Subject: Question about SubRange.
Posted by Novo on Thu, 04 Jan 2018 20:58:35 GMT
View Forum Message <> Reply to Message

Function SubRange has two overloads below.

```
template <class C>
auto SubRange(C& c, int pos, int count) -> decltype(SubRange(c.begin() + pos, count))
{
 return SubRange(c.begin() + pos, count);
}

template <class C>
auto SubRange(C&& c, int pos, int count) -> decltype(SubRange(c.begin() + pos, count))
{
 return SubRange(c.begin() + pos, count);
}
```

The second one doesn't actually move anything, so, it is just not needed, IMHO.
The first one can always be used instead. And it should look like below. IMHO.

```
template <class C>
auto SubRange(const C& c, int pos, int count) -> decltype(SubRange(c.begin() + pos, count))
{
 return SubRange(c.begin() + pos, count);
}
```

Method begin() is always const I believe ...
And you can drop this decltype in C++14 ...

Subject: Re: Question about SubRange.
Posted by mirek on Mon, 08 Jan 2018 10:52:02 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 04 January 2018 21:58Function SubRange has two overloads below.

```
template <class C>
auto SubRange(C& c, int pos, int count) -> decltype(SubRange(c.begin() + pos, count))
{
 return SubRange(c.begin() + pos, count);
}

template <class C>
auto SubRange(C&& c, int pos, int count) -> decltype(SubRange(c.begin() + pos, count))
```

```
{
 return SubRange(c.begin() + pos, count);
}
```

The second one doesn't actually move anything, so, it is just not needed, IMHO.
The first one can always be used instead. And it should look like below. IMHO.

```
template <class C>
auto SubRange(const C& c, int pos, int count) -> decltype(SubRange(c.begin() + pos, count))
{
 return SubRange(c.begin() + pos, count);
}
```

Method begin() is always const I believe ...

Not true (not in U++ nor STL). That is basically a reason for those 2 overloads. E.g. try Sort with just const variant...

Quote:
And you can drop this decltype in C++14 ...

Not if we want to maintain C++11 compatibility (current status is that we do and we will as long as it is cheap. I think decltype here is cheap).

Subject: Re: Question about SubRange.
Posted by Novo on Mon, 08 Jan 2018 14:55:01 GMT
View Forum Message <> Reply to Message

mirek wrote on Mon, 08 January 2018 05:52
Not true (not in U++ nor STL). That is basically a reason for those 2 overloads. E.g. try Sort with just const variant...

The "C& c" variant is definitely not needed. The "C&& c" variant is an universal reference and it is a complete replacement of the "C& c" variant. I tried to compile TheIDE with "C& c" commented out and it compiles just fine.

The problem with the "const C& c" variant is that when you try to construct the SubRangeClass you get a SubRangeClass<const I> version.
I tried to strip the constness like below

```
template <class I>
SubRangeClass<I> SubRange(I l, I h)
{
```

```
 return SubRangeClass<typename std::remove_const<I>::type>(l, h);
}

template <class I>
SubRangeClass<I> SubRange(I l, int count)
{
 return SubRangeClass<typename std::remove_const<I>::type>(l, count);
}
```

But I'm still getting weird compilation problems with TabBar::Tab.

And I'm still a little bit confused about what is IterSwap for. What is wrong with the regular Swap?

Something is not right about all this ...

Subject: Re: Question about SubRange.
Posted by mirek on Mon, 08 Jan 2018 15:21:04 GMT
View Forum Message <> Reply to Message

Novo wrote on Mon, 08 January 2018 15:55mirek wrote on Mon, 08 January 2018 05:52
Not true (not in U++ nor STL). That is basically a reason for those 2 overloads. E.g. try Sort with just const variant...

The "C& c" variant is definitely not needed. The "C&& c" variant is an universal reference and it is a complete replacement of the "C& c" variant. I tried to compile TheIDE with "C& c" commented out and it compiles just fine.

theide is not a good representative, as there probably is not a single use of subrange.

It is quite possible that C& is not needed, but I think I had some issues with at least some compilers with that approach. But it would really be a great if I could remove them...

Quote:
And I'm still a little bit confused about what is IterSwap for. What is wrong with the regular Swap?

When sorting Array, you can swap just pointers instead of objects - it is optimization.

Mirek

Subject: Re: Question about SubRange.
Posted by mirek on Wed, 10 Jan 2018 19:56:04 GMT
View Forum Message <> Reply to Message

I have tried to remove l-value references, leaving only universal T&& references just as you suggest. So far seems good.

Thanks.

Mirek
