Subject: [PROPOSAL] CppCheck support
Posted by Oblivion on Sun, 22 Jan 2023 17:00:46 GMT
View Forum Message <> Reply to Message

Hi,

I've been using CppCheck for a long time, and found it very useful. It is a very popular free static analyzer tool with lots of features and it is easy to integrate.

While its own parser is very good, it also provides an experimental clang AST backend, so it is very flexible.

I am currently integrating it with theIDE, and the integration is very smooth. So I am proposing to add cppcheck support to TheIDE.

My Plan is to make it available to public before U++ 2023 1, in a usable/stable state.

What it requires.

0. CppCheck binary.
1. A single file in TheIDE's codebase (CppCheck.cpp)
2. Detection of CppCheck binary (HasCppCheck() function)
3. A settings pane, in theIDE's settings dialog (will be available only if the binary is detected).
4. Color entries in (Settings/Syntax Highligting pane) for CppCheck error severity types (style, warning, performance, portability, etc.)

5. It will have the ability to check
   a) A single file
   b) A package
   c) Workspace (all packages)
   This will require adding menu and toolbar entries when cppcheck binary is detected.
   (Similar to existing "Compile $FILENAME" or "Build $PACKAGE" entries -> "Analyze $PACKAGE)

See the below screenshot for a working version (initial)

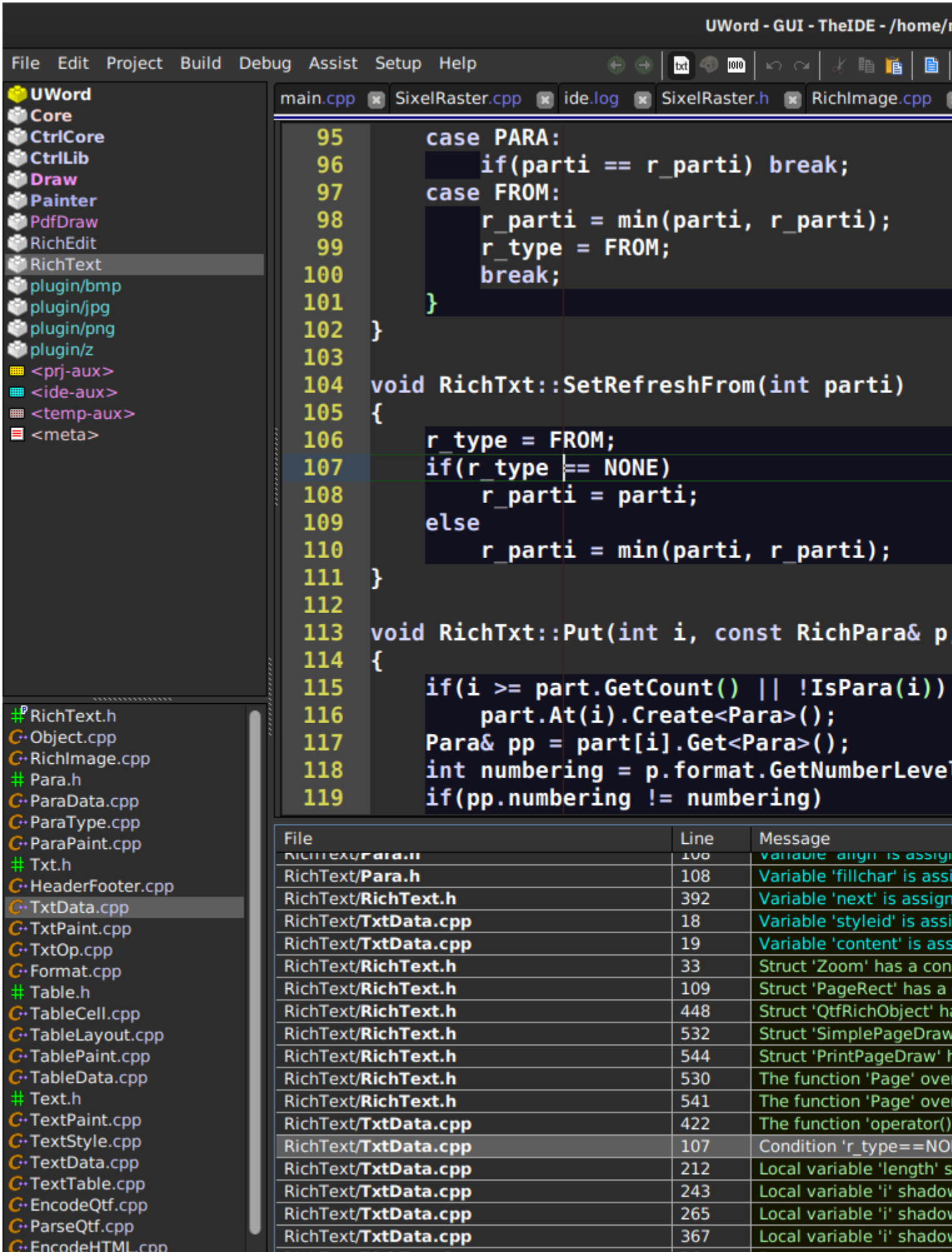
I will create a branch (theide_cppcheck) in my fork of upp and make the source code available so others can participate.

