

Re: determining if a vector is constant

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.stl/2004-06/0041.html>

From: SteveK (asasd_at_asdfasdfs.com)

Date: 06/15/04

Date: Tue, 15 Jun 2004 13:42:29 -0700

Nice, thanks Carl!

I know that class does not have a properly defined class, so if I could, I would fix it, but it's not mine and I don't have the rights at this time.

Your solution is nice and it teaches me how it all works. Thanks again.

"Carl Daniel [VC++ MVP]" <cpdaniel_remove_this_and_nospam@mvps.org.nospam> wrote in message news:e1DAhexUEHA.3664@TK2MSFTNGP12.phx.gbl...

> *SteveK wrote:*

> > *OK, so I implemented this and I think I have hit a small snag,*

> > *actually I know I have hit a small snag.*

> > *First off, I'm a newb. to STL.*

> >

> > *I'm getting an error 2678" no operator != which takes left-hand*

> > *operand of type 'const PosKey'*

> > *PosKey is a class (that I did not create) that has an != operator,*

> > *but it doesn't take a const PosKey, just a PosKey&*

>

> *The class is broken.*

>

> >

> > *I'm actually having a hard time with this, the problem seems simple,*

> > *the solution, while simple... is not an option because I can't change*

> > *the class PosKey and it is this fact that makes me think there is no*

> > *way the STL would force something like this..*

>

> *Force? Not really – it just prefers that your classes are properly defined.*

> *Basically, in order to use `std::adjacent_find` you'll have to write your own*

> *predicate instead of using `std::not_equal_to<T>`.*

>

> >

> > *I must be wrong in interpreting this error.*

>

> *Nope, you're not wrong.*

>

```
> >
> > Any ideas or thoughts for me??
>
> #include <vector>
> #include <functional>
> #include <algorithm>
>
> struct Value
> {
>     bool operator != (Value& rhs)
>     {
>         return true;
>     }
> };
>
> template <class T> struct non_const_not_equal : public
>     std::binary_function<T,T,bool>
> {
>     bool operator()(const T& lhs, const T& rhs) const
>     {
>         return const_cast<T&>(lhs) != const_cast<T&>(rhs);
>     }
> };
>
> void foo()
> {
>     std::vector<Value> v;
>     // put values in v...
>     bool bSame = v.end() !=
>     std::adjacent_find(v.begin(),v.end(),non_const_not_equal<Value>());
> }
>
> -cd
>
>
```