intuisup

COLLABORATORS

	TITLE : intuisup		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		March 1, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

intuisup

1.1 Intuition Support Library Doc

Table of Contents:

Author:
  Torsten Jürgeleit
  Am Sandberg 4
  W-5270 Gummersbach
  Germany
  Phone: ++49 2261 27400
  EMail: mike@miba.obb.sub.org

Thanks to:
  Michael Bjerking - for helping with this document and the new
        vewrsion of the template editor
  Johann Semsrott - for the Modula II support
  Mauricio Hunt Rosales - for the Benchmark Modula II support
  Gilles Andre - for helping with DICE support
  Mika Saastamoinen - for reporting some bugs

  and all other who supporting IntuiSup

1.2 IntuiSup/Intro

This is a quick reference guide for the Intuition Support Library (ISUP).

ISUP is a shared library so you have to make a call to 'OpenLibrary' before using the Library in your code. Also you have to include the header file 'intuisup.h'.

```
We have made an
                index
                 table to all function, to get easy access to the
description of the functions.

Last the
                Data Structures and Defines
                 are also describe.
```

1.3 IntuiSup/Library Functions

1.4 IntuiSup/IGetRenderInfo

```
                NAME
  IGetRenderInfo

SYNOPSIS
  ri = IGetRenderInfo( screen, flags )
  d0                   a0      d0
  APTR IGetRenderInfo( struct Screen *, USHORT );

FUNCTION
  Returns a pointer to an internal data structure with some visual
  infos needed for creation of ISUP objects. Use the SAME pointer
  (only ONE call of IGetRenderInfo for all ISUP objects displayed on
  the SAME screen) as parameter APTR ri to the ISUP functions. This
  data structure MUST be released with the function
                IFreeRenderInfo
                  before closing library.

INPUTS
  screen - ptr to screen the ISUP object should be displayed on or
     NULL for the workbench screen

  flags -
                RENDER_INFO_FLAG_xxx
                RETURNS
  ri - pointer to an internal data structure or NULL if function
       failed

SEE ALSO

                IFreeRenderInfo
```

1.5 IntuiSup/IFreeRenderInfo

```
                NAME
  IFreeRenderInfo

SYNOPSIS
  IFreeRenderInfo( ri )
                   a0

  VOID IFreeRenderInfo( APTR );

FUNCTION
  Releases memory for internal data structure allocated by

                IGetRenderInfo
                .

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                RETURNS
  VOID

SEE ALSO

                IGetRenderInfo
```

1.6 IntuiSup/IOpenWindow

```
                NAME
  IOpenWindow

SYNOPSIS
  win = IOpenWindow( ri, nw, flags )
  d0                 a0  a1  d0

  struct Window *IOpenWindow( APTR, struct NewWindow *, USHORT );

FUNCTION
  Manipulates NewWindow structure according to given flags and opens
  window from it.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  nw - pointer to initialized NewWindow structure

  flags -
                OPEN_WINDOW_FLAG_xxx
                RETURNS
  win - pointer to standard window structure or NULL if function
        failed

SEE ALSO

                IClearWindow
                ,
                ICloseWindow
```

1.7 IntuiSup/IClearWindow

```
                NAME
  IClearWindow

SYNOPSIS
  IClearWindow( ri, win, leftedge, topedge, width, height, flags )
                a0  a1   d0        d1       d2     d3      d4

  VOID IClearWindow( APTR, struct Window *, USHORT, USHORT, USHORT,
         USHORT, USHORT );

FUNCTION
  Clears area of given window according to visual infos (APTR ri). The
  area will be clipped to the window dimension if necessary.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  win - pointer to window opened by
                (I)OpenWindow
                  leftedge, topedge, width, height - dimension of area to clear
  flags -
                CLEAR_WINDOW_FLAG_xxx
                RETURNS
  VOID

SEE ALSO

                IOpenWindow
                ,
                ICloseWindow
```

1.8 IntuiSup/ICloseWindow

```
                NAME
  ICloseWindow

SYNOPSIS
  ICloseWindow( win, morewindows )
                a0   d0

  VOID ICloseWindow( struct Window *, BOOL );

FUNCTION
  Closes window in a savely manner (all IntuiMessages are replyed,
  user port is only closed if not shared [morewindows == FALSE], ...).

INPUTS
  win - pointer to window opened by
                (I)OpenWindow
                  morewindows - TRUE if user port of window shared with other  ↩
                      windows

RETURNS
  VOID

SEE ALSO

                IOpenWindow
                ,
                IClearWindow
```

1.9 IntuiSup/IAvailFonts

```
                NAME
  IAvailFonts

SYNOPSIS
  afh = IAvailFonts( ri )
                    a0

  struct AvailFontsHeader *IAvailFonts( APTR );
```

```
FUNCTION
  Creates a list of all available fonts (ROM + disk) and saves this in
  the internal data structure (APTR ri).

INPUTS
  ri - pointer to internal data structure returned by
              IGetRenderInfo
              RETURNS
  afh - pointer to initialized structure with available fonts

SEE ALSO

              IAskFont
              ,
              IOpenFont
```

1.10 IntuiSup/IAskFont

```
              NAME
  IAskFont

SYNOPSIS
  ta = IAskFont( ri, ta )
  d0             a0  a1

  struct TextAttr *IAskFont( APTR, struct TextAttr * );

FUNCTION
  Check if given font exists in internal font list (APTR ri) created
  with IAvailFonts.

INPUTS
  ri - pointer to internal data structure returned by
              IGetRenderInfo
                ta - pointer to initialized structure with text attributes

RETURNS
  ta - pointer to initialized structure with text attributes
       describing the most fitting font for given TextAttr structure.

SEE ALSO

              IAvailFonts
              ,
              IOpenFont
```

1.11 IntuiSup/IOpenFont

```
              NAME
  IOpenFont
```

```
SYNOPSIS
  tf = IOpenFont( ri, ta )
  d0              a0  a1

  struct TextFont  *IOpenFont( APTR, struct TextAttr * );

FUNCTION
  Open font according to given TextAttr structure from internal font
  list (APTR ri).

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  ta - pointer to initialized structure with text attributes

RETURNS
  tf - pointer to initialized structure with font data

SEE ALSO

                IAvailFonts
                ,
                IAskFont
```

1.12 IntuiSup/IDisplayTexts

```
                NAME
  IDisplayTexts

SYNOPSIS
  IDisplayTexts( ri, win, td, hoffset, voffset, languagetextarray )
                 a0  a1   a2  d0       d1       a3

  VOID IDisplayTexts( APTR, struct Window *, struct TextData *, SHORT,
          SHORT, BYTE ** );

FUNCTION
  Displays texts described in given data structure array. If given
  language_text_array is not NULL then td_Text doesn't contain a
  pointer to string but an offset of the string pointer in given array
  with pointers to text in foreign languages.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  win - pointer to window opened by
                (I)OpenWindow
                  td - pointer to ARRAY of initialized
                TextData
                 structures

  hoffset, voffset - offsets added to positions of ALL texts

  languagetextarray - pointer to string pointer array created with
```

```
                IBuildLanguageTextArray
                 or NULL

RETURNS
  VOID

SEE ALSO

                IPrintText
```

