ClassAction

Martin R. Gasmi

COLLABORATORS

	TITLE : ClassAction	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Martin R. Gasmi	July 31, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 ClassAction **1**

1.1 ClassAction 4.2 Guide . . . 1
1.2 Index . . . 2
1.3 What is ClassAction ? . . . 3
1.4 more about ClassAction . . . 4
1.5 System Requirements . . . 4
1.6 Configuration . . . 5
1.7 Options . . . 6
1.8 Commands . . . 10
1.9 Classes . . . 11
1.10 Using the learn function . . . 12
1.11 What is a Class . . . 13
1.12 Creating a new Class . . . 14
1.13 Defining a new action . . . 16
1.14 Modes . . . 17
1.15 Variables . . . 19
1.16 Future Improvements . . . 21
1.17 About . . . 22
1.18 The Author . . . 23
1.19 Using ClassAction . . . 24
1.20 The Buttons . . . 25
1.21 The Menu . . . 26
1.22 Technicals Infos . . . 27
1.23 Installation . . . 27
1.24 Legal words . . . 28
1.25 History . . . 28
1.26 The Arexx Commands . . . 33
1.27 The archive support . . . 35
1.28 FTPMount . . . 35
1.29 What should you do after install . . . 37
1.30 Message for Philippe THOMAS . . . 38
1.31 Known Bugs . . . 39

Chapter 1

ClassAction

1.1 ClassAction 4.2 Guide

```
                    ClassAction
                    Version 4.2


   For beginners

   o  What is ClassAction    Read this first
   o  System Requirements    What you need
   o  Licence                What you may do

   Installation and Configuration

   o  Installation           How to install ClassAction
   o  After Install          What still has to be done
   o  ClassActionPrefs       How to adapt ClassAction
   o  Options                Changing some parameters
   o  Commands               Creating your own interface
   o  Classes                Configuring classes and actions

   Using ClassAction

   o  General notes          How to use this tool
   o  The Buttons            What they mean and do
   o  The Menu               The menu items

   Additional functions

   o  Arexx commands         The ARexx port and commands
   o  Archive support        How you can work with *.lha and *.zip
   o  FTPMount               How you use FTP with ClassAction

   About the program

   o  History                Since the beginning...
   o  The Authors            Who wrote this program
   o  Greetings              To all who helped !
   o  Known Bugs             Nobody is perfect
   o  The Future             What should be added in the future
```

Index See button on the top

1.2 Index

A

Actions
ARexx commands
Archives
Authoren

B

Bugs
Buttons

C

ClassAction
ClassActionPrefs
Classes
Commands
Commands (internal)
Configuration

D

Devices
Directory Opus

F

FTP
FTPMount
Future Improvements

G

Greetings

H

History

I

Installation

L

Lha files
Licence

M

Menu

O

Options

R

Requirements

S

Settings (menu item)
Settings (preferences)
Start modes
System Requirements

W

Warranty

Z

Zip files

1.3 What is ClassAction ?

What is CLASSACTION ?

ClassAction is a little tool which will simplify the life of
all hard disk users.

When you own a hard disk, you have always a bunch of files :
executables, modules, pictures, sources, animations, sounds...

ClassAction determine for you the kind of a selected file and
displays a list of actions to perform on the file.

For example, when you select a GIF picture, it will be recognized
by ClassAction as a GIF Class file, and a list of actions
will be shown with Actions like 'Display' or 'Edit'.

ClassAction is also a full file manager with many options.
Copy, move, rename, delete, add icons, show information, ...
Just try it and you will see all the possibilities!

ClassAction is highly configurable:
You can add your own classes and actions.
Actions use external programs, so you can use your prefered
Gif viewer to show your Gif Files (or anything else).
Specify your own drive or directory menu, your own icons and shortcut keys.

ClassAction has an AppIcon, an Arexx port and is localized.

ClassAction uses xfdmaster.library to auto decrunch crunched files.
With this feature, even a crunched class can be detected.

If you take time to configure ClassAction, you can do everything
with it !!! It's an easy way to handle files.

Before deciding to erase it, just try it !!!!!!

See also

More about ClassAction

1.4 more about ClassAction

More about ClassAction

ClassAction is now MailWare, so please send a mail to the author if you
use ClassAction frequently.

PD Distributors are allowed to include the ClassAction Package
into their collection as long as they let me know that they have
included it.

I hope you will find this tool useful.
(at least, I find it useful :])

See also

Technicals Infos

1.5 System Requirements

Requirements

- Amiga OS 3.5 or greater
- a Hard Disk
  (ClassAction is quite useless without a hard disk)

Recommended:

```
o  The utilities "Lha" and "Zip"
o  A high resolution like 1024 x 768
o  More speed, like a 68060 ...
```

1.6 Configuration

```
Configuration


All the configuration is done with ClassActionPrefs.
You don't have to edit tooltypes or config files, just
start ClassActionPrefs - for example from the ClassAction
menu - and change the settings.

You can choose between three parts:

o  Options

   Some global properties of the ClassAction window, the
   way ClassAction works and how it looks.

o  Commands

   The icons on the top and bottom of the window and the
   key commands.

o  Classes

   The classes and actions - the most important part.


The menu has items known from standard preference programs:

Project:

o Open...   - open a definition file from somewhere on disk
o SaveAs... - save a definition file to a file

As ClassAction needs several config files, these actions only
apply to the active page (options/commands/classes).

Edit:

o Reset to defaults - is not yet implemented
o Last Saved        - loads the settings you have last saved
                      in ClassActionPrefs
o Restore           - resets the settings to the state as you
                      started ClassActionPrefs



When you have done all changes, use Save to make this the
standard for ClassAction, or Use to try it only this session
(until you reboot). Cancel or the close button will discard your
```

changes.

1.7 Options

Options

Some ClassAction features are configurable. To change them, just start ClassActionPrefs, adjust the options and use "Save" or "Use".

Remark: If you have changed the options while ClassAction is running, some options can not be adjusted immediately; they are here marked with a * behind it. ClassAction will use all the new settings if you restart it.

Some of the options can also be modified inside ClassAction. They can be found in the Settings menu .

This is the list of available options. Some are switches and get a hook if you click on them, some have values that can be modified by a double-click.

General:

Start as icon *

If you choose this option, ClassAction will start as an AppIcon. Use it for example if you start ClassAction from WBStartup.

Iconify by close gadget

MUI applications can be iconified when you click on the close gadget of the window. As ClassAction was such an application and doesn't want you to change your habits ;) you can activate this behaviour.

