一．布局优化

1.<include>标签可以将一个指定的布局文件加载到当前的布局文件中

2.<merge>标签一般和<include>标签一起使用从而减少布局的层级

二．绘制优化

1.onDraw中不要创建新的局部对象。因为ondraw方法可能会被频繁调用，这样就会创建大量的局部对象。

2.ondraw方法中不要做耗时操作，会造成view的绘制过程不流畅。

三．内存优化

1.减小对象的内存占用

1）使用更加轻量的数据结构（ArrayMap替换HashMap）

2）避免在android中使用枚举

3）减小bitmap对象的内存占用（使用前先压缩图片）

4）使用更小的图片

2.内存对象重复利用

1）复用系统自带的资源（字符串/颜色/图片/动画/布局）

2）ListView/GradView的ConvertView复用、使用单例减少对象的重复创建

If(conVertView==null){

Holder=new ViewHolder();

}else{

Holder=(ViewHolder)convertView.getTag();

}

3）StringBuilder:如果使用Stgring进行字符串多次拼接，会产生多个对象，而StringBuilder是对一个对象多次操作

3.避免对象的内存泄漏

1. 单例模式导致的内存泄漏
2. 静态常量导致的内存泄漏
3. 非静态内部类创建静态常量实例导致的内存溢出
4. Handler导致的内存溢出（创建非静态匿名内部handler,handler长期持有activity的引用，handler又不断的进行消息的轮询，导致activity不能销毁）
5. 线程造成的内存泄漏（在activity中创建的匿名内部线程，在线程还没有执行完毕时，关闭activity,匿名内部类又持有着activity的引用）
6. webview内存泄漏
7. 资源未关闭导致内存泄漏