EasyWB

COLLABORATORS			
	TITLE : EasyWB		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		November 24, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 EasyWB **1**

1.1 EasyWB guide . . . 1
1.2 What is EasyWB? . . . 1
1.3 How does it work? . . . 1
1.4 System requirements . . . 2
1.5 Installation . . . 2
1.6 Usage . . . 2
1.7 Usage from GUI . . . 3
1.8 Icon tool types . . . 4
1.9 EasyWB menu . . . 4
1.10 Usage from CLI . . . 5
1.11 Using EasyWB as MultiView . . . 6
1.12 Thanks to . . . 6
1.13 About the author . . . 6
1.14 Copyright . . . 7
1.15 What's an AppIcon? . . . 8
1.16 What's pattern-matching? . . . 8
1.17 My system . . . 8
1.18 Program objects . . . 8
1.19 Filetypes . . . 10
1.20 File ID String example values . . . 11
1.21 Things to do/Bugs in EasyWB . . . 12
1.22 EasyWB History . . . 13

Chapter 1

EasyWB

1.1 EasyWB guide

```
EasyWB - Version 1.2
  Copyright © 1995 Arian T. Kulp
All Rights Reserved

  What is EasyWB
 How does it work?
System Requirements
   Installation
      Usage
   To Do/Bugs
    Thanks to...
 About the author
     History
    Copyright
```

1.2 What is EasyWB?

What is EasyWB?

EasyWB is the result of many hours of my toiling at the computer, trying to make a simple way to handle an arbitrary file. It finally works!

With EasyWB, just define a few things, then you can drop any file (even if the file has no icon of its own) onto the AppIcon to execute the appropriate program.

For example: just drop a picture on the icon and EasyWB will tell ViewTek to view it. Drop a text file on it and your favorite text editor or view will handle the rest. File recognition is based on either standard AmigaDOS file pattern-matching or up to the first 12 bytes of the file.

1.3 How does it work?

How does it work?

In my mind, what makes EasyWB easy to use, is its object-oriented approach to handling files. When you drop a file on its AppIcon, EasyWB scans through a list of filetypes that you define. When it matches the file up with one of the filetypes you created, it uses the program object you assigned to that filetype.
So you first make a list of program objects i.e. ViewTek, AmigaGuide, more, then you make a list of filetypes i.e. IFF picture, AmigaGuide document, etc., then link them together.

This may not sound easy to use at this point, but once you try it out, it won't seem so bad.

1.4 System requirements

System Requirements:

Any Amiga with OS2.1 or greater. (or does 2.0 support AppIcons?)

EasyWB will use ReqTools (reqtools.library), if present, for the optional string requestor used in arguments. This library is not required, but without it you cannot use the "String Requestor" argument in your program objects. As most people have ReqTools, this should not be a problem.

Also, EasyWB consumes about 23k plus memory allocated for your preferences. The more objects defined, the more memory used (though it uses very little).

No other files or libraries are needed.

1.5 Installation

Installation:

Installation is simlple. Simple copy EasyWB and its icon (EasyWB.info) to the directory of your choice. If you want to always have EasyWB available, you should copy it your WBStartup drawer.

If you are interested in using the sample preferences file, simple double-click the Install-Prefs icon. This file will define file types for you, but you will have to define programs to use these types.

1.6 Usage

Usage:

For the most part, EasyWB is intuitive. But until you understand it's basic principles, you may need a little help.

First of all, EasyWB has two different ways of using it. The first is its GUI (Graphic User Interface). This is the only way to configure EasyWB. The second is its CLI interface.

Please note: Only one copy of EasyWB can be run at one time. This made sense to me, as what would be the point of starting a second process to do the same thing using the same preferences? Trying to start a new EasyWB will simply bring up the requestor from the pre-existing EasyWB. This is transparent and will not look any different than if you had simply double-clicked on the EasyWB AppIconto begin with.

