MITSUBISHI ELECTRIC

Changes for the Better

Numerical Control (CNC)

Function Reference Manual

NC Designer2

MELSOFT

IB-1501792(ENG)-A

Introduction

This instruction manual describes how to use NC Designer2. Incorrect handling may lead to unforeseen accidents, so make sure to read this instruction manual thoroughly before operation to ensure correct usage. NC Designer2 supports the following NC series. Some of the functions are not adapted to the NC Designer2, depending on its series.

Supported models	Abbreviations in this manual
M800VW (Windows-based display unit)	M800V/M80V (Windows-based display unit)
M80VW (Windows-based display unit)	
M800W (Windows-based display unit)	M800/M80 (Windows-based display unit)
M80W (Windows-based display unit)	
M800VW (Windows-less display unit)	M800V/M80V (Windows-less display unit)
M800VS	
M80VW (Windows-less display unit)	
M80V	
M800W (Windows-less display unit)	M800/M80 (Windows-less display unit)
M800S	
M80W (Windows-less display unit)	
M80	
E80	
M700VW	M700VW
M700VS	M700VS/M70V/E70
M70V	
E70	

Notes on Reading This Manual

(1) For the specifications of individual machine tools, refer to the manuals issued by the respective machine tool builders. The "restrictions" and "available functions" described by the machine tool builders have precedence over this manual.

(2) This manual describes as many special operations as possible, but it should be kept in mind that operations not mentioned in this manual cannot be performed.

In this manual, the following abbreviations might be used.
MTB: Machine tool builder

Notes on Using This Software

(1) Decimal point

Regardless of the language used on the OS which NC Designer 2 is installed on, "." is used for decimal points.

(2) Properties in property sheets and property setup dialogs for which a numerical value is entered

(a) "-" or "+" at the start if the numerical value is handled as a number.

(b) When characters (other than numbers and decimal points".") are included, the sequence of numbers and decimal points from the start are handled as numerical values.
Example: When "0123AB" is entered, it is handled as"123".

When "67@89" is entered, it is handled as "67".

(c) When multiple decimal points "." are included, the second decimal point and the values that follow are ignored.
Example: When "78.9.12" is entered, it is handled as"78.9".

(d) When two-byte numbers are entered, they are handled as a character string and the error message "It is not a numerical value" is displayed. Enter one-byte numbers.

Precautions for Safety

Always read the specifications issued by the machine tool builder, this manual, related manuals and attached documents before installation, operation, programming, maintenance or inspection to ensure correct use.
Understand this numerical controller, safety items and cautions before using the unit.
This manual ranks the safety precautions into "DANGER", "WARNING" and "CAUTION".

DANGER	When the user may be subject to imminent fatalities or major injuries if handling is mistaken.
WARNING	When the user may be subject to fatalities or major injuries if handling is mistaken.
CAUTION	When the user may be subject to injuries or when property damage may occur if handling is mistaken.

The following sings indicate prohibition and compulsory.

	This sign indicates prohibited behavior (must not do). For example,  indicates "Keep fire away".
	This sign indicated a thing that is pompously (must do). For example,  indicates "it must be grounded".

The meaning of each pictorial sing is as follows.

CAUTION	CAUTION rotated object	CAUTION HOT	Danger Electric shock risk	Danger explosive
Prohibited	Disassembly is prohibited	KEEP FIRE AWAY	General instruction	Earth ground

Note that even items ranked as " ⚠ **CAUTION**", may lead to major results depending on the situation. In any case, important information that must always be observed is described.

⚠ DANGER

Not applicable in this manual.

⚠ WARNING

Not applicable in this manual.

⚠ CAUTION

1. Items related to product and manual

⚠ If the descriptions relating to the "restrictions" and "allowable conditions" conflict between this manual and the machine tool builder's instruction manual, the latter has priority over the former.

⚠ The operations to which no reference is made in this manual should be considered impossible.

⚠ This manual is compiled on the assumption that your machine is provided with all optional functions. Confirm the functions available for your machine before proceeding to operation by referring to the specification issued by the machine tool builder.

⚠ In some NC system versions, there may be cases that different pictures appear on the screen, the machine operates in a different way on some function is not activated.

⚠ To protect the availability, integrity and confidentiality of the NC system against cyber-attacks including unauthorized access, denial-of-service (Dos) (*1) attack, and computer virus from external sources via a network, take security measures such as firewall, VPN, and anti-virus software.

(*1) Denial-of-service (Dos) refers to a type of cyber-attack that disrupts services by overloading the system or by exploiting a vulnerability of the system.

⚠ Mitsubishi Electric assumes no responsibility for any problems caused to the NC system by any type of cyber-attacks including DoS attack, unauthorized access and computer virus.

Trademarks

MELDAS, MELSEC, EZSocket, EZMotion, iQ Platform, MELSOFT, GOT, CC-Link, CC-Link/LT and CC-Link IE are either trademarks or registered trademarks of Mitsubishi Electric Corporation in Japan and/or other countries.

Ethernet is a registered trademark of Xerox Corporation in the United States and/or other countries.
TRELLIX is a registered trademark or a trademark of Musarubra US LLC or its subsidiaries in the United States and other countries.
Microsoft and Windows are either trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.
CompactFlash and CF are either trademarks or registered trademarks of SanDisk Corporation in the United States and/or other countries.
SD Logos are trademarks or registered trademarks of SD-3C, LLC in the United States, other countries or both.
Wind River Systems, Inc., VxWorks and Tornado are either trademarks or registered trademarks of Wind River Systems, Inc. in the United States and/or other countries.

Other company and product names that appear in this manual are trademarks or registered trademarks of the respective companies.

CONTENTS

1. Outline 1
    1.1 GUI Library 1
2. GDraw 3
    2.1 List of GDraw Functions 3
    2.2 Description of GDraw Functions 10
3. GResource 96
    3.1 List of GResource Functions 96
    3.2 Description of GResource Functions 98
4. GEvent 108
    4.1 List of GEvent Functions 108
    4.2 Description of GEvent Functions 109
5. GCS 117
    5.1 List of GCS Functions 117
        5.1.1 NC Designer2 Compatibility 117
        5.1.2 Button 125
        5.1.3 TextBox 126
        5.1.4 Label 128
        5.1.5 List 129
        5.1.6 Picture 131
        5.1.7 CheckBox 132
        5.1.8 RadioButton 133
        5.1.9 ProgressBar 134
        5.1.10 HtmlBrowser 134
        5.1.11 ScrollBar 136
        5.1.12 Edit 138
        5.1.13 NCTable 139
        5.1.14 InputBox 140
        5.1.15 SoftKey 142
        5.1.16 Counter 142
        5.1.17 CycleTime 142
        5.1.18 LoadMeter 143
        5.1.19 SPCommand 143
        5.1.20 NCPLCButton 143
        5.1.21 NCPLCTextBox 144
        5.1.22 NCDataTextBox 147
        5.1.23 Menu 150
        5.1.24 Extension Menu 151
        5.1.25 FileInOut 155
        5.1.26 AlarmMessage 155
        5.1.27 MonitorStatus 155
        5.1.28 NCPageChangeButton 156
        5.1.29 Graph 156
        5.1.30 AlarmList 157
        5.1.31 NC Data Access Setting 158
    5.2 Description of GCS Functions 159
        5.2.1 NC Designer2 Compatibility 159
        5.2.2 Button 304
        5.2.3 TextBox 317
        5.2.4 Label 345
        5.2.5 List 348
        5.2.6 Picture 379
        5.2.7 CheckBox 388
        5.2.8 RadioButton 400
        5.2.9 ProgressBar 412
        5.2.10 HtmlBrowser 418
        5.2.11 ScrollBar 432
        5.2.12 Edit 451
        5.2.13 NCTable 480
        5.2.14 InputBox 503
        5.2.15 SoftKey 518
        5.2.16 Counter 530
        5.2.17 CycleTime 532
        5.2.18 LoadMeter 533
        5.2.19 SPCommand 534
        5.2.20 NCPLCButton 535

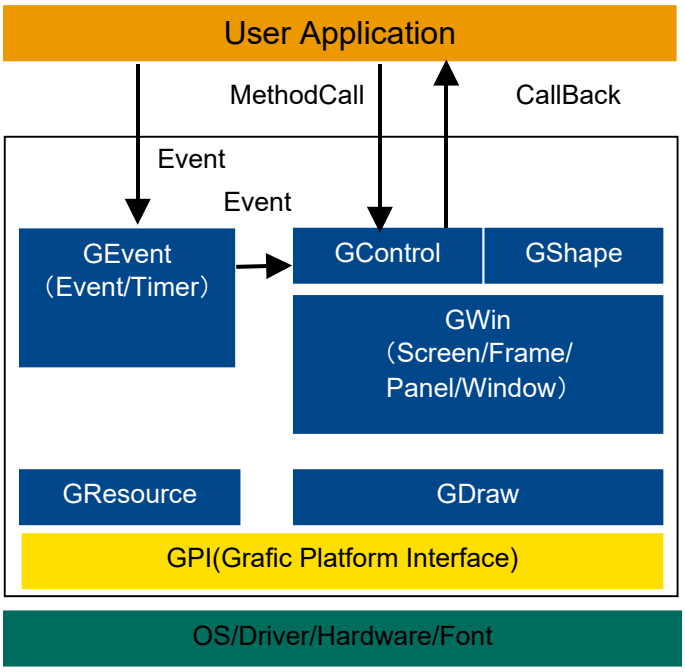
5.2.21 NCPLCTextBox 557
5.2.22 NCDataTextBox 592
5.2.23 Menu 621
5.2.24 Extension Menu 640
5.2.25 FileInOut 702
5.2.26 AlarmMessage 711
5.2.27 MonitorStatus 712
5.2.28 NCPageChangeButton 713
5.2.29 Graph 717
5.2.30 AlarmList 740
5.2.31 NC Data Access Setting 743
6. GIO 754
6.1 List of GIO Functions 754
6.2 Description of GIO Functions 755
7. GMMemory 759
7.1 List of GMMemory Functions 759
7.2 Description of GMMemory Functions 760
Appendix 767
Appendix 1. Data Type Definitions 767

1. Outline

This section gives an outline of the GUI library.

1.1 GUI Library

The GUI library is a C/C++ language GUI library that gives powerful support for GUI development. The GUI library is equipped with the functions necessary for the construction of not only event management but also GUI elements such as window systems, an it supports simple construction of GUIs. In addition, it is equipped with a Graphic Platform Interface (GPI), which is an arrangement for calling up processing dependent on hardware to make porting to various platforms easy. Porting the GUI library to other platforms is done by creating one with this GPI processing matched to the platform.



GUI Library Configuration

The functions for the various function groups are as follows.

Function Name	Outline
GEvent	Function group for managing various types of events such as the mouse, keyboard and timer and window system events such as GWin/GControl/GShape
GResource	Function group for handling resources such as character strings, 3D borders and images
GWin	Function group for window system management
GControl	Function group for controlling buttons, pictures and the like that operate on GWin
GShape	Function group for managing rectangles, circles, straight lines and other figures that operate on GWin
GDraw	Function group for making settings for the drawing environment such as basic figures, character drawing, coordinates, colors, etc.
GPI	Function group for handling parts that are dependent on hardware or OS

2. GDraw

This section describes the GDraw functions.

2.1 List of GDraw Functions

No.	Name	Type	File name	Outline
1	GDBeginDraw	void	GDBeginDraw.c	Begin drawing
2	GDBitBlt	void	GDBitBlt.c	Block transfer
3	GDCalcBorderRect	void	GDCalcBorderRect.c	Calculate rectangular area based on 3D border settings
4	GDCloseDrawHandle	void	GDCloseDrawHandle.c	Release drawing handle
5	GDDefineFillPattern	void	GDDefineFillPattern.c	Define fill pattern
6	GDDefinePenDash	void	GDDefinePenDash.c	Define line pattern
7	GDDrawArc	void	GDDrawArc.c	Draw arc
8	GDDrawBezier	void	GDDrawBezier.c	Draw Bezier curve
9	GDDrawCaptionCursorString	void	GDDrawCaptionCursorString.c	Draw caption character string (with cursor display)
10	GDDrawCaptionString	void	GDDrawCaptionString.c	Draw caption character string
11	GDDrawCaptionStringResource	void	GDDrawCaptionStringResource.c	Draw caption based on character string resources
12	GDDrawGrf	short	GDDrawGrf.c	Draw BMP
13	GDDrawImage	void	GDDrawImage.c	Draw image
14	GDDrawImageResource	void	GDDrawImageResource.c	Draw resource image
15	GDDrawOval	void	GDDrawOval.c	Draw oval
16	GDDrawOvalBorder	void	GDDrawOvalBorder.c	Draw (oval) circular 3D border
17	GDDrawOvalBorderResource	void	GDDrawOvalBorderResource.c	Draw (oval) circular resource 3D border
18	GDDrawPie	void	GDDrawPie.c	Draw sector
19	GDDrawPoly	void	GDDrawPoly.c	Draw polygon
20	GDDrawRect	void	GDDrawRect.c	Draw rectangle
21	GDDrawRectBorder	void	GDDrawRectBorder.c	Draw rectangular 3D border

No.	Name	Type	File name	Outline
22	GDDrawRectBorderResource	void	GDDrawRectBorderResource.c	Draw rectangular resource 3D border
23	GDDrawRRect	void	GDDrawRRect.c	Draw rounded corner rectangle
24	GDDrawString	void	GDDrawString.c	Draw character string
25	GDDrawStringResource	void	GDDrawStringResource.c	Draw resource character string
26	GDEndDraw	void	GDEndDraw.c	End drawing
27	GDFlushScreen	void	GDFlushScreen.c	Refresh screen
28	GDGetCharHeight	short	GDGetCharHeight.c	Get specified character height
29	GDGetCharWidth	short	GDGetCharWidth.c	Get specified character width
30	GDGetNextChar	char*	GDGetNextChar.c	Get next character
31	GDGetPalette	void	GDGetPalette.c	Get palette color code
32	GDGetPixel	GColor	GDGetPixel.c	Get color code for specified coordinates
33	GDGetPrevChar	char*	GDGetPrevChar.c	Get previous character
34	GDGetStringLength	unsigned long	GDGetStringLength.c	Get character string length
35	GDGetStringWidth	short	GDGetStringWidth.c	Get specified character width
36	GDInceptClipRect	short	GDInceptClipRect.c	Clip the area of overlap between the specified rectangle and the current clip area
37	GDInitPlatform	short	GDInitPlatform.c	Initialization processing for GUI library environment
38	GDLineTo	void	GDLineTo.c	Draw line
39	GDMoveTo	void	GDMoveTo.c	Move pen position
40	GDOffsetLocalOrigin	void	GDOffsetLocalOrigin.c	Offset to local coordinate origin
41	GDOpenDrawHandle	HGDRAW	GDOpenDrawHandle.c	Get drawing handle
42	GDQueryError	unsigned short	GDQueryError.c	Get final error code

No.	Name	Type	File name	Outline
43	GDQuitPlatform	short	GDQuitPlatform.c	Quit processing for GUI library environment
44	GDSetBackColor	void	GDSetBackColor.c	Set background color
45	GDSetBrush	void	GDSetBrush.c	Set brush
46	GDSetClipRect	void	GDSetClipRect.c	Define clipping area
47	GDSetDisplay	short	GDSetDisplay.c	Set graphics mode
48	GDSetDrawingMode	void	GDSetDrawingMode.c	Set raster operations for drawing
49	GDSetFillAttribute	void	GDSetFillAttribute.c	Set fill attributes
50	GDSetFillColor	void	GDSetFillColor.c	Set colors for foreground and background
51	GDSetFillPattern	void	GDSetFillPattern.c	Set fill pattern
52	GDSetFont	short	GDSetFont.c	Set font
53	GDSetForeColor	void	GDSetForeColor.c	Set foreground color
54	GDSetLocalMode	void	GDSetLocalMode.c	Set to local (physical) coordinate system
55	GDSetPenColor	void	GDSetPenColor.c	Set line and character color
56	GDSetPenDash	void	GDSetPenDash.c	Set line pattern
57	GDSetPenJoin	void	GDSetPenJoin.c	Specify joint for rectangles and polygons
58	GDSetPenSize	void	GDSetPenSize.c	Set line width (pen size)
59	GDSetPixel	void	GDSetPixel.c	Draw pixels
60	GDSetSimpleFont	HGFONT	GDSetSimpleFont.c	Set simple font
61	GDSetSimpleFontResource	HGFONT	GDSetSimpleFontResource.c	Set resource simple font
62	GDSetVirtualMode	void	GDSetVirtualMode.c	Set to virtual coordinate system
63	GDSetVirtualRect	void	GDSetVirtualRect.c	Set rectangular area in virtual coordinates
64	GDSetVram	short	GDSetVram.c	Set VRAM for drawing destination
65	GFGetFontAttribute	short	GFGetFontAttribute.c	Get font attributes

No.	Name	Type	File name	Outline
66	GFGetFontCount	short	GFGetFontCount.c	Get number of fonts that can be used
67	GFGetFontInformation	short	GFGetFontInformation.c	Get information for font of specified No.
68	GFGetFontUserAttribute	short	GFGetFontUserAttribute.c	Get user font attributes
69	GFGetSize	short	GFGetSize.c	Get font size
70	GFLoadFont	HGFONT	GFLoadFont.c	Load specified font
71	GFReleaseFont	void	GFReleaseFont.c	Release font
72	GFSetFontAttribute	short	GFSetFontAttribute.c	Set font attributes
73	GFSetFontUserAttribute	short	GFSetFontUserAttribute.c	Set user font attributes
74	GFSetSize	short	GFSetSize.c	Set character size
75	GRDuplicateRect	void	GRDuplicateRect.c	Copy rectangle
76	GREqualRect	short	GREqualRect.c	Confirm that rectangle areas are the same
77	GRGetRect	void	GRGetRect.c	Get rectangle area
78	GRInceptRect	short	GRInceptRect.c	Confirm rectangular area overlap
79	GRMapRect	void	GRMapRect.c	Enlarge/reduce rectangular area
80	GRMapToScreenRect	void	GRMapToScreenRect.c	Convert rectangular area defined with pgrRect to screen coordinate system
81	GRMapToVirtualRect	void	GRMapToVirtualRect.c	Convert rectangular area defined with pgrRect to virtual screen coordinate system
82	GROffsetRect	void	GROffsetRect.c	Rectangle offset processing
83	GRPhysicalToVirtualRect	void	GRPhysicalToVirtualRect.c	Convert rectangle physical coordinates to virtual coordinates

No.	Name	Type	File name	Outline
84	GRPointInRect	short	GRPointInRect.c	Check if point coordinates fall within rectangle coordinates
86	GRSetRect	void	GRSetRect.c	Define rectangular area
87	GRShiftRect	short	GRShiftRect.c	Rectangle shift processing
88	GRUnionRect	void	GRUnionRect.c	Join rectangle areas
89	GRVirtualToPhysicalRect	void	GRVirtualToPhysicalRect.c	Convert rectangular area virtual coordinates to physical coordinates
90	GLDuplicateLine	void	GLDuplicateLine.c	Copy line structure
91	GLGetLine	void	GLGetLine.c	Get line
92	GLMapLine	void	GLMapLine.c	Enlarge/reduce line coordinates
93	GLOffsetLine	void	GLOffsetLine.c	Offset processing for line structure
94	GLPhysicalToVirtualLine	void	GLPhysicalToVirtualLine.c	Convert line structure physical coordinates to virtual coordinates
95	GLSetLine	void	GLSetLine.c	Define line structure
96	GLVirtualToPhysicalLine	void	GLVirtualToPhysicalLine.c	Convert line structure virtual coordinates to physical coordinates
97	GPTDuplicatePoint	void	GPTDuplicatePoint.c	Copy point structure
98	GPTGetPoint	void	GPTGetPoint.c	Get point structure content
99	GPTMapPoint	void	GPTMapPoint.c	Map rectangle defined with pgrFrom to rectangle defined with pgrTo
100	GPTMapToScreenPoint	void	GPTMapToScreenPoint.c	Convert point structure to screen coordinates

No.	Name	Type	File name	Outline
101	GPTMapToVirtualPoint	void	GPTMapToVirtualPoint.c	Convert point defined in pgptPoint to virtual screen coordinate system
102	GPTOffsetPoint	void	GPTOffsetPoint.c	Point structure offset processing
103	GPTPhysicalToVirtualPoint	void	GPTPhysicalToVirtualPoint.c	Convert point structure physical coordinates to virtual coordinates
104	GPTSetPoint	void	GPTSetPoint.c	Define point structure
105	GPTVirtualToPhysicalPoint	void	GPTVirtualToPhysicalPoint.c	Convert point structure virtual coordinates to physical coordinates
106	GPGetPoly	void	GPGetPoly.c	Get polygon settings
107	GPMapPoly	void	GPMapPoly.c	Enlarge/reduce polygon
108	GPOffsetPoly	void	GPOffsetPoly.c	Polygon offset processing
109	GPPhysicalToVirtualPoly	void	GPPhysicalToVirtualPoly.c	Convert polygon physical coordinates to virtual coordinates
112	GMSetMemorySpace	short	GMemory.c	Register memory area information
113	GMDeleteMemorySpace	short	GMemory.c	Delete registered memory area
114	GMMemoryAlloc	void*	GMemory.c	Allocate memory area
115	GMMemoryFree	short	GMemory.c	Release memory area
116	GDCreateVram	HGVRAM	GDCreateVram.c	Create virtual VRAM
117	GDDeleteVram	void	GDDeleteVram.c	Release virtual VRAM
118	GDFlushScreenForce	void	GDFlushScreenForce.c	Forcible refresh screen
119	GDGetGradationInfo	void	GDGetGradationInfo.c	Get gradation information

No.	Name	Type	File name	Outline
120	GDGetLocalOrigin	void	GDGetLocalOrigin.c	Get local coordinate origin
121	GDGetVram	HGVRAM	GDGetVram.c	Get system VRAM
122	GDRemoveDrawableRegion	short	GDRemoveDrawableRegion.c	Delete rectangular area from drawable area
123	GDSetDrawableRegion	short	GDSetDrawableRegion.c	Set drawable area
124	GDSetGradationInfo	void	GDSetGradationInfo.c	Set gradation information
125	GDSetLocalOrigin	void	GDSetLocalOrigin.c	Set local coordinate origin

2.2 Description of GDraw Functions

GDBeginDraw	
	GDBeginDraw.c
Outline	Begin drawing
Syntax	void GDBeginDraw(HGDRAW *hDraw*)
Argument	(i/o) HGDRAW hDraw : draw handle
Return value	None
Details	This must be called just once when drawing on the screen for drawing basic figures, displaying image data and the like.
File name	GDBeginDraw.c

GDBitBlt

GDBitBlt.c

Outline Block transfer

Syntax

```
void GDBitBlt(
  HGDRAW    hDraw,
  HGVRAM    hSrc,
  short     nSrcX,
  short     nSrcY,
  short     nWidth,
  short     nHeight,
  HGVRAM    hDest,
  short     nDestX,
  short     nDestY,
  short     nDrawingMode
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) HGVRAM hSrc	: source VRAM handle
(i) short nSrcX	: source X coordinate
(i) short nSrcY	: source Y coordinate
(i) short nWidth	: transmission area width
(i) short nHeight	: transmission area height
(i) HGVRAM hDest	: destination VRAM handle
(i) nDestX	: destination X coordinate
(i) nDestY	: destination Y coordinate
(i) nDrawingMode	: raster operation

Return value None

Details Transfers the nWidth*nHeight sized area based on the specified point (nSrcX, nSrcY) in the VRAM of the transmission source (hSrc) to the nWidth*nHeight sized area based on another specified point (nDestX, nDestY) in the transmission destination (hDest).
The transfer method depends on nDrawingMode.
Corresponds to physical coordinates BitBlt only, and not virtual coordinates.

File name GDBitBlt.c

GDCalcBorderRect

GDCalcBorderRect.c

Outline Calculate rectangular area based on 3D border settings

Syntax
```
void GDCalcBorderRect(
  GRect    *pBounds,
  GRect    *pInside,
  GRect    *pBorderRect,
  GBorder  *pBorder
)
```

Argument

(i) GRect *pBounds	:	perimeter rectangle area
(o) GRect *pInside	:	rectangle area inside 3D border
(o) GRect *pBorderRect	:	3D border area
(i) GBorder *pBorder	:	3D border

Return value None

Details Calculates the position of the rectangle when a 3D border is drawn within pBounds based on the pBorder settings.
The rectangular area within the 3D border is stored in pInside.
The perimeter rectangle for the 3D border is stored in pBorderRect.

File name GDCalcBorderRect.c

GDCloseDrawHandle

GDCloseDrawHandle.c

Outline Release drawing handle

Syntax
```
void GDCloseDrawHandle(
  HGDRAW  hDraw
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle to be released

Return value None

Details The drawing handle is released.

File name GDCloseDrawHandle.c

GDDefineFillPattern

GDDefineFillPattern.c

Outline	Define fill pattern
Syntax	void GDDefineFillPattern(HGDRAW *hDraw,* short *nFillPattern,* char **pcPattern*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nFillPattern : fill pattern specification (i) char *pcPattern : fill pattern definition
Return value	None
Details	Set pcPattern in nFillPattern No. fill pattern. When a pattern No. outside of the range is set, GERROR_BAD_PATTERN_NUM is set as the error.
File name	GDDefineFillPattern.c

GDDefinePenDash

GDDefinePenDash.c

Outline	Define line pattern
Syntax	void GDDefinePenDash(HGDRAW *hDraw,* short *nPenDash,* char **pcDash*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nPenDash : line pattern No. specification (i) char *pcDash : line pattern
Return value	None
Details	Set pcDash in nPenDash No. line pattern. Since nPenDash number 0 is set to solid line, specify 1 or higher.
File name	GDDefinePenDash.c

GDDrawArc

GDDrawArc.c

Outline

Draw arc

Syntax

```
void GDDrawArc(
  HGDRAW   hDraw,
  GRect    *pgrRect,
  short    nStartAngle,
  short    nEndAngle
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle
(i) GRect *pgrRect	:	rectangle area
(i) short nStartAngle	:	10 times of the starting angle
(i) short nEndAngle	:	10 times of the ending angle

Return value

None

Details

Draws an arc in contact with the inside of the rectangle defined with pgrRect.
The starting angle is nStartAngle, and the ending angle is nEndAngle.
Line color, line type, line thickness and raster operation are enabled as drawing attributes.

File name

GDDrawArc.c

GDDrawBezier

GDDrawBezier.c

Outline	Draw Bezier curve
Syntax	void GDDrawBezier(HGDRAW *hDraw,* GPoly **pgpPoly*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GPoly *pgpPoly : polygon structure
Return value	None
Details	Draws a Bezier curve joining the points defined with pgpPoly. Line color, line type, line thickness and raster operation are enabled as drawing attributes.
File name	GDDrawBezier.c

GDDrawCaptionCursorString

GDDrawCaptionCursorString.c

Outline	Draw caption character string (with cursor display)

Syntax

```
void GDDrawCaptionCursorString(
  HGDRAW    hDraw,
  GRect     *pBounds,
  GCaption  * pCaption,
  GCursor   *pCursor,
  long      lCursorPos,
  char      *pString
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) GRect *pBounds	: rectangle area
(i) GCaption *pCaption	: caption information
(i/o) GCursor *pCursor	: data with cursor displayed
(i) long lCursorPos	: character cursor display position
(i) char *pString	: character string

Return value	None
Details	pString character string is drawn with the cursor displayed according to the pCaption settings so that it fills the pBounds rectangular area.
File name	GDDrawCaptionCursorString.c

GDDrawCaptionString

GDDrawCaptionString.c

Outline Draw caption character string

Syntax

```
void GDDrawCaptionString(
  HGDRAW    hDraw,
  GRect     *pBounds,
  GCaption  *pCaption,
  char      *pString
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle
(i) GRect *pBounds	:	rectangle area
(i) GCaption *pCaption	:	caption information
(i) char *pString	:	character string

Return value None

Details pString character string is drawn according to the pCaption settings so that it fills the pBounds rectangular area.

File name GDDrawCaptionString.c

GDDrawCaptionStringResource

GDDrawCaptionStringResource.c

Outline

Draw caption based on character string resources

Syntax

```
void GDDrawCaptionStringResource(
  HGDRAW        hDraw,
  GRect         *pBounds,
  GCaption      *pCaption,
  unsigned short usResourceID
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) GRect *pBounds	: rectangle area
(i) GCaption *pCaption	: caption information
(i) unsigned short usResourceID	: character string resource ID

Return value

None

Details

usResourceID resource character string is drawn according to the pCaption settings so that it fills the pBounds rectangular area.

File name

GDDrawCaptionStringResource.c

GDDrawGrf

GDDrawGrf.c

Outline Draw BMP

Syntax

```
short GDDrawGrf(
  HGDRAW         hDraw,
  GRFHeader      *pGHeader,
  unsigned char  *pData,
  short          nX,
  short          nY,
  short          nWidth,
  short          nHeight
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) GRFHeader *pGHeader	: BMP information structure
(i) unsigned char *pData	: BMP data
(i) short nX	: coordinates for starting drawing (upper left of BMP)
(i) short nY	: coordinates for starting drawing (upper left of BMP)
(i) short nWidth	: display all when drawing area size is 0 or lower
(i) short nHeight	: display all when drawing area size is 0 or lower

Return value

FALSE : drawing failed
TRUE : drawing succeeded

Details The pData image having pGHeader information is drawn in the size of nWidth*nHeight starting at (nX,nY).
When 0 is specified for nWidth, it is displayed in the size of the image.
When 0 is specified for nHeight, it is displayed in the size of the image.

File name GDDrawGrf.c

GDDrawImage

GDDrawImage.c

Outline

Draw image

Syntax

```
void GDDrawImage(
  HGDRAW   hDraw,
  GImage   *pImage,
  short    nX,
  short    nY,
  short    nWidth,
  shor     nHeight
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) GImage *pImage	: image
(i) short nX	: drawing start position (x coordinate)
(i) short nY	: drawing start position (y coordinate)
(i) short nWidth	: drawing area width
(i) short nHeight	: drawing area height

Return value

None

Details

Draws the GImage image in the size of nWidth*nHeight starting at (nX, nY).
GImage contains the data for any of the images that are supported in the GUI library as standard format data, and the processing function is called according to the type of image.
At present this is only Graphic Raster Format (GRF)

File name

GDDrawImage.c

GDDrawImageResource

GDDrawImageResource.c

Outline Draw resource image

Syntax

```
void GDDrawImageResource(
  HGDRAW          hDraw,
  unsigned short  usResourceID,
  short           nX,
  short           nY,
  short           nWidth,
  short           nHeight
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) usResourceID	: image resource ID
(i) short nX	: drawing start position (X coordinate)
(i) short nY	: drawing start position (Y coordinate)
(i) short nWidth	: drawing area width
(i) short nHeight	: drawing area height

Return value None

Details Draws the usResourceID resource image in the size of nWidth*nHeight starting at (nX, nY).

File name GDDrawImageResource.c

GDDrawOval

GDDrawOval.c

Outline	Draw oval
Syntax	void GDDrawOval(HGDRAW *hDraw,* GRect **pgrRect*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GRect *pgrRect : rectangle area
Return value	None
Details	Draws an (oval) circle in contact with the inside of the rectangle defined with pgrRect. For the fill inside the shape, the fill pattern, foreground color and background color, and for the perimeter, line color, line thickness, line type and raster operation are enabled as drawing attributes.
File name	GDDrawOval.c

GDDrawOvalBorder

GDDrawOvalBorder.c

Outline

Draw (oval) circular 3D border

Syntax

```
void GDDrawOvalBorder(
  HGDRAW   hDraw,
  GRect    *pBounds,
  GRect    *pInside,
  GBorder  *pBorder,
  short    nStatus
)
```

Argument

(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pBounds : rectangle area
(i) GRect *pInside : rectangle in contact inside
(i) GBorder *pBorder : 3D border
(i) short nStatus : status
FALSE : convex status
TRUE : concave status

Return value

None

Details

Draws the 3D border in pBorder in an (oval) circle shape in the size of the rectangle in contact with the inside of the rectangular area in pBounds. When, in this instance, nStatus is FALSE, filling of the upper left and lower right borders is done with the colors that have been set, but if it is TRUE, the colors for filling of the upper left and lower right borders is reversed.

File name

GDDrawOvalBorder.c

GDDrawOvalBorderResource

GDDrawOvalBorderResource.c

Outline Draw (oval) circular resource 3D border

Syntax

```
void GDDrawOvalBorderResource(
  HGDRAW          hDraw,
  GRect           *pBounds,
  GRect           *pInside,
  unsigned short  usResourceID,
  short           nStatus
)
```

Argument

(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pBounds : rectangle area
(i) GRect *pInside : rectangle in contact inside
(i) unsigned short usResourceID : 3D border resource ID
(i) short nStatus : status
FALSE : convex status
TRUE : concave status

Return value None

Details Draws the 3D border in usResourceID in an (oval) circle shape in the size of the rectangle in contact with the inside of the rectangular area in pBounds. When, in this instance, nStatus is FALSE, filling of the upper left and lower right borders is done with the colors that have been set, but if it is TRUE, the colors for filling of the upper left and lower right borders is reversed.

File name GDDrawOvalBorderResource.c

GDDrawPie

GDDrawPie.c

Outline

Draw sector

Syntax

```
void GDDrawPie(
  HGDRAW   hDraw,
  GRect    *pgrRect,
  short    nStartAngle,
  short    nEndAngle
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) GRect *pgrRect	: rectangle area
(i) short nStartAngle	: 10 times of the starting angle
(i) short nEndAngle	: 10 times of the ending angle

Return value

None

Details

Draws a sector in contact with the inside of the rectangle defined with pgrRect.
The starting angle is nStartAngle, and the ending angle is nEndAngle.
For the fill inside the shape, the fill pattern, foreground color and background color, and for the perimeter, line color, line thickness, line type and raster operation are enabled as drawing attributes.

File name

GDDrawPie.c

GDDrawPoly

GDDrawPoly.c

Outline Draw polygon

Syntax
```
void    GDDrawPoly(
  HGDRAW   hDraw,
  GPoly    *pgpPoly
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GPoly *pgpPoly : polygon structure

Return value None

Details Draws the polygon defined in pgpPoly.
For the fill inside the shape, the fill pattern, foreground color and background color, and for the perimeter, line color, line thickness, line type and raster operation are enabled as drawing attributes.

File name GDDrawPoly.c

GDDrawRect

GDDrawRect.c

Outline Draw rectangle

Syntax
```
void GDDrawRect(
  HGDRAW   hDraw,
  GRect    *pgrRect
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pgrRect : rectangle area

Return value None

Details Draws the rectangle in the rectangular area defined in pgrRect.
For the fill inside the shape, the fill pattern, foreground color and background color, and for the perimeter, line color, line thickness, line type and raster operation are enabled as drawing attributes.

File name GDDrawRect.c

GDDrawRectBorder

GDDrawRectBorder.c

Outline Draw rectangular 3D border

Syntax

```
void GDDrawRectBorder(
  HGDRAW   hDraw,
  GRect    *pBounds,
  GRect    *pInside,
  GBorder  *pBorder,
  short    nStatus
)
```

Argument

(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pBounds : rectangle area
(i) GRect *pInside : rectangle in contact inside
(i) GBorder *pBorder : 3D border
(i) short nStatus : status
FALSE : convex status
TRUE : concave status

Return value None

Details Draws a 3D border in pBorder in an rectangle in the size of the rectangle in contact with the inside of the rectangular area in pBounds.
When, in this instance, nStatus is FALSE, filling of the upper left and lower right borders is done with the colors that have been set, but if it is TRUE, the colors for filling of the upper left and lower right borders is reversed.

File name GDDrawRectBorder.c

GDDrawRectBorderResource

GDDrawRectBorderResource.c

Outline Draw resource rectangular 3D border

Syntax

```
void GDDrawRectBorderResource(
  HGDRAW          hDraw,
  GRect           *pBounds,
  GRect           *pInside,
  unsigned short  usResourceID,
  short           nStatus
)
```

Argument

(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pBounds : rectangle area
(i) GRect *pInside : rectangle in contact inside
(i) unsigned short usResourceID : 3D border
(i) short nStatus : status
FALSE : convex status
TRUE : concave status

Return value None

Details Draws a border in usResourceID in a rectangle in the size of the rectangle in contact with the inside of the rectangular area in pBounds. When, in this instance, nStatus is FALSE, filling of the upper left and lower right borders is done with the colors that have been set, but if it is TRUE, the colors for filling of the upper left and lower right borders is reversed.

File name GDDrawRectBorderResource.c

GDDrawRRect

GDDrawRRect.c

Outline Draw rounded corner rectangle

Syntax

```
void GDDrawRRect(
  HGDRAW   hDraw,
  GRect    *pgrRect,
  short    nX,
  short    nY
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) GRect *pgrRect	: rectangle area
(i) short nX	: X axis of corner arc
(i) short nY	: Y axis of corner arc

Return value None

Details Draws the rounded corner rectangle in the rectangular area defined in pgrRect.
The rounding of the corners is defined with nX and nY.
For the fill inside the shape, the fill pattern, foreground color and background color, and for the perimeter, line color, line thickness, line type and raster operation are enabled as drawing attributes.

File name GDDrawRRect.c

GDDrawString

GDDrawString.c

Outline	Draw character string
Syntax	void GDDrawString(HGDRAW *hDraw,* char **pszString*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) char *pszString : draw character string
Return value	None
Details	Draws the pszString character string starting at the current pen position. Pen color, background color and text mode are enabled as the drawing attributes.
File name	GDDrawString.c

GDDrawStringResource

GDDrawStringResource.c

Outline	Draw resource character string
Syntax	void GDDrawStringResource(HGDRAW *hDraw,* unsigned short *usResourceID*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) unsigned short usResourceID : draw resource character string
Return value	None
Details	Draws the usResourceID resource character string starting at the current pen position. Pen color, background color and text mode are enabled as the drawing attributes.
File name	GDDrawStringResource.c

GDEndDraw

GDEndDraw.c

Outline End drawing

Syntax
```
void GDEndDraw(
  HGDRAW   hDraw
)
```

Argument (i/o) HGDRAW hDraw : drawing handle

Return value None

Details Quits screen drawing. This is called after drawing on the screen for basic figure drawing, data display and the like.

File name GDEndDraw.c

GDFlushScreen

GDFlushScreen.c

Outline Refresh screen

Syntax
```
void GDFlushScreen(
  HGDRAW   hDraw,
  short    nX,
  short    nY,
  short    nWidth,
  short    nHeight
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) short nX : X coordinate for drawing starting point
(i) short nY : Y coordinate for drawing starting point
(i) short nWidth : drawing width
(i) short nHeight : drawing height

Return value None

Details A screen for the size of nWidth*nHeight is refreshed in the current coordinate system with (nX, nY) as the starting point.

File name GDFlushScreen.c

GDGetCharHeight

GDGetCharHeight.c

Outline	Get specified character height
Syntax	short GDGetCharHeight(HGDRAW *hDraw,* short *nCharCode*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nCharCode : character code
Return value	Character height
Details	Get specified character height with current font settings.
File name	GDGetCharHeight.c

GDGetCharWidth

GDGetCharWidth.c

Outline	Get specified character width
Syntax	short GDGetCharWidth(HGDRAW *hDraw,* short *nCharCode*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nCharCode : character code
Return value	Character width
Details	Get specified character width with current font settings.
File name	GDGetCharWidth.c

GDGetNextChar

GDGetNextChar.c

Outline Get next character

Syntax

```
char* GDGetNextChar(
  HGDRAW   hDraw,
  char     *pszString,
  char     *pszNow
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) char *pszString	: character string
(i) char *pszNow	: character position - starting point for the next character search

Return value Pointer to the next character of the specified character

Details Searches for the next character using the specified character in the specified character string as the starting point.

File name GDGetNextChar.c

GDGetPalette

GDGetPalette.c

Outline	Get palette color code
Syntax	void GDGetPalette(HGDRAW *hDraw,* short *nStart,* short *nCount,* GColor **pgcColorArray*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nStart : starting No. to get (i) short nCount : number of colors to get (o) GColor *pgcColorArray : array storing color code
Return value	None
Details	nCount color codes starting at nStart in the color pallet are acquired in pgcColorArray.
File name	GDGetPalette.c

GDGetPixel

GDGetPixel.c

Outline Get color code for specified coordinates

Syntax

```
GColor GDGetPixel(
  HGDRAW   hDraw,
  short    nX,
  short    nY
)
```

Argument

(i/o) HGDRAW	hDraw	: drawing handle
(i) short	nX	: X coordinate
(i) short	nY	: Y coordinate

Return value Color code

Details Gets the color code for (nX, nY).
The color code depends on the current color environment.

File name GDGetPixel.c

GDGetPrevChar

GDGetPrevChar.c

Outline Get previous character

Syntax
```
char* GDGetPrevChar(
  HGDRAW   hDraw,
  char     * pszString,
  char     * pszNow
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) char *pszString : character string
(i) char *pszNow : character position - starting point for the previous character search

Character position

Return value Pointer to the previous character of the specified character

Details Searches for the previous character using the specified character in the specified character string as the starting point.

File name GDGetPrevChar.c

GDGetStringLength

GDGetStringLength.c

Outline Get character length

Syntax
```
unsigned long GDGetStringLength(
  HGDRAW   hDraw,
  char     *pszString
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) char *pszString : character string in which the characters are to be counted

Return value Number of characters

Details Gets the number of characters in the specified character string.

File name GDGetStringLength.c

GDGetStringWidth

GDGetStringWidth.c

Outline Get specified character width

Syntax

```
short GDGetStringWidth(
  HGDRAW   hDraw,
  char     *pszString
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle
(i) char *pszString	:	character string

Return value Character string width

Details Gets the width of a character string specified by pszString.

File name GDGetStringWidth.c

GDInceptClipRect

GDInceptClipRect.c

Outline Clip the area of overlap between the specified rectangle and the current clip area

Syntax
```
short GDInceptClipRect(
  HGDRAW    hDraw,
  GRect     *pgrClip,
  GRect     *pOldClip
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pgrClip : rectangle area
(o) GRect *pOldClip : clip area before change

Return value
FALSE : clipping failed
TRUE : clipping succeeded

Details Defines the portion of overlap between the pgrClip rectangular area displaying characters or graphics on the screen and the current clip area.
The clip area up to that point is stored in pOldClip.
Either physical or virtual coordinates can be used for the clip area, and the setting is applied to the coordinate system at the time this function is called.

File name GDInceptClipRect.c

GDInitPlatform

GDInitPlatform.c

Outline Carrying out initialization processing for GUI library environment

Syntax
```
short GDInitPlatform(
  void    *pParameter
)
```

Argument
(i) void* pParameter : additional platform information

Return value
FALSE : initialization failed
TRUE : initialization succeeded

Details Initializes the platform that uses the GUI library.

File name GDInitPlatform.c

GDLineTo

GDLineTo.c

Outline Draw line

Syntax
```
void GDLineTo(
  HGDRAW   hDraw,
  short    nX,
  short    nY
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle
(i) short nX	:	X coordinate
(i) short nY	:	Y coordinate

Return value None

Details Draws a line from the current pen position to (nX, nY).

File name GDLineTo.c

GDMoveTo

GDMoveTo.c

Outline Move pen position

Syntax
```
void GDMoveTo(
  HGDRAW   hDraw,
  short    nX,
  short    nY
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle
(i) short nX	:	X coordinate
(i) short nY	:	Y coordinate

Return value None

Details Moves pen position to (nX, nY).

File name GDMoveTo.c

GDOffsetLocalOrigin

GDOffsetLocalOrigin.c

Outline	Offset to local coordinate origin
Syntax	void GDOffsetLocalOrigin(HGDRAW *hDraw,* GPoint **pgptPoint*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GPoint *pgptPoint : point structure
Return value	None
Details	An offset of pgptPoint.nX in the X direction and pgptPoint.nY in the Y direction is applied to the origin for the local (physical) coordinates.
File name	GDOffsetLocalOrigin.c

GDOpenDrawHandle

GDOpenDrawHandle.c

Outline	Get drawing handle
Syntax	HGDRAW GDOpenDrawHandle(short *nType,* void **pParameter*)
Argument	(i) short nType : display mode (i) void *pParameter : additional handle information
Return value	NULL : acquisition failed Other than NULL : drawing handle
Details	Gets the drawing handle.
File name	GDOpenDrawHandle.c

GDQueryError

GDQueryError.c

Outline	Get final error code
Syntax	unsigned short GDQueryError(HGDRAW *hDraw*)
Argument	(i/o) HGDRAW hDraw : drawing handle
Return value	Error No.
Details	The final error code within the GUI library is acquired.
File name	GDQueryError.c

GDQuitPlatform

GDQuitPlatform.c

Outline	Quit processing for GUI library environment
Syntax	short GDQuitPlatform()
Argument	None
Return value	FALSE : failed TRUE : succeeded
Details	Carries out end processing for the platform environment used by the GUI library.
File name	GDQuitPlatform.c

GDSetBackColor

GDSetBackColor.c

Outline Set background color

Syntax
```
void GDSetBackColor(
  HGDRAW    hDraw,
  GColor    gcBackColor
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GColor gcBackColor : background color to be set

Return value None

Details Sets the background color.
When NO_DRAW is set for the background color, the background color pixels are not filled in during drawing.

NO_DRAW : no background fill
Color code : fill background using specified color code

File name GDSetBackColor.c

GDSetBrush

GDSetBrush.c

Outline Set brush

Syntax
```
void GDSetBrush(
  HGDRAW    hDraw,
  GBrush    *pBrush
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GBrush *pBrush : brush being set

Return value None

Details The brush to be used is set as a fill attribute.
The brush set is registered as a fill pattern, foreground color and background color setting.

File name GDSetBrush.c

GDSetClipRect

GDSetClipRect

Outline	Define clipping area
Syntax	void GDSetClipRect(HGDRAW *hDraw,* GRect **pgrClip*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GRect *pgrClip : rectangle area
Return value	None
Details	A rectangular area for displaying characters or graphics on the screen is set as pgrClip. Either physical or virtual coordinates can be used for the clip area, and the setting is applied to the coordinate system at the time this function is called.
File name	GDSetClipRect.c

GDSetDisplay

GDSetDisplay.c

Outline	Set graphics mode
Syntax	short GDSetDisplay(HGDRAW *hDraw,* short *nType*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nType : graphics mode to be set
Return value	FALSE : setting failed TRUE : setting succeeded
Details	Sets the graphics mode. The mode depends on the platform, so it is not defined as the GUI library.
File name	GDSetDisplay.c

GDSetDrawingMode

GDSetDrawingMode.c

Outline	Set raster operations for drawing
Syntax	void GDSetDrawingMode(HGDRAW *hDraw,* short *nMode*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nMode : raster operation
Return value	None
Details	Sets the raster operation for drawing. GRASTER_OPERATION_REPLACE : overwrite GRASTER_OPERATION_AND : AND with base GRASTER_OPERATION_OR : OR with base GRASTER_OPERATION_XOR : XOR with base
File name	GDSetDrawingMode.c

GDSetFillAttribute

GDSetFillAttribute.c

Outline Set fill attributes

Syntax
```
void GDSetFillAttribute(
  HGDRAW   hDraw,
  short    nPattern,
  GColor   gcForeColor,
  GColor   gcBackColor
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) short nPattern : fill pattern
(i) GColor gcForeColor : foreground color
(i) GColor gcBackColor : background color

Return value None

Details Sets the fill attributes.

nPattern
- NO_DRAW : no fill
- FILL_FORE_COLOR : fill with single foreground color
- FILL_BACK_COLOR : fill with single background color
- 0 to 37 : fill using pattern for specified No.

gcForeColor
- NO_DRAW : no foreground fill
- Color code : fill foreground using specified color code

gcBackColor
- NO_DRAW : no background fill
- Color code : fill background using specified color code

File name GDSetFillAttribute.c

GDSetFillColor

GDSetFillColor.c

Outline

Set colors for foreground and background

Syntax

```
void GDSetFillColor(
  HGDRAW   hDraw,
  GColor   gcForeColor,
  GColor   gcBackColor
)
```

Argument

(i/o) HGDRAW hDraw : drawing handle
(i) GColor gcForeColort : foreground color to be se
(i) GColor gcBackColor : background color to be set

Return value

None

Details

Sets the foreground color and background color.
gcForeColor
NO_DRAW : no foreground fill
Color code : fill foreground using specified color code
gcBackColor
NO_DRAW : no background fill
Color code : fill background using specified color code

File name

GDSetFillColor.c

GDSetFillPattern

GDSetFillPattern.c

Outline	Set fill pattern
Syntax	void GDSetFillPattern(HGDRAW *hDraw,* short *nPattern*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nPattern : fill pattern
Return value	None
Details	Sets the fill pattern. nPattern NO_DRAW : no fill FILL_FORE_COLOR : fill with single foreground color FILL_BACK_COLOR : fill with single background color 0 to 37 : fill using pattern for specified No.
File name	GDSetFillPattern.c

GDSetFont

GDSetFont.c

Outline Set font

Syntax
```
short GDSetFont(
  HGDRAW   hDraw,
  HGFONT   hFont
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) HGFONT hFont : font handle

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Set the hFont font as the font for character drawing.

File name GDSetFont.c

GDSetForeColor

GDSetForeColor.c

Outline Set foreground color

Syntax
```
void GDSetForeColor(
  HGDRAW   hDraw,
  GColor   gcForeColor
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GColor gcForeColor : foreground color to be set

Return value None

Details Sets the foreground color.
gcForeColor
NO_DRAW : no foreground fill
Color code : fill foreground using specified color code

File name GDSetForeColor.c

GDSetLocalMode

GDSetLocalMode.c

Outline Set to local (physical) coordinate system

Syntax
```
void GDSetLocalMode(
  HGDRAW   hDraw
)
```

Argument (i/o) HGDRAW hDraw : drawing handle

Return value None

Details Sets the coordinate system to local (physical) coordinates.
After the coordinate system is set to local coordinates, all GUI library functions are executed using the local (physical) coordinates system.

File name GDSetLocalMode.c

GDSetPenColor

GDSetPenColor.c

Outline Set line and character color

Syntax
```
void GDSetPenColor(
  HGDRAW   hDraw,
  GColor   gcPenColor
)
```

Argument (i/o) HGDRAW hDraw : drawing handle
(i) GColor gcPenColor : color to be set for lines/characters

Return value None

Details Sets the color for lines and characters.
gcPenColor
NO_DRAW : no drawing
Color code : draw lines and characters using the specified color code

File name GDSetPenColor.c

GDSetPenDash

GDSetPenDash.c

Outline	Set line pattern
Syntax	void GDSetPenDash(HGDRAW *hDraw,* short *nDash*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nDash : line pattern
Return value	None
Details	Specifies the line pattern to use when drawing lines. nDash 0 : solid line 1 to 8 : defined line pattern
File name	GDSetPenDash.c

GDSetPenJoin

GDSetPenJoin.c

Outline	Specify joint for rectangles and polygons + D968
Syntax	void GDSetPenJoin(HGDRAW *hDraw,* short *nJoin*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) short nJoin : type of joint
Return value	None
Details	Specifies the joint for rectangle and polygon perimeters. nJoin NONE : none BEVEL : line ROUND : rounded corners
File name	GDSetPenJoin.c

GDSetPenSize

GDSetPenSize.c

Outline Set line width (pen size)

Syntax
```
void GDSetPenSize(
  HGDRAW   hDraw,
  short    nX,
  short    nY
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) short nX	: thickness in X direction
(i) short nY	: thickness in Y direction

Return value None

Details Sets the thickness of lines and perimeter lines.
nX is the horizontal thickness, and nY is the vertical thickness.

File name GDSetPenSize.c

GDSetPixel

GDSetPixel.c

Outline Draw pixels

Syntax
```
void GDSetPixel(
  HGDRAW   hDraw,
  short    nX,
  short    nY
)
```

Argument

(i/o) HGDRAW hDraw	: drawing handle
(i) short nX	: X coordinate
(i) short nY	: Y coordinate

Return value None

Details Draws a point in the specified (nX, nY) coordinates.
After point drawing, the current position changes to the specified position.

File name GDSetPixel.c

GDSetSimpleFont

GDSetSimpleFont.c

Outline Set simple font

Syntax
```
HGFONT GDSetSimpleFont(
  HGDRAW       hDraw,
  GSimpleFont  *pFont
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i)GSimpleFont *pFont : set simple font

Return value Font handle set

Details Changes the current font to a simple font pFont.

File name GDSetSimpleFont.c

GDSetSimpleFontResource

GDSetSimpleFontResource.c

Outline Set resource simple font

Syntax
```
HGFONT GDSetSimpleFontResource(
  HGDRAW          hDraw,
  unsigned short  usResourceID
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) unsigned short usResourceID : set simple font

Return value Font handle set

Details Changes the current font to a resource simple font usResourceID.

File name GDSetSimpleFontResource.c

GDSetVirtualMode

GDSetVirtualMode.c

Outline	Set to virtual coordinate system
Syntax	void GDSetVirtualMode(HGDRAW *hDraw*)
Argument	(i/o) HGDRAW hDraw : drawing handle
Return value	None
Details	The coordinate system is set to the virtual coordinate system. After the coordinate system is set to the virtual coordinate system, all GUI library functions are executed using the virtual coordinate system.
File name	GDSetVirtualMode.c

GDSetVirtualRect

GDSetVirtualRect.c

Outline	Set rectangular area in virtual coordinates
Syntax	void GDSetVirtualRect(HGDRAW *hDraw,* GRect **pgrVirtual*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GRect *pgrVirtual : virtual coordinate rectangle area
Return value	None
Details	The rectangular area for the virtual coordinate system is set in pgrVirtual.
File name	GDSetVirtualRect.c

GDSetVram

GDSetVram.c

Outline Set VRAM for drawing destination

Syntax
```
short GDSetVram(
  HGDRAW    hDraw,
  short     nType,
  HGVRAM    hVram
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) short nType
  DEFAULT_SCREEN : set actual VRAM for drawing destination
  MEMORY_SCREEN : set virtual VRAM for drawing destination
(i) HGVRAM hVram : VRAM handle when specifying virtual VRAM

Return value
FALSE : failed
TRUE : succeeded

Details The drawing destination VRAM is specified in hVram.

File name GDSetVram.c

GFGetFontAttribute

GFGetFontAttribute.c

Outline Get font attributes

Syntax
```
short GFGetFontAttribute(
  HGFONT          hFont,
  GFontAttribute  *pgfaAttribute
)
```

Argument
(i) HGFONT hFont : font
(o) GFontattribute *pgfaAttribute : attribute

Return value
FALSE : failed
TRUE : succeeded

Details Gets the attributes for the font handle font.

File name GFGetFontAttribute.c

GFGetFontCount

GFGetFontCount.c

Outline	Get number of fonts that can be used
Syntax	short GFGetFontCount()
Argument	None
Return value	Number of fonts
Details	This is supported by the current platform. Gets fonts.
File name	GFGetFontCount.c

GFGetFontInformation

GFGetFontInformation.c

Outline	Get font information for the specified number
Syntax	short GFGetFontInformation(short *nIndex,* GFont ***pgfFont*)
Argument	(i) short nIndex : font No. (o) GFont **pgfFont : font information
Return value	FALSE : failed TRUE : succeeded
Details	This is supported by the platform. Gets the font information for the specified font No.
File name	GFGetFontInformation.c

GFGetFontUserAttribute

GFGetFontUserAttribute.c

Outline Get user font attributes

Syntax
```
short GFGetFontUserAttribute(
  HGFONT   hFont,
  short    *pnAttributeArray
)
```

Argument
(i) HGFONT hFont : font
(o) short *pnAttributeArray : attribute unique to platform

Return value
FALSE : failed
TRUE : succeeded

Details Gets the font attributes peculiar to the font handle.

File name GFGetFontUserAttribute.c

GFGetSize

GFGetSize.c

Outline Get font size

Syntax
```
short GFGetSize(
  HGFONT   hFont,
  short    *nWidth,
  short    *nHeight
)
```

Argument
(i) HGFONT hFont : font
(o) short *nWidth : width
(o) short *nHeight : height

Return value
FALSE : failed
TRUE : succeeded

Details Gets the current font size for the specified font handle.

File name GFGetSize.c

GFLoadFont

GFLoadFont.c

Outline	Load specified font
Syntax	HGFONT GFLoadFont(char *pszFontName)
Argument	(i) char *pszFontName : font name
Return value	Font handle
Details	Sets the specified font to a usable status and returns the handle.
File name	GFLoadFont.c

GFReleaseFont

GFReleaseFont.c

Outline	Release font
Syntax	void GFReleaseFont(HGFONT hFont)
Argument	(i) HGFONT hFont : font
Return value	None
Details	Releases the font for the specified handle.
File name	GFReleaseFont.c

GFSetFontAttribute

GFSetFontAttribute.c

Outline Set font attributes

Syntax
```
short GFSetFontAttribute(
  HGFONT          hFont,
  GFontAttribute  *pgfaAttribute
)
```

Argument
(i) HGFONT hFont : font
(i) GFontattribute *pgfaAttribute : attribute

Return value
FALSE : failed
TRUE : succeeded

Details Sets the attributes for the specified font.

File name GFSetFontAttribute.c

GFSetFontUserAttribute

GFSetFontUserAttribute.c

Outline Set font user attributes

Syntax
```
short GFSetFontUserAttribute(
  HGFONT  hFont,
  short   *pnAttributeArray
)
```

Argument
(i) HGFONT hFont : font
(i) short *pnAttributeArray : attribute unique to platform

Return value
FALSE : failed
TRUE : succeeded

Details Sets the font attributes peculiar to the font handle.

File name GFSetFontUserAttribute.c

GFSetSize

GFSetSize.c

Outline Set character size

Syntax
```
short GFSetSize(
  HGFONT   hFont,
  short    nWidth,
  short    nHeight
)
```

Argument
(i) HGFONT hFont : font being specified
(i) short nWidth : width
(i) short nHeight : height

Return value
FALSE : failed
TRUE : succeeded

Details Sets the character size for the specified font.

File name GFSetSize.c

GRDuplicateRect

GRDuplicateRect.c

Outline Copy rectangle

Syntax
```
void GRDuplicateRect(
  GRect   *pgrSrc,
  GRect   *pgrDest
)
```

Argument
(i) GRect *pgrSrc : copy source
(o) GRect *pgrDest : copy destination

Return value None

Details Copies the rectangular area defined in pgrSrc to the pgrDest rectangular area.

File name GRDuplicateRect.c

GREqualRect

GREqualRect.c

Outline	Confirm that rectangle areas are the same
Syntax	short GREqualRect(GRect **pgrSrc1,* GRect **pgrSrc2*)
Argument	(i) GRect *pgrSrc1 : rectangle area (i) GRect *pgrSrc2 : rectangle area
Return value	FALSE : not the same TRUE : same
Details	Finds out if the rectangular areas defined in pgrSrc1 and pgrSrc2 are the same.
File name	GREqualRect.c

GRGetRect

GRGetRect.c

Outline Get rectangle area

Syntax

```
void GRGetRect(
  GRect   *pgrRect,
  short   *nX1,
  short   *nY1,
  short   *nX2,
  short   *nY2
)
```

Argument

(i) GRect *pgrRect	:	rectangle area
(o) short *nX1	:	X coordinate for upper left corner
(o) short *nY1	:	Y coordinate for upper left corner
(o) short *nX2	:	X coordinate for lower right corner
(o) short *nY2	:	Y coordinate for lower right corner

Return value None

Details Gets the rectangle area.
The rectangle's upper left corner coordinates will be set in (nX1, nY1) and the lower right corner coordinates will be set in (nX2, nY2).

File name GRGetRect.c

GRInceptRect

GRInceptRect.c

Outline Confirm rectangular area overlap

Syntax
```
short GRInceptRect(
  GRect    *pgrSrc1,
  GRect    *pgrSrc2,
  GRect    *pgrDest
)
```

Argument
(i) GRect *pgrSrc1 : rectangle area
(i) GRect *pgrSrc2 : rectangle area
(o) GRect *pgrDest : rectangle area

Return value
FALSE : pgrSrc1 and pgrSrc2 do not overlap. (pgrDest not determined)
TRUE : pgrSrc1 and pgrSrc2 overlap.

Details Finds out if the two rectangular areas in pgrSrc1 and pgrSrc2 overlap.
If they overlap, set that area in pgrDest.

File name GRInceptRect.c

GRMapRect

GRMapRect.c

Outline Enlarge/reduce rectangular area

Syntax
```
void GRMapRect(
  GRect   *pgrRect,
  GRect   *pgrFrom,
  GRect   *pgrTo
)
```

Argument
(i/o) GRect *pgrRect : rectangle area
(i) GRect *pgrFrom : rectangle area
(i) GRect *pgrTo : rectangle area

Return value None

Details Converts the rectangular area defined in pgrRect in the same proportion as the conversion of the rectangular area defined in pgrFrom to the rectangular area defined in PgrTo.

File name GRMapRect.c

GRMapToScreenRect

GRMapToScreenRect.c

Outline Convert rectangular area defined in pgrRect to actual screen coordinate system

Syntax
```
void GRMapToScreenRect(
  HGDRAW   hDraw,
  GRect    *pgrRect
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i/o) GRect *pgrRect : rectangle

Return value None

Details Converts the rectangular area defined in pgrRect to the actual screen coordinate system

File name GRMapToScreenRect.c

GRMapToVirtualRect

GRMapToVirtualRect.c

Outline	Convert the rectangular area defined in pgrRect to the virtual screen coordinate system
Syntax	void GRMapToVirtualRect(HGDRAW *hDraw,* GRect **pgrRect*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GRect *pgrRect : rectangular area
Return value	None
Details	Converts the rectangular area defined in pgrRect to the virtual screen coordinate system
File name	GRMapToVirtualRect.c

GROffsetRect

GROffsetRect.c

Outline	Rectangle offset processing
Syntax	void GROffsetRect(GRect **pgrRect,* short *nDx,* short *nDy*)
Argument	(i/o) GRect *pgrRect : source rectangular area (i) short nDx : increase in X direction (i) short nDy : increase in Y direction
Return value	None
Details	Moves the rectangle defined in pgrRect nDx and nDy in the X direction and Y direction, respectively.
File name	GROffsetRect.c

GRPhysicalToVirtualRect

GRPhysicalToVirtualRect.c

Outline	Convert rectangle physical coordinates to virtual coordinates
Syntax	void GRPhysicalToVirtualRect(HGDRAW *hDraw,* GRect **pgrRect*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i/o) GRect *pgrRect : rectangle area
Return value	None
Details	Converts the rectangular area defined in pgrRect from physical (local) coordinates to virtual coordinates.
File name	GRPhysicalToVirtualRect.c

GRPointInRect

GRPointInRect.c

Outline	Check if point coordinates fall within rectangle coordinates
Syntax	short GRPointInRect(GPoint **pptPoint,* GRect **pgrRect*)
Argument	(i) GPoint *pptPoint : point coordinates (i) GRect *pgrRect : rectangle area
Return value	FALSE : not inside TRUE : inside (including on the pgrRect)
Details	Finds out if the point coordinates defined in pptPoint are within the rectangular area defined in pgrRect.
File name	GRPointInRect.c

GRResizeRect

GRResizeRect.c

Outline Resize rectangle

Syntax

```
void GRResizeRect(
  GRect   *pgrRect,
  short   nDx,
  short   nDy
)
```

Argument

(i/o) GRect *pgrRect	: rectangle area
(i) short nDx	: increase in X direction
(i) short nDy	: increase in Y direction

Return value None

Details nDx is subtracted from the initial X coordinate of pgrRect and nDx is added to the final X coordinate.

Furthermore, nDy is subtracted from the initial Y coordinate and nDy is added to the final Y coordinate.

File name GRResizeRect.c

GRSetRect

GRSetRect.c

Outline Define rectangular area

Syntax

```
void GRSetRect(
  GRect   *pgrRect,
  short   nX1,
  short   nY1,
  short   nX2,
  short   nY2
)
```

Argument

(o) GRect *pgrRect1	: rectangle area
(i) short nX1	: X coordinate for upper left corner
(i) short nY1	: Y coordinate for upper left corner
(i) short nX2	: X coordinate for lower right corner
(i) short nY2	: Y coordinate for lower right corner

Return value None

Details The coordinates for the rectangle’s upper left corner are specified as (nX1, nY1) and those for the lower right corner as (nX2, nY2).

File name GRSetRect.c

GRShiftRect

GRShiftRect.c

Outline

Rectangle shift processing

Syntax

```
short GRShiftRect(
  GRect   *pgrRect,
  short   nDx,
  short   nDy,
  GRect   *pgrRect1,
  GRect   *pgrRect2
)
```

Argument

(i) GRect *pgrRect	: source rectangular area
(i) short nDx	: increase in X direction
(i) short nDy	: increase in Y direction
(o) GRect *pgrRect1	: empty rectangular area
(o) GRect *pgrRect2	: empty rectangular area

Return value

0 : no empty rectangular area
1 : one empty rectangular area
2 : two empty rectangular areas

Details

Moves the rectangle defined in pgrRect nDx and nDy in the X direction and Y direction, respectively.
Furthermore, the empty rectangular area created by the movement is set in pgrRect1 and pgrRect2.

File name

GRShiftRect.c

GRUnionRect

GRUnionRect.c

Outline Join rectangle areas

Syntax
```
void GRUnionRect(
  GRect    *pgrSrc1,
  GRect    *pgrSrc2,
  GRect    *pgrDest
)
```

Argument
(i) GRect *pgrSrc1 : rectangle area
(i) GRect *pgrSrc2 : rectangle area
(o) GRect *pgrDest : combined rectangular area

Return value None

Details The minimum sized rectangular area surrounding the specified rectangular areas pgrSrc1 and pgrSrc2 is set in pgrDest.

File name GRUnionRect.c

GRVirtualToPhysicalRect

GRVirtualToPhysicalRect.c

Outline Convert rectangular area virtual coordinates to physical coordinates

Syntax
```
void GRVirtualToPhysicalRect(
  HGDRAW   hDraw,
  GRect    *pgrRect
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i) GRect *pgrRect : rectangle area

Return value None

Details Converts the rectangular area defined in pgrRect from virtual coordinates to physical (local) coordinates.

File name GRVirtualToPhysicalRect.c

GLDuplicateLine

GLDuplicateLine.c

Outline	Copy line structure
Syntax	void GLDuplicateLine(GLine **pglSrc,* GLine **pglDest*)
Argument	(i) GLine *pglSrc : copy source (o) GLine *pglDest : copy destination
Return value	None
Details	The line structure defined in pglSrc is copied to the line structure in pglDest.
File name	GLDuplicateLine.c

GLGetLine

GLGetLine.c

Outline	Get line
Syntax	void GLGetLine(GLine **pglLine,* short **nX1,* short **nY1,* short **nX2,* short **nY2*)
Argument	(i) GLine *pglLine : source line structure being acquired (o) short *nX1 : X coordinate for starting point of line structure (o) short *nY1 : Y coordinate for starting point of line structure (o) short *nX2 : X coordinate for ending point of line structure (o) short *nY2 : Y coordinate for ending point of line structure
Return value	None
Details	Gets the content of the line structure pglLine in (nX1,nY1) and (nX2,nY2)
File name	GLGetLine.c

GLMapLine

GLMapLine.c

Outline	Enlarge/reduce line coordinates
Syntax	void GLMapLine(GLine **pglLine,* GRect **pgrFrom,* GRect **pgrTo*)
Argument	(i/o) GLine *pglLine : line structure (i) GRect *pgrForm : rectangle area (i) GRect *pgrTo : rectangle area
Return value	None
Details	Converts coordinates when the line structure in the pgrFrom rectangular area will be projected onto to the pgrTo rectangular area.
File name	GLMapLine.c

GLOffsetLine

GLOffsetLine.c

Outline Offset processing for line structure

Syntax
```
void GLOffsetLine(
  GLine   *pglLine,
  short   nDx,
  short   nDy
)
```

Argument
(i/o) GLine *pglLine : source line
(i) short nDx : increase in X direction
(i) short nDy : increase in Y direction

Return value None

Details Moves the line defined in pglLine nDx and nDy in the X direction and Y direction, respectively.

