```
--File:  WEMain.mesa
--Edited by:
--          Barbara  July 18, 1978  9:49 AM
--          Sandman  May 4, 1978  9:32 AM

DIRECTORY
  AltoDefs: FROM "altodefs" USING [BytesPerPage],
  DebugMiscDefs: FROM "debugmiscdefs" USING [DebugAbort, DebugProceed],
  InlineDefs: FROM "inlinedefs" USING [BITOR, BITXOR],
  IODefs: FROM "iodefs" USING [BS, CR, DEL, ESC],
  KeyDefs: FROM "keydefs" USING [Keys, Mouse],
  OsStaticDefs: FROM "osstaticdefs" USING [OsStatics],
  ProcessDefs: FROM "processdefs" USING [Aborted, Yield],
  RectangleDefs: FROM "rectangledefs" USING [
    ClearBoxInRectangle, CursorToRecCoords, GrayArray, GrayPtr, leftmargin,
    Rptr, xCoord, yCoord],
  SegmentDefs: FROM "segmentdefs" USING [FileNameError, InvalidFP],
  StreamDefs: FROM "streamdefs" USING [
    DisplayHandle, GetCurrentKey, GetIndex, KeyboardHandle, OpenKeyStream,
    StreamError, StreamIndex],
  WindExDefs: FROM "windexdefs" USING [
    AMouseButton, CursorArray, cursormap, CursorType, JumpStrip, KeySet,
    LMult, maxscratch, NullIndex, ReadEditChar, Shorten, slop, WEDataHandle,
    xcursorloc, ycursorloc],
  WindowDefs: FROM "windowdefs" USING [
    AlterWindowType, BlinkCursor, FindDisplayWindow, GetCurrentDisplayWindow,
    GetSelection, Selection, SetCurrentDisplayWindow, SetFileForWindow,
    WindowHandle];

DEFINITIONS FROM WindExDefs;

WEMain:  PROGRAM [WEState: WEDataHandle]
  IMPORTS ProcessDefs, SegmentDefs, StreamDefs, RectangleDefs, WindowDefs,
    DebugMiscDefs, WindExDefs
  EXPORTS WindExDefs
  SHARES WindExDefs, StreamDefs =
  BEGIN

  -- common types

  WindowHandle: TYPE = WindowDefs.WindowHandle;
  DisplayHandle: TYPE = StreamDefs.DisplayHandle;
  KeyboardHandle: TYPE = StreamDefs.KeyboardHandle;
  StreamIndex: TYPE = StreamDefs.StreamIndex;
  Selection: TYPE = WindowDefs.Selection;
  Rptr: TYPE = RectangleDefs.Rptr;
  xCoord: TYPE = RectangleDefs.xCoord;
  yCoord: TYPE = RectangleDefs.yCoord;

  -- Window Executive Main Control Routine

  WindowExecutive: PUBLIC PROCEDURE =
    BEGIN OPEN DebugMiscDefs;
    timer: POINTER TO INTEGER ← LOOPHOLE[430B];
    blinktime: INTEGER ← timer↑+8;
    DO -- forever
      DoWork[ !
        DebugProceed => RESUME;
        ProcessDefs.Aborted, DebugAbort, StreamDefs.StreamError => CONTINUE];
      IF blinktime-timer↑ ~IN[0..13] THEN
        BEGIN [] ← WindowDefs.BlinkCursor[]; blinktime ← timer↑+13 END;
      ProcessDefs.Yield[];
      ENDLOOP;
    END;

  DoWork: PROCEDURE =
    BEGIN OPEN WEState;
    -- Declare Locals
    x, y,i: INTEGER;
    k: KeySet;
    inJumpStrip: BOOLEAN ← FALSE;
    mousewindow: WindowHandle;
    buttons: AMouseButton;
    cw: WindowHandle;
    currentks: KeyboardHandle;
```

```
    currentks ← StreamDefs.GetCurrentKey[];
    cw ← WindowDefs.GetCurrentDisplayWindow[];
    -- check if need to service KeyStream
    IF cw.ks # defaultks AND NOT cw.ks.endof[cw.ks] THEN
      FOR i IN [0..maxscratch) DO ENABLE StreamDefs.StreamError => EXIT;
        IF scratchfiles[i] = cw.file THEN
          BEGIN
          ReadEditChar[cw.ks.get[cw.ks], cw];
          EXIT;
          END;
        ENDLOOP;
    -- check if some part of cursor is in jump bar
    [x, y] ← CursorToRectangleCoords[cw.rectangle, xcursorloc↑, ycursorloc↑];
    inJumpStrip ← x+slop > 0 AND x <= JumpStrip AND y+slop > 0 AND
      y-slop <= cw.rectangle.ch;
    SetJumpStripe[cw, inJumpStrip];
    -- check mouse buttons
    buttons ← GetMouseButton[];
    -- look at the mouse and flip from one to the other
    cw ← WindowDefs.GetCurrentDisplayWindow[];
    [mousewindow, x, y] ←
      WindowDefs.FindDisplayWindow[xcursorloc↑+cxa, ycursorloc↑+cya];
    IF ~inJumpStrip AND cw # mousewindow AND mousewindow # NIL AND
      buttons # None THEN
      BEGIN
      WindowDefs.SetCurrentDisplayWindow[mousewindow !
