Subject: How to handle a lack of memory
Posted by koldo on Wed, 06 Apr 2016 08:47:07 GMT
View Forum Message <> Reply to Message

Hello all

Now String::Cat() as many U++ functions calls SysAllocRaw() to allocate a String.

However if requested size is too big, SysAllocRaw() unconditionally shows a panic message and stops the program.

I propose you to throw an exception in this case, so that the main program can handle this properly.

In the old days we did this :) :
```
char *a = malloc(size);
if (a == 0) {
 printf("\nNot enough memory available");
 ... code that handles this properly ...
 ... program can follow running without crashing ...
}
```

Subject: Re: How to handle a lack of memory
Posted by mirek on Thu, 07 Apr 2016 20:59:46 GMT
View Forum Message <> Reply to Message

It is not so easy...

Frankly, it all boils down to VectorMap::Add implementation as example:

```
 T&      Add(const K& k, const T& x)          { key.Add(k); return value.Add(x); }
```

Now if value.Add runs out of memory, VectorMap is left in inconsistent state. Not that this is just example, practical code is full of examples like this. E.g. what is wrong with this code:

```
struct Foo {
   Bar *ptr;

   Foo() { ptr = new Bar; }
   ~Foo() { delete ptr; }
}
```

It is certainly possible to solve that by adding exception block into Add, but correctly resolving this issue everywhere would result in significant increase in codebase (and probably decreased

performace). And it is virtually untestable.

Plus, in 64 bit platform (but often with 32-bits too), you are rather going to freeze system first (because of swap trashing) that actually get to out-of-memory.

In practice, I do not remember getting out-of-memory in other situations that program bugs. Have you ever got out-of-mem in theide?

Mirek

Subject: Re: How to handle a lack of memory
Posted by koldo on Fri, 08 Apr 2016 07:00:36 GMT
View Forum Message <> Reply to Message

Hello Mirek

For example there is a program with a virtual ArrayCtrl that takes data from a huge file.
If user selects and copy the data to the Clipboard, program may crash.

Subject: Re: How to handle a lack of memory
Posted by mirek on Fri, 08 Apr 2016 07:09:34 GMT
View Forum Message <> Reply to Message

koldo wrote on Fri, 08 April 2016 09:00Hello Mirek

For example there is a program with a virtual ArrayCtrl that takes data from a huge file.
If user selects and copy the data to the Clipboard, program may crash.

Well, I have to say that generally, this is one of areas where I am not really that much sure that used approach (Panic) is correct.

However, somewhat I suspect that here the sensible solution is rather checking the size of selection before starting the process. The advantage is that it would solve "freezing" problem too.

I know that it is not always possible. And sometimes such check is difficult - but making the code "memory exception safe" reliably is much more difficult than that, I am afraid...

Mirek

Subject: Re: How to handle a lack of memory
Posted by koldo on Mon, 11 Apr 2016 06:33:57 GMT
View Forum Message <> Reply to Message

Hello Mirek

I agree with you. However I think that this issue should have to be solved some time.
Maybe it is not possible to solve it soon for all cases, but maybe it is possible to solve it for a big percentage of them.
Remember the Pareto principle (for example, with 20% of the effort you may solve the 80% of the problem) :).

Subject: Re: How to handle a lack of memory
Posted by mirek on Mon, 11 Apr 2016 10:06:00 GMT
View Forum Message <> Reply to Message

BTW, for reference, Qt is not much advanced in handling this problem either:

http://doc.qt.io/qt-4.8/exceptionsafety.html

IMO: Really not worth spending time with this. I could imagine replacement function for Panic, but I do not think that it is worth the effort to make the code "bad_alloc exception safe".