File name GLOffsetLine.c

GLPhysicalToVirtualLine

GLPhysicalToVirtualLine.c

Outline Convert line structure physical coordinates to virtual coordinates

Syntax
```
void GLPhysicalToVirtualLine(
  HGDRAW   hDraw,
  GLine    *pglLine
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i/o) GLine *pglLine : line structure

Return value None

Details Converts the coordinates defined in pglLine from physical (local) coordinates to virtual coordinates.

File name GLPhysicalToVirtualLine.c

GLSetLine

GLSetLine.c

Outline Define line structure

Syntax

```
void GLSetLine(
  GLine   *pglLine,
  short   nX1,
  short   nY1,
  short   nX2,
  short   nY2
)
```

Argument

(o) GLine *pglLine	: line structure
(i) short nX1	: X coordinate for starting point
(i) short nY1	: Y coordinate for starting point
(i) short nX2	: X coordinate for ending point
(i) short nY2	: Y coordinate for ending point

Return value None

Details Defines a line structure. The coordinates for the starting point of a line are specified as (nX1, nY1) and those for the ending point as (nX2, nY2).

File name GLSetLine.c

GLVirtualToPhysicalLine

GLVirtualToPhysicalLine.c

Outline	Convert line structure virtual coordinates to physical coordinates
Syntax	void GLVirtualToPhysicalLine(HGDRAW *hDraw,* GLine **pglLine*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i/o) GLine *pglLine : line structure
Return value	None
Details	Converts the coordinates defined in pglLine from virtual coordinates to physical (local) coordinates.
File name	GLVirtualToPhysicalLine.c

GPTDuplicatePoint

GPTDuplicatePoint.c

Outline	Copy point structure
Syntax	void GPTDuplicatePoint(GPoint **pgptSrc,* GPoint **pgptDest*)
Argument	(i) GPoint *pgptSrc : copy source (o) GPoint *pgptDest : copy destination
Return value	None
Details	The point structure defined in pgptSrc is copied to pgptDest.
File name	GPTDuplicatePoint.c

GPTGetPoint

GPTGetPoint.c

Outline Get point structure content

Syntax
```
void GPTGetPoint(
  GPoint   *pgptPoint,
  short    *nX,
  short    *nY
)
```

Argument

(i) GPoint *pgptPoint	:	point structure for acquisition source
(o) short *nX	:	X coordinate for point structure
(o) short *nY	:	Y coordinate for point structure

Return value None

Details Gets the content of point structure pgptPoint in (nX, nY).

File name GPTGetPoint.c

GPTMapPoint

GPTMapPoint.c

Outline Relative conversion of point coordinates

Syntax
```
void GPTMapPoint(
  GPoint   *pgptPoint,
  GRect    *pgrFrom,
  GRect    *pgrTo
)
```

Argument

(i/o) GPoint *pgptPoint	:	point structure
(i) GRect *pgrFrom	:	rectangle area
(i) GRect *pgrTo	:	rectangle area

Return value None

Details Converts coordinates when the point structure in the pgrFrom rectangular area will be projected onto to the pgrTo rectangular area.

File name GPTMapPoint.c

GPTMapToScreenPoint

GPTMapToScreenPoint.c

Outline	Convert point structure to screen coordinates
Syntax	void GPTMapToScreenPoint(HGDRAW *hDraw,* GPoint **pgptPoint*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i/o) GPoint *pgptPoint : point coordinates
Return value	None
Details	Converts the point defined in pgptPoint to the actual screen coordinate system.
File name	GPTMapToScreenPoint.c

GPTMapToVirtualPoint

GPTMapToVirtualPoint.c

Outline	Convert point defined in pgptPoint to virtual screen coordinate system
Syntax	void GPTMapToVirtualPoint(HGDRAW *hDraw,* GPoint ** pgptPoint*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GPoint *pgptPoint : point structure
Return value	None
Details	Converts the point defined in pgptPoint to the virtual screen coordinate system.
File name	GPTMapToVirtualPoint.c

GPTOffsetPoint

GPTOffsetPoint.c

Outline Point structure offset processing

Syntax
```
void GPTOffsetPoint(
  GPoint   *pgptPoint,
  short    nDx,
  short    nDy
)
```

Argument
(i/o) GPoint *pgptPoint : point structure for movement source
(i) short nDx : increase in X direction
(i) short nDy : increase in Y direction

Return value None

Details Moves the point structure pgptPoint nDx and nDy in the X direction and Y direction, respectively.

File name GPTOffsetPoint.c

GPTPhysicalToVirtualPoint

GPTPhysicalToVirtualPoint.c

Outline Convert point structure physical coordinates to virtual coordinates

Syntax
```
void GPTPhysicalToVirtualPoint(
  HGDRAW   hDraw,
  GPoint   *pgptPoint
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i/o) GPoint *pgptPoint : point structure

Return value None

Details Converts the point defined in pgptPoint from physical (local) coordinates to virtual coordinates.

File name GPTPhysicalToVirtualPoint.c

GPTSetPoint

GPTSetPoint.c

Outline Define point structure

Syntax
```
void GPTSetPoint(
  GPoint   *pgptPoint,
  short    nX,
  short    nY
)
```

Argument
(o) GPoint *pgptPoint : point structure
(i) short nX : coordinate in X direction
(i) short nY : coordinate in Y direction

Return value None

Details Sets the (nX, nY) coordinates in the point structure pgptPoint.

File name GPTSetPoint.c

GPTVirtualToPhysicalPoint

GPTVirtualToPhysicalPoint.c

Outline Convert point structure virtual coordinates to physical coordinates

Syntax
```
void GPTVirtualToPhysicalPoint(
  HGDRAW   hDraw,
  GPoint   *pgptPoint
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i/o) GPoint *pgptPoint : point structure

Return value None

Details Converts the point specified by pgptPoint from virtual coordinates to physical coordinates.

File name GPTVirtualToPhysicalPoint.c

GPGetPoly

GPGetPoly.c

Outline Get polygon settings

Syntax

```
void GPGetPoly(
  GPoly     *pgpPolygon,
  short     *nNumPoints,
  GPoint    **pgptPoints,
  GRect     *pgrBounds
)
```

Argument

(i) GPoly *pgpPolygon	: polygon structure for acquisition source
(o) short *nNumPoints	: number of vertices
(o) GPoint **pgptPoints	: point structure
(o) GRect *pgrBounds	: rectangle area

Return value None

Details Gets the polygon structure pgpPolygon content in nNumPoints, pgptPoints and pgrBounds.

File name GPGetPoly.c

GPMapPoly

GPMapPoly.c

Outline	Enlarge/reduce polygon
Syntax	void GPMapPoly(GPoly * *pgpPolygon,* GRect * *pgrFrom,* GRect * *pgrTo*)
Argument	(i/o) GPoly *pgpPolygon : polygon structure (i) GRect *pgrFrom : rectangular area (i) GRect *pgrTo : rectangular area
Return value	None
Details	Converts coordinates when the polygon structure in the pgrFrom rectangular area will be projected onto to the pgrTo rectangular area.
File name	GPMapPoly.c

GPOffsetPoly

GPOffsetPoly.c

Outline Polygon offset processing

Syntax
```
void GPOffsetPoly(
  GPoly  *pgpPolygon,
  short  nDx,
  short  nDy
)
```

Argument

(i/o) GPoly *pgpPolygon	:	polygon for movement source
(i) short nDx	:	increase in X direction
(i) short nDy	:	increase in Y direction

Return value None

Details Moves the polygon defined in pgpPolygon nDx and nDy in the X direction and Y direction, respectively.

File name GPOffsetPoly.c

GPPhysicalToVirtualPoly

GPPhysicalToVirtualPoly.c

Outline Convert polygon physical coordinates to virtual coordinates

Syntax
```
void GPPhysicalToVirtualPoly(
  HGDRAW  hDraw,
  GPoly   *pgpPolygon
)
```

Argument

(i/o) HGDRAW hDraw	:	drawing handle
(i/o) GPoly *pgpPolygon	:	polygon structure

Return value None

Details Converts the polygon structure defined in pgpPolygon from physical (local) coordinates to virtual coordinates.

File name GPPhysicalToVirtualPoly.c

GPSetPoly

GPSetPoly.c

Outline Define polygon

Syntax
```
void GPSetPoly(
  GPoly    *pgpPolygon,
  short    nNumPoints,
  GPoint   *pgptPoints,
  GRect    *pgrBounds
)
```

Argument
(o) GPoly *pgpPolygon : polygon structure
(i) short nNumPoints : number of vertices
(i) GPoint *pgptPoints : point structure
(i) GRect *pgrBounds : rectangle area

Return value None

Details Sets nNumPoints, pgptPoints and pgrBounds in the polygon structure pgpPolygon.

File name GPSetPoly.c

GPVirtualToPhysicalPoly

GPVirtualToPhysicalPoly.c

Outline Convert polygon virtual coordinates to physical coordinates

Syntax
```
void GPVirtualToPhysicalPoly(
  HGDRAW   hDraw,
  GPoly    *pgpPolygon
)
```

Argument
(i/o) HGDRAW hDraw : drawing handle
(i/o) GPoly *pgpPolygon : polygon structure

Return value None

Details Converts the polygon structure defined in pgpPolygon from virtual coordinates to local (physical) coordinates.

File name GPVirtualToPhysicalPoly.c

GMSetMemorySpace

GMemory.c

Outline	Register memory area information
Syntax	short GMSetMemorySpace(unsigned char *ucMemorySpaceCount,* GMemorySpaceInformation *paMemorySpace*)
Argument	(i) unsigned char ucMemorySpaceCount : number of pieces of data for memory area to be registered (i) GMemorySpaceInformation *paMemorySpace : pointer to memory area information array
Return value	0 : registration failed 1 or more : number of pieces of memory area data successfully registered
Details	GMemorySpaceInformation is set in GUI library 2 for the number specified by ucMemorySpaceCount. GMemorySpaceInformation is passed to paMemorySpace as an array for the number specified by ucMemorySpaceCount. GPISetMemorySpace is called for the internal processing.
File name	GMemory.c

GMDeleteMemorySpace	
	GMemory.c

Outline	Delete registered memory area
Syntax	short GMDeleteMemorySpace(unsigned char *ucType,* unsigned char *ucPlane*)
Argument	(i) unsigned char ucType : memory area type to be deleted (i) unsigned char ucPlane : memory plane No. for deletion
Return value	FALSE : Deletion failed TRUE : Deletion succeeded
Details	The registered memory area specified by ucType and ucPlane is deleted. GPIDeleteMemorySpace is called for the internal processing.
File name	GMemory.c

GMMemoryAlloc

GMemory.c

Outline Allocate memory area

Syntax

```
void* GMMemoryAlloc(
  unsigned char   ucType,
  unsigned char   ucPlane,
  unsigned long   ulMemorySpaceSize
)
```

Argument

(i) unsigned char ucType		: type of memory area for memory being allocated
	GMEM_SPACE_GDRAW(1)	: memory area used by Gdraw
	GMEM_SPACE_GPI(4)	: memory area used by GPI
	GMEM_SPACE_GWIN(5)	: memory area used by GWin
	GMEM_SPACE_GWIN_PANEL(6)	: memory area used by GWin base
	GMEM_SPACE_GWIN_WINDOW(7)	: memory area used by GWin window
(i) unsigned char ucPlane		: memory area plane No. for memory being allocated
(i) unsigned long ulMemorySpaceSize		: size of memory being allocated

Return value

NULL	: memory allocation failed
Other than NULL	: start of address for memory allocated

Details Allocates memory of the specified size from the specified memory area.
When allocation fails, a NULL is returned as a return value.
GPIMemoryAlloc is called for the internal processing.

File name GMemory.c

GMMemoryFree

GMemory.c

Outline Release memory area

Syntax

```
short GMMemoryFree (
  unsigned char    ucType,
  unsigned char    ucPlane,
  void             *pvMemory
)
```

Argument

(i) unsigned char ucType		: type of memory area for memory to be released
	GMEM_SPACE_GDRAW(1)	: memory area used by Gdraw
	GMEM_SPACE_GPI(4)	: memory area used by GPI
	GMEM_SPACE_GWIN(5)	: memory area used by GWin
	GMEM_SPACE_GWIN_PANEL(6)	: memory area used by GWin base
	GMEM_SPACE_GWIN_WINDOW(7)	: memory area used by GWin window
	GMEM_SPACE_USER(8)	: memory area used by user
(i) unsigned char ucPlane		: memory area plane No. for memory being released
(i) void *pvMemory		: address of memory being released

Return value

FALSE : memory release failed
TRUE : memory release succeeded

Details Releases memory allocated from the specified memory area.
GPIMemoryFree is called for the internal processing.

File name GMemory.c

GDCreateVram

GDCreateVram.c

Outline	Create virtual VRAM
Syntax	HGVRAM GDCreateVram(GVram *pVram)
Argument	(i) GVram *pVram : VRAM structure
Return value	VRAM handle
Details	Creates virtual VRAM based on the information set in the VRAM structure.
File name	GDCreateVram.c

GDDeleteVram

GDDeleteVram.c

Outline	Release virtual VRAM
Syntax	void GDDeleteVram(HGVRAM hVram)
Argument	(i/o) HGVRAM hVram : VRAM handle
Return value	VRAM handle
Details	Releases the virtual VRAM corresponding to the specified VRAM handle.
File name	GDDeleteVram.c

GDFlushScreenForce

GDFlushScreenForce.c

Outline	Forcible refresh screen

Syntax

```
void GDFlushScreenForce(
  HGDRAW   hDraw,
  short    nX,
  short    nY,
  short    nWidth,
  short    nHeight
)
```

Argument		
	(i/o) HGDRAW hDraw	: drawing handle
	(i) short nX	: X coordinate for drawing starting point
	(i) short nY	: Y coordinate for drawing starting point
	(i) short nWidth	: drawing area width
	(i) short nHeight	: drawing area height

Return value	None
Details	A screen for the size of nWidth∗Height is forcibly refreshed in the current coordinate system with the specified point (nX, nY) as the starting point.
File name	GDFlushScreenForce.c

GDGetGradationInfo

GDGetGradationInfo.c

Outline	Get gradation information
Syntax	void GDGetGradationInfo(HGDRAW *hDraw,* GGradationInfo **pggInfo*)
Argument	(i/o) HGDRAW hDraw : drawing handle (o) GGradationInfo *pggInfo : gradation information structure
Return value	None
Details	Gets gradation information.
File name	GDGetGradationInfo.c

GDGetLocalOrigin

GDGetLocalOrigin.c

Outline	Get local coordinate origin
Syntax	void GDGetLocalOrigin(HGDRAW *hDraw,* GPoint **pgptPoint*)
Argument	(i/o) HGDRAW hDraw : drawing handle (o) GPoint *pgptPoint : point structure
Return value	None
Details	Gets local (physical) coordinate origin.
File name	GDGetLocalOrigin.c

GDGetVram

GDGetVram.c

Outline	Get system VRAM
Syntax	HGVRAM GDGetVram(HGDRAW *hDraw*)
Argument	(i/o) HGDRAW hDraw : drawing handle
Return value	HGVRAM : system VRAM handle
Details	Gets system VRAM.
File name	GDGetVram.c

GDRemoveDrawableRegion

GDRemoveDrawableRegion.c

Outline	Delete specified rectangular area from drawable area
Syntax	short GDRemoveDrawableRegion(HGDRAW *hDraw,* GRect * *pRect*)
Argument	(i/o) HGDRAW hDraw : drawing handle (o) GRect *pRect : rectangle area to be deleted
Return value	FALSE : failed TRUE : succeeded
Details	Deletes the rectangular area specified by pRect from the drawable area held by the platform.
File name	GDRemoveDrawableRegion.c

GDSetDrawableRegion

GDSetDrawableRegion.c

Outline	Set drawable area
Syntax	short GDSetDrawableRegion(HGDRAW *hDraw,* GRect * *pRect*)
Argument	(i/o)HGDRAW hDraw : drawing handle (i)GRect *pRect : rectangular area being set
Return value	FALSE : failed TRUE : succeeded
Details	Sets the drawable area held by the platform.
File name	GDSetDrawableRegion.c

GDSetGradationInfo

GDSetGradationInfo.c

Outline	Set gradation information
Syntax	void GDSetGradationInfo(HGDRAW *hDraw,* GGradationInfo **pggInfo*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GGradationInfo *pggInfo : gradation information structure
Return value	None
Details	Sets gradation information.
File name	GDSetGradationInfo.c

GDSetLocalOrigin

GDSetLocalOrigin.c

Outline	Set local coordinate origin
Syntax	void GDSetLocalOrigin(HGDRAW *hDraw,* GPoint **pgptPoint*)
Argument	(i/o) HGDRAW hDraw : drawing handle (i) GPoint *pgptPoint : point structure
Return value	None
Details	Values for pgptPoint.nX in the X direction and pgptPoint.nY in the Y direction are set for the origin for the local (physical) coordinates.
File name	GDSetLocalOrigin.c

3. GResource

This section describes the GResource functions.

3.1 List of GResource Functions

No.	Name	Type	File name	Outline
1	GRCLoadBorder	GBorder *	GRCLoadBorder.c	Get specified ID 3D border resource
2	GRCLoadFont	GSimpleFont *	GRCLoadFont.c	Get specified ID font resource
3	GRCLoadImage	GImage *	GRCLoadImage.c	Get specified ID image resource
4	GRCLoadString	char *	GRCLoadString.c	Get specified ID character string resource
5	GRCLoadLocaleFont	GSimpleFont *	GRCLoadLocaleFont.c	Get font resource for specified locale specified ID
6	GRCLoadLocaleString	GTCHAR *	GRCLoadLocaleString.c	Get character string resource for specified locale specified ID
7	GRCLoadPalette	GPaletteHeader *	GRCLoadPalette.c	Get specified ID palette resource
8	GRCSetLocale	short	GRCLocale.c	Current locale setting
9	GRCGetLocale	unsigned char	GRCLocale.c	Get current locale
10	GRCLoadFillPattern	char *	GRCLoadFillPattern.c	Get fill pattern for specified No.
11	GRCLoadHtmlFileName	GFileName*	GRCLoadHtmlFileName.c	Get HTML file name for specified ID
12	GRCLoadLanguageStr	GTCHAR *	GRCLoadLanguageStr.c	Get specified locale language-identifying character strings resource
13	GRCLoadNowLanguageStr	GTCHAR *	GRCLoadLanguageStr.c	Get current locale language-identifying character strings resource
14	GRCLoadLinePattern	char *	GRCLoadLinePattern.c	Get line drawing pattern for specified No.
15	GRCSetResourceHandle	void	GRCResourceHandle.c	Set current resource handle
16	GRCGetResourceHandle	HGRESOURCE	GRCResourceHandle.c	Get current resource handle
17	GRCUnicodeToMultiByte	unsigned long	GRCUnicode.c	UNICODE → multibyte conversion

No.	Name	Type	File name	Outline
18	GRCMultiByteToUnicode	unsigned long	GRCUnicode.c	Multibyte → UNICODE conversion

3.2 Description of GResource Functions

GRCLoadBorder

GRCLoadBorder.c

Outline	Get specified ID 3D border resource
Syntax	GBorder *GRCLoadBorder(unsigned short *usResourceID*)
Argument	(i) unsigned short usResourceID : resource ID
Return value	Specified resource
Details	Gets the pointer to the 3D border resource structure with the specified ID.
File name	GRCLoadBorder.c

GRCLoadFont

GRCLoadFont.c

Outline	Get specified ID font resource
Syntax	GSimpleFont *GRCLoadFont(unsigned short *usResourceID*)
Argument	(i) unsigned short usResourceID : resource ID
Return value	Specified resource
Details	Gets the pointer to the simple font resource structure with the specified ID.
File name	GRCLoadFont.c

GRCLoadImage

GRCLoadImage.c

Outline	Get specified ID image resource
Syntax	GImage *GRCLoadImage(unsigned short *usResourceID*)
Argument	(i) unsigned short usResourceID : resource ID
Return value	Specified resource
Details	Gets the pointer to the image resource structure with the specified ID.
File name	GRCLoadImage.c

GRCLoadString

GRCLoadString.c

Outline	Get specified ID character string resource
Syntax	char *GRCLoadString(unsigned short *usResourceID*)
Argument	(i) unsigned short usResourceID : resource ID
Return value	Specified resource
Details	Gets the pointer to the character string resource structure with the specified ID.
File name	GRCLoadString.c

GRCLoadLocaleFont

GRCLoadLocaleFont.c

Outline Get font resource for specified locale specified ID

Syntax
```
GSimpleFont *GRCLoadLocaleFont(
  unsigned char   ucLocale,
  unsigned short  usResourceID
)
```

Argument
(i) unsigned char ucLocale : locale No.
(i) unsigned short usResourceID : resource ID

Return value Pointer for the specified resource for the specified locale

Details Gets the pointer to the simple font resource structure with the specified locale specified ID.

File name GRCLoadLocaleFont.c

GRCLoadLocaleString

GRCLoadLocaleString.c

Outline Get character string resource for specified locale specified ID

Syntax
```
GTCHAR *GRCLoadLocaleString(
  unsigned char   ucLocale,
  unsigned short  usResourceID
)
```

Argument
(i) unsigned char ucLocale : locale No.
(i) unsigned short usResourceID : resource ID

Return value Pointer for the specified resource for the specified locale

Details Gets the pointer to character string resource with the specified ID for the specified locale.

File name GRCLoadLocaleString.c

GRCLoadPalette

GRCLoadPalette.c

Outline	Get specified ID palette resource
Syntax	GPaletteHeader *GRCLoadPalette(unsigned short *usResourceID*)
Argument	(i) unsigned short usResourceID : resource ID
Return value	Specified resource
Details	Gets the pointer to resource structure with the specified ID.
File name	GRCLoadPalette.c

GRCSetLocale

GRCLocale.c

Outline	Current locale setting
Syntax	short GRCSetLocale(unsigned char *ucNewLocale*)
Argument	(i) unsigned char ucNewLocale : locale No. being set
Return value	TRUE : setting succeeded FALSE : setting failed
Details	Sets the current locale.
File name	GRCLocale.c

GRCGetLocale

GRCLocale.c

Outline	Get current locale
Syntax	unsigned char GRCGetLocale()
Argument	None
Return value	Current locale No.
Details	Gets the current locale.
File name	GRCLocale.c

GRCLoadFillPattern

GRCLoadFillPattern.c

Outline	Get fill pattern for specified No.
Syntax	char *GRCLoadFillPattern(unsigned short *usIndex*)
Argument	(i) unsigned short usIndex : fill No.
Return value	Fill pattern with specified No.
Details	Gets the fill pattern for the specified No.
File name	GRCLoadFillPattern.c

GRCLoadHtmlFileName

GRCLoadHtmlFileName.c

Outline	Get HTML file name for specified ID
Syntax	GFileName* GRCLoadHtmlFileName(unsigned short *usResourceID*)
Argument	(i) unsigned short usResourceID : resource ID
Return value	Specified resource HTML file name
Details	Gets the HTML file name with the specified ID.
File name	GRCLoadHtmlFileName.c

GRCLoadLanguageStr

GRCLoadLanguageStr.c

Outline	Get specified locale language-identifying character strings resource
Syntax	GTCHAR *GRCLoadLanguageStr(unsigned short *usLocaleID*)
Argument	(i) unsigned short usLocaleID : locale ID
Return value	Pointer to specified locale character string
Details	Gets the pointer for the specified locale language-identifying chracter strings resource.
File name	GRCLoadLanguageStr.c

GRCLoadNowLanguageStr

GRCLoadLanguageStr.c

Outline	Get current locale language-identifying character strings resource
Syntax	GTCHAR *GRCLoadNowLanguageStr()
Argument	None
Return value	Pointer to current locale character string
Details	Gets the pointer for the current locale language-identifying chracter strings resource.
File name	GRCLoadLanguageStr.c

GRCLoadLinePattern

GRCLoadLinePattern.c

Outline	Get line drawing pattern for specified No.
Syntax	char *GRCLoadLinePattern(unsigned short *usIndex*)
Argument	(i) unsigned short usIndex : line drawing No.
Return value	Line drawing pattern for specified No.
Details	Gets the pointer to the line drawing pattern for the specified No.
File name	GRCLoadLinePattern.c

GRCSetResourceHandle

GRCResourceHandle.c

Outline	Set current resource handle
Syntax	void GRCSetResourceHandle(HGRESOURCE *hResource*)
Argument	(i) HGRESOURCE hResource : resource handle to be set
Return value	None
Details	Sets the current resource handle.
File name	GRCResourceHandle.c

GRCGetResourceHandle

GRCResourceHandle.c

Outline	Get current resource handle
Syntax	HGRESOURCE GRCGetResourceHandle()
Argument	None
Return value	Current resource handle
Details	Gets the pointer to the current resource handle.
File name	GRCResourceHandle.c

GRCUnicodeToMultiByte

GRCUnicode.c

Outline UNICODE → multibyte conversion

Syntax

```
unsigned long GRCUnicodeToMultiByte(
  GTCHAR         *pszLangStr,
  char           *mbstr,
  wchar_t        *wcstr,
  unsigned long  ulCount
)
```

Argument

(i) GTCHAR *pszLangStr	:	language-identifying character string
(o) char *mbstr	:	buffer for saving multibyte character string
(i) wchar_t *wcstr	:	buffer for saving unicode character string
(i) unsigned long ulCount	:	maximum size that can be stored in output destination buffer

Return value Write size in output destination multibyte character string.

Details Converts a unicode character string to a multibyte character string.
When NULL is specified in mbstr, the size necessary for the output destination buffer is returned to the return values.

File name GRCUnicode.c

GRCMultiByteToUnicode

GRCUnicode.c

Outline multibyte → UNICODE conversion

Syntax

```
unsigned long GRCMultiByteToUnicode(
  GTCHAR        *pszLangStr,
  wchar_t       *wcstr,
  char          *mbstr,
  unsigned long  ulCount
)
```

Argument

(i) GTCHAR *pszLangStr	:	language-identifying character string
(o) wchar_t *wcstr	:	buffer storing the unicode character string
(i) char *mbstr	:	buffer storing the multibyte character string
(i) unsigned long ulCount	:	maximum size that can be stored in output destination buffer

Return value Write size in output destination unicode character string.

Details Converts a multibyte character string to a unicode character string. When NULL is specified in wcstr, the size necessary for the output destination buffer is returned to the return values.

File name GRCUnicode.c

4. GEvent

This section describes the GEvent function group.

4.1 List of GEvent Functions

No.	Name	Type	File name	Outline
1	GEInitEvent	short	GEvent.c	Initialize event system
2	GEQuitEvent	void	GEvent.c	Process event system termination
3	GECreateEventMessage	GEventMessage	GEvent.c	Define event structure
4	GESetEvent	short	GEvent.c	Register event
5	GEGetEvent	short	GEvent.c	Acquire event
6	GEClearTimerEvent	void	GEvent.c	Clear timer event registration from event queue
7	GESetTimer	short	GTimer.c	Register timer event
8	GEKillTimer	void	GTimer.c	Clear timer event registration
9	GECheckTimer	void	GTimer.c	Confirm that timer event has been issued
10	GESetSystemTime	short	GTimer.c	Set system time
11	GEGetSystemTime	short	GTimer.c	Get system time
12	GECompareSystemTime	long	GTimer.c	System time comparison
13	GEClearTargetEvent	void	GEvent.c	Clear registration for specified objects event to be sent

4.2 Description of GEvent Functions

GEInitEvent	
	GEvent.c
Outline	Initialize event system
Syntax	short GEInitEvent()
Argument	None
Return value	FALSE : initialization failed TRUE : initialization succeeded
Details	Initializes the event system.
File name	GEvent.c

GEQuitEvent	
	GEvent.c
Outline	Process event system termination
Syntax	void GEQuitEvent()
Argument	None
Return value	None
Details	Quits event system.
File name	GEvent.c

GECreateEventMessage

GEvent.c

Outline Define event structure

Syntax

```
GEventMessage GECreateEventMessage(
  unsigned short  usMessage,
  void            *pvTarget,
  long            lLParam,
  long            lUParam
)
```

Argument

(i)	unsigned short usMessage	: event message
(i)	void *pvTarget	: event target object
(i)	long lLParam	: lower-side of additional information
(i)	long lUParam	: upper-side of additional information

Return value Event structure

Details Defines event structure.

File name GEvent.c

GESetEvent

GEvent.c

Outline

Register event

Syntax

