

New in SmartHeap 8.0

Improved performance

The SmartHeap 8.0 multi-threaded libraries contain performance optimizations making them up to 2x faster than version 7. The performance improvement is greatest on the Windows operating system.

Note that the general performance improvements in SmartHeap version 8, and those described under `MemDefaultPoolThreadExclusive`, below, are available only when running on single-processor (including hyperthread-enabled Intel processor) systems. The performance enhancements are enabled on SMP systems only in the SmartHeap/SMP product.

New API controls thread-specific heap usage

SmartHeap 8.0 introduces a new API, `MemDefaultPoolThreadExclusive`, that can be used to greatly enhance performance and simplify heap object deallocation in applications that use heap exclusively in one or more threads. Applications that will benefit from this new API are those that have one or more heap-intensive threads which exclusively free the memory they allocate.

Use `MemDefaultPoolThreadExclusive` only in threads that exclusively free all heap pointers that are allocated by that thread after the call to `MemDefaultPoolThreadExclusive`.

The new API has the following prototype:

```
MEM_BOOL MemDefaultPoolThreadExclusive(unsigned flags);
```

Specify one of the following values for flags:

- **MEM_THREAD_EXCLUSIVE_OFF**: default behavior – current thread uses process-wide default memory pool. Heap operations are synchronized with other threads.
- **MEM_THREAD_EXCLUSIVE_ON**: creates thread-specific default memory pool for the current thread. Synchronization with other threads is not needed, resulting in improved performance.

Specify this value only if all allocations created with `malloc`, `calloc`, `new`, or `realloc` in the calling thread are freed exclusively by the same thread. Can be combined disjunctively with **MEM_FREE_ON_THREAD_TERM**.

- **MEM_FREE_ON_THREAD_TERM**: creates thread-specific default memory pool for the current thread. All allocations created in the current thread are automatically freed at thread termination, simplifying and improving performance of storage management. Note that you still must call `delete` for objects created with operator `new` in order for destructors to execute, though you can override operator `delete` for such objects with an empty operator `delete` definition if desired (if the objects are used exclusively in thread-specific memory pools) since SmartHeap will automatically free the

memory at thread termination.

Specify this value only if all allocations created with **malloc**, **calloc**, **new**, or **realloc** in the calling thread are either not freed or are freed exclusively by the same thread, and if there are no references to allocations created in the calling thread from other threads after the calling thread terminates. Can be combined disjunctively with **MEM_THREAD_EXCLUSIVE_ON**.

Call this API from each thread where you want one or both of the above behaviors.

[windows only] Improved Debug SmartHeap compiler debug info support

SmartHeap Version 8 now uses dbghelp.dll for symbol and file/line information to support heap error diagnostics, which provides greater compatibility with more compiler and debugging information formats.

