog代码
          getActivity().showConfirmDialog(...);
}
```

同理MainActivity的BFragment，CFragment的重构与AFragment类似。

MainActivity的关键代码片段：

```
@Override
public void onClick(DialogInterface dialogInterface, int i) {                    //伪代码
               if(mId == aFragment传递Id) {
                           aFragment.onClick(dialogInterface,i);
               }else if(mId == bFragment传递Id) {
               }else if(mId == cFragment传递Id) {
               }else if(mId == mainActiviy传递Id){
                         调用自己的方法
               }
}
```

产生的问题

看了MainActivity的onClick方法里面代码我都对自己无语了，onClick方法里面充斥着各种的if else 语句，并且当前的MainActivity里，若再有别的显示ConfirmDialog的调用者，onClick方法里少不了要增加对应的else if语句。MainActivity只是项目中所有Activity的一个缩影。其他的Activity也会遇到同样的问题（这不是我意淫的，提早预估到问题，提早入手进行解决总是好的）

我们拿MainActivity来代表所有的Activity总结下使用DialogFragment创建Dialog产生的问题：

- MainActivity里的onClick方法维护、扩展性不好，充斥着各种if else if语句，可读性也不

好。

- MainActivity里的onClick方法除了把Dialog的事件转发给相对应的调用者之外，没有多任何其他操作，所以是多余的
- 显示Dialog的方法不灵活

存在这些问题严重影响了我后面的重构工作，于是乎我就去国内国外网站上搜索对应的解决方法，但是也没有找到好的方法，最后我就想办法自己解决上面的问题，这也是我为何要封装DialogFragment的缘由。

封装DialogFragment，让DialogFragment使用非常简单、灵活

我们仔细的分析下上文的问题的主要原因是显示Dialog的方法没有把Dialog里面的开放的接口作为参数导致的，假如能像下面的使用方式：

```
 //某一个Activity中显示ConfirmDialog
showConfirmDialog(title,message,confirmDialogListener);      //某一个Fragment中显示ConfirmDialog
getActivity().showConfirmDialog(this,message,confirmDialogListener);      //非Activity和非Fragment的类中显示ConfirmDialog
mActivity.showConfirmDialog(this,message,confirmDialogListener);
```

上文中所有的问题都可以解决。

为什么不按下面的做法做

做法1:那我们直接把ConfirmDialogListener 的实例赋值给ConfirmDialogFragment 的实例的mListener属性，以下为代码：

