原 简单的程序诠释C++ STL算法系列之十六：iter_swap

上文中阐述了元素交换算法swap以及容器中swap成员函数的使用，尤其是通过vector成员函数的交换技巧实现容器内存的收缩，今天，我们要看到的是另一个变易算法，迭代器的交换算法iter_swap，顾名思义，该算法是通过迭代器来完成元素的交换。首先我们来看看函数的原型：

函数原型：

```
template<class ForwardIterator1, class ForwardIterator2>
   void iter_swap(
      ForwardIterator1 _Left,
      ForwardIterator2 _Right
   );
```

参数说明：

_Left，_Right指向要交换的两个迭代器

程序示例：

在泛型编程里面，iterator被称为“泛型指针”，因此我们可以通过iterator作为指针来交换两个数组的元素，为了展示swap和iter_swap的区别，在下面这个示例中，我们分别通过这两个算法来实现数组元素的简单交换。

```
/*****************************************************************
 * Copyright (C) Jerry Jiang
 *
 * File Name   : iter_swap.cpp
 * Author      : Jerry Jiang
 * Create Time : 2012-3-26 23:22:18
 * Mail        : jbiaojerry@gmail.com
 * Blog        : http://blog.csdn.net/jerryjbiao
 *
 * Description : 简单的程序诠释C++ STL算法系列之十六
 *               变易算法 : 迭代器交换iter_swap
 *
 *****************************************************************/

#include <iostream>
#include <algorithm>
#include <iterator>

using namespace std;
int main()
{
 //初始化数组
 int b[ 9 ] = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
 ostream_iterator< int > otpt( cout, " " );

 //通过copy+ostream_iterator的方式输出原始数组
 cout << "Array a contains:\n ";
 copy( b, b + 9, otpt );

 //调用iter_swap交换b[0]和b[1]
 iter_swap( &b[0], &b[1] );

 //调用swap交换b[2]和b[3]，展示两者的区别
 swap( b[2], b[3] );

 //通过copy+ostream_iterator的方式输出交换后的数组
 cout << "\n Array a after swapping :\n ";
 copy( b, b + 9, otpt );
 cout << endl;

 return 0;
}
```

上例中，迭代器是作为泛型指针的形式来实现数组元素的交换，现在我们通过iter_swap算法来实现同一种容器之间元素的交换以及不同容器之间的元算交换。

```
/*****************************************************************
 * Copyright (C) Jerry Jiang
 *
 * File Name   : iter_swap02.cpp
 * Author      : Jerry Jiang
 * Create Time : 2012-3-26 23:58:29
```

```
 * Mail        : jbiaojerry@gmail.com
 * Blog        : http://blog.csdn.net/jerryjbiao
 *
 * Description : 简单的程序诠释C++ STL算法系列之十六
 *               变易算法 : 迭代器交换iter_swap
 *
 ****************************************************************/

#include <vector>
#include <deque>
#include <algorithm>
#include <iostream>
#include <ostream>

using namespace std;

int main( )
{
   deque<int> deq1;
   deque<int>::iterator d1_Iter;
   ostream_iterator< int > otpt( cout, " " );

   deq1.push_back ( 2 );
   deq1.push_back ( 4 );
   deq1.push_back ( 9 );

   //通过copy输出队列初始序列
   cout << "The deque is:\n";
   copy(deq1.begin(), deq1.end(), otpt);

   //通过iter_swap算法交换队列中第一个和最后一个元素
   iter_swap(deq1.begin() , --deq1.end());

   //输出通过iter_swap交换后的队列
   cout << "\n\nThe deque of CInts with first & last elements swapped is:\n ";
   copy(deq1.begin(), deq1.end(), otpt);

   //通过swap交换算法还原队列中的元素
   swap (*deq1.begin(), *(deq1.end()-1));

   cout << "\n\nThe deque of CInts with first & last elements swapped back is:\n ";
   copy(deq1.begin(), deq1.end(), otpt);
   cout << endl;

   cout << "*********************************" << endl;


   // 通过iter_swap交换vector和deque两个不同容器中的元素
   vector <int> v1;
   deque <int> deq2;

   //初始化容器v1
   for ( size_t i = 0 ; i <= 3 ; ++i )
   {
      v1.push_back( i );
   }

   //初始化队列deq2
   for ( size_t ii = 4 ; ii <= 5 ; ++ii )
   {
      deq2.push_back( ii );
   }

   cout << "\nVector v1 is : " ;
   copy(v1.begin(), v1.end(), otpt);

   cout << "\nDeque deq2 is : " ;
   copy(deq2.begin(), deq2.end(), otpt);
   cout << endl;

   //交换容器v1和队列deq2的第一个元素
   iter_swap( v1.begin(), deq2.begin() );

   cout << "\n\nAfter exchanging first elements,\n vector v1 is:  " ;
   copy(v1.begin(), v1.end(), otpt);

   cout << " \n deque deq2 is: ";
   copy(deq2.begin(), deq2.end(), otpt);

   cout << endl;
```

```
    cout << endl;

    return 0;
}
```

****

C++经典书目索引及资源下载：http://blog.csdn.net/jerryjbiao/article/details/7358796

****