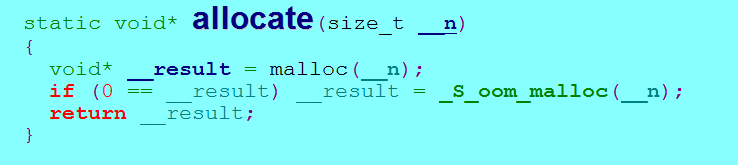
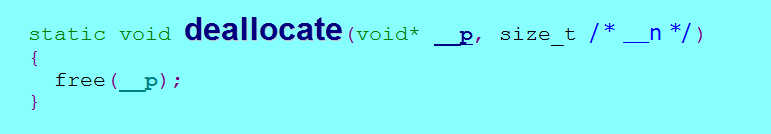
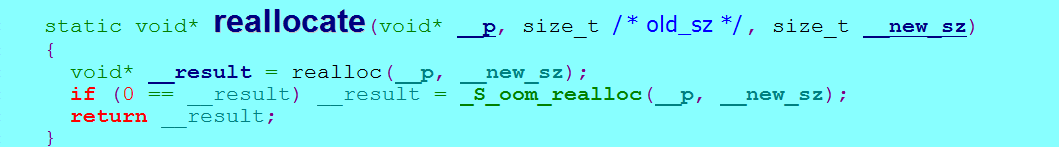
Stl\_alloc.h

STL 的内存分配有两级机制

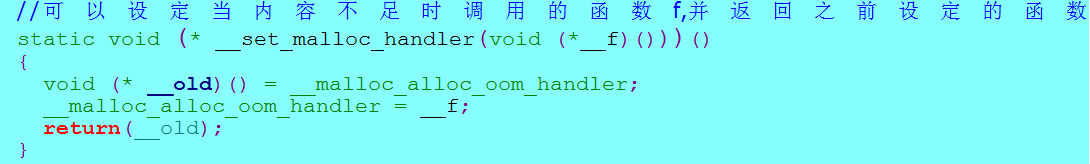
一级分配：当申请的内存大于128字节，采用该分配方式，该分配方式主要通过调用malloc和free申请和释放内存，若申请失败，调用设定的回调函数，通过该自定义函数可以处理内存不足问题。







通过该函数设定回调函数：



二级分配：当申请的内存小于128字节时，采用该分配方式。

首先从内存中申请一个内存池(start\_free——end\_free)

接着当客户端有内存请求时，首先将请求内存大小字节提升为8的整数倍，然后从free\_list数组中得到指定大小内存区域。free\_list数组定义为16个元素，0~15个元素存储指向8字节、16字节、24字节.....128字节的内存链表。

该内存链表的数据结构为union共享体：

union \_Obj {

union \_Obj\* \_M\_free\_list\_link;

char \_M\_client\_data[1]; /\* The client sees this. \*/

};