1.13 IntuiSup/IPrintText

```
                NAME
  IPrintText

SYNOPSIS
  len = IPrintText( ri, win, text, leftedge, topedge, type, flags,
  d0                a0  a1   a2    d0        d1       d2    d3
        textattr )
        a3

  USHORT IPrintText( APTR, struct Window *, BYTE *, USHORT, USHORT,
         USHORT, USHORT, struct TextAttr * );

FUNCTION
  Displays text at given position and returns width of printed text in
  pixels. With (flags &
                TEXT_DATA_FLAG_NO_PRINT
                ) no text will be
  printed but only the width calculated.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  win - pointer to window opened by
                (I)OpenWindow
                  text - pointer to string with text

  leftedge, topedge - position of text in window

  type -
                TEXT_DATA_TYPE_xxx
                  flags -
                TEXT_DATA_FLAG_xxx
                  textattr - font used for text

RETURNS
  len - width of printed text in pixels

SEE ALSO

                IDisplayTexts
```

1.14 IntuiSup/IConvertUnsignedDec

```
              NAME
  IConvertUnsignedDec

SYNOPSIS
  len = IConvertUnsignedDec( num, buffer, flags )
  d0                         d0   a0      d1

  USHORT IConvertUnsignedDec( ULONG, BYTE *, USHORT );

FUNCTION
  Converts binary number to text string in unsigned decimal format and
  and returns length of result string.

INPUTS
  num - number to be converted

  buffer - pointer to buffer for converted number

  flags -
              CONVERT_FLAG_xxx
              RETURNS
  len - length of text string

SEE ALSO

              IConvertSignedDec
              ,
              IConvertHex
              ,
              IConvertBin
```

1.15 IntuiSup/IConvertSignedDec

```
              NAME
  IConvertSignedDec

SYNOPSIS
  len = IConvertSignedDec( num, buffer, flags )
  d0                       d0   a0      d1

  USHORT IConvertSignedDec( LONG, BYTE *, USHORT );

FUNCTION
  Converts binary number to text string in signed decimal format and
  and returns length of result string.

INPUTS
  num - number to be converted

  buffer - pointer to buffer for converted number

  flags -
```

```
                CONVERT_FLAG_xxx
                RETURNS
  len - length of text string

SEE ALSO

                IConvertUnsignedDec
                ,
                IConvertHex
                ,
                IConvertBin
```

1.16 IntuiSup/IConvertHex

```
                NAME
  IConvertHex

SYNOPSIS
  result =   IConvertHex( num, buffer, flags )
  d0                      d0   a0      d1

  USHORT IConvertHex( ULONG, BYTE *, USHORT );

FUNCTION
  Converts binary number to text string in hexa decimal format and
  and returns length of result string.

INPUTS
  num - number to be converted

  buffer - pointer to buffer for converted number

  flags -
                CONVERT_FLAG_xxx
                RETURNS
  len - length of text string

SEE ALSO

                IConvertUnsignedDec
                ,
                IConvertSignedDec
                ,
                IConvertBin
```

1.17 IntuiSup/IConvertBin

```
                NAME
  IConvertBin

SYNOPSIS
  result = IConvertBin( num, buffer, flags )
  d0                    d0   a0      d1

  USHORT IConvertBin( ULONG, BYTE *, USHORT );
```

FUNCTION
```
  Converts binary number to text string in binary format and returns
  length of result string.
```

INPUTS
```
  num - number to be converted

  buffer - pointer to buffer for converted number

  flags -
              CONVERT_FLAG_xxx
              RETURNS
  len - length of text string
```

SEE ALSO

```
              IConvertUnsignedDec
              ,
              IConvertSignedDec
              ,
              IConvertHex
```

1.18 IntuiSup/IDisplayBorders

```
              NAME
  IDisplayBorders
```

SYNOPSIS
```
  IDisplayBorders( ri, win, bd, hoffset, voffset )
                   a0  a1   a2  d0       d1

  VOID IDisplayBorders( APTR, struct Window *, struct BorderData *,
            SHORT, SHORT );
```

FUNCTION
```
  Displays borders described in given data structure array.
```

INPUTS
```
  ri - pointer to internal data structure returned by
              IGetRenderInfo
                win - pointer to window opened by
              (I)OpenWindow
                bd - pointer to ARRAY of initialized
              BorderData
               structures

  hoffset, voffset - offsets added to positions of ALL texts
```

RETURNS
```
  VOID
```

```
SEE ALSO

                IDrawBorder
```

1.19 IntuiSup/IDrawBorder

```
                NAME
  IDrawBorder

SYNOPSIS
  IDrawBorder( ri, win, leftedge, topedge, width, height, type )
               a0  a1   d0        d1       d2     d3      d4

  VOID IDrawBorder( APTR, struct Window *, USHORT, USHORT, USHORT,
        USHORT, USHORT );

FUNCTION
  Draws border at given position on display.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  win - pointer to window opened by
                (I)OpenWindow
                  leftedge, topedge, width, height - dimension of border

  type -
                BORDER_DATA_TYPE_xxx
                RETURNS
  VOID

SEE ALSO

                IDisplayBorders
```

1.20 IntuiSup/ICreateGadgets

```
                NAME
  ICreateGadgets

SYNOPSIS
  gl = ICreateGadgets( ri, gd, hoffset, voffset, languagetextarray )
  d0                   a0  a1  d0       d1       a2

  APTR ICreateGadgets( APTR, struct GadgetData *, SHORT, SHORT,
           BYTE ** );

FUNCTION
  Create internal data structure for ISUP gadgets from given array of
  data structures. This function DON'T display any object. Displaying
  will be done with
                IDisplayGadgets
                . Internal data structure MUST be
  released with
                IFreeGadgets
                . If given language_text_array are not
  NULL then gd_Text doesn’t contain a pointer to string but an offset
  of the string pointer in given array with pointers to text in
  foreign languages.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  gd - pointer to ARRAY of initialized
                GadgetData
                 structures

  hoffset, voffset - offsets added to positions of ALL texts

  languagetextarray - pointer to string pointer array created with

                IBuildLanguageTextArray
                 or NULL

RETURNS
  gl - pointer to internal data structure (GadgetList) or NULL if
       function failed

SEE ALSO

                IFreeGadgets
                ,
                IDisplayGadgets
                ,
                IRefreshGadgets
                ,
                IModifyGadget
                ,

                ISetGadgetAttributes
                ,
                IActivateInputGadget
                ,
                IGadgetAddress
                ,

                IRemoveGadgets
```

1.21 IntuiSup/IFreeGadgets

```
                NAME
  IFreeGadgets

SYNOPSIS
  IFreeGadgets( gl )
                a0
  VOID IFreeGadgets( APTR );

FUNCTION
  Releases memory for internal data structure allocated by

                ICreateGadgets
                . If gadgets are currently displayed then they MUST be
  removed with
                IRemoveGadgets
                 first.

INPUTS
  gl - pointer to internal data structure returned by
                ICreateGadgets
                RETURNS
  VOID

SEE ALSO

                ICreateGadgets
                ,
                IDisplayGadgets
                ,
                IRefreshGadgets
                ,
                IModifyGadget
                ,

                ISetGadgetAttributes
                ,
                IActivateInputGadget
                ,
                IGadgetAddress
                ,

                IRemoveGadgets
```

1.22 IntuiSup/IDisplayGadgets

```
                NAME
  IDisplayGadgets

SYNOPSIS
  IDisplayGadgets( win, gl )
                   a0   a1

  VOID IDisplayGadgets( struct Window *, APTR );

FUNCTION
  Displays all gadgets from given internal data structure (gadgets now
  are selectable). To remove gadgets from display use
                IRemoveGadgets
                  (gadgets aren't selectable any more).

INPUTS
```

```
  win - pointer to window opened by
                (I)OpenWindow
                  gl - pointer to internal data structure returned by
                ICreateGadgets
                RETURNS
  VOID

SEE ALSO

                ICreateGadgets
                ,
                IFreeGadgets
                ,
                IRefreshGadgets
                ,
                IModifyGadget
                ,

                ISetGadgetAttributes
                ,
                IActivateInputGadget
                ,
                IGadgetAddress
                ,

                IRemoveGadgets
```

