MemError Return error code of last **Memory Manager** function

#include <Memory.h> **Memory Manager**

OSErr **MemError( );**
returns Error code of recent Memory Manager function; 0=noErr

MemError returns the OSErr code of the most recent call to a Memory Manager function.

Returns: an OSErr (a.k.a. short). Common Error/Return Code returns:

noErr	(0)	No error
memFullErr	(-108)	No room in heap
nilHandleErr	(-109)	Illegal operation on a NIL handle
memWZErr	(-111)	Illegal operation on a free block
memPurErr	(-112)	Illegal operation on a locked block
memLockedErr	(-117)	Can't move a locked block

Notes: Before returning to an application, the Memory Manager stores an error/return code into a global variable. If you call **MemError**, you get a copy of that value. Note that this value is affected only by calls made directly by the application (and not errors made indirectly; e.g. via a toolbox function that calls the Memory Manager indirectly).

C programmers may access the global variable MemErr (at 0x0220) directly. For instance:

```
HPurge( myHandle );
if ( MemError() ) { . . . process the error . . . }

/* faster alternative . . . */
if ( MemErr )  { . . . process the error . . . }
```

ASM programs can check for return code values in the low word of the D0 register (with some exceptions).