
Subject: Re: How to respond when memory is exceeded

Posted by [mirek](#) on Mon, 18 Nov 2024 18:48:20 GMT

[View Forum Message](#) <> [Reply to Message](#)

koldo wrote on Mon, 18 November 2024 19:36mirek wrote on Mon, 18 November 2024 12:31koldo wrote on Mon, 18 November 2024 10:09Hello Mirek

This is a very simple test code to force the problem:

```
GUI_APP_MAIN
{
    WithMain<TopWindow> dlg;

    CtrlLayout(dlg, "Memory test");

    dlg.butTest.WhenAction = []{
        try {
            Buffer<int> data;
            data.Alloc(123456789012, 0);

            PromptOK("Memory is alloc");
        } catch (const std::bad_alloc& e) {
            Exclamation("Caught bad_alloc");
        }
    };

    dlg.Execute();
}
```

And this is the change in U++:

```
void OutOfMemoryPanic(size_t size)
{
    throw std::bad_alloc();

    /*char h[200];
    snprintf(h, 200, "Out of memory!\nU++ allocated memory: %d KB", MemoryUsedKb());
    Panic(h);*/
}
```

When pushing the button 5 times, no problem happens. Visually monitoring the Windows Task Manager, MemoryTest.exe seems to use the same memory all the time.

However, pushing the button one more time, MemoryLimitKb breached! Panic appears. It happens the same in debug and in release.

That is "buggy code" example. For something more real, try

```
#include <CtrlLib/CtrlLib.h>
```

```
using namespace Upp;
```

```
GUI_APP_MAIN
{
    TopWindow win;
    win.Title("Testing");
    win.WhenClose << [] {
        try {
            for(;;) {
                byte *data = new byte[1024 * 4096];
                for(int i = 0; i < 1024; i++)
                    data[i * 4096] = 1;
            }
        } catch (const std::bad_alloc&) {
            Exclamation("Caught bad_alloc");
        }
    };
    win.Run();
}
```

And try that in Linux...

That said, whatever, I think this really is totally futile, but I am going to change the code to actually throw `bad_alloc`, then catch it in main and do `Panic` there. I am NOT going to change the U++ code to work correctly in case `bad_alloc` is thrown, but maybe I will be able to compile some rules (like everything in try block has to be destructed). What a waste of time...

Hi Mirek

I think that the for with allocs is really an example of bad code.

Imagine that you are doing a new Notepad++, and the user wants to open a file of, let's say, XXX Tb.

Two options:

- a) Your Notepad++ would say "File too big", and will follow working normally.
- b) `Exclamation()` dialog, and program closes

I prefer option a)

- c) theide just views the file in read-only mode.

I prefer option c)

But really, notepad will need to allocate individual lines while loading, so the allocation pattern will look just like my example. Not like yours. Billions of small allocations until the memory is exhausted.

Mirek
