Subject: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by fudadmin on Wed, 07 Dec 2005 20:02:13 GMT
View Forum Message <> Reply to Message

It would be good to have this poll in "%" but I don't know howto...
so, please write your "%"...
Are there any other reasons?

biggest drawback of U++(total votes: 38)

Articles & Reviews 3/(8%)
Features 2/(5%)
Lack of documentation 33/(87%)

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by Garry on Sun, 08 Jan 2006 15:29:55 GMT
View Forum Message <> Reply to Message

Personally I think the best way to improve "popularity" is what's happening now - this forum.

Even without reams of documentation, a good forum is what's needed to help out newbies, like myself. I've followed the progress of UPP for quite some time, but it's only now since the forum got up and running, that I've gotten a bit more confidence in sticking more time and effort into doing something with U++. The original mailing list was far too daunting to try and look for information. Keep up the good work here.

Magazine articles would certainly also raise the profile of the project too. It's unusual to see such a powerful and well developed tool have such a low media impact. It's nearly impossible to find any information out there on the web except for download sites which always feature the same description of the project.

Anyway, hopefully the download figures are indicating a trend in the right direction.

Garry

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by fudadmin on Sun, 08 Jan 2006 15:51:57 GMT
View Forum Message <> Reply to Message

Garry wrote on Sun, 08 January 2006 10:29Personally I think the best way to improve "popularity" is what's happening now - this forum.

Even without reams of documentation, a good forum is what's needed to help out newbies, like myself. I've followed the progress of UPP for quite some time, but it's only now since the forum got up and running, that I've gotten a bit more confidence in sticking more time and effort into doing

something with U++. The original mailing list was far too daunting to try and look for information. Keep up the good work here.

Garry

Thank you for your encouraging words!

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by prof on Tue, 25 Apr 2006 03:41:11 GMT
View Forum Message <> Reply to Message

I would also consider the following:

- stability (Is it useable or buggy? Is it possible to build stable real world applications using it?)
- maturity (Are there many upp-based real world applications besides TheIDE? Is it used for years to develop commercial products? Is it likely that my application will still compile with future versions of upp? Is Assist++ parser good enough?)
- support (Is Upp actively maintained? Is upp community alive?)
- performance and effectivity (size of binaries, memory and cpu usage, ease of use, learning curve)
- look and feel
- interoperability, compatibility and portability (Is it easy to migrate from VC/boost.build/autotools/whatever to TheIDE and package system? Is it possible to build upp packages from boost build/msvc ide/ant/whatever?)
- scalability/useablity for large projects (Does it take ages to open Assist++ completion dialogs in large projects? Is version control/collaborative work supported?)

Etc etc... Well, I don't know yet...

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by riri on Tue, 25 Apr 2006 07:13:52 GMT
View Forum Message <> Reply to Message

Hi

I agree with prof at least for one term: look&feel.
Upp is designed to be portable, but has a Windows look&feel. I know this will be improved in the future, but for now, I beleave this's one of the reasons to make Ultimate++ unpopular.

Another reason could be it's particular design (like the foundations of NTL). Even if it seems to be a pleasure for certain persons, it way be a non sense for others ?

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?

Posted by fudadmin on Tue, 25 Apr 2006 07:29:03 GMT
View Forum Message <> Reply to Message

riri wrote on Tue, 25 April 2006 08:13Hi

I agree with prof at least for one term: look&feel.
Upp is designed to be portable, but has a Windows look&feel. I know this will be improved in the future, but for now, I beleave this's one of the reasons to make Ultimate++ unpopular.

Another reason could be it's particular design (like the foundations of NTL). Even if it seems to be a pleasure for certain persons, it way be a non sense for others ?

Look&feel is the most uninmportant thing. The most commercial programs are for Windows. Linux with X.org is just for kids at its current state.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Tue, 25 Apr 2006 07:32:25 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Tue, 25 April 2006 03:29riri wrote on Tue, 25 April 2006 08:13Hi

I agree with prof at least for one term: look&feel.
Upp is designed to be portable, but has a Windows look&feel. I know this will be improved in the future, but for now, I beleave this's one of the reasons to make Ultimate++ unpopular.

Another reason could be it's particular design (like the foundations of NTL). Even if it seems to be a pleasure for certain persons, it way be a non sense for others ?

Look&feel is the most uninmportant thing. The most commercial programs are for Windows. Linux with X.org is just for kids at its current state.

Bah, you have just tried the wrong distro

We will also need look&feel adjustments on Win too (think Vista if not theming...)

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by riri on Tue, 25 Apr 2006 07:43:07 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Tue, 25 April 2006 09:29Look&feel is the most uninmportant thing.

Really ?
u++ implements a GUI, so for me Look&Feel is important for people.

Quote:The most commercial programs are for Windows.

Must have commercial ambitions to be proud of using u++ ?

Quote:Linux with X.org is just for kids at its current state.

I've not tried really u++ under GNU/Linux so I can't tell.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Tue, 25 Apr 2006 07:50:27 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Tue, 25 April 2006 03:29
The most commercial programs are for Windows.

BTW, just a sidenote, things seem to be changing slowly for our bussines there. We already maintain a couple of linux apps now for our customers. And it even seems that one of my municipal agendas system will at least partly trash Windows at the end of year and replace with Linux - that would mean I will have to support Linux version soon.

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by fudadmin on Tue, 25 Apr 2006 08:57:24 GMT
View Forum Message <> Reply to Message

riri wrote on Tue, 25 April 2006 08:43

Quote:The most commercial programs are for Windows.

Must have commercial ambitions to be proud of using u++ ?

Wrong formulation of the question which shows wrong understanding of the situation.
I want, at least, u++ users (and creators) not to be ashamed and ridiculed (which is a normal stage of every inovation...) because of using it...
And, I'm very sure, it's in everyone's interests to make u++ better and more popular.
Because - unfortunately - pshycological labeling prevails at this stage of humans development.
That's why police, army and priests etc. wear uniforms... That's why most people choose not the better things but "more popular".
You need to reach some critical point when mass and popularity becomes "self expanding".
Endless story of "The King's dead! Long live the King!..."

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by riri on Tue, 25 Apr 2006 09:03:26 GMT
View Forum Message <> Reply to Message

luzr wrote on Tue, 25 April 2006 09:50... - that would mean I will have to support Linux version soon.

What a good news for Linux users

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by pivica on Thu, 27 Apr 2006 07:29:15 GMT
View Forum Message <> Reply to Message

Quote:... - that would mean I will have to support Linux version soon.

finally

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by prof on Thu, 27 Apr 2006 19:07:58 GMT
View Forum Message <> Reply to Message

It seems to me that the whole talk was focused just on "GUI toolkit" part of upp, and everybody left TheIDE and build system used by UPP without attention they deserve.

By "look and feel" I didn't mean the widgets - I meant TheIDE in comparison to VS IDE, upp build system in comparison to MCVC solutions, Boost.Build, Perforce Jam, makefiles, autotools etc.

I'm not a GUI developer. 90% of my needs for UI are satisfied by printf() call. The toolkit is good, but it isn't revolutionary. There are such things like Mozilla XUL/XRE and Tk. But the idea of packages impressed me. Finally there is a chance for source tree to become ordered. Finally, someone tried to introduce "modular concepts to C++ programming" (a quote from upp home page).

Talking about the widgets, they look ugly for me. I wouldn't use them in a software product I want to sell to plain home or corporate users. Of course, if I just want a quick and dirty UI, or if I make a prog for one customer for his personal use, it's ok. The widgets resemble early linux desktops and win16 (especially when I try to open a file). I hope that themes will fix this issue, and the UI will look more beautiful. Like Firefox, like Visual Studio, like Office.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Thu, 27 Apr 2006 19:17:10 GMT
View Forum Message <> Reply to Message

prof wrote on Thu, 27 April 2006 15:07

Talking about the widgets, they look ugly for me.

Hm, are you linux user or XP?

I think that look is now pretty damned close to XP Luna... (or Win2K in second variation).

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by forlano on Thu, 27 Apr 2006 19:59:07 GMT
View Forum Message <> Reply to Message

prof wrote on Thu, 27 April 2006 21:07Talking about the widgets, they look ugly for me.

