Subject: C++11 library features finished
Posted by mirek on Mon, 31 Aug 2015 11:38:04 GMT
View Forum Message <> Reply to Message

I have finished adding C++11 features for the next release. Basically, there is support for C++11 initializers and also support for lambdas in Callbacks.

Lambdas have problem as all-catch templated constructor in std::function creates overloading issues. Previously, I was trying to solve it using LAMBDA macro, but it proved cumbersome. New approach allows direct us of lambda via operator<< (beacause we do not overload that one much for callbacks).

Demo in reference/Cpp11

```
#include "CtrlLib/CtrlLib.h"

using namespace Upp;

#define LAYOUTFILE <Cpp11/Cpp11.lay>
#include <CtrlCore/lay.h>

GUI_APP_MAIN
{
 WithMyAppLayout<TopWindow> dlg;
 CtrlLayout(dlg, "C++11 demo");
 dlg.list.NoHeader().AddColumn();
 Vector<int> x = { 1, 2, 12, 34, 15, 11 };
 for(auto i : x)
  dlg.list.Add(i);
 dlg.add << [&] {
  if(dlg.list.Find(~dlg.number) < 0)
   dlg.list.Add(~dlg.number);
 };
 dlg.list.WhenSel << [&] {
  dlg.number <<= dlg.list.GetKey();
 };
 dlg.Execute();
}
```

Subject: Re: C++11 library features finished
Posted by Mindtraveller on Mon, 31 Aug 2015 13:16:16 GMT
View Forum Message <> Reply to Message

Very cool, thank you!
It looks like time to get used to new coding style. Finally we have inlined (lambda) callbacks

without declaring new member functions.

Subject: Re: C++11 library features finished
Posted by slashupp on Sun, 15 Nov 2015 16:16:05 GMT
View Forum Message <> Reply to Message

Am using this feature all-over the place now
brilliant time & effort saver

thx

Subject: Re: C++11 library features finished
Posted by Mindtraveller on Wed, 18 Nov 2015 23:59:17 GMT
View Forum Message <> Reply to Message

Still having rare issues with newest C++11-style code:

```
 struct GLTextureDescriptor : Moveable<GLTextureDescriptor>
 {
  int w,h,wTex,hTex;
  struct GLTex : Moveable<GLTex>
  {
   int w,h;
   GLuint glId;
  };
  Vector<Vector<GLTex>> glTex;
 };
 VectorMap<String,GLTextureDescriptor> glTextureDescriptors;

 //...
 GLTextureDescriptor desc;
 //...
 glTextureDescriptors.AddPick(key, desc); // <-- compiler error, cannot cast 2nd argument to GLTextureDescriptor &&
```

I wasn't able to find quick & easy solution for this case.

Subject: Re: C++11 library features finished
Posted by Novo on Thu, 19 Nov 2015 03:41:51 GMT
View Forum Message <> Reply to Message

There is another C++11 problem. Checked with MSVC2015.

```
One<MFile> mfile;
One<MFile> cur_mfile(new MFile);
mfile = cur_mfile; // <-- Upp::One<MFile> &Upp::One<MFile>::operator =(const Upp::One<MFile>
&)': attempting to reference a deleted function
```

I need to detach value explicitly.

```
mfile = cur_mfile.Detach();
```

Thanks.

Subject: Re: C++11 library features finished
Posted by mirek on Thu, 19 Nov 2015 14:29:26 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Thu, 19 November 2015 00:59Still having rare issues with newest C++11-style code:

```
 struct GLTextureDescriptor : Moveable<GLTextureDescriptor>
 {
  int w,h,wTex,hTex;
  struct GLTex : Moveable<GLTex>
  {
   int w,h;
   GLuint glId;
  };
  Vector<Vector<GLTex>> glTex;
 };
 VectorMap<String,GLTextureDescriptor> glTextureDescriptors;

 //...
 GLTextureDescriptor desc;
 //...
 glTextureDescriptors.AddPick(key, desc); // <-- compiler error, cannot cast 2nd argument to
GLTextureDescriptor &&
```

I wasn't able to find quick & easy solution for this case.

AddPick(key, pick(desc)) should work here. Does it not?

Subject: Re: C++11 library features finished
Posted by mirek on Thu, 19 Nov 2015 14:31:09 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 19 November 2015 04:41There is another C++11 problem. Checked with

MSVC2015.

One<MFile> mfile;
One<MFile> cur_mfile(new MFile);
mfile = cur_mfile; // <-- Upp::One<MFile> &Upp::One<MFile>::operator =(const Upp::One<MFile> &)': attempting to reference a deleted function

I need to detach value explicitly.

mfile = cur_mfile.Detach();

Thanks.

Same thing... mfile = pick(cur_mfile).

Simple: You now need to be explicit, except for temporaries.

