Modern C++ (Part 1/N) Companion

Examples

If with Init

Allowing variable initialization within `if` statements is a useful feature because it variables that are only used in the context of the conditional statements from bleeding out into the external scope.

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:1,endLineNumber:9,positionColumn:1,positionLineNumber:9,selectionStartColumn:1,selectionStartLineNumber:9,startColumn:1,startLineNumber:9),source:%27//+If+with+Init+Example%0A%0A%23include+%3Ciostream%3E%0A%23include+%3Cstring%3E%0A%0Aint+main()+%7B%0A++++//+Create+a+string%0A++++std::string+s(%22prefix_something%22)%3B%0A%0A++++//+Check+if+the+string+has+a+prefix%0A++++auto+underscore+%3D+s.find(!%27_!%27)%3B+%0A++++if(underscore+!!%3D+std::string::npos)+%7B%0A++++++++std::cout+%3C%3C+%22Found+%22+%3C%3C+s%5Bunderscore%5D+%3C%3C+!%27%5Cn!%27%3B%0A++++%7D+else+%7B%0A++++++++std::cout+%3C%3C+%22_+not+present+in+string%5Cn%22%3B%0A++++%7D%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:50,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27)),k:50,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4
- https://en.cppreference.com/w/cpp/language/if

Override/Final

`override` and `final` are useful keywords for expressing the intent behind our class/struct implementation decisions. They can be used (respectively) to prevent:

- Typos in function names when you intend to override a virtual function
- Prevent virtual functions from being overloaded by derived classes

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:17,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:18,endLineNumber:18,positionColumn:18,positionLineNumber:18,selectionStartColumn:18,selectionStartLineNumber:18,startColumn:18,startLineNumber:18),source:%27//+Override/Final+Example%0A%0A%23include+%3Ciostream%3E%0A%0Astruct+Animal+%7B%0A++++virtual+void+speak()+%7B%0A++++++++std::cout+%3C%3C+%22Default+speak+function!!%5Cn%22%3B%0A++++%7D%0A%7D%3B%0A%0Astruct+Dog+:+Animal+%7B%0A++++void+speak()+%7B%0A++++++++std::cout+%3C%3C+%

22Woof!!%5Cn%22%3B%0A++++%7D%0A%7D%3B%0A%0Astruct+Cat+:+Animal+%7B%0A++++void+speak()+%7B%0A++++++++std::cout+%3C%3C+%22Meow!!%5Cn%22%3B%0A++++%7D%0A%7D%3B%0A%0Aint+main()+%7B%0A++++//+Create+instances+of+our+derived+classes%0A++++Dog+d%3B%0A++++Cat+c%3B%0A%0A++++//+Upcast+to+base+class+type%0A++++Animal+%26a1+%3D+d%3B%0A++++Animal+%26a2+%3D+c%3B%0A%0A++++//+Call+speak+methods%0A++++a1.speak()%3B%0A++++a2.speak()%3B%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:45.146726862302486,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:49.94708994708994,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2+-lpthread%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50.05291005291006,n:%270%27,o:%27%27,s:0,t:%270%27)),k:54.85327313769752,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4

- https://en.cppreference.com/w/cpp/language/override
- https://en.cppreference.com/w/cpp/language/final

Rvalue References/std::move

Move construction allows a new object to steal away the resources of an existing object (and avoid the performance penalty of copying).

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:1,endLineNumber:13,positionColumn:1,positionLineNumber:13,selectionStartColumn:1,selectionStartLineNumber:13,startColumn:1,startLineNumber:13),source:%27//+RValue+Example+%0A%0A%23include+%3Ciostream%3E%0A%23include+%3Cutility%3E%0A%23include+%3Cvector%3E%0A%0A//+A+simple+struct+that+implements+construct/copy/move%0Astruct+S+%7B%0A++++S()+%7B+std::cout+%3C%3C+%22Calling+Constructor!!%5Cn%22%3B+%7D%0A++++S(const+S+%26rhs)+%7B+std::cout+%3C%3C+%22Calling+Copy+Constructor!!%5Cn%22%3B+%7D%0A++++S(S+%26%26rhs)+%7Bstd::cout+%3C%3C+%22Calling+Move+Constructor!!%5Cn%22%3B+%7D%0A%7D%3B%0A%0Aint+main()+%7B%0A++++//+Create+a+vector+of+S%0A++++std::vector%3CS%3E+vector%3B%0A++++%0A++++//+Construct+an+S%0A++++S+s%3B%0A%0A++++//+Add+it+to+the+vector%0A++++vector.push_back(s)%3B%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:50,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27)),k:50,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4
- https://en.cppreference.com/w/cpp/utility/move