What do you think?

Best regards
Oblivion,

File Attachments

1) Ekran Görüntüsü - 2023-01-22 19-41-52.png , downloaded 1416 times


UWord - GUI - TheIDE - /home/
File Edit Project Build Debug Assist Setup Help
UWord
Core
CtrlCore
CtrlLib
Draw
Painter
PdfDraw
RichEdit
RichText
plugin/bmp
plugin/jpg
plugin/png
plugin/z
<prj-aux>
<ide-aux>
<temp-aux>
<meta>
main.cpp
SixelRaster.cpp
ide.log
SixelRaster.h
RichImage.cpp
95 case PARA:
96 if(parti == r_parti) break;
97 case FROM:
98 r_parti = min(parti, r_parti);
99 r_type = FROM;
100 break;
101 }
102 }
103
104 void RichTxt::SetRefreshFrom(int parti)
105 {
106 r_type = FROM;
107 if(r_type == NONE)
108 r_parti = parti;
109 else
110 r_parti = min(parti, r_parti);
111 }
112
113 void RichTxt::Put(int i, const RichPara& p
114 {
115 if(i >= part.GetCount() || !IsPara(i))
116 part.At(i).Create<Para>();
117 Para& pp = part[i].Get<Para>();
118 int numbering = p.format.GetNumberLeve
119 if(pp.numbering != numbering)
RichText.h
Object.cpp
RichImage.cpp
Para.h
ParaData.cpp
ParaType.cpp
ParaPaint.cpp
Txt.h
HeaderFooter.cpp
TxtData.cpp
TxtPaint.cpp
TxtOp.cpp
Format.cpp
Table.h
TableCell.cpp
TableLayout.cpp
TablePaint.cpp
TableData.cpp
Text.h
TextPaint.cpp
TextStyle.cpp
TextData.cpp
TextTable.cpp
EncodeQtf.cpp
ParseQtf.cpp
EncodeHTML.cpp
File Line Message
RichText/Para.h 108 Variable 'fillchar' is assi
RichText/RichText.h 392 Variable 'next' is assign
RichText/TxtData.cpp 18 Variable 'styleid' is assi
RichText/TxtData.cpp 19 Variable 'content' is ass
RichText/RichText.h 33 Struct 'Zoom' has a con
RichText/RichText.h 109 Struct 'PageRect' has a
RichText/RichText.h 448 Struct 'QtfRichObject' ha
RichText/RichText.h 532 Struct 'SimplePageDraw
RichText/RichText.h 544 Struct 'PrintPageDraw' h
RichText/RichText.h 530 The function 'Page' ove
RichText/RichText.h 541 The function 'Page' ove
RichText/TxtData.cpp 422 The function 'operator()
RichText/TxtData.cpp 107 Condition 'r_type==NO
RichText/TxtData.cpp 212 Local variable 'length' s
RichText/TxtData.cpp 243 Local variable 'i' shadow
RichText/TxtData.cpp 265 Local variable 'i' shadow
RichText/TxtData.cpp 367 Local variable 'i' shadow

