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: Introduction                Tulevagen 22       */
/* File:    Example2.c                  181 41  LIDINGO    */
/* Author:  Anders Bjerin               SWEDEN             */
/* Date:    93-09-24                                      */
/* Version: 1.1                                           */
/*                                                       */
/*   Copyright 1993, Anders Bjerin - Amiga C Club (ACC)    */
/*                                                       */
/* Registered members may use this program freely in their */
/*     own commercial/noncommercial programs/articles.     */
/*                                                       */
/*********************************************************/

/* This example demonstrates how to allocate some memory which    */
/* has to be long word aligned. We will allocate a FileInfoBlock  */
/* structure in this example, but the procedure of allocating     */
/* long word aligned memory is the same for all types of objects. */



/* Include the normal dos header file: */
#include <libraries/dos.h>

/* Include memory definitions: (MEMF_ANY...) */
#include <exec/memory.h>

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

```
/* Set name and version number: */
UBYTE *version = "$VER: AmigaDOS/AmigaDOS/Example2 1.1";



/* Declared our own function(s): */
int main( int argc, char *argv[] );



/* The main function: */

int main( int argc, char *argv[] )
{
  /* A pointer to our memory which we will allocate: */
  struct FileInfoBlock *my_fib_ptr;



  /* Allocate some memory. The memory will be long word aligned   */
  /* which means that the data will start (and end) on a complete */
  /* 32-bit address (even word address, on a 4 byte boundary).    */
  my_fib_ptr = AllocMem( sizeof( struct FileInfoBlock ),
    MEMF_ANY | MEMF_CLEAR );


  /* Have we successfully allocated the memory? */
  if( my_fib_ptr == NULL )
  {
    /* Not enough memory! Inform the user and quit: */
    printf( "Could not allocate enough memory!\n" );

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



  /* You can now use the memory... */
  printf( "We have successfully allocated a FileInfoBlock structure!\n" );



  /* Deallocate the memory when we do not need it any more: */
  FreeMem( my_fib_ptr, sizeof( struct FileInfoBlock ) );

  /* Remember that you may not use the memory any */
  /* more after you have deallocated it!          */
  printf( "The memory has been deallocated!\n" );

  /* The End (0 = success): */
  exit( 0 );
}
```