Gadget

COLLABORATORS			
	TITLE : Gadget		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 23, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Gadget

1.1 Gadget

```
PureBasic 'Gadget' library

  Gadgets in PureBasic are based on the AmigaOS 'GadTools' library
  and provide a very easy way to setup the layout. All the gadgets
  types are supported.

Commands summary:

   AttachGadgetList
   ActivateGadget
   ButtonGadget
   CheckBoxGadget
   CreateGadgetList
   CycleGadget
   DisableGadget
   GetStringText
   FreeGadgetList
   InitGadget
   IntegerGadget
   ListViewGadget
   NumberGadget
   NoGadgetBorder
   OptionGadget
   PaletteGadget
   RefreshGadget
   RefreshGadgetList
   ScrollerGadget
   SetGadgetAttrs
   SetGadgetFlags
   SetGadgetFont
   SetStringText
   SliderGadget
   StringGadget
   TextGadget
   UseGadgetList

Examples:
```

```
    Program 1
    Wild Manager
```

1.2 initgadget

```
  SYNTAX
result = InitGadget(#GadgetLists)

  COMMAND
Try to open the gadtools.library and initialize the gadget
environments. You must put this command before using any of the
gadget set.
```

1.3 buttongadget

```
  SYNTAX
ButtonGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Create a button gadget in the GadgetList. #Gadget will be  the number
returned by EventGadget() command.


Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V36)
  #GA_Immediate (BOOL) - Hear #IDCMP_GADGETDOWN events from button gadget
      (defaults to FALSE). (V39)
```

1.4 checkboxgadget

```
  SYNTAX
CheckBoxGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Create a checkbox gadget in the GadgetList. #Gadget  will  be the number
number returned by EventGadget() command. The 'Text$' is an optionnal
description of the checkbox which will be displayed at the right of it:

  _
 |_| Enable automatic saving.


Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE).
  #GTCB_Checked (BOOL) - Initial state of checkbox (defaults to FALSE)
```

```
      (V36)
  #GTCB_Scaled (BOOL) - If true, then checkbox imagery will be scaled to
      fit the gadget's width & height.  Otherwise, a fixed size of
      CHECKBOXWIDTH by CHECKBOXHEIGHT will be used. (defaults to FALSE)
      (V39)
```

1.5 integergadget

```
  SYNTAX
IntegerGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Creates an Integer gadget in the GadgetList. #Gadget will be the number
returned  by  EventGadget() command.


Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V36)
  #GA_Immediate (BOOL) - Hear #IDCMP_GADGETDOWN events from integer
      gadget (defaults to FALSE). (V39)
  #GA_TabCycle (BOOL) - Set to TRUE so that pressing <TAB> or <Shift-TAB>
      will activate the next or previous such gadget. (defaults to TRUE,
      unlike regular Intuition string gadgets which default to FALSE).
      (V37)
  #GTIN_Number (LONG) - The initial contents of the integer gadget
      (defaults to 0). (V36)
  #GTIN_MaxChars (UWORD) - The maximum number of digits that the
      integer gadget is to hold (defaults to 10). (V36)
  #GTIN_EditHook (struct Hook *) - Hook to use as a custom
      integer gadget edit hook (StringExtend->EditHook) for this gadget.
      GadTools will allocate the StringExtend->WorkBuffer for you.
      (defaults to NULL). (V37)
  #STRINGA_ExitHelp (BOOL) - Set to TRUE to have the help-key cause an
      exit from the integer gadget.  You will then receive an
      #IDCMP_GADGETUP event with Code = 0x5F (rawkey for help).
      (defaults to FALSE) (V37)
  #STRINGA_Justification - Controls the justification of the contents of
      an integer gadget.  Choose one of STRINGLEFT, STRINGRIGHT, or
      STRINGCENTER (defaults to STRINGLEFT). (V37)
  #STRINGA_ReplaceMode (BOOL) - If TRUE, this integer gadget is in
      replace-mode (defaults to FALSE (insert-mode)). (V37)
```