1.23 IntuiSup/IRefreshGadgets

```
                NAME
  IRefreshGadgets

SYNOPSIS
  IRefreshGadgets( gl )
                   a0

  VOID IRefreshGadgets( APTR );

FUNCTION
  Refresh images of ALL gadgets from given internal data structure.
  ONLY NEEDED FOR WINDOWREFRESH OR NEWSIZE IDCMP EVENTS.

INPUTS
  gl - pointer to internal data structure returned by
                ICreateGadgets
                RETURNS
  VOID

SEE ALSO

                ICreateGadgets
                ,
                IFreeGadgets
                ,
                IDisplayGadgets
                ,
                IModifyGadget
                ,

                ISetGadgetAttributes
                ,
                IActivateInputGadget
                ,
                IGadgetAddress
                ,

                IRemoveGadgets
                EXAMPLE
  struct IntuiMessage *im;

  while (im = IGetMsg(win->UserPort)) {
     switch (im->Class) {
        case REFRESHWINDOW :
        case NEWSIZE :
      BeginRefresh(win);

      /* Refresh ISUP gadget list(s) displayed on this window */
      IRefreshGadgets(gl1);
      IRefreshGadgets(gl2);
         :
      IRefreshGadgets(gln);

      /* Custom window refresh */
         :
      EndRefresh(win);
      break;

        /* Handle other IDCMP events */
         :
     }
     IReplyMsg(im);
  }
```

1.24 IntuiSup/IModifyGadget

```
                NAME
  IModifyGadget

SYNOPSIS
  IModifyGadget( gl, dataentry, leftedge, topedge, width, height )
                 a0  d0         d1        d2       d3     d4

  VOID IModifyGadget( APTR, USHORT, LONG, LONG, ULONG, ULONG );

FUNCTION
  Repositions and/or resizes a gadget. All gadgets can be repositioned
  but only some
                gadgets
                 can be modified: buttons, sliders and
  scrollers. NO ADDITIONAL REFRESH NEEDED.
```

```
INPUTS
  gl - pointer to internal data structure returned by
                ICreateGadgets
                  dataentry - offset (in array of
                GadgetData structures
                ) of gadget to modify

  leftedge, topedge, width, height - new dimension of gadget or

                USE_CURRENT_VALUE
                 for old value

RETURNS
  VOID

SEE ALSO

                ICreateGadgets
                ,
                IFreeGadgets
                ,
                IDisplayGadgets
                ,
                IRefreshGadgets
                ,

                ISetGadgetAttributes
                ,
                IActivateInputGadget
                ,
                IGadgetAddress
                ,

                IRemoveGadgets
```

1.25 IntuiSup/ISetGadgetAttributes

```
                NAME
  ISetGadgetAttributes

SYNOPSIS
  old_value = ISetGadgetAttributes( gl, dataentry, flagmask, flagbits,
  d0                                a0  d0         d1        d2
            data1, data2, data3 )
            d3     d4     a1

  ULONG ISetGadgetAttributes( APTR, USHORT, ULONG, ULONG, ULONG,
            ULONG, VOID * );

FUNCTION
  Changes flags or special data of a gadget. Not all special data
  members can be changed of different gadgets. Some are fixed while
  creating. It returns the old value of a gadget, so with data1..data2
  set to
```

```
                USE_CURRENT_VALUE
                 you can get the actual value.
  NO ADDITIONAL REFRESH NEEDED.

INPUTS
  gl - pointer to internal data structure returned by
                ICreateGadgets
                  dataentry - offset (in array of
                GadgetData structures
                ) of gadget to
        change attributes

  flagmask - mask with bits set for flag bits to change

  flagbits - new flag bits (only bits with flag mask bit set are
       changed)

  data1, data2, data3 - new values for appropriate union

                gd_SpecialData
                 or
                USE_CURENT_VALUE
                 for old
            value

RETURNS
  ULONG old_value - old value of gadget

SEE ALSO

                ICreateGadgets
                ,
                IFreeGadgets
                ,
                IDisplayGadgets
                ,
                IRefreshGadgets
                ,

                IModifyGadget
                ,
                IActivateInputGadget
                ,
                IGadgetAddress
                ,
                IRemoveGadgets
                EXAMPLE
  To disable a gadget:

     ISetGadgetAttributes(<gl>, <dataentry>, GADGET_DATA_FLAG_DISABLED,
       GADGET_DATA_FLAG_DISABLED, USE_CURRENT_VALUE,
       USE_CURRENT_VALUE, (VOID *)USE_CURRENT_VALUE);

  To enable a gadget:

     ISetGadgetAttributes(<gl>, <dataentry>, GADGET_DATA_FLAG_DISABLED,
       0, USE_CURRENT_VALUE, USE_CURRENT_VALUE,
       (VOID *)USE_CURRENT_VALUE);

  To change the contents of an input gadget buffer to "Test":

     ISetGadgetAttributes(<gl>, <dataentry>, 0, 0, USE_CURRENT_VALUE,
      USE_CURRENT_VALUE, "Test");
```

1.26 IntuiSup/IActivateInputGadget

```
              NAME
  IActivateInputGadget

SYNOPSIS
  IActivateInputGadget( gl, dataentry )
                        a0  d0

  VOID IActivateInputGadget( APTR, USHORT );

FUNCTION
  Activates an input gadget (string or integer gadget).

INPUTS
  gl - pointer to internal data structure returned by
                ICreateGadgets
                  dataentry - offset (in array of
                GadgetData structures
                ) of gadget to
        activate

RETURNS
  VOID

SEE ALSO

                ICreateGadgets
                ,
                IFreeGadgets
                ,
                IDisplayGadgets
                ,
                IRefreshGadgets
                ,

                IModifyGadget
                ,
                ISetGadgetAttributes
                ,
                IGadgetAddress
                ,
                IRemoveGadgets
```

1.27 IntuiSup/IGadgetAddress

```
                NAME
  IGadgetAddress

SYNOPSIS
  gad = IGadgetAddress( gl, dataentry )
  d0                    a0  d0

  struct Gadget *IGadgetAddress( APTR, USHORT );

FUNCTION
  Returns pointer to the appropriate standard gadget structure. This
  function is normally not used, because no access to the standard
  gadget structures is required. All changes to ISUP objects MUST be
  performed via
                ISetGadgetAttributes
                .

INPUTS
  gl - pointer to internal data structure returned by
                ICreateGadgets
                  dataentry - offset (in array of
                GadgetData structures
                ) of gadget to
        get pointer of its standard gadget structure

RETURNS
  gad - pointer to standard gadget structure or NULL if non existent
        gadget selected

SEE ALSO

                ICreateGadgets
                ,
                IFreeGadgets
                ,
                IDisplayGadgets
                ,
                IRefreshGadgets
                ,

                IModifyGadget
                ,
                ISetGadgetAttributes
                ,
                IActivateInputGadget
                ,

                IRemoveGadgets
```

1.28 IntuiSup/IRemoveGadgets

```
                NAME
  IRemoveGadgets
```

```
SYNOPSIS
  win = IRemoveGadgets( gl )
  d0                    a0

  struct Window *IRemoveGadgets( APTR );

FUNCTION
  Removes all gadgets belonging to given internal data structure from
  display (gadgets aren't selectable any more). Pointer to window
  gadgets displayed before is returned.

INPUTS
  gl - pointer to internal data structure returned by
              ICreateGadgets
              RETURNS
  win - pointer to window gadgets displayed before

SEE ALSO

              ICreateGadgets
              ,
              IFreeGadgets
              ,
              IDisplayGadgets
              ,
              IRefreshGadgets
              ,

              IModifyGadget
              ,
              ISetGadgetAttributes
              ,
              IActivateInputGadget
              ,

              IGadgetAddress
```

