

```

/*****
/*          C D R O M . C          */
**/
/* Task          : Demo Program for MSCDEX programming */
/*              : Uses CDUTIL          */
**/
/* Author        : Michael Tischer / Bruno Jennrich   */
/* Developed on   : 04/08/94                          */
/* Last update    : 04/18/94                          */
**/
/* COMPILER OPT. : 1 Byte Struct Member Alignment     */
/* COMPILER       : Borland C++ 3.1, Microsoft Visual C++ 1.5 */
*****/
#include <dos.h>
#include <stdio.h>
#include <ctype.h>
#include <conio.h>
#include <stdlib.h>
#include <process.h>

#include "args.h"
#include "cddev.h"
#include "cdutil.h"

/*- Remove the following lines within a project: -----*/
#include "cdutil.c"
#include "cddev.c"
#include "args.c"

/*****
/*--  M A I N  P R O G R A M  --*/
*****/
VOID main( INT argc, PCHAR argv[] )
{
    MSCDEX_DevStat  DS;
    MSCDEX_MedChng  MC;
    MSCDEX_DiskInfo DI;
    MSCDEX_TnoInfo  TI;
    MSCDEX_UPCode   UPC;

    INT i, j;
    INT iVersion;
    INT iDrive = 0;
    INT iNumTrack = 5;
    LONG lSector;
    LPCHAR lpFileName;

    if( argc == 1 )
    {
        printf("Parameter:\n");
        printf("A-Z:      Drive letter\s");
        printf(" -DEVSTAT  shows device status\n");
        printf(" -VTOC     shows Volume Table Of Contents\n");
        printf(" -UPC      shows Universal Product Code\n");
        printf(" -CONTENTS shows Tracks\n");
        printf(" -SHOW:X   shows sector number X");
        printf(" [with -RAW as Raw-Data]\n");
        printf(" -PLAY:X   plays audio track number X");
        printf(" [with -WAIT status info is displayed]\n");
        printf(" -ENTRY:name shows directory entry of");
        printf(" specified file\s");
        exit(0);
    }

    iDrive = -1;
    for( i = 1; i < argc; i++ )
        for( j = 0; j < 26; j++ )
            if( ( toupper( argv[ i ][ 0 ] ) == ( char ) ( j + 'A' ) ) &&
                ( argv[ i ][ 1 ] == ':' ) &&
                ( argv[ i ][ 2 ] == '\0' ) ) iDrive = j;

    if( iDrive < 0 )
    {
        printf("Invalid drive specification!\n");
        exit( 0 );
    }
}

```

```

if( !MSCDEX_Installed() )
{
    printf("Couldnot find MSCDEX!");
    exit( 0 );
}

iVersion = MSCDEX200_GetVersion();
printf("MSCDEX V%d.%02d detected\n", ( INT )HIBYTE( iVersion ),
      ( INT )LOBYTE( iVersion ) );

if( iVersion > 0x200 )
if( !MSCDEX200_CDROMDriveCheck( iDrive ) )
{
    printf("Specified drive is not a CD-ROM drive!");
    exit( 0 );
}

/*- See to it that any CD changes are recognized -----*/
cd_GetMediaChanged( iDrive, &MC );
cd_GetDevStat( iDrive, &DS );
if( GetArg( argc, argv, "-DEVSTAT", _none, NULL, 0 ) )
    cd_PrintDevStat( &DS );
else
if( DS.lDeviceStatus & DS_NO_DISC_IN_DRIVE )
{
    printf("No CD in drive!\n");
    exit( 0 );
}

/*- Display Volume Table Of Contents -----*/
if( GetArg( argc, argv, "-VTOC", _none, NULL, 0 ) )
{
    INT i, iResult;
    BYTE Sector[ 2048 ];
    i = 0;
    do
    {
        iResult = MSCDEX_ReadVTOC( iDrive, Sector, i );
        if( iResult ) cd_PrintSector( Sector, 2048, 16, 24 );
        i++;
    } while( iResult );
}

/*- Display Universal Product Code -----*/
if( GetArg( argc, argv, "-UPC", _none, NULL, 0 ) )
{
    cd_GetUPCode( iDrive, &UPC );
    cd_PrintUPCode( &UPC );
}

/*- Display directory (title) -----*/
if( GetArg( argc, argv, "-CONTENTS", _none, NULL, 0 ) )
    cd_PrintDiskTracks( iDrive );

/*- Display sector contents -----*/
lSector = 0L;
if( GetArg( argc, argv, "-SHOW:", _long, &lSector, 1 ) )
{
    char cSector[ 2352 ];
    INT iSize;
    iSize = 2048;
    if( GetArg( argc, argv, "-RAW", _none, NULL, 1 ) ) iSize = 2352;
    /*- Warning! Does not read audio CDs Copyright! -*/
    if( !cd_IsError( cd_ReadLong( iDrive, HSG, lSector, 1, cSector,
        ( iSize == 2048 ) ? COOKED : RAW ) ) )
        cd_PrintSector( cSector, iSize, 16, 24 );
    else printf("Cannot read sector! (audio CD?)\n");
}

/*- Display directory entry -----*/
if( GetArg( argc, argv, "-ENTRY:", _string, &lpFileName, 1 ) )
{
    DIR_ENTRY DirEntry;
    INT iIsHsg;

    if( MSCDEX200_GetDirectoryEntry( iDrive, STRUCT_COPY,
        lpFileName, &DirEntry, &iIsHsg ) )
        cd_PrintDirEntry( &DirEntry );
}

```

```

    else printf("Error: %d\n", _doserrno );
}

/*- Play title -----*/
iNumTrack = 0;
if( GetArg( argc, argv, "-PLAY:", _int, &iNumTrack, 1 ) )
{
    if( !( DS.lDeviceStatus & DS_AUDIO_VIDEO ) )
    {
        printf("Drive does not support audio playback!\n");
        exit( 0 );
    }
    if( DS.lDeviceStatus & DS_DOOR_OPEN ) cd_CloseTray( iDrive );
    cd_GetAudioDiskInfo( iDrive, &DI );
    if( ( iNumTrack < DI.bLowestTrack ) ||
        ( iNumTrack > DI.bHighestTrack ) )
    {
        printf("Invalid song number\n");
        exit(0);
    }
    cd_GetAudioTrackInfo( iDrive, iNumTrack, &TI );
    if( cd_IsDataTrack( &TI ) )
    {
        printf("Not an audio track!\n");
        exit(0);
    }
    cd_StopAudio( iDrive );
    if( !cd_IsError ( cd_PlayAudio( iDrive, REDBOOK, TI.lStartingPoint,
                                   cd_GetTrackLen( iDrive, iNumTrack ) ) ) )
    {
        if( GetArg( argc, argv, "-WAIT", _none, NULL, 0 ) )
        {
            while( cd_PrintActPlay( iDrive ) );
            printf("\n");
        }
        printf("Play - OK\n");
    }
}
}

```