std::unique_ptr and std::make_unique

Raw pointers and `new`/ `delete` are heavily discouraged in modern C++ due to how easily it is to make mistakes in their use (out of bounds accesses, memory leaks, etc.). Smart points, like `std::unique_ptr` can be used to prevent common issues while maintaining the same interface.

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:2,endLineNumber:14,positionColumn:2,positionLineNumber:14,selectionStartColumn:2,selectionStartLineNumber:14,startColumn:2,startLineNumber:14),source:%27//+Make+Unique+Example+%0A%0A%23include+%3Ciostream%3E%0A%23include+%3Cmemory%3E%0A%0Aint+main()+%7B%0A++++//+Dynamically+allocate+an+array%0A++++int+*ptr+%3D+new+int%5B10%5D%3B%0A%0A++++//+Fill+it+with+some+values%0A++++for(int+i+%3D+0%3B+i+%3C+10%3B+i%2B%2B)+%7B%0A++++++++ptr%5Bi%5D+%3D+i*i%3B%0A++++%7D%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:50,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27)),k:50,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4
- https://en.cppreference.com/w/cpp/memory/unique_ptr
- https://en.cppreference.com/w/cpp/memory/unique_ptr/make_unique

Structured Bindings

Structured bindings is added syntactic sugar to C++ that make it easier to unpack values from containers.

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:31,endLineNumber:1,positionColumn:31,positionLineNumber:1,selectionStartColumn:31,selectionStartLineNumber:1,startColumn:31,startLineNumber:1),source:%27//+Structured+Bindings+Example%0A//+Includes+range-base+for+loop+and+auto%0A%0A%23include+%3Ciostream%3E%0A%23include+%3Cstring%3E%0A%23include+%3Cunordered_map%3E%0A%0Aint+main()+%7B%0A++++//+Create+an+unordered+map%0A++++std::unordered_map%3Cstd::string,+std::string%3E+table+%3D+%7B%0A++++++++%7B%22Roland%22,+%22North+Dakota%22%7D,%0A++++++++%7B%22Ben%22,+%22Texas%22%7D,%0A++++++++%7B%22Christina%22,+%22Texas%22%7D%0A++++%7D%3B%0A%0A++++//+Iterate+over+the+container%0A++++for(const+std::pair%3Cstd::string,+std::string%3E+%26item+:+table)+%7B%0A++++++++std::cout+%3C%3C+%22Name:+%22+%3C%3C+item.first+%3C%3C+%22,+State:+%22+%3C%3C+item.second+%3C%3C+!%27%5Cn!%27%3B%0A++++%7D%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:50,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27)),k:50,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4
- https://en.cppreference.com/w/cpp/language/structured_binding

std::string_view

`std::string_view` allows us to express our intent when we want to use a string, but not manage that string's memory.

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:24,endLineNumber:14,positionColumn:24,positionLineNumber:14,selectionStartColumn:24,selectionStartLineNumber:14,startColumn:24,startLineNumber:14),source:%27//+String+View+Example%0A%0A%23include+%3Ciostream%3E%0A%23include+%3Cstring%3E%0A%23include+%3Cstring_view%3E%0A%0Aint+main()+%7B%0A++++//+Create+a+string%0A++++std::string+s(%22prefix_something%22)%3B%0A++++%0A++++//+Trim+off+the+prefix%0A++++auto+trimmed+%3D+s.substr(s.find_first_of(!%27_!%27)+%2B+1)%3B%0A++++%0A++++//+Print+the+result%0A++++std::cout+%3C%3C+trimmed+%3C%3C+!%27%5Cn!%27%3B%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:50,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27)),k:50,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4
- https://en.cppreference.com/w/cpp/string/basic_string_view

Lambdas

Function objects are incredibly useful in C++, but often require a fair bit of boilerplate code to implement as structs/classes. Lambdas provide often provide a more simple write anonymous function objects in our code.