Confirm replace

If you click on 'copy' or a related button, there exists a probability that an old file is replaced by the new one. If "Confirm replace" is set, you will be asked in such a case whether you do really want to delete the old file.

Check space before copying

To check if the selected files fit on the destination volume before ClassAction starts copying, switch this on.

Show help

If you don't know which icon is for which purpose, turn this option on and you will see a short hint on the bottom when the mouse is over each button.

Public screen

Perhaps you use another screen and want to start ClassAction on this public screen. If you know its name, just enter it here and ClassAction and ClassActionPrefs will use this screen instead of the workbench. If no name or "Workbench" is entered, the Workbench will bes used as usual.

Cli size

It is the size of the CLI output window, if a command of type CLI is executed.

The syntax is : DEVICE:TopX/TopY/Width/Height/Title

Thus you can select another device than CON: for the CLI's or define a new window position and dimension.

WARNING :

If you set this to a weird value, ClassAction will not launch CLI.
DON'T modify it unless you know what you are doing.

** NEVER ** add device commands such AUTO, CLOSE, WAIT to this string, ClassAction will do it for you.

** NEVER ** put spaces into the title.

Default Value is :
CON:0/0/640/100/ClassAction_Output_Window

Window:

Left Edge, Top Edge

Specify the distance of the ClassAction window from the left and top margin of the screen.
If they are both -1, the window is centred on the screen.

Width, Height

Give the width and height of the window.
If not given, the size is adjusted to the window gadgets.

AppIcon left edge, AppIcon top edge

Usually the AppIcon will be put on the workbench according to the position of the program icon. If this is not your preferred place, enter these values. A value of -1 won't change the position.

Popup wait time

In some cases you can choose an action from a popup menu,

which is usually closed after some seconds. Here you can specify this delay. Enter the amount of seconds the menu shall stay open, or a value of -1 if you don't want that the menu closes itself (you can close it anyway by a click somewhere else or by pressing the right mouse button or a key).

Buttons:

Parent *

Show and use the parent gadget left to the path in source and destination listers.

Path menu *

Show and use the path menu right to the path in source and destination listers. You can enter the paths in the group "Path menu".

History *

Show and use the history gadget right to the path in source and destination listers.

Listers:

Show images

Images are really nice, but if you don't have much space or you think the icons are ugly :(turn this option off. You should then turn on "Highlight directories".

Use small images

Check this option to have smaller images left to the files/dirs.

Show devices

In most cases you are content with volumes and assigns - but if you want to see also the devices select this option. (This can be interesting if you have installed an env: handler etc.)

Highlight directories

Turn this on to show directories and volumes in white colour. Else they are written black like other entries.

Show drawers first

Use this option to adjust the way the file listers are sorted. If active, directories are positioned at the top of the list.

Show date as text

If this is activated, some dates are replaced by a text, like "Today","Monday" etc.

Display also time

ClassAction shows the date of the last changes for each file, and aditionally the time if you activate this switch.

Source start path

This is the directory that is listed in the source lister when ClassAction starts.

Destination start path

The same as "Source start path" for the destination.

Icons:

Display icons

If you select this option, .info files will be shown in the listers.

Create icons

The 'makedir' button creates a new directory. If "Create Icons" is set, it also creates an icon for this directory.

Copy icons

With this option activated, the 'copy' and related buttons will copy also the icon if you copy a file. This is useful if "Display icons" is not set.

Paths:

Path to the AppIcon *

You can select your own AppIcon for ClassAction.

Example: Sys:icons/head
will use the icon head.info from directiry Sys:Icons/ as AppIcon.

If ClassAction fails to load your Icon file, it will use the default one (ClassAction.info).

Path to the manual *

If you store this documentation in an other directory than HELP:, then you have to tell ClassAction where it can be found. Example: sys:docs/classaction.guide

Path to ClassActionPrefs

ClassActionPrefs saves also another path to the preferences:
its own path !
As it knows where it is, you do not have to change that ;)

Path menu:

You have 20 free places for your own paths that are shown in the path popup menu.

Make a double click on one of them to set it. You are first asked for a directory that shall be listet, then for a name that shall be used as title in the popup menu.

You can leave entries free, then they are omitted.

See also

Settings menu

1.8 Commands

Commands

You can configure the icons on the top and on the bottom, and you can define key commands that do not have an icon.

To get an overview, just start ClassActionPrefs and look at the predefined icons.

You can add a command with the button "Add" on the left bottom.
Then move it with the arrows to the place you want to have it.

Name and help

Specify a short name. This is only used in ClassActionPrefs and has no meaning for ClassAction. It is the text you will see in the left lister.

Specify a help text that is shown in ClassAction, if
"Show help" is turned on.

Command

The most important part is the command itself:

first decide, if you want to start something by Cli (CLI), by Cli without a window (No CLI), by Workbench (WB), as ARexx script (ARexx) or as internal command (Internal), see modes .
If you configure the icon bars, you may want to have some space between

icons, so create a command and use SPACE as type.

Now choose the command. You have several possibilities:

o Enter the command in the string gadget.
o Choose "Load" and then search for the program you want to be inserted.
o If you want to use an internal command, click on "Internal commands"
  and select the one you want.

Additionally you can use some internal variables that are replaced by
the selected file, the shown paths, a string or file requested from the
user ... See variables for a complete description.

Icon

If you want to see the command as icon in the ClassAction window, you
have to specify an icon file. Enter the name of the file that has an
icon (ClassAction will hide the ending ".info" if you enter it).
You can have a preview of the icon by clicking on "Preview", or even
of the whole icon bar when you choose the group in the lister and then
the preview button.

Key

Finally you can choose also a key for this command. If it is a key
command, this is the only way you can call it.
You have two choices: ASCII or RAWKEY.

ASCII takes a printable character like ’a’, ’W’ or ’:’ and is case
sensitive.
RAWKEY identifies the number of the pressed key. Thereby you can define
keys like F1,...,F10 or ESC,DEL,TAB,RETURN ...
If you don’t know the number of the key ;) press "Select RAW key"
and then the key you want to define. The number will immediately be
shown.

Be careful not to overwrite other commands you want to use; HELP key and
the latin characters (’a’-’z’,’A’-’Z’) are already used by ClassAction,
but your defined commands will overwrite the internal commands.

See also

 Internal Commands
 Modes
 Variables

1.9 Classes

Classes

Defining new classes and actions is the heart of the program.
To do so, you must use ClassActionPrefs program.

Using ClassActionPrefs should be easy... so let's go !!

The window is divided into two parts : Classes & Actions.

First select a class, the associated actions will be displayed
on the actions part.

To add a class or delete a class, simply click on the associated
buttons.

