COLLABORATORS

	TITLE : Dignet	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		April 16, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Dignet **1**

1.1 Dignet library 1

1.2 Introduction 2

1.3 Important notes 3

1.4 Requirements 5

1.5 Resource Tracking 6

1.6 Library Functions 7

1.7 Tools 9

1.8 Author addresses 13

Chapter 1

Dignet

1.1 Dignet library

```
                ============================================================================

                           DIGNET LIBRARY GUIDE

                Author is Kenneth "Kenny" Nilsen (kenny@bgnett.no)
 ============================================================================

    >  THE easy(tm) way to serial networking with games and programs!  <



                 INTRODUCTION
                    Background



                 IMPORTANT
                    Copyrights, disclaimer and more



                 REQUIREMENTS
                    What you need to use this library



                 RESOURCE TRACKING
                    About the powerful feature



                 FUNCTIONS
                    Short descriptions of functions



                 TOOLS
```

Usage of the tools in archive

AUTHOR
Contact addresses

==========

1.2 Introduction

==========

DIGNET LIBRARY INTRODUCTION

==========

PREFACE

Always wanted to make programs or games that you could use via a nullmodem cable or modem (or even internet) ?

Now you can in a pretty comfortable way.

This library is developed based on the need of such a feature. Most programmers doesn't want to get into device handling and that is understandable since device handling is complicated and frustrating.
There are many aspects to think about, timing, cleaning up and so forth. Code that handle devices often get "spagetthi" to use that frase. The more the spagetti code get the harder it is to maintain it and locate bugs.

We made this library with that in mind. We would do the work once and take benifit of that later without going through the whole process again. We released this library since we also believe that others would like to have benefit of it and we didn;t release it to earn any money (it's freeware). We would be glad to give something useful to the Amiga community and to let programmers use more unused aspects in programming such as games- and programnetworks.

We intented to make the library safe to use. There are no hacks in it and we consider it to be more than good enough for serious programs that needs a reliable network system. You don't need to be an expert in programming either. If you know how to open/read/write dos files you can use this library.

DEVELOPER

You'll find functions to send and receive bytes, strings and binary datas. You find modem dial up functions and modem receive functions so you can actually create your own BBS system in a

very simple way. You find converter routines, parameter
functions and more.

Some of the most important features in this library are:

-
        resource tracking
         (!)
- own functions for dialup/receive modem calls
- string/text/binary send/receive functions
- converter routines for PC <-> Amiga texts, connect strings ++
- painless to use, straight forward.
- possibility for more advanced uses (eg. with signal/Wait())
- fully re-entrant (more connections open at the same time)
- ready-for-use when allocating a net
- easy to change default parameters
- fast routines written in assembler.
- safe to call functions even with faulty arguments
  (such as read/write no bytes, freeing net twice etc.).
  (Please tokenize this as beeing not that sensitive to errors).

        Read more about resource tracking
        .

The library is even freeware so you can use it with no cost. You
can bundle the library file with your programs, even if you are a
shareware or commercial author (
        read details
).

The basic library functions are build around the same call three
as the dos.library. That means a handler, buffer and a size. The
function does the rest for you.

Take a look at the autodoc for more information about each
function.

If you create a game or program using this library, please notify
me so I can have a look (just of curriousity :).

INTERNET:

You can't use this library directly against Internet (via TCP/IP
stacks) unless there will be available a serial.device compatible
device that works agains such. Please
        let me know
         if you know
something like that is in progress!

==========

1.3 Important notes

==========

DIGNET LIBRARY IMPORTANT NOTES

==========

DISCLAIMER

You use this library and the programs in the archive entirly at your own risk. We are not responsible for any damage on your software nor your hardware. You should read the documentation confirmly before you use any
functions
in this library or
programs using this library.

LIBRARY BUNDLING

- Freeware authors

You are free to bundle the library file alone in your archive as long as you provide a visable (to the user) credit line saying something like "Using dignet.library by Kenneth "Kenny" Nilsen (Kenny@bgnett.no)". If you don't want to have such a credit in your program you must bundle the whole archive as it came from the Aminet or not bundle at all.

