Subject: Tarball issues
Posted by mirek on Mon, 09 Jan 2017 08:47:56 GMT
View Forum Message <> Reply to Message

amrein wrote on Mon, 09 January 2017 02:07I can change the domake script and force the use of clang++ instead of gcc if gcc version is lower than 4.9.0 for example.

I didn't found a quick fix for abs and other methods producing compilation errors with old gcc versions.

Well, I am not 100% what is the correct solution here, but

- we need to add to README that required gcc should be >= 5.0 (as older have problems with C++14 standard)

- maybe add warning (or even error) to domake

- we need to provide a way how to proceeed in that case, which is IMO:

  - install latest GCC from e.g. tarball, then provide a name of the compiler (like g++-6.1.0) to domake / make somehowe
  - install clang from distro package, as usually it is more OK with c++14 and again, change the compiler used by make
  - maybe install clang from tarball...

Now how to provide the name of compiler I am not 100++ sure, but IMO either environment variable/commandline switch?

Or maybe that warning in domake should try to pick the correct compiler and ask user? Let him choose?

Mirek

Subject: Re: Tarball issues
Posted by amrein on Mon, 09 Jan 2017 10:22:44 GMT
View Forum Message <> Reply to Message

This will force the use of clang++:

make -e CXX="clang++" -e CXXFLAGS="-O3 -ffunction-sections -fdata-sections -Wno-logical-op-parentheses -std=c++11"

[TODO] - Add to readme and web documentation: gcc should be >= 5.0 (as older versions have problems with C++14 standard).

[TODO] - Add to readme and web documentation: solutions like (1) install clang++ or (2) install latest GCC.
[TODO] - Add to readme and web documentation: how to force make to use those compilers from command line.
[TODO] - Add to domake: warnings and errors.
[TODO] - Add to domake: pick clang++ instead of gcc if gcc version is too old and if clang is available.

---

Subject: Re: Tarball issues
Posted by amrein on Tue, 10 Jan 2017 18:57:38 GMT
View Forum Message <> Reply to Message

[DONE] - Add to domake: pick clang++ instead of gcc if gcc version is too old and if clang is available. (in fact, I did a complete rewrite of domake)
[DONE] - Add to domake: warnings and errors.

But also:

[DONE] - Add to doinstall: several parameter options (in fact, I did a complete rewrite of doinstall).
[DONE] - Add to doinstall: standard POSIX installation.
[DONE] - Add to doinstall: warnings and errors.

I didn't commit those two scripts (domake and doinstall) in svn. I'm not sure if you (as the stable release manager) want me to, before stable release.
Please tell me.

Not done obviously:

[TODO] - Add to readme and web documentation: gcc should be >= 5.0 (as older versions have problems with C++14 standard).
[TODO] - Add to readme and web documentation: solutions like (1) install clang++ or (2) install latest GCC.
[TODO] - Add to readme and web documentation: how to force make to use those compilers from command line.

---

Subject: Re: Tarball issues
Posted by Klugier on Tue, 10 Jan 2017 20:09:39 GMT
View Forum Message <> Reply to Message

Hello amrein,

Quote:
[TODO] - Add to readme and web documentation: gcc should be >= 5.0 (as older versions have

problems with C++14 standard).

Our target c++ standard for this release is c++11, not c++14 and GCC versions lesser that 5.0 have got problems with the older one. Currently, only Android Builder supports c++14 by default. When, you will write documentation - please keep this digression in mind.

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by mirek on Tue, 10 Jan 2017 21:47:34 GMT
View Forum Message <> Reply to Message

Klugier wrote on Tue, 10 January 2017 21:09

Our target c++ standard for this release is c++11, not c++14 and GCC versions lesser that 5.0 have got problems with the older one. Currently, only Android Builder supports c++14 by default. When, you will write documentation - please keep this digression in mind.

Sincerely,
Klugier

Actually, it is sort of fuzzy: We are using some C++14 features that are already available in all conforming C++11 compilers...

GCC options are certainly set to C++14.

Mirek

---

Subject: Re: Tarball issues
Posted by mirek on Tue, 10 Jan 2017 21:49:20 GMT
View Forum Message <> Reply to Message

amrein wrote on Tue, 10 January 2017 19:57[DONE] - Add to domake: pick clang++ instead of gcc if gcc version is too old and if clang is available. (in fact, I did a complete rewrite of domake)
[DONE] - Add to domake: warnings and errors.

But also:

[DONE] - Add to doinstall: several parameter options (in fact, I did a complete rewrite of doinstall).
[DONE] - Add to doinstall: standard POSIX installation.

[DONE] - Add to doinstall: warnings and errors.

I didn't commit those two scripts (domake and doinstall) in svn. I'm not sure if you (as the stable release manager) want me to, before stable release.
Please tell me.

Please commit.

Mirek

---

Subject: Re: Tarball issues
Posted by amrein on Wed, 11 Jan 2017 03:54:51 GMT
View Forum Message <> Reply to Message

Committed.

If it works as it should (me testing my own code is not enough), I will modify the documentation accordingly.

---

Subject: Re: Tarball issues
Posted by Klugier on Wed, 11 Jan 2017 12:01:16 GMT
View Forum Message <> Reply to Message

Hello amrein,

While running - "make" command on Ubuntu 16.04 I came across following error:

/bin/sh domake
domake: 14: domake: Bad substitution

I think the problem is with 14 line of domake script:

script_name=${0^^}

When I removed that line I have got one more error:

/bin/sh domake
domake: 36: domake: Syntax error: "(" unexpected

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by amrein on Wed, 11 Jan 2017 14:15:42 GMT
View Forum Message <> Reply to Message

Thank you for that.

Causes:

- ${variable^^} uppercase substitution only work in bash
- function only work in bash
- echo is replaced by internal sh version that doesn't support escape character (so no color without calling /bin/echo explicitly)
- on most rpm based distribution, calling /bin/sh will always call /bin/bash and so none of those errors will ever occurs. :/

Fixed in SVN. Tested. Should be ok and ready for release.

Does it works for you now?

---

Subject: Re: Tarball issues
Posted by Klugier on Wed, 11 Jan 2017 16:35:19 GMT
View Forum Message <> Reply to Message

Hello,

The build process starts without error on latest SVN version of the script.

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by mirek on Wed, 11 Jan 2017 19:21:30 GMT
View Forum Message <> Reply to Message

Klugier wrote on Wed, 11 January 2017 17:35Hello,

The build process starts without error on latest SVN version of the script.

Sincerely,
Klugier

Does it finish too? Producing theide?

Mirek

---

Subject: Re: Tarball issues
Posted by amrein on Wed, 11 Jan 2017 21:09:03 GMT
View Forum Message <> Reply to Message

'make' and 'make install' do work (at least for me) on Debian and on a few rpm based distributions.

More feedback will be awesome. I couldn't test BSD and alike.

If you want to compile faster (not as fast as BLITZ) and install somewhere without messing up your current installation:

```
if which nproc
then
  make -j$(nproc)
else
  make
fi

make install "DESTDIR=$HOME/UPP-TEMP"
```

It will install U++ in $HOME/UPP-TEMP/$HOME as if it was your home directory.

If you want to see the POSIX installation in action:

```
if which nproc
then
  make -j$(nproc)
else
  make
fi

make install "DESTDIR=$HOME/UPP-TEMP" prefix=/usr/local
```

---

Subject: Re: Tarball issues
Posted by amrein on Wed, 11 Jan 2017 21:42:51 GMT
View Forum Message <> Reply to Message

Question:

- When creating POSIX binary packages, where should I copy the dictionaries (like en-gb.scd and en-us.scd) for theide to pick them?

