[**vector利用swap()函数进行内存的释放**](http://blog.csdn.net/sukhoi27smk/article/details/27505467)

分类： [muduo](http://blog.csdn.net/SUKHOI27SMK/article/category/2250791)2014-05-29 11:2345人阅读[评论](http://blog.csdn.net/sukhoi27smk/article/details/27505467#comments)(0)[收藏](javascript:void(0);)[举报](http://blog.csdn.net/sukhoi27smk/article/details/27505467#report)

首先，vector与deque不同，其内存占用空间只会增长，不会减小。比如你首先分配了10,000个字节，然后erase掉后面9,999个，则虽然有效元素只有一个，但是内存占用仍为10,000个。所有空间在vector析构时回收。

1、释放内存:  
empty()是用来检测容器是否为空的，clear()可以清空所有元素。但是即使clear()，所占用的内存空间依然如故。如果你需要空间动态缩小，可以考虑使用deque。如果非要用vector，这里有一个办法：

在《effective STL》和其实很多C++文章中都有指明，用clear()无法保证内存回收。但是swap技法可以。具体方法如下所示：  
vector<int> ivec;  
ivec.push\_back(1);ivec.push\_back(1);ivec.push\_back(2);ivec.push\_back(2);  
vector<int>().swap(ivec); //或者ivec.swap(vector<int>())；

vector<int>().swap(ivec); 或者如下所示 加一对大括号都可以，意思一样的：  
{  
std::vector<int> tmp;   
ivec.swap(tmp);  
}   
加一对大括号是可以让tmp退出{}的时候自动析构

2、修整空间  
在一个应用中，可能会需要向一个vector中插入很多记录，比如说100000条，为了避免在插入过程中移动内存，咱实现向系统预订一段足够的连续的空间，例如  
vector<int> ivec;  
ivec.reserve(100000);  
这个问题是解决了。  
但是，如果后来这个vector不再需要存那么多的元素了，已经通过erase删除了。但是以前咱们预留的空间却无法被其他程序再度利用，这样会造成内存一定程度上的浪费。于是，我们利用目前的vector构造一个一模一样的vector，他并没有预留空间，于是以前预留的空间也被释放以作他用了：  
ivec.swap(vector<int>(ivec)); // or vector<int>(ivec).swap(ivec)  
或者如下所示 加一对大括号都可以，意思一样的：  
{  
std::vector<int> tmp = ivec;   
ivec.swap(tmp);  
}   
加一对大括号是可以让tmp退出{}的时候自动析构  
  
使用这种方法的前提是vector从前存储了大量数据，比如10000000，经过各种处理后，现在只有100条，那么向清空原来数据所占有的空间，就可以通过这种交换技术swap技法就是通过交换函数swap（），使得vector离开其自身的作用域，从而强制释放vector所占的内存空间。