Android退出应用最优雅的方式(改进版)

2016-01-07 安卓应用频道 安卓应用频道

(点击上方公众号，可快速关注)

来源：_Hi_xiaoyu

链接：http://blog.csdn.net/soul_code/article/details/50453934

本系列：

Android退出应用最优雅的方式(未改进版)

我们先来看看几种常见的退出方法（不优雅的方式）

一、容器式

建立一个全局容器，把所有的Activity存储起来，退出时循环遍历finish所有Activity

```
import java.util.ArrayList;
import java.util.List;
import android.app.Activity;
import android.os.Bundle;
public class BaseActivity extends Activity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
// 添加Activity到堆栈
AtyContainer.getInstance().addActivity(this);
}
@Override
protected void onDestroy() {
super.onDestroy();
// 结束Activity&从栈中移除该Activity
AtyContainer.getInstance().removeActivity(this);
}
}
class AtyContainer {
private AtyContainer() {
}
private static AtyContainer instance = new AtyContainer();
private static List activityStack = new ArrayList();
public static AtyContainer getInstance() {
return instance;
}
```

```
public void addActivity(Activity aty) {
activityStack.add(aty);
}
public void removeActivity(Activity aty) {
activityStack.remove(aty);
}
/**
* 结束所有Activity
*/
public void finishAllActivity() {
for (int i = 0, size = activityStack.size(); i if (null != activityStack.get(i)) {
activityStack.get(i).finish();
}
}
activityStack.clear();
}
}
```

这种方法比较简单， 但是可以看到activityStack持有这Activity的强引用，也就是说当某个Activity异常退出时，activityStack没有即使释放掉引用，就会导致内存问题，接下来我们看一种类似的方式，但是会稍微优雅一点点

二、广播式

通过在BaseActivity中注册一个广播，当退出时发送一个广播，finish退出

```
public class BaseActivity extends Activity {
private static final String EXITACTION = "action.exit";
private ExitReceiver exitReceiver = new ExitReceiver();
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
IntentFilter filter = new IntentFilter();
filter.addAction(EXITACTION);
registerReceiver(exitReceiver, filter);
}
@Override
protected void onDestroy() {
super.onDestroy();
unregisterReceiver(exitReceiver);
}
class ExitReceiver extends BroadcastReceiver {
@Override
public void onReceive(Context context, Intent intent) {
BaseActivity.this.finish();
}
}
}
```

三、进程式

通过直接杀死当前应用的进程来结束应用，简单粗暴，而且有（wu）效！

```
android.os.Process.killProcess(android.os.Process.myPid());
System.exit(0);
ActivityManager manager = (ActivityManager) getSystemService(ACTIVITY_SERVICE);
manager.killBackgroundProcesses(getPackageName());
```

这三种都能达到同样的效果，但是在模拟器上都会弹出 Unfortunately , XXX has stopped 消息提示框，但确实能退出应用。部分真机直接失效，只能finish当前Activity（比如我手上这台小米note，国产的几款ROM fw层改动太多，使用这种方式需慎重）

四、 RS优雅式

什么是RS式呢？即Receiver+singleTask 。我们知道Activity有四种加载模式，而singleTask就是其中的一种，使用这个模式之后，当startActivity时，它先会在当前栈中查询是否存在Activity的实例，如果存在，则将其至于栈顶，并将其之上的所有Activity移除栈。我们打开一个app，首先是一个splash页面，然后会finish掉splash页面。跳转到主页。然后会在主页进行N次的跳转，期间会产生数量不定的Activity，有的被销毁，有的驻留在栈中，但是栈底永远是我们的HomeActivity。这样就让问题变得简单很多了。我们只需两步操作即可优雅的实现app的退出。

1、在HomeActivity注册一个退出广播，和第二个广播式一样，但是这里只需要在HomeActivity一个页面注册即可。

2、设置HomeActivity的启动模式为singleTask。

当我们需要退出的时候只需要startActivity(this,HomeActivity,class)， 再发送一个退出广播。上面代码首先会把栈中HomeActivity之上的所有Activity移除出栈，然后接到广播finish自己。一切OK！没有弹框，不用考虑机型Rom适配。不会有内存问题，就是那么的优雅，简单！

五、SingleTask改版式

和一些小伙交流之后，很多小伙伴说注册广播略显麻烦，在楼下的小伙伴提出了一种更简单的方式，思路也很简单，

1、设置MainActivity的加载模式为singleTask

2、重写MainActivity中的onNewIntent方法

3、需要退出时在Intent中添加退出的tag

由于很多小伙伴对源码需求比较热切，我们这里就直接以代码的形式为大家讲解这种方式

第一步设置MainActivity的加载模式为singleTask

```
android:launchMode="singleTask"
```

第二步重写onNewIntent()方法

```
private static final String TAG_EXIT = "exit";
@Override
protected void onNewIntent(Intent intent) {
super.onNewIntent(intent);
if (intent != null) {
boolean isExit = intent.getBooleanExtra(TAG_EXIT, false);
if (isExit) {
this.finish();
}
}
}
```

第三步 退出

```
Intent intent = new Intent(this,MainActivity.class);
intent.putExtra(MainActivity.TAG_EXIT, true);
startActivity(intent);
```

六、懒人式

这种方式更加简单，只需要如下两步操作

1、将MainActivity设置为singleTask
2、将退出出口放置在MainActivity

我们可以看到很多应用都是双击两次home键退出应用，就是基于这样的方式来实现的，这里在贴一下如何处理连续两次点击退出的源码

```
private boolean mIsExit;
@Override
/**
* 双击返回键退出
*/
public boolean onKeyDown(int keyCode, KeyEvent event) {
if (keyCode == KeyEvent.KEYCODE_BACK) {
if (mIsExit) {
this.finish();
} else {
Toast.makeText(this, "再按一次退出", Toast.LENGTH_SHORT).show();
mIsExit = true;
```

```
new Handler().postDelayed(new Runnable() {
@Override
public void run() {
mIsExit = false;
}
}, 2000);
}
return true;
}
return super.onKeyDown(keyCode, event);
}
```


安卓应用频道

专注分享安卓应用相关内容

微信号：AndroidPD

长按,识别二维码关注

商务合作QQ：2302462408


拉勾 专注互联网职业机会
www.lagou.com
挑工作
不妨看看面试评价
上拉勾网
听面试过的人怎么说
速戳阅读原文进入拉勾主战场

阅读原文 举报