-> I can't put them in /usr/bin, even if I know that it will work.
-> theide doesn't find them in /usr/share/upp nor does it in /usr/share/upp/scd and I can't figure it out (where does theide search for those files at first start...).

Could, but didn't help:
https:// sourceforge.net/projects/upp/files/SpellerDictionaries/Aspel l/

Subject: Re: Tarball issues
Posted by dolik.rce on Thu, 12 Jan 2017 06:47:21 GMT
View Forum Message <> Reply to Message

Hi amrein,
amrein wrote on Wed, 11 January 2017 22:42Question:

- When creating POSIX binary packages, where should I copy the dictionaries (like en-gb.scd and en-us.scd) for theide to pick them?

-> I can't put them in /usr/bin, even if I know that it will work.
-> theide doesn't find them in /usr/share/upp nor does it in /usr/share/upp/scd and I can't figure it out (where does theide search for those files at first start...).

Same problem as with the default GCC.bm... IIRC, I ended up putting it in /usr/share and copying it to $HOME/.upp/theide/ in the install step, which launches when the user first starts TheIDE. The dictionaries will be picked up by TheIDE when placed into the config directory. The unpleasant thing about this is, that it requires makeing this in code in TheIDE itself...

It would by great if U++ or at least TheIDE supported having global configuration in /etc and separate user configs that would overload the global settings. This is kind of standard behavior in posix world... But I think it would be pretty big change.

Best regards,
Honza

Subject: Re: Tarball issues

Posted by mirek on Thu, 12 Jan 2017 07:49:26 GMT
View Forum Message <> Reply to Message

amrein wrote on Wed, 11 January 2017 22:42
-> theide doesn't find them in /usr/share/upp nor does it in /usr/share/upp/scd and I can't figure it out (where does theide search for those files at first start...).

Core/Speller.cpp 149

I have added some for POSIX:

```
Speller *sGetSpeller(int lang)
{
 static ArrayMap<int, Speller> speller;
 int q = speller.Find(lang);
 if(q < 0) {
  String pp = spell_path;
  DoSpellerPath(pp, GetExeDirFile("scd"));
  DoSpellerPath(pp, ConfigFile("scd"));
  pp << spell_path << ';' << getenv("LIB") << ';' << getenv("PATH") << ';';
#ifdef PLATFORM_POSIX
  pp << "/usr/share/upp;/usr/local/share/upp;/usr/share/upp/scd;/usr/local/share/upp/scd";
#endif
  String path = GetFileOnPath(ToLower(LNGAsText(lang)) + ".udc", pp);
```

Subject: Re: Tarball issues
Posted by Tom1 on Thu, 12 Jan 2017 08:10:55 GMT
View Forum Message <> Reply to Message

Hi Amrein,

I can confirm a successful build and install from source tarball 10699M using the instructions (make and make install) in readme file on Linux Mint 18.1 (based on Ubuntu). It was a clean install on a machine which did not have even gcc installed. And actually I left gcc out intentionally to test the tarball with clang which worked out of the box just as expected.

Best regards,

Tom

Subject: Re: Tarball issues

Posted by Tom1 on Thu, 12 Jan 2017 11:27:34 GMT
View Forum Message <> Reply to Message

Hi,

Tested on a freshly installed TrueOS (based on FreeBSD).

After installing llvm39 and libnotify, make starts working. Making theide fails but umk gets built.

...
make[1]: don't know how to make ide/app.tpp/Aboutn-us.tppi. Stop

make[1]: stopped in /usr/home/tom/upp-x11-src-10699M/uppsrc

When compiling, strangely, the flags defined in compilation include flagGCC and flagLINUX, while compiling with CLANG on FreeBSD:
...
clang++ -c -x c++ -O3 -ffunction-sections -fdata-sections -Wno-logical-op-parentheses -std=c++11 -I./ -I/usr/local/include/gtk-2.0 -I/usr/local/include/pango-1.0 -I/usr/local/include/glib-2.0 -I/usr/local/lib/glib-2.0/include -I/usr/local/include -I/usr/local/include/cairo -I/usr/local/include/pixman-1 -I/usr/local/include/freetype2 -I/usr/local/include/libdrm -I/usr/local/include/libpng16 -I/usr/local/include/harfbuzz -I/usr/local/include/gdk-pixbuf-2.0 -I/usr/local/include/atk-1.0 -I/usr/include/freetype2 -I/usr/include/gtk-2.0 -I/usr/include/glib-2.0 -I/usr/lib/glib-2.0/include -I/usr/lib/gtk-2.0/include -I/usr/include/cairo -I/usr/include/pango-1.0 -I/usr/include/atk-1.0 -I/usr/X11R6/include -I/usr/X11R6/include/freetype2 -I/usr/X11R6/include/gtk-2.0 -I/usr/X11R6/include/glib-2.0 -I/usr/X11R6/lib/glib-2.0/include -I/usr/X11R6/lib/gtk-2.0/include -I/usr/X11R6/include/cairo -I/usr/X11R6/include/pango-1.0 -I/usr/X11R6/include/atk-1.0 -I/usr/include/gdk-pixbuf-2.0 -I/usr/include/gtkglext-1.0 -I/usr/lib/gtkglext-1.0/include -I/usr/lib/i386-linux-gnu/glib-2.0/include -I/usr/lib/x86_64-linux-gnu/glib-2.0/include -I/usr/lib/i386-linux-gnu/gtk-2.0/include -I/usr/lib/x86_64-linux-gnu/gtk-2.0/include -DflagGUI -DflagMT -DflagGCC -DflagSHARED -DflagLINUX -DflagPOSIX -DflagMAIN ide/BaseDlg.cpp -o _out/ide//home/cxl/Scripts/GCCMK.bm-Gcc-Gui-Linux-Main-Mt-Posix-Shared/BaseDlg.o
...

Using umk I tried to rebuild theide but linking fails at -lfontconfig. Fontconfig is installed, but does not get linked. The library resides below /usr/local/lib, but only /usr/lib is included in library path.

Best regards,

Tom

Subject: Re: Tarball issues
Posted by amrein on Thu, 12 Jan 2017 15:41:07 GMT
View Forum Message <> Reply to Message

-> Thank Tom1.
So:
[ OK ] - Linux Mint 18.1
[ KO ] - TrueOS

Fontconfig in /usr/local? Manual install?
If possible, can you post the output on TrueOS of those three commands:

pkg-config --variable pcfiledir gtk+-2.0
pkg-config --variable pcfiledir fontconfig
cat "$(pkg-config --variable pcfiledir fontconfig)/fontconfig.pc"

My guess: pkg-config won't find fontconfig and "fontconfig.pc" is in /usr/local/lib*/pkgconfig/

-> Mirek: I tested with the new Core/Speller.cpp. It didn't work. Build Falgs: -DflagGCC -DflagSHARED -DflagLINUX -DflagPOSIX
There is somewhere in U++ ide package the code coping files and folders from /usr/share/upp/. This is what I fail to find. Because there I would be able to not just copy *.scd but instead move them to ~/.upp/theide.
I guess the good directory name could be /usr/share/upp/scd-usd/ or something similar.

-> dolik.rce: I agree. Great point. Hard-coded path can easily break because POSIX OS are moving targets.
This kind of thing needs coordination between packagers and developers as U++ is cross-platform.

