Sourcecode: Example2.c

COLLABORATORS			
	TITLE : Sourcecode: Example2.c		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Sourcecode: Example2.c

1.1 Example2.c

```
/*********************************************************/
/*                                                       */
/* Amiga C Encyclopedia (ACE)           Amiga C Club (ACC) */
/* --------------------------           ------------------ */
/*                                                       */
/* Manual:  AmigaDOS                    Amiga C Club       */
/* Chapter: Files                       Tulevagen 22       */
/* File:    Example2.c                  181 41  LIDINGO    */
/* Author:  Anders Bjerin               SWEDEN             */
/* Date:    93-03-11                                     */
/* Version: 1.0                                          */
/*                                                       */
/*   Copyright 1993, Anders Bjerin - Amiga C Club (ACC)  */
/*                                                       */
/* Registered members may use this program freely in their */
/*     own commercial/noncommercial programs/articles.   */
/*                                                       */
/*********************************************************/

/* This program will reads ten integer values from an */
/* already existing file called "HighScore.dat" which */
/* is located on the RAM disk. (This file was created */
/* by Example1.)                                      */



/* Include the dos library definitions: */
#include <dos/dos.h>

/* Now we include the necessary function prototype files:         */
#include <clib/dos_protos.h>       /* General dos functions...    */
#include <stdio.h>                 /* Std functions [printf()...] */
#include <stdlib.h>                /* Std functions [exit()...]   */



/* Set name and version number: */
UBYTE *version = "$VER: AmigaDOS/InputOutput/Example2 1.0";
```

```
/* Declared our own function(s): */

/* Our main function: */
int main( int argc, char *argv[] );



/* Main function: */

int main( int argc, char *argv[] )
{
  /* A "BCPL" pointer to our file: */
  BPTR my_file;

  /* The numbers: (10 integers will be collected) */
  int my_highscore[ 10 ];

  /* Store here the number of bytes actually collected: */
  long bytes_read;

  /* A simple loop variable: */
  int loop;



  /* Since we want to collect some data from an already existing */
  /* file we must open the file "HighScore.dat" as an old file:  */
  my_file = Open( "RAM:HighScore.dat", MODE_OLDFILE );

  /* Have we opened the file successfully? */
  if( !my_file )
  {
    /* Inform the user: */
    printf( "Error! Could not open the file!\n" );

    /* Exit with an error code: */
    exit( 20 );
  }

  /* The file has now been opened: */
  printf( "File open!\n" );



  /* Load the values: */
  printf( "Loading values...\n" );

  /* Collect 10 integers (40 bytes): */
  bytes_read = Read( my_file, my_highscore, sizeof( my_highscore ) );

  /* Did we get all data? */
  if( bytes_read != sizeof( my_highscore ) )
  {
    /* No! We could not read all values! */
```

```
    printf( "Error! Could read all values!\n" );

    /* Close the file: */
    Close( my_file );

    /* Exit with an error code: */
    exit( 21 );
  }
  else
  {
    /* OK! */
    printf( "All values were successfully collected!\n" );
  }



  /* Print the values: */
  for( loop=0; loop < 10; loop++ )
    printf( "Highscore[%d] = %8d\n", loop, my_highscore[ loop ] );



  /* Close the file: */
  if( Close( my_file ) )
    printf( "File closed!\n" );
  else
    printf( "Error! File could not be closed!\n" );

  /* Remember that even if the file could not be */
  /* closed we must NOT try to close it again!   */



  /* The End! */
  exit( 0 );
}
```