Subject: MSC15 and 2014 sources - problem with Stream
Posted by NilaT on Fri, 22 Jul 2016 06:26:51 GMT
View Forum Message <> Reply to Message

Hello,

I already started a thread on a wrong forum, please move or delete the Thread from there and put it in here.
http://www.ultimatepp.org/forums/index.php?t=msg&th=9720 &start=0&

My problem is, I'm using the latest Upp (9251) with 2014 sources and compile with MSC15.
But when I call Cout() in my prog, it crashes.
This only happens with MT flag and in optimal mode (debug runs fine)

The same error can be produced on the newest 2015 sources, when putting CoutUTF8() before the first Cout().
I'm running Win 10 Pro.

You can see the error in the other thread as well, but it's:
Exception at 0x00AB60C7 (address changes every time) in (Projectname).exe: 0xC0000005: Access violation at read at position 0x0000004.

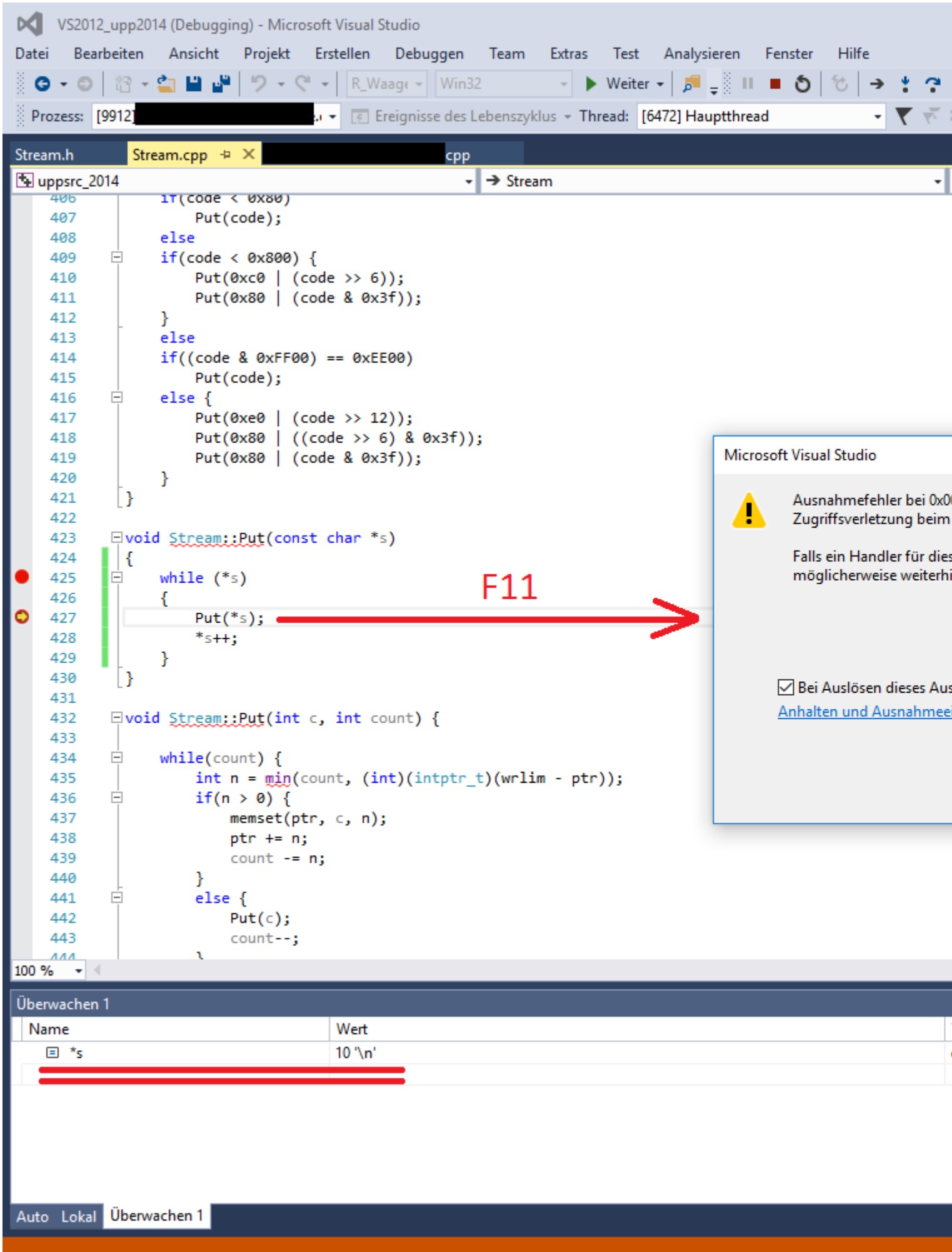
The error seems to occur in function void Stream::Put(const char *s).
See attachment.

