Subject: New Core
Posted by mirek on Tue, 03 May 2016 14:17:23 GMT
View Forum Message <> Reply to Message

This week I am in the process to finally switching to "new" Core.

Right now, it is already in trunk and most things are fixed.

New Core is "C++11 only". Details of what is new/changed will be published later.

Anyway, as there is not a 100% compatibility maintained, the most important information is that the 'legacy' U++ will be available at svn://ultimatepp.org/upp/classic and perhaps maintained for some time to go...

Mirek

Subject: Re: New Core
Posted by koldo on Wed, 04 May 2016 07:38:57 GMT
View Forum Message <> Reply to Message

Hello Mirek

What will be the improvements and where do you expect the compatibility problems may appear?

Subject: Re: New Core
Posted by cbpporter on Wed, 04 May 2016 12:55:05 GMT
View Forum Message <> Reply to Message

And what kind of changes are these? Superficial or deeper? Did you manage to clean up the weird more legacy stuff or is it just an update of containers for C++1x?

And speaking of C++14, this version supports 0b1111 like constants. Just today I needed them in CodeEditor. I will submit a patch for it if you think we should support it? Do you?

And if yes, the question is if you want me to add a:

HL_COLOR(INK_CONST_BIN, t_("Binary constant"), 1)

to the highlighting infrastructure or will we reuse some existing color?

Subject: Re: New Core

Posted by mirek on Thu, 05 May 2016 07:26:38 GMT
View Forum Message <> Reply to Message

koldo wrote on Wed, 04 May 2016 09:38Hello Mirek

What will be the improvements and where do you expect the compatibility problems may appear?

I will detail changes soon (but mostly it is about C++11 and multithreading).

So far, apart for one or two minor changes, suprisingly the most compatibility problems are caused by new Core detecting subtle bugs.

It took me 5 minutes to fix those is my largest app.

Mirek

Subject: Re: New Core
Posted by cbpporter on Fri, 06 May 2016 08:24:42 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 05 May 2016 10:26koldo wrote on Wed, 04 May 2016 09:38Hello Mirek

What will be the improvements and where do you expect the compatibility problems may appear?

I will detail changes soon (but mostly it is about C++11 and multithreading).

So far, apart for one or two minor changes, suprisingly the most compatibility problems are caused by new Core detecting subtle bugs.

It took me 5 minutes to fix those is my largest app.

Mirek
Ah OK!

I shall wait then!

Honestly, I was hoping for a more substantial update to Core. Most classes are fine as they are, even great, but all the free functions are falling out of favor and considered sub-optimal design nowadays. As an example, all the file name related stuff could be grouped under a class called Path with static members and so on.

I help with remembering where everything goes and what names it has. I often know that there is a free function that solves my problem, but I'm not sure how it is called.

Today I've spent like 5 minutes trying to find the function that copies text to clipboard, only to find that in CtrlCore.h there are things like ClearClipboard and WriteClipboardUnicodeText. Things like this should really be Clipboard::Clear and Clipboard::Write.

Subject: Re: New Core
Posted by mirek on Fri, 06 May 2016 11:10:55 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Fri, 06 May 2016 10:24
Today I've spent like 5 minutes trying to find the function that copies text to clipboard, only to find that in CtrlCore.h there are things like ClearClipboard and WriteClipboardUnicodeText. Things like this should really be Clipboard::Clear and Clipboard::Write.
[/quote]

Trouble with this is that Clipboard class is 'closed'. You can put some formats to it, but it is not extensible in another module.

Mirek

Subject: Re: New Core
Posted by dolik.rce on Fri, 06 May 2016 20:53:30 GMT
View Forum Message <> Reply to Message

Hi Mirek,

I have updated the packaging to use C++11, but TheIDE build still fail. The error says DeepClone is not declared, you can see it in the build logs, e.g. for Debian Wheezy. Could you have a look at it, please?

Honza

Subject: Re: New Core
Posted by mirek on Sun, 08 May 2016 04:31:23 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Fri, 06 May 2016 22:53Hi Mirek,

I have updated the packaging to use C++11, but TheIDE build still fail. The error says DeepClone is not declared, you can see it in the build logs, e.g. for Debian Wheezy. Could you have a look at it, please?

Honza

Most likely it is missing MT in main package config.

Subject: Re: New Core
Posted by dolik.rce on Sun, 08 May 2016 13:37:31 GMT
View Forum Message <> Reply to Message

mirek wrote on Sun, 08 May 2016 06:31Most likely it is missing MT in main package config.
Oh, right, I missed that it is necessary now.