1.6 listviewgadget

```
  SYNTAX
ListViewGadget(#Gadget, x, y, Width, Heigth, Text$, Labels(), TagList)

  COMMAND
Create a ListView gadget in the GadgetList.  #Gadget  will  be the
number  returned by EventGadget() command. 'Labels()' is a linked
```

```
linked (created with 'NewList') with a structured type similar
to this one:

  Structure ListViewStructure
    Pad.w    ; Skip this 2 first bytes
    String.s ; This is the strings which will be displayed in the listview.
  EndStructure

The 'Text$' is an optional description which will be displayed above
the ListView like that:


    List Content:
 _____________________ _
|Item 1               | |
|Item 2               | |
|Item 3               |_|
|                     |_|
|_____________________|_|



Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V39)
  #GTLV_Top (WORD) - Top item visible in the listview.  This value
      will be made reasonable if out-of-range (defaults to 0). (V36)
  #GTLV_MakeVisible (WORD) - Number of an item that should be forced
      within the visible area of the listview by doing minimal scrolling.
      This tag overrides #GTLV_Top. (V39)
  #GTLV_Labels (struct List *) - List of nodes whose ln_Name fields
      are to be displayed in the listview. (V36)
  #GTLV_ReadOnly (BOOL) - If TRUE, then listview is read-only
      (defaults to FALSE). (V36)
  #GTLV_ScrollWidth (UWORD) - Width of scroll bar for listview.
      Must be greater than zero (defaults to 16). (V36)
  #GTLV_ShowSelected (struct Gadget *) - NULL to have the currently
      selected item displayed beneath the listview under V37 or with
      a highlight bar in V39. If not NULL, this is a pointer to
      an already-created GadTools #STRING_KIND gadget to have an
      editable display of the currently selected item. If the tag is
      not present, the currently selected item will not be displayed.
      (V36)
  #GTLV_Selected (UWORD) - Ordinal number of currently selected
      item, or ~0 to have no current selection (defaults to ~0). (V36)
  #LAYOUTA_Spacing (UWORD) - Extra space to place between lines of
      listview (defaults to 0). (V36)
  #GTLV_ItemHeight (UWORD) - The exact height of an item. This is
      normally useful for listviews that use the #GTLV_CallBack
      rendering hook (defaults to ng->ng_TextAttr->ta_YSize). (V39)
  #GTLV_CallBack (struct Hook *) - Callback hook for various listview
      operations. As of V39, the only callback supported is for custom
      rendering of individual items in the listview. The call back hook
      is called with:
      A0 - struct Hook *
      A1 - struct LVDrawMsg *
```

```
      A2 - struct Node *
      The callback hook *must* check the lvdm_MethodID field of the
      message and only do processing if it equals LV_DRAW. If any
      other value is passed, the callback hook must return LVCB_UNKNOWN
  #GTLV_MaxPen (UWORD) - The maximum pen number used by rendering
      in a custom rendering callback hook. This is used to optimize
      the rendering and scrolling of the listview display (default is
      the maximum pen number used by all of TEXTPEN, BACKGROUNDPEN,
      FILLPEN, TEXTFILLPEN, and BLOCKPEN. (V39)
```

1.7 numbergadget

```
  SYNTAX
NumberGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Create a Number gadget in the GadgetList. #Gadget will be  the number
returned  by  EventGadget()  command.


Available Tags:

  #GTNM_Number (LONG) - A signed long integer to be displayed as a read-only
      number (defaults to 0). (V36)
  #GTNM_Border (BOOL) - If TRUE, this flag asks for a recessed border to
      be placed around the gadget. (V36)
  #GTNM_FrontPen (UBYTE) - The pen to use when rendering the number
      (defaults to DrawInfo->dri_Pens[TEXTPEN]). (V39)
  #GTNM_BackPen (UBYTE) - The pen to use when rendering the background
      of the number (defaults to leaving the background untouched).
      (V39)
  #GTNM_Justification (UBYTE) - Determines how the number is rendered
      within the gadget box. GTJ_LEFT will make the rendering be
      flush with the left side of the gadget, GTJ_RIGHT will make it
      flush with the right side, and GTJ_CENTER will center the number
      within the gadget box. Under V39, using this tag also required
      using {#GTNM_Clipped, TRUE}, otherwise the text would not show
      up in the gadget. This has been fixed in V40.
      (defaults to GTJ_LEFT). (V39)
  #GTNM_Format (STRPTR) - C-Style formatting string to apply on the number
      before display. Be sure to use the 'l' (long) modifier. This string
      is processed using exec.library/RawDoFmt(), so refer to that
      function for details. (defaults to "%ld") (V39)
  #GTNM_MaxNumberLen (ULONG) - Maximum number of bytes that can be
      generated by applying the #GTNM_Format formatting string to the
      number (excluding the NULL terminator). (defaults to 10). (V39)
  #GTNM_Clipped (BOOL) - Determine whether text should be clipped to
      the gadget dimensions (defaults to FALSE for gadgets without
      borders, TRUE for gadgets with borders). (V39)
```

