```
/*****************************************************************************

    FnTE.cp

*****************************************************************************/

/*
    Displaying text should be easier than it is.   This library
    provides a group of functions which allow the basics of displaying
    and editing of text in the form of a TE Record.

    Functions Include:

      FnTE_SetUpTEWindow        Creates TE Record to fill window
      FnTE_MaintainCursor       Call this and TEIdle() in event loop
      FnTE_UpdateWindow         Redraws window contents
      FnTE_DoActivate           Activates/Deactivates window
      FnTE_DoKeyDown            Handles keyboard inputs
      FnTE_DoEditMenu           Handles cut and paste of TE data
      FnTE_DoContent            Handles mousedown in window
      FnTE_DetSBarIncr          Refreshes scroll bar parameters
      FnTE_GrowWindow           Handles resizing of window
      FnTE_FrameRect            Puts frame around text in window
      FnTE_GetTEXT              Puts TEXT resource into TE Record

      // Semi-private
      FnTE_AdjustText
      FnTE_SetToCursorPos
      FnTE_CheckScrollBar
      FnTE_ScrollProc
*/

#define FnCUT           1
#define FnCOPY          2
#define FnPASTE         3
#define FnCLEAR         4
#define FnSELECT_ALL 5

#define DEL_KEY         8
#define DOWN_ARROW     31
#define UP_ARROW       30
#define LEFT_ARROW     28
#define RIGHT_ARROW    29
#define PAGE_DOWN      12
#define PAGE_UP        11
#define HOME            1
#define END             4

#define MAX_CHAR       32000

TEHandle        *tempTEHptr;
ControlHandle *tempVScrollptr;
Cursor          IBeamCursor;
Boolean         IBeamCursorDefined = FALSE;

// Prototypes

Boolean FnTE_SetUpTEWindow   ( WindowPtr       w,
                               Boolean         hasGrowIcon,
                               Boolean         wordWrap,
                               short           font,
                               short           textSize,
```

```
                                short           textInsetPixels,
                                // results //
                                TEHandle        *te,
                                ControlHandle *vScroll );
void    FnTE_MaintainCursor ( WindowPtr     w,
                                TEHandle        *te );
void    FnTE_UpdateWindow   ( WindowPtr     w,
                                TEHandle        *te,
                                Boolean         hasGrowIcon );
void    FnTE_DoActivate     ( WindowPtr     w,
                                TEHandle        *te,
                                ControlHandle *vScroll,
                                Boolean         hasGrowIcon,
                                Boolean         activate );
void    FnTE_DoKeyDown      ( char          theChar,
                                TEHandle        *te,
                                ControlHandle *vScroll,
                                char            *dirty );
void    FnTE_DoEditMenu     ( int           item,
                                TEHandle        *te,
                                ControlHandle *vScroll,
                                char            *dirty,
                                int             cutCommand,
                                int             copyCommand,
                                int             pasteCommand,
                                int             clearCommand,
                                int             selectAllCommand );
void    FnTE_DoContent      ( WindowPtr     w,
                                EventRecord   *e,
                                TEHandle        *te,
                                ControlHandle *vScroll );
void    FnTE_DetSBarIncr    ( TEHandle      *te,
                                ControlHandle *vScroll );
void    FnTE_GrowWindow     ( WindowPtr     w,
                                TEHandle        *te,
                                ControlHandle *vScroll,
                                short           textInsetPixels,
                                Point           clickLoc );
void    FnTE_FrameRect      ( WindowPtr     w,
                                TEHandle        *te,
                                short           inset );
void    FnTE_GetTEXT        ( short         rsrcID,
                                TEHandle        *te,
                                ControlHandle *vScroll );

void    FnTE_AdjustText     ( TEHandle *te, ControlHandle *vScroll );
void    FnTE_SetToCursorPos ( TEHandle *te, ControlHandle *vScroll );
void    FnTE_CheckScrollBar ( TEHandle *te, ControlHandle *vScroll );
pascal void FnTE_ScrollProc ( ControlHandle theControl,
                                short           theCode );

extern void FnErr_DisplayStr(   Str255 s1,
                                Str255 s2,
                                Str255 s3,
                                Str255 s4,
                                int     quitFlag );


/********** SetUpTEWindow */

Boolean FnTE_SetUpTEWindow(
    WindowPtr   w,
```

```
    Boolean      hasGrowIcon,
    Boolean      wordWrap,
    short        font,
    short        textSize,
    short        textInsetPixels,
    // results //
    TEHandle        *te,
    ControlHandle *vScroll )
/*
    FnSetUpWindow takes information you supply about a window
    and it's default text parameters then returns a TEHandle to
    a text pane that fills the window and a ControlHandle to
    a scroll bar that is located at the right of the window.