So the situation is much better now. There is just few problems left:

Ubuntu 12.4 and ScientificLinux 6 both fail, because their compiler (GCC 4.6 and 4.7, respectively) is so old it doesn't even support the option -std=c++11. Also the compiler in Debian Wheezy (GCC 4.7) doesn't work. This one already knows c++11, but thread_local keyword is not implemented, so the compilation fails.

And the last problem is in ArchLinux packages where the compilation fails on conflicting declaration of abs(int64), which is already present in stdlib in C++11.

Is it safe to drop support for the old distros, even though they are still supported? Ubuntu 12.4 has end-of-life scheduled for April 2017, Debian Wheezy has LTS until May 2018 and ScientificLinux 6 even has full support until Q2 2017 (and then maintenance updates until November 2020).

Honza

Subject: Re: New Core
Posted by Zbych on Sun, 08 May 2016 16:08:03 GMT
View Forum Message <> Reply to Message

I tried to compile last version (upp-x11-src-9809.tar.gz) from source but it failed with lack of std=c++11.
Where it should be added?
src/Makefile?
src/Makefile.in?
src/UMakefile?

Subject: Re: New Core
Posted by mirek on Mon, 09 May 2016 09:23:08 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sun, 08 May 2016 15:37mirek wrote on Sun, 08 May 2016 06:31Most likely it is missing MT in main package config.
Oh, right, I missed that it is necessary now.

So the situation is much better now. There is just few problems left:

Ubuntu 12.4 and ScientificLinux 6 both fail, because their compiler (GCC 4.6 and 4.7, respectively) is so old it doesn't even support the option -std=c++11. Also the compiler in Debian Wheezy (GCC 4.7) doesn't work. This one already knows c++11, but thread_local keyword is not

implemented, so the compilation fails.

And the last problem is in ArchLinux packages where the compilation fails on conflicting declaration of abs(int64), which is already present in stdlib in C++11.

Is it safe to drop support for the old distros, even though they are still supported? Ubuntu 12.4 has end-of-life scheduled for April 2017, Debian Wheezy has LTS until May 2018 and ScientificLinux 6 even has full support until Q2 2017 (and then maintenance updates until November 2020).

Honza

Perhaps we can switch it to 'classic'?

Mirek

Subject: Re: New Core
Posted by mirek on Mon, 09 May 2016 09:23:33 GMT
View Forum Message <> Reply to Message

Zbych wrote on Sun, 08 May 2016 18:08I tried to compile last version (upp-x11-src-9809.tar.gz) from source but it failed with lack of std=c++11.
Where it should be added?
src/Makefile?
src/Makefile.in?
src/UMakefile?

Sorry, that is a bug in umk. Fixing it now.

Mirek

Subject: Re: New Core
Posted by dolik.rce on Mon, 09 May 2016 14:32:45 GMT
View Forum Message <> Reply to Message

mirek wrote on Mon, 09 May 2016 11:23dolik.rce wrote on Sun, 08 May 2016 15:37
Is it safe to drop support for the old distros, even though they are still supported? Ubuntu 12.4 has end-of-life scheduled for April 2017, Debian Wheezy has LTS until May 2018 and ScientificLinux 6 even has full support until Q2 2017 (and then maintenance updates until November 2020).

Perhaps we can switch it to 'classic'?

It could be done, if you package the 'classic' uppsrc into the nigthly tarball. I could then try to detect GCC version and set the assembly to classic version if the compiler is too old. It doesn't

even need to be whole uppsrc, only packages that are necessary to build theide and umk.

Honza

Subject: Re: New Core
Posted by mirek on Mon, 09 May 2016 15:25:22 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Mon, 09 May 2016 16:32mirek wrote on Mon, 09 May 2016 11:23dolik.rce wrote on Sun, 08 May 2016 15:37
Is it safe to drop support for the old distros, even though they are still supported? Ubuntu 12.4 has end-of-life scheduled for April 2017, Debian Wheezy has LTS until May 2018 and ScientificLinux 6 even has full support until Q2 2017 (and then maintenance updates until November 2020).

Perhaps we can switch it to 'classic'?

It could be done, if you package the 'classic' uppsrc into the nigthly tarball.

Well, I do not. 'classic' is basically branch for systems not supporting C++11.

I perhaps could generate complete 'classic' tarball.