I strongly disagree with you. Some widget could be improved, but their general appearance is excellent. Please have a look at this page , http://free-soft.org/guitool/ , to see what there is around. I've seen all of them and U++ has one of the best set of widgets.

Luigi

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by prof on Thu, 27 Apr 2006 20:19:47 GMT
View Forum Message <> Reply to Message

I'm XP user and I use "windows classic" theme. I think that these look better than TheIDE. Maybe, it's just due to colorful icons and fancy toolbar grips I used to?

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by unodgs on Thu, 27 Apr 2006 20:56:29 GMT
View Forum Message <> Reply to Message

prof wrote on Thu, 27 April 2006 16:19I'm XP user and I use "windows classic" theme. I think that these look better than TheIDE. Maybe, it's just due to colorful icons and fancy toolbar grips I used to?

Well, I don't see nothing better on these screenshots..

IMO Xp style looks very well, even better than oryginal. Yes there are some places to improvement like adding shaded left pane in menu bar or making toolbars similar to those from vs, but u++ gui look dosn't deserve to calling it ugly.
BTW TheIde goal is not to be beautiful but useful and comfortable. I coded long time in vs and bcb and IMO neither of them was better than theide (I'm not talking about vs + visual assist)

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by prof on Thu, 27 Apr 2006 22:34:53 GMT
View Forum Message <> Reply to Message

Yes. TheIDE is comfortable for me as for a developer. VisualAssist is just a code completion engine smarter than Intellisense. TheIDE already has a completion engine, and just need to make the c++ parser more complete. So there's nothing wrong with TheIDE.

I unfortunately do see a big difference between the widgets in those apps I mentioned and upp widgets. Maybe we should make a new poll - how many other people do see a difference in looks significant enough in order not to use upp GUI for their programs.

Compare Office 97 and Office 2003, 7zip and WinZip/WinRar, early Mozilla themes and Firefox. I've seen Mozilla before and refused to migrate due to its awful themes. But when I saw Firefox 0.8 with it's perfect Qute theme, it was the first open source GUI I really liked. When in 0.9 Qute was replaced with a more conservative theme, a lot of people were disappointed. So the community fixed the default theme in Firefox and now I'm equally comfortable with both.

And also two examples from the Web: http://www.visual-eiffel.com/front_content.php and http://www.sgi.com/support/custeducation/courses/linux/altix _adv_sys_adminii.html

The two pages just represent some textual information. No super DHTML/AJAX, no drop down menus etc on both pages. All usability recomendations seems to be followed. I can't tell what's wrong with the first page, but I definitely liked the latter more. It looks more balanced, more professional.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Thu, 27 Apr 2006 22:39:30 GMT
View Forum Message <> Reply to Message

prof wrote on Thu, 27 April 2006 16:19I'm XP user and I use "windows classic" theme. I think that these look better than TheIDE. Maybe, it's just due to colorful icons and fancy toolbar grips I used to?

Well, this was the first "visual style" created and I guess we are 99% equivalent there.

However, perhaps you are really speaking about how TheIDE icons look like - OK, there are

drawbacks based on our art skills

Please, note also that many of screenshots you have posted are in fact NOT using "classic theme". E.g. that svn screenshot has quite alternative appearance. In "classic theme", menus should look like menus in the windows explorer.

(And yes, toolbar grips are on the list

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Thu, 27 Apr 2006 22:55:04 GMT
View Forum Message <> Reply to Message

Well, I think you are mixing the "application visual design" and "widget appearance".

In other words, if you see some differences between U++ "classic" widgets (buttons, options etc...) and Win32 "classics", simply take and post screenshots of both to demonstrate differences and post to bugs database, as we are trying to minimize them (but maybe wait a month or so for the final skinning engine).

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by prof on Thu, 27 Apr 2006 23:22:40 GMT
View Forum Message <> Reply to Message

Well, you two an

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by cioannou on Wed, 25 Oct 2006 12:01:06 GMT
View Forum Message <> Reply to Message

This is the second time I try to use Upp and try to drop other tools.

Before reading my opinion please bear in mind:

(http://www.artima.com/cppsource/cpp0x.html)
In an article of Bjarn Stroustrup I read the following sentence which I strongly believe that is absolutely correct.

In my opinion, C++ has become too "expert friendly"

IMHO the most important drawbacks are:

1) Documentation (I am trying to find where "FindFile" is documented and all filesystem related functions).

2)I am not a C++ expert and took me half an hour forum searching to understand how to trap and handle events coming from my "form".

3)Upp Looks "closed" and proprietary.

4)Controls are not very nice looking (e.g. treeview looks terrible, no gradient toolbars) Modern applications should look like this(Sample Powerbuilder application):

http://kodigo.sourceforge.net/images/Screenshots/screenshot4 .png

5)Everybody likes to write code fast and easy, that's why VB,Powerbuilder and similar tools have loyal friends. Except from the C++ gurus/professionals every other mortal soul hates pointers and strange syntax. So the closer you get to VB or Powerbuilder the more friends you'll get.

6) Comments in examples. Not everyone is a C++ guru.

Suggestions:

1) Better docs
2) A simple but nice looking upp example (with comments in the code) that demonstrates all controls, event handling, database connections, filesystem access, common dialogs (fileopen, browseforfolder, fontselect), external library integration (e.g. Boost), imagebuttons, toolbar, statusbar,resizeable controls.
And please avoid the <expr>?<foo>:<bar> confusing C++ shortcuts.
3)Take a look in Powerbuilder, I think it's the friendliest and RADest tool I've ever seen.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Wed, 25 Oct 2006 12:25:52 GMT
View Forum Message <> Reply to Message

Quote:
In my opinion, C++ has become too "expert friendly"

BTW, I am thinking about makeing Bjarne aware about our efforts, maybe he could like what we are trying to achieve (well, most likely not (no STL), but we can try

Any ideas how to do that right?

Mirek

P.S.:

As for the comments, I think it is still the same story, documentation comes first. But I think quite a lot effort was spend there, so current status is much better (and will gradually continue to improve).

As for the look, current goal is to achieve host platform defined look and feel with skinning option - 80% of that is now achieved (minus GTK themes). Anyway, looking at the screenshot, its look is unstandard for XP. I think we should start with standard and move to advanced later (which in the end can be achieved by skinning).

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by cioannou on Wed, 25 Oct 2006 12:43:35 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 25 October 2006 15:25

As for the comments, I think it is still the same story, documentation comes first. But I think quite a lot effort was spend there, so current status is much better (and will gradually continue to improve).

As for the look, current goal is to achieve host platform defined look and feel with skinning option - 80% of that is now achieved (minus GTK themes). Anyway, looking at the screenshot, its look is unstandard for XP. I think we should start with standard and move to advanced later (which in the end can be achieved by skinning).

Mirek

Maybe you're right but since January (The last time I visited Upp) I didn't notice any major changes.

What do you mean by "unstandard"?

Besides the docs, I believe that my suggestion No2 would be helpful enough and faster to do compared to analytical docs.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Wed, 25 Oct 2006 12:56:37 GMT
View Forum Message <> Reply to Message

cioannou wrote on Wed, 25 October 2006 08:43

Maybe you're right but since January (The last time I visited Upp) I didn't notice any major changes.

Have you downloaded dev version? (Note: Right now, there are not gradients in toolbar yet. But rest is more or less used from XP theming engine).

Quote:
What do you mean by "unstandard"?

E.g. unless you have different theme installed (which U++ now can use too).

Quote:
Besides the docs, I believe that my suggestion No2 would be helpful enough and faster to do compared to analytical docs.

You might be right. I agree that all examples are written for C++ programmers. And even for them, initial learning curve is steep.

In fact, I believe there is something like "being ready" factor in U++ now: We have designed it because we were sick of other solutions. So when you are aged C++ programmer sick of problems, chances are hight you will understand U++ quick, because it primarily tries to find solution of problems almost every programmer has... (but is unaware of them before creating a couple of applications). (That of course does not mean that U++ is problem-less, but most of problems now are about lacking features, not core idea - I believe that one is sound for solving problems U++ was designed to solve).

