CFX_HEX ™

A Useful Debugging Tool For Programmers

Installation & User Guide

For Windows, Linux & Solaris

Software Version	2.0
Document	cfxhex.pdf
Published	05/01/2003

CFXWorks, Inc.
5365 Chelsen Wood Drive, Duluth, Georgia 30097

Email: sales@CFXWorks.com http://www.CFXWorks-Coldfusion.com

Printed in the United States of America.

© 2002 CFXWorks, Inc. All Rights Reserved

1. INTRODUCTION 3

1.1. THANK YOU 3
1.2. OVERVIEW 3

2. TECHNICAL ISSUES 5

2.1. SPECIFICATIONS 5
2.2. INSTALLATION OF COLDFUSION CFX TAG 6
2.3. MINIMUM SYSTEM REQUIREMENTS 7

3. OTHER STUFF 8

3.1. SOFTWARE LICENSE 8
3.2. EXPORT LIMITATIONS 8
3.3. SUPPORT 8
3.4. WARRANTY 8

APPENDIX A. TAG PARAMETERS 9

APPENDIX B. ERROR CODES 10

APPENDIX C. SAMPLES 11

1. INTRODUCTION

1.1. Thank You

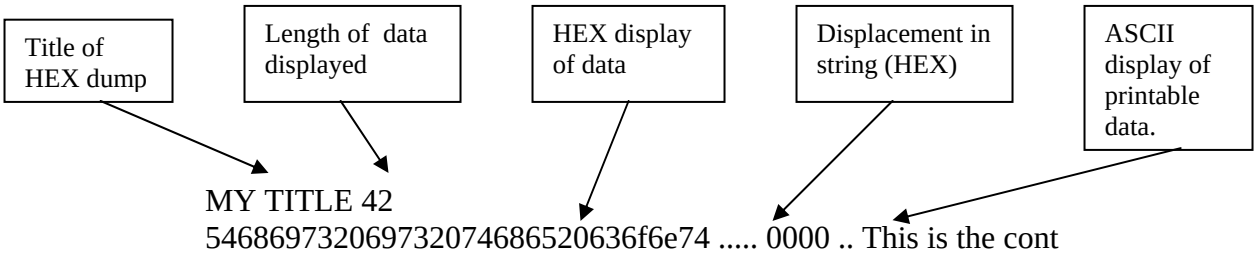
Thank you for downloading this offering. I developed the original code for this routine in 1978 on an Apple 2 computer and have ported it to nearly every platform imaginable. I hope you find it as useful as I have over the years. If you feel a need to contact me directly, please send an email to anickles@cfxworks.com.

A companion product to CFX_HEX is CFXHEX. CFXHEX is a Windows DLL that has been compiled using the using the __stdcall calling convention. Therefore, it should be callable from any programming language capable of calling DLLs using this calling convention. This includes C, C++, VB, Access, Pascal and Fortran. A shared object version of CFXHEX is also available for Linux and Solaris.

1.2. Overview

Have you ever spent hours trying to debug a program just to find out that the data you were working with was different than you expected? Perhaps the data contained unprintable characters, or contained characters that were translated by ColdFusion. The CFX_HEX offering is useful because it can be used to create a HEX display of a ColdFusion variable or of a file. Once you see the exact contents of the data, it takes a lot of the guesswork out of the programming process.

CFX_HEX is a ColdFusion custom tag that will HEX encode either a variable or a data file. It writes its output either to a file or to a variable. The format of data dumped to a file is as follows:




```
MY TITLE 42
546869732069732074686520636f6e74 ..... 0000 .. This is the cont
656e7473206f66206d7920696e707574 ..... 0010 .. ents of my input
207661726961626c6521             ..... 0020 ..  variable!
END OF DATA
```

The format of data returned in a log file is as follows. First it displays the "MESSAGE" parameter coded by the programmer, followed by the length of the data displayed. In the above illustration the "MESSAGE" parameter contained "MY TITLE" and the length of data displayed is 42. Next it displays for each line of data, the HEX data followed by the displacement (in hex) from the start of the input data, Finally, it displays for each line, the ASCII character representation of each character if it is printable. This format is left over from mainframe core dumps from a prior life.

