in

Georg Steger

COLLABORATORS			
	TITLE : in		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Georg Steger	January 19, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

in

1.1 ReqAttack Tools

```
                ReqAttack Tools - (C) 1999 by Georg Steger

                          RAIM_Convert
                          RAIM_SetAttributes
                          RAIM_GetInfo
                          RAIM_Anim2Brush
                          RAIM_Brush2Anim
                          RAIM_Anim2IFF

                          WinInfo
                          CyReq
                          QuitReqAttack
```

1.2 RAIM_Convert

With RAIM_Convert you can convert an IFF image or any other image format for which you have installed the corresponding Datatype to ReqAttack's own image format which is the only format supported by ReqAttack. The program can only be started from Shell and has the following argument template:

```
    SOURCE/A,DEST/A,FRAMEWIDTH=FW/N/K,FRAMEHEIGHT=FH/N/K,FRAMES=F/N/K,
    FRAMESPERROW=FPR/N/K,ANIMSPEED=AS/N/K,NOTRANSP=NT/S,DONTPACK=DP/S,
    PINGPONG=PP/S
```

Main arguments which are required for both brushes and animbrushes:

SOURCE: Name of source image file.

DEST : Name of destination image file.

Arguments which are additionally required to create an animbrush. Note that The source image must have all frames grouped together in a way like this:

```
    +---+---+---+---+---+
```

```
! A | B | C | D | E |
+---+---+---+---+---+
! F | G | H | I | J |
+---+---+---+---+---+
! K | L | M | N | O |
+---+---+---+---+---+
! P | Q |
+---+---+
```

The first frame (A) must start at pixel position (0,0). There must not be any space between the frames.

FRAMEWIDTH : Width of one frame in pixels.

FRAMEHEIGHT : Height of one frame in pixels.

FRAMES : Total number of frames. In the above example we have a total number of 17 frames.

FRAMESPERROW: Number of frames in one row. In the above example we have 5 frames per row. (The last row does not matter).

ANIMSPEED : Delay in 1/50 seconds between frames. Can be later changed with RAIM_SetAttributes utility.

The following arguments are optional:

NOTRANSP: Switch transparency of color 0 off. Color 0 usually will not be remapped. By specifying NOTRANSP it will be remapped. For more information see RAIM_SetAttributes doc.

PINGPONG: Usually after having displayed the last frame of an animation, playback will resume from the first frame. With this option the playback direction is set to backwards when the last frame of the animation has been displayed and reset to forward when the first frame of the animation has been reached again. For more information see RAIM_SetAttributes doc.

DONTPACK: By default RAIM_Convert compresses the image data with a rather primitive and not very effective compression algorithm called ByteRun1, that is the same used by the IFF ILBM format. In some cases it may happen that the image files gets bigger with the compression than without, but RAIM_Convert does not autocheck this. With this option compression is turned off.

1.3 RAIM_SetAttributes

With RAIM_SetAttributes you can change some attributes of a [R]eq[A]ttack [Im]age file. The program can only be started from Shell and has the following argument template:

```
FILE/A,TRANSP=TR/S,NOTRANSP=NOTR/S,PINGPONG=PP/S,NOPINGPONG=NOPP/S,
ANIMSPEED=AS/N/K
```

FILE : Name of the image file.

TRANSP : Switches transparency of color 0 on. This means that no matter what palette color 0 has, it will not be remapped. So if for example the palette (RGB value) of color 0 of the image is black and the palette of color 0 on the screen on which the requester appears is gray then the parts of the image which are painted with color 0 will appear gray.

NOTRANSP : Switches transparency of color 0 off. Parts of the image painted with color 0 will be remapped with the color of the screen whose palette matches best the image's color 0 palette.

PINGPONG : Switches ping pong animation on. If for example you have a ReqAttack AnimBrush with 10 frames the animation will be played like this:

1 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1 2 3 4 5 6 7 8 9 10 9 8 7

NOPINGPONG: Switches ping pong animation off. After the last frame is shown the animation will restart from frame 1:

1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 1

ANIMSPEED : Delay in 1/50 seconds between frames.

1.4 RAIM_GetInfo

This utility can be used to get some information about the specified ReqAttack image file. The program can only be started from Shell and has the following argument template:

```
FILE/A,WIDTH=W/S,HEIGHT=H/S,FRAMEWIDTH=FW/S,FRAMEHEIGHT=FH/S,
FRAMES=FR/S,COLORS=CO/S,PLANES=PL/S,ANIMSPEED=AS/S,FLAGS=FL/S,
ANIMFLAGS=AF/S
```

If only the FILE argument is specified all information will be printed!

FILE : Name of the image file

WIDTH : Prints the width of the image. For ReqAttack AnimBruhes this is equivalent to FRAMEWIDTH x FRAMES!

HEIGHT : Prints the height of the image.

FRAMEWIDTH : Prints the width of one frame. If the image file is not a ReqAttack AnimBrush but a ReqAttack Brush then the image's width is returned.

FRAMEHEIGHT: Prints the height of one frame. This is always equivalent to HEIGHT.

FRAMES : Prints the number of frames if the file is a ReqAttack AnimBrush or "1" if the file is a ReqAttack Brush.

