```
-- StreamDefs.Mesa  Edited by Sandman on April 10, 1978  10:59 AM

DIRECTORY
  AltoDefs: FROM "altodefs",
  AltoFileDefs: FROM "altofiledefs",
  SegmentDefs: FROM "segmentdefs",
  RectangleDefs: FROM "rectangledefs";

StreamDefs: DEFINITIONS = BEGIN

  KeyBufChars: PRIVATE INTEGER = 80;

  StreamHandle: TYPE = POINTER TO StreamObject;
  KeyboardHandle: TYPE = POINTER TO Keyboard StreamObject;
  DisplayHandle: TYPE = POINTER TO Display StreamObject;
  DiskHandle: TYPE = POINTER TO Disk StreamObject;
  OtherStreamHandle: TYPE = POINTER TO Other StreamObject;

  StreamObject: TYPE = RECORD [
    reset: PROCEDURE [StreamHandle],
    get: PROCEDURE [StreamHandle] RETURNS [UNSPECIFIED],
    putback: PROCEDURE [StreamHandle, UNSPECIFIED],
    put: PROCEDURE [StreamHandle, UNSPECIFIED],
    endof: PROCEDURE [StreamHandle] RETURNS [BOOLEAN],
    destroy: PROCEDURE [StreamHandle],
    body: PRIVATE SELECT type: PUBLIC * FROM
      Keyboard => [
        in, out: CARDINAL,
        buffer: PACKED ARRAY [0..KeyBufChars) OF CHARACTER],
      Display => [
        options: PUBLIC DSOptions,
        charx: RectangleDefs.xCoord,
        TABindex: [0..9],
        chary: RectangleDefs.yCoord,
        link: PUBLIC DisplayHandle,
        pfont: PUBLIC RectangleDefs.FAptr,
        rectangle: PUBLIC RectangleDefs.Rptr,
        lineheight: PUBLIC CARDINAL,
        line: PUBLIC CARDINAL,
        TABs: ARRAY [0..9] OF CARDINAL],
      Disk => [
        eof, dirty: BOOLEAN,
        read, write, append: PUBLIC BOOLEAN,
        unit: [1..2],
        index, size: CARDINAL,
        getOverflow: PROCEDURE [StreamHandle],
        savedGet: PROCEDURE [StreamHandle] RETURNS [UNSPECIFIED],
        putOverflow: PROCEDURE [StreamHandle],
        savedPut: PROCEDURE [StreamHandle, UNSPECIFIED],
        page: AltoDefs.PageNumber,
        char: CARDINAL,
        buffer: RECORD [
          SELECT OVERLAID * FROM
            byte => [
              byte: POINTER TO PACKED ARRAY [0..0) OF AltoDefs.BYTE],
            word => [
              word: POINTER TO ARRAY [0..0) OF WORD],
            ENDCASE],
        file: PUBLIC SegmentDefs.FileHandle,
        das: ARRAY {last, current, next} OF AltoFileDefs.vDA],
      Other => [data: PUBLIC POINTER],
      ENDCASE];

  StreamError: SIGNAL [stream:StreamHandle, error:StreamErrorCode];

  StreamErrorCode: TYPE = {
    StreamType, StreamAccess, StreamOperation,
    StreamPosition, StreamEnd, StreamBug};


  -- extensions applicable to keyboard streams

  CreateKeyStream: PROCEDURE RETURNS [KeyboardHandle];
  GetDefaultKey, GetCurrentKey: PROCEDURE RETURNS [KeyboardHandle];
  OpenKeyStream, CloseKeyStream: PROCEDURE [stream:StreamHandle];
  CursorTrack: PROCEDURE [BOOLEAN];
  ControlDELtyped: PROCEDURE RETURNS [BOOLEAN];
  ResetControlDEL: PROCEDURE;
  KeyStreams: PROGRAM;

  -- extensions applicable to display streams

  -- Display Stream options field definitions

  DSOptions: TYPE = RECORD [    -- DisplayStream options
    StopRight: BOOLEAN, -- discard rather than wrap line
    StopBottom: BOOLEAN,        -- discard rather than scroll
    NoteLineBreak: BOOLEAN,     -- SIGNAL on line breaks
    NoteScrolling: BOOLEAN];    -- SIGNAL on Scroll

  GetDefaultDisplayStream: PROCEDURE RETURNS [DisplayHandle];
  GetDisplayStreamList: PROCEDURE RETURNS [DisplayHandle];
  ClearCurrentLine: PROCEDURE [stream:StreamHandle];
  ClearDisplayLine: PROCEDURE [stream:StreamHandle, line:CARDINAL];
  ClearDisplayChar: PROCEDURE [stream:StreamHandle, char:CHARACTER];
  CreateDisplayStream: PROCEDURE [rectangle:RectangleDefs.Rptr] RETURNS [DisplayHandle];

  -- display stream procedures used by Window stuff

  ScrollDisplay: PROCEDURE [ds:DisplayHandle, char:CHARACTER];
  SetDisplayLine: PROCEDURE [ds:DisplayHandle, line, pos:CARDINAL];
  WriteDisplayChar: PROCEDURE [stream:StreamHandle, char:CHARACTER];

  -- extensions applicable to disk streams

  StreamIndex: TYPE = RECORD [
    page: AltoDefs.PageNumber,
    byte: CARDINAL];

  AccessOptions: TYPE = SegmentDefs.AccessOptions;
    Read: AccessOptions = SegmentDefs.Read;
    Write: AccessOptions = SegmentDefs.Write;
    Append: AccessOptions = SegmentDefs.Append;
--  Delete: AccessOptions = SegmentDefs.Delete;

  DefaultAccess: AccessOptions = SegmentDefs.DefaultAccess;

  NewByteStream, NewWordStream: PROCEDURE [name: STRING, access: AccessOptions]
    RETURNS [DiskHandle];
  CreateByteStream, CreateWordStream: PROCEDURE [
    file: SegmentDefs.FileHandle, access: SegmentDefs.AccessOptions]
    RETURNS [DiskHandle];
  OpenDiskStream, CloseDiskStream: PROCEDURE [stream: StreamHandle];
  CleanupDiskStream, TruncateDiskStream: PROCEDURE [stream: StreamHandle];
  FileLength: PROCEDURE [stream: StreamHandle] RETURNS [StreamIndex];
  GetFA: PROCEDURE [stream:StreamHandle, fa:POINTER TO AltoFileDefs.FA];
  JumpToFA: PROCEDURE [stream:StreamHandle, fa:POINTER TO AltoFileDefs.FA];
  ReadBlock, WriteBlock: PROCEDURE [stream:StreamHandle, address:POINTER, words:CARDINAL]
    RETURNS [CARDINAL];

  -- StreamIndex Stuff

  GetIndex: PROCEDURE [stream: StreamHandle] RETURNS [StreamIndex];
  SetIndex: PROCEDURE [stream: StreamHandle, index: StreamIndex];
  NormalizeIndex: PROCEDURE [index: StreamIndex] RETURNS [StreamIndex];
  ModifyIndex: PROCEDURE [index: StreamIndex, change: INTEGER] RETURNS [StreamIndex];
  EqualIndex:  PUBLIC PROCEDURE [i1, i2:  StreamIndex] RETURNS[BOOLEAN];
  GrEqualIndex:  PUBLIC PROCEDURE [i1, i2:  StreamIndex] RETURNS[BOOLEAN];
  GrIndex:  PUBLIC PROCEDURE [i1, i2:  StreamIndex] RETURNS[BOOLEAN];

  END.
```