    Returns FALSE if error detected during routine.
*/

    Rect          destRect, viewRect;
    Rect          vSBarRect;
    GrafPtr      oldPort;
    int           SBarWidth = 15;

    // define basic te stuff
    GetPort( &oldPort );
    SetPort( w );
    TextFont( font );
    TextSize( textSize );

    // set scroll bar rect
    vSBarRect = (*w).portRect;
    vSBarRect.left = vSBarRect.right - SBarWidth;
    vSBarRect.right += 1;
    vSBarRect.top -= 1;
    if( hasGrowIcon )
        vSBarRect.bottom -= 14;
    else
        vSBarRect.bottom += 1;

    // create new control
    *vScroll = NewControl( w, &vSBarRect, "\p", 1, 0, 0, 0,
        scrollBarProc, 0L);
    if( *vScroll == NULL )
        return FALSE;

    // define te view and dest rects
    viewRect = w->portRect;
    viewRect.right -= SBarWidth;
    if( hasGrowIcon )
        viewRect.bottom -= SBarWidth;
    InsetRect(&viewRect, textInsetPixels, textInsetPixels);
    destRect = viewRect;

    // create new te record
    *te = TENew( &destRect, &viewRect );
    if( *te == NULL )
        return FALSE;
    if( wordWrap )
        (***te).crOnly = 1;
    else
        (***te).crOnly = -1;

    // set autoview flag for automatic scrolling
    TEAutoView( TRUE, *te );
```

```
    SetPort( oldPort );
    return TRUE;


/********** MaintainCursor */

void FnTE_MaintainCursor( WindowPtr w, TEHandle *te )
/*
    Maintains cursor icon in TE window.   Call this function each time
    through event loop (also call Macintosh function TEIdle to blink
    cursor).
*/

    Point          pt;
    WindowPeek   wPtr;
    GrafPtr       savePort;
    CursHandle   hCurs;

    /*
        Variables hIBeamCursor and IBeamCursorDefined must be defined
        at top of this FnTELibrary.c file.
    */

    if( !IBeamCursorDefined )

        hCurs = GetCursor(1);   /* IBeam Cusor ID */
        IBeamCursor = **hCurs;
        IBeamCursorDefined = TRUE;


    wPtr = (WindowPeek)FrontWindow();
    if( (WindowPtr)wPtr == w )

        if( wPtr->windowKind >= 0 )

            GetPort( &savePort );
            SetPort( (GrafPtr)wPtr );
            GetMouse( &pt );
            if( PtInRect( pt, &(***te).viewRect ) )

                SetCursor( &IBeamCursor );

            else

                SetCursor( &qd.arrow );

            SetPort( savePort );




/********** UpdateWindow */

void FnTE_UpdateWindow( WindowPtr w, TEHandle *te, Boolean hasGrowIcon )
/*
    Updates a text window.
*/

    GrafPtr savePort;
```

```
    GetPort(&savePort);
    SetPort(w);

    BeginUpdate(w);
    EraseRect(&w->portRect);
    DrawControls(w);
    if( hasGrowIcon )
        DrawGrowIcon(w);
    TEUpdate(&w->portRect, *te);
    EndUpdate(w);

    SetPort(savePort);



/********** DoActivate */
void FnTE_DoActivate( WindowPtr       w,
                      TEHandle        *te,
                      ControlHandle *vScroll,
                      Boolean         hasGrowIcon,
                      Boolean         activate )
/*
    Handles activateEvt for TE window.
*/

    if( activate )

        TEActivate( *te );
        ShowControl( *vScroll );
        if( hasGrowIcon ) DrawGrowIcon( w );
        TEFromScrap();

    else

        TEDeactivate( *te );
        HideControl( *vScroll );
        if( hasGrowIcon ) DrawGrowIcon( w );
        ZeroScrap();
        TEToScrap();



/********** DoKeyDown */

void FnTE_DoKeyDown( char              theChar,
                     TEHandle        *te,
                     ControlHandle *vScroll,
                     char              *dirty )
/*
    Handles key press in TE region.   Uses functions FnTE_SetToCursorPos
    and FnTE_AdjustText to postion text and scroll bar.