Anyway, back to the comment. Besides lack of time, I think another problem is that core U++ developers are not quite surewhat to comment and what not. For me, all examples are too simple and understandable, I am really not sure what should be commented and what not.

Also, I am not sure we should try to teach U++ users how to program in C++...

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Wed, 25 Oct 2006 12:59:28 GMT
View Forum Message <> Reply to Message

Quote:
What do you mean by "unstandard"?

It contains elements not defined or defined otherwise (in standard theme) in XP. E.g. XP tab widget cannot look like that.

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by cioannou on Wed, 25 Oct 2006 13:48:59 GMT
View Forum Message <> Reply to Message

Quote:
Anyway, back to the comment. Besides lack of time, I think another problem is that core U++ developers are not quite surewhat to comment and what not. For me, all examples are too simple and understandable, I am really not sure what should be commented and what not.

Also, I am not sure we should try to teach U++ users how to program in C++...

Can't tell you what to comment exactly, but I guess that whatever is UPP related and since we're talking about examples code almost everything needs a comment.

U++ users are supposed to have read one or two "C++ for Dummies" or "Teach yourself C++" books.

btw, IMHO this is the biggest problem in C++, the most difficult part in this language is to get from beginner to intermediate by writing real-world useful programs.There is absolutely no book/guide/tutorial that would get you from simple "cout << 'Hello World'" subjects to real-world GUI programming without having to really master templates, vectors, pointers that point to pointers that point to functions etc. If we're talking about users, all this is crap, especially for business application developers. All they need is a full featured IDE/RAD tool, ready made functionality (e.g. data editing made easy) and a bunch of commands and controls to work with. (Reminder: VB, Powerbuilder, Delphi etc.)

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by yeohhs on Wed, 25 Oct 2006 16:11:44 GMT
View Forum Message <> Reply to Message

cioannou wrote on Wed, 25 October 2006 20:01
2) A simple but nice looking upp example (with comments in the code) that demonstrates all controls, event handling, database connections, filesystem access, common dialogs (fileopen, browseforfolder, fontselect), external library integration (e.g. Boost), imagebuttons, toolbar, statusbar,resizeable controls.
And please avoid the <expr>?<foo>:<bar> confusing C++ shortcuts.

I'm working on something like this now. I had known about Ultimate++ since 2005 but had not committed to learning it until several days ago.

After having used MFC, FLTK, FOX GUI and wxWidgets for the past few years I can say that Ultimate++ certainly could do better than all of them. I see it has huge potential.

The example I'm working on now is a simple "dialog-based" U++ application with splash screen, menu, shortcut keys, toolbar, status bar, about dialog, user-created dialog, auto-save config and Topic++ help for the program, etc. This version aims to show how the simple program features are implemented in U++. I want to keep this project simple but close to real world applications.

Once this is done, I will re-use the same application but focus on other aspects of Ultimate++ like database, web, etc. This way, I think will be easy for newbies to learn U++.

I'll post a zip of the first app when it is ready.

Best Regards,
Yeoh

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by forlano on Wed, 25 Oct 2006 16:53:11 GMT
View Forum Message <> Reply to Message

yeohhs wrote on Wed, 25 October 2006 18:11

Once this is done, I will re-use the same application but focus on other aspects of Ultimate++ like database, web, etc. This way, I think will be easy for newbies to learn U++.

I'll post a zip of the first app when it is ready.

Best Regards,
Yeoh

WOW... I'm waiting...
Although I'm using U++ I feel to use it only at 1% of its real possibility.

Luigi

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by forlano on Wed, 25 Oct 2006 17:07:37 GMT
View Forum Message <> Reply to Message

cioannou wrote on Wed, 25 October 2006 14:01This is the second time I try to use Upp and try to drop other tools.

Before reading my opinion please bear in mind:

(http://www.artima.com/cppsource/cpp0x.html)
In an article of Bjarn Stroustrup I read the following sentence which I strongly believe that is absolutely correct.

In my opinion, C++ has become too "expert friendly"

IMHO the most important drawbacks are:

1) Documentation (I am trying to find where "FindFile" is documented and all filesystem related functions).

2)I am not a C++ expert and took me half an hour forum searching to understand how to trap and handle events coming from my "form".

3)Upp Looks "closed" and proprietary.

4)Controls are not very nice looking (e.g. treeview looks terrible, no gradient toolbars) Modern applications should look like this(Sample Powerbuilder application):

Hi,

the U++ widget are now very nice, especially under XP since the introduction of chamaleon technology.

Recently I've used some application designed to work with .NET platform. Wonderfull looking and amazing gradient... unfortunately it run terribly slow! Up some threshold the application become horrible despite of the gradient.

Now U++ is very fast and I would not change this aspect with heavy but fancy colored window.

Luigi

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Wed, 25 Oct 2006 23:07:58 GMT
View Forum Message <> Reply to Message

Quote:
The example I'm working on now is a simple "dialog-based" U++ application with splash screen, menu, shortcut keys, toolbar, status bar, about dialog, user-created dialog, auto-save config and

Topic++ help for the program, etc. This version aims to show how the simple program features are implemented in U++. I want to keep this project simple but close to real world applications.

It would be nice to make it an article too, similiar to that one of Matt Ezell on Codeproject (good for U++ PR). It would be excellent to get all main C++ related sites "populated" with some U++ articles....

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by yeohhs on Wed, 25 Oct 2006 23:55:27 GMT
View Forum Message <> Reply to Message

[quote title=luzr wrote on Thu, 26 October 2006 07:07]Quote:
It would be nice to make it an article too, similiar to that one of Matt Ezell on Codeproject (good for U++ PR). It would be excellent to get all main C++ related sites "populated" with some U++ articles....

Mirek

Yes, this is a very good idea. I like writing.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by cioannou on Thu, 26 Oct 2006 13:42:42 GMT
View Forum Message <> Reply to Message

Very good idea indeed!

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by richardt on Thu, 26 Oct 2006 20:24:08 GMT
View Forum Message <> Reply to Message

I agree. Ive read more books on C++ than you can shake a stick at. None of them can help in with the transition to GUI programming when it comes to Ultimate. Usefull business applications need to store data, and the combination of Ultimate with Sqlite could be in my opinion unbeatable. (Try looking on the web for such a combination that is mature, open and multiplatform). You would have more luck searching for rocking horse dung!

I think that a few simple examples would lead to an enormous increase in interest and more cross

Richard

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Thu, 26 Oct 2006 23:36:28 GMT
View Forum Message <> Reply to Message

Quote:

I think that a few simple examples would lead to an enormous increase in interest and more cross

BTW, there is now quite complete SQLApp in examples. Do you find it helpful?

MIrek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by richardt on Fri, 27 Oct 2006 08:39:11 GMT
View Forum Message <> Reply to Message

luzr wrote on Thu, 26 October 2006 19:36Quote:

I think that a few simple examples would lead to an enormous increase in interest and more cross

BTW, there is now quite complete SQLApp in examples. Do you find it helpful?