Mirek

Subject: Re: New Core
Posted by dolik.rce on Mon, 09 May 2016 18:51:05 GMT
View Forum Message <> Reply to Message

mirek wrote on Mon, 09 May 2016 17:25dolik.rce wrote on Mon, 09 May 2016 16:32mirek wrote on Mon, 09 May 2016 11:23dolik.rce wrote on Sun, 08 May 2016 15:37
Is it safe to drop support for the old distros, even though they are still supported? Ubuntu 12.4 has end-of-life scheduled for April 2017, Debian Wheezy has LTS until May 2018 and ScientificLinux 6 even has full support until Q2 2017 (and then maintenance updates until November 2020).

Perhaps we can switch it to 'classic'?

It could be done, if you package the 'classic' uppsrc into the nigthly tarball.

Well, I do not. 'classic' is basically branch for systems not supporting C++11.

I perhaps could generate complete 'classic' tarball.
I understand that it is a branch, but I think it could still be present in the tarball as a "fallback" for older systems. It would also be simpler for the users, because having two tarballs requires

explaining them in great detail which one they should use...

Also it would save me a lot of work when packaging, because I wouldn't have to keep track of compiler version for each of the 15 distros and manually choose which one should be used. It would be especially troublesome for ubuntu and debian, where all 7 versions is currently built from the same package.

Honza

Subject: Re: New Core
Posted by mr_ped on Mon, 09 May 2016 22:31:41 GMT
View Forum Message <> Reply to Message

Maybe the package should be left as is, and upp2 can be introduced? (like kde4 vs kde5 during transition period)
Or that's too complicated to maintain either?

Subject: Re: New Core
Posted by Mindtraveller on Tue, 10 May 2016 16:46:55 GMT
View Forum Message <> Reply to Message

I'm afraid Web package is not compiled successfully with new Core. Compiling with MSC 14/15 fails on a number of files from Web package.

Subject: Re: New Core
Posted by mirek on Wed, 11 May 2016 08:33:46 GMT
View Forum Message <> Reply to Message

Web was planned to be removed. It was DEPRECATED years ago.

Is there anything in Web that you need? I guess I can still fix it...

Subject: Re: New Core
Posted by mirek on Wed, 11 May 2016 08:34:40 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Tue, 10 May 2016 00:31Maybe the package should be left as is, and upp2 can be introduced? (like kde4 vs kde5 during transition period)
Or that's too complicated to maintain either?

Is not that exactly what happened? 'classic' being left as it is and upp moved on to C++11?

Mirek

Subject: Re: New Core
Posted by mirek on Wed, 11 May 2016 08:40:30 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Mon, 09 May 2016 20:51mirek wrote on Mon, 09 May 2016 17:25dolik.rce wrote on Mon, 09 May 2016 16:32mirek wrote on Mon, 09 May 2016 11:23dolik.rce wrote on Sun, 08 May 2016 15:37
Is it safe to drop support for the old distros, even though they are still supported? Ubuntu 12.4 has end-of-life scheduled for April 2017, Debian Wheezy has LTS until May 2018 and ScientificLinux 6 even has full support until Q2 2017 (and then maintenance updates until November 2020).

Perhaps we can switch it to 'classic'?

It could be done, if you package the 'classic' uppsrc into the nigthly tarball.

Well, I do not. 'classic' is basically branch for systems not supporting C++11.

I perhaps could generate complete 'classic' tarball.
I understand that it is a branch, but I think it could still be present in the tarball as a "fallback" for older systems. It would also be simpler for the users, because having two tarballs requires explaining them in great detail which one they should use...
Honza

Not sure. For starters, it will complicate the tarball. Either I will have to include both source trees, or invent some patching.

I really would like to keep 'classic' as separate thing. Like gtk2-gtk3, KDE4/5 etc.. Those are not coming in single tarball, right?

Subject: Re: New Core
Posted by mirek on Wed, 11 May 2016 08:54:47 GMT
View Forum Message <> Reply to Message

koldo wrote on Wed, 04 May 2016 09:38Hello Mirek

What will be the improvements and where do you expect the compatibility problems may appear?

Here is the brief summary:

http://www.ultimatepp.org/srcdoc$Core$Core2016$en-us.html

Subject: Re: New Core
Posted by Mindtraveller on Wed, 11 May 2016 13:30:21 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 11 May 2016 11:33Web was planned to be removed. It was DEPRECATED years ago.

Is there anything in Web that you need? I guess I can still fix it...

Oh, sorry I wasn't aware of deprecated status of Web package.
After looking into code, it looks like HttpQuery is the only dependant class from Web package I still use.

Subject: Re: New Core
Posted by cbpporter on Wed, 11 May 2016 13:32:40 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Wed, 11 May 2016 16:30mirek wrote on Wed, 11 May 2016 11:33Web was planned to be removed. It was DEPRECATED years ago.