COLORS : Prints the number of colors of the image file.

PLANES : Prints the number of planes (bits per color) of the image file.

ANIMSPEED : Prints the animspeed (unit is 1/50 secs) or 0 if the image file is a ReqAttack Brush and not a ReqAttack AnimBrush.

FLAGS : Prints image flags (PACKED, NOTRANSPARENCY).

ANIMFLAGS : Prints anim flags (PINGPONG).

1.5 RAIM_Anim2Brush

With this utility you can convert a ReqAttack AnimBrush to a ReqAttack Brush. The brush will have all frames grouped together in one row. This program can only be used from Shell and has the following argument template:

FILE/A

FILE: Name of the image file.

You probably will never need this program as the ReqAttack AnimBrush Datatype has a special feature which allows you to disable it on the fly and let the ReqAttack Brush Datatype handle everything. All you need to do is hold down LEFT ALT + LEFT SHIFT + CONTROL while the image is loaded (by MultiView or some other program which uses DataTypes to load pictures). Unfortunately this will not work from a Shell because of it's special behaviour when the qualifiers (keys) mentioned above are hold down while pressing RETURN.

With programs which only load picture DataTypes you should not have any problems at all anyway as such programs (should) ignore anim Datatypes automatically unless they first check the datatypes GID (group ID).

1.6 RAIM_Brush2Anim

With this utility you can convert a ReqAttack image file which was earlier converted from a ReqAttack AnimBrush to a ReqAttack Brush (with the utility RAIM_Anim2Brush) back to a ReqAttack AnimBrush. This program can only be used from Shell and has the following argument template:

FILE/A

FILE: Name of the image file.

1.7 RAIM_Anim2IFF

With this utility you can either convert single frames of a ReqAttack AnimBrush to IFF images or all frames of the ReqAttack AnimBrush to one IFF containing all frames. This program can only be started from Shell and has the

following argument template:

```
SOURCE/A,DEST/A,FRAMESPERROW=FPR/N/K,STARTFRAME=SF/N/K,ENDFRAME=EF/N/K,
FRAME=FR/N/K,ALLFRAMES=AF/S,VERBOSE=V/S
```

SOURCE : Name of the ReqAttack AnimBrush file.

DEST : Name of destination IFF file(s) to be created. If you want to convert several frames into single IFF files you must write "%ld" somewhere in the filename. This will be replaced by the corresponding frame number. For example "Frame%ld.iff" will result in the following filenames:

```
"Frame1.iff"
"Frame2.iff"
"Frame3.iff"
...
"Frame10.iff"
"Frame11.iff"
...
```

Instead of "%ld" you can also write "%02ld", "%03ld", "%04ld", ... to get leading zeros:

"Frame%02ld.iff"	"Frame%03ld.iff"	"Frame%04ld.iff"
"Frame01.iff"	"Frame001,iff"	"Frame0001.iff"
"Frame02.iff"	"Frame002.iff"	"Frame0002.iff"
"Frame03.iff"	"Frame003.iff"	"Frame0003.iff"
...	...	...
"Frame10.iff"	"Frame010.iff"	"Frame0010.iff"
...	...	...

FRAMESPERROW: If you want to convert the ReqAttack AnimBrush into one IFF image containing all frames you can specify the number of frames you want to have per row in the IFF image with this option. By default RAIM_Anim2IFF tries to calculate a "good" FRAMESPERROW value, that is a value that gives about the same number of frames per row and frames per column.

STARTFRAME : If you want to convert single frames into IFF files use this option to specify the start frame (default = 1 = first frame).

ENDFRAME : With this option you can specify the last frame to be converted into an IFF file (default = last frame).

FRAME : Same as specifying the same value for STARTFRAME end ENDFRAME.

ALLFRAMES : Same as specifying 1 for STARTFRAME and <NUMBER OF FRAMES> for ENDFRAME.

VERBOSE : Verbose output after each converted frame.

By specifying any of STARTFRAME, ENDFRAME, FRAME or ALLFRAMES the utility is switched into "single frame mode", that is one IFF file per frame will be

generated. If you don't specify any of this options the program will convert all frames into one IFF file. Examples:

```
MyImage is a ReqAttack AnimBrush with 10 frames.

RAIM_Anim2IFF MyImage MyImage.iff
   ... groups together all 10 frames in the IFF image "MyImage.iff".

RAIM_Anim2IFF MyImage MyImage%ld.iff ALLFRAMES
   ... creates 10 IFF images (one per frame) called "MyImage1.iff",
   "MyImage2.iff", ..., "MyImage10.iff".

RAIM_Anim2IFF MyImage MyImage%ld.iff STARTFRAME 5 ENDFRAME 8
   ... creates 4 IFF images from the frames 5 - 8 called "MyImage5.iff",
   "MyImage6.iff", "MyImage7.iff", "MyImage8.iff".

RAIM_Anim2IFF MyImage MyImage.iff STARTFRAME 6 ENDFRAME 6
   ... creates an IFF image called "MyImage.iff" from the frame number
   6.

RAIM_Anim2IFF MyImage MyImage.iff FRAME 6
   ... does exactly the same.

RAIM_Anim2IFF MyImage MyImage%ld.iff ENDFRAME 3
   ... creates 3 IFF images from the frames 1 - 3 called "MyImage1.iff",
   "MyImage2.iff", "MyImage3.iff".

RAIM_Anim2IFF MyImage MyImage%02ld.iff STARTFRAME 8
   ... creates 3 IFF images from the frames 8 - 10 called
   "MyImage08.iff", "MyImage09.iff", "MyImage10.iff".
```