```
short GESetEvent(
  GEventMessage  gemMsg,
  short          fMarge
)
```

Argument

(i)GEventMessage gemMsg : Event information
(i)short fMarge : merge flag

Return value

FALSE : registration failed
TRUE : registration succeeded

Details

Registers an event.
When there is a same event for the merge flag, this makes the setting either for merging or not for merging.
fMarge
FALSE : no merge
TRUE : merge

File name

GEvent.c

GEGetEvent

GEvent.c

Outline	Acquire event
Syntax	short GEGetEvent(GEventMessage *gemMsg, long lTimeOut)
Argument	(i)GEventMessage *gemMsg : event information (i)long lTimeOut : timeout
Return value	FALSE : no event TRUE : event present
Details	Gets an event. When the event is not generated until the timeout time passes, the status is changed to the waiting.
File name	GEvent.c

GEClearTimerEvent

GEvent.c

Outline	Clear timer event registration from even queue
Syntax	void GEClearTimerEvent(void *pvTarget, unsigned short usTimerID)
Argument	(i) void * pvTarget : timer event target object (i) unsigned short usTimerID : timer ID
Return value	None
Details	Clears the registration of the timer event for the ID specified from the event queue. Specify the object where the timer event is set in pvTarget.
File name	GEvent.c

GESetTimer

GTimer.c

Outline Register timer event

Syntax
```
short GESetTimer(
  void           *pvTarget,
  unsigned short  usTimerID,
  unsigned long   ulTimeOut
)
```

Argument
(i)void *pvTarget : timer event target object
(i)unsigned short usTimerID : timer ID
(i)unsigned long ulTimeOut : timer interval

Return value
FALSE : timer event registration failed
TRUE : timer event registration succeeded

Details Registers a timer event.

File name GTimer.c

GEKillTimer

GTimer.c

Outline Clear timer event registration

Syntax
```
void GEKillTimer(
  void           *pvTarget,
  unsigned short  usTimerID
)
```

Argument
(i)void *pvTarget : timer event target object
(i)unsigned short usTimerID : timer ID to be cleared

Return value None

Details Clears the timer event registration.

File name GTimer.c

GECheckTimer

GTimer.c

Outline	Confirm that timer event has been issued
Syntax	void GECheckTimer()
Argument	None
Return value	None
Details	Checks the issuance of a timer event. If the timer event being issued is present, the event is registered.
File name	GTimer.c

GESetSystemTime

GTimer.c

Outline	Set system time
Syntax	short GESetSystemTime(GSystemTime *pSystemTime*)
Argument	GSystemTime *pSystemTime : system time
Return value	FALSE : setting failed TRUE : setting succeeded
Details	Sets the current system time.
File name	GTimer.c

GEGetSystemTime

GTimer.c

Outline	Get system time
Syntax	short GEGetSystemTime(GSystemTime *pSystemTime)
Argument	GSystemTime *pSystemTime : for system time storage
Return value	FALSE : acquisition failed TRUE : acquisition succeeded
Details	Gets the current system time.
File name	GTimer.c

GECompareSystemTime

GTimer.c

Outline	System time comparison
Syntax	long GECompareSystemTime(GSystemTime *pSystemTime1, GSystemTime *pSystemTime2)
Argument	GSystemTime *pSystemTime1 : system time GSystemTime *pSystemTime2 : system time
Return value	Difference in system times
Details	Compares the two times.
File name	GTimer.c

GEClearTargetEvent

GEvent.c

Outline	Clear specified target event registration
Syntax	void GEClearTargetEvent(void * pvTarget)
Argument	(i) void * pvTarget : target object
Return value	None
Details	Clears the target event registration set specified in pvTarget.
File name	GEvent.c

5. GCS

This section describes the GCS function group.

5.1 List of GCS Functions

5.1.1 NC Designer2 Compatibility

No.	Name	Type	File name	Outline
1	GCSGetType	unsigned short	GWin.h	Get object type
2	GCSGetID	unsigned short	GWin.h	Get object ID
3	GCSGetXPosition	short	GWin.h	Get horizontal position
4	GCSGetYPosition	short	GWin.h	Get vertical position
5	GCSSetPosition	void	GWin.h	Set position
6	GCSGetPosition	void	GWin.h	Get position
7	GCSGetWidth	short	GWin.h	Get width
8	GCSGetHeight	short	GWin.h	Get height
9	GCSGetSize	void	GWin.h	Get size
10	GCSSetBounds	void	GWin.h	Set relative position within parent object
11	GCSGetBounds	void	GWin.h	Get relative position within parent object
12	GCSCalcClientRect	void	GWin.h	Get coordinate position of client area
13	GCSGetClientRect	void	GWin.h	Get object client area
14	GCSClientToScreen	void	GWin.h	Convert client coordinates to screen coordinates
15	GCSGetStyle	unsigned long	GWin.h	Batch style acquisition
16	GCSSetVisibleStatus	void	GWin.h	Set display/non-display status
17	GCSGetVisibleStatus	short	GWin.h	Get display/non-display status
18	GCSSetEnableStatus	void	GWin.h	Set controllable or uncontrollable status
19	GCSGetEnableStatus	short	GWin.h	Get controllable or uncontrollable status
20	GCSSetConsumableStatus	void	GWin.h	Set input event consumable or input event not consumable status
21	GCSGetConsumableStatus	short	GWin.h	Get input event consumable or input event not consumable status
22	GCSSetCaptionStatus	void	GWin.h	Set caption display status
23	GCSGetCaptionStatus	short	GWin.h	Get caption display status
24	GCSSetFocusStatus	void	GWin.h	Set focus status
25	GCSGetFocusStatus	short	GWin.h	Get focus status

No.	Name	Type	File name	Outline
26	GCSSetPanelStatus	void	GWin.h	Set management panel object status
27	GCSGetPanelStatus	short	GWin.h	Get management panel object status
28	GCSSetStyleStatus	void	GWin.h	Set style
29	GCSGetStyleStatus	short	GWin.h	Get style
30	GCSSetParent	void	GWin.h	Set parent object
31	GCSGetParent	GBaseObject*	GWin.h	Get parent object
32	GCSImportProperty	short	GWin.h	Import property settings
33	GCSExportProperty	void	GWin.h	Export property settings
34	GCSCreate	short	GWin.h	Create process
35	GCSCreateVram	short	GWin.h	Create process (create VRAM)
36	GCSDelete	void	GWin.h	Delete process
37	GCSPaint	void	GWin.h	Drawing process
38	GCSOnDraw	void	GWin.h	Client area drawing process
39	GCSOnNcDraw	void	GWin.h	Non-client area drawing process
40	GCSAddRefreshRect	void	GWin.h	Client area redraw area registration
41	GCSLButtonPress	long	GWin.h	Process for pressing left mouse button
42	GCSOnLButtonPress	long	GWin.h	Client area process for pressing left mouse button
43	GCSLButtonRelease	void	GWin.h	Process for releasing left mouse button
44	GCSOnLButtonRelease	void	GWin.h	Client area process for releasing left mouse button
45	GCSMouseMove	void	GWin.h	Pointing device movement process
46	GCSOnMouseMove	void	GWin.h	Pointing device client area movement process
47	GCSOnNcMouseMove	void	GWin.h	Pointing device non-client area movement process
48	GCSKeyPress	long	GWin.h	Process for pressing key
49	GCSKeyRelease	void	GWin.h	Process for releasing key
50	GCSSetFocus	void	GWin.h	Focus setting process
51	GCSKillFocus	void	GWin.h	Focus removing process
52	GCSTimer	void	GWin.h	Timer process
53	GCSUser	long	GWin.h	User process
54	GCSSetFocusInformation	void	GWin.h	Set focus movement information
55	GCSGetFocusInformation	GFocusInformation *	GWin.h	Get focus movement information
56	GCSChar	void	GWin.h	Character input process
57	GCSClose	void	GWin.h	Issue object close process request
58	GCSDeleteChild	void	GWin.h	Child object deletion process
59	GCSSetMemorySpace	void	GWin.h	Set memory information
60	GCSGetMemorySpace	void	GWin.h	Get memory information

No.	Name	Type	File name	Outline
61	GCSSetDrawForceStatus	void	GWin.h	Set forced drawing status
62	GCSGetDrawForceStatus	short	GWin.h	Get forced drawing status
63	GCSSetRedrawStatus	void	GWin.h	Set redraw status
64	GCSGetRedrawStatus	short	GWin.h	Get redraw status
65	GCSClearDrawStatus	void	GWin.h	Clear drawing status
66	GCSAddDirtyRect	void	GWin.h	Client area redraw area registration (internal processing function)
67	GCSSetPrevTimerObject	void	GWin.h	Set previous object on timer bidirectional list
68	GCSGetPrevTimerObject	GBaseObject*	GWin.h	Get previous object on timer bidirectional list
69	GCSSetNextTimerObject	void	GWin.h	Set next object on timer bidirectional list
70	GCSGetNextTimerObject	GBaseObject*	GWin.h	Get next object on timer bidirectional list
71	GCSSetBlinkStatus	void	GWin.h	Set blink status
72	GCSGetBlinkStatus	unsigned char	GWin.h	Get blink status
73	GCSGetParentPanel	void	GWin.h	Get management panel object
74	GCSSetPrevBlinkControl	void	GWin.h	Set previous object on blink bidirectional list
75	GCSGetPrevBlinkControl	GBaseObject*	GWin.h	Get previous object on blink bidirectional list
76	GCSSetNextBlinkControl	void	GWin.h	Set next object on blink bidirectional list
77	GCSGetNextBlinkControl	GBaseObject*	GWin.h	Get next object on blink bidirectional list
78	GCSSetPrevScrollControl	void	GWin.h	Set previous object on scroll bidirectional list
79	GCSGetPrevScrollControl	GBaseObject*	GWin.h	Get previous object on scroll bidirectional list
80	GCSSetNextScrollControl	void	GWin.h	Set next object on scroll bidirectional list
81	GCSGetNextScrollControl	GBaseObject*	GWin.h	Get next object on scroll bidirectional list
82	GCSLoadData	int	GWin.h	Load interpreter screen file
83	GCSSetHWDCEnvironment	void	GWin.h	Set drawing environment
84	GCSGetScreen	GBaseObject*	GWin.h	Get screen object
85	GCSSetGBaseObject	void	GWin.h	Set object management structure
86	GCSGetGBaseObject	GBaseObject*	GWin.h	Get object management structure
87	GCSFreeLoadData	int	GWin.h	Release memory acquired by LoadData
88	GCSSetModalStatus	void	GWin.h	Set modal status

No.	Name	Type	File name	Outline
89	GCSGetModalStatus	short	GWin.h	Get modal window status
90	GCSGetRetValue	short	GWin.h	Get modal window return value
91	GCSBeginModal	void	GWin.h	Start process for modal window
92	GCSEndModal	void	GWin.h	End process for modal window
93	GCSSetTitleBarStatus	void	GWin.h	Set title bar present/none status
94	GCSGetTitleBarStatus	short	GWin.h	Get title bar present/none status
95	GCSSetCloseButtonStatus	void	GWin.h	Set close button present/none status
96	GCSGetCloseButtonStatus	short	GWin.h	Get close button present/none status
97	GCSSetEdgeStatus	void	GWin.h	Set window frame present/none status
98	GCSGetEdgeStatus	short	GWin.h	Get window frame present/none status
99	GCSAddChild	short	GWin.h	Add child object
100	GCSGetChildList	HGBWL	GWin.h	Get child object list
101	GCSSetBackGroundDesign	void	GWin.h	Set background design
102	GCSGetBackGroundDesign	GDesign*	GWin.h	Get background design
103	GCSCreateChildren	short	GWin.h	Process for creating child object
104	GCSDeleteAllChildren	void	GWin.h	Delete all child objects
105	GCSGetChild	GBaseObject *	GWin.h	Search child object
106	GCSGetChildAll	GBaseObject *	GWin.h	Search child object search in frame
107	GCSSetActiveChild	short	GWin.h	Set active object
108	GCSGetActiveChild	GBaseObject*	GWin.h	Get active object (only direct child object)
109	GCSGetActiveControl	GBaseObject*	GWin.h	Get active control (including terminal child objects)
110	GCSSetBlinkOffInterval	void	GWin.h	Set blinking OFF interval
111	GCSGetBlinkOffInterval	long	GWin.h	Get blinking OFF interval
112	GCSSetBlinkOnInterval	void	GWin.h	Set blinking ON interval
113	GCSGetBlinkOnInterval	long	GWin.h	Get blinking ON interval
114	GCSAddBlinkControl	void	GWin.h	Add caption blink control
115	GCSRemoveBlinkControl	void	GWin.h	Delete caption blink control
116	GCSAddScrollControl	void	GWin.h	Add caption scroll control
117	GCSRemoveScrollControl	void	GWin.h	Delete caption scroll control
118	GCSDrawChildren	void	GWin.h	Child object drawing process
119	GCSAttachPanel	GBaseObject *	GWin.h	Attach panel object
120	GCSGetPanel	GBaseObject *	GWin.h	Get attached GCPanel object
121	GCSSetSystemFontID	void	GWin.h	Set system font resource ID

No.	Name	Type	File name	Outline
122	GCSGetSystemFontID	unsigned short	GWin.h	Get system font resource ID
123	GCSSetSystemBorderID	void	GWin.h	Set system 3D border resource ID
124	GCSGetSystemBorderID	unsigned short	GWin.h	Get system 3D border resource ID
125	GCSSetSystemForeColor	void	GWin.h	Set system foreground color
126	GCSGetSystemForeColor	GColor	GWin.h	Get system foreground color
127	GCSSetSystemBackColor	void	GWin.h	Set system background color
128	GCSGetSystemBackColor	GColor	GWin.h	Get system background color
129	GCSSetSystemActiveColor	void	GWin.h	Set system active color
130	GCSGetSystemActiveColor	GColor	GWin.h	Get system active color
131	GCSSetSystemInactiveColor	void	GWin.h	Set system inactive color
132	GCSGetSystemInactiveColor	GColor	GWin.h	Get system inactive color
133	GCSGetFrame	GBaseObject *	GWin.h	Get frame
134	GCSAddWindow	short	GWin.h	Add window
135	GCSChangeFocus	void	GWin.h	Change focus
136	GCSGetFocus	GBaseObject *	GWin.h	Get focus
137	GCSDispatchMessage	long	GWin.h	Event dispatch process
138	GCSGetWDC	HWDC	GWin.h	Get drawing context
139	GCSGetHWDC	HWDC	GWin.h	Get drawing context
140	GCSSetPanelPlane	void	GWin.h	Set memory area panel plane management information
141	GCSGetPanelPlane	unsigned char	GWin.h	Get memory area panel plane management information
142	GCSSetWindowPlane	void	GWin.h	Set memory area window plane management information
143	GCSGetWindowPlane	unsigned char	GWin.h	Get memory area window plane management information
144	GCSSetExitStatus	void	GWin.h	Set exit information
145	GCSGetExitStatus	short	GWin.h	Get exit information
146	GCSSetSystemStatus	void	GWin.h	Set system information
147	GCSGetSystemStatus	short	GWin.h	Get system information
148	GCSDoModal	short	GWin.h	Window modal display
149	GCSSetPaletteStatus	void	GWin.h	Set system palette utilization status
150	GCSGetPaletteStatus	short	GWin.h	Get system palette utilization status
151	GCSSetSystemPaletteID	void	GWin.h	Set system palette resource ID
152	GCSGetSystemPaletteID	unsigned short	GWin.h	Get system palette resource ID
153	GCSSetPaletteColors	void	GWin.h	Set number of colors in system palette

No.	Name	Type	File name	Outline
154	GCSGetPaletteColors	unsigned short	GWin.h	Get number of colors in system palette
155	GCSAddTimerObject	void	GWin.h	Add object to system timer list
156	GCSRemoveTimerObject	void	GWin.h	Delete object from system timer list
157	GCSGetWindow	GBaseObject*	GWin.h	Get window
158	GCSGetActiveWindow	GBaseObject*	GWin.h	Get active window
159	GCSShowPanel	short	GWin.h	Display panel switching process
160	GCSCreateInstance	GBaseObject *	GWin.h	Create instance process
161	GCSPageLogOperation	short	GWin.h	Switch screen according to screen history
162	GCSDrawFrame	void	GWin.h	Related panel drawing process
163	GCSDrawPanel	void	GWin.h	Related GCPanel object drawing process
164	GCSOnDrawWindow	void	GWin.h	Window drawing process
165	GCSSetSystemPalette	void	GWin.h	Set system palette
166	GCSDrawBackGround	void	GWin.h	Background drawing process
167	GCSCreateGWindow	short	GWin.h	Create window
168	GCSSetScreenManager	void	GWin.h	Set screen manager
169	GCSSetModuleID	void	GWin.h	Set module ID
170	GCSClearPageLog	void	GWin.h	Clear screen history
171	GCSAddPageLog	short	GWin.h	Add screen history
172	GCSPrevPage	short	GWin.h	Issue request for displaying previous screen
173	GCSNextPage	short	GWin.h	Issue request for displaying next screen
174	GCSSetPageOffset	void	GWin.h	Set module page offset
175	GCSGetPageOffset	long	GWin.h	Get module page offset
176	GCSGetOperationLog	short	GWin.h	Get operation log
177	GCSClearOperationLog	short	GWin.h	Clear operation log
178	GCSAddOperationLog	short	GWin.h	Add operation log (operation event)
179	GCSAddChangePageLog	short	GWin.h	Add operation log (screen switching)
180	GCSGetPageNumber	long	GWin.h	Get screen No.
181	GCSLoadProjectFile	int	GWin.h	Load interpreter project file
182	GCSLoadGWindow	GBaseObject*	GWin.h	Window loading process
183	GCSModalGWindow	short	GWin.h	Modal window process
184	GCSSetBlinkType	void	GWin.h	Set blink type
185	GCSGetBlinkType	unsigned char	GWin.h	Get blink type
186	GCSSetBlinkEnableStatus	void	GWin.h	Set blink operation
187	GCSGetBlinkEnableStatus	unsigned char	GWin.h	Get blink operation
188	GCSStartCaptionBlink	void	GWin.h	Start blink operation
189	GCSStopCaptionBlink	void	GWin.h	Stop blink operation

No.	Name	Type	File name	Outline
190	GCSSetCaptionScrollEnable	void	GWin.h	Set scroll operation
191	GCSGetCaptionScrollEnable	unsigned char	GWin.h	Get scroll operation
192	GCSStartCaptionScroll	void	GWin.h	Start scroll operation
193	GCSStopCaptionScroll	void	GWin.h	Stop scroll operation
194	GCSPauseCaptionScroll	void	GWin.h	Pause scroll operation
195	GCSRestartCaptionScroll	void	GWin.h	Restart scroll operation
196	GCSSetCaptionScrollStatus	void	GWin.h	Set scroll operation status
197	GCSGetCaptionScrollStatus	unsigned char	GWin.h	Get scroll operation status
198	GCSIsCaptionOutOfBounds	short	GWin.h	Confirm overflowing of caption character string
199	GCSSetCaptionScrollDelayTime	void	GWin.h	Set scroll delay time
200	GCSGetCaptionScrollDelayTime	unsigned long	GWin.h	Get scroll delay time
201	GCSSetCaptionScrollRefreshTime	void	GWin.h	Set scroll refresh time
202	GCSGetCaptionScrollRefreshTime	unsigned long	GWin.h	Get scroll refresh time
203	GCSSetCaptionScrollMovementValue	void	GWin.h	Set scroll movement value
204	GCSGetCaptionScrollMovementValue	unsigned short	GWin.h	Get scroll movement value
205	GCSSetCaptionScrollStartPosition	void	GWin.h	Set scroll start position
206	GCSGetCaptionScrollStartPosition	unsigned char	GWin.h	Get scroll start position
207	GCSSetCaptionScrollPosition	void	GWin.h	Set total scroll movement value
208	GCSGetCaptionScrollPosition	long	GWin.h	Get total scroll movement value
209	GCSUpdateScroll	void	GWin.h	Scroll update process
210	GCSIsCaptionScrollFinish	short	GWin.h	Confirm caption scroll completion
211	GCSCallBackMessage	long	GWin.h	Call back message execution process
212	GCSLoadScreenFile	int	GWin.h	Load interpreter screen file
213	GCSSetPenColor	void	GWin.h	Set line color
214	GCSGetPenColor	GColor	GWin.h	Get line color
215	GCSSetPenDash	void	GWin.h	Set line type
216	GCSGetPenDash	unsigned short	GWin.h	Get line type
217	GCSSetPenSize	void	GWin.h	Set line width
218	GCSGetPenSize	unsigned short	GWin.h	Get line width
219	GCSSetAngle	void	GWin.h	Set start angle and finish angle for sector and arc
220	GCSGetAngle	void	GWin.h	Get start angle and finish angle for sector and arc
221	GCSSetBrush	void	GWin.h	Set fill brush
222	GCSGetBrush	GBrush*	GWin.h	Get fill brush
223	GCSSetStartPoint	void	GWin.h	Set line drawing starting point
224	GCSGetStartPoint	void	GWin.h	Get line drawing starting point

No.	Name	Type	File name	Outline
225	GCSSetEndPoint	void	GWin.h	Set line drawing end point
226	GCSGetEndPoint	void	GWin.h	Get line drawing end point
227	GCSSetPoly	short	GWin.h	Set polygon/continuous line
228	GCSGetPoly	GPoly*	GWin.h	Get polygon/continuous line
229	GCSSetDrawAttribute	void	GWin.h	Set drawing attributes for figure
230	GCSBWLInit	void	GWin.h	Initialization
231	GCSBWLAdd	short	GWin.h	Addition
232	GCSBWLRemove	short	GWin.h	Deletion
233	GCSBWLClear	void	GWin.h	All clear
234	GCSBWLMoveFirst	short	GWin.h	Move to first
235	GCSBWLMoveLast	short	GWin.h	Move to last
236	GCSBWLGetFirst	GBaseObject*	GWin.h	Get first object
237	GCSBWLGetLast	GBaseObject*	GWin.h	Get last object
238	GCSBWLFindListItem	short	GWin.h	Find object
239	GCSBWLFindFirstItem	short	GWin.h	Move search position to first item
240	GCSBWLFindNextItem	short	GWin.h	Acquire object at search position and move search position to next
241	GCSBWLFindLastItem	short	GWin.h	Move search position to last item
242	GCSBWLFindPrevItem	short	GWin.h	Acquire object at search position and move search position to previous
243	GCSBWLCount	long	GWin.h	Get number of list items
244	GCSBWLSetItemPosition	void	GWin.h	Set current list search position
245	GCSBWLGetItemPosition	void*	GWin.h	Get current list search position
246	GCSBWLNew	HGBWL	GWin.h	Create list object
247	GCSBWLDelete	void	GWin.h	Delete list object
248	GCSWDCGetHGDRAW	HGDRAW	GWin.h	Get drawing handle
249	GCSWDCOffsetOrigin	void	GWin.h	Move position of origin
250	GCSWDCSetClipRect	void	GWin.h	Define clipping area
251	GCSWDCGetClipRect	void	GWin.h	Get clipping area
252	GCSWDCInceptClipRect	short	GWin.h	Define clipping for the part overlapping the current clipping area
253	GCSWDCBeginDraw	void	GWin.h	Start drawing process
254	GCSWDCEndDraw	void	GWin.h	End drawing process
255	GCSWDCNew	HWDC	GWin.h	Create device context
256	GCSWDCDelete	void	GWin.h	Delete device context
257	GCSWDCSetOrigin	void	GWin.h	Set position of origin
258	GCSChangeActiveFocus	void	GWin.h	Change active focus
259	GCSGetStandardColor	GColor	GWin.h	Get color information according to the theme color of the specified display area
260	GCSCheckUnlockFile	long	GWin.h	Determine whether an unlock key for the custom screen exists

5.1.2 Button

No.	Name	Type	File name	Outline
1	GCSButtonSetAction	void	GCSCButton.h	Set button operations
2	GCSButtonGetAction	unsigned char	GCSCButton.h	Get button operations
3	GCSButtonSetDisplay	void	GCSCButton.h	Set button display
4	GCSButtonGetDisplay	unsigned char	GCSCButton.h	Get button display
5	GCSButtonSetBorderID	void	GCSCButton.h	Set 3D border resource ID
6	GCSButtonGetBorderID	unsigned short	GCSCButton.h	Get 3D border resource ID
7	GCSButtonSetOnDesign	void	GCSCButton.h	Set ON status design
8	GCSButtonGetOnDesign	GDesign*	GCSCButton.h	Get ON status design
9	GCSButtonSetOffDesign	void	GCSCButton.h	Set OFF status design
10	GCSButtonGetOffDesign	GDesign*	GCSCButton.h	Get OFF status design
11	GCSButtonSetFocusDesign	void	GCSCButton.h	Set FOCUS status design
12	GCSButtonGetFocusDesign	GDesign*	GCSCButton.h	Get FOCUS status design
13	GCSButtonSetDisableDesign	void	GCSCButton.h	Set Disable status design
14	GCSButtonGetDisableDesign	GDesign*	GCSCButton.h	Get Disable status design
15	GCSButtonSetFontID	void	GCSCButton.h	Set font resource ID
16	GCSButtonGetFontID	unsigned short	GCSCButton.h	Get font resource ID
17	GCSButtonSetStringID	void	GCSCButton.h	Set caption character string resource ID
18	GCSButtonGetStringID	unsigned short	GCSCButton.h	Get caption character string resource ID
19	GCSButtonSetCaption	void	GCSCButton.h	Set caption information
20	GCSButtonGetCaption	GCaption *	GCSCButton.h	Get caption information
21	GCSButtonSetFocusEffect	void	GCSCButton.h	Set effect during focus
22	GCSButtonGetFocusEffect	unsigned char	GCSCButton.h	Get effect during focus
23	GCSButtonSetStatus	void	GCSCButton.h	Set object status
24	GCSButtonGetStatus	unsigned char	GCSCButton.h	Get object status

5.1.3 TextBox

No.	Name	Type	File name	Outline
1	GCSTextboxSetBorderID	void	GCSCTextBox.h	Set 3D border resource ID
2	GCSTextboxGetBorderID	unsigned short	GCSCTextBox.h	Get 3D border resource ID
3	GCSTextboxSetFontID	void	GCSCTextBox.h	Set font resource ID
4	GCSTextboxGetFontID	unsigned short	GCSCTextBox.h	Get font resource ID
5	GCSTextboxSetCaption	void	GCSCTextBox.h	Set caption information
6	GCSTextboxGetCaption	GCaption *	GCSCTextBox.h	Get caption information
7	GCSTextboxSetFocusEffect	void	GCSCTextBox.h	Set effect during focus
8	GCSTextboxGetFocusEffect	unsigned char	GCSCTextBox.h	Get effect during focus
9	GCSTextboxSetFocusColor	void	GCSCTextBox.h	Set background color during focus
10	GCSTextboxGetFocusColor	GColor	GCSCTextBox.h	Get background color during focus
11	GCSTextboxSetNormalColor	void	GCSCTextBox.h	Set normal background color
12	GCSTextboxGetNormalColor	GColor	GCSCTextBox.h	Get normal background color
13	GCSTextboxSetDisableColor	void	GCSCTextBox.h	Set background color when disabled
14	GCSTextboxGetDisableColor	GColor	GCSCTextBox.h	Get background color when disabled
15	GCSTextboxSetTextType	void	GCSCTextBox.h	Set character string type
16	GCSTextboxGetTextType	unsigned char	GCSCTextBox.h	Get character string type
17	GCSTextboxSetFormatID	void	GCSCTextBox.h	Set character string display format resource ID
18	GCSTextboxGetFormatID	unsigned short	GCSCTextBox.h	Get character string display format resource ID
19	GCSTextboxSetPasswordStatus	void	GCSCTextBox.h	Set password display status
20	GCSTextboxGetPasswordStatus	short	GCSCTextBox.h	Get password display status
21	GCSTextboxSetCommaStatus	void	GCSCTextBox.h	Set comma display status
22	GCSTextboxGetCommaStatus	short	GCSCTextBox.h	Get comma display status

No.	Name	Type	File name	Outline
23	GCSTextboxSetRefuseInputNumberStatus	void	GCSCTextBox.h	Set no number input status
24	GCSTextboxGetRefuseInputNumberStatus	short	GCSCTextBox.h	Get no number input status
25	GCSTextboxSetRefuseInputSmallLetterStatus	void	GCSCTextBox.h	Set one-byte lower case character input disabled status
26	GCSTextboxGetRefuseInputSmallLetterStatus	short	GCSCTextBox.h	Get one-byte lower case character input disabled status
27	GCSTextboxSetRefuseInputCapitalLetterStatus	void	GCSCTextBox.h	Set one-byte upper case character input disabled status
28	GCSTextboxGetRefuseInputCapitalLetterStatus	short	GCSCTextBox.h	Get one-byte upper case character input disabled status
29	GCSTextboxSetRefuseInputSymbolLetterStatus	void	GCSCTextBox.h	Set one-byte symbol input disabled status
30	GCSTextboxGetRefuseInputSymbolLetterStatus	short	GCSCTextBox.h	Get one-byte symbol input disabled status
31	GCSTextboxSetRefuseInputFullLetterStatus	void	GCSCTextBox.h	Set two-byte character input disabled status
32	GCSTextboxGetRefuseInputFullLetterStatus	short	GCSCTextBox.h	Get two-byte character input disabled status
33	GCSTextboxSetCheckMaxStatus	void	GCSCTextBox.h	Set maximum value check status
34	GCSTextboxGetCheckMaxStatus	short	GCSCTextBox.h	Get maximum value check status
35	GCSTextboxSetCheckMinStatus	void	GCSCTextBox.h	Set minimum value check status
36	GCSTextboxGetCheckMinStatus	short	GCSCTextBox.h	Get minimum value check status
37	GCSTextboxSetMaxGValue	void	GCSCTextBox.h	Set maximum value
38	GCSTextboxGetMaxGValue	GValue	GCSCTextBox.h	Get maximum value
39	GCSTextboxSetMinGValue	void	GCSCTextBox.h	Set minimum value
40	GCSTextboxGetMinGValue	GValue	GCSCTextBox.h	Get minimum value
41	GCSTextboxSetString	short	GCSCTextBox.h	Set display character string
42	GCSTextboxGetString	short	GCSCTextBox.h	Get display character string

No.	Name	Type	File name	Outline
43	GCSTextboxSetGValue	short	GCSCTextBox.h	Set display value
44	GCSTextboxGetGValue	short	GCSCTextBox.h	Get display value
45	GCSTextboxSetStringBuffer	short	GCSCTextBox.h	Set display buffer
46	GCSTextboxGetStringBuffer	GTCHAR*	GCSCTextBox.h	Get display buffer
47	GCSTextboxSetBackGroundPattern	void	GCSCTextBox.h	Set background fill status
48	GCSTextboxGetBackGroundPattern	short	GCSCTextBox.h	Get background fill status
49	GCSTextboxSetSelectStringStartPos	void	GCSCTextBox.h	Set selected character string start position
50	GCSTextboxGetSelectStringStartPos	unsigned short	GCSCTextBox.h	Get selected character string start position
51	GCSTextboxSetSelectStringEndPos	void	GCSCTextBox.h	Set selected character string end position
52	GCSTextboxGetSelectStringEndPos	unsigned short	GCSCTextBox.h	Get selected character string end position

5.1.4 Label

No.	Name	Type	File name	Outline
1	GCSLabelSetFontID	void	GCSCLabel.h	Set font resource ID
2	GCSLabelGetFontID	unsigned short	GCSCLabel.h	Get font resource ID
3	GCSLabelSetStringID	void	GCSCLabel.h	Set caption character string resource ID
4	GCSLabelGetStringID	unsigned short	GCSCLabel.h	Get caption character string resource ID
5	GCSLabelSetCaption	void	GCSCLabel.h	Set caption information
6	GCSLabelGetCaption	GCaption *	GCSCLabel.h	Get caption information

5.1.5 List

No.	Name	Type	File name	Outline
1	GCSListSetBorderID	void	GCSCList.h	Set 3D border resource ID
2	GCSListGetBorderID	unsigned short	GCSCList.h	Get 3D border resource ID
3	GCSListSetFontID	void	GCSCList.h	Set font resource ID
4	GCSListGetFontID	unsigned short	GCSCList.h	Get font resource ID
5	GCSListSetCaption	void	GCSCList.h	Set caption information
6	GCSListGetCaption	GCaption *	GCSCList.h	Get caption information
7	GCSListSetFocusColor	void	GCSCList.h	Set background color during focus
8	GCSListGetFocusColor	GColor	GCSCList.h	Get background color during focus
9	GCSListSetScrollButtonColor	void	GCSCList.h	Set scroll button color
10	GCSListGetScrollButtonColor	GColor	GCSCList.h	Get scroll button color
11	GCSListSetScrollBarColor	void	GCSCList.h	Set scroll bar color
12	GCSListGetScrollBarColor	GColor	GCSCList.h	Get scroll bar color
13	GCSListSetNormalColor	void	GCSCList.h	Set normal background color
14	GCSListGetNormalColor	GColor	GCSCList.h	Get normal background color
15	GCSListSetDisableColor	void	GCSCList.h	Set background color when disabled
16	GCSListGetDisableColor	GColor	GCSCList.h	Get background color when disabled
17	GCSListSetScrollBarWidth	void	GCSCList.h	Set scroll bar width
18	GCSListGetScrollBarWidth	unsigned char	GCSCList.h	Get scroll bar width
19	GCSListSetSelectBarColor	void	GCSCList.h	Set selection bar color
20	GCSListGetSelectBarColor	GColor	GCSCList.h	Get selection bar color
21	GCSListSetMaxListLines	void	GCSCList.h	Set maximum number of lines in list
22	GCSListGetMaxListLines	unsigned short	GCSCList.h	Get maximum number of lines in list
23	GCSListAddString	short	GCSCList.h	Add list character string
24	GCSListGetListString	short	GCSCList.h	Get list character string
25	GCSListRemoveAllStrings	void	GCSCList.h	Delete all list character strings
26	GCSListSetCurrentSelect	short	GCSCList.h	Set selected line
27	GCSListGetCurrentSelect	long	GCSCList.h	Get selected line
28	GCSListSetVisibleScrollBarStatus	void	GCSCList.h	Set scroll bar display/non-display status
29	GCSListGetVisibleScrollBarStatus	short	GCSCList.h	Get scroll bar display/non-display status

No.	Name	Type	File name	Outline
30	GCSListSetEnableScrollBarStatus	void	GCSCList.h	Set scroll bar controllable or uncontrollable status
31	GCSListGetEnableScrollBarStatus	short	GCSCList.h	Get scroll bar controllable or uncontrollable status
32	GCSListSetTopLine	void	GCSCList.h	Set list top line
33	GCSListGetTopLine	long	GCSCList.h	Get list top line
34	GCSListInsertString	short	GCSCList.h	Insert list character string
35	GCSListGetLineCount	short	GCSCList.h	Get number of lines in list
36	GCSListRemoveString	short	GCSCList.h	Delete list character string
37	GCSListSetScrollBarBorderID	void	GCSCList.h	Set scroll bar 3D border resource ID
38	GCSListGetScrollBarBorderID	unsigned short	GCSCList.h	Get scroll bar 3D border resource ID
39	GCSListSetSelectStringColor	void	GCSCList.h	Set selected character color
40	GCSListGetSelectStringColor	GColor	GCSCList.h	Get selected character color
41	GCSListSetScrollBarDisplay	void	GCSCList.h	Set display type
42	GCSListGetScrollBarDisplay	unsigned char	GCSCList.h	Get display type
43	GCSListSetScrollArrowColor	void	GCSCList.h	Set arrow color
44	GCSListGetScrollArrowColor	GColor	GCSCList.h	Get arrow color
45	GCSListSetPrevBtnOnImgID	void	GCSCList.h	Set design (upper) resource when button is ON
46	GCSListSetPrevBtnOffImgID	void	GCSCList.h	Set design (upper) resource when button is OFF
47	GCSListGetPrevBtnOnImgID	unsigned short	GCSCList.h	Get design (upper) resource when button is ON
48	GCSListGetPrevBtnOffImgID	unsigned short	GCSCList.h	Get design (upper) resource when button is OFF
49	GCSListSetNextBtnOnImgID	void	GCSCList.h	Set design (lower) resource when button is ON
50	GCSListSetNextBtnOffImgID	void	GCSCList.h	Set design (lower) resource when button is OFF
51	GCSListGetNextBtnOnImgID	unsigned short	GCSCList.h	Get design (lower) resource when button is ON

No.	Name	Type	File name	Outline
52	GCSListGetNextBtnOffImgID	unsigned short	GCSCList.h	Get design (lower) resource when button is OFF
53	GCSListSetScrollBarBackGroundColor	void	GCSCList.h	Set background color
54	GCSListGetScrollBarBackGroundColor	GColor	GCSCList.h	Get background color
55	GCSListSetPinchWidth	void	GCSCList.h	Set pinch width
56	GCSListGetPinchWidth	unsigned short	GCSCList.h	Get pinch width
57	GCSListSetPinchColor	void	GCSCList.h	Set pinch color
58	GCSListGetPinchColor	GColor	GCSCList.h	Get pinch color
59	GCSListSetScrollBarOutLineColor	void	GCSCList.h	Set color of scroll outer line
60	GCSListGetScrollBarOutLineColor	GColor	GCSCList.h	Get color of scroll outer line

5.1.6 Picture

No.	Name	Type	File name	Outline
1	GCSPictureSetDisplay	void	GCSCPicture.h	Set picture display
2	GCSPictureGetDisplay	unsigned char	GCSCPicture.h	Get picture display
3	GCSPictureSetBorderID	void	GCSCPicture.h	Set 3D border resource ID
4	GCSPictureGetBorderID	unsigned short	GCSCPicture.h	Get 3D border resource ID
5	GCSPictureSetFontID	void	GCSCPicture.h	Set font resource ID
6	GCSPictureGetFontID	unsigned short	GCSCPicture.h	Get font resource ID
7	GCSPictureSetStringID	void	GCSCPicture.h	Set caption character string resource ID
8	GCSPictureGetStringID	unsigned short	GCSCPicture.h	Get caption character string resource ID
9	GCSPictureSetCaption	void	GCSCPicture.h	Set caption information
10	GCSPictureGetCaption	GCaption*	GCSCPicture.h	Get caption information
11	GCSPictureSetStatus	void	GCSCPicture.h	Set object status
12	GCSPictureGetStatus	unsigned char	GCSCPicture.h	Get object status
13	GCSPictureSetMaxStatus	void	GCSCPicture.h	Set maximum picture status
14	GCSPictureGetMaxStatus	unsigned char	GCSCPicture.h	Get maximum picture status
15	GCSPictureSetStatusDesign	void	GCSCPicture.h	Set picture status design
16	GCSPictureGetStatusDesign	GDesign*	GCSCPicture.h	Get picture status design

5.1.7 CheckBox

No.	Name	Type	File name	Outline
1	GCSCheckboxSetBorderID	void	GCSCCheckBox.h	Set 3D border resource ID
2	GCSCheckboxGetBorderID	unsigned short	GCSCCheckBox.h	Get 3D border resource ID
3	GCSCheckboxSetFontID	void	GCSCCheckBox.h	Set font resource ID
4	GCSCheckboxGetFontID	unsigned short	GCSCCheckBox.h	Get font resource ID
5	GCSCheckboxSetStringID	void	GCSCCheckBox.h	Set caption character string resource ID
6	GCSCheckboxGetStringID	unsigned short	GCSCCheckBox.h	Get caption character string resource ID
7	GCSCheckboxSetCaption	void	GCSCCheckBox.h	Set caption information
8	GCSCheckboxGetCaption	GCaption *	GCSCCheckBox.h	Get caption information
9	GCSCheckboxSetFocusEffect	void	GCSCCheckBox.h	Set effect during focus
10	GCSCheckboxGetFocusEffect	unsigned char	GCSCCheckBox.h	Get effect during focus
11	GCSCheckboxSetStatus	void	GCSCCheckBox.h	Set object status
12	GCSCheckboxGetStatus	unsigned char	GCSCCheckBox.h	Get object status
13	GCSCheckboxSetBoxSize	void	GCSCCheckBox.h	Set box size
14	GCSCheckboxGetBoxSize	unsigned short	GCSCCheckBox.h	Get box size
15	GCSCheckboxSetBoxColor	void	GCSCCheckBox.h	Set box color
16	GCSCheckboxGetBoxColor	GColor	GCSCCheckBox.h	Get box color
17	GCSCheckboxSetFocusColor	void	GCSCCheckBox.h	Set background color during focus
18	GCSCheckboxGetFocusColor	GColor	GCSCCheckBox.h	Get background color during focus
19	GCSCheckboxSetDisableCaptionColor	void	GCSCCheckBox.h	Set character color when disabled
20	GCSCheckboxGetDisableCaptionColor	GColor	GCSCCheckBox.h	Get character color when disabled
21	GCSCheckboxSetDisableBoxColor	void	GCSCCheckBox.h	Set box color when disabled
22	GCSCheckboxGetDisableBoxColor	GColor	GCSCCheckBox.h	Get box color when disabled

5.1.8 RadioButton

No.	Name	Type	File name	Outline
1	GCSRadiobuttonSetBorderID	void	GCSCRadioButton.h	Set 3D border resource ID
2	GCSRadiobuttonGetBorderID	unsigned short	GCSCRadioButton.h	Get 3D border resource ID
3	GCSRadiobuttonSetFontID	void	GCSCRadioButton.h	Set font resource ID
4	GCSRadiobuttonGetFontID	unsigned short	GCSCRadioButton.h	Get font resource ID
5	GCSRadiobuttonSetStringID	void	GCSCRadioButton.h	Set caption character string resource ID
6	GCSRadiobuttonGetStringID	unsigned short	GCSCRadioButton.h	Get caption character string resource ID
7	GCSRadiobuttonSetCaption	void	GCSCRadioButton.h	Set caption information
8	GCSRadiobuttonGetCaption	GCaption *	GCSCRadioButton.h	Get caption information
9	GCSRadiobuttonSetFocusEffect	void	GCSCRadioButton.h	Set effect during focus
10	GCSRadiobuttonGetFocusEffect	unsigned char	GCSCRadioButton.h	Get effect during focus
11	GCSRadiobuttonSetStatus	void	GCSCRadioButton.h	Set object status
12	GCSRadiobuttonGetStatus	unsigned char	GCSCRadioButton.h	Get object status
13	GCSRadiobuttonSetBoxSize	void	GCSCRadioButton.h	Set box size
14	GCSRadiobuttonGetBoxSize	unsigned short	GCSCRadioButton.h	Get box size
15	GCSRadiobuttonSetBoxColor	void	GCSCRadioButton.h	Set box color
16	GCSRadiobuttonGetBoxColor	GColor	GCSCRadioButton.h	Get box color
17	GCSRadiobuttonSetFocusColor	void	GCSCRadioButton.h	Set background color during focus
18	GCSRadiobuttonGetFocusColor	GColor	GCSCRadioButton.h	Get background color during focus
19	GCSRadiobuttonSetDisableCaptionColor	void	GCSCRadioButton.h	Set character color when disable
20	GCSRadiobuttonGetDisableCaptionColor	GColor	GCSCRadioButton.h	Get character color when disable
21	GCSRadiobuttonSetDisableBoxColor	void	GCSCRadioButton.h	Set box color when disable
22	GCSRadiobuttonGetDisableBoxColor	GColor	GCSCRadioButton.h	Get box color when disable
23	GCSRadiobuttonSetRadioGroup	void	GCSCRadioButton.h	Set radio group No.
24	GCSRadiobuttonGetRadioGroup	unsigned short	GCSCRadioButton.h	Get radio group No.

5.1.9 ProgressBar

No.	Name	Type	File name	Outline
1	GCSProgressbarSetBackGroundDesign	void	GCSCProgressBar.h	Set background design
2	GCSProgressbarGetBackGroundDesign	GDesign*	GCSCProgressBar.h	Get background design
3	GCSProgressbarSetBarDesign	void	GCSCProgressBar.h	Set bar design
4	GCSProgressbarGetBarDesign	GDesign*	GCSCProgressBar.h	Get bar design
5	GCSProgressbarSetRange	void	GCSCProgressBar.h	Set progress bar range
6	GCSProgressbarGetRange	void	GCSCProgressBar.h	Get progress bar range
7	GCSProgressbarSetDirection	void	GCSCProgressBar.h	Set fill direction
8	GCSProgressbarGetDirection	unsigned char	GCSCProgressBar.h	Get fill direction
9	GCSProgressbarGetValue	short	GCSCProgressBar.h	Get current progress bar value
10	GCSProgressbarSetValue	void	GCSCProgressBar.h	Set current progress bar value

5.1.10 HtmlBrowser

No.	Name	Type	File name	Outline
1	GCSHtmlbrowserSetBorderID	void	GCSCHtmlBrowser.h	Set 3D border resource ID
2	GCSHtmlbrowserGetBorderID	unsigned short	GCSCHtmlBrowser.h	Get 3D border resource ID
3	GCSHtmlbrowserSetFontID	void	GCSCHtmlBrowser.h	Set font resource ID
4	GCSHtmlbrowserGetFontID	unsigned short	GCSCHtmlBrowser.h	Get font resource ID
5	GCSHtmlbrowserSetScrollBarWidth	void	GCSCHtmlBrowser.h	Set scroll bar width
6	GCSHtmlbrowserGetScrollBarWidth	unsigned char	GCSCHtmlBrowser.h	Get scroll bar width
7	GCSHtmlbrowserSetTextColor	void	GCSCHtmlBrowser.h	Set text display color
8	GCSHtmlbrowserGetTextColor	GColor	GCSCHtmlBrowser.h	Get text display color
9	GCSHtmlbrowserSetBackColor	void	GCSCHtmlBrowser.h	Set background color
10	GCSHtmlbrowserGetBackColor	GColor	GCSCHtmlBrowser.h	Get background color
11	GCSHtmlbrowserSetLinkColor	void	GCSCHtmlBrowser.h	Set link color
12	GCSHtmlbrowserGetLinkColor	GColor	GCSCHtmlBrowser.h	Get link color
13	GCSHtmlbrowserSetHtmlFileName	void	GCSCHtmlBrowser.h	Set HTML file name
14	GCSHtmlbrowserGetHtmlFileName	GTCHAR*	GCSCHtmlBrowser.h	Get HTML file name
15	GCSHtmlbrowserSetHScrollPosition	void	GCSCHtmlBrowser.h	Set horizontal scroll position

No.	Name	Type	File name	Outline
16	GCSHtmlbrowserGetHScrollPosition	long	GCSCHtmlBrowser.h	Get horizontal scroll position
17	GCSHtmlbrowserSetVScrollPosition	void	GCSCHtmlBrowser.h	Set vertical scroll position
18	GCSHtmlbrowserGetVScrollPosition	long	GCSCHtmlBrowser.h	Get vertical scroll position
19	GCSHtmlbrowserSetLineScrollWidth	void	GCSCHtmlBrowser.h	Set width of one line scroll for horizontal scroll
20	GCSHtmlbrowserGetLineScrollWidth	long	GCSCHtmlBrowser.h	Get width of one line scroll for horizontal scroll
21	GCSHtmlbrowserSetLineScrollHeight	void	GCSCHtmlBrowser.h	Set width of one line scroll for vertical scroll
22	GCSHtmlbrowserGetLineScrollHeight	long	GCSCHtmlBrowser.h	Get width of one line scroll for vertical scroll
23	GCSHtmlbrowserSetPageScrollWidth	void	GCSCHtmlBrowser.h	Set width of page scroll for horizontal scroll
24	GCSHtmlbrowserGetPageScrollWidth	long	GCSCHtmlBrowser.h	Get width of page scroll for horizontal scroll
25	GCSHtmlbrowserSetPageScrollHeight	void	GCSCHtmlBrowser.h	Set width of page scroll for vertical scroll
26	GCSHtmlbrowserGetPageScrollHeight	long	GCSCHtmlBrowser.h	Get width of page scroll for vertical scroll
27	GCSHtmlbrowserGetHtmlWidth	long	GCSCHtmlBrowser.h	Get total width of HTML file
28	GCSHtmlbrowserGetHtmlHeight	long	GCSCHtmlBrowser.h	Get total height of HTML file

5.1.11 ScrollBar

No.	Name	Type	File name	Outline
1	GCSScrollbarexSetDisplay	void	GCSCScrollBarEx.h	Set display type
2	GCSScrollbarexGetDisplay	unsigned char	GCSCScrollBarEx.h	Get display type
3	GCSScrollbarexSetDirection	void	GCSCScrollBarEx.h	Set scroll bar direction
4	GCSScrollbarexGetDirection	unsigned char	GCSCScrollBarEx.h	Get scroll bar direction
5	GCSScrollbarexSetPageSize	void	GCSCScrollBarEx.h	Set page size
6	GCSScrollbarexGetPageSize	unsigned long	GCSCScrollBarEx.h	Get page size
7	GCSScrollbarexSetRange	void	GCSCScrollBarEx.h	Set scroll bar range
8	GCSScrollbarexGetRange	void	GCSCScrollBarEx.h	Get scroll bar range
9	GCSScrollbarexSetScrollArrowColor	void	GCSCScrollBarEx.h	Set scroll arrow color
10	GCSScrollbarexGetScrollArrowColor	GColor	GCSCScrollBarEx.h	Get scroll arrow color
11	GCSScrollbarexSetScrollPosition	void	GCSCScrollBarEx.h	Set scroll bar current position
12	GCSScrollbarexGetScrollPosition	long	GCSCScrollBarEx.h	Get scroll bar current position
13	GCSScrollbarexSetPrevBtnOnImgID	void	GCSCScrollBarEx.h	Set design (upper) resource ID when button is ON
14	GCSScrollbarexSetPrevBtnOffImgID	void	GCSCScrollBarEx.h	Set design (upper) resource ID when button is OFF
15	GCSScrollbarexGetPrevBtnOnImgID	unsigned short	GCSCScrollBarEx.h	Get design (upper) resource ID when button is ON
16	GCSScrollbarexGetPrevBtnOffImgID	unsigned short	GCSCScrollBarEx.h	Get design (upper) resource ID when button is OFF
17	GCSScrollbarexSetNextBtnOnImgID	void	GCSCScrollBarEx.h	Set design (lower) resource ID when button is ON
18	GCSScrollbarexSetNextBtnOffImgID	void	GCSCScrollBarEx.h	Set design (lower) resource ID when button is OFF
19	GCSScrollbarexGetNextBtnOnImgID	unsigned short	GCSCScrollBarEx.h	Get design (lower) resource ID when button is ON
20	GCSScrollbarexGetNextBtnOffImgID	unsigned short	GCSCScrollBarEx.h	Get design (lower) resource ID when button is OFF
21	GCSScrollbarexSetBackGroundColor	void	GCSCScrollBarEx.h	Set background color
22	GCSScrollbarexGetBackGroundColor	GColor	GCSCScrollBarEx.h	Get background color
23	GCSScrollbarexSetOutLineBorderID	void	GCSCScrollBarEx.h	Set outline 3D border

No.	Name	Type	File name	Outline
24	GCSScrollbarexGetOutLineBorderID	unsigned short	GCSCScrollBarEx.h	Get outline 3D border
25	GCSScrollbarexSetPinchWidth	void	GCSCScrollBarEx.h	Set pinch width
26	GCSScrollbarexGetPinchWidth	unsigned short	GCSCScrollBarEx.h	Get pinch width
27	GCSScrollbarexSetPinchColor	void	GCSCScrollBarEx.h	Set pinch color
28	GCSScrollbarexGetPinchColor	GColor	GCSCScrollBarEx.h	Get pinch color
29	GCSScrollbarexSetBarEnableStatus	void	GCSCScrollBarEx.h	Set bar display/non-display
30	GCSScrollbarexGetBarEnableStatus	short	GCSCScrollBarEx.h	Get bar display/non-display
31	GCSScrollbarexSetBorderID	void	GCSCScrollBarEx.h	Set 3D border resource ID
32	GCSScrollbarexGetBorderID	unsigned short	GCSCScrollBarEx.h	Get 3D border resource ID
33	GCSScrollbarexSetScrollButtonColor	void	GCSCScrollBarEx.h	Set scroll button color
34	GCSScrollbarexGetScrollButtonColor	GColor	GCSCScrollBarEx.h	Get scroll button color
35	GCSScrollbarexSetScrollBarColor	void	GCSCScrollBarEx.h	Set scroll bar color
36	GCSScrollbarexGetScrollBarColor	GColor	GCSCScrollBarEx.h	Get scroll bar color
37	GCSScrollbarexSetOutLineColor	void	GCSCScrollBarEx.h	Set color of outline
38	GCSScrollbarexGetOutLineColor	GColor	GCSCScrollBarEx.h	Get color of outline

5.1.12 Edit

No.	Name	Type	File name	Outline
1	GCSEditSetBorderID	void	GCSCEdit.h	Set 3D border resource ID
2	GCSEditGetBorderID	unsigned short	GCSCEdit.h	Get 3D border resource ID
3	GCSEditSetFontID	void	GCSCEdit.h	Set font resource ID
4	GCSEditGetFontID	unsigned short	GCSCEdit.h	Get font resource ID
5	GCSEditAddString	short	GCSCEdit.h	Add character string
6	GCSEditSetString	short	GCSCEdit.h	Set display character string
7	GCSEditGetString	short	GCSCEdit.h	Get display character string
8	GCSEditSetTextColor	void	GCSCEdit.h	Set character display color
9	GCSEditGetTextColor	GColor	GCSCEdit.h	Get character display color
10	GCSEditSetBackColor	void	GCSCEdit.h	Set background color
11	GCSEditGetBackColor	GColor	GCSCEdit.h	Get background color
12	GCSEditSetBrush	void	GCSCEdit.h	Set brush
13	GCSEditDeleteString	void	GCSCEdit.h	Delete character string
14	GCSEditAddLine	short	GCSCEdit.h	Add line
15	GCSEditInsertLine	short	GCSCEdit.h	Insert line
16	GCSEditDeleteLine	short	GCSCEdit.h	Delete line
17	GCSEditGetLineString	short	GCSCEdit.h	Get line character string
18	GCSEditSetLineFeedCode	void	GCSCEdit.h	Set line feed character string
19	GCSEditGetLineFeedCode	unsigned char	GCSCEdit.h	Get line feed character string
20	GCSEditSetCursor	short	GCSCEdit.h	Set cursor position
21	GCSEditGetCursor	short	GCSCEdit.h	Get cursor position
22	GCSEditSetForeColor	void	GCSCEdit.h	Set foreground color
23	GCSEditGetForeColor	GColor	GCSCEdit.h	Get foreground color
24	GCSEditSetFillPattern	void	GCSCEdit.h	Set fill pattern
25	GCSEditGetFillPattern	short	GCSCEdit.h	Get fill pattern
26	GCSEditSetInsertMode	void	GCSCEdit.h	Set insert/overwrite mode
27	GCSEditGetInsertMode	short	GCSCEdit.h	Get insert/overwrite mode
28	GCSEditSetLineBrush	short	GCSCEdit.h	Set line brush
29	GCSEditGetLineBrush	short	GCSCEdit.h	Get line brush
30	GCSEditSetLineTextColor	short	GCSCEdit.h	Set line text display color
31	GCSEditGetLineTextColor	GColor	GCSCEdit.h	Get line text display color
32	GCSEditGetLength	unsigned long	GCSCEdit.h	Get character length
33	GCSEditFind	short	GCSCEdit.h	Search forward
34	GCSEditReverseFind	short	GCSCEdit.h	Search backward
35	GCSEditReplace	short	GCSCEdit.h	Replace
36	GCSEditReplaceAll	unsigned long	GCSCEdit.h	Replace all
37	GCSEditInsertString	short	GCSCEdit.h	Insert character string
38	GCSEditGetLineCount	unsigned long	GCSCEdit.h	Get number of lines
39	GCSEditGetBrush	short	GCSCEdit.h	Get brush
40	GCSEditRemoveString	short	GCSCEdit.h	Delete character string

No.	Name	Type	File name	Outline
41	GCSEditSetTopLine	void	GCSCEdit.h	Set first line displayed for edit control
42	GCSEditGetTopLine	long	GCSCEdit.h	Get first line displayed for edit control
43	GCSEditSetDispCursor	short	GCSCEdit.h	Set cursor position to specified position
44	GCSEditGetDispCursor	short	GCSCEdit.h	Get cursor position
45	GCSEditSetLineDisplayMode	void	GCSCEdit.h	Set "Yes/None" for turn back mode
46	GCSEditGetLineDisplayMode	unsigned char	GCSCEdit.h	Get "Yes/None" for turn back mode
47	GCSEditSetModifyFlag	void	GCSCEdit.h	Set data modify flag
48	GCSEditGetModifyFlag	unsigned char	GCSCEdit.h	Get data modify flag
49	GCSEditConvertLogicalLineToDisplayLine	unsigned long	GCSCEdit.h	Convert logical line number to display line number
50	GCSEditConvertDisplayLineToLogicalLine	unsigned long	GCSCEdit.h	Convert display line number to logical line number
51	GCSEditGetDisplayLineCount	unsigned long	GCSCEdit.h	Get current number of lines displayed

5.1.13 NCTable

No.	Name	Type	File name	Outline
1	GCSNCTableSetCellForeColor	void	GNCWin.h	Set foreground color of each cell
2	GCSNCTableSetCellBackColor	void	GNCWin.h	Set background color of each cell
3	GCSNCTableSetCellString	void	GNCWin.h	Set character string of each cell
4	GCSNCTableGetCellString	void	GNCWin.h	Get character string of each cell
5	GCSNCTableSetSubCursorNum	void	GNCWin.h	Set sub-cursor position
6	GCSNCTableGetSubCursorNum	void	GNCWin.h	Get sub-cursor position
7	GCSNCTableSetCellWidth	void	GNCWin.h	Set cell width
8	GCSNCTableSetCellHight	void	GNCWin.h	Set cell height
9	GCSNCTableGetCellNumFromPoint	void	GNCWin.h	Get cell No. including pertinent coordinates
6	GCSNCTableGetSubCursorNum	void	GNCWin.h	Get sub-cursor position
7	GCSNCTableSetCellWidth	void	GNCWin.h	Set cell width
8	GCSNCTableSetCellHight	void	GNCWin.h	Set cell height

No.	Name	Type	File name	Outline
9	GCSNCTableGetCellNumFromPoint	void	GNCWin.h	Get cell No. including pertinent coordinates
10	GCSNCTableSetLineTitleString	void	GNCWin.h	Set line title
11	GCSNCTableGetLineTitleString	void	GNCWin.h	Get line title
12	GCSNCTableSetRowTitleString	void	GNCWin.h	Set row title
13	GCSNCTableGetRowTitleString	void	GNCWin.h	Get row title
14	GCSNCTableSetLineTitleStringPosition	void	GNCWin.h	Set line title character string position
15	GCSNCTableSetRowTitleStringPosition	void	GNCWin.h	Set row title character string position
16	GCSNCTableSetCellStringPosition	void	GNCWin.h	Set area part character string position
17	GCSNCTableSetBeforeSubCursorNum	void	GNCWin.h	Set previous sub-cursor No.
18	GCSNCTableGetBeforeSubCursorNum	void	GNCWin.h	Get previous sub-cursor No.
19	GCSNCTableSetLineTitleForeColor	void	GNCWin.h	Set foreground color of line title
20	GCSNCTableSetLineTitleBackColor	void	GNCWin.h	Set background color of line title
21	GCSNCTableSetRowTitleForeColor	void	GNCWin.h	Set foreground color of row title
22	GCSNCTableSetRowTitleBackColor	void	GNCWin.h	Set background color of row title
23	GCSNCTableSetSubCursorForeColor	void	GNCWin.h	Set foreground color of sub-cursor
24	GCSNCTableSetSubCursorBackColor	void	GNCWin.h	Set background color of sub-cursor

5.1.14 InputBox

No.	Name	Type	File name	Outline
1	GCSInputBoxSetPasswordStatus	void	GCSCInputBox.h	Set password display status
2	GCSInputBoxGetPasswordStatus	short	GCSCInputBox.h	Get password display status
3	GCSInputBoxSetString	short	GCSCInputBox.h	Set display character string
4	GCSInputBoxGetString	short	GCSCInputBox.h	Get display character string
5	GCSInputBoxSetGValue	short	GCSCInputBox.h	Set display value
6	GCSInputBoxGetGValue	short	GCSCInputBox.h	Get display value
7	GCSInputBoxSetFocusEffect	void	GCSCInputBox.h	Set effect during focus
8	GCSInputBoxGetFocusEffect	unsigned char	GCSCInputBox.h	Get effect during focus

No.	Name	Type	File name	Outline
9	GCSInputBoxSetBackGroundPattern	void	GCSCInputBox.h	Set background fill status
10	GCSInputBoxGetBackGroundPattern	short	GCSCInputBox.h	Get background fill status
11	GCSInputBoxSetSelectStringStartPos	void	GCSCInputBox.h	Set selected character string start position
12	GCSInputBoxGetSelectStringStartPos	unsigned short	GCSCInputBox.h	Get selected character string start position
13	GCSInputBoxSetSelectStringEndPos	void	GCSCInputBox.h	Set selected character string end position
14	GCSInputBoxGetSelectStringEndPos	unsigned short	GCSCInputBox.h	Get selected character string end position
15	GCSInputBoxSetCalcFlag	void	GCSCInputBox.h	Set availability of operation function
16	GCSInputBoxGetCalcFlag	unsigned char	GCSCInputBox.h	Get availability of operation function
17	GCSInputBoxSetInputValue	short	GCSCInputBox.h	Set input content absolutely/incrementally to the target control
18	GCSInputBoxSetEchoback	void	GCSCInputBox.h	Set availability of echo back
19	GCSInputBoxGetEchoback	unsigned char	GCSCInputBox.h	Get availability of echo back
20	GCSInputBoxSetAbsIncType	void	GCSCInputBox.h	Set absolute/incremental input method
21	GCSInputBoxGetAbsIncType	unsigned char	GCSCInputBox.h	Get absolute/incremental input method
22	GCSSetRefrectControl	short	GCSCInputBox.h	Set the control to reflect the INPUT
23	GCSInputBoxGetRefrectControlTextType	unsigned char	GCSCInputBox.h	Set character string type of control to reflect the INPUT
24	GCSInputBoxGetInputMode	unsigned char	GCSCInputBox.h	Get input mode (insert/overwrite)
25	GCSGetSubCursorControl	short	GCSCInputBox.h	Get the ID of the sub cursor display control

5.1.15 SoftKey

No.	Name	Type	File name	Outline
1	GCSSoftKeySetPasswordStatus	void	GCSCSoftKey.h	Set password display status
2	GCSSoftKeyGetPasswordStatus	short	GCSCSoftKey.h	Get password display status
3	GCSSoftKeySetString	short	GCSCSoftKey.h	Set display character string
4	GCSSoftKeyGetString	short	GCSCSoftKey.h	Get display character string
5	GCSSoftKeySetGValue	short	GCSCSoftKey.h	Set display value
6	GCSSoftKeyGetGValue	short	GCSCSoftKey.h	Get display value
7	GCSSoftKeySetFocusEffect	void	GCSCSoftKey.h	Set effect during focus
8	GCSSoftKeyGetFocusEffect	unsigned char	GCSCSoftKey.h	Get effect during focus
9	GCSSoftKeySetCalcFlag	void	GCSCSoftKey.h	Set operation function
10	GCSSoftKeyGetCalcFlag	unsigned char	GCSCSoftKey.h	Get operation function
11	GCSSoftKeySetInputValue	short	GCSCSoftKey.h	Set to the control to reflect the INPUT
12	GCSSoftKeySetAbsIncType	void	GCSCSoftKey.h	Set input method (absolute/incremental)
13	GCSSoftKeyGetAbsIncType	unsigned char	GCSCSoftKey.h	Get input method (absolute/incremental)
14	GCSSetRefrectControl	short	GWin.h	Set to the control to reflect the INPUT
15	GCSSoftKeyGetRefrectControlTextType	unsigned char	GCSCSoftKey.h	Get character string type of INPUT reflection destination control
16	GCSSoftKeyGetInputMode	unsigned char	GCSCSoftKey.h	Get input mode (insert/overwrite)
17	GCSSoftKeySetTitleString	void	GCSCSoftKey.h	Set character string to be displayed on title bar
18	GCSGetSubCursorControl	short	GWin.h	Get the ID of the sub cursor display control

5.1.16 Counter

No.	Name	Type	File name	Outline
1	GCSAxisChange	long	GNXWin.h	Switch displayed axis
2	GCSSetCounterType	long	GNXWin.h	Set counter type

5.1.17 CycleTime

No.	Name	Type	File name	Outline
1	GCSSetTimeType	long	GNXWin.h	Displayed time type setting

5.1.18 LoadMeter

No.	Name	Type	File name	Outline
1	GCSLoadMeterSetDispType	void	GNXWin.h	Set display type of load meter

5.1.19 SPCommand

No.	Name	Type	File name	Outline
1	GCSSetDispSpAxis	long	GNXWin.h	Set spindle to be displayed

5.1.20 NCPLCButton

No.	Name	Type	File name	Outline
1	GCSNCPLCButtonSetAction	void	GNCWin.h	Set PLC button operations
2	GCSNCPLCButtonGetAction	unsigned char	GNCWin.h	Get PLC button operations
3	GCSNCPLCButtonSetDisplay	void	GNCWin.h	Set PLC button display
4	GCSNCPLCButtonGetDisplay	unsigned char	GNCWin.h	Get PLC button display
5	GCSNCPLCButtonSetBorderID	void	GNCWin.h	Set 3D border resource ID
6	GCSNCPLCButtonGetBorderID	unsigned short	GNCWin.h	Get 3D border resource ID
7	GCSNCPLCButtonSetOnDesign	void	GNCWin.h	Set ON status design
8	GCSNCPLCButtonGetOnDesign	GDesign*	GNCWin.h	Get ON status design
9	GCSNCPLCButtonSetOffDesign	void	GNCWin.h	Set OFF status design
10	GCSNCPLCButtonGetOffDesign	GDesign*	GNCWin.h	Get OFF status design
11	GCSNCPLCButtonSetOnOnDesign	void	GNCWin.h	Set ONON status design
12	GCSNCPLCButtonGetOnOnDesign	GDesign*	GNCWin.h	Get ONON status design
13	GCSNCPLCButtonSetOnOffDesign	void	GNCWin.h	Set ONOFF status design
14	GCSNCPLCButtonGetOnOffDesign	GDesign*	GNCWin.h	Get ONOFF status design
15	GCSNCPLCButtonSetFocusDesign	void	GNCWin.h	Set FOCUS status design
16	GCSNCPLCButtonGetFocusDesign	GDesign*	GNCWin.h	Get FOCUS status design
17	GCSNCPLCButtonSetDisableDesign	void	GNCWin.h	Set Disable status design
18	GCSNCPLCButtonGetDisableDesign	GDesign*	GNCWin.h	Get Disable status design
19	GCSNCPLCButtonSetFontID	void	GNCWin.h	Set font resource ID
20	GCSNCPLCButtonGetFontID	unsigned short	GNCWin.h	Get font resource ID
21	GCSNCPLCButtonSetOnStringID	void	GNCWin.h	Set ON status caption character string resource ID
22	GCSNCPLCButtonGetOnStringID	unsigned short	GNCWin.h	Get ON status caption character string resource ID
23	GCSNCPLCButtonSetOffStringID	void	GNCWin.h	Set OFF status caption character string resource ID
24	GCSNCPLCButtonGetOffStringID	unsigned short	GNCWin.h	Get OFF status caption character string resource ID

No.	Name	Type	File name	Outline
25	GCSNCPLCButtonSetOnOnStringID	void	GNCWin.h	Set ONON status caption character string resource ID
26	GCSNCPLCButtonGetOnOnStringID	unsigned short	GNCWin.h	Get ONON status caption character string resource ID
27	GCSNCPLCButtonSetOnOffStringID	void	GNCWin.h	Set ONOFF status caption character string resource ID
28	GCSNCPLCButtonGetOnOffStringID	unsigned short	GNCWin.h	Get ONOFF status caption character string resource ID
29	GCSNCPLCButtonSetCaption	void	GNCWin.h	Set caption information
30	GCSNCPLCButtonGetCaption	Gcaption *	GNCWin.h	Get caption information
31	GCSNCPLCButtonSetFocusEffect	void	GNCWin.h	Set effect during focus
32	GCSNCPLCButtonGetFocusEffect	unsigned char	GNCWin.h	Get effect during focus
33	GCSNCPLCButtonSetStatus	void	GNCWin.h	Set object status
34	GCSNCPLCButtonGetStatus	unsigned char	GNCWin.h	Get object status
35	GCSNCPLCButtonSetDeviceType	short	GNCWin.h	Set PLC device operations
36	GCSNCPLCButtonGetDeviceType	short	GNCWin.h	Get PLC device operations
37	GCSNCPLCButtonSetDevice	short	GNCWin.h	Set PLC device
38	GCSNCPLCButtonGetDevice	short	GNCWin.h	Get PLC device
39	GCSNCPLCButtonSetDeviceVal	short	GNCWin.h	Set PLC device value
40	GCSNCPLCButtonGetDeviceVal	short	GNCWin.h	Get PLC device value

5.1.21 NCPLCTextBox

No.	Name	Type	File name	Outline
1	GCSNCPLCTextboxSetBorderID	void	GNCWin.h	Set 3D border resource ID
2	GCSNCPLCTextboxGetBorderID	unsigned short	GNCWin.h	Get 3D border resource ID
3	GCSNCPLCTextboxSetFontID	void	GNCWin.h	Set font resource ID
4	GCSNCPLCTextboxGetFontID	unsigned short	GNCWin.h	Get font resource ID
5	GCSNCPLCTextboxSetCaption	void	GNCWin.h	Set caption information
6	GCSNCPLCTextboxGetCaption	Gcaption *	GNCWin.h	Get caption information
7	GCSNCPLCTextboxSetFocusEffect	void	GNCWin.h	Set effect during focus
8	GCSNCPLCTextboxGetFocusEffect	unsigned char	GNCWin.h	Get effect during focus
9	GCSNCPLCTextboxSetFocusColor	void	GNCWin.h	Set background color during focus
10	GCSNCPLCTextboxGetFocusColor	GColor	GNCWin.h	Get background color during focus
11	GCSNCPLCTextboxSetNormalColor	void	GNCWin.h	Set normal background color
12	GCSNCPLCTextboxGetNormalColor	GColor	GNCWin.h	Get normal background color

No.	Name	Type	File name	Outline
13	GCSNCPLCTextboxSetDisableColor	void	GNCWin.h	Set background color when disabled
14	GCSNCPLCTextboxGetDisableColor	GColor	GNCWin.h	Get background color when disabled
15	GCSNCPLCTextboxSetTextType	void	GNCWin.h	Set character string type
16	GCSNCPLCTextboxGetTextType	unsigned char	GNCWin.h	Get character string type
17	GCSNCPLCTextboxSetPasswordStatus	void	GNCWin.h	Set password display status
18	GCSNCPLCTextboxGetPasswordStatus	short	GNCWin.h	Get password display status
19	GCSNCPLCTextboxSetCommaStatus	void	GNCWin.h	Set comma display status
20	GCSNCPLCTextboxGetCommaStatus	short	GNCWin.h	Get comma display status
21	GCSNCPLCTextboxSetCheckMaxStatus	void	GNCWin.h	Set maximum value check status
22	GCSNCPLCTextboxSetCheckMaxStatus	short	GNCWin.h	Get maximum value check status
23	GCSNCPLCTextboxSetCheckMinStatus	void	GNCWin.h	Set minimum value check status
24	GCSNCPLCTextboxGetCheckMinStatus	short	GNCWin.h	Get minimum value check status
25	GCSNCPLCTextboxSetMaxGValue	void	GNCWin.h	Set maximum value
26	GCSNCPLCTextboxGetMaxGValue	GValue	GNCWin.h	Get maximum value
27	GCSNCPLCTextboxSetMinGValue	void	GNCWin.h	Set minimum value
28	GCSNCPLCTextboxGetMinGValue	GValue	GNCWin.h	Get minimum value
29	GCSNCPLCTextboxSetString	short	GNCWin.h	Set display character string
30	GCSNCPLCTextboxGetString	short	GNCWin.h	Get display character string
31	GCSNCPLCTextboxSetGValue	short	GNCWin.h	Set display value
32	GCSNCPLCTextboxGetGValue	short	GNCWin.h	Get display value
33	GCSNCPLCTextboxSetStringBuffer	short	GNCWin.h	Set display buffer
34	GCSNCPLCTextboxGetStringBuffer	GTCHAR*	GNCWin.h	Get display buffer
35	GCSNCPLCTextboxSetBackGroundPattern	void	GNCWin.h	Set background fill status
36	GCSNCPLCTextboxGetBackGroundPattern	short	GNCWin.h	Get background fill status
37	GCSNCPLCTextboxSetSelectStringStartPos	void	GNCWin.h	Set selected character string start position
38	GCSNCPLCTextboxGetSelectStringStartPos	unsigned short	GNCWin.h	Get selected character string start position

No.	Name	Type	File name	Outline
39	GCSNCPLCTextboxSetSelectStringEndPos	void	GNCWin.h	Set selected character string end position
40	GCSNCPLCTextboxGetSelectStringEndPos	unsigned short	GNCWin.h	Get selected character string end position
41	GCSNCPLCTextboxSetIntegerWidth	void	GNCWin.h	Set integer part width
42	GCSNCPLCTextboxGetIntegerWidth	unsigned char	GNCWin.h	Get integer part width
43	GCSNCPLCTextboxSetDecimalWidth	void	GNCWin.h	Set decimal part width
44	GCSNCPLCTextboxGetDecimalWidth	unsigned char	GNCWin.h	Get decimal part width
45	GCSNCPLCTextboxSetScale	void	GNCWin.h	Set scale
46	GCSNCPLCTextboxGetScale	void	GNCWin.h	Get scale
47	GCSNCPLCTextboxSetOffset	void	GNCWin.h	Set offset
48	GCSNCPLCTextboxGetOffset	long	GNCWin.h	Get offset
49	GCSNCPLCTextboxSetDevice	short	GNCWin.h	Set PLC device
50	GCSNCPLCTextboxGetDevice	short	GNCWin.h	Get PLC device
51	GCSNCPLCTextboxSetTextSize	void	GNCWin.h	Set size
52	GCSNCPLCTextboxGetTextSize	unsigned long	GNCWin.h	Get size
53	GCSNCPLCTextboxSetZeroSuppressStatus	void	GNCWin.h	Set zero suppress display status
54	GCSNCPLCTextboxGetZeroSuppressStatus	short	GNCWin.h	Get zero suppress display status
55	GCSNCPLCTextboxSetInterlockDevice	short	GNCWin.h	Set interlock device
56	GCSNCPLCTextboxGetInterlockDevice	short	GNCWin.h	Get interlock device
57	GCSNCPLCTextboxSetInterlockDeviceAction	short	GNCWin.h	Set interlock device operations
58	GCSNCPLCTextboxGetInterlockDeviceAction	unsigned char	GNCWin.h	Get interlock device operations
59	GCSNCPLCTextboxSetInterlockDeviceBitPos	short	GNCWin.h	Set interlock device bit position
60	GCSNCPLCTextboxGetInterlockDeviceBitPos	unsigned char	GNCWin.h	Get interlock device bit position
61	GCSNCPLCTextboxSetInterlockDeviceProjectNo	short	GNCWin.h	Set interlock device project number
62	GCSNCPLCTextboxGetInterlockDeviceProjectNo	unsigned char	GNCWin.h	Get interlock device project number
63	GCSNCPLCTextboxSetInterlockColor	void	GNCWin.h	Set background color during interlock
64	GCSNCPLCTextboxGetInterlockColor	GColor	GNCWin.h	Get background color during interlock

5.1.22 NCDataTextBox

No.	Name	Type	File name	Outline
1	GCSNCDataTextboxSetBorderID	void	GNCWin.h	Set 3D border resource ID
2	GCSNCDataTextboxGetBorderID	unsigned short	GNCWin.h	Get 3D border resource ID
3	GCSNCDataTextboxSetFontID	void	GNCWin.h	Set font resource ID
4	GCSNCDataTextboxGetFontID	unsigned short	GNCWin.h	Get font resource ID
5	GCSNCDataTextboxSetCaption	void	GNCWin.h	Set caption information
6	GCSNCDataTextboxGetCaption	GCaption*	GNCWin.h	Get caption information
7	GCSNCDataTextboxSetFocusEffect	void	GNCWin.h	Set effect during focus
8	GCSNCDataTextboxGetFocusEffect	unsigned char	GNCWin.h	Get effect during focus
9	GCSNCDataTextboxSetFocusColor	void	GNCWin.h	Set background color during focus
10	GCSNCDataTextboxGetFocusColor	GColor	GNCWin.h	Get background color during focus
11	GCSNCDataTextboxSetNormalColor	void	GNCWin.h	Set normal background color
12	GCSNCDataTextboxGetNormalColor	GColor	GNCWin.h	Get normal background color
13	GCSNCDataTextboxSetDisableColor	void	GNCWin.h	Set background color when disabled
14	GCSNCDataTextboxGetDisableColor	GColor	GNCWin.h	Get background color when disabled
15	GCSNCDataTextboxSetTextType	void	GNCWin.h	Set character string type
16	GCSNCDataTextboxGetTextType	unsigned char	GNCWin.h	Get character string type
17	GCSNCDataTextboxSetCommaStatus	void	GNCWin.h	Set comma display status
18	GCSNCDataTextboxGetCommaStatus	short	GNCWin.h	Get comma display status
19	GCSNCDataTextboxSetRefuseInputNumberStatus	void	GNCWin.h	Set no number input status

No.	Name	Type	File name	Outline
20	GCSNCDataTextboxGetRefuseInputNumberStatus	short	GNCWin.h	Get no number input status
21	GCSNCDataTextboxSetRefuseInputSmallLetterStatus	void	GNCWin.h	Set one-byte lower case character input disabled status
22	GCSNCDataTextboxGetRefuseInputSmallLetterStatus	short	GNCWin.h	Get one-byte lower case character input disabled status
23	GCSNCDataTextboxSetRefuseInputCapitalLetterStatus	void	GNCWin.h	Set one-byte upper case character input disabled status
24	GCSNCDataTextboxGetRefuseInputCapitalLetterStatus	short	GNCWin.h	Get one-byte upper case character input disabled status
25	GCSNCDataTextboxSetRefuseInputSymbolLetterStatus	void	GNCWin.h	Set one-byte symbol input disabled status
26	GCSNCDataTextboxGetRefuseInputSymbolLetterStatus	short	GNCWin.h	Get one-byte symbol input disabled status
27	GCSNCDataTextboxSetZeroSuppressStatus	void	GNCWin.h	Set zero suppress display status
28	GCSNCDataTextboxGetZeroSuppressStatus	short	GNCWin.h	Get zero suppress display status
29	GCSNCDataTextboxSetString	short	GNCWin.h	Set display character string
30	GCSNCDataTextboxGetString	short	GNCWin.h	Get display character string
31	GCSNCDataTextboxSetStringBuffer	short	GNCWin.h	Set input character string
32	GCSNCDataTextboxGetStringBuffer	GTCHAR*	GNCWin.h	Get input character string
33	GCSNCDataTextboxSetGNCValue	short	GNCWin.h	Set display value
34	GCSNCDataTextboxGetGNCValue	short	GNCWin.h	Get display value
35	GCSNCDataTextboxSetBackGroundPattern	void	GNCWin.h	Set background fill status

No.	Name	Type	File name	Outline
36	GCSNCDataTextboxGetBackGroundPattern	short	GNCWin.h	Get background fill status
37	GCSNCDataTextboxSetIntegerWidth	void	GNCWin.h	Set integer part width
38	GCSNCDataTextboxGetIntegerWidth	unsigned char	GNCWin.h	Get integer part width
39	GCSNCDataTextboxSetDecimalWidth	void	GNCWin.h	Set decimal part width
40	GCSNCDataTextboxGetDecimalWidth	unsigned char	GNCWin.h	Get decimal part width
41	GCSNCDataTextboxSetScale	void	GNCWin.h	Set scale
42	GCSNCDataTextboxGetScale	void	GNCWin.h	Get scale
43	GCSNCDataTextboxSetOffset	void	GNCWin.h	Set offset
44	GCSNCDataTextboxGetOffset	long	GNCWin.h	Get offset
45	GCSNCDataTextboxSetSystemNumber	void	GNCWin.h	Set system number
46	GCSNCDataTextboxGetSystemNumber	long	GNCWin.h	Get system number
47	GCSNCDataTextboxSetGround	void	GNCWin.h	Set ground information
48	GCSNCDataTextboxGetGround	long	GNCWin.h	Get ground information
49	GCSNCDataTextboxSetAxisInfo	void	GNCWin.h	Set axis informmation
50	GCSNCDataTextboxGetAxisInfo	unsigned long	GNCWin.h	Get axis informmation
51	GCSNCDataTextboxSetSection	void	GNCWin.h	Set section number
52	GCSNCDataTextboxGetSection	long	GNCWin.h	Get section number
53	GCSNCDataTextboxSetSubSection	void	GNCWin.h	Set sub-section number
54	GCSNCDataTextboxGetSubSection	long	GNCWin.h	Get sub-section number
55	GCSNCDataTextboxSetDataType	void	GNCWin.h	Set data type
56	GCSNCDataTextboxGetDataType	long	GNCWin.h	Get data type

5.1.23 Menu

No.	Name	Type	File name	Outline
1	GCSRefresh	void	GNXWin.h	Redraw menu
2	GCSSetRefreshValidFlag	void	GNXWin.h	Set menu display
3	GCSSetMenuButtonName_all	long	GNXWin.h	Set character string of level 1 menu (x10)
4	GCSSetMenuButtonUpperName_all	long	GNXWin.h	Set character string of level 2 menu (upper) (x10)
5	GCSSetMenuButtonLowerName_all	long	GNXWin.h	Set character string of level 2 menu (lower) (x10)
6	GCSSetMenuButtonName_one	long	GNXWin.h	Set character string of level 1 menu (x1)
7	GCSSetMenuButtonUpperName_one	long	GNXWin.h	Set character string of level 2 menu (upper) (x1)
8	GCSSetMenuButtonLowerName_one	long	GNXWin.h	Set character string of level 2 menu (lower) (x1)
9	GCSSetMenuButtonState_one	long	GNXWin.h	Set status of level 1 menu (x1)
10	GCSSetMenuButtonUpperState_one	long	GNXWin.h	Set status of level 2 menu (upper) (x1)
11	GCSSetMenuButtonLowerState_one	long	GNXWin.h	Set status of level 2 menu (lower) (x1)
12	GCSGetMenuButtonState_one	long	GNXWin.h	Get status of level 1 menu (x1)
13	GCSGetMenuButtonUpperState_one	long	GNXWin.h	Get status of level 2 menu (upper) (x1)
14	GCSGetMenuButtonLowerState_one	long	GNXWin.h	Get status of level 2 menu (lower) (x1)
15	GCSSetMenuButtonOffIcon_one	long	GNXWin.h	Set icon when level 1 menu (x1) is OFF
16	GCSSetMenuButtonOnIcon_one	long	GNXWin.h	Set icon when level 1 menu (x1) is ON
17	GCSChangeMenuType	long	GNXWin.h	Switch to level 1/level 2 display
18	GCSSetDstObject	long	GNXWin.h	Specify target control for calling
19	GCSSetNextMenuButtonState	long	GNXWin.h	Set selection status of menu switching button
20	GCSSetOpeMessage	long	GNXWin.h	Set menu operation message
21	GCSBeginAnmState	long	GNXWin.h	Start setting of menu animation operation
22	GCSEndAnmState	long	GNXWin.h	End setting of menu animation operation
23	GCSStopAnm	long	GNXWin.h	Stop menu animation

5.1.24 Extension Menu

No.	Name	Type	File name	Outline
1	GCSNCExMenuSetFontID	void	GNCWin.h	Set font resource ID
2	GCSNCExMenuGetFontID	unsigned short	GNCWin.h	Get font resource ID
3	GCSNCExMenuGetMenuNumFromPoint	unsigned short	GNCWin.h	Get menu number from position where mouse is clicked
4	GCSNCExMenuGetDispMenuGroup	unsigned short	GNCWin.h	Get displayed menu group
5	GCSNCExMenuSetDisplay	void	GNCWin.h	Set display type of extension menu
6	GCSNCExMenuGetDisplay	unsigned char	GNCWin.h	Get display type of extension menu
7	GCSNCExMenuSetUpperMenuOnPattern	void	GNCWin.h	Set pattern at the time of upper row extension menu ON
8	GCSNCExMenuGetUpperMenuOnPattern	short	GNCWin.h	Get pattern at the time of upper row extension menu ON
9	GCSNCExMenuSetLowerMenuOnPattern	void	GNCWin.h	Set pattern at the time of lower row extension menu ON
10	GCSNCExMenuGetLowerMenuOnPattern	short	GNCWin.h	Get pattern at the time of lower row extension menu ON
11	GCSNCExMenuSetUpperMenuOnForeColor	void	GNCWin.h	Set foreground color at the time of upper row extension menu ON
12	GCSNCExMenuGetUpperMenuOnForeColor	long	GNCWin.h	Get foreground color at the time of upper row extension menu ON
13	GCSNCExMenuSetLowerMenuOnForeColor	void	GNCWin.h	Set foreground color at the time of lower row extension menu ON
14	GCSNCExMenuGetLowerMenuOnForeColor	long	GNCWin.h	Get foreground color at the time of lower row extension menu ON
15	GCSNCExMenuSetUpperMenuOnBackColor	void	GNCWin.h	Set background color at the time of upper row extension menu ON
16	GCSNCExMenuGetUpperMenuOnBackColor	long	GNCWin.h	Get background color at the time of upper row extension menu ON
17	GCSNCExMenuSetLowerMenuOnBackColor	void	GNCWin.h	Set background color at the time of lower row extension menu ON

No.	Name	Type	File name	Outline
18	GCSNCExMenuGetLowerMenuOnBackColor	long	GNCWin.h	Get background color at the time of lower row extension menu ON
19	GCSNCExMenuSetMenuOnImage	void	GNCWin.h	Set image ID at the time of extension menu ON
20	GCSNCExMenuGetMenuOnImage	unsigned short	GNCWin.h	Get image ID at the time of extension menu ON
21	GCSNCExMenuSetUpperMenuOnStringColor	void	GNCWin.h	Set character color at the time of upper row extension menu ON
22	GCSNCExMenuGetUpperMenuOnStringColor	long	GNCWin.h	Get character color at the time of upper row extension menu ON
23	GCSNCExMenuSetLowerMenuOnStringColor	void	GNCWin.h	Set character color at the time of lower row extension menu ON
24	GCSNCExMenuGetLowerMenuOnStringColor	long	GNCWin.h	Get character color at the time of lower row extension menu ON
25	GCSNCExMenuSetUpperMenuOffPattern	void	GNCWin.h	Set pattern at the time of upper row extension menu OFF
26	GCSNCExMenuGetUpperMenuOffPattern	short	GNCWin.h	Get pattern at the time of upper row extension menu OFF
27	GCSNCExMenuSetLowerMenuOffPattern	void	GNCWin.h	Set pattern at the time of lower row extension menu OFF
28	GCSNCExMenuGetLowerMenuOffPattern	short	GNCWin.h	Get pattern at the time of lower row extension menu OFF
29	GCSNCExMenuSetUpperMenuOffForeColor	void	GNCWin.h	Set foreground color at the time of upper row extension menu OFF
30	GCSNCExMenuGetUpperMenuOffForeColor	long	GNCWin.h	Get foreground color at the time of upper row extension menu OFF
31	GCSNCExMenuSetLowerMenuOffForeColor	void	GNCWin.h	Set foreground color at the time of lower row extension menu OFF
32	GCSNCExMenuGetLowerMenuOffForeColor	long	GNCWin.h	Get foreground color at the time of lower row extension menu OFF

No.	Name	Type	File name	Outline
33	GCSNCExMenuSetUpperMenuOffBackColor	void	GNCWin.h	Set background color at the time of upper row extension menu OFF
34	GCSNCExMenuGetUpperMenuOffBackColor	long	GNCWin.h	Get background color at the time of upper row extension menu OFF
35	GCSNCExMenuSetLowerMenuOffBackColor	void	GNCWin.h	Set background color at the time of upper row extension menu OFF
36	GCSNCExMenuGetLowerMenuOffBackColor	long	GNCWin.h	Get background color at the time of lower row extension menu OFF
37	GCSNCExMenuSetMenuOffImage	void	GNCWin.h	Set image ID at the time of extension menu OFF
38	GCSNCExMenuGetMenuOffImage	unsigned short	GNCWin.h	Get image ID when extension menu is OFF
39	GCSNCExMenuSetUpperMenuOffStringColor	void	GNCWin.h	Set character color at the time of upper row extension menu OFF
40	GCSNCExMenuGetUpperMenuOffStringColor	long	GNCWin.h	Get character color at the time of upper row extension menu OFF
41	GCSNCExMenuSetLowerMenuOffStringColor	void	GNCWin.h	Set character color at the time of lower row extension menu OFF
42	GCSNCExMenuGetLowerMenuOffStringColor	long	GNCWin.h	Get character color at the time of lower row extension menu OFF
43	GCSNCExMenuSetMenuAction	void	GNCWin.h	Set button operations of extension menu
44	GCSNCExMenuGetMenuAction	unsigned short	GNCWin.h	Get button operations of extension menu
45	GCSNCExMenuSetMenuString	void	GNCWin.h	Set character string of extension menu
46	GCSNCExMenuGetMenuString	GTCHAR*	GNCWin.h	Get character string of extension menu
47	GCSNCExMenuSetMenuStringID	void	GNCWin.h	Set character string resource ID of extension menu
48	GCSNCExMenuGetMenuStringID	unsigned short	GNCWin.h	Get character string resource ID of extension menu

No.	Name	Type	File name	Outline
49	GCSNCExMenuSetMenuPageAction	void	GNCWin.h	Set screen switch operation
50	GCSNCExMenuGetMenuPageAction	unsigned short	GNCWin.h	Get screen switch operation
51	GCSNCExMenuSetMenuPageNum	void	GNCWin.h	Set Panel/Window number
52	GCSNCExMenuGetMenuPageNum	short	GNCWin.h	Get Panel/Window number
53	GCSNCExMenuSetMenuStatus	void	GNCWin.h	Set extension menu button status
54	GCSNCExMenuGetMenuStatus	unsigned char	GNCWin.h	Get extension menu button status
55	GCSNCExMenuSetMenuViewStatus	void	GNCWin.h	Set display status of extension menu
56	GCSNCExMenuGetMenuViewStatus	unsigned short	GNCWin.h	Get display status of extension menu
57	GCSNCExMenuSetMenuStringPosition	void	GNCWin.h	Set character string display position of extension menu
58	GCSNCExMenuGetMenuStringPosition	void	GNCWin.h	Get character string display position of extension menu
59	GCSNCExMenuSetUpperMenuGroupNum	void	GNCWin.h	Set upper row menu group number
60	GCSNCExMenuGetUpperMenuGroupNum	unsigned short	GNCWin.h	Get upper row menu group number
61	GCSNCExMenuSetNextMenuGroupNum	void	GNCWin.h	Set destination menu group number
62	GCSNCExMenuGetNextMenuGroupNum	unsigned short	GNCWin.h	Get destination menu group number
63	GCSNCExMenuChangeMenuGroupNum	void	GNCWin.h	Change menu group number to be displayed
64	GCSNCExMenuSetOpeMessage	void	GNCWin.h	Set operation message of extension menu (specify character string)
65	GCSNCExMenuSetOpeMessageID	void	GNCWin.h	Set operation message (specify resource ID)
66	GCSNCExMenuSetMenuDevice	short	GNCWin.h	Set PLC device
67	GCSNCExMenuGetMenuDevice	char*	GNCWin.h	Get PLC device
68	GCSNCExMenuSetMenuDeviceEnable	short	GNCWin.h	Set PLC device enabled
69	GCSNCExMenuGetMenuDeviceEnable	unsigned short	GNCWin.h	Get PLC device enabled
70	GCSNCExMenuSetMenuDeviceBitPosition	short	GNCWin.h	Set bit position of PLC device

No.	Name	Type	File name	Outline
71	GCSNCExMenuGetMenuDeviceBitPosition	unsigned char	GNCWin.h	Get bit position of PLC device
72	GCSNCExMenuSetMenuDeviceVal	short	GNCWin.h	Set PLC device value
73	GCSNCExMenuGetMenuDeviceVal	unsigned char	GNCWin.h	Get PLC device value
74	GCSNCExMenuSetButtonGroupAllOffState	void	GNCWin.h	Set all OFF state of button group
75	GCSNCExMenuGetButtonGroupAllOffState	unsigned char	GNCWin.h	Get all OFF state of button group
76	GCSNCExMenuBeginAnimState	void	GNCWin.h	Start setting of extension menu animation operation
77	GCSNCExMenuEndAnimState	void	GNCWin.h	End setting of extension menu animation operation (animation start)
78	GCSNCExMenuStopAnim	void	GNCWin.h	Stop extension menu animation

5.1.25 FileInOut

No.	Name	Type	File name	Outline
1	GCSIOCheckUpDateFile	long	GNXWin.h	Check existence of file
2	GCSIOFileTransfer	long	GNXWin.h	Start file transfer
3	GCSIODeleteFile	long	GNXWin.h	Delete file/directory
4	GCSIOCreateDirectory	long	GNXWin.h	Create directory
5	GCSIORenameFile	long	GNXWin.h	Rename file/directory
6	GCSInterrupt	long	GNXWin.h	Cancel file transfer
7	GCSIOInitializeMessage	void	GNXWin.h	Initialize file input/output part
8	GCSGetLastError	long	GNXWin.h	Get error information
9	GCSIOGetTransferStatus	long	GNXWin.h	Get transfer status

5.1.26 AlarmMessage

No.	Name	Type	File name	Outline
1	GCSSetDispMessageType	long	GNXWin.h	Set character string display format

5.1.27 MonitorStatus

No.	Name	Type	File name	Outline
1	GCSSetDispSysNumber	short	GNXWin.h	Set part system number to display
2	GCSGetDispSysNumber	short	GNXWin.h	Get part system number to display

5.1.28 NCPageChangeButton

No.	Name	Type	File name	Outline
1	GCSNCPageChgBtnSetDestinationPageID	long	GNCWin.h	Set destination screen number
2	GCSNCPageChgBtnGetDestinationPageID	unsigned short	GNCWin.h	Get destination screen number
3	GCSNCPageChgBtnSetFocusMoveToWindow	long	GNCWin.h	Set focus move to window
4	GCSNCPageChgBtnGetFocusMoveToWindow	unsigned short	GNCWin.h	Get focus move to window
5	GCSNCPageChgBtnSetCloseWindow	long	GNCWin.h	Set close status of source window
6	GCSNCPageChgBtnGetCloseWindow	unsigned short	GNCWin.h	Get close status of source window
7	GCSNCPageChgBtnSetInputKeyID	long	GNCWin.h	Set input key ID
8	GCSNCPageChgBtnGetInputKeyID	unsigned short	GNCWin.h	Get input key ID

5.1.29 Graph

No.	Name	Type	File name	Outline
1	GCSNCGraphSetChartType	void	GNCWin.h	Set graph type
2	GCSNCGraphGetChartType	unsigned char	GNCWin.h	Get graph type
3	GCSNCGraphSetDispDirection	void	GNCWin.h	Set display direction
4	GCSNCGraphGetDispDirection	unsigned char	GNCWin.h	Get display direction
5	GCSNCGraphSetItemNum	void	GNCWin.h	Set number of items
6	GCSNCGraphGetItemNum	unsigned short	GNCWin.h	Get number of items
7	GCSNCGraphSetSection	void	GNCWin.h	Set section number
8	GCSNCGraphGetSection	long	GNCWin.h	Get section number
9	GCSNCGraphSetSubSection	void	GNCWin.h	Set sub-section number
10	GCSNCGraphGetSubSection	long	GNCWin.h	Get sub-section number
11	GCSNCGraphSetGround	void	GNCWin.h	Set ground
12	GCSNCGraphGetGround	long	GNCWin.h	Get ground
13	GCSNCGraphSetAxisInfo	void	GNCWin.h	Set axis number
14	GCSNCGraphGetAxisInfo	long	GNCWin.h	Get axis number
15	GCSNCGraphSetDataType	void	GNCWin.h	Set data type
16	GCSNCGraphGetDataType	long	GNCWin.h	Get data type
17	GCSNCGraphSetReadMethod	void	GNCWin.h	Set reading method
18	GCSNCGraphGetReadMethod	unsigned char	GNCWin.h	Get reading method
19	GCSNCGraphSetReadOffset	void	GNCWin.h	Set read offset
20	GCSNCGraphGetReadOffset	long	GNCWin.h	Get read offset
21	GCSNCGraphSetGraphFillPattern	void	GNCWin.h	Set pattern
22	GCSNCGraphGetGraphFillPattern	short	GNCWin.h	Get pattern
23	GCSNCGraphSetGraphForeColor	void	GNCWin.h	Set foreground color

No.	Name	Type	File name	Outline
24	GCSNCGraphGetGraphForeColor	short	GNCWin.h	Get foreground color
25	GCSNCGraphSetGraphBackColor	void	GNCWin.h	Set background color
26	GCSNCGraphGetGraphBackColor	short	GNCWin.h	Get background color
27	GCSNCGraphSetScaleColor	void	GNCWin.h	Set scale color
28	GCSNCGraphGetScaleColor	long	GNCWin.h	Get scale color
29	GCSNCGraphSetScaleNum	void	GNCWin.h	Set scale number
30	GCSNCGraphGetScaleNum	unsigned short	GNCWin.h	Get scale number
31	GCSNCGraphSetScaleWidth	void	GNCWin.h	Set scale width
32	GCSNCGraphGetScaleWidth	unsigned short	GNCWin.h	Get scale width
33	GCSNCGraphSetScaleMin	void	GNCWin.h	Set scale (minimum)
34	GCSNCGraphGetScaleMin	void	GNCWin.h	Get scale (minimum)
35	GCSNCGraphSetScaleMax	void	GNCWin.h	Set scale (maximum)
36	GCSNCGraphGetScaleMax	void	GNCWin.h	Get scale (maximum)
37	GCSNCGraphSetLowerLimit	void	GNCWin.h	Set boundary value (lower limit)
38	GCSNCGraphGetLowerLimit	void	GNCWin.h	Get boundary value (lower limit)
39	GCSNCGraphSetUpperLimit	void	GNCWin.h	Set boundary value (upper limit)
40	GCSNCGraphGetUpperLimit	void	GNCWin.h	Get boundary value (upper limit)
41	GCSNCGraphSetSort	void	GNCWin.h	Set sort type
42	GCSNCGraphGetSort	unsigned char	GNCWin.h	Get sort type
43	GCSNCGraphGetItemData	short	GNCWin.h	Get graph item data (reading order)
44	GCSNCGraphSetSystemNumber	void	GNCWin.h	Set part system number
45	GCSNCGraphGetSystemNumber	long	GNCWin.h	Get part system number

5.1.30 AlarmList

No.	Name	Type	File name	Outline
1	GCSNCAlarmListUpdateHistory	void	GNCWin.h	Update alarm history
2	GCSNCAlarmListSetDispType	void	GNCWin.h	Set display type
3	GCSNCAlarmListGetDispType	unsigned char	GNCWin.h	Get display type
4	GCSNCAlarmListGetSelectLine	short	GNCWin.h	Get selected line number
5	GCSNCAlarmListGetAlarmInfo	long	GNCWin.h	Get alarm information

5.1.31 NC Data Access Setting

No.	Name	Type	File name	Outline
1	GNCDataSetGNCValue	short	genifa.h	Set value to CNC
2	GNCDataGetGNCValue	short	genifa.h	Get value from CNC
3	GNCDataSetString	short	genifa.h	Set character string to CNC
4	GNCDataGetString	short	genifa.h	Get character string from CNC
5	melSetData	DWORD	ncmcapi.h	Set data
6	melGetData	DWORD	ncmcapi.h	Get data
7	melGetLumpData	DWORD	ncmcapi.h	Get continuous data in a batch
8	melSelectExecPrg	DWORD	ncmcapi.h	Search operation program

5.2 Description of GCS Functions

5.2.1 NC Designer2 Compatibility

GCSGetType	
Outline	Get object type
Syntax	unsigned short GCSGetType(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Object type
Details	Gets the object type.

GCSGetID	
Outline	Get object ID
Syntax	Unsigned short GCSGetID(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Object ID
Details	Gets the object ID.

GCSGetXPosition

Outline	Get horizontal position
Syntax	short GCSGetXPosition(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Horizontal position
Details	Gets the horizontal position of the object.

GCSGetYPosition

Outline	Get vertical position
Syntax	short GCSGetYPosition(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Vertical position
Details	Gets the vertical position of the object.

GCSSetPosition

Outline	Set position
Syntax	void GCSSetPosition(GBaseObject *pSelf, GPoint *pPosition)
Argument	(i) GBaseObject *pSelf : object for operation (i) GPoint *pPosition : point structure being set
Return value	None
Details	Sets the position of the object using a point structure.

GCSGetPosition

Outline	Get position
Syntax	void GCSGetPosition(GBaseObject *pSelf, GPoint *pPosition)
Argument	(i) GBaseObject *pSelf : object for operation (o) GPoint *pPosition : point structure being stored
Return value	None
Details	Gets the position of the object using a point structure.

GCSGetWidth

Outline	Get width
Syntax	short GCSGetWidth(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Width
Details	Gets the object width.

GCSGetHeight

Outline	Get height
Syntax	short GCSGetHeight(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Height
Details	Gets the object height.

GCSGetSize	

Outline	Get size
Syntax	void GCSGetSize(GBaseObject *pSelf, GPoint *pSize)
Argument	(i) GBaseObject *pSelf : object for operation (o) GPoint *pSize : point structure for which the size is being stored
Return value	None
Details	Gets the object size.

GCSSetBounds

Outline Set relative position within parent object

Syntax
```
void GCSSetBounds(
  GBaseObject  *pSelf,
  GRect        *pBounds
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GRect *pBounds : rectangle variable

Return value None

Details Sets the relative position of the object within parent object.

GCSGetBounds

Outline Get relative position within parent object

Syntax
```
void GCSGetBounds(
  GBaseObject  *pSelf,
  GRect        *pBounds
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(o) GRect *pBounds : rectangle variable

Return value None

Details Gets the relative position of the object within parent object.

GCSCalcClientRect

Outline	Get coordinate position of client area
Syntax	void GCSCalcClientRect(GBaseObject *pSelf, GRect *pClient)
Argument	(i) GBaseObject *pSelf : object for operation (o) GRect *pClinet : client area coordinates
Return value	None
Details	Gets the client area for the object with the upper left of the object as the origin.

GCSGetClientRect

Outline Get object client area

Syntax
```
void GCSGetClientRect(
  GBaseObject  *pSelf,
  GRect        *pClient
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(o) GRect *pClinet : client area

Return value None

Details
Gets the client area for the object.
Gets the rectangle with the upper left of the object as the origin.

GCSClientToScreen

Outline Convert client coordinates to screen coordinates

Syntax
```
void GCSClientToScreen(
  GBaseObject  *pSelf,
  GRect        *pRect
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i/o) GRect *pRect : client coordinate rectangle being converted

Return value None

Details
Converts the object client coordinates to screen coordinates.

GCSGetStyle	

Outline	Batch style acquisition
Syntax	unsigned long GCSGetStyle(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Style value
Details	Gets the styles of the object all at once.

GCSSetVisibleStatus

Outline Set display/non-display status

Syntax
```
void GCSSetVisibleStatus(
  GBaseObject   *pSelf,
  short         fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : display/non-display status
FALSE : non-display
TRUE : display

Return value None

Details Sets the display/non-display status for the object.
When switching the display/non-display, the object area is registered as a redraw area.

GCSGetVisibleStatus

Outline Get display/non-display status

Syntax
```
short GCSGetVisibleStatus(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Display/non-display status
FALSE : non-display
TRUE : display

Details Gets the display/non-display status for the object.

GCSSetEnableStatus

Outline Set controllable or uncontrollable status

Syntax
```
void GCSSetEnableStatus(
  GBaseObject   *pSelf,
  short         fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : controllable or uncontrollable status
FALSE : uncontrollable
TRUE : controllable

Return value None

Details Sets the controllable or uncontrollable status for the object.
Controllability with mouse input or key input is determined according to this flag.

GCSGetEnableStatus

Outline Get controllable or uncontrollable status

Syntax
```
short GCSGetEnableStatus(
  GBaseObject   *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value
FALSE : uncontrollable
TRUE : controllable

Details Gets the controllable or uncontrollable status for the object.

GCSSetConsumableStatus

Outline Set input event consumable or input event not consumable status

Syntax
```
void GCSSetConsumableStatus(
  GBaseObject   *pSelf,
  short         fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : consumable/not consumable status
FALSE : not consumable
TRUE : consumable

Return value None

Details Sets the event consumption or non-consumption when the object has uncontrollable status.
Whether the event is consumed or not consumed is controlled by this flag for this object.

GCSGetConsumableStatus

Outline Get input event consumable or input event not consumable status

Syntax
```
short GCSGetConsumableStatus(
  GBaseObject   *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value Consumable/not consumable status
FALSE : not consumable
TRUE : consumable

Details Gets the input event consumable or input event not consumable status.

GCSSetCaptionStatus

Outline

Set caption display status

Syntax

```
void GCSSetCaptionStatus(
  GBaseObject  *pSelf,
  short        fStatus
)
```

Argument

(i) GBaseObject *pSelf		: object for operation
(i) short fStatus		: display/non-display status
	FALSE	: display
	TRUE	: non-display

Return value

None

Details

Sets the object caption display status.
Displaying or not displaying captions for the objects that have captions is switched according to this setting.

GCSGetCaptionStatus

Outline Get caption display status

Syntax
```
short GCSGetCaptionStatus(
  GBaseObject  *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value Caption display status
FALSE : display
TRUE : non-display

Details Gets the object caption display status.

GCSSetFocusStatus

Outline Set focus status

Syntax
```
void GCSSetFocusStatus(
  GBaseObject  *pSelf,
  short        fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : focused/not focused status
FALSE : not focused
TRUE : focused

Return value None

Details Sets the object focus status for the object.

GCSGetFocusStatus

Outline Get focus status

Syntax
```
short GCSGetFocusStatus(
  GBaseObject  *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
Focus status
FALSE : not focused
TRUE : focused

Details Gets the object focus status for the object.

GCSSetPanelStatus

Outline Set management panel object status

Syntax
```
void GCSSetPanelStatus(
  GBaseObject  *pSelf,
  short        fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : management panel status
FALSE : other than management panel object
TRUE : management panel object

Return value None

Details Sets the object as a management panel object or not.

GCSGetPanelStatus

Outline Get management panel object status

Syntax
```
short GCSGetPanelStatus(
  GBaseObject  *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
Management panel status
FALSE : other than management panel object
TRUE : management panel object

Details Gets the object focus status for the object.

GCSSetStyleStatus

Outline Set style

Syntax
```
void GCSSetStyleStatus(
  GBaseObject     *pSelf,
  unsigned long   ulStyle,
  short           fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStyle : style to be set
(i) short fStatus : specified style status

Return value None

Details Sets the object style.
With this method, it is necessary to redraw when redrawing is necessary to make this setting.

GCSGetStyleStatus

Outline Get style

Syntax
```
short GCSGetStyleStatus(
  GBaseObject     *pSelf,
  unsigned long   ulStyle
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStyle : style being acquired

Return value Specified style status (depends on style being acquired)

Details Gets the specified style status for the object.

GCSSetParent

Outline	Set parent object
Syntax	void GCSSetParent(GBaseObject *pSelf, GBaseObject *pParent)
Argument	(i) GBaseObject *pSelf : object for operation (i) GBaseObject *pParent : parent object being set
Return value	None
Details	Sets the parent object for managing the object.

GCSGetParent

Outline	Get parent object
Syntax	GBaseObject* GCSGetParent(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	NULL : no parent object Other than NULL : parent object
Details	Gets the parent object for managing the object.

GCSImportProperty

Outline Import property settings

Syntax
```
short GCSImportProperty(
  GBaseObject                 *pSelf,
  const GBaseWindowProperty   *pImport
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) const GBaseWindowProperty *pImport : property structure

Return value
FALSE : acquisition failed
TRUE : acquisition succeeded

Details Imports the property setting as a setting object.

GCSExportProperty

Outline Export property settings

Syntax
```
void GCSExportProperty(
  GBaseObject             *pSelf,
  GBaseWindowProperty     * pExport
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(o) GBaseWindowProperty *pExport : property structure

Return value None

Details Exports the object setting to a property structure.

GCSCreate

Outline Create process

Syntax
```
short GCSCreate(
  GBaseObject  *pSelf,
  GBaseObject  *pParent
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pParent : management object

Return value
FALSE : creation failed
TRUE : creation succeeded

Details Creates an object.

GCSCreateVram

Outline Create process (create VRAM)

Syntax
```
short GCSCreateVram(
  GBaseObject  *pScreen,
  GBaseObject  *pParent,
  void         *pData
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pParent : management object
(i) void *pData : additional information when drawing handle is created

Return value
FALSE : creation failed
TRUE : creation succeeded

Details Creates an object.
In the process for creating a GCScreen object, a drawing handle and drawing context are created.

GCSDelete

Outline Delete process

Syntax
```
void GCSDelete(
  GBaseObject   *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value None

Details Deletes an object.

GCSPaint

Outline Drawing process

Syntax
```
void GCSPaint(
  GBaseObject   *pSelf,
  HWDC          hWDC
)
```

Argument (i) GBaseObject *pSelf : object for operation
(i) HWDC hWDC : drawing context

Return value None

Details Carries out drawing process for object GM_PAINT message.
Since there is no special distinction between the client area and non-client area with the GBaseObject class, only client area drawing is called.

GCSOnDraw

Outline Client area drawing process

Syntax
```
void GCSOnDraw(
  GBaseObject  *pSelf,
  HWDC         hWDC
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) HWDC hWDC : drawing context

Return value None

Details Carries out drawing process for object client area.
Since there is no special distinction between the client area and non-client area with the GBaseObject class, calling is only for the client area.

GCSOnNcDraw

Outline Non-client area drawing process

Syntax
```
void GCSOnNcDraw(
  GBaseObject  *pSelf,
  HWDC         hWDC
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) HWDC hWDC : drawing context

Return value None

Details Carries out drawing process for object non-client area.
Since there is no special distinction between the client area and non-client area with the GBaseObject class, non-client area is not called.

GCSAddRefreshRect

Outline Client area redraw area registration

Syntax

```
void GCSAddRefreshRect(
  GBaseObject   *pSelf,
  GRect         *pRefreshRect,
  short         fErase
)
```

Argument

(i) GBaseObject *pSelf		: object for operation
(i) GRect *pRefreshRect		: area for redrawing (client coordinates)
(i) short fErase		: background drawing present/none
	GW_REFRESH_BGERASE	: background drawing
	GW_REFRESH_BGNOERASE	: no background drawing

Return value None

Details Registers the redrawing area in object client coordinates. Background redrawing present/none is selected using fErase.

GCSLButtonPress

Outline

Process for pressing left mouse button

Syntax

```
long GCSLButtonPress(
  GBaseObject   *pSelf,
  unsigned long  ulStatus,
  GPoint         gptPoint
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value

FALSE : do not consume mouse event
TRUE : consume mouse event

Details

Carries out the processing for pressing the left mouse button for the object.
Calls the left mouse button press processing for the client area.
When the mouse event is not consumed, FALSE is returned, and when the mouse event is consumed, TRUE is returned.

GCSOnLButtonPress

Outline

Client area process for pressing left mouse button

Syntax

```
long GCSOnLButtonPress(
  GBaseObject   *pSelf,
  unsigned long  ulStatus,
  GPoint         gptPoint
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value

FALSE : do not consume mouse event
TRUE : consume mouse event

Details

Carries out the processing for pressing the left mouse button for the client area on the object.
When the mouse event is not consumed, FALSE is returned, and when the mouse event is consumed, TRUE is returned.

GCSLButtonRelease

Outline Process for releasing left mouse button

Syntax
```
void GCSLButtonRelease(
  GBaseObject   *pSelf,
  unsigned long  ulStatus,
  GPoint         gptPoint
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value None

Details Carries out the processing for releasing the left mouse button for the object.
Calls the left mouse button release processing for the client area.