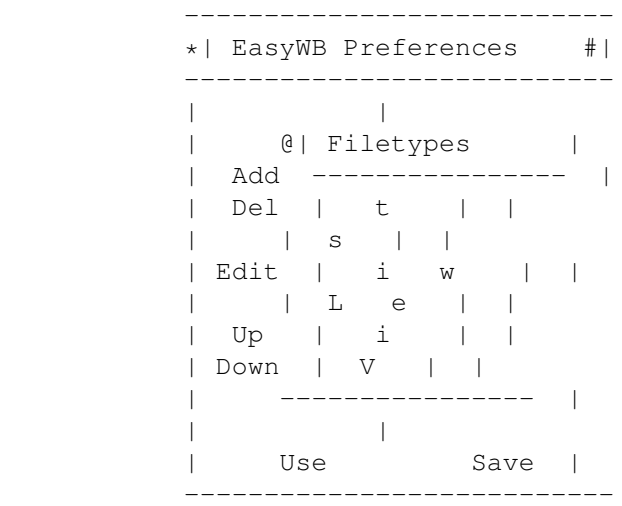
Also: I have done a few things to make EasyWB a good MultiView replacement for pre OS3.0 users.

1.7 Usage from GUI

From the GUI:

First of all, when you first start EasyWB, you won't see much. After a few seconds, an AppIcon with a black background and an asterisk (yes, that's what it's supposed to be!) appears on your WorkBench. At this point (the first time anyway), there isn't anything you can do with it except double-click on it.

Upon double-clicking, a window should appear as follows:



We'll address each gadget one at a time.

Add: The add gadget brings up a new window allowing you to define a new object based on your current list.

Del: The del gadget deletes the last object you selected. No effect if there is no current object.

Edit: The edit gadget brings up a window to edit the last object you selected.

Up: Moves the last selected object up the list one position (unless it is at the top).

Down: Moves the last selected object down the list one position (unless it
is at the bottom).

Use: Saves the current settings to env: and hides the window. These
settings will stay in effect until the next reboot.

Save: Saves the current settings to envarc: and hides the window. These
settings will survive a reboot.

The gadget showing Filetypes in the example above is a cycle gadget. By
pressing this gadgets you toggle between the two lists: "Filetypes," and
"Program Objects." When you select this gadget, the listview below it
updates itself to reflect the item in the corresponding list.

The Listview gadget shows all objects in the current list (as defined by the
cycle gadget above it).

For info on adding or editing filetypes click here:
For info on adding or editing program objects click here:

EasyWB also has a menu and the icon has tool types.

1.8 Icon tool types

Tool types are accessed from the Workbench by selecting the icon (not
AppIcon), then going to the Workbench Icon menu and selecting
Information.

The following tool types are supported by EasyWB:

ICONX Use this to specify the X position of the AppIcon on WorkBench.
ICONY Ditto for Y.
QUIET Normally, EasyWB will bring up a simple requestor when it is started.
Use QUIET to suppress it. You want it suppressed if it is in your
WBStartup drawer.

1.9 EasyWB menu

With the EasyWB prefs window open, an Intuition menu is available.
Two submenus are available: Project and Options:

```
Project      Options
-------      -------
Open Prefs O     Reset to Defaults  D
Save As... A     Last Saved   L
~~~~~~~~~~~~
About      ?
~~~~~~~~~~~~
Hide       H
~~~~~~~~~~~~
Quit       Q
```

Open Prefs: Opens a file requestor to select a prefs file to open.

Save As...: Opens a file requestor to select a prefs file to which to
save the current settings.

About : Brings up an informative requestor with a few facts about
EasyWB.

Hide : Closes the prefs window while leaving the program running.

Quit : Hmmmmm.... (does not save anything)

Reset to Defaults : Clears all current prefs. This is internal only, and
will not affect any prefs files you have selected or
previously utilized.

Last Saved: Loads the preferences previously saved using the Save icon
in the prefs window itself (not from Save As... in the menu).

1.10 Usage from CLI

From the CLI:

Using EasyWB from the CLI (or WorkBench Execute Command... option) has a few peculiarities to get used to. As noted before, you can only have one EasyWB running at one time. I decided that this was a good idea for the following reason.

My biggest use for EasyWB is to drop files on, but I also use it from within ToolManager, and from within the shell to handle files. Simply type:

```
EasyWB <filename>
```

to let EasyWB figure out the file and act on it. If EasyWB is already running, typing this will simply send a message to the already running process (thereby avoiding loading in preferences a second time), then quit. The pre-existing process handles the file.

If EasyWB is not already running, it will load in the preferences, act on the file, then quit.

Please note: You cannot only invoke the EasyWB AppIcon/GUI from CLI if you use the command:

```
EasyWB
```

Notice no filename after it. This will start it up, but you will not get your shell back unless you run it:

```
run <>NIL: EasyWB
```

I feel the best way to use it is to keep EasyWB in WBStartup. Then you can always drop files on it, or call it from a shell. It’s pretty compact in size, so you won’t notice it much.

Options for running EasyWB from CLI are:

```
FILENAME,CLIPBOARD/K/S,UNIT/S
```

What this means, is you can either use a filename:

```
EasyWB sky.iff
```

```
or, view the clipboard:
    EasyWB CLIPBOARD
which would view clipboard unit 0.
To view other units, use:
    EasyWB CLIPBOARD UNIT [0..255]
Note: To use the UNIT keyword, you must have already used the CLIPBOARD
keyword.
```

1.11 Using EasyWB as MultiView

```
Under 3.0+ there is a program called MultiView which uses external libraries
to handle arbitrary files.  You can view pictures, play sounds, read Amiga-
Guide documents, etc. all with this one program.  Unfortunately, below 3.0
we have nothing like it.

For people like me, this creates a problem.  I get a lot of new archives from
Aminet and BBS's, and many of them have MultiView set as the default tool.
Now instead of having to be forever editting the default tool string, just
rename EasyWB to MultiView, and as long as you have defined all of the file
types, it will work almost as well.

Even better, create a link from SYS:utilities/MultiView to EasyWB, and you
can keep track of EasyWB better.

I have also made the command line options look like MultiView by giving it
options to use the clipboard instead of a specific file.
```

1.12 Thanks to

```
Thanks go to:

Anthony Moringello.  For his assistance with many C questions, and example
         source from an AppIcon program he wrote (not very
         recognizable anymore!).

Jess Sosnoski.       For acting as a guinea pig, and offering much
         constructive criticism during the development of EasyWB.

Jan van den Baard.   For the wonderful program GadToolsBox.

Nico Francois.       For the easy-to-use but so powerful ReqTools.

Commodore-Amiga.     For writing an operating system which is just so fun to
         program!
```

1.13 About the author

```
About the author.

Hmmm, not much to say.  I've been writing C programs for my Amiga for several
```

years now. This is the only one which I've actually felt comfortable enough with the idea of distributing. It works! Consistently!!

I also enjoy music, reading, and writing. I love my Amiga, but not to the point of fanatacism, and I love programming on it. I enjoy talking to others that are the same way.

As for other work I've done, if anyone has the Xetec American Heritage Illustrated Encyclopedic Dictionary CDROM and cannot run it (I can't on my system), I wrote a pretty good program for viewing words and definitions. It's WorkBench-friendly, saves to the clipboard or another file, prints, and has a GUI, or shell based interface in the same program for about 20k!

To get in touch,

My email address is: ronnie@mmc.mtmercy.edu

or USPS address: Arian T. Kulp
     240½ Heritage Dr.
     North Liberty, IA 52317

and my phone number is: 319/626-6098

I love to hear from other Amiga owners, especially programmers.

1.14 Copyright

performance of this software is assumed entirely by the user.

1.15 What's an AppIcon?

AppIcon:
A gateway to a program, which utilizes an icon drawn directly onto the WorkBench screen. To access the program, just double-click on the AppIcon, or drop a file onto it.

1.16 What's pattern-matching?

Pattern matching:
A method of specifying files from a list, by identifying similarities in the desired files.

Example
Pattern : *.txt
Result : Any files with ".txt" as the last four characters of the filename.
README.txt, BBSLIST.txt would show up, whereas README or txt.BBS would not.

Pattern : *(~.info)
Result : Every file in the directory, except icons (files with ".info" at the end.

Notice the tilde (~) character can be used to negate a pattern. Also, parentheses ()'s can be used to specify just a portion of the name.

1.17 My system

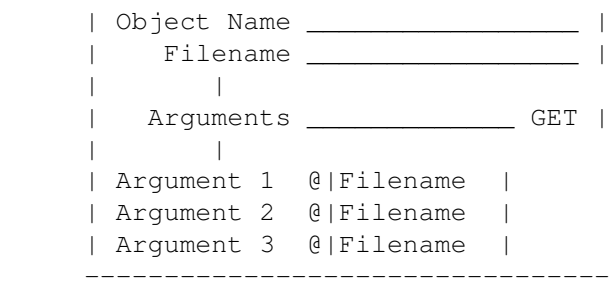
My system consists of an Amiga 2000 with A2620 card (screaming 14Mz 68020 card with a blazing 14Mz 68881 math co-processor), OS2.1, a whopping 1x speed NEC CDROM drive, Viva (sit down for this) 2400 modem (with fax, of course), external Pyramid MIDI interface, 2 whole megs of Fast RAM, 1 complete meg of Chip RAM, and a 120M SCSI HD. (Oh yeah, and 2 (two) low-density floppy drives)

1.18 Program objects

Adding/editing program objects:

Upon pressing "edit" on an existing object, or "add" while in the program objects list, a new window will appear:

```
---------------------------------
*| Program Object Edit    #
---------------------------------
```



Taking the gadgets in order:

Object Name: This string gadget is used for you to give the current program a name. This is useful only to you, and EasyWB never uses it. For my own config, I use Viewtek to view pictures and animations, and I name that object "PicView." I use muchmore to display text files, and name that object "TextView."

Filename : This is what tells EasyWB what to use. This must be a fully-qualified path name, or something in the system path list which identifies the program. Examples would be:
SYS:utilities/more
WORK:viewers/vt

Get: Calls an ASL file requestor to select a program filename.

Arguments : If the program requires any arguments (most do), specify them here. For example:
the sample player oplay will play a digitized sample, if invoked from a shell or CLI as follows:
oplay raiders.snd
In order to define this, you would enter oplay as Filename (with a path if needed), then for File Arguments, enter %s. %s is how you tell EasyWB to insert a word. EasyWB will substitute, based on what is selected in Argument 1, a value for %s. For many programs, it is a good idea to enclose %s in quotes. This way, if the filename dropped on the AppIcon has spaces, they will be handled properly.

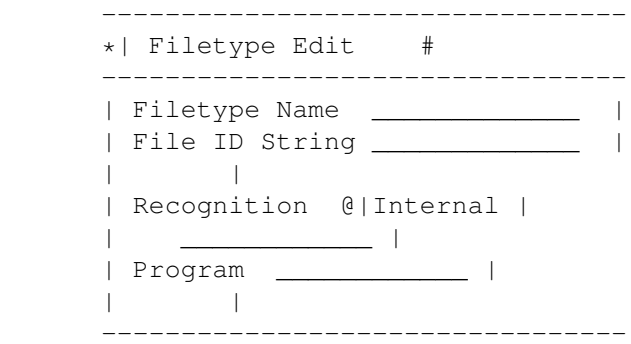
Argument 1-3 : These cycle gadgets tell EasyWB how to replace the %s's in the File Arguments string gadget. It defaults to Filename, which is how most programs want it. Clicking on it once changes it to Directory (replaces the %s with just the directory of the file). Clicking on it once again, changes it to String Requestor. The option requires reqtools.library to be installed in your libs: directory. This option makes EasyWB bring up a string requestor for you to manually enter that argument each time a file is dropped on it. This could be useful for a program like lha. This way you could enter "x" to uncompress the archive, or "v" to view the archive each time a different archive was dropped onto the AppIcon.

Play around with these and experiment. They only make sense as you use them!

1.19 Filetypes

Adding/editing filetypes:

Upon pressing "edit" on an existing object, or "add" while in the file objects list, a new window will appear:



Object Name: This string gadget is used for you to give the current file type a name. This way you can define a Gif picture and name it GIFPIC or something similar. This just makes it easier for you when you look in the list to add, edit, or delete.
This is useful only to you, as EasyWB almost never uses it.

I say "almost" never uses it, because there are two exceptions. I have created two internal filetype specifications which you can use in your configuration.

1) If you name the object "ascii" (all lower case, as shown), EasyWB uses this type if nothing else matches and the file is plain ASCII text. This should probably call a text viewer, hex viewer, or text editor.

2) If you name the object "default" (again, all lower case), EasyWB only calls this object if nothing else matches up. If you have an "ascii" object, it will catch plain text files, but if it filters past all your objects, and it is not ASCII, this object will be used. NOTE: You may not want to use this type until you have defined all the objects you need. By defualt, EasyWB will present you with a requestor asking if you want to add unrecognized file types when it first sees them. Once you have a "default" type, every file will be recognized one way or another.

Also: types "ascii," and "default" need not be at the end of the list.

File ID String: This string gadget allows you to enter up to 12 characters for EasyWB to use in identifying this filetype. This can be a tough one to figure out for some files, so I have provided a few examples in the included preferences file.

The use of this gadget changes based on the Recognition gadget.