---

Subject: Re: Tarball issues
Posted by Klugier on Thu, 12 Jan 2017 20:17:57 GMT
View Forum Message <> Reply to Message

Hello,

Yes, it compiles fine on my Ubuntu 16.04 machine, but at the end I have got following output:

```
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/Builders.a \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/CppBuilder.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/MakeFile.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/GccBuilder.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/MscBuilder.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/JavaBuilder.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/ScriptBuilder.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidProject.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidApplicationMak
eFile.o \
 _out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidMakeFile.o \
```

_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidModuleMakeFile.o \
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidBuilder.o \
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidBuilderCommands.o \
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidBuilderUtils.o \
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/AndroidModuleMakeFileBuilder.o \
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/Blitz.o \
_out/ide/Builders//home/cxl/Scripts/GCCMK.bm-Gcc-Linux-Posix-Shared/Build.o

And, I have one question - why this directory is appended "/home/cxl/". Am I cxl? This is probably cosmetic issue, but should be fixed.

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by amrein on Thu, 12 Jan 2017 22:06:03 GMT
View Forum Message <> Reply to Message

You have "/home/cxl/" appended because Makefile.in and uMakefile.in were built on the U++ server using umk.
The corresponding script is svn/trunk/uppbox/Scripts/src.

Those lines are what you are looking for:

Quote:
cp -p ~/Scripts/GCCMK.bm ~/.upp/theide

~/bin/umk ~/upp.src/uppsrc ide ~/Scripts/GCCMK.bm -asrXk ~/upp.tmp/upp/uppsrc

mv ~/upp.tmp/upp/uppsrc/Makefile ~/upp.tmp/upp/uppsrc/Makefile.in

~/bin/umk ~/upp.src/uppsrc umk ~/Scripts/GCCMK.bm -asrXk ~/upp.tmp/upp/uppsrc

mv ~/upp.tmp/upp/uppsrc/Makefile ~/upp.tmp/upp/uppsrc/uMakefile.in

Those modifications should fix it:

Quote:
cp -p ~/Scripts/GCCMK.bm ~/.upp/theide

~/bin/umk ~/upp.src/uppsrc ide GCCMK -asrXk ~/upp.tmp/upp/uppsrc

mv ~/upp.tmp/upp/uppsrc/Makefile ~/upp.tmp/upp/uppsrc/Makefile.in

~/bin/umk ~/upp.src/uppsrc umk GCCMK -asrXk ~/upp.tmp/upp/uppsrc

mv ~/upp.tmp/upp/uppsrc/Makefile ~/upp.tmp/upp/uppsrc/uMakefile.in

Committing...

---

Subject: Re: Tarball issues
Posted by Tom1 on Fri, 13 Jan 2017 09:19:20 GMT
View Forum Message <> Reply to Message

Hi amrein,

It was a fresh install and I have not installed anything 'manually'. I just had to install libnotify and llvm39, but did it with the default package manager.

* Please note that working on TrueOS, or *BSD in general, is not a critical issue for me at the moment. Just wanted to help you in testing. *

Here's the output of commands you asked for:

[tom@trueos-7040] ~% pkg-config --variable pcfiledir gtk+-2.0
/usr/local/libdata/pkgconfig
[tom@trueos-7040] ~% pkg-config --variable pcfiledir fontconfig
/usr/local/libdata/pkgconfig
[tom@trueos-7040] ~% cat "$(pkg-config --variable pcfiledir fontconfig)/fontconfig.pc"
Illegal variable name.
[tom@trueos-7040] ~%

Just for reference, here's the result of locate fontconfig.

[tom@trueos-7040] ~% locate fontconfig
/usr/local/include/fontconfig
/usr/local/include/fontconfig/fcfreetype.h
/usr/local/include/fontconfig/fcprivate.h
/usr/local/include/fontconfig/fontconfig.h
/usr/local/include/qt5/QtPlatformSupport/5.6.2/QtPlatformSupport/private/qfontconfigdatabase_p.h
/usr/local/include/qt5/QtPlatformSupport/5.6.2/QtPlatformSupport/private/qfontenginemultifontconfig_p.h
/usr/local/lib/girepository-1.0/fontconfig-2.0.typelib
/usr/local/lib/libfontconfig.a
/usr/local/lib/libfontconfig.so
/usr/local/lib/libfontconfig.so.1
/usr/local/lib/libfontconfig.so.1.9.2

/usr/local/libdata/pkgconfig/fontconfig.pc
/usr/local/pkg-cache/fontconfig-2.12.1,1-581b37418c.txz
/usr/local/pkg-cache/fontconfig-2.12.1,1.txz
/usr/local/share/doc/fontconfig
/usr/local/share/doc/fontconfig/fontconfig-user.html
/usr/local/share/doc/fontconfig/fontconfig-user.pdf
/usr/local/share/doc/fontconfig/fontconfig-user.txt
/usr/local/share/gir-1.0/fontconfig-2.0.gir
/usr/local/share/licenses/fontconfig-2.12.1,1
/usr/local/share/licenses/fontconfig-2.12.1,1/catalog.mk
/usr/local/share/licenses/fontconfig-2.12.1,1/LICENSE
/usr/local/share/licenses/fontconfig-2.12.1,1/MIT
/var/db/fontconfig
/var/db/fontconfig/1fe06f2352c4cb897117a756a050a9d4-le64.cache-7
/var/db/fontconfig/317e92aa1a4d3e601fb38e2d3c7b366a-le64.cache-7
/var/db/fontconfig/4c599c202bc5c08e2d34565a40eac3b2-le64.cache-7
/var/db/fontconfig/5590eef8711d78f75a1d19f78ae9af8f-le64.cache-7
/var/db/fontconfig/830bb1cbfd3d582459af9eb69ef1dd53-le64.cache-7
/var/db/fontconfig/b505adbf72d7253408dd67084a8aa967-le64.cache-7
/var/db/fontconfig/ba1d92d9e40780c65c2952558e6fa6f5-le64.cache-7
/var/db/fontconfig/bd930d03fc0401d0f1dcfa7b9bdc1687-le64.cache-7
/var/db/fontconfig/CACHEDIR.TAG
/var/db/fontconfig/d3b21a501470a17bfd0b9b6aedc735bd-le64.cache-7

/var/db/fontconfig/ece4193fc5c4a4effb1dc60970b2e31e-le64.cache-7

Finally, please note that the flags flagGCC and flagLINUX were erroneously active, while I assume they should have been flagCLANG and flagBSD.

Best regards,

Tom

---

Subject: Re: Tarball issues
Posted by amrein on Sat, 14 Jan 2017 21:18:56 GMT
View Forum Message <> Reply to Message

What I found:
After having installed all dependencies:

[1]- Make will stop with error:

make[1]: don't know how to make ide/app.tpp/Aboutn-us.tppi. Stop

It's because make from BSD try to handle '$$' in Makefiles like the beginning of variable (About$$en-us.tppi become Aboutn-us.tppi, ...).
Several lines will crash in Makefile.in because of '$$'.

If I'm correct, the perfect way to escape '$', according to internet forums, is to use four '$'.
Yes, four '$' meaning '$' become '$$$$'.

Solution: modify umk source to double escape $ characters.

[2]- Then make will stop with another error:

/usr/bin/ld: cannot find -lfreetype
clang-3.9: error: linker command failed with exit code 1 (use -v to see invocation)
*** Error code 1

It's because freetype libraries on BSD are named differently (libttf.so* instead of libfreetype.so*). I found that running 'pkg info -l freetype'.

