OD.doc

COLLABORATORS			
	TITLE : OD.doc		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		December 7, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

OD.doc

1.1 OD -- The Oberon-A module definition utility

```
   $RCSfile: OD.doc $
Description: Documentation for the Oberon-A module definition utility.

 Created by: fjc (Frank Copeland)
  $Revision: 1.2 $
    $Author: fjc $
      $Date: 1995/01/25 23:37:04 $
```

1.2 What is OD?

OD is the Oberon-A module definition utility. Its purpose is to create a summary of the objects exported by a module, to act as a reference for programmers. It is similar in most ways to the Oberon System's 'browser' utility.

The definition file created by OD closely resembles an Oberon-2 module containing only declarations and procedure headings. It is produced directly from a module's symbol file, and contains only those declarations exported by the module and visible to its clients. It is

structured roughly as follows:

```
DEFINITION <module>;

CONST
  <name> = <value>;
  ...

TYPE
  <name> = <type>;
  ...

VAR
  <name> : <type>;
  ...

<procedure heading>
...

END <module>.
```

Within each division, identifiers are listed alphabetically. It is not possible to reproduce the structure of the original module.

Type-bound procedures and library call procedures are shown as part of the declaration of the associated record type.

1.3 Copyright and distribution

OD is part of Oberon-A and is:

See Oberon-A.doc for its conditions of use and distribution.

1.4 What do I need to run OD?

OD requires Release 2.04 (V37) of the Amiga operating system, or a later version.

1.5 Running OD from the CLI

```
Format:        OD <files> [TO <directory>] [MAKEICONS]
               [EXTERNAL] [SIZE] [EXPAND]

Template:      FILE/A,MAKEICONS/S,TO/K,
               EXTERNAL/S,SIZE/S,EXPAND/S
```

```
Purpose:       Generates definition files from symbol files.

Path:          OBERON-A:OD

Specification: OD reads the symbol file specified in the FILE
  parameter and generates a definition file which is output in the
  directory specified in the TO parameter, or in the current directory
  if there is none. You can specify several symbol files to be
  processed by giving an AmigaDOS pattern as the FILE argument.

OD uses the standard AmigaDOS pattern matching routines, so the FILE
arguments must fully specify the symbol file name, including the
extension. If a TO parameter is given, it must be the name of an
existing directory.

If the EXTERNAL argument is given, additional information about
external type and procedure declarations is output. If the SIZE
argument is given the size of all objects is output. If the EXPAND
argument is given extended types are output with the fields and
type-bound procedures of the base type(s) are output.

If the MAKEICONS argument is given, an icon is created for the
definition file. The default icon is found in "ENV:OD/def_file.info".

OD requires a stack of at least 10000 bytes. See the AmigaDOS manual
entry for the STACK command.
```

1.6 Running OD from the Workbench

```
See Running OD from the Shell for a general description of OD's
operation and the effect of the various arguments.

OD can be run in two ways:

1. Use extended selection to select one or more symbol files. Then hold
   down the shift key and double-click on the OD icon.

2. Select the icon for the drawer in which the symbol file(s) are
   stored, then double-click the OD icon. In this case the pattern for
   the name(s) of the symbol file(s) to be processed must be specified
   in a tooltype as "FILE=<pattern>".

All arguments available when running OD from the Shell can be specified
as tooltypes in OD's icon. The tooltypes can be edited by clicking the
icon and selecting the "Information" item from the Workbench "Icons"
menu.

For switch arguments like MAKEICONS the name of the argument is entered
as a tooltype. Keyword arguments like FILE are entered as
"FILE=<argument>". The standard WINDOW tooltype is also understood by
OD. If it is omitted a default console window is opened.

A typical list of tooltypes might look like this:

  WINDOW=CON:0/0/640/200/OD'ing.../CLOSE/WAIT
```

```
FILE=#?.sym
EXTERNAL
(SIZE)
(EXPAND)
```

Enclosing the SIZE argument in parentheses disables it without the need to delete the entire tooltype. To enable it, edit the tooltype to remove the parentheses and save the icon.

Set the default stack to at least 10000 bytes.

1.7 Running OD from the FPE utility

A tool button in the FPE window can be configured to run OD (see FPE.doc). In the button editor, set the Command field to the full path name of the OD program. Set the Arguments field to "Code/!M.Sym", or wherever else the module's symbol file may be found. If you wish the definition file to go somewhere else than the current directory, add the destination to the Arguments field. Specify a console window as the Console field. Put at least 10000 in the stack field.

For example:

```
Command="DH1:Oberon-A/OD"
Arguments="Code/!M.Sym TO OBERON-A:Defs"
Console="CON:0/11/540/189/OD'ing !M.../CLOSE/WAIT"
Stack=10000
```

To create a definition file:

1. select the module in the Module gadget.
2. click on the tool button OD is bound to.
3. sit back and relax for a bit.

1.8 Reporting bugs and suggestions

You are encouraged to report any and all bugs you find, as well as any comments or suggestions for improvements you may have.

Before reporting a suspected bug, check the file ToDo.doc to see if it has already been noted. If it is a new insect, clearly describe its behaviour including the actions necessary to make it repeatable. Indicate in your report which version of OD you are using. Include an example of a definition file or symbol file that demonstrates the bug.