MIrek

My comment was based on the 605 release, maybe I should have looked at the DEV releases . Ok. Ive now installed 609 and found the rather nice example that you refer to. I get errors when I try to compile it, what am I doing wrong?:

----- CtrlLib (GUI GCC32 DEBUG SHARED DEBUG_FULL BLITZ LINUX) (1 / 12)
----- SqlCtrl (GUI GCC32 DEBUG SHARED DEBUG_FULL BLITZ LINUX) (2 / 12)
BLITZ: SqlCtrl.cpp SqlArray.cpp SqlDetail.cpp SqlDlg.cpp SqlConsole.cpp SqlObjectTree.cpp
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:255: error: declaration of ‘typedef class SqlConsole SqlCon
sole::CLASSNAME’

/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:196: error: conflicts with previous declaration ‘typedef cl
 ass SqlConsole SqlConsole::CLASSNAME’
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp: In member function ‘void SqlConsole::ListPrintRow()’:
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:533: error: ‘DocReport’ was not declared in this scope
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:533: error: expected `;' before ‘report’
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:534: error: ‘report’ was not declared in this scope
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp: In member function ‘void SqlConsole::ListPrintList()’:
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:551: error: ‘DocReport’ was not declared in this scope
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:551: error: expected `;' before ‘report’
/home/richard/upp/uppsrc/SqlCtrl/SqlConsole.cpp:552: error: ‘report’ was not declared in this scope
/home/richard/upp/uppsrc/Oracle/Oracle8.h: At global scope:
/home/richard/upp/uppsrc/Oracle/Oracle8.h:63: error: ‘OCI8Connection’ was not declared in this scope
/home/richard/upp/uppsrc/Oracle/Oracle8.h:63: error: template argument 1 is invalid
/home/richard/upp/uppsrc/SqlCtrl/SqlObjectTree.cpp: In member function ‘void SqlObjectTree::OpenTableColumn
 s(int, const SqlObjectTree::Item&)’:
/home/richard/upp/uppsrc/SqlCtrl/SqlObjectTree.cpp:358: error: call of overloaded ‘Add(int&, const Image&,
 SqlObjectTree::Item, String&)’ is ambiguous
/home/richard/upp/uppsrc/CtrlLib/TreeCtrl.h:137: note: candidates are: int TreeCtrl::Add(int, const Image&,
  Value, bool)
/home/richard/upp/uppsrc/CtrlLib/TreeCtrl.h:138: note: int TreeCtrl::Add(int, const Image&,
  Value, Value, bool)
/home/richard/upp/uppsrc/SqlCtrl/SqlObjectTree.cpp:364: error: call of overloaded ‘Add(int&, const Nuller&,
  SqlObjectTree::Item, String&)’ is ambiguous
/home/richard/upp/uppsrc/CtrlLib/TreeCtrl.h:137: note: candidates are: int TreeCtrl::Add(int, const Image&,
  Value, bool)
/home/richard/upp/uppsrc/CtrlLib/TreeCtrl.h:138: note: int TreeCtrl::Add(int, const Image&,
  Value, Value, bool)
/home/richard/upp/uppsrc/SqlCtrl/SqlObjectTree.cpp:368: error: call of overloaded ‘Add(int&, const Nuller&,
  SqlObjectTree::Item, String)’ is ambiguous
/home/richard/upp/uppsrc/CtrlLib/TreeCtrl.h:137: note: candidates are: int TreeCtrl::Add(int, const Image&,
  Value, bool)

/home/richard/upp/uppsrc/CtrlLib/TreeCtrl.h:138: note: int TreeCtrl::Add(int, const Image&,
Value, Value, bool)
SqlCtrl: 6 file(s) built in (0:03.65), 608 msecs / file, duration = 3663 msecs

There were errors. (0:03.69)

Thanks in advance for your help.

Richard

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by yeohhs on Fri, 27 Oct 2006 09:23:34 GMT
View Forum Message <> Reply to Message

I'm also unable to compile the SQLApp example.

Here is the error message.
Quote:
----- SQLApp (GUI MAIN MSC8 WIN32 MSC) (13 / 13)
query.cpp
C:\upp\uppsrc\CtrlLib/EditCtrl.h(130) : error C2248: 'Ctrl::operator =' : cannot access private member
declared in class 'Ctrl'
C:\upp\uppsrc\CtrlCore/CtrlCore.h(234) : see declaration of 'Ctrl::operator ='
C:\upp\uppsrc\CtrlCore/CtrlCore.h(172) : see declaration of 'Ctrl'
This diagnostic occurred in the compiler generated function 'EditField &EditField::operator =(EditField &)'

Thanks for any help.

Best Regards.
Yeoh

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by yeohhs on Sat, 28 Oct 2006 07:35:08 GMT
View Forum Message <> Reply to Message

forlano wrote on Thu, 26 October 2006 00:53

WOW... I'm waiting...
Although I'm using U++ I feel to use it only at 1% of its real possibility.

Luigi

I've posted the zip in the "U++ users applications in progress..." subforum.

Best Regards
Yeoh

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by agent86 on Mon, 25 Dec 2006 03:12:53 GMT
View Forum Message <> Reply to Message

I just got to using U++ a week ago. My first impression was that it was strange. I was a stranger in a strange land. The very first screen asked me about copying some items to my local directory. I had no idea how to answer that. The next screen was equally unfamiliar. I needed to pick something called an assembly or a package or something. Whatever they are...

I was expecting a 3-piece window with a welcome screen etc. I didn't get the comformtable feeling of VC++ or eclipse or MinGWStudio or Code::Blocks or wxHatch or Anjuta or DevC++ or Kdevelop or....

U++ and TheIDE was strange. I had to expend a fair amount of energy to investigate. I had to investigat to see if I wanted to learn more. If MinGWStudio were not dead (dying?), I might have just stuck with that. I believe that the work I have done with U++ and TheIDE has been and will be very beneficial.

Forgive the military analogy, but I feel that with the other IDSs I was flying biplanes in the 1920s. After a week with TheIDE I feel like someone sat me in the cockpit of a supersonic fighter/bomber. I am in an unknown and maybe a little scary environment but have more weapons at my disposal than I can begin to appreciate at my current learning level. Slightly overwhelmed...

So I needed to get past the "What the heck is a package and a nest and an assembly and why doesn't U++ talk about projects?" stage. I'm doing ok so far.

So the short answer is that the immediate impression of U++ is that it is "very different". There is a learning curve just to see what it does.

As for the look and feel of the widgets, they seem great to me. I don't want much more than they look reasonable and work the way one would expect. Users should not be supprised about how they work. (EG. in Java Swing, if you tab across a dialog box and highlight the desired button and hit Return, you don't get the selected button action. You get the default button. Space bar gets the highlighted button. Not intuitive. I hate it.)

Anyway, I develop on both Linux and Windows. I like that my apps can be based on the same code set. I have been using wxWidgets and am ready to switch, but I am not sure yet...cause I

need to read some documentation. Maybe I haven't found it yet, but a simple overview of the widgets and how they fit together would help. What constitutes an application, the typical flow, etc.

Thanks for reading, Sometimes I get carried away.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by fallingdutch on Mon, 25 Dec 2006 07:01:25 GMT
View Forum Message <> Reply to Message

nice little story ... enjoyed reading it

A list of all Widgets:
http://www.ultimatepp.org/src$CtrlLib$index$en-us.html

Documentation:
hit ctrl+Enter in TheIDE will open Topic++ and show you the latest documentation.

within Linux i prefer not to copy the sources to my home dir because with a update you won't get the newest code.

A Project can consist of one or more Packages.
A Package is Part of a Assembly, which can contain other Assemblys to be able to compile. (right click on an Assably and select edit and you will see the Package nests, a line with all Assamblies used to get Packages compiled. output dir is the part where the compiled and linked files will be. Common files is - afaik - not used very much if at all.

Hope i could help some,
Bas
[Update:] i don't know wich version you use, but 612-dev1 is worth it to be downloaded ...
612-dev2 is not so clean but the most current

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by agent86 on Thu, 04 Jan 2007 07:17:37 GMT
View Forum Message <> Reply to Message

Thanks for the response. I had no idea that all that information was sitting behind the Ctrl-Enter key.

I've seen the Topic++ entry in the Assist menu, but somehow I got the idea that it was to generate docs, not read them...

Sigh.

The list of widgets is going to be a great help. My current work involves console apps, but I have plans...

Anyway, Thanks again.

chuck

ps, I am now running 612 dev 3 on Debian Etch, Fedora Core 6, Windows 2000, and Windows XP. Seems OK except for some of the hot-key listings on the menus (eg ctrl+100dc for split horizontal).

I guess they will be resolved before the next release.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by fallingdutch on Thu, 04 Jan 2007 07:31:38 GMT
View Forum Message <> Reply to Message

Hi Chuck,

agent86 wrote on Thu, 04 January 2007 08:17My current work involves console apps, but I have plans...

Console Apps work with Upp, too
CONSOLE_APP_MAIN{
//your code goes here ;)
}
agent86 wrote on Thu, 04 January 2007 08:17
Seems OK except for some of the hot-key listings on the menus (eg ctrl+100dc for split horizontal).
I don't know at the moment, wether this bug is already known, but i remember something with Hotkeys.

