

```

***** C D R O M . P A S *****
{-----*
Task      : Demo Program for MSCDEX programming
           Uses CDUTIL
{-----*
Author    : Michael Tischer / Bruno Jennrich
Developed on : 04/08/1994
Last update : 10/08/1994
*****}

{$A-}          { no word alignment of structures }
{$X+}          { Extended syntax, evaluation of Fct results optional }

Uses ARGs,CDDEV,CDUTIL,DOS;

Var
DS      : MSCDEX_DevStat;
MC      : MSCDEX_MedChng;
Di      : MSCDEX_DiskInfo;
TI      : MSCDEX_TnoInfo;
UPC     : MSCDEX_UPCode;
i,j,
iVersion,
iresult,
iIsHsg,
iIsSize : Integer;
arg      : String;
lpFileName : String;
lSector  : Longint;
Sector   : Array[0..2047] of Byte;
cSector  : Array[0..2351] of Char;
DirEntry : DIR_ENTRY ;

Const
iDrive      : Integer = 0;
iNumTrack   : Integer = 5;

{-----*
-- MAIN PROGRAM --
{-----*
Begin
  if ParamCount = 0 then
  Begin
    Writeln( 'Parameter:');
    Writeln( 'A-Z:           Drive letter');
    Writeln( ' -DEVSTAT       shows Device Status');
    Writeln( ' -VTOC          shows Volume Table Of Contents');
    Writeln( ' -UPC           shows Universal Product Code');
    Writeln( ' -CONTENTS      shows Tracks');
    Writeln( ' -SHOW:X        shows Sector Number X [with -RAW as Raw-Data]');
    Writeln( ' -PLAY:X        plays Audio Track Number X');
    Writeln( '                [Status infos displayed with -WAIT]');
    Write ( ' -ENTRY:name shows Directory Entry of');
    Writeln( ' the specified file');
    Halt(0);
  End;

  iDrive := -1;
  for i := 1 to ParamCount do
    for j := 0 to 25 do
      begin
        arg := ParamStr(i);
        if( ( upcase( arg[1] ) = char ( j + ord( 'A' ) ) ) and
            ( arg[ 2 ] = ':' ) and
            ( length(arg) = 2 ) ) then iDrive := j;
      end;

  if iDrive < 0 then
  Begin
    Writeln( 'Invalid drive specification! ');
    Halt( 0 );
  End;

  if not MSCDEX_Installed then
  Begin
    Writeln( 'Could not find MSCDEX!' );
    Halt( 0 );
  End;

  iVersion := MSCDEX200_GetVersion;
  Writeln('MSCDEX V', HI( iVersion ), '.', LO( iVersion ):2,' detected');

  if iVersion > $200 then
    if not MSCDEX200_CDROMDriveCheck( iDrive ) then

```

```

Begin
    Writeln( 'Specified drive is not a CD-ROM drive!');
    Halt( 0 );
End;

{- See to it that any CD change is recognized -----}
cd_GetMediaChanged( iDrive, MC );
cd_GetDevStat( iDrive, DS );
if GetArg( '-DEVSTAT', _none, NIL, 0 ) <> 0 then
    cd_PrintDevStat( DS )
else
    if ( DS.lDeviceStatus and DS_NO_DISC_IN_DRIVE ) <> 0 then
        Begin
            Writeln( 'No CD in drive!' );
            Halt( 0 );
        End;

{- Display Volume Table Of Contents -----}
if GetArg( '-VTOC', _none, NIL, 0 ) <> 0 then
Begin
    i := 0;
    repeat
        iResult := MSCDEX_ReadVTOC( iDrive, @Sector, i );
        if ( iResult ) <> 0 then
            cd_PrintSector( @Sector, 2048, 16, 24 );
        Inc(i);
    Until iResult = 0;
End;

{- Display Universal Product Code -----}
if GetArg( '-UPC', _none, NIL, 0 ) <> 0 then
Begin
    cd_GetUPCode( iDrive, UPC );
    cd_PrintUPCode( UPC );
End;

{- Display table of contents (title) -----}
if GetArg( '-CONTENTS', _none, NIL, 0 ) <> 0 then
    cd_PrintDiskTracks( iDrive );

{- Display sector contents -----}
lSector := 0;
if GetArg( '-SHOW:', _long, @lSector, 1 ) <> 0 then
    Begin
        iSize := 2048; i := COOKED;
        if GetArg( '-RAW', _none, NIL, 1 ) <> 0 then
            Begin
                iSize := 2352;
                i := RAW;
            End;
        { Warning! Does not read audio CDs Copyright!! }
        if not cd_IsError( cd_ReadLong( iDrive, HSG, lSector, 1, @cSector, i ) ) then
            cd_PrintSector( @cSector, iSize, 16, 24 )
        else
            Writeln( 'Cannot read sector! (Audio CD?)' );
    End;

{- Display directory entry -----}
if GetArg( '-ENTRY:', _string, @lpFileName, 1 ) <> 0 then
    Begin
        if MSCDEX200_GetDirectoryEntry( iDrive, STRUCT_COPY,
                                         lpFileName, @DirEntry, iIsHsg ) then
            cd_PrintDirEntry( DirEntry )
        else
            Writeln( 'Error: ', DOSERROR );
    End;

{- Play title -----}
iNumTrack := 0;
if GetArg( '-PLAY:', _int, @iNumTrack, 1 ) <> 0 then
    Begin
        if ( DS.lDeviceStatus and DS_AUDIO_VIDEO ) = 0 then
            Begin
                Writeln( 'Drive does not support audio playback!' );
                Halt( 0 );
            End;
        if ( DS.lDeviceStatus and DS_DOOR_OPEN ) <> 0 then
            cd_CloseTray( iDrive );
        cd_GetAudioDiskInfo( iDrive, DI );
        if ( ( iNumTrack < DI.bLowestTrack ) or
              ( iNumTrack > DI.bHighestTrack ) ) then
            Begin
                Writeln( 'invalid song number' );
                Halt(0);
            End;
        cd_GetAudioTrackInfo( iDrive, iNumTrack, TI );
    End;

```

```

if cd_IsDataTrack( TI ) then
  Begin
    Writeln('Not an audio track!');
    Halt(0);
  End;
cd_StopAudio( iDrive );
if not cd_IsError ( cd_PlayAudio( iDrive, REDBOOK, TI.lStartingPoint,
                                cd_GetTrackLen( iDrive, iNumTrack ) ) ) then
  Begin
    if GetArg( '-WAIT', _none, NIL, 0 ) <> 0 then
      Begin
        Repeat Until not ( cd_PrintActPlay( iDrive ) );
        writeln;
      End;
      Writeln( 'Play - OK' );
    End;
  End;
End;
End.

```