```
//直接把listener传递
public static ConfirmDialogFragment newInstance(..., ConfirmDialogListener listener){
        ConfirmDialogFragment instance = new ConfirmDialogFragment();
        ......
        instance.mListener = listener;
        ......
  }
```

那我就详细的解释下为什么不这样做的具体原因：

- 在创建Fragment的时候，最好是把传递给Fragment的数据存放在Bundle中，然后在调用fragment的setArguments（bundle）方法进行数据的设置，这种做法好处是：系统会

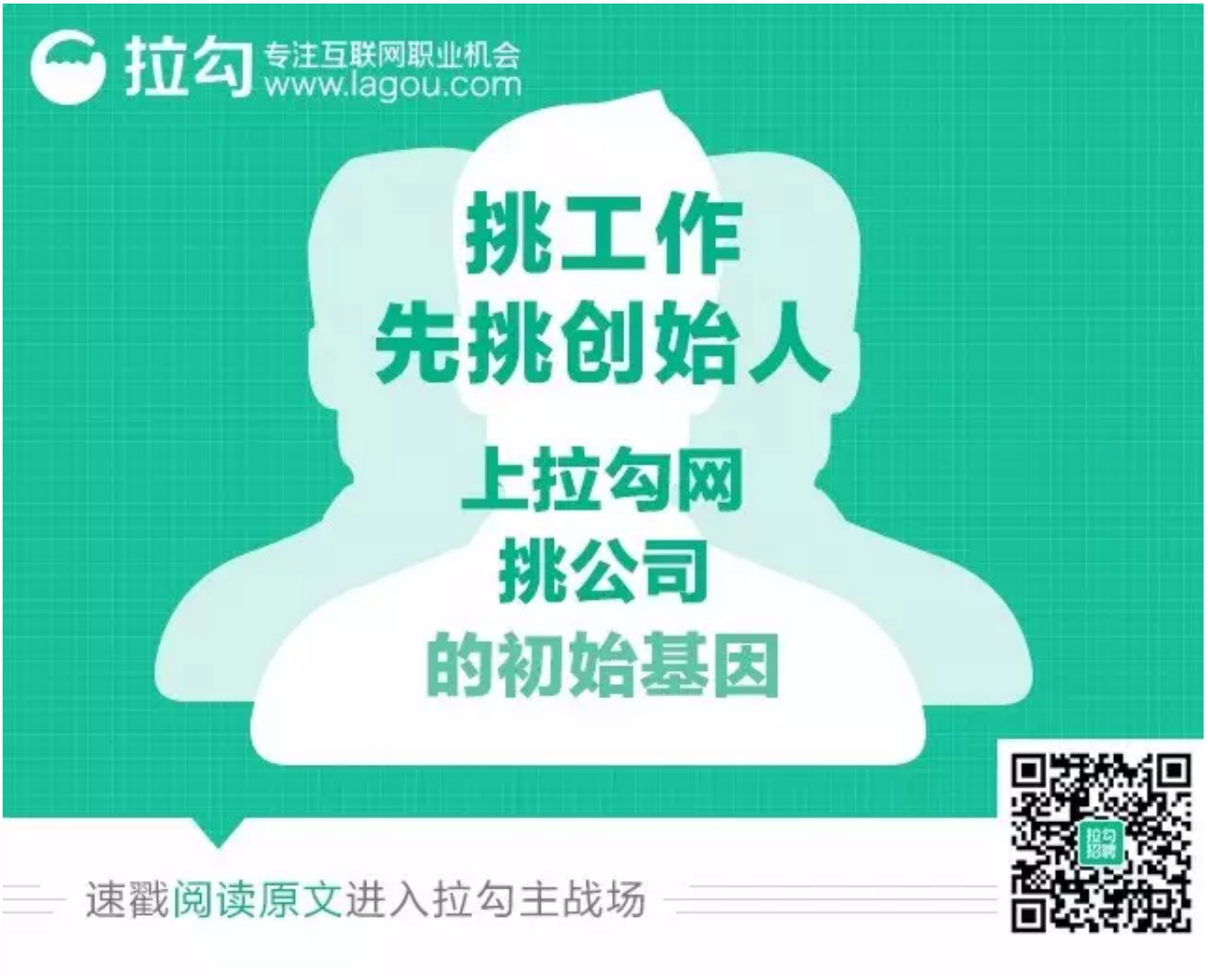
保存Fragment的数据，在手机配置发生变化后（比如旋屏），系统会把保存的Fragment数据进行恢复。

以上做法，ConfirmDialogFragment 中mListener属性系统没有为之保存，所以手机配置发生变化后，ConfirmDialogFragment 中的mListener 是null。

做法2：那我们是否可以把ConfirmDialogListener实例（ConfirmDialogListener是ConfirmDialogFragment 对外开放的接口）存放在Bundle中？

答案是不可以，首先 Bundle对存放的数据是有限制的，把ConfirmDialogListener的实例存入Bundle中是比较复杂的操作。其次即使通过艰辛万苦把ConfirmDialogListener实例存入了Bundle中，保存ConfirmDialogListener实例是毫无意义的，只有保存数据对于Fragment来说才有意义，保存行为对Fragment是无意义的。

字数超限制了，请看下篇啦~



阅读原文

微信扫一扫
关注该公众号