Subject: container.At(not_existed_item_index)
Posted by qwerty on Wed, 31 May 2006 08:46:19 GMT
View Forum Message <> Reply to Message

...what is "default constructed" value for item inside container, where I getting value w/ .At and there is not such a value / that index ?

I guess, its constructor like thong, but generic types(int, double...)?

parsed through code(in rush), not found...

thanx

Subject: Re: container.At(not_existed_item_index)
Posted by mirek on Wed, 31 May 2006 09:15:53 GMT
View Forum Message <> Reply to Message

qwerty wrote on Wed, 31 May 2006 04:46...what is "default constructed" value for item inside container, where I getting value w/ .At and there is not such a value / that index ?

I guess, its constructor like thong, but generic types(int, double...)?

parsed through code(in rush), not found...

thanx

A good question. I guess this needs discussion / better documentation.

Right now, int, double etc... are left uninitialized, whereas for types with constructor default constructor is used. I guess this is faster and you can always specify "init".

It is true that this is unsimiliar to STL, where quite often fundamental types are zeroed using the T() "constructor". Frankly, this C++ feature is something I never really liked...

Mirek

Subject: Re: container.At(not_existed_item_index)
Posted by qwerty on Wed, 31 May 2006 09:32:43 GMT
View Forum Message <> Reply to Message

It would be nice if there can be:

Array<int> my_array(default_value)

or
Array<int> my_array(default_function(default_args))

or something_like_that...

Subject: Re: container.At(not_existed_item_index)
Posted by qwerty on Wed, 31 May 2006 10:02:32 GMT
View Forum Message <> Reply to Message

...and there's a items with argumented constructor only too...

maybee use of callbacks come in handy for all mentioned

Subject: Re: container.At(not_existed_item_index)
Posted by qwerty on Wed, 31 May 2006 10:28:29 GMT
View Forum Message <> Reply to Message

yes, callback could be fine if .At(non_existet_item)

Subject: Re: container.At(not_existed_item_index)
Posted by mirek on Wed, 31 May 2006 11:49:25 GMT
View Forum Message <> Reply to Message

qwerty wrote on Wed, 31 May 2006 05:32It would be nice if there can be:

Array<int> my_array(default_value)
or
Array<int> my_array(default_function(default_args))

or something_like_that...

I do not believe so. It would make containers too heavy and is not really needed in practice.

Adding another parameter to At is really very easy and in fact, does it matter whether the default_value is specified in constructor, or in At call? What is the point of storing value inside container just because of this?

Mirek

Subject: Re: container.At(not_existed_item_index)

Posted by qwerty on Wed, 31 May 2006 14:49:30 GMT
View Forum Message <> Reply to Message

I like flowing throught my mind and there is so many ideas and so low knowledge of upp

Subject: Re: container.At(not_existed_item_index)
Posted by mirek on Wed, 31 May 2006 15:33:01 GMT
View Forum Message <> Reply to Message

qwerty wrote on Wed, 31 May 2006 10:49I like flowing throught my mind and there is so many ideas and so low knowledge of upp

That is OK.

Actually, the real thing to perhaps reconsider here is "zero" vs "initialized" for fundamentals... But I believe that additional "init" parameter makes that OK...

Mirek

Subject: Re: container.At(not_existed_item_index)
Posted by qwerty on Wed, 31 May 2006 15:36:57 GMT
View Forum Message <> Reply to Message

so do I, thank you for your response...

...to be complete, is there any other way, than for(...) to traverze throught container in conjunction with possibility of modifying container during traversing ?

it's something like foreach, I guess, but there's nothing too much to add...

Subject: Re: container.At(not_existed_item_index)
Posted by mirek on Wed, 31 May 2006 16:00:33 GMT
View Forum Message <> Reply to Message

qwerty wrote on Wed, 31 May 2006 11:36so do I, thank you for your response...

...to be complete, is there any other way, than for(...) to traverze throught container in conjunction with possibility of modifying container during traversing ?

it's something like foreach, I guess, but there's nothing too much to add...

No, I even consider iterators deprecated now...

Last 5 years, we were almost exclusively using simple index based iteration. It has quite a lot of advantages:

- cheap range checking
- no problems with invalidated iterators
- possibility of coupling containers with other structures

That is why NTL is so index-centric now..

Mirek

Subject: Re: container.At(not_existed_item_index)
Posted by qwerty on Wed, 31 May 2006 16:19:49 GMT
View Forum Message <> Reply to Message

after reading one special article, it seems that much clearer to me

---