The format of data returned in a variable, if the programmer selects this option, is simply the HEX output. For example for the above input variable, the output would be:

```
546869732074686520636f6e74656e7473206f66206d7920696e707574207
661726961626c6521
```

If data is returned in a variable, the return code contains the length of the variable. If data is returned in a file, a zero return code indicates success.

2. TECHNICAL ISSUES

2.1. Specifications

The programming specification for CFX_HEX is as follows:

```
                                        DefaultValue
<CFX_HEX
        ACTION =        "tf"            "tf"            hex encode text to file
                        "ff"                            hex encode file to file
                        "tt"                            hex encode text to text
                        "ft"                            hex encode file to text
        MODE =          0               0               create a new output file
                        1                               concatinate to existing output file
        TEXT =          "1234567890"                    text to be encoded
        LENGTH =        N                               display this many characters
        NAME =          "RESULT"        "TEXT"          result
        FILEIN =        "FILEIN"        "filein.txt"    input file
        FILEOUT =       "FILEOUT"       "hex.log"       output file
        MESSAGE =       "MY DUMP"       "HEX DUMP"      title displayed in log file
        DISPLAY =       0                               display nothing
                        1                               display log
                        2                               write log to logfile "hex.log"
>

Response:

RC =                    0                               success for "tf" and "ff"
                        N                               length of output  for "tt" and ""ft"
                        -1                              zero length input
                        -2                              name error
                        -3                              invalid filein
                        -4                              invalid fileout
                        -5                              memory error
                        -99                             dll corruption error
```

Figure 1 – Specification For ColdFusion Tag

For a detailed discussion of CFX_HEX parameters, please see ***Appendix A. Tag Parameters***. For a detailed discussion of CFX_HEX error codes please see ***Appendix B. Error Codes.*** For specific examples of how to deploy this tag, please see ***Appendix C. Samples.***

2.2. Installation of ColdFusion CFX Tag

The installation process for ColdFusion CFX tag versions of this offering is a simple three-step process:

1. CFX_HEX is distributed as zip file (filename.zip). The actual filename is specific to each licensee. You will need a "password" to unzip this file. The password is sent to the licensee by email when the tag is ordered. The command you must enter at a command line is as follows:

Windows:	pkunzip filename –s[password]
Linux:	unzip –j filename
Solaris:	unzip –j filename

   Note that commands and filenames in Linux and Solaris are case sensitive. The "-j" in the above Linux and Solaris commands stands for "junk", or do not make directories. Without this parameter, the unzip command will create subdirectories within the current working directory. On Linux and Solaris, after you have unzipped the zip file, you will have to use the "chmod u+x cfx_checkemail.so" command to make this file executable.

   When you unzip this file, the code and documentation is extracted to the current working directory. The zip file contains the following files:

   - CFX_HEX.dll (Windows) or "cfx_hex.so" (Linux and Solaris) – the executable.
   - HEX0.cfm – sample of a hex dump of a ColdFusion variable.
   - HEX1.cfm – sample of a hex dump of a file.
   - HEX2.cfm – sample of a hex display of a ColdFusion variable.
   - HEX3.cfm – sample of a hex display of a file.
   - CFXHEX.pdf – this document.

2. On versions of Coldfusion prior to MX, ColdFusion recommends that you copy the CFX_HEX executable to the \CFUSION\CFX directory for execution. This wasn't absolutely necessary, but it was best practices. On Windows versions of MX, this directory would be \CFusionMX\CustomTags\cfx. On Linux, this directory would be /opt/coldfusionmx/cfx.

