```
-- file: FSPDefs.Mesa; edited by Sandman on August 23, 1977  9:26 PM

DIRECTORY
  AltoDefs: FROM "altodefs";

DEFINITIONS FROM AltoDefs;

FSPDefs: DEFINITIONS =
  BEGIN

  -- types and formats of zone headers and nodes

  BlockSize: TYPE = INTEGER [0..77777B--VMLimit/2--];

  NodePointer: TYPE = POINTER TO NodeHeader;
    FreeNodePointer: TYPE = POINTER TO free NodeHeader;

  NodeHeader: TYPE = PRIVATE RECORD[
    length: BlockSize,
    extension: SELECT state: * FROM
      inuse =>
        NULL,
      free => [
        fwdp, backp: FreeNodePointer],
      ENDCASE];


  Deallocator: TYPE = PROCEDURE [POINTER];

  ZonePointer: TYPE = POINTER TO ZoneHeader;

  ZoneHeader: TYPE = PRIVATE RECORD[
    node: free NodeHeader,
    rover: FreeNodePointer,
      -- roving pointer to slow down fragmentation
      -- (see Knuth, Vol I, p. 597 #6)
    restofzone: ZonePointer,  -- link to additional segments of zone
    length: BlockSize,
    deallocate: Deallocator,
    threshold: PUBLIC BlockSize,
    checking: PUBLIC BOOLEAN];

  ZoneOverhead: INTEGER = SIZE[ZoneHeader]+SIZE[inuse NodeHeader];


  -- public procedures and signals

  MakeNewZone: PROCEDURE [base: POINTER, length: BlockSize, deallocate: Deallocator]
    RETURNS [z: ZonePointer];
  MakeZone: PROCEDURE [base: POINTER, length: BlockSize] RETURNS [z: ZonePointer];
  AddToNewZone: PROCEDURE
    [z: ZonePointer, base: POINTER, length: BlockSize, deallocate: Deallocator];
  AddToZone: PROCEDURE [z: ZonePointer, base: POINTER, length: BlockSize];
  PruneZone: PROCEDURE [z: ZonePointer] RETURNS [BOOLEAN];
  DestroyZone: PROCEDURE [z: ZonePointer];
  DoNothingDeallocate: Deallocator;

  NoRoomInZone: SIGNAL [z: ZonePointer];  -- not enough space to fill a request

  MakeNode: PROCEDURE [z: ZonePointer, n: BlockSize] RETURNS [POINTER];
  FreeNode: PROCEDURE [z: ZonePointer, p: POINTER];
  SplitNode: PROCEDURE [z: ZonePointer, p: POINTER, n: BlockSize];
  NodeSize: PROCEDURE [p: POINTER] RETURNS [BlockSize];

  ZoneTooSmall: ERROR [POINTER];
  InvalidZone: ERROR [POINTER];  -- zone header looks fishy
  NodeLoop: ERROR [ZonePointer];
  InvalidNode: ERROR [POINTER];  -- node appears damaged

  END.
```

```
-- ImageDefs.mesa  Modified by: Sandman, August 29, 1977  22:47 PM

DIRECTORY
  AltoDefs: FROM "altodefs",
  BcdDefs: FROM "bcddefs",
  ControlDefs: FROM "controldefs",
  Mopcodes: FROM "mopcodes",
  NovaOps: FROM "novaops",
  SegmentDefs: FROM "segmentdefs",
  StringDefs: FROM "stringdefs";

ImageDefs: DEFINITIONS =
  PRIVATE BEGIN

  MapItem: TYPE = MACHINE DEPENDENT RECORD [page, count: [0..255]];
  MapIndexType: TYPE =
    [0..AltoDefs.PageSize-5-SIZE[ControlDefs.StateVector]-2*SIZE[BcdDefs.VersionStamp]);
  FirstImageDataPage: AltoDefs.PageNumber = 2;
  HeaderPages: CARDINAL = 1;

  VersionID: CARDINAL = 08277;  -- must match in Mesa.bcpl

  ImageHeader: TYPE = MACHINE DEPENDENT RECORD [
    versionident: CARDINAL,
    version, creator: BcdDefs.VersionStamp,
    options: WORD,
    av, gft, sd: POINTER,
    state: ControlDefs.StateVector,
    map: ARRAY MapIndexType OF MapItem];

  FileRequest: PUBLIC TYPE = MACHINE DEPENDENT RECORD[
    link: POINTER TO FileRequest,
    file: SegmentDefs.FileHandle,
    access: SegmentDefs.AccessOptions,
    body: SELECT tag:* FROM
      short => [fill: [0..7777B], name: STRING],
      long => [fill: [0..7777B], name: StringDefs.SubStringDescriptor],
      ENDCASE];

  AddFileRequest: PUBLIC MACHINE CODE [POINTER TO FileRequest] =
    INLINE [Mopcodes.zKFCB, ControlDefs.sAddFileRequest];
  MakeImage: PUBLIC PROCEDURE [name: STRING, symbolsToImage: BOOLEAN];
  MakeUnMergedImage: PUBLIC PROCEDURE [name: STRING, symbolsToImage: BOOLEAN];
  SwapOutDuringMakeImage: PUBLIC SIGNAL;
  SwapTrapDuringMakeImage: PUBLIC SIGNAL;
  SwapErrorDuringMakeImage: PUBLIC SIGNAL;

  ImageVersion: PUBLIC PROCEDURE RETURNS [BcdDefs.VersionStamp];

  CleanupItem: PUBLIC TYPE = RECORD [
    link: POINTER TO CleanupItem,
    proc: CleanupProcedure];

  CleanupProcedure: PUBLIC TYPE = PROCEDURE [why: CleanupReason];

  CleanupReason: PUBLIC TYPE = NovaOps.CleanupReason;
  NovaOpcode: TYPE = NovaOps.NovaOpcode;

  AddCleanupProcedure, RemoveCleanupProcedure: PUBLIC PROCEDURE [POINTER TO CleanupItem];
  UserCleanupProc: PUBLIC CleanupProcedure;

  StopMesa: PUBLIC MACHINE CODE =
    INLINE[Mopcodes.zLIn+LOOPHOLE[NovaOpcode[Finish], CARDINAL], Mopcodes.zSTOP];
  AbortMesa: PUBLIC MACHINE CODE =
    INLINE[Mopcodes.zLIn+LOOPHOLE[NovaOpcode[Abort], CARDINAL], Mopcodes.zSTOP];
  PuntMesa: PUBLIC MACHINE CODE =
    INLINE[Mopcodes.zLIn+LOOPHOLE[NovaOpcode[Punt], CARDINAL], Mopcodes.zSTOP];

  END..
```