**Set的执行过程**

Set的命令分层两个部分：命令行部分和value部分：

1. 首先是解析命令行，然后初始化item，用于存放缓存数据
2. 命令行部分解析完毕，继续读取socket中的剩余数据报文，边读取边复制到item的数据结构，直到value数据长度和命令行提供的vlen长度一致才会结束。然后存储item，如果存储成功后，则会将item挂带hashtable和LRU链上；如果存储失败，则会删除item。

**Set执行流程**

1. 收到set命令
2. 调用process\_update\_cammand函数：（1）解析命令行（2）分配item数据结构用于存储
3. 调用item\_alloc函数分配一块可用item
4. 调用drive\_machine函数, 进入case ‘conn\_read’，读取缓冲区的数据（value值）
5. 调用complete\_nread函数：（1）调用存储数据item的方法（2）调用删除item的方法
6. 调用do\_store\_item函数存储item（1）add/replace命令，会判断item是否存在，如果已经存在则add命令失败（2）set命令，不管item存在与否，都会创建新的item替换老的item
7. 调用do\_store\_item（1）将item挂到hashtable上（2）将item挂到LRU链上

**Get执行流程**

1. 收到get命令
2. 调用process\_get\_command函数（1）分解get命令（2）通过key去hashtable上找到item的地址值（3）分会找到的item数据

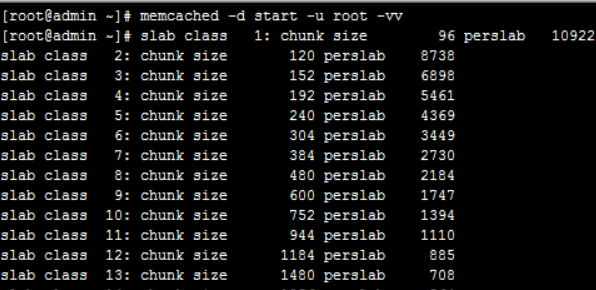
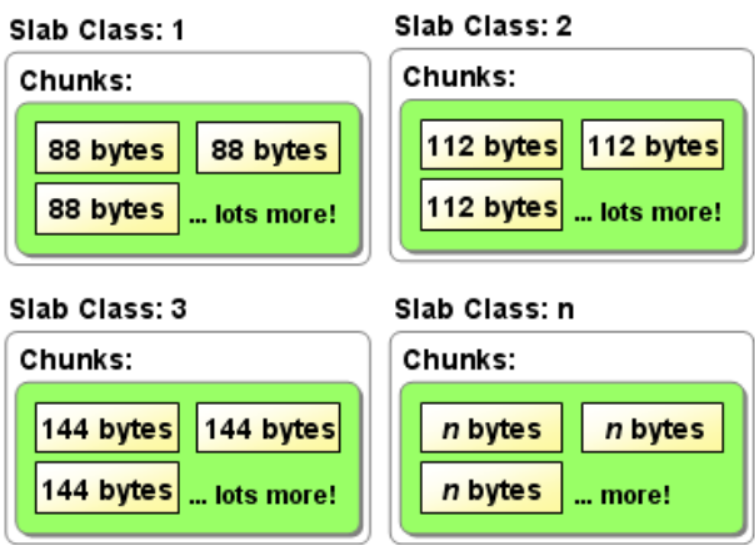
**Delete执行流程**

1. 收到delete命令
2. 调用process\_delete\_command函数（1）查询item是否存在（2）如果存在则删除item，不存在，则返回NOT FOUND
3. 调用item\_unlink和do\_item\_unlink函数将item分别从hashtable和LRU链表上的地址删除
4. 调用item\_remove函数释放item

**Memcached内存分配流程**

1. 调用slab\_init初始化内存
2. 调用slabs\_preallocate对预申请的内存进行划分
3. 调用do\_slabs\_newslab根据每个slabclass区域大小进行划分
4. 调用grow\_slab\_list获取chunk指针数组，不存在则创建，存在且空间不够则扩容
5. 调用memory\_allocate申请一块chunk区域，并根性内存使用量
6. 调用split\_slab\_page\_into\_freelist根据给定的chunk区域指针进行划分item
7. 调用do\_slabs\_free划分item

Memcached是在内存里面维护一块空间（hashtable），将这块空间分割成了很多个slab\_class



然后set命令正确执行会将创建好的item添加到合适的slab\_class里面的chunk里也就是添加到了hashtable中。具体更改操作是在do\_item\_link函数里面调用的assoc\_insert()函数（插入进哈希表），item\_link\_q函数（插入进LRU队列）