3. ColdFusion also requires you to register the tag using the ColdFusion Administrator. The registration dialog will look similar to the following:

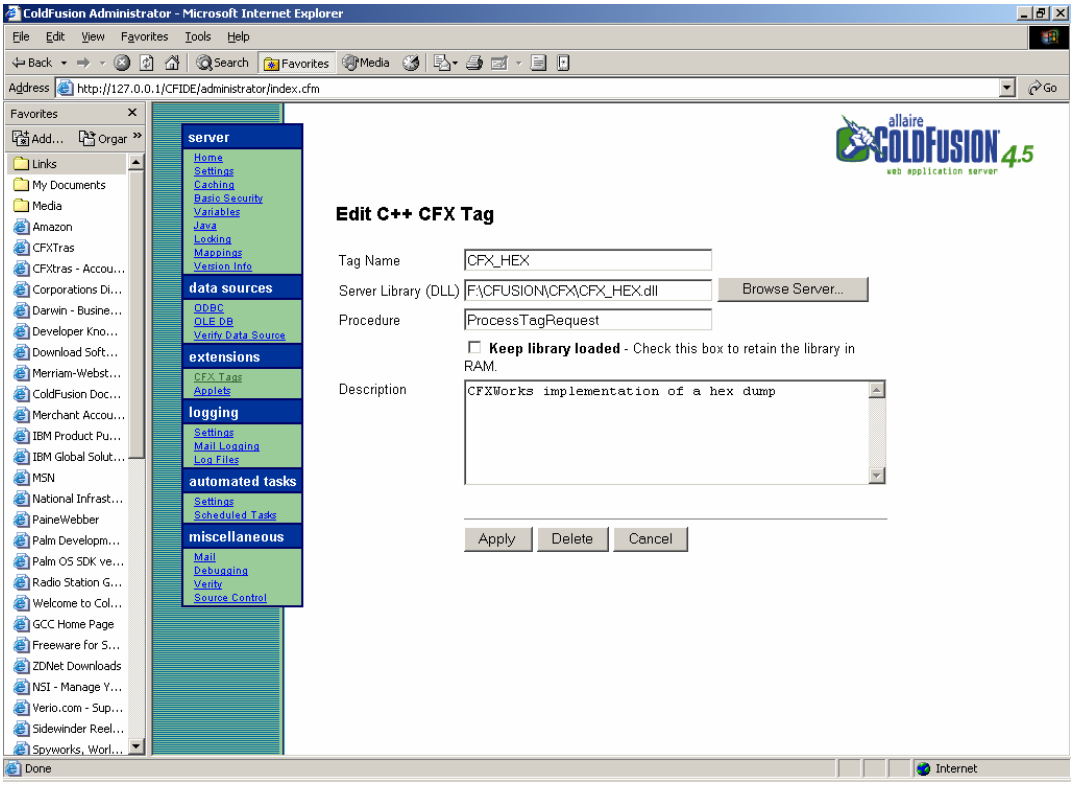


Figure 2- Registering CFX_HEX

You should name the tag as shown in the ***Figure 2***. You must tell ColdFusion where the executable is stored. For performance reasons you should select "Keep library loaded" however this is not required if the tag will not be in heavy use. You can enter a description as shown above. For Windows, if you do not check the "Keep library loaded" checkbox, the dll is reloaded each time the tag is executed. Therefore, if you install a new version of the dll, it is automatically refreshed at runtime. On Linux, if you install a new version of the executable, you must stop and start the Coldfusion server to refresh the executable.

That's it! The entire installation process should take only a few minutes to complete.

2.3. Minimum System Requirements

These offerings have been tested on Windows 2000, Linux (Red Hat) and Solaris 8 platforms. The Windows version should function properly on all Win32 platforms. The Linux version should function on all Intel versions of Linux. The Solaris offering should function on Solaris 8 and 9.

The ColdFusion tag versions of this offering requires that it be installed on a system using a properly configured copy of ColdFusion 4.0, 4.5, 5.0 or MX.

3. OTHER STUFF

3.1. Software License

Users are granted a license to use this software without restriction.

3.2. Export Limitations

None.

3.3. Support

Please send your questions to support@cfxworks.com.

