TrapDispatcher

COLLABORATORS			
	TITLE : TrapDispatcher		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		November 2, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

TrapDispatcher

1.1 TrapDispatcher.guide

TrapDispatcher 1.1

1.2 Introduction

Introduction
============

As you probably know, TrapDoor's built-in functions for calling nodes aren't very good, so until recently I was using one of all the ARexx scripts that exist. But that wasn't good enough for me; I wanted to be able to add more nodes to call when the script was already running, and I wanted to be able to call a node until a specific time. TrapDispatcher can do this and more!

TrapDispatcher is a program for calling nodes via TrapDoor. It has a nice, resizable GUI and an ARexx port. You can also install external programs in a menu.

TrapDispatcher requires a nodelist compiled with TrapList, a registered version of TrapDoor and reqtools.library version 38 or higher by Nico François.

1.3 Disclaimer and License Agreement

Disclaimer and License Agreement
================================
I do not guarantee that this stuff will work, and if something goes wrong... Well, don’t blame me! There is no guarantee at all. Use it at your own risk!

The program is released as FreeWare. You may use it, give it to your friends, upload it to any BBS, but you may NOT SELL IT FOR PROFIT! This is what I want, so please respect it!

1.4 Configuration

Configuration
=============
The default name of the settings file is MAIL:TrapDispatcher.prefs, but you can tell TrapDispatcher to load another file by specifying the file either in the CLI or in the tooltypes with the SETTINGS keyword.

Keywords in the Settings file
=============================

Window <xpos> <ypos> <width> <height>

With this command, you can specify the position and the size of the window. Since the window size can be saved from within TrapDispatcher, you won’t need to change this.

DelayCall <time in seconds>

This is the time TrapDispatcher will wait between the calls.

DelayAll <time in seconds>

This is the time TrapDispatcher will wait after it has called the last node in the list.

DefaultNode <node>

When you add a call, manually or via the ARexx port, it will be completed using this node if it is uncomplete. Here are a few examples of what would happen if the default node was 2:200/207.0:

Entered string		Result
.6	-->	2:200/207.6
230	-->	2:200/230.0
203/600	-->	2:203/600
.	-->	2:200/207.0

Please set the default node to a node number, not to a point. If you set it to a point, most of the examples won’t work...

DefaultRetries <retries>

This is the default number of retries TrapDispatcher will use when a node is added using the GUI, or when a node is added via the ARexx port with no limit is specified.

MaxFailed <max>

If a call to a node has failed more times than specified here, the call will be removed from the list. This was added so that wrong passwords, unreliable modems etc can’t give you a high phonebill...

DispatchPort <name>

The name of TrapDispatcher’s ARexx port. Please remember that portnames are case-sensitive.

TrapPort <name>

The name of TrapDoor’s ARexx port. Port names are case-sensitive.

PubScreen <name>

This is the name of the public screen TrapDispatcher should try to open its window on.

StartTrapDoor <command>

If TrapDoor isn’t running, TrapDispatcher can try to start it. In most cases, this will be set to "mail:TrapDoor answer". There is no need to specify "Run", since all commands executed by TrapDispatcher are run in the background.

Nodelist <dir>

The directory where TrapDispatcher can find a nodelist compiled with TrapList. A nodelist is always needed!

InactiveWindow

If this is specified, TrapDispatcher’s window is opened in inactive state. This could e.g. be useful if you want to run TrapDispatcher in WBStartup and want to keep the Workbench active.

QuitTrapDoor

If this is specified, TrapDispatcher will try to quit TrapDoor as well when you quit TrapDispatcher.

External <title> <command>

With this keyword, you can add items to TrapDispatchers menu for external commands. <Title> is the string that will be shown in the menu and <command> is the command that will be executed.

If you want to have a hotkey for the item, enter a "@" and the hotkey first in the title. If you enter a "@" first in the command, the command will not be executed , the string will be sent to TrapDoor’s ARexx port instead.

A few examples:

```
External "@AAnswer" "@answer immediate"
External "@BBreak" "@@abort cde"
External "@VView log" "MultiView MAIL:TrapDoor.log PUBSCREEN TRAPDOOR"
External "@SScanOut" "ScanOut >CON:/20//200/ScanOut/AUTO/WAIT/SCREENTRAPDOOR"
External "@LLocal login" "BBS:bbs local"
```

1.5 Usage

Usage
=====

TrapDispatcher can be used in several ways, and here are the two I thought of when designing the program:

- You can use TrapDispatcher to get a small GUI for TrapDoor and to be able to call nodes in a comfortable way. You can even put TrapDispatcher in you WBStartup and let it start TrapDoor.