Is there anything in Web that you need? I guess I can still fix it...

Oh, sorry I wasn't aware of deprecated status of Web package.
After looking into code, it looks like HttpQuery is the only dependant class from Web package I still use.
Really? I had no idea .

What are we supposed to use instead?

Subject: Re: New Core
Posted by mirek on Wed, 11 May 2016 16:40:59 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Wed, 11 May 2016 15:32Mindtraveller wrote on Wed, 11 May 2016 16:30mirek wrote on Wed, 11 May 2016 11:33Web was planned to be removed. It was DEPRECATED years ago.

Is there anything in Web that you need? I guess I can still fix it...

Oh, sorry I wasn't aware of deprecated status of Web package.
After looking into code, it looks like HttpQuery is the only dependant class from Web package I still use.
Really? I had no idea .

What are we supposed to use instead?

For years, there is a superior alternative in Core, HttpRequest. It can handle much more of http and is (if needed) async.

E.g.

http://www.codeproject.com/Articles/1068378/Simple-multi-req uest-web-crawler-using-Uplusplus

Mirek

Subject: Re: New Core
Posted by cbpporter on Thu, 12 May 2016 01:04:26 GMT
View Forum Message <> Reply to Message

Which is the last "old core" version. For bookmarking .

Subject: Re: New Core
Posted by Novo on Thu, 12 May 2016 02:45:44 GMT
View Forum Message <> Reply to Message

I'm still getting crashes with the new core and mingw x64 (tdm-gcc 5.1). My app is crashing in Optimal and Size configurations. It is crashing with and without SSE2 flag. BLITZ is disabled, precompiled headers are enabled. x86 configuration seems to be fine.

Interestingly, cc1plus.exe is crashing when I'm trying to compile Optimal with debug info

Update: I've checked mingw 5.3.0 x64 Optimal with debug info. I can get a call stack. It is crashing in Upp::Rect_<int>::Rect_, which is inside of Upp::Ctrl::CtrlPaint.

Thanks.

Subject: Re: New Core
Posted by dolik.rce on Thu, 12 May 2016 04:42:31 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 11 May 2016 10:40dolik.rce wrote on Mon, 09 May 2016 20:51mirek wrote on Mon, 09 May 2016 17:25I perhaps could generate complete 'classic' tarball.
I understand that it is a branch, but I think it could still be present in the tarball as a "fallback" for older systems. It would also be simpler for the users, because having two tarballs requires explaining them in great detail which one they should use...
Honza

Not sure. For starters, it will complicate the tarball. Either I will have to include both source trees, or invent some patching.

I really would like to keep 'classic' as separate thing. Like gtk2-gtk3, KDE4/5 etc.. Those are not coming in single tarball, right?
Ok, it'll be more complicated, but I guess I can make it work somehow with two distinct tarballs. Please let me know when the classic archive is ready.

BTW: What about the issue with Arch Linux, did you look at it? I'm totally swamped lately, so I didn't have time to investigate it at all

Honza

Subject: Re: New Core
Posted by mirek on Thu, 12 May 2016 07:09:38 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 12 May 2016 04:45I'm still getting crashes with the new core and mingw x64 (tdm-gcc 5.1). My app is crashing in Optimal and Size configurations. It is crashing with and without SSE2 flag. BLITZ is disabled, precompiled headers are enabled. x86 configuration seems to be fine.

Interestingly, cc1plus.exe is crashing when I'm trying to compile Optimal with debug info

Update: I've checked mingw 5.3.0 x64 Optimal with debug info. I can get a call stack. It is crashing in Upp::Rect_<int>::Rect_, which is inside of Upp::Ctrl::CtrlPaint.

Thanks.

Weird. That is the place it was always crashing, before I have fixed it with 'finetuning' -O options.

Are you using "Instant setup" default options?

Mirek

Subject: Re: New Core
Posted by Mindtraveller on Thu, 12 May 2016 13:10:12 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 11 May 2016 19:40For years, there is a superior alternative in Core, HttpRequest. It can handle much more of http and is (if needed) async.
Yes, I was aware of the HttpRequest alternative. But HttpQuery represented slightly different functionality. It gave a number of useful routines for URL parsing. I used it in http server code to parse URL variables of incoming http request.

```
  while (!shutdown)
  {
   if (server->IsError())
```

```
  {
   server.Clear();
   server.Create().Listen(answerPort,100);
   Sleep(200);
  }
  TcpSocket socket;
  socket.Timeout(2000);
  if (!socket.Accept(*server))
  {
   Sleep(50);
   continue;
  }

  HttpHeader http;
  http.Read(socket);
  int len = static_cast<int>(http.GetContentLength());
  if (len > 0)
   socket.GetAll(len);

  HttpQuery query(http.GetURI());

  String rq = query.GetString("RQ"); // this is where HttpQuery is actually used

  if (rq == "REQ1")
  {
  }
  else if (rq == "REQ2")
  {
  }
  else if (rq == "REQ3")
  {
  }
...
```