Bas

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by indiocolifa on Mon, 21 Apr 2008 19:40:31 GMT
View Forum Message <> Reply to Message

Thanks god I've discovered U++, instead of going through the MFC-ish WxWidget route...

When 2008 is released, all the U++ community (We) should build a proper documentation. 2009 the year of Ultimate++

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by Mindtraveller on Tue, 22 Apr 2008 07:20:32 GMT
View Forum Message <> Reply to Message

The most significant causes of current U++ state are it`s targeting for experienced C++ programmers and lack of adequate documentation. The first one isn`t drawback, the second one should be improved in the near future.

1) U++ is heavily targeted for experienced C++ programmers.
That`s it. And there`s no way of feeling comfortable in U++ without strong experience in C++. This is NOT a drawback, it is just a professional instrument. It mustn`t be as pretty and fancy looking as more beginner` ones. It just should work effectively.
This is what I love in U++. And, yes, it solves many problems which experienced programmers meet before U++ was created.
As for me, I can compare U++ team efforts with many others: starting with Turbo Vision (yes! good old under DOS by Borland), plain Win32 programming, custom GUI for DirectDraw, MFC, VCL (Delphi & BC++B), QT, wxWidgets - and finally U++ beats them. But it requires strong knowledge of C++ language and tricks. Because it is created with C++ tricks. So, to use it effectively, one must be good enough to understand all behind these tricks. This way you have smaller code, well organized program structure and powerful abilities of U++. Again - this professional tool mustn`t have fancy look. It must be effective. And it is - if we talk about U++ and TheIDE.

2) Currently U++ authors targeted for widening framework abilities. And for now it has no appropriate documentation/manual/help system. Current documentation state covers most of basic things about basic abilities. Everything else you should find in forum or from sources. It is drawback but Mirek promised to focus on that in next major release. I hope he will also take into consideration community` ideas about documentation.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by tvanriper on Tue, 22 Apr 2008 12:09:59 GMT
View Forum Message <> Reply to Message

First and foremost, the documentation holds Ultimate++ back from mainstream adoption. It took me almost a month to get mostly comfortable with the toolkit, not because it isn't easy to use, but because I had to literally study the code in order to figure out how to work with it. This acts as a barrier to entry, when you have other toolkits with far better documentation (e.g. MFC *choke*).

Next, you need to get the attention of some heavy hitters. Most of these guys, from what I can tell, hang out at boost.org. In the United States, I've worked for about five different large-ish companies, and in all of them, they held a tremendous respect for the efforts at boost.org, because of the whole peer-review system, and because often the code at boost.org makes its way

into the standard library over time.

You mentioned an interest in getting Bjarne Stroustrup's attention. You can find him participating at boost.org. If you seriously want to cause Ultimate++ to be put into the standard C++ library (which would be an amazing feat for a library like this), put the library up for review.

If you do this, I can tell you right now what will happen, at least initially.

Ultimate++ will be shot down.

Poor documentation; they're picky about having helpful documentation.
The use of Ultimate++-centric interfaces instead of standard interfaces. More on this later.
Possible portability issues.
Possible size of system.
namespace isn't in boost.
Duplication of technologies already available in boost (e.g. boost::thread already exists for multi-threading, and boost already has a system for handling signals and slots).
Difficulty in using Ultimate++ without using TheIDE. I know you're already working (or maybe finished working) on addressing this issue.

The NTL system you developed will most likely be an issue for the boost.org team, not because the underlying technology is terrible, but because the interfaces do not match the kind of interfaces used with the standard library. It doesn't 'look' like something you'd see in the standard library.

I think that's a small issue. You can accomplish both your technical goals (fast, agile code) and still have the library 'feel' like it belongs as part of the standard template library.

Although I'm confident you'd get shot down, there are several significant benefits in submitting the library for review, regardless.

You'll get serious, useful information for improving Ultimate++.
You'll get serious, useful exposure to a very large community of developers, some of whom are surely interested in the problems solved by Ultimate++.

Note that almost all first attempts at a library get shot down at boost.org, so it isn't necessarily a bad thing. Personally, of all the toolkits out there, this is the only one I think could come close to being put up for review at boost.org without being outright mocked. You already have the right kind of license. You tend to follow ideals that seem in line with what I've seen at boost.org. You solve a problem domain that a lot of people need solved.

However, maybe you do not have a goal to cause Ultimate++ to become part of the standard library. In that case, don't bother submitting it to boost. You'd only be wasting each other's time. Personally, I think that'd be a pity, as I think both of you (boost and Ultimate++) could benefit from

each other.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Tue, 22 Apr 2008 12:44:36 GMT
View Forum Message <> Reply to Message

tvanriper wrote on Tue, 22 April 2008 08:09
I think that's a small issue. You can accomplish both your technical goals (fast, agile code) and still have the library 'feel' like it belongs as part of the standard template library.

I am not quite sure about it being a "small issue".

IMO it all starts with reasons for String vs std::string...

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by tvanriper on Fri, 25 Apr 2008 11:14:41 GMT
View Forum Message <> Reply to Message

luzr wrote on Tue, 22 April 2008 08:44tvanriper wrote on Tue, 22 April 2008 08:09
I think that's a small issue. You can accomplish both your technical goals (fast, agile code) and still have the library 'feel' like it belongs as part of the standard template library.

I am not quite sure about it being a "small issue".

IMO it all starts with reasons for String vs std::string...

Mirek
This is where the peer review system is very helpful, I think.

To get an idea of what the review system looks like, you can view past reviews online, if you like:

http://blog.gmane.org/gmane.comp.lib.boost.announce

I don't think the Egg review is the best one to examine (not enough people participated)... I'd start with the review some folks did of the Logging library, to get a flavor for how the review process helps.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Fri, 25 Apr 2008 11:52:44 GMT

tvanriper wrote on Fri, 25 April 2008 07:14luzr wrote on Tue, 22 April 2008 08:44tvanriper wrote on Tue, 22 April 2008 08:09
I think that's a small issue. You can accomplish both your technical goals (fast, agile code) and still have the library 'feel' like it belongs as part of the standard template library.

I am not quite sure about it being a "small issue".

IMO it all starts with reasons for String vs std::string...

Mirek
This is where the peer review system is very helpful, I think.

How do you think it could help to resolve this problem?

The real issue IMO is that boost is std:: extension, while U++ is std:: replacement... The basic design fundamentals and requirements are quite alien to std:: / boost concepts. Therefore I just fail to see how std:: replacement could do anything with boost.

Well, the only advantage would be to "get the attention", but I am not quite sure that is a good idea either....

(OTOH, peer review process itself is a very good idea, I wish we had something like that for U++ too...).

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by tvanriper on Sat, 26 Apr 2008 00:36:48 GMT

luzr wrote on Fri, 25 April 2008 07:52tvanriper wrote on Fri, 25 April 2008 07:14luzr wrote on Tue, 22 April 2008 08:44tvanriper wrote on Tue, 22 April 2008 08:09
I think that's a small issue. You can accomplish both your technical goals (fast, agile code) and still have the library 'feel' like it belongs as part of the standard template library.

I am not quite sure about it being a "small issue".

IMO it all starts with reasons for String vs std::string...

Mirek
This is where the peer review system is very helpful, I think.

How do you think it could help to resolve this problem?

The real issue IMO is that boost is std:: extension, while U++ is std:: replacement... The basic design fundamentals and requirements are quite alien to std:: / boost concepts. Therefore I just fail to see how std:: replacement could do anything with boost.

Well, the only advantage would be to "get the attention", but I am not quite sure that is a good idea either....

(OTOH, peer review process itself is a very good idea, I wish we had something like that for U++ too...).

Mirek

If I have it right, your primary concern with std:: involves its relatively terrible performance, and I think the technical issue had to do with the way you manipulate memory. If that's the concern, someone could potentially help you find a way to achieve the same performance you currently get with NTL, while using a more std::-like interface.

I could, of course, be mistaken. I'm not completely clear on why you feel these are so incompatible... as perhaps I'm not 100% clear on your design goals, or I'm ignorant of the fundamental problem you see in std::. I've read and re-read this page, but I still can't quite see how U++ and std:: can be so incompatible that there's no hope of improving the std::-style system to the point of matching U++ performance. I've certainly seen quite a few tricks used at boost:: that involved improving performance in a variety of ways (sometimes by providing a replacement for an existing library in std::, I think)... I should think they'd take quite a bit of interest in your techniques, and could possibly figure out a way to keep the same performance while making fewer changes to the existing interfaces.