Subject: Re: [PROPOSAL] CppCheck support
Posted by koldo on Mon, 23 Jan 2023 07:20:40 GMT
View Forum Message <> Reply to Message

Thank you Oblivion

I would like to have it.

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sat, 04 Feb 2023 10:54:02 GMT
View Forum Message <> Reply to Message

Hello,

The initial linter (cppcheck) support has landed in my fork of U++, under ide_linter branch.

The basic functionality is there. It is tested on Linux. Windows exe detection will be availble tomorrow, so as of today it only supports Linux.

DONE
+ Check file, package, project (all) commands.
+ TheIDe linter keyboard shortcuts for check file, package, and project commands.
+ TheIDE linter setup pane in Setup dialog.
+ Ability to select severity message types.
+ Core configuration settings (language, standard, platform, etc.)

TODO:
- Add Windows cppcheck executable detection.
- Add configurable cppcheck build dir path (to speed-up analysis)
- Add library file support.
- Add Package/File list menu entries.
- Add CLANG backend switch.
- Add the ability to pass additional command line options.
- Filter out non C/C++ files (*.tpp, *.log, etc).

To make it less intrusive, I have put it into its own package under ide (ide/Linter).

I'll add a more detailed overview tomorrow.

Reviews, patches, suggestions and testing is welcome.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by deep on Sat, 04 Feb 2023 12:38:33 GMT
View Forum Message <> Reply to Message

Hi Oblivion,

Downloaded your branch.

Works great. I was waiting for This feature.

Subject: Re: [PROPOSAL] CppCheck support
Posted by Klugier on Sat, 04 Feb 2023 20:10:36 GMT
View Forum Message <> Reply to Message

Hello Oblivion,

I like the idea to addlinter support for TheIDE, however I am not sure we should go for cppcheck. When I used it in the past, the tool had some problems like false positives. Also, the decision to relay on this particular one in the main code base is very difficult. We have very good alternatives such as clang-tidy and the true thing is that we are now basing on clang ecosystem like never before. With the usage of libclang and recent clang-format integration.

What about creating plugin system for TheIDE to allowing creation of extensions? Thanks to that there will be possible to support multiple linters. I understand that it will require a lot of work, but in the long term it is the way to go.

Klugier

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sat, 04 Feb 2023 21:05:51 GMT
View Forum Message <> Reply to Message

Hello Klugier,

Quote:Hello Oblivion,

I like the idea to addlinter support for TheIDE, however I am not sure we should go for cppcheck. When I used it in the past, the tool had some problems like false positives. Also, the decision to relay on this particular one in the main code base is very difficult. We have very good alternatives such as clang-tidy and the true thing is that we are now basing on clang ecosystem like never before. With the usage of libclang and recent clang-format integration.

What about creating plugin system for TheIDE to allowing creation of extensions? Thanks to that there will be possible to support multiple linters. I understand that it will require a lot of work, but in the long term it is the way to go.

Klugier

Thank you very much for your comments.

CppCheck has matured nicely. Yes, it occasionaly gives false positives, but so does clang-tidy. Not to mention cppcheck is famous for catching *interesting* issues where other solutions fail. The main reason why I opt for cppcheck is:

It is extremely simple to integrate and remove.
a) The ide/Linter package is almost self-contained package in the sense that there are only several Lines of code (7 lines of code in cpp files, and one line of code in ide.h: only menu entries, and config stuff) in TheIde's codebase.
IMO that's not much of a maintenanca burden. Removing it completely from the codebase if required won't take more than 15 seconds.
b) It uses it's own config file (JSON), meaning it does not mess with TheIDE's config file.
c) Implementation does not have to rely on low level, raw stuff, so future changes won't really break it unless U++ somehow gets broken.
d) It is activated only if the cppcheck executable is found.