Subject: Re: New Core
Posted by Novo on Thu, 12 May 2016 14:23:57 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 12 May 2016 03:09
Weird. That is the place it was always crashing, before I have fixed it with 'finetuning' -O options.

Are you using "Instant setup" default options?

Mirek

No, I was using old -O3 and -Os options.

Replacing -Os with -O2 fixed crashes with Optimal and Size configuration.
I didn't change -O3 to -O2 for the Speed configuration.
I do not understand where you are taking options for the Optimal configuration from. They are not declared explicitly anywhere.
Another interesting thing. A quote from documentation: "-Os: optimizes code for size. It activates all -O2 options that do not increase the size of the generated code. It can be useful for machines that have extremely limited disk storage space and/or CPUs with small cache sizes."
So, basically, -Os is -O2 with some limitations.

Thanks.

Subject: Re: New Core
Posted by Novo on Thu, 12 May 2016 15:08:56 GMT
View Forum Message <> Reply to Message

Another thing.

Could you please convert all Windows-related include file names to lower case? They are all lower case with mingw on Linux.
It looks like this is the right time to do that.

A patch file created by git looks weird, but still ...
You need to use WIN32 flag with mingw on Linux, and TheIDE won't handle rc-files.

Update: if I use WIN32 flag along with GUI and MT, then I get a problem with rc-files. If I just add a -DflagWIN32 common option, then everything compiles, but TheIDE is still linking against Linux libs ...

I guess I'm missing something.

TIA

File Attachments
1) mingw.patch, downloaded 383 times

Subject: Re: New Core
Posted by cbpporter on Thu, 12 May 2016 17:29:09 GMT
View Forum Message <> Reply to Message

And a question: I can fully understand the 16 bit alignment for allocation and I've used a lot in the past SSE, so this is a good move.

But why make the smallest allocation size 32, not 16?

Subject: Re: New Core
Posted by cbpporter on Fri, 13 May 2016 14:48:45 GMT
View Forum Message <> Reply to Message

I've been slightly reluctant with the changes since I'm suing an older compiler.

And while I'm not sure I'll switch over to the new core, I'm done with the old compiler. In some micro benchmarks with VS2010, my code runs 4 times slower than with MINGW with 100000 iterations.

So I'm done with VS2010 .

Subject: Re: New Core
Posted by cbpporter on Fri, 13 May 2016 15:05:23 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Fri, 13 May 2016 17:48I've been slightly reluctant with the changes since I'm suing an older compiler.

And while I'm not sure I'll switch over to the new core, I'm done with the old compiler. In some micro benchmarks with VS2010, my code runs 4 times slower than with MINGW with 100000 iterations.

So I'm done with VS2010 .
Well, MSC11 is not any better than 10.

I'll try the latest MSC on Monday.

But can actually MINGW or TDM or whatever is called be actually so good at optimizing my code or is there something else going on?

Subject: Re: New Core
Posted by mirek on Sat, 14 May 2016 05:52:57 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Thu, 12 May 2016 19:29And a question: I can fully understand the 16 bit alignment for allocation and I've used a lot in the past SSE, so this is a good move.

But why make the smallest allocation size 32, not 16?

It is related to the implementation of allocator.

In short: There are small blocks and there are large blocks. At some point, I need to know if freed block is small or large. Fast method to do that is to use single bit in address - small blocks are always 32 bytes aligned, large blocks are 32 bytes misaligned. Therefore small block has to be multiple of 32.

Previous core used the same process, but 16 byte misalignment. So the smallest block was 16 bytes and alignment 8 bytes.

Of course, I have tested this (with theide and other apps) and the increase in memory consumption is negligible.

Subject: Re: New Core
Posted by mirek on Sat, 14 May 2016 05:59:16 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 12 May 2016 16:23mirek wrote on Thu, 12 May 2016 03:09
Weird. That is the place it was always crashing, before I have fixed it with 'finetuning' -O options.

Are you using "Instant setup" default options?

