

Listing: RECLOCKC.C

```

/*****
/*
/*-----*
/* Task : Demonstrates the DOS record locking functions.*
/*-----*
/* Memory model : SMALL *
/*-----*
/* Author : Michael Tischer *
/* developed on : 02/10/1992 *
/* last Update : 04/07/1995 *
*****/

#include "netfilec.c" /* include network routine*/
#include <stdio.h>

/*== Constants =====*/

#define TFILENAME "reclockc.dat" /* file name for test file*/
#define NumOfRecs 10 /* number of records*/

/*== Type definitions =====*/

typedef char Test[ 160 ]; /* Data type for Test*/
typedef char TestString[ 161 ]; /* Data type for screen output*/

#ifdef __TURBOC__ /* Microsoft C?*/

#define clrscr() clearwindow( 1, 1, 80, 25 )

/*****
/* Gotoxy : Positions the Cursor */

```

```

/* Input          : coordinates of the Cursor          */
/* Output         : none                                */
/*****/

void gotoxy( int x, int y )

{
    regs.h.ah = 0x02;          /* function number for Interrupt call*/
    regs.h.bh = 0;             /* color*/
    regs.h.dh = y - 1;
    regs.h.dl = x - 1;
    int86( 0x10, &regs, &regs );          /* call Interrupt*/
}
#endif

/*****/
/* clearwindow    : Clear a portion of the screen.    */
/* Input          : s.u.                               */
/* Output         : none                                */
/*****/

void clearwindow( int x1, int y1, int x2, int y2 )
{
    regs.h.ah = 0x07;          /* function number for the interrupt call*/
    regs.h.al = 0x00;
    regs.h.bh = 0x07;
    regs.h.ch = y1 - 1;
    regs.h.cl = x1 - 1;
    regs.h.dh = y2 - 1;
    regs.h.dl = x2 - 1;
    int86( 0x10, &regs, &regs );          /* call interrupt*/
    gotoxy( x1, y1 );             /* position cursor*/
}

```

```

}

/*****
/* OpenNetFile : Open available network file. If one does not exist,*/
/*               create a new one and fill this new file with    */
/*               test data records.                               */
/* Input        : s.u.                                           */
/* Output       : file                                           */
*****/

int OpenNetFile( NFILE *DFile )                                /* Network file*/
{
    int i;                                                    /* loop counter*/
    Test TestDRec;                                           /* needed for creating the test file*/

/*-- Open file for input and output in Deny None mode -----*/

    NetReset( TFILENAME, FM_RW | SM_NO, sizeof( Test ), DFile );
    if ( NetError == NE_FileNotFound )                      /* file not found?*/
    {
/*-- Create file and fill with test data records -----*/

        NetRewrite( TFILENAME, FM_RW | SM_NO, sizeof( Test ), DFile );
        if ( NetError == NE_OK )                            /* no errors during creation?*/
        {
            if ( NetLock( DFile, 0L, (long) NumOfRecs ) )
            {
                NetSeek( DFile, 0L );                        /* pointer to beginning of the file*/
                for ( i = 0; i < NumOfRecs; i++ )
                {
                    memset( TestDRec, 'A' + i, 160 );
                    NetWrite( DFile, TestDRec );              /* write test file*/
                }
            }
        }
    }
}

```

```

    }
    return NetUnLock( DFile, 0L, (long) NumOfRecs );
}
else
    return FALSE;                                /* error when locking*/
}
else
    return FALSE;                                /* error while creating the file*/
}
else
    return ( NetError == 0 );                    /* no errors while opening?*/
}

/*****
/* NetEdits      : Demonstrates network functions          */
/* Input         : s.u                                     */
/* Output        : File                                     */
*****/

void NetEdits( NFILE *TestFile )                  /* network file*/
{
    unsigned long CurRecord;                       /* current Record number*/
    TestString    CurDRec;                         /* current data record*/
    int           Action;                          /* desired action*/
    int           Status;                          /* Record locked?*/
    char          TChar[ 10 ];
    char          SDummy[ 50 ];                    /* Network status*/
    int           LStatus[ NumOfRecs ];
    int           i;                               /* loop counter*/

    /--- Display menu -----*/