*/

    int count = 256;              // same as used in FnIO_Library
    tempTEHptr = te;              // used in Scroll_Proc
    tempVScrollptr = vScroll; // used in Scroll_Proc

    if( (((***te).teLength + count) >= MAX_CHAR) &&
        (theChar != DEL_KEY) )
```

```
            FnErr_DisplayStr(
                "\pFile size limits have been exceeded.   ",
                "\pPlease save your work and then try to ",
                "\preduce file size.",
                "\p",
                FALSE );   // don't quit

        else

            switch( theChar )

                case DOWN_ARROW:
                case UP_ARROW:
                case LEFT_ARROW:
                case RIGHT_ARROW:
                    TEKey( theChar, *te );
                    FnTE_SetToCursorPos( te, vScroll );
                    FnTE_AdjustText( te, vScroll );
                    break;
                case PAGE_DOWN:
                    FnTE_ScrollProc( *vScroll, (short)inPageDown );
                    break;
                case PAGE_UP:
                    FnTE_ScrollProc( *vScroll, (short)inPageUp );
                    break;
                case HOME:
                    SetCtlValue( *vScroll, 0 );
                    FnTE_AdjustText( te, vScroll );
                    break;
                case END:
                    SetCtlValue( *vScroll, GetCtlMax(*vScroll) );
                    FnTE_AdjustText( te, vScroll );
                    break;
                default:

                        TEKey( theChar, *te );
                        FnTE_SetToCursorPos( te, vScroll );
                        FnTE_AdjustText( te, vScroll );
                        *dirty = 1;




/********** DoEditMenu */

void FnTE_DoEditMenu( int                theItem,
                      TEHandle         *te,
                      ControlHandle *vScroll,
                      char              *dirty,
                      int               cutCommand,
                      int               copyCommand,
                      int               pasteCommand,
                      int               clearCommand,
                      int               selectAll )
/*
    Performs cut and paste function of standard Edit menu items.
*/

    if       ( theItem == cutCommand )     theItem = FnCUT;
    else if( theItem == copyCommand )   theItem = FnCOPY;
```

```
    else if( theItem == pasteCommand ) theItem = FnPASTE;
    else if( theItem == clearCommand ) theItem = FnCLEAR;
    else if( theItem == selectAll )    theItem = FnSELECT_ALL;

    switch( theItem )

        case FnCUT:
            TECut( *te );
            FnTE_DetSBarIncr( te, vScroll );
            FnTE_SetToCursorPos( te, vScroll );
            FnTE_AdjustText( te, vScroll );
            *dirty = 1;
            break;
        case FnCOPY:
            TECopy( *te );
            break;
        case FnPASTE:
            if(TEGetScrapLen() + (***te).teLength < MAX_CHAR)

                TEPaste( *te );
                FnTE_DetSBarIncr( te, vScroll );
                FnTE_SetToCursorPos( te, vScroll );
                FnTE_AdjustText( te, vScroll );
                *dirty = 1;

            else

                FnErr_DisplayStr(
                    "\pPaste size too large.","\p","\p","\p", FALSE);

            break;
        case FnCLEAR:
            TEDelete( *te );
            FnTE_DetSBarIncr( te, vScroll );
            FnTE_SetToCursorPos( te, vScroll );
            FnTE_AdjustText( te, vScroll );
            *dirty = 1;
            break;
        case FnSELECT_ALL:
            TESetSelect( 0, 32767, *te );
            FnTE_SetToCursorPos( te, vScroll );
            FnTE_AdjustText( te, vScroll );
            break;



/********** DoContent */

void FnTE_DoContent( WindowPtr       w,
                     EventRecord    *e,
                     TEHandle        *te,
                     ControlHandle *vScroll )
/*
    Use this procedure when user clicks in content region of TE window.
    Uses the TnTE_ScrollProc below to take care of scroll bar function.
    Also uses FnTE_AdjustText function.
