```
-- file SymDefs.Mesa
-- last modified by Satterthwaite, July 10, 1978  4:08 PM

DIRECTORY
  AltoDefs: FROM "altodefs",
  BcdDefs: FROM "bcddefs",
  TableDefs: FROM "tabledefs";

SymDefs: DEFINITIONS =
  BEGIN

  VersionID: CARDINAL = 04118;

 -- hash table declarations

  HVLength: PRIVATE CARDINAL = 71;
  HVIndex: TYPE = CARDINAL [0..HVLength);


  HTRecord: TYPE = RECORD [
    anyInternal, anyPublic: BOOLEAN,
    link: HTIndex,
    ssIndex: CARDINAL];

  HTIndex: TYPE = CARDINAL [0..TableDefs.TableLimit/2);

  HTNull: HTIndex = FIRST[HTIndex];


 -- semantic entry table declarations

  TypeClass: TYPE = {
    mode,
    basic,
    enumerated,
    record,
    pointer,
    array,
    arraydesc,
    transfer,
    definition,
    union,
    relative,
    subrange,
    long,
    real,
    nil
    };

  TransferMode: TYPE = {procedure, port, signal, error, process, program, none};

  SERecord: TYPE = RECORD [
    mark3, mark4: BOOLEAN,
    sebody: SELECT setag: * FROM
      id => [
        extended: BOOLEAN,
        public: BOOLEAN,
        ctxnum: CTXIndex,
        writeonce, constant: BOOLEAN,
        idtype: SEIndex,
        idinfo: UNSPECIFIED,
        idvalue: UNSPECIFIED,
        htptr: HTIndex,
        linkSpace: BOOLEAN,
        ctxlink: SELECT linktag: * FROM
          terminal => NULL,
          sequential => NULL,
          linked => [link: ISEIndex],
          ENDCASE],
      constructor => [
        typeinfo: SELECT typetag: TypeClass FROM
          mode => NULL,
          basic => [
            ordered: BOOLEAN,
            code: [0..16),
            length: CARDINAL],
        enumerated => [
          ordered: BOOLEAN,
          valuectx: CTXIndex,
          nvalues: CARDINAL],
        record => [
          machineDep: BOOLEAN,
          monitored: BOOLEAN,
          unifield, argument: BOOLEAN,
          defaultFields: BOOLEAN,
          comparable: BOOLEAN,
          privateFields: BOOLEAN,
          lengthUsed: BOOLEAN,
          length: CARDINAL,
          fieldctx: CTXIndex,
          variant: BOOLEAN,
          linkpart: SELECT linktag: * FROM
            notlinked => NULL,
            linked => [linktype: SEIndex],
            ENDCASE],
        pointer => [
          ordered, readonly, basing: BOOLEAN,
          dereferenced: BOOLEAN,
          pointedtotype: SEIndex],
        array => [
          packed: BOOLEAN,
          comparable: BOOLEAN,
          lengthUsed: BOOLEAN,
          indextype: SEIndex,
          componenttype: SEIndex],
        arraydesc => [describedType: SEIndex],
        transfer => [
          mode: TransferMode,
          inrecord, outrecord: recordCSEIndex],
        definition => [
          nGfi: [1 .. 4],
          defCtx: CTXIndex],
        union => [
          equalLengths: BOOLEAN,
          casectx: CTXIndex,
          overlayed, controlled: BOOLEAN,
          tagsei: ISEIndex],
        relative => [
          baseType: SEIndex,
          offsetType: SEIndex,
          resultType: SEIndex],
        subrange => [
          filled, empty, flexible: BOOLEAN,
          rangetype: SEIndex,
          origin: INTEGER,
          range: CARDINAL],
        long, real => [rangetype: SEIndex],
        nil => NULL,
        ENDCASE],
    ENDCASE];

  SEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO SERecord;

  ISEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO id SERecord;
  CSEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO constructor SERecord;
    recordCSEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO record constructor SERecord;
    arrayCSEIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO array constructor SERecord;

  SENull: SEIndex = FIRST[SEIndex];
    ISENull: ISEIndex = LOOPHOLE[SENull];
    CSENull: CSEIndex = LOOPHOLE[SENull];
      recordCSENull: recordCSEIndex = LOOPHOLE[SENull];
      arrayCSENull: arrayCSEIndex = LOOPHOLE[SENull];

-- the following two values are guaranteed by the compiler
  typeTYPE: CSEIndex = FIRST[CSEIndex] + SIZE[nil constructor SERecord];
  typeANY: CSEIndex = typeTYPE + SIZE[mode constructor SERecord];

-- codes identifying the basic types (extensible)
  codeANY: CARDINAL = 0;
  codeINTEGER: CARDINAL = 1;
  codeBOOLEAN: CARDINAL = 2;
  codeCHARACTER: CARDINAL = 3;

  BitAddress: TYPE = RECORD[
    wd: [0..AltoDefs.VMLimit/AltoDefs.wordlength],      -- word displacement
    bd: [0..AltoDefs.wordlength)];              -- bit displacement


 -- context table declarations

  MaxContextLevel: CARDINAL = 7;

  ContextLevel: TYPE = [0..MaxContextLevel];
    lZ: ContextLevel = 0;       -- context level of non-frame records
    lG: ContextLevel = 1;       -- context level of global frame
    lL: ContextLevel = lG+1;    -- context level of outer procedures

  CTXRecord: TYPE = RECORD [
    sn: Sn,      -- for DeSoto
    selist: ISEIndex,
    ctxlevel: ContextLevel,
    extension: SELECT ctxType: * FROM
      simple => [ctxNew: CTXIndex],      -- for DeSoto
      included => [
        ctxchain: includedCTXIndex,
        ctxmodule: MDIndex,
        ctxmap: CTXIndex,
        ctxclosed, ctxcomplete, restricted: BOOLEAN,
        ctxreset: BOOLEAN],
      imported => [includeLink: includedCTXIndex],
      nil => NULL,
      ENDCASE];

  CTXIndex: TYPE = ORDERED POINTER [0..3777B] TO CTXRecord;
   includedCTXIndex: TYPE = ORDERED POINTER [0..3777B] TO included CTXRecord;

  CTXNull: CTXIndex = FIRST[CTXIndex];
    includedCTXNull: includedCTXIndex = LOOPHOLE[CTXNull];


 -- module table declarations

  FileIndex: TYPE = [0..77777B];         -- internal file handle
  nullFileIndex: FileIndex = LAST[FileIndex];

  MDRecord: TYPE = RECORD [
    mdhti: HTIndex,             -- hash entry for file name
    mdctx: includedCTXIndex,    -- context of copied entries
    mdshared: BOOLEAN,          -- overrides PRIVATE, etc.