1.8 WinInfo

With this tool you can find out which task/process a certain window (for instance a requester) belongs to. In most cases this equals the task/process that opened the window. However there are some hacks/commodities that change the UserPort of some windows. Then it could be, that the task/process names shown by WinInfo look strange. For example WinInfo will say "ToolsDaemon" for Workbench Windows instead of "Workbench" if ToolsDaemon is running. Another problem is windows that do not have a userport, because they do not want to get any IDCMP messages. Then WinInfo can't find out to whom the window belongs. Anyway, with requester windows these problems do not exist and therefore WinInfo can be of big help if for example you want to give requesters of certain programs a special look by using according conditions in the ReqAttack configuration file.

Although it is possible to start WinInfo directly from a Shell, it is better to use a commodity which allows starting external programs through function keys or hotkeys. Once started you can move the mouse over any window and the window will be hilighted by a scolling rectangle. An Info Box will appear as well. To quit WinInfo it is enough to press the left mouse button. The argument template of WinInfo looks like this:

```
FPS/N/K,HIDDENAREA=HA/N/K,LINEPATTERN1=LP1/K,LINEPATTERN2=LP2/K,
LINECOL1=LC1/K,LINECOL2=LC2/K,HLINECOL1=HLC1/K,HLINECOL2=HLC2/K,
```

```
    INFOBGCOL1=IBGC1/K,INFOBGCOL2=IBGC2/K,LABELCOL=LC/K,TEXTCOL=TC/K,
    INFOBORCOL=IBC/K,INFOTITLECOL=ITK/K

FPS         : Speed in frames per second at which the rectangle line pattern
              will be scrolled.

HIDDENAREA  : Look of the scrolling rectangle  in hidden areas of the window
              under the mouse pointer.

                  0: Invisible
                  1: Visible
                  2: Visible but with alternative colors/line pattern

LINEPATTERN1: Normale 16 pixel line pattern (0xFF00 = 1111111100000000).

LINEPATTERN2: Alternative 16 pixel line pattern (for hidden areas).

LINECOL1    : Color 1 in normal line pattern (where bits are set).

LINECOL2    : Color 2 in normal line pattern (where bits are cleared).

HLINECOL1   : Color 1 in alternative line pattern.

HLINECOL2   : Color 2 in alternative line pattern.

INFOBGCOL1  : Background color 1 of the info box.

INFOBGCOL2  : Background color 2 of the info box.

LABELCOL    : Color of the label texts.

TEXTCOL     : Color of task/process names.

INFOBORCOL  : Color of the info box border.

INFOTITLECOL: Color of the info box title.

For arguments which expect a color you can either specify a 24 Bit RGB value
(best in HEX format,  for example 0xFFFF00 = yellow)  or a DrawInfo pen name
(SHINEPEN, SHADOWPEN, FILLPEN, FILLTEXTPEN, TEXTPEN, ...).
```

1.9 CyReq

```
CyReq redirects EasyRequestArgs() to:

  BuildEasyRequestArgs()
  SysReqHandler()
  FreeSysRequest()

and AutoRequest() to:

  BuildSysRequest()
  SysReqHandler()
  FreeSysRequest()
```

This is done because of EasyRequestArgs() and AutoRequest() calling FreeSys-Request() directly instead of using the LVO. Therefore one cannot catch calls to FreeSysRequest (which calls CloseWindow(), etc.). With CyReq you can.

It is possible to build the CyReq functionality into the patching program (ReqAttack, PowerWindows) but then there's the problem that certain other programs which replace or improve requester won't work anymore, unless these programs install their patches (usually a EasyRequestArgs() patch) after ReqAttack, PowerWindows, ...

By putting CyReq into the Startup-Sequence somewhere soon after the SetPatch line the redirection patches are done there and don't need to be done by the other programs (ReqAttack, PowerWindows) which either detect CyReq automatically (ReqAttack) or are instructed to avoid installing the requester redirection patches a second time by a startup option.

Once CyReq is installed it should not matter when ReqAttack, PowerWindows, ... are started. AssignWedge and similiar programs should keep working no matter if started before or after ReqAttack, PowerWindows, ...

It is strongly suggested to install CyReq, also because of CyReq using stack swapping in it's EasyRequestArgs() patch which should prevent crashes that on some systems happen after replying "Please insert volume XYZ ..." requesters. For more info see ReqAttack History of V 1.0!

CyReq can be installed with the ReqAttack installation script.

1.10 QuitReqAttack

Sends ReqAttack a CONTROL C break, which tells ReqAttack to (try to) quit. If ReqAttack is not running this Shell program returns WARN (5) as primary result code ($RC).