        SegmentDefs.InvalidFP =>
          BEGIN
          WindowDefs.SetFileForWindow[mousewindow, mousewindow.name !
            SegmentDefs.FileNameError =>
              BEGIN
              WindowDefs.AlterWindowType[mousewindow, mousewindow.type, NIL];
              CONTINUE;
              END];
          RETRY;
          END];
      StreamDefs.OpenKeyStream[mousewindow.ks];
      END
    ELSE IF buttons = None THEN
      BEGIN OPEN OsStaticDefs;
      IF useKeyset AND OsStatics.AltoVersion.engineeringnumber # 4 --DO
        AND (k ← GetKeySet[]) # 0 THEN
          SELECT k FROM
            1B => StuffSel[cw];
            2B => PutChar[IODefs.CR];
            3B => BEGIN PutChar[IODefs.CR]; StuffSel[cw]; END;
            4B => PutChar[IODefs.ESC];
            10B => PutChar[IODefs.DEL];
            20B => PutChar[IODefs.BS];
            ENDCASE
      ELSE BEGIN
        IF OsStatics.AltoVersion.engineeringnumber = 4 THEN useKeyset ← FALSE;
        IF FL4Down[] THEN
          BEGIN StuffSel[cw]; WHILE FL4Down[] DO NULL ENDLOOP; END;
        END;
      END
    ELSE
      BEGIN
      THROUGH [0..700) DO NULL ENDLOOP;  -- Debounce Mouse
      ButtonProcArray[GetMouseButton[]][cw, xcursorloc↑+cxa,ycursorloc↑+cya];
      END;
    END;

  SetJumpStripe: PUBLIC PROCEDURE[w: WindowHandle, flag: BOOLEAN] =
    BEGIN OPEN RectangleDefs, WEState;
   -- Declare Locals
    r: Rptr = w.rectangle;
    y1, y2: yCoord;
    bytepos, wendpos, eof: LONG INTEGER;
    longZero: LONG INTEGER ← 0;
    longOne: LONG INTEGER ← 1;
    barheight: CARDINAL = r.ch-defaultlineheight-2;
    barwidth: INTEGER = leftmargin-1;
    current, wendindex: StreamIndex;
    gray: GrayArray ← [125252B, 52525B, 125252B, 52525B];
    grayarray: GrayPtr = @gray;
```

```
    zeros: GrayArray ← [0,0,0,0];
    zeroarray: GrayPtr = @zeros;
   -- check if visible
    IF (w.rectangle.visible = FALSE) OR (w.file = NIL) THEN
      RETURN;
   -- now set or reset state
    IF flag THEN
      BEGIN
      IF currentcursor = botharrow THEN RETURN;
      SetCursor[botharrow];
      ButtonProcArray ← ScrollProcArray;
      ClearBoxInRectangle[r, 1, barwidth, defaultlineheight+1, barheight, grayarray];
      current ← IF w.tempindex = NullIndex THEN w.fileindex ELSE w.tempindex;
      -- compute position in file and paint(reset) position
      bytepos ← IF current.byte=177777B THEN longZero
        ELSE LMult[current.page, AltoDefs.BytesPerPage] + current.byte;
      wendindex ← StreamDefs.GetIndex[w.file];
      wendpos ← LMult[wendindex.page, AltoDefs.BytesPerPage] + wendindex.byte;
      eof ← IF w.eofindex.byte=177777B THEN longOne
        ELSE LMult[w.eofindex.page, AltoDefs.BytesPerPage] + w.eofindex.byte;
      IF wendpos > eof THEN eof ← wendpos;
      y1 ← Shorten[(bytepos*barheight)/eof];
      y2 ← Shorten[(wendpos*barheight)/eof];
      y2 ← MAX[y2-y1, 2];
      ClearBoxInRectangle[
        r, 3, barwidth-5, y1+defaultlineheight+1, y2, zeroarray];
      END
    ELSE
      BEGIN
      SetCursor[textpointer];
      ButtonProcArray ← TextProcArray;
      ClearBoxInRectangle[
        r, 1, barwidth, defaultlineheight+1, barheight, zeroarray]
      END;
    END;

  GetMouseButton: PUBLIC PROCEDURE RETURNS[AMouseButton]=
    BEGIN
