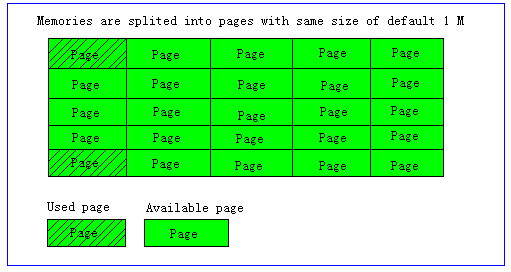
# 一、memcached究竟是如何运作的

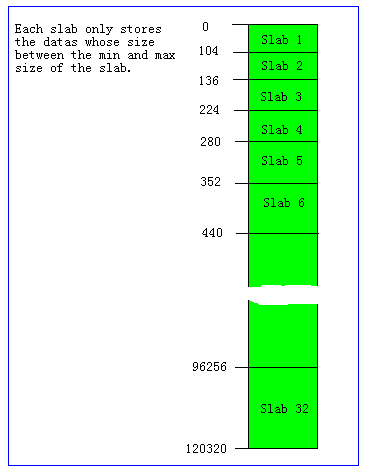
## 1.1、Page为内存分配的单位

Memcached的内存分配以page为单位，默认情况下一个page是1M，可以通过-I参数修改，最小1K，最大128M。如果 需要申请内存时，memcached会划分出一个新的page并分配给需要的slab区域。page一旦被分配在memcached重启前不会被回收或者 重新分配（page ressign已经从1.2.8版移除了）。

[](http://xenojoshua.com/wp-content/uploads/2011/04/memcached-01.png)

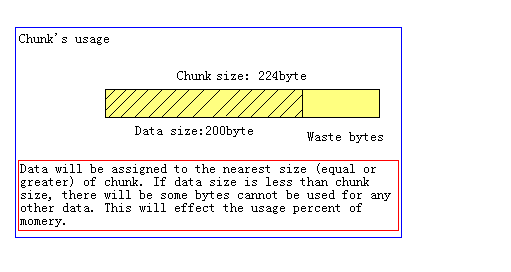
## 1.2、Slabs划分数据空间

Memcached并不是将所有大小的数据都放在一起的，而是预先将数据空间划分为一系列大小的slabs，每个slab只负责一定大 小范围内的数据存储。每个slab只存储大于其上一个slab的size并小于或者等于自己最大size的数据。例如：slab 3只存储大小介于137 到 224 bytes的数据。如果一个数据大小为230byte的数据进行存储，它将被分配到slab 4中。每个slab负责的空间其实是不等的，memcached默认情况下下一个slab的最大值为前一个的1.25倍，这个可以通过修改-f参数来修改 增长比例。

[](http://xenojoshua.com/wp-content/uploads/2011/04/memcached-02.png)

## 1.3、Chunk才是存放缓存数据的单

Chunk是一系列固定的内存空间，这个大小就是管理它的slab的最大存放大小。例如：slab 1的所有chunk都是104byte，而slab 4的所有chunk都是280byte。chunk是memcached实际存放缓存数据的地方，因为chunk的大小固定为slab能够存放的最大值， 所以所有分配给当前slab的数据都可以被chunk存下。如果实际的数据大小小于chunk的大小，空余的空间将会被闲置，这个是为了防止内存碎片而设 计的。举例来说，如果chunk size是224byte，而存储的数据只有200byte，剩下的24byte将被闲置。此外，memcached允许配置的最小的chunk空间为 48个字节（key+value+flags），通过-n参数可以调节这个数值。

[](http://xenojoshua.com/wp-content/uploads/2011/04/memcached-03.png)

## 1.4、理解这三者之间的关系

要理解memcached是如何分配内存的就要从理解上述三个东西之间的关系开始。

page是memcached在收到内存不够的请求，并进行内存分配的单位。举例来说，slab2的所有空间都用完 了，又有大小适合slab2的数据过来了，那么slab2就会向memcached请求新的内存空间，memcached就会划分一个page大小的内存 量到slab2。page的默认大小是1M，这个数值可以通过参数-I来修改。