- Shareware/commecial authors

You must include a credit line (see above) under any circumstances if you use this library in your program(s). You are free to bundle the library file itself too in your archive. If the library has more than 20 percent of functionality in your program (as used by user) you are not allowed to bundle this library. You can however buy yourself free from these regulations. Take
contact
for details.

DISTRIBUTION

This archive is distributed as FREEWARE on these conditions:

- The archive must be intact (as it came from Aminet or homepage)
- There can not be charged any fee for this archive or the programs or files in the archive. Only resonable fee to cover media expences.

Aminet, Fred Fish, Meeting Pearls and BBS'es are allowed to distribute this archive and its files.

BUGS

```
There are bugs in every software. We can't gurantee that this
software is bugfree, but we will try to make it as bugfree as
possible. To do so
        we need your help
        . If you discover strange
behaviour or bugs please contact us and describe what you did to
get the behaviour or the bug appearance.

We would also like to hear from you what you think about the
library, if you have ideas, suggestions, comments or critisism.

=============================================================================
```

1.4 Requirements

```
                =============================================================================

                        DIGNET LIBRARY REQUIREMENTS
=============================================================================


  You would need a computer with:


          - MC68000 or better

          - Kickstart version 37 or later

          - A device that is serial.device compatible (or the
            serial.device itself of course)


  Optional:
 ---------------------------------------------------------------------

  - Modem

  - Nullmodem cable


  Suggestions:
 ---------------------------------------------------------------------

  The AllocNet() uses the default system configuration. We would
  recommend that to be:

  - No parity
  - 8 bits
  - 1 stopbit
  - Atleast 38400 baud (020 -> 030/25)
  - RTS/CTS handshaking (don't use XON/XOFF)
  - BOGGIE flag set (highspeed modems -> 14k4+)
  - 8 KB buffers (I/O)
```

```
   If you don't know how to set this configuration you could use the
   function SetDefault() which sets the above datas to the allocated
   net (device).


   Tested with:
 ------------------------------------------------------------------

   - serial.device
   - duart.device (MFC III)
   - nullmodem.device

   - modem send/receive
   - nullmodem cable (MFC port 0 <-> 1)

   Betatesters needed! I need someone to test all functions. See

           contact page
           .


   Developed on an A4000/030FPU/25 with 26 MB ram using
   duart.device and MultiFaceCard III.


==========================================================================
```

1.5 Resource Tracking

```
                ============================================================================

                    DIGNET LIBRARY RESOURCE TRACKING
==========================================================================


   Very few programs, almost none, comes with built-in resource
   tracking. dignet.library does.

   Resource tracking means you can track back resources and free
   them if a program forget to free them it self or it gets
   suspended, removed or crashes.

   The library keep tracks of all resources such as device name,
   unit, io structures, messageports and more. If a program get in
   one of the situations mentioned above and the system doesn't go
   down, you can track back the resources and free them yourself.

   If you have a serial.device unit 0 used and the program that
   opened it goes down you would normally not be able to use the
   device again unless if you reset or hack pointers. If the latter
   the memory used is still unavailable.

   The built-in resource tracking doesn't hack anything. It uses a
   special structure that keeps all needed pointers and then checks
   for the tasks status and free the resources allocated by that
```

task. You should, after freeing the resources, be able to use the device again.

(Even the functions not using the net structure is tracked).

Note that resource tracking isn't always successful. If something went wrong using the device you could experience that the system crash. If a program still lives and you free its resources you are almost guaranteed that your system will crash!

The resource tracking only uses two assembler instructions for each function call. In practice you will not be able to notice the speed difference at all.

Always be careful when using the debug tools. You use them at your own risk and as mentioned, you are not guranteed success.

Read about the
usage of the tools
.

==========

1.6 Library Functions

==========
DIGNET LIBRARY FUNCTIONS
==========

Here you will find a list of all the functions in the library and a short description of what they do. They are sorted based on the FD file.

From version 1:

AllocNet - allocates a net structure, opens the device and initialize so everything is ready to use if it return successfully

FreeNet - Frees the above structure.