I only pose this idea because it feels to me like you and boost have similar goals. I could, of course, be wrong. I know, for example, that boost has less of an emphasis on performance and more of an emphasis on their idea of 'correctness', so you may differ significantly there. (This is certainly not to say you have no concern for 'correctness', but that you may have a slightly different idea of what is 'correct' from boost).

You can best decide for yourself whether or not to approach them by reading some of the reviews people have left, and examine the decisions used to arrive at their conclusions.

Regardless, you could probably have your own peer-review system... it'd be interesting to see how people here would work with that. At the very least, such a system might help you gain some continuity between various objects in the U++ system.

As for gaining the attention of some of those heavy-hitters... well, there can be positive and negative results, to be sure. I thought it was possibly of interest to you only because you had mentioned attracting the attention of Stroustrup.

Perhaps someone could submit an article to Dr. Dobb's Journal showcasing the use of Ultimate++; that's a fairly popular magazine, at least here in the United States (the CUJ folded to Dr. Dobb's a few years ago, sadly, or I would have recommended it instead).

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mr_ped on Sat, 26 Apr 2008 01:23:56 GMT
View Forum Message <> Reply to Message

Quote:
I could, of course, be mistaken. I'm not completely clear on why you feel these are so incompatible...

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

I agree the interfaces could have been much more similar, no matter how much it differs inside. Also in U++ there are still some inconsistencies across different APIs for similar things, so even if it has to be different from std::, at least some things should be united in the future.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Sat, 26 Apr 2008 06:11:39 GMT
View Forum Message <> Reply to Message

tvanriper wrote on Fri, 25 April 2008 20:36
If I have it right, your primary concern with std:: involves its relatively terrible performance,

Well, not really. If am to put it in a very simple way, the main problem with std:: is that it makes you wish the C++ had garbage collector....

Quote:
If that's the concern, someone could potentially help you find a way to achieve the same performance you currently get with NTL, while using a more std::-like interface.

Well, what would be that? Something like these macros at the end of Core/topt.h?

```
// STL compatibility hacks

#define STL_INDEX_COMPATIBILITY(C) \
 typedef T            value_type; \
 typedef ConstIterator const_iterator; \
 typedef const T&     const_reference; \
 typedef int          size_type; \
 typedef int          difference_type; \
```

```
 const_iterator       begin() const         { return B::Begin(); } \
 const_iterator       end() const           { return B::End(); } \
 void                 clear()               { B::Clear(); } \
 size_type            size()                { return B::GetCount(); } \

#define STL_BI_COMPATIBILITY(C) \
 typedef T            value_type; \
 typedef ConstIterator const_iterator; \
 typedef const T&     const_reference; \
 typedef int          size_type; \
 typedef int          difference_type; \
 const_iterator       begin() const         { return Begin(); } \
 const_iterator       end() const           { return End(); } \
 void                 clear()               { Clear(); } \
 size_type            size()                { return GetCount(); } \
 typedef Iterator     iterator; \
 typedef T&           reference; \
 iterator             begin()               { return Begin(); } \
 iterator             end()                 { return End(); } \

#define STL_MAP_COMPATIBILITY(C) \
 typedef T            value_type; \
 typedef ConstIterator const_iterator; \
 typedef const T&     const_reference; \
 typedef int          size_type; \
 typedef int          difference_type; \
 const_iterator       begin() const         { return B::Begin(); } \
 const_iterator       end() const           { return B::End(); } \
 void                 clear()               { B::Clear(); } \
 size_type            size()                { return B::GetCount(); } \
 typedef Iterator     iterator; \
 typedef T&           reference; \
 iterator             begin()               { return B::Begin(); } \
 iterator             end()                 { return B::End(); } \

#define STL_VECTOR_COMPATIBILITY(C) \
 typedef T            value_type; \
 typedef ConstIterator const_iterator; \
 typedef const T&     const_reference; \
 typedef int          size_type; \
 typedef int          difference_type; \
 const_iterator       begin() const         { return Begin(); } \
 const_iterator       end() const           { return End(); } \
 void                 clear()               { Clear(); } \
 size_type            size()                { return GetCount(); } \
 typedef Iterator     iterator; \
 typedef T&           reference; \
 iterator             begin()               { return Begin(); } \
```

```
iterator            end()                { return End(); } \
reference           front()              { return (*this)[0]; } \
const_reference     front() const        { return (*this)[0]; } \
reference           back()               { return Top(); } \
const_reference     back() const         { return Top(); } \
void                push_back(const T& x)  { Add(x); } \
void                pop_back()           { Drop(); }
```

Quote:
I could, of course, be mistaken. I'm not completely clear on why you feel these are so incompatible... as perhaps I'm not 100% clear on your design goals, or I'm ignorant of the fundamental problem you see in std::.

The real trouble starts with the fact that you cannot use std::string as map keys. You cannot use any concrete class defined in std:: as element of any Vector flavor U++ container.

So far, the main "incompatibility complaint" was that "U++ guys seem to define their own containers and string". This is not easy to fix

I've read and re-read [url=http://www.ultimatepp.org/www$uppweb$vsstd$en-us.html]this page[/url], but I still can't quite see how U++ and std:: can be so incompatible that there's no hope of improving the std::-style system to the point of matching U++ performance.

Ah, but you could fix std::. But it is not likely to happen.

Moreover, adopting all U++ tricks into std:: would change its semantics and break existing code.

I only pose this idea because it feels to me like you and boost have similar goals. I could, of course, be wrong. I know, for example, that boost has less of an emphasis on performance and more of an emphasis on their idea of 'correctness', so you may differ significantly there. (This is certainly not to say you have no concern for 'correctness', but that you may have a slightly different idea of what is 'correct' from boost).

Oh, I have a very strong concern for 'correctness' - to the degree that I often rather break existing code by fixing some "incorrectness" in U++ Core.

Also, please, do not think I am not aware about boost or that I think these people are stupid. Of course not, boost is a very good effort and the code is pretty good.

I just feel U++ is not a good fit there. It is almost like suggesting boost to adopt Java

BTW: I mostly care about "optimality" with U++. If I would care about "popularity" more, I would

certainly use another path and boost would be the part of it.

Quote:
Perhaps someone could submit an article to Dr. Dobb's Journal showcasing the use of Ultimate++; that's a fairly popular magazine, at least here in the United States (the CUJ folded to Dr. Dobb's a few years ago, sadly, or I would have recommended it instead).

I guess that would be much better idea:)

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Sat, 26 Apr 2008 06:13:55 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Fri, 25 April 2008 21:23Quote:
I could, of course, be mistaken. I'm not completely clear on why you feel these are so incompatible...

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

I agree the interfaces could have been much more similar, no matter how much it differs inside. Also in U++ there are still some inconsistencies across different APIs for similar things, so even if it has to be different from std::, at least some things should be united in the future.

Definitely. This happens if you develop/extend library to solve the problem:)

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by tvanriper on Sat, 26 Apr 2008 11:41:31 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 26 April 2008 02:11tvanriper wrote on Fri, 25 April 2008 20:36
If I have it right, your primary concern with std:: involves its relatively terrible performance,

Well, not really. If am to put it in a very simple way, the main problem with std:: is that it makes you wish the C++ had garbage collector....

I can appreciate that. I get frustrated sometimes, trying to write code that doesn't require 'new' when using std::. And I strongly agree with you that an object belongs somewhere.

luzr
tvanriper
If that's the concern, someone could potentially help you find a way to achieve the same performance you currently get with NTL, while using a more std::-like interface.

Well, what would be that? Something like these macros at the end of Core/topt.h?

(I only removed the quoted code to reduce the size of the message.)

Partially... but you also appear to introduce difference concepts for use with your collections than the standard. I would expect, if you were to incorporate this into boost::, you'd provide concept-checking mechanisms that would cause clear compiler errors if someone attempted to use an inappropriate data type... one that didn't support your collections' concept.

luzr
tvanriper
I could, of course, be mistaken. I'm not completely clear on why you feel these are so incompatible... as perhaps I'm not 100% clear on your design goals, or I'm ignorant of the fundamental problem you see in std::.