- You can use TrapDispatcher instead of "TrapDoor call x:y/z". TrapDispatcher can dial more than one node and calls can be added when TrapDispatcher is running.

Window

Command-line

Workbench tooltypes

ARexx port

Example usage

Advanced node usage

1.6 Window

Window
======

Gadgets

With the Active/Inactive cyclegadget you can toggle TrapDispatcher’s state. When inactive, TrapDispatcher will not try to call any nodes.

If you click on a node in the list, information about it will be shown in the box below the list.

Node: Node to call.

Retries: The number of retries that already have been made

Last: Last response from TrapDoor

Remove: When TrapDispatcher will remove this call from the list even if it could not be done successfully.

There are also three buttons in the window:

Add... Add a node to the list of nodes to call. If the node you enter is incomplete, it will be completed using the configured default node. A few examples with the default node 2:200/207.0:

Entered string		Result
.6	-->	2:200/207.6
230	-->	2:200/230.0
203/600	-->	2:203/600
.	-->	2:200/207.0

Remove Remove the selected node

Call now Stop the delay countdown and make the next call immediately

Menus

The Project menu

About

Shows information about TrapDispatcher

Quit

Quits TrapDispatcher. If TrapDispatcher is currently calling a node it can’t quit immediately and will wait until the call has been made.

The Settings menu

Quit TrapDoor

If this is selected, TrapDispatcher will try to quit TrapDoor as well when it quits.

Quit after last

If this is selected, TrapDispatcher will quit when the last call has been made, or when the limit for the last call has been reached.

Zone Mail Hour

If this is selected, TrapDispatcher assumes that it is ZMH. This only makes a difference if the node isn't marked as CM (continuous mail) in the nodelist, non-CM nodes can only be called during ZMH. When the ZMH flag is removed, all nodes without CM are removed from the list.

Save window position

TrapDispatcher will save the current window position and size to the settings file.

External menu

In this menu, you can put commands of your own choice. See the External configuration keyword for more information.

1.7 Command-line switches

```
Command-line switches
=====================

CALL/M,RETRIES/K/N,UNTIL/K,TIME/K/N,FORCE/S,REQ/S,ZMH/S,NOZMH/S,
QUITAFTERLAST/S,QUITTRAPDOOR/S,KEEPTRAPDOOR/S,INACTIVEWINDOW/S,SETTINGS/K

CALL                Here you can enter one or several nodes that
                    TrapDispatcher should try to call.

RETRIES             The number of times TrapDispatcher should try to call
                    the nodes specified with CALL.

UNTIL               The time (hh:mm) when TrapDispatcher should stop trying
                    to call the nodes specified with CALL. Example: 10:15

TIME                The number of minutes TrapDispatcher should try to call
                    the nodes specified with CALL.

FORCE               TrapDispatcher will call the node specified on
                    command-line even if it isn't marked as CM in the
                    nodelist and normally shouldn't be called at this time.
```

```
REQ                 If the node specified on command-line can't be called
                    for some reason, TrapDispatcher will inform you in a
                    requester.

QUITAFTERLAST       When the last call has been carried out, or when the
                    limit for the last call has been reached, TrapDispatcher
                    will quit.

QUITTRAPDOOR        When TrapDispatcher quits, it will try to quit TrapDoor as well ↩
    .
                    This overrides what has been specified in the settings file.

KEEPTRAPDOOR        When TrapDispatcher quits, it will leave TrapDoor running.
                    This overrides what has been specified in the settings file.

INACTIVEWINDOW      Make TrapDispatcher open its windows in inactive state. This
                    overrides what has been was specified in the settings file.

SETTINGS            With this keyword, you can specify an alternative
                    settings file that TrapDispatcher should use instead of
                    "MAIL:TrapDispatcher.prefs".
```

Note: If TrapDispatcher is started with commands like "call x:y/z retries 10 quitafterlast" and the program is already running, the commands will be passed on to the running copy of TrapDispatcher and the call will be carried out.

1.8 Workbench tooltypes

Tooltypes
=========

There is only one tooltype: SETTINGS. You can use it to specify a settings file that TrapDispatcher should use instead of "MAIL:TrapDispatcher.prefs".

