

Re: STL support for 64 bit applications

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.stl/2006-06/msg00048.html>

- *From:* "P.J. Plauger" <pjp@xxxxxxxxxxxxxxxx>
 - *Date:* Sat, 24 Jun 2006 09:30:35 -0400
-

"Stephen Howe" <sjhoweATdialDOTpipexDOTcom> wrote in message
news:u4KgWx4lGHA.4100@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

SGI was the last maintainer of the STL, before large parts of it were incorporated into the C++ standard library. Referring to STL when you mean the C++ stdlib is technically not correct...

It is technically correct. Observe the name of this newsgroup. Do I not see "STL" ?

The fact that large parts of it have been incorporated into C++ standard library is neither here nor there.

What if the OP wanted to talk about the complexity requirements of single-linked lists : <slst>?
That is not in the C++ standard library.

Or ropes?
That is not in the C++ standard library.

Or hash container classes?
That is not in the C++ standard library (yes I know it will be in tr1)

Or algorithms like copy_if() ? is_sorted() ? is_heap() ?
They are not in the C++ standard library.

This newsgroup covers portions of the STL subsumed into the standard library
But it also covers parts of the STL that were not, including founding STL concepts like iterators, iterator traits, algorithms, containers, complexity requirements. That may include new containers, new algorithms, new iterators.

I think the issue is that people have *broadened* the term STL to include the entire Standard C++ library. It's an easy enough mistake to make, and colloquially accurate enough for most purposes. Moreover, it has the added advantage that it gives purists yet another thing to quibble about...

Re: STL support for 64 bit applications

P.J. Plauger
Dinkumware, Ltd.
<http://www.dinkumware.com>