It's the same concept for the actions...

1. What is a Class
2. Creating a new Class
3. Defining a new action
4. The learn function

1.10 Using the learn function

Using the learn function

The learn function is provided to help you defining new classes.

When you define a new class, you may have to define offsets for it,
and it may be long and boring to edit files to guess which offsets
could define this new class.

This is why the marvellous function learn has been made !

To use it, just follow this procedure :

1 : Define your class as normal, fill name and classname.

2 : Click on the 'learn' button.

3 : Select with the provided requester as many files you want
    as long as they belong to the class you want to define.
    The more files you select, the better will be the result.
    ClassAction will try to find out the offsets definition.
    (to select multiple files, use the shift key.)

4 : After analysis, you have a window with found offsets.

5 : Modify them by hand if needed.

6 : Click on 'Accept' to use the offsets for your class.
    or on 'Cancel' to cancel.

WARNING :

If you want the learn function works you must be sure that :

* ALL CHOOSEN FILES BELONG TO THE SAME CLASS.

* ALL CHOOSEN FILES ARE NOT CRUNCHED.

1.11 What is a Class

What is a class?

A class is a familly of files. For example C files can be
considered as a Class, let's call it C Class.

With ClassActionPrefs you can define as many classes as you
want, as long as you explain how to recognize it.

To explain how to recognize a Class, there are two methods :
the match name and the file contents.

Matchname attribute is used to recognize a file regarding
it's name.
Offsets attributes are used to recognize a file with it's contents.

There are three Built-in Classes that you cannot remove, they are
displayed in white in the Classes ListView and cannot be ranamed.

The first one is called "Unknown Class"
This Class contains all the files that ClassAction cannot recognize.

The second one is called "Generic Actions"
this class contains Actions that will be displayed in ALL
other classes...

Interest : if you want to have an action 'Copy' for all the
classes, you can create it for each class you define, but it's
looooong and boring.
A better way to do that is to create this action in the
"Generic Action" Class, so 'Copy' will be displayed for all the
classes.

The generic actions are displayed in white in the ClassAction
action listview and are only visible in the ClassAction Window.
They will not be displayed when ClassAction is an AppIcon,
or via Arexx command Load.

The third one is called "Directory" and holds actions available
when you select a directory.

1.12 Creating a new Class

```
Creating a new class


Just Click on the 'add' button on the classes part to add a class.

A class has 3 properties :

    - a name
    - a matchname
    - offsets


o   The name is simply the class name, it's up to you to choose it.
    WARNING : a class name must be unique.


o   The matchname is any regular AmigaDos expression, like :

        #?.c , mod.#? , #?.c|#?.h , #?b[a|c] , #?toto? .......

    (read the AmigaDOS manual for all wildcards)
    WARNING : Don't use wilcard * but use #? instead.

    The matchname is not case sensitive, thus toto.C match with #?.c


    If you define a class using a matchname, you must be sure that
    the definition is always good.

    Example : if you define the GIF class with the matchname #?.gif
              all files with .gif will be recognized as GIF files.

              But are you sure that ALL .gif files are GIFs, or
              that all your GIFs have the .gif extension ?

    So, you should use matchname only in two situations :

    - the matchname is a bijection of the class
      (ex: #?.info is a good enough matchname for Icon class)

    - you don't have the choice
      (ex: how to recognize a C source, excepted with #?.c)



o   Otherwise, you should use Offsets.

    An offset is a place in a file where we should find something
    to recognize it.

    For example, GIF pictures always begin with string 'GIF' at the
    offset 0.
```

```
There are three syntaxes for defining offsets :

=====================================================================

Syntax #1 :   Offset,HexString

    Offset is a DECIMAL number holding the Offset.
    HexString is an HEX string that should be found at that offset.

    Example 1 : 0,4f4a   means that the file must begin with bytes
                         $4f and $4a at the position 0

    Example 2 : 9,448b3c means that at byte #9, we should find
                         $44 $8b $3c

=====================================================================

Syntax #2 :   Offset,'String'

    Offset is a DECIMAL number holding the Offset.
    String is an ASCII string that should be found at that offset.

    Example 1 : 0,'GIF'   means that the file must begins with
                          string 'GIF'
    Example 2 : 9,'FuBar' means that at byte #9, we should find
                          string 'FuBar'

=====================================================================

Syntax #3 :   Offset,"String"

    Offset is a DECIMAL number holding the Offset.
    String is an ASCII string that should be found at that offset.

    Note the difference with the previous syntax, here we use "
    to define the string, and in syntax #2 we used '.

    It's the same concept than in Syntax #2 but here, the string
    comparison is NOT CASE SENSITIVE.

    For example : an AmigaGuide file always begin with string :
    @database in lower or upper case.

    If you use method #2 to recognize an amigaguide file, with
    0,'@database', ClassAction will not declare a file beginning
    with @DATABASE as an AmigaGuide file.

    It works if offset is defined with syntax #3 : 0,"@database"

=====================================================================


You can define up to 5 Offsets to define a class.
```

A file is recognized as a class if all the Offsets match.
If the file fits more than one class, it will be recognized as
the class with the most offsets.

Ex : if the class X is defined like :

Offset #1 : 0,4a8b6c
Offset #2 : 58,14

All the files beginning with 4a8b6c AND having $14 at byte #58
will be declared as X.

To define several offets just click on cycle gadget 'Offset #'
to activate the next offset.

Remark : The ASCII Class

There is a built-in Offset command named : ASCII[]

If you put this Command into Offset #1 (i.e Offset#1=ASCII[]),
it will match with ASCII Files.
But ClassAction will try this after everything has failed.
Thank to this, Amigaguide files (that are ascii) will not be
recognized as ascii if you have already defined an Amigaguide
Class.

Normally you shoudn't use it as I have provided a standard
Prefs file where the Class 'ASCII' is defined using this
command.

1.13 Defining a new action

Defining a new action

Once the class is defined you should define actions for it.
Each Class can have as many Actions as you want.

Simply click on the 'Add' button on the action part to add
an action.

An action has 6 properties:

- a name
- a run mode
- a stack size (only if run mode is Cli)
- a delay (only if run mode is Cli)
- an exec command
- a RescanDir flag

o Name is the name of the action.

o Mode can be : Cli, WB, No Cli, ARexx or Internal

(see modes).

- Command is an AmigaDOS valid command line and can contain parameters.
  YOU SHOULD always use the full path for the executables.

  Example : use C:Copy instead of Copy in exec line.

  You can put in exec lines, Arguments and Commands built into ClassAction.