1.29 IntuiSup/IGetMsg

```
              NAME
  IGetMsg

SYNOPSIS
  imsg = IGetMsg( uport )
  d0              a0

  struct IntuiMessage *IGetMsg( struct MsgPort * );

FUNCTION
  MUST be used instead of Exec's GetMsg to handle all actions
  belonging to ISUP objects. For all events produced by ISUP objects
  a modified IntuiMessage structure will be returned. Some of their
  members are (mis)used for special ISUP data:

    Class = ISUP_ID -> need to identify an modified ISUP message
    Code = id of the appropriate ISUP object -> offset of object
           data structure in array of
                GadgetData structures
                           given to
                ICreateGadgets
                    IAddress = value returned from ISUP object, e.g. state (0|1)
         of check gadget, count of count gadget,...
         ATTENTION: for string gadgets IAddress contains
              a pointer to the gadget's input buffer,
              so no ptr to gad->StringInfo.Buffer
              needed
    SpecialLink = internal ptr returned by
                ICreateGadgets
                            according to appropriate ISUP object
            ATTENTION: if more than one lists with ISUP
           objects displayed on the same
           window, SpecialLink must be checked
           first for the list the ISUP object
           belongs to

  All other members of the IntuiMessage structure contains their
  normal values. All special IntuiMessage structures MUST be replied
  with
                IReplyMsg
                 instead of Exec's ReplyMsg. Normal IntuiMessage can
  replied with this function too.

INPUTS
  uport - window's user port

RETURNS
  imsg - pointer to message from intuition

SEE ALSO

                IReplyMsg
                EXAMPLE
  struct IntuiMessage *im;

  while (im = IGetMsg(win->UserPort)) {
        ^
        |
     switch (im->Class) {
              case ISUP_ID :
      ULONG value;

      /* Handle event from ISUP object */
      switch (im->Code) {
         case 0 :    /* first object in GadgetData array */
      value = (ULONG)im->IAddress;   /* value returned from this object */
      break;
        :
        case n :    /* n-th object in GadgetData array */
      value = (ULONG)im->IAddress;   /* value returned from this object */
      break;
      }
      break;
```

```
        /* Handle other IDCMP events */
         :
     }
     IReplyMsg(im);
     ^
     |
  }
```

1.30 IntuiSup/IReplyMsg

```
                NAME
  IReplyMsg

SYNOPSIS
  IReplyMsg( imsg )
             a0

  VOID IReplyMsg( struct IntuiMessage * );

FUNCTION
  Replys special IntuiMessage built by
                IGetMsg
                INPUTS
  imsg - IntuiMessage received with
                IGetMsg
                RETURNS
  VOID

SEE ALSO

                IGetMsg
```

1.31 IntuiSup/IConvertRawKeyToASCII

```
                NAME
  IConvertRawKeyToASCII

SYNOPSIS
  IConvertRawKeyToASCII( imsg )
             a0

  UBYTE IConvertRawKeyToASCII( struct IntuiMessage  *imsg );

FUNCTION
  Returns ASCII code of given RAWKEY IntuiMessage or ZERO if no
  ASCII character.

INPUTS
  imsg - IntuiMessage received with
                IGetMsg
                RETURNS
  UBYTE - ASCII code or ZERO if no valid ASCII character

SEE ALSO

                IGetMsg
                ,
                IReplyMsg
```

1.32 IntuiSup/IAutoRequest

```
                NAME
  IAutoRequest

SYNOPSIS
  result = IAutoRequest( reqwin, title, bodytext, postext, negtext,
  d0                     a0      a1     a2        a3       d0
             posidcmpflags, negidcmpflags, reqflags,
             d1             d2             d3
             languagetextarray )
             d4

  BOOL IAutoRequest( struct Window *, BYTE *, BYTE *, BYTE *, BYTE *,
         LONG, LONG, USHORT, BYTE ** );

FUNCTION
  Displays an auto requester from given data and waits for it's
  termination with the positive or negative gadget. In body text a new
  line is started with '\n'. If given language_text_array is not NULL
  then all text pointer don't contain pointer to string but offsets of
  the appropriate string pointers in given array with pointers to text
  in foreign languages.

INPUTS
  reqwin - pointer to window opened by
                (I)OpenWindow
                 or NULL for
     window on workbench screen

  title - pointer to title string for requester window or NULL for
    default title

  bodytext - pointer to text string for requester body

  postext - pointer to text string for positive gadget or NULL for no
      positive gadget

  negtext - pointer to text string for negative gadget or NULL for no
      negative gadget

  posidcmpflags - IDCMP flags for activating positive gadget

  negidcmpflags - IDCMP flags for activating negative gadget

  reqflags -
                AUTO_REQ_DATA_FLAG_xxx
```

```
                  languagetextarray - pointer to string pointer array created with

                IBuildLanguageTextArray
                 or NULL

RETURNS
  result - TRUE if positive gadget or FALSE if negative gadget
     selected (or function failed)

SEE ALSO

                IDisplayRequester
                ,
                IRemoveRequester
```

1.33 IntuiSup/IDisplayRequester

```
                NAME
  IDisplayRequester

SYNOPSIS
  rl = IDisplayRequester( reqwin, rd, languagetextarray )
  d0                      a0      a1  a2

  APTR IDisplayRequester( struct Window *, struct RequesterData *,
        BYTE ** );

FUNCTION
  Displays a requester defined by given
                RequesterData structure
                . All
  other gadgets displayed on window the requester window (req_win)
  opened from are disabled till removing of requester. So any
  IntuiMessages with Class == ISUP_ID received by

                IGetMsg(req_win->UserPort)
                 come from requester. If given
  language_text_array is not NULL then
                td_Text
                 doesn't contain a
  pointer to string but an offset of the string pointer in given array
  with pointers to text in foreign languages. Pointer to internal
  data structure belonging to requester is returned. Requester MUST be
  removed from display with
                IRemoveRequester
                .

INPUTS
  reqwin - pointer to window opened by
                (I)OpenWindow
                 or NULL for
     window on workbench screen

  rd - pointer to initialized
                RequesterData structure
```

```
                  languagetextarray - pointer to string pointer array created with

                IBuildLanguageTextArray
                 or NULL

RETURNS
  rl - pointer to internal data structure (RequesterList) or NULL if
       function failed

SEE ALSO

                IAutoRequest
                ,
                IDisplayRequester
                ,
                IRemoveRequester
```

1.34 IntuiSup/IRemoveRequester

```
                NAME
  IRemoveRequester

SYNOPSIS
  IRemoveRequester( rl )
                    a0

  VOID IRemoveRequester( APTR );

FUNCTION
  Removes requester belonging to given internal data structure from
  display. All gadgets disabled by

                IDisplayRequester
                 are reenabled now.

INPUTS
  rl - pointer to internal data structure returned by
                IDisplayRequester
                RETURNS
  VOID

SEE ALSO

                IAutoRequest
                ,
                IDisplayRequester
```

