

Re: Reading text file characters

Source: <http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2004-12/3263.html>

From: Jim Mack (jmack_at_mdxi.nospam.com)

Date: 12/28/04

Date: Tue, 28 Dec 2004 12:47:45 -0500

Don wrote:

- > *If anyone is still with this thread, I did a hex dump suggested by*
- > *Duane and got the following that appears AFTER what should be the end*
- > *of the file – or at least what appears as the end of the file when the*
- > *file is displayed in a text editor or word processor.*
- >
- > *What doesn't show up in the numbers below is the last line of the hex*
- > *dump which is just a double underscore, but that seems to show up with*
- > *all files so I'm assuming it just marks the end up the dump?*
- >
- > *Any idea what these hex numbers represent in the text file that seem*
- > *to occur right at the point where the ^@ character appears which seems*
- > *to be the source of problems?*
- >
- > *00 0d 0a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a 1a*
- > *1a 1a*

Hex 1A is ASCII EOF. Older text editors (really old) would sometimes fill their file buffer with EOF characters, then overwrite with text as needed. When they closed the file they wrote the entire buffer, which left a trail of EOFs in the file. Since the first EOF marks the end, the extras were ignored. What created this file?

The ^@ is the null byte you show as the first byte in the dump. It shouldn't cause any problems, but it depends on how you're opening and reading the file whether null bytes are an issue. The usual strategy would be to open as binary, Get the file into a byte array, then replace all 0 bytes with spaces.

--

Jim Mack
MicroDexterity Inc
www.microdexterity.com