Solution: well, I don't know. There is no pkg-config file for freetype in most distributions (no freetype.pc I mean). The only way to handle this efficiently would be to switch to GNU Autoconf and this is not as easy as it sound.

---

Subject: Re: Tarball issues
Posted by amrein on Sat, 14 Jan 2017 21:35:36 GMT
View Forum Message <> Reply to Message

In fact it's 'pkg-config --libs gtk+-2.0' that output '-lfreetype'
-> This is a TrueOS configuration bug. Not a U++ issue then.

---

Subject: Re: Tarball issues
Posted by Klugier on Sat, 14 Jan 2017 21:40:41 GMT
View Forum Message <> Reply to Message

Hello,

Quote:
Solution: well, I don't know. There is no pkg-config file for freetype in most distributions (no freetype.pc I mean). The only way to handle this efficiently would be to switch to GNU Autoconf and this is not as easy as it sound.

I think it would be not problem to modify Draw package to link against libttf when BSD or FREEBSD is detected. Dose it happen on all BSD distribution or it is related only to True OS?

_____ _____
You can try to modify linking libs in Draw package organizer. This is shown on below screenshot:

Below is the list of all flags that we can handle in package organizer (ide/Core/Host.cpp - line

316):

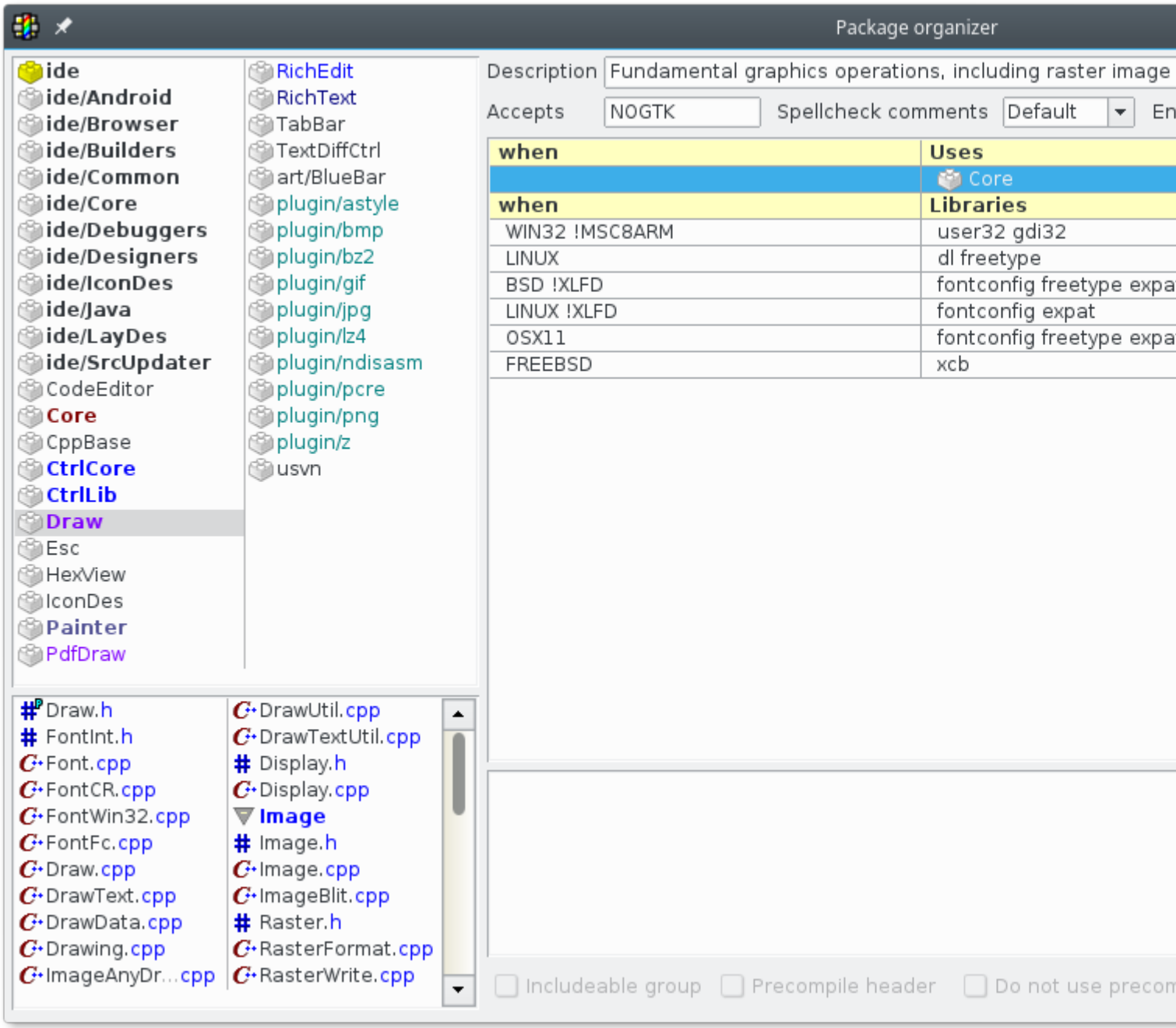
```
void LocalHost::AddFlags(Index<String>& cfg)
{
#if   defined(PLATFORM_WIN32)
 cfg.Add("WIN32");
#endif

#ifdef PLATFORM_LINUX
 cfg.Add("LINUX");
#endif

#ifdef PLATFORM_POSIX
 cfg.Add("POSIX");
#endif

#ifdef PLATFORM_BSD
 cfg.Add("BSD");
#endif

#ifdef PLATFORM_FREEBSD
 cfg.Add("FREEBSD");
#endif

#ifdef PLATFORM_OPENBSD
 cfg.Add("OPENBSD");
#endif

#ifdef PLATFORM_NETBSD
 cfg.Add("NETBSD");
#endif

#ifdef PLATFORM_DRAGONFLY
 cfg.Add("DRAGONFLY");
#endif

#ifdef PLATFORM_SOLARIS
 cfg.Add("SOLARIS");
#endif

#ifdef PLATFORM_OSX11
 cfg.Add("OSX11");
#endif
}
```

So, as you can see we can make hacks for all BSD related system and only for that using FREE BSD kernel. Dependency in this case is BSD > FREEBSD.

Sincerely,
Klugier

## File Attachments

1) Screenshot_20170114_222728.png, downloaded 1032 times




---

Subject: Re: Tarball issues
Posted by amrein on Sat, 14 Jan 2017 21:50:53 GMT
View Forum Message <> Reply to Message

I understand but we shouldn't. This is a TrueOS issue. What can be done is reporting that pkg-config for gtk doesn't give the right library name for freetype and add that freetype libraries use wrong names.
Other than that, we shouldn't interfere. Tomorrow, this will certainly be fixed and we would have to change it again and again.

---

Subject: Re: Tarball issues
Posted by Klugier on Sat, 14 Jan 2017 21:53:20 GMT
View Forum Message <> Reply to Message

amrein wrote on Sat, 14 January 2017 22:35In fact it's 'pkg-config --libs gtk+-2.0' that output '-lfreetype'
-> This is a TrueOS configuration bug. Not a U++ issue then.