Thanks for the quick reply, as I need this program really quick.

//edit:
Okay guys, I'm kind of confused now.
When I start the little test program from the other thread with 2015 sources, it works fine. In 2014 sources, it crashes.
BUT, when I start MY program in 2015 sources, it crashes as well as on 2014 sources.
Very confusing.
Any help would be very appreciated.....

File Attachments

1) crash.png, downloaded 704 times


VS2012_upp2014 (Debugging) - Microsoft Visual Studio
Datei Bearbeiten Ansicht Projekt Erstellen Debuggen Team Extras Test Analysieren Fenster Hilfe
Weiter
Prozess:
Ereignisse des Lebenszyklus
Thread: [6472] Hauptthread
Stream.h
Stream.cpp
cpp
uppsrc_2014
Stream
406 if(code < 0x80)
407 Put(code);
408 else
409 if(code < 0x800) {
410 Put(0xc0 | (code >> 6));
411 Put(0x80 | (code & 0x3f));
412 }
413 else
414 if((code & 0xFF00) == 0xEE00)
415 Put(code);
416 else {
417 Put(0xe0 | (code >> 12));
418 Put(0x80 | ((code >> 6) & 0x3f));
419 Put(0x80 | (code & 0x3f));
420 }
421 }
422
423 void Stream::Put(const char *s)
424 {
425 while (*s)
426 {
427 Put(*s);
428 *s++;
429 }
430 }
431
432 void Stream::Put(int c, int count) {
433
434 while(count) {
435 int n = min(count, (int)(intptr_t)(wrlim - ptr));
436 if(n > 0) {
437 memset(ptr, c, n);
438 ptr += n;
439 count -= n;
440 }
441 else {
442 Put(c);
443 count--;
F11
Microsoft Visual Studio
Ausnahmefehler bei 0x0
Zugriffsverletzung beim
Falls ein Handler für dies
möglicherweise weiterhi
Bei Auslösen dieses Aus
Anhalten und Ausnahmee
100 %
Überwachen 1
Name
Wert
*s
10 '\n'
Auto Lokal Überwachen 1

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by mirek on Fri, 22 Jul 2016 08:15:48 GMT
View Forum Message <> Reply to Message

Do you call Cout from threads?

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by NilaT on Fri, 22 Jul 2016 08:23:53 GMT
View Forum Message <> Reply to Message

In this particular example, no.
My code starts as following:

```
CONSOLE_APP_MAIN
{
    String cmd = "";
    const Vector<String>& argc = CommandLine();
    if (argc.GetCount()>0)
        cmd = argc[0];

    bool progCheck = true;
    for (int c=0;c<argc.GetCount();c++)
    {
     //if(_stricmp(argv[c], "noProgCheck") == 0)
     if(argc[c] == "noProgCheck")
  { progCheck = false;
   break;
  }
    }

    Cout() << "\n---------------------------------\n";
    Cout() << PROG_DESCRIPTION;
    Cout() << "   ";
    Cout() << PROG_COPYRIGHT;
    Cout() << "\n";
    Cout() << "Rel.: ";
    Cout() << PROG_VERSION;
    Cout() << " / ";
    Cout() << DATABASE_VERSION;
    Cout() << " ";
    Cout() << cmd;
    Cout() << "\n---------------------------------\n";
```

and crashes at the very first Cout().
We use sockets in this program as well, that's why we need MT.

It seems to be a compiler problem, as it happens with 2014 sources and 2015 sources.

The funny thing is, that my little testprogram worked...
And when we compile our program with older upp (same sources though), and older compiler (VS2012) it works fine.

Do you know any compiler flag we should try??

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by mirek on Fri, 22 Jul 2016 09:18:01 GMT
View Forum Message <> Reply to Message

Looking at screenshot, I have to say I am a bit confused what is "sources" and what is "compiler"

Looks like you are using in Visual Studio to compile?

Mirek

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by NilaT on Fri, 22 Jul 2016 09:51:41 GMT
View Forum Message <> Reply to Message

Not to compile but to debug, because VS has more capabilities in debugging release/optimal versions.
May I can call you and you can do a TV session?
Because I don't know what to do anymore.
Things are very strange... Because our project works fine with 2012 compiler and 2014 sources.
But won't work with 2014 sources and 2015 compiler.
And it won't work with 2015 sources and 2015 compiler either.
When I do a little testprog, it works with 2015 sources, but won't with 2014. Compiler is 2015.
I think it's better if you actually see what I do...

