AssignList

Markus Adamski

COLLABORATORS

	TITLE : AssignList		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Markus Adamski	July 20, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

AssignList

1.1 AssignList - Overview

```
           AssignList

           Define assigns via GUI

     Copyright © 1996 by Markus Adamski

          - Freeware -
```

1.2 About AssignList

Use AssignList to execute as many assigns as you want at boot-time by one single call.

1.3 Requirements

AssignList requires at least OS 2.04 or higher and MUI version 3.1 or higher if the GUI should be used.

To call AssignList at boot-time only dos.library version 37 or higher is required.

Any 680x0 cpu is supported.

1.4 How to install AssignList ?

Just double-click on that installer-icon in the drawer Install in your preferred langnguage.

All files will be copied and your user-startup file will be modified.

1.5 Getting started

There are two ways to start AssignList:

1. From Shell or within your user-startup file:

   - Just type "AssignList" in your shell window or call it from within your user-startup and AssignList will start. Some arguments are supported:

     * HELP or ?
       Prints usage.

     * <Filename>
       The file <filename> will be loaded automatically.

     * <Filename> USE
       The file <filename> will be loaded, all contained assign definitions and afterwards AssignList terminates. For this MUI is not required. Also this is the standard configuration if you chose to modify your user-startup file on installation .

1. From workbench:

   - Simply double-click on the AssignList-icon. You may put AssignList in the WBStartup-drawer as well. Some tooltypes are supported:

     * DONOTWAIT
       If AssignList is installed in the WBStartup-drawer tell the system not to wait for AssignList to terminate.

     * CX_POPKEY=<Keys>

Defines a sequence of keys to pop up the main window
if AssignList was iconified.

* CX_POPUP=<YES|NO>
Open the main window or show an icon on
the workbench screen only.

* FILE=
Load file specified after "=" automatically.

* MAKEICONS=<YES|NO>
Create an icon automatically when saving a definition
file. A double-click on a created icon would start
AssignList and load the attached file.
(See also menu item "Create icon".)

1.6 Main window

The main window has the following structure:

```
  .-----------------------------------------------------.
  |             |
  |             |
  |           List of all definied assigns    |
  |             |
  |             |
  | --------------------------------------------------- |
  |       Assign:                          Mode   |
  |           to:
 assign-modes   |
  | --------------------------------------------------- |
  |    Execute...          Add             Delete   |
  |       Save          Save & End          Quit    |
  | --------------------------------------------------- |
  |                     Current file        |
  .-----------------------------------------------------.
```

If the pointer remains for a couple of seconds over one button a
short help text will pop up.

1.7 List of all definied assigns

Here you find all added definitions.

In the first column you find the assign- mode . Single entries may
be moved around the list and placed where ever you want. One click
on an entry makes it the current. A double-click will copy the entry
into the name , path und mode gadgets below the list.

The second column contains the name associated to the path in the third column.

1.8 Assign name

Type here the name associated to a path .

The name is limited to 31 characters.

1.9 Assign path

Type here the path associated to a namen .

The path is limited to 127 characters.

1.10 Assign mode

Define the type of assign to be used:

Locked = Usual method. A path is attached to a name. An old definition will be overwritten.

Add = Addes a path to an already defined name. With this you can read from different paths by using only one single name. For example:

```
    Assign Locked Test: RAM:
    Assign Add Test: Work:
```

"Dir Test:" would print all files contained in the drawers RAM: and Work: as if they where all in one single drawer.

Though, a file would be saved in the drawer that was assigned locked.

Path = This is a not binding definition, though similar to mode locked. A name will be attached to a path, but with every use of that name, the path will be newly identified. For example:

```
Assign Path FONTS: DF0:FONTS
```

"Dir FONTS:" will print all files in the fonts-drawer of the disk currently in df0: but it will not ask for the initially inserted disk, when the assign was called.

Late = Similar to mode locked, but the associated path will be evaluated on first use.

