

# 移动生活，幸福代码

移动天下，IOS，Android

个人资料



Birds



访问：135356次

积分：2045

等级：[BLOG > 5](#)

排名：第11657名

原创：64篇

[高并发程序设计入门](#) [【活动】云计算行业圆桌论坛](#) [【知识库】一张大图看懂Android架构](#) [【征文】Hadoop](#)

[Hadoop——我与Hadoop不得不说的故事](#)

## Android App框架设计之编写基类BaseActivity

标签：[Android](#) [app架构](#) [设计模式](#)

2015-05-20 18:16 5946人阅读 [评论\(3\)](#) [收藏](#) [举报](#)

分类：[Android 技术研究 \(39\)](#)

版权声明：本文为博主原创文章，未经博主允许不得转载。

### Android App框架设计之编写基类BaseActivity

编写基类BaseActivity

- OOP里面子类里面能够共享父类的方法，提高代码的可复用性
- 基类的编写要根据实际项目的情况，有的可能很简单 比如只是显示“关于”信息，业务功能非常少 你可以不用继承基类进行扩展

BaseActivity继承Activity 还是 FragmentActivity 还是其它? 也是根据你的技术要求，因为解决问题的方法有很多种，或者你根本不用写基类也能完成工作。我们的目的是编写较好的代码，提高代码可复用性，可扩展性，尤其是团队协作非常重要，这就是OOP为什么出现的原因之一。

转载： 5篇

译文： 1篇

评论： 38条

文章搜索

文章分类

- Android 技术研究 (40)
- 设计模式 (1)
- JAVAE E (1)
- 数据库 (1)
- java基础学习 (5)
- java学习 (7)
- 个人问题 (0)
- 手机移动开发 (19)
- 算法 (0)
- 脚本 (2)
- IOS学习与实践 (4)

文章存档

- 2015年12月 (1)
- 2015年09月 (2)
- 2015年08月 (1)
- 2015年06月 (2)
- 2015年05月 (4)

展开

阅读排行

考虑基类都应写什么？通用的代码。

- 网络环境监控
- 统一的数据接口
- 生命周期不同方法应该做的事情

刚工作的时候，看Spring源码 根本就不知道为什么搞那么多接口和对象，一层层的。 随着经验的积累才知道OOP的好处。

下面我们编写一个基础的基类，并不复杂。 假设我们的业务是公司的移动OA系统 我们需要管理员工。

抽象一个员工

```
1 //Domine为通用实体类，后面可作为泛型参数有用
2 public class Employee extends Domine {
3
4     private static final long serialVersionUID = 1L;
5
6     public Employee() {
7     }
8
9     public String fullName;
10    public String mobileTel;
11    public String email;
12    public String empNo;
13 }
14
15 //可以添加公共属性
16 public class Domine implements Serializable {
17
18     private static final long serialVersionUID = 1L;
19
20     public Domine() {
21     }
22
23     public int id;
24     public String desc;
```

Android 图片切换，Imag	
Android 5.0 CardView 应	(11184)
JavaEE5 Jax-ws 学习	(7989)
Android App框架设计之	(6060)
Android使用Gradle进行	(5902)
NDK 4 使用Socket与服务	(5346)
纯代码实现 Android 登陆	(5057)
apache mina 长连接	(4913)
JAVA JDBC 连接Mysql 1	(4612)
Android中使用Apache cc	(4504)
	(4324)

评论排行	
Android使用Messenger	(10)
AsyncTask测试多任务	(5)
Android App框架设计之	(3)
IOS AFNetworking基本学	(3)
蓝牙技术	(2)
Android 图片切换，Imag	(2)
java thread synchronizec	(2)
从简单的android 登陆应	(1)
监听网络，启动Service	(1)
Android中使用Apache cc	(1)

推荐文章	
*Android自定义ViewGroup打造各种风格的SlidingMenu	
* Android 6.0 运行时权限处理完全解析	

25	
26	}
...	