1.35 IntuiSup/ICreateMenu

```
                NAME
  ICreateMenu
```

```
SYNOPSIS
  ml = ICreateMenu( ri, win, md, ta, languagetextarray )
  d0                a0  a1   a2  a3  d0

  APTR ICreateMenu( APTR, struct Window *, struct MenuData *,
        struct TextAttr *, BYTE ** );

FUNCTION
  Create internal data structure for a menu from given array of data
  structures. This function DON'T display the menu. Displaying will be
  done with
                IAttachMenu
                , Internal data structure MUST be released
  released with
                IFreeMenu
                . If given language_text_array are not
  NULL then
                md_Text
                 doesn't contain a pointer to string but an offset
  of the string pointer in given array with pointers to text in
  foreign languages. Pointer to internal data structure belonging to
  menu is returned.

INPUTS
  ri - pointer to internal data structure returned by
                IGetRenderInfo
                  win - pointer to window opened by
                (I)OpenWindow
                  md - pointer to ARRAY of initialized
                MenuData structures
                  ta - pointer to initialized structure with text attributes

  languagetextarray - pointer to string pointer array created with

                IBuildLanguageTextArray
                 or NULL

RETURNS
  ml - pointer to internal data structure (MenuList) or NULL if
       function failed

SEE ALSO

                IAttachMenu
                ,
                IMenuItemAddress
                ,
                IRemoveMenu
                ,
                IFreeMenu
```

1.36 IntuiSup/IAttachMenu

```
                NAME
  IAttachMenu
```

```
SYNOPSIS
  IAttachMenu( win, ml )
               a0   a1

  VOID IAttachMenu( struct Window *, APTR );

FUNCTION
  Make menu built with
                ICreateMenu
                 available by attaching it to
  given window. Menu MUST be removed with
                IRemoveMenu
                .

INPUTS
  win - pointer to window opened by
                (I)OpenWindow
                  ml - pointer to internal data structure returned by
                ICreateMenu
                RETURNS
  VOID

SEE ALSO

                ICreateMenu
                ,
                IMenuItemAddress
                ,
                IRemoveMenu
                ,
                IFreeMenu
```

1.37 IntuiSup/IMenuItemAddress

```
                NAME
  IMenuItemAddress

SYNOPSIS
  mi = IMenuItemAddress( ml, menunum )
  d0                     a0  d0

  struct MenuItem *IMenuItemAddress( APTR, USHORT );

FUNCTION
  Return pointer to normal MenuItem structure of specified menu item.

INPUTS
  ml - pointer to internal data structure returned by
                ICreateMenu
                  menunum - offset (in array of MenuData structures) of menu item ↩
                     to
      get address

RETURNS
```

```
  mi - pointer to standard menu item structure or NULL if non existent
       menu item selected

SEE ALSO

                ICreateMenu
                ,
                IAttachMenu
                ,
                IRemoveMenu
                ,
                IFreeMenu
```

1.38 IntuiSup/IRemoveMenu

```
                NAME
  IRemoveMenu

SYNOPSIS
  win = IRemoveMenu( ml )
  d0                 a0

  struct Window *IRemoveMenu( APTR );

FUNCTION
  Remove menu attached with
                IAttachMenu
                 from display.

INPUTS
  ml - pointer to internal data structure returned by
                ICreateMenu
                RETURNS
  win - pointer to window menu attached before

SEE ALSO

                ICreateMenu
                ,
                IAttachMenu
                ,
                IMenuItemAddress
                ,
                IFreeMenu
```

1.39 IntuiSup/IFreeMenu

```
                NAME
  IFreeMenu

SYNOPSIS
  IFreeMenu( ml )
```

```
             a0

  VOID IFreeMenu( APTR );

FUNCTION
  Releases memory for internal data structure allocated by

                ICreateMenu
                . If menu is currently attached then it MUST be
  removed with
                IRemoveMenu
                 first.

INPUTS
  ml - pointer to internal data structure returned by
                ICreateMenu
                RETURNS
  VOID

SEE ALSO

                ICreateMenu
                ,
                IAttachMenu
                ,
                IMenuItemAddress
                ,
                IRemoveMenu
```

1.40 IntuiSup/IOpenTextFile

```
                NAME
  IOpenTextFile

SYNOPSIS
  fd = IOpenTextFile( name, readbuffersize, linebuffersize, flags )
  d0                  a0    d0              d1              d2

  struct FileData *IOpenTextFile( BYTE *, USHORT, USHORT, USHORT );

FUNCTION
  Opens given text file and returns pointer to
                data structure
                 with
  allocated buffers. This structure MUST be freed with

                ICloseTextFile
                .

INPUTS
  name - pointer to string with file name

  readbuffersize - size (in bytes) of buffer used for reading text
       file
  linebuffersize - number of bytes used for longest line of text

  flags -
                TEXT_FILE_FLAG_xxx
                RETURNS
  fd - pointer to initialized
                FileData structure
                SEE ALSO

                IReadTextLine
                ,
                ICloseTextFile
```

1.41 IntuiSup/IReadTextLine

```
                NAME
  IReadTextLine

SYNOPSIS
  status = IReadTextLine( fd )
  d0                      a0

FUNCTION
  Read next line from text file opened with
                IOpenTextFile
                . This
  line can be found in given
                FileData structure
                .

INPUTS
  fd - pointer to
                data structure
                 returned by
                IOpenTextFile
                RETURNS
  status -
                TEXT_FILE_STATUS_NORMAL
                 if function succeeded or

                TEXT_FILE_ERROR_xxx
                 if error occured

SEE ALSO

                IOpenTextFile
                ,
                ICloseTextFile
```

1.42 IntuiSup/ICloseTextFile

```
                NAME
  ICloseTextFile

SYNOPSIS
  ICloseTextFile( fd )
                  a0

  VOID ICloseTextFile( struct FileData * );

FUNCTION
  Close text file opened with
                IOpenTextFile
                 and releases allocated
  buffers.

INPUTS
  fd - pointer to data structure returned by
                IOpenTextFile
                RETURNS
  VOID

SEE ALSO

                IOpenTextFile
                ,
                IReadTextLine
```

1.43 IntuiSup/IBuildLanguageTextArray

```
                NAME
  IBuildLanguageTextArray

SYNOPSIS
  languagetextarray = IBuildLanguageTextArray( name, entries )
  d0                                           a0    d0

  BYTE **IBuildLanguageTextArray( BYTE *, USHORT );

FUNCTION
  Parse given text file and return pointer to array of text strings
  read from language file. This array MUST be released with

                IFreeLanguageTextArray
                .

INPUTS
  name - pointer to string with name of file with language texts

  entries - number of text entries in file

RETURNS
  languagetextarray - pointer to language text array or NULL if
          function failed
```

```
SEE ALSO

                IGetLanguageText
                ,
                IFreeLanguageTextArray
```

1.44 IntuiSup/IGetLanguageText

```
                NAME
  IGetLanguageText

SYNOPSIS
  text = IGetLanguageText( text, languagetextarray )
  d0                        a0    a1

  BYTE *IGetLanguageText( BYTE *, BYTE ** );

FUNCTION
  Returns specified entry from within text pointer array created with

                IBuildLanguageTextArray
                .

INPUTS
  text - offset (in array of text strings) of language text (starting
         with 1 instead of 0!!!)

  languagetextarray - pointer to array of text strings created by

                IBuildLanguageTextArray
                RETURNS
  text - pointer to text belonging to this entry in language text
         array

SEE ALSO

                IBuildLanguageTextArray
                ,
                IFreeLanguageTextArray
```

1.45 IntuiSup/IFreeLanguageTextArray

```
                NAME
  IFreeLanguageTextArray

SYNOPSIS
  IFreeLanguageTextArray( languagetextarray )
                          a0

  VOID IFreeLanguageTextArray( BYTE ** );

FUNCTION
```

```
  Releases memory of array created with
                IBuildLanguageTextArray
                .

INPUTS
  languagetextarray - pointer to array of text strings created by

                IBuildLanguageTextArray
                RETURNS
  VOID

SEE ALSO

                IBuildLanguageTextArray
                ,
                IGetLanguageText
```

