Author: Kris Lee

Email: kris1987@qq.com

Web: www.krislq.com

Contents

应用缓存机制

eoe wiki 客户端

故事

http://www.eoeandroid.com/thread-194188-1-1.html

【应用简介】
移动开发百科(eoeWIKI)是一款移动开发知识百科软件，依靠 eoe 开发者社区，将高质量移动开发内容用百科的形势进行组织和展示，以方便大家能更快更好的学习移动开发的相关知识，尽快的成长。

【应用故事】
eoe 的理念是'让移动开发更简单', 如何让移动开发更简单是我们不断思考和探索的，我们认为有组织、有体系的内容对开发者的学习和成长是最有价值的，于是大家都希望有这么一款应用让我们零碎的时间里学习到最有价值的知识，于是移动开发百科(eoeWIKI)诞生了。

数据库+文件

- 模式
  - 数据库+文件
- 适用情况
  - 对速度要求不高，数据量大

如果只是一些不多的文本，可以只存放在数据库中，而不必再存入文件。

《三种东西永远不要放到数据库里》

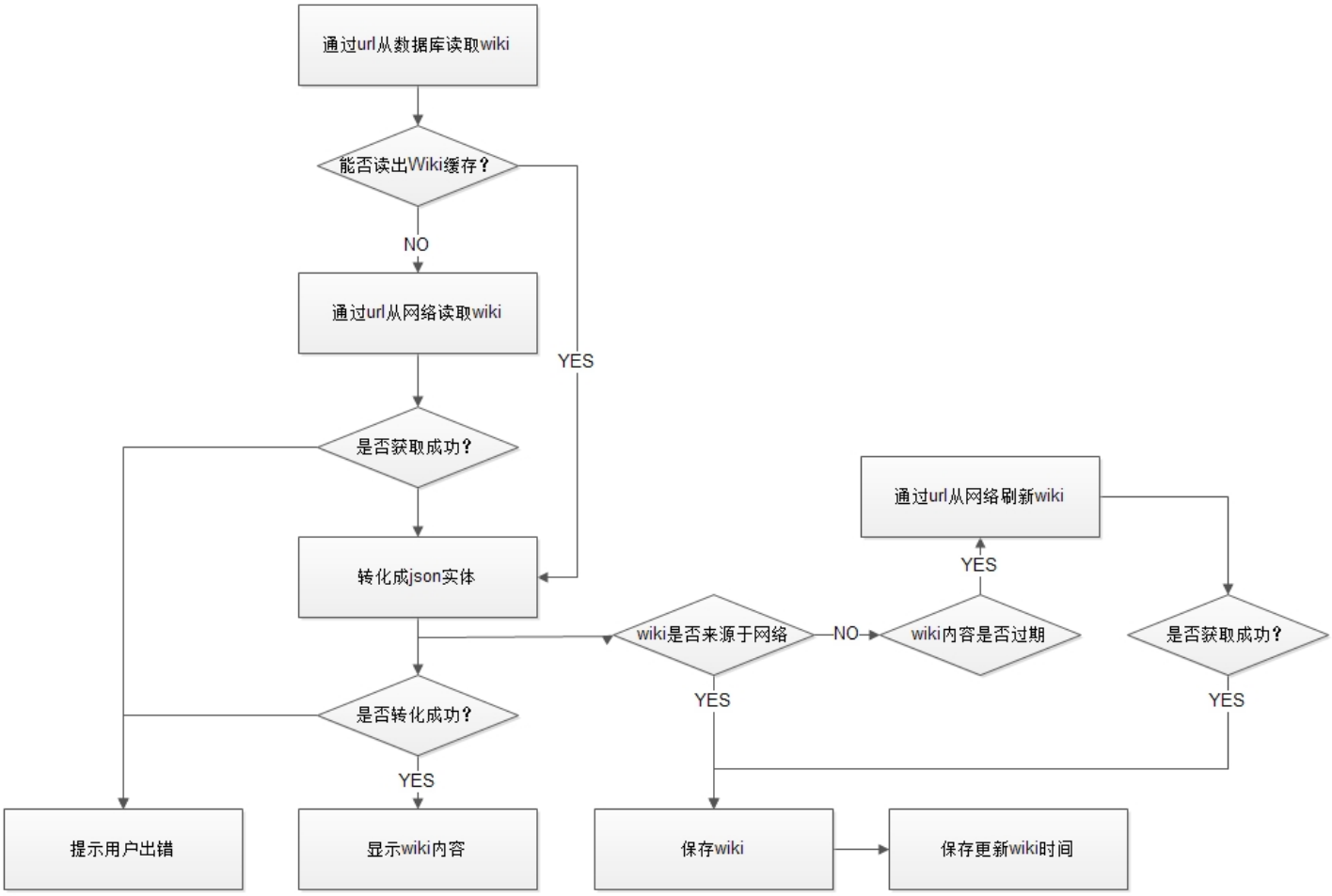
http://www.eoeandroid.com/thread-173766-1-1.html