*/

    int                 cntlCode;
    ControlHandle       cntl;
    int                 pageSize;
```

```
    GrafPtr                 savePort;

    GetPort(&savePort);
    SetPort(w);

    GlobalToLocal(&e->where);
    if( (cntlCode = FindControl(e->where, w, &cntl)) == 0 )

        if( PtInRect( e->where, &(***te).viewRect) )

            TEClick( e->where, (e->modifiers & shiftKey) != 0, *te );
            FnTE_CheckScrollBar( te, vScroll );


    else if( cntlCode == inThumb )

        TrackControl( cntl, e->where, 0L );
        FnTE_AdjustText( te, vScroll );

    else

        tempTEHptr = te;
        tempVScrollptr = vScroll;
        TrackControl( cntl, e->where, FnTE_ScrollProc );


    SetPort(savePort);



/********** DetSBarIncr */

void FnTE_DetSBarIncr( TEHandle *te, ControlHandle *vScroll )
/*
    Determines number of scroll bar increments based on amount of
    text pointed to in TEHandle te.
*/

    register int nIncr, nLinesDisplayed, len;

    nLinesDisplayed = ((***te).viewRect.bottom-(***te).viewRect.top) /
        (***te).lineHeight;
    nIncr = (***te).nLines - nLinesDisplayed;
    len = (***te).teLength;
    if( len <= 1 ) len = 1;
    if( (*((***te).hText))[len-1] == '\r' )
      nIncr++;

    if( nIncr < 0 )
        nIncr = 0;
    if( nIncr != GetCtlMax( *vScroll ) )

SetCtlMax( *vScroll, nIncr );



/********** GrowWindow */

void FnTE_GrowWindow( WindowPtr       w,
                      TEHandle        *te,
                      ControlHandle *vScroll,
                      short           textInsetPixels,
```

```
                        Point           clickLoc )
/*
    Use this function to resize a TE window.
*/

    GrafPtr savePort;
    long    theResult;
    Rect    oldHorizBar;
    Rect    r;
    int     SBarWidth = 15;
    float   percent;
    float   currentValue;
    float   currentMax;

    GetPort( &savePort );
    SetPort( w );
    // although not used, calculate size of bottom scroll bar
    oldHorizBar = w->portRect;
    oldHorizBar.top = oldHorizBar.bottom - (SBarWidth + 1);
    currentValue = GetCtlValue( *vScroll );
    currentMax = GetCtlMax( *vScroll );
    if( currentMax > 0 )
        percent = currentValue / currentMax;
    else
        percent = 0.0;
    SetRect( &r, 100, 100, qd.screenBits.bounds.right,
        qd.screenBits.bounds.bottom );
    theResult = GrowWindow( w, clickLoc, &r );
    if (theResult == 0)
      return;
    SizeWindow( w, LoWord(theResult), HiWord(theResult), FALSE );
    InvalRect( &w->portRect );
    EraseRect( &w->portRect );

    (***te).viewRect = w->portRect;
    (***te).viewRect.right -= SBarWidth;
    (***te).viewRect.bottom -= SBarWidth;
    InsetRect( &(***te).viewRect, textInsetPixels, textInsetPixels );
    (***te).destRect = (***te).viewRect;
    TECalText( *te );
    MoveControl( *vScroll,
                    w->portRect.right - SBarWidth,
                    w->portRect.top - 1 );
    SizeControl( *vScroll,
                    SBarWidth + 1,
                    w->portRect.bottom - w->portRect.top - (SBarWidth-2) );
    FnTE_DetSBarIncr( te, vScroll );
    percent = percent * (float)GetCtlMax( *vScroll );
    SetCtlValue( *vScroll, int( percent ) );
    FnTE_AdjustText( te, vScroll );
    DrawGrowIcon( w );
    TEUpdate( &w->portRect, *te );
    ValidRect( &w->portRect );

    SetPort( savePort );



/********** FrameRect */

void FnTE_FrameRect( WindowPtr w, TEHandle *te, short inset )
```

```
    Rect        itemRect;
    PenState thePnState;
    GrafPtr    oldPort;

    GetPort( &oldPort );
    SetPort( w );
    itemRect = (***te).viewRect;
    InsetRect( &itemRect, -inset, -inset );
    itemRect.right += 1;
    GetPenState( &thePnState );
    PenNormal();
    FrameRect( &itemRect );
    SetPenState( &thePnState );
    SetPort( oldPort );



/********** GetTEXT */

void FnTE_GetTEXT( short rsrcID, TEHandle *te, ControlHandle *vScroll )
/*
    Reads in TEXT resource into TE record and resets scroll bar.    Need
    to call FnTE_UpdateWindow to display text.
