Subject: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by nixnixnix on Fri, 22 Aug 2008 22:53:02 GMT
View Forum Message <> Reply to Message

Having being persuaded that new[] is bad and that Buffer::Alloc() is way better, I have hit a snag. When new[] fails it returns a NULL pointer but when Alloc fails it panics and we get an "Out of Memory" message. Is there a way to check if something can be built before we try please?

This is particularly awkward when creating a large Image. Is there a way to know ahead of time that it will fail and is it possible to make bigger ones under a 64bit OS?

Cheers,

Nick

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by cas_ on Sat, 23 Aug 2008 18:10:48 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Sat, 23 August 2008 00:53Having being persuaded that new[] is bad

Bad? In what way?

Quote:When new[] fails it returns a NULL pointer

No, it throws std::bad_alloc.

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by mirek on Sun, 24 Aug 2008 14:31:46 GMT
View Forum Message <> Reply to Message

Well, not in U++.

In U++, it panics with "out of memory".

While this practice might seem a little bit harsh, it "solves" a couple of quite tedious problems. E.g. we can require that copy constructors never thrown an exception

In practice, in the light of existence of OOM killer, I guess this issue hardly matters in any way.

Mirek

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by mirek on Sun, 24 Aug 2008 14:36:53 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Fri, 22 August 2008 18:53
This is particularly awkward when creating a large Image. Is there a way to know ahead of time that it will fail and is it possible to make bigger ones under a 64bit OS?

I guess using Raster and GetSize, then do cx * cy * 4 and make sure it is less than 100MB should work... and is the only reasonable solution to the problem anyway.

Other than that, it is really quite hard to guess which allocation will fail. The real problem is "define failure". On system with 10GB swap and 512MB RAM, you might be able to open very big image, but system can become completely unresponsive (HD light on for hours, mouse does not move, ... etc...)

Mirek

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by nixnixnix on Tue, 26 Aug 2008 23:49:22 GMT
View Forum Message <> Reply to Message

Because of the data types my app handles, it really isn't practical to run it with less than 2GB RAM and preferably 4GB and probably a dual or quad core machine (in fact a lot of my users are on dual quad core CPUs with 4 GB or more). Even with my super small grid2d objects instead of 2d arrays, I am still going to get people crashing through the 2GB 32 bit memory limit so the idea that an image over 100MB is unwieldy does not apply in this case.

Is there a way that UPP users could setup their app for their chosen maximum image size?

Nick

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by mirek on Wed, 27 Aug 2008 10:49:45 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Tue, 26 August 2008 19:49
Is there a way that UPP users could setup their app for their chosen maximum image size?

Nick

I am not sure I understand the question....

Mirek

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by amrein on Fri, 29 Aug 2008 17:52:38 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Wed, 27 August 2008 01:49Because of the data types my app handles, it really isn't practical to run it with less than 2GB RAM and preferably 4GB and probably a dual or quad core machine (in fact a lot of my users are on dual quad core CPUs with 4 GB or more). Even with my super small grid2d objects instead of 2d arrays, I am still going to get people crashing through the 2GB 32 bit memory limit so the idea that an image over 100MB is unwieldy does not apply in this case.

Is there a way that UPP users could setup their app for their chosen maximum image size?

Nick

If you mean "How can I know the image size in memory or image x,y,dx,dy attributes before loading it?" then I think the short answer is you can't at present. If you want to know the file size on disk before loading it then you can use GetSize() from FileIn.

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by nixnixnix on Sat, 30 Aug 2008 17:54:50 GMT
View Forum Message <> Reply to Message

No I tend to create the huge images I want to display but the memory alloc fails not because of lack of memory but because of some other preset limit so far as I can tell. It is this limit that I would like to remove or be able to change.

Nick

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by mirek on Sat, 30 Aug 2008 18:20:00 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Sat, 30 August 2008 13:54No I tend to create the huge images I want to display but the memory alloc fails not because of lack of memory but because of some other preset limit so far as I can tell. It is this limit that I would like to remove or be able to change.

Nick

Well, that is interesting.

Other possible issue that comes to mind is Win32 address space fragmentation. Are you in 32-bit?

Besides, I was also thinking about your problem and I believe that one possible correct solution is

not to load everything in memory. Maybe you could process the image as file stream? That would allow processing of images that are REALLY BIG

Mirek

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by zsolt on Sat, 30 Aug 2008 20:26:24 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 30 August 2008 20:20nixnixnix wrote on Sat, 30 August 2008 13:54No I tend to create the huge images I want to display but the memory alloc fails not because of lack of memory but because of some other preset limit so far as I can tell. It is this limit that I would like to remove or be able to change.

Nick

Well, that is interesting.

Other possible issue that comes to mind is Win32 address space fragmentation. Are you in 32-bit?

Besides, I was also thinking about your problem and I believe that one possible correct solution is not to load everything in memory. Maybe you could process the image as file stream? That would allow processing of images that are REALLY BIG

Mirek

Or as memory mapped files?

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by mirek on Sun, 31 Aug 2008 07:19:54 GMT
View Forum Message <> Reply to Message

zsolt wrote on Sat, 30 August 2008 16:26
Or as memory mapped files?

That does not necessary help. Memory mapped files have the same address space fragmentation problem as normal large allocs.

Mirek

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by mirek on Sun, 31 Aug 2008 07:22:05 GMT
View Forum Message <> Reply to Message

P.S.: In bazaar, there is RasterCtrl (and RasterCtrlTest example), that demonstrates how to handle large images without loading them into memory.

There is however a possible problem that raster decoder has to implement stream loading (some decoders just load the whole Image in and then simulate the stream).

Mirek

Subject: Re: [FEATURE REQUEST] could Alloc fail a little more gracefully?
Posted by nixnixnix on Mon, 01 Sep 2008 18:36:10 GMT
View Forum Message <> Reply to Message

Hi Mirek,

This is specifically for images that I want to display on screen.

I already process some very large ~2GB data grids by doing something similar to what you suggest.

So far I am having trouble compiling for 64 bit under vista 64 (different thread) but once I get past that I'll let you know if I still get the same problems with large images.

At present, I am handling this by sampling every one in 2 or one in 4 pixels(grid values) but I think I'll need to move to support image pyramids eventually. My worry about the image size limit was that it would not go away under 64 bit but you're suggesting it probably will so that is good.

Cheers,

Nick