图片，文件，二进制数据

短生命期数据

日志文件

流程图



线程池

为什么需要线程池？

T1= 创建线程时间，T2 =线程执行时间，T3 =销毁线程时间。

如果 T1 + T3 > T2，并且需要许多线程时，可以采用线程池，以提高服务器性能。

比如：每次请求图片我们需要一个线程去操作，一个 listView 里面有许多图片（假设 5000 个），我们利用线程池则只需要远远小于 5000 的线程去重复执行即可。

类	特征
newCachedThreadPool	1. 缓存型池子通常用于执行一些生存期很短的异步型任务 2. 能 reuse 的线程，必须是 timeout IDLE 内的池中线程 3. 缺省 timeout 是 60s,超过这个 IDLE 时长，线程实例将被终止及移出池。
newFixedThreadPool	1．最多只能有固定数目的活动线程存在。此时如果有新的线程要建立，只能放在另外的队列中等待，直到当前的线程中某个线程终止直接被移出池子 2.FixedThreadPool 多数针对一些很稳定很固定的正规并发线程，多用于服务器
ScheduledThreadPool	调度型线程池，池子里的线程可以按 schedule 依次 delay 执行，或周期执行
SingleThreadExecutor	单例线程，任意时间池中只能有一个线程

图片错位

原因：

共用 convertView。

如果我们永远都只生成一个 convertView，而我们的图片有多个，那么异步下载完成生设置图片的 view 都是指向一个实际的对象的。

解决办法：

1. 如果不需要异步下载图片（比如缓存中有），可以共用。否则就生成一个 convertView。
2. 允许生成一定数量的 convertView，但是需要与异步下载的线程同步。否则滑动太快的情况下，也会发生错位。比如说我已经显示到第 50 个了，可能异步下载线程刚下载完前 10 个的图片。
3. 如果对速度要求特别高，则实行预加载。即在显示前先把 convertView 创建出来。

思想：

通向罗马的路有千万条，找到最适合自己的一条。

所以这也要求我们在项目的前期，需求阶段，需要尽量的详细，才能在组建项目的时候找到最合适的。如果开始实施项目了，还三天一小变，五天一大变，再牛 b 的程序猿也累跨。

OOM(内存溢出)

原因：

1. 内存中加载数据过于庞大
2. 集合中有对对象的引用，没及时清空，jvm 不能回收（内存泄漏）
3. 代码中有死循环或者循环中产生过多的重复对象
4. Jvm 参数太小了

网友推荐 Application 对象的使用-数据传递以及内存泄漏...

http://www.eoeandroid.com/thread-187029-1-1.html

解决办法：

1. 修改应用运行参数 。android:largeHeap="true"
2. 载入大数据时，能否伪载入。比如解析图片时：inJustDecodeBounds= true 获取图片的信息。
3. 防止内存的泄漏，可以找有经验的程序员来 review
4. 内存查看工具

Listview 性能优化

1. 尽量创建少的 item。 共用 convertView（是在不需要异步加载的情况下）。如果在异步下面共用，则会引起内容的错位。
2. 尽量减小在 getView 中获取数据的时间，可以先缓存起来（如果结构比较复杂）。在 getView 只需要通过 get 显示及可
3. 尽量在 item 中显示小数据，速度与大小成正比。再怎么优化，数据大了，也会耗时的。
4. 使用 Holder，可以减小每次查找 view 的时间。特别是一些界面上控件较多的时候，非常用用的
5. 分页加载，10，20.。。这样可以减少等待的时间，也可以让每次分析的数据较小。因为如果一次载入内存的东西太多了，OOM 就越容易发生了
6. 预加载，在显示前先把数据或者甚至是 convertView 准备好。然后再显示
7. android:largeHeap="true“(3.0+)　可以通知系统说我们应用需要申请更多的内存。但并不是所有的手机上面都需要申请大内存。我们可以通过 android 资源包的机制来使应用在大屏幕的手机上面才开户这个设置
8. 直接用 LinearLayout(数据不多，内存不多，动态变化的控件少).因为我们都知道每次刷新一次 listview ,都会导致多次调用 getView .而用 LinearLayout 就可以一次生成，永远使用。在屏幕滚动的时候还不会再去销毁对象再生成。速度会更快。