- Set the RescanDir Checkmark if you want this action rescan the current directory , usefull for actions that change the directory contents such delete or rename a file.

Buttons

- The Up and Down arrows buttons permit to sort the actions.
- The 'Load' button asks you to select an executable in exec line.
- The 'Commands' button pops up a requester which lists all the possible arguments and commands for an exec line.

1.14 Modes

CLI mode

If 'Cli' is choosen, then when selecting the action, the action will be launched from a cli and the stack size of the cli will be determined by the stack value (default is 4096).

The run mode Cli will only open a Cli if it's needed (if the executable displays something).

You can define the delay property for CLI:

If Delay is negative (i.e. Delay = -1), the Cli will wait until you close it by hand with the close gadget in the top left of the window.

If Delay is zero (i.e. Delay = 0), the Cli will close itself as soon as the task is terminated.

If Delay is positive (i.e. Delay = n with n>0), the Cli will wait n seconds before closing itself, but you can force the Cli to close by clicking the close gadget.

The dimension of the used Cli can be found in the ToolType CLISIZE.

WB mode

If 'WB' is choosen, then no cli will be opened, and ClassAction

will simulate a Workbench launching of the action.
The first string will be considered an icon, and the next
arguments are handled as if they would have been dropped on
the icon.
Enter '"[F]"' as command, and the filename will be started as if
you had double-clicked on it.

WARNING : this mode is only valid with files that have icons,
and only filenames (full name) should be used as arguments.

NO CLI mode

If 'No Cli' is choosen, no cli will be opened even if the
program displays something, but the task is still running
from a CLI.

AREXX mode

If 'Arexx' is choosen, it will launch rx with the given exec
command. Exec command MUST be an arexx script.
Of course RexxMaster should be Active and Rx in the directory
Sys:rexxc/ to work.

INTERNAL mode

ClassAction has some built in commands that you can call.
They must be entered in big letters. Variables are indicated
by a small letter. n allways stands for the lister, where
0 indicates the source and 1 the destination.

ADDICON	Add an icon to the selected files. This is chosen by the system according to your DefIcons settings.
ALL	Select all files in the source list
COPY	Copy all selected files to the destination
COPYAS	Copy all selected files and rename them
DELETE	Delete all selected files
GETSIZE	Compute and show the size of all selected files
LIST n dir	List the drawer 'dir' in lister 'n' dir: a valid path like sys:tools, can also be omitted for a list of devices
MAKEDIR n	Creates a new directory in lister 'n'. An icon is created if this option is activated.
MOVE	Move all selected files to the destination
MOVEAS	Move all selected files and rename them
NONE	Deselect all files in the source list
PARENT n	Go to the parent directory in lister 'n'
PATTERN	Select all files that match a pattern. You are asked for a pattern according to the standard rules (e.g. #? for several unknown characters etc.).
REFRESH n	Refresh the lister n
SWAP	Swap the contents of source and destination lister
TOGGLE	Swap the selection state of each file selected become unselected and v.v.
TYPE	Selects all files that are in the same class as the

```
            selected one. This can last some seconds as each file
            in the directory must be examined.
WBINFO      Show WBInfo for all selected files. You must quit
            the requester to continue work.

Remarks:

o  The transfer commands (COPY,COPYAS,MOVE,MOVEAS)
   will check if files would be replaced and then ask
   you what to do with these files. This is done only
   for the first level, so it asks you only once for a
   directory.

o  COPY works also with archives.

o  Some commands can be interrupted by the stop button
   at the bottom right. It will become coloured if this
   is possible. Then the not processed files will remain
   selected.

o  Some of the commands can be rebuilt with cli commands, e.g.
   'SURE[Really delete?]delete "[F]"' which does nearly the same
   as 'DELETE', but the internal commands have some advantages,
   like the test for replacement or the active progress bar.


SPACE mode

This is no real mode, it only means that this a no command
but a space between icons. SPACE can not be used for actions
and is senseless for key commands.


See also

 Variables
```

1.15 Variables

```
Variables


Currently, the following variables are possible :

Path variables:

[F]    : full path of selected file
[S]    : full path of selected file without suffix
[B]    : filename  of selected file
[X]    : filename  of selected file without suffix
[SRC]  : path of the source lister
[SRCS] : path of the source lister, with trailing '/' if necessary
[DST]  : path of the destination lister
[DSTS] : path of the destination lister, with trailing '/' if necessary
```

```
Requester variables:

REQD[text] : Requests for a Directory.
REQF[text] : Requests for a File.
REQT[text] : Requests for a text.
SURE[text] : Asks the user to confirm.

=============================================================================

Path variables:

Example : let say you select the file ram:env/sys.prefs

    [F]    = ram:env/sys.prefs
    [S]    = ram:env/sys
    [B]    = sys.prefs
    [X]    = sys
    [SRC]  = ram:env
    [SRCS] = ram:env/

    Use quotes if you can't ensure that blanks are in the path and
    would confuse the command:

    "[F]" = "ram:env/sys.prefs"
    "[S]" = "ram:env/sys"
    ...


Example : let imagine you've selected the file ram:main.c

o   The exec line
             c:copy "[F]" "[F].bak"
    will be replaced by :
             c:copy "ram:main.c" "ram:main.c.bak"

o   The exec line
             c:copy "[F]" "[S].bak"
    will be replaced by :
             c:copy "ram:main.c" "ram:main.bak"


=============================================================================

Requester variables:


    REQs commands popup a requester with the title [text].
    This is useful when you need interactive command lines.

    Example :
        bin:lha x "[F]" to REQD[Choose a Directory to unarchive]

    This will popup a directory requester letting the user to choose
    the target directory, and the selected file [F] will be unarchived
    to the selected directory.
```

```
    REQF[] is the same except than it asks for a file.

    Example :
        c:dir "[F]" > REQF[Choose a file]


    REQT[] asks for a text, it's usefull to asks args for example:

        c:cpu REQT[Enter arguments for CPU]


SURE Command:

    SURE[text] command will popup a requester with text [text]
    and with 2 buttons : yes / no.
    If the user choose no, the exec line is aborted.
    If the user choose yes, ClassAction will execute the exec line
    on the RIGHT part of the Sure command.

    Example :
        SURE[Really delete this file ?]C:delete "[F]"

    This will popup a requester asking the user to reply yes or no
    to the question "Really delete this file ?".
    If the user reply no, nothing is done ; if the user reply yes,
    then C:delete "[F]" is executed.

=========================================================================


Of course you can combine any number of arguments / commands in an exec
line.

Example :
    SURE[Really rename this file]c:rename "[F]" REQF[Give me a new name]
```