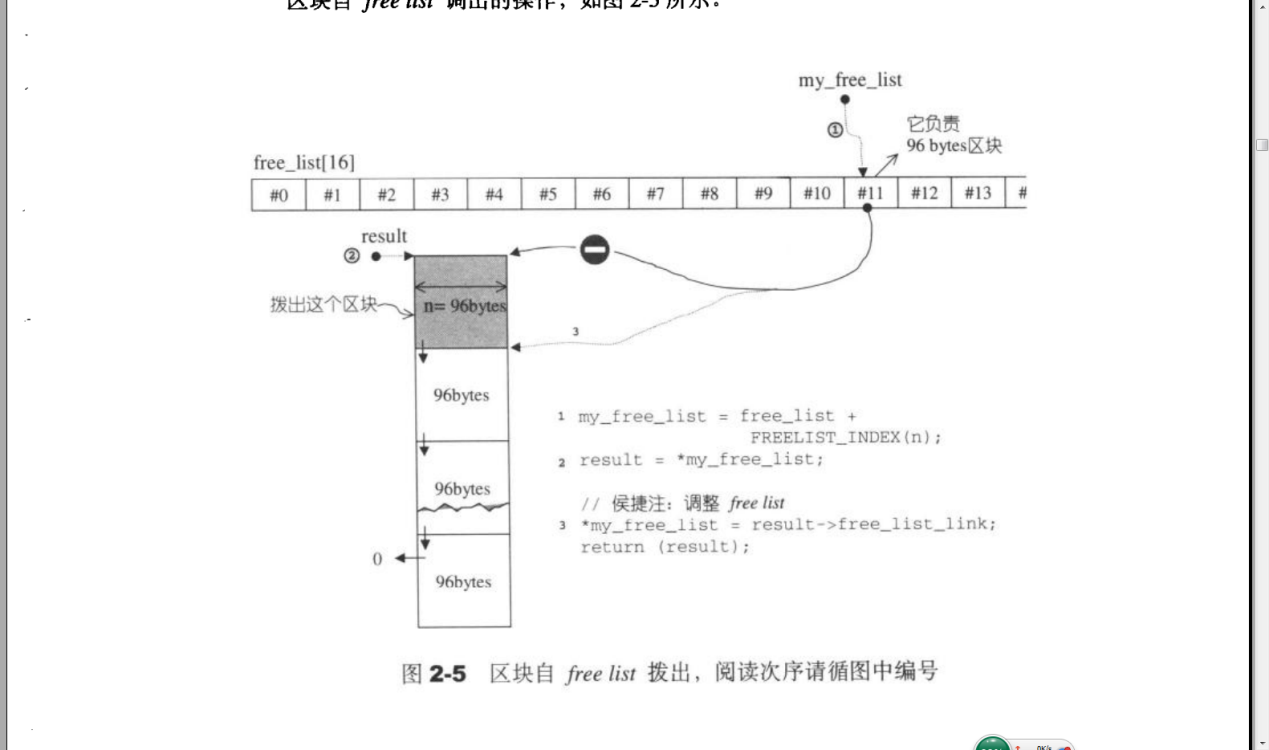
\_M\_free\_list\_link为指针指向下一个union共享体

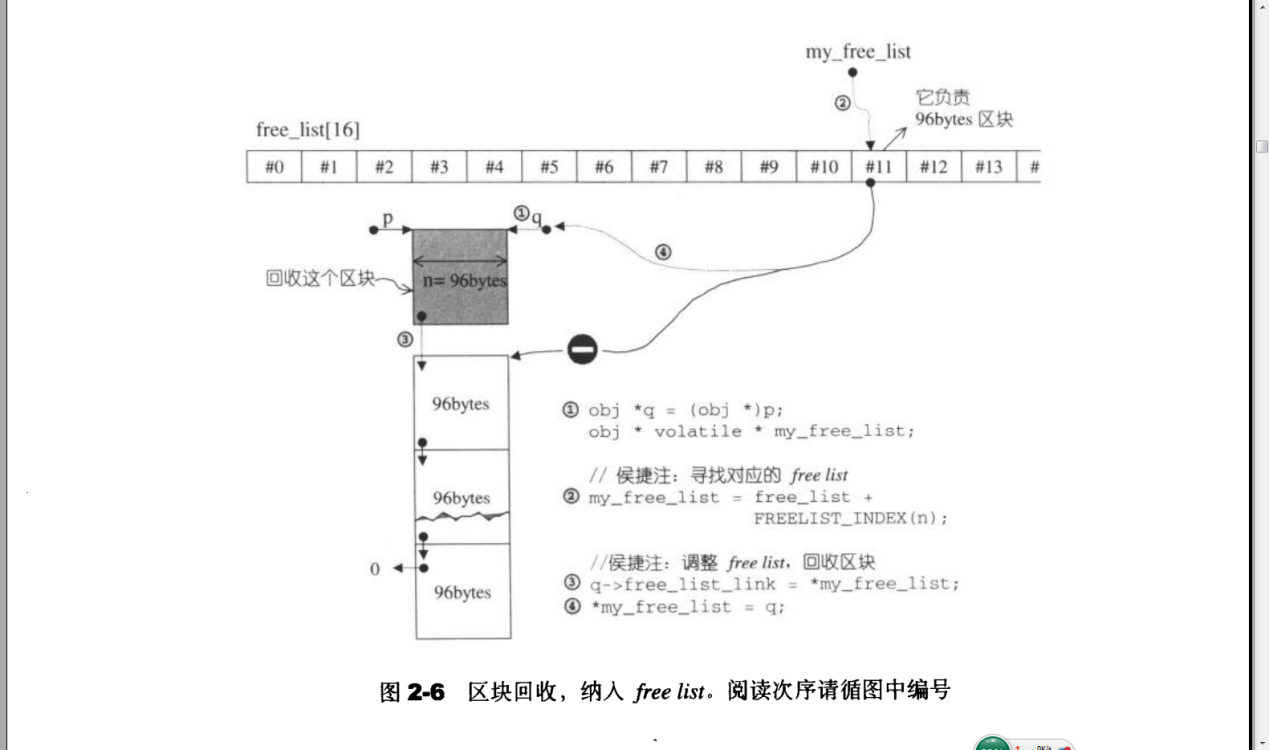
\_M\_client\_data：为指向分配给客户的内存地址

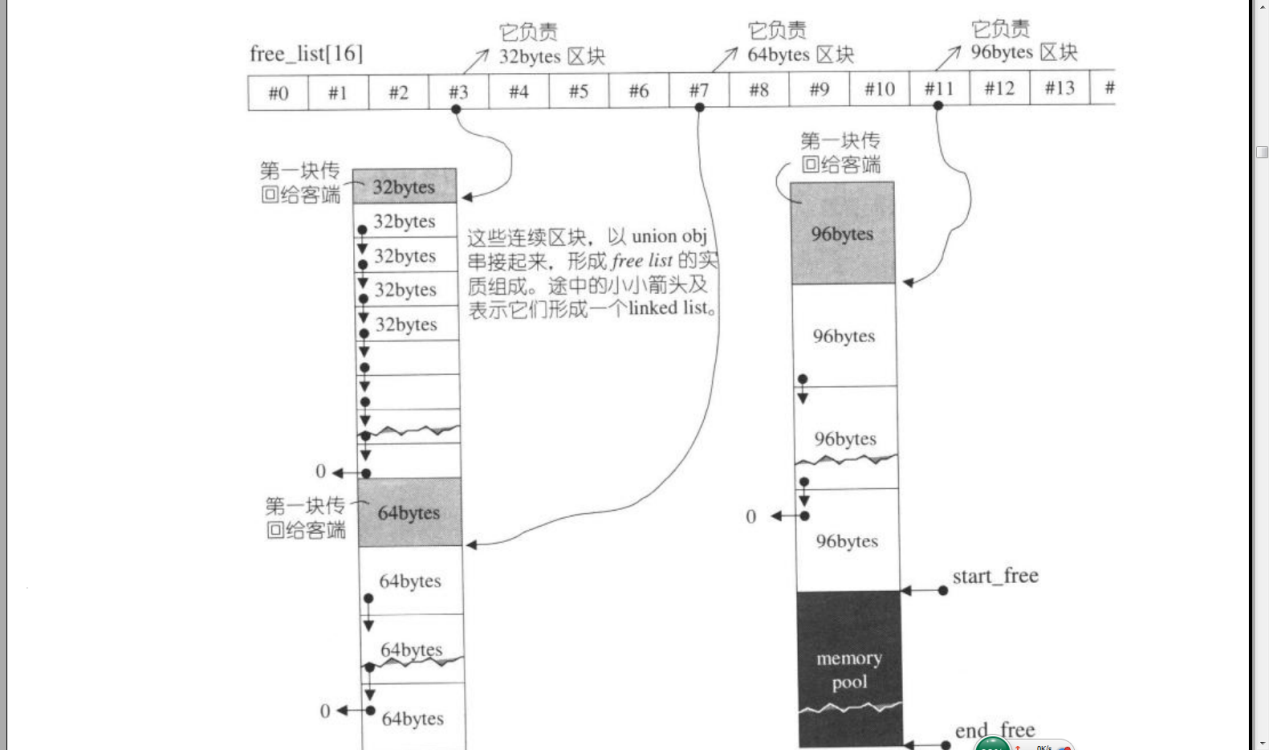
当该内存为空闲时，\_M\_free\_list\_link有效，当该内存被分配给客户是，\_M\_client\_data有效。即该内存区的首地址。