Links

- https://godbolt.org/#g:!((g:!((g:!((h:codeEditor,i:(filename:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,selection:(endColumn:1,endLineNumber:2,positionColumn:1,positionLineNumber:2,selectionStartColumn:1,selectionStartLineNumber:2,startColumn:1,startLineNumber:2),source:%27//+Lambda+Example+%0A%0A%23include+%3Calgorithm%3E%0A%23include+%3Ciostream%3E%0A%23include+%3Cvector%3E%0A%0A//+A+simple+function+object+to+check+if+a+number+is+evenly+divisable+by+another%0Astruct+IsDivisable+%7B%0A++++int+divisor%3B%0A++++IsDivisable(int+new_divisor)+:+divisor(new_divisor)+%7B%7D%0A++++bool+operator()(int+dividend)+%7B%0A++++++++return+dividend+%25+divisor+%3D%3D+0%3B%0A++++%7D%0A%7D%3B%0A%0Aint+main()+%7B%0A++++//+Create+a+vector+of+some+numbers%0A++++std::vector%3Cint%3E+vector+%3D+%7B244,+41,+6,+20,

+11%7D%3B%0A++++%0A++++//+Construct+our+function+object%0A++++IsDivisable+is_divisable_by_10(10)%3B%0A++++%0A++++//+Find+the+first+number+divisable+by+10+in+the+vector%0A++++auto+itr+%3D+std::find_if(begin(vector),+end(vector),+is_divisable_by_10)%3B%0A%0A++++//+Print+the+first+element+we+find%0A++++std::cout+%3C%3C+*itr+%3C%3C+!%27%5Cn!%27%3B%0A%7D%27),l:%275%27,n:%270%27,o:%27C%2B%2B+source+%231%27,t:%270%27)),k:50,l:%274%27,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((g:!((h:output,i:(compilerName:%27x86-64+gcc+8.3%27,editorid:1,fontScale:14,fontUsePx:%270%27,j:1,wrap:%271%27),l:%275%27,n:%270%27,o:%27Output+of+x86-64+gcc+8.3+(Compiler+%231)%27,t:%270%27)),header:(),l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27),(g:!((h:compiler,i:(compiler:g83,deviceViewOpen:%271%27,filters:(b:%270%27,binary:%271%27,binaryObject:%271%27,commentOnly:%270%27,demangle:%270%27,directives:%270%27,execute:%270%27,intel:%270%27,libraryCode:%270%27,trim:%271%27),flagsViewOpen:%271%27,fontScale:14,fontUsePx:%270%27,j:1,lang:c%2B%2B,libs:!(),options:%27--std%3Dc%2B%2B17+-O2%27,selection:(endColumn:1,endLineNumber:1,positionColumn:1,positionLineNumber:1,selectionStartColumn:1,selectionStartLineNumber:1,startColumn:1,startLineNumber:1),source:1),l:%275%27,n:%270%27,o:%27+x86-64+gcc+8.3+(Editor+%231)%27,t:%270%27)),k:50,l:%274%27,m:50,n:%270%27,o:%27%27,s:0,t:%270%27)),k:50,l:%273%27,n:%270%27,o:%27%27,t:%270%27)),l:%272%27,n:%270%27,o:%27%27,t:%270%27)),version:4

- https://en.cppreference.com/w/cpp/language/lambda

C++11 Concurrency

C++11 provided the basis for concurrency support in C++. This includes things like:

- Threads (`std::thread`)
- Mutexes (`std::mutex`)
- Atomics (`std::atomic<T>`)

as well as numerous other support items that make writing parallel C++ code safer and easier.