数据交互的模型

1	
2	public class MData<T> implements Serializable {
3	
4	private static final long serialVersionUID = 1L;
5	
6	public String id;
7	public String type;
8	public T dataList;//多种类型数据，一般是List集合，比如获取所有员工列表
9	
10	}

下面是数据回调接口，我们从网络端获取数据 需要通知给UI

1	
2	public interface IDataCallback<T> { //我们一样传入通用类型
3	public void onNewData(T data);
4	public void onError(String msg,int code);
5	}

下面是独立出来一个UIHandler，一般是写在Activity里面作为内部类，我这里没有

1	public class UIHandler extends Handler {
2	
3	private IHandler handler;//回调接口，消息传递给注册者
4	
5	public UIHandler(Looper looper) {
6	super(looper);
7	}
8	
9	public UIHandler(Looper looper,IHandler handler) {

* 数据库性能优化之SQL语句优化
*Animation动画详解(七)——ObjectAnimator基本使用
* Chromium网页URL加载过程分析
* 大数据三种典型云服务模式

最新评论
Android App框架设计之编写基类 孤心泣: 写的很好，赞一个
开源Android-PullToRefresh下拉 a153614131: 楼主写的好清晰，豁然开朗。已关注~
AsyncTask测试多任务 LinterChan: AsyncTask有两种执行方式，一种是execute串行执行，一种是executeOnExecut...
Android App框架设计之编写基类 hymbo: 例子虽小，却充分看到了基类的好处
Android App框架设计之编写基类 Wilshion: 谢谢博主，对我这种新手而言写的很好。。
Android自定义View绘制闹钟 Kern_: 谢谢哥们了
Android DrawerLayout 抽屉 momoney: 博主现在哪里高就？对你很感兴趣
Android Toast 简单封装 ItJavawfc: 楼主，封装借用。。
IOS AFNetworking基本学习 haohao123nana: 我找到原因了，是因为我的workspace是在虚拟机外的硬盘里面，通过虚拟机共享设置的。我把代码弄到...
Android使用Messenger进行Serv java路上: 请问，为什么Messenger只支持单线程，我可以有多个客户端和服务端通信吗？

10	super(looper);
11	this.handler = handler;
12	}
13	
14	public void setHandler(IHandler handler) {
15	this.handler = handler;
16	}
17	
18	@Override
19	public void handleMessage(Message msg) {
20	super.handleMessage(msg);
21	if (handler != null) {
22	handler.handleMessage(msg); //有消息，就传递
23	}
24	}
25	}
26	
27	public interface IHandler {
28	public void handleMessage(Message msg);
29	}
30	

我们有了基本的东西，现在组合到BaseActivity中

1	public abstract class BaseActivity extends Activity {
2	
3	// 可以把常量单独放到一个Class中
4	public static final String ACTION_NETWORK_CHANGE = "android.net.conn.CONNECTIVITY_CHANGE";
5	public static final String ACTION_PUSH_DATA = "fm.data.push.action";
6	public static final String ACTION_NEW_VERSION = "apk.update.action";
7	
8	protected static UIHandler handler = new UIHandler(Looper.getMainLooper());
9	
10	// 数据回调接口，都传递Domine的子类实体
11	protected IDataCallback<MData<? extends Domine>> dataCallback;
12	

```
13 public BaseActivity() {
14
15 }
16
17 //这个地方有点“模板方法”的设计模式样子
18 @Override
19 protected void onCreate(Bundle savedInstanceState) {
20     super.onCreate(savedInstanceState);
21     setBase();
22     setHandler();
23     setContentView(savedInstanceState);
24 }
25
26 private void setHandler() {
27     handler.setHandler(new IHandler() {
28         public void handleMessage(Message msg) {
29             handler(msg); //有消息就提交给子类实现的方法
30         }
31     });
32 }
33
34 // 初始化UI , setContentView等
35 protected abstract void setContentView(Bundle savedInstanceState);
36
37 // 可能全屏或者没有ActionBar等
38 private void setBase() {
39     requestWindowFeature(Window.FEATURE_NO_TITLE); // 例
40 }
41
42 protected void addLeftMenu(boolean enable) {
43     // 如果你的项目有侧滑栏可以处理此方法
44     if (enable) { // 是否能有侧滑栏
45
46     } else {
47
48     }
```

```
49     }
50     //让子类处理消息
51     protected abstract void handler(Message msg);
52
53
54     // 横竖屏切换，键盘等
55     @Override
56     public void onConfigurationChanged(Configuration newConfig) {
57         super.onConfigurationChanged(newConfig);
58     }
59
60     @Override
61     protected void onRestoreInstanceState(Bundle savedInstanceState) {
62         try {
63             super.onRestoreInstanceState(savedInstanceState);
64         } catch (Exception e) {
65         }
66     }
67
68     @Override
69     protected void onResume() {
70         super.onResume();
71         // 你可以添加多个Action捕获
72         IntentFilter filter = new IntentFilter();
73         filter.addAction(ACTION_NETWORK_CHANGE);
74         filter.addAction(ACTION_PUSH_DATA);
75         filter.addAction(ACTION_NEW_VERSION);
76         registerReceiver(receiver, filter);
77         //还可能发送统计数据，比如第三方的SDK 做统计需求
78     }
79
80     @Override
81     protected void onPause() {
82         super.onPause();
83         unregisterReceiver(receiver);
84         //还可能发送统计数据，比如第三方的SDK 做统计需求
```

```

85     }
86
87     BroadcastReceiver receiver = new BroadcastReceiver() {
88         @Override
89         public void onReceive(Context context, Intent intent) {
90             // 处理各种情况
91             String action = intent.getAction();
92             if (ACTION_NETWORK_CHANGE.equals(action)) { // 网络发生变化
93                 // 处理网络问题
94             } else if (ACTION_PUSH_DATA.equals(action)) { // 可能有新数据
95                 Bundle b = intent.getExtras();
96                 MData<Employee> mdata = (MData<Employee>) b.get("data");
97                 if (dataCallback != null) { // 数据通知
98                     dataCallback.onNewData(mdata);
99                 }
100             } else if (ACTION_NEW_VERSION.equals(action)) { // 可能发现新版本
101                 // VersionDialog 可能是版本提示是否需要下载的对话框
102             }
103         }
104     };
105
106     public void setDataCallback(
107         IDataCallback<MData<? extends Domine>> dataCallback) {
108         this.dataCallback = dataCallback;
109     }
110 }

