Subject: unsigned long / uint64 from Value type
Posted by EspressoMan on Mon, 08 Jan 2024 11:41:34 GMT
View Forum Message <> Reply to Message

In September 2006, a member posted this title: Value with unsigned support [message #5214]

17 years on, I need to ask this question again. To be fair, as a new user, I could simply be missing something basic due to inexperience with U++

On my form I have an EditInt64 control. I would like users to be able to enter any integer up to 2^64-1, i.e, the maximum unsigned 64 bit integer.
However, there appears to be some kind of "mask" which triggers a colour change in the field when anything bigger than a signed 64 bit integer is input. (see pic below). Now, typecasting a signed 64 bit int into an unsigned 64 bit int will of course work but the maximum signed 64 bit int is 2^63-1 and the EditInt64 won't allow anything bigger to be input. Can some kind member please advise how my requirement can be realised.

Thanks: Colin

File Attachments
1) Screenshot from 2024-01-08 23-50-12.png, downloaded 190 times

Subject: Re: unsigned long / uint64 from Value type
Posted by Klugier on Mon, 08 Jan 2024 20:47:05 GMT
View Forum Message <> Reply to Message

Hello,

Looking at the code it seems that EditInt64 is basing on int64 type (CtrlLib/EditCtrl.h - line 309):

```
typedef EditMinMax<int64, ConvertInt64>         EditInt64;
```

So, to add support for unsigned type (uint64) there is a need for expansion of this code. There is a need to add following entries:

```
typedef EditMinMax<uint64, ConvertUInt64>         EditUInt64;
// ...
typedef EditMinMaxNotNull<uint, EditUInt64>      EditUInt64NotNull;
```

```
// ... (Many lines later there is something called spin variant)
typedef WithSpin<int64, EditInt64>          EditInt64Spin;
```

Also, there is a need to implement following classes:

```
class ConvertUInt64
class EditUInt64
```

I personally think that supporting unsigned variants would be nice. However, I don't have much time now to implement it by myself. However, if you want to help you can try to expand CtrlLib in the places I have mentioned. After expansion you can create PR to our official repo and then these changes should be in master and in future releases of U++.

Don't forget to modify CtrlLib.usc. There would be good to have these new controls out of the box in layout designer!

In case of any questions please fell free to ask.

Klugier

Subject: Re: unsigned long / uint64 from Value type
Posted by EspressoMan on Tue, 16 Jan 2024 10:36:09 GMT

View Forum Message <> Reply to Message

Ok Klugier
It's a total distraction from my current project but it's probably a good exercise to try - I have questions...

1) Is there a diagram of how all the (control) classes inter-relate? I mean the full U++ framework hierarchy?

2) I am assuming that the proposed 'EditUint64' control is similar to the implementation of 'EditInt64'? I searched through the source dirs and found:
 struct ConvertInt64 : public ConvertInt in Core/Convert.h
But I couldn't find the declaration for EditInt64. Which header is it in?

3) When searching for the above, I found quite a few references to the int64 type in the context of converting that to/from 'Value'
for example:
Class Ref in ValueUtil.h
Struct IniInt64 in Util.h
ReadInt64() and ReadNumber64() in Parser.h

I'm also wondering about other unsigned integer types... Any comments or guidance would help to clarify the process for me.

Cheers
Colin

Subject: Re: unsigned long / uint64 from Value type
Posted by EspressoMan on Sat, 20 Jan 2024 02:01:41 GMT
View Forum Message <> Reply to Message

Hi again Klugier
So I added the following section to CtrlLib.usc

```
 // cjm 2024-01-17 17:32:45
 // Effectively just copied from IntStr & EditInt64 above but changed types to uint64

 fn IntStr64(x)
 {
  return x == :IntNull || x < :DblNullLim ? "" : to_string(x); // :IntNull, :DblNullLimit ???
 }

 ctrl EditUint64
 {
  group "Input fields";
  >EditNotNull;
  uint64 Min;
  uint64 Max;

  PaintData(w) { PaintMinMax(w, "Uint64", IntStr64(.Min), IntStr64(.Max)); } // Note function name
change
 }

 // cjm 2024-01-17 17:32:45
```

This compiles no problem. However, there's something a strange when I try to add the control to my Layout. Clearly the Paint process is not happy!

Specifically, the PaintData(w) { PaintMinMax(...); } is giving me the following error message in the console of Layout designer

```
...PaintData: lambda.PaintData(): PaintMinMax: IntStr64: lambda.IntStr64():
/opt/upp/uppsrc/CtrlLib/CtrlLib.usc(895,49): invalid values for comparison array < double
```

};

I have searched the site and forums and can find no reference to :IntNull and :DblNullLim

Can you please suggest what is happening? It's fairly obviously to do with the type of Min and Max being set as uint64

Thanks
Colin

File Attachments

1) Screenshot from 2024-01-20 14-55-36.png, downloaded 114 times

---

Subject: Re: unsigned long / uint64 from Value type
Posted by mirek on Sat, 16 Mar 2024 15:59:14 GMT
View Forum Message <> Reply to Message

Hi,

I suppose this is one-off case, so adding this to U++ or even .usc is not worth it. I would to this:

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

struct U64Convert : Convert {
 Value Format(const Value& q) const override {
  return FormatUInt64((uint64)(int64)q);
 }

 Value Scan(const Value& text) const override {
  uint64 x;
  bool overflow = false;
  String s = ~text;
  if(ScanUint<char, byte, uint64, 10>(x, s))
   return (int64)x;
  return ErrorValue("Invalid number");
 }
 int Filter(int chr) const override {
  return CharFilterDigit(chr);
 }
};

GUI_APP_MAIN
```

```
{
 TopWindow win;
 EditField h;
 h.SetConvert(Single<U64Convert>());
 win << h.HSizePos().TopPos(0);
 uint64 x = UINT64_MAX;
 h <<= (int64)x;
 win.Run();
 x = (int64)~h;
 DDUMP(x);
}
```