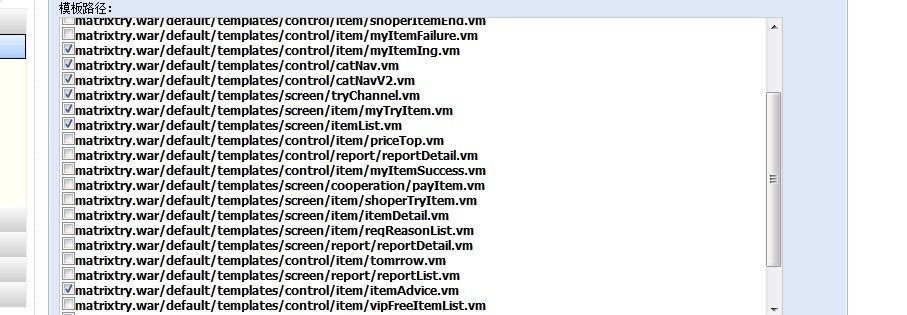
本地缓存的设计

涉及的问题

* 替换策略: LRU、MRU、FIFO



缓存命中率的提升：

测试场景：

页面渲染：20ms

取数据 ：200ms

并发： 10 100 1000

服务器线程:

误区：控制线程穿透次数意义不大，而且增加了复杂度。

控制失效时请求穿透数：

n大于零

定义： -1 永不穿透

1. 全部穿透，默认0

n 有且仅穿透n个

一致性Hash：问题节点当机引发雪崩效应。

延迟清除策略。

其它实现：

LRUMap Apache Commons Collections not thread-safe。 Must using Collections.synchronizedMap(Map) in multiple thread

分布式缓存：

JCS、ehcache

本地缓存策略： <http://blog.csdn.net/cenwenchu79/article/details/6076513>

一致性Hash算法及

翻译

清理操作仅仅会阻塞当前线程，而不会

put 清理 vs 线程清理（清理操作很快，并且put本身不是很频繁）， vs 部分清理策略。

内存操作，清理非常快速。

缓存不存在时的清理策略。

put 时更新key 检查key中的数据是否和object一致。

缓存的失效，

缓存不存在情况处理，大量的穿透，对数据进行预热的处理。

强弱引用规则

统计信息用

自动变换强弱引用

相关资源：

<http://www.codeinstructions.com/2008/09/weakhashmap-is-not-cache-understanding.html>

<http://blog.csdn.net/cenwenchu79/article/details/6076513>

那个cache策略的文档：xxx

参考文档：

<http://code.google.com/p/concurrentlinkedhashmap/>

<http://people.csail.mit.edu/karger/Talks/Hash/index.htm>

Cache类型： <http://www.cacheonix.com/articles/Caching_for_Java_Applications.htm>