
Subject: Re: Core chat...

Posted by [mirek](#) on Thu, 25 Oct 2007 21:38:04 GMT

[View Forum Message](#) <> [Reply to Message](#)

mdelfede wrote on Thu, 25 October 2007 17:26luzr wrote on Thu, 25 October 2007 21:33

You can try. However, long time ago, such class template was part of U++. But there was no use for it. "pick" is confusing at first, but quite powerful concept.

I do find "pick" quite clear. You get the top performance at at the expense of some loss of easy to use stuff.

Well, performance is nice, but really not that important.

What IS imporant is that there is only ONE PLACE where instance of (possibly) non-copyable object can exist.

Quote:

```
array<int> a, b;  
a.At(1000) = 1;  
b = a;    <== b and a share same memory area  
a[10] = 2; <== here an automatic deep copy, ok  
b[10] = 2; <== here no deep copy, just array access, ok
```

Note that above is impossible to implement reliably in C++ (as long as you want read operator[] access to perform no copy at all).

Quote:

array a gets 2 references to it for a while, just when MyFunc returns a value. Then, temporary from MyFunc get destroyed, leaving a with a single reference. Any subsequent access to a[] don't need a deep copy.

Sure. Anyway, the real point of pick is here:

```
Array<Ctrl> CreateWidgets()  
{  
    Array<Ctrl> x;  
    ...  
    return x;  
}
```

Mirek