3.4. Warranty

No warranty is expressed or implied for the CFX_HEX offering.

APPENDIX A. TAG PARAMETERS

Parameter		Default Value	Requred	Description
ACTION =	"tf" "ff" "tt" "ft"	“tf”	optional	hex encode text to file hex encode file to file hex encode text to text hex encode file to text
MODE =	0 1	0		create a new output file concatinate to existing output file
TEXT = “Text to be encoded”		none	required for “tf” & “tt”	Null terminated text string.
LENGTH = N		input length	optional	Encode only the first N characters in the input.
NAME = “RESULT”		“TEXT”	optional	Null terminated text string.
FILEIN = “filename”		“filein.txt”	optional	Fully qualified file name (1)
FILEOUT = “filename”		“hex.log”	optional	Fully qualified file name (1).
MESSAGE		“HEX DUMP”	optional	Text displayed as title in log file.
DISPLAY = value		0	optional	0 – no display (2) 1 – display log 2 – create a disk log

(1) If a fully qualified filename is not specified, the file is read from or written to the current working directory. This directory most likely is c:\winnt\system32. If you execute the tag using DISPLAY=1, the current working directory will be displayed.

(2) If ACTION = “tf” or “ff” is coded, the output is automatically written to log file. If DISPLAY = “1” or “2” is coded, additional information is written to the log file. In either case the log is written to the current working directory of ColdFusion using the filename “hex.log” which most likely is c:\winnt\system32 unless you code the FILEOUT parameter.

APPENDIX B. ERROR CODES

RC	Description	Action of tag
n	For “tt” and “ft”, a positive value indicates that HEX encoding was been successful. For “tf” and “ff” a zero value indicates success	The value is the length of the output text.
-1	Indicates that the length of the input variable or file was zero.	The tag takes no action.
-2	Invalid NAME. This means that the variable name passed to the tag contained blanks or non-ASCII characters.	No output is created.
-3	FILIIN error. Most likely the file cannot be found or the directory specified does not exist.	The status of the output text or file is unknown.
-4	FILEOUT error. Most likely the directory or drive specified does not exist.	The status of the output text or file is unknown.
-5	Memory allocation error. This means that you do not have enough memory on your system to complete the encoding/decoding task.	No output is created.
-99	This means that the DLL or shared object is corrupt. Please notify CFXWorks and download a fresh copy.	No output is created.

© 2002 CFXWorks, Inc. All Rights Reserved

APPENDIX C. SAMPLES

ColdFusion CFX Tag Example:

***Figure** 4* illustrates just how simple it is to use this tag. Two of the included samples are shown in the following figure.

```
Sample - ·HEX0.cfm - sample of a hex dump of a ColdFusion variable.

        <cfset INPUT = "1234567890">
        <CFX_HEX
                ACTION="tf"
                MODE=0
                FILEOUT="hex.log"
                MESSAGE="MY TITLE"
                TEXT="#INPUT#"
                DISPLAY=0
                >

        Response:
        RESPONSE = OK-39!
        RC = 39

        Contents of hex.log:
        MY TITLE 39
        54686973207468652063 6f6e74656e74 ..... 0000 .. This the content
        73206f66206d7920696e707574207661 ..... 0010 .. s of my input va
        726961626c6521                   ..... 0020 .. riable!
        END OF DATA

Sample - HEX1.cfm - sample of a hex dump of a file .

        <CFX_HEX
                ACTION="ff"
                MODE=0
                FILEOUT="hex.log"
                MESSAGE="MY TITLE"
                FILEIN="f:\aen\cfx_hex\text"
                DISPLAY=0
                >

        Response:
        RESPONSE = OK-34!
        RC = 34

        Contents of hex.log:
        MY TITLE 34
        54686973206973207468652063 6f6e74 ..... 0000 .. This is the cont
        656e74206f66206d792066696c65210d ..... 0010 .. ent of my file!
        0a1a                             ..... 0020 ..
        END OF DATA
```

Figure 4 - Samples