If you look closely to the linter package, you'll notice that the main class (Linter) is actually an abstraction.
This is a stripped down version of my own linters system (I use a commercial solution besides cppcheck).
Hence the name is Linter, not CppCheck. We can definitely (re)add clang-tidy later.

As for the plugin system:
Writing a plugin system has always been a good idea, given that I intend to participate more in TheIDE's development this year, I'll likely look into it.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sat, 04 Feb 2023 23:26:59 GMT
View Forum Message <> Reply to Message

By the way,

I decided to re-implement clang-tidy support in Linter package too. Lets have both.
First version will likely be available next weekend.

Best regards,

Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sun, 05 Feb 2023 14:08:54 GMT
View Forum Message <> Reply to Message

Hi,

ide/Linter package now supports Windows too.
If you download the official cppcheck executable, theIde should detect it.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sat, 11 Feb 2023 22:55:46 GMT
View Forum Message <> Reply to Message

Hi,

TheIde/Linter (Cppcheck) has made some more progress.

+ CppCheck library files support is enabled.
+ CppCheck plugins support is enabled.
+ Additional command line options can now be passed manually, via setup dialog. (If needed)
+ U++ X11 (NOGTK) backend crash fixed on linux.

Note that, cppcheck library files (configurable checkers) are a powerful and nice way to allow cppcheck to recognize external libraries' (U++, GTK, QT, etc.) stuff (functions, etc.) Now it is enabled. ide/Linter will now check for .cfg files under a user-configurable path.

Similarly, 3rd party plugins for cppcheck can now be enabled.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Tue, 18 Apr 2023 22:06:33 GMT
View Forum Message <> Reply to Message

Hi,

Experimental CppCheck branch of my U++ fork is updated.

Changes:

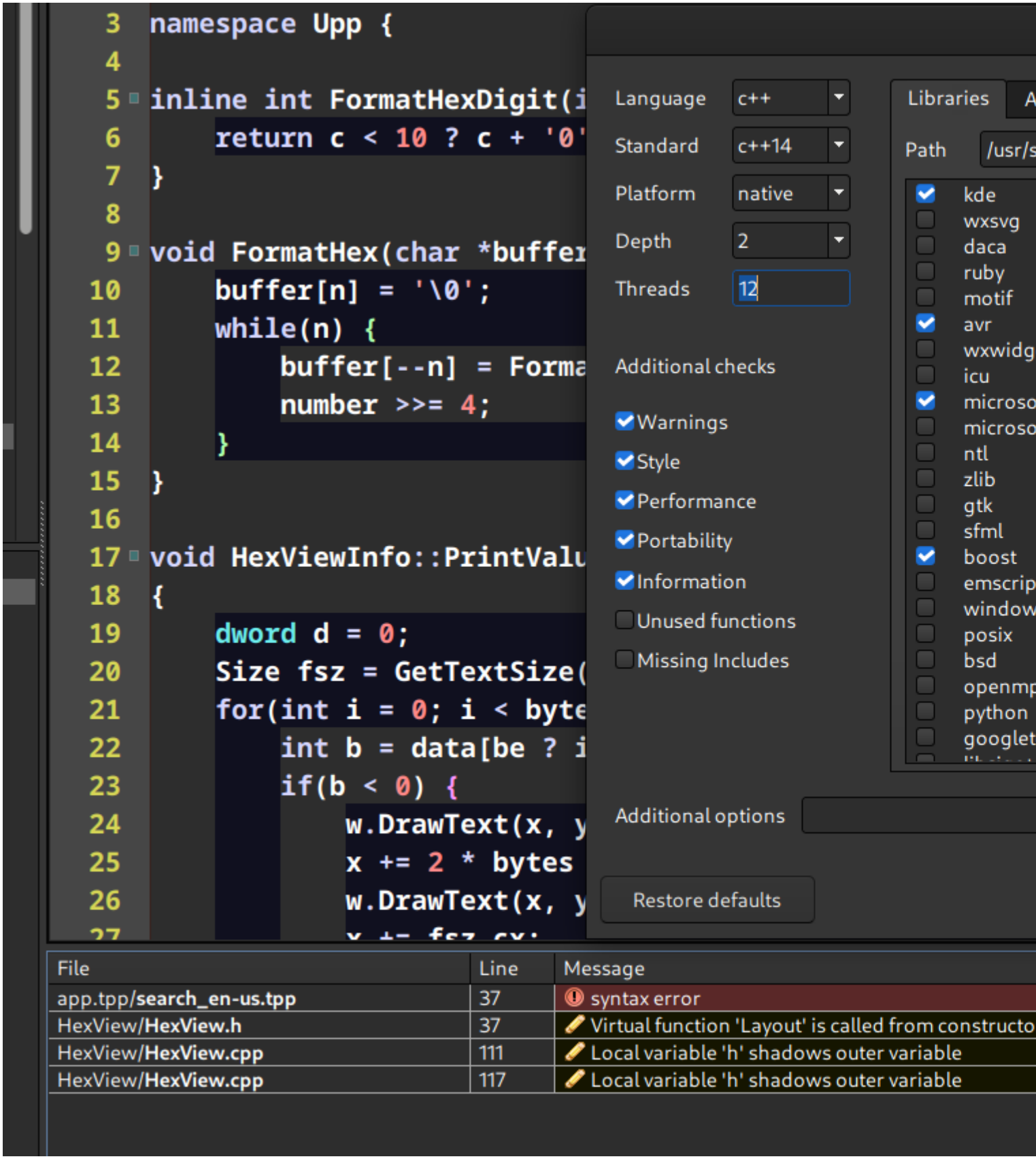
Configuration dialog is redesigned.
Ability to select individual libraries and addons.

Screenshot:

Best regards,
Oblivion

File Attachments

1) Ekran Görüntüsü - 2023-04-19 01-02-14.png , downloaded 1272 times


namespace Upp {
inline int FormatHexDigit(i
return c < 10 ? c + '0'
}
void FormatHex(char *buffer
buffer[n] = '\0';
while(n) {
buffer[--n] = Forma
number >>= 4;
}
}
void HexViewInfo::PrintValu
{
dword d = 0;
Size fsz = GetTextSize(
for(int i = 0; i < byte
int b = data[be ? i
if(b < 0) {
w.DrawText(x, y
x += 2 * bytes
w.DrawText(x, y
Language
c++
Standard
c++14
Platform
native
Depth
2
Threads
12
Additional checks
Warnings
Style
Performance
Portability
Information
Unused functions
Missing Includes
Additional options
Restore defaults
Libraries
Path
/usr/s
kde
wxsvg
daca
ruby
motif
avr
wxwidg
icu
microso
microso
ntl
zlib
gtk
sfml
boost
emscrip
window
posix
bsd
openmp
python
googlet
File
Line
Message
app.tpp/search_en-us.tpp
37
syntax error
HexView/HexView.h
37
Virtual function 'Layout' is called from constructo
HexView/HexView.cpp
111
Local variable 'h' shadows outer variable
HexView/HexView.cpp
117
Local variable 'h' shadows outer variable

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Thu, 20 Apr 2023 19:40:07 GMT
View Forum Message <> Reply to Message

Hi,

Experimenal linter package for TheIDE is updated.

This is a big update since the package is now reimplemented as a generic framework for interfacing with command-line-driven static analysis tools.

Accordingly, CppCheck interface is reimplemented as a linter module.I've added the essential API docs.
It is now possible to add multiple linters easily. (Meaning that clang-tidy module is on its way.)