GCSOnLButtonRelease

Outline Client area process for releasing left mouse button

Syntax
```
void GCSOnLButtonRelease(
  GBaseObject    *pSelf,
  unsigned long   ulStatus,
  GPoint          gptPoint
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value None

Details Carries out the processing for releasing the left mouse button for the client area on the object.

GCSMouseMove

Outline Pointing device movement process

Syntax
```
void GCSMouseMove(
  GBaseObject  *pSelf,
  long         ulStatus,
  GPoint       gptPoint
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value None

Details Carries out the processing for pointing device movement for the object. Calls the pointing device movement processing for the client area.

GCSOnMouseMove

Outline Pointing device client area movement process

Syntax
```
void GCSOnMouseMove(
  GBaseObject  *pSelf,
  long         ulStatus,
  GPoint       gptPoint
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value None

Details Carries out the processing for pointing device movement for the object for the client area.

GCSOnNcMouseMove

Outline

Pointing device non-client area movement process

Syntax

```
void GCSOnNcMouseMove(
  GBaseObject    *pSelf,
  long           ulStatus,
  GPoint         gptPoint
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) long ulStatus : pressed button status
(i) GPoint gptPoint : position of point device coordinates

Return value

None

Details

Carries out the processing for pointing device movement for the object for the non-client area.

GCSKeyPress

Outline Process for pressing key

Syntax
```
long GCSKeyPress(
  GBaseObject     *pSelf,
  unsigned short  usKeyCode,
  unsigned long   ulStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short usKeyCode : virtual key code
(i) unsigned long ulStatus : key status

Return value
FALSE : do not consume key event
TRUE : consume key event

Details Carries out the processing for pressing a key for the object.
When the key event is not consumed, FALSE is returned, and when the key event is consumed, TRUE is returned.

GCSKeyRelease

Outline Process for releasing key

Syntax
```
void GCSKeyRelease(
  GBaseObject     *pSelf,
  unsigned short  usKeyCode,
  unsigned long   ulStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short usKeyCode : virtual key code
(i) unsigned long ulStatus : key status

Return value None

Details Carries out the processing for releasing a key for the object.

GCSSetFocus

Outline	Focus setting process
Syntax	void GCSSetFocus(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	None
Details	Carries out the processing for setting the focus for the object.

GCSKillFocus

Outline	Focus removing process
Syntax	void GCSKillFocus(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	None
Details	Carries out the processing for removing the focus for the object.

GCSTimer

Outline	Timer process
Syntax	void GCSTimer(GBaseObject **pSelf,* unsigned short *usTimerID*)
Argument	(i) GBaseObject *pSelf : object for operation (i) unsigned short usTimerID : timer ID
Return value	None
Details	Carries out the timer processing for the object.

GCSUser

Outline	User process
Syntax	long GCSUser(GBaseObject **pSelf,* long *lLParam,* long *lUParam*)
Argument	(i) GBaseObject *pSelf : object for operation (i) long lLParam : appended information (lower-side) (i) long lUParam : appended information (upper-side)
Return value	-1 : not supported 0 or more : arbitrary
Details	Carries out the user processing for the object. Null processing.

GCSSetFocusInformation

Outline Set focus movement information

Syntax
```
void GCSSetFocusInformation(
  GBaseObject          *pSelf,
  GFocusInformation    *pFocusInfo
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GFocusInformation short *pFocusInfo : focus movement information

Return value None

Details Set focus movement information for the object.
In the case of a virtual key code where the object is not received in a key operation for an object, focus movement follows this setting.

GCSGetFocusInformation

Outline Get focus movement information

Syntax
```
GFocusInformation *GCSGetFocusInformation(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Pointer to focus movement information

Details Gets focus movement information for the object.
In the case of a virtual key code where the object is not received in a key operation for an object, focus movement follows this setting.

GCSChar

Outline Character input process

Syntax
```
void GCSChar(
  GBaseObject     *pSelf,
  unsigned short  usCharCode,
  unsigned long   ulStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
unsigned short usCharCode : character code
unsigned long ulStatus : status

Return value None

Details Carries out the character input processing for the object.
When one byte size of characters is set as the character code in usCharCode, processing is carried out for one-byte characters, and when two bytes size is set, processing is carried out two-byte characters.

GCSClose

Outline Issue object close process request

Syntax
```
void GCSClose(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value None

Details For a related parent object, a request for termination processing for the object is issued.

GCSDeleteChild	

Outline	Child object deletion process
Syntax	void GCSDeleteChild(GBaseObject *pSelf, GBaseObject *pChild)
Argument	(i) GBaseObject *pSelf : object for operation (i) GBaseObject *pChild : child object to be deleted
Return value	None
Details	Executes deletion processing for related child object.

GCSSetMemorySpace

Outline Set memory information

Syntax
```
void GCSSetMemorySpace(
  GBaseObject     *pSelf,
  unsigned char   ucType,
  unsigned char   ucPlane
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned char ucType : memory management information type
(i) unsigned char ucPlane : memory management information plane

Return value None

Details Sets the memory management information used by the object.

GCSGetMemorySpace

Outline Get memory information

Syntax
```
void GCSGetMemorySpace(
  GBaseObject     *pSelf,
  unsigned char   *pType,
  unsigned char   *pPlane
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(o) unsigned char *pType : memory management information type
(o) unsigned char *pPlane : memory management information plane

Return value None

Details Gets the memory management information used by the object.

GCSSetDrawForceStatus

Outline Set forced drawing status

Syntax
```
void GCSSetDrawForceStatus(
  GBaseObject   *pSelf,
  short         fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : forced drawing status
FALSE : no forced drawing
TRUE : forced drawing

Return value None

Details Sets the forced drawing status for the object.

GCSGetDrawForceStatus

Outline Get forced drawing status

Syntax
```
short GCSGetDrawForceStatus(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Forced drawing status
FALSE : no forced drawing
TRUE : forced drawing

Details Get object forced drawing status.

GCSSetRedrawStatus

Outline Set redraw status

Syntax
```
void GCSSetRedrawStatus(
  GBaseObject     *pSelf,
  short           fStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short fStatus : redraw status
FALSE : no redrawing
TRUE : redraw

Return value None

Details Sets the redrawing status for the object.

GCSGetRedrawStatus

Outline Get redraw status

Syntax
```
short GCSGetRedrawStatus(
  GBaseObject     *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Redraw status
FALSE : no redrawing
TRUE : redraw

Details Gets the redrawing status for the object.

GCSClearDrawStatus

Outline Clear drawing status

Syntax
```
void GCSClearDrawStatus(
  GBaseObject    *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value None

Details Clears the object drawing status.

GCSAddDirtyRect

Outline Client area redraw area registration (internal processing function)

Syntax
```
void GCSAddDirtyRect(
  GBaseObject  * pSelf,
  GRect        *pRefreshRect,
  short        fErase
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GRect *pRefreshRect : area for redrawing (client coordinates)
(i) short fErase : background drawing present/none
GW_REFRESH_BGERASE : background drawing
GW_REFRESH_BGNOERASE : no background drawing

Return value None

Details Registers the redrawing area in object client coordinates.
Background redrawing present/none is selected using fErase.

GCSSetPrevTimerObject

Outline	Set previous object on timer bidirectional list
Syntax	void GCSSetPrevTimerObject(GBaseObject **pSelf,* GBaseObject **pbwPrev*)
Argument	(i) GBaseObject *pSelf : object for operation (i) GBaseObject *pbwPrev : object being set
Return value	None
Details	Sets the previous list object on the timer bidirectional list held in the object.

GCSGetPrevTimerObject

Outline	Get previous object on timer bidirectional list
Syntax	GBaseObject* GCSGetPrevTimerObject(GBaseObject **pSelf*)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Previous object on timer bidirectional list NULL : do not set Other than NULL : pointer to previous object
Details	Gets the previous list object on the timer bidirectional list held in the object.

GCSSetNextTimerObject

Outline	Set next object on timer bidirectional list
Syntax	void GCSSetNextTimerObject(GBaseObject *pSelf, GBaseObject *pbwNext)
Argument	(i) GBaseObject *pSelf : object for operation (i) GBaseObject *pbwNext : object being set
Return value	None
Details	Sets the next list object on the timer bidirectional list held in the object.

GCSGetNextTimerObject

Outline	Get next object on timer bidirectional list
Syntax	GBaseObject* GCSGetNextTimerObject(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Next object on timer bidirectional list NULL : do not set Other than NULL : pointer to next object
Details	Gets the next list object on the timer bidirectional list held in the object.

GCSSetBlinkStatus

Outline Set blink status

Syntax
```
void GCSSetBlinkStatus(
  GBaseObject    *pSelf,
  unsigned char   ucBlinkStatus
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned char ucBlinkStatus : blink status
GCBLINK_BLINK_OFF : OFF status
GCBLINK_BLINK_ON : ON status

Return value None

Details Sets the object blinking status for the object.

GCSGetBlinkStatus

Outline Get blink status

Syntax
```
unsigned char GCSGetBlinkStatus(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value blink status
GCBLINK_BLINK_OFF : OFF status
GCBLINK_BLINK_ON : ON status

Details Gets the object blinking status for the object.

GCSGetParentPanel

Outline

Get management panel object

Syntax

```
void GCSGetParentPanel(
  GBaseObject    *pSelf,
  GBaseObject    **pPanel,
  GBaseObject    ** pWindow
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(o) GBaseObject *pPanel	:	management panel object
(o) GBaseObject *pWindow	:	management window object

Return value

None

Details

Gets the panel object for managing the object.
Gets the window object matched to the object in the window.

GCSSetPrevBlinkControl

Outline	Set previous object on blink bidirectional list
Syntax	void GCSSetPrevBlinkControl(GBaseObject **pSelf,* GBaseObject **pvPrev*)
Argument	(i) GBaseObject *pSelf : object for operation (i) GBaseObject *pvPrev : object being set
Return value	None
Details	Sets the previous list object on the blink bidirectional list held in the object.

GCSGetPrevBlinkControl

Outline	Get previous object on blink bidirectional list
Syntax	GBaseObject* GCSGetPrevBlinkControl(GBaseObject **pSelf*)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Previous object on blink bidirectional list NULL : do not set Other than NULL : pointer to previous object
Details	Gets the previous list object on the blink bidirectional list held in the object.

GCSSetNextBlinkControl

Outline Set next object on blink bidirectional list

Syntax
```
void GCSSetNextBlinkControl(
  GBaseObject  *pSelf,
  GBaseObject  *pvNext
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pvNext : object being set

Return value None

Details Sets the next list object on the blink bidirectional list held in the object.

GCSGetNextBlinkControl

Outline Get next object on blink bidirectional list

Syntax
```
GBaseObject* GCSGetNextBlinkControl(
  GBaseObject  *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
Next object on blink bidirectional list
NULL : do not set
Other than NULL : pointer to next object

Details Gets the next list object on the blink bidirectional list held in the object.

GCSSetPrevScrollControl

Outline Set previous object on scroll bidirectional list

Syntax
```
void GCSSetPrevScrollControl(
  GBaseObject    *pSelf,
  GBaseObject    *pvPrev
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pbPrev : object being set

Return value None

Details Sets the previous list object on the scroll bidirectional list held in the object.

GCSGetPrevScrollControl

Outline Get previous object on scroll bidirectional list

Syntax
```
GBaseObject* GCSGetPrevScrollControl(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
previous object on scroll bidirectional list
NULL : do not set
Other than NULL : pointer to previous object

Details Gets the previous list object on the scroll bidirectional list held in the object.

GCSSetNextScrollControl

Outline Set next object on scroll bidirectional list

Syntax
```
void GCSSetNextScrollControl(
  GBaseObject      * pSelf,
  GBaseObject      * pvNext
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pvNext : object being set

Return value None

Details Sets the next list object on the scroll bidirectional list held in the object.

GCSGetNextScrollControl

Outline Get next object on scroll bidirectional list

Syntax
```
GBaseObject* GCSGetNextScrollControl(
  GBaseObject      *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
Next object on scroll bidirectional list
NULL : do not set
Other than NULL : pointer to next object

Details Gets the next list object on the scroll bidirectional list held in the object.

GCSLoadData

Outline

Load interpreter screen file

Syntax

```
int GCSLoadData(
  GBaseObject              *pSelf,
  HGIO                     fp,
  GBaseWindowProperty      *ptPropery,
  short                    fImport
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) HGIO fp	: file descriptor
(i/o)GBaseWindowProperty *ptProperty	: area storing the read property
(i) short fImport	: indicates whether to import or not

Return value

0	: normal end
Other than 0	: error

Details

Reads the screen data from current position in interpreter screen file.

GCSSetHWDCEnvironment

Outline Set drawing environment

Syntax
```
void GCSSetHWDCEnvironment(
  GBaseObject    *pSelf,
  HWDC           hWDC
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i/o) HWDC hWDC : drawing handle

Return value None

Details Sets drawing environment to match object.

GCSGetScreen

Outline Get screen object

Syntax
```
GBaseObject* GCSGetScreen(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Pointer to screen object

Details Gets screen object belonging to object.

GCSSetGBaseObject

Outline Set object management structure

Syntax
```
void GCSSetGBaseObject(
  GBaseObject     *pSelf,
  GBaseObject     *pBaseObject
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pBaseObject : object management structure

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the management structure for the object being managed.

GCSGetGBaseObject

Outline Get object management structure

Syntax
```
GBaseObject* GCSGetGBaseObject(
  GBaseObject     *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
NULL : management structure does not exist
Other than NULL : pointer to management structure

Details Gets the management structure for the object being managed.

GCSFreeLoadData

Outline

Release memory acquired by LoadData

Syntax

```
int GCSFreeLoadData(
  GBaseObject          *pSelf,
  GBaseWindowProperty  *ptProperty
)
```

Argument

(i) GBaseObject *pSelf		: object for operation
(i) GBaseWindowProperty *ptProperty		: property

Return value

0	: normal end
Other than 0	: error

Details

Releases the memory acquired by LoadData.

GCSSetModalStatus

Outline Set modal status

Syntax
```
void GCSSetModalStatus(
  GBaseObject   *pWindow,
  short         fStatus
)
```

Argument
(i) GBaseObject *pWindow : object for operation
(i) short fStatus : window modal status
FALSE : none
TRUE : present

Return value None

Details Sets the modal window status.

GCSGetModalStatus

Outline Get modal window status

Syntax
```
short GCSGetModalStatus(
  GBaseObject   *pWindow
)
```

Argument (i) GBaseObject *pWindow : object for operation

Return value
FALSE : none
TRUE : present

Details Gets the modal window status.

GCSGetRetValue

Outline	Get modal window return value
Syntax	short GCSGetRetValue(GBaseObject *pWindow)
Argument	(i) GBaseObject *pWindow : object for operation
Return value	Modal window return value
Details	Gets the return value returned when the modal window closes.

GCSBeginModal

Outline	Start process for modal window
Syntax	void GCSBeginModal(GBaseObject *pWindow)
Argument	(i) GBaseObject *pWindow : object for operation
Return value	None
Details	Gives notice of the modal start for a window.

GCSEndModal

Outline	End process for modal window
Syntax	void GCSEndModal(GBaseObject *pWindow, short nRetValue)
Argument	(i) GBaseObject *pWindow : object for operation (i) short nRetValue : return value when modal window is closed
Return value	None
Details	Gives notice of the modal end for a window.

GCSSetTitleBarStatus

Outline Set title bar present/none status

Syntax
```
void GCSSetTitleBarStatus(
  GBaseObject   *pWindow,
  short         fStatus
)
```

Argument
(i) GBaseObject *pWindow : object for operation
(i) short fStatus : title bar present/none status
FALSE : none
TRUE : present

Return value None

Details Sets the window title bar present or none.
When switching on/off, the object area is registered as a redraw area.

GCSGetTitleBarStatus

Outline Get title bar present/none status

Syntax
```
short GCSGetTitleBarStatus(
  GBaseObject   *pWindow
)
```

Argument (i) GBaseObject *pWindow : object for operation

Return value
FALSE : none
TRUE : present

Details Gets the title bar present/none status.

GCSSetCloseButtonStatus

Outline Set close button present/none status

Syntax
```
void GCSSetCloseButtonStatus(
  GBaseObject   *pWindow,
  short         fStatus
)
```

Argument
(i) GBaseObject *pWindow : object for operation
(i) short fStatus : close button present/none status
FALSE : none
TRUE : present

Return value None

Details Sets the window close button present or none.
When switching present or none, the object area is registered as a redraw area.

GCSGetCloseButtonStatus

Outline Get close button present/none status

Syntax
```
short GCSGetCloseButtonStatus(
  GBaseObject   *pWindow
)
```

Argument (i) GBaseObject *pWindow : object for operation

Return value
FALSE : none
TRUE : present

Details Gets the close button present/none status.

GCSSetEdgeStatus

Outline Set window frame present/none status

Syntax
```
void GCSSetEdgeStatus(
  GBaseObject   *pWindow,
  short         fStatus
)
```

Argument

(i) GBaseObject *pWindow		: object for operation
(i) short fStatus		: window frame present/none status
	FALSE	: none
	TRUE	: present

Return value None

Details Sets the window frame present or none.
When switching present or none, the object area is registered as a redraw area.

GCSGetEdgeStatus

Outline Get window frame present/none status

Syntax
```
short GCSGetEdgeStatus(
  GBaseObject   *pWindow
)
```

Argument (i) GBaseObject *pWindow : object for operation

Return value
FALSE : none
TRUE : present

Details Gets the window frame present/none status.

GCSAddChild

Outline	Add child object
Syntax	short GCSAddChild(GBaseObject *pPanel, GBaseObject *pChild)
Argument	(i) GBaseObject *pPanel : object for operation (i) GBaseObject *pChild : child object to be added
Return value	FALSE : addition failed TRUE : addition succeeded
Details	The child object specified by pChild is added to the child objects managed by the panel.

GCSGetChildList

Outline	Get child object list
Syntax	HGBWL GCSGetChildList(GBaseObject *pPanel)
Argument	(i) GBaseObject *pPanel : object for operation
Return value	Child object list handle
Details	Gets child object list handle.

GCSSetBackGroundDesign

Outline	Set background design
Syntax	void GCSSetBackGroundDesign(GBaseObject *pPanel, GDesign *pDesign)
Argument	(i) GBaseObject *pPanel : object for operation (i) GDesign* pDesign : background design
Return value	None
Details	Sets background design. After the background design is set, the panel area is registered as a redraw area.

GCSGetBackGroundDesign

Outline	Get background design
Syntax	GDesign* GCSGetBackGroundDesign(GBaseObject *pPanel)
Argument	(i) GBaseObject *pPanel : object for operation
Return value	Pointer to background design
Details	Gets pointer for background design.

GCSCreateChildren

Outline Process for creating child object

Syntax
```
short GCSCreateChildren(
  GBaseObject  *pPanel
)
```

Argument (i) GBaseObject *pPanel : object for operation

Return value FALSE : creation failed
TRUE : creation succeeded

Details Creates child object managed by GCPanel object. No processing by GCPanel. Actual processing creates GCPanel in a derivative class.

GCSDeleteAllChildren

Outline Delete all child objects

Syntax
```
void GCSDeleteAllChildren(
  GBaseObject  *pPanel
)
```

Argument (i) GBaseObject *pPanel : object for operation

Return value None

Details Deletes all child objects.

GCSGetChild

Outline	Search child object
Syntax	GBaseObject *GCSGetChild(GBaseObject *pPanel, unsigned short usID)
Argument	(i) GBaseObject *pPanel : object for operation (i) unsigned short usID : ID for child object being searched
Return value	NULL : child object with specified ID was not searched. Other than NULL : pointer to child object with specified ID
Details	Searches for a GCPanel child object having the specified ID.

GCSGetChildAll

Outline	Search child object search in frame
Syntax	GBaseObject *GCSGetChildAll(GBaseObject *pPanel, unsigned short usID)
Argument	(i) GBaseObject *pPanel : object for operation (i) unsigned short usID : ID for child object being searched
Return value	NULL : child object with specified ID was not searched. Other than NULL : pointer to child object with specified ID
Details	Searches for a GCPanel child object having the specified ID. When the child object is a frame, the search is also performed in the GCPanel for the page currently displayed in that frame.

GCSSetActiveChild

Outline

Set active object

Syntax

```
short GCSSetActiveChild(
  GBaseObject     *pPanel,
  short           nSearch,
  GBaseObject     *pActiveChild
)
```

Argument

(i) GBaseObject *pPanel : object for operation
(i) short nSearch : method for specifying object
(i) GBaseObject *pChild : active child object

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

When the panel status is active, the child object receiving the focus is set.
The setting methods are the method of directly specifying the object and the method of searching for the beginning and end of the child object list.
The following value is specified in nSearch:
GWINDOW_SEARCH_NON:

GCSGetActiveChild

Outline Get active object (only direct child object)

Syntax
```
GBaseObject* GCSGetActiveChild(
  GBaseObject   *pPanel
)
```

Argument (i) GBaseObject *pPanel : object for operation

Return value
NULL : no setting
Other than NULL : get active object

Details When the panel status is active, the child object receiving the focus is acquired.

GCSGetActiveControl

Outline Get active control (including terminal child objects)

Syntax
```
GBaseObject* GCSGetActiveControl(
  GBaseObject       *pPanel
)
```

Argument (i) GBaseObject *pPanel : object for operation

Return value
NULL : no setting
Other than NULL : get active control

Details When the panel status is active, the child object receiving the control is acquired.

GCSSetBlinkOffInterval

Outline Set blinking OFF interval

Syntax
```
void GCSSetBlinkOffInterval(
  GBaseObject    *pPanel,
  long           lInterval
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) long lInterval : blinking OFF interval

Return value None

Details Sets the interval for blinking OFF.

GCSGetBlinkOffInterval

Outline Get blinking OFF interval

Syntax
```
long GCSGetBlinkOffInterval(
  GBaseObject   *pPanel
)
```

Argument
(i) GBaseObject *pPanel : object for operation

Return value blinking OFF interval

Details Gets the interval for blinking OFF.

GCSSetBlinkOnInterval

Outline Set blinking ON interval

Syntax
```
void GCSSetBlinkOnInterval(
  GBaseObject    *pPanel,
  long           lInterval
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) long lInterval : blinking ON interval

Return value None

Details Sets the interval for blinking ON.

GCSGetBlinkOnInterval

Outline Get blinking ON interval

Syntax
```
long GCSGetBlinkOnInterval(
  GBaseObject  *pPanel
)
```

Argument
(i) GBaseObject *pPanel : object for operation

Return value Blinking ON interval

Details Gets the interval for blinking ON.

GCSAddBlinkControl

Outline Add caption blink control

Syntax
```
void GCSAddBlinkControl(
  GBaseObject  *pPanel,
  GBaseObject  *pControl
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) GBaseObject *pControl : control to be added

Return value None

Details Among the controls in the panel, the control for caption blinking is added to the list.

GCSRemoveBlinkControl

Outline Delete caption blink control

Syntax
```
void GCSRemoveBlinkControl(
  GBaseObject  *pPanel,
  GBaseObject  *pControl
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) GBaseObject *pControl : control to be deleted

Return value None

Details Among the controls in the panel, the control for terminating caption blink is deleted from the list.

GCSAddScrollControl

Outline Add caption scroll control

Syntax
```
void GCSAddScrollControl(
  GBaseObject  *pPanel,
  GBaseObject  *pControl
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) GBaseObject *pControl : control to be added

Return value None

Details Among the controls in the panel, the control for caption scrolling is added to the list.

GCSRemoveScrollControl

Outline Delete caption scroll control

Syntax
```
void GCSRemoveScrollControl(
  GBaseObject  *pPanel,
  GBaseObject  *pControl
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) GBaseObject *pControl : control to be deleted

Return value None

Details Among the controls in the panel, the control for terminating caption scroll is deleted from the list.

GCSDrawChildren

Outline Child object drawing process

Syntax
```
void GCSDrawChildren(
  GBaseObject  *pPanel,
  HWDC         pDC
)
```

Argument
(i) GBaseObject *pPanel : object for operation
(i) HWDC pDC : drawing context

Return value None

Details Carries out child object drawing process.
The drawing order is in order from the beginning of the list.

GCSAttachPanel

Outline Attach panel object

Syntax
```
GBaseObject *GCSAttachPanel(
  GBaseObject  *pScreen,
  GBaseObject  *pNewPanel
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) GBaseObject *pNewPanel : related GCPanel class

Return value Pointer to related panel object

Details Attaches panel object to screen/window.

GCSGetPanel	

Outline	Get attached GCPanel object
Syntax	GBaseObject *GCSGetPanel(GBaseObject *pScreen)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	Pointer to related panel
Details	Carries out acquisition of GCPanel object related to screen/window.

GCSSetSystemFontID

Outline Set system font resource ID

Syntax
```
void GCSSetSystemFontID(
  GBaseObject    *pScreen,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets system font resource ID.
After the system font resource ID is set, the entire screen is registered as a redraw area.

GCSGetSystemFontID

Outline Get system font resource ID

Syntax
```
unsigned short GCSGetSystemFontID(
  GBaseObject   *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value Font resource ID

Details Gets system font resource ID.

GCSSetSystemBorderID

Outline Set system 3D border resource ID

Syntax
```
void GCSSetSystemBorderID(
  GBaseObject    *pScreen,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned short usID : 3D border resource ID

Return value None

Details Sets system 3D border resource ID.
After the system 3D border resource ID is set, the entire screen is registered as a redraw area.

GCSGetSystemBorderID

Outline Get system 3D border resource ID

Syntax
```
unsigned short GCSGetSystemBorderID(
  GBaseObject  *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value 3D border resource ID

Details Gets system 3D border resource ID setting.

GCSSetSystemForeColor

Outline	Set system foreground color
Syntax	void GCSSetSystemForeColor(GBaseObject *pScreen, GColor gcColor)
Argument	(i) GBaseObject *pScreen : object for operation (i) GColor gcColor : foreground color
Return value	None
Details	Sets system foreground color. After the system foreground color is set, the entire screen is registered as a redraw area.

GCSGetSystemForeColor

Outline	Get system foreground color
Syntax	GColor GCSGetSystemForeColor(GBaseObject *pScreen)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	System foreground color
Details	Gets system foreground color.

GCSSetSystemBackColor

Outline Set system background color

Syntax
```
void GCSSetSystemBackColor(
  GBaseObject  *pScreen,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) GColor gcColor : background color

Return value None

Details Sets system background color.
After the system background color is set, the entire screen is registered as a redraw area.

GCSGetSystemBackColor

Outline Get system background color

Syntax
```
GColor GCSGetSystemBackColor(
  GBaseObject  *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value System background color

Details Gets system background color.

GCSSetSystemActiveColor

Outline	Set system active color
Syntax	void GCSSetSystemActiveColor(GBaseObject **pScreen,* GColor *gcColor*)
Argument	(i) GBaseObject *pScreen : object for operation (i) GColor gcColor : active color
Return value	None
Details	Sets system active color. After the system active color is set, the entire screen is registered as a redraw area.

GCSGetSystemActiveColor

Outline	Get system active color
Syntax	GColor GCSGetSystemActiveColor(GBaseObject **pScreen*)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	System active color
Details	Gets system active color.

GCSSetSystemInactiveColor

Outline Set system inactive color

Syntax
```
void GCSSetSystemInactiveColor(
  GBaseObject   *pScreen,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) GColor gcColor : inactive color

Return value None

Details Sets system inactive color.
After the system inactive color is set, the entire screen is registered as a redraw area.

GCSGetSystemInactiveColor

Outline Get system inactive color

Syntax
```
GColor GCSGetSystemInactiveColor(
  GBaseObject   *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value System inactive color

Details Gets system inactive color.

GCSGetFrame

Outline Get frame

Syntax
```
GBaseObject *GCSGetFrame(
  GBaseObject   *pScreen
)
```

Argument (i) GBaseObject *pScreen : object for operation

Return value Pointer to frame managed by screen

Details Gets pointer to frame being managed by the screen.

GCSAddWindow

Outline Add window

Syntax
```
short GCSAddWindow(
  GBaseObject   *pScreen,
  GBaseObject   *pWindow
)
```

Argument (i) GBaseObject *pScreen : object for operation
(i) GBaseObject *pWindow : window object to be added

Return value FALSE : addition failed
TRUE : addition succeeded

Details Add window to the frame on the screen.

GCSChangeFocus

Outline	Change focus
Syntax	void GCSChangeFocus(GBaseObject *pScreen, GBaseObject *pNewObject)
Argument	(i) GBaseObject *pScreen : object for operation (i) GBaseObject *pNewObject : object acquiring the focus
Return value	None
Details	Focus given to specified object.

GCSGetFocus

Outline	Get focus
Syntax	GBaseObject *GCSGetFocus(GBaseObject *pScreen)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	NULL : no focus Other than NULL : object having focus
Details	Gets the pointer to the object having the focus.

GCSDispatchMessage	

Outline	Event dispatch process
Syntax	long GCSDispatchMessage(GBaseObject *pScreen, GEventMessage *gemMsg)
Argument	(i) GBaseObject *pScreen : object for operation (i) GEventMessage *gemMsgt : event message
Return value	Supported event return values
Details	Calls a message procedure for each event according to the event.

GCSGetWDC

Outline	Get drawing context
Syntax	HWDC GCSGetWDC(GBaseObject *pScreen)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	Pointer to drawing context
Details	Gets drawing context being managed by the screen.

GCSGetHWDC

Outline	Get drawing context
Syntax	HWDC GCSGetHWDC(GBaseObject *pScreen)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	Pointer to drawing context
Details	Gets drawing context being managed by the screen.

GCSSetPanelPlane

Outline Set memory area panel plane management information

Syntax
```
void GCSSetPanelPlane(
  GBaseObject    *pScreen,
  unsigned char  ucIndex,
  unsigned char  ucFlag
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned char ucIndex : panel plane No.
(i) unsigned char ucFlag : usage status (FALSE: unused, TRUE: used)

Return value None

Details Sets usage status for the panel plane in the memory area.

GCSGetPanelPlane

Outline Get memory area panel plane management information

Syntax
```
unsigned char GCSGetPanelPlane(
  GBaseObject    *pScreen,
  unsigned char  ucIndex
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned char ucIndex : panel plane No.

Return value Usage status
FALSE : unused
TRUE : used

Details Gets usage status for the panel plane in the memory area.

GCSSetWindowPlane

Outline Set memory area window plane management information

Syntax
```
void GCSSetWindowPlane(
  GBaseObject    *pScreen,
  unsigned char  ucIndex,
  unsigned char  ucFlag
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned char ucIndex : window plane No.
(i) unsigned char ucFlag : usage status (FALSE: unused, TRUE: used)

Return value None

Details Sets usage status for the panel plane in the memory area.

GCSGetWindowPlane

Outline Get memory area window plane management information

Syntax
```
unsigned char GCSGetWindowPlane(
  GBaseObject    *pScreen,
  unsigned char  ucIndex
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned char ucIndex : panel plane No.

Return value Usage status
FALSE : unused
TRUE : used

Details Gets usage status for the window plane in the memory area.

GCSSetExitStatus

Outline Set exit information

Syntax
```
void GCSSetExitStatus(
  GBaseObject  *pScreen,
  short        fStatus
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) short fStatus : exit information (TRUE: exit)

Return value None

Details Sets the exit information for applications using the GUI library.

GCSGetExitStatus

Outline Get exit information

Syntax
```
short GCSGetExitStatus(
  GBaseObject  *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value Exit information
FALSE : in operation
TRUE : exit

Details Gets the exit information for applications using the GUI library.

GCSSetSystemStatus

Outline Set system information

Syntax
```
void GCSSetSystemStatus(
  GBaseObject   *pScreen,
  unsigned long  ulSystem,
  short          fStatus
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned long ulSystem : system flag
(i) short fStatus : status (0: FALSE, 1: TRUE)

Return value None

Details Sets system status specified by ulSystem.

GCSGetSystemStatus

Outline Get system information

Syntax
```
short GCSGetSystemStatus(
  GBaseObject   *pScreen,
  unsigned long  ulSystem
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned long ulSystem : system flag

Return value Status
0 : FALSE
1 : TRUE

Details Gets system status specified by ulSystem.

GCSDoModal

Outline Window modal display

Syntax

```
short GCSDoModal(
  GBaseObject   *pScreen,
  GBaseObject   *pWindow,
  long          lLParam,
  long          lUParam
)
```

Argument

(i) GBaseObject *pScreen	: object for operation
(i) GBaseObject *pWindow	: window object
(i) long lLParam	: additional information (lower-side)
(i) long lUParam	: additional information (upper-side)

Return value Modal window return value

Details Displays the window object specified by pWindow as a modal window.

GCSSetPaletteStatus

Outline Set system palette utilization status

Syntax
```
void GCSSetPaletteStatus(
  GBaseObject   *pScreen,
  short         nStatus
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) short nStatus : palette usage status

Return value None

Details Sets the usage status of the palette.
When FALSE is set, unused is the status.
When TRUE is set, used is the status.

GCSGetPaletteStatus

Outline Get system palette utilization status

Syntax
```
short GCSGetPaletteStatus(
  GBaseObject   *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value System palette usage status
FALSE : unused
TRUE : used

Details Gets the usage status of the palette.

GCSSetSystemPaletteID

Outline	Set system palette resource ID
Syntax	void GCSSetSystemPaletteID(GBaseObject **pScreen,* unsigned short *usID*)
Argument	(i) GBaseObject *pScreen : object for operation (i) unsigned short usID : palette resource ID
Return value	None
Details	Sets the palette resource ID used by the system palette.

GCSGetSystemPaletteID

Outline	Get system palette resource ID
Syntax	unsigned short GCSGetSystemPaletteID(GBaseObject **pScreen*)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	System palette resource ID
Details	Gets the palette resource ID used by the system palette.

GCSSetPaletteColors

Outline Set number of colors in system palette

Syntax
```
void GCSSetPaletteColors(
  GBaseObject      *pScreen,
  unsigned short   usPaletteColors
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned short usPaletteColors : number of colors used by system palette

Return value None

Details Sets the number of colors used by the system palette.

GCSGetPaletteColors

Outline Get number of colors in system palette

Syntax
```
unsigned short GCSGetPaletteColors(
  GBaseObject   *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value Number of colors used by system palette

Details Gets the number of colors used by the system palette.

GCSAddTimerObject

Outline Add object to system timer list

Syntax
```
void GCSAddTimerObject(
  GBaseObject   *pScreen,
  GBaseObject   *pObject
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) GBaseObject *pObject : object to be added

Return value None

Details Adds the object specified by pObject to the system timer list. When registered into this system timer list, the object's function is call at a constant frequency with usTimerID==GCSCREEN_BASE_TIMER_ID.

GCSRemoveTimerObject

Outline Delete object from system timer list

Syntax
```
void GCSRemoveTimerObject(
  GBaseObject   *pScreen,
  GBaseObject   *pObject
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) GBaseObject *pObject : object to be deleted

Return value None

Details Deletes the object specified by pObject from the system timer list.

GCSGetWindow

Outline	Get window
Syntax	GBaseObject* GCSGetWindow(GBaseObject *pScreen, nsigned short usID)
Argument	(i) GBaseObject *pScreen : object for operation (i) unsigned short usID : ID of window being acquired
Return value	NULL : window for specified ID does not exist Other than NULL : window object having specified ID
Details	Gets the window having the ID specified by usID.

GCSGetActiveWindow

Outline	Get active window
Syntax	GBaseObject* GCSGetActiveWindow(GBaseObject *pScreen)
Argument	(i) GBaseObject *pScreen : object for operation
Return value	NULL : active window does not exist Other than NULL : active window object
Details	Gets the window having the ID specified by usID.

GCSShowPanel

Outline Display panel switching process

Syntax
```
short GCSShowPanel(
  GBaseObject   *pScreen,
  long          lPage
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lPage : screen No. to be displayed

Return value
FALSE : switching failed
TRUE : switching succeeded

Details Switched to the page specifying the panel currently displayed in the object.

GCSCreateInstance

Outline Create instance process

Syntax
```
GBaseObject *GCSCreateInstance(
  GBaseObject                   *pScreen,
  unsigned short                usType,
  const GBaseWindowProperty     *pProperty,
  GBaseObject                   *pParent
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) unsigned short usType : object type
(i) const GBaseWindowProperty *pImport : object property
(i) GBaseObject *pParent : parent object

Return value
NULL : creation failed
Other than NULL : created object

Details Creates instance of object registered in the project.

GCSPageLogOperation

Outline Switch screen according to screen history

Syntax
```
short GCSPageLogOperation(
  GBaseObject   *pScreen,
  short         nOperation,
  short         nShowWindow
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) short nOperation : screen transition method
(i) short nShowWindow : window recovery status

Return value
FALSE : failed
TRUE : succeeded

Details Switches screens according to the screen history.

GCSDrawFrame

Outline Related panel drawing process

Syntax
```
void GCSDrawFrame(
  GBaseObject   *pScreen,
  HWDC          pDC
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) HWDC pDC : drawing context

Return value None

Details Carries out drawing process for frame object related to GCScreen object.

GCSDrawPanel	

Outline	Related GCPanel object drawing process
Syntax	void GCSDrawPanel(GBaseObject *pScreen, HWDC pDC)
Argument	(i) GBaseObject *pScreen : object for operation (i) HWDC pDC : drawing context
Return value	None
Details	Carries out drawing process for panel object related to GCScreen object.

GCSOnDrawWindow

Outline

Window drawing process

Syntax

```
void GCSOnDrawWindow(
  GBaseObject   *pScreen,
  HWDC          pDC,
  long          lStart,
  long          lEnd,
  long          lNow,
  GBaseObject   **pWinArray,
  GRect         *pClip
)
```

Argument

(i) GBaseObject *pScreen	: object for operation
(i) HWDC pDC	: drawing context
(i) long lStart	: depth of starting window
(i) long lEnd	: depth of ending window
(i) long lNow	: depth of current window
(i) GBaseObject **pWinArray	: drawing window array
(i) GRect *pClip	: drawing area

Return value

None

Details

Carries out drawing process for object client area.
Calls related GCPanel object drawing process in GCScreen object.

GCSSetSystemPalette

Outline Set system palette

Syntax
```
void GCSSetSystemPalette(
  GBaseObject   *pScreen
)
```

Argument (i) GBaseObject *pScreen : object for operation

Return value None

Details Carries out system palette setup according to the system palette settings. When no palette is used, ends without making palette settings. When the number of colors differs in the palette resource and system palette settings, the lower number of colors is set.

GCSDrawBackGround

Outline Background drawing process

Syntax
```
void GCSDrawBackGround(
  GBaseObject   *pScreen,
  HWDC          pDC
)
```

Argument (i) GBaseObject *pScreen : object for operation
(i) HWDC pDC : drawing context

Return value None

Details Carries out drawing of GCScreen object background.

GCSCreateGWindow

Outline Create window

Syntax
```
short GCSCreateGWindow(
  GBaseObject   *pScreen,
  long          lPage
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lPage : display specified window

Return value
FALSE : display failed
TRUE : display succeeded

Details Creates a window and displays it.

GCSSetScreenManager

Outline Set screen manager

Syntax
```
void GCSSetScreenManager(
  GBaseObject   *pScreen,
  GBaseObject   *pScreenManager
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) GBaseObject *pScreenManager : screen manager object

Return value None

Details Sets up screen manager in screen objects for each module.

GCSSetModuleID	

Outline	Set module ID
Syntax	void GCSSetModuleID(GBaseObject **pScreen,* unsigned long *ulModuleID*)
Argument	(i) GBaseObject *pScreen : object for operation (i) unsigned long ulModuleID : module ID
Return value	None
Details	Sets up ID in screen objects for each module.

GCSClearPageLog

Outline Clear screen history

Syntax
```
void GCSClearPageLog(
  GBaseObject   *pScreen
)
```

Argument (i) GBaseObject *pScreen : object for operation

Return value None

Details Clears the screen history.

GCSAddPageLog

Outline Add to screen history

Syntax
```
short GCSAddPageLog(
  GBaseObject   *pScreen,
  long          lPage
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lPage : screen No. to be added

Return value
TRUE : addition succeeded
FALSE : addition failed

Details Adds to the screen history.

GCSPrevPage

Outline

Issue request for displaying previous screen

Syntax

```
short GCSPrevPage(
  GBaseObject   *pScreen,
  short         nShowWindow
)
```

Argument

(i) GBaseObject *pScreen : object for operation
(i) short nShowWindow : restore window status
FALSE : do not restore
TRUE : restore

Return value

TRUE : succeeded
FALSE : failed

Details

Issues a request to display the screen displayed previously based on the screen history information.
Note that if the previous screen is the standard screen, setting TRUE for nShowWindow does not restore the window.

GCSNextPage

Outline

Issue request for displaying next screen

Syntax

```
short GCSNextPage(
  GBaseObject   *pScreen,
  short         nShowWindow
)
```

Argument

(i) GBaseObject *pScreen		: object for operation
(i) short nShowWindow		: restore window status
	FALSE	: do not restore
	TRUE	: restore

Return value

TRUE : succeeded
FALSE : failed

Details

Issues a request to display the screen displayed afterward based on the screen history information.

GCSSetPageOffset

Outline Set module page offset

Syntax
```
void GCSSetPageOffset(
  GBaseObject   *pScreen,
  long          lOffset
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lOffset : page offset

Return value None

Details Sets the page offset for the module to which the screen belongs.

GCSGetPageOffset

Outline Get module page offset

Syntax
```
long GCSGetPageOffset(
  GBaseObject   *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value Page offset

Details Gets the page offset for the module to which the screen belongs.

GCSGetOperationLog

Outline

Get operation log

Syntax

```
short GCSGetOperationLog(
  GBaseObject            *pScreen,
  unsigned short         usIndex,
  GOperationInformation  *pLog
)
```

Argument

(i) GBaseObject *pScreen	:	object for operation
(i) unsigned short usIndex	:	index No.
(i) GOperationInformation *pLog	:	variable storing operation log being acquired

Return value

FALSE	:	failed
TRUE	:	succeeded

Details

Gets the operation log in the position specified by usIndex.

GCSClearOperationLog

Outline Clear operation log

Syntax
```
short GCSClearOperationLog(
  GBaseObject   *pScreen
)
```

Argument (i) GBaseObject *pScreen : object for operation

Return value FALSE : failed
TRUE : succeeded

Details Clears the content of the operation log.

GCSAddOperationLog

Outline Add operation log (operation event)

Syntax
```
short GCSAddOperationLog(
  GBaseObject     *pScreen,
  long            lPage,
  unsigned short  usID,
  unsigned short  usMessage
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lPage : screen No.
(i) unsigned short usID : object ID
(i) unsigned short usMessage : event

Return value FALSE : failed
TRUE : succeeded

Details Adds the operation log with the specified log content.

GCSAddChangePageLog

Outline Add operation log (screen switching)

Syntax
```
short GCSAddChangePageLog(
  GBaseObject  *pScreen,
  long         lOldPage,
  long         lNewPage
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lOldPage : screen No. prior to screen switching
(i) long lNewPage : screen No. following screen switching

Return value
FALSE : failed
TRUE : succeeded

Details Adds the operation log with the specified log content.

GCSGetPageNumber

Outline Get screen No.

Syntax
```
long GCSGetPageNumber(
  GBaseObject  *pScreen
)
```

Argument
(i) GBaseObject *pScreen : object for operation

Return value Currently displayed screen No.

Details Returns the currently displayed screen No.

GCSLoadProjectFile

Outline Load interpreter project file

Syntax
```
int GCSLoadProjectFile(
  GBaseObject   *pIScreen,
  GTCHAR        *ptszProjectPath,
  short         fShowStartPanel
)
```

Argument
(i) GBaseObject *pIScreen; : object for operation
(i) GTCHAR *ptszProjectPath : project file to be loaded
(i) short fShowStartPanel : display/non-display of initial display page
FALSE : non-display
TRUE : display

Return value
0 : normal end
Other than 0 : error

Details Reads project data from the interpreter project file.

GCSLoadGWindow

Outline Window loading process

Syntax
```
GBaseObject* GCSLoadGWindow(
  GBaseObject   *pScreen,
  long          lPage
)
```

Argument
(i) GBaseObject *pScreen : object for operation
(i) long lPage : window No. to load

Return value
NULL : load failed
Other than NULL : load succeeded

Details Loads the window with the specified No.

GCSModalGWindow

Outline	Modal window process
Syntax	short GCSModalGWindow(GBaseObject *pScreen, long lPage)
Argument	(i) GBaseObject *pScreen : object for operation (i) long lPage : modal window No.
Return value	Modal window return value
Details	Displays the window with the number specified as a modal window.

GCSSetBlinkType

Outline Set blink type

Syntax
```
void GCSSetBlinkType(
  GBaseObject     *pControl,
  unsigned char   ucBlinkType
)
```

Argument
(i) GBaseObject *pControl : object for operation
(i) unsigned char ucBlinkType
- GCBLINK_CHANGE_VISIBLE_STRING : display/non-display of character string
- GCBLINK_CHANGE_STRING_COLOR : change character color
- GCBLINK_CHANGE_WHOLE_COLOR : change whole color

Return value None

Details Sets the object blink type.

GCSGetBlinkType

Outline Get blink type

Syntax
```
unsigned char GCSGetBlinkType(
  GBaseObject   *pControl
)
```

Argument (i) GBaseObject *pControl : object for operation

Return value Blink type:
- GCBLINK_CHANGE_VISIBLE_STRING : display/non-display of character string
- GCBLINK_CHANGE_STRING_COLOR : change character color
- GCBLINK_CHANGE_WHOLE_COLOR : change whole color

Details Gets the object blink type.

GCSSetBlinkEnableStatus

Outline Set blink operation

Syntax
```
void GCSSetBlinkEnableStatus(
  GBaseObject   *pControl,
  unsigned char  ucEnable
)
```

Argument

(i) GBaseObject *pControl		: object for operation
(i) unsigned char ucEnable		: blink operation setting
	GCBLINK_BLINK_DISABLE	: disable
	GCBLINK_BLINK_ENABLE	: enable

Return value None

Details Sets the object blink operation.

GCSGetBlinkEnableStatus

Outline Get blink operation

Syntax
```
unsigned char GCSGetBlinkEnableStatus(
  GBaseObject   *pControl
)
```

Argument (i) GBaseObject *pControl : object for operation

Return value Setup of blink operation
FALSE : disable
TRUE : enable

Details Gets the object blink operation.

GCSStartCaptionBlink	
Outline	Start blink operation
Syntax	void GCSStartCaptionBlink(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Starts the object blink operation.

GCSStopCaptionBlink	
Outline	Stop blink operation
Syntax	void GCSStopCaptionBlink(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Stops the object blink operation.

GCSSetCaptionScrollEnable

Outline Set scroll operation

Syntax
```
void GCSSetCaptionScrollEnable(
  GBaseObject   *pControl,
  unsigned char  ucEnable
)
```

Argument

(i) GBaseObject *pControl		: object for operation
(i) unsigned char ucEnable		: scroll operation setting
	GCSCROLL_SCROLL_DISABLE	: disable
	GCSCROLL_SCROLL_ENABLE	: enable
	GCSCROLL_SCROLL_OUTOFBOUNDS	: enable when out of bounds

Return value None

Details Sets the object scroll operation.

GCSGetCaptionScrollEnable

Outline Get scroll operation

Syntax
```
unsigned char GCSGetCaptionScrollEnable(
  GBaseObject   *pControl
)
```

Argument (i) GBaseObject *pControl : object for operation

Return value Scroll operation

GCSCROLL_SCROLL_DISABLE	: disable
GCSCROLL_SCROLL_ENABLE	: enable
GCSCROLL_SCROLL_OUTOFBOUNDS	: enable when out of bounds

Details Gets the object scroll operation.

GCSStartCaptionScroll

Outline	Start scroll operation
Syntax	void GCSStartCaptionScroll(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Starts the object scroll operation. The scroll operation must be set in GCSCROLL_SCROLL_ENABLE or GCSCROLL_SCROLL_OUTOFBOUNDS in advance.

GCSStopCaptionScroll

Outline	Stop scroll operation
Syntax	void GCSStopCaptionScroll(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Stops the object scroll operation.

GCSPauseCaptionScroll	
Outline	Pause scroll operation
Syntax	void GCSPauseCaptionScroll(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Pauses the object scroll operation.

GCSRestartCaptionScroll	
Outline	Restart scroll operation
Syntax	void GCSRestartCaptionScroll(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Restart the object scroll operation.

GCSSetCaptionScrollStatus

Outline Set scroll operation status

Syntax
```
void GCSSetCaptionScrollStatus(
  GBaseObject     *pControl,
  unsigned char   ucStatus
)
```

Argument
(i) GBaseObject *pControl : object for operation
(i) unsigned char ucStatus : scroll operation status

Return value None

Details Sets the object scroll operation status.

GCSGetCaptionScrollStatus

Outline Get scroll operation status

Syntax
```
unsigned char GCSGetCaptionScrollStatus(
  GBaseObject   *pControl
)
```

Argument
(i) GBaseObject *pControl : object for operation

Return value Scroll operation status

Details Gets the object scroll operation status.

GCSIsCaptionOutOfBounds

Outline Confirm overflowing of caption character string

Syntax
```
short GCSIsCaptionOutOfBounds(
  GBaseObject  *pControl
)
```

Argument (i) GBaseObject *pControl : object for operation

Return value Out of bounds status
FALSE : not out of bounds
TRUE : out of bounds

Details Confirms whether or not the caption character string set in the control goes out of the bounds of the control caption display area.

GCSSetCaptionScrollDelayTime

Outline	Set scroll delay time
Syntax	void GCSSetCaptionScrollDelayTime(GBaseObject **pControl,* unsigned long *lTime*)
Argument	(i) GBaseObject *pControl : object for operation (i) unsigned long ulTime : delay time setting
Return value	None
Details	Sets the delay time for the object scroll starting time.

GCSGetCaptionScrollDelayTime

Outline	Get scroll delay time
Syntax	unsigned long GCSGetCaptionScrollDelayTime(GBaseObject **pControl*)
Argument	(i) GBaseObject *pControl : object for operation
Return value	Delay time
Details	Gets the delay time for the object scroll starting time.

GCSSetCaptionScrollRefreshTime

Outline	Set scroll refresh time
Syntax	void GCSSetCaptionScrollRefreshTime(GBaseObject **pControl,* unsigned long *lTime*)
Argument	(i) GBaseObject *pControl : object for operation (i) unsigned long ulTime : refresh time setting
Return value	None
Details	Sets the object scroll refresh time.

GCSGetCaptionScrollRefreshTime

Outline	Get scroll refresh time
Syntax	unsigned long GCSGetCaptionScrollRefreshTime(GBaseObject **pControl*)
Argument	(i) GBaseObject *pControl : object for operation
Return value	Refresh time
Details	Gets the object scroll refresh time.

GCSSetCaptionScrollMovementValue

Outline	Set scroll movement value
Syntax	void GCSSetCaptionScrollMovementValue(GBaseObject **pControl,* unsigned short *usMovementValue*)
Argument	(i) GBaseObject *pControl : object for operation (i) unsigned short usMovementValue : movement value setting
Return value	None
Details	Sets the object scroll movement value.

GCSGetCaptionScrollMovementValue

Outline	Get scroll movement value
Syntax	unsigned short GCSGetCaptionScrollMovementValue(GBaseObject **pControl*)
Argument	(i) GBaseObject *pControl : object for operation
Return value	Movement value
Details	Gets the object scroll movement value.

GCSSetCaptionScrollStartPosition

Outline Set scroll start position

Syntax
```
void GCSSetCaptionScrollStartPosition(
  GBaseObject     *pControl,
  unsigned char   ucStartPosition
)
```

Argument
(i) GBaseObject *pControl : object for operation
(i) unsigned char ucStartPosition : start position setting
GCSCROLL_START_NOW : follow current caption setting
GCSCROLL_START_RIGHT : from right

Return value None

Details Sets the object scroll start position.

GCSGetCaptionScrollStartPosition

Outline Get scroll start position

Syntax
```
unsigned char GCSGetCaptionScrollStartPosition(
  GBaseObject   *pControl
)
```

Argument
(i) GBaseObject *pControl : object for operation

Return value Start position
GCSCROLL_START_NOW : follow current caption setting
GCSCROLL_START_RIGHT : from right

Details Gets the object scroll start position.

GCSSetCaptionScrollPosition

Outline	Set total scroll movement value
Syntax	void GCSSetCaptionScrollPosition(GBaseObject *pControl, long lPosition)
Argument	(i) GBaseObject *pControl : object for operation (i) unsigned short usMovementValue : total movement value setting
Return value	None
Details	Sets the total scroll movement value from the start position for the object.

GCSGetCaptionScrollPosition

Outline	Get total scroll movement value
Syntax	long GCSGetCaptionScrollPosition(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	Total movement value
Details	Gets the total scroll movement value from the start position for the object.

GCSUpdateScroll

Outline	Scroll update process
Syntax	void GCSUpdateScroll(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	None
Details	Carries out object scroll update process.

GCSIsCaptionScrollFinish

Outline	Confirm caption scroll completion
Syntax	short GCSIsCaptionScrollFinish(GBaseObject *pControl)
Argument	(i) GBaseObject *pControl : object for operation
Return value	Scroll finish status FALSE : scrolling TRUE : scrolling finish
Details	Confirms whether the caption character string set in the control is being scrolled.

GCSCallBackMessage

Outline Call back message execution process

Syntax
```
long GCSCallBackMessage(
  GBaseObject     *pControl,
  unsigned short  usMessage,
  long            lLParam,
  long            lUParam
)
```

Argument

(i) GBaseObject *pControl	: object for operation
(i) unsigned short usMessage	: call back message type
(i) long lLParam	: appended information (lower-side)
(i) long lUParam	: appended information (upper-side)

Return value Arbitrary

Details For a parent class managing the object, executes call-back processing prepared for an object ID.

GCSLoadScreenFile

Outline Load interpreter screen file

Syntax
```
int GCSLoadScreenFile(
  GBaseObject   *pIScreen,
  long          nPageNo
)
```

Argument

(i) GBaseObject *pIScreen	: object for operation
(i) long nPageNo	: screen No. to load

Return value

0	: normal end
Other than 0	: error

Details Reads the interpreter screen file.

GCSSetPenColor

Outline Set line color

Syntax
```
void GCSSetPenColor(
  GBaseObject   *pShape,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pShape : object for operation
(i) GColor gcColor : line color

Return value None

Details Sets the line color for the drawing/line drawing.
After the line color is set, the drawing/line drawing area is registered as a redraw area.

GCSGetPenColor

Outline Get line color

Syntax
```
GColor GCSGetPenColor(
  GBaseObject   *pShape
)
```

Argument
(i) GBaseObject *pShape : object for operation

Return value Line color

Details Gets the line color for the drawing/line drawing.

GCSSetPenDash

Outline Set line type

Syntax
```
void GCSSetPenDash(
  GBaseObject     *pShape,
  unsigned short  usPenDash
)
```

Argument
(i) GBaseObject *pShape : object for operation
(i) unsigned short usPenDash : line type

Return value None

Details Sets the line type for the drawing/line drawing.
One of the following values is set for usPenDash:
0 : solid line
1 to 8 : line type
When a value outside of the range is set, there is no forced change of the setting.
After the line type is set, the drawing/line drawing area is registered as a redraw area.

GCSGetPenDash

Outline Get line type

Syntax
```
unsigned short GCSGetPenDash(
  GBaseObject  *pShape
)
```

Argument
(i) GBaseObject *pShape : object for operation

Return value
0 : solid line
1 to 8 : line type

Details Gets the line type for the drawing/line drawing.

GCSSetPenSize

Outline Set line width

Syntax
```
void GCSSetPenSize(
  GBaseObject     *pShape,
  unsigned short  usPenSize
)
```

Argument
(i) GBaseObject *pShape : object for operation
(i) unsigned short usPenSize : line width

Return value None

Details Sets the line width for the drawing/line drawing.
After the line width is set, the drawing/line drawing area is registered as a redraw area.

GCSGetPenSize

Outline Get line width

Syntax
```
unsigned short GCSGetPenSize(
  GBaseObject  *pShape
)
```

Argument
(i) GBaseObject *pShape : object for operation

Return value Line width

Details Gets the line width for the drawing/line drawing.

GCSSetAngle

Outline Set start angle and finish angle for sector and arc

Syntax
```
void GCSSetAngle(
  GBaseObject  *pSArc,
  short        nStartAngle,
  short        nEndAngle
)
```

Argument
(i) GBaseObject *pSArc : object for operation
(i) short nStartAngle : start angle
(i) short nEndAngle : end angle

Return value None

Details Sets the start angle and end angle for sector and arcs.

GCSGetAngle

Outline Get start angle and finish angle for sector and arc

Syntax
```
void GCSGetAngle(
  GBaseObject  *pSArc,
  short        *nStartAngle,
  short        *nEndAngle
)
```

Argument
(i) GBaseObject *pSArc : object for operation
(o) short *nStartAngle : start angle
(o) short *nEndAngle : end angle

Return value None

Details Gets the start angle and end angle for sector and arcs.

GCSSetBrush

Outline Set fill brush

Syntax
```
void GCSSetBrush(
  GBaseObject   *pSRect,
  GBrush        *pBrush
)
```

Argument
(i) GBaseObject *pSRect : object for operation
(i) GBrush *pBrush : brush being set

Return value None

Details Sets the brush used for drawing fill.
After the brush is set, the drawing area is registered as a redraw area.

GCSGetBrush

Outline Get fill brush

Syntax
```
GBrush* GCSGetBrush(
  GBaseObject   *pSRect
)
```

Argument
(i) GBaseObject *pSRect : object for operation

Return value Pointer to fill brush

Details Gets the brush used for drawing fill.

GCSSetStartPoint

Outline	Set line drawing starting point
Syntax	void GCSSetStartPoint(GBaseObject *pSLine, GPoint *pPoint)
Argument	(i) GBaseObject *pSLine : object for operation (i) GPoint *pPoint : line drawing starting point
Return value	None
Details	Sets the line drawing starting point.

GCSGetStartPoint

Outline	Get line drawing starting point
Syntax	void GCSGetStartPoint(GBaseObject *pSLine, GPoint *pPoint)
Argument	(i) GBaseObject *pSLine : object for operation (o) GPoint *pPoint : variable storing line drawing starting point
Return value	None
Details	Gets the line drawing starting point.

GCSSetEndPoint

Outline	Set line drawing end point
Syntax	void GCSSetEndPoint(GBaseObject *pSLine, GPoint *pPoint)
Argument	(i) GBaseObject *pSLine : object for operation (i) GPoint *pPoint : line drawing end point
Return value	None
Details	Sets the line drawing end point.

GCSGetEndPoint

Outline	Get line drawing end point
Syntax	void GCSGetEndPoint(GBaseObject *pSLine, .GPoint *pPoint)
Argument	(i) GBaseObject *pSLine : object for operation (o) GPoint *pPoint : variable storing line drawing end point
Return value	None
Details	Gets the line drawing end point.

GCSSetPoly

Outline Set polygon/continuous line

Syntax
```
short GCSSetPoly(
  GBaseObject   *pSPoly,
  GPoly         *pPoly
)
```

Argument
(i) GBaseObject *pSPoly : object for operation
(i) GPoly *pPoly : polygon structure

Return value
FALSE : setting failed
TRUE : setting succeeded

Details
Sets polygon/continuous line data.
Vertex data has memory allocated and is held as internal data.

GCSGetPoly

Outline Get polygon/continuous line

Syntax
```
GPoly* GCSGetPoly(
  GBaseObject   *pSPoly
)
```

Argument
(i) GBaseObject *pSPoly : object for operation

Return value
Pointer to polygon structure

Details
Gets polygon/continuous line data.

GCSSetDrawAttribute

Outline	Set drawing attributes for figure
Syntax	void GCSSetDrawAttribute(GBaseObject *pShape, HWDC pDC)
Argument	(i) GBaseObject *pShape : object for operation (i) HWDC pDC : drawing handle
Return value	None
Details	Sets the drawing attributes. The drawing attributes set are perimeter color, perimeter line type, perimeter line width and fill brush.

GCSBWLInit

Outline	Initialization
Syntax	void GCSBWLInit(HGBWL *hList*)
Argument	(i) HGBWL hList : list handle
Return value	None
Details	Initialization.

GCSBWLAdd

Outline	Addition
Syntax	short GCSBWLAdd(HGBWL *hList,* GBaseObject **pgbwBaseWindow*)
Argument	(i) HGBWL hList : list handle (i) GbaseObject *pgbwBaseWindow : object to be added
Return value	FALSE : addition failed TRUE : addition succeeded
Details	Addition.

GCSBWLRemove

Outline Deletion

Syntax
```
short GCSBWLRemove(
  HGBWL         hList,
  GBaseObject   *pgbwBaseWindow
)
```

Argument
(i) HGBWL hList : list handle
(i) GBaseObject *pgbwBaseWindow : object to be deleted

Return value
FALSE : deletion failed
TRUE : deletion succeeded

Details Deletion.

GCSBWLClear

Outline All clear

Syntax
```
void    GCSBWLClear(
  HGBWL     hList
)
```

Argument (i) HGBWL hList : list handle

Return value None

Details Clears all.

GCSBWLMoveFirst

Outline Move to first

Syntax
```
short GCSBWLMoveFirst(
  HGBWL        hList,
  GBaseObject  *pgbwBaseWindow
)
```

Argument
(i) HGBWL hList : list handle
(i) GBaseObject *pgbwBaseWindow : object to be moved

Return value
FALSE : specified object does not exist
TRUE : move succeeded

Details Moves to the first.

GCSBWLMoveLast

Outline Move to last

Syntax
```
short GCSBWLMoveLast(
  HGBWL        hList,
  GBaseObject  *pgbwBaseWindow
)
```

Argument
(i) HGBWL hList : list handle
(i) GbaseObject *pgbwBaseWindow : object to be moved

Return value
FALSE : specified object does not exist
TRUE : move succeeded

Details Moves to the last.

GCSBWLGetFirst

Outline	Get first object
Syntax	GBaseObject* GCSBWLGetFirst(HGBWL *hList*)
Argument	(i) HGBWL hList : list handle
Return value	NULL : object does not exist Other than NULL : first object
Details	Gets the first object.

GCSBWLGetLast

Outline	Get last object
Syntax	GBaseObject* GCSBWLGetLast(HGBWL *hList*)
Argument	(i) HGBWL hList :list handle
Return value	NULL : object does not exist Other than NULL : last object
Details	Gets the last object.

GCSBWLFindListItem

Outline Find object

Syntax
```
short GCSBWLFindListItem(
  HGBWL        hList,
  GBaseObject  *pBaseWindow
)
```

Argument
(i) HGBWL hList : list handle
(i) GbaseObject *pBaseWindow : object being searched for

Return value
FALSE : none
TRUE : present

Details Finds the object.

GCSBWLFindFirstItem

Outline Move search position to first item

Syntax
```
short GCSBWLFindFirstItem(
  HGBWL  hList
)
```

Argument
(i) HGBWL hList : list handle

Return value
FALSE : move failed
TRUE : move succeeded

Details Moves the search to the first item.

GCSBWLFindNextItem

Outline Acquire object at search position and move search position to next

Syntax
```
short GCSBWLFindNextItem(
  HGBWL         hList,
  GBaseObject   **pgbwBaseWindow
)
```

Argument
(i) HGBWL hList : list handle
(o) GBaseObject **pgbwBaseWindow : store object to be acquired

Return value
FALSE : move failed
TRUE : move succeeded

Details Gets the object at search position and moves search position to the next.

GCSBWLFindLastItem

Outline Move search position to last item

Syntax
```
short GCSBWLFindLastItem(
  HGBWL    hList
)
```

Argument
(i) HGBWL hList : list handle

Return value
FALSE : move failed
TRUE : move succeeded

Details Moves the search position to the last item.

GCSBWLFindPrevItem

Outline Acquire object at search position and move search position to previous

Syntax
```
short GCSBWLFindPrevItem(
  HGBWL         hList,
  GBaseObject   **pgbwBaseWindow
)
```

Argument
(i) HGBWL hList : list handle
(o) GBaseObject **pgbwBaseWindow : store object to be acquired

Return value
FALSE : move failed
TRUE : move succeeded

Details Gets the object at search position and moves search position to the previous.

GCSBWLCount

Outline Get number of list items

Syntax
```
long GCSBWLCount(
  HGBWL    hList
)
```

Argument
(i) HGBWL hList : list handle

Return value Number of list items

Details Gets the number of list items.

GCSBWLSetItemPosition

Outline Set current list search position

Syntax
```
void GCSBWLSetItemPosition(
  HGBWL   hList,
  void    *pvItemPosition
)
```

Argument
(i) HGBWL hList : list handle
(i) void *pvItemPosition : list search position

Return value None

Details Sets the current list search position.

GCSBWLGetItemPosition

Outline Get current list search position

Syntax
```
void* GCSBWLGetItemPosition(
  HGBWL   hList
)
```

Argument
(i) HGBWL hList : list handle

Return value Pointer showing current list search position

Details Gets the current list search position.

GCSBWLNew

Outline	Create list object
Syntax	HGBWL GCSBWLNew()
Argument	None
Return value	List handle
Details	Creates the list object.

GCSBWLDelete

Outline	Delete list object
Syntax	void GCSBWLDelete(HGBWL *hList*)
Argument	(i) HGBWL hList : list handle
Return value	None
Details	Deletes the list object.

GCSWDCGetHGDRAW

Outline Get drawing handle

Syntax
```
HGDRAW GCSWDCGetHGDRAW(
  HWDC   hWDC
)
```

Argument (i) HWDC hWDC : device context handle

Return value None

Details Gets the drawing handle (HGDRAW).

GCSWDCOffsetOrigin

Outline Move position of origin

Syntax
```
void GCSWDCOffsetOrigin(
  HWDC   hWDC,
  short  nX,
  short  nY
)
```

Argument
(i) HWDC hWDC : device context handle
(i) short nX : X coordinate offset amount
(i) short nY : Y coordinate offset amount

Return value None

Details Moves the position of the origin.

GCSWDCSetClipRect

Outline	Define clipping area
Syntax	void GCSWDCSetClipRect(HWDC *hWDC,* GRect **pClip*)
Argument	(i) HWDC hWDC : device context handle (i) GRect *pClip : clipping area
Return value	None
Details	Defines clipping area as pClip.

GCSWDCGetClipRect

Outline	Get clipping area
Syntax	void GCSWDCGetClipRect(HWDC *hWDC,* GRect **pClip*)
Argument	(i) HWDC hWDC : device context handle (i) GRect *pClip : clipping area
Return value	None
Details	Gets the clipping area.

GCSWDCInceptClipRect

Outline

Define clipping for the part overlapping the current clipping area

Syntax

```
short GCSWDCInceptClipRect(
  HWDC   hWDC,
  GRect  *pClip,
  GRect  *pOldClip
)
```

Argument

(i) HWDC hWDC : device context handle
(i) GRect *pClip : clipping area
(o) GRect *pOldClip : previous clipping area

Return value

FALSE : no clipping area
TRUE : clipping area present

Details

Defines the clipping area as the part overlapping between the current clipping area and pClip.

GCSWDCBeginDraw

Outline	Start drawing process
Syntax	void GCSWDCBeginDraw(HWDC *hWDC,* GRect **pRefreshRect*)
Argument	(i) HWDC hWDC : device context handle (i) GRect *pRefreshRect : area for redrawing
Return value	None
Details	Starts the drawing process.

GCSWDCEndDraw

Outline	End drawing process
Syntax	void GCSWDCEndDraw(HWDC *hWDC*)
Argument	(i) HWDC hWDC : device context handle
Return value	None
Details	Ends the drawing process.

GCSWDCNew

Outline	Create device context
Syntax	HWDC GCSWDCNew(HGDRAW *hDraw*)
Argument	(i) HGDRAW hDraw : drawing handle
Return value	Device context NULL : creation failed Other than NULL : device context created
Details	Creates the device context.

GCSWDCDelete

Outline	Delete device context
Syntax	void GCSWDCDelete(HWDC *hWDC*)
Argument	(i) HWDC hWDC : device context handle
Return value	None
Details	Deletes the device context.

GCSWDCSetOrigin

Outline Set position of origin

Syntax
```
void GCSWDCSetOrigin(
  HWDC  hWDC,
  short  nX,
  short  nY
)
```

Argument
(i) HWDC hWDC : device context handle
(i) short nX : X coordinate
(i) short nY : Y coordinate

Return value None

Details Sets the position of the origin.

GCSChangeActiveFocus

Outline Change active focus

Syntax
```
void GCSChangeActiveFocus(
  GBaseObject  *pPanel,
  GBaseObject  *pObject
)
```

Argument
(i) GBaseObject *pPanel : pointer to management panel object
(i) GBaseObject *pObject : object setting the focus

Return value None

Details Changes the active focus to specified object.

GCSGetStandardColor

Outline

Get color information according to the theme color of the specified display area

Syntax

```
GColor GCSGetStandardColor(
 GBaseObject    *pScreen,
 unsigned short  usPartsKind
)
```

Argument

(i) GBaseObject *pScreen : object for operation
(i) unsigned short usPartsKind : color type of part to be acquired

Return value

Color information according to theme color (RGB888)

Details

Gets color information according to the theme color for each part being used in the standard screen.
Any of the following can be specified for usPartsKind (color type of part). When color information cannot be acquired, the return value is -1.

0: Window background color
1: Window frame (inside)
2: Window title frame (upper)
3: Window title frame (lower)
4: Window title gradation (start)
5: Window title gradation (end)
6: Value part normal background color (active area)
7: Value part normal background color (inactive area)
8: Label background color
9: Cursor color
10: Scroll bar - bar color
11: Scroll bar - button color
12: Scroll bar - background color
13: Scroll bar - pinch color
14: Background color during switch-ON state
15: Entry area - background color at the time of disabled state

GCSCheckUnlockFile

Outline Determine whether an unlock key for the custom screen exists

Syntax
```
long GCSCheckUnlockFile(
  GTCHAR          *pszPassword
)
```

Argument (i) GTCHAR *pszPassword : password (key specified by the rights holder)

Return value
1 : unlock key exists
0 : unlock key does not exist

Details Generates an unlock key from pszPassword and determines whether the same unlock key as the one for the unlock file stored in the custom data folder can be generated.

5.2.2 Button

GCSButtonSetAction

Outline	Set button operations
Syntax	void GCSButtonSetAction(GBaseObject *pButton, unsigned char ucAction)
Argument	(i) GBaseObject *pButton : object for operation (i) unsigned char ucAction : button operation
Return value	None
Details	Sets the button operation. One of the following values is set for ucAction: GBTN_ACT_NOACTION : no operation GBTN_ACT_MOMENTARY : momentary operation GBTN_ACT_ALTERNATE : alternative operation When a value outside of the range is set, there is no forced change of the setting.

GCSButtonGetAction

Outline	Get button operations
Syntax	unsigned char GCSButtonGetAction(GBaseObject *pButton)
Argument	(i) GBaseObject *pButton : object for operation
Return value	GBTN_ACT_NOACTION : no operation GBTN_ACT_MOMENTARY : momentary operation GBTN_ACT_ALTERNATE : alternative operation Besides the above : illegal operation setting
Details	Gets the button operation setting. When a setting outside the range is made, that setting value is returned.

GCSButtonSetDisplay

Outline

Set button display

Syntax

```
void GCSButtonSetDisplay(
  GBaseObject     *pButton,
  unsigned char   ucDisplay
)
```

Argument

(i) GBaseObject *pButton : object for operation
(i) unsigned char ucDisplay : button display

Return value

None

Details

Sets the button display.
One of the following values is set for ucDisplay:
GBTN_DISP_RECT : rectangle display
GBTN_DISP_OVAL : round shape display
GBTN_DISP_IMAGE : image display
When a value outside of the range is set, there is no forced change of the setting.
After the button display is set, the control area is registered as a redraw area.

GCSButtonGetDisplay

Outline

Get button display

Syntax

```
unsigned char GCSButtonGetDisplay(
  GBaseObject   *pButton
)
```

Argument

(i) GBaseObject *pButton : object for operation

Return value

GBTN_DISP_RECT : rectangle display
GBTN_DISP_OVAL : round shape display
GBTN_DISP_IMAGE : image display
Besides the above : illegal display setting

Details

Gets the button display setting.
When a setting outside the range is made, that setting value is returned.

GCSButtonSetBorderID

Outline Set 3D border resource ID

Syntax
```
void GCSButtonSetBorderID(
  GBaseObject     *pButton,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) unsigned short usID : 3D border resource ID

Return value None

Details Sets the 3D border resource ID.
After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSButtonGetBorderID

Outline Get 3D border resource ID

Syntax
```
unsigned short GCSButtonGetBorderID(
  GBaseObject   *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value 3D border resource ID

Details Gets the 3D border resource ID.

GCSButtonSetOnDesign

Outline	Set ON status design
Syntax	void GCSButtonSetOnDesign(GBaseObject *pButton, GDesign* pDesign)
Argument	(i) GBaseObject *pButton : object for operation (i) GDesign* pDesign : design
Return value	None
Details	Sets the design for ON status. After the design for ON status is set, the control area is registered as a redraw area.

GCSButtonGetOnDesign

Outline	Get ON status design
Syntax	GDesign* GCSButtonGetOnDesign(GBaseObject *pButton)
Argument	(i) GBaseObject *pButton : object for operation
Return value	Pointer to design
Details	Gets the design for ON status.

GCSButtonSetOffDesign

Outline Set OFF status design

Syntax
```
void GCSButtonSetOffDesign(
  GBaseObject   *pButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) GDesign* pDesign : design

Return value None

Details Sets the design for OFF status.
After the design for OFF status is set, the control area is registered as a redraw area.

