**Cppcheck guide**

**Table of Contents**

* 1. about cppcheck
* 2. Cppcheck using
* 3. cppcheck work with other softwares
  + 3.1 work with VisualStudio
  + 3.2 work with TortoiseSVN

**1. about cppcheck**

Cppcheck is a static code checking tool, support c/ c++ code;can be used as a supplemental check for the compiler.

Check include：

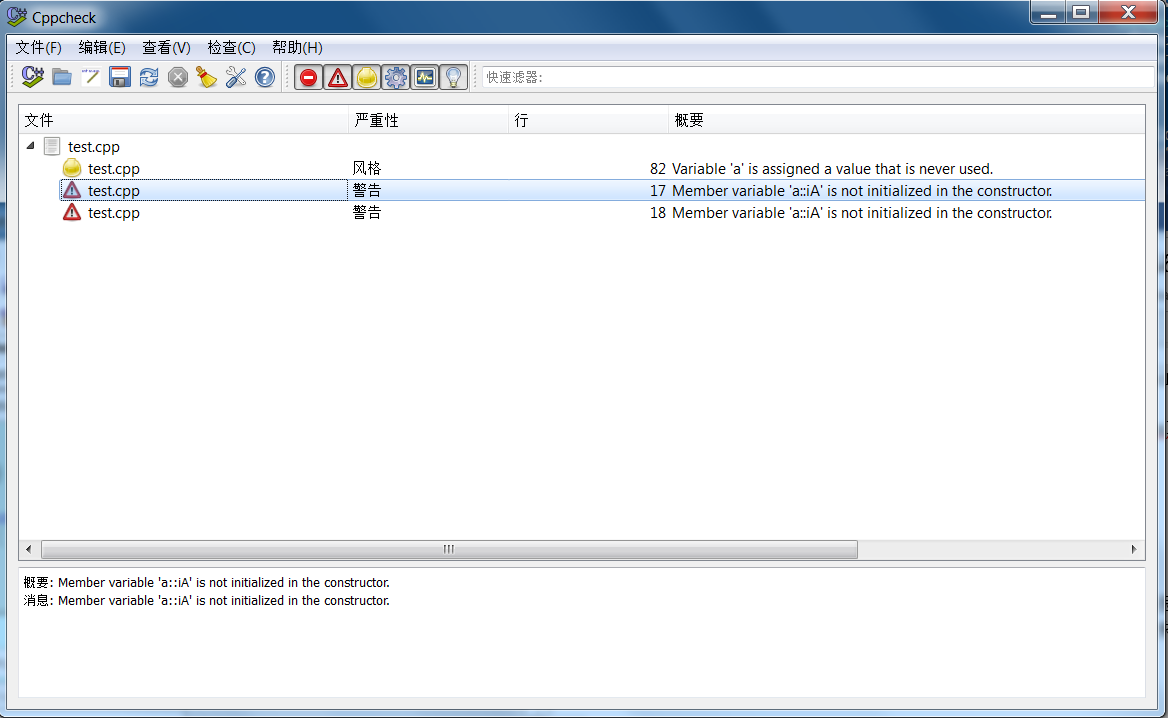
* Variable checking
* Array boundary checking
* Class checking
* Waste function call checking
* Abnormal memory usage checking
* Memory leak, mainly about pointers
* The exception STL function uses
* Code format

**2. cppcheck using**

Download address: https://sourceforge.net/projects/cppcheck/

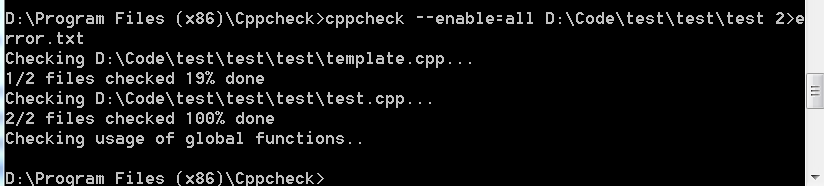
After installing cppcheck,there are two tools in the installating directory, and one is a program with graphical interfaces(cppcheckgui.exe), another is a console program(cppcheck.exe).

* cppcheckgui.exe



Load file or folder(note:it’s better to use English folder) ,after load it automatically check all possible problems and list them out.

* cppcheck.exe



enter the cppcheck installation directory through the command line and type the following command.

cppcheck –enable=all D:\Code\test\test\test\test.cpp (full of the path)  2>error.txt

the error will be generated in the file under the installation directory called error.txt. for example：

[D:\Code\test\test\test\test.cpp:82]: (style) Variable 'a' is assigned a value that is never used.

[D:\Code\test\test\test\test.cpp:17]: (warning) Member variable 'a::iA' is not initialized in the constructor.