Hello,

I think we can modify Core to support flag PLATFORM_TRUE_OS (Core/config.h - line 37):

```
#if __FreeBSD__ || __OpenBSD__ || __NetBSD__ || __DragonFly__
  #define PLATFORM_BSD 1
  #if __FreeBSD__
   #define PLATFORM_FREEBSD 1
  #endif
  #if __OpenBSD__
   #define PLATFORM_OPENBSD 1
  #endif
  #if __NetBSD__
   #define PLATFORM_NETBSD 1
  #endif
  #if __DragonFly__
   #define PLATFORM_DRAGONFLY 1
  #endif
```

After doing this, we are going to modify ide/Builders (Please read my post above) and the we can make linking hack for the TRUE_OS. Is the True OS the most FreeBSD popular distro?

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by Tom1 on Sat, 14 Jan 2017 22:21:56 GMT
View Forum Message <> Reply to Message

Hi both,

Thanks for looking into this issue.

From my point of view TrueOS is just a convenient way to quickly install FreeBSD with a usable desktop. Klugier: Please note that the BSD world is not anywhere nearly as fragmented as Linux world is. If FreeBSD is supported, the coverage is already more than adequate. OpenBSD support would be nice, but likely out of reach for now. Please read more on that below.

Supporting TrueOS as a separate system from FreeBSD is in my opinion both unnecessary and wasting energy. If TrueOS differs from FreeBSD in a bad way, just forget TrueOS. However, it would be nice to have FreeBSD working. When I get to the office on Monday, I will try to install a clean FreeBSD and test with that instead of TrueOS. (Or maybe someone have done it already???)

--

I also did a test installation of OpenBSD on Friday, but first found out that the default gcc version on OpenBSD is more or less about ten years old (4.2.x) and no way it's going to support C++11. Also, after installing clang -- which is eventually going to replace gcc on OpenBSD too -- I ran in trouble with problems on thread local storage not being supported. At least that's what the clang compiler thought when compiling Core. I could not really figure out if this is really the case with OpenBSD, or just a clang compatibility issue, but never the less, Upp is not going to work on OpenBSD without substantial effort I guess.

Thanks and best regards,

Tom

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Sat, 14 Jan 2017 23:47:58 GMT
View Forum Message <> Reply to Message

amrein wrote on Sat, 14 January 2017 21:18It's because freetype libraries on BSD are named differently (libttf.so* instead of libfreetype.so*). I found that running 'pkg info -l freetype'.
I guess, that you installed outdated version of FreeType from print/freetype FreeBSD port, which has libttf.so library. The libfreetype.so library installed by print/freetype2 FreeBSD port. Example for v2.7.1:
% pkg info -l freetype2 | grep /lib
 /usr/local/lib/libfreetype.a
 /usr/local/lib/libfreetype.so
 /usr/local/lib/libfreetype.so.6
 /usr/local/lib/libfreetype.so.6.13.0
 /usr/local/libdata/pkgconfig/freetype2.pc
% pkg-config freetype2 --libs
-L/usr/local/lib -lfreetype

As I understood, the exported Makefile (including for nightly builds) is done for Linux (with flagLINUX). The exported Makefile for FreeBSD (with flagBSD and flagFREEBSD) has other contents. Take a look at configuration of current revision of uppsrc/CtrlCore package (or with using TheIDE), for example:
Toggle excerpt from uppsrc/CtrlCore/CtrlCore.upp file

```
library(WIN32 !MSC8ARM) "advapi32 comdlg32 comctl32";

library((LINUX | BSD) & !NOGTK & !RAINBOW & !GTK3 & GUI) "gtk-x11-2.0  gdk-x11-2.0  atk-1.0  gdk_pixbuf-2.0  m  pangocairo-1.0  fontconfig  Xext  Xrender  Xinerama  Xi  Xrandr  Xcursor  Xfixes pango-1.0  cairo  X11  gobject-2.0  gmodule-2.0  glib-2.0";

library((LINUX | BSD) & GTK3) "gtk-3.0  gdk-3.0  atk-1.0  gdk_3.0  m  pangocairo-1.0  fontconfig  Xext  Xrender  Xinerama  Xi  Xrandr  Xcursor Xfixes pango-1.0  cairo  X11  gobject-2.0  gmodule-2.0  glib-2.0";

library(WIN32 !MSC8ARM) "user32 gdi32";

library(LINUX !RAINBOW GUI) "X11 Xrender Xinerama";

library(LINUX  !RAINBOW) dl;

library(LINUX !XLFD  !RAINBOW GUI) Xft;

library(BSD !RAINBOW) "X11  Xau Xdmcp";

library(BSD !XLFD  !RAINBOW) "Xft fontconfig Xrender freetype expat";

library(LINUX !XLFD !SHARED  !RAINBOW GUI) "fontconfig Xrender freetype";

library(OSX11) "X11 Xft fontconfig Xrender freetype expat";

library(FREEBSD | DRAGONFLY) xcb;

library(POSIX !NOGTK !RAINBOW GUI) notify;

library(DRAGONFLY) "Xext Xinerama";
```

Perhaps, you trying to create some universal solution and already know about this information.

There is a devel/upp FreeBSD port, which used for installation of release version. If you look at its contents, you may understand, that it adapts exported Makefile for FreeBSD, uses related configuration flags and generate files for build methods.

There is also a devel/upp-nightly port on the forum for some nightly revision, which was created for other purposes. I updated it to current 10703 revision and it builds fine (on FreeBSD 10.3).

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 16:47:12 GMT
View Forum Message <> Reply to Message

Well, great point. Freetype2 was already there. So I only had to add its path to LIBPATH in Makefile.in

LIBPATH = -L"/usr/X11R6/lib" -L"/usr/lib" -L"/usr/local/lib"

To summarize, from a clean install, here what I had to do to compile and run theide:

```
# As root, install all available updates
su
pc-updatemanager pkgupdate
# then reboot
shutdown -r now

# Install gtk2, freetype2, libnotify and llvm39 (clang++):
su
pkg install gtk2
pkg install freetype2
pkg install libnotify
pkg install llvm39
exit

# As standard user:
tar zxvf upp-x11-src-10703.tar.gz
cd upp-x11-src-10703
sed -i -e 's|LIBPATH = -L"/usr/X11R6/lib" -L"/usr/lib"|LIBPATH = -L"/usr/X11R6/lib" -L"/usr/lib"
-L"/usr/local/lib"|g' uppsrc/Makefile.in
sed -i -e 's|\$\$|$$$$|g' uppsrc/Makefile.in
make
```

I don't know how to deal with flagBSD. I can't find it in U++ snapshots or in SVN. What I would like to see is packagers creating standard POSIX packages using as much as possible standardized installation procedures. I mean:
- spec file for rpm based distribution -> done
- debian source package for debian based distributions (standard source directory using debuild and not from binary using dpkg-deb) -> todo
- BSD like package -> todo (something similar as http://www.freshports.org/devel/upp)
- Easy terminal compilation (make, make install) -> partially done (works on Linux but will fail on BSD distributions without tricks)
- Windows package

Should we create a new topic where all packagers with svn access will share how they would like to build all this?

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 16:50:26 GMT
View Forum Message <> Reply to Message

I'm really a bad C++ programmer. I couldn't find where U++ duplicate the '$' character in file name when it create Makefiles.
Really, I need to improve my C++ skill.

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 20:32:40 GMT
View Forum Message <> Reply to Message

Klugier, I'm modifying a few scripts in Scripts directory. That way, U++ will build out of the box on BSD distributions.
But I still have a big problem: the need to fix Makefile generation with umk.

