iffpack

COLLABORATORS			
	TITLE : iffpack		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 23, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 iffpack **1**

1.1 contents 1

1.2 iffcleanup 1

1.3 readbody 1

1.4 readpicsize 2

1.5 setcolors 3

1.6 writewindow 4

1.7 newcomp 4

Chapter 1

iffpack

1.1 contents

```
                TABLE OF CONTENTS
```

```
                 IFFCleanup

                 ReadBody

                 ReadPicSize

                 SetColors

                 WriteWindow

                 New compression format
```

1.2 iffcleanup

```
IFFCleanup                                                        IFFCleanup

   NAME
        IFFCleanup -- Clean up all allocated memory

   SYNOPSIS
        void IFFCleanup(void);

   FUNCTION
        Frees all memory that was allocated by ReadBody, ReadPicSize or
        WriteWindow
```

1.3 readbody

```
                ReadBody  ←
                                                                    ←
                   ReadBody

   NAME
        ReadBody -- Read the bitmap data (i.e. the BODY-Chunk)

   SYNOPSIS
        error = ReadBody(rp,fp);

        int ReadBody(struct RastPort *,FILE *);

   FUNCTION
        This fucntion reads the bitmap data, after the size of the picture
        has been read with ReadPicSize.

   INPUTS
        rp - pointer to the RastPort structure, where the picture will be
             loaded to
        fp - standard C filepointer


   RESULT
        errorcode:  NO_ERROR            0
                    NO_MEMORY           1
                    BAD_IFF             2
                    READ_ERROR          3
                    UNKNOWN_COMPRESSION 5

   SEE ALSO

                 ReadPicSize()
                ,
                 SetColors()
```

1.4 readpicsize

```
                ReadPicSize                                           ←
                   ReadPicSize

   NAME
        ReadPicSize -- Read the size of the picture

   SYNOPSIS
        error = ReadPicSize(fp,win_width,win_height,
                                       scr_width,scr_height,depth,viewmode);

        int ReadPicSize(FILE *,SHORT *,SHORT *,
                                       SHORT *,SHORT *,SHORT *,USHORT *);

   FUNCTION
        This fucntion reads the size of the picture and stores the values
        in the variables the pointers point to. Furthermore it reads the
        colortable of the picture.
```

```
INPUTS
     fp                    - standard C filepointer
     win_width,win_height - pointers to the variables for width and height
                            of the window
     scr_width,scr_height - pointers to the variables for width and height
                            of the screen
     depth                 - pointer to the variable for the depth of the
                            screen
     viewmode              - pointer to the variable for the viewmode
                            (<-> CAMG-Chunk)


RESULT
     errorcode:  NO_ERROR            0
                 NO_MEMORY           1
                 BAD_IFF             2
                 READ_ERROR          3
                 UNKNOWN_COMPRESSION 5

SEE ALSO

             ReadBody()
            ,
             SetColors()
```

1.5 setcolors

```
            SetColors  ↩
                                                                     ↩
               SetColors

NAME
     SetColors -- Set the colors, that has been read by ReadPicSize

SYNOPSIS
     SetColors(vp);

     void SetColors(struct ViewPort *);

FUNCTION
     This fucntion sets the colors, that has been read by ReadPicSize()
     in the specified ViewPort.

INPUTS
     vp - pointer to the ViewPort structure of the screen, where the
          colors will be set.

SEE ALSO

             ReadPicSize()
            ,
             ReadBody()
```

1.6 writewindow

```
                WriteWindow                                                    ←
                   WriteWindow

   NAME
        WriteWindow -- Write the contents of a window as an ILBM-Picture

   SYNOPSIS
        error = WriteWindow(fp,window,mode,read_colors);

        int WriteWindow(FILE *,struct Window *,int,int);

   FUNCTION
        This functions writes the contents of a window as an ILBM-Picture.
        It supports three different compression modes:

           mode:

              0  - no compression
              1  - normal horizontal byte compare run compression
              2  - new vertical byte compare run compression

   INPUTS
        fp          - standard C filepointer
        window      - pointer to the Window structure of the Window that has
                      to be written as an ILBM-Picture
        mode        - compression mode
        read_colors - specifies if the colors that have been read by the
                      last ReadPicSize will be used for saving.

                      FALSE: Read the colors from the colortable of the
                             screen
                      TRUE:  use the old color-values

                      This enables to keep the full 24-bit palette when
                      modifying pictures on non-AGA machines.

   RESULT
        errorcode:  NO_ERROR            0
                    NO_MEMORY           1
                    BAD_IFF             2
                    WRITE_ERROR         4
                    UNKNOWN_COMPRESSION 5

   SEE ALSO

                 ReadPicSize()
                ,
                 New compression format
```

1.7 newcomp

```
New  compression format                             New compression format
```

```
The new compression method does not compress row by row but column by
column. Each column has a width of 1 byte (8 pixel). First the 1st
column of the 1st bitplane is compressed, then the 1st column of the
2nd bitplane and so on.
The new compression format is marked with 2 in the compression field
of BMHD. The new format is very similar to the old compression format.
It is as follows:
   The first byte v is a control byte. The values have the following
   meaning:
      v>=0:   v+1 uncompressed bytes are following
      v<0:    the next byte is repeated -v+1 times
      v=-128: The next unsigned (!) byte is the number of bytes that
              has to be copied from the column left of this column of
              the same bitplane.

   NOTE: never compress across the borders of the columns, like it was
         not allowed in the old format to compress across the rows.
```