1.46 IntuiSup/IChangeMousePointer

```
                NAME
  IChangeMousePointer

SYNOPSIS
  IChangeMousePointer( win, pd, removegadgets )
                       a0   a1  d0

  VOID IChangeMousePointer( struct Window *, struct PointerData *,
          BOOL );

FUNCTION
  Replace current mouse pointer of selected window with one described
  in given
                data structure
                . Old mouse pointer will be saved and can be
  restored later with
                IRestoreMousePointer
                .

INPUTS
  win - pointer to window opened by
                (I)OpenWindow
                  pd - pointer to initialized
                PointerData structure
                 or NULL for busy
       mouse pointer

  removegadgets - TRUE if all visisible ISUP gadget lists should be
      removed for busy mouse pointer

RETURNS
  VOID

SEE ALSO

                IRestoreMousePointer
                ,
                IMoveMousePointer
```

1.47 IntuiSup/IRestoreMousePointer

```
                NAME
  IRestoreMousePointer

SYNOPSIS
  IRestoreMousePointer( win )
                        a0

  VOID IRestoreMousePointer( struct Window * );

FUNCTION
  Restore old mouse pointer saved with
                IChangeMousePointer
                .

INPUTS
  win - pointer to window opened by
                (I)OpenWindow
                RETURNS
  VOID

SEE ALSO

                IChangeMousePointer
                ,
                IMoveMousePointer
```

1.48 IntuiSup/IMoveMousePointer

```
                NAME
  IMoveMousePointer

SYNOPSIS
  IMoveMousePointer( win, x, y, button )
                     a0   d0 d1 d2

  VOID IMoveMousePointer( struct Window *, SHORT, SHORT, BOOL );

FUNCTION
  Move mouse pointer of given window to new position.

INPUTS
  win - pointer to window opened by
                (I)OpenWindow
                  x, y - new position (relative to upper left corner of given
         window!!!) for mouse pointer

  button - TRUE for left mouse button pressed
```

```
RETURNS
  VOID

SEE ALSO

                IChangeMousePointer
                ,
                IRestoreMousePointer
```

1.49 IntuiSup/Structures and Defines

```
                Flags for IOpenTextFile

                Status for IReadTextLine

                Text file data structure

                Data structure for IChangeMousePointer
```

1.50 Some notes data IntuiSup structures

```
All ISUP objects are created from special data structures (struct xxxData)
via library functions (Createxxx). Some of these functions expect (a pointer
to) an ARRAY of data structures to create multiple objects with one call.
These arrays are terminated with an entry (data structure) with it's type
member (xxx_Type) set to the special value INTUISUP_DATA_END (0).
DON'T FORGET THIS TERMINATION ENTRY OR YOU'RE VISITED BY THE GURU.

The creation functions return a pointer (APTR) to the internal data
environment according to these ISUP objects. This pointer is used later as
paramter for the other functions to access the object data.
THE POINTERS TO INTERNAL DATA OF THE DIFFERENT ISUP OBJECTS ARE ALL OF THE
SAME TYPE (APTR), SO DON'T CONFUSE WITH THEM.
```

1.51 IntuiSup/Defines for library

```
IntuiSupName - text string containing the name of library
IntuiSupVersion - current version number of library

ISUP_ID - used for identifying IntuiMessages belonging to IntuiSup gadgets
    (imsg->Class == ISUP_ID)

INTUISUP_DATA_END - mark end of data arry (xxx_Type = INTUISUP_DATA_END)
```

1.52 IntuiSup/Flags for GetRenderInfo

```
                RENDER_INFO_FLAG_INNER_WINDOW - use upper left corner of inner  ↩
                   window as
  location (0,0)
RENDER_INFO_FLAG_BACK_FILL - fill window back ground with different color
RENDER_INFO_FLAG_AVAIL_FONTS - scan available fonts and use this list for

                IAskFont
                /
                IOpenFont
```

1.53 IntuiSup/Flags for IOpenWindow

```
OPEN_WINDOW_FLAG_CENTER_SCREEN - center window on screen
OPEN_WINDOW_FLAG_RENDER_PENS - use render pens for detail and backfill pen
OPEN_WINDOW_FLAG_CENTER_MOUSE - center window over current position of mouse
  pointer
OPEN_WINDOW_FLAG_NO_INER_WINDOW - don't add inner window offsets for
  RENDER_INFO_FLAG_INNER_WINDOW
```

1.54 IntuiSup/Flags for IClearWindow

```
CLEAR_WINDOW_FLAG_CUSTOM_DRAW_MODE - don't change draw mode
CLEAR_WINDOW_FLAG_CUSTOM_COLOR - don't change background color
CLEAR_WINDOW_FLAG_NORMAL_COLOR - use normal background color
```

1.55 IntuiSup/Text data types

```
TEXT_DATA_TYPE_TEXT - pointer to normal text string
TEXT_DATA_TYPE_NUM_UNSIGNED_DEC - no pointer to text string but unsigned
  decimal number
TEXT_DATA_TYPE_NUM_SIGNED_DEC - no pointer to text string but signed decimal
  number
TEXT_DATA_TYPE_NUM_HEX - no pointer to text string but hexadecimal number
TEXT_DATA_TYPE_NUM_BIN - no pointer to text string but binary number
```

1.56 IntuiSup/Text data flags

```
TEXT_DATA_FLAG_BOLD - text font attribute: bold
TEXT_DATA_FLAG_ITALIC - text font attribute: italic
TEXT_DATA_FLAG_UNDERLINED - text font attribute: underlined
TEXT_DATA_FLAG_ABSOLUTE_POS - absolute text pos given so don't add border
  offsets
TEXT_DATA_FLAG_CENTER - center text within window width
TEXT_DATA_FLAG_PLACE_LEFT - place text left from given left edge
TEXT_DATA_FLAG_COLOR2 - use 2nd text render pen
TEXT_DATA_FLAG_COMPLEMENT - use complement of front and back pen
TEXT_DATA_FLAG_BACK_FILL - use draw mode JAM2 to fill text background with
  ri_BackPen
TEXT_DATA_FLAG_NO_PRINT - don't print text (only calc width)
TEXT_DATA_FLAG_NUM_IDENTIFIER - prepend converted number with assember
  style identifiers e.g. `$' or `%'
TEXT_DATA_FLAG_NUM_C_STYLE - prepend converted number with C style
  identifiers e.g. `0x'
TEXT_DATA_FLAG_NUM_LEADING_ZEROES - print converted number with leading
  zeros
TEXT_DATA_FLAG_NUM_UPPER_CASE - use upper case characters for hex number
```

1.57 IntuiSup/Flags for converting functions

```
CONVERT_FLAG_IDENTIFIER - prepend converted number with assember syle
  identifiers e.g. `$' or `%'
CONVERT_FLAG_C_STYLE - prepend converted number with C style identifiers
  e.g. `0x'
CONVERT_FLAG_LEADING_ZEROES - include leading zeros
CONVERT_FLAG_UPPER_CASE - use upper case characters for hex numbers
```

1.58 IntuiSup/Text data structure

```
                struct TextData {
   USHORT
                td_Type
                ;
   USHORT
                td_Flags
                ;
   SHORT  td_LeftEdge;
   SHORT  td_TopEdge;
   BYTE   *td_Text;
   struct TextAttr  *td_TextAttr;
};
```

1.59 IntuiSup/Border types

```
BORDER_DATA_TYPE_BOX1_OUT - bevelled border of type 1
BORDER_DATA_TYPE_BOX1_IN - recessed border of type 1
BORDER_DATA_TYPE_BOX2_OUT - bevelled border of type 2
BORDER_DATA_TYPE_BOX2_IN - recessed border of type 1
```

