# Cedar Style Sheet

```
-- FILE: CedarSample.mesa
-- Last edited by Mitchell, April 23, 1980 12:03 PM

CedarSample:  DEFINITIONS  =
BEGIN
upperLimit:  INTEGER  =  32;    -- name the upper limit of an interval type
IntervalType: TYPE  =  [0 .. upperLimit];   -- preferred form for intervals
Sample:  TYPE =  REF SampleRec;
SampleRec: TYPE = RECORD[val: Value, next: Sample];
SampleSet: TYPE = REF SampleSetRec;
SampleSetRec: TYPE = RECORD[head: Sample, count: Value];
Problem: ERROR [reason: ErrorCode];
ErrorCode: TYPE = {damagedSample, callingError, programError};
NewSample :  PROCEDURE [val: Value]  RETURNS  [Sample];
NewSampleSet: PROCEDURE RETURNS[nilSet: SampleSet];
RemoveSample: PROCEDURE[toBeRemoved: Sample, from: SampleSet]
                       RETURNS [inSet: BOOLEAN, setMinus: SampleSet];
  -- ERRORs: Problem[damagedSample]
  -- SIGNALs: ResumableCondition, SomeSignal
emptySet:  SampleSet  =  NIL;
END.
This is where a global description of the module goes
CHANGE LOG
Changed by: YourName:  DateTime
  DescriptionOfChange
```

No common suffix on the names of DEFINITIONS modules

Standard prelude for a module

Capitalize the first letter of module, procedure, signal, or type names

Naming a type is better than using anonymous type constructor.

Use a small set of ERRORs with an error code parameter.

attach stylized comments to proce- dures that generate ERRORs or SIGNALs.

Capitalize the first letter of each imbedded word of a multi-word name

Standard postlude for a module includes a history log of (non-triv changes to the module

```
-- FILE: CedarSampleImpl.mesa
-- Last edited by Mitchell, April 23, 1980 3:19 PM
DIRECTORY
  SomeInterface USING [SomeProc, SomeType],
  CedarSample;
CedarSampleImpl  PROGRAM  IMPORTS  SI: SomeInterface
               EXPORTS CedarSample  =
BEGIN  OPEN  CedarSample;
Problem: ERROR[reason: ErrorCode]  =  CODE;
ResumableCondition: PUBLIC SIGNAL = CODE;
SomeSignal: PUBLIC SIGNAL = CODE;
NewSample: PUBLIC PROCEDURE [val: Value]  RETURNS [Sample]  =
  BEGIN
  sample: Sample;
  x: SI.SomeType=  0;
  IF  val = 0  THEN  SI.SomeProc[x];
  . . .
  RETURN [sample];
  END;
NewSampleSet: PUBLIC  PROCEDURE  RETURNS [nilSet: SampleSet]  =
  BEGIN
  nilSet _ AllocateSampleSet[ ! AllocFault =>  ERROR  Problem[programError]];
  nilSet  [head: NIL,  count: 0];
  END;
Bug: ERROR = CODE;
RemoveSample: PUBLIC  PROCEDURE [toBeRemoved: Sample, from: SampleSet]
                       RETURNS [inSet: BOOLEAN, setMinus: SampleSet]  =
  -- ERRORs: Problem[damagedSample]
   BEGIN OPEN  SI;
   IF . . . THEN ERROR Problem[damagedSample];
   WITH refAnyVar SELECT FROM
    r: REF REAL => r^ _ r^ + 1.0;
    i: REF INT => { . . SomeProc[y]; . . .  };
    b: REF  BOOLEAN => SIGNAL  ResumableCondition;
    ENDCASE  => ERROR Bug;
  . . .
  END;
AllocFault: ERROR =  CODE;         --  error for local use in this module
AllocateSampleSet : PROCEDURE RETURNS [uninitedSet: SampleSet]  =
  BEGIN
  . . .
  END;
END.
CHANGE LOG
Changed by: YourName:  DateTime
  DescriptionOfChange
```

OK to use unnamed OPEN of interface if DIRECTORY entry has a USING list

PROGRAM module name is normaly formed by suffixing interface name with "Impl". Alternatively, name may be totally different than interface's

OK to OPEN interface that module implements

NO names in same scope differing only by letter case distinctions except a value with same name as its type but with lowercase first letter

Qualify identifiers from interfaces

Only let signals that are part of t abstraction escape out of it.

Keyword constructors, argument lists, and extractors preferred for multiple-component constructors

OK to OPEN an interface in a local scope where it is heavily used

Use REF ANY instead of variant records and discriminate

Only raise SIGNALs using SIGNAL, and ERROR using ERROR

Calls on single-argument procedures don't have to use keyword notation

NO using ENDCASE to handle a single remaining case: use as "OTHERWISE" or to generate an ERROR

OK to have locally defined ERRORs and SIGNALs

JGM   16 AUG 82 Print on color printer if you can.          Filed on [Indigo]<CedarDocs>Language>StyleSheet.sil