For now on, I do a manual fix with before running make:
sed -i -e 's|\$\$|$$$$|g' uppsrc/Makefile.in

I don't know how to fix umk. It needs to not just duplicate but quadruplicate the '$' character found in file names when it creates Makefile.

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 20:37:49 GMT
View Forum Message <> Reply to Message

Sender Ghost, I don't know an easy way to know that we are running on BSD like distributions.

'uname -s' will give me the operating system name, but there are a lot of BSD fork out there:
https://en.wikipedia.org/wiki/List_of_BSD_operating_systems
and most of them don't even have BSD in their name.

I tried flagLINUX and flagBSD. Both configuration will compile up to the end (if the '$' bug in Makefile is fixed).

Subject: Re: Tarball issues
Posted by Klugier on Sun, 15 Jan 2017 20:38:27 GMT
View Forum Message <> Reply to Message

Hello,

In Makefile.in in 4 line there is following code:

Macro =  -DflagGUI -DflagMT -DflagGCC -DflagSHARED -DflagLINUX -DflagPOSIX

Then when we detects that our current operating system is BSD, the following line should have following form

Macro =  -DflagGUI -DflagMT -DflagGCC -DflagSHARED -DflagBSD -DflagPOSIX

Can we add there if and detects operating system in makefile. I am not big makefile expert, so I would like to ask?

To detect that we are running on BSD we can call following command:

[www@cb.vu]~> uname -a

FreeBSD cb.vu 7.1-STABLE FreeBSD 7.1-STABLE #2: Wed Jan 30 16:21:05 CET 2009 c@cb.vu:/usr/obj/usr/src/sys/CB  i386

Alterantivly, we can do exactly like we do on Core, so detects with 'uname -s' that the operating system is Linux and treat all others like BSD. What do you think Amrein?

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by Klugier on Sun, 15 Jan 2017 21:13:37 GMT
View Forum Message <> Reply to Message

One more alternative - using compiler version to detects operating system especially BSD:

For clang try:

```
clang++ -v
clang version 3.8.0-2ubuntu4 (tags/RELEASE_380/final)
Target: x86_64-pc-linux-gnu
Thread model: posix
InstalledDir: /usr/bin
Found candidate GCC installation: /usr/bin/../lib/gcc/i686-linux-gnu/5.4.0
Found candidate GCC installation: /usr/bin/../lib/gcc/i686-linux-gnu/6.0.0
Found candidate GCC installation: /usr/bin/../lib/gcc/x86_64-linux-gnu/5.4.0
Found candidate GCC installation: /usr/bin/../lib/gcc/x86_64-linux-gnu/6.0.0
Found candidate GCC installation: /usr/lib/gcc/i686-linux-gnu/5.4.0
Found candidate GCC installation: /usr/lib/gcc/i686-linux-gnu/6.0.0
Found candidate GCC installation: /usr/lib/gcc/x86_64-linux-gnu/5.4.0
Found candidate GCC installation: /usr/lib/gcc/x86_64-linux-gnu/6.0.0
```

Selected GCC installation: /usr/bin/../lib/gcc/x86_64-linux-gnu/5.4.0
Candidate multilib: .;@m64
Selected multilib: .;@m64

For GCC try:

g++ -v
Using built-in specs.
COLLECT_GCC=g++
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/5/lto-wrapper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1~16.04.4'
--with-bugurl=file:///usr/share/doc/gcc-5/README.Bugs
--enable-languages=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix=/usr --program-suffix=-5
--enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext
--enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu
--enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new
--enable-gnu-unique-object --disable-vtable-verify --enable-libmpx --enable-plugin
--with-system-zlib --disable-browser-plugin --enable-java-awt=gtk --enable-gtk-cairo
--with-java-home=/usr/lib/jvm/java-1.5.0-gcj-5-amd64/jre --enable-java-home
--with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-5-amd64
--with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-5-amd64 --with-arch-directory=amd64
--with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --enable-multiarch --disable-werror
--with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib
--with-tune=generic --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu
--target=x86_64-linux-gnu
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.4)

As you may notice, we are interesting in target field. For clang ("Target: x86_64-pc-linux-gnu") and for GCC "--target=x86_64-linux-gnu". Maybe it will help.

---

Subject: Re: Tarball issues
Posted by Klugier on Sun, 15 Jan 2017 21:33:42 GMT
View Forum Message <> Reply to Message

amrein wrote on Sun, 15 January 2017 21:32Klugier, I'm modifying a few scripts in Scripts directory. That way, U++ will build out of the box on BSD distributions.
But I still have a big problem: the need to fix Makefile generation with umk.

For now on, I do a manual fix with before running make:
sed -i -e 's|\$\$|$$$$|g' uppsrc/Makefile.in

I don't know how to fix umk. It needs to not just duplicate but quadruplicate the '$' character found

in file names when it creates Makefile.

Hello,

The MakeFile code base is hidden in following file ide/Builders/MakeFile.cpp. And probably you are interested in following function (line 11):

static String MakeSourcePath(const Vector<String>& dirs, String fn, bool raw, bool exporting)

Try to play with it - for example replace in fn. If it starts working just post your patch here - to discuss it before committing. You know how to build UMK from TheIDE?

I can try to modify this cpp file to support BSD, but probably I could do it until the next Saturday

Sincerely,
Klugier

---

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 21:59:34 GMT
View Forum Message <> Reply to Message

Thank

I found what I need in uppsrc/ide/Core/Core.cpp.
The patch:

```
--- uppsrc/ide/Core/Core.cpp.OLD        2016-10-15 16:42:07.000000000 +0200
+++ uppsrc/ide/Core/Core.cpp    2017-01-15 22:57:18.174354625 +0100
@@ -78,7 +78,7 @@
        String out;
        for(; *fn; fn++)
                if(*fn == '$')
-                       out << '$' << '$';
+                       out << "$$$$";
                else
                        out << *fn;
        return out;
```

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Sun, 15 Jan 2017 22:09:33 GMT
View Forum Message <> Reply to Message

amrein wrote on Sun, 15 January 2017 20:32For now on, I do a manual fix with before running

make:
sed -i -e 's|\$\$|$$$$|g' uppsrc/Makefile.in

I don't know how to fix umk. It needs to not just duplicate but quadruplicate the '$' character found in file names when it creates Makefile.

I tried following sed patch instead:

% sed -i '.bak' -e '/.tpp/s|\$|$$|g' Makefile

But this makes Makefile incompatible with GNU make (gmake on FreeBSD):

% gmake
<...>
gmake: *** No rule to make target 'ide/app.tpp/About$$en-us.tppi', needed by '_out/ide/CLANG-Bsd-Clang-Freebsd-Gui-Main-Mt-Posix-Shared/About.o'.  Stop.

I guess, this is not a solution for current (umk/theide) generator of exported Makefile, because on other platforms GNU make may be used. Otherwise, possible to generate BSD and GNU compatible make files with different names: BSDmakefile and GNUmakefile. The GNU make may use GNUmakefile and BSD make may use BSDmakefile first.

The mentioned issue was partially fixed with using compiler generated dependency information, some patch for which I attached in another topic (as a fix for another issue). For example, Clang compiler (on FreeBSD 10.3) escapes "$" character(s) to "$$" for dependencies:
% touch build_info.h
% c++ -MM -c -pipe -Wno-logical-op-parentheses -std=c++11 -DflagGUI -DflagMT -DflagCLANG -DflagSHARED -DflagPOSIX -DflagBSD -DflagFREEBSD -DflagNOGTK -I./ `pkg-config freetype2 -cflags` ide/About.cpp | grep .tpp
  Core/topic_group.h ide/app.tpp/all.i ide/app.tpp/About$$en-us.tppi \
  ide/app.tpp/AdvancedReplace$$en-us.tppi \
  ide/app.tpp/AndroidBuilder$$en-us.tppi ide/app.tpp/Assist$$en-us.tppi \
  ide/app.tpp/BSD$$en-us.tppi ide/app.tpp/Blitz$$en-us.tppi \
  ide/app.tpp/ConfiguringPackagesAssemblies$$en-us.tppi \
  ide/app.tpp/Contact$$en-us.tppi ide/app.tpp/Cpp$$en-us.tppi \
  ide/app.tpp/CrossComp$$en-gb.tppi ide/app.tpp/Files$$en-us.tppi \
  ide/app.tpp/FindFile$$en-us.tppi ide/app.tpp/Flags$$en-us.tppi \
  ide/app.tpp/GPL$$en-us.tppi ide/app.tpp/GettingStarted$$en-us.tppi \
  ide/app.tpp/GettingStarted$$ru-ru.tppi ide/app.tpp/IconDes$$en-us.tppi \
  ide/app.tpp/IntroductionToUPP$$en-us.tppi ide/app.tpp/Keys$$en-us.tppi \
  ide/app.tpp/NewProject$$en-us.tppi \
  ide/app.tpp/PackageTemplates$$en-us.tppi \
  ide/app.tpp/PackagesAssembliesAndNests$$en-us.tppi \
  ide/app.tpp/PackagesAssembliesAndNests$$ru-ru.tppi \
  ide/app.tpp/Sponsor$$en-us.tppi ide/app.tpp/Topic$$en-us.tppi \
  ide/app.tpp/cmdline$$en-us.tppi ide/app.tpp/esc$$en-us.tppi \
  ide/app.tpp/importext$$en-us.tppi ide/app.tpp/index$$en-us.tppi \
  ide/app.tpp/index$$ru-ru.tppi ide/app.tpp/install$$en-us.tppi \

```
  ide/app.tpp/install$$ru-ru.tppi ide/app.tpp/macros$$en-us.tppi \
  ide/app.tpp/special$$en-us.tppi ide/app.tpp/umk$$en-us.tppi \
  ide/app.tpp/upp$$en-us.tppi build_info.h
```

This works for BSD make just for first run, while dependency information wasn't generated yet (which may be enough for first build without interruptions).

The other possible solution is to rename directories/files with "$" characters, but this also may require to change other parts of source code.

The devel/upp FreeBSD port just uses gmake to build release version, which doesn't have this issue.

---

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 22:37:02 GMT
View Forum Message <> Reply to Message

Quote:we can do exactly like we do on Core, so detects with 'uname -s' that the operating system is Linux and treat all others like BSD. What do you think Amrein?

It's a solution. For that, I could add this in domake:

```
posix_os=$(uname -s)
if [ "$posix_os" != "Linux" ]
then
  sed -i -e 's/flagLINUX/flagBSD/g' uppsrc/Makefile.in
  sed -i -e 's/flagLINUX/flagBSD/g' uppsrc/uMakefile.in
fi
```

But there're many POSIX OS out there https://en.wikipedia.org/wiki/POSIX
If we use this solution, it won't be clean.

Is there a flag for clang++? I can see flagGCC in Makefiles.

Mirek, if my patch for Core.cpp is ok and if all goes well for tomorrow snapshots, what are the remaining things to do before releasing U++ stable 2017.1?

---

Subject: Re: Tarball issues
Posted by amrein on Sun, 15 Jan 2017 23:51:33 GMT
View Forum Message <> Reply to Message

I tested gmake on TrueOS and as you said it failed with '$$$$'.

Gmake on TrueOS (4.2) is newer than on my computer (4.1).

And so:
- make on BSD won't work without '$$$$'
- gmake 4.2 on BSD won't work with it
. gmake 4.1 on Fedora will work with or without it.

This is a bit weird.

---

Subject: Re: Tarball issues
Posted by mirek on Mon, 16 Jan 2017 10:20:24 GMT
View Forum Message <> Reply to Message

amrein wrote on Sun, 15 January 2017 22:59Thank

I found what I need in uppsrc/ide/Core/Core.cpp.
The patch:

```
--- uppsrc/ide/Core/Core.cpp.OLD        2016-10-15 16:42:07.000000000 +0200
+++ uppsrc/ide/Core/Core.cpp    2017-01-15 22:57:18.174354625 +0100
@@ -78,7 +78,7 @@
        String out;
        for(; *fn; fn++)
                if(*fn == '$')
-                       out << '$' << '$';
+                        out << "$$$$";
                else
                        out << *fn;
        return out;
```

Applied, now running nightly build.

---

Subject: Re: Tarball issues
Posted by amrein on Mon, 16 Jan 2017 18:21:16 GMT
View Forum Message <> Reply to Message

I modified the documentation for BSD users to force them to use gmake instead of make, added some tips about this issue and reverted my patch.
'$' -> '$$' is apparently now the default substitution to use for all new gmake release (and four dollars will fail). :/

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Tue, 17 Jan 2017 01:35:23 GMT

amrein wrote on Mon, 16 January 2017 18:21I modified the documentation for BSD users to force them to use gmake instead of make, added some tips about this issue and reverted my patch. '$' -> '$$' is apparently now the default substitution to use for all new gmake release (and four dollars will fail). :/
I read it in 10720 revision. But why did you use word "will" (instead of "may") in the following snippet of text?:
On BSD distributions, if you use make instead of gmake, U++ compilation will fail
For example, this is not an issue for uppsrc/umk (which also exported for current tarball), where there are no files with "$" characters for dependencies and current revision builds fine with BSD make. This is an issue for uppsrc/ide or other projects, where files with "$" character(s) may be used.

Also, I noticed about build requirements:
Build requires: gmake gtk2 freetype2 libnotify llvm39 (clang++)
Maybe this is true for some *BSD operating systems without Clang compiler in base, but Clang 3.4.1 on FreeBSD 10.3 is capable to build uppsrc/ide and uppsrc/umk projects in tarball (was tested for 10703 revision):
% which clang++
/usr/bin/clang++
% clang++ --version | head -1
FreeBSD clang version 3.4.1 (tags/RELEASE_34/dot1-final 208032) 20140512

Installation of devel/llvm39 FreeBSD port is not mandatory for building of uppsrc/ide and uppsrc/umk (current revisions), at least on FreeBSD 10.3 amd64. But even if build/install it, the clang++39 (wrapper script; as part of devel/llvm39 FreeBSD port) may be used, because clang++ may be installed to different directory (which may be not exposed in (common) PATH, because other clang++ may exist in /usr/bin).

I guess, the same is possible for Clang 3.8.0 (in base) on FreeBSD 11.0.

---

Subject: Re: Tarball issues
Posted by amrein on Tue, 17 Jan 2017 08:23:45 GMT


Quote:
I read it in 10720 revision. But why did you use word "will" (instead of "may") in the following snippet of text?:
<<On BSD distributions, if you use make instead of gmake, U++ compilation will fail>>

For example, this is not an issue for uppsrc/umk (which also exported for current tarball), where there are no files with "$" characters for dependencies and current revision builds fine with BSD make. This is an issue for uppsrc/ide or other projects, where files with "$" character(s) may be used.