Sources = Uppsrc
Compiler = MSC15

//edit: I did get another error recently, but only once and I didn't made a screenshot, sorry. But the callstack said something like: Put > printf (is put really calling printf?!) > lambda ....
So, could it be some sort of C++11 issue?

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by mirek on Fri, 22 Jul 2016 13:57:49 GMT
View Forum Message <> Reply to Message

It really looks like Cout is not properly initialized...

Mirek

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by mirek on Fri, 22 Jul 2016 14:00:19 GMT
View Forum Message <> Reply to Message

Try to replace

```
Stream& Cout()
{
 return Single<CoutStream>();
}
```

with

```
Stream& Cout()
{
   static CoutStream x;
   return x;
}
```

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by NilaT on Mon, 25 Jul 2016 07:13:25 GMT
View Forum Message <> Reply to Message

Hi mirek,

thanks, works like a charm now.

Could you explain what happened?
Must be a ONCELOCK issue?!

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by mirek on Mon, 25 Jul 2016 13:50:50 GMT
View Forum Message <> Reply to Message

Well, I am afraid that this is in fact very bad news....

It looks like MSC15 sometimes has troubles with static inside template (as in Single) I have already experienced something somewhat similar. It looks like only some installations of MSC15 are affected though, compile on different computer and it works.

Mirek

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by NilaT on Tue, 26 Jul 2016 08:33:16 GMT
View Forum Message <> Reply to Message

Well, it seems like there are more and more problems coming up with MSC15, this time with 2010 sources (in another project).
A simple Format("%s %d", "bla", 1); results in a crash in optimal mode, debug works fine though.
Exception: C0000005 at 150C170
EXCEPTION_ACCESS_VIOLATION
reading at 0000001C

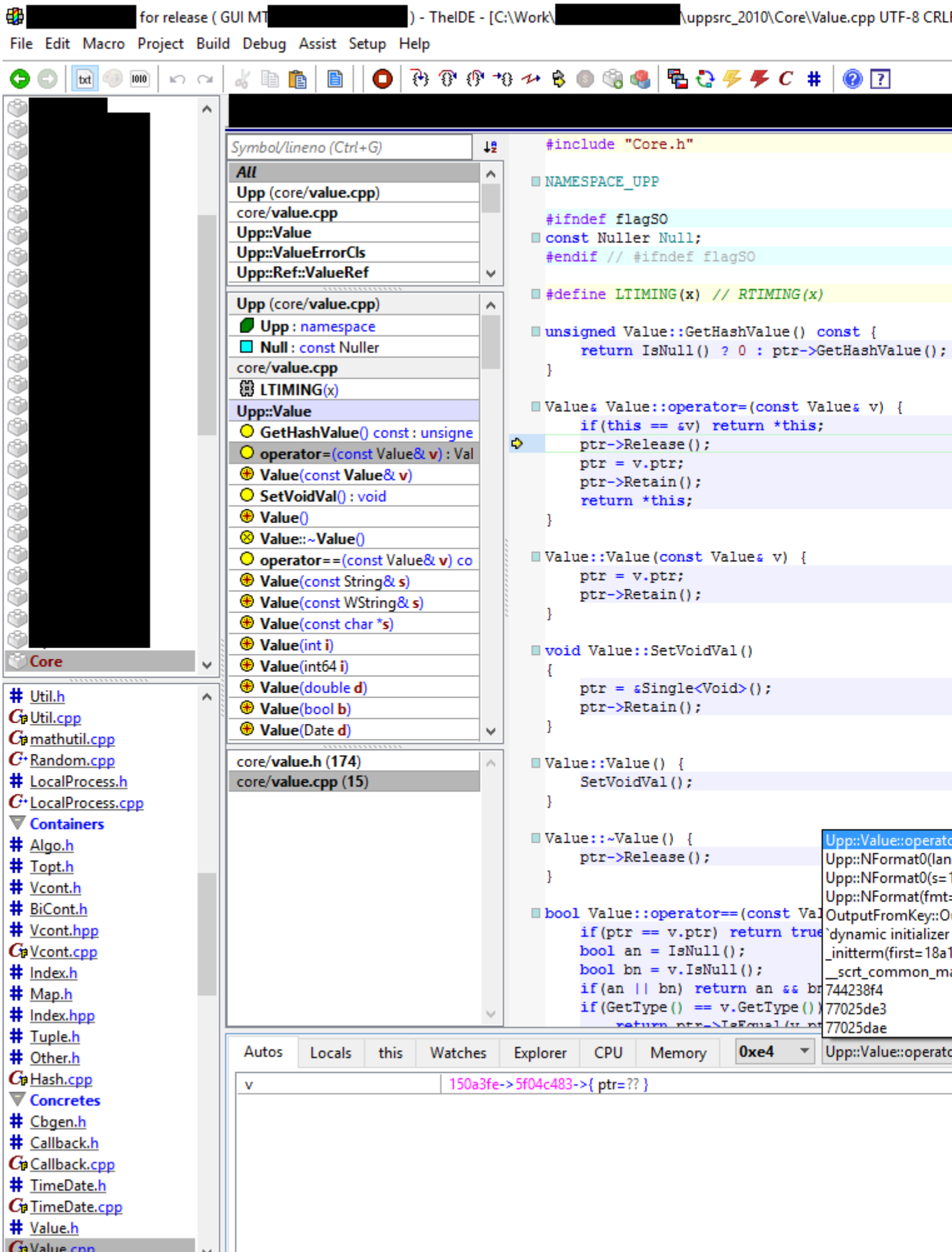
Callstack is in the picture.
Thanks for help.

