Subject: Re: CTRL + C = 659 Heap leaks
Posted by Novo on Thu, 11 Jul 2019 22:46:23 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 11 July 2019 18:15
I believe you are wrong. I believe Ctrl+C is equivalent of calling exit(). In that case, global variables get destructed, but local variables do not. This is also quite clear for the implementation of signals - they are asynchronouse and thus do not have to use the same stack so no stack unwinding is possible.

BTW, destructing global variables is the mechanism used to activate the leak checker :)

Mirek
I saw that MemDiagCls is a global static ...
Let's say only global statics get destroyed, and stack doesn't get unwinded. In such case every time you press CTRL-C you should get memory leaks. But this never happens to my apps ...
BTW, info for stack unwinding is always included into app, especially when you compile it with exception support (this is related to x64 Win and POSIX ABI).

"no stack unwinding is possible" - try to run an app with gdb and press CTRL-C. gdb will stop each thread in your app and will show you call stacks for each thread. Call stack for each thread can we unwinded, IMHO ...

This is my understanding of this process ...