```

有了基类现在编写一个子类Activity

```

1
2 public class EmployeeDisplayActivity extends BaseActivity implements IDataCallback<MData<? extends Domine>
3
4     public EmployeeDisplayActivity() {
5     }
6
7     @Override

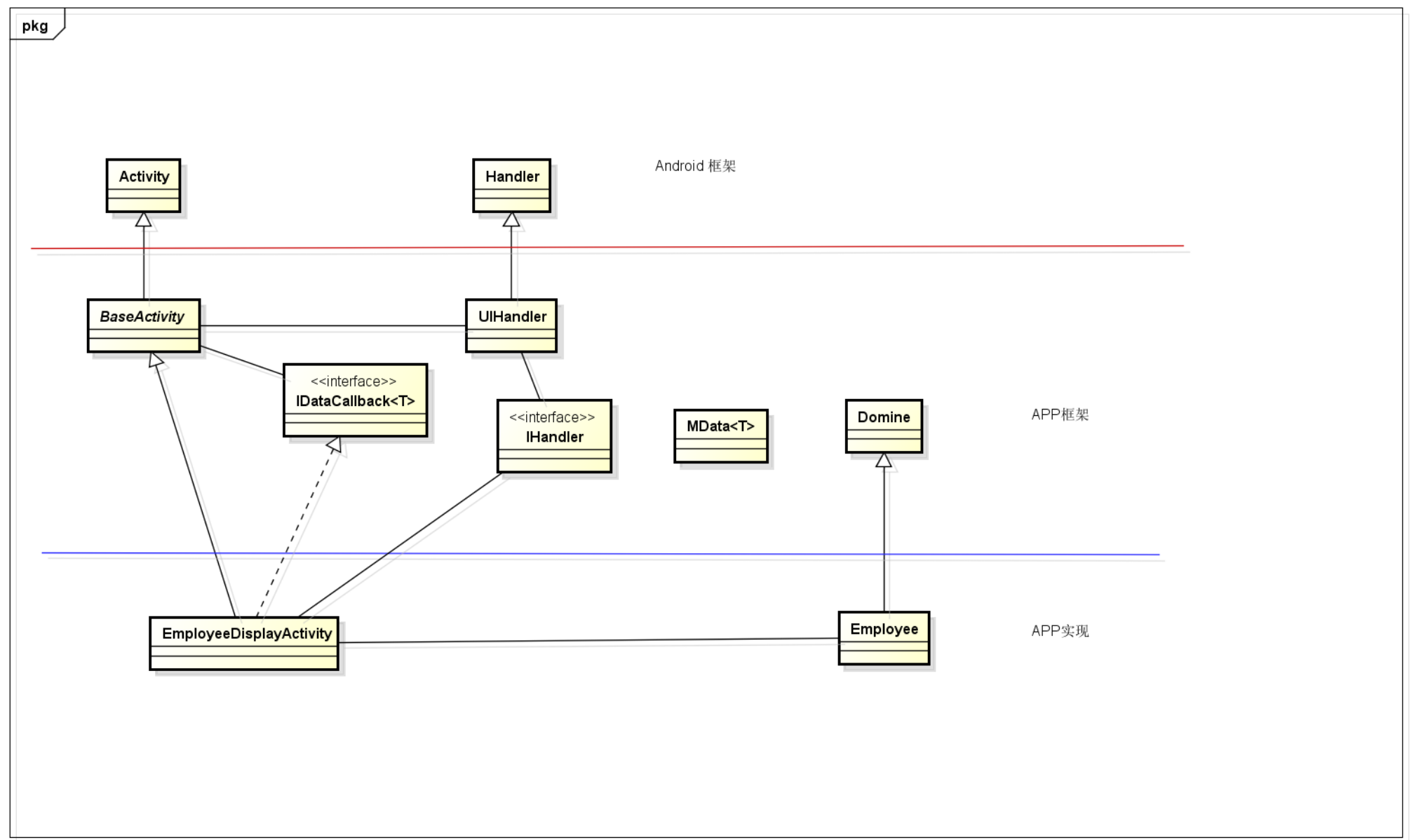
```

```
8     protected void initView(Bundle savedInstanceState) {
9         setContentView(R.layout.activity_main);
10        loadData();
11    }
12
13    private void loadData() {
14        setDataCallback(this); //设置回调函数
15        //我们还可以把这个Callback传给其它获取数据的类，比如HTTP获取数据
16        //比如 HttpClient.get(url, this);
17    }
18
19    @Override
20    public void onNewData(MData<? extends Domine> data) {
21        //update UI 更新UI数据
22        final List<Employee> list = (List<Employee>) data.dataList;
23        handler.post(new Runnable() {
24            public void run() {
25                //更新UI
26            }
27        });
28        //或者
29        handler.sendMessage(0); //通知Handler
30    }
31
32    @Override
33    public void onError(final String msg, final int code) {
34        handler.post(new Runnable() {
35            public void run() {
36                //通知错误消息
37            }
38        });
39        //或者
40        handler.sendMessage(0); //通知Handler
41    }
42
43    @Override
```



```
44     protected void handler(Message msg) { //我们可以处理数据消息了
45         switch (msg.what) {
46             case 0:
47                 break;
48             case -1:
49                 break;
50             default:
51                 break;
52         }
53     }
54
55 }
```