1.8 cyclegadget

```
  SYNTAX
CycleGadget(#Gadget, x, y, Width, Heigth, Text$, Labels(), TagList)

  COMMAND
Create a Cycle gadget in the GadgetList. #Gadget will  be  the number
returned  by  EventGadget()  command. The 'Labels()' parameter represents
an array of 'String' which the content of the CycleGadget:

  Dim CycleGadgetItems.s(3)
  CycleGadgetItems(0) = "Item1"
  CycleGadgetItems(1) = "Item2"
  CycleGadgetItems(2) = "Item3"

The 'Text$' parameter is an optional description which will be displayed
before the cycle gadget like that:

                           -----------
  Choose your processor: | |  680x0  |
                           -----------

Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V37)
  #GTCY_Labels (STRPTR *) - Pointer to NULL-terminated array of strings
      that are the choices offered by the cycle gadget. This tag is
      required. (V36)
  #GTCY_Active (UWORD) - The ordinal number (counting from zero) of
      the initially active choice of a cycle gadget (defaults to zero).
      (V36)
```

1.9 optiongadget

```
  SYNTAX
OptionGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Creates an exclusive option gadget in the GadgetList. #Gadget will
be the number  returned  by  EventGadget()  command. The 'Text$' is
a desciption which will be displayed above the option gadget:

  Months:

   -
  | | January
   -

   -
  | | February
   -



Available Tags:
```

```
  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V39)
  #GTMX_Labels (STRPTR *) - Pointer to a NULL-terminated array of
      strings which are to be the labels beside each choice in a
      set of mutually exclusive gadgets. This tag is required. (V36)
  #GTMX_Active (UWORD) - The ordinal number (counting from zero) of
      the initially active choice of an mx gadget (defaults to 0). (V36)
  #GTMX_Spacing (UWORD) - The amount of space between each choice
      of a set of mutually exclusive gadgets.  This amount is added
      to the font height to produce the vertical shift between
      choices (defaults to 1). (V36)
  #GTMX_Scaled (BOOL) - If true, then mx gadget imagery will be scaled
      to fit the gadget's width & height.  Otherwise, a fixed size of
      MXWIDTH by MXHEIGHT will be used. When setting this tag to TRUE,
      you should typically set the height of the gadget to be
      (ng.ng_TextAttr->ta_YSize + 1). (defaults to FALSE.) (V39)
  #GTMX_TitlePlace - One of PLACETEXT_LEFT, PLACETEXT_RIGHT,
      PLACETEXT_ABOVE, or PLACETEXT_BELOW, indicating where the title
      of the gadget is to be displayed. Without this tag, the
      NewGadget.ng_GadgetText field is ignored for MX_KIND gadgets.
      (V39)
```