1.16 Future Improvements

```
Future Improvements


Well, there are many proposals, but some need a lot of time, some make
the program very big or slow, and some are only interesting for a small
group of users.

What could be implemented:

o support for long filenames (well, I must admit that I do not know how)

o file popup menu like the one in DOpus (but how make Reaction report me
  a RMB click on a label?)

o sorting the classes as tree or import DefIcons definitions (not really
  necessary; DefIcons uses another structure and doesn't know actions).
```

See also

Contact

1.17 About

Current ClassAction team:

Planing, development and coordination:

Martin R. Elsner

Graphics:

Martin Eriksson (the other Martin ;)

Beta testers:

Christian Aichinger
Jan Britsch
Jean-Pierre Riviere
Jimmy Sulter
Jocke Sjöblom
Martin Eriksson
Patric Hofmann
Rolf Max Rotvel
Stefan Funke

Translators:

Abdul Alkaç (Dutch)
Arturo Franzin (Italian)
Bigo (French)
Javier de las Rivas (Spanish)
Martin R. Elsner (German)
Márton Dósa (the Hungarian Martin ;) (Hungarian)
Sinan Gürkan (Turkish)

I would like to thank the following people for their contribution to the renaissance of ClassAction:

- all the users who sent me emails and told me their opinion (if I have forgotten to add your name, please don't be sad ;)
- Evan Scott and Thies Wellpott for their agreement to use and continue the work on FTPMount
- Mick Saile for the permission to use his GlowIcons (opusgibar, available in Aminet)
- Dietmar Eilert for the great GoldED which I use for programming

o and Salim Gasmi for the development of the versions 1.0
to 3.6 of this fine piece of software!

Martin

I would like to thank the following people :

o Philippe Thomas (for suggestions, help, beta testing,
The French guide and UTT used for the install)

Hey Phil, if you read this, click here

o Bruno Durremberger (for Beta testing)

o Jean Michel Dessolas (for beta testing on a A4000/40)

o Joern Krueger for Beta Testing and good ideas.

o Obvious Implementations Corp (for Dice C Pro)

o Nico Francois for the ReqTools library

o Georg Hörmann for the *GREAT* xfdmaster library

o Stefan Stuntz for the marvellous Magic User Interface.

o All users contacted me to report bugs or for suggestions.

Salim

1.18 The Author

You can contact the author at the following address :

Martin R. Elsner
Müsener Straße 46

57223 Kreuztal
Germany

Web: http://www.martin-elsner.de

E-Mail: classaction@martin-elsner.de

or the coauthor Salim :

Gasmi Salim
4b rue des petits champs

67300 Shiltigheim

France

Irc: Dr_Unix (#amigafr, #amiga)

Web: http://www.gasmi.net

E-Mail: salim@gasmi.net or salim@sdv.fr

1.19 Using ClassAction

Using ClassAction

Using ClassAction is simple. If you do not know what icon is for
which purpose, just move the mouse over the icon and read the hint
at the bottom (you must have turned on the option ShowHelp).

In the DirLists you can move as in a normal file requester.
Left to the path string there is an arrow which leads to the parent
directory.
Right to the path string there are two popup menus: the first shows
a set of defined paths that you can also change; the second shows paths
you have already visited, it is the lister history. Select one of the
paths to go to this location.
Special actions are available through the
 icons at the top and bottom .
Some of these methods can be accessed by keys; the keys are defined
with ClassActionPrefs and can be viewed and changed.
In the standard setting, you can for example delete the selected
files by DEL. HELP allways shows this manual.

After you have selected a file in the lister, you will see in the
right listview, the file class and the corresponding defined actions.
Just Select the action you want...

Pressing a key will show the 1st file starting with this letter,
and Shift+Key will does the same for directories.
(This can be overwritten by your own commands that could for example
copy the selected files by pressing "c".)

If you double-click on a file, the first defined action
will be launched.

ClassAction window is an AppWindow, thus you can throw icons
on the main window.

Start the preference program by selecting 'Preferences' from the
 menu .
To quit, just click on the Close icon or select 'Quit' from the
 menu .

To transform the window into an AppIcon just choose the iconify icon
or choose 'iconify' from the menu .
Double click on the AppIcon for the window popup.

When ClassAction is AppIconified, you can throw icons on it so the
file type will be recognised.
A window pops up with the list of all possible actions, you
just have to choose one.

Using the file manager is REALLY simple, the LEFT listview is
ALWAYS the source directory and the RIGHT one on the group
"manager" is ALWAYS the destination directory.

See also

ARexx Commands
Buttons
Configuration
Menu

1.20 The Buttons

This is a list of predefined commands. You can change them and
create new ones with ClassActionPrefs .

Selection Buttons

On top of the source lister there are 6 buttons that make
selection easier:

All: Select all entries in the lister, even directories.
None: Deselect all entries.
Pattern: Select all files that match the pattern you enter.
         Directories will not be selected.
Type: Select all files of the same class as the selected one.
         Directories will not be selected.
Toggle: Switch selected/unselected entries.
         Affects also directories.
GetSize: Compute and show the size of all selected files and
         directories.
Swap: Swap the contents of the source and destination lister.

Directory Buttons

For both listers there are buttons to move inside and create
new drawers:

Devices: Show volumes~and assigns.
MakeDir: Create a new directory; you will be asked for its name.
         If the option 'CreateIcons' is turned on, also an icon
         is created.

Action Buttons

```
Some actions can be executed on the selected files directly by
these buttons:

WbInfo:  Show the Workbench icon information of the selected files.
         There you can modify tooltypes, protection bits, ...
AddIcon: Create an icon for the selected files.
         If there already exists an icon you will be informed, and
         nothing will happen.
Rename:  Rename the selected files.
Delete:  Delete the selected files. You will be asked if you
         really want it.

Transfer Buttons

You can transfer files with the aid of following buttons:

Copy:    Copy the selected files to the destination directory.
CopyAs:  Copy the selected files to the destination directory, and
         give them the name you enter.
Move:    Move the selected files to the destination directory.
MoveAs:  Move the selected files to the destination directory, and
         give them the name you enter.

Other buttons

There can be other buttons to show the possibilities. Look into
the preferences to get more information about the commands.


See also

 ClassActionPrefs
 Commands
```

1.21 The Menu

```
Project

Help:        Show this manual.
             As alternative you can press the 'HELP' key.
Preferences: Start  ClassActionPrefs  to modify the
             settings.
About:       Show information about the program.
Iconify:     Close the window and create an appicon.
Quit:        Leave the program.

Settings

Snapshot Window: Save the current window position and size.
                 The next time ClassAction starts it will use these
                 dimensions. You can also specify exact values
                 in  ClassActionPrefs .

The following options are identically to the  options
with the same name.
```

So if you want to change one of the options permanently, you have
to change the preferences .