//edit: Same fix seems to work. I changed Single to:

```
template <class T>
T& Single() {
 #ifndef flagMSC15
 static T *p;
 ONCELOCK {
  static T o;
  p = &o;
 }
 return *p;
 #else
 static T x;
 return x;
 #endif
}
```

File Attachments

1) crash.png, downloaded 657 times


[redacted] for release (GUI MT[redacted]) - TheIDE - [C:\Work\[redacted]\uppsrc_2010\Core\Value.cpp UTF-8 CRLF
File Edit Macro Project Build Debug Assist Setup Help
Symbol/lineno (Ctrl+G)
All
Upp (core/value.cpp)
core/value.cpp
Upp::Value
Upp::ValueErrorCls
Upp::Ref::ValueRef
Upp (core/value.cpp)
Upp : namespace
Null : const Nuller
core/value.cpp
LTIMING(x)
Upp::Value
GetHashValue() const : unsigne
operator=(const Value& v) : Val
Value(const Value& v)
SetVoidVal() : void
Value()
Value::~Value()
operator==(const Value& v) co
Value(const String& s)
Value(const WString& s)
Value(const char *s)
Value(int i)
Value(int64 i)
Value(double d)
Value(bool b)
Value(Date d)
core/value.h (174)
core/value.cpp (15)
#include "Core.h"
NAMESPACE_UPP
#ifndef flagSO
const Nuller Null;
#endif // #ifndef flagSO
#define LTIMING(x) // RTIMING(x)
unsigned Value::GetHashValue() const {
return IsNull() ? 0 : ptr->GetHashValue();
}
Value& Value::operator=(const Value& v) {
if(this == &v) return *this;
ptr->Release();
ptr = v.ptr;
ptr->Retain();
return *this;
}
Value::Value(const Value& v) {
ptr = v.ptr;
ptr->Retain();
}
void Value::SetVoidVal()
{
ptr = &Single<Void>();
ptr->Retain();
}
Value::Value() {
SetVoidVal();
}
Value::~Value() {
ptr->Release();
}
bool Value::operator==(const Val
if(ptr == v.ptr) return true
bool an = IsNull();
bool bn = v.IsNull();
if(an || bn) return an && bn
if(GetType() == v.GetType())
Upp::Value::operato
Upp::NFormat0(lan
Upp::NFormat0(s=1
Upp::NFormat(fmt=
OutputFromKey::O
`dynamic initializer
_initterm(first=18a1
__scrt_common_ma
744238f4
77025de3
77025dae
Autos Locals this Watches Explorer CPU Memory
0xe4
Upp::Value::operato
v
150a3fe->5f04c483->{ ptr=?? }
Core
Util.h
Util.cpp
mathutil.cpp
Random.cpp
LocalProcess.h
LocalProcess.cpp
Containers
Algo.h
Topt.h
Vcont.h
BiCont.h
Vcont.hpp
Vcont.cpp
Index.h
Map.h
Index.hpp
Tuple.h
Other.h
Hash.cpp
Concretes
Cbgen.h
Callback.h
Callback.cpp
TimeDate.h
TimeDate.cpp
Value.h
Value.cpp

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by mirek on Wed, 27 Jul 2016 04:29:22 GMT
View Forum Message <> Reply to Message

Actually, that sounds like a good news for a change. For C++11, this is a correct implementation. The whole point of Single was that before C++11, this code was not MT safe.

Mirek

Subject: Re: MSC15 and 2014 sources - problem with Stream
Posted by NilaT on Wed, 27 Jul 2016 06:38:23 GMT
View Forum Message <> Reply to Message

Well, all projects seem to work for now, maybe we face some detailed errors when we test specific packages/functions, but a first test was fine.

Thanks for the help!