```
-- file DisplayControl.Mesa
-- last edited by Johnsson, October 14, 1977  2:57 PM

DIRECTORY

  AltoFileDefs: FROM "AltoFileDefs",
  ControlDefs: FROM "ControlDefs",
  DirectoryDefs: FROM "DirectoryDefs",
  DisplayDefs: FROM "DisplayDefs",
  FontDefs: FROM "FontDefs",
  ImageDefs: FROM "ImageDefs",
  SegmentDefs: FROM "SegmentDefs",
  StreamDefs: FROM "StreamDefs",
  StringDefs: FROM "StringDefs",
  SystemDisplay: FROM "SystemDisplay";

DisplayControl: PROGRAM
  IMPORTS DirectoryDefs, DisplayDefs, FontDefs, ImageDefs, SegmentDefs, StreamDefs, StringDefs, SystemD
**isplay =

  BEGIN

  mesafont: short ImageDefs.FileRequest ← [
    link: NIL, file:, access: SegmentDefs.Read,
    body: short[fill:,name: "MesaFont.al."]];
  sysfont: short ImageDefs.FileRequest ← [
    link: @mesafont, file:, access: SegmentDefs.Read,
    body: short[fill:,name: "SysFont.al."]];
  typescript: short ImageDefs.FileRequest ← [
    link: @sysfont, file:,
    access: SegmentDefs.Read+SegmentDefs.Write+SegmentDefs.Append,
    body: short[fill:,name: "Mesa.Typescript."]];

  font: FontDefs.FontHandle;
  fontseg: SegmentDefs.FileSegmentHandle;
  initialize: BOOLEAN ← TRUE;
  imaging: BOOLEAN;

  cleanupitem: ImageDefs.CleanupItem ← [
    link:, proc: Cleanup];

  Cleanup: ImageDefs.CleanupProcedure =
    BEGIN
    file: SegmentDefs.FileHandle;
    i: CARDINAL;
    si: StreamDefs.StreamIndex;
    ts: StreamDefs.StreamHandle;
    SELECT why FROM
      Finish, Abort, Save =>
        BEGIN
        IF ~initialize AND SystemDisplay.typescript # NIL THEN
          BEGIN
          StreamDefs.TruncateDiskStream[SystemDisplay.typescript];
          SystemDisplay.typescript ← NIL;
          END;
        IF why = Save AND ~initialize THEN
          BEGIN
          DisplayDefs.DisplayOff[black];
          font.destroy[font];
          SegmentDefs.DeleteFileSegment[fontseg];
          END;
        IF why # Save THEN RETURN;
        imaging ← (REGISTER[ControlDefs.SDreg]+ControlDefs.sAddFileRequest)↑ # 0;
        mesafont.file ← NIL;
        IF imaging THEN ImageDefs.AddFileRequest[@mesafont];
        sysfont.file ← NIL;
        IF imaging THEN ImageDefs.AddFileRequest[@sysfont];
        typescript.file ← NIL;
        IF imaging THEN ImageDefs.AddFileRequest[@typescript];
        END;
      Restore =>
        BEGIN OPEN SegmentDefs;
        IF (file←mesafont.file) = NIL THEN file ← sysfont.file
        ELSE ReleaseFile[sysfont.file];
        fontseg ← NewFileSegment[file,DefaultBase,DefaultPages,Read];
        font ← FontDefs.CreateFont[fontseg];
```

```
      IF initialize THEN
        BEGIN initialize ← FALSE; DisplayDefs.InitDisplay[72,30,20,font] END
      ELSE
        BEGIN SystemDisplay.SetFont[font]; DisplayDefs.DisplayOn[] END;
      IF (file←typescript.file) = NIL THEN
        file ← NewFile[typescript.name, Read+Write+Append, DefaultVersion];
      SystemDisplay.SetTypeScript[
        StreamDefs.CreateByteStream[file, Read+Write+Append]];
      END;
    InLd =>
      IF SystemDisplay.typescript # NIL THEN
        StreamDefs.OpenDiskStream[SystemDisplay.typescript];
    OutLd =>
      BEGIN OPEN StreamDefs;
      IF (ts←SystemDisplay.typescript) = NIL THEN RETURN;
      si ← GetIndex[ts];
      ts.put[ts,15C];
      FOR i IN [0..9) DO ts.put[ts,'~] ENDLOOP;
      SetIndex[ts, si];
      CloseDiskStream[ts];
      END;
    ENDCASE;
  END;

-- file requests

ProcessFileRequests: PROCEDURE [rHead: POINTER TO ImageDefs.FileRequest] =
  BEGIN OPEN AltoFileDefs;
  checkone: PROCEDURE [fp: POINTER TO FP, dname: STRING] RETURNS [BOOLEAN] =
    BEGIN
    ss: StringDefs.SubStringDescriptor ← [dname,0,dname.length];
    r: POINTER TO ImageDefs.FileRequest;
    prev: POINTER TO ImageDefs.FileRequest ← NIL;
    FOR r ← rHead, r.link UNTIL r = NIL DO
      IF (WITH r SELECT FROM
            long => StringDefs.EquivalentSubStrings[@ss,@name],
            short => StringDefs.EquivalentString[dname,name],
            ENDCASE => FALSE) THEN
        BEGIN
        IF r.file = NIL THEN r.file ← SegmentDefs.InsertFile[fp,r.access]
        ELSE r.file.fp ← fp↑;
        IF prev = NIL THEN rHead ← r.link
        ELSE prev.link ← r.link;
        END
      ELSE prev ← r;
      ENDLOOP;
    RETURN[rHead = NIL]
    END;

  DirectoryDefs.EnumerateDirectory[checkone];
  END;

ImageDefs.AddCleanupProcedure[@cleanupitem];
Cleanup[Save];
IF imaging THEN STOP ELSE ProcessFileRequests[@typescript];
Cleanup[Restore];

END...
```