Mainly because calling 'make' will try to build theide and umk at the same time.

Quote:
Also, I noticed about build requirements:

Build requires: gmake gtk2 freetype2 libnotify llvm39 (clang++)

Maybe this is true for some *BSD operating systems without Clang compiler in base, but Clang 3.4.1 on FreeBSD 10.3 is capable to build uppsrc/ide and uppsrc/umk projects in tarball (was tested for 10703 revision):

% which clang++
/usr/bin/clang++
% clang++ --version | head -1
FreeBSD clang version 3.4.1 (tags/RELEASE_34/dot1-final 208032) 20140512

Installation of devel/llvm39 FreeBSD port is not mandatory for building of uppsrc/ide and uppsrc/umk (current revisions), at least on FreeBSD 10.3 amd64. But even if build/install it, the clang++39 (wrapper script; as part of devel/llvm39 FreeBSD port) may be used, because clang++ may be installed to different directory (which may be not exposed in (common) PATH, because other clang++ may exist in /usr/bin).

What package should I install? llvm39 is what TrueOS tells me to install by default if clang++ is not installed.

---

Subject: Re: Tarball issues
Posted by amrein on Tue, 17 Jan 2017 08:37:56 GMT
View Forum Message <> Reply to Message

Found it: llvm39 -> clang-devel
-> documentation modified

But note that clang-devel clang++ binary on TrueOS is in fact a script that tells you to install llvm39 first. Compilation will fail because of this.
This script search for clang++ in /usr/local/bin.

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Tue, 17 Jan 2017 16:13:24 GMT
View Forum Message <> Reply to Message

amrein wrote on Tue, 17 January 2017 08:23What package should I install? llvm39 is what TrueOS tells me to install by default if clang++ is not installed.
Looks like, this is behaviour on TrueOS.
Previously, I said about FreeBSD 10.3 and 11.0 release versions, where some C/C++ compiler available in base:
% uname -sr

FreeBSD 10.3-RELEASE-p15
% which cc clang c++ clang++
/usr/bin/cc
/usr/bin/clang
/usr/bin/c++
/usr/bin/clang++
% cd /usr/bin && sha256 cc clang c++ clang++
SHA256 (cc) = 2f86f05d3bb79bfcb9be965df7c89e3066c39b7b119796e79e69664c92c53ce0
SHA256 (clang) = 2f86f05d3bb79bfcb9be965df7c89e3066c39b7b119796e79e69664c92c53ce0
SHA256 (c++) = 2f86f05d3bb79bfcb9be965df7c89e3066c39b7b119796e79e69664c92c53ce0
SHA256 (clang++) =
2f86f05d3bb79bfcb9be965df7c89e3066c39b7b119796e79e69664c92c53ce0
% c++ --version | head -1
FreeBSD clang version 3.4.1 (tags/RELEASE_34/dot1-final 208032) 20140512

As I understand, the current revisions of uppsrc/umk and uppsrc/ide projects requires C/C++ compiler for build, which may support C++11 (or C++0x) features.
According to some C++ compiler support page:
C++11 (complete): Clang >= 3.3, GCC >= 4.8.1.
C++14 (complete): Clang >= 3.4, GCC >= 5.0.
There is maybe related C++ library support needed also.

The current devel/llvm39 FreeBSD port just build/install LLVM/Clang (and other components for) 3.9.1, which is latest release version.

---

Subject: Re: Tarball issues
Posted by amrein on Tue, 17 Jan 2017 18:01:33 GMT
View Forum Message <> Reply to Message

It's part of the core system, great. Which package does clang++ belong?
"pkg_info -W /usr/bin/clang++" can tell I guess? If I add this to the documentation, it should work for everyone using BSD clones.

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Wed, 18 Jan 2017 06:27:07 GMT
View Forum Message <> Reply to Message

amrein wrote on Tue, 17 January 2017 18:01It's part of the core system, great. Which package does clang++ belong?
"pkg_info -W /usr/bin/clang++" can tell I guess? If I add this to the documentation, it should work for everyone using BSD clones.
It's already available in /usr/bin on FreeBSD operating system (at least for current release versions).
Usually, the user may specify C and C++ compiler with using CC and CXX variables for (BSD) make. This is what on FreeBSD 10.3 by default:

% make -V CC -V CXX -V CPP
cc
c++
cpp

Obviously, the (current version of) TrueOS has some differences (e.g. some script proposes to install C/C++ compiler as a package).

---

Subject: Re: Tarball issues
Posted by amrein on Wed, 18 Jan 2017 06:49:35 GMT
View Forum Message <> Reply to Message

Ok, so if I understand you correctly, now upp build scripts (Makefiles, domake, doinstall, upp-devel.spec, ...) are stable enough.
Those scripts take care of all know configurations (c++, g++, clang++, make, domake and pkg-config, Linux, BSD) for the best.

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Wed, 18 Jan 2017 08:17:44 GMT
View Forum Message <> Reply to Message

amrein wrote on Wed, 18 January 2017 06:49Ok, so if I understand you correctly, now upp build scripts (Makefiles, domake, doinstall, upp-devel.spec, ...) are stable enough.
This is what on FreeBSD 10.3 (in my configuration) for 10724 revision from Nightly builds page:
% fetch http://www.ultimatepp.org/downloads/upp-x11-src-10724.tar.gz
<...>
% sha256 upp-x11-src-10724.tar.gz
SHA256 (upp-x11-src-10724.tar.gz) =
2cec4d158dcc2af851d3b0072ca0a2e7d7ff485da80be1fd4f1c1ba1b0849b4b
% tar -xf upp-x11-src-10724.tar.gz
% cd upp-x11-src-10724
% make CXX=c++
<...>
% ls umk theide
theide umk
% ./umk
Usage: umk assembly main_package build_method -options [+flags] [output]

Examples: umk examples Bombs GCC -ab +GUI,SHARED ~/bombs
umk examples,uppsrc Bombs ~/GCC.bm -rv +GUI,SHARED ~/bin

See http://www.ultimatepp.org/app$ide$umk$en-us.html for details
% ./theide --help
Usage: theide assembly package

theide assembly package build_method [-[a][b][e][r][s][S][v][1][2][m][d][M][l][x][X][Hn]] [+FLAG[,FLAG]...] [out]
theide -f [file..]
theide [file..] // autodetection mode

---

Subject: Re: Tarball issues
Posted by amrein on Wed, 18 Jan 2017 12:44:09 GMT
View Forum Message <> Reply to Message

Cool.

Note:
- Even if you don't force c++ (with CXX=c++), domake will search in your path for g++, than clang++ and finally c++.
- You can also have more information about domake and doinstall running './domake --help' or './doinstall --help'.

---

Subject: Re: Tarball issues
Posted by Sender Ghost on Wed, 18 Jan 2017 14:32:18 GMT
View Forum Message <> Reply to Message

amrein wrote on Wed, 18 January 2017 12:44Note:
- Even if you don't force c++ (with CXX=c++), domake will search in your path for g++, than clang++ and finally c++.

Currently, I have lang/gcc port installed, as a build/runtime dependency for some port:
% which g++ | xargs pkg which -o
/usr/local/bin/g++ was installed by package lang/gcc
% pkg info -r lang/gcc
gcc-4.9.4:
kBuild-0.1.9998_6

I used CXX variable to set "c++" compiler for build, in this case.

---