

**Arts**

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> Arts		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		November 24, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>Arts</b>	<b>1</b>
1.1	Arts . . . . .	1
1.2	TMP:Modula-2/Arts.def . . . . .	1

# Chapter 1

## Arts

### 1.1 Arts

#### Prozeduren

Assert	BreakPoint	CloseLib
Divs32	Divu32	Error
Exit	Muls32	Mulu32
OpenLib	Requester	Startup
StkChk	SystemError	Terminate

#### Konstanten

maxKeys	maxModName
---------	------------

#### Variablen

dosCmdBuf	dosCmdLen	errorFrame
kickVersion	oldCurrentDir	programName
reserved2	reserved3	reserved4
returnVal	startupMsg	thisTask
wbStarted		

#### Typ-Deklarationen

ErrorFrame	ErrorType	ModKeys
ModName	ModType	StrPtr
SysErr		

### 1.2 TMP:Modula-2/Arts.def

```
(*
 *      // / _____ \      //
 *      // \      Amiga Run Time System      /      //
 * \ // /      4.02 / 23.02.91 / bp      \ \ //
 * \X/ \      _____ /      \X/
 *)
```

```
DEFINITION MODULE Arts;

(*$ LargeVars:=FALSE
   LongAlign:=FALSE
   NameChk:=FALSE
*)

FROM SYSTEM IMPORT
  ADDRESS, LONGSET, BPTR;

CONST
  maxModName=32; maxKeys=3;

TYPE
  (* Typ für M2Debug: Arts.StrPtr *)
  StrPtr = POINTER TO ARRAY [0..63] OF CHAR;
  ErrorType =
    (trap, exception, system, assertion, breakPoint, explicit);
  SysErr = (halt, illCase, fctReturn, stkOvl, illCall);
  ErrorFrame = RECORD
    pc: ADDRESS;
    dRegs: ARRAY [0..7] OF LONGINT;
    aRegs: ARRAY [8..15] OF ADDRESS;
    CASE error: ErrorType OF
      | trap: trapNr: INTEGER;
      | exception: exceptionMask: LONGSET
      | system: sysErr: SysErr
      | assertion, breakPoint, explicit:
    END;
    header, body: ADDRESS
  END;
  ModType = (none, mod, lib, noImp);
  ModKeys = ARRAY [0..maxKeys-1] OF CARDINAL;
  ModName = ARRAY [0..maxModName-1] OF CHAR;

VAR
  (*
   * Umgebungsvariablen von M2Amiga.
   *
   * Einzig returnVal darf verändert,
   * alle anderen nur gelesen werden!!
   *)

  wbStarted: BOOLEAN;
  (*
   * TRUE: Start von Workbench, FALSE: Start von CLI
   *)

  returnVal: LONGINT;
  (*
   * Dies ist der Rückgabewert an das Dos,
   * kann beliebig gesetzt werden.
   * Bei Workbench-Start wird returnVal als INTEGER
   * in das Feld length der WBStartupMsg eingetragen.
   *)
```

---

```
startupMsg: ADDRESS;
(*
 * Zeiger auf die Workbench.WBStartup Message
 * Nur gültig bei Workbench-Start (wbStarted),
 * sonst NIL!
 *)

dosCmdBuf: ADDRESS;
dosCmdLen: LONGINT;
(*
 * Adresse und Länge der CLI-Argumente.
 * Nur gültig bei CLI-Start (NOT wbStarted)!
 *)

programName: ADDRESS;
(*
 * Zeiger auf den Aufrufnamen des Programmes.
 * Der String endet mit 0C.
 *)

thisTask: ADDRESS;
(*
 * Zeiger auf den eigenen Task/Process
 *)

oldCurrentDir:BPTR;
(*
 * CurrentDir bei Programmstart
 *)

errorFrame: ErrorFrame ;
(*
 * errorFrame wird bei einem Laufzeitfehler mit den
 * aktuellen Werten gefüllt.
 *)

kickVersion: INTEGER;
(*
 * Die Version der exec.library (32,33,36,...)
 * Hiermit kann die Version von Kickstart,Workbench
 * festgestellt werden.
 *)

(*
 * Für Erweiterungen der ENTWICKLER!
 * Verhindert eine ständige Neudefinition von Arts.
 *)
reserved2: ADDRESS;
reserved3: ADDRESS;
reserved4: ADDRESS;

(* Diese Prozedur ist IMMER die erste jedes Programms,
 * NIEMALS aufrufen!!!!
 *)
PROCEDURE Startup(cl{0}:LONGINT; cb{8}:ADDRESS): LONGINT;
```

---

```
(* Compiler Unterstützung *)
PROCEDURE StkChk(need{0}: LONGINT);
  (* SystemError(halt) = HALT *)
PROCEDURE SystemError(err{0}: SysErr );
PROCEDURE Mulu32(x{0},y{1}: LONGINT): LONGINT;
  (* D0=REM, D1=QUO *)
PROCEDURE Divu32(x{0},y{1}: LONGINT): LONGINT;
PROCEDURE Muls32(x{0},y{1}: LONGINT): LONGINT;
  (* D0=REM, D1=QUO *)
PROCEDURE Divs32(x{0},y{1}: LONGINT): LONGINT;
(* Nur für die "automatischen" Libraries! *)
PROCEDURE OpenLib(version{0}:LONGINT;
                  name{9}:ADDRESS):ADDRESS;
PROCEDURE CloseLib(base{9}:ADDRESS);

(* Laufzeitunterstützung *)
PROCEDURE Assert(condition: BOOLEAN; msg: ADDRESS);
PROCEDURE BreakPoint(msg: ADDRESS);
PROCEDURE Error(header,body: ADDRESS);
PROCEDURE Requester(header,body,pos,neg:ADDRESS):BOOLEAN;
PROCEDURE Terminate;
PROCEDURE Exit (retVal{0}:LONGINT);

END Arts.
```

---