Mirek

No, I was using old -O3 and -Os options.
Replacing -Os with -O2 fixed crashes with Optimal and Size configuration.
I didn't change -O3 to -O2 for the Speed configuration.
I do not understand where you are taking options for the Optimal configuration from. They are not declared explicitly anywhere.

I do not quite understand the question. I was just combining until it worked. Then adjusted 'instant setup' to found options.

BTW, my hypothesis here is that it is a bug linker that makes it impossible to combine size and speed optimized code (after all, there are no issues in Posix). Perhaps something related to inlines (-O3 is more aggresive in inlining everything). SO whatever, all is now -O2.

Mirek

Subject: Re: New Core
Posted by mirek on Sat, 14 May 2016 06:00:19 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Thu, 12 May 2016 06:42
BTW: What about the issue with Arch Linux, did you look at it? I'm totally swamped lately, so I didn't have time to investigate it at all

Honza

Hopefully fixed (I have just removed our abs definitions, which were there to support old compilers...)

Subject: Re: New Core
Posted by mirek on Sat, 14 May 2016 06:56:52 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 12 May 2016 17:08
Update: if I use WIN32 flag along with GUI and MT, then I get a problem with rc-files. If I just add a -DflagWIN32 common option, then everything compiles, but TheIDE is still linking against Linux libs ...

You would need "cross-compling" ide... (does not exist at the moment).

Why not just use wine and current nightly build? Works like charm. (Actually, nightly build is built this way - wine mingw).

Mirek

Subject: Re: New Core
Posted by mirek on Sat, 14 May 2016 07:03:32 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Thu, 12 May 2016 15:10mirek wrote on Wed, 11 May 2016 19:40For years, there is a superior alternative in Core, HttpRequest. It can handle much more of http and is (if needed) async.
Yes, I was aware of the HttpRequest alternative. But HttpQuery represented slightly different functionality. It gave a number of useful routines for URL parsing. I used it in http server code to parse URL variables of incoming http request.

```
 while (!shutdown)
 {
  if (server->IsError())
  {
   server.Clear();
   server.Create().Listen(answerPort,100);
   Sleep(200);
  }
  TcpSocket socket;
  socket.Timeout(2000);
  if (!socket.Accept(*server))
```

```
   {
    Sleep(50);
    continue;
   }

   HttpHeader http;
   http.Read(socket);
   int len = static_cast<int>(http.GetContentLength());
   if (len > 0)
    socket.GetAll(len);

   HttpQuery query(http.GetURI());

   String rq = query.GetString("RQ"); // this is where HttpQuery is actually used

   if (rq == "REQ1")
   {
   }
   else if (rq == "REQ2")
   {
   }
   else if (rq == "REQ3")
   {
   }
...
```

I see.

Well, looks like we need to move this piece of code from Skylark to Core, right?

```
void Http::ParseRequest(const char *p)
{
 while(*p) {
  const char *last = p;
  while(*p && *p != '=' && *p != '&')
   p++;
  String key = UrlDecode(last, p);
  if(*p == '=')
   p++;
  last = p;
  while(*p && *p != '&')
   p++;
  if(*key != '.' && *key != '@') {
   if(key.EndsWith("[]")) {
    Value &v = var.GetAdd(key);
    if(v.IsNull())
```

```
      v = ValueArray();
     (ValueArray &)v << UrlDecode(last, p);
    }
    else
     var.GetAdd(key) = UrlDecode(last, p);
   }
   if(*p)
    p++;
  }
}
```

Subject: Re: New Core
Posted by Mindtraveller on Sat, 14 May 2016 21:57:46 GMT
View Forum Message <> Reply to Message

mirek wrote on Sat, 14 May 2016 10:03
I see.

Well, looks like we need to move this piece of code from Skylark to Core, right?

Yes. It will be handy little addition.

Subject: Re: New Core
Posted by mirek on Sun, 15 May 2016 18:40:05 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Sat, 14 May 2016 23:57mirek wrote on Sat, 14 May 2016 10:03
I see.

Well, looks like we need to move this piece of code from Skylark to Core, right?

Yes. It will be handy little addition.

So this is now in Core:

UrlInfo
f("http://username:password@hostname:9090/path?arg=value&aarg[]=item1&aarg[]=item2#anchor");

f.url =

http://username:password@hostname:9090/path?arg=value&aarg[]=item1&aarg[]=item2#anchor
f.scheme = http
f.host = hostname
f.port = 9090
f.user = username
f.password = password
f.path = /path
f.query = arg=value&aarg[]=item1&aarg[]=item2
f.fragment = anchor
f.parameters = {arg: value}
f.array_parameters = {aarg: [item1, item2]}

Subject: Re: New Core
Posted by cbpporter on Tue, 17 May 2016 09:18:45 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Fri, 13 May 2016 18:05cbpporter wrote on Fri, 13 May 2016 17:48I've been slightly reluctant with the changes since I'm suing an older compiler.