ReadNet - Reads x chars from the device.

WriteNet - Writes x bytes to the device

QueryNet - Checks how many chars are available for reading in the device buffer

AbortNet - Stop all read/writes/++ in the device

InitIOEXT - Set new device parameters

```
  ReadIOEXT               - Reads the device parameters into a
                            buffer

  SendNet                 - Still not in use (protocol send ala
                            Zmodem)

  ReceiveNet              - Still not in use (protocol receive)

  Timeout                 - Give device x seconds to get some chars
                            to read

  From version 2
-------------------------------------------------------------------

  ReadString              - Read a nullterminated string from
                            device

  WriteString             - Write a null terminated string to
                            device

  GetNetport              - For advanced use. Pointer to device
                            message port

  From version 3
-------------------------------------------------------------------

  InitModem               - Sends a init string to a modem. The
                            init string can be in C style
                            (for example "ATZ/r")

  CallModemDT             - Dials a phone number and return connect
                            speed if ok. DT = Dial Touchtone

  CallModemDP             - Same as above, just with Dial Pulse

  WaitModem               - Wait for a modem to call. Handles
                            none-modem calls too. Returns with
                            connect speed

  HangupModem             - Sends a hangup command to the modem

  StopWaiting             - Aborts the WaitModem() since
                            WaitModem() nevner returns unless it
                            get a RING signal.

  FlushNet                - Flushes (empties) the IO buffers.
                            Built-in NIL read to empty the buffers
                            for sure.

  SetBaud                 - Sets internal baud rate for use.

  ConvertModemString      - Convert a, for example init string to
                            binary format (ATZ\r -> ATZ(13))

  GetIOmsg                - For advanced use. Returns pointer to
                            the message structure used by the
```

device.

GetModemStatus - Assumes the modem is on- or offline. (currently for Call/WaitModem only)

GetCurrentDevice - ptr. to device name in use

GetCurrentunit - unit number of device in use

GetBaudrate - Get current internal baudrate

ParseConnect - Converts CONNECT string to baudrate (f.ex "CONNECT 28800/ARQ" -> value 28800)

SetDefault - Init device with a good default setup.

ConvertToPC - Converts a text buffer to PC format (LineFeed + CarriageReturn)

ConvertToAmiga - Converts a text buffer to Amiga format (strips CarriageReturn)

FreeConvert - Free the buffer returned by one of the two above functions.

ObtainNet - A function another task can call to obtain a net structure allocated from this task (or vica verca).

RTNumberofNets - Returns number of nets in use

RTNetInfo - Copies valid information about a task for debug reasons.

RTKillNet - Kills a net owned by a dead task

RTAbortNet - Aborts a net. For global use.

RTFlushNet - Flushes a net. For global use.

RTCheck - Checks the resource list if it's valid. If invalid you should be careful using the debug tools/functions.

RTHighestID - returns the highest net ID used.

MORE TO COME!

==========

1.7 Tools

==========

DIGNET LIBRARY TOOLS

==========

In this section you will find descriptions and usage of the tools provided in this archive.

Most are for debug purposes.

DignetTERM

USAGE: <device> [unit]

Device: device name, f.ex. "serial.device"
Unit: optional unit number. Defaults to 0 if not provided.

This little terminal program shows you some of the functions that dignet.library provides for your programs. The DignetTERM uses several functions of the library. The terminal program should work fine, but it has no built-in charmap converters (eg. IBN to ISO) and such. It's just a simple demo program. You will find source in assembler for this program. The source and program is public domain. It sends a ATZ string to the modem at startup.

DignetLIST

USAGE: DignetLIST

List nets in use. That means devices that are in use. It uses the

resource track
list in the library to give additional information such as task pointer and name on the task using the device, pointer to the net structure and so on. I can however logically not list devices that aren't allocated with this library.

At the end of a list line you'll get an OK or ERROR. If OK the task is in the tasklist of the system. If ERROR the task is gone.

First on the line is a number. This number is used with some of the other debug tools. DignetLIST is for information only.

