UtilityD

COLLABORATORS			
	TITLE : UtilityD		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		November 24, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

UtilityD

1.1 UtilityD

Konstanten

tagDone	tagEnd	tagFilterAND
tagFilterNOT	tagIgnore	tagMore
tagSkip	tagUser	utilityName

Typ-Deklarationen

ClockData	ClockDataPtr	Hook
HookProc	HookPtr	Tag
TagItem	TagItemPtr	TagPtr

1.2 TMP:Modula-2/UtilityD.def

```
DEFINITION MODULE UtilityD; (*$Implementation:=FALSE*)
(* 26-Feb-1992/cn *)

FROM SYSTEM IMPORT ADDRESS;

FROM ExecD IMPORT
  MinNode ;

IMPORT
 R;

CONST
 utilityName="utility.library";

TYPE
  Tag =LONGCARD;
  TagItem =RECORD
  tag: Tag ;
  data:LONGCARD;
```

```
 END;

  TagPtr =POINTER TO  Tag ;
  TagItemPtr =POINTER TO  TagItem ;
(*
 Genaugenommen zeigen TagPtr und TagItemPtr meist auf eine Folge von
 Tag bzw. Tagitem.
*)

CONST
 tagDone= Tag (0); tagEnd=tagDone;
 tagIgnore= Tag (1);
 tagMore= Tag (2);
 tagSkip= Tag (3);

 tagUser= Tag (80000000H);

 tagFilterAND=0;
 tagFilterNOT=1;

TYPE
  ClockData =RECORD
  sec,min,hour:CARDINAL;
  mday,month,year:CARDINAL;
  wday:CARDINAL;
 END;
  ClockDataPtr =POINTER TO  ClockData ;

  HookPtr =POINTER TO  Hook ;
  HookProc =PROCEDURE(HookPtr{R.A0},
                   (*object*)ADDRESS{R.A2},
                   (*message*)ADDRESS{R.A1}):ADDRESS;
  Hook =RECORD
  node: MinNode ;
  entry: HookProc ;
  subEntry:ADDRESS;
  data:ADDRESS;
 END;
(*
 "entry" ist die Prozedur die tatsächlich aufgerufen wird. Bei
 Hochsprachen, die keine Registervariabeln kennen, würde "entry" eine
 Assemblerprozedur enthalten, dass die Register auf den Stack lädt,
 und dann die in "subEntry" eingetragene Prozedur aufruft.

 Da M2Amiga eine Parameterübergabe in den Registern erlaubt, kann
 eine korrekt deklarierte Prozedur direkt in "entry" eingetragen
 werden.
*)

END UtilityD.noimp
```