1.9 Arexx port

ARexx port
==========

TrapDispatcher strips all leading "@" from the command and uses CALL just like TrapDoor, so tools like TrapCall and TrapPoll work perfectly with TrapDispatcher if you specify the correct ARexx port.

```
QUIT
```

This will quit TrapDispatcher. If the program is currently calling a node it can't quit immediately, and will wait until the call has been made.

```
CALL <node> [TIME <mins>/RETRIES <retries>/UNTIL <hh:mm>] [REQ] [FORCE]
```

This will add a node to TrapDispatcher's list of nodes to call.

TIME/RETRIES/UNTIL With TIME you can specify the number of minutes TrapDispatcher should try to call the node, with RETRIES the maximum number of retries and with UNTIL a time when TrapDispatcher should stop trying to call this node. You can only specify one of TIME, RETRIES or UNTIL. If no limit is specified, TrapDispatcher will use the default number of retries specified in the settings file.

FORCE TrapDispatcher will call the node even if it isn’t marked as CM in the nodelist and normally shouldn’t be called at this time.

REQ If the node can’t be called for some reason, TrapDispatcher will inform you in a requester.

Errors: 10 Invalid node number, node not found in nodelist or non-CM node outside of ZMH.
5 The node, or another node with the same telephone number, is already in the list.

GETERROR

This command gets the result of the last CALL command as a brief string describing the error. You can find the string in the "result" variable. Make sure that you have "options results" in your ARexx script.

ZMH

Tells TrapDispatcher that is is Zone Mail Hour. Nodes without the flag CM (continuous mail) in the nodelist can only be called during ZMH. Put this command in the script you normally run when ZMH starts. Then it might be a good idea to use Deliver.rexx to send all crashmail...

NOZMH

Tells TrapDispatcher that ZMH is over. All nodes without CM are removed from the list of nodes to call.

REMOVE <node>

Remove a node from the list. If TrapDispatcher is currently calling the node, it can’t be removed until after the call has been made.

REMOVEALL

Remove all the nodes from the list. The node TrapDispatcher is currently calling can’t be remove immediately, but will be removed later.

```
QUITTRAPDOOR

When TrapDispatcher quits, it will quit TrapDoor as well. You can use this
to override the settings.


KEEPTRAPDOOR

When TrapDispatcher quits, it will leave TrapDoor running. You can use this
to override the settings.


INACTIVE

Makes TrapDispatcher inactive. When the program is inactive, it won’t make any
calls.


ACTIVE

Makes TrapDispatcher active again.


QUITAFTERLAST

When the last call has been made, or when the limit for the last call
has been reached, TrapDispatcher will quit.


COMMANDAFTERLAST <command>

When the last call has been carrier out or when limit for the last call
has been reached, TrapDispatcher will execute this command. If the first
character is a "@", the command will not be executed, the string will be
sent to TrapDoors ARexx port instead.
```

1.10 Example usage

```
Replacement for "TrapDoor call x:y/z"
====================================
If you want to use TrapDispatcher this way, make sure that it is configured
to start TrapDoor. Use this command in the CLI:

   TrapDispatcher call <node> quitafterlast

To make it really comfortable, you can use an alias. Put this line in your
shell-startup:

   Alias Call MAIL:TrapDispatcher call [] quitafterlast


As a GUI for TrapDoor
=====================
Make sure that TrapDispatcher starts TrapDoor in a correct way, then just
run this program instead of TrapDoor. You can even put it in your WBStartup
```

drawer if you want to!

Poll a node via ARexx port
==========================

Your script might look like this:

```
/* TrapDispatcher call */

address ’TRAPDISPATCHER’ call ’2:666/4711’
```

And if you don’t want TrapDoor to answer any incoming calls when TrapDispatcher is calling, use this:

```
/* Call without answering incoming calls */

address ’TrapDoor’ rings 9999
address ’TRAPDISPATCHER’ call ’2:666/4711’
address ’TRAPDISPATCHER’ ’commandafterlast "@rings 1"’
```

Don’t forget that you can specify a time limit for the calls, which is very useful if you only want to call a node as long as the phone rates still are cheap.

TrapPoll
========

TrapDispatcher strips all leading "@" from ARexx commands and uses CALL just like TrapDoor so it is possible to use TrapPoll with TrapDispatcher. This is a very easy way to deliver all files in your outbound directory.

All you have to do, is to make sure TrapDispatcher is running and run the following command:

```
   TrapPoll +N +C +D -pTRAPDISPATCHER
```

This will deliver all normal, direct and priority (crash) mail.