The real trouble starts with the fact that you cannot use std::string as map keys. You cannot use any concrete class defined in std:: as element of any Vector flavor U++ container.

So far, the main "incompatibility complaint" was that "U++ guys seem to define their own containers and string". This is not easy to fix

I do not view this, necessarily, as a problem.

I note that boost currently has specialized containers to solve specific sorts of problems. Your containers are no different in that respect.

luzr
tvanriper
I've read and re-read this page, but I still can't quite see how U++ and std:: can be so incompatible that there's no hope of improving the std::-style system to the point of matching U++ performance.

Ah, but you could fix std::. But it is not likely to happen.

Moreover, adopting all U++ tricks into std:: would change its semantics and break existing code.

Bad choice of words on my part... a consequence of trying to write quickly.

By 'improving the std::style system', I didn't necessarily mean 'replace std:: containers'. I meant

that you could create another set of containers that feel std::-like, but have the qualities you prefer.

luzr
tvanriper
I only pose this idea because it feels to me like you and boost have similar goals. I could, of course, be wrong. I know, for example, that boost has less of an emphasis on performance and more of an emphasis on their idea of 'correctness', so you may differ significantly there. (This is certainly not to say you have no concern for 'correctness', but that you may have a slightly different idea of what is 'correct' from boost).

Oh, I have a very strong concern for 'correctness' - to the degree that I often rather break existing code by fixing some "incorrectness" in U++ Core.

Also, please, do not think I am not aware about boost or that I think these people are stupid. Of course not, boost is a very good effort and the code is pretty good.

I just feel U++ is not a good fit there. It is almost like suggesting boost to adopt Java

Well, firstly, I apologize if I seem pushy, or condescending. I suppose I just wanted this conversation to take place, just to fully explore the idea.

Secondly, while it might be true that this is like suggesting boost adopt Java, I wanted to at least explore the idea to see if that was really true. Other folks can eventually read all of this and see that we've covered all the points to cover on the subject.

At the very least, I thank you for your patience.

luzr
BTW: I mostly care about "optimality" with U++. If I would care about "popularity" more, I would certainly use another path and boost would be the part of it.

That is understandable.

luzr
tvanriper
Perhaps someone could submit an article to Dr. Dobb's Journal showcasing the use of Ultimate++; that's a fairly popular magazine, at least here in the United States (the CUJ folded to Dr. Dobb's a few years ago, sadly, or I would have recommended it instead).

I guess that would be much better idea:)

Hmmm... well, I need to buy some issues of Dr. Dobb's Journal, review the past articles, and see what sort of article they'd accept. I could probably write something... it's been a long time since I

wrote anything for a journalistic venue, but I could probably do it, if they'd accept the article.

Oh, and many thanks to mr_ped for pointing out the other link. I had read that a long time ago, but couldn't remember where I saw it.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Sun, 27 Apr 2008 14:31:44 GMT
View Forum Message <> Reply to Message

tvanriper wrote on Sat, 26 April 2008 07:41
Well, firstly, I apologize if I seem pushy, or condescending. I suppose I just wanted this conversation to take place, just to fully explore the idea.

It is all OK. Actually, it is not bad to reiterate all these issues from time to time.

Quote:
Hmmm... well, I need to buy some issues of Dr. Dobb's Journal, review the past articles, and see what sort of article they'd accept. I could probably write something... it's been a long time since I wrote anything for a journalistic venue, but I could probably do it, if they'd accept the article.

This would be excellent. I must admit I am a little bit short of energy to do something like this (or other forms of U++ propagation), besides, 3rd party article is always better.

If you would need any help (mostly with code , I would be glad to deliver

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by Novo on Sun, 27 Apr 2008 16:26:45 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 26 April 2008 02:11

BTW: I mostly care about "optimality" with U++. If I would care about "popularity" more, I would certainly use another path and boost would be the part of it.

I still do not get the strategy behind U++. You are saying you do not care about "popularity", but you have posted almost five thousand messages on this forum. It looks to me like you care very much about users and "popularity".

What "popularity" means to you? And how do you understand "optimality"?

As I understand, U++ is a self-contained application framework, which is planned with two goals in mind: performance and easiness to use.

Everything that doesn't match these goals gets ignored. That includes third-party libraries like boost, dll-based builds, makefiles for Unix and project files for MSVS.

1) Boost is not just containers and algorithms. It intersects with U++ at least in Boost.Thread, Boost.Spirit, Boost.Wave, and Boost.Function. But that is an external dependency.

2) Dll-based architecture seems to introduce some performance worsening. Automatic exporting of all classes and functions would impair optimization. Manual marking functions and classes for export would require some time, and there still be some performance worsening.

3) Project files / make files for other build systems would definitely improve popularity of U++. It could be included in popular Linux distributives. That doesn't seem to break either "popularity" or "optimality" principles. Without make files it is not possible to build U++ in regular way.

IMHO, before investing resources in making U++ more popular it is absolutely necessary to understand (and clearly explain to others) what actually U++ is, and what it is going to be. Otherwise you will just confuse new users.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Sun, 27 Apr 2008 16:49:33 GMT
View Forum Message <> Reply to Message

Novo wrote on Sun, 27 April 2008 12:26luzr wrote on Sat, 26 April 2008 02:11

BTW: I mostly care about "optimality" with U++. If I would care about "popularity" more, I would certainly use another path and boost would be the part of it.

I still do not get the strategy behind U++. You are saying you do not care about "popularity", but you have posted almost five thousand messages on this forum. It looks to me like you care very much about users and "popularity".

Eh, I wanted to say that "optimality" has priority. Not that I do not care about "popularity" at all. Sure I do.

Quote:
And how do you understand "optimality"?

Something that makes my life as programmer easier.

Quote:
As I understand, U++ is a self-contained application framework, which is planned with two goals in mind: performance and easiness to use.

Everything that doesn't match these goals gets ignored. That includes third-party libraries like boost, dll-based builds, makefiles for Unix and project files for MSVS.

I guess you got it right. Although most things are now ignored only because of lack of time/energy and it seems they are going to be supplied by community over time.

Quote:
IMHO, before investing resources in making U++ more popular it is absolutely necessary to understand (and clearly explain to others) what actually U++ is, and what it is going to be. Otherwise you will just confuse new users.

I agree. Note that this thread was revived after almost a year.

In fact I am in process of reviewing U++ strategy. We are now in quite different state than 4 years ago where basically there were 2 people in the world with deep knowledge of U++. This has changed quite a lot

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by royalstream on Tue, 17 Jun 2008 07:55:29 GMT
View Forum Message <> Reply to Message

Hi, I'm mostly a newcomer. I say mostly because I used UPP a few years ago for some small utility apps I needed.

With that said, I really can say my eyes are fresh and I can give you my humble opinion on my second first impression with UPP

First of all, I like TheIDE. Yes, its different. Yes, it looks different than Visual Studio or Eclipse. But it works and its fast and as long as I can plug-in different compilers I'm happy. My only suggestion: lets get some new icons TheIDE deserves to look as good as it works.

I love the way the API avoids unnecessary heap/pointer use. This very well may be the strongest reason why I like UPP. The naming convention is also clear and modern looking.

Love the String class and the NTL is fine, even though the naming of the classes (Vector/Array/etc) doesn't suggest how or why they differ from each other.

As you can see, I'm relatively open minded to new ways of doing things. So far the only characteristic I haven't been able to "like" is the frequent use of macros, especially the APP_MAIN ones.

I'm sure the GUI_APP_MAIN macro is awkward enough to scare away some curious newbies. Keep in mind "GUI_APP_MAIN" is about one fifth of what they see when the look at your Hello World's source code.

Obviously removing it at this point is no easy task but it could remain supported for backwards compatibility while a new way is offered. I think even having to include certain header in your main code file (depending on the type of application) would look more acceptable to some people. But that's just an quick idea, maybe there are even better ways of doing it.