UML关系图



本文编写了基本的BaseActivity代码，显然不可能是完美的，需要不断学习和改进，有问题欢迎探讨，Thanks.

上一篇 Android roboguice 开源框架使用

下一篇 Android LoaderCallbacks

我的同类文章

Android 技术研究 (39)

• Android开源项目 Universal imageleade...

2015-12-30

阅读 62

• ViewDragHelper实现相册图片查看

2015-09-11

阅读 209

• Android LoaderCallbacks

2015-06-01

阅读 431

• 使用EventBus进行Fragment和Activity...

2015-05-11

阅读 2094

• Google官方提供的有用的Android范例...

2015-04-24

阅读 3232

• Android自定义View绘制闹钟

2015-08-29

阅读 364

• Android DrawerLayout 抽屉

2015-06-03

阅读 535

• 序列化和反序列化技术

2015-05-12

阅读 211

• Android roboguice 开源框架使用

2015-05-13

阅读 806

更多文章

主题推荐

android   spring   oa   string   框架   设计   app   class

猜你在找

- Android设计模式精解(第7课)：Adapter模式
- Android底层技术：Linux驱动框架与开发
- Android设计模式精解(第10课)：状态模式(State pattern)
- Android设计模式精解(第9课)：策略模式(Strategy pattern)
- Struts2架构设计之路 自主编写Web开发框架

查看评论

3楼 [孤心泣](#) 5天前 09:05发表



写的很好，赞一个

2楼 [hymbo](#) 2015-11-11 01:05发表



例子虽小，却充分看到了基类的好处

1楼 [Wilshion](#) 2015-11-03 19:16发表



谢谢博主，对我这种新手而言写的很好。。

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

- 全部主题
- Hadoop    AWS    移动游戏    Java    Android    iOS    Swift    智能硬件    Docker    OpenStack    VPN    Spark    ERP    IE10
- Eclipse    CRM    JavaScript    数据库    Ubuntu    NFC    WAP    jQuery    BI    HTML5    Spring    Apache    .NET    API    HTML    SDK    IIS
- Fedora    XML    LBS    Unity    Splashtop    UML    components    Windows Mobile    Rails    QEMU    KDE    Cassandra    CloudStack    FTC
- coremail    OPhone    CouchBase    云计算    iOS6    Rackspace    Web App    SpringSide    Maemo    Compuware    大数据    aptech    Perl
- Tornado    Ruby    Hibernate    ThinkPHP    HBase    Pure    Solr    Angular    Cloud Foundry    Redis    Scala    Django    Bootstrap