Doc No: J16/99-0040, WG21/N1216
Date: October 22, 1999
Reply to: Stephen D. Clamage,
`stephen.clamage@sun.com`

Allocating Zero Bytes or Objects

This paper addresses Library Working Group issue 9: `operator new(0)` calls should not yield the same pointer.

The issues:

Section 3.7.3 discusses properties of allocation and deallocation functions. It seems to say it applies only to user-written functions. If it doesn't apply to the default library versions of `operator new` and `operator delete`, those functions are under-specified.

Section 3.7.3.1/2 says an allocator cannot return a null pointer value in response to a request for zero size. A `nothrow` allocator in particular must therefore never fail. Since the available address space could become exhausted, this requirement means that an allocator need not return different pointer values for consecutive requests.

The choices:

1. A request for zero size cannot fail, meaning returned values need not be distinct.
2. A request for zero size can fail, but returns distinct values if it succeeds.

The advantage of option 1 seems to be that an implementation could choose always to return a pointer to the same static byte, and ignore requests to delete the pointer. Allocation would be very fast, and no resources would be consumed by a zero-size request. On the other hand, one does not expect many zero-size allocations in a program.

The advantage of option 2, returning unique addresses, is that algorithms on arrays can be simplified. The typed pointer plus the number of elements always points just past the last element, and we get behavior like `begin` and `end` iterators. No special checks for zero elements or a special pointer value are needed. In addition, two arrays of zero elements are in fact different objects, and therefore should have different addresses.

Possible options already rejected are to require or to allow a null return for successful allocation of zero size.

Proposed handling:

The Library Working Group recommends adopting option 2, which requires the following changes to the standard:

The last paragraph of 3.7.3 should be changed
from

> Any allocation and/or deallocation functions defined in a C++ program shall conform to the semantics specified in 3.7.3.1 and 3.7.3.2.

to

> Any allocation and/or deallocation functions defined in a C++ program, including the default versions in the library, shall conform to the semantics specified in 3.7.3.1 and 3.7.3.2.

3.7.3.1/2, next-to-last sentence should be changed
from

> If the size of the space requested is zero, the value returned shall not be a null pointer value (4.10).

to

> Even if the size of the space requested is zero, the request can fail. If the request succeeds, the value returned shall be a non-null pointer value (4.10) *p0* different from any previously returned value *p1,* unless that value *p1* was since passed to an `operator delete`.

5.3.4/7 currently reads:

> When the value of the expression in a direct-new-declarator is zero, the allocation function is called to allocate an array with no elements. The pointer returned by the new-expression is non-null. [Note: If the library allocation function is called, the pointer returned is distinct from the pointer to any other object.]

Retain the first sentence, and *delete* the remainder.

18.4.1 currently has no text. *Add* the following:

> Except where otherwise specified, the provisions of 3.7.3 apply to the library versions of `operator new` and `operator delete`.

18.4.1.3, *add* the following text:

> The provisions of 3.7.3 do not apply to these reserved placement forms of `operator new` and `operator delete`.