slab是memcached用来划定存储空间的大小概念，每当memcached启动的时候，它会按照-n参数配置 的值（如果有的话，否则为默认值）来决定第一个slab的大小，然后根据-f参数的值来决定后续slab大小的增长速率，一个一个地决定后续的slab的 大小，直到slab的大小达到设定的page大小（一般是1M）。  
chunk是实际用来存储数据的内存空间，它的大小和包含它的slab的大小是一致的。当page大小的内存分配到slab的时候，slab会根据自身的大小将page大小的内存分割成 page / slabsize 个chunk。

memcached启动时候，slab创建以及chunk分配的细节可以参照下面的数据（使用-vv命令查看的详细内存分配过程）。

/usr/bin/memcached -u nobody -m 64 -p 11211 -l 127.0.0.1 -vv

slab class 1: chunk size 96 perslab 10922

slab class 2: chunk size 120 perslab 8738

slab class 3: chunk size 152 perslab 6898

……………………………………………………………………

slab class 42: chunk size 1048576 perslab 1

## 1.5、举个例子来分析

首先，是memcached启动时候的情况：

商人A很有钱，他有100个大小一摸一样的仓库（100M的memcached服务器，每个page大小1M，就是一个仓库）。商人A根据自己的商品尺 寸，将自己的仓库分成了42种（42个slab），定义为最小一种的仓库是专门用来存放尺寸为96的货物的（slab1大小为96个字节），然后每种仓库 存放的货物大小都是之前一种的1.25倍（增长因子-f为1.25）。商人预先将42个仓库按照预定义的42种货物大小整理、装修了下 （memcached启动时候的42个slab预分配、chunk分割）。1号仓库（slab1）中有10922个（1M \* 1024 \* 1024 / 96）货物存储空间（chunk），后续的仓库类型的装修、空间分配都以此类推。

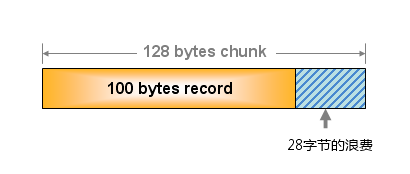
其次，来看下slab满了的时候的情况：

商人A进了一批尺寸是150的货物，共6899个。货物按大小分配，进入3号仓库（ slab3）。因为3号仓库是仓库类型3，其大小只有6898个位置（6898个chunk），6898个货物被安置到仓库类型3（slab3）的3号仓 库里去。然后还多出来一个货物没地方放，商人就安排了一个新的仓库装修成仓库类型3（1M的空间分配给slab3，大小为152个字节，含6898个 chunk），然后将多余的一个货物放入到新的仓库里。

这个例子看过以后，相信大家都已经很明白前述的三个概念之间的关系以及memcached是如何分配内存空间的了。

## 1.6、memcached里的内存浪费

读过上文之后大家应该很明白memcached的内存分配方式了。memcached这样分配内存的好处是不会存在内存碎片，但是坏处也很明显，就 是内存的浪费。就拿前面的商人例子来说，如果遇到一种极端的情况，所有的货物进来的都是121个字节的大小，那么按逻辑他们都会被分到slab3里面去， 也就是分到大小是152的slab里，也就是说每塞进一个对象，就会有31个字节的内存空间被浪费掉了。

[](http://xenojoshua.com/wp-content/uploads/2011/04/memcached-04.png)

## 1.7、memcached的数据回收机制

memcached内部不会监视记录是否过期，而是在get时查看记录的时间戳，检查记录是否过期。 这种技术被称为lazy（惰性）expiration。因此，memcached不会在过期监视上耗费CPU时间。如果某一个item在 memcached里过期了，这个东西并不会被删除，而是客户端无法再看见该记录（invisible，透明）， 其存储空间即可重复使用。一般情况下memcached会优先使用已超时的记录的空间，但即使如此，也会发生追加新记录时空间不足的情况， 此时就要使用名为 Least Recently Used（LRU）机制来分配空间。 顾名思义，这是删除“最近最少使用”的记录的机制。 因此，当memcached的内存空间不足时（无法从slab class 获取到新的空间时），就从最近未被使用的记录中搜索，并将其空间分配给新的记录。