若当前的free\_list数组所指向的元素为空，则从内存池申请20个该内存大小的区域块，将第一个分配给客户，后19个加入free\_list所指向的元素中。

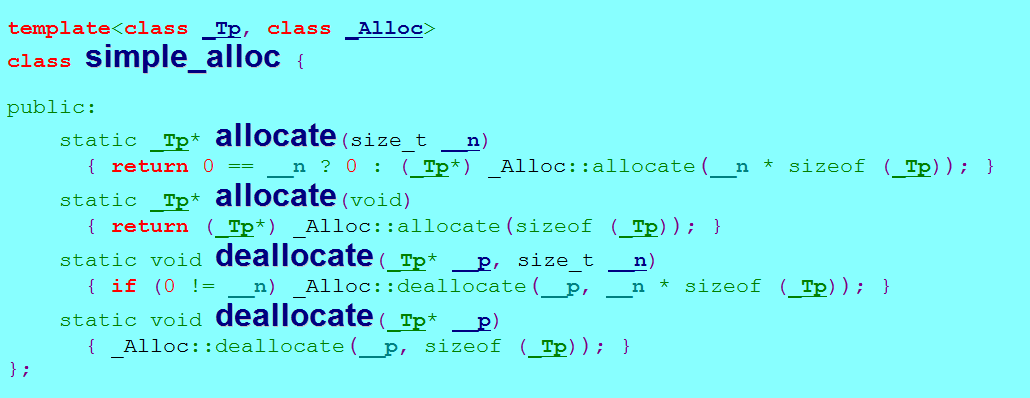
若内存池没有多余空间，则重新向内存申请，空间，若仍然失败，则在free\_list 中找用户要求的内存，比如客户申请8字节，可找16字节的元素是否为0，接着24字节元素是否为0，若不为0，得到内存返给客户。







该内存分配机制提供了一个模板类，通过传递不同的类型，以及不同的内存分配方式



下面是STL源码库中使用该模板的例子：