1.10 palettegadget

```
  SYNTAX
PaletteGadget(#Gadget, x, y, Width, Heigth, Text$, Depth, TagList)

  COMMAND
Creates a Palette gadget in the GadgetList. #Gadget will be the number
returned  by  EventGadget()  command.  The 'Depth' (ie: number of colours)
of this gadget can be specified. The 'Text$' is an optional description
above of this gadget:


    Choose your colour:

 ----------------------------
|      |      |      |       |
|      |      |      |       |
 ----------------------------


Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V36)
  #GTPA_Depth (UWORD) - Number of bitplanes in the palette
      (defaults to 1). (V36)
  #GTPA_Color (UBYTE) - Initially selected color of the palette. This
      number is a pen number, and not the ordinal colour number within
      the palette gadget itself. (defaults to 1). (V36)
  #GTPA_ColorOffset (UBYTE) - First colour to use in palette
      (defaults to 0). (V36)
  #GTPA_IndicatorWidth (UWORD) - The desired width of the current-colour
```

```
      indicator, if you want one to the left of the palette. (V36)
  #GTPA_IndicatorHeight (UWORD) - The desired height of the current-color
      indicator, if you want one above the palette. (V36)
  #GTPA_ColorTable (UBYTE *) - Pointer to a table of pen numbers
      indicating  which colors should be used and edited by the palette
      gadget. This array must contain as many entries as there are
      colors displayed in the palette gadget. The array provided with
      this tag must remain valid for the life of the gadget or until a
      new table is provided. (default is NULL, which causes a 1-to-1
      mapping of pen numbers). (V39)
  #GTPA_NumColors (UWORD) - Number of colors to display in the palette
      gadget. This override #GTPA_Depth and allows numbers which aren't
      powers of 2. (defaults to 2) (V39)
```

1.11 scrollergadget

```
  SYNTAX
ScrollerGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Creates a Scroller gadget in the GadgetList.  #Gadget  will  be the
number  returned by EventGadget() command.


Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V36)
  #GA_RelVerify (BOOL) - Hear every #IDCMP_GADGETUP event from scroller
      (defaults to FALSE). (V36)
  #GA_Immediate (BOOL) - Hear every #IDCMP_GADGETDOWN event from scroller
      (defaults to FALSE). (V36)
  #GTSC_Top (WORD) - Top visible in area scroller represents
      (defaults to 0). (V36)
  #GTSC_Total (WORD) - Total in area scroller represents
      (defaults to 0). (V36)
  #GTSC_Visible (WORD) - Number visible in scroller (defaults to 2). (V36)
  #GTSC_Arrows (UWORD) - Asks for arrows to be attached to the scroller.
      The value supplied will be taken as the width of each arrow button
      for a horizontal scroller, or the height of each button for a
      vertical scroller (the other dimension will match the whole
      scroller). (V36)
  #PGA_Freedom - Whether scroller is horizontal or vertical.
      Choose LORIENT_VERT or LORIENT_HORIZ (defaults to LORIENT_HORIZ).
      (V36)
```

1.12 slidergadget

```
  SYNTAX
SliderGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
```

```
Creates a Slider gadget in the GadgetList. #Gadget will be  the number
returned  by  EventGadget()  command.


Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V36)
  #GA_RelVerify (BOOL) - If you want to hear each slider #IDCMP_GADGETUP
      event (defaults to FALSE). (V36)
  #GA_Immediate (BOOL) - If you want to hear each slider #IDCMP_GADGETDOWN
      event (defaults to FALSE). (V36)
  #GTSL_Min (WORD) - Minimum level for slider (defaults to 0). (V36)
  #GTSL_Max (WORD) - Maximum level for slider (defaults to 15). (V36)
  #GTSL_Level (WORD) - Current level of slider (defaults to 0). (V36)
  #GTSL_MaxLevelLen (UWORD) - Maximum length in characters of level string
      when rendered beside slider (defaults to 2). (V36)
  #GTSL_LevelFormat (STRPTR) - C-Style formatting string for slider
      level.  Be sure to use the 'l' (long) modifier.  This string
      is processed using exec.library/RawDoFmt(), so refer to that
      function for details. (defaults to "%ld"). (V36)
  #GTSL_LevelPlace - One of PLACETEXT_LEFT, PLACETEXT_RIGHT,
      PLACETEXT_ABOVE, or PLACETEXT_BELOW, indicating where the level
      indicator is to go relative to slider (default to PLACETEXT_LEFT).
      (V36)
  #GTSL_DispFunc ( LONG (*function)(struct Gadget *, WORD) ) - Function
      to calculate level to be displayed.  A number-of-colors slider
      might want to set the slider up to think depth, and have a
      (1 << n) function here.  Defaults to none.  Your function must
      take a pointer to gadget as the first parameter, the level
      (a WORD) as the second, and return the result as a LONG. (V36)
  #GTSL_MaxPixelLen (ULONG) - Indicates the maximum pixel size used up
      by the level display for any value of the slider. This is mostly
      useful when dealing with proportional fonts. (defaults to
      FontWidth*MaxLevelLen). (V39)
  #GTSL_Justification (UBYTE) - Determines how the level display is to
      be justified within its alotted space. Choose one of GTJ_LEFT,
      GTJ_RIGHT, or GTJ_CENTER (defaults to GTJ_LEFT). (V39)
  #PGA_Freedom - Set to LORIENT_VERT or LORIENT_HORIZ to have a
      vertical or horizontal slider (defaults to LORIENT_HORIZ). (V36)
```