GCSButtonGetOffDesign

Outline Get OFF status design

Syntax
```
GDesign* GCSButtonGetOffDesign(
  GBaseObject   *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value Pointer to design

Details Gets the design for OFF status.

GCSButtonSetFocusDesign

Outline Set FOCUS status design

Syntax
```
void GCSButtonSetFocusDesign(
  GBaseObject   *pButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) GDesign* pDesign : design

Return value None

Details Sets the design for FOCUS status.
After the design for FOCUS status is set, the control area is registered as a redraw area.

GCSButtonGetFocusDesign

Outline Get FOCUS status design

Syntax
```
GDesign* GCSButtonGetFocusDesign(
  GBaseObject   *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value Pointer to design

Details Gets the design for FOCUS status.

GCSButtonSetDisableDesign

Outline	Set Disable status design
Syntax	void GCSButtonSetDisableDesign(GBaseObject *pButton, GDesign *pDesign)
Argument	(i) GBaseObject *pButton : object for operation (i) GDesign* pDesign : design
Return value	None
Details	Sets the design for Disable status. After the design for Disable status is set, the control area is registered as a redraw area.

GCSButtonGetDisableDesign

Outline	Get Disable status design
Syntax	GDesign* GCSButtonGetDisableDesign(GBaseObject *pButton)
Argument	(i) GBaseObject *pButton : object for operation
Return value	Pointer to design
Details	Gets the design for Disable status.

GCSButtonSetFontID

Outline Set font resource ID

Syntax
```
void GCSButtonSetFontID(
  GBaseObject     *pButton,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSButtonGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSButtonGetFontID(
  GBaseObject  *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSButtonSetStringID

Outline Set caption character string resource ID

Syntax
```
void GCSButtonSetStringID(
  GBaseObject     *pButton,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) unsigned short usID : caption character string resource ID

Return value None

Details Sets the caption character string resource ID.
After the caption character string resource ID is set, the control area is registered as a redraw area.

GCSButtonGetStringID

Outline Get caption character string resource ID

Syntax
```
unsigned short GCSButtonGetStringID(
  GBaseObject   *pButton
)
```

Argument (i) GBaseObject *pButton : object for operation

Return value Caption character string resource ID

Details Gets the caption character string resource ID.

GCSButtonSetCaption

Outline Set caption information

Syntax
```
void GCSButtonSetCaption(
  GBaseObject   *pButton,
  GCaption *    pCaption
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets pCaption in caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSButtonGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSButtonGetCaption(
  GBaseObject   *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value Pointer to caption information

Details Gets the caption information.

GCSButtonSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSButtonSetFocusEffect(
  GBaseObject     *pButton,
  unsigned char   ucFocusEffect
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
After the effect is set, the control area is registered as a redraw area.

GCSButtonGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSButtonGetFocusEffect(
  GBaseObject   *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSButtonSetStatus

Outline Set object status

Syntax
```
void GCSButtonSetStatus(
  GBaseObject    *pButton,
  unsigned char   ucStatus
)
```

Argument
(i) GBaseObject *pButton : object for operation
(i) unsigned char ucStatus : set object status

Return value None

Details Sets the GCButton object status.
After the GCButton object status is set, the control area is registered as a redraw area.

GCSButtonGetStatus

Outline Get object status

Syntax
```
unsigned char GCSButtonGetStatus(
  GBaseObject   *pButton
)
```

Argument
(i) GBaseObject *pButton : object for operation

Return value Object status

Details Gets the GCButton object status.

5.2.3 TextBox

GCSTextboxSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSTextboxSetBorderID(GBaseObject **pTextBox,* unsigned short *usID*)
Argument	(i) GBaseObject *pTextBox : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSTextboxGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSTextboxGetBorderID(GBaseObject **pTextBox*)
Argument	(i) GBaseObject *pTextBox : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID setting.

GCSTextboxSetFontID

Outline Set font resource ID

Syntax
```
void GCSTextboxSetFontID(
  GBaseObject    *pTextBox,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSTextboxGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSTextboxGetFontID(
  GBaseObject  *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSTextboxSetCaption

Outline Set caption information

Syntax
```
void GCSTextboxSetCaption(
  GBaseObject   *pTextBox,
  GCaption      *pCaption
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets the pCaption in caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSTextboxGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSTextboxGetCaption(
  GBaseObject   *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.

GCSTextboxSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSTextboxSetFocusEffect(
  GBaseObject   *pTextBox,
  unsigned char  ucFocusEffect
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
After the effect is set, the control area is registered as a redraw area.

GCSTextboxGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSTextboxGetFocusEffect(
  GBaseObject  *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSTextboxSetFocusColor

Outline Set background color during focus

Syntax
```
void GCSTextboxSetFocusColor(
  GBaseObject  *pTextBox,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) GColor gcColor : color code for background color during focus

Return value None

Details Sets the background color during focus.
After the background color is set, the control area is registered as a redraw area.

GCSTextboxGetFocusColor

Outline Get background color during focus

Syntax
```
GColor GCSTextboxGetFocusColor(
  GBaseObject    *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value Color code for background color during focus

Details Gets the background color during focus.

GCSTextboxSetNormalColor

Outline Set normal background color

Syntax
```
void GCSTextboxSetNormalColor(
  GBaseObject   *pTextBox,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) GColor gcColor : color code for normal background color

Return value None

Details Sets the normal background color.
After the background color is set, the control area is registered as a redraw area.

GCSTextboxGetNormalColor

Outline Get normal background color

Syntax
```
GColor GCSTextboxGetNormalColor(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value Color code for normal background color

Details Gets the normal background color.

GCSTextboxSetDisableColor

Outline Set background color when disabled

Syntax
```
void GCSTextboxSetDisableColor(
  GBaseObject  *pTextBox,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) GColor gcColor : color code for background color when disabled

Return value None

Details Sets the background color when disabled.
After the background color is set, the control area is registered as a redraw area.

GCSTextboxGetDisableColor

Outline Get background color when disabled

Syntax
```
GColor GCSTextboxGetDisableColor(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value Color code for background color setting when disabled

Details Gets the background color when disabled.

GCSTextboxSetTextType

Outline	Set character string type
Syntax	void GCSTextboxSetTextType(GBaseObject **pTextBox,* unsigned char *ucType*)
Argument	(i) GBaseObject *pTextBox : object for operation (i) unsigned char ucType : character string type
Return value	None
Details	Sets the character string type. One of the following values is set for ucType: GTEXT_TYPE_STRING : character string GTEXT_TYPE_SHORT : short value GTEXT_TYPE_USHORT : unsigned short value GTEXT_TYPE_LONG : long value GTEXT_TYPE_ULONG : unsigned long value GTEXT_TYPE_FLOAT : float value

GCSTextboxGetTextType

Outline

Get character string type

Syntax

```
unsigned char GCSTextboxGetTextType(
  GBaseObject  *pTextBox
)
```

Argument

(i) GBaseObject *pTextBox : object for operation

Return value

GTEXT_TYPE_STRING	: character string
GTEXT_TYPE_SHORT	: short value
GTEXT_TYPE_USHORT	: unsigned short value
GTEXT_TYPE_LONG	: long value
GTEXT_TYPE_ULONG	: unsigned long value
GTEXT_TYPE_FLOAT	: float value
Besides the above	: illegal setting

Details

Gets the display character string type.

GCSTextboxSetFormatID

Outline Set character string display format resource ID

Syntax
```
void GCSTextboxSetFormatID(
  GBaseObject    *pTextBox,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned short usID : character string resource ID as display format

Return value None

Details Sets the character string resource ID used as the format when display in a text box.
After the character string resource ID is set, the control area is registered as a redraw area.

GCSTextboxGetFormatID

Outline Get character string display format resource ID

Syntax
```
unsigned short GCSTextboxGetFormatID(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox :object for operation

Return value Character string resource ID

Details Gets the character string resource ID used as the display format.

GCSTextboxSetPasswordStatus

Outline Set password display status

Syntax
```
void GCSTextboxSetPasswordStatus(
  GBaseObject   *pTextBox,
  short         fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : password display status
FALSE : normal display status
TRUE : password display status

Return value None

Details Sets whether the password will be displayed for text boxes or not.
After the password display status is set, the control area is registered as a redraw area.

GCSTextboxGetPasswordStatus

Outline Get password display status

Syntax
```
short GCSTextboxGetPasswordStatus(
  GBaseObject  *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value
FALSE : normal display status
TRUE : password display status

Details Gets whether the test box display status is normal display status or the password display status.

GCSTextboxSetCommaStatus

Outline Set comma display status

Syntax
```
void GCSTextboxSetCommaStatus(
  GBaseObject  *pTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : comma display status
FALSE : normal display status
TRUE : comma display status

Return value None

Details Sets whether commas will be displayed for text boxes or not.
After the comma display status is set, the control area is registered as a redraw area.

GCSTextboxGetCommaStatus

Outline Get comma display status

Syntax
```
short GCSTextboxGetCommaStatus(
  GBaseObject  *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value
FALSE : normal display status
TRUE : comma display status

Details Gets the setting as to whether the test box display status is normal display status or the comma display status.

GCSTextboxSetRefuseInputNumberStatus

Outline Set numerical input disabled status

Syntax
```
void GCSTextboxSetRefuseInputNumberStatus(
  GBaseObject   *pTextBox,
  short         fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : number input status
FALSE : input enabled
TRUE : input disabled

Return value None

Details Sets whether numerical input will be rejected for text boxes or not.

GCSTextboxGetRefuseInputNumberStatus

Outline Get numerical input disabled status

Syntax
```
short GCSTextboxGetRefuseInputNumberStatus(
  GBaseObject   *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value
FALSE : input enabled
TRUE : input disabled

Details Gets the setting as to whether numerical input will be rejected for text boxes or not.

GCSTextboxSetRefuseInputSmallLetterStatus

Outline Set one-byte lower case character input disabled status

Syntax
```
void GCSTextboxSetRefuseInputSmallLetterStatus(
  GBaseObject  *pTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : one-byte lower case character input status
FALSE : input enabled
TRUE : input disabled

Return value None

Details Sets whether one-byte lower case character input will be rejected for text boxes or not.

GCSTextboxGetRefuseInputSmallLetterStatus

Outline Get one-byte lower case character input disabled status

Syntax
```
short GCSTextboxGetRefuseInputSmallLetterStatus(
  GBaseObject  *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value
FALSE : input enabled
TRUE : input disabled

Details Gets the setting as to whether one-byte lower case character input will be rejected for text boxes or not.

GCSTextboxSetRefuseInputCapitalLetterStatus

Outline Set one-byte upper case character input disabled status

Syntax
```
void GCSTextboxSetRefuseInputCapitalLetterStatus(
  GBaseObject  *pTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : one-byte upper case character input status
FALSE : input enabled
TRUE : input disabled

Return value None

Details Sets whether one-byte upper case character input will be rejected for text boxes or not.

GCSTextboxGetRefuseInputCapitalLetterStatus

Outline Get one-byte upper case character input disabled status

Syntax
```
short GCSTextboxGetRefuseInputCapitalLetterStatus(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value
FALSE : input enabled
TRUE : input disabled

Details Gets whether one-byte upper case character input will be rejected for text boxes or not.

GCSTextboxSetRefuseInputSymbolLetterStatus

Outline Set one-byte symbol input disabled status

Syntax
```
void GCSTextboxSetRefuseInputSymbolLetterStatus(
  GBaseObject   *pTextBox,
  short         fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : one-byte symbol input status
FALSE : input enabled
TRUE : input disabled

Return value None

Details Sets whether one-byte symbol input will be rejected for text boxes or not.

GCSTextboxGetRefuseInputSymbolLetterStatus

Outline Get one-byte symbol input disabled status

Syntax
```
short GCSTextboxGetRefuseInputSymbolLetterStatus(
  GBaseObject   *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value
FALSE : input enabled
TRUE : input disabled

Details Gets whether one-byte symbol input will be rejected for text boxes or not.

GCSTextboxSetRefuseInputFullLetterStatus

Outline Set two-byte character input disabled status

Syntax
```
void GCSTextboxSetRefuseInputFullLetterStatus(
  GBaseObject  *pTextBox,
  short        fStatus
)
```

Argument

(i) GBaseObject *pTextBox		: object for operation
(i) short fStatus		: two-byte character input status
	FALSE	: input enabled
	TRUE	: input disabled

Return value None

Details Sets whether two-byte character input will be rejected for text boxes or not.

GCSTextboxGetRefuseInputFullLetterStatus

Outline Get two-byte character input disabled status

Syntax
```
short GCSTextboxGetRefuseInputFullLetterStatus(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value
FALSE : input enabled
TRUE : input disabled

Details Gets setting as to whether two-byte character input will be rejected for text boxes or not.

GCSTextboxSetCheckMaxStatus

Outline Set maximum value check status

Syntax
```
void GCSTextboxSetCheckMaxStatus(
  GBaseObject  *pTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : setting for maximum value check status
FALSE : check disabled
TRUE : check enabled

Return value None

Details Sets whether the maximum value will be checked with SetValue and GetValue for text boxes.

GCSTextboxGetCheckMaxStatus

Outline Get maximum value check status

Syntax
```
short GCSTextboxGetCheckMaxStatus(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value
FALSE : check disabled
TRUE : check enabled

Details Gets the setting as to whether the maximum value will be checked with SetValue and GetValue from text boxes.

GCSTextboxSetCheckMinStatus

Outline Set minimum value check status

Syntax
```
void GCSTextboxSetCheckMinStatus(
  GBaseObject   *pTextBox,
  short         fStatus
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short fStatus : setting for minimum value check status
FALSE : check disabled
TRUE : check enabled

Return value None

Details Sets whether the maximum value will be checked with SetValue and GetValue for text boxes.

GCSTextboxGetCheckMinStatus

Outline Get minimum value check status

Syntax
```
short GCSTextboxGetCheckMinStatus(
  GBaseObject   *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value
FALSE : check disabled
TRUE : check enabled

Details Gets the setting as to whether the minimum value will be checked with SetValue and GetValue from text boxes.

GCSTextboxSetMaxGValue

Outline Set maximum value

Syntax
```
void GCSTextboxSetMaxGValue(
  GBaseObject  *pTextBox,
  GValue       gvValue
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) GValue gvValue : maximum value

Return value None

Details Sets the maximum value when checking the maximum value with SetValue and GetValue for text boxes.

GCSTextboxGetMaxGValue

Outline Get maximum value

Syntax
```
GValue GCSTextboxGetMaxGValue(
  GBaseObject  *pTextBox
)
```

Argument (i) GBaseObject *pTextBox : object for operation

Return value Maximum value (GValue)

Details Gets the maximum value when checking the maximum value for SetValue and GetValue from text boxes.

GCSTextboxSetMinGValue

Outline	Set minimum value
Syntax	void GCSTextboxSetMinGValue(GBaseObject **pTextBox,* GValue *gvValue*)
Argument	(i) GBaseObject *pTextBox : object for operation (i) GValue gvValue : minimum value
Return value	None
Details	Sets the minimum value when checking the minimum value with SetValue and GetValue for text boxes.

GCSTextboxGetMinGValue

Outline	Get minimum value
Syntax	GValue GCSTextboxGetMinGValue(GBaseObject **pTextBox*)
Argument	(i) GBaseObject *pTextBox : object for operation
Return value	Minimum value (GValue)
Details	Gets the minimum value when checking the minimum value with SetValue and GetValue from text boxes.

GCSTextboxSetString

Outline Set display character string

Syntax
```
short GCSTextboxSetString(
  GBaseObject      *pTextBox,
  const GTCHAR     *pString
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) const char *pString : display character string

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the display character string.

GCSTextboxGetString

Outline Get display character string

Syntax
```
short GCSTextboxGetString(
  GBaseObject   *pTextBox,
  GTCHAR        *pString,
  long          lMaxCount
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(o) char *pString : buffer storing display character string
(i) long lMaxCount : buffer size

Return value
FALSE : acquisition failed
TRUE : acquisition succeeded

Details Stores the display character string in pString.

GCSTextboxSetGValue

Outline

Set display value

Syntax

```
short   GCSTextboxSetGValue(
  GBaseObject  *pTextBox,
  GValue       gvValue
)
```

Argument

(i) GBaseObject *pTextBox : object for operation
(i) GValue gvValue : setting value

Return value

0	: normal value
GTEXT_ERROR_UNDER_MIN_LIMIT	: less than minimum value
GTEXT_ERROR_OVER_MAX_LIMIT	: greater than maximum value

Details

Sets the value displayed in a text box.
Numbers are converted to character strings according to the display format.
At this time, the maximum value and minimum value are checked, and if it is outside of the range, an error is returned in the return value.
Error checks are performed by:
GTEXT_TYPE_SHORT
GTEXT_TYPE_USHORT
GTEXT_TYPE_LONG

GCSTextboxGetGValue

Outline

Get display value

Syntax

```
short   GCSTextboxGetGValue(
  GBaseObject   *pTextBox,
  GValue        *gvValue
)
```

Argument

(i) GBaseObject *pTextBox : object for operation
(i) GValue *gvValue : GValue variable storing the value

Return value

0	: normal value
GTEXT_ERROR_UNDER_MIN_LIMIT	: less than minimum value
GTEXT_ERROR_OVER_MAX_LIMIT	: greater than maximum value

Details

Gets the value displayed in a text box.
Character strings are converted to numbers according to the display format.
At this time, the maximum value and minimum value are checked, and if it is outside of the range, an error is returned in the return value.
Error checks are performed by:
GTEXT_TYPE_SHORT
GTEXT_TYPE_USHORT

GCSTextboxSetStringBuffer

Outline Set display buffer

Syntax
```
short GCSTextboxSetStringBuffer(
  GBaseObject     *pTextBox,
  unsigned long   ulBufferSize,
  GTCHAR          *pStringBuffer
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned long ulBufferSize : buffer size
(i) char *pStringBuffer : character string buffer

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the display buffer.
Allocates memory internally when pStringBuffer is NULL. When a character string buffer is specified, that buffer is set.

GCSTextboxGetStringBuffer

Outline Get display buffer

Syntax
```
GTCHAR* GCSTextboxGetStringBuffer(
  GBaseObject   *pTextBox,
  unsigned long  *ulBufferSize
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned long *ulBufferSize : variable storing the buffer size

Return value Display buffer address

Details Gets the display buffer.

GCSTextboxSetBackGroundPattern

Outline Set background fill status

Syntax
```
void GCSTextboxSetBackGroundPattern(
  GBaseObject  *pTextBox,
  short        nBackGroundPattern
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) short nBackGroundPattern : background fill status

Return value None

Details Sets the background color fill status.
NO_DRAW and FILL_BACK_COLOR are the corresponding fill statuses.

GCSTextboxGetBackGroundPattern

Outline Get background fill status

Syntax
```
short GCSTextboxGetBackGroundPattern(
  GBaseObject  *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value
Background fill status
NO_DRAW : no fill
FILL_BACK_COLOR : background fill

Details Sets the background color fill status.
NO_DRAW and FILL_BACK_COLOR are the corresponding fill statuses.

GCSTextboxSetSelectStringStartPos

Outline Set selected character string start position

Syntax
```
void GCSTextboxSetSelectStringStartPos(
  GBaseObject    *pTextBox,
  unsigned short  usStartPos
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned short usStartPos : selected character string start position

Return value None

Details Sets the selected character string start position.

GCSTextboxGetSelectStringStartPos

Outline Get selected character string start position

Syntax
```
unsigned short GCSTextboxGetSelectStringStartPos(
GBaseObject    *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value Selected character string start position

Details Gets the selected character string start position.

GCSTextboxSetSelectStringEndPos

Outline Set selected character string end position

Syntax
```
void GCSTextboxSetSelectStringEndPos(
  GBaseObject    *pTextBox,
  unsigned short  usEndPos
)
```

Argument
(i) GBaseObject *pTextBox : object for operation
(i) unsigned short usEndPos : selected character string end position

Return value None

Details Sets the selected character string end position.

GCSTextboxGetSelectStringEndPos

Outline Get selected character string end position

Syntax
```
unsigned short GCSTextboxGetSelectStringEndPos(
  GBaseObject    *pTextBox
)
```

Argument
(i) GBaseObject *pTextBox : object for operation

Return value Selected character string end position

Details Gets the selected character string end position.

5.2.4 Label

GCSLabelSetFontID

Outline	Set font resource ID
Syntax	void GCSLabelSetFontID(GBaseObject **pLabel,* unsigned short *usID*)
Argument	(i) GBaseObject *pLabel : object for operation (i) unsigned short usID : font resource ID
Return value	None
Details	Sets the font resource ID. After the font resource ID is set, the control area is registered as a redraw area.

GCSLabelGetFontID

Outline	Get font resource ID
Syntax	unsigned short GCSLabelGetFontID(GBaseObject **pLabel*)
Argument	(i) GBaseObject *pLabel : object for operation
Return value	Font resource ID
Details	Gets the font resource ID.

GCSLabelSetStringID

Outline	Set caption character string resource ID
Syntax	void GCSLabelSetStringID(GBaseObject **pLabel,* unsigned short *usID*)
Argument	(i) GBaseObject *pLabel : object for operation (i) unsigned short usID : caption character string resource ID
Return value	None
Details	Sets the caption character string resource ID. After the caption character string resource ID is set, the control area is registered as a redraw area.

GCSLabelGetStringID

Outline	Get caption character string resource ID
Syntax	unsigned short GCSLabelGetStringID(GBaseObject **pLabel*)
Argument	(i) GBaseObject *pLabel : object for operation
Return value	Caption character string resource ID
Details	Gets the caption character string resource ID.

GCSLabelSetCaption

Outline Set caption information

Syntax
```
void GCSLabelSetCaption(
  GBaseObject   *pLabel,
  GCaption      *pCaption
)
```

Argument
(i) GBaseObject *pLabel : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets pCaption in caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSLabelGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSLabelGetCaption(
  GBaseObject   *pLabel
)
```

Argument
(i) GBaseObject *pLabel : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.

5.2.5 List

GCSListSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSListSetBorderID(GBaseObject **pList,* unsigned short *usID*)
Argument	(i) GBaseObject *pList : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSListGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSListGetBorderID(GBaseObject **pList*)
Argument	(i) GBaseObject *pList : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID setting.

GCSListSetFontID

Outline Set font resource ID

Syntax
```
void GCSListSetFontID(
  GBaseObject     *pList,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSListGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSListGetFontID(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSListSetCaption

Outline Set caption information

Syntax
```
void GCSListSetCaption(
  GBaseObject   *pList,
  GCaption      *pCaption
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets pCaption in caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSListGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSListGetCaption(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.

GCSListSetFocusColor

Outline Set background color during focus

Syntax
```
void GCSListSetFocusColor(
  GBaseObject   *pList,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for background color during focus

Return value None

Details Sets the background color during focus.
After the background color is set, the control area is registered as a redraw area.

GCSListGetFocusColor

Outline Get background color during focus

Syntax
```
GColor GCSListGetFocusColor(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code for background color during focus

Details Gets the background color during focus.

GCSListSetScrollButtonColor

Outline Set scroll button color

Syntax
```
void GCSListSetScrollButtonColor(
  GBaseObject   *pList,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for scroll button color

Return value None

Details Sets the scroll button color.
After the scroll button color is set, the control area is registered as a redraw area.

GCSListGetScrollButtonColor

Outline Get scroll button color

Syntax
```
GColor GCSListGetScrollButtonColor(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code

Details Gets the scroll button color.

GCSListSetScrollBarColor

Outline Set scroll bar color

Syntax
```
void GCSListSetScrollBarColor(
  GBaseObject   *pList,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for scroll bar color

Return value None

Details Sets the scroll bar color.
After the scroll bar color is set, the control area is registered as a redraw area.

GCSListGetScrollBarColor

Outline Get scroll bar color

Syntax
```
GColor GCSListGetScrollBarColor(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code

Details Gets the scroll bar color.

GCSListSetNormalColor

Outline Set normal background color

Syntax
```
void GCSListSetNormalColor(
  GBaseObject   *pList,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for normal background color

Return value None

Details Sets the normal background color.
After the normal background color is set, the control area is registered as a redraw area.

GCSListGetNormalColor

Outline Get normal background color

Syntax
```
GColor GCSListGetNormalColor(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code for normal background color

Details Gets the normal background color.

GCSListSetDisableColor

Outline Set background color when disabled

Syntax
```
void GCSListSetDisableColor(
  GBaseObject   *pList,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for background color when disabled

Return value None

Details Sets the background color when disabled.
After the background color is set, the control area is registered as a redraw area.

GCSListGetDisableColor

Outline Get background color when disabled

Syntax
```
GColor GCSListGetDisableColor(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code for background color setting when disabled

Details Gets the background color when disabled.

GCSListSetScrollBarWidth

Outline Set scroll bar width

Syntax
```
void GCSListSetScrollBarWidth(
  GBaseObject     *pList,
  unsigned char   ucScroolBarWidth
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned char ucScrollBarWidth : scroll bar width

Return value None

Details Sets the scroll bar width.
After the scroll bar width is set, the control area is registered as a redraw area.

GCSListGetScrollBarWidth

Outline Get scroll bar width

Syntax
```
unsigned char GCSListGetScrollBarWidth(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Scroll bar width

Details Gets the scroll bar width.

GCSListSetSelectBarColor

Outline Set selection bar color

Syntax
```
void GCSListSetSelectBarColor(
  GBaseObject   *pList,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for selection bar color

Return value None

Details Sets the selection bar color.
After the selection bar color is set, the control area is registered as a redraw area.

GCSListGetSelectBarColor

Outline Get selection bar color

Syntax
```
GColor GCSListGetSelectBarColor(
  GBaseObject   *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code for selection bar

Details Gets the selection bar color.

GCSListSetMaxListLines

Outline Set maximum number of lines in list

Syntax
```
void GCSListSetMaxListLines(
  GBaseObject     *pList,
  unsigned short  usMaxListLines
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned short usMaxListLines : maximum number 0 to 32767

Return value None

Details Sets the maximum number of lines for list control.
The range of settings is 0 to 32767.
When the maximum number of lines for display that is set is less than the number of lines in a particular list already registered, the character strings in the list that are over the setting are deleted.

GCSListGetMaxListLines

Outline Get maximum number of lines in list

Syntax
```
unsigned short GCSListGetMaxListLines(
  GBaseObject  * pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Maximum number of lines in list

Details Gets the maximum number of lines for list control.

GCSListAddString

Outline Add list character string

Syntax
```
short GCSListAddString(
  GBaseObject  *pList,
  GTCHAR       *pszString
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) char *pszString : character string to be added

Return value
-1 : addition failed
0 or more : line added

Details Adds a character string to the end of a list.

GCSListGetListString

Outline Get list character string

Syntax
```
short GCSListGetListString(
  GBaseObject  *pList,
  short        nIndex,
  GTCHAR       *pszString,
  short        nLength
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) short nIndex : line No. to be acquired
(o) char* pszString : buffer storing the acquired character string
(i) short nLength : buffer size

Return value
FALSE : acquisition failed
TRUE : acquisition succeeded

Details Gets the character string for the specified line in a list.

GCSListRemoveAllStrings

Outline	Delete all list character strings
Syntax	void GCSListRemoveAllStrings(GBaseObject **pList*)
Argument	(i) GBaseObject *pList : object for operation
Return value	None
Details	Deletes all the character strings in a list.

GCSListSetCurrentSelect

Outline Set selected line

Syntax
```
short GCSListSetCurrentSelect(
  GBaseObject  *pList,
  long         nSel
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) long nSel : set selected line
-1 : no selected line
0 or more : specify selected line

Return value
FALSE : setting failed (illegal setting)
TRUE : setting succeeded

Details Sets the selected line in a list.
When -1 is specified, a selection is not made.
When the maximum number of lines is set, FALSE is returned for the return value.

GCSListGetCurrentSelect

Outline Get selected line

Syntax
```
long GCSListGetCurrentSelect(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value
0 or more : current selected line
-1 : no selection

Details Gets the selected line in a list.

GCSListSetVisibleScrollBarStatus

Outline Set scroll bar display/non-display status

Syntax
```
void GCSListSetVisibleScrollBarStatus(
  GBaseObject  *pList,
  short        fStatus
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) short fStatus : display/non-display status
FALSE : non-display
TRUE : display

Return value None

Details Sets the display/non-display status for the scroll bar.
When switching the display/non-display, the object area is registered as a redraw area.

GCSListGetVisibleScrollBarStatus

Outline Get scroll bar display/non-display status

Syntax
```
short GCSListGetVisibleScrollBarStatus(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value
FALSE : non-display
TRUE : display

Details Gets the display/non-display status for the scroll bar.

GCSListSetEnableScrollBarStatus

Outline Set scroll bar controllable or not controllable status

Syntax
```
void GCSListSetEnableScrollBarStatus(
  GBaseObject  *pList,
  short        fStatus
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) short fStatus : controllable or not controllable status
FALSE : not controllable
TRUE : controllable

Return value None

Details Sets the controllable or not controllable status for the scroll bar.
Controllability with mouse input or key input is determined according to this flag.

GCSListGetEnableScrollBarStatus

Outline Get scroll bar controllable or not controllable status

Syntax
```
short GCSListGetEnableScrollBarStatus(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value
FALSE : not controllable
TRUE : controllable

Details Gets the controllable or not controllable status for the scroll bar.

GCSListSetTopLine

Outline Set list top line

Syntax
```
void GCSListSetTopLine(
  GBaseObject  *pList,
  long         lIndex
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) long lIndex : top line No.

Return value None

Details Changes the top line of the list to 1Position.
When a value outside of the range is set, it is changed to the smallest line or the largest line.

GCSListGetTopLine

Outline Get list top line

Syntax
```
long GCSListGetTopLine(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Top line No.

Details Gets the current first line in a list.

GCSListInsertString

Outline Insert list character string

Syntax
```
short GCSListInsertString(
  GBaseObject  *pList,
  short        nIndex,
  GTCHAR       *pszString
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) short nIndex : line No. for insertion (-1: add at the end)
(i) char *pszString : character string to be added

Return value
-1 : addition failed
0 or more : line added

Details
Adds a character string to the specified line in a list.
Specifies a line No. to be added to nIndex.
When -1 is specified in nIndex, it is added to the end of the list.
When the line specified by nIndex does not exist, it is an addition failure.

GCSListGetLineCount

Outline Get number of lines in list

Syntax
```
short GCSListGetLineCount(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value
Current number of lines in list

Details
Gets the current number of lines in a list.

GCSListRemoveString

Outline Delete list character string

Syntax
```
short GCSListRemoveString(
  GBaseObject  *pList,
  short        nIndex
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) short nIndex : line No. for deletion

Return value
FALSE : deletion failed
TRUE : deletion succeeded

Details Deletes the character string for the specified line in a list.

GCSListSetScrollBarBorderID

Outline	Set scroll bar 3D border resource ID
Syntax	void GCSListSetScrollBarBorderID(GBaseObject **pList,* unsigned short *usID*)
Argument	(i) GBaseObject *pList : object for operation (i) unsigned short usID : scroll bar 3D border resource ID
Return value	None
Details	Sets the resource ID of the scroll bar 3D border. After the resource ID is set, the control area is registered as a redraw area.

GCSListGetScrollBarBorderID

Outline	Get scroll bar 3D border resource ID
Syntax	unsigned short GCSListGetScrollBarBorderID(GBaseObject **pList,* unsigned short *usID*)
Argument	(i) GBaseObject *pList : object for operation (i) unsigned short usID : scroll bar 3D border resource ID
Return value	Scroll bar 3D border resource ID
Details	Gets the resource ID settings of the scroll bar 3D border.

GCSListSetSelectStringColor

Outline	Set selected character color
Syntax	void GCSListSetSelectStringColor(GBaseObject *pList, GColor gcColor)
Argument	(i) GBaseObject *pList : object for operation (i) GColor gcColor : selected character color
Return value	None
Details	Sets the selected character color.

GCSListGetSelectStringColor

Outline	Get selected character color
Syntax	GColor GCSListGetSelectStringColor(GBaseObject *pList)
Argument	(i) GBaseObject *pList : object for operation
Return value	Selected character color
Details	Gets the selected character color.

GCSListSetScrollBarDisplay

Outline Set display type

Syntax
```
void GCSListSetScrollBarDisplay(
  GBaseObject    *pList,
  unsigned char  ucDisplay
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned char ucDisplay : display type

Return value None

Details Sets the display type.
Set any of the following values for ucDisplay:
GSBAR_DISP_IMAGE: image
GSBAR_DISP_NORMAL: fill

GCSListGetScrollBarDisplay

Outline Get display type

Syntax
```
unsigned char GCSListGetScrollBarDisplay(
  GBaseObject    *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value
GSBAR_DISP_IMAGE: image
GSBAR_DISP_NORMAL: fill

Details Gets the display type.

GCSListSetScrollArrowColor

Outline Set arrow color

Syntax
```
void GCSListSetScrollArrowColor(
  GBaseObject  *pList,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : arrow color

Return value None

Details Sets the arrow color.
After the arrow color is set, the control area is registered as a redraw area.

GCSListGetScrollArrowColor

Outline Get arrow color

Syntax
```
GColor GCSListGetScrollArrowColor(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Arrow color

Details Gets the arrow color.

GCSListSetPrevBtnOnImgID

Outline Set design (upper) resource when button is ON

Syntax
```
void GCSListSetPrevBtnOnImgID(
  GBaseObject    *pList,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned short usID : resource ID

Return value None

Details Sets the design (upper) when the button is ON.
After the design is set, the control area is registered as a redraw area.

GCSListSetPrevBtnOffImgID

Outline Set design (upper) resource when button is OFF

Syntax
```
void GCSListSetPrevBtnOffImgID(
  GBaseObject    *pList,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned short usID : resource ID

Return value None

Details Sets the design (upper) when the button is OFF.
After the design is set, the control area is registered as a redraw area.

GCSListGetPrevBtnOnImgID

Outline	Get design (upper) resource when button is ON
Syntax	unsigned short GCSListGetPrevBtnOnImgID(GBaseObject *pList)
Argument	(i) GBaseObject *pList : object for operation
Return value	Design (upper) resource ID when the button is ON
Details	Gets the design (upper) when the button is ON.

GCSListGetPrevBtnOffImgID

Outline	Get design (upper) resource when button is OFF
Syntax	unsigned short GCSListGetPrevBtnOffImgID(GBaseObject *pList)
Argument	(i) GBaseObject *pList : object for operation
Return value	Design (upper) resource ID when the button is OFF
Details	Gets the design (upper) when the button is OFF.

GCSListSetNextBtnOnImgID

Outline	Set design (lower) resource when button is ON
Syntax	void GCSListSetNextBtnOnImgID(GBaseObject **pList,* unsigned short *usID*)
Argument	(i) GBaseObject *pList : object for operation (i) unsigned short usID : resource ID
Return value	None
Details	Sets the design (lower) when the button is ON. After the design is set, the control area is registered as a redraw area.

GCSListSetNextBtnOffImgID

Outline	Set design (lower) resource when button is OFF
Syntax	void GCSListSetNextBtnOffImgID(GBaseObject **pList,* unsigned short *usID*)
Argument	(i) GBaseObject *pList : object for operation (i) unsigned short usID : resource ID
Return value	None
Details	Sets the design (lower) when the button is OFF. After the design is set, the control area is registered as a redraw area.

GCSListGetNextBtnOnImgID

Outline	Get design (lower) resource when button is ON
Syntax	unsigned short GCSListGetNextBtnOnImgID(GBaseObject *pList)
Argument	(i) GBaseObject *pList : object for operation
Return value	Design (lower) resource ID when the button is ON
Details	Gets the design (lower) when the button is ON.

GCSListGetNextBtnOffImgID

Outline	Get design (lower) resource when button is OFF
Syntax	unsigned short GCSListGetNextBtnOffImgID(GBaseObject *pList)
Argument	(i) GBaseObject *pList : object for operation
Return value	Design (lower) resource ID when the button is OFF
Details	Gets the design (lower) when the button is OFF.

GCSListSetScrollBarBackGroundColor

Outline Set background color

Syntax
```
void GCSListSetScrollBarBackGroundColor(
  GBaseObject  *pList,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : background color

Return value None

Details Sets the background color. After the background color is set, the control area is registered as a redraw area.

GCSListGetScrollBarBackGroundColor

Outline Get background color

Syntax
```
GColor GCSListGetScrollBarBackGroundColor(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Background color

Details Gets the background color.

GCSListSetPinchWidth

Outline Set pinch width

Syntax
```
void GCSListSetPinchWidth(
  GBaseObject    *pList,
  unsigned short  nWidth
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) unsigned short nWidth : pinch width

Return value None

Details Sets the pinch width.

GCSListGetPinchWidth

Outline Get pinch width

Syntax
```
unsigned short GCSListGetPinchWidth(
  GBaseObject    *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Pinch width

Details Gets the pinch width.

GCSListSetPinchColor

Outline Set pinch color

Syntax
```
void GCSListSetPinchColor(
  GBaseObject  *pList,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : pinch color

Return value None

Details Sets the pinch color.

GCSListGetPinchColor

Outline Get pinch color

Syntax
```
GColor GCSListGetPinchColor(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Pinch color

Details Gets the pinch color.

GCSListSetScrollBarOutLineColor

Outline Set color of scroll outer line

Syntax
```
void GCSListSetScrollBarOutLineColor(
  GBaseObject  *pList,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pList : object for operation
(i) GColor gcColor : color code for scroll outer line

Return value None

Details Sets the color of scroll outer line.
After the color is set, the control area is registered as a redraw area.

GCSListGetScrollBarOutLineColor

Outline Get color of scroll outer line

Syntax
```
GColor GCSListGetScrollBarOutLineColor(
  GBaseObject  *pList
)
```

Argument
(i) GBaseObject *pList : object for operation

Return value Color code

Details Gets the color of scroll outer line.

5.2.6 Picture

GCSPictureSetDisplay

Outline	Set picture display
Syntax	void GCSPictureSetDisplay(GBaseObject **pPicture,* unsigned char *ucDisplay*)
Argument	(i) GBaseObject *pPicture : object for operation (i) unsigned char ucDisplay : picture display
Return value	None
Details	Sets the picture display. One of the following values is set for ucDisplay: GPCTR_DISP_RECT : rectangle display GPCTR_DISP_OVAL : (oval) round shape display GPCTR_DISP_IMAGE : image display When a value outside of the range is set, there is no forced change of the setting.

GCSPictureGetDisplay

Outline	Get picture display
Syntax	unsigned char GCSPictureGetDisplay(GBaseObject *pPicture)
Argument	(i) GBaseObject *pPicture : object for operation
Return value	GPCTR_DISP_RECT : rectangle display GPCTR_DISP_OVAL : (oval) round shape display GPCTR_DISP_IMAGE : image display Besides the above : illegal display setting
Details	Gets the picture display setting. When a setting outside the range is made, that setting value is returned.

GCSPictureSetBorderID

Outline Set 3D border resource ID

Syntax
```
void GCSPictureSetBorderID(
  GBaseObject    *pPicture,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pPicture : object for operation
(i) unsigned short usID : 3D border resource ID

Return value None

Details Sets the 3D border resource ID.
After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSPictureGetBorderID

Outline Get 3D border resource ID

Syntax
```
unsigned short GCSPictureGetBorderID(
  GBaseObject    *pPicture
)
```

Argument
(i) GBaseObject *pPicture : object for operation

Return value 3D border resource ID

Details Gets the 3D border resource ID.

GCSPictureSetFontID

Outline Set font resource ID

Syntax
```
void GCSPictureSetFontID(
  GBaseObject     *pPicture,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pPicture : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSPictureGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSPictureGetFontID(
  GBaseObject  *pPicture
)
```

Argument (i) GBaseObject *pPicture : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSPictureSetStringID

Outline Set caption character string resource ID

Syntax
```
void GCSPictureSetStringID(
  GBaseObject     *pPicture,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pPicture : object for operation
(i) unsigned short usID : caption character string resource ID

Return value None

Details Sets the caption character string resource ID.
After the caption character string resource ID is set, the control area is registered as a redraw area.

GCSPictureGetStringID

Outline Get caption character string resource ID

Syntax
```
unsigned short GCSPictureGetStringID(
  GBaseObject  *pPicture
)
```

Argument (i) GBaseObject *pPicture : object for operation

Return value Caption character string resource ID

Details Gets the caption character string resource ID.

GCSPictureSetCaption

Outline Set caption information

Syntax
```
void GCSPictureSetCaption(
  GBaseObject   *pPicture,
  GCaption      *pCaption
)
```

Argument
(i) GBaseObject *pPicture : object for operation
(i) GCaption *pCaption : pointer to caption information

Return value None

Details Sets pCaption in caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSPictureGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSPictureGetCaption(
  GBaseObject   *pPicture
)
```

Argument
(i) GBaseObject *pPicture : object for operation

Return value Pointer to caption information

Details Gets the caption information.

GCSPictureSetStatus

Outline	Set object status
Syntax	void GCSPictureSetStatus(GBaseObject **pPicture,* unsigned char *ucStatus*)
Argument	(i) GBaseObject *pPicture : object for operation (i) unsigned char ucStatus : set object status
Return value	None
Details	Sets the GCButton object status. After the GCButton object status is set, the control area is registered as a redraw area.

GCSPictureGetStatus

Outline	Get object status
Syntax	unsigned char GCSPictureGetStatus(GBaseObject **pPicture*)
Argument	(i) GBaseObject *pPicture : object for operation
Return value	Object status
Details	Gets the GCButton object status.

GCSPictureSetMaxStatus

Outline Set maximum picture status

Syntax
```
void GCSPictureSetMaxStatus(
  GBaseObject    *pPicture,
  unsigned char  ucStatus
)
```

Argument
(i) GBaseObject *pPicture : object for operation
(i) unsigned char ucMaxStatus : maximum number of states

Return value None

Details Sets the maximum number of states for the picture.
The design of the picture indicates that there are this many states.

GCSPictureGetMaxStatus

Outline Get maximum picture status

Syntax
```
unsigned char GCSPictureGetMaxStatus(
  GBaseObject  *pPicture
)
```

Argument (i) GBaseObject *pPicture : object for operation

Return value Maximum number of states

Details Gets the maximum number of states for the picture.

GCSPictureSetStatusDesign

Outline	Set picture status design
Syntax	void GCSPictureSetStatusDesign(GBaseObject *pPicture, GDesign *pDesign)
Argument	(i) GBaseObject *pPicture : object for operation (i) GDesign *pDesign : pointer to status design arrangement
Return value	None
Details	Sets the pointer to picture status design arrangement. After the pointer is set, the control area is registered as a redraw area.

GCSPictureGetStatusDesign

Outline	Get picture status design
Syntax	GDesign* GCSPictureGetStatusDesign(GBaseObject *pPicture)
Argument	(i) GBaseObject *pPicture : object for operation
Return value	NULL : no setting Other than NULL : Pointer to status design arrangement
Details	Gets the pointer to picture status design arrangement.

5.2.7 CheckBox

GCSCheckboxSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSCheckboxSetBorderID(GBaseObject **pCheckBox,* unsigned short *usID*)
Argument	(i) GBaseObject *pCheckBox : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSCheckboxGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSCheckboxGetBorderID(GBaseObject **pCheckBox*)
Argument	(i) GBaseObject *pCheckBox : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID setting.

GCSCheckboxSetFontID

Outline Set font resource ID

Syntax
```
void GCSCheckboxSetFontID(
  GBaseObject     *pCheckBox,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSCheckboxGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSCheckboxGetFontID(
  GBaseObject  *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSCheckboxSetStringID

Outline Set caption character string resource ID

Syntax
```
void GCSCheckboxSetStringID(
  GBaseObject     *pCheckBox,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) unsigned short usID : caption character string resource ID.

Return value None

Details Sets the caption character string resource ID.
After the caption character string resource ID is set, the control area is registered as a redraw area.

GCSCheckboxGetStringID

Outline Get caption character string resource ID

Syntax
```
unsigned short GCSCheckboxGetStringID(
  GBaseObject   *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Caption character string resource ID.

Details Gets the caption character string resource ID.

GCSCheckboxSetCaption

Outline Set caption information

Syntax
```
void GCSCheckboxSetCaption(
  GBaseObject   *pCheckBox,
  GCaption      *pCaption
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets pCaption in caption information.
All except character color are invalid.
After the caption information is set, the control area is registered as a redraw area.

GCSCheckboxGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSCheckboxGetCaption(
  GBaseObject   *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.
All settings except character color are invalid.

GCSCheckboxSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSCheckboxSetFocusEffect(
  GBaseObject     *pCheckBox,
  unsigned char   ucFocusEffect
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
After the effect is set, the control area is registered as a redraw area.

GCSCheckboxGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSCheckboxGetFocusEffect(
  GBaseObject   *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSCheckboxSetStatus

Outline

Set object status

Syntax

```
void GCSCheckboxSetStatus(
  GBaseObject     *pCheckBox,
  unsigned char   ucStatus
)
```

Argument

(i) GBaseObject *pCheckBox : object for operation
(i) unsigned char ucStatus : set object status

Return value

None

Details

Sets status for GCCheckBox object.
One of the following values is set for ucStatus:

GCHK_STATE_OFF	: OFF status
GCHK_STATE_ON	: ON status
GCHK_STATE_OFFSELECT	: OFF selection status
GCHK_STATE_ONSELECT	: ON selection status

When a setting outside of the range is made, there is no forced change of the setting.

GCSCheckboxGetStatus

Outline

Get object status

Syntax

```
unsigned char GCSCheckboxGetStatus(
  GBaseObject   *pCheckBox
)
```

Argument

(i) GBaseObject *pCheckBox : object for operation

Return value

GCHK_STATE_OFF	: OFF status
GCHK_STATE_ON	: ON status
GCHK_STATE_OFFSELECT	: OFF selection status
GCHK_STATE_ONSELECT	: ON selection status
Besides the above	: undefined status

Details

Gets status for GCCheckBox object.

GCSCheckboxSetBoxSize

Outline Set box size

Syntax
```
void GCSCheckboxSetBoxSize(
  GBaseObject     *pCheckBox,
  unsigned short  ucBoxSize
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) unsigned short usID : check box size

Return value None

Details Sets the box size.
After the box size is set, the control area is registered as a redraw area.

GCSCheckboxGetBoxSize

Outline Get box size

Syntax
```
unsigned short GCSCheckboxGetBoxSize(
  GBaseObject  *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Box size

Details Gets the box size settings.

GCSCheckboxSetBoxColor

Outline Set box color

Syntax
```
void GCSCheckboxSetBoxColor(
  GBaseObject    *pCheckBox,
  GColor         gcColor
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) GColor gcColor : box color

Return value None

Details Sets the box color.
After the box color is set, the control area is registered as a redraw area.

GCSCheckboxGetBoxColor

Outline Get box color

Syntax
```
GColor GCSCheckboxGetBoxColor(
  GBaseObject   *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Box color

Details Gets the box color.

GCSCheckboxSetFocusColor

Outline Set background color during focus

Syntax
```
void GCSCheckboxSetFocusColor(
  GBaseObject     *pCheckBox,
  GColor          gcColor
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) GColor gcColor : background color during focus

Return value None

Details Sets the background color when the effect during focus is GFOCUS_EFFECT_CHG_COLOR.
After the background color is set, the control area is registered as a redraw area.

GCSCheckboxGetFocusColor

Outline Get background color during focus

Syntax
```
GColor GCSCheckboxGetFocusColor(
  GBaseObject     *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Background color during focus

Details Gets the background color when the effect during focus is GFOCUS_EFFECT_CHG_COLOR.

GCSCheckboxSetDisableCaptionColor

Outline Set character color when disabled

Syntax
```
void GCSCheckboxSetDisableCaptionColor(
  GBaseObject    *pCheckBox,
  GColor         gcColor
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) GColor gcColor : character color when disabled

Return value None

Details Sets the character color during disabled control.
After the character color is set, the control area is registered as a redraw area.

GCSCheckboxGetDisableCaptionColor

Outline Get character color when disabled

Syntax
```
GColor GCSCheckboxGetDisableCaptionColor(
  GBaseObject   *pCheckBox
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation

Return value Character color when disabled

Details Gets the character color during disabled control.

GCSCheckboxSetDisableBoxColor

Outline Set box color when disabled

Syntax
```
void GCSCheckboxSetDisableBoxColor(
  GBaseObject   *pCheckBox,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pCheckBox : object for operation
(i) GColor gcColor : box color

Return value None

Details Sets the box color during disabled control.
After the box color is set, the control area is registered as a redraw area.

GCSCheckboxGetDisableBoxColor

Outline Get box color when disabled

Syntax
```
GColor GCSCheckboxGetDisableBoxColor(
  GBaseObject   *pCheckBox
)
```

Argument (i) GBaseObject *pCheckBox : object for operation

Return value Box color when disabled

Details Gets the box color during disabled control.

5.2.8 RadioButton

GCSRadiobuttonSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSRadiobuttonSetBorderID(GBaseObject **pRadioButton,* unsigned short *usID*)
Argument	(i) GBaseObject *pRadioButton : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSRadiobuttonGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSRadiobuttonGetBorderID(GBaseObject **pRadioButton*)
Argument	(i) GBaseObject *pRadioButton : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID setting.

GCSRadiobuttonSetFontID

Outline Set font resource ID

Syntax
```
void GCSRadiobuttonSetFontID(
  GBaseObject     *pRadioButton,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After font resource ID is set, the control area is registered as a redraw area.

GCSRadiobuttonGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSRadiobuttonGetFontID(
  GBaseObject  *pRadioButton
)
```

Argument (i) GBaseObject *pRadioButton : object for operation

Return value font resource ID

Details Gets the font resource ID.

GCSRadiobuttonSetStringID

Outline Set caption character string resource ID

Syntax
```
void GCSRadiobuttonSetStringID(
  GBaseObject     *pRadioButton,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) unsigned short usID : caption character string resource ID

Return value None

Details Sets the caption character string resource ID.
After the caption character string resource ID is set, the control area is registered as a redraw area.

GCSRadiobuttonGetStringID

Outline Get caption character string resource ID

Syntax
```
unsigned short GCSRadiobuttonGetStringID(
  GBaseObject  *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value Caption character string resource ID

Details Gets the caption character string resource ID.

GCSRadiobuttonSetCaption

Outline Set caption information

Syntax
```
void GCSRadiobuttonSetCaption(
  GBaseObject  *pRadioButton,
  GCaption     *pCaption
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets the pCaption in caption information.
All except character color are disabled.
After the caption information is set, the control area is registered as a redraw area.

GCSRadiobuttonGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSRadiobuttonGetCaption(
  GBaseObject  *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.
All settings except character color are disabled.

GCSRadiobuttonSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSRadiobuttonSetFocusEffect(
  GBaseObject   *pRadioButton,
  unsigned char  ucFocusEffect
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
After the effect is set, the control area is registered as a redraw area.

GCSRadiobuttonGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSRadiobuttonGetFocusEffect(
  GBaseObject   * pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSRadiobuttonSetStatus

Outline Set object status

Syntax
```
void GCSRadiobuttonSetStatus(
  GBaseObject    *pRadioButton,
  unsigned char   ucStatus
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) unsigned char ucStatus : set object status

Return value None

Details Sets the GCRadioButton object status.
One of the following values is set for ucStatus:
GRADIO_STATE_OFF : OFF status
GRADIO_STATE_ON : ON status
GRADIO_STATE_OFFSELECT : OFF selection status
When GRADIO_STATE_ON is set, other controls within the same screen are set to OFF.

GCSRadiobuttonGetStatus

Outline Get object status

Syntax
```
unsigned char GCSRadiobuttonGetStatus(
  GBaseObject   *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value
GRADIO_STATE_OFF : OFF status
GRADIO_STATE_ON : ON status
GRADIO_STATE_OFFSELECT : OFF selection status
Besides the above : undefined status

Details Gets the GCRadioButton object status.

GCSRadiobuttonSetBoxSize

Outline Set box size

Syntax
```
void GCSRadiobuttonSetBoxSize(
  GBaseObject     *pRadioButton,
  unsigned short  ucBoxSize
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) unsigned short ucBoxSize : check box size

Return value None

Details Sets the box size.
After the box size is set, the control area is registered as a redraw area.

GCSRadiobuttonGetBoxSize

Outline Get box size

Syntax
```
unsigned short GCSRadiobuttonGetBoxSize(
  GBaseObject  *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value Box size

Details Gets the box size settings.

GCSRadiobuttonSetBoxColor

Outline	Set box color
Syntax	void GCSRadiobuttonSetBoxColor(GBaseObject *pRadioButton, GColor gcColor)
Argument	(i) GBaseObject *pRadioButton : object for operation (i) GColor gcColor : box color
Return value	None
Details	Sets the box color. After the box color is set, the control area is registered as a redraw area.

GCSRadiobuttonGetBoxColor

Outline	Get box color
Syntax	GColor GCSRadiobuttonGetBoxColor(GBaseObject *pRadioButton)
Argument	(i) GBaseObject *pRadioButton : object for operation
Return value	Box color
Details	Gets the box color.

GCSRadiobuttonSetFocusColor

Outline Set background color during focus

Syntax
```
void GCSRadiobuttonSetFocusColor(
  GBaseObject  *pRadioButton,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) GColor gcColor : background color during focus

Return value None

Details Sets the background color when the effect during focus is GFOCUS_EFFECT_CHG_COLOR.
After the background color is set, the control area is registered as a redraw area.

GCSRadiobuttonGetFocusColor

Outline Get background color during focus

Syntax
```
GColor GCSRadiobuttonGetFocusColor(
  GBaseObject  *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value Background color during focus

Details Gets the background color when the effect during focus is GFOCUS_EFFECT_CHG_COLOR.

GCSRadiobuttonSetDisableCaptionColor

Outline Set character color when disabled

Syntax
```
void GCSRadiobuttonSetDisableCaptionColor(
  GBaseObject  *pRadioButton,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) GColor gcColor : character color when disabled

Return value None

Details Sets the character color during disabled control.
After the character color is set, the control area is registered as a redraw area.

GCSRadiobuttonGetDisableCaptionColor

Outline Get character color when disabled

Syntax
```
GColor GCSRadiobuttonGetDisableCaptionColor(
  GBaseObject  *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value Character color when disabled

Details Gets the character color during disabled control.

GCSRadiobuttonSetDisableBoxColor

Outline Set box color when disabled

Syntax
```
void GCSRadiobuttonSetDisableBoxColor(
  GBaseObject  *pRadioButton,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
(i) GColor gcColor : box color

Return value None

Details Sets the box color during disabled control.
After the box color is set, the control area is registered as a redraw area.

GCSRadiobuttonGetDisableBoxColor

Outline Get box color when disabled

Syntax
```
GColor GCSRadiobuttonGetDisableBoxColor(
  GBaseObject  *pRadioButton
)
```

Argument (i) GBaseObject *pRadioButton : object for operation

Return value Box color when disabled

Details Gets the box color during disabled control.

GCSRadiobuttonSetRadioGroup

Outline Set radio group No.

Syntax
```
void GCSRadiobuttonSetRadioGroup(
  GBaseObject     *pRadioButton,
  unsigned short  usRadioGroup
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation
unsigned short usRadioGroup : radio group No.

Return value None

Details Sets the group No. for radio buttons.
The radio button control in the screen having the same group No. carries out an exclusion operation.

GCSRadiobuttonGetRadioGroup

Outline Get radio group No.

Syntax
```
unsigned short GCSRadiobuttonGetRadioGroup(
  GBaseObject  *pRadioButton
)
```

Argument
(i) GBaseObject *pRadioButton : object for operation

Return value Radio group No.

Details Gets the group No. for radio buttons.

5.2.9 ProgressBar

GCSProgressbarSetBackGroundDesign

Outline	Set background design
Syntax	void GCSProgressbarSetBackGroundDesign(GBaseObject **pProgressBar,* GDesign **pDesign*)
Argument	(i) GBaseObject *pProgressBar : object for operation (i) GDesign* pDesign : design
Return value	None
Details	Sets the background design for the progress bar. After the background design is set, the control area is registered as a redraw area.

GCSProgressbarGetBackGroundDesign

Outline	Get background design
Syntax	GDesign* GCSProgressbarGetBackGroundDesign(GBaseObject **pProgressBar*)
Argument	(i) GBaseObject *pProgressBar : object for operation
Return value	Design
Details	Gets the background design for the progress bar.

GCSProgressbarSetBarDesign

Outline	Set bar design
Syntax	void GCSProgressbarSetBarDesign(GBaseObject *pProgressBar, GDesign *pDesign)
Argument	(i) GBaseObject *pProgressBar : object for operation (i) GDesign* pDesign : design
Return value	None
Details	Sets the design for the progress bar. After the bar design is set, the control area is registered as a redraw area.

GCSProgressbarGetBarDesign

Outline	Get bar design
Syntax	GDesign* GCSProgressbarGetBarDesign(GBaseObject *pProgressBar)
Argument	(i) GBaseObject *pProgressBar : object for operation
Return value	Pointer to design
Details	Gets the design for the progress bar.

GCSProgressbarSetRange

Outline Set progress bar range

Syntax
```
void GCSProgressbarSetRange(
  GBaseObject  *pProgressBar,
  short        nMin,
  short        nMax
)
```

Argument

(i) GBaseObject *pProgressBar	: object for operation
(i) short lMin	: minimum value
(i) short lMax	: maximum value

Return value None

Details Sets the range for the progress bar.
When the smallest value and largest value settings are inverted, they are reversed.

GCSProgressbarGetRange

Outline Get progress bar range

Syntax
```
void GCSProgressbarGetRange(
  GBaseObject  *pProgressBar,
  short        *pMin,
  short        *pMax
)
```

Argument

(i) GBaseObject *pProgressBar	: object for operation
(o) short *lMin	: variable storing smallest value
(o) short *lMax	: variable storing largest value

Return value None

Details Gets the range for the progress bar.

GCSProgressbarSetDirection

Outline

Set fill direction

Syntax

```
void GCSProgressbarSetDirection(
  GBaseObject   *pProgressBar,
  unsigned char  ucDirection
)
```

Argument

(i) GBaseObject *pProgressBar : object for operation
(i) unsigned char ucDirection : fill direction

Return value

None

Details

Sets the fill direction for the progress bar.
One of the followings is set for ucDirection:
GPROG_DIR_LEFT2RIGHT : left to right
GPROG_DIR_RIGHT2LEFT : right to left
GPROG_DIR_TOP2BOTTOM : top to bottom
GPROG_DIR_BOTTOM2TOP : bottom to top
When a value outside of the range is set, there is no forced change of the setting.

GCSProgressbarGetDirection

Outline

Get fill direction

Syntax

```
unsigned char GCSProgressbarGetDirection(
  GBaseObject  *pProgressBar
)
```

Argument

(i) GBaseObject *pProgressBar : object for operation

Return value

GPROG_DIR_LEFT2RIGHT	: left to right
GPROG_DIR_RIGHT2LEFT	: right to left
GPROG_DIR_TOP2BOTTOM	: top to bottom
GPROG_DIR_BOTTOM2TOP	: bottom to top
Besides the above	: illegal display setting

Details

Gets the fill direction setting.

GCSProgressbarGetValue

Outline Get current progress bar value

Syntax
```
short GCSProgressbarGetValue(
  GBaseObject  *pProgressBar
)
```

Argument (i) GBaseObject *pProgressBar : object for operation

Return value Current value

Details Gets the current value for the progress bar.

GCSProgressbarSetValue

Outline Set current progress bar value

Syntax
```
void GCSProgressbarSetValue(
  GBaseObject  *pProgressBar,
  long         nValue
)
```

Argument (i) GBaseObject *pProgressBar : object for operation
(i) long lValue : current value

Return value None

Details Sets the current value for the progress bar.
The permissible range for the setting is the range specified by SetRange, and values outside of the range become the maximum value or the minimum value.

5.2.10 HtmlBrowser

GCSHtmlbrowserSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSHtmlbrowserSetBorderID(GBaseObject **pHtmlBrowser,* unsigned short *usID*)
Argument	(i) GBaseObject *pHtmlBrowser : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSHtmlbrowserGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSHtmlbrowserGetBorderID(GBaseObject **pHtmlBrowser*)
Argument	(i) GBaseObject *pHtmlBrowser : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID setting.

GCSHtmlbrowserSetFontID

Outline Set font resource ID

Syntax
```
void GCSHtmlbrowserSetFontID(
  GBaseObject     *pHtmlBrowser,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSHtmlbrowserGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSHtmlbrowserGetFontID(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Font the resource ID

Details Gets the font resource ID.

GCSHtmlbrowserSetScrollBarWidth

Outline Set scroll bar width

Syntax
```
void GCSHtmlbrowserSetScrollBarWidth(
  GBaseObject     *pHtmlBrowser,
  unsigned char   ucScroolBarWidth
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) unsigned char ucScrollBarWidth : scroll bar width

Return value None

Details Sets the scroll bar width.
After the scroll bar width is set, the control area is registered as a redraw area.

GCSHtmlbrowserGetScrollBarWidth

Outline Get scroll bar width

Syntax
```
unsigned char GCSHtmlbrowserGetScrollBarWidth(
  GBaseObject   *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Scroll bar width

Details Gets the scroll bar width.

GCSHtmlbrowserSetTextColor

Outline Set text display color

Syntax
```
void GCSHtmlbrowserSetTextColor(
  GBaseObject   *pHtmlBrowser,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) GColor gcColor : color code for text display color

Return value None

Details Sets the text display color.
After the text display color is set, the control area is registered as a redraw area.

GCSHtmlbrowserGetTextColor

Outline Get text display color

Syntax
```
GColor GCSHtmlbrowserGetTextColor(
  GBaseObject   *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Color code for text display color

Details Gets the text display color.

GCSHtmlbrowserSetBackColor

Outline	Set background color
Syntax	void GCSHtmlbrowserSetBackColor(GBaseObject **pHtmlBrowser,* GColor *gcColor*)
Argument	(i) GBaseObject *pHtmlBrowser : object for operation (i) GColor gcColor : color code for background color
Return value	None
Details	Sets the background color. After the background color is set, the control area is registered as a redraw area.

GCSHtmlbrowserGetBackColor

Outline	Get background color
Syntax	GColor GCSHtmlbrowserGetBackColor(GBaseObject **pHtmlBrowser*)
Argument	(i) GBaseObject *pHtmlBrowser : object for operation
Return value	Color code for background color
Details	Gets the background color.

GCSHtmlbrowserSetLinkColor

Outline Set link color

Syntax
```
void GCSHtmlbrowserSetLinkColor(
  GBaseObject  *pHtmlBrowser,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) GColor gcColor : color code for link color

Return value None

Details Sets the link color.
After the link color is set, the control area is registered as a redraw area.

GCSHtmlbrowserGetLinkColor

Outline Get link color

Syntax
```
GColor GCSHtmlbrowserGetLinkColor(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Color code for link color

Details Gets the link color.

GCSHtmlbrowserSetHtmlFileName

Outline Set HTML file name

Syntax
```
void GCSHtmlbrowserSetHtmlFileName(
  GBaseObject   *pHtmlBrowser,
  GTCHAR        *FileName
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) GTCHAR *FileName : HTML file name

Return value None

Details Sets the HTML file name.
After the HTML file name is set, the read complete flag is cleared to 0 for reading again.

GCSHtmlbrowserGetHtmlFileName

Outline Get HTML file name

Syntax
```
GTCHAR* GCSHtmlbrowserGetHtmlFileName(
  GBaseObject   *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value HTML file name

Details Gets the HTML file name.

GCSHtmlbrowserSetHScrollPosition

Outline Set horizontal scroll position

Syntax
```
void GCSHtmlbrowserSetHScrollPosition(
  GBaseObject  *pHtmlBrowser,
  long         lPosition
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) long lPosition : horizontal scroll position

Return value None

Details Sets the position of the horizontal scroll incorporated in the browser control specified by pHtmlBrowser in dots. This function cannot be set when the horizontal scroll bar is not displayed. In addition, when attempting to set a negative value, 0 will be set. When attempting to set a value that exceeds (maximum value – page size + 1), (maximum value – page size + 1) will be set.

GCSHtmlbrowserGetHScrollPosition

Outline Get horizontal scroll position

Syntax
```
long GCSHtmlbrowserGetHScrollPosition(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Current horizontal scroll position

Details Gets the position of the horizontal scroll incorporated in the browser control specified by pHtmlBrowser in dots.

GCSHtmlbrowserSetVScrollPosition

Outline Set vertical scroll position

Syntax
```
void GCSHtmlbrowserSetVScrollPosition(
  GBaseObject  *pHtmlBrowser,
  long         lPosition
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) long lPosition : vertical scroll position

Return value None

Details Sets the position of the vertical scroll incorporated in the browser control specified by pHtmlBrowser in dots. This function cannot be set when the vertical scroll bar is not displayed. In addition, when attempting to set a negative value, 0 will be set. When attempting to set a value that exceeds (maximum value – page size + 1), (maximum value – page size + 1) will be set.

GCSHtmlbrowserGetVScrollPosition

Outline Get vertical scroll position

Syntax
```
long GCSHtmlbrowserGetVScrollPosition(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Current vertical scroll position

Details Gets the position of the vertical scroll incorporated in the browser control specified by pHtmlBrowser in dots.

GCSHtmlbrowserSetLineScrollWidth

Outline Set width of one line scroll for horizontal scroll

Syntax
```
void GCSHtmlbrowserSetLineScrollWidth(
  GBaseObject  *pHtmlBrowser,
  long         lWidth
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) long lWidth : scroll size

Return value None

Details Sets the width of one line scroll for the horizontal scroll bar incorporated in the browser control specified by pHtmlBrowser. The scroll size that can be set is from 0 to 2560. When attempting to set a negative value, 0 will be set. When attempting to set a value that exceeds 2560, 2560 will be set.