*/

    (**te)->hText = GetResource( 'TEXT', rsrcID );
    TECalText( *te );
    (***te).destRect.top = (***te).viewRect.top;
    SetCtlValue( *vScroll, GetCtlMin( *vScroll ) );
    FnTE_DetSBarIncr( te, vScroll );



/********** AdjustText */

void FnTE_AdjustText( TEHandle *te, ControlHandle *vScroll )
/*
    Takes current value of the scroll bar position and adjusts text
    for specified TextEdit pane.    Call this function after user
    clicks in scroll bar.
*/

    int    oldScroll, newScroll, delta;

    oldScroll = (***te).viewRect.top - (***te).destRect.top;
    newScroll = GetCtlValue( *vScroll ) * (***te).lineHeight;
    delta = oldScroll - newScroll;
    if( delta != 0 )
        TEScroll( 0, delta, *te );



/********** SetToCursorPos */

void FnTE_SetToCursorPos( TEHandle *te, ControlHandle *vScroll )
/*
    Use this function to adjust scroll bar position such that the
    currently selected text and/or insertion cursor position is
    displayed in middle of te window.    You must make a call to a
    function like FnTE_AdjustText after calling this function to redraw
    text based on new scroll bar position.
*/
```

```
    register int topLine,bottomLine,theLine,nIncr,nLinesDisplayed,len;

    // following could be replaced by call to FnDetSBarIncr
    nLinesDisplayed = ((***te).viewRect.bottom-(***te).viewRect.top) /
        (***te).lineHeight;
    nIncr = (***te).nLines - nLinesDisplayed;

    len = (***te).teLength;
    if( len <= 1 ) len = 1;
    if( (*((***te).hText))[len-1] == '\r' )
      nIncr++;

    if( nIncr < 0 )
        nIncr = 0;
    SetCtlMax( *vScroll, nIncr );

    topLine = GetCtlValue( *vScroll );
    bottomLine = topLine + nLinesDisplayed;
    if( (***te).selStart < (***te).lineStarts[topLine] ||
        (***te).selStart >= (***te).lineStarts[bottomLine] )

        for( theLine = 0;
            (***te).selStart >= (***te).lineStarts[theLine];
            theLine++ )
            /* count */ ;
        SetCtlValue( *vScroll, theLine - nLinesDisplayed / 2 );



/********** CheckScrollBar */

void FnTE_CheckScrollBar( TEHandle *te, ControlHandle *vScroll )
/*
    Use this function to check (and adjust) scroll bar such that the
    current top of the TE window matches the scroll bar increment.   This
    should be called when a mouseDown occurs in a TE window which has
    auto scrolling enabled.
*/

    register int incr;

    incr = ((***te).viewRect.top-(***te).destRect.top) /
        (***te).lineHeight;
    if( incr < 0 )
        incr = 0;
    if( incr > GetCtlMax( *vScroll ) )
        incr = GetCtlMax( *vScroll );
    if( incr != GetCtlValue( *vScroll ) )
        SetCtlValue( *vScroll, incr );



/********** PASCAL ScrollProc */

pascal void FnTE_ScrollProc( ControlHandle theControl, short theCode )
/*
    Scroll procedure used for TE scroll bar.   Uses FnTE_AdjustText
    function to update text.
*/
```

```
    int     pageSize;
    int     scrollAmt;
    int     oldCtl;

    if( theCode == 0 )
        return;

    pageSize =
        ((***tempTEHptr).viewRect.bottom -
        (***tempTEHptr).viewRect.top) /
        (***tempTEHptr).lineHeight - 1;

    switch( theCode )

        case inUpButton:
            scrollAmt = -1;
            break;
        case inDownButton:
            scrollAmt = 1;
            break;
        case inPageUp:
            scrollAmt = -pageSize;
            break;
        case inPageDown:
            scrollAmt = pageSize;
            break;


    oldCtl = GetCtlValue( theControl );
    SetCtlValue( theControl, oldCtl + scrollAmt );

    FnTE_AdjustText( tempTEHptr, tempVScrollptr );


// End of File
```