1.13 stringgadget

```
  SYNTAX
StringGadget(#Gadget, x, y, Width, Heigth, Text$, Content$, TagList)

  COMMAND
Creates a String gadget in the GadgetList. #Gadget will be the number
returned  by  EventGadget()  command. You can specify the initial content
of this StringGadget. Later the content can be changed with the commands
SetStringText() and GetStringText(). The 'Text$' is an optional description
located at the left of the gadget:

                      ---------------------
```

```
    Enter your name: |                    |
                     ---------------------
```

```
Available Tags:

  #GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
      (defaults to FALSE). (V36)
  #GA_Immediate (BOOL) - Hear #IDCMP_GADGETDOWN events from string
      gadget (defaults to FALSE). (V39)
  #GA_TabCycle (BOOL) - Set to TRUE so that pressing <TAB> or <Shift-TAB>
      will activate the next or previous such gadget. (defaults to TRUE,
      unlike regular Intuition string gadgets which default to FALSE).
      (V37)
  #GTST_String (STRPTR) - The initial contents of the string gadget,
      or NULL (default) if string is to start empty. (V36)
  #GTST_MaxChars (UWORD) - The maximum number of characters that the
      string gadget is to hold. (V36)
  #GTST_EditHook (struct Hook *) - Hook to use as a custom string gadget
      edit hook (StringExtend->EditHook) for this gadget. GadTools will
      allocate the StringExtend->WorkBuffer for you. (defaults to NULL).
      (V37)
  #STRINGA_ExitHelp (BOOL) - Set to TRUE to have the help-key cause an
      exit from the string gadget.  You will then receive an
      #IDCMP_GADGETUP event with Code = 0x5F (rawkey for help).
      (V37)
  #STRINGA_Justification - Controls the justification of the contents of
      a string gadget.  Choose one of #STRINGLEFT, #STRINGRIGHT, or
      #STRINGCENTER (defaults to #STRINGLEFT). (V37)
  #STRINGA_ReplaceMode (BOOL) - If TRUE, this string gadget is in
      replace-mode (defaults to FALSE (insert-mode)). (V37)
```