GCSHtmlbrowserGetLineScrollWidth

Outline Get width of one line scroll for horizontal scroll

Syntax
```
long GCSHtmlbrowserGetLineScrollWidth(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Width of one line scroll for horizontal scroll

Details Gets the width of one line scroll for the horizontal scroll bar incorporated in the browser control specified by pHtmlBrowser.

GCSHtmlbrowserSetLineScrollHeight

Outline Set width of one line scroll for vertical scroll

Syntax
```
void GCSHtmlbrowserSetLineScrollHeight(
  GBaseObject  *pHtmlBrowser,
  long         lWidth
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) long lWidth : scroll size

Return value None

Details Sets the width of one line scroll for the vertical scroll bar incorporated in the browser control specified by pHtmlBrowser. The scroll size that can be set is from 0 to 1920. When attempting to set a negative value, 0 will be set. When attempting to set a value that exceeds 1920, 1920 will be set.

GCSHtmlbrowserGetLineScrollHeight

Outline Get width of one line scroll for vertical scroll

Syntax
```
long GCSHtmlbrowserGetLineScrollHeight(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Width of one line scroll for vertical scroll

Details Gets the width of one line scroll for the vertical scroll bar incorporated in the browser control specified by pHtmlBrowser.

GCSHtmlbrowserSetPageScrollWidth

Outline Set width of page scroll for horizontal scroll

Syntax
```
void GCSHtmlbrowserSetPageScrollWidth(
  GBaseObject  *pHtmlBrowser,
  long         lWidth
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) long lWidth : scroll size

Return value None

Details Sets the width of the page scroll for the horizontal scroll bar incorporated in the browser control specified by pHtmlBrowser in dots. The scroll size that can be set is from 0 to the width of the Html file. When attempting to set a negative value, 0 will be set. When attempting to set a value that exceeds the width of the Html file, the width of the Html file will be set.

GCSHtmlbrowserGetPageScrollWidth

Outline Get width of page scroll for horizontal scroll

Syntax
```
long GCSHtmlbrowserGetPageScrollWidth(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Width of page scroll for horizontal scroll

Details Gets the width of the page scroll for the horizontal scroll bar incorporated in the browser control specified by pHtmlBrowser.

GCSHtmlbrowserSetPageScrollHeight

Outline Set width of page scroll for vertical scroll

Syntax
```
void GCSHtmlbrowserSetPageScrollHeight(
  GBaseObject  *pHtmlBrowser,
  long         lHeight
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation
(i) long lHeight : scroll size

Return value None

Details Sets the width of the page scroll for the vertical scroll bar incorporated in the browser control specified by pHtmlBrowser in dots. The scroll size that can be set is from 0 to the height of the Html file. When attempting to set a negative value, 0 will be set. When attempting to set a value that exceeds the height of the Html file, the height of the Html file will be set.

GCSHtmlbrowserGetPageScrollHeight

Outline Get width of page scroll for vertical scroll

Syntax
```
long GCSHtmlbrowserGetPageScrollHeight(
  GBaseObject  *pHtmlBrowser
)
```

Argument
(i) GBaseObject *pHtmlBrowser : object for operation

Return value Width of page scroll for vertical scroll

Details Gets the width of the page scroll for the vertical scroll bar incorporated in the browser control specified by pHtmlBrowser.

GCSHtmlbrowserGetHtmlWidth

Outline	Get total width of HTML file
Syntax	long GCSHtmlbrowserGetHtmlWidth(GBaseObject *pHtmlBrowser)
Argument	(i) GBaseObject *pHtmlBrowser : object for operation
Return value	Total width of HTML file
Details	Gets the total width of the HTML file incorporated in the browser control specified by pHtmlBrowser in dots.

GCSHtmlbrowserGetHtmlHeight

Outline	Get total height of HTML file
Syntax	long GCSHtmlbrowserGetHtmlHeight(GBaseObject *pHtmlBrowser)
Argument	(i) GBaseObject *pHtmlBrowser : object for operation
Return value	Total height of HTML file
Details	Gets the total height of the HTML file incorporated in the browser control specified by pHtmlBrowser in dots.

5.2.11 ScrollBar

GCSScrollbarexSetDisplay

Outline	Set display type
Syntax	void GCSScrollbarexSetDisplay(GBaseObject **pScrollBarEx,* unsigned char *ucDisplay*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation (i) unsigned char ucDisplay : display type
Return value	None
Details	Sets the display type. One of the following values is set for ucDisplay: GSBAR_DISP_NORMAL : normal display GSBAR_DISP_IMAGE : image display When a value outside of the range is set, there is no forced change of the setting. After the display type is set, the control area is registered as a redraw area.

GCSScrollbarexGetDisplay

Outline	Get display type
Syntax	unsigned char GCSScrollbarexGetDisplay(GBaseObject **pScrollBarEx*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	GSBAR_DISP_NORMAL : normal display GSBAR_DISP_IMAGE : image display Besides the above : illegal display setting
Details	Gets the display type setting. When a setting outside the range is made, that setting value is returned.

GCSScrollbarexSetDirection

Outline Set scroll bar direction

Syntax
```
void GCSScrollbarexSetDirection(
  GBaseObject    *pScrollBarEx,
  unsigned char  ucDirection
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) unsigned char ucDirection : scroll bar direction

Return value None

Details Sets the scroll bar direction.
One of the following values is set for ucDirection:
GSBAR_VERTICAL : vertical direction
GSBAR_HORIZONTAL : horizontal direction
When a value outside of the range is set, there is no forced change of the setting.
After the scroll bar direction is set, the control area is registered as a redraw area.

GCSScrollbarexGetDirection

Outline Get scroll bar direction

Syntax
```
unsigned char GCSScrollbarexGetDirection(
  GBaseObject  *pScrollBarEx
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation

Return value
GSBAR_VERTICAL : vertical direction
GSBAR_HORIZONTAL : horizontal direction
Besides the above : illegal display setting

Details Gets the scroll bar direction setting.
When a setting outside the range is made, that setting value is returned.

GCSScrollbarexSetPageSize

Outline	Set page size
Syntax	void GCSScrollbarexSetPageSize(GBaseObject **pScrollBarEx,* unsigned long *ulPageSize*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation (i) unsigned long ulPageSize : page size
Return value	None
Details	Sets the page size. After the page size is set, the control area is registered as a redraw area.

GCSScrollbarexGetPageSize

Outline	Get page size
Syntax	unsigned long GCSScrollbarexGetPageSize(GBaseObject **pScrollBarEx*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Page size
Details	Gets the page size.

GCSScrollbarexSetRange

Outline Set scroll bar range

Syntax
```
void GCSScrollbarexSetRange(
  GBaseObject  *pScrollBar,
  long         lMin,
  long         lMax
)
```

Argument

(i) GBaseObject *pScrollBar	: object for operation
(i) long lMin	: minimum value
(i) long lMax	: maximum value

Return value None

Details Sets the range for the scroll bar.
When the smallest value and largest value settings are inverted, they are reversed.

GCSScrollbarexGetRange

Outline Get scroll bar range

Syntax
```
void GCSScrollbarexGetRange(
  GBaseObject  *pScrollBar,
  long         *pMin,
  long         *pMax
)
```

Argument

(i) GBaseObject *pScrollBar	: object for operation
(i) long *pMin	: pointer to variable storing smallest value
(i) long *pMax	: pointer to variable storing largest value

Return value None

Details Gets the range for the scroll bar.

GCSScrollbarexSetScrollArrowColor

Outline Set scroll arrow color

Syntax
```
void GCSScrollbarexSetScrollArrowColor(
  GBaseObject  *pScrollBarEx,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) GColor gcColor : color code for scroll arrow color

Return value None

Details Sets the scroll bar arrow color.
After the scroll bar arrow color is set, the control area is registered as a redraw area.

GCSScrollbarexGetScrollArrowColor

Outline Get scroll arrow color

Syntax
```
GColor GCSScrollbarexGetScrollArrowColor(
  GBaseObject  *pScrollBarEx
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation

Return value Color code

Details Gets the scroll bar arrow color.

GCSScrollbarexSetScrollPosition

Outline Set scroll bar current position

Syntax
```
void GCSScrollbarexSetScrollPosition(
  GBaseObject   *pScrollBar,
  long          lPosition
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation
(i) long lPosition : current position

Return value None

Details Sets the scroll bar current position.
When a value outside of the maximum/minimum value is set, that setting value is set to maximum/minimum value.

GCSScrollbarexGetScrollPosition

Outline Get scroll bar current position

Syntax
```
long GCSScrollbarexGetScrollPosition(
  GBaseObject   *pScrollBar
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation

Return value Current position

Details Gets the scroll bar current position.

GCSScrollbarexSetPrevBtnOnImgID

Outline Set design (upper) resource ID when button is ON

Syntax
```
void GCSScrollbarexSetPrevBtnOnImgID(
  GBaseObject    *pScrollBarEx,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) unsigned short usID : design (upper) resource ID when button is ON

Return value None

Details Sets the design (upper) resource ID when the button is ON.
After the design resource ID is set, the control area is registered as a redraw area.

GCSScrollbarexSetPrevBtnOffImgID

Outline Set design (upper) resource ID when button is OFF

Syntax
```
void GCSScrollbarexSetPrevBtnOffImgID(
  GBaseObject    *pScrollBarEx,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) unsigned short usID : design (upper) resource ID when button is OFF

Return value None

Details Sets the design (upper) resource ID when the button is OFF.
After the design resource ID is set, the control area is registered as a redraw area.

GCSScrollbarexGetPrevBtnOnImgID

Outline	Get design (upper) resource ID when button is ON
Syntax	unsigned short GCSScrollbarexGetPrevBtnOnImgID(GBaseObject *pScrollBarEx)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Design (upper) resource ID when button is ON
Details	Gets the design (upper) resource ID setting when the button is ON.

GCSScrollbarexGetPrevBtnOffImgID

Outline	Get design (upper) resource ID when button is OFF
Syntax	unsigned short GCSScrollbarexGetPrevBtnOffImgID(GBaseObject *pScrollBarEx)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Design (upper) resource ID when button is OFF
Details	Gets the design (upper) resource ID setting when the button is OFF.

GCSScrollbarexSetNextBtnOnImgID

Outline Set design (lower) resource ID when button is ON

Syntax
```
void GCSScrollbarexSetNextBtnOnImgID(
  GBaseObject    *pScrollBarEx,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) unsigned short usID : design (lower) resource ID when button is ON

Return value None

Details Sets the design (lower) resource ID when the button is ON.
After the design resource ID is set, the control area is registered as a redraw area.

GCSScrollbarexSetNextBtnOffImgID

Outline Set design (lower) resource ID when button is OFF

Syntax
```
void GCSScrollbarexSetNextBtnOffImgID(
  GBaseObject    *pScrollBarEx,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) unsigned short usID : design (lower) resource ID when button is OFF

Return value None

Details Sets the design (lower) resource ID when the button is OFF.
After the design resource ID is set, the control area is registered as a redraw area.

GCSScrollbarexGetNextBtnOnImgID

Outline	Get design (lower) resource ID when button is ON
Syntax	unsigned short GCSScrollbarexGetNextBtnOnImgID(GBaseObject *pScrollBarEx)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Design (lower) resource ID when button is ON
Details	Gets the design (lower) resource ID setting when the button is ON.

GCSScrollbarexGetNextBtnOffImgID

Outline	Get design (lower) resource ID when button is OFF
Syntax	unsigned short GCSScrollbarexGetNextBtnOffImgID(GBaseObject *pScrollBarEx)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Design (lower) resource ID when button is OFF
Details	Gets the design (lower) resource ID setting when the button is OFF.

GCSScrollbarexSetBackGroundColor

Outline Set background color

Syntax
```
void GCSScrollbarexSetBackGroundColor(
  GBaseObject  *pScrollBarEx,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) GColor gcColor : color code for background color

Return value None

Details Sets the background color.
After the background color is set, the control area is registered as a redraw area.

GCSScrollbarexGetBackGroundColor

Outline Get background color

Syntax
```
GColor GCSScrollbarexGetBackGroundColor(
  GBaseObject  *pScrollBarEx
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation

Return value Color code

Details Gets the background color.

GCSScrollbarexSetOutLineBorderID

Outline	Set outline 3D border
Syntax	void GCSScrollbarexSetOutLineBorderID(GBaseObject **pScrollBarEx,* unsigned short *usID*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the outline 3D border.

GCSScrollbarexGetOutLineBorderID

Outline	Get outline 3D border
Syntax	unsigned short GCSScrollbarexGetOutLineBorderID(GBaseObject **pScrollBarEx*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Outline 3D border resource ID
Details	Gets outline 3D border resource ID.

GCSScrollbarexSetPinchWidth

Outline	Set pinch width
Syntax	void GCSScrollbarexSetPinchWidth(GBaseObject **pScrollBarEx,* unsigned short *nWidth*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation (i) unsigned short nWidth : pinch width
Return value	None
Details	Sets the pinch width.

GCSScrollbarexGetPinchWidth

Outline	Get pinch width
Syntax	unsigned short GCSScrollbarexGetPinchWidth(GBaseObject **pScrollBarEx*)
Argument	(i) GBaseObject *pScrollBarEx : object for operation
Return value	Pinch width
Details	Gets the pinch width.

GCSScrollbarexSetPinchColor

Outline Set pinch color

Syntax
```
void GCSScrollbarexSetPinchColor(
  GBaseObject  *pScrollBarEx,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) GColor gcColor : pinch color

Return value None

Details Sets the pinch color.

GCSScrollbarexGetPinchColor

Outline Get pinch color

Syntax
```
GColor GCSScrollbarexGetPinchColor(
  GBaseObject  *pScrollBarEx
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation

Return value Pinch color

Details Gets the pinch color.

GCSScrollbarexSetBarEnableStatus

Outline Set bar display/non-display

Syntax
```
void GCSScrollbarexSetBarEnableStatus(
  GBaseObject  *pScrollBarEx,
  short        nStatus
)
```

Argument
(i) GBaseObject *pScrollBarEx : object for operation
(i) short nStatus : scroll bar display/non-display status
TRUE : display
FALSE : non-display

Return value None

Details Sets the display/non-display for the bar.

GCSScrollbarexGetBarEnableStatus

Outline Get bar display/non-display

Syntax
```
short GCSScrollbarexGetBarEnableStatus(
  GBaseObject  *pScrollBarEx
)
```

Argument (i) GBaseObject *pScrollBarEx : object for operation

Return value
TRUE : display
FALSE : non-display

Details Gets the display/non-display setting for the bar.

GCSScrollbarexSetBorderID

Outline Set 3D border resource ID

Syntax
```
void GCSScrollbarexSetBorderID(
  GBaseObject    *pScrollBar,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation
(i) unsigned short usID : 3D border resource ID

Return value None

Details Sets the 3D border resource ID.
After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSScrollbarexGetBorderID

Outline Get 3D border resource ID

Syntax
```
unsigned short GCSScrollbarexGetBorderID(
  GBaseObject  *pScrollBar
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation

Return value 3D border resource ID

Details Gets the 3D border resource ID setting.

GCSScrollbarexSetScrollButtonColor

Outline Set scroll button color

Syntax
```
void GCSScrollbarexSetScrollButtonColor(
  GBaseObject  *pScrollBar,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation
(i) GColor gcColor : color code for scroll button color

Return value None

Details Sets the scroll button color.
After the scroll button color is set, the control area is registered as a redraw area.

GCSScrollbarexGetScrollButtonColor

Outline Get scroll button color

Syntax
```
GColor GCSScrollbarexGetScrollButtonColor(
  GBaseObject  *pScrollBar
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation

Return value Color code

Details Gets the scroll button color.

GCSScrollbarexSetScrollBarColor

Outline Set scroll bar color

Syntax
```
void GCSScrollbarexSetScrollBarColor(
  GBaseObject  *pScrollBar,
  GColor       gcColor
```

Argument
(i) GBaseObject *pScrollBar : object for operation
(i) GColor gcColor : color code for scroll bar color

Return value None

Details Sets the scroll bar color.
After the scroll bar color is set, the control area is registered as a redraw area.

GCSScrollbarexGetScrollBarColor

Outline Get scroll bar color

Syntax
```
GColor GCSScrollbarexGetScrollBarColor(
  GBaseObject  *pScrollBar
)
```

Argument
(i) GBaseObject *pScrollBar : object for operation

Return value Color code

Details Gets the scroll bar color.

GCSScrollbarexSetOutLineColor

Outline	Set color of outline
Syntax	void GCSScrollbarexSetOutLineColor(GBaseObject *pScrollBarEx, GColor gcColor)
Argument	(i) GBaseObject *pScrollBar : object for operation (i) GColor gcColor : color of outline
Return value	None
Details	Sets the color of outline.

GCSScrollbarexGetOutLineColor

Outline	Get color of outline
Syntax	GColor GCSScrollbarexGetOutLineColor(GBaseObject *pScrollBarEx)
Argument	(i) GBaseObject *pScrollBar : object for operation
Return value	Color of outline
Details	Gets the color of outline.

5.2.12 Edit

GCSEditSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSEditSetBorderID(GBaseObject **pEdit,* unsigned short *usID*)
Argument	(i) GBaseObject *pEdit : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSEditGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSEditGetBorderID(GBaseObject **pEdit*)
Argument	(i) GBaseObject *pEdit : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID setting.

GCSEditSetFontID

Outline Set font resource ID

Syntax
```
void GCSEditSetFontID(
  GBaseObject    *pEdit,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSEditGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSEditGetFontID(
  GBaseObject  *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSEditAddString

Outline	Add character string
Syntax	short GCSEditAddString(GBaseObject *pEdit, GTCHAR *pszString)
Argument	(i) GBaseObject *pEdit : object for operation (i) char *pszString : character string to be added
Return value	FALSE : processing failed TRUE : processing succeeded
Details	Adds the character string to the end.

GCSEditSetString

Outline Set display character string

Syntax
```
short GCSEditSetString(
  GBaseObject      *pEdit,
  const GTCHAR   * pString
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) const char *pString : display character string

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the display character string.

GCSEditGetString

Outline Get display character string

Syntax
```
short GCSEditGetString(
  GBaseObject   *pEdit,
  GTCHAR        *pString,
  long          lMaxCount
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(o) char *pString : buffer where display character string is stored
(i) long lMaxCount : buffer size

Return value
FALSE : acquisition failed
TRUE : acquisition succeeded

Details Stores the display character string in pString.

GCSEditSetTextColor

Outline	Set character display color
Syntax	void GCSEditSetTextColor(GBaseObject *pEdit, GColor gcColor)
Argument	(i) GBaseObject *pEdit : object for operation (i) GColor gcColor : color code for character display color
Return value	None
Details	Sets the character display color.

GCSEditGetTextColor

Outline	Get character display color
Syntax	GColor GCSEditGetTextColor(GBaseObject *pEdit)
Argument	(i) GBaseObject *pEdit : object for operation
Return value	Color code for character display color
Details	Gets the character display color.

GCSEditSetBackColor

Outline Set background color

Syntax
```
void GCSEditSetBackColor(
  GBaseObject   *pEdit,
  GColor        gcColor
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) GColor gcColor : color code for background color

Return value None

Details Sets the background color.

GCSEditGetBackColor

Outline Get background color

Syntax
```
GColor GCSEditGetBackColor(
  GBaseObject   *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value Color code for background color

Details Gets the background color.

GCSEditSetBrush

Outline Set brush

Syntax
```
void GCSEditSetBrush(
  GBaseObject  *pEdit,
  GBrush       *pBrush
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) GBrush *pBrush : brush structure

Return value
TRUE : processing succeeded
FALSE : processing failed

Details Sets the brush.

GCSEditDeleteString

Outline Delete character string

Syntax
```
void GCSEditDeleteString(
  GBaseObject  *pEdit
)
```

Argument (i) GBaseObject *pEdit : object for operation

Return value None

Details Deletes all the character strings.

GCSEditAddLine

Outline Add line

Syntax
```
short GCSEditAddLine(
  GBaseObject   *pEdit,
  GTCHAR        *pszLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) GTCHAR *pszLine : line to be added

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Adds the character string as a new line at the end.

GCSEditInsertLine

Outline Insert line

Syntax
```
short GCSEditInsertLine(
  GBaseObject     *pEdit,
  GTCHAR          *pszLine,
  unsigned long   ulLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) GTCHAR *pszLine : line to be inserted
(i) unsigned long ulLine : line No.

Return value
0 : processing failed
Other than 0 : number of lines inserted

Details Inserts the character string as a new line at specified position.

GCSEditDeleteLine

Outline Delete line

Syntax
```
short GCSEditDeleteLine(
  GBaseObject   *pEdit,
  unsigned long  ulLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulLine : line No.

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Deletes the specified line.

GCSEditGetLineString

Outline Get line character string

Syntax
```
short GCSEditGetLineString(
  GBaseObject   *pEdit,
  unsigned long  ulLine,
  GTCHAR        *pszString,
  long           lMaxCount
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulLine : line No.
(i) GTCHAR *pszString : buffer where character string is stored
(i) long lMaxCount : buffer size

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Gets the character string for the specified line.

GCSEditSetLineFeedCode

Outline Set line feed character string

Syntax
```
void GCSEditSetLineFeedCode(
  GBaseObject     *pEdit,
  unsigned char   ucLineFeedCode
)
```

Argument
(i) GBaseObject *pEdit : object for operation
unsigned short usLineFeedType : line feed character type

Return value None

Details Sets the line feed character string type.

GCSEditGetLineFeedCode

Outline Get line feed character string

Syntax
```
unsigned char GCSEditGetLineFeedCode(
  GBaseObject   *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value
Line feed character string type
GEDIT_LF_TYPE_CRLF : /r/n
GEDIT_LF_TYPE_LF : /n
GEDIT_LF_TYPE_CR : /r

Details Gets the line feed character string type.

GCSEditSetCursor

Outline Set cursor position

Syntax
```
short GCSEditSetCursor(
  GBaseObject   *pEdit,
  unsigned long  ulLine,
  unsigned long  ulIndex
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulLine : line No.
(i) unsigned long ulIndex : specified index

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Sets the cursor position to the specified position.

GCSEditGetCursor

Outline Get cursor position

Syntax
```
short GCSEditGetCursor(
  GBaseObject     *pEdit,
  unsigned long   *pulLine,
  unsigned long   *pulIndex
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long *pulLine : line No.
(i) unsigned long *pulIndex : specified index

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Gets the cursor position.

GCSEditSetForeColor

Outline	Set foreground color
Syntax	void GCSEditSetForeColor(GBaseObject **pEdit,* GColor *gcColor*)
Argument	(i) GBaseObject *pEdit : object for operation (i) GColor gcColor : color code for foreground color
Return value	None
Details	Sets the foreground color. After the foreground color is set, the control area is registered as a redraw area.

GCSEditGetForeColor

Outline	Get foreground color
Syntax	GColor GCSEditGetForeColor(GBaseObject **pEdit*)
Argument	(i) GBaseObject *pEdit : object for operation
Return value	Color code for foreground color
Details	Gets the foreground color.

GCSEditSetFillPattern

Outline Set fill pattern

Syntax
```
void GCSEditSetFillPattern(
  GBaseObject   *pEdit,
  short         nFillPattern
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) short nFillPattern : fill pattern

Return value None

Details Sets the fill pattern.
After the fill pattern is set, the control area is registered as a redraw area.

GCSEditGetFillPattern

Outline Get fill pattern

Syntax
```
short GCSEditGetFillPattern(
  GBaseObject   *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value Fill pattern

Details Gets the fill pattern.

GCSEditSetInsertMode

Outline Set insert/overwrite mode

Syntax
```
void GCSEditSetInsertMode(
  GBaseObject   *pEdit,
  short         nIsInsert
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) short nIsInsert : set insert/overwrite mode
TRUE : insert mode
FALSE : overwrite mode

Return value None

Details Sets the insert/overwrite mode.

GCSEditGetInsertMode

Outline Get insert/overwrite mode

Syntax
```
short GCSEditGetInsertMode(
  GBaseObject   *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value
TRUE : insert mode
FALSE : overwrite mode

Details Gets the insert/overwrite mode.

GCSEditSetLineBrush

Outline Set line brush

Syntax
```
short GCSEditSetLineBrush(
  GBaseObject    *pEdit,
  GBrush         *pBrush,
  unsigned long  ulLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulLine : line No.
(i) GBrush *pBrush : brush structure

Return value None

Details Sets the brush for the specified line.

GCSEditGetLineBrush

Outline Get line brush

Syntax
```
short GCSEditGetLineBrush(
  GBaseObject    *pEdit,
  GBrush         *pBrush,
  unsigned long  ulLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulLine : line No.
(i) GBrush *pBrush : brush structure

Return value
TRUE : processing succeeded
FALSE : processing failed

Details Gets the brush for the specified line.

GCSEditSetLineTextColor

Outline Set line text display color

Syntax
```
short GCSEditSetLineTextColor(
  GBaseObject   *pEdit,
  GColor        gcLineColor,
  unsigned long ulLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulLine : line No.
(i) GColor gcColor : color code for text display color

Return value None

Details Sets the text display color for the specified line.

GCSEditGetLineTextColor

Outline Get line text display color

Syntax
```
GColor GCSEditGetLineTextColor(
  GBaseObject   *pEdit,
  unsigned long ulLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value Color code for text display color for specified line

Details Gets the text display color for the specified line.

GCSEditGetLength	

Outline	Get character length
Syntax	unsigned long GCSEditGetLength(GBaseObject *pEdit)
Argument	(i) GBaseObject *pEdit : object for operation
Return value	Number of characters
Details	Gets the current number of characters.

GCSEditFind

Outline Search forward

Syntax

```
short GCSEditFind(
  GBaseObject     *pEdit,
  GTCHAR          *pszString,
  unsigned long   ulStartLine,
  unsigned long   ulStartIndex,
  unsigned long   *pulEditFindLine,
  unsigned long   *pulEditFindPos
)
```

Argument

(i) GBaseObject *pEdit	: object for operation
(i) GTCHAR *pszString	: character string for search
(i) unsigned long ulStartLine	: starting line for search
(i) unsigned long ulStartIndex	: starting position for search
(i) unsigned long *pulFindLine	: search result line
(i) unsigned long *pulFindIndex	: search result position

Return value

FALSE	: search failed
TRUE	: search succeeded

Details Searches the character string for search from the starting position for search toward the end. Returns the position of the character string found first.

GCSEditReverseFind

Outline

Search backward

Syntax

```
short GCSEditReverseFind(
  GBaseObject     *pEdit,
  GTCHAR          *pszString,
  unsigned long   ulStartLine,
  unsigned long   ulStartIndex,
  unsigned long   *pulFindLine,
  unsigned long   *pulFindPos
)
```

Argument

(i) GBaseObject *pEdit	: object for operation
(i) GTCHAR *pszString	: character string for search
(i) unsigned long ulStartLine	: starting line for search
(i) unsigned long ulStartIndex	: starting position for search
(i) unsigned long *pulFindLine	: search results line
(i) unsigned long *pulFindIndex	: search result position

Return value

FALSE	: search failed
TRUE	: search succeeded

Details

Searches the character string for search from the starting position for search toward the beginning. Returns the position of the character string found first.

GCSEditReplace

Outline

Replace

Syntax

```
short GCSEditReplace(
  GBaseObject   *pEdit,
  GTCHAR        *pszString,
  unsigned long  ulStartLine,
  unsigned long  ulStartIndex,
  unsigned long  ulSize
)
```

Argument

(i) GBaseObject *pEdit	: object for operation
(i) GTCHAR *pszString	: replacement character string
(i) unsigned long ulStartLine	: starting line for replacement
(i) unsigned long ulStartIndex	: starting position for replacement
(i) unsigned long ulSize	: replacement size

Return value

FALSE	: replacement failed
TRUE	: replacement succeeded

Details

Replaces a character string with a character string of the replacement size starting from the replacement starting position.

GCSEditReplaceAll

Outline Replace all

Syntax

```
unsigned long GCSEditReplaceAll(
  GBaseObject  *pEdit,
  GTCHAR       *pszReplace,
  GTCHAR       *pszFind
)
```

Argument

(i) GBaseObject *pEdit	: object for operation
(i) GTCHAR *pszReplace	: replacement character string
(i) GTCHAR *pszFind	: character string for search

Return value Number of replacements

Details Searches the character string for search starting from the character string in the buffer and replaces it with the replacement character string.

GCSEditInsertString

Outline Insert character string

Syntax
```
short GCSEditInsertString(
  GBaseObject     *pEdit,
  GTCHAR          *pszString,
  unsigned long   ulLine,
  unsigned long   ulIndex
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) char *pszString : character string to be inserted
(i) unsigned long ulLine : line for insertion
(i) unsigned long ulIndex : insertion position

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Inserts character string.

GCSEditGetLineCount

Outline Get number of lines

Syntax
```
unsigned long GCSEditGetLineCount(
  GBaseObject   *pEdit
)
```

Argument (i) GBaseObject *pEdit : object for operation

Return value Number of lines

Details Gets the current number of lines.

GCSEditGetBrush

Outline Get brush

Syntax
```
short GCSEditGetBrush(
  GBaseObject  *pEdit,
  GBrush       *pBrush
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) GBrush *pBrush : brush structure

Return value
TRUE : processing succeeded
FALSE : processing failed

Details Gets the brush.

GCSEditRemoveString

Outline Delete character string

Syntax
```
short GCSEditRemoveString(
  GBaseObject  *pEdit,
  unsigned long  ulRemoveSize
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned long ulRemoveSize : size of character string for deletion

Return value
FALSE : processing failed
TRUE : processing succeeded

Details Deletes character string from the end.

GCSEditSetTopLine

Outline Set first line displayed for edit control

Syntax
```
void GCSEditSetTopLine(
  GBaseObject  *pEdit,
  long         lTopLogicalLine
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) long lTopLogicalLine : first line displayed

Return value None

Details Sets the first line displayed.
The range of settings is 0 to the largest line.
When a negative value or a value exceeding the largest line is specified in the first line displayed, no processing is performed.

GCSEditGetTopLine

Outline Get first line displayed for edit control

Syntax
```
long GCSEditGetTopLine(
  GBaseObject  *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value First line displayed

Details Gets the first line displayed.

GCSEditSetDispCursor

Outline Set cursor position to specified posiiton

Syntax
```
short GCSEditSetDispCursor(
  GBaseObject   *pEdit,
  unsigned long  ulDisplayLine,
  unsigned long  ulIndex
)
```

Argument

(i) GBaseObject *pEdit	:	object for operation
(i) unsigned long ulDisplayLine	:	displayed line number
(i) unsigned long ulIndex	:	specified index

Return value

FALSE	:	processing failed
TRUE	:	processing succeeded

Details Sets the cursor position to specified position.

GCSEditGetDispCursor

Outline Get cursor position

Syntax
```
short GCSEditGetDispCursor(
  GBaseObject   *pEdit,
  unsigned long *pulDisplayLine
  unsigned long *pulIndex
)
```

Argument

(i) GBaseObject *pEdit	:	object for operation
(i) unsigned long *pulDisplayLine	:	displayed line number
(i) unsigned long *pulIndex	:	specified index

Return value

FALSE	:	processing failed
TRUE	:	processing succeeded

Details Gets the cursor position.

GCSEditSetLineDisplayMode

Outline	Set "Yes/None" for turn back mode
Syntax	void GCSEditSetLineDisplayMode(GBaseObject **pEdit,* unsigned char *ucFlag*)
Argument	(i) GBaseObject *pEdit : object for operation (i) unsigned char ucFlag : turn back mode
Return value	None
Details	Sets "Yes/None" for turn back mode. One of the following values is set for ucFlag: 0 : without turning back 1 : with turning back

GCSEditGetLineDisplayMode

Outline	Get "Yes/None" for turn back mode
Syntax	unsigned char GCSEditGetLineDisplayMode(GBaseObject **pEdit*)
Argument	(i) GBaseObject *pEdit : object for operation
Return value	0 : without turning back 1 : with turning back
Details	Gets "Yes/None" for turn back mode.

GCSEditSetModifyFlag

Outline Set data modify flag

Syntax
```
void GCSEditSetModifyFlag(
  GBaseObject    *pEdit,
  unsigned char  ucFlag
)
```

Argument
(i) GBaseObject *pEdit : object for operation
(i) unsigned char ucFlag : data modify flag

Return value None

Details Sets the data modify flag.

GCSEditGetModifyFlag

Outline Get data modify flag

Syntax
```
unsigned char GCSEditGetModifyFlag(
  GBaseObject  *pEdit
)
```

Argument
(i) GBaseObject *pEdit : object for operation

Return value None

Details Gets the data modify flag.

GCSEditConvertLogicalLineToDisplayLine

Outline Convert logical line number to display line number

Syntax
```
unsigned long GCSEditConvertLogicalLineToDisplayLine(
  GBaseObject   *pEdit,
  unsigned long  ulLogicalLine,
  unsigned long  ulLColumn
)
```

Argument

(i) GBaseObject *pEdit	:	object for operation
(i) unsigned long ulLogicalLine	:	logical line number
(i) unsigned long ulLColumn	:	position in logical line

Return value Display line number

Details Converts the logical line number to the display line number.

GCSEditConvertDisplayLineToLogicalLine

Outline Convert display line number to logical line number

Syntax
```
unsigned long GCSEditConvertDisplayLineToLogicalLine(
  GBaseObject   *pEdit,
  unsigned long  ulDisplayLine
)
```

Argument

(i) GBaseObject *pEdit	:	object for operation
(i) unsigned long ulDisplayLine	:	display line number

Return value Logical line number

Details Converts the display line number to the logical line number.

GCSEditGetDisplayLineCount	
Outline	Get current number of lines displayed
Syntax	unsigned long GCSEditGetDisplayLineCount(GBaseObject *pEdit)
Argument	(i) GBaseObject *pEdit : object for operation
Return value	Number of lines displayed
Details	Gets the current number of lines displayed.

5.2.13 NCTable

GCSNCTableSetCellForeColor

Outline	Set foreground color of each cell
Syntax	void GCSNCTableSetCellForeColor(GBaseObject **pSelf,* GColor *gcColor,* unsigned short *usLine,* unsigned short *usRow,* unsigned short **pusErrCode*)
Argument	(i) GBaseObject *pSelf : object for operation (i) Gcolor gcColor : color (i) unsigned short usLine : line No. (0 or more) (i) unsigned short usRow : row No. (0 or more) (o) unsigned short *pusErrCode : error code GERR_NOERR : normal GERR_NCTABLE_RANGEOVER : cell No. outside range
Return value	None
Details	Sets the foreground color of each cell.

GCSNCTableSetCellBackColor

Outline Set background color of each cell

Syntax
```
void GCSNCTableSetCellBackColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  usLine,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) Gcolor gcColor : color
(i) unsigned short usLine : line No. (0 or more)
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code
GERR_NOERR : normal
GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Sets the background color of each cell.

GCSNCTableSetCellString

Outline Set character string of each cell

Syntax
```
void GCSNCTableSetCellString(
  GBaseObject     *pSelf,
  const GTCHAR    *pString,
  unsigned short  usLine,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) const GTCHAR *pString : set character string (0 to 128 characters)
(i) unsigned short usLine : line No. (0 or more)
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_MEMORYERROR	: securing memory failed

Return value None

Details Sets the character string of each cell.

GCSNCTableGetCellString

Outline Get character string of each cell

Syntax

```
void GCSNCTableGetCellString(
  GBaseObject     *pSelf,
  GTCHAR          *pString,
  long            lMaxCount,
  unsigned short  usLine,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(o) GTCHAR *pString	: character string to be acquired
(i) long lMaxCount	: number of characters to be acquired
(i) unsigned short usLine	: line No. (0 or more)
(i) unsigned short usRow	: row No. (0 or more)
(o) unsigned short *pusErrCode	: error code
GERR_NOERR	: normal
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_WRONGSTRINGGETSIZE	: character string size to be acquired illegal

Return value None

Details Gets the character string of each cell.

GCSNCTableSetSubCursorNum

Outline Set sub-cursor position

Syntax
```
void GCSNCTableSetSubCursorNum(
  GBaseObject     *pSelf,
  unsigned short  usLine,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short usLine : line No. (0 or more)
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_DISABLESUBCURSOR	: sub-cursor invalid
GERR_NCTABLE_RANGEOVER	: cell No. outside range

Return value None

Details Sets the sub-cursor position.

GCSNCTableGetSubCursorNum

Outline Get sub-cursor position

Syntax
```
void GCSNCTableGetSubCursorNum(
  GBaseObject     *pSelf,
  unsigned short  *uspLine,
  unsigned short  *usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(o) unsigned short *uspLine : line No. (0 or more)
(o) unsigned short *usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code
GERR_NOERR : normal
GERR_NCTABLE_DISABLESUBCURSOR : sub-cursor invalid

Return value None

Details Gets the sub-cursor position.

GCSNCTableSetCellWidth

Outline Set cell width

Syntax

```
void GCSNCTableSetCellWidth(
  GBaseObject     *pSelf,
  unsigned short  usRowNum,
  unsigned short  usWidth,
  unsigned short  usRatioMode,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short usRowNum : row No. (0 or more)
(i) unsigned short usWidth : row width
(i) unsigned short usRatioMode : row proportion mode (0: proportion 1: pixel)
(o) unsigned short *pusErrCode : error code
GERR_NOERR : normal
GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Sets the cell width.

GCSNCTableSetCellHight

Outline Set cell height

Syntax
```
void GCSNCTableSetCellHight(
  GBaseObject      *pSelf,
  unsigned short   usHight,
  unsigned short   *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short usHight : height
(o) unsigned short *pusErrCode : error code
GERR_NOERR : normal

Return value None

Details Sets the cell height.

GCSNCTableGetCellNumFromPoint

Outline Get cell No. including pertinent coordinates

Syntax
```
void GCSNCTableGetCellNumFromPoint(
  GBaseObject     *pSelf,
  GPoint          gptPoint,
  unsigned short  *uspLine,
  unsigned short  *uspRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GPoint gptPoint : coordinates
(o) unsigned short *uspLine : line No. (0 or more)
(o) unsigned short *uspRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code
GERR_NOERR : normal
GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Gets the cell No. including pertinent coordinates.

GCSNCTableSetLineTitleString

Outline Set line title

Syntax
```
void GCSNCTableSetLineTitleString(
  GBaseObject      *pSelf,
  const GTCHAR     *pString,
  unsigned short   usLine,
  unsigned short   *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) const GTCHAR *pString : set character string
(i) unsigned short usLine : line No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_NOTITLEMODE	: title non-display mode
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_MEMORYERROR	: securing memory failed

Return value None

Details Sets the line title.

GCSNCTableGetLineTitleString

Outline Get line title

Syntax
```
void GCSNCTableGetLineTitleString(
  GBaseObject     *pSelf,
  GTCHAR          *pString,
  long            lMaxCount,
  unsigned short  usLine,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(o) GTCHAR *pString : character string to be acquired
(i) long lMaxCount : number of characters to be acquired
(i) unsigned short usLine : line No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_NOTITLEMODE	: title non-display mode
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_MEMORYERROR	: securing memory failed

Return value None

Details Gets the line title.

GCSNCTableSetRowTitleString

Outline Set row title

Syntax
```
void GCSNCTableSetRowTitleString(
  GBaseObject     *pSelf,
  GTCHAR          *pString,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GTCHAR *pString : set character string
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_NOTITLEMODE	: title non-display mode
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_MEMORYERROR	: securing memory failed

Return value None

Details Sets the row title.

GCSNCTableGetRowTitleString

Outline

Get row title

Syntax

```
void GCSNCTableGetRowTitleString(
  GBaseObject     *pSelf,
  GTCHAR          *pString,
  long            lMaxCount,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf	: object for operation	
(o) GTCHAR *pString	: character string to be acquired	
(i) long lMaxCount	: number of characters to be acquired	
(i) unsigned short usRow	: row No. (0 or more)	
(o) unsigned short *pusErrCode	: error code	
GERR_NOERR		: normal
GERR_NCTABLE_NOTITLEMODE		: title non-display mode
GERR_NCTABLE_RANGEOVER		: cell No. outside range
GERR_NCTABLE_WRONGSTRINGGETSIZE		: character string size to be acquired illegal

Return value

None

Details

Gets the row title.

GCSNCTableSetLineTitleStringPosition

Outline Set line title character string position

Syntax
```
void GCSNCTableSetLineTitleStringPosition(
  GBaseObject     *pSelf,
  unsigned char   ucHPosition,
  unsigned char   ucVPosition,
  unsigned short  usLine,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned char ucHPosition : character display position horizontal
(i) unsigned char ucVPosition : character display position vertical
(i) unsigned short usLine : line No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_NOTITLEMODE	: title non-display mode
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_WRONGVALUE	: value outside setting range

Return value None

Details Sets the line title character string position.

GCSNCTableSetRowTitleStringPosition

Outline Set row title character string position

Syntax
```
void GCSNCTableSetRowTitleStringPosition(
  GBaseObject     *pSelf,
  unsigned char   ucHPosition,
  unsigned char   ucVPosition,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned char ucHPosition : character display position horizontal
(i) unsigned char ucVPosition : character display position vertical
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_NOTITLEMODE	: title non-display mode
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_WRONGVALUE	: value outside setting range

Return value None

Details Sets the row title character string position.

GCSNCTableSetCellStringPosition

Outline

Set area part character string position

Syntax

```
void GCSNCTableSetCellStringPosition(
  GBaseObject     *pSelf,
  unsigned char   ucHPosition,
  unsigned char   ucVPosition,
  unsigned short  usLine,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned char ucHPosition : character display position horizontal
(i) unsigned char ucVPosition : character display position vertical
(i) unsigned short usLine : line No. (0 or more)
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_RANGEOVER	: cell No. outside range
GERR_NCTABLE_WRONGVALUE	: value outside setting range

Return value

None

Details

Sets the area part character string position.

GCSNCTableSetBeforeSubCursorNum

Outline Set previous sub-cursor No.

Syntax
```
void GCSNCTableSetBeforeSubCursorNum(
  GBaseObject     *pSelf,
  unsigned short  usLine,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short usLine : line No. (0 or more)
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code

GERR_NOERR	: normal
GERR_NCTABLE_DISABLESUBCURSOR	: sub-cursor invalid
GERR_NCTABLE_RANGEOVER	: cell No. outside range

Return value None

Details Sets the previous sub-cursor No.

GCSNCTableGetBeforeSubCursorNum

Outline Get previous sub-cursor No.

Syntax
```
void GCSNCTableGetBeforeSubCursorNum(
  GBaseObject     *pSelf,
  unsigned short  *pusLine,
  unsigned short  *pusRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short *pusLine : line No. (0 or more)
(i) unsigned short *pusRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code
GERR_NOERR : normal
GERR_NCTABLE_DISABLESUBCURSOR : sub-cursor invalid

Return value None

Details Gets the previous sub-cursor No.

GCSNCTableSetLineTitleForeColor

Outline Set foreground color of line title

Syntax
```
void GCSNCTableSetLineTitleForeColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  usLine,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GColor gcColor : color
(i) unsigned short usLine : line No. (0 or more)
(o) unsigned short *pusErrCode : error code
- GERR_NOERR : normal
- GERR_NCTABLE_NOTITLEMODE : title non-display mode
- GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Sets the foreground color of line title.

GCSNCTableSetLineTitleBackColor

Outline Set background color of line title

Syntax
```
void GCSNCTableSetLineTitleBackColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  usLine,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GColor gcColor : color
(i) unsigned short usLine : line No. (0 or more)
(o) unsigned short *pusErrCode : error code
- GERR_NOERR : normal
- GERR_NCTABLE_NOTITLEMODE : title non-display mode
- GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Sets the background color of line title.

GCSNCTableSetRowTitleForeColor

Outline Set foreground color of row title

Syntax
```
void GCSNCTableSetRowTitleForeColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GColor gcColor : color
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code
- GERR_NOERR : normal
- GERR_NCTABLE_NOTITLEMODE : title non-display mode
- GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Sets the foreground color of row title.

GCSNCTableSetRowTitleBackColor

Outline Set background color of row title

Syntax
```
void GCSNCTableSetRowTitleBackColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  usRow,
  unsigned short  *pusErrCode
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GColor gcColor : color
(i) unsigned short usRow : row No. (0 or more)
(o) unsigned short *pusErrCode : error code
- GERR_NOERR : normal
- GERR_NCTABLE_NOTITLEMODE : title non-display mode
- GERR_NCTABLE_RANGEOVER : cell No. outside range

Return value None

Details Sets the background color of row title.

GCSNCTableSetSubCursorForeColor

Outline Set foreground color of sub-cursor

Syntax
```
void GCSNCTableSetSubCursorForeColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  *pusErrCode
)
```

Argument
```
(i) GBaseObject *pSelf                : object for operation
(i) GColor gcColor                    : color
(o) unsigned short *pusErrCode        : error code
                      GERR_NOERR      : normal
```

Return value None

Details Sets the foreground color of sub-cursor.

GCSNCTableSetSubCursorBackColor

Outline Set background color of sub-cursor

Syntax
```
void GCSNCTableSetSubCursorBackColor(
  GBaseObject     *pSelf,
  GColor          gcColor,
  unsigned short  *pusErrCode
)
```

Argument
```
(i) GBaseObject *pSelf                : object for operation
(i) GColor gcColor                    : color
(o) unsigned short *pusErrCode        : error code
                      GERR_NOERR      : normal
```

Return value None

Details Sets the background color of sub-cursor.

5.2.14 InputBox

GCSInputBoxSetPasswordStatus

Outline Set password display status

Syntax
```
void GCSInputBoxSetPasswordStatus(
 GBaseObject   *pInputBox,
 short         nStatus
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) short nStatus : password display status

Return value None

Details Sets whether the password will be displayed for input boxes or not.
One of the following values is set for nStatus:
0 : normal display status
1 : password display status
After the password display status is set, the control area is registered as a redraw area.

GCSInputBoxGetPasswordStatus

Outline Get password display status

Syntax
```
short GCSInputBoxGetPasswordStatus(
 GBaseObject   *pInputBox
)
```

Argument
(i) GBaseObject *pInputBox : object for operation

Return value
0 : normal display status
1 : password display status

Details Gets whether the input box display status is normal display status or the password display status.

GCSInputBoxSetString

Outline Set display character string

Syntax
```
short GCSInputBoxSetString(
  GBaseObject      *pInputBox,
  const GTCHAR     *pString
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) const GTCHAR *pString : display character string

Return value
FALSE : setting failed
TRUE : setting succeeded

Details
Sets the display character string.
After the display character string is set, the control area is registered as a redraw area.

GCSInputBoxGetString

Outline Get display character string

Syntax
```
short GCSInputBoxGetString(
  GBaseObject   *pInputBox,
  GTCHAR        *pString,
  long          lMaxCount
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(o) GTCHAR *pString : buffer where display character string is stored
(i) long lMaxCount : buffer size

Return value
FALSE : acquisition failed
TRUE : acquisition succeeded

Details
Stores the display character string in pString.

GCSInputBoxSetGValue

Outline

Set display value

Syntax

```
short GCSInputBoxSetGValue(
 GBaseObject   *pInputBox,
 GNCValue      gvValue
)
```

Argument

(i) GBaseObject *pInputBox : object for operation
(i) GNCValue gvValue : setting value

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

Sets the value displayed in an input box.
Numbers are converted to character strings according to the format defined by the target control.
At this time, if the conversion of numbers to character strings fails, or the converted character strings exceed the maximum number of characters, or if any control is not specified as the target control, an error is returned in the return value.
After the display value is set, the control area is registered as a redraw area.

GCSInputBoxGetGValue

Outline

Get display value

Syntax

```
short GCSInputBoxGetGValue(
  GBaseObject   *pInputBox,
  GNCValue      *gvValue
)
```

Argument

(i) GBaseObject *pInputBox : object for operation
(o) GNCValue *gvValue : GNCValue variable storing the value

Return value

FALSE : acquisition failed
TRUE : acquisition succeeded

Details

Gets the value displayed in an input box.
Display character strings are converted to numbers according to the character string type of the target control and stored in the argument.
At this time, if the value is not displayed or the conversion into numbers fails, or if any control is not specified as the target control, an error is returned in the return value.

GCSInputBoxSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSInputBoxSetFocusEffect(
  GBaseObject    *pInputBox,
  unsigned char  ucFocusEffect
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_NON : no effect
GFOCUS_EFFECT_CURSOR : with cursor
GFOCUS_EFFECT_SELECT : select
After the effect during focus is set, the control area is registered as a redraw area.

GCSInputBoxGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSInputBoxGetFocusEffect(
  GBaseObject  *pInputBox
)
```

Argument
(i) GBaseObject *pInputBox : object for operation

Return value
GFOCUS_EFFECT_NON : no effect
GFOCUS_EFFECT_CURSOR : with cursor
GFOCUS_EFFECT_SELECT : select
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSInputBoxSetBackGroundPattern

Outline Set background fill status

Syntax
```
void GCSInputBoxSetBackGroundPattern(
  GBaseObject   *pInputBox,
  short         nBackGroundPattern
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) short nBackGroundPattern : background fill status

Return value None

Details Sets the background color fill status.
NO_DRAW and FILL_BACK_COLOR are the corresponding fill statuses.
After the background fill status is set, the control area is registered as a redraw area.

GCSInputBoxGetBackGroundPattern

Outline Get background fill status

Syntax
```
short GCSInputBoxGetBackGroundPattern(
  GBaseObject   *pInputBox
)
```

Argument
(i) GBaseObject *pInputBox : object for operation

Return value
Background fill status
NO_DRAW : no fill
FILL_BACK_COLOR : with background fill

Details Gets the background color fill status.

GCSInputBoxSetSelectStringStartPos

Outline	Set selected character string start position
Syntax	void GCSInputBoxSetSelectStringStartPos(GBaseObject **pInputBox,* unsigned short *usStartPos*)
Argument	(i) GBaseObject *pInputBox : object for operation (i) unsigned short usStartPos : selected character string start position
Return value	None
Details	Sets the selected character string start position.

GCSInputBoxGetSelectStringStartPos

Outline	Get selected character string start position
Syntax	unsigned short GCSInputBoxGetSelectStringStartPos(GBaseObject **pInputBox*)
Argument	(i) GBaseObject *pInputBox : object for operation
Return value	Selected character string start position
Details	Gets the selected character string start position.

GCSInputBoxSetSelectStringEndPos

Outline Set selected character string end position

Syntax
```
void GCSInputBoxSetSelectStringEndPos(
  GBaseObject      *pInputBox,
  unsigned short   usEndPos
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) unsigned short usEndPos : selected character string end position

Return value None

Details Sets the selected character string end position.

GCSInputBoxGetSelectStringEndPos

Outline Get selected character string end position

Syntax
```
unsigned short GCSInputBoxGetSelectStringEndPos(
  GBaseObject  *pInputBox
)
```

Argument
(i) GBaseObject *pInputBox : object for operation

Return value Selected character string end position

Details Gets the selected character string end position.

GCSInputBoxSetCalcFlag

Outline Set availability of operation function

Syntax
```
void GCSInputBoxSetCalcFlag(
  GBaseObject    *pInputBox,
  unsigned char  ucCalcFlag
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) unsigned char ucCalcFlag : availability of operation function

Return value None

Details Specify the availability of operation function.
One of the following values is set for ucCalcFlag:
0 : without operation function
1 : with operation function

GCSInputBoxGetCalcFlag

Outline Get availability of operation function

Syntax
```
unsigned char GCSInputBoxGetCalcFlag(
  GBaseObject  *pInputBox
)
```

Argument
(i) GBaseObject *pInputBox : object for operation

Return value
0 : without operation function
1 : with operation function
Besides the above : illegal setting

Details Gets the availability of operation function.

GCSInputBoxSetInputValue

Outline

Set input content absolutely/incrementally to the target control

Syntax

```
short GCSInputBoxSetInputValue(
  GBaseObject    *pInputBox,
  unsigned char  ucSetType
)
```

Argument

(i) GBaseObject *pInputBox : object for operation
(i) unsigned char ucSetType : setting method (absolute/incremental)

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

Sets the input content as it is (absolute) to the target control or adds it to the value of the target control.
One of the following values is set for ucSetType:
0 : absolute setting
1 : incremental setting
After the setting is made, the control area is registered as a redraw area.
If any control is not specified as the target control, an error is returned in the return value.

GCSInputBoxSetEchoback

Outline Set availability of echo back

Syntax
```
void GCSInputBoxSetEchoback(
  GBaseObject    *pInputBox,
  unsigned char  ucEchoBackType
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) unsigned char ucEchoBackType : availability of echo back

Return value None

Details Set whether to echo back or not.
One of the following values is set for ucEchoback:
0 : do not echo back
1 : echo back
When 1 (Echo back) is set, the content of the target control will be reflected to and displayed in the input box when a control is specified as the target control.

GCSInputBoxGetEchoback

Outline Get availability of echo back

Syntax
```
unsigned char GCSInputBoxGetEchoback(
  GBaseObject   *pInputBox
)
```

Argument (i) GBaseObject *pInputBox : object for operation

Return value
0 : do not echo back
1 : echo back
Besides the above : illegal setting

Details Gets whether to echo back or not to echo back.

GCSInputBoxSetAbsIncType

Outline Set absolute/incremental input method

Syntax
```
void GCSInputBoxSetAbsIncType(
  GBaseObject    *pInputBox,
  unsigned char  ucAbsIncType
)
```

Argument
(i) GBaseObject *pInputBox : object for operation
(i) unsigned char ucAbsIncType : absolute/incremental input method

Return value None

Details Select the absolute/incremental input method
One of the following values is set for ucAbsIncType:
0 : absolute input
1 : incremental input

GCSInputBoxGetAbsIncType

Outline Get absolute/incremental input method

Syntax
```
unsigned char GCSInputBoxGetAbsIncType(
  GBaseObject  *pInputBox
)
```

Argument
(i) GBaseObject *pInputBox : object for operation

Return value
0 : absolute input
1 : incremental input
Besides the above : illegal setting

Details Gets the absolute/incremental input method

GCSSetRefrectControl

Outline

Set the control to reflect the INPUT

Syntax

```
short GCSSetRefrectControl(
  GBaseObject     *pInputBox,
  unsigned short  usRefrectControlId
)
```

Argument

(i) GBaseObject *pInputBox : object for operation
(i) unsigned short usRefrectControlId : ID of the control to reflect the INPUT

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

Sets the control to reflect the INPUT.
One of the following control ID is set for usRefrectControlId:
- Text box control
- PLC text box control
- NC data text box control

If a control ID not mentioned above is set, an error is returned in the return value.

GCSInputBoxGetRefrectControlTextType

Outline Get character string type of control to reflect the INPUT

Syntax
```
unsigned char GCSInputBoxGetRefrectControlTextType(
  GBaseObject  *pInputBox
)
```

Argument (i) GBaseObject *pInputBox : object for operation

Return value Character string type

When the control to reflect the INPUT is a "text box".

GTEXT_TYPE_STRING(0)	: character string
GTEXT_TYPE_SHORT(1)	: short value
GTEXT_TYPE_USHORT(2)	: unsigned short value
GTEXT_TYPE_LONG(3)	: long value
GTEXT_TYPE_ULONG(4)	: unsigned long value
GTEXT_TYPE_FLOAT(5)	: float value
GTEXT_TYPE_DOUBLE(6)	: double value
Besides the above	: illegal setting

When the control to reflect the INPUT is a "PLC text box".

GPLCTEXT_TYPE_DEC(1)	: signed decimal integer
GPLCTEXT_TYPE_UDEC(2)	: unsigned decimal integer
GPLCTEXT_TYPE_HEX(3)	: hexadecimal integer
GPLCTEXT_TYPE_FLOAT(4)	: real number (float)
GPLCTEXT_TYPE_DOUBLE(5)	: real number (double)
GPLCTEXT_TYPE_STRING(6)	: character string
Besides the above	: illegal setting

When the control to reflect the INPUT is an "NC data text box".

GNCTEXT_TYPE_STRING(0)	: character string
GNCTEXT_TYPE_BIN(1)	: binary integer
GNCTEXT_TYPE_DEC(2)	: signed decimal integer
GNCTEXT_TYPE_UDEC(3)	: unsigned decimal integer
GNCTEXT_TYPE_HEX(4)	: hexadecimal integer
GNCTEXT_TYPE_DOUBLE(5)	: real number (double)
Besides the above	: illegal setting

Details Gets the character string type (character string/numerical value/real number etc.) of the control to reflect the INPUT.

GCSInputBoxGetInputMode

Outline Get input mode (insert/overwrite)

Syntax
```
unsigned char GCSInputBoxGetInputMode(
  GBaseObject  *pInputBox
)
```

Argument (i) GBaseObject *pInputBox : object for operation

Return value
GINPUTBOX_INPUTMODE_INSERT(0) : insert mode
GINPUTBOX_INPUTMODE_OVERWRITE(1) : overwrite mode

Details Gets the current input mode (insert/overwrite).

GCSGetSubCursorControl

Outline Get the ID of the sub cursor display control

Syntax
```
short GCSGetSubCursorControl(
  GBaseObject  *pInputBox
)
```

Argument (i) GBaseObject *pInputBox : object for operation

Return value
-1 : acquisition failed
ID of the control where the sub cursor is pointed : acquisition succeeded

Details Gets the ID of the control where the sub cursor is pointed at.
When the sub cursor settings are not made, an error (-1) is returned to the return value.

5.2.15 SoftKey

GCSSoftKeySetPasswordStatus

Outline Set password display status

Syntax
```
unsigned char GCSSoftKeySetPasswordStatus(
  GBaseObject    *pSoftKey,
  short          nStatus
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation
(i) short nStatus : password display status

Return value None

Details Sets whether the password will be displayed for the ten-key control.
One of the following values is set for nStatus:
FALSE : normal display status
TRUE : password display status
After the password display status is set, the control area is registered as a redraw area.

GCSSoftKeyGetPasswordStatus

Outline Get password display status

Syntax
```
short GCSSoftKeyGetPasswordStatus(
  GBaseObject    *pSoftKey
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation

Return value
FALSE : normal display status
TRUE : password display status

Details Gets whether the ten-key control display status is normal display status or the password display status.

GCSSoftKeySetString

Outline Set display character string

Syntax
```
short GCSSoftKeySetString(
  GBaseObject       *pSoftKey,
  const GTCHAR      *pString
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation
(i) const GTCHAR *pString : display character string

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the character string to display in the input data display part.
After the display character string is set, the control area is registered as a redraw area.

GCSSoftKeyGetString

Outline Get display character string

Syntax
```
short GCSSoftKeyGetString(
  GBaseObject     *pSoftKey,
  GTCHAR          *pString,
  long            lMaxCount
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation
(i) const GTCHAR *pString : buffer where display character string is stored
(i) long lMaxCount : buffer size

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Stores the display character string in pString.

GCSSoftKeySetGValue

Outline

Set display value

Syntax

```
short GCSSoftKeySetGValue(
  GBaseObject    *pSoftKey,
  GNCValue       gvValue
)
```

Argument

(i) GBaseObject *pSoftKey : object for operation
(i) GNCValue gvValue : setting value

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

Sets the value displayed in the ten-key control part.
Numbers are converted to character strings according to the format defined by the control to reflect the INPUT.
At this time, if the conversion of numbers to character strings fails, or the converted character strings exceed the maximum number of characters, or if any control is not specified as the destination to reflect the INPUT, an error is returned in the return value.
After the display value is set, the control area is registered as a redraw area.

GCSSoftKeyGetGValue

Outline

Get display value

Syntax

```
short GCSSoftKeyGetGValue(
  GBaseObject    *pSoftKey
  GNCValue       *gvValue
)
```

Argument

(i) GBaseObject *pSoftKey : object for operation
(i) GNCValue *gvValue : pointer to the GNCValue variable storing the value

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

Gets the value to be displayed in the ten-key control part.
Character strings are converted to numbers according to the character string type of the control to reflect the INPUT and stored in the argument.
At this time, if the value is not displayed or the conversion into numbers fails, an error is returned in the return value.

GCSSoftKeySetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSSoftKeySetFocusEffect(
  GBaseObject     *pSoftKey,
  unsigned char   ucFocusEffect
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_NON : no effect
GFOCUS_EFFECT_CURSOR : with cursor
After the effect during focus is set, the control area is registered as a redraw area.

GCSSoftKeyGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSSoftKeyGetFocusEffect(
  GBaseObject     *pSoftKey
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect during focus

GCSSoftKeySetCalcFlag

Outline Set operation function

Syntax
```
void GCSSoftKeySetCalcFlag(
  GBaseObject    *pSoftKey,
  unsigned char  ucCalcFlag
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation
(i) unsigned char ucCalcFlag : availability of operation function

Return value None

Details Sets the availability of operation function.
One of the following values is set for ucCalcFlag:
TRUE : with operation function
FALSE : without operation function

GCSSoftKeyGetCalcFlag

Outline Get operation function

Syntax
```
unsigned char GCSSoftKeyGetCalcFlag(
  GBaseObject  *pSoftKey
)
```

Argument
(i) GBaseObject *pSoftKey : object for operation

Return value
FALSE : without operation function
TRUE : with operation function

Details Gets the availability of operation function.

GCSSoftKeySetInputValue

Outline

Set to the control to reflect the INPUT

Syntax

```
short GCSSoftKeySetInputValue(
  GBaseObject    *pSoftKey,
  unsigned char  ucSetType
)
```

Argument

(i) GBaseObject *pSoftKey : object for operation
(i) unsigned char ucSetType : setting method (absolute/incremental)

Return value

None

Details

Sets the input content as it is (absolute) to the target control, or adds it to the value of the target control.
One of the following values is set for ucSetType:
0 : absolute setting
1 : incremental setting
After the setting is made, the control area is registered as a redraw area.
If any control is not specified as the target control, an error is returned in the return value.

GCSSoftKeySetAbsIncType

Outline	Set input method (absolute/incremental)
Syntax	void GCSSoftKeySetAbsIncType(GBaseObject **pSoftKey,* unsigned char *ucAbsIncType*)
Argument	(i) GBaseObject *pSoftKey : object for operation (i) unsigned char ucAbsIncType : absolute/incremental input method
Return value	None
Details	Select the absolute/incremental input method. One of the following values is set for ucAbsIncType: 0 : absolute input 1 : incremental input

GCSSoftKeyGetAbsIncType

Outline	Get input method (absolute/incremental)
Syntax	unsigned char GCSSoftKeyGetAbsIncType(GBaseObject **pSoftKey*)
Argument	(i) GBaseObject *pSoftKey : object for operation
Return value	Absolute/incremental input method
Details	Gets the absolute/incremental input method.

GCSSetRefrectControl

Outline	Set the control to reflect the INPUT. * As this function is the same as the function in the InputBox section, refer to the GCSSetRefrectControl function in 5.2.14 InputBox.

GCSSoftKeyGetRefrectControlTextType

Outline Get character string type of control to reflect the INPUT

Syntax
```
unsigned char GCSSoftKeyGetRefrectControlTextType(
  GBaseObject    *pSoftKey
)
```

Argument (i) GBaseObject *pSoftKey : object for operation

Return value Character string type

Details Gets the character string type (character string/numerical value/real number etc.) of the control to reflect the INPUT.

[When the control to reflect the INPUT is a "text box".]

GTEXT_TYPE_STRING	0x00	: character string
GTEXT_TYPE_SHORT	0x01	: short value
GTEXT_TYPE_USHORT	0x02	: unsigned short value
GTEXT_TYPE_LONG	0x03	: long value
GTEXT_TYPE_ULONG	0x04	: unsigned long value
GTEXT_TYPE_FLOAT	0x05	: float value
GTEXT_TYPE_DOUBLE	0x06	: double value
Besides the above		: illegal setting

[When the control to reflect the INPUT is a "PLC text box".]

GPLCTEXT_TYPE_DEC	0x01	: signed decimal integer
GPLCTEXT_TYPE_UDEC	0x02	: unsigned decimal integer
GPLCTEXT_TYPE_HEX	0x03	: hexadecimal integer
GPLCTEXT_TYPE_FLOAT	0x04	: real number
GPLCTEXT_TYPE_DOUBLE	0x05	: real number (double)
GPLCTEXT_TYPE_STRING	0x06	: character string
Besides the above		: illegal setting

[When the control to reflect the INPUT is an "NC data text box".]

GNCTEXT_TYPE_STRING	0x00	: character string
GNCTEXT_TYPE_BIN	0x01	: binary integer
GNCTEXT_TYPE_DEC	0x02	: signed decimal integer
GNCTEXT_TYPE_UDEC	0x03	: unsigned decimal integer
GNCTEXT_TYPE_HEX	0x04	: hexadecimal integer
GNCTEXT_TYPE_DOUBLE	0x05	: real number (double)
Besides the above		: illegal setting

GCSSoftKeyGetInputMode

Outline Get input mode (insert/overwrite)

Syntax
```
unsigned char GCSSoftKeyGetInputMode(
  GBaseObject    *pSoftKey
)
```

Argument (i) GBaseObject *pSoftKey : object for operation

Return value Input mode status

Details Gets the current input mode (insert/overwrite).

GCSSoftKeySetTitleString

Outline Set character string to be displayed on title bar

Syntax
```
void GCSSoftKeySetTitleString(
  GBaseObject  *pSoftKey,
  GTCHAR       *pTitleString
)
```

Argument (i) GBaseObject *pSoftKey : object for operation
(i) GTCHAR *pTitleString : title bar character string

Return value FALSE : setting failed
TRUE : setting succeeded

Details Gets the character string to be displayed on the title bar.

GCSGetSubCursorControl

Outline	Get the ID of the sub cursor display control. * As this function is the same as the function in the InputBox section, refer to the GCSGetSubCursorControl function in 5.2.14 InputBox.

5.2.16 Counter

GCSAxisChange

Outline	Switch displayed axis
Syntax	long GCSAxisChange(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	FALSE : setting failed TRUE : setting succeeded
Details	Changes the axis to be displayed when the number of control axes has exceeded the maximum number of lines for display. After the axis is set, the control area is registered as a redraw area.

