ToolTypesEditor

James T. Steichen

	COLLABORATORS		
	TITLE : ToolTypesEditor		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	James T. Steichen	August 11, 2022	

	REVISION HISTORY		
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ToolTypesEditor

1.1 ToolTypes Editor Index:

```
General Info

FileName
- The icon being examined.

ASL Button
- Find another icon to look at.


ToolTypes
- The program Icon junk.

ToolTypes ListView
- Your ToolTypes.

ToolType String
- The Mystery Gadget


Save Button
- Save the new ToolTypes & exit.

Restore Button
- Undo the last change made.

Cancel Button
- Exit without saving changes.


Vanilla Keys
- Satanic KeyPresses


Author
- Who wrote this great program?
```

1.2 ToolTypes Editor General Information:

```
                  A GUI for editing & viewing ToolTypes contained in an Icon.

The main reason for this program is to use it within other programs
as a ToolType Editor, thus saving you (but mostly myself) the headache
of having to write your own ToolTypes editor (again & again).

Here is an
              example
              of how to call it from your own program.

When the program is called from a CLI/Shell, it expects at least one
argument, the name of the Icon you wish to view/change.  There is an
optional second argument that you can specify, the name of the Public
Screen to open the program onto, which allows you use the program
from your own programs.

SYNOPSIS
   ToolTypesEditor <IconName> [PublicScreenName]

The program can be called from Workbench (but why would you want to when
there are other, more complete Icon editors available?).

HISTORY
    1.0 ToolTypes Editor is now working correctly!

SEE ALSO,
              ToolTypes
```

1.3 How to call ToolTypesEditor from your code:

```
                  Here is an example of how to use ToolTypesEditor inside your  ←
                    own
program:


void CallToolTypesEditor( void )
{
   char command[256];

   strcpy( &command[0], "ToolTypesEditor" );
   strcat( &command[0], " " );
   strcat( &command[0], IconFileName ); // has path also.
   strcat( &command[0], " " );
   strcat( &command[0], PublicScreenName );

   /*
   ** or perhaps you prefer:

      sprintf( command, "ToolTypesEditor %s %s",
                        IconFileName, PublicScreenName
             );
```

```
    */
    // System() is an AmigaDOS call (but you knew that didn't you?).

    if (System( &command[0], TAG_DONE ) < 0)
        {
        // Tell the User about an error condition:

        sprintf( ErrMsg,
                  "%s\n couldn't be run by the System,\n"
                  "check your spelling!", command
               );

        // My personal EasyRequest() call:
        UserInfo( ErrMsg, "Invalid Icon Name?" );
        }
}
```

```
If anyone would care to send
                me
                an example of how to use LoadSeg()
so that ToolTypesEditor doesn't have to be external to a program,
I'll be happy to send that person a copy of the source code for
ToolTypesEditor.
```

1.4   ToolTypes recognized by ToolTypesEditor:

```
ToolTypes used by ToolTypesEditor:

     DEFAULTDIR    = System:Tools (Default)

  DefaultDir specifies where to begin looking for icons.

   PUBLICSCREEN   = Workbench     (Default)

  PublicScreen tells the program which screen to open its GUI onto.
```

1.5   tooltypeslv

```
                When you select a ToolType from this ListView Gadget, it gets  ←
                   written
into the
                ToolType
                string Gadget below.  You may change the ToolType
there, but it will not take effect unless you save the ToolTypes.

Any changes made to the ToolTypes won't be sent to the Icon until the

                Save
                Button is pressed & released.
```

1.6 tooltypestring

When you select a ToolType from the
ListView Gadget
, it gets written
into this string Gadget. You may change the ToolType here, but it
will not change in the Icon until you press & release the
Save
Button.

1.7 Keys that ToolTypes Editor recognizes:

s or S: Same as pressing the
SAVE
Button.

r or R: Same as pressing the
RESTORE
Button.

c or C: Same as pressing the
CANCEL
Button.

i or I: Shows an Information requester.

1.8 Cancel Button:

Pressing & releasing this button will exit the program without making
any changes to the Icon ToolTypes.

1.9 Save Button:

Pressing & releasing this button will exit the program after saving
all of your changes to the Icon ToolTypes.

1.10 Restore Button:

Pressing & releasing this button will undo all changes that you
made to the ToolTypes. The other two buttons will exit the program.

1.11   File Name string Gadget:

                         Either enter the Icon name you wish to change/view in this ←
                      gadget, or
 press the
                ASL file requester
                button to the right in order to select an
 Icon name from the filing system. ToolTypesEditor is smart enough
 to remove the .info from the end of your selection so the program won't
 crash.

1.12   aslbutton

                         Either press this button in order in order to select an Icon ←
                      name
 from the filing system or enter the Icon name you wish to change/view in
 the
                File Name
                string Gadget.

1.13   The Author (what a hoopy guy!):

 If your conscience is bugging you to contribute some cash for my
 programming efforts, please send $5 (US) or more for the program to:

 The Author (a real cool frood!):

 James T. Steichen
 2217 N. Tamarack Dr.
 Slayton, Mn. 56172-1155 (USA)

 email: jimbot@rconnect.com   (best place to send bug reports)

 ToolTypesEditor was written using SAS-C V6.58.

 The GUI was put together with GadToolsBox V37.300 (14.3.93) (C)
 1991-1993 by Jaba Development. I wish that there was an easy way to pay
 for this godsend of a program (Thanx guys!).