Bit more verbose, but in fact I was able to find some bugs at compile time already thanks to this..

Subject: Re: C++11 library features finished
Posted by Mindtraveller on Thu, 19 Nov 2015 17:05:14 GMT
View Forum Message <> Reply to Message

Mirek, you are absolutely right. Somehow I forgot to use pick/AddPick combination. C++11-style coding rules seem more strict but it should help avoid bugs. Thanks!

Subject: Re: C++11 library features finished
Posted by Novo on Fri, 20 Nov 2015 01:50:20 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 19 November 2015 09:31
Same thing... mfile = pick(cur_mfile).

Simple: You now need to be explicit, except for temporaries.

Bit more verbose, but in fact I was able to find some bugs at compile time already thanks to this..

Thanks! I missed that.

Subject: Re: C++11 library features finished
Posted by Alboni on Wed, 01 Jun 2016 01:22:16 GMT
View Forum Message <> Reply to Message

Something weird going on. I had an accident with my .upp folder and deleted it. Now that hasn't been a disaster before; upp will recreate it, I had no special configuration in my build method on linux. But now I can't compile anything without adding -std=gnu++11
I didn't have to add anything before. But that still doesn't work, as AtomicWrite() doesn't seem to exist anymore and there seems to be a problem with CallbackNP.i see attachment.

so it sounds like extra flags are needed in the build method that is not put there by default (any more?) in the current version?
But everything compiled right yesterday. Haven't changed anything else.

File Attachments
1) weirdness.png, downloaded 920 times

Subject: Re: C++11 library features finished
Posted by mirek on Wed, 01 Jun 2016 21:24:28 GMT
View Forum Message <> Reply to Message

Uhm, perhaps you have missed the annoucement that trunk now requires c++11?

As for AtomicWrite, it is removed because it is not necessary. Just assign the damned variable...

Mirek

P.S.: You can still have 'old' U++ is you use 'classic' instead of 'trunk'.

Subject: Re: C++11 library features finished
Posted by Novo on Fri, 03 Jun 2016 17:27:55 GMT
View Forum Message <> Reply to Message

code\gui\upp\trunk\uppsrc\core\map.h(178) : warning C4717:
'Upp::AMap<int,Upp::Vector<int>,Upp::Vector<Upp::Vector<int> > >::FindPutPick': recursive on all control paths, function will cause runtime stack overflow

MSVC 2015 x64.
Just to let you know.
The same problem should happen to FindAddPick and GetAddPick.
Although they are deprecated, they still are really recursive, and apps crash because of that.

I believe you wanted to implement them this way:
FindPutPick(const K& k, T&& init) { return FindPut(k, pick(init)); }

TIA.

Subject: Re: C++11 library features finished
Posted by Novo on Fri, 03 Jun 2016 17:38:52 GMT
View Forum Message <> Reply to Message

Another thing. IMHO, AMap::GetPutPick(const K& k, T&& x) should be renamed into GetPut, and GetPutPick should be deprecated.

TIA

Subject: Re: C++11 library features finished
Posted by Novo on Sat, 04 Jun 2016 03:36:25 GMT
View Forum Message <> Reply to Message

There is a bug in AMap<K, T, V>::FindPut(const K& k, T&& init). It should look like below.

```
template <class K, class T, class V>
int AMap<K, T, V>::FindPut(const K& k, T&& init)
{
 unsigned hash = key.hashfn(k);
 int i = Find(k, hash);
 if(i < 0) {
  i = key.Put(k, hash);
  value.At(i) = pick(init);
 }
 return i;
}
```

The same problem can be found in other methods where you are using r-value directly (without "pick").
Below is a test case.

```
 VectorMap<int, Vector<int> > x;
 x.FindPut(1, Vector<int>());
```

You can just try to compile all methods taking r-value and see what happens.

EDIT: I've attached a patch.

Regards

File Attachments

1) AMap.zip, downloaded 282 times

Subject: Re: C++11 library features finished
Posted by mirek on Thu, 09 Jun 2016 12:05:12 GMT
View Forum Message <> Reply to Message

Thanks, you are completely correct.

(BTW, perhaps you have some testing code now - if yes, I can add it to nightly tests, just send it to me).

Subject: Re: C++11 library features finished
Posted by Novo on Fri, 10 Jun 2016 03:09:49 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 09 June 2016 08:05Thanks, you are completely correct.

(BTW, perhaps you have some testing code now - if yes, I can add it to nightly tests, just send it to me).
I do not have special tests for this problem. I spotted it by trying to run an old app.

I've attached another similar patch. I'm not 100% sure about each change, but it looks like I'm right.

Regards

File Attachments
1) r-value.patch, downloaded 288 times

Subject: Re: C++11 library features finished
Posted by mirek on Sat, 11 Jun 2016 18:55:48 GMT
View Forum Message <> Reply to Message

Thanks, applied as well.