    mdExported: BOOLEAN,
    mdStamp: BcdDefs.VersionStamp,
    mdFile: FileIndex];         -- associated file

  MDIndex: TYPE = ORDERED POINTER [0..TableDefs.TableLimit) TO MDRecord;
  MDNull: MDIndex = LAST[MDIndex];

  OwnMdi: MDIndex = FIRST[MDIndex];


 -- body table declarations

  BodyLink: TYPE = RECORD [which: {sibling, parent}, index: BTIndex];

  BodyRecord: TYPE = RECORD [
    link: BodyLink,
    firstSon: BTIndex,
    localCtx: CTXIndex,
    level: ContextLevel,
    info: BodyInfo,
    extension: SELECT kind: * FROM
      Callable => [
        id: ISEIndex,
        ioType: SEIndex,
        monitored, stopping: BOOLEAN,
        entryIndex: [0..128),
        entry, internal: BOOLEAN,
        closure: SELECT nesting: * FROM
        Outer => NULL,
        Inner => [frameOffset: [0..AltoDefs.VMLimit]],
        ENDCASE],
    Other => NULL,
    ENDCASE];

  BodyInfo: TYPE = RECORD [
    SELECT mark: * FROM
      Internal => [
        bodyTree: --TreeIndex-- POINTER [0..TableDefs.TableLimit),
        sourceIndex: CARDINAL,
        stOrigin: --LitDefs.STIndex-- POINTER [0..TableDefs.TableLimit/2),
        frameSize: [0..4096)],
      External => [
        origin: [0..AltoDefs.VMLimit/2],
        bytes: CARDINAL,
        startIndex, indexLength: CARDINAL],
      ENDCASE];

 BTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO BodyRecord;
   CBTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO Callable BodyRecord;
     ICBTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO Inner Callable BodyRecord;
     OCBTIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO Outer Callable BodyRecord;
 BTNull: BTIndex = LAST[BTIndex];
   CBTNull: CBTIndex = LOOPHOLE[BTNull];


-- definitions for use by DeSoto

 Sn: TYPE = {snNil, snValid, snInvalid, snIndirect};


-- allocation codes for table components

 setype: TableDefs.TableSelector = 1;
 httype: TableDefs.TableSelector = 2;
 sstype: TableDefs.TableSelector = 3;
 ctxtype: TableDefs.TableSelector = 4;
 mdtype: TableDefs.TableSelector = 5;
 bodytype: TableDefs.TableSelector = 6;


-- symbol segment headers

 WordOffset: TYPE = CARDINAL;
 BlockDescriptor: TYPE = RECORD [offset: WordOffset, size: CARDINAL];

 STHeader: TYPE = RECORD [
   versionIdent: CARDINAL,
   version: BcdDefs.VersionStamp,
   sourceVersion: BcdDefs.VersionStamp,
   creator: BcdDefs.VersionStamp,
   definitionsFile: BOOLEAN,
   directoryCtx, importCtx, outerCtx: CTXIndex,
   hvBlock: BlockDescriptor,
   htBlock: BlockDescriptor,
   ssBlock: BlockDescriptor,
   seBlock: BlockDescriptor,
   ctxBlock: BlockDescriptor,
   mdBlock: BlockDescriptor,
   bodyBlock: BlockDescriptor,
   extBlock: BlockDescriptor,
   treeBlock: BlockDescriptor,
   litBlock: BlockDescriptor,
   fgRelPgBase: CARDINAL,
   fgPgCount: AltoDefs.PageCount];


-- fine grain table header

 fgHeader: TYPE = RECORD [
   fgoffset: WordOffset,
   fglength: CARDINAL,
   sourcefile: StringBody -- text follows --];

-- fine grain table declarations
```

```
ByteIndex: TYPE = CARDINAL;

FGTEntry: TYPE =  RECORD [
  findex: ByteIndex,
  cindex: ByteIndex];

END.
```