1.22 Technicals Infos

Technicals Infos

ClassAction was first coded with DICE C 3.0 and is now developped in C++
with StormC 3.0, using ReAction (nice name ;-).
It's real C++ with some nice classes for ARexx, the menu, the preference
files, ... Contact the author if you want to look into the source code.

Miscelleaneous info :

The prefs file is an ASCII file named ENVARC:ClassAction.prefs

DISK libraries used if found :

    xadmaster.library
    xfdmaster.library   V30+

How ClassAction determines a class :

    1- test if the filename matches matchnames of defined classes.
    2- test if the file matches offsets of defined classes.
    3- Decrunch the file using xfdmaster.library (if turned on).
    4- test if the decrunched buffer matches offsets of defined classes.
    5- test if the file is ASCII (if ASCII Offset Command exists)

    If everything fails, the file is declared as 'Unknown Class'.

1.23 Installation

Installing

Use the installer script given with the archive.
To use it, just click on the install icon.

If you move the files after installation, you will have to change
the settings. Just start ClassActionPrefs and look at the paths
(manual, ClassActionPrefs, icons). Change them and save.
The next time ClassAction starts, it knows what has happened ;)

See also

 After Install

1.24 Legal words

Copyright

Warranty

No responsibility or liability will be accepted for any damage that
may appear to have resulted from use of this program. All use is at
your own risk. The software is provided "as is" without any warranty
implied or otherwise to the fitness or accuracy of the software and
documentation. The documentation is believed to be correct but the
author reserves the right to update the software and/or documentation
without notice.