[D:\Code\test\test\test\test.cpp:18]: (warning) Member variable 'a::iA' is not initialized in the constructor.

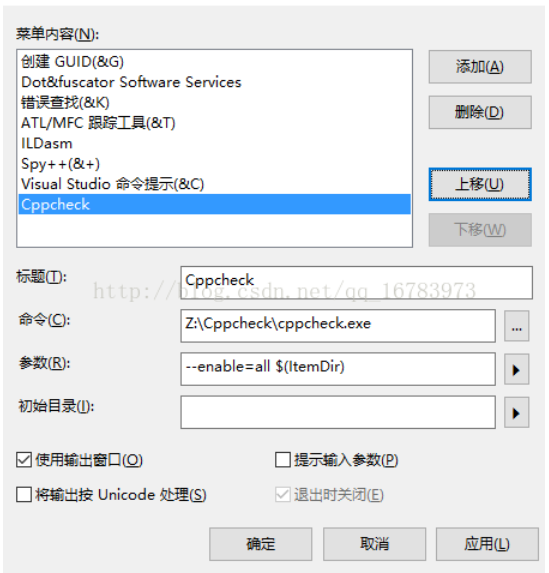
[D:\Code\test\test\test\test.cpp:24]: (style) The function 'func' is never used.

(information) Cppcheck cannot find all the include files (use --check-config for details)

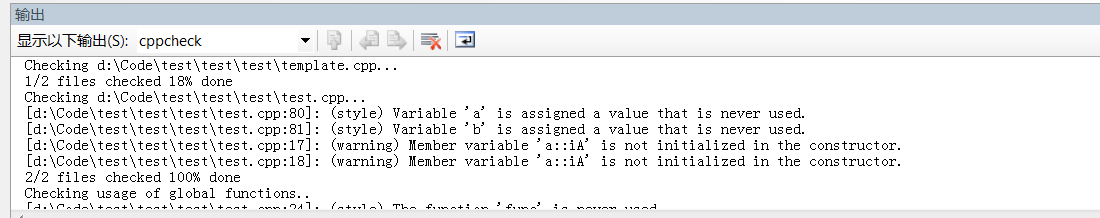
**3. cppcheck work with other softwares**

**3.1 work with VisualStudio**

1. in the visualstudio menu bar choose tools->external tools



1. after set up, just point tool cppcheck menu, will output code,check the results.



**3.2 work with TortoiseSVN**

Configuration

1.Download the "SVN\_Pre\_Commit\_Hook\_\_CppCheck\_Validate.bat" script to your machine and edit it:

* cppCheckPath - Full path to your Cppcheck.exe (not CppcheckGui.exe).
* supportedFileTypes - Add or remove file types to check. This variable is here so the script won't check '.sln', '.vxproj' and other non-source file types.
* enableScript - '1' or '0' to enable/disable running the script.

2.Right click (somewhere on desktop) –> TortoiseSVN –> Settings –> Hook Scripts –> Add…

3.Configure Hook Scripts:

* Hook Type: Choose 'Pre-Commit Hook' (upper right corner).
* Working Copy Path: The directory that all of your SVN checkouts are done. Use the top most directory (or just use 'C:\' for example).
* Command Line To Execute: Full path to the attached script.
* Make sure that both 'Wait for the script to finish' and 'Hide the script while running' checkboxes are checked –> OK –> OK.

Hints

1.Even if the commit failed because of the Cppcheck, SVN gives you the option to easily recommit disregarding the failure by clicking the 'Retry without hooks' button. If commit succeeded (meaning, Cppcheck did not find any issues), it will look like nothing happened (so developers will still see a commit end message just like before).

2.If you want to implement this solution in your organization/team you can do it in two different approaches:

* Client side solution - Meaning, the steps above should be taken for all of your development machines. The benefit in this approach is that only relevant teams can use this solution and not all of the developers that are working on the SVN server. Besides, ignoring this Cppcheck (in case of false-positives for example) is quite easy using one button click integrated in the TortoiseSVN Client ('Retry without hooks'). This approach means that Cppcheck must be installed on all of the relevant developers machines of course.
* Server side solution - Meaning, Cppcheck should be installed only on the SVN server and the steps above should be taken only once (server side only). So clients (developers' machines) should take no action since every commit will trigger the hook at server side. The benefit is this is taken only once, but this solution may be to restrictive for some organizations. In addition, in order to ignore the hook (once again, false-positive for example) - you need to create some 'back-door' script that will allow developers to bypass it with a specific keyword in the commit message.

3.More about SVN hook scripts - Client Hook Scripts, Server Hook Scripts.

All you need to do is take the Configuration steps above just once. Afterwards, you can work with SVN the same as before, just now you get to see your failures before code is committed to the SVN server.