1.60 IntuiSup/Border data structure

```
                struct BorderData {
   USHORT
                bd_Type
                ;
   SHORT  bd_LeftEdge;
   SHORT  bd_TopEdge;
   USHORT bd_Width;
   USHORT bd_Height;
};
```

1.61 IntuiSup/Gadget types

```
GADGET_DATA_TYPE_BUTTON - button gadget
GADGET_DATA_TYPE_CHECK - check mark gadget
GADGET_DATA_TYPE_MX - mutual exclude gadget
GADGET_DATA_TYPE_STRING - string input gadget
```

```
GADGET_DATA_TYPE_INTEGER - integer input gadget
GADGET_DATA_TYPE_SLIDER - slider gadget
GADGET_DATA_TYPE_SCROLLER - scroller gadget
GADGET_DATA_TYPE_CYCLE - cycle gadget
GADGET_DATA_TYPE_COUNT - count gadget
GADGET_DATA_TYPE_LISTVIEW - list view gadget
GADGET_DATA_TYPE_PALETTE - palette gadget
```

1.62 IntuiSup/Gadget flags

```
  General flags:

GADGET_DATA_FLAG_DISABLED - gadget disabled (ghosted) -> default enabled
GADGET_DATA_FLAG_NO_BORDER - no gadget border -> default with border
GADGET_DATA_FLAG_HIGH_COMP - highliting by complement -> default highliting
  by select border
GADGET_DATA_FLAG_ORIENTATION_VERT - vertical orientation -> default
  horizontal
GADGET_DATA_FLAG_HOTKEY - hotkey given -> default none
GADGET_DATA_FLAG_MOVE_POINTER - move mouse pointer to center of this gadget
GADGET_DATA_FLAG_NO_CLEAR - don’t clear area occupied by this gadget before
  drawing
GADGET_DATA_FLAG_NO_TEXT_OUTPUT - no text output, but scan gadget text for
  hotkey
GADGET_DATA_FLAG_TEXT_LEFT - place text left of gadget
GADGET_DATA_FLAG_TEXT_RIGHT - place text right of gadget
GADGET_DATA_FLAG_TEXT_ABOVE - place text above of gadget
GADGET_DATA_FLAG_TEXT_BELOW - place text below of gadget
GADGET_DATA_FLAG_TEXT_COLOR2 - use 2nd text render pen for gadget text

  Special flags:

GADGET_DATA_FLAG_BUTTON_TOGGLE - button gadgets: toggle button - default no
  toggle
GADGET_DATA_FLAG_BUTTON_IMAGE - button gadgets: render image - default no
  image

GADGET_DATA_FLAG_INPUT_AUTO_ACTIVATE - input gadgets: activate after
  GADGETUP next or previous input gadget (given in gd_SpecialData)
GADGET_DATA_FLAG_INPUT_CENTER - center input string within gadget
GADGET_DATA_FLAG_INPUT_RIGHT - right justify input string within gadget

GADGET_DATA_FLAG_STRING_UNSIGNED_DEC - string gadgets: input default no
  pointer to string but an unsigned decimal number
GADGET_DATA_FLAG_STRING_SIGNED_DEC - string gadgets: input default no
  pointer to string but an signed decimal number
GADGET_DATA_FLAG_STRING_HEX - string gadgets: input default no pointer to
  string but an hex number
GADGET_DATA_FLAG_STRING_BIN - string gadgets: input default no pointer to
  string but an binary number

GADGET_DATA_FLAG_SCROLLER_NO_ARROWS - scroller gadget: no arrows - default
  with arrows

GADGET_DATA_FLAG_SLIDER_IMAGE - kludge to define image for knob of
```

```
  proportional gadget in gd_TextAttr (if text then default TextAttr
  used)

GADGET_DATA_FLAG_COUNT_SIGNED_DEC - count gadget: signed dec - default
  unsigned dec

GADGET_DATA_FLAG_LISTVIEW_READ_ONLY - list view gadget: read only - default
  selection enabled
GADGET_DATA_FLAG_LISTVIEW_SHOW_SELECTED - list view gadget: show last
  selected entry - default no
#define GADGET_DATA_FLAG_LISTVIEW_ENTRY_COLOR - if first char of an entry
  text equals <Ctrl A> ($01) then this char will be skipped and the
  rest of this entry text will be printed in a different color

GADGET_DATA_FLAG_PALETTE_NO_INDICATOR - palette gadget: no current color
  indicator - default with indicator
GADGET_DATA_FLAG_PALETTE_INDICATOR_TOP - palette gadget: place indicator at
  top - default at left
```

1.63 IntuiSup/Other gadget defines

```
                  IDCMP flags for gadgets:

GADGET_IDCMP_FLAGS_BUTTON (GADGETUP | RAWKEY)
GADGET_IDCMP_FLAGS_CHECK  (GADGETDOWN | RAWKEY)
GADGET_IDCMP_FLAGS_MX   (GADGETDOWN | RAWKEY)
GADGET_IDCMP_FLAGS_STRING (GADGETUP | RAWKEY)
GADGET_IDCMP_FLAGS_INTEGER  (GADGETUP | RAWKEY)
GADGET_IDCMP_FLAGS_SLIDER (GADGETUP | MOUSEMOVE | RAWKEY)
GADGET_IDCMP_FLAGS_SCROLLER (GADGETDOWN | GADGETUP | MOUSEMOVE | INTUITICKS |  ↩
   RAWKEY)
GADGET_IDCMP_FLAGS_CYCLE  (GADGETUP | RAWKEY)
GADGET_IDCMP_FLAGS_COUNT  (GADGETDOWN | GADGETUP | MOUSEMOVE | RAWKEY)
GADGET_IDCMP_FLAGS_LISTVIEW (GADGETDOWN | GADGETUP | MOUSEMOVE | INTUITICKS |  ↩
   RAWKEY)
GADGET_IDCMP_FLAGS_PALETTE  (GADGETUP | RAWKEY)
GADGET_IDCMP_FLAGS_ALL    (GADGETDOWN | GADGETUP | MOUSEMOVE | INTUITICKS | RAWKEY ↩
   )

  Macros and constants:

INPUT_AUTO_ACTIVATE(next,prev) - macro to generate longword with next and
         previous input gadget to activate for
         gd_SpecialData.gd_Data3 for input gadgets

USE_CURRENT_VALUE - used for
                ISetGadgetAttributes
                 to indicate special
        data for which to use the current value
```

