```
--File:  WESelection.mesa
--Edited by:
--          Sandman  April 21, 1978  11:55 AM
--          Barbara  July 24, 1978  3:55 PM

DIRECTORY
  AltoFileDefs: FROM "altofiledefs" USING [FA],
  RectangleDefs: FROM "rectangledefs" USING [
    ComputeCharWidth, FAptr, leftmargin, xCoord, yCoord],
  StreamDefs: FROM "streamdefs" USING [
    GetFA, GrIndex, JumpToFA, ModifyIndex, SetIndex, StreamError],
  WindExDefs: FROM "windexdefs" USING [
    AMouseButton, GetKeySet, GetMouseButton, MenuSelect, NullProc,
    WEDataHandle, xcursorloc, ycursorloc],
  WindowDefs: FROM "windowdefs" USING [
    GetLineTable, MakeSelection, ResolveBugToPosition, Selection, StreamIndex,
    WindowHandle];

DEFINITIONS FROM StreamDefs, WindowDefs, RectangleDefs, WindExDefs;

WESelection:  PROGRAM [WEState:  WEDataHandle]
  IMPORTS WindowDefs, StreamDefs, RectangleDefs, WindExDefs
  EXPORTS WindExDefs
  SHARES StreamDefs, WindExDefs =
BEGIN

OPEN WEState;

CR: CHARACTER = 15C;
Space: CHARACTER = 40C;

Finder: TYPE = PROCEDURE [
  w: WindowHandle, sel: POINTER TO Selection, x: xCoord, y: yCoord];

FindTheChar: Finder =
  BEGIN
  width: INTEGER;
  [sel.leftline, sel.leftx, width, sel.leftindex]
    ← ResolveBugToPosition[w, x, y];
  sel.rightline ← sel.leftline;
  sel.rightindex ← sel.leftindex;
  sel.rightx ← sel.leftx + width;
  END;

TextSelect: PUBLIC PROCEDURE [w: WindowHandle, x: xCoord, y: yCoord] =
  BEGIN
  Select[w, x, y, FindTheChar, Red];
  END;

WordSelect: PUBLIC PROCEDURE [w: WindowHandle, x: xCoord, y: yCoord] =
  BEGIN
  Select[w, x, y, FindTheWord, Yellow];
  END;

Select: PROCEDURE [
  w: WindowHandle, x: xCoord, y: yCoord, find: Finder, button: AMouseButton] =
  BEGIN
  -- Declare Locals
  selection: Selection;
  exselection: Selection;
  sel: POINTER TO Selection ← @selection;
  exsel: POINTER TO Selection ← @exselection;
  fa: AltoFileDefs.FA;

  IF w.file = NIL THEN RETURN;
  GetFA[w.file, @fa];
  find[w, sel, x, y];
  MakeSelection[w, sel];
  --  check for extensions
  WHILE GetMouseButton[] = button DO
    IF x # xcursorloc↑+cxa OR y # ycursorloc↑+cya THEN
      BEGIN
      x ← xcursorloc↑; y ← ycursorloc↑;
      find[w, exsel, x, y];
      IF GrIndex[sel.leftindex, exsel.leftindex] THEN
        BEGIN
```

```
        exsel.rightx ← sel.rightx;
        exsel.rightline ← sel.rightline;
        exsel.rightindex ← sel.rightindex;
        END;
      IF GrIndex[exsel.rightindex, sel.rightindex] THEN
        BEGIN
        exsel.leftx ← sel.leftx;
        exsel.leftline ← sel.leftline;
        exsel.leftindex ← sel.leftindex;
        END;
      IF w.selection # exsel↑ THEN MakeSelection[w, exsel];
      END;
    ENDLOOP;
  JumpToFA[w.file, @fa];
  RETURN
  END;

FindTheWord: Finder =
  BEGIN
  char, newchar:  CHARACTER;
  linestarts:  DESCRIPTOR FOR ARRAY OF StreamIndex;
  index, newindex, charindex:  StreamIndex;
  pos, charpos:  xCoord;
  nlines, charline, line, width, charwidth: CARDINAL;
  class, newclass: Class;

  nlines ← (w.rectangle.ch/w.ds.lineheight)-1;
  linestarts ← DESCRIPTOR[GetLineTable[],nlines];
  [charline, charpos, width, charindex] ← ResolveBugToPosition[w, x, y];
  line ← charline - 1;  -- line 0 is first line of text in window
  SetIndex[w.file, charindex];
  char ← w.file.get[w.file];
  class ← CharClass[char];