See also

Bugs

1.25 History

History

22/04/01 : V4.1
- Both listers now show name, size, date and protection bits, and
  can be sorted by name, size or date (click on column title).
  Adjust the columns by clicking on the "invisible" separators.
- Lha and zip archives can be browsed like real directories!
  You can copy (=add) files to an archive or copy (=extract) files
  from archives (with path).
  All other methods are impossible inside archives.
  (For viewing and extraction you only need xadmaster.library,
  adding is done via the commands "lha" and "zip" which have to
  be somewhere in the systems path. It is only possible to add
  files to the root dir of the archive.)
- The PopUpWindow now contains buttons, POPUPWIDTH and
  POPUPHEIGHT are obsolete, window will allways open.
- Added tooltype DATESTRING to specify if you want "today" etc.
  instead of the dd-mmm-yy date.
- Added tooltype MUIQUIT to iconify the window with the close
  gadget.
- Added keys KEY_DEL (standard: <DEL>) and KEY_RENAME (<F4>) as
  shortcut for deleting and renaming.
- "Swap" rewritten and much faster.

```
- "Copy" now uses option CLONE to keep date and protection bits.
- Replace requester shows size and date of the two files.
- Text requester opens allways under the mouse.
- After actions the old position in the list is shown, not the top.
- Busy pointer and filename display while copying,...
- Path buttons have allways the same size and are really removed
  if you specify less than 13 paths.
- Bugs removed: copy .info works correct, "move as" also,
  WBInfo now displays devices correct (thanks to Stephan Rupprecht!)

31/03/01 : V4.0
- "Rewritten" the program to use Reaction and new OS3.5+ features.
- Now only 13 path buttons can be configured (should be enough!).
- Manager actions are now images!
- Devices are now shown in the normal file list.
- New buttons: MakeDir also for right file list, WbInfo, AddIcon,
  MoveAs, all the selecting methods.
- Added a menu(!) with help, preferences, settings, ...
- Added asynchronous help!
- Added a progress bar for copying/moving/...
- Now the class of a file is the one with the MOST fitting offsets,
  not the first one that matches.
- Comment optimization and Tooltype NOCOMMENT removed.
- New tooltypes: COPYICONS,CREATEICONS,DRAWERSFIRST,HELPFILE,SHOWHELP
  X,Y,WIDTH,HEIGHT,POPUPWIDTH,POPUPHEIGHT,more keys
- ARexx port is now CLASSACTION.1 to make handling easier


28/03/98 : V3.6
- Added a new executable named CAStart wich can be set as project
  for your icons, and thus opening the CA window choices.
- ClassAction now set the filetype in file comments in order
  to speed up file recognition.
- Tooltype NOCOMMENT added to disable file comments.
- Main DirList of ClassAction is now keyboard usable
  With up/down/space/Enter keys
- Added key filestart, if you press on a key, the first file
  begining with this letter is selected (Shift+Key for Directories)
- Added a key to allow pattern selection (configurable via tooltype)
- Added a key to select all Files (configurable via tooltype)
- Added a key to unselect all files (configurable via tooltype)
- Added a key to Toggle files (configurable via tooltype)
- Added a key for Parent Dir (configurable via tooltype)
- Added a key for SelectFile (configurable via tooltype)
- MoveDir bug fixed
- Makedir requester bug fixed
- Some routines optimized and thus faster.
- Command Crash/Hang/Hit in ClassActionPrefs Fixed
- Some minor other bugs fixed

27/06/97 : V3.5
- Bonus Version.
- CA_Register is no more needed, registration is done on-line
  on my site http://www.gasmi.net
- File Manager now handle icons as well
- Added tooltype DISPINFO
- Modifyed the tooltype GTLIKE
```

```
- Minor bugs removed

07/02/97 : V3.4
- This was supposed to be the LAST release of ClassAction
- Corrected some minor bugs
- Added tooltype ICONSTART to force ClassAction to start iconified
  for people who does not know how to do this using MUI Prefs

23/08/96 : V3.3
- ClassAction is no more ShareWare but MailWare !!!!
- Included Ca_Register to register for free
- Some Bugs removed .

07/07/96 : V3.2
- ClassAction has been recompiled with latest release of Dice
  and is more stable.
- Arexx engine is now full .
- Some "bugs" removed.

09/04/96 : V3.1
- ClassAction have now a tiny built-in File manager.
- Some bugs removed.

12/03/96 : V3.0 (Major Update)
- ClassAction and ClassActionPrefs now use MUI 3.0+
- ClassAction can have up to 50 path buttons
- ClassAction recognize directories.
- ClassAction handle multi selection.
- ClassAction has an appwindow.
- You can set the Learning Accuracy now.
- Tooltypes CAPREFS,GTLIKE added.
- Tooltypes HEIGHT,APPSTART,WINX,WINY,ICONX,ICONY,CX_PRIORITY,CX_HOTKEY
            REQBUG,STARTDIR,WBFONT,PUBSCREEN removed and no more needed.

25/09/95 : V2.8
- ClassAction have now a FULL commodtity support
  and a HotKey to show/hide ClassAction.
- Tooltype APPSTART can be set to HIDE (APPSTART=HIDE)
  if you want that ClassAction start hidden.
- Tooltype CX_HOTKEY added
- Tooltype PUBSCREEN added to allow using Public Screens
- The Internal File Selector of ClassAction was not freeing all
  the memory allocated, fixed now .

11/09/95 : V2.75
- The REQD,REQF,REQV commands were not really incompatibles
  with the appicon mode, but incompatibles with some programs
  such MagicMenu (Bad luck I use MagicMenu...)
  A lot of users complained about the automatic swith into
  a REQT command even if they don't use an incompatible program.
  I have added a tooltype (REQBUG) to let the user choose to
  auto-switch into REQT or not.

03/09/95 : V2.7
- Added commands [b],[x],[B],[X],[F],[S]
- Added the Action arexx command.
- removed some bugs in rendering routines
```

```
- removed a bug with multiple icons throwed on the AppIcon.
- The REQD,REQF,REQV commands are now swapped into REQT
  when using them from the AppIcon or Arexx , they are only
  compatible with the Window mode.

28/08/95 : V2.6
- ClassAction window is now resizable.
- ClassAction and ClassActionPrefs are now using ReqTools.library.
- Added REQV command to request a volume.
- Added REQT command to request a text.
- Added WINX and WINY tooltypes.
- Learn requester has now an ALL button.
- Selected File is now in all the REQs requesters.
- ClassAction use now a Key file for the registerd versions,
  this key file is placed in S: .

17/07/95 : V2.5
- ClassAction and ClassActionPrefs are now localised.
  and a French catalog is provided with the archive.
- Learn function added to ClassActionPrefs.
- To select the first action of a file, you must now Double Click
  on it instead of reselecting the file.
- Generic Actions are now Synchro and rescans the current directory.
- AUTOSELECT ToolType is no more used.
- REQs requesters opens now in current directory.
- minor improvements made.

12/06/95 : V2.1
- Added the 'Generic Actions' Built-in Class.
- Added SURE[] exec command.
- Added ASCII[] Offset Command to recognize ASCII files.
- Added Arexx commands: AppIconify, Show, Status, GetClass.
- Changed running tasks system, now I use Systemtags().
  We do not need tmp files anymore.
- you can now define a delay for CLI run mode.
- Added 'string' and "string" for offsets definition.
- ToolType OUTPUT is obsolete now and not used anymore,
  we use the new ToolType CLISIZE in replacement.
- Actions requester is now well sized, appear below the
  mouse pointer and uses to frontmost public screen.
- Swapped the buttons 'Use' and 'Save" and added 'Cancel'
  in ClassActionPrefs to follow the Amiga prefs look,
  moved the button 'about' in top right corner as '?'.
- ClassActionPrefs Cycle gadgets routines weren't
  100 % system friendly and some patches like Cycle2Menu
  makes bugs with ClassActionPrefs; It's fixed now.
- Improved the recognizer code and the Info routine:
  they are up to 400% faster.
- Listview hilight color error removed.
- ClassAction does not anymore lock the Workbench
  screen when AppIconified.
- A nasty bug found and removed in ClassActionPrefs.
- Right Mouse button shows Assigns only if mouse
  is in the requester and Right Mouse again brings
  back to the Directory.
- ClassAction remember now the window position.
```

```
23/05/95 : V2.00 ( Major Update )
- ClassAction has now an AppIcon.
- ClassAction is now a commodity.
- ClassAction has now an Arexx port.
- ClassAction use now the default WB Font.
- Exec mode 'Arexx' added.
- Different color for Directories/Files.
- Up/Down gadgets added to ClassActionPrefs.
- 'Use' gadget added to ClassActionPrefs.
- Classes are now sorted into ClassActionPrefs Listview.
- APPSTART, ICONNAME, ICONX, ICONY, CX_PRIORITY,
  WBFONT, OUTPUT, ICONFILE ToolTypes added.
- When the window is iconnified it have now the right height
  regarding the screen default font.
- New Save Format (CASF20).
- Suffix/Prefix Button removed. Replaced by MatchName Gadget
  who accept any Wildcard.
- Config file moved into ENVARC:
- an installer is now provided with the archive.
- some Code optimization done.

05/05/95 : V1.43
- Cleaned up the requester code, now 5% faster.
- ClassAction now look for his name using WBstartup structure
  and then can be renamed .

02/05/95 : V1.42
- 'No Cli' Exec mode added to ClassActionPrefs.
- "" are always added to filenames even if not needed
  it's easier for AREXX scripts.

06/04/95 : V 1.4
- ToolType HEIGHT added.
- Some code optimization added.

22/02/95 : V 1.31
- If you click twice on the same file the first action
  will be lanched (it's faster than selected the first action
  by hand).
- this version is now ShareWare and you must register to get
  the registered version.

15/01/95 : V 1.3
- Added REQD[] and REQF[] interactive commands.
- Copy gadget added to ClassActionPrefs.
- minor improvments made.
- Beta testers reported this version is really stable.

25/11/94 : V 1.22
- First Public Release.
- Button 'Info' Added.
- ClassAction was Locking() the directories it read without
  UnLocking() them .... *FIXED*
- Code Optimization.
- Minor other Bugs removed.

08/11/94 : V 1.21
```

```
    - Program was crashing with empty floppy units  .. *FIXED*
    - Volume/Name Bug Fixed
    - <..> Item removed when root of a volume.

    07/11/94 : V 1.2
    - New interface, I have included my own fast file requester.
    - STARTDIR & DRIVE1 to DRIVE11 ToolTypes added.

    01/11/94 : V 1.1
    - Now using xfdmaster library to recognize and decrunch files.
    - Configuration with ToolTypes added (DECRUNCH,AUTOSELECT).
    - 'Unknown Class' is now a built in class with unlimited actions.
    - New save format (CASF11).

    16/10/94 : V 1.0
    - New Save Format (CASF10).
    - Window has now a zoom gadget.
    - a lot of classes definitions added.

    10/10/94 : Beta Version
    - tmp file bug removed.
    - Offset increment error removed.
    - using asl library for the file requester.

    01/10/94 : Alpha Version
```

1.26 The Arexx Commands

```
    ARexx Commands


    ClassAction has an Arexx port called : CLASSACTION.1

    This is the list of all the Arexx commands :


=========================================================================

    Quit

        Just quit ClassAction...

=========================================================================

    Use

        Force ClassAction to reload the prefs file.

=========================================================================

    Ver

        Return the version of ClassAction.

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

```
    Status

        Return the current status of ClassAction.