1.64 IntuiSup/Gadget data structure

```
             struct GadgetData {
ULONG
             gd_Type
             ;
ULONG
             gd_Flags
             ;
USHORT gd_LeftEdge;
USHORT gd_TopEdge;
USHORT gd_Width;
USHORT gd_Height;
BYTE   *gd_Text;
struct TextAttr  *gd_TextAttr;

/* union with special data */
union {

   /* standard special data */
   struct {
LONG gd_Data1;
      LONG gd_Data2;
VOID *gd_Data3;
   } gd_Data;

   /* special data for button gadgets */
   struct {

/* selection state for toggle buttons - ZERO = unselected
 *                                       non ZERO = selected
 */
ULONG gd_ButtonSelected;

/* normal render image */
struct Image  *gd_ButtonNormalRender;

/* select render image */
struct Image  *gd_ButtonSelectRender;
   } gd_ButtonData;

   /* special data for check gadgets */
   struct {

/* selection state - ZERO = unselected
 *                   non ZERO = selected
 */
ULONG gd_CheckSelected;
ULONG gd_CheckPad1;
ULONG gd_CheckPad2;
   } gd_CheckData;

   /* special data for mutual exclude gadgets */
   struct {

/* pixel spacing between MX gadgets */
ULONG gd_MXSpacing;
```

```
/* num of active entry from text array */
ULONG gd_MXActiveEntry;

/* pointer to MX text pointer array */
BYTE  **gd_MXTextArray;
   } gd_MXData;

   /* special data for string and integer gadgets */
   struct {

/* len of input buffer */
ULONG  gd_InputLen;

/* num of next string/num gadget to activate */
USHORT gd_InputActivateNext;

/* num of previous string/num gadget to activate */
USHORT gd_InputActivatePrev;

/* default input - string: default text [syntax: "text"]
 *                integer: default number [syntax: (VOID *)num]
 */
BYTE   *gd_InputDefault;
   } gd_InputData;

   /* special data for slider gadgets */
   struct {

/* minimal level */
LONG gd_SliderMin;

/* maximal level */
LONG gd_SliderMax;

/* current slider level */
LONG gd_SliderLevel;
   } gd_SliderData;

   /* special data for scroller gadgets */
   struct {

/* number of visible entries */
ULONG gd_ScrollerVisible;

/* number of total entries */
ULONG gd_ScrollerTotal;

/* number of current top entry */
ULONG gd_ScrollerTop;
   } gd_ScrollerData;

   /* special data for cycle gadget */
   struct {

/* pixel spacing between pop up cycle list entries */
ULONG gd_CycleSpacing;
```

```
   /* number of current cycle text pointer array entry */
   ULONG gd_CycleActive;

   /* pointer to cycle text pointer array */
   BYTE  **gd_CycleTextArray;
      } gd_CycleData;

      /* special data for count gadget */
      struct {

   /* minimal value */
   ULONG gd_CountMin;

   /* maximal value */
   ULONG gd_CountMax;

   /* current count value */
   ULONG gd_CountValue;
      } gd_CountData;

      /* special data for list view gadget */
      struct {

   /* pixel spacing between list view entries */
   ULONG gd_ListViewSpacing;

   /* current top entry */
   ULONG gd_ListViewTop;

   /* current list pointer */
   struct List  *gd_ListViewList;
      } gd_ListViewData;

      /* special data for palette gadget */
      struct {

   /* number of bitplanes for palette */
   ULONG gd_PaletteDepth;

   /* first color of palette */
   ULONG gd_PaletteColorOffset;

   /* selected color */
   ULONG gd_PaletteActiveColor;
      } gd_PaletteData;
   } gd_SpecialData;
};
```

1.65 IntuiSup/Auto Requester flags

```
AUTO_REQ_FLAG_BACK_FILL - fill background with background color
AUTO_REQ_FLAG_RENDER_PENS - use render pens for detail and backfill pens of
  requester window
AUTO_REQ_FLAG_TEXT_CENTER - center text within requester window
AUTO_REQ_FLAG_TEXT_COLOR2 - use 2nd text color for requester text
```

```
AUTO_REQ_FLAG_HOTKEY - get hotkey from gadget texts
AUTO_REQ_FLAG_BEEP - beep with Intuition’s DisplayBeep when opening
  requester window
AUTO_REQ_FLAG_MOVE_POINTER_POS - center move mouse pointer over positive
  gadget
AUTO_REQ_FLAG_MOVE_POINTER_NEG - center move mouse pointer over ngative
  gadget
AUTO_REQ_FLAG_DRAW_RASTER - draw raster around text area
AUTO_REQ_FLAG_CENTER_MOUSE - center last gadget of auto requester over
  current position of mouse pointer
```

1.66 IntuiSup/Requester flags

```
REQ_DATA_FLAG_BACK_FILL - fill background with background color
REQ_DATA_FLAG_RENDER_PENS - use render pens for detail and backfill pens of
  requester window
REQ_DATA_FLAG_INNER_WINDOW - use upper left corner of inner requester window
  as location (0,0)
REQ_DATA_FLAG_AVAIL_FONTS - scan available fonts and use this list for
  IAskFont/IOpenFont
REQ_DATA_FLAG_CENTER_SCREEN - center requester window on given window’s
  screen
REQ_DATA_FLAG_DRAG_GADGET - enable window’s drag gadget
REQ_DATA_FLAG_DEPTH_GADGET - enable window’s depth gadget
REQ_DATA_FLAG_CENTER_WINDOW - center requester window on given window
REQ_DATA_FLAG_CENTER_MOUSE - center requester window over current position
  of mouse pointer
```

1.67 IntuiSup/Requester data structure

```
             struct RequesterData {
   SHORT rd_LeftEdge;
   SHORT rd_TopEdge;
   SHORT rd_Width;
   SHORT rd_Height;
   ULONG
             rd_Flags
             ;
   BYTE  *rd_Title;
   struct TextData    *rd_Texts;
   struct BorderData  *rd_Borders;
   struct GadgetData  *rd_Gadgets;
};
```

1.68 IntuiSup/Menu types

```
MENU_DATA_TYPE_TITLE - start new menu
MENU_DATA_TYPE_ITEM - new menu item
MENU_DATA_TYPE_SUBITEM - attach subitem to previous menu item
```

1.69 IntuiSup/Menu flags

```
MENU_DATA_FLAG_DISABLED - disable menu or menu item
MENU_DATA_FLAG_ATTRIBUTE - attribute menu item
MENU_DATA_FLAG_SELECTED - selected attribute menu item
MENU_DATA_FLAG_EMPTY_LINE - insert empty line before this item
MENU_DATA_FLAG_HIGH_NONE - no highliting
MENU_DATA_FLAG_HIGH_BOX - highliting with box, otherwise with complement
```

1.70 IntuiSup/Menu data structure

```
                struct MenuData {
   USHORT
                md_Type
                ;
   USHORT
                md_Flags
                ;
   BYTE   *md_Name;
   BYTE   *md_CommandKey;
   ULONG  md_MutualExclude;   /* bit mask for mutual excluding menu items */
};
```

1.71 IntuiSup/Flags for IOpenTextFile

```
TEXT_FILE_FLAG_TRIM_LINE - strip leading and trailing white space
TEXT_FILE_FLAG_SKIP_COMMENTS - skip C style comments
TEXT_FILE_FLAG_SKIP_EMPTY_LINES - skip empty lines
TEXT_FILE_FLAG_LINE_CONTINUATION - continue line with last character '\' in
  next line
```

1.72 IntuiSup/Status for IReadTextLine

```
  Status codes:

TEXT_FILE_STATUS_NORMAL - normal status
TEXT_FILE_STATUS_EOF - end of file reached

  Error codes:

TEXT_FILE_ERROR_NO_FILE_DATA - invalid pointer to FileData structure given
TEXT_FILE_ERROR_LINE_TOO_LONG - line too long to fit into line buffer
TEXT_FILE_ERROR_NO_COMMENT_END - missing end of C style comment
TEXT_FILE_ERROR_READ_FAILED - AmigaDOS function Read failed
```

1.73 IntuiSup/Text file data structure

```
struct FileData {
   BYTE   *fd_Line;
   USHORT fd_LineLen;
   USHORT fd_LineNum;
};
```

1.74 IntuiSup/Data structure for IChangeMousePointer

```
struct PointerData {
   UBYTE pd_Width;  /* width of image */
   UBYTE pd_Height; /* height of image */
   BYTE  pd_XOffset;  /* vertical offset of pointer’s hotspot */
   BYTE  pd_YOffset;  /* horizontal offset of pointer’s hotspot */
   UWORD *pd_Data;  /* pointer to image data */
};
```