Links

- https://godbolt.org/#z:OYLghAFBqd5QCxAYwPYBMCmBRdBLAF1QCcAaPECAMzwBtMA7AQwFtMQByARg9KtQYEAysib0QXACx8BBAKoBnTAAUAHpwAMvAFYTStJg1DIApACYAQuYukl9ZATwDKjdAGFUtAK4sGEgOykrgAyeAyYAHI%2BAEaYxBIAHKQADqgKhE4MHt6%2BASlpGQKh4VEssfFcSXaYDplCBEzEBNk%2BflyB1bUC9Y0ExZExcYm2DU0tue0jvf2l5YkAlLaoXsTI7BwA9BsA1B4MyCvEjMgAntvYqqzJ9CYaAIK3D2YAzGHI3ljbJs9uTEQseFMz2wj3Mr32H0wXx%2BTgUBCOrG%2BIPuYLekOhbhYXgImFUSNBLzRXk%2B3zcBAQCPQ%2BJRhIhxKhpIAbjUiMRqQ97mECNsWEwwhB5l9/FZ7tsxdsttshJhueSoQxBsRtqgqNtaKhUMltoQ4n9MgpHuLtmgGHDtYJtgqWAB9HXEPUCBTQgAi2y4GNJ2zMZm%2BIo5dyNkulsoQ8sVytV5MpTsM6G2mHobEETuScW2UcwTCpovFJrNXMtPmtGazTu%2Brp9zz9Rrz3ILCcwSYICmtqeIxYpmbj5cLNrtDtNEt7HejvtBObFkoAkvsjk306HtgpWFCuZhgGnDeLGZ49fRzdyFD4XdsNGOUROJTs3AiccqVtsAO4kADW2yoXn2jgEW7FTGxqBPq%2BJ4mAArFYZgAGxgc6ApCtWRrivwxAQAWeAnmeVbahi8aJowzatnEI5dr62rWNYgpHiw5GWOeAbiiY/jOnRv5XlKyRMI%2BDALqOl5wugIAgMyDgkKS/GCSWVLAjxXYGlWrHIahFroT2mEWNhnpWsRpakXgNEWIKjEIYhMmlgAdI21xMGs1rRNZL4QM%2BxAvvMdFGoxzEXvRk47AAUqgYSmegBqXop/5EF8UGSdsIBBQogqSWZ2gBQwApuQxl6SsoxAFnK6aoA0tCseJKDLNypKelRHo/NsYAcGBbgMHVLFMRwiy0JwoG8H4HBaKQqCcG4%2BlLssqwMi8PCkAQmhtYsL4gKBGj6JwkjdTN/WcLwCggEt029W1pBwLAMCICAZXJNi5CUGgLDJHQcQRCunCqAkkEALSQZI2zAMgyDbAkZnPLwmD4KyeACVwMiCCIYjsFIUPyEoajrbokOPvayScDw7WdWt%2B0bRwADy2IXdyKrbC972fd9v3/YD2wQB4t33UqYJcPMvB7Vo8yLKGWZDAKy0cKtpAsAtS09X1A0cFtO1TTNixHadN13fQZAUBAKssyAwAJM8pBYIygKYAAangmCPoTqY9ZNNC0DixDbRA0TrdEYSNCcWO8G7zDECchPRNoLJe6QN1NoTDC0J7%2BNYLyRjiDHeBHLUzLbfjuI1Ni6x9WuHX47QeDRPafseFg63wngYvcAdVAGMAChmxbVuMCH/DQ6I4jw23iMqOo%2BOo/ohjGPp%2BiF9tkCLJq36mpwb1vfx5amLRljtNsb2E2Ya%2B0MkklbQ2XR%2BBArjjG0QQMOgMyKnoqTpNPJ/XwU0%2BX2UQyQ50089GMnitHo791KMfQwgDBfhUKYX8cinzhNMIBsxX6LAUKNNYEgcYcC6qQSWvBpYU1eh9L6P0/oA2eAzXAhASCRWeOzTmCs5riyFiLMWi10HrWlrLXa1DSDzUYXnIGTD8YsPlvtHmQszB4ylptAR3NFjMkdpkEAkggA
- https://en.cppreference.com/w/cpp/thread