Android性能优化

武汉数字政通 张硕

Java代码

- **字符串**

  +连接 → StringBuilder

- **哈希**

  HashMap → SparseArray：前者key为Integer对象，后者key为int

- **缓存**

  Lrucache<K,V>：LRU丢弃最近最少使用，缓存下载数据

响应

- **原则**：不仅关注性能，也要让用户真正感觉到快
- **目的**：保持应用的持续响应，让主线程尽可能快地完成任务
- **做法**：

  主线程 → 后台线程

  推迟初始化

响应

- **后台线程**

  在AsyncTask中进行网络操作或者访问文件操作。

  若症结在算法，就必须优化算法，不然只是欲盖弥彰。

- **推迟初始化**

  ViewStub：因为内存分配需要耗时，直到对象真正需要时（即运行期需要展现时）才创建，用相应资源替换，自己就可以被垃圾回收。

布局

- **做法：**

  降低复杂性，保持扁平化：嵌套的LinearLayout → RelativeLayout

  重用布局：<include>

  合并FrameLayout：<merge>

- **工具：**
  - hierarchyviewer和layoutopt

SQLite

- **原则：**降低IO
  不要重度使用SQLite
  只读取需要的数据

内存

- **目标：**高效使用内存
- **原因：**

  物理内存小

  虚拟内存交换能力差
- **做法：**

  处理大量数据时，使用满足需要的最小数据类型

  int → short，内存消耗小，排序速度快

  double → float，精度要求不苛刻时

  尽量避免类型转换，保持类型一致

内存

- **内存泄露检测：**StrictMode
  Activity与其他对象泄漏
  Closeable类没关闭造成的泄漏

评测

- 时间测量

  System.currentTimeMillis() → System.nanoTime()：精度准确度高
- 方法调用跟踪

  Debug.startMethodTracing()
- 分析跟踪文件

  Traceview工具
- 日志

  LogCat

电量

- **原则**：节省电量，代码不做无用功
- **做法**：

  广播接收：在onResume启用，在onPause禁用
  网络传输：先压缩数据，再传输
  位置监听：不需时注销，设置合适更新频率
  传感器：不需时禁用，降低通知频率
  WakeLock：视频暂停时释放，继续播放时再次获取

NDK

- **Java调用C++**
  1. 用Java声明native方法
  2. 利用JDK建立JNI粘合层
  3. 创建Android makefile文件
  4. 用C++实现本地方法
  5. 编译本地库
  6. 加载本地库
- **完全使用C++**

  NativeActivity，但是不必要所有activity都使用

谢谢

2014-09-03