```

```

printf( "Available functions\n" );
printf( "  1: Position file pointer\n" );
printf( "  2: Lock record\n" );
printf( "  3: Read Record\n" );
printf( "  4: Edit data record\n" );
printf( "  5: Write record\n" );
printf( "  6: Unlock record\n" );
printf( "  7: Exit\n" );

/*-- Initialize data record -----*/

gotoxy( 58, 4 );
printf( "Status:" );
for ( i = 0; i < NumOfRecs; ++i )
{
    LStatus[i] = FALSE;
    gotoxy( 60, i+5 );
    printf( "%2d   Unlocked", i );
}

CurRecord = 0;                                     /* current data record*/
Status = FALSE;                                     /* Record not locked*/
memset( CurDRec, 32, 160 );                         /* create empty data record*/

do
{
/*-- Display information -----*/

gotoxy( 1, 16 );                                     /* display file pointer position*/
printf( "Current Record: %4li\n", CurRecord );
printf( "Status          : %s\n",
        LStatus[CurRecord] ? "Locked" : "Unlocked" );

```

```

NetErrorMsg( NetError, SDummy );
printf( "Network Status   : %4i = %s", NetError, SDummy );
gotoxy( 1, 21 );                                     /* display test record*/
printf( "Current Data Record:\n" );
CurDRec[ 160 ] = 0;
printf( "%s", CurDRec );

NetSeek( TestFile, CurRecord );                      /* Position file pointer*/
gotoxy( 1, 13 );
printf( "Select:                                     " );
gotoxy( 10, 13 );
scanf( "%i", &Action );
switch( Action )
{
    case 1 : gotoxy( 1, 13 );
              printf( "New data record number: " );
              do
              {
                  gotoxy( 25, 13 );
                  printf( "                               " );
                  gotoxy( 25, 13 );
                  scanf( "%li", &CurRecord );
              }
              while ( !( CurRecord >= 0  &&  CurRecord < NumOfRecs ) );
              break;

    case 2 : Status = NetLock( TestFile, CurRecord, 1L );
              if ( Status )
              {
                  LStatus[ CurRecord ] = TRUE;
                  gotoxy( 60, (int) CurRecord +5 );
                  printf( "%2d   Locked   ", CurRecord );
              }

```

```

        }
        break;

case 3 : NetRead( TestFile, CurDRec );          /* read data record*/
        break;

case 4 : gotoxy( 1, 13 );
        printf( "New character: " );
        scanf( "%s", TChar );
        memset( CurDRec, TChar[ 0 ], 160 );
        break;

case 5 : NetWrite( TestFile, CurDRec );          /* write data record*/
        break;

case 6 : Status = NetUnLock( TestFile, CurRecord, 1L );
        if ( Status )
        {
            LStatus[ CurRecord ] = FALSE;
            gotoxy( 60, (int) CurRecord+5 );
            printf( "%2d  UnLocked  ", CurRecord);
        }
        break;
    }
}

while ( Action != 7 );
}

/*****
/*          M A I N   P R O G R A M          */
*****/

```

```

void main( )

{
    NFILE DFile;                                /* Test file*/

    clrscr();
    printf( "Demonstration of DOS File Locking Funcitons" \
        " (C)1992 by Michael Tischer\n" );
    printf( "===== \
        "=====\\n\\n" );

    if ( ShareInst() )                          /* Share program installed?*/
    {
        if ( OpenNetFile( &DFile ) )           /* file open or created?*/
        {
            NetEdits( &DFile );                 /* Demonstration of network functions*/
            NetClose( &DFile );                 /* close file*/
            clrscr( );
        }
        else
            printf( "\\nError %i while opening network file", NetError );
    }
    else
        printf( "\\nPlease install SHARE before running this program" );
}

```