    RETURN[KeyDefs.Mouse.buttons];
    END;

  GetKeySet: PUBLIC PROCEDURE RETURNS [KeySet]=
    BEGIN OPEN InlineDefs;
    n, keyvalues: KeySet ← 0;
    DO
      n ← BITXOR[KeyDefs.Mouse.keyset, 37B];
      IF n = 0 THEN
        BEGIN
        THROUGH [0..200) DO NULL ENDLOOP;
        n ← BITXOR[KeyDefs.Mouse.keyset, 37B];
        END;
      IF n = 0 THEN RETURN[keyvalues] ELSE keyvalues ← BITOR[keyvalues, n];
      ENDLOOP;
    END;

  FL4Down: PROCEDURE RETURNS [BOOLEAN] =
    BEGIN OPEN KeyDefs;
    RETURN[KeyDefs.Keys.FL4 = down];
    END;

  PutChar: PROCEDURE [c: CHARACTER]=
    BEGIN OPEN WEState;
    defaultks.putback[defaultks, c];
    END;

  StuffSel: PROCEDURE [w: WindowHandle]=
    BEGIN OPEN WEState;
    s: STRING ← WindowDefs.GetSelection[w];
    i: CARDINAL;
    FOR i DECREASING IN [0..s.length) DO defaultks.putback[defaultks, s[i]]; ENDLOOP;
    END;

  CursorToRectangleCoords: PUBLIC PROCEDURE [rectangle: Rptr, x: xCoord, y: yCoord]
    RETURNS[xCoord, yCoord] =
    BEGIN OPEN WEState;
    -- refinements for sensitive points of each cursor
```

```
    x ← x + cxa;
    y ← y + cya;
    -- convert cursor coordinates to window coordinates
    [x, y] ← RectangleDefs.CursorToRecCoords[rectangle, x, y];
    RETURN[x, y];
    END;

NullProc:  PUBLIC PROCEDURE[w:  WindowHandle, x:  xCoord, y:  yCoord] =
  BEGIN
  RETURN;
  END;

  Cursors: ARRAY CursorType OF CursorArray  = [
    -- textpointer
    [100000B, 140000B, 160000B, 170000B, 174000B, 176000B, 177000B, 170000B, 154000B,
      114000B, 6000B, 6000B, 3000B, 3000B, 1400B,1400B],

    -- arrow
    [40B, 60B, 70B, 74B, 177776B, 177777B, 177776B, 74B, 70B, 60B, 40B, 0, 0, 0, 0, 0],

    -- bullseye
    [3700B, 7740B, 14060B, 30030B, 60014B, 140006B, 141606B, 141606B, 141606B,
      140006B, 60014B, 30030B, 14060B, 7740B, 3700B, 0B],

    -- leftbutton
    [177740B, 100040B, 135240B, 135240B, 135240B, 135240B, 135240B, 100040B, 100040B,
      100040B, 100040B, 100040B, 100040B, 100040B, 100040B, 177740B],

    -- uparrow
    [600B, 1700B, 7760B, 37776B, 177777B, 7760B, 7760B, 7760B, 7760B, 7760B, 7760B,7760B,
      7760B, 7760B,7760B, 7760B],

    -- downarrow
    [7760B, 7760B, 7760B,7760B, 7760B, 7760B,7760B, 7760B, 7760B,7760B, 7760B,177777B,
      37776B, 7760B, 1700B, 600B],

    -- botharrow
    [600B, 1700B, 7760B, 37776B, 177777B, 7760B, 7760B, 7760B, 7760B, 7760B, 7760B,
      177777B, 37776B, 7760B, 1700B, 600B],

    -- hourglass
    [177777B, 100001B, 40002B, 34034B, 17170B, 7660B, 3740B, 1700B, 1100B, 2440B, 4220B,
      10610B, 21704B, 47762B, 177777B, 177777B],

    -- norm
    [37774B, 37774B, 34034B, 34034B, 34034B, 34034B, 34034B, 34034B, 34034B, 34034B,
      34034B, 34034B, 34034B, 34034B, 37774B, 37774B],

    -- menu
    [1000B, 3001B, 7003B, 36007B, 177776B, 177776B, 36007B, 7003B,
      3001B, 1000B, 0B, 0B, 0B, 0B, 0B, 0B]];

  SetCursor: PUBLIC PROCEDURE [type: CursorType] =
    BEGIN OPEN WEState;
    XAdjust: PACKED ARRAY CursorType OF [0..16) = [0,15,7,0,8,8,0,0,8,0];
    YAdjust: PACKED ARRAY CursorType OF [0..16) = [0,8,7,0,0,15,0,0,8,5];
    Cursor: POINTER TO CursorArray = cursormap;
    currentcursor ← type;
    cya ← YAdjust[type];
    cxa ← XAdjust[type];
    Cursor↑ ← Cursors[type];
    END;

END. of WEControl
```