And while I'm not sure I'll switch over to the new core, I'm done with the old compiler. In some micro benchmarks with VS2010, my code runs 4 times slower than with MINGW with 100000 iterations.

So I'm done with VS2010 .
Well, MSC11 is not any better than 10.

I'll try the latest MSC on Monday.

But can actually MINGW or TDM or whatever is called be actually so good at optimizing my code or is there something else going on?

Quote:
But can actually MINGW or TDM or whatever is called be actually so good at optimizing my code or is there something else going on?

Nope, of course not. While I have full respect for GCC and it's family, I am yet to see a it have a 2 to 4 times better performance in the generated code than it's competitors.

Changing compilers around and having the same bad performance led me to disregard what is changing as a non-contributing factor and focus my attention on what was constant.

And I was right: it was TheIDE. The settings for "Optimal" cause the performance degrade. Switching to speed makes the performance be roughly equal to TDM with Optimal. But TDM with

Speed is even faster.

So what is up with "Optimal". I can see that it uses /O1. That is the option for "optimized for speed" from MSC. MSC recommends /O2 for release builds. Why does Optimal do /O1 and what is the difference then between Optimal and Size.

And more importantly: why was I compiling with "Optimal" for years now?

Good thing that I caught it. I was just about to write a blog post detailing the horrible performance of MSC10 and singing the praises of GCC. I would have made a fool of myself publicly when no one else could have gotten the same results in the benchmarks .

Subject: Re: New Core
Posted by mirek on Wed, 18 May 2016 12:25:02 GMT
View Forum Message <> Reply to Message

I guess you misunderstand Optimal mode.

-O1 is "optimize for size"

Optimal mode means "optimize for size unless told otherwise". You can "tell otherwise" by activating "optimize for speed" in either package or file settings.

The motivation here was that majority of code does not have impact on speed. So you can just mark some files or packages to be speed optimized and save size elsewhere. You can e.g. look at Core and see which files are speed optimized (they have little "F" on them, as "FAST").

That said, maybe whole that thing is over-engineered. It dates some 15 years back when it was still important to fit something on 3.5inch diskettes (yes, U++ remembers such times) and having .exe 1.2MB instead of 1.7MB was still a big deal.

Subject: Re: New Core
Posted by mirek on Wed, 18 May 2016 12:36:30 GMT
View Forum Message <> Reply to Message

P.S.: Funny fact, theide 64 optimal size is 15920KB, speed is 17354KB. I guess I will drop this confusion soon...

Subject: Re: New Core
Posted by cbpporter on Wed, 18 May 2016 13:24:51 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 18 May 2016 15:36P.S.: Funny fact, theide 64 optimal size is 15920KB, speed is 17354KB. I guess I will drop this confusion soon...
I tested it too and it is 400 KiB on my 32 bit exe.

But a round trip (and not necessarily round trip, works one way too) conversion from Int to String is about 3 time "slower" on size for 100 million iterations than on "speed". And this with a bunch of hacks I added to speed it up. Without, it is even slower. So I'm sticking with -O2 from now on. I would have never used anything else but TheIDE made this choice for me.

It would be great if at least this would be less confusing.

Optimal means optimal. It means that either it is everybody else's optimal, i.e. Microsoft's or the open source community's recommendation or your personal best pick if you think you know better. Nobody understand though optimal -O1! -O1 on GCC means optimized, but do cheap as compile time optimizations and on MSC means optimize if it doesn't increase code size.

Subject: Re: New Core
Posted by mirek on Wed, 18 May 2016 18:44:14 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Wed, 18 May 2016 15:24mirek wrote on Wed, 18 May 2016 15:36P.S.: Funny fact, theide 64 optimal size is 15920KB, speed is 17354KB. I guess I will drop this confusion soon...
I tested it too and it is 400 KiB on my 32 bit exe.

But a round trip (and not necessarily round trip, works one way too) conversion from Int to String is about 3 time "slower" on size for 100 million iterations than on "speed". And this with a bunch of hacks I added to speed it up. Without, it is even slower. So I'm sticking with -O2 from now on.

This absolutely expected.

Quote:
I would have never used anything else but TheIDE made this choice for me.

Have you told it that your package/file needs to be speed optimized?

Quote:
Optimal means optimal.