1.11 Execute a definition

A requester opens to choose one from these options:

1. Input:
   The path will be attached to the name regarding the assign- mode .

2. All:
   The whole list will be executed.

3. Part

4. Abort
   Close requester without any further action.

1.12 Execute partial list

The execute window has following structure:

```
.-------------------------------------------------------.
|         |     All       |
|         |     Mark      |
|  List of temporary assigns    |    Delete     |
|         |---------------|
|         |    Append     |
|         |     Quit      |
.-------------------------------------------------------.
```

Drag items from the list in the main window onto this list by mouse. Proceed as follows:
Click on one item, hold down left mouse button and drag the item off right or left the listview gadget's boundaries. The item will then not appear ghosted any more. Now release the left mouse button when the pointer is over this listview gadget and the item will be added.

Proceed reverse to drag items from this onto the main window's listview gadget.

Attention: In this procedure no testing will be done to verify wether an added name has already been definied or not.

1.13 List of temporary assigns

In opposite to the list of all definied assigns this list, as you already assumed, is only temporary, so it can not be saved directly.

1.14 Execute all temporary assigns

All in the temporary list definied assigns will be executed one after another.

Error handling will be done.

1.15 Execute marked temporary assign

Only the marked assign will be executed.

1.16 Delete marked temporary assign

The mark temporary assign will be removed from the list without any further request.

(See also delete marked definition.)

1.17 Append temporary assign to main list

Appends the temporary list to the main list .

1.18 Delete temporary assigns

The temporary list will be removed from memory and the execute window will be closed.

1.19 Add current definition to list

The definitions of name , path and mode will be added to the list. If the name is already definied, a warning will be displayed.

In addition, this definition will also be added to the temporary assign list automatically. No verifying will be done, wether the name has already been definied or not.

1.20 Delete current definition

The marked definition will be removed from the list with a warning.

1.21 Save list

The list will be saved with the current filename . Is the current filename empty, a file requester will pop up.

1.22 Save and end

The list will be saved with the current filename and the program terminates.

1.23 End program

AssignList will terminate without saving the list. Though, if the list has been modified a requester will pop up.

1.24 Current file

Shows the name, that will be used when the file is saved.

1.25 The menu

The menu-strip contains two items, that will provide you with almost the same functions as the main window.

1. Project:

    * New: The list will be deleted. If it has been modified, a requester will pop up.

    * Load: Loads a definition file. If the current list has been modified, a requester will pop up.

    * Insert: A definition file will be attached to the end of the current list. If names where used twice, a warning will be displayed.

    * Save

    * Save as: See Save except, that a file requester will pop up.

    * About: A copyright information will be displayed.

    * About Project: Informations about the current project will be displayed.

    * About MUI: An information about MUI will be displayed.

```
    *  Quit

2. Options:

    * Create icons: Determines wether an icons will be created
         when a file is saved or not. (See also tooltypes.)

    * MUI: Starts  MUI -preferences.
```

1.26 Definition-file's structure

A definition file may be as well created using a simple text editor.
Every line must be structured as follows:

A line starts with a key word locked, add, path or late.
(See also assign mode .)

Then separated by a "," follows the assign name and again
separated by a "," follows the associated path .

The line must be ended by a LineFeed (CHR$(10)).

Just print a definition file using for example "type".

1.27 To do...

```
    ???
```

1.28 ARexx support

```
Command   Template
===========================================================
 AddItem  MODE/A,NAME/A,PATH/A,FORCE/S
```

1.29 ARexx: AddItem

This adds an item to the main and the temporary list. The parameters
are as follows:

```
MODE=[Locked|Add|Path|Late]

NAME=Name

PATH=Path

FORCE       Force adding.
```

```
Return values:

RC =  0:  Adding was successful.
RC =  5:  If FORCE was submitted:
      Item added to lists successfully, though name
      was already used.
    Else:
      Name was already used and the item has not
      been added.
RC = 10:  MODE was invalid.
```

1.30 Über MUI

This application uses

MUI - MagicUserInterface

(c) Copyright 1993/94 by Stefan Stuntz

MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to register directly, feel free to send

DM 30.- or US$ 20.-

to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY