HyperCard: General Problems with XCMDs and XFCNs

This article last reviewed: 24 August 1989

Using your own or someone else's XCMD or XFCN involves some risks. Some XCMDs may cause system crashes or unusual problems to appear. These problems usually result from not programming enough error protection into an XCMD and from lack of testing of the XCMD.

When running a new XCMD, first execute it within a test HyperCard stack. If a problem does occur within and corrupting the test stack, your real stack will not be damaged. Also, be sure to check the result of the XCMD or XFCN for any error messages. It's a good idea to put this statement into the script after executing the command:

"Put the result into the message"

Here are some possible XMCD or XFCN causes of system locks up:

- The memory management system is purging needed data of the XCMD or XFCN, because the XCMD or XFCN has not taken precautions to protect it.

- The XCMD or XFCN has broken the master-pointer block, and the problem is not uncovered until some random, future action.

- The memory manager's application stack has become fragmented causing an out-of-memory error, because the XCMD or XFCN has allocated memory, locked it down, and has not removed the data after it was finished.

- There just is not enough memory available.

There are several ways for XCMDs or XFCNs to crash. Almost all are due to lack of error checking and or defensive programming. Some are simply due to trying to use HyperCard in ways for which it was not designed. Anyone using an XCMD or XFCN needs to test the XCMD or XFCN in a stack by itself and slowly add and execute other XCMDs. Any one of the additions may fragment the memory manager's stack, corrupting the master pointer block, or using up memory by leaving data floating around.

An XCMD or XFCN may not immediately cause problems. For example, a corrupted, master-pointer block may continue to work until a call is made to a once-valid pointer to your data. Once but is now corrupted the pointer points to garbage. The effect could be returning garbage to the stack, but usually, HyperCard

chokes on the unexpected garbage data and crashes. Such a crash can take the system with it.

Apple does not support extensions that are not part of the HyperTalk language. The ability to add extensions was created, so that customers can enhance their stacks. However, each programmer must take the responsibility to check for errors and to report any problems to the user when writing the XCMD or XFCN. Be wary when faced with extensions from unknown sources.

Finally, a user's needs may not be appropriate for the capabilities of HyperCard. There are some supported XCMDs, like those available within legitimate HyperCard software products. We suggest HyperTalk programmers use only those XCMDs that they are confident with or that carry a guarantee for future support.

There are many other programming environments that support and handle advanced programming requirements. HyperTalk is not meant to be a replacement for programming languages like Pascal or C.

Keywords: <None>

==========

This information is from the Apple Technical Information Library.

19960215 11:05:19.00

Tech Info Library Article Number: 4713