  IF charindex = linestarts[line] AND line = 0 THEN pos ← leftmargin
  ELSE
    BEGIN
    IF charindex = linestarts[line] THEN line ← line - 1;
    newindex ← charindex;
    DO
      newindex ← ModifyIndex[newindex, -1];
      SetIndex[w.file, newindex];
      newchar ← w.file.get[w.file];
      newclass ← CharClass[newchar];
      IF newclass # class THEN GOTO foundend;
      IF newindex = linestarts[line] THEN -- linestarts[0] is first line
        IF line # 0 THEN line ← line - 1 ELSE GOTO startwindow;
      REPEAT
        startwindow => pos ← leftmargin;
        foundend =>
        BEGIN
        newindex ← ModifyIndex[newindex, 1];
        IF newindex = linestarts[line+1] THEN line ← line+1;
        SetIndex[w.file, linestarts[line]];
        pos ← leftmargin;
        FOR index ← linestarts[line], ModifyIndex[index, 1]
        UNTIL index = newindex DO
          newchar ← w.file.get[w.file];
          pos ← pos + (IF newchar = 11C THEN ComputeTabWidth[w.ds.pfont, pos]
            ELSE ComputeCharWidth[newchar, w.ds.pfont]);
          ENDLOOP;
        END;
      ENDLOOP;
    END;
  sel.leftx ← pos; sel.leftline ← line+1; sel.leftindex ← newindex;

  line ← charline - 1;  -- line 0 is first line of text in window
  newindex ← ModifyIndex[charindex, 1];
  SetIndex[w.file, newindex];
  pos ← charpos + width;
  DO
    newchar ← w.file.get[w.file ! StreamError => EXIT];
    newclass ← CharClass[newchar];
    IF newclass # class THEN GOTO foundend;
    charwidth ← IF newchar = 11C THEN ComputeTabWidth[w.ds.pfont, pos]
      ELSE ComputeCharWidth[newchar, w.ds.pfont];
    IF line+1 = nlines AND pos + charwidth >= w.rectangle.cw THEN
```

```
      GOTO foundend;
    IF newindex = linestarts[line+1] THEN
      BEGIN
      line ← line + 1;
      pos ← leftmargin;
      END;
    pos ← pos + charwidth;
    newindex ← ModifyIndex[newindex, 1];
    REPEAT
      foundend => newindex ← ModifyIndex[newindex, -1];
    ENDLOOP;
  sel.rightx ← pos; sel.rightline ← line+1; sel.rightindex ← newindex;
  RETURN
  END;

Class: TYPE = {alphanumeric, other, return};

CharClass: PROCEDURE [char: CHARACTER] RETURNS [Class] =
  BEGIN
  RETURN[SELECT char FROM
    IN ['a..'z], IN ['A..'Z], IN ['0..'9] => alphanumeric,
    CR => return,
    ENDCASE => other]
  END;

ComputeTabWidth: PROCEDURE [font: FAptr, x: xCoord] RETURNS [CARDINAL] =
  BEGIN
  tw: CARDINAL = ComputeCharWidth[' ,font] * 8;
  RETURN[tw - LOOPHOLE[x-leftmargin, CARDINAL] MOD tw]
  END;

CommandStuff: PUBLIC PROCEDURE [w: WindowHandle, x: xCoord, y: yCoord]=
  BEGIN
  n: CARDINAL;
  IF ~useKeyset THEN RETURN;
  n ← GetKeySet[];
  IF w.ks # NIL THEN
    SELECT n FROM
      IN [1..26] => w.ks.putback[w.ks, 101B+n-1];
      27 => w.ks.putback[w.ks, '+];
      31 => w.ks.putback[w.ks, 1C]; -- Control A
      ENDCASE;
  END;

-- initialization for selection module

InitSelection:  PROCEDURE =
  BEGIN
  TextProcArray[RedYellowBlue] ← NullProc;
  TextProcArray[RedBlue] ← NullProc;
  TextProcArray[RedYellow] ← CommandStuff;
  TextProcArray[Red] ← TextSelect;
  TextProcArray[BlueYellow] ← NullProc;
  TextProcArray[Blue] ← MenuSelect;
  TextProcArray[Yellow] ← WordSelect;
  TextProcArray[None] ← NullProc;
  END;

-- MAIN BODY CODE
InitSelection[];

END. of weselection
```