        Return 0 : ClassAction is AppIconnified.
        Return 1 : ClassAction is in Window mode.

=========================================================================

    AppIconify

        Force ClassAction to AppIconify.

        If ClassAction is already an AppIcon, this command does
        nothing.

=========================================================================

    Show

        Force ClassAction to show the main window.

        If ClassAction is already a window, this command does nothing.

=========================================================================

    Load <filename>

        ClassAction will try to load the file <filename>, and pop up
        the actions requester, letting the user choose one of them.

        If the file does not exist, this command does nothing.

=========================================================================

    GetClass <filename>

        ClassAction will try to load the file <filename> and return the
        class of the loaded file.

        If the file does not exist, this command does nothing.

=========================================================================

    Action <filename> <Action Pattern>


        ClassAction will execute the first matching action for the
        file <filename>.

        example : Action ram:toto.lha extr

        will run the first action with the name containing
        string 'extr' on the file ram:toto.lha

        The pattern is useful , you dont have to give the exact name
```

```
    of the action, just giving a part of it is enough.


    Note: Look on Rexx/ directory of the archive
          to find Arexx scripts included.
```

1.27 The archive support

```
Archive Support


ClassAction supports some special functions on archives.
If you double click on an archive file, it will be opened like a
real directory, and its contents are shown. You can move in this
archive, and you can copy a file from an archive to a volume or
from a volume into an archive and even start actions on files.
Other actions are still available in the action list, for example
for extracting the whole archive.
The file recognition is based on the filename for fast use. So
don't name files as *.lha that are no archives!

Survey

Supported file types:

o   *.lha files (respectively *.lzx / *.lzh)
o   *.zip files

Supported actions:

o   Double click in source lister opens a popup menu
    with the class specific actions
o   COPY (internal command) into archive / from archive
    (NOT from one to another archive!)

Technical info:

o   Reading the archive and extracting files is done via
    xadmaster.library.
o   Archiving files (copying to an archive) is realized by the
    dos commands "lha" and "zip" that have to be somewhere in
    your system's path.
o   For execution of actions the file will be stored in t:
```

1.28 FTPMount

```
FTPMount


What is FTPMount ?
```

Well, FTPMount can help you transfer files with FTP.
The difference to well known FTPManagers is the way to handle
the connections and the transfer:

A connection is handled like a normal directory. This means, you
can use standard tools or the shell instead of FTP commands.

That’s the cause that FTPMount is a great enhancement to
ClassAction: ClassAction becomes an FTP program, and you don’t
have to change between different filemanagers.

Install FTPMount

First you have to get FTPMount. Try to download it at the
ClassAction support side, there is the current version.

Unpack the archive (with ClassAction ...) and start the
install program.

The best choices are:

User: average
Put DOSDriver mount entry in: sys:wbstartup

The rest depends on your system, so take any choice you want.

This has only copied the needed files and made the assigns
FTPMount needs. You still have to configure it:

Start ClassAction and go to the directory where FTPMount
is stored (not by FTPMountDir:, you have to use the real path,
for example work:ftpmount); then move into the dir "Hosts".

You will see some directories. Create a new one (with icon!),
name it like the connection you want to add (the name is not
important anyway).

Then click on this directory and call WBInfo.
You must add several tooltypes here:

At least

HOST=..

for example HOST=ftp.uni-paderborn.de if you want to connect to
this site. Depending on the site you have to enter also

USER=...
PASSWORD=...

You can add

ROOT=...

to specify the path you will start in, for example
ROOT=/pub/aminet

If you want to see a status window while the connection is
established, add

STATUS

For other options please read the FTPMount documentation.

Now you can save the tooltypes and all configuration (for this
connection!) is done.

Using FTPMount

If you have successfully installed and configured FTPMount,
it could be useful to reboot your computer (depending on the
place you put the DOS driver; you can also click on the
DOS driver to activate it, that should suffice).

Then the game can begin:

Start ClassAction.
Start a connection program like Genesis or Miami to set up the
AmiTCP port, go online.
Then double click on the volume FTPMount:
Choose the directory you have inserted and configured.

If everything is well installed, a window will appear, showing
that FTPMount tries to open the connection.
If it succeeds, the contents of the FTP site are shown in
the ClassAction file lister.

Use methods like copy, delete, ... to transfer data. Some
methods will not be possible (especially class recognition
with offsets).

If you have specified the option "STATUS" for your connection,
you will allways see some information about current things
FTPMount does.

The connection is normally closed after some minutes of idle
state, but can be reopened as long as you are online.

When you have finished your work, you have to go offline,
that’s all, FTPMount will close the connections and you can
do other tasks!

1.29 What should you do after install

What now ?

After the installation, you should have a working ClassAction,

with a lot of classes definitions, but with very few Actions
defined for them (most of the defined classes does not have any
actions defined).

It's up to you to define actions regarding to your system
configuration and your prefered programs to use.

Then just load ClassActionPrefs and configure it for your
convenience...

If you don't know how to configure ClassAction, just read
this guide :)).

It may take a long time to perform a 'nice' config.
But once it's done, it's really GREAT !!!

One little hint: don't use too many classes:
the more you define, the longer ClassAction needs to find
the class the file belongs to. Just remove the classes you
never use. Remember that you can also define actions for the
generic class that suffice for many file types.

See also

Classes
Configuration

1.30 Message for Philippe THOMAS

Salut Philippe !!!

Je voulais simplement te remercier pour toute l'aide que
tu m'as apporté à la création de ce programme.

Quasiment toutes les améliorations de la version 2.0, c'est
toi qui me les a proposées, parfois même avec insistance,
style le resize de la fenêtre que je n'ai toujours pas fait.. :(

Encore merçi pour tous les appels téléphoniques ; parfois plus
d'une heure à discuter des Hooks, SystemTagList, de bugs etc...
et ce, même en periode d'exams.

Franchement si ClassAction commence à être cool, c'est beaucoup
grâce à toi, je crois que j'aurais eu la flemme de le paufinner
autant si tu n'étais pas là.

Bref, ce programme est aussi un peu le tien.

Okay Phil, à la prochaine.

Salim.

PS : Non, non ton processeur il est bien, il est pas buggé... :)).

1.31 Known Bugs

Bugs ??

There can occur some problems when DirectoryOpus is installed,
because not all workbench functions are accessable.
If you realize such an error, contact me .

See also

Author
Warranty