GCSSetCounterType

Outline

Set counter type

Syntax

```
long GCSSetCounterType(
  GBaseObject   *pSelf,
  unsigned long  _CounterType
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned long _CounterType : counter type

Return value

FALSE : setting failed
TRUE : setting succeeded

Details

Sets the counter type.
One of the following values is set for _CounterTyp:

GCURRENT_ID	: current position
GWORK_ID	: workpiece coordinate position
GMACHINE_ID	: machine position
GPRGM_ID	: program position
GPRG_REF_ID	: program origin
GPIT_ERR_ID	: error correction amount
GDIS_ID	: remaining command
GMANUAL_ID	: manual interruption amount
GNEXTPOS_ID	: next command
GRESTART_ID	: restart position
GRES_DIS_ID	: restart remain distance
GTIP_POS_ID	: tool tip position
GDISTANCE_ID	: machine axis movement
GPULSE_ID	: tool axis movement
GTIP2_POS_ID	: tool tip machine position
GRELATIVE_ID	: relative position
GPLCAXIS_IDPLC	: axis position

After the counter type is set, the control area is registered as a redraw area.

5.2.17 CycleTime

GCSSetTimeType

Outline	Displayed time type setting
Syntax	long GCSSetTimeType(GBaseObject *pSelf, long _lArea, long _lTimeType)
Argument	(i) GBaseObject *pSelf : object for operation (i) long _lArea : display position (i) long _lTimeType : displayed time type
Return value	None
Details	Sets the type of time to be displayed. One of the following values is set for _lArea: 0 : upper tier 1 : lower tier One of the following values is set for _lTimeType: 0 : default 1 : date 2 : time 3 : power-ON 4 : automatic operation 5 : automatic start 6 : external accumulation 1 7 : external accumulation 2 8 : cycle time After the time type is set, the control area is registered as a redraw area.

5.2.18 LoadMeter

GCSLoadMeterSetDispType

Outline	Set display type of load meter
Syntax	void GCSLoadMeterSetDispType(GBaseObject **pSelf,* long *lDispType*)
Argument	(i) GBaseObject *pSelf : object for operation (i) long lDispType : display type
Return value	None
Details	Sets the display type. One of the following values is set for lDispType: 0 : display spindle load only 1 : display Z axis load only 2 : display both spindle and Z axis loads

5.2.19 SPCommand

GCSSetDispSpAxis

Outline	Set spindle to be displayed
Syntax	long GCSSetDispSpAxis(GBaseObject *pSelf, long *_lSPs, long _lSPNum)
Argument	(i) GBaseObject *pSelf : object for operation (i) long *_lSPs : displayed spindle arrangement (i) long _lSPNum : number of displayed spindle arrangements
Return value	None
Details	Sets the spindle to be displayed.

5.2.20 NCPLCButton

GCSNCPLCButtonSetAction

Outline	Set PLC button operations
Syntax	void GCSNCPLCButtonSetAction(GBaseObject **pPLCButton,* unsigned char *ucAction*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) unsigned char ucAction : PLC button operation
Return value	None
Details	Sets the PLC button operation. One of the following values is set for ucAction: GPLCBTN_ACT_NOACTION : no operation GPLCBTN_ACT_MOMENTARY : momentary operation GPLCBTN_ACT_ALTERNATE : alternative operation When a value outside of the range is set, there is no forced change of the setting.

GCSNCPLCButtonGetAction

Outline

Get PLC button operations

Syntax

```
unsigned char GCSNCPLCButtonGetAction(
  GBaseObject   *pPLCButton
)
```

Argument

(i) GBaseObject *pPLCButton : object for operation

Return value

GPLCBTN_ACT_NOACTION	: no operation
GPLCBTN_ACT_MOMENTARY	: momentary operation
GPLCBTN_ACT_ALTERNATE	: alternative operation
Besides the above	: illegal operation setting

Details

Gets the PLC button operation setting.
When a setting outside the range is made, that setting value is returned.

GCSNCPLCButtonSetDisplay

Outline

Set PLC button display

Syntax

```
void GCSNCPLCButtonSetDisplay(
  GBaseObject     *pPLCButton,
  unsigned char   ucDisplay
)
```

Argument

(i) GBaseObject *pPLCButton : object for operation
(i) unsigned char ucDisplay : PLC button display

Return value

None

Details

Sets the PLC button display.
One of the following values is set for ucDisplay:
GPLCBTN_DISP_RECT : rectangle display
GPLCBTN_DISP_OVAL : round shape display
GPLCBTN_DISP_IMAGE : image display
When a value outside of the range is set, there is no forced change of the setting.
After the button display is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetDisplay

Outline

Get PLC button display

Syntax

```
unsigned char GCSNCPLCButtonGetDisplay(
  GBaseObject   *pPLCButton
)
```

Argument

(i) GBaseObject *pPLCButton : object for operation

Return value

GPLCBTN_DISP_RECT	: rectangle display
GPLCBTN_DISP_OVAL	: round shape display
GPLCBTN_DISP_IMAGE	: image display
Besides the above	: illegal display setting

Details

Gets the PLC button display setting.
When a setting outside the range is made, that setting value is returned.

GCSNCPLCButtonSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSNCPLCButtonSetBorderID(GBaseObject **pPLCButton,* unsigned short *usID*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSNCPLCButtonGetBorderID(GBaseObject **pPLCButton*)
Argument	(i) GBaseObject *pPLCButton : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID.

GCSNCPLCButtonSetOnDesign

Outline	Set ON status design
Syntax	void GCSNCPLCButtonSetOnDesign(GBaseObject **pPLCButton,* GDesign **pDesign*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) GDesign *pDesign : pointer to design
Return value	None
Details	Sets the design for ON status. After the design for ON status is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOnDesign

Outline	Get ON status design
Syntax	GDesign *GCSNCPLCButtonGetOnDesign(GBaseObject **pPLCButton*)
Argument	(i) GBaseObject *pPLCButton : object for operation
Return value	Pointer to design
Details	Gets the design for ON status.

GCSNCPLCButtonSetOffDesign

Outline Set OFF status design

Syntax
```
void GCSNCPLCButtonSetOffDesign(
  GBaseObject   *pPLCButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) GDesign *pDesign : pointer to design

Return value None

Details Sets the design for OFF status.
After the design for OFF status is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOffDesign

Outline Get OFF status design

Syntax
```
GDesign *GCSNCPLCButtonGetOffDesign(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Pointer to design

Details Gets the design for OFF status.

GCSNCPLCButtonSetOnOnDesign

Outline Set ONON status design

Syntax
```
void GCSNCPLCButtonSetOnOnDesign(
  GBaseObject   *pPLCButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) GDesign *pDesign : pointer to design

Return value None

Details Sets the design for ONON status.
After the design for ONON status is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOnOnDesign

Outline Get ONON status design

Syntax
```
GDesign *GCSNCPLCButtonGetOnOnDesign(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Pointer to design

Details Gets the design for ONON status.

GCSNCPLCButtonSetOnOffDesign

Outline Set ONOFF status design

Syntax
```
void GCSNCPLCButtonSetOnOffDesign(
  GBaseObject   *pPLCButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) GDesign *pDesign : pointer to design

Return value None

Details Sets the design for ONOFF status.
After the design for ONOFF status is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOnOffDesign

Outline Get ONOFF status design

Syntax
```
GDesign *GCSNCPLCButtonGetOnOffDesign(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Pointer to design

Details Gets the design for ONOFF status.

GCSNCPLCButtonSetFocusDesign

Outline Set FOCUS status design

Syntax
```
void GCSNCPLCButtonSetFocusDesign(
  GBaseObject   *pPLCButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) GDesign *pDesign : pointer to design

Return value None

Details Sets the design for FOCUS status.
After the design for FOCUS status is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetFocusDesign

Outline Get FOCUS status design

Syntax
```
GDesign *GCSNCPLCButtonGetFocusDesign(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Pointer to design

Details Gets the design for FOCUS status.

GCSNCPLCButtonSetDisableDesign

Outline Set Disable status design

Syntax
```
void GCSNCPLCButtonSetDisableDesign(
  GBaseObject   *pPLCButton,
  GDesign       *pDesign
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) GDesign *pDesign : pointer to design

Return value None

Details Sets the design for Disable status.
After the design for Disable status is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetDisableDesign

Outline Get Disable status design

Syntax
```
GDesign *GCSNCPLCButtonGetDisableDesign(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Pointer to design

Details Gets the design for Disable status.

GCSNCPLCButtonSetFontID

Outline	Set font resource ID
Syntax	void GCSNCPLCButtonSetFontID(GBaseObject **pPLCButton,* unsigned short *usID*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) unsigned short usID : font resource ID
Return value	None
Details	Sets the font resource ID. After the font resource ID is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetFontID

Outline	Get font resource ID
Syntax	unsigned short GCSNCPLCButtonGetFontID(GBaseObject **pPLCButton*)
Argument	(i) GBaseObject *pPLCButton : object for operation
Return value	Font resource ID
Details	Gets the font resource ID.

GCSNCPLCButtonSetOnStringID

Outline Set ON status caption character string resource ID

Syntax
```
void GCSNCPLCButtonSetOnStringID(
  GBaseObject      *pPLCButton,
  unsigned short   usID
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) unsigned short usID : caption character string resource ID

Return value None

Details Sets the ON status caption character string resource ID.
After the ON status caption character string resource ID is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOnStringID

Outline Get ON status caption character string resource ID

Syntax
```
unsigned short GCSNCPLCButtonGetOnStringID(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Caption character string resource ID

Details Gets the ON status caption character string resource ID.

GCSNCPLCButtonSetOffStringID

Outline	Set OFF status caption character string resource ID
Syntax	void GCSNCPLCButtonSetOffStringID(GBaseObject **pPLCButton,* unsigned short *usID*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) unsigned short usID : caption character string resource ID
Return value	None
Details	Sets the OFF status caption character string resource ID. After the OFF status caption character string resource ID is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOffStringID

Outline	Get OFF status caption character string resource ID
Syntax	unsigned short GCSNCPLCButtonGetOffStringID(GBaseObject **pPLCButton*)
Argument	(i) GBaseObject *pPLCButton : object for operation
Return value	Caption character string resource ID
Details	Gets the OFF status caption character string resource ID.

GCSNCPLCButtonSetOnOnStringID

Outline	Set ONON status caption character string resource ID
Syntax	void GCSNCPLCButtonSetOnOnStringID(GBaseObject **pPLCButton,* unsigned short *usID*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) unsigned short usID : caption character string resource ID
Return value	None
Details	Sets the ONON status caption character string resource ID. After the ONON status caption character string resource ID is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOnOnStringID

Outline	Get ONON status caption character string resource ID
Syntax	unsigned short GCSNCPLCButtonGetOnOnStringID(GBaseObject **pPLCButton*)
Argument	(i) GBaseObject *pPLCButton : object for operation
Return value	Caption character string resource ID
Details	Gets the ONON status caption character string resource ID.

GCSNCPLCButtonSetOnOffStringID

Outline Set ONOFF status caption character string resource ID

Syntax
```
void GCSNCPLCButtonSetOnOffStringID(
  GBaseObject      *pPLCButton,
  unsigned short   usID
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) unsigned short usID : caption character string resource ID

Return value None

Details Sets the ONOFF status caption character string resource ID.
After the ONOFF status caption character string resource ID is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetOnOffStringID

Outline Get ONOFF status caption character string resource ID

Syntax
```
unsigned short GCSNCPLCButtonGetOnOffStringID(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Caption character string resource ID

Details Gets the ONOFF status caption character string resource ID.

GCSNCPLCButtonSetCaption

Outline Set caption information

Syntax
```
void GCSNCPLCButtonSetCaption(
  GBaseObject      *pPLCButton,
  GCaption         *pCaption
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) GCaption *pCaption : pointer to caption information

Return value None

Details Sets pCaption to the caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSNCPLCButtonGetCaption(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.

GCSNCPLCButtonSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSNCPLCButtonSetFocusEffect(
  GBaseObject      *pPLCButton,
  unsigned char    ucFocusEffect
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
After the effect during focus is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSNCPLCButtonGetFocusEffect(
  GBaseObject   *pPLCButton
)
```

Argument (i) GBaseObject *pPLCButton : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSNCPLCButtonSetStatus

Outline	Set object status
Syntax	void GCSNCPLCButtonSetStatus(GBaseObject **pPLCButton,* unsigned char *ucStatus*)
Argument	(i) GBaseObject *pPLCButton : object for operation (i) unsigned char ucStatus : set object status
Return value	None
Details	Sets the GCNCPLCButton object ON/OFF status. After the GCNCPLCButton object ON/OFF is set, the control area is registered as a redraw area.

GCSNCPLCButtonGetStatus

Outline	Get object status
Syntax	unsigned char GCSNCPLCButtonGetStatus(GBaseObject **pPLCButton*)
Argument	(i) GBaseObject *pPLCButton : object for operation
Return value	Object status
Details	Gets the GCNCPLCButton object status.

GCSNCPLCButtonSetDeviceType

Outline Set PLC device operations

Syntax
```
short GCSNCPLCButtonSetDeviceType(
  GBaseObject     *pPLCButton,
  short           nIndex,
  unsigned char   ucDeviceType
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) short nIndex : index No.
(i) unsigned char ucDeviceType : PLC device type

Return value
FALSE : abnormal (index No. illegal)
TRUE : normal

Details Sets the PLC device type.

GCSNCPLCButtonGetDeviceType

Outline Get PLC device operations

Syntax
```
short GCSNCPLCButtonGetDeviceType(
  GBaseObject     *pPLCButton,
  short           nIndex,
  unsigned char   *pDeviceType
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) short nIndex : index No.
(i) unsigned char *pDeviceType : PLC device type

Return value
FALSE : abnormal (index No. illegal)
TRUE : normal

Details Gets the PLC device type.

GCSNCPLCButtonSetDevice

Outline Set PLC device

Syntax
```
short GCSNCPLCButtonSetDevice(
  GBaseObject   *pPLCButton,
  short         nIndex,
  char          *szDevice
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) short nIndex : index No.
(i) char *szDevice : PLC device

Return value
FALSE : abnormal (index No. illegal)
TRUE : normal

Details Sets the PLC device.

GCSNCPLCButtonGetDevice

Outline Get PLC device

Syntax
```
short GCSNCPLCButtonGetDevice(
  GBaseObject   *pPLCButton,
  short         nIndex,
  char          *szDevice
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) short nIndex : index No.
(i) char *szDevice : PLC device

Return value
FALSE : abnormal (index No. illegal)
TRUE : normal

Details Gets the PLC device.

GCSNCPLCButtonSetDeviceVal

Outline Set PLC device value

Syntax
```
short GCSNCPLCButtonSetDeviceVal(
  GBaseObject     *pPLCButton,
  short           nIndex,
  unsigned char   ucDeviceVal
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) short nIndex : index No.
(i) unsigned char ucDeviceVal : PLC device value

Return value
FALSE : abnormal (index No. illegal)
TRUE : normal

Details Sets the PLC device value.

GCSNCPLCButtonGetDeviceVal

Outline Get PLC device value

Syntax
```
short GCSNCPLCButtonGetDeviceVal(
  GBaseObject     *pPLCButton,
  short           nIndex,
  unsigned char   *pDeviceVal
)
```

Argument
(i) GBaseObject *pPLCButton : object for operation
(i) short nIndex : index No.
(i) unsigned char *pDeviceVal : PLC device value

Return value
FALSE : abnormal (index No. illegal)
TRUE : normal

Details Gets the PLC device value.

5.2.21 NCPLCTextBox

GCSNCPLCTextboxSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSNCPLCTextboxSetBorderID(GBaseObject **pPLCTextBox,* unsigned char *usID*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation (i) unsigned char usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSNCPLCTextboxGetBorderID(GBaseObject **pPLCTextBox*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID.

GCSNCPLCTextboxSetFontID

Outline Set font resource ID

Syntax
```
void GCSNCPLCTextboxSetFontID(
  GBaseObject      *pPLCTextBox,
  unsigned short   usID
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSNCPLCTextboxGetFontID(
  GBaseObject      *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSNCPLCTextboxSetCaption

Outline Set caption information

Syntax
```
void GCSNCPLCTextboxSetCaption(
  GBaseObject     *pPLCTextBox,
  GCaption        *pCaption
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GCaption *pCaption : pointer to caption information

Return value None

Details Sets pCaption to the caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSNCPLCTextboxGetCaption(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Pointer to caption information

Details Gets the caption information setting.

GCSNCPLCTextboxSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSNCPLCTextboxSetFocusEffect(
  GBaseObject      *pPLCTextBox,
  unsigned char    ucFocusEffect
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
After the effect during focus is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSNCPLCTextboxGetFocusEffect(
  GBaseObject      *pPLCTextBox
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value
GFOCUS_EFFECT_CHG_COLOR : design change
GFOCUS_EFFECT_NON : no effect
Besides the above : illegal display setting

Details Gets the effect used during focus.

GCSNCPLCTextboxSetFocusColor

Outline Set background color during focus

Syntax
```
void GCSNCPLCTextboxSetFocusColor(
  GBaseObject     *pPLCTextBox,
  GColor          gcColor
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GColor gcColor : color code for background color during focus

Return value None

Details Sets the background color during focus.
After the background color during focus is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetFocusColor

Outline Get background color during focus

Syntax
```
GColor GCSNCPLCTextboxGetFocusColor(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Color code for background color during focus

Details Gets the background color during focus.

GCSNCPLCTextboxSetNormalColor

Outline Set normal background color

Syntax
```
void GCSNCPLCTextboxSetNormalColor(
  GBaseObject      *pPLCTextBox,
  GColor           gcColor
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GColor gcColor : color code for normal background color

Return value None

Details Sets the normal background color.
After the normal background color is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetNormalColor

Outline Get normal background color

Syntax
```
GColor GCSNCPLCTextboxGetNormalColor(
  GBaseObject      *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Color code for normal background color

Details Gets the normal background color.

GCSNCPLCTextboxSetDisableColor

Outline Set background color when disabled

Syntax
```
void GCSNCPLCTextboxSetDisableColor(
  GBaseObject    *pPLCTextBox,
  GColor         gcColor
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GColor gcColor : color code for background color when disabled

Return value None

Details Sets the background color when disabled.
After the background color when disabled is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetDisableColor

Outline Get background color when disabled

Syntax
```
GColor GCSNCPLCTextboxGetDisableColor(
  GBaseObject    *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Color code for background color setting when disabled

Details Gets the background color when disabled.

GCSNCPLCTextboxSetTextType

Outline

Set character string type

Syntax

```
void GCSNCPLCTextboxSetTextType(
 GBaseObject      *pPLCTextBox,
 unsigned char    ucType
)
```

Argument

(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned char ucType : setting value

Return value

None

Details

Sets the display character string type.
One of the following values is set for ucType:

GPLCTEXT_TYPE_DEC	: signed decimal integer
GPLCTEXT_TYPE_UDEC	: unsigned decimal integer
GPLCTEXT_TYPE_HEX	: hexadecimal integer
GPLCTEXT_TYPE_FLOAT	: real number (float)
GPLCTEXT_TYPE_DOUBLE	: real number (double)
GPLCTEXT_TYPE_STRING	: character sequence

After the character string type is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetTextType

Outline	Get character string type
Syntax	unsigned char GCSNCPLCTextboxGetTextType(GBaseObject **pPLCTextBox*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation
Return value	GPLCTEXT_TYPE_DEC : signed decimal integer GPLCTEXT_TYPE_UDEC : unsigned decimal integer GPLCTEXT_TYPE_HEX : hexadecimal integer GPLCTEXT_TYPE_FLOAT : real number (float) GPLCTEXT_TYPE_DOUBLE : real number (double) GPLCTEXT_TYPE_STRING : character sequence Besides the above : illegal setting
Details	Gets the display character string type.

GCSNCPLCTextboxSetPasswordStatus

Outline Set password display status

Syntax
```
void GCSNCPLCTextboxSetPasswordStatus(
  GBaseObject      *pPLCTextBox,
  short            fStatus
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) short fStatus : password display status
FALSE : normal display status
TRUE : password display status

Return value None

Details Sets whether the password will be displayed for text boxes or not.
After the password display status is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetPasswordStatus

Outline Get password display status

Syntax
```
short GCSNCPLCTextboxGetPasswordStatus(
  GBaseObject      *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value
FALSE : normal display status
TRUE : password display status

Details Gets whether the test box display status is normal display status or the password display status.

GCSNCPLCTextboxSetCommaStatus

Outline Set comma display status

Syntax
```
void GCSNCPLCTextboxSetCommaStatus(
  GBaseObject     *pPLCTextBox,
  short           fStatus
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) short fStatus : comma display status
FALSE : normal display status
TRUE : comma display status

Return value None

Details Sets whether commas will be displayed or not for text boxes.
After the comma display status is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetCommaStatus

Outline Get comma display status

Syntax
```
short GCSNCPLCTextboxGetCommaStatus(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value
FALSE : normal display status
TRUE : comma display status

Details Gets the setting as to whether the test box display status is normal display status or the comma display status.

GCSNCPLCTextboxSetCheckMaxStatus

Outline Set maximum value check status

Syntax
```
void GCSNCPLCTextboxSetCheckMaxStatus(
  GBaseObject     *pPLCTextBox,
  short           fStatus
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) short fStatus : setting for maximum value check status
FALSE : check disabled
TRUE : check enabled

Return value None

Details Sets the status of the maximum value check during SetValue/GetValue for the PLC text box.

GCSNCPLCTextboxGetCheckMaxStatus

Outline Get maximum value check status

Syntax
```
short GCSNCPLCTextboxGetCheckMaxStatus(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox

Return value
FALSE : check disabled
TRUE : check enabled

Details Gets the status of the maximum value check during SetValue/GetValue from the PLC text box.

GCSNCPLCTextboxSetCheckMinStatus

Outline Set minimum value check status

Syntax
```
void GCSNCPLCTextboxSetCheckMinStatus(
  GBaseObject     *pPLCTextBox,
  short           fStatus
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) short fStatus : setting for minimum value check status
FALSE : check disabled
TRUE : check enabled

Return value None

Details Sets the status of the maximum value check during SetValue/GetValue for the PLC text box.

GCSNCPLCTextboxGetCheckMinStatus

Outline Get minimum value check status

Syntax
```
short GCSNCPLCTextboxGetCheckMinStatus(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value
FALSE : check disabled
TRUE : check enabled

Details Gets the status of the minimum value check during SetValue/GetValue from the PLC text box.

GCSNCPLCTextboxSetMaxGValue

Outline Set maximum value

Syntax
```
void GCSNCPLCTextboxSetMaxGValue(
  GBaseObject     *pPLCTextBox,
  GValue          gvValue
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GValue gvValue : maximum value

Return value None

Details Sets the value of the maximum value check during SetValue/GetValue for the PLC text box.

GCSNCPLCTextboxGetMaxGValue

Outline Get maximum value

Syntax
```
GValue GCSNCPLCTextboxGetMaxGValue(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Maximum value (GValue)

Details Gets the value of the maximum value check during SetValue/GetValue from the PLC text box.

GCSNCPLCTextboxSetMinGValue

Outline Set minimum value

Syntax
```
void GCSNCPLCTextboxSetMinGValue(
  GBaseObject     *pPLCTextBox,
  GValue          gvValue
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GValue gvValue : minimum value

Return value None

Details Sets the value of the minimum value check during SetValue/GetValue for the PLC text box.

GCSNCPLCTextboxGetMinGValue

Outline Get minimum value

Syntax
```
GValue GCSNCPLCTextboxGetMinGValue(
  GBaseObject     *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Minimum value (GValue)

Details Gets the value of the minimum value check during SetValue/GetValue from the PLC text box.

GCSNCPLCTextboxSetString

Outline Set display character string

Syntax
```
short GCSNCPLCTextboxSetString(
  GBaseObject      *pPLCTextBox,
  const GTCHAR     *pString
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) const char *pString : display character string

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the display character string.

GCSNCPLCTextboxGetString

Outline Get display character string

Syntax
```
short GCSNCPLCTextboxGetString(
  GBaseObject      *pPLCTextBox,
  GTCHAR           *pString,
  long             lMaxCount
)
```

Argument
(i) GbaseObject *pPLCTextBox : object for operation
(o) char *pString : buffer to store display character string
(i) long lMaxCount : buffer size

Return value
FALSE : acquisition failed
TRUE : acquisition succeeded

Details Stores the display character string in pString.

GCSNCPLCTextboxSetGValue

Outline

Set display value

Syntax

```
short GCSNCPLCTextboxSetGValue(
 GBaseObject    *pPLCTextBox,
 GValue         gvValue
)
```

Argument

(i) GBaseObject *pPLCTextBox : object for operation
(i) GValue gvValue : setting value

Return value

0	: normal value
GERR_NCPLCTEXT_UNDER_MIN_LIMIT	: less than minimum value
GERR_NCPLCTEXT_OVER_MAX_LIMIT	: greater than maximum value
GERR_NCPLCTEXT_LIMITLENGTHOVER	: over than maximum number of characters
GERR_NCPLCTEXT_INVALID_TYPE	: illegal type

Details

Sets the PLC device value to allocate to the control.
The maximum value and minimum value for the PLC device value are checked, and if the value is outside of the range, an error is returned in the return value.
The error check is executed for the following:
GPLCTEXT_TYPE_DEC: signed decimal integer
GPLCTEXT_TYPE_UDEC: unsigned decimal integer
GPLCTEXT_TYPE_HEX: hexadecimal integer
After the character string type is set, the control area is registered as a redraw area.

(Note 1) If the number of characters after converting the writing value to a display character string is over the value of the "maximum number of characters" of the properties, "over maximum number of characters" is returned on the return value. Even when it is over the "maximum number of characters", a character string after the conversion which is limited to the "maximum number of characters" is displayed in the PLC text box.

(Note 2) Make sure to store the setting value in the float type variable (GValue.fValue) if a real number (float) or a real number (double) is set to the "type" of the properties.
When stored in a variable other than the float type variable, the setting value is not written correctly.

(Note 3) Since this function does not support the double type variable, the range of settings follows a real number (float) limitation even when a real number (double) is set to the "type" of the properties.

(Note 4) If "Character string" is set to the "type" of the properties, "illegal type" will be returned.

GCSNCPLCTextboxGetGValue

Outline

Get display value

Syntax

```
short GCSNCPLCTextboxGetGValue(
  GBaseObject    *pPLCTextBox,
  GValue         *gvValue
)
```

Argument

(i) GBaseObject *pPLCTextBox : object for operation
(i) GValue *gvValue : pointer to GValue variable stored value

Return value

0	: normal value
GERR_NCPLCTEXT_INVALID_VALUE	: illegal value
GERR_NCPLCTEXT_UNDER_MIN_LIMIT	: less than minimum value
GERR_NCPLCTEXT_OVER_MAX_LIMIT	: more than maximum value
GERR_NCPLCTEXT_INVALID_TYPE	: illegal type

Details

Gets the PLC device value allocated to the control.
The maximum value and minimum value for the PLC device value are checked, and if it is outside of the range, an error is returned in the return value.
The error check is executed for the following:
GPLCTEXT_TYPE_DEC: Signed decimal integer
GPLCTEXT_TYPE_UDEC: Unsigned decimal integer
GPLCTEXT_TYPE_HEX: Hexadecimal integer

(Note 1) Make sure to store the pointer in the float type variable (GValue.fValue) if a real number (float) or a real number (double) is set to the "type" of the properties. When stored in a variable other than the float type variable, the read value is not obtained correctly.

(Note 2) Since this function does not support the double type variable, the range of settings follows a real number (float) limitation even when a real number (double) is set to the "type" of the properties.

(Note 3) If "Character string" is set to the "type" of the properties, "illegal type" will be returned.

GCSNCPLCTextboxSetStringBuffer

Outline Set display buffer

Syntax
```
short GCSNCPLCTextboxSetStringBuffer(
  GBaseObject      *pPLCTextBox,
  unsigned long    ulBufferSize,
  GTCHAR           *pStringBuffer
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned long ulBufferSize : buffer size
(i) char *pStringBuffer : character string buffer

Return value
FALSE : setting failed
TRUE : setting succeeded

Details Sets the display buffer.
Allocates memory internally when pStringBuffer is NULL.
When a character string buffer is specified, that buffer is set.

GCSNCPLCTextboxGetStringBuffer

Outline Get display buffer

Syntax
```
GTCHAR *GCSNCPLCTextboxGetStringBuffer(
  GBaseObject    *pPLCTextBox,
  unsigned long  *ulBufferSize
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned long *ulBufferSize : pointer to variable stored buffer size

Return value Display buffer address

Details Gets the display buffer.

GCSNCPLCTextboxSetBackGroundPattern

Outline Set background fill status

Syntax
```
void GCSNCPLCTextboxSetBackGroundPattern(
  GBaseObject      *pPLCTextBox,
  short            nBackGroundPattern
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) short nBackGroundPattern : background fill status

Return value None

Details Sets the background color fill status.
NO_DRAW and FILL_BACK_COLOR are the corresponding fill statuses.

GCSNCPLCTextboxGetBackGroundPattern

Outline Get background fill status

Syntax
```
short GCSNCPLCTextboxGetBackGroundPattern(
  GBaseObject      *pPLCTextBox
)
```

Argument (i) GBaseObject *pPLCTextBox : object for operation

Return value Background fill status
NO_DRAW : no fill
FILL_BACK_COLOR : background fill

Details Sets the background color fill status.
NO_DRAW and FILL_BACK_COLOR are the corresponding fill statuses.

GCSNCPLCTextboxSetSelectStringStartPos

Outline	Set selected character string start position
Syntax	void GCSNCPLCTextboxSetSelectStringStartPos(GBaseObject **pPLCTextBox,* unsigned short *usStartPos*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation (i) unsigned short usStartPos : selected character string start position
Return value	None
Details	Sets the selected character string start position.

GCSNCPLCTextboxGetSelectStringStartPos

Outline	Get selected character string start position
Syntax	unsigned short GCSNCPLCTextboxGetSelectStringStartPos(GBaseObject **pPLCTextBox*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation
Return value	Selected character string start position
Details	Gets the selected character string start position.

GCSNCPLCTextboxSetSelectStringEndPos

Outline	Set selected character string end position
Syntax	void GCSNCPLCTextboxSetSelectStringEndPos(GBaseObject **pPLCTextBox,* unsigned short *usEndPos*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation (i) unsigned short usEndPos : selected character string end position
Return value	None
Details	Sets the selected character string end position.

GCSNCPLCTextboxGetSelectStringEndPos

Outline	Get selected character string end position
Syntax	unsigned short GCSNCPLCTextboxGetSelectStringEndPos(GBaseObject **pPLCTextBox*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation
Return value	Selected character string end position
Details	Gets the selected character string end position.

GCSNCPLCTextboxSetIntegerWidth

Outline	Set integer part width
Syntax	void GCSNCPLCTextboxSetIntegerWidth(GBaseObject **pPLCTextBox,* unsigned char *ucWidth*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation (i) unsigned char ucWidth : number of integer part figures
Return value	None
Details	Sets the number of integer part figures.

GCSNCPLCTextboxGetIntegerWidth

Outline	Get integer part width
Syntax	unsigned char GCSNCPLCTextboxGetIntegerWidth(GBaseObject **pPLCTextBox*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation
Return value	The number of integer part figures
Details	Gets the number of integer part figures.

GCSNCPLCTextboxSetDecimalWidth

Outline	Set decimal part width
Syntax	void GCSNCPLCTextboxSetDecimalWidth(GBaseObject **pPLCTextBox,* unsigned char *ucWidth*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation (i) unsigned char ucWidth : number of decimal part figures
Return value	None
Details	Sets the number of decimal part figures.

GCSNCPLCTextboxGetDecimalWidth

Outline	Get decimal part width
Syntax	unsigned char GCSNCPLCTextboxGetDecimalWidth(GBaseObject **pPLCTextBox*)
Argument	(i) GBaseObject *pPLCTextBox : object for operation
Return value	The number of decimal part figures
Details	Gets the number of decimal part figures.

GCSNCPLCTextboxSetScale

Outline Set scale

Syntax
```
void GCSNCPLCTextboxSetScale(
  GBaseObject     *pPLCTextBox,
  double          *pScale
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) double *pScale : scale

Return value None

Details Sets the scale.

GCSNCPLCTextboxGetScale

Outline Get scale

Syntax
```
void GCSNCPLCTextboxGetScale(
  GBaseObject     *pPLCTextBox,
  double          *pScale
)
```

Argument
(i) GbaseObject *pPLCTextBox : object for operation
(i) double *pScale : scale

Return value None

Details Gets the scale.

GCSNCPLCTextboxSetOffset

Outline Set offset

Syntax
```
void GCSNCPLCTextboxSetOffset(
  GBaseObject     *pPLCTextBox,
  long            lOffset
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) long lOffset : offset

Return value None

Details Sets the offset.

GCSNCPLCTextboxGetOffset

Outline Get offset

Syntax
```
long GCSNCPLCTextboxGetOffset(
  GBaseObject     *pPLCTextBox
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value Offset

Details Gets the offset.

GCSNCPLCTextboxSetDevice

Outline Set PLC device

Syntax
```
short GCSNCPLCTextboxSetDevice(
  GBaseObject     *pPLCTextBox,
  char            *szDevice
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) char *szDevice : PLC device

Return value
TRUE : acquisition succeeded
FALSE : acquisition failed

Details Sets the PLC device.

GCSNCPLCTextboxGetDevice

Outline Get PLC device

Syntax
```
short GCSNCPLCTextboxGetDevice(
  GBaseObject     *pPLCTextBox,
  char            *szDevice
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) char *szDevice : PLC device

Return value
TRUE : acquisition succeeded
FALSE : acquisition failed

Details Gets the PLC device.

GCSNCPLCTextboxSetTextSize

Outline
Set size

Syntax
```
void GCSNCPLCTextboxSetTextSize(
  GBaseObject      *pPLCTextBox,
  unsigned long    ulSize
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned long ulSize : PLC device size

Return value
None

Details
Sets the PLC device size.
One of the following values is set for ucType:
GPLCTEXT_SIZE_2 : 2 byte
GPLCTEXT_SIZE_4 : 4 byte
After the PLC device size is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetTextSize

Outline
Get size

Syntax
```
unsigned long GCSNCPLCTextboxGetTextSize(
  GBaseObject      *pPLCTextBox
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value
GPLCTEXT_SIZE_2 : 2 byte
GPLCTEXT_SIZE_4 : 4 byte
Besides the above : illegal setting

Details
Gets the PLC device size.

GCSNCPLCTextboxSetZeroSuppressStatus

Outline
Set zero suppress display status

Syntax
```
void GCSNCPLCTextboxSetZeroSuppressStatus(
  GBaseObject     *pPLCTextBox,
  short           fStatus
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) short fStatus : zero suppress display status
FALSE : normal display status
TRUE : zero suppress display status

Return value
None

Details
Sets whether the zero suppress will be displayed for text boxes or not.
After the zero suppress display status is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetZeroSuppressStatus

Outline
Get zero suppress display status

Syntax
```
short GCSNCPLCTextboxGetZeroSuppressStatus(
  GBaseObject     *pPLCTextBox
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value
FALSE : normal display status
TRUE : zero suppress display status

Details
Gets whether the test box display status is normal display status or the zero suppress display status.

GCSNCPLCTextboxSetInterlockDevice

Outline Set interlock device

Syntax
```
short GCSNCPLCTextboxSetInterlockDevice(
  GBaseObject  *pPLCTextBox,
  char         *szInterlockDevice
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) char *szInterlockDevice : interlock character string

Return value
FALSE : abnormal
TRUE : normal

Details Sets the interlock device address to allocate to the control.

GCSNCPLCTextboxGetInterlockDevice

Outline Get interlock device

Syntax
```
short GCSNCPLCTextboxGetInterlockDevice(
  GBaseObject  *pPLCTextBox,
  char         *pInterlockDevice
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(o) char *pInterlockDevice : interlock device character string storage

Return value
FALSE : abnormal
TRUE : normal

Details Gets the interlock device address allocated to the control.

GCSNCPLCTextboxSetInterlockDeviceAction

Outline Set interlock device operation

Syntax
```
short GCSNCPLCTextboxSetInterlockDeviceAction(
  GBaseObject     *pPLCTextBox,
  unsigned char   ucInterlockDeviceAction
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned char ucInterlockDeviceAction : interlock device operation

Return value
FALSE : abnormal
TRUE : normal

Details
Sets the interlock device operation.
One of the following values is set for ucInterlockDeviceAction:
0: none
1: ON
2: OFF
When a setting outside the range is made, the setting value is not changed and "abnormal" is returned in the return value.

GCSNCPLCTextboxGetInterlockDeviceAction

Outline Get interlock device operation

Syntax
```
unsigned char GCSNCPLCTextboxGetInterlockDeviceAction(
  GBaseObject     *pPLCTextBox,
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value
Getting result
0: none
1: ON
2: OFF

Details
Gets the PLC textbox operation setting.
When a value outside the range is got, the value is returned.

GCSNCPLCTextboxSetInterlockDeviceBitPos

Outline
Set bit position of interlock device

Syntax
```
short GCSNCPLCTextboxSetInterlockDeviceBitPos(
  GBaseObject     *pPLCTextBox,
  unsigned char   ucInterlockDeviceBitPos
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) unsigned char ucInterlockDeviceBitPos : bit position of interlock device

Return value
FALSE : abnormal
TRUE : normal

Details
Sets the bit position of the interlock device.
One of the following values is set for ucInterlockDeviceBitPos:
0 to 15
When a setting outside the range is made, the setting value is not changed and "Abnormal" is returned in the return value.

GCSNCPLCTextboxGetInterlockDeviceBitPos

Outline
Get bit position of interlock device

Syntax
```
unsigned char GCSNCPLCTextboxGetInterlockDeviceBitPos(
  GBaseObject  *pPLCTextBox
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value
Getting result
0 to 15

Details
Gets the bit position of the PLC textbox interlock device.
When a setting outside the range has been made, that setting value is returned.

GCSNCPLCTextboxSetInterlockDeviceProjectNo

Outline
Set project number of interlock device

Syntax
```
short GCSNCPLCTextboxSetInterlockDeviceProjectNo(
  GBaseObject     *pPLCTextBox,
  unsigned char   ucInterlockDeviceProjectNo
)
```

Argument

(i) GBaseObject *pPLCTextBox	:	object for operation
(i) unsigned char ucInterlockDeviceProjectNo	:	project number of interlock device

Return value
FALSE : Abnormal
TRUE : Normal

Details
Sets the project number of the interlock device.
One of the following values is set for ucInterlockDeviceProjectNo:
0 to 6
When a setting outside the range is made, the setting value is not changed and "Abnormal" is returned in the return value.

GCSNCPLCTextboxGetInterlockDeviceProjectNo

Outline
Get project number of interlock device

Syntax
```
unsigned char GCSNCPLCTextboxGetInterlockDeviceProjectNo(
  GBaseObject  *pPLCTextBox
)
```

Argument

(i) GBaseObject *pPLCTextBox	:	object for operation

Return value
Getting result
0 to 6

Details
Gets the project number of the PLC textbox interlock device.
When a setting outside the range is made, that setting value is returned.

GCSNCPLCTextboxSetInterlockColor

Outline Set background color during interlock

Syntax
```
void GCSNCPLCTextboxSetInterlockColor(
  GBaseObject  *pPLCTextBox,
  GColor       gcInterlockColor
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation
(i) GColor gcInterlockColor : color code for background color during interlock

Return value None

Details Sets the background color during interlock.
After the background color is set, the control area is registered as a redraw area.

GCSNCPLCTextboxGetInterlockColor

Outline Get background color during interlock

Syntax
```
GColor GCSNCPLCTextboxGetInterlockColor(
  GBaseObject  *pPLCTextBox
)
```

Argument
(i) GBaseObject *pPLCTextBox : object for operation

Return value Color code for background color during interlock

Details Gets the background color during interlock.

5.2.22 NCDataTextBox

GCSNCDataTextboxSetBorderID

Outline	Set 3D border resource ID
Syntax	void GCSNCDataTextboxSetBorderID(GBaseObject **pNCDataTextBox,* unsigned short *usID*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation (i) unsigned short usID : 3D border resource ID
Return value	None
Details	Sets the 3D border resource ID. After the 3D border resource ID is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetBorderID

Outline	Get 3D border resource ID
Syntax	unsigned short GCSNCDataTextboxGetBorderID(GBaseObject **pNCDataTextBox*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation
Return value	3D border resource ID
Details	Gets the 3D border resource ID.

GCSNCDataTextboxSetFontID

Outline Set font resource ID

Syntax
```
void GCSNCDataTextboxSetFontID(
  GBaseObject   *pNCDataTextBox,
  unsigned short  usID
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) unsigned short usID : font resource ID

Return value None

Details Sets the font resource ID.
After the font resource ID is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetFontID

Outline Get font resource ID

Syntax
```
unsigned short GCSNCDataTextboxGetFontID(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Font resource ID

Details Gets the font resource ID.

GCSNCDataTextboxSetCaption

Outline Set caption information

Syntax
```
void GCSNCDataTextboxSetCaption(
  GBaseObject  *pNCDataTextBox,
  GCaption     *pCaption
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) GCaption *pCaption : caption information

Return value None

Details Sets caption information.
After the caption information is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetCaption

Outline Get caption information

Syntax
```
GCaption *GCSNCDataTextboxGetCaption(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Caption information

Details Gets the caption information.

GCSNCDataTextboxSetFocusEffect

Outline Set effect during focus

Syntax
```
void GCSNCDataTextboxSetFocusEffect(
  GBaseObject    *pNCDataTextBox,
  unsigned char  ucFocusEffect
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) unsigned char ucFocusEffect : effect during focus

Return value None

Details Sets the effect during focus.
One of the following values is set for ucFocusEffect:
1: no effect
2: cursor display
3: select all
After the effect is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetFocusEffect

Outline Get effect during focus

Syntax
```
unsigned char GCSNCDataTextboxGetFocusEffect(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Effect used during focus

Details Gets the effect used during focus.

GCSNCDataTextboxSetFocusColor

Outline	Set background color during focus
Syntax	void GCSNCDataTextboxSetFocusColor(GBaseObject **pNCDataTextBox,* GColor *gcColor*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation (i) GColor gcColor : color code for background color during focus
Return value	None
Details	Sets the background color during focus. After the background color is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetFocusColor

Outline	Get background color during focus
Syntax	GColor GCSNCDataTextboxGetFocusColor(GBaseObject **pNCDataTextBox*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation
Return value	Color code for background color during focus
Details	Gets the background color during focus.

GCSNCDataTextboxSetNormalColor

Outline Set normal background color

Syntax
```
void GCSNCDataTextboxSetNormalColor(
  GBaseObject  *pNCDataTextBox,
  GColor       gcColor
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) GColor gcColor : color code for normal background color

Return value None

Details Sets the normal background color.
After the normal background color is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetNormalColor

Outline Get normal background color

Syntax
```
GColor GCSNCDataTextboxGetNormalColor(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Color code for normal background color

Details Gets the normal background color.

GCSNCDataTextboxSetDisableColor

Outline	Set background color when disabled
Syntax	void GCSNCDataTextboxSetDisableColor(GBaseObject **pNCDataTextBox,* GColor *gcColor*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation (i) GColor gcColor : color code for background color when disabled
Return value	None
Details	Sets the background color when disabled. After the background color is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetDisableColor

Outline	Get background color when disabled
Syntax	GColor GCSNCDataTextboxGetDisableColor(GBaseObject **pNCDataTextBox*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation
Return value	Color code for background color setting when disabled
Details	Gets the background color when disabled.

GCSNCDataTextboxSetTextType

Outline Set character string type

Syntax
```
void GCSNCDataTextboxSetTextType(
  GBaseObject    *pNCDataTextBox,
  unsigned char  ucType
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) unsigned char ucType : character string type

Return value None

Details Sets the character string type.
One of the following values is set for ucType:
0: character string
1: binary integer value
2: signed decimal integer value
3: unsigned decimal integer value
4: hexadecimal integer value
5: real number
After the character string type is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetTextType

Outline Get character string type

Syntax
```
unsigned char GCSNCDataTextboxGetTextType(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Character string type

Details Gets the display character string type.

GCSNCDataTextboxSetCommaStatus

Outline Set comma display status

Syntax
```
void GCSNCDataTextboxSetCommaStatus(
  GBaseObject  *pNCDataTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short fStatus : comma display status

Return value None

Details Sets whether commas will be displayed for text boxes or not.
One of the following values is set for fStatus:
0: normal display status
1: comma display status
After the comma display status is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetCommaStatus

Outline Get comma display status

Syntax
```
short GCSNCDataTextboxGetCommaStatus(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Comma display status

Details Gets the setting as to whether the test box display status is normal display status or the comma display status.

GCSNCDataTextboxSetRefuseInputNumberStatus

Outline Set numerical input disabled status

Syntax
```
void GCSNCDataTextboxSetRefuseInputNumberStatus(
  GBaseObject  *pNCDataTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short fStatus : number input status

Return value None

Details Sets whether numerical input will be rejected for text boxes or not.
One of the following values is set for fStatus:
0: input enabled
1: input disabled

GCSNCDataTextboxGetRefuseInputNumberStatus

Outline Get numerical input disabled status

Syntax
```
short GCSNCDataTextboxGetRefuseInputNumberStatus(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Numerical input disabled status

Details Gets the setting as to whether numerical input will be rejected for text boxes or not.

GCSNCDataTextboxSetRefuseInputSmallLetterStatus

Outline Set one-byte lower case character input disabled status

Syntax
```
void GCSNCDataTextboxSetRefuseInputSmallLetterStatus(
  GBaseObject  *pNCDataTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short fStatus : one-byte lower case character input status

Return value None

Details Sets whether one-byte lower case character input will be rejected for text boxes or not.
One of the following values is set for fStatus:
0: input enabled
1: input disabled

GCSNCDataTextboxGetRefuseInputSmallLetterStatus

Outline Get one-byte lower case character input disabled status

Syntax
```
short GCSNCDataTextboxGetRefuseInputSmallLetterStatus(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value One-byte lower case character input disabled status

Details Gets the setting as to whether one-byte lower case character input will be rejected for text boxes or not.

GCSNCDataTextboxSetRefuseInputCapitalLetterStatus

Outline Set one-byte upper case character input disabled status

Syntax
```
void GCSNCDataTextboxSetRefuseInputCapitalLetterStatus(
  GBaseObject  *pNCDataTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short fStatus : one-byte upper case character input status

Return value None

Details Sets whether one-byte upper case character input will be rejected for text boxes or not.
One of the following values is set for fStatus:
0: input enabled
1: input disabled

GCSNCDataTextboxGetRefuseInputCapitalLetterStatus

Outline Get one-byte upper case character input disabled status

Syntax
```
short GCSNCDataTextboxGetRefuseInputCapitalLetterStatus(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value One-byte upper case character input disabled status

Details Gets the setting as to whether one-byte upper case character input will be rejected for text boxes or not.

GCSNCDataTextboxSetRefuseInputSymbolLetterStatus

Outline Set one-byte symbol input disabled status

Syntax
```
void GCSNCDataTextboxSetRefuseInputSymbolLetterStatus(
  GBaseObject  *pNCDataTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short fStatus : one-byte symbol input status

Return value None

Details Sets whether one-byte symbol input will be rejected for text boxes or not.
One of the following values is set for fStatus:
0: input enabled
1: input disabled

GCSNCDataTextboxGetRefuseInputSymbolLetterStatus

Outline Get one-byte symbol input disabled status

Syntax
```
short GCSNCDataTextboxGetRefuseInputSymbolLetterStatus(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value One-byte symbol input disabled status

Details Gets the setting as to whether one-byte symbol input will be rejected for text boxes or not.

GCSNCDataTextboxSetZeroSuppressStatus

Outline Set zero suppress display status

Syntax
```
void GCSNCDataTextboxSetZeroSuppressStatus(
  GBaseObject  *pNCDataTextBox,
  short        fStatus
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short fStatus : zero suppress display status

Return value None

Details Sets whether to enable zero suppress display for text boxes or not.
One of the following values is set for fStatus:
0: normal display status
1: zero suppress display status

GCSNCDataTextboxGetZeroSuppressStatus

Outline Get zero suppress display status

Syntax
```
short GCSNCDataTextboxGetZeroSuppressStatus(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Zero suppress display status

Details Gets whether the text box display is in normal display status or in zero suppress display status.

GCSNCDataTextboxSetString

Outline Set display character string

Syntax
```
short GCSNCDataTextboxSetString(
  GBaseObject       *pNCDataTextBox,
  const GTCHAR      *pString
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) const GTCHAR *pString : display character string

Return value
0 : setting failed
1 : setting succeeded

Details Sets the display character string.
After the display character string is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetString

Outline Get display character string

Syntax
```
short GCSNCDataTextboxGetString(
  GBaseObject   *pNCDataTextBox,
  GTCHAR        *pString,
  long          lMaxCount
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(o) GTCHAR *pString : display character string
(i) long lMaxCount : length of the acquired character string

Return value
0 : acquisition failed
1 : acquisition succeeded

Details Stores the display character string in pString.

GCSNCDataTextboxSetStringBuffer

Outline

Set input character string

Syntax

```
short GCSNCDataTextboxSetStringBuffer(
  GBaseObject    *pNCDataTextBox,
  unsigned long  ulBufferSize,
  GTCHAR         *pStringBuffer
)
```

Argument

(i) GBaseObject *pNCDataTextBox : object for operation
(i) unsigned long ulBufferSize : length of input character string
(i) GTCHAR *pStringBuffer : input character string

Return value

0: setting failed
1: setting succeeded

Details

Enter [INPUT] to write the input character string to the NC.
After the input character string is set, the control area is registered as a redraw area.

(Note 1) The input character string will be displayed only when the focus is placed on the text box. When the focus is not placed on the text box, display data is rewritten according to the settings of the display update cycle.

(Note 2) The length of input character string is limited by the property setting "Number of the maximum characters". If the input character string exceeds the limit, only the maximum number of characters from the top will be set.

(Note 3) When the securing of memory fails, or a character that is not allowed by the property is set, the setting fails and 0 is returned as a return value.

GCSNCDataTextboxGetStringBuffer

Outline

Get input character string

Syntax

```
GTCHAR *GCSNCDataTextboxGetStringBuffer(
  GBaseObject    *pNCDataTextBox,
  unsigned long  *ulBufferSize
)
```

Argument

(i) GBaseObject *pNCDataTextBox : object for operation
(o) unsigned long *ulBufferSize : pointer for saving the length of the acquired character string

Return value

Input character string

Details

Gets the input character string when the focus is placed on the NC data text box for operation.
When the focus is not placed on the textbox, the display character string which matches the NC data will be acquired.

GCSNCDataTextboxSetGNCValue

Outline Set display value

Syntax
```
short GCSNCDataTextboxSetGNCValue(
  GBaseObject  *pNCDataTextBox,
  GNCValue     gvValue
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) GNCValue gvValue : setting value

Return value
0: setting failed
1: setting succeeded

Details Sets the value displayed in a text box.
Numbers are converted to character strings according to the display format.
After the value displayed in a text box is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetGNCValue

Outline Get display value

Syntax
```
short GCSNCDataTextboxGetGNCValue(
  GBaseObject  *pNCDataTextBox,
  GNCValue     *gvValue
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(o) GNCValue *gvValue : display value

Return value
0: acquisition failed
1: acquisition succeeded
2: variable <empty>

Details Gets the value displayed in a text box.
Character strings are converted to numbers according to the display format.

GCSNCDataTextboxSetBackGroundPattern

Outline Set background fill status

Syntax
```
void GCSNCDataTextboxSetBackGroundPattern(
  GBaseObject  *pNCDataTextBox,
  short        nBackGroundPattern
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) short nBackGroundPattern : background fill status

Return value None

Details Sets the background color fill status.
One of the following values is set for nBackGroundPattern:
-1: no fill
-2: with background fill
After the background fill status is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetBackGroundPattern

Outline Get background fill status

Syntax
```
short GCSNCDataTextboxGetBackGroundPattern(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Background fill status

Details Gets the background color fill status.

GCSNCDataTextboxSetIntegerWidth

Outline	Set integer part width
Syntax	void GCSNCDataTextboxSetIntegerWidth(GBaseObject **pNCDataTextBox,* unsigned char *ucIntegerWidth*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation (i) unsigned char ucIntegerWidth : the number of integer part figures
Return value	None
Details	Sets the number of integer part figures.

GCSNCDataTextboxGetIntegerWidth

Outline	Get integer part width
Syntax	unsigned char GCSNCDataTextboxGetIntegerWidth(GBaseObject **pNCDataTextBox*)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation
Return value	The number of integer part figures
Details	Gets the number of integer part figures.

GCSNCDataTextboxSetDecimalWidth

Outline Set decimal part width

Syntax
```
void GCSNCDataTextboxSetDecimalWidth(
  GBaseObject    *pNCDataTextBox,
  unsigned char  ucDecimalWidth
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) unsigned char ucDecimalWidth : the number of decimal part figures

Return value None

Details Sets the number of decimal part figures.

GCSNCDataTextboxGetDecimalWidth

Outline Get decimal part width

Syntax
```
unsigned char GCSNCDataTextboxGetDecimalWidth(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value The number of decimal part figures

Details Gets the number of decimal part figures.

GCSNCDataTextboxSetScale

Outline	Set scale
Syntax	void GCSNCDataTextboxSetScale(GBaseObject *pNCDataTextBox, double *pdScale)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation (i) double *pdScale : scale of fractional part
Return value	None
Details	Sets the scale.

GCSNCDataTextboxGetScale

Outline	Get scale
Syntax	void GCSNCDataTextboxGetScale(GBaseObject *pNCDataTextBox, double *pdScale)
Argument	(i) GBaseObject *pNCDataTextBox : object for operation (o) double *pdScale : scale of fractional part
Return value	None
Details	Gets the scale.

GCSNCDataTextboxSetOffset

Outline Set offset

Syntax
```
void GCSNCDataTextboxSetOffset(
  GBaseObject  *pNCDataTextBox,
  long         lOffset
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) long lOffset : offset

Return value None

Details Sets the offset.

GCSNCDataTextboxGetOffset

Outline Get offset

Syntax
```
long GCSNCDataTextboxGetOffset(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Offset

Details Gets the offset.

GCSNCDataTextboxSetSystemNumber

Outline Set part system number

Syntax
```
void GCSNCDataTextboxSetSystemNumber(
  GBaseObject  *pNCDataTextBox,
  long         lMachine
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) long lMachine : part system number

Return value None

Details Sets the part system number.
One of the following values is set for lMachine:
1 to 10 : part system number
After the part system number is set, the control area is registered as a redraw area.

GCSNCDataTextboxGetSystemNumber

Outline Get part system number

Syntax
```
long GCSNCDataTextboxGetSystemNumber(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Part system number

Details Gets the part system number.

GCSNCDataTextboxSetGround

Outline Set ground information

Syntax
```
void GCSNCDataTextboxSetGround(
  GBaseObject  *pNCDataTextBox,
  long         lGround
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) long lGround : ground information

Return value None

Details Sets the ground information.
One of the following values is set for lGround:
0 : foreground
1 : background

GCSNCDataTextboxGetGround

Outline Get ground information

Syntax
```
long GCSNCDataTextboxGetGround(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Ground information

Details Gets the ground information.

GCSNCDataTextboxSetAxisInfo

Outline Set axis information

Syntax
```
void GCSNCDataTextboxSetAxisInfo(
  GBaseObject    *pNCDataTextBox,
  unsigned long  ulAxis
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) unsigned long ulAxis : axis information

Return value None

Details Sets the axis information.
One of the following values is set for ulAxis:
0 : no axis designation
1 to 32 : axis number

GCSNCDataTextboxGetAxisInfo

Outline Get axis information

Syntax
```
unsigned long GCSNCDataTextboxGetAxisInfo(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Axis information

Details Gets the axis information.

GCSNCDataTextboxSetSection

Outline Set section No.

Syntax
```
void GCSNCDataTextboxSetSection(
  GBaseObject  *pNCDataTextBox,
  long         lSection
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) long lSection : section No.

Return value None

Details Sets the section No.
One of the following values is set for lSection:
0 to 999 : section No.

GCSNCDataTextboxGetSection

Outline Get section No.

Syntax
```
long GCSNCDataTextboxGetSection(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Section No.

Details Gets the section No.

GCSNCDataTextboxSetSubSection

Outline Set sub-section No.

Syntax
```
void GCSNCDataTextboxSetSubSection(
  GBaseObject  *pNCDataTextBox,
  long         lSubSection
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) long lSubSection : sub-section No.

Return value None

Details Sets the sub-section No.
One of the following values is set for lSubSection:
0 to 1000000000 : sub-section No.

GCSNCDataTextboxGetSubSection

Outline Get sub-section No.

Syntax
```
long GCSNCDataTextboxGetSubSection(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Sub-section No.

Details Gets the sub-section No.

GCSNCDataTextboxSetDataType

Outline Set data type

Syntax
```
void GCSNCDataTextboxSetDataType(
  GBaseObject  *pNCDataTextBox,
  long         lDataType
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation
(i) long lDataType : data type

Return value None

Details Sets the data type.
One of the following values is set for lDataType:
0x1 : 1-byte integer type
0x2 : 2-byte integer type
0x3 : 4-byte integer type
0x5 : 8-byte real type
0x10 : character string type

GCSNCDataTextboxGetDataType

Outline Get data type

Syntax
```
long GCSNCDataTextboxGetDataType(
  GBaseObject  *pNCDataTextBox
)
```

Argument
(i) GBaseObject *pNCDataTextBox : object for operation

Return value Data type

Details Gets the data type.

5.2.23 Menu

GCSRefresh

Outline	Redraw menu
Syntax	void GCSRefresh(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	None
Details	Redraws the menu button.

GCSSetRefreshValidFlag

Outline	Set menu display
Syntax	void GCSSetRefreshValidFlag(GBaseObject *pSelf, short _RefreshValidFlag)
Argument	(i) GBaseObject *pSelf : object for operation (i) short _RefreshValidFlag : display valid flag
Return value	None
Details	Sets whether to enable or disable the menu display. One of the following values is set for _RefreshValidFlag: 1 : menu display enabled 0 : menu display disabled When the menu display is changed from disabled to enabled, always execute GCSRefresh().

GCSSetMenuButtonName_all

Outline

Set character string on one-row menu (10 menus)

Syntax

```
long GCSSetMenuButtonName_all(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  char                 *_ptbMenuString[10]
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) char *_ptbMenuString[10]	: character string to be set

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010004	: length of the button name is illegal
0x00010005	: button type is illegal

Details

Sets the character strings (names) for all the menu buttons (10 buttons) of one-row menu.

Up to seven characters can be displayed in 1 row for the menu name. Put the line break code between the rows to make two rows.

GCSSetMenuButtonUpperName_all

Outline

Set character string on upper row of two-row menu (10 menus)

Syntax

```
long GCSSetMenuButtonUpperName_all(
  GBaseObject              *pSelf,
  const unsigned short     _unDispMenuGroupID,
  char                     *_ptbMenuString[10]
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) char *_ptbMenuString[10]	: character string to be set

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010004	: length of the button name is illegal
0x00010005	: button type is illegal

Details

Sets the character strings (names) for all the menu buttons (10 buttons) of the upper row of two-row menu.

Up to seven characters can be displayed in 1 row for the menu name.

GCSSetMenuButtonLowerName_all

Outline

Set character string on lower row of two-row menu (10 menus)

Syntax

```
long GCSSetMenuButtonLowerName_all(
  GBaseObject              *pSelf,
  const unsigned short     _unDispMenuGroupID,
  char                     *_ptbMenuString[10]
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) char *_ptbMenuString[10]	: character string to be set

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010004	: length of the button name is illegal
0x00010005	: button type is illegal

Details

Sets the character strings (names) for all the menu buttons (10 buttons) of the lower row of two-row menu.

Seven characters maximum can be displayed on a row for the menu name.
Put the line break code between the rows to make two rows.

GCSSetMenuButtonState_one

Outline

Set state of one-row menu (one menu)

Syntax

```
long GCSSetMenuButtonState_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  const long           _lMenuStat
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) const long _lMenuStat	: menu selection status

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010005	: button type is illegal
0x00010009	: status of the button is illegal

Details

Sets the selection state for the specified one-row menu button (one button).

One of the following values is set for _lMenuStat:

0 : Menu button OFF (not selected, normal)

1 : Menu button ON (selected)

-1 : Menu button disabled

GCSSetMenuButtonUpperName_one

Outline

Set character string on upper row of two-row menu (one menu)

Syntax

```
long GCSSetMenuButtonUpperName_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  char*                _pMenuString
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) char* _pMenuString	: character string to be set

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010004	: length of the button name is illegal
0x00010005	: button type is illegal

Details

Sets the character string (name) for the specified menu button (one button) of the upper row of two-row menu.

Up to seven characters can be displayed in 1 row for the menu name.

GCSSetMenuButtonLowerName_one

Outline

Set character string on lower row of two-row menu (one menu)

Syntax

```
long GCSSetMenuButtonLowerName_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  char*                _pMenuString
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) char* _pMenuString	: character string to be set

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010004	: length of the button name is illegal
0x00010005	: button type is illegal

Details

Sets the character string (name) for the specified menu button (one button) of the lower row of two-row menu.

Seven characters maximum can be displayed on a row for the menu name.
Put the line break code between the rows to make two rows.

GCSSetMenuButtonUpperState_one

Outline

Set state of upper row of two-row menu (one menu)

Syntax

```
long GCSSetMenuButtonUpperState_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  const long           _lMenuStat
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) const long _lMenuStat	: menu selection status

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010005	: button type is illegal

Details

Sets the selection state for the upper row of the specified two-row menu button (one button).

One of the following values is set for _lMenuStat:

0 : menu button OFF (not selected, normal)

1 : menu button ON (selected)

-1 : menu button disabled

GCSSetMenuButtonLowerState_one

Outline

Set state of lower row of two-row menu (one menu)

Syntax

```
long GCSSetMenuButtonLowerState_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  const long           _lMenuStat
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) const long _lMenuStat	: menu selection status

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010005	: button type is illegal
0x00010009	: status of the button is illegal

Details

Sets the selection state for the lower row of the specified two-row menu button (one button).

One of the following values is set for _lMenuStat:

0 : menu button OFF (not selected, normal)

1 : menu button ON (selected)

-1 : menu button disabled

GCSGetMenuButtonState_one

Outline

Get state of one-row menu (one menu)

Syntax

```
long GCSGetMenuButtonState_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010005	: button type is illegal

Details

Gets the selection state for the specified one-row menu button (one button).

GCSGetMenuButtonUpperState_one

Outline

Get state of upper row of two-row menu (one menu)

Syntax

```
long GCSGetMenuButtonUpperState_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010005	: button type is illegal

Details

Gets the selection state for the upper row of the specified two-row menu button (one button).

GCSGetMenuButtonLowerState_one

Outline

Get state of lower row of two-row menu (one menu)

Syntax

```
long GCSGetMenuButtonLowerState_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.

Return value

Setting result

0	: normal end
0x00010002	: menu ID error
0x00010005	: button type is illegal

Details

Gets the selection state for the lower row of the specified two-row menu button (one button).

GCSSetMenuButtonOffIcon_one

Outline

Set icon of one-row menu (one menu) at OFF

Syntax

```
long GCSSetMenuButtonOffIcon_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  const short          _nIconID
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) const short _nIconID	: icon resource ID at the time of OFF

Return value

Setting result

0	: normal end
0x00010002	: menu ID error

Details

Sets the icon resource ID at the time of OFF for the specified one-row menu button (one button).

GCSSetMenuButtonOnIcon_one

Outline Set icon of one-row menu (one menu) at ON

Syntax

```
long GCSSetMenuButtonOnIcon_one(
  GBaseObject          *pSelf,
  const unsigned short _unDispMenuGroupID,
  const short          _nMenuNo,
  const short          _nIconID
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const unsigned short _unDispMenuGroupID	: menu group ID (fixed to 1)
(i) const short _nMenuNo	: menu No.
(i) const short _nIconID	: icon resource ID at the time of ON

Return value Setting result

0	: normal end
0x00010002	: menu ID error

Details Sets the icon resource ID at the time of ON for the specified one-row menu button (one button).

GCSChangeMenuType

Outline Switch display between one-row menu and two-row menu

Syntax
```
long GCSChangeMenuType(
  GBaseObject  *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value 0: normal end

Details Switches the display between the one-row menu and the two-row menu.
Switching can be executed between the following menu types:
- One-row menu (VGA) (Classic) <--> Two-row menu (VGA) (Classic)
- One-row menu (XGA) (Classic) <--> Two-row menu (XGA) (Classic)
- One-row menu (VGA) <--> Two-row menu (VGA)
- One-row menu (XGA) <--> Two-row menu (XGA)

GCSSetDstObject

Outline Specify called control

Syntax
```
long GCSSetDstObject(
  GBaseObject  *pSelf,
  GBaseObject  *pDestObj
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) GBaseObject *pDestObj : called control object

Return value Setting result
0: object for operation and the called control object are same
1: normal end

Details Specify the called control.
(Note) After the specified called control is called, the called control will be focused.

GCSSetNextMenuButtonState

Outline

Set menu switching button selection state

Syntax

```
long GCSSetNextMenuButtonState(
  GBaseObject  *pSelf,
  const long       _eButtonState,
  const long       _lType
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) const long _eButtonState : a button selection state
(i) const long _lType : type

Return value

0: normal end

Details

Sets the selection state for the menu switching button.
Set any of the following values for_eButtonState:
0 : Button OFF (Not selected, default)
-1 : Button disabled
Set any of the following values for_lType:
1 : Does not update drawing
0 : Updates drawing

GCSSetOpeMessage

Outline

Set menu operation message

Syntax

```
long GCSSetOpeMessage(
  GBaseObject  *pSelf,
  char*        _pOpeMessageString
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) char* _pOpeMessageString : operation message

Return value

0: normal end

Details

Sets operation messages to be displayed on menu.
Up to 40 characters can be used for an operation message.

GCSBeginAnmState

Outline

Start menu animated motion setting

Syntax

```
long GCSBeginAnmState(
  GBaseObject  *pSelf,
  short        nMenuAnmType
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) short nMenuAnmType : type of menu animation

Return value

0: normal end

Details

Starts the setting for animated motion.
After the start setting for animated motion has been completed, set the updating function for the menu to be displayed by animation (setting of name, normal/highlighted/hidden display, icons, etc.). After the GCSEndAnmState() setting has been completed, animated motion starts.

Set any of the following values for nMenuAnimType:
0: no animation
1: rightward sliding
2: upward sliding
3: downward sliding

(Note) If the operation parameter (#8976 Menu animation disabled) is enabled, menu animation is not executed.

GCSEndAnmState

Outline Complete menu animated motion setting

Syntax
```
long GCSEndAnmState(
  GBaseObject  *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value 0: normal end

Details Sets the completion for animated motion setting.
Set the updating function for the menu to be displayed by animation (setting of name, normal/highlighted/hidden display, icons, etc.). Calling this function starts animated motion.

(Note) If the control parameter (#8976 Menu Anim Invalid) is enabled, menu animation is not executed.

GCSStopAnm

Outline Stop menu animation

Syntax
```
long GCSStopAnm(
  GBaseObject  *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value 0: normal end

Details Stops menu animation and displays the menu after the state has been changed.
Use this function to stop menu animation that is running using GCSEndAnmState().
If you call this function when menu animation is not executed, the stop process is not executed and the program ends normally.

5.2.24 Extension Menu

GCSNCExMenuSetFontID

Outline	Set font resource ID
Syntax	void GCSNCExMenuSetFontID(GBaseObject **pSelf,* unsigned short *wID*)
Argument	(i) GBaseObject *pSelf : object for operation (i) unsigned short wID : font resource ID
Return value	None
Details	Sets the font resource ID.

GCSNCExMenuGetFontID

Outline	Get font resource ID
Syntax	unsigned short GCSNCExMenuGetFontID(GBaseObject **pSelf*)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Font resource ID
Details	Gets the font resource ID.

GCSNCExMenuGetMenuNumFromPoint

Outline Get menu No. from mouse press position

Syntax
```
unsigned short GCSNCExMenuGetMenuNumFromPoint(
  GBaseObject  *pSelf,
  short        nPointX,
  short        nPointY
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) short nPointX : X coordinate of pointing device
(i) short nPointY : Y coordinate of pointing device

Return value Button No.

Menu button No.	: 1 to 10
Menu return button	: 11
Menu switching button	: 12
No press button, or "display status" property is set to other than "full display"	: 0

Details Gets the menu No. from the mouse press position.
The upper left of the control is the origin (0, 0) of the pointing device coordinates.

GCSNCExMenuGetDispMenuGroup

Outline Get menu group currently displayed

Syntax
```
unsigned short GCSNCExMenuGetDispMenuGroup(
  GBaseObject  *pSelf
)
```

Argument (i) GBaseObject *pSelf : object for operation

Return value Menu group No. (1 to 10)

Details Gets the menu group No. currently displayed.

GCSNCExMenuSetDisplay

Outline

Set display type of extension menu

Syntax

```
void GCSNCExMenuSetDisplay(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned char   ucDType
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned char ucDType	: display type

Return value

None

Details

Sets the display type of extension menu.
Sets ucDType to one of the following values:

0 : square
1 : image

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetDisplay

Outline

Get display type of extension menu

Syntax

```
unsigned char GCSNCExMenuGetDisplay(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) unsigned short wMBNum : menu button No. (1 to 10)

Return value

Display type
0 : square
1 : image

Details

Gets the display type of extension menu.
When a setting outside the range is made, "0: square" is returned.

GCSNCExMenuSetUpperMenuOnPattern

Outline Set pattern at the time of upper row extension menu ON

Syntax
```
void GCSNCExMenuSetUpperMenuOnPattern(
  GBaseObject    *pSelf,
  unsigned short wMGNum,
  short          nPattern
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) short nPattern : pattern

Return value None

Details Sets the pattern at the time of the upper row extension menu ON.
One of the following values is set for nPattern:

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOnPattern

Outline Get pattern at the time of upper row extension menu ON

Syntax
```
short GCSNCExMenuGetUpperMenuOnPattern(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Pattern

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

Details Gets the pattern at the time of the upper row extension menu ON.
When a setting outside the range is made, "0: background painting out" is returned.

GCSNCExMenuSetLowerMenuOnPattern

Outline Set pattern at the time of lower row extension menu ON

Syntax
```
void GCSNCExMenuSetLowerMenuOnPattern(
  GBaseObject    *pSelf,
  unsigned short wMGNum,
  short          nPattern
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) short nPattern : pattern

Return value None

Details Sets the pattern at the time of the lower row extension menu ON.
One of the following values is set for nPattern:

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOnPattern

Outline Get pattern at the time of lower row extension menu ON

Syntax
```
short GCSNCExMenuGetLowerMenuOnPattern(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Pattern

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

Details Gets the pattern at the time of the lower row extension menu ON.
When a setting outside the range is made, "0: Background painting out" is returned.

GCSNCExMenuSetUpperMenuOnForeColor

Outline Set foreground color at the time of upper row extension menu ON

Syntax
```
void GCSNCExMenuSetUpperMenuOnForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lForeColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lForeColor : foreground color

Return value None

Details Sets the foreground color at the time of the upper row extension menu ON.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOnForeColor

Outline Get foreground color at the time of upper row extension menu ON

Syntax
```
long GCSNCExMenuGetUpperMenuOnForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Foreground color

Details Gets the foreground color at the time of the upper row extension menu ON.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetLowerMenuOnForeColor

Outline Set foreground color at the time of lower row extension menu ON

Syntax
```
void GCSNCExMenuSetLowerMenuOnForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lForeColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lForeColor : foreground color

Return value None

Details Sets the foreground color at the time of the lower row extension menu ON.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOnForeColor

Outline Get foreground color at the time of lower row extension menu ON

Syntax
```
long GCSNCExMenuGetLowerMenuOnForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Foreground color

Details Gets the foreground color at the time of the lower row extension menu ON.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetUpperMenuOnBackColor

Outline Set background color at the time of upper row extension menu ON

Syntax
```
void GCSNCExMenuSetUpperMenuOnBackColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lBackColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lBackColor : background color

Return value None

Details Sets the background color at the time of the upper row extension menu ON.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOnBackColor

Outline Get background color at the time of upper row extension menu ON

Syntax
```
long GCSNCExMenuGetUpperMenuOnBackColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Background color

Details Gets the background color at the time of the upper row extension menu ON.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetLowerMenuOnBackColor

Outline Set background color at the time of lower row extension menu ON

Syntax
```
void GCSNCExMenuSetLowerMenuOnBackColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lBackColor
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)
(i) long lBackColor	:	background color

Return value None

Details Sets the background color at the time of the lower row extension menu ON.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOnBackColor

Outline Get background color at the time of lower row extension menu ON

Syntax
```
long GCSNCExMenuGetLowerMenuOnBackColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)

Return value Background color

Details Gets the background color at the time of the lower row extension menu ON.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetMenuOnImage

Outline

Set image ID at the time of extension menu ON

Syntax

```
void GCSNCExMenuSetMenuOnImage(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wImageID
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) unsigned short wMBNum : menu button No. (1 to 10)
(i) unsigned short wImageID : image ID

Return value

None

Details

Sets the image ID at the time of the extension menu ON.
One of the following values is set for usImageID:

0 : no image
1 or more : user definition image ID

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuOnImage

Outline

Get image ID at the time of extension menu ON

Syntax

```
unsigned short GCSNCExMenuGetMenuOnImage(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) unsigned short wMBNum : menu button No. (1 to 10)

Return value

Image ID
0 : no image
1 or more : user definition image ID

Details

Gets the image ID at the time of the extension menu ON.
When a setting outside the range is made, 0 is returned.

