

# traits help

**Source:** <http://coding.derkeiler.com/Archive/C/ CPP/comp.lang.cpp/2003-10/3697.html>

---

**From:** Rex\_chaos (*rex\_chaos\_at\_21cn.com*)

**Date:** 10/23/03

Date: 23 Oct 2003 13:22:26 -0700

Hi there,

I am learning template programming and there is a problem about traits. Now consider a container and an iterator. Here is the code

```
// tag for const iterator and non-const iterator
struct non_const_iterator_tag {};
struct const_iterator_tag {};

// traits
template <typename T,
          typename tag=non_const_iterator_tag>
struct Iter_traits
{
    typedef random_access_iterator_tag iterator_category;
    typedef T value_type;
    typedef ptrdiff_t difference_type;
    typedef T* pointer;
    typedef T& reference;
};

// For const iterator ...
template <typename T>
struct Iter_traits<T, const_iterator_tag>
{
    typedef random_access_iterator_tag iterator_category;
    typedef T value_type;
    typedef ptrdiff_t difference_type;
    typedef const T* pointer;
    typedef const T& reference;
};

//Iter is an iterator

template <typename T, typename tag=non_const_iterator_tag>
class Iter
{
    typedef typename Iter_traits<T, tag>::value_type value_type;
    typedef typename Iter_traits<T, tag>::reference reference;
```

```

public:
    Iter(T *c) :cont(c) {...}

    // copy constructor to shift the non-const iterator to const
    iterator
    Iter(const Iter<T, non_const_iterator_tag>& it) :cont(it.c)
    {...}

private:
    T *cont;
};

// Container is a container.

template <typename T>
class Container
{
    ...
public:
    typedef Iter<T> iterator;
    typedef Iter<T, const_iterator_tag> const_iterator;

    iterator begin() {...};
    const_iterator begin() const {...};
    iterator end() {...};
    const_iterator end() const {...};
private:
    T *data;
    ...
};

int main(void)
{
    Container<int> c;

    // copy constructor should be actived here
    Container<int>::const_iterator cit=c.begin();
    return 0;
}

```

An error occurs while compliation.

```
`int* Iter<int, non_const_iterator_tag>::cont' is private
```

I don't know why. It seems that it's due to the copy constructor.  
Please help!

BTW, if I define begin() outside the class. e.g.

```

template <typename T>
Container<T>::iterator Container<T>::begin(void)

```

```
{  
...  
}
```

The compiler will give a warning:

`typename Container<T>::iterator' is implicitly a typename  
implicit typename is deprecated, please see the documentation for  
details

Is that anything wrong? Should I do something to prevent that warning?

Thanks in advance.