DignetKILL

USAGE: DignetKILL <number from DignetLIST>

If you find a net number with ERROR in DignetLIST you could use this program to free the resources used by that program (note: resources used by dignet.library such as device, io structures, message port etc.).

What it does is to obtain the net structure from the resource track list of that task. It then calls the AbortNet(), FlushNet() and finally FreeNet().

If it returns ok you should now be able to use the device again and the memory allocated with the net is freed. If your system crash it means a more serious bug have occured.

DignetDEBUG

USAGE: DignetDEBUG <number from DignetLIST>

This tool can be used to obtain more information about a net. If net is in the resourcelist you get the following output (for example):

```
        Net number: 1
       Net address: $7485D28
              Task: $75F6DA8
         Taskname: DignetTERM
           Device: serial.device
             Unit: 0
   Call returns to: $761251A
Device messageport: $7485F90
 Device IO message: $7487F68
Last function used: QueryNet()
       Task status: -1  (-1 = OK)
```

Netnumber is the net's ID in the resource list. You can have a maximum of 1024 nets open (that is 1024 different devices/units).

Net address is the pointer to the net structure allocated with AllocNet().

Task is pointer to the task that allocated the net stucture, eg. the task that use the device.

Taskname is the name of that task.

Device is the name of the device used by the net/task.

Unit is unit number used of the device.

Call returns to. This is the pointer to where the task returned to after calling the last function (see below).

Device messagesport is pointing at the port which the device uses.

Device IO message is the pointer to the message structure (including IOEXT structure) that the device uses.

Last function used: is the name of the function the program used last time it called the dignet.library. With this line and the above you have a powerful debug tool. If your program crashes you can easily disassemble memory from Call returns to pointer and check your source for the function name. The error should be nearby.

Task status indicates if task still exist in the tasklist or not. TRUE (-1) if the task is in the systems tasklist else FALSE (0).

DignetABORT

USAGE: DignetABORT <number from DignetLIST>

This command will send an abort command to the net. Useful if a program locks up waiting for a char to be read that never will arrive. Using this program might help since the Read..() will be aborted and return. You might have to use this program several times in sequence (or remove task and kill net).

DignetLIB

USAGE: DignetLIB [+|-]

If your program crashes and leave the dignet.library open, you could use this program to increase or more likely decrease the library open counter. When increasing it opens the library once and exits. When decreasing the library it opens it and closes it twice. The last methode require that you know what you’re doing.

Useful if you want to flush the library.

Flushing libraries

If you use a flush command on the libraries in your system you will notice that when dignet.library get a Expunge() function call activated it will pop up a requester if there are resources left in the system. If this happands you should use the DignetLIST and DignetKILL commands to free the resources. You can of course choose to leave them, but after a Expunge()/Flush of the library there are no leagl way to obtain the resources anymore. After a cleanup or a decission you press OK to continue the Expunge().

```
        MORE DEBUG TOOLS TO COME

 ========================================================================
```

1.8 Author addresses

```
                 ========================================================================

                             DIGNET LIBRARY AUTHOR
 ========================================================================


        The dignet.library is made by Kenneth "Kenny" Nilsen.
        Copyright (C) 1997 by Digital Surface.

        This program is FREEWARE.
                Read about the details
                .

        Contact author:

        - if bugs are found
        - if you have ideas for functions
        - if you wonder about the usage of the functions
        - if you have other comments (author like nice-job type of
          comments..  helps motivation ;)
        - suggestions to improve the library/speed etc.


        ADDRESSES
      ------------------------------------------------------------------

        Kenneth "Kenny" Nilsen
        Briskåsen 1
        N-3535 Krøderen
        (Norway)

        EMAIL: kenny@bgnett.no

        URL: http://www.bgnett.no/~kenny/

        You will also find many other programs by me on this page with
        links to Aminet.

        This page is also the official support page of IXODEV. What is
        IXODEV ?  A developer and GUI system for the Amiga. Visit the
        page and read all about it..


        Thanks for taking time to look at the library and this guide
        file!
```

==========