以上，主要是memcached的内存分配利用的一些经验。当然，memcached的配置、调优、监控在这篇文章里是没有涉及的，以后有机会的话会补上

## 1.8、总结



### MemCached主要有两点优势

（1）、高性能，每次查询先从缓存中寻找，找到则直接返回结果；否则，从数据中查询结果并保存到内存中。

（2）、分布式，Memcached通过Hash运算到对应位置查询或缓存其查询结果

（3）、多线程

但是，Memcached只支持k/v类型数据，不支持list，set等数据结果存储，这一点不如redis；还有Memcached由于将数据存入缓存中，因此不利于数据的持久化；一旦及其发生故障，缓存中所有数据将会丢失。

### 在哪些情况下比较适合使用分布式缓存?

（1）、海量数据查询

（2）、访问的集中

（3）、数据增、删、改不频繁

通过缓存数据库查询结果，减少数据访问次数，提高可扩展性。

分布式存储有利于减少其查询范围，提高访问速率；MemCached通过Hash运算，通过哈希值快速找到其位置。

# 二、Java语言实现Memcached缓存技术

## 2.1、在windows下实现Memcached缓存技术步骤

（1）、下载memcached-1.2.1-win32.zip  并且解压.

（2）、在项目中导入必要的包，alisoft-xplatform-asf-cache-2.5.1.jar，

commons-logging-1.0.3.jar，

log4j-1.2.13.jar

（3）、创建工具类MemCachedUtil

（4）、新建一个配置文件MemCached.xml

## 2.2、Windows安装memcached

2.2.1、Memcached-win64 下载：

A、下载最新版：http://blog.couchbase.com/memcached-windows-64-bit-pre-release-available

B、直接下载： memcached-win64-1.4.4-14.zip

http://www.2cto.com/uploadfile/2012/0713/20120713110308123.zip

2.2.2、解压放某个盘下面，比如:

D:\WampServer\bin\memcached\memcached.exe

2.2.3. 在终端（也即cmd命令界面）下输入以下命令安装windows服务:

D:\WampServer\bin\memcached>memcached.exe -d install

2.2.4、再输入下面命令启动：

D:\WampServer\bin\memcached>memcached.exe -d start

## 2.3、Java实现Memcached缓存

### 2.3.1、向POM.xml中添加如下代码

<!-- xmemcached缓存 -->

<dependency>

<groupId>com.googlecode.xmemcached</groupId>

<artifactId>xmemcached</artifactId>

<version>2.0.0</version>

</dependency>

### 2.3.2、添加一个管理类MemcachedManager

**public** **class** MemcachedManager {

**private** **static** MemcachedClient *mcc*;

**static** {

**try** {

String ip = "192.168.6.195";//设置memcached所在的Ip地址

**int** port = 20000;//设置memcached的端口号

**if** (*mcc* == **null**) {

*mcc* = **new** XMemcachedClient(ip, port);

}

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

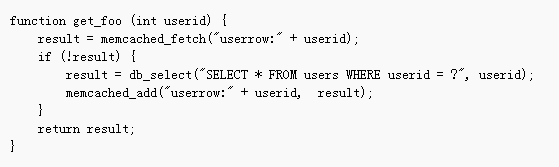
**public** **static** MemcachedClient getInstance() {

**return** *mcc*;

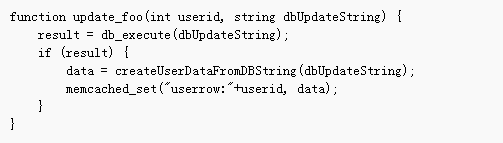
}

}

### 2.3.3、查询数据



### 2.3.4、更新数据



## 2.4、Memcached现状

1、支持更多协议，在已有协议支持的基础上添加了append、prepend、gets、批量gets、cas协议的支持，具体请查看XMemcachedClient类的实例方法。重点是cas操作。

2、memcached分布支持，支持连接多个memcached server，支持简单的余数分布和一致性哈希分布。

3、0.60版本以来的bug修复。

当前1.0-beta版本，支持memcached的分布式（余数哈希和一致性哈希算法）。目前已经支持get、set、add、replace、cas、 append、prepend、批量get/gets、delete、incr、decr、version这几个协议。API为阻塞模型，而非 spymemcached的异步模型，异步模型在批处理的时候有优势，但是阻塞模型在编程难度和使用上会容易很多。

# 三、不理解的知识点

1、使用什么软件测试

2、怎么使用java实现Memcached缓存技术

3、MemCached怎么实现列表的缓存

4、数据库中增加、删除、修改数据时，缓存如何及时的反应