GCSNCExMenuSetUpperMenuOnStringColor

Outline Set character color at the time of upper row extension menu ON

Syntax
```
void GCSNCExMenuSetUpperMenuOnStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lStrColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lStrColor : character color

Return value None

Details Sets the character color at the time of the upper row extension menu ON.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOnStringColor

Outline Get character color at the time of upper row extension menu ON

Syntax
```
long GCSNCExMenuGetUpperMenuOnStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Character color

Details Gets the character color at the time of the upper row extension menu ON.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetLowerMenuOnStringColor

Outline Set character color at the time of lower row extension menu ON

Syntax
```
void GCSNCExMenuSetLowerMenuOnStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lStrColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lStrColor : character color

Return value None

Details Sets the character color at the time of the lower row extension menu ON.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOnStringColor

Outline Get character color at the time of lower row extension menu ON

Syntax
```
long GCSNCExMenuGetLowerMenuOnStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Character color

Details Gets the character color at the time of the lower row extension menu ON.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetUpperMenuOffPattern

Outline Set pattern at the time of upper row extension menu OFF

Syntax
```
void GCSNCExMenuSetUpperMenuOffPattern(
  GBaseObject    *pSelf,
  unsigned short wMGNum,
  short          nPattern
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)
(i) short nPattern	:	pattern setting value

Return value None

Details Sets the pattern at the time of the upper row extension menu OFF.
One of the following values is set for nPattern:

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOffPattern

Outline Get pattern at the time of upper row extension menu OFF

Syntax
```
short GCSNCExMenuGetUpperMenuOffPattern(
  GBaseObject     *pSelf,
  unsigned short  wMGNum
 )
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Pattern

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

Details Gets the pattern at the time of the upper row extension menu OFF.
When a setting outside the range is made, "0: background painting out" is returned.

GCSNCExMenuSetLowerMenuOffPattern

Outline Set pattern at the time of lower row extension menu OFF

Syntax
```
void GCSNCExMenuSetLowerMenuOffPattern(
  GBaseObject    *pSelf,
  unsigned short wMGNum,
  short          nPattern
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) short nPattern : pattern setting value

Return value None

Details Sets the pattern at the time of the lower row extension menu OFF.
One of the following values is set for nPattern:

0	: background painting out
1	: foreground painting out
2 to 39	: pattern 0 to 37
40	: with no painting out

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOffPattern

Outline

Get pattern at the time of lower row extension menu OFF

Syntax

```
short GCSNCExMenuGetLowerMenuOffPattern(
  GBaseObject     *pSelf,
  unsigned short  wMGNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value

Pattern
0 : background painting out
1 : foreground painting out
2 to 39 : pattern 0 to 37
40 : with no painting out

Details

Gets the pattern at the time of lower row extension menu OFF.
When a setting outside the range is made, "0: background painting out" is returned.

GCSNCExMenuSetUpperMenuOffForeColor

Outline Set foreground color at the time of upper row extension menu OFF

Syntax
```
void GCSNCExMenuSetUpperMenuOffForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lForeColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lForeColor : foreground color

Return value None

Details Sets the foreground color at the time of the upper row extension menu OFF.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOffForeColor

Outline Get foreground color at the time of upper row extension menu OFF

Syntax
```
long GCSNCExMenuGetUpperMenuOffForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Foreground color

Details Gets the foreground color at the time of the upper row extension menu OFF.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetLowerMenuOffForeColor

Outline Set foreground color at the time of lower row extension menu OFF

Syntax
```
void GCSNCExMenuSetLowerMenuOffForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lForeColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lForeColor : foreground color

Return value None

Details Sets the foreground color at the time of lower row extension menu OFF.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOffForeColor

Outline Get foreground color at the time of lower row extension menu OFF

Syntax
```
long GCSNCExMenuGetLowerMenuOffForeColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Foreground color

Details Gets the foreground color at the time of lower row extension menu OFF.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetUpperMenuOffBackColor

Outline Set background color at the time of upper row extension menu OFF

Syntax
```
void GCSNCExMenuSetUpperMenuOffBackColor(
  GBaseObject    *pSelf,
  unsigned short wMGNum,
  long           lBackColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lBackColor : background color

Return value None

Details Sets the background color at the time of upper row extension menu OFF.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOffBackColor

Outline Get background color at the time of upper row extension menu OFF

Syntax
```
long GCSNCExMenuGetUpperMenuOffBackColor(
  GBaseObject    *pSelf,
  unsigned short wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Background color

Details Gets the background color at the time of upper row extension menu OFF.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetLowerMenuOffBackColor

Outline Set background color at the time of lower row extension menu OFF

Syntax
```
void GCSNCExMenuSetLowerMenuOffBackColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lBackColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lBackColor : background color

Return value None

Details Sets the background color at the time of lower row extension menu OFF.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOffBackColor

Outline Get background color at the time of lower row extension menu OFF

Syntax
```
long GCSNCExMenuGetLowerMenuOffBackColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Background color

Details Gets the background color at the time of lower row extension menu OFF.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetMenuOffImage

Outline

Set image ID at the time of extension menu OFF

Syntax

```
void GCSNCExMenuSetMenuOffImage(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wImageID
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned short wImageID	: image ID

Return value

None

Details

Sets the image ID at the time of the extension menu OFF.
One of the following values is set for usImageID:
0: no image
1 or more: user definition image ID
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuOffImage

Outline

Get image ID at the time of extension menu OFF

Syntax

```
unsigned short GCSNCExMenuGetMenuOffImage(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) unsigned short wMBNum : menu button No. (1 to 10)

Return value

Image ID
0: no image
1 or more: user definition image ID

Details

Gets the image ID at the time of the extension menu OFF.
When a setting outside the range is made, 0 is returned.

GCSNCExMenuSetUpperMenuOffStringColor

Outline Set character color at the time of upper row extension menu OFF

Syntax
```
void GCSNCExMenuSetUpperMenuOffStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lStrColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lStrColor : character color

Return value None

Details Sets the character color at the time of upper row extension menu OFF.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuOffStringColor

Outline Get character color at the time of upper row extension menu OFF

Syntax
```
long GCSNCExMenuGetUpperMenuOffStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Character color

Details Gets the character color at the time of upper row extension menu OFF.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetLowerMenuOffStringColor

Outline Set character color at the time of lower row extension menu OFF

Syntax
```
void GCSNCExMenuSetLowerMenuOffStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  long            lStrColor
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) long lStrColor : character color

Return value None

Details Sets the character color at the time of lower row extension menu OFF.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetLowerMenuOffStringColor

Outline Get character color at the time of lower row extension menu OFF

Syntax
```
long GCSNCExMenuGetLowerMenuOffStringColor(
  GBaseObject    *pSelf,
  unsigned short  wMGNum
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)

Return value Character color

Details Gets the character color at the time of lower row extension menu OFF.
When a setting outside the range is made, "0: black" is returned.

GCSNCExMenuSetMenuAction

Outline

Set button operations of extension menu

Syntax

```
void GCSNCExMenuSetMenuAction(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wAction
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned short wAction	: button operation

Return value

None

Details

Sets the button operation of the extension menu.
One of the following values is set for wAction:

0: no operation
1: momentary
2: alternate
3: button group 1
4: button group 2
5: button group 3
6: button group 4
7: button group 5

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuAction

Outline

Get button operations of extension menu

Syntax

```
unsigned short GCSNCExMenuGetMenuAction(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group number (1 to 10)
(i) unsigned short wMBNum	:	menu button number (1 to 10)

Return value

Button operation
- 0: no operation
- 1: momentary
- 2: alternate
- 3: button group 1
- 4: button group 2
- 5: button group 3
- 6: button group 4
- 7: button group 5

Details

Gets the button operations of the extension menu.
When a setting outside the range is made, "0: no operation" is returned.

GCSNCExMenuSetMenuString

Outline

Set character string of extension menu

Syntax

```
void GCSNCExMenuSetMenuString(
  GBaseObject    *pSelf,
  unsigned short wMGNum,
  unsigned short wMBNum,
  GTCHAR         *pMenuStr
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) GTCHAR *pMenuStr	: character string

Return value

None

Details

Sets the character string of the extension menu.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuString

Outline

Get character string of extension menu

Syntax

```
GTCHAR* GCSNCExMenuGetMenuString(
  GBaseObject    *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)
(i) unsigned short wMBNum	:	menu button No. (1 to 10)

Return value

Character string

Details

Gets the character string of the extension menu.
When a setting outside the range is made, the null character is returned.

GCSNCExMenuSetMenuStringID

Outline

Set character string resource ID of extension menu

Syntax

```
void GCSNCExMenuSetMenuStringID(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wMenuStrID
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned short wMenuStrID	: character string resource ID

Return value

None

Details

Sets the character string resource ID of extension menu.
(Sets ten designated menu group buttons)
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuStringID

Outline

Get character string resource ID of extension menu

Syntax

```
unsigned short GCSNCExMenuGetMenuStringID(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)

Return value

Character string resource ID

Details

Gets the character string resource ID of extension menu.
When a setting outside the range is made, 0 is returned.

GCSNCExMenuSetMenuPageAction

Outline

Set screen switch operation

Syntax

```
void GCSNCExMenuSetMenuPageAction(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wPageAction
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned short wPageAction	: screen switch operation

Return value

None

Details

Sets the screen switch operation.
One of the following values is set for wPageAction:

0: none
1: open panel
2: open window
3: close window
4: ON: open window, OFF: close window
5: ON: close window, OFF: open window

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuPageAction

Outline

Get screen switch operation

Syntax

```
unsigned short GCSNCExMenuGetMenuPageAction(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)
(i) unsigned short wMBNum	:	menu button No. (1 to 10)

Return value

Screen switch operation

0: none
1: open panel
2: open window
3: close window
4: ON: open window, OFF: close window
5: ON: close window, OFF: open window

Details

Gets the screen switch operation.
When a setting outside the range is made, "0: none" is returned.

GCSNCExMenuSetMenuPageNum

Outline

Set Panel/Window number

Syntax

```
void GCSNCExMenuSetMenuPageNum(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nPageNum
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nPageNum	: Panel/Window number

Return value

None

Details

Sets the Panel/Window number.
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuPageNum

Outline

Get Panel/Window number

Syntax

```
short GCSNCExMenuGetMenuPageNum(
  GBaseObject      *pSelf,
  unsigned short   wMGNum,
  unsigned short   wMBNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) unsigned short wMBNum : menu button No. (1 to 10)

Return value

Panel/Window number

Details

Gets the Panel/Window number.
When a setting outside the range is made, 0 is returned.

GCSNCExMenuSetMenuStatus

Outline

Set extension menu button status

Syntax

```
void GCSNCExMenuSetMenuStatus(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wStatus
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned short wStatus	: menu button status

Return value

None

Details

Sets the extension menu button status.
One of the following values is set for wStatus:
0: OFF status
1: ON status
When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuStatus

Outline

Get extension menu button status

Syntax

```
unsigned char GCSNCExMenuGetMenuStatus(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) unsigned short wMGNum : menu group No. (1 to 10)
(i) unsigned short wMBNum : menu button No. (1 to 10)

Return value

Menu button status
0: OFF status
1: ON status

Details

Gets the extension menu button status.
When a setting outside the range is made, "0: OFF status" is returned.

GCSNCExMenuSetMenuViewStatus

Outline Set display status of extension menu

Syntax
```
void GCSNCExMenuSetMenuViewStatus(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wViewSts
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) unsigned short wViewSts	: display status

Return value None

Details Sets the display status of the extension menu.
One of the following values is set for wViewSts:
- 0: hide
- 1: partial display
- 2: whole display
- 3: invalid display

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuViewStatus

Outline

Get display status of extension menu

Syntax

```
unsigned short GCSNCExMenuGetMenuViewStatus(
  GBaseObject      *pSelf,
  unsigned short   wMGNum,
  unsigned short   wMBNum
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)

Return value

Display status
- 0: hide
- 1: partial display
- 2: whole display
- 3: invalid display

Details

Gets the display status of the extension menu.
When a setting outside the range is made, "0: hide" is returned.

GCSNCExMenuSetMenuStringPosition

Outline

Set character string display position of extension menu

Syntax

```
void GCSNCExMenuSetMenuStringPosition(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned char   ucHPosition,
  unsigned char   ucVPosition
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned char ucHPosition	: horizontal position
(i) unsigned char ucVPosition	: vertical position

Return value

None

Details

Sets the character string display position of the extension menu.
One of the following values is set for ucHPosition:
- 0: align left
- 1: center
- 2: align right

One of the following values is set for ucVPosition:
- 0: align top
- 1: center
- 2: align bottom

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetMenuStringPosition

Outline

Get character string display position of extension menu

Syntax

```
void GCSNCExMenuGetMenuStringPosition(
  GBaseObject      *pSelf,
  unsigned short   wMGNum,
  unsigned char    *pucHPosition,
  unsigned char    *pucVPosition
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(o) unsigned char *pucHPosition	: horizontal position
(o) unsigned char *pucVPosition	: vertical position

Return value

None

Details

Gets the character string display position of the extension menu.
Gets one of the following values for pucHPosition:
Horizontal position

- 0 : align left
- 1 : center
- 2 : align right

Gets one of the following values for pucVPosition:
Vertical position

- 0 : align top
- 1 : center
- 2 : align bottom

When a setting outside the range is made, "1: center" is returned for horizontal position and vertical position.

GCSNCExMenuSetUpperMenuGroupNum

Outline Set upper row menu group number

Syntax
```
void GCSNCExMenuSetUpperMenuGroupNum(
  GBaseObject      *pSelf,
  unsigned short   wMGNum,
  unsigned short   wSetMGNum
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)
(i) unsigned short wSetMGNum	:	upper row menu group No. (1 to 10)

Return value None

Details Sets the menu group number of the upper row in the two-row display.
One of the following values is set for wSetMGNum:

0	: hide upper row menu
1 to 10	: displays the specified group number menu to the upper row

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetUpperMenuGroupNum

Outline

Get upper row menu group number

Syntax

```
unsigned short GCSNCExMenuGetUpperMenuGroupNum(
  GBaseObject      *pSelf,
  unsigned short   wMGNum
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) unsigned short wMGNum	:	menu group No. (1 to 10)

Return value

Upper row menu group number

0	: hide upper row menu
1 to 10	: menu group number set to the upper row menu

Details

Gets the menu group number of the upper row in the two-row display.
When a setting outside the range is made, 0 is returned.

GCSNCExMenuSetNextMenuGroupNum

Outline

Set destination menu group number

Syntax

```
void GCSNCExMenuSetNextMenuGroupNum(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  unsigned short  wSetMGNum
)
```

Argument

(i) GBaseObject *pSelf : object for operation

(i) unsigned short wMGNum : menu group No. (1 to 10)
return button, switch button, right-flick, left-flick : 0

(i) unsigned short wMBNum : menu button No. (1 to 10)
return button : 11
switch button, right-flick : 12
left-flick : 13

(i) unsigned short wSetMGNum : destination menu group number (0 to 10)

Return value

None

Details

Sets the destination menu group number.
One of the following values is set for usSetMGNum:

0 : no movement
1 to 10 : displays the designated group number menu

When a setting outside the range is made, the setting value is not changed.

GCSNCExMenuGetNextMenuGroupNum

Outline

Get destination menu group number

Syntax

```
unsigned short GCSNCExMenuGetNextMenuGroupNum(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum
)
```

Argument

(i) GBaseObject *pSelf	: object for operation	
(i) unsigned short wMGNum	: menu group No. (1 to 10)	
	return button, switch button, right-flick, left-flick	: 0
(i) unsigned short wMBNum	: menu button No. (1 to 10)	
	return button	: 11
	switch button, right-flick	: 12
	left-flick	: 13

Return value

Destination menu group number

0	: no movement
1 to 10	: displays the designated group number menu

Details

Gets the destination menu group number.
When a setting outside the range is made, 0 is returned.

GCSNCExMenuChangeMenuGroupNum

Outline Change menu group number to be displayed

Syntax
```
void GCSNCExMenuChangeMenuGroupNum(
  GBaseObject      *pSelf,
  unsigned short   wMGNum
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)

Return value None

Details Changes the menu group number to be displayed.
When a setting outside the range is made, the setting value is not changed.
* The upper row menu displays the group set in the "upper row menu group number" property of the designated menu group number.

GCSNCExMenuSetOpeMessage

Outline Set operation message of extension menu (specify character string)

Syntax
```
long GCSNCExMenuSetOpeMessage(
  GBaseObject      *pSelf,
  char*            _pOpeMessageString
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) char* _pOpeMessageString : operation message

Return value None

Details Sets the operation message of the extension menu to be displayed.
Up to 40 characters can be used for the operation message.

GCSNCExMenuSetOpeMessageID

Outline Set operation message of extension menu (specify resource ID)

Syntax
```
void GCSNCExMenuSetOpeMessageID(
  GBaseObject      *pSelf,
  unsigned short   wOpeMessageString
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wOpeMessageString : character string resource ID of operation message

Return value None

Details Sets the operation message using character string resource ID.

GCSNCExMenuSetMenuDevice

Outline

Set PLC device

Syntax

```
short GCSNCExMenuSetMenuDevice(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex,
  char*           szDevice
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device
(i) char* szDevice	: PLC device character string

Return value

Setting result

0xA001	: menu group No. is not correct
0xA002	: menu button No. is not correct
0xA003	: PLC device index is not correct
0xA004	: PLC device is not correct
0x0	: normal

Details

Sets the PLC device address to allocate to the control.

GCSNCExMenuGetMenuDevice

Outline

Get PLC device

Syntax

```
char* GCSNCExMenuGetMenuDevice(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device

Return value

PLC device character string

Details

Gets the PLC device address allocated to the control.
* Gets "X0008" when set to "X0008," and gets "X8" when set to "X8".

GCSNCExMenuSetMenuDeviceEnable

Outline

Set PLC device enabled

Syntax

```
short GCSNCExMenuSetMenuDeviceEnable(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex,
  unsigned short  usDeviceEnable
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device
(i) unsigned short usDeviceEnable	: PLC device enabled

Return value

Setting result

0xA001	: menu group No. is not correct
0xA002	: menu button No. is not correct
0xA003	: PLC device index is not correct
0xA005	: PLC device enabled is not correct
0x0	: normal

Details

Sets the PLC device enabled status to allocate to the control.
One of the following values is set for usDeviceEnable:

0	: disabled
1	: enabled
2	: same setting as Write PLC device

GCSNCExMenuGetMenuDeviceEnable

Outline

Get PLC device enabled

Syntax

```
unsigned short GCSNCExMenuGetMenuDeviceEnable(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device

Return value

Setting result

0 : disabled
1 : enabled
2 : Same setting as Write PLC device

Details

Gets the PLC device enabled status allocated to the control.

GCSNCExMenuSetMenuDeviceBitPosition

Outline

Set bit position of PLC device

Syntax

```
short GCSNCExMenuSetMenuDeviceBitPosition(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex,
  unsigned char   ucDeviceBitPos
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device
(i) unsigned char ucDeviceBitPos	: Bit position of PLC device (0 to 15)

Return value

Setting result

0xA001	: menu group No. is not correct
0xA002	: menu button No. is not correct
0xA003	: PLC device index is not correct
0xA006	: bit position of PLC device is not correct
0x0	: normal

Details

Sets the bit position of the PLC device to allocate to the control.

GCSNCExMenuGetMenuDeviceBitPosition

Outline

Get bit position of PLC device

Syntax

```
unsigned char GCSNCExMenuGetMenuDeviceBitPosition(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device

Return value

Bit position of PLC device (0 to 15)

Details

Gets the bit position of the PLC device allocated to the control.

GCSNCExMenuSetMenuDeviceVal

Outline

Set PLC device value

Syntax

```
short GCSNCExMenuSetMenuDeviceVal(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex,
  unsigned char   ucDeviceVal
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device
(i) unsigned char ucDeviceVal	: PLC device value

Return value

Setting result

0xA001	: menu group No. is not correct
0xA002	: menu button No. is not correct
0xA003	: PLC device index is not correct
0xA007	: writing failure
0x0	: normal

Details

Sets the PLC device value to allocate to the control.

* Monitors the write PLC device and the read PLC device using Timer event. Changing the device by ladder side changes the menu button status according to the device status. When writing a PLC device by both custom screen side and ladder side, the last writing is valid.

GCSNCExMenuGetMenuDeviceVal

Outline

Get PLC device value

Syntax

```
unsigned char GCSNCExMenuGetMenuDeviceVal(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device

Return value

PLC device value

Details

Gets the PLC device value allocated to the control.

GCSNCExMenuSetButtonGroupAllOffState

Outline

Set all OFF state of button group

Syntax

```
void GCSNCExMenuSetButtonGroupAllOffState(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex,
  unsigned char   ucStatus
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device
(i) unsigned char ucStatus	: all OFF state

Return value

None

Details

Sets the all OFF state of the button group.
One of the following values is set for ucStatus:
0: permitted
1: not permitted

GCSNCExMenuGetButtonGroupAllOffState

Outline

Get all OFF state of button group

Syntax

```
unsigned char GCSNCExMenuGetButtonGroupAllOffState(
  GBaseObject     *pSelf,
  unsigned short  wMGNum,
  unsigned short  wMBNum,
  short           nIndex
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) unsigned short wMGNum	: menu group No. (1 to 10)
(i) unsigned short wMBNum	: menu button No. (1 to 10)
(i) short nIndex	: PLC device index No. 0: Write PLC device 1: Read PLC device

Return value

All OFF state
0: permitted
1: not permitted

Details

Sets the all OFF state of the button group.

GCSNCExMenuBeginAnimState

Outline Start setting of extension menu animation operation

Syntax
```
void GCSNCExMenuBeginAnimState(
  GBaseObject     *pSelf,
  short           nMenuAnmType
)
```

Argument

(i) GBaseObject *pSelf	:	object for operation
(i) short nMenuAnmType	:	menu animation direction

Return value None

Details Starts the extension menu status change.
One of the following values is set for nMenuAnmType:

0: no animation
1: upward
2: downward
3: leftward
4: rightward

GCSNCExMenuEndAnimState

Outline	End setting of extension menu animation operation (animation start)
Syntax	void GCSNCExMenuEndAnimState(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	None
Details	Ends the extension menu status change and starts the extension menu animation.

GCSNCExMenuStopAnim

Outline	Stop extension menu animation
Syntax	void GCSNCExMenuStopAnim(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	None
Details	Stops the extension menu animation, and displays the extension menu after status change.

5.2.25 FileInOut

GCSIOCheckUpDateFile

Outline	Check existence of file
Syntax	long GCSIOCheckUpDateFile(GBaseObject *pSelf, const GTCHAR *pFileInfo)
Argument	(i) GBaseObject *pSelf : object for operation (i) const GTCHAR *pFileInfo : file path to be checked
Return value	FALSE : file does not exist TRUE : file exists
Details	Checks if the specified file exists.

GCSIOFileTransfer

Outline Start file transfer

Syntax

```
long GCSIOFileTransfer(
  GBaseObject      *pSelf,
  const GTCHAR     *pSrcFileInfo,
  const GTCHAR     *pDstFileInfo
)
```

Argument

(i) GBaseObject *pSelf : object for operation
(i) const GTCHAR *pSrcFileInfo : source file path
(i) const GTCHAR *pDstFileInfo : destination file path

Return value

IO_NOERR (0x00)	: normal end
IO_SRCOPEN_ERR (0x01)	: unable to open the transfer source file
IO_DSTOPEN_ERR (0x02)	: unable to open the transfer destination file
IO_SRCREAD_ERR (0x05)	: unable to read the transfer source file
IO_DSTWRITE_ERR (0x08)	: unable to write to the transfer destination file
IO_NOFILE_ERR (0x16)	: specified file does not exist
IO_MEMOVER_ERR (0x18)	: memory capacity over
IO_AUTORUNNING (0x1F)	: specified file is in automatic operation
IO_CHECKRUNNING (0x20)	: specified file is in program check
IO_RESEARCHING (0x21)	: specified file is in program restart
IO_SET_ERR (0x22)	: an illegal character is included in argument
IO_SYSPARA_NOPASSWD_ERR (0x23)	: unable to transfer without password input
IO_SAMEFILE_SET_ERR (0x25)	: the same file is specified for the transfer source and destination
IO_FILEOVER_ERR (0x27)	: excessive number of programs are registered
IO_OVER_FILE_LEN_MAX_SET_ERR (0x3C)	: file name length is longer than the maximum
IO_SRCFILE_IS_DIR_ERR (0x3D)	: directory is specified for transfer source file name

IO_FLD_UNINSTAL_ERR (0x3F)	: FLD drive is not connected
IO_FILEMAX_ERR (0x42)	: the number of files registered is excessive
IO_PROTECT_LOCKB (0x43)	: edit lock B is active
IO_PROTECT_LOCKC (0x44)	: edit lock C is active
IO_DS_UNINSTAL_ERR (0x46)	: data server is not connected (M700)
IO_PLCRUNNING (0x47)	: in PLC RUN
IO_DATA_PROTECT (0x48)	: data protect is active
IO_PCMCIA_UNINSTAL_ERR (0x4A)	: memory card is not connected
IO_DEVPATH_ERR (0x54)	: write protect is active on the specified device
IO_SET_PATH_ERR (0x56)	: specified path is incorrect
IO_RUNTRANSFER (0x57)	: in transferring
IO_UNSPECIFIED_FILE (0x58)	: the file to transfer is not specified
IO_EOR_ERR (0x5F)	: no EOR (%) in file

Details	Starts file transfer between NC memory/NC memory 2 and an external device.

GCSIODeleteFile

Outline

Delete file/directory

Syntax

```
long GCSIODeleteFile(
  GBaseObject      *pSelf,
  const GTCHAR     *pDeleteFileInfo
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const GTCHAR *pDeleteFileInfo	: file path to be deleted

Return value

IO_NOERR (0x00)	: normal end
IO_DELETE_ERR (0x0F)	: unable to delete the specified file (directory)
IO_NOFILE_ERR (0x16)	: specified file does not exist
IO_AUTORUNNING (0x1F)	: specified file is in automatic operation
IO_CHECKRUNNING (0x20)	: specified file is in program check
IO_RESEARCHING (0x21)	: specified file is in program restart
IO_SET_ERR (0x22)	: setting error (the specified argument is illegal)
IO_DIRHASFILES (0x39)	: a file exists in the directory
IO_FLD_UNINSTAL_ERR (0x3F)	: FLD drive is not connected
IO_PROTECT_LOCKB (0x43)	: edit lock B is active
IO_PROTECT_LOCKC (0x44)	: edit lock C is active
IO_DS_UNINSTAL_ERR (0x46)	: data server is not connected (M700)
IO_DATA_PROTECT (0x48)	: data protect is active
IO_PCMCIA_UNINSTAL_ERR (0x4A)	: memory card is not connected
IO_DEVPATH_ERR (0x54)	: write protect is active on the specified device
IO_SET_PATH_ERR (0x56)	: specified path is incorrect
IO_EOR_ERR (0x5F)	: no EOR (%) in file

Details

Deletes the specified file (directory).

GCSIOCreateDirectory

Outline

Create directory

Syntax

```
long GCSIOCreateDirectory(
  GBaseObject      *pSelf,
  const GTCHAR    * pDirInfo
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const GTCHAR * pDirInfo	: information of directory to be created

Return value

IO_NOERR (0x00)	: normal end
IO_MKDIR_ERR (0x12)	: unable to create directory
IO_MKDIRNOTSUP_ERR (0x15)	: unable to create directory on this device
IO_SET_ERR (0x22)	: setting error (the specified argument is illegal)
IO_FILEOVER_ERR (0x27)	: excessive number of programs are registered
IO_FLD_UNINSTAL_ERR (0x3F)	: FLD drive is not connected
IO_FILEMAX_ERR (0x42)	: the number of files registered is excessive
IO_DS_UNINSTAL_ERR (0x46)	: data server is not connected (M700)
IO_PCMCIA_UNINSTAL_ERR (0x4A)	: memory card is not connected
IO_DEVPATH_ERR (0x54)	: write protect is active on the specified device
IO_DIREXIST_ERR (0x55)	: specified directory already exists
IO_SET_PATH_ERR (0x56)	: specified path is incorrect
IO_EOR_ERR (0x5F)	: no EOR (%) in file

Details

Creates the specified directory.

GCSIORenameFile

Outline Rename file/directory

Syntax

```
long GCSIORenameFile(
  GBaseObject      *pSelf,
  const GTCHAR     *pOldFileInfo,
  const GTCHAR     *pNewFileInfo
)
```

Argument

(i) GBaseObject *pSelf	: object for operation
(i) const GTCHAR *pOldFileInfo	: file path before change
(i) const GTCHAR *pNewFileInfo	: file path after change

Return value

IO_NOERR (0x00)	: normal end
IO_RENAME_ERR (0x10)	: unable to rename selected file
IO_DEVICE_ERR (0x1D)	: different device has been specified
IO_AUTORUNNING (0x1F)	: specified file is in automatic operation
IO_CHECKRUNNING (0x20)	: specified file is in program check
IO_RESEARCHING (0x21)	: specified file is in program restart
IO_SET_ERR (0x22)	: setting error (the specified argument is illegal)
IO_SAMEFILE_SET_ERR (0x25)	: the same file is specified for the names before and after rename
IO_OVER_FILE_LEN_MAX_SET_ERR (0x3C)	: file name length is longer than the maximum
IO_FLD_UNINSTAL_ERR (0x3F)	: FLD drive is not connected
IO_PROTECT_LOCKB (0x43)	: edit lock B is active
IO_PROTECT_LOCKC (0x44)	: edit lock C is active
IO_DS_UNINSTAL_ERR (0x46)	: data server is not connected (M700)
IO_DATA_PROTECT (0x48)	: data protect is active
IO_PCMCIA_UNINSTAL_ERR (0x4A)	: memory card is not connected
IO_DEVPATH_ERR (0x54)	: write protect is active on the specified device
IO_SET_PATH_ERR (0x56)	: specified path is incorrect
IO_EOR_ERR (0x5F)	: no EOR (%) in file

Details Changes the file (directory) name to the specified one.

GCSInterrupt

Outline	Cancel file transfer
Syntax	long GCSInterrupt(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	IO_NOERR (0x00) : normal end
Details	Cancels the file transfer process that is being executed.

GCSIOInitializeMessage

Outline	Initialize file input/output part
Syntax	void GCSIOInitializeMessage(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	None
Details	Initializes the drawing of message and bar graph.

GCSGetLastError

Outline Get error information

Syntax
```
long GCSGetLastError(
  GBaseObject      *pSelf
)
```

Argument

(i) GBaseObject *pSelf		: object for operation

Return value

IO_NOERR (0x00)	: normal end
IO_SRCOPEN_ERR (0x01)	: unable to open the transfer source file
IO_DSTOPEN_ERR (0x02)	: unable to open the transfer destination file
IO_SRCREAD_ERR (0x05)	: unable to read the transfer source file
IO_DSTWRITE_ERR (0x08)	: unable to write to the transfer destination file
IO_DELETE_ERR (0x0F)	: unable to delete the specified file (directory)
IO_RENAME_ERR (0x10)	: unable to rename selected file
IO_MKDIR_ERR (0x12)	: unable to create directory
IO_MKDIRNOTSUP_ERR (0x15)	: unable to create directory on this device
IO_NOFILE_ERR (0x16)	: specified file does not exist
IO_MEMOVER_ERR (0x18)	: memory capacity over
IO_DEVICE_ERR (0x1D)	: different device has been specified
IO_AUTORUNNING (0x1F)	: specified file is in automatic operation
IO_CHECKRUNNING (0x20)	: specified file is in program check
IO_RESEARCHING (0x21)	: specified file is in program restart
IO_SET_ERR (0x22)	: an illegal character is included in argument
IO_SYSPARA_NOPASSWD_ERR (0x23)	: unable to transfer without password input
IO_SAMEFILE_SET_ERR (0x25)	: the same file is specified for the transfer source and destination
IO_FILEOVER_ERR (0x27)	: excessive number of programs are registered
IO_DIRHASFILES (0x39)	: a file exists in the directory
IO_OVER_FILE_LEN_MAX_SET_ERR (0x3C)	: file name length is longer than the maximum

IO_SRCFILE_IS_DIR_ERR (0x3D)	: directory is specified for transfer source file name
IO_FLD_UNINSTAL_ERR (0x3F)	: FLD drive is not connected
IO_FILEMAX_ERR (0x42)	: the number of files registered is excessive
IO_PROTECT_LOCKB (0x43)	: edit lock B is active
IO_PROTECT_LOCKC (0x44)	: edit lock C is active
IO_DS_UNINSTAL_ERR (0x46)	: data server is not connected (M700)
IO_PLCRUNNING (0x47)	: in PLC RUN
IO_DATA_PROTECT (0x48)	: data protect is active
IO_PCMCIA_UNINSTAL_ERR (0x4A)	: memory card is not connected
IO_SERIALNUMBER_ERR (0x4F)	: serial number is incorrect
IO_DEVPATH_ERR (0x54)	: write protect is active on the specified device
IO_DIREXIST_ERR (0x55)	: specified directory already exists
IO_SET_PATH_ERR (0x56)	: specified path is incorrect
IO_RUNTRANSFER (0X57)	: in transferring
IO_UNSPECIFIED_FILE (0X58)	: the file to transfer is not specified
IO_EOR_ERR (0x5F)	: no EOR (%) in file

Details: Gets the information of currently occurring errors, including the error that occurred during file transfer.

GCSIOGetTransferStatus

Outline: Get transfer status

Syntax:
```
long GCSIOGetTransferStatus(
  GBaseObject  *pSelf
)
```

Argument: (i) GBaseObject *pSelf : object for operation

Return value:
(0x00) : no status
(0x01) : being transferred
(0x02) : transfer completed
(0x03) : error

Details: Gets the file transfer status of the input/output control part.

5.2.26 AlarmMessage

GCSSetDispMessageType

Outline	Set character string display format
Syntax	long GCSSetDispMessageType(GBaseObject **pSelf,* short *_nDispType*)
Argument	(i) GBaseObject *pSelf : object for operation (i) short _nDispType : display type
Return value	None
Details	Sets the property "Character string display format" to change the display format of the character string. One of the following values is set for _lDispTyp: 0 : When the message is split, the residual message is not displayed at the next drawing update. 1 : When the message is split, the residual message is displayed at the next drawing update. 2 : A character string is displayed according to the setting of the parameter (#11021 PLC mesg disp type).

5.2.27 MonitorStatus

GCSSetDispSysNumber

Outline	Set part system number to display
Syntax	short GCSSetDispSysNumber(GBaseObject *pSelf, long _lSysNumber)
Argument	(i) GBaseObject *pSelf : object for operation (i) short _lSysNumber : part system number
Return value	None
Details	Sets the property "Part system designation" to change the system to display.

GCSGetDispSysNumber

Outline	Get part system number to display
Syntax	short GCSGetDispSysNumber(GBaseObject *pSelf)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Part system number
Details	Gets the value of the property "Part system designation".

5.2.28 NCPageChangeButton

GCSNCPageChgBtnSetDestinationPageID

Outline	Set destination screen number
Syntax	long GCSNCPageChgBtnSetDestinationPageID(GBaseObject **pSelf,* unsigned short *wDstPageID*)
Argument	(i) GBaseObject *pSelf : object for operation (i) unsigned short wDstPageID : destination screen number
Return value	FALSE : abnormal TRUE : normal
Details	Sets the destination screen number.

GCSNCPageChgBtnGetDestinationPageID

Outline	Get destination screen number
Syntax	unsigned short GCSNCPageChgBtnGetDestinationPageID(GBaseObject **pSelf*)
Argument	(i) GBaseObject *pSelf : object for operation
Return value	Destination screen number
Details	Gets the destination screen number.

GCSNCPageChgBtnSetFocusMoveToWindow

Outline Set focus move to window

Syntax
```
long GCSNCPageChgBtnSetFocusMoveToWindow(
  GBaseObject     *pSelf,
  unsigned short  wFocusMove
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wFocusMove : status of focus move to the target window

Return value
FALSE : abnormal
TRUE : normal

Details
Sets the focus move to the target window.
One of the following values is set for wFocusMove:

GNCPAGE_FOCUSMOVE(0)	perform focus move
GNCPAGE_NOTFOCUSMOVE(1)	do not perform focus move

GCSNCPageChgBtnGetFocusMoveToWindow

Outline Get focus move to window

Syntax
```
unsigned short GCSNCPageChgBtnGetFocusMoveToWindow(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value
Status of focus move to the target window

Details
Gets the focus move to the target window.

GCSNCPageChgBtnSetCloseWindow

Outline Set close status of source window

Syntax
```
long GCSNCPageChgBtnSetCloseWindow(
  GBaseObject     *pSelf,
  unsigned short  wCloseWindow
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wCloseWindow : close status of the source window

Return value
FALSE : abnormal
TRUE : normal

Details Sets the close status of the source window.
One of the following values is set for wCloseWindow:

GNCPAGE_NOTCLOSEWINDOW(0)	do not close the source window
GNCPAGE_CLOSEWINDOW(1)	close the source window

GCSNCPageChgBtnGetCloseWindow

Outline Get close status of source window

Syntax
```
unsinged short GCSNCPageChgBtnGetCloseWindow(
  GBaseObject   *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Close status of the source window

Details Gets the close status of the source window.

GCSNCPageChgBtnSetInputKeyID

Outline Set input key ID

Syntax
```
long GCSNCPageChgBtnSetInputKeyID(
  GBaseObject     *pSelf,
  unsigned short  wInputKeyID
)
```

Argument
(i) GBaseObject *pSelf : object for operation
(i) unsigned short wInputKeyID : input key ID

Return value
FALSE : abnormal
TRUE : normal

Details Sets the input key ID.

GCSNCPageChgBtnGetInputKeyID

Outline Get input key ID

Syntax
```
unsigned short GCSNCPageChgBtnGetInputKeyID(
  GBaseObject  *pSelf
)
```

Argument
(i) GBaseObject *pSelf : object for operation

Return value Input key ID

Details Gets the input key ID.

5.2.29 Graph

GCSNCGraphSetChartType

Outline	Set graph type
Syntax	void GCSNCGraphSetChartType(GBaseObject *pNCGraph, unsigned char byChartType)
Argument	(i) GBaseObject *pNCGraph : object for operation (i) unsigned char byChartType : graph type
Return value	None
Details	Sets the graph type. One of the following values is set for byChartType: GNCGRAPH_TYPE_CIRCLE(0) pie graph GNCGRAPH_TYPE_BAND(1) band graph

GCSNCGraphGetChartType

Outline	Get graph type
Syntax	unsigned char GCSNCGraphGetChartType(GBaseObject *pNCGraph)
Argument	(i) GBaseObject *pNCGraph : object for operation
Return value	Graph type
Details	Gets the graph type.

GCSNCGraphSetDispDirection

Outline Set display direction

Syntax
```
void GCSNCGraphSetDispDirection(
  GBaseObject    *pNCGraph,
  unsigned char  byDispDirection
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned char byDispDirection : display direction setting

Return value None

Details Sets the display direction.
One of the following values is set for byDispDirection:
GNCGRAPH_DISP_VERTICAL(0) vertical direction
GNCGRAPH_DISP_HORIZONTAL(1) horizontal direction

GCSNCGraphGetDispDirection

Outline Get display direction

Syntax
```
unsigned char GCSNCGraphGetDispDirection(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Display direction

Details Gets the display direction (vertical/horizontal) of band graphs.

GCSNCGraphSetItemNum

Outline Set number of items

Syntax
```
void GCSNCGraphSetItemNum(
  GBaseObject    *pNCGraph,
  unsigned short  wItemNum
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned short wItemNum : number of graph items

Return value None

Details Sets the number of items to be displayed in the graph.
Set wItemNum within the following range:
1 to 32

GCSNCGraphGetItemNum

Outline Get number of items

Syntax
```
unsigned short GCSNCGraphGetItemNum(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Number of graph items

Details Gets the number of items to be displayed in the graph.

GCSNCGraphSetSection

Outline Set section number

Syntax
```
void GCSNCGraphSetSection(
  GBaseObject   *pNCGraph,
  long          lSection
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) long lSection : section number

Return value None

Details Sets the section number.

GCSNCGraphGetSection

Outline Get section number

Syntax
```
long GCSNCGraphGetSection(
  GBaseObject   *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Section number

Details Gets the section number.

GCSNCGraphSetSubSection

Outline Set sub-section number

Syntax
```
void GCSNCGraphSetSubSection(
  GBaseObject    *pNCGraph,
  long           lSubSection
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) long lSubSection : sub-section number

Return value None

Details Sets the sub-section number.

GCSNCGraphGetSubSection

Outline Get sub-section number

Syntax
```
long GCSNCGraphGetSubSection(
  GBaseObject   *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Sub-section number

Details Gets the sub-section number.

GCSNCGraphSetGround

Outline Set ground

Syntax
```
void GCSNCGraphSetGround(
  GBaseObject   *pNCGraph,
  long          lGround
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) long lGround : ground

Return value None

Details Sets the ground.
Set lGround within the following range:
1 to 3

GCSNCGraphGetGround

Outline Get ground

Syntax
```
long GCSNCGraphGetGround(
  GBaseObject   *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Ground

Details Gets the ground.

GCSNCGraphSetAxisInfo

Outline Set axis number

Syntax
```
void GCSNCGraphSetAxisInfo(
  GBaseObject    *pNCGraph,
  unsigned long  dwAxis
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned long dwAxis : axis number

Return value None

Details Sets the axis number.

GCSNCGraphGetAxisInfo

Outline Get axis number

Syntax
```
long GCSNCGraphGetAxisInfo(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Axis number

Details Gets the axis number.

GCSNCGraphSetDataType

Outline Set data type

Syntax
```
void GCSNCGraphSetDataType(
  GBaseObject   *pNCGraph,
  long          lType
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) long lType : data type

Return value None

Details Sets the data type
One of the following values is set for lType:

GNCGRAPH_DTYPE_CHAR(1)	: char type
GNCGRAPH_DTYPE_SHORT(2)	: short value
GNCGRAPH_DTYPE_LONG(3)	: long value
GNCGRAPH_DTYPE_DOUBLE(5)	: double value
GNCGRAPH_DTYPE_UCHAR(33)	: unsigned char value
GNCGRAPH_DTYPE_USHORT(34)	: unsigned short value
GNCGRAPH_DTYPE_ULONG(35)	: unsigned long value

GCSNCGraphGetDataType

Outline Get data type

Syntax
```
long GCSNCGraphGetDataType(
  GBaseObject   *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Data type

Details Gets the data type.

GCSNCGraphSetReadMethod

Outline Set reading method

Syntax
```
void GCSNCGraphSetReadMethod(
  GBaseObject    *pNCGraph,
  unsigned char  byMethod
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned char byMethod : reading method

Return value None

Details Sets the reading method.
One of the following values is set for byMethod:

GNCGRAPH_RD_METHOD_SUBSECTION (0)	reading method: sub-section number
GNCGRAPH_RD_METHOD_SYSTEM (1)	reading method: part system number
GNCGRAPH_RD_METHOD_AXIS (2)	reading method: axis number

GCSNCGraphGetReadMethod

Outline Get reading method

Syntax
```
unsigned char GCSNCGraphGetReadMethod(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Reading method

Details Gets the reading method.

GCSNCGraphSetReadOffset

Outline Set read offset

Syntax
```
void GCSNCGraphSetReadOffset(
  GBaseObject   *pNCGraph,
  long          lOffset
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) long lOffset : read offset

Return value None

Details Sets the read offset.
Set lOffset within a range of 1 to 1000.

GCSNCGraphGetReadOffset

Outline Get read offset

Syntax
```
long GCSNCGraphGetReadOffset(
  GBaseObject   *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Read offset

Details Gets the read offset.

GCSNCGraphSetGraphFillPattern

Outline Set pattern

Syntax
```
void GCSNCGraphSetGraphFillPattern(
  GBaseObject   *pNCGraph,
  short         nIndex,
  short         nPattern
)
```

Argument

(i) GBaseObject *pNCGraph	:	object for operation
(i) short nIndex	:	specified graph pattern index
(i) short nPattern	:	graph pattern 1 to 32

Return value None

Details Sets the graph pattern 1 to 32.
Specify nIndex within the following range:
1 to 32

GCSNCGraphGetGraphFillPattern

Outline Get pattern

Syntax
```
short GCSNCGraphGetGraphFillPattern(
  GBaseObject   *pNCGraph,
  short         nIndex,
  short         *pnPattern
)
```

Argument

(i) GBaseObject *pNCGraph	:	object for operation
(i) short nIndex	:	specified graph pattern item number
(o) short *pnPattern	:	pointer to graph pattern 1 to 32

Return value

FALSE	:	error end
TRUE	:	normal end

Details Gets the graph pattern 1 to 32.
Specify nIndex within the following range:
1 to 32

GCSNCGraphSetGraphForeColor

Outline Set foreground color

Syntax
```
void GCSNCGraphSetGraphForeColor(
  GBaseObject   *pNCGraph,
  short         nIndex,
  long          lColor
)
```

Argument

(i) GBaseObject *pNCGraph	:	object for operation
(i) short nIndex	:	specified graph pattern number
(i) long lColor	:	graph foreground color 1 to 32

Return value None

Details Sets the graph foreground color 1 to 32.
Specify nIndex within the following range:
1 to 32

GCSNCGraphGetGraphForeColor

Outline Get foreground color

Syntax
```
short GCSNCGraphGetGraphForeColor(
  GBaseObject   *pNCGraph,
  short         nIndex,
  long          *plColor
)
```

Argument

(i) GBaseObject *pNCGraph	:	object for operation
(i) short nIndex	:	specified graph pattern number
(o) long *plColor	:	pointer to graph foreground color 1 to 32

Return value

FALSE	:	error end
TRUE	:	normal end

Details Gets the graph foreground color 1 to 32.
Specify nIndex within the following range:
1 to 32

GCSNCGraphSetGraphBackColor

Outline Set background color

Syntax
```
void GCSNCGraphSetGraphBackColor(
  GBaseObject   *pNCGraph,
  short         nIndex,
  long          lColor
)
```

Argument

(i) GBaseObject *pNCGraph	:	object for operation
(i) short nIndex	:	specified graph pattern item number
(i) long lColor	:	graph background color 1 to 32

Return value None

Details Sets the graph background color 1 to 32.
Specify nIndex within the following range:
1 to 32

GCSNCGraphGetGraphBackColor

Outline Get background color

Syntax
```
short GCSNCGraphGetGraphBackColor(
  GBaseObject   *pNCGraph,
  short         nIndex,
  long          *plColor
)
```

Argument

(i) GBaseObject *pNCGraph	:	object for operation
(i) short nIndex	:	specified graph pattern item number
(o) long *plColor	:	pointer to graph background color 1 to 32

Return value

FALSE	:	error end
TRUE	:	normal end

Details Gets the graph background color 1 to 32.
Specify nIndex within the following range:
1 to 32

GCSNCGraphSetScaleColor

Outline	Set scale color
Syntax	void GCSNCGraphSetScaleColor(GBaseObject *pNCGraph, long lColor)
Argument	(i) GBaseObject *pNCGraph : object for operation (i) long lColor : scale color
Return value	None
Details	Sets the scale color.

GCSNCGraphGetScaleColor

Outline	Get scale color
Syntax	long GCSNCGraphGetScaleColor(GBaseObject *pNCGraph)
Argument	(i) GBaseObject *pNCGraph : object for operation
Return value	Scale color
Details	Gets the scale color.

GCSNCGraphSetScaleNum

Outline Set scale number

Syntax
```
void GCSNCGraphSetScaleNum(
  GBaseObject    *pNCGraph,
  unsigned short  wScaleNum
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned short wScaleNum : scale number

Return value None

Details Sets the scale number.

GCSNCGraphGetScaleNum

Outline Get scale number

Syntax
```
unsigned short GCSNCGraphGetScaleNum(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Scale number

Details Gets the scale number.

GCSNCGraphSetScaleWidth

Outline Set scale width

Syntax
```
void GCSNCGraphSetScaleWidth(
  GBaseObject     *pNCGraph,
  unsigned short  wScaleWidth
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned short wScaleWidth : scale width

Return value None

Details Sets the scale width.

GCSNCGraphGetScaleWidth

Outline Get scale width

Syntax
```
unsigned short GCSNCGraphGetScaleWidth(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Scale width

Details Gets the scale width.

GCSNCGraphSetScaleMin

Outline	Set scale (minimum)
Syntax	void GCSNCGraphSetScaleMin(GBaseObject *pNCGraph, double *pdScaleMin)
Argument	(i) GBaseObject *pNCGraph : object for operation (i) double *pdScaleMin : pointer to scale (minimum)
Return value	None
Details	Sets the scale (mimimum) of the graph.

GCSNCGraphGetScaleMin

Outline	Get scale (mimimum)
Syntax	void GCSNCGraphGetScaleMin(GBaseObject *pNCGraph, double *pdScaleMin)
Argument	(i) GBaseObject *pNCGraph : object for operation (o) double *pdScaleMin : pointer to scale (mimimum)
Return value	None
Details	Gets the scale (mimimum) of the graph.

GCSNCGraphSetScaleMax

Outline Set scale (maximum)

Syntax
```
void GCSNCGraphSetScaleMax(
  GBaseObject   *pNCGraph,
  double        *pdScaleMax
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) double *pdScaleMax : pointer to scale (maximum)

Return value None

Details Sets the scale (maximum) of the graph.

GCSNCGraphGetScaleMax

Outline Get scale (maximum)

Syntax
```
void GCSNCGraphGetScaleMax(
  GBaseObject   *pNCGraph,
  double        *pdScaleMax
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(o) double *pdScaleMax : pointer to scale (maximum)

Return value None

Details Gets the scale (maximum) of the graph.

GCSNCGraphSetLowerLimit

Outline Set boundary value (lower limit)

Syntax
```
void GCSNCGraphSetLowerLimit(
  GBaseObject   *pNCGraph,
  double        *pdLowerLimit
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) double *pdLowerLimit : pointer to boundary value (lower limit)

Return value None

Details Sets the boundary value (lower limit).

GCSNCGraphGetLowerLimit

Outline Get boundary value (lower limit)

Syntax
```
void GCSNCGraphGetLowerLimit(
  GBaseObject   *pNCGraph,
  double        *pdLowerLimit
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(o) double *pdLowerLimit : pointer to boundary value (lower limit)

Return value None

Details Gets the boundary value (lower limit).

GCSNCGraphSetUpperLimit

Outline Set boundary value (upper limit)

Syntax
```
void GCSNCGraphSetUpperLimit(
  GBaseObject   *pNCGraph,
  double        *pdUpperLimit
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) double *pdUpperLimit : pointer to boundary value (upper limit)

Return value None

Details Sets the boundary value (upper limit).

GCSNCGraphGetUpperLimit

Outline Get boundary value (upper limit)

Syntax
```
void GCSNCGraphGetUpperLimit(
  GBaseObject   *pNCGraph,
  double        *pdUpperLimit
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(o) double *pdUpperLimit : pointer to boundary value (upper limit)

Return value None

Details Gets the boundary value (upper limit).

GCSNCGraphSetSort

Outline Set sort type

Syntax
```
void GCSNCGraphSetSort(
  GBaseObject    *pNCGraph,
  unsigned char  bySort
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) unsigned char bySort : sort type

Return value None

Details Sets the graph (item) display order.

GCSNCGraphGetSort

Outline Get sort type

Syntax
```
unsigned char GCSNCGraphGetSort(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value Sort type

Details Gets the sort type.

GCSNCGraphGetItemData

Outline

Get graph item data (reading order)

Syntax

```
short GCSNCGraphGetItemData(
  GBaseObject   *pNCGraph,
  short         nIndex,
  double        *pdGData
)
```

Argument

(i) GBaseObject *pNCGraph	: object for operation
(i) short nIndex	: specified graph pattern item number
(o) double *pdGData	: pointer to graph item data

Return value

FALSE	: error end
TRUE	: normal end

Details

Specifies the graph item, and gets the data read from NC.
(Item numbers are specified in the reading order.)

GCSNCGraphSetSystemNumber

Outline Set part system number

Syntax
```
void GCSNCGraphSetSystemNumber(
  GBaseObject  *pNCGraph,
  long         lSystem
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation
(i) long lSystem : part system number

Return value None

Details Sets the part system number.

GCSNCGraphGetSystemNumber

Outline Get part system number

Syntax
```
long GCSNCGraphGetSystemNumber(
  GBaseObject  *pNCGraph
)
```

Argument
(i) GBaseObject *pNCGraph : object for operation

Return value None

Details Gets the part system number.

5.2.30 AlarmList

GCSNCAlarmListUpdateHistory

Outline	Update alarm history
Syntax	void GCSNCAlarmListUpdateHistory(GBaseObject **pNCAlarmList*)
Argument	(i) GBaseObject *pNCAlarmList : object for operation
Return value	None
Details	Updates the alarm history.

GCSNCAlarmListSetDispType

Outline Set display type

Syntax
```
void GCSNCAlarmListSetDispType(
  GBaseObject    *pNCAlarmList,
  unsigned char  ucDispType
)
```

Argument
(i) GBaseObject *pNCAlarmList : object for operation
(i) unsigned char ucDispType : display type

Return value None

Details Sets the alarm type to be displayed in the alarm list.

GCSNCAlarmListGetDispType

Outline Get display type

Syntax
```
unsigned char GCSNCAlarmListGetDispType (
  GBaseObject  *pNCAlarmList
)
```

Argument
(i) GBaseObject *pNCAlarmList : object for operation

Return value Display type

Details Gets the alarm type to be displayed in the alarm list.

GCSNCAlarmListGetSelectLine

Outline Get selected line number

Syntax
```
short GCSNCAlarmListGetSelectLine(
  GBaseObject   *pNCAlarmList
)
```

Argument (i) GBaseObject *pNCAlarmList : object for operation

Return value Selected line

Details Gets the selected line number in the message display area.

GCSNCAlarmListGetAlarmInfo

Outline Get alarm information

Syntax
```
long GCSNCAlarmListGetAlarmInfo(
  GBaseObject   *pNCAlarmList,
  long          lLine,
  unsigned long ulItem,
  void          *pData
)
```

Argument
(i) GBaseObject *pNCAlarmList : object for operation
(i) long lLine : (I/) selected line number
(i) unsigned long ulItem : (I/) alarm information retrieval item frame
(o) void *pData : (/O) alarm information
(the storage area is allocated by the caller)

Return value
FALSE : abnormal
TRUE : normal

Details Gets the alarm information currently displayed in the alarm list or the alarm history.

5.2.31 NC Data Access Setting

GNCDataSetGNCValue

Outline Set value to CNC

Syntax
```
DWORD GNCDataSetGNCValue(
    GNCControl  *pNCControl,
    long        lSection,
    long        lSubSection,
    long        lDataType,
    GNCValue    *pNCValue
)
```

Argument

(i) GNCControl *pNCControl	: pointer to the NC information structure
(i) long lSection	: section number
(i) long lSubSection	: sub-section number
(i) long lDataType	: data type
(i) GNCValue *pNCValue	: pointer to store numerical data to be set to the NC

Return value
0: setting failed
1: setting succeeded

Details Sets the value stored in pNCValue to the NC.
Set any of the following value for lDataType:
0x1: char
0x2: short
0x3: long
0x5: real number

GNCDataGetGNCValue

Outline

Get value from CNC

Syntax

```
DWORD GNCDataGetGNCValue(
  GNCControl    *pNCControl,
  long          lSection,
  long          lSubSection,
  long          lDataType,
  GNCValue      *pNCValue
)
```

Argument

(i) GNCControl *pNCControl	: pointer to the NC information structure
(i) long lSection	: section number
(i) long lSubSection	: sub-section number
(i) long lDataType	: data type
(o) GNCValue *pNCValue	: pointer to store numerical data to be acquired from the NC

Return value

0: acquisition failed
1: acquisition succeeded
2: variable <empty>

Details

Stores the numerical data acquired from the NC in pNCValue.
Set any of the following value for lDataType:
0x1: char
0x2: short
0x3: long
0x5: real number

GCSNCDataSetString

Outline

Set character string to CNC

Syntax

```
DWORD GCSNCDataSetString(
  GNCControl      *pNCControl,
  long            lSection,
  long            lSubSection,
  long            lLength,
  const GTCHAR    *pString
)
```

Argument

(i) GNCControl *pNCControl	: pointer to the NC information structure
(i) long lSection	: section number
(i) long lSubSection	: sub-section number
(i) long lLength	: maximum number of characters for the character string data to be set to the NC
(i) const char *pString	: pointer to store character string data to be set to the NC

Return value

0: setting failed
1: setting succeeded

Details

Sets the character strings stored in pString to the NC.

GCSNCDataGetString

Outline

Get character string from CNC

Syntax

```
DWORD GCSNCDataGetString(
  GNCControl   *pNCControl,
  long         lSection,
  long         lSubSection,
  long         lLength,
  GTCHAR       *pString
)
```

Argument

(i) GNCControl *pNCControl	: pointer to the NC information structure
(i) long lSection	: section number
(i) long lSubSection	: sub-section number
(i) long lLength	: maximum number of characters for the character string data to be acquired from the NC
(o) GTCHAR *pString	: pointer to store character string data to be acquired from the NC

Return value

0: setting failed
1: setting succeeded

Details

Stores the character string data acquired from the NC in pString.

melSetData

Outline

Set data

Syntax

```
DWORD melSetData(
 HWND    hWnd,
 long    lAddress,
 long    lSectionNum,
 long    lSubSectionNum,
 long    lAxisFlag,
 LPVOID  lpSetData,
 long    lSetType
)
```

Argument

(i) HWND hWnd	: specify 0.
(i) long lAddress	: address Specify the following items. - Specify NC. - Specify the basic part system or current part system during cross control. (Required depending on the data) - Specify the part system. (Required depending on the data) - Specify the forced write mode.
(i) long lSectionNum	: section number
(i) long lSubSectionNum	: sub-section number
(i) long lAxisFlag	: specify axis. (Required depending on the data)
(i) LPVOID lpSetData	: pointer to the data area
(i) long lSetType	: data type

Return value

dwStatus	: error code
0	: normal end
ME_DATA_WRITE_ADDR	: the specified address is not correct.
ME_DATA_WRITE_SECT	: the section number is not correct.
ME_DATA_WRITE_SUBSECT	: the sub-section number is not correct.
ME_DATA_WRITE_AXIS	: the specified axis is not correct.
ME_DATA_WRITE_DATASIZE	: data size over
ME_DATA_WRITE_DATATYPE	: the specified data type is not correct.
ME_DATA_WRITE_READONLY	: read only data
ME_DATA_WRITE_WRITE	: cannot write data.

ME_DATA_WRITE_SORT	: cannot write data because the tool data is being sorted.
ME_DATA_WRITE_ULPROTECT_CHK	: data protection file for each user level
ME_DATA_WRITE_PROTECT	: data protection error
ME_DATA_WRITE_NODATA	: the writing destination data is not found.
ME_DATA_WRITE_OCCUPIED	: cannot write data because it is being used.
ME_DATA_WRITE_RUNNING	: cannot write data. (In operation or running)
ME_DATA_WRITE_RBT_NOTCONNECT	: robot controller unconnected
ME_DATA_WRITE_RBT_RUNNING	: cannot write data because the robot is in automatic operation.
ME_DATA_WRITE_RBT_TIMEOUT	: robot connection timeout
ME_DATA_WRITE_RBT_ENABLEOFF	: robot enable device disabled

Details	Sets data.

melGetData

Outline Get data

Syntax

```
DWORD melGetData(
  HWND    hWnd,
  long    lAddress,
  long    lSectionNum,
  long    lSubSectionNum,
  long    lAxisFlag,
  LPVOID  lpGetData,
  long    lGetType
)
```

Argument

(i) HWND hWnd	: specify 0.
(i) long lAddress	: address Specify the following items. - Specify NC. - Specify the basic part system or current part system during cross control. (Required depending on the data) - Specify the part system. (Required depending on the data) - Specify the ground (Fore/Back). (Required depending on the data)
(i) long lSectionNum	: section number
(i) long lSubSectionNum	: sub-section number
(i) long lAxisFlag	: specify axis. (Required depending on the data)
(i) LPVOID lpGetData	: pointer to the area to store data
(i) long lGetType	: data type

Return value

dwStatus	: error code
0	: normal end
ME_DATA_READ_ADDR	: the specified address is not correct.
ME_DATA_READ_SECT	: the section number is not correct.
ME_DATA_READ_SUBSECT	: the sub-section number is not correct.
ME_DATA_READ_DATASIZE	: data size over
ME_DATA_READ_READ	: unreadable data
ME_DATA_READ_DATATYPE	: the specified data type is not correct.
ME_DATA_READ_AXIS	: the specified axis is not correct.

ME_DATA_READ_WRITEONLY	: write only data
ME_DATA_READ_NODATA	: the data to be read is not found.
ME_DATA_READ_RBT_NOTCONNECT	: robot controller unconnected
ME_DATA_READ_RBT_TIMEOUT	: robot connection timeout
*lpGetData	: acquired data

Details	Gets data.

melGetLumpData

Outline Get continuous data in a batch

Syntax

```
DWORD melGetLumpData(
  HWND    hWnd,
  long    lAddress,
  long    lSectionNum,
  long    lSubSectTop,
  long    lGetNos,
  LPVOID  lpGetData,
  long    lGetType
)
```

Argument

(i) HWND hWnd	: specify 0.
(i) long lAddress	: address Specify the following items. - Specify NC. - Specify the basic part system or current part system during cross control. (Required depending on the data) - Specify the part system. (Required depending on the data) - Specify the ground (Fore/Back). (Required depending on the data)
(i) long lSectionNum	: section number
(i) long lSubSectTop	: sub-section number
(i) long lGetNos	: number of continuous data to be acquired
(i) LPVOID lpGetData	: pointer to the area to store data
(i) long lGetType	: data type (T_CHAR、T_SHORT、T_LONG、T_DOUBLE)

Return value

dwStatus	: error code
0	: normal end
ME_DATA_READ_ADDR	: the specified address is not correct.
ME_DATA_READ_SECT	: the section number is not correct.
ME_DATA_READ_SUBSECT	: the sub-section number is not correct.
ME_DATA_READ_DATASIZE	: data size over
ME_DATA_READ_READ	: unreadable data
ME_DATA_READ_DATATYPE	: the specified data type is not correct.
*lpGetData	: acquired data

Details Gets continuous data in a batch.

melSelectExecPrg

Outline

Search operation program

Syntax

```
DWORD melSelectExecPrg(
  HWND    hWnd,
  long    lAddress,
  LPVOID  lpSelectPrg,
  long    lDataType,
  long    lSequenceNum,
  long    lBlockNum
)
```

Argument

(i) HWND hWnd	: specify 0.
(i) long lAddress	: address Specify the following items. - Specify NC. - Specify the part system.
LPVOID lpSelectPrg	: program file name for which the operation search is performed
long lDataType	: data type of lpSelectPrg (T_STR)
long lSequenceNum	: sequence No. to be searched for
long lBlockNum	

Return value

dwStatus	: error code
0	: normal end
ME_OPE_SELECTPRG_ADDR	: the specified address is not correct.
ME_OPE_SELECTPRG_PRGFORMAT	: the format of the program file name is not correct.
ME_OPE_SELECTPRG_NOTPRG	: the specified program is not found.
ME_OPE_SELECTPRG_RUNNING	: the program is running.
ME_OPE_SELECTPRG_RESET	: in process of resetting
ME_OPE_SELECTPRG_FILESYSTEM	: some kind of error occurred in the file system.
ME_OPE_SELECTPRG_SEARCHING	: cannot perform search (currently carrying out another search).
ME_OPE_SELECTPRG_CHECKING	: cannot perform search (currently carrying out a check or simulation).
ME_OPE_SELECTPRG_MODE	: the specified operation mode is not correct.

ME_OPE_SELECTPRG_NOTSUPPORTED : a file of the unsupported device has been specified.

Details	Searches the operation program.

6. GIO

This section describes the GIO function group.

6.1 List of GIO Functions

No.	Name	Type	File name	Outline
1	GIOOpen	HGIO	GIO.c	I/O open process
2	GIOClose	void	GIO.c	I/O close process
3	GIORead	unsigned long	GIO.c	I/O read
4	GIOWrite	unsigned long	GIO.c	I/O write
5	GIOSeek	long	GIO.c	File pointer movement

6.2 Description of GIO Functions

GIOOpen

GIO.c

Outline	I/O open process
Syntax	HGIO GIOOpen(unsigned short *usDirType,* GTCHAR **pIOName,* GTCHAR **pOperation*)
Argument	(i) unsigned short usDirType : Standard directory (i) GTCHAR *pIOName : name of I/O being opened (i) GTCHAR *pOperation : opening operation
Return value	NULL : open failed Other than NULL : HGIO handle
Details	Carries out the process for opening file I/O.
File name	GIO.c

GIOClose

GIO.c

Outline	I/O close process
Syntax	void GIOClose(HGIO *hIO*)
Argument	(i) HGIO hIO : handle being closed
Return value	None
Details	Carries out the process for closing file I/O.
File name	GIO.c

GIORead

GIO.c

Outline I/O read

Syntax

```
unsigned long GIORead(
  void          *pBuffer,
  unsigned long  ulSize,
  unsigned long  ulCount,
  HGIO           hIO
)
```

Argument

(o) void *pBuffer : area for storing read results
(i) unsigned long ulSize : read data size units
(i) unsigned long ulCount : number of pieces of read data
(i) HGIO hIO : file handle

Return value Number of pieces of data read

Details Carries out the reading of data from file I/O.

File name GIO.c

GIOWrite	
	GIO.c

Outline	I/O write

Syntax

```
unsigned long GIOWrite(
  void          *pBuffer,
  unsigned long  ulSize,
  unsigned long  ulCount,
  HGIO           hIO
)
```

Argument	(i) void *pBuffer	: area for storing write results
	(i) unsigned long ulSize	: write data size units
	(i) unsigned long ulCount	: number of pieces of data written
	(i) HGIO hIO	: file handle

Return value	Number of pieces of data written
Details	Carries out the writing of data for file I/O.
File name	GIO.c

GIOSeek

GIO.c

Outline	File pointer movement
Syntax	long GIOSeek(HGIO *hIO,* long *lOffset,* long *lOrigin*)
Argument	(i) long lOffset : offset position from initial position (i) long lOrigin : initial position
Return value	Results from execution 0 : succeeded Other than 0 : failed
Details	Moves the file pointer to the specified position.
File name	GIO.c

7. GMMemory

This section describes the GMMemory function group.

7.1 List of GMMemory Functions

No.	Name	Type	File name	Outline
1	GMGMemCreate	GMGMEM	GMMemory.c	Create named shared memory area
2	GMGMemSelect	GMGMEM	GMMemory.c	Select memory area
3	GMGMemDelete	void	GMMemory.c	Release shared memory area
4	GMGMemGetShort	short	GMMemory.c	Get shared memory 16 bit integer
5	GMGMemSetShort	void	GMMemory.c	Set shared memory 16 bit integer
6	GMGMemGetChar	char	GMMemory.c	Get shared memory 8 bit integer
7	GMGMemSetChar	void	GMMemory.c	Set shared memory 8 bit integer
8	GMGMemGetLong	long	GMMemory.c	Get shared memory 32 bit integer
9	GMGMemSetLong	void	GMMemory.c	Set shared memory 32 bit integer
10	GMGMemGetDouble	double	GMMemory.c	Get shared memory 64 bit floating point value
11	GMGMemSetDouble	void	GMMemory.c	Set shared memory 64 bit floating point value
12	GMGMemGetString	GTCHAR *	GMMemory.c	Get shared memory character string
13	GMGMemSetString	void	GMMemory.c	Set shared memory character string

7.2 Description of GMMemory Functions

GMGMemCreate

GMMemory.c

Outline	Create named shared memory area
Syntax	GMGMEM GMGMemCreate(GTCHAR *pszName, unsigned long ulSize)
Argument	(i) GTCHAR *pszName : shared memory name (i) unsigned long ulSize : allocation size
Return value	Allocated region
Details	Shared memory with the name specified in pszName and the size specified in ulSize is allocated.
File name	GMMemory.c

GMGMemSelect

GMMemory.c

Outline	Select memory area
Syntax	GMGMEM GMGMemSelect(GTCHAR *pszLangStr)
Argument	(i) GTCHAR *pszLangStr : shared memory name
Return value	Shared memory area
Details	Selects the region for the specified name from the shared memory areas previously created.
File name	GMMemory.c

GMGMemDelete

GMMemory.c

Outline	Release shared memory area
Syntax	void GMGMemDelete(GMGMEM *gmem*)
Argument	(i) GMGMEM gmem : shared memory area
Return value	None
Details	Releases the specified shared memory area.
File name	GMMemory.c

GMGMemGetShort

GMMemory.c

Outline	Get shared memory 16 bit integer
Syntax	short GMGMemGetShort(GMGMEM *gmem,* unsigned long *ulOffset*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position
Return value	16 bit integer value
Details	Gets the ulOffset byte data from the beginning of the specified shared memory as a 16 bit integer value.
File name	GMMemory.c