Exactly. Optimal .exe is as small as possible and as fast as possible. By optimizing for speed only what is necessary, you can achieve that goal.

Anyway, while this is sound idea, as I said 10% in size is not worth the trouble. So from the next

release on, it will be just speed.

Mirek

Subject: Re: New Core
Posted by cbpporter on Wed, 18 May 2016 19:53:21 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Thu, 12 May 2016 04:04Which is the last "old core" version. For bookmarking . And not just for bookmarking. For bug-fixing and submitting test-cases. I'm a bit stuck. I need to patch both my code and submit one for TheIDE to detect better older compiler versions. I know that it is legacy mode and no longer supported, but there is still autodecting code, which doesn't work as good as it should. So either remove it or fix it. I really need to take my ugly code and write a package that autodetects really well the compiler versions and add it to bazaar and as a dependency in my code.

Then I'm finding bugs left and right, some probably just because I'm doing something wrong, but at least one is in U++, but I can't compile yet with the new core.

Then I'm stuck since my command line project is not supposed to support C++11, but it is also used as a library by the GUI, which doesn't care about C++ version. So if I update, I need to make my code old school C++ and use U++ Core in a way that compiles at least on both.

mirek wrote on Wed, 18 May 2016 21:44
Anyway, while this is sound idea, as I said 10% in size is not worth the trouble. So from the next release on, it will be just speed.

Speed and size is fine.

Actually, I'm having this problem too. I need to support MSC, several versions, TDM, Clang overrides for them, Linux G++ and Linux Clang.

And they hate to agree upon what option means what . So I'm thinking of only supporting two options, Debug and Release, and to have compiler dependent options as overrides. For MSC: O1, O2, Ox and Od. For G++: O0-O3 and Os. So I'm thinking of adding these options to the GUI based on compiler and no even trying to give them meaningful names. Debug and Release come with sensible defaults and if you touch the Onn options, it means you know what you are doing.

Subject: Re: New Core
Posted by mdelfede on Fri, 20 May 2016 23:22:56 GMT
View Forum Message <> Reply to Message

Well... today I updated from svn and I got the "nice" surprise on my windows box
I guess it's time to fix some code and, the hard part, the Protect package.
I hope to find some spare time....

Subject: Re: New Core
Posted by mirek on Sat, 21 May 2016 11:30:11 GMT
View Forum Message <> Reply to Message

mdelfede wrote on Sat, 21 May 2016 01:22Well... today I updated from svn and I got the "nice" surprise on my windows box
I guess it's time to fix some code and, the hard part, the Protect package.
I hope to find some spare time....

You can still use 'classic' instead of trunk.

Subject: Re: New Core
Posted by mirek on Sat, 21 May 2016 11:40:10 GMT
View Forum Message <> Reply to Message

Novo wrote on Thu, 12 May 2016 17:08Another thing.

Could you please convert all Windows-related include file names to lower case? They are all lower case with mingw on Linux.
It looks like this is the right time to do that.

A patch file created by git looks weird, but still ...
You need to use WIN32 flag with mingw on Linux, and TheIDE won't handle rc-files.

Update: if I use WIN32 flag along with GUI and MT, then I get a problem with rc-files. If I just add a -DflagWIN32 common option, then everything compiles, but TheIDE is still linking against Linux libs ...

I guess I'm missing something.

TIA

Patch applied (I do not think it will solve your issues, but it does not harm either).

Subject: Re: New Core
Posted by Novo on Thu, 02 Jun 2016 23:23:04 GMT
View Forum Message <> Reply to Message

mirek wrote on Sat, 14 May 2016 01:59Novo wrote on Thu, 12 May 2016 16:23mirek wrote on Thu, 12 May 2016 03:09
Weird. That is the place it was always crashing, before I have fixed it with 'finetuning' -O options.

Are you using "Instant setup" default options?

Mirek

No, I was using old -O3 and -Os options.
Replacing -Os with -O2 fixed crashes with Optimal and Size configuration.
I didn't change -O3 to -O2 for the Speed configuration.
I do not understand where you are taking options for the Optimal configuration from. They are not declared explicitly anywhere.

I do not quite understand the question. I was just combining until it worked. Then adjusted 'instant setup' to found options.

BTW, my hypothesis here is that it is a bug linker that makes it impossible to combine size and speed optimized code (after all, there are no issues in Posix). Perhaps something related to inlines (-O3 is more aggresive in inlining everything). SO whatever, all is now -O2.

Mirek
I was trying to say that "Optimal" was hard-coded somewhere, and there was no way to change it without recompilation of TheIDE. Anyway, it is gone now.