Just my two cents and I'm not here to criticize, I really like UPP and I'd say we really "clicked". My true wish is for it to become as widely adopted as possible.

Regards,

Steven

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by unodgs on Tue, 17 Jun 2008 08:28:45 GMT
View Forum Message <> Reply to Message

Quote:First of all, I like TheIDE. Yes, its different. Yes, it looks different than Visual Studio or Eclipse. But it works and its fast and as long as I can plug-in different compilers I'm happy. My only suggestion: lets get some new icons TheIDE deserves to look as good as it works. TheIDE is our target right now. Soon it will get a docking system with new quicktabs (I have some nice ideas). Mirek is trying to improve C++ parser and Topic++. Don't worry about docking. It will be possible to turn off windows headers so everyone who likes thin splitters ant take care of screen space will be satisfied. As for icons I agree - especialy in assist. We need a good painter here. For assist we could use eclipse icons. I really like them.

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Tue, 17 Jun 2008 11:46:38 GMT

royalstream wrote on Tue, 17 June 2008 03:55
I'm sure the GUI_APP_MAIN macro is awkward enough to scare away some curious newbies. Keep in mind "GUI_APP_MAIN" is about one fifth of what they see when the look at your Hello World's source code.

Well, it is not very nice indeed, but one has to consider the only possible alternative:

```
#ifdef PLATFORM_WIN32
int APIENTRY WinMain(HINSTANCE hInstance, HINSTANCE, LPTSTR, lpCmdLine, int
nCmdShow) {
   InitWin32(hInstance, lpCmdLine);
#else
int main(int argc, const char **argv, const char **envptr) {
   InitPosix(argc, argv, envptr);
#endif
```

That is why GUI_APP_MAIN was introduced.

Mirek

P.S.: Concerning macros, I think it is good idea to avoid them when possible. Anyway, if they can avoid repetitive tasks (and that way often bugs too), I am not the one to avoid macros just because they are "ugly".

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by TeCNoYoTTa on Sat, 19 Jul 2008 17:47:02 GMT

i really dont understand why this library isnt popular

i used this library from about only one week and quickly i realized that this is the best library for my program that i want to make

---

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Sat, 19 Jul 2008 17:56:25 GMT

Well, note that the thread is 3 years old

(OTOH, lack of docs is still a problem. We have to focus hard on that now...)

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by TeCNoYoTTa on Sat, 19 Jul 2008 18:09:02 GMT
View Forum Message <> Reply to Message

OMG

i really dont understand where is the Ultimate++ fans ??

i think it's a good idea to make a video that demonstrate how easy is it to make C++ applications with Ultimate++ and Upload it on Youtube

and i am a student in Computer And Information science collage in AinShams University in Egypt and i think i can tell my friends and colleagues about it and i already told some teachers and friends in my collage about it

And i really want to thank every body contributed in this Great library

---

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by Didier on Sun, 30 Nov 2008 22:09:05 GMT
View Forum Message <> Reply to Message

Hello,

This is my first post on this forum.

I'm not used to doing GUIs since I mostly do firmware on specialized hardware but I find U++ amazing in it's concepts and ease of use.

The main drawback I see to U++ is its code editor and the apparent complexity to build a U++ graphical application without using the editor (sorry, I haven't looked very much).

I'm perfectly aware of the time and energy spent creating the code editor (and it's debuger) but it's far from being perfect and why create what already exists free and much more complete (I'm thinking about Eclipse CDT for example).
If Writing and debuging the code editor wasn't necessary, much more things could be done for the part that really matters, I 'meen:
- The U++ core library and concepts : FANTASTIC
- The Graphical editor and the class inheritance mechanism
- Additional evolutions / libraries.

My bet is that with few modification U++ could spread out very easily:

1 - Have an eclipse plugin that manages the graphical part.
(CDT is starting to be a very popular C/C++ code editor)

2 - Change the build process to something more classical (blitz++ is perfect as far as i'm concerned but when you try to convince you're boss or a work colleague the usual reaction is Uhhh? a specific compilation process ?!? Can we trust it? is it bug free ? Is there support for it ? An d finally they say : "Forget it". and I understand that position).
'Boost.build' could be a good candidate :it's free, multi-plateform, multi-tread, simple to use, has complete documentation.

But again The work accomplished is fantastic : great job !

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Sun, 30 Nov 2008 22:41:12 GMT
View Forum Message <> Reply to Message

Didier wrote on Sun, 30 November 2008 17:09
The main drawback I see to U++ is its code editor and the apparent complexity to build a U++ graphical application without using the editor (sorry, I haven't looked very much).

I understand the sentiment. However, theide was created for purpose.

The main focus was managing modular dependencies and do it across platforms. There is nothing comparable available AFAIK.

Quote:
I'm perfectly aware of the time and energy spent creating the code editor (and it's debuger) but it's far from being perfect and why create what already exists free and much more complete (I'm thinking about Eclipse CDT for example).
If Writing and debuging the code editor wasn't necessary, much more things could be done for the part that really matters, I 'meen:
 - The U++ core library and concepts : FANTASTIC
 - The Graphical editor and the class inheritance mechanism
 - Additional evolutions / libraries.

BTW, there was quite limited development of theide in 2007 and 2008. Maybe that is the reason why it lagged. It is the focus now, if you have not checked the latest svn version (note that current svn is on google-code), maybe do so first.

I see no tool able to solve problems theide was designed to solve.

Quote:

My bet is that with few modification U++ could spread out very easily:
1 - Have an eclipse plugin that manages the graphical part.
(CDT is starting to be a very popular C/C++ code editor)

2 - Change the build process to something more classical

You can switch BLITZ off with single click It is off for release builds by default anyway - nothing special about build process then.

You can also export the whole project with makefile and build that way. Frankly, I was forced to add this feature for the exact reason:

Quote:
(blitz++ is perfect as far as i'm concerned but when you try to convince you're boss or a work colleague the usual reaction is Uhhh? a specific compilation process ?!? Can we trust it? is it bug free ? Is there support for it ? An d finally they say : "Forget it". and I understand that position).

If they see the code buildable with makefile in svn repo, they do not care what tool I have used to edit it. Frankly, they do not even care that the project was created with U++ - it is just regular C++ code with no strings attached.

All that said - I will repeat myself, but I strongly support the idea of "library version". This was discussed here a couple of times. I am only not going to do it myself - for me, there is no benefit (except more popularity, of course). I am offering all the support for the project - but in the same time I have to say that currently I do not see how to solve some issues, especially how to regroup current highly modular structure of U++ packages into something more orthodox.

Mirek

P.S.: Maybe, after (eventually) checking the latest svn version, you can write a list of issues you dislike about theide. Even if you are not going to like the idea of its further development, I would like to see it.

Mirek

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by Didier on Thu, 04 Dec 2008 19:48:47 GMT
View Forum Message <> Reply to Message

OK thanks,

I compiled and launched successfully the latest svn version on FC8.
I had to modify the build scripts in order to get it running and find out what files where needed for it

to run correctly.

The main problems I found with the editor was :
- code completion only worked in main.cpp file of my app ???
  I never found out why ? (this was on 2007 version)
- when i passed to 2008.1 the same app doesn't compile any more
  I get some obscure syntax error and TheIde freezes !

Maybe I'll try to pass to boost.build style, after viewing some
generated makefiles it seems that the dependency problems and includes may just get resolved naturally.

Didier

Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by mirek on Thu, 04 Dec 2008 22:27:03 GMT
View Forum Message <> Reply to Message

Didier wrote on Thu, 04 December 2008 14:48OK thanks,

I compiled and launched successfully the latest svn version on FC8.
I had to modify the build scripts in order to get it running and find out what files where needed for it to run correctly.

The main problems I found with the editor was :
- code completion only worked in main.cpp file of my app ???
  I never found out why ? (this was on 2007 version)
- when i passed to 2008.1 the same app doesn't compile any more
  I get some obscure syntax error and TheIde freezes !

Maybe I'll try to pass to boost.build style, after viewing some
generated makefiles it seems that the dependency problems and includes may just get resolved naturally.

Didier

I have to say I am little bit confused about what was wrong with what version:)

Anyway, code completion, while still not perfect, should be greatly improved in latest svn.

Mirek