Also, cppcheck module can now display verbose messages and inconclusive results.

DONE & TODO List:

+ Linter package: Redesigned as a framework to utilize multiple command-line-driven static analysis tools.
+ Linter package: Added initial API docs for implmenting linter modules.
+ CppCheck module: Re-implemented as a linter module.
+ CppCheck module: Can now show verbose messages.
+ CppCheck module: Can now show inconclusive results.
+ CppCheck module: Filtering out non C/C++ files and directories (*.tpp, *.log, etc).

TODO:
- CppCheck module: Add configurable build dir path (to speed-up analysis).
- CppCheck module: Add CLANG backend switch to CppCheck module.
- CppCheck module: Allow per-project configuration file.
- Linter package: Add a mechanism to switch between linter modules on-the-fly.
- Linter package: Add a clang-tidy module.

You can download the code from this address.

Any questions, suggestions, bug reports, etc. are welcome.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sun, 23 Apr 2023 16:14:56 GMT

View Forum Message <> Reply to Message

Hi,

Last round of updates for this week:

CppCheck module: Now allows per-project configuration files.
Configuration files (json) are stored in cppcheck folder, under the upp's config directory.
And they are automatically loaded when the linter is invoked.
Format of the name of each config file is [name of the main package]-cppcheck.json.

Code can be found here.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sat, 29 Apr 2023 11:36:53 GMT
View Forum Message <> Reply to Message

Hi,

A round of weekly updates to the experimental TheIDe/Linter framework:

+ Linter package: Linter package is now a compile-time option, using the LINTER flag.
+ Linter package: Footprint in TheIDE's source code is further narrowed down.
+ Linter package: A module registration mechanism is implemented.
+ Linter package: Added a mechanism to switch between available linter modules on-the-fly.

Now the registered modules that have a valid/installed backend will appear in the menu and can be selected by the user.

Since now the module registration and selection is implemented, it is time to send a pull request for review and start implementing clang-tidy module.

Any questions, reviews, testing, help is welcome.

Best regards,
Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by deep on Sun, 08 Sep 2024 09:26:19 GMT
View Forum Message <> Reply to Message

Hi Oblivion,

I use the theide with cppcheck.

What is your frequency of update for LINTER (ide_linter) branch.

Quote:
This branch is 78 commits ahead of, 76 commits behind ultimatepp/ultimatepp:master

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Sun, 08 Sep 2024 11:03:06 GMT
View Forum Message <> Reply to Message

Hello deep,

Quote:What is your frequency of update for LINTER (ide_linter) branch.

Monthly, but I am working on clang-tidy module in the background, so the update was somewhat delayed.

Anyways, I have now updated the linter branch to latest U++. IT works fine here, but let me know if you encounter any problems.

Best regards,

Oblivion

Subject: Re: [PROPOSAL] CppCheck support
Posted by deep on Sun, 08 Sep 2024 18:08:08 GMT
View Forum Message <> Reply to Message

Hello Oblivion

Thanks,

It is working.

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Wed, 23 Apr 2025 17:43:46 GMT
View Forum Message <> Reply to Message

Hi,

Long time no announcement.
But thanks to exposed commands.json file support in theIDE, I have finally implemented Linter::clang-tidy support
I haven't pushed the changes to github yet as it needs more testing before being published, but here's a screenshot of what's to come:

Best regards,
Oblivion

File Attachments

1) theide-linter-clang-tidy.png, downloaded 899 times

```
18                  ColorPusher& bt = pushers.Add();
19                  bt <<= Color(Random(256), Random(256), Random(256));
20                  bt.SetRect(0, 0, 100, 100);
21                  Add(bt);
22                  ctrls.Add(&bt);
23                  targets.Add(RectC(col * 100, row * 100, 100, 100).Deflated(2));
24              }
25          }
26
27          SetTimeCallback(1000, [&]{ Animate(ctrls, targets, 300); });
28      }
29  };
30
31  GUI_APP_MAIN
```