```
    For internal strings, case is significant, and question
    marks can be substituted for values that are not always the
    same in the same types of values.  Example:  IFF files usually
    start with FORM, but the next four bytes reflect the size of
    the file.  Since most files have different sizes, use four
    question marks here.

    For filename strings, case is insignificant, and standard
    AmigaDOS pattern matching is used.

    NOTE: This string has no use for types ascii and default.

Recognition   :   This cycle gadget toggles between "Internal-ASC,"
    "Filename," "Filesize," and "Internal-Hex."

    Internal-ASC means EasyWB compares the File ID String against
    the first 12 characters of the file (character by character).

    Filename means EasyWB compares the File ID String against the
    actual file name.

    Filesize means EasyWB compares the File ID String against the
    actual size of the file in bytes.  Note: with this option,
    the string must be a number.  (duh!)

    Internal-Hex acts like Internal-ASC, except translates ASCII
    string into hex equivalent:
        typing 00 00 03 f3
        would find the values 000003f3, not 30 30 30 30 30 33...

    NOTE: This gadget has no use for types ascii and default.

Program       :   This gadget is used to actually assign a program object to
    the current filetype.  A window with only a listview containing
    all program objects will appear.  Simply select the object you
    want.  This is where it is a good idea to have your program
    objects named appropriately.

Play around with these and experiment.  They only make sense as you use them!
```

1.20 File ID String example values

```
File ID String example values:

Internal:
    FORM????ILBM    standard IFF picture
    FORM????ANIM    standard IFF animation
    FORM????ANIM    standard IFF animation
    FORM????8SVX    standard IFF sampled sound
    GIF87 or GIF89  GIF format picture
    ??????JFIF      JPEG picture
    ??-lh     LHArc, LZH archive
    @database     AmigaGuide database

Filename:
```

```
    *.c       C source file
    *.h       C header file
    *.mod     many mod music files
    mod.*     many more mod music files
    *.FLI     IBM FLIck animations
```

Just a few examples. Most data files these days have some kind of ID string. In most files, it's internal, but in a few (like most MOD's), you still must use filename identifiers.

1.21 Things to do/Bugs in EasyWB

Bugs:

Actually, there really aren't many that I can find. The biggest problem that I cannot figure out, is why it won't lock onto a volume with a space in it. I *had* the clipboard support working, but now the system asks for DISK: when I make a reference to volume RAM DISK:. As I use the standard Lock() on the filename, this makes no sense.

Also, if you make a link to EasyWB and start it from that, it will not be able to lock its icon, and will not open. This is not a problem unless you are trying to start the GUI from CLI.

To do:

Well first of all, I need to make all strings dynamic. Currently you only have 32 characters for all strings (except the ID string which is 12). I realize that this is a bad limitation, and I will fix it by the next release.

Also, I am considering making it possible to define the same filetype multiple times pointing to different program objects. This would be done so you could, for example, define a picture twice, then have it point to a viewer, and an editor (maybe DPaint). Then when you drop a picture on the AppIcon, a requestor would come up like this:

```
    ---------------------------------
    |@|EasyWB         |
    ---------------------------------
    | The file you have dropped:    |
    |       Pic.IFF       |
    | is defined more than once.    |
    |           |
    | Please choose program to use: |
    |           |
    |   --------- ----------      |
    |   |DPaint | |ViewTek |      |
    |   --------- ----------      |
    ---------------------------------
```

Please tell me if this is something I should spend my time on.

Another nice thing would be batch support. With this, you could drop several files on it, and EasyWB would deal with them one at a time. The reason why I don't do that, is I would have to make the program not detach itself when it runs. If I do this I risk a lockup with

```
    EasyWB until the invoked program returns.  If I do detach each program
    in a batch, it EasyWB would essentially process each file simultaneously.
    This would not generally be a good way to handle things.

If you find any bugs, I won't be surprised ;) and I want to know about them.
Get in touch with me and I'll do my best.
```

1.22 EasyWB History

```
Not much here yet.

v1.1  Beta release.
  Fixed problem of crashing if a requestor was brought up without an
  open window.

v1.2  Beta release.
  File requestors added.
  Internal hex recognition added.
  File size recognition added.
  Improved text entry gadget handling by not requiring the user to
   hit enter or return to make EasyWB notice changes.
  Fixed file pattern matching to be case-insensitive.
```