1.14 textgadget

```
  SYNTAX
TextGadget(#Gadget, x, y, Width, Heigth, Text$, TagList)

  COMMAND
Creates a Text gadget in the GadgetList. #Gadget will be the number
returned  by  EventGadget()  command. 'Text$' is the text initially
displayed in the TextGadget()


Available Tags:

  #GTTX_Text - Pointer to a NULL terminated string to be displayed,
      as a read-only text-display gadget, or NULL. (defaults to NULL)
      (V36)
  #GTTX_CopyText (BOOL) -  This flag instructs the text-display gadget
      to copy the supplied text string, instead of using only
      pointer to the string.  This only works for the initial value
      of GTTX_Text set at CreateGadget() time.  If you subsequently
      change GTTX_Text, the new text will be referenced by pointer,
      not copied.  Do not use this tag with a NULL GTTX_Text. (V37)
  #GTTX_Border (BOOL) - If TRUE, this flag asks for a recessed
```

```
      border to be placed around the gadget. (V36)
  #GTTX_FrontPen (UBYTE) - The pen to use when rendering the text
      (defaults to DrawInfo->dri_Pens[TEXTPEN]). (V39)
  #GTTX_BackPen (UBYTE) - The pen to use when rendering the background
      of the text (defaults to leaving the background untouched).
      (V39)
  #GTTX_Justification (UBYTE) - Determines how the text is rendered
      within the gadget box. GTJ_LEFT will make the rendering be
      flush with the left side of the gadget, GTJ_RIGHT will make it
      flush with the right side, and GTJ_CENTER will center the text
      within the gadget box. Under V39, using this tag also required
      using {GTNM_Clipped, TRUE}, otherwise the text would not show
      up in the gadget. This has been fixed in V40.
      (defaults to GTJ_LEFT). (V39)
  #GTTX_Clipped (BOOL) - Determine whether text should be clipped to
      the gadget dimensions (defaults to FALSE for gadgets without
      borders, TRUE for gadgets with borders). (V39)
```

1.15 setgadgetfont

```
  SYNTAX
SetGadgetFont(&TextAttr)

  STATEMENT
Sets the font which will be used by newly created gadgets.
It must be a TextAttr structure, so be careful when using it...
```

1.16 setgadgetflags

```
  SYNTAX
SetGadgetFlags(&Text$)

  STATEMENT
Set the flags that will be used for newly-created gadgets.
```

1.17 usegadgetlist

```
  SYNTAX
UseGadgetList(#GadgetList)

  STATEMENT
Make the specified Gadgetlist the currently-used Gadgetlist.
```

1.18 creategadgetlist

```
  SYNTAX
result.l = CreateGadgetList(#GadgetList, ScreenID)

  FUNCTION
Try to allocate the resource for a future gadgetlist.  The  ScreenID  is
needed, so be sure to pass it !
```

1.19 attachgadgetlist

```
  SYNTAX
AttachGadgetList(#GadgetList, WindowID)

  STATEMENT
Attach the specified gadgetlist to an open window specified  by  the
WindowID. Gadgets are automatically refreshed.
```

1.20 disablegadget

```
  SYNTAX
DisableGadget(#Gadget, State)

  STATEMENT
Disable or enable a gadget. If State = 1, gadget will be
disable, if State = 0 will be enabled.

NOTE: You must use the NRefreshGadget to reflect the changes
on the display.
```

1.21 activategadget

```
  SYNTAX
ActivateGadget(#Gadget)

  STATEMENT
Cause the specified gadget to be activated. Useful for StringGadget.
```

1.22 refreshgadget

```
  SYNTAX
RefreshGadget(#Gadget)

  STATEMENT
The specified gadget display will be refreshed.
```

1.23 refreshgadgetlist

```
  SYNTAX
RefreshGadgetList()

  STATEMENT
Refresh the current gadgetlist: all the gadgets will be redrawn
in the window to which they are attached.
```

1.24 nogadgetborder

```
  SYNTAX
NoGadgetBorder(#Gadget)

  STATEMENT
Must be put after a gadget declaration and will remove the border around
the specified gadget

ie: ButtonGadget 1, 10,10,100,100,"Hello",0
    NNoGadgetBorder 1
```

1.25 freegadgetlist

```
  SYNTAX
FreeGadgetList(#GadgetList)

  STATEMENT
Free the memory taken by the specified gadgetlist. Be sure that this gadget
list is not referenced again by any windows, or you will have a crash !
Call it typically after a window close.
```

1.26 setstringtext

```
  SYNTAX
SetStringText(#Gadget, Text$)

  STATEMENT
Change the text content of a string gadget.
```

1.27 getstringtext

```
  SYNTAX
Text$ = GetStringText(#Gadget)

  STATEMENT
Return the text content of a string gadget.
```

1.28 setgadgetattrs

```
  SYNTAX
SetGadgetAttrs(#Gadget, #TAG_ITEM, #TAG_DATA)

  STATEMENT
Change the attributes of the given gadget.
```