File	Line	Message
AnimateCtrlGeometry/**main.cpp**	19	'Random' must resolve to a function declared within the na
AnimateCtrlGeometry/**main.cpp**	19	256 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	20	100 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	20	100 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	23	'RectC' must resolve to a function declared within the name
AnimateCtrlGeometry/**main.cpp**	23	no header providing "Upp::RectC" is directly included [misc
AnimateCtrlGeometry/**main.cpp**	23	100 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	23	100 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	23	100 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	23	100 is a magic number; consider replacing it with a named c
AnimateCtrlGeometry/**main.cpp**	27	calling a function that uses a default argument is disallowe
AnimateCtrlGeometry/**main.cpp**	27	1000 is a magic number; consider replacing it with a named

Subject: Re: [PROPOSAL] CppCheck support
Posted by deep on Sat, 26 Apr 2025 12:13:11 GMT
View Forum Message <> Reply to Message

Hi Oblivion

Great.

Looking forward to have it.

Subject: Re: [PROPOSAL] CppCheck support
Posted by deep on Sat, 19 Jul 2025 11:43:00 GMT
View Forum Message <> Reply to Message

Hi

I updated my local git to merge CPP check with nightly build with following changes

Linter.cpp file patch

```
diff -U 3 /WC_Linter/uppsrc/ide/Linter/Linter.cpp /upp_lint/uppsrc/ide/Linter/Linter.cpp
--- /WC_Linter/uppsrc/ide/Linter/Linter.cpp Fri Jul 18 19:18:37 2025
+++ /upp_lint/uppsrc/ide/Linter/Linter.cpp Sat Jul 19 16:45:54 2025
@@ -74,7 +74,7 @@

 String Linter::GetPackagePath() const
 {
- return TheIde()->GetActivePackagePath();
+ return TheIde()->GetActivePackageDir();
 }

 void Linter::CheckFile()
@@ -89,7 +89,8 @@
 {
  if(!Exists())
   return;
- Vector<String> paths = { GetFileFolder(GetPackagePath()) };
+ Vector<String> paths = { GetPackagePath() };
  DoCheck(Scope::Package, paths);
 }

@@ -100,7 +101,8 @@
  Vector<String> paths;
  const Workspace& wspc = GetIdeWorkspace();
  for(int i = 0; i < wspc.GetCount(); i++)
-  paths.Add() = GetFileFolder(PackagePath(wspc[i]));
+  paths.Add() = PackageDirectory(wspc[i]);
  DoCheck(Scope::Project, paths);
```

```
 }
```

And applied changes manually to following files.
Lines to be inserted marked ++

/ide/idebar.cpp file

```
  #ifdef PLATFORM_POSIX
   if(IsValgrind())
    menu.Add(b, AK_VALGRIND, THISBACK(Valgrind))
     .Help("Build application & run in valgring");
  #endif

   menu.Separator();
++  #ifdef flagLINTER // Experimental static analyzer tools support.
++   Linter::StdMenu(menu);
++  #endif
  }
 }
```

/ide/ide.upp file

```
 Report,
 Core/SSL,
 plugin/md,
 ide/clang,
++ ide/Linter;
```

/ide/ide.h file

```
#include <TextDiffCtrl/TextDiffCtrl.h>
#include <ide/Designers/Designers.h>
#include <ide/Android/Android.h>
#include <plugin/md/Markdown.h>

++ #ifdef flagLINTER
++ #include <ide/Linter/Linter.h>
++ #endif

#include "About.h"
#include "MethodsCtrls.h"

#define LAYOUTFILE <ide/ide.lay>
#include <CtrlCore/lay.h>
```

Use flags for compilation GUI LINTER

Subject: Re: [PROPOSAL] CppCheck support
Posted by Oblivion on Tue, 09 Sep 2025 21:27:36 GMT
View Forum Message <> Reply to Message

Hello Deepak,

Thanks for the patch. I've updated the branch.

Best regards,
Oblivion