If you have mail for two AKAs of the same node, the node will not be called twice since TrapDispatcher refuses to have two nodes with the same telephone number in the list.

Deliver.rexx
============

This is basically the same as TrapPoll, but allows you to specify the number of the retries, the time TrapDispatcher should stop at or for how long TrapDispatcher should try to deliver the mail...

```
rx Deliver.rexx +N +C +D UNTIL 04:30
```

This will try to deliver all normal, direct and priority (crash) mail until 04:30.

1.11 Advanced node usage

Advanced node usage

In this section, a typical node usage of TrapDispatcher is described. Of course you may want to use the program differently...

Normally, a node will have TrapDoor running all the time and then you should also keep TrapDispatcher running.

After export

After mail has been exported from your messagebase, it may be wise to deliver all crashmail immediately. Execute this line after your mail has been exported to try to send all crashmail for 60 minutes. You can of course also receive calls when TrapDispatcher is delaying between calls, so users will be able to call your BBS. TrapDispatcher will only call nodes that accept mail all day (has the CM flag in the nodelist) unless it is in ZMH mode.

```
rx mail:Rexx/Deliver.rexx +C TIME 60
```

Polling your uplink

In order to have automatic polling, you must use a "cron"-program or a similar utility that can start commands at different times. This is a command that you can use:

```
rx "address ’TRAPDISPATCHER’ call ’2:200/207’ time 60"
```

The "’" are very important, without them the "rx" command will fail. Of you only want to call if the node has mail waiting, you may use this:

```
rx mail:Rexx/CallExist.rexx 2:200/207 time 60
```

ZMH starts

Here you will normally send a "BBSMODE ZMH" to TrapDoor and also disable file-requests etc. I suggest you to also add these two lines two the scriptfile you run when ZMH starts. This will send all crashmail to nodes that don’t accept mail all day.

```
rx "address ’TRAPDISPATCHER’ zmh"
rx mail:Rexx/Deliver.rexx +C TIME 60
```

ZMH ends

Put the following line in the scriptfile you run when ZMH ends. All calls to nodes that only accept mail during ZMH will be removed.

```
rx "address ’TRAPDISPATCHER’ nozmh"
```

1.12 Acknowledgements

Acknowledgements

I wish to thank the following persons:

Johannes Nilsson for beta-testing.

Maximilian Hantsch and Martin Laubach for TrapDoor.

Matt Dillon for providing the Amiga community with the cheap and good C-compiler I compiled TrapDispatcher with: DICE!

1.13 How to contact the author

```
How to contact the author
===========================
FidoNet:   Johan Billing@2:200/207.6
e-mail:    johan.billing@kcc.ct.se

SnailMail: Johan Billing
           Östra Storgatan 22
           260 60 Kvidinge

(FidoNet/e-mail preferred)
```

1.14 Program History

```
History
=======

1.0 - First release

1.1 - The ARexx command CALL would return random result codes when it
      was successful.

    - Incomplete node numbers will now be completed using the default
      node. See configuration and usage for more details.

    - TrapDispatcher will now start without an configuration file
      and then use the default values.

    - TrapDispatcher caused Enforcerhits when there was an error
      in the configuration file.

    - New configuration keyword: "MaxFailed". If a call to a node
      has failed more times than allowed, the call will be removed
      from the list. This was added so that wrong passwords,
      unreliable modems etc can't give you a high phonebill...

    - You can now control the listview using the cursor keys.

    - Two new switches for the ARexx command "CALL" and for the
      command-line: REQ and FORCE

    - REQ is now used in the ScanPoll.rexx script. If an error
```

occurs, a requester will appear.

- New ARexx command: GETERROR. This gets the result of the last CALL command as a brief string describing the error. You can find the string in the "result" variable. Make sure that you have "options results" in your ARexx script.

- Three new useful ARexx scripts are included: CallExist.rexx, Deliver.rexx and IntuiNode.rexx.

- Support for Zone Mail Hour added. Now TrapDispatcher will only call non-CM nodes during ZMH. ZMH can be toggled either using ARexx or manually using the menus.

1.15 Future plans

I am currently thinking of implementing the following features:

o Support for nodes running multiple lines. That is quite a lot of work and I am not quite sure how many Amigas there are that run multiline...

o Adding FrontDoor-like features like View outbound, Send files, Request files and configuration of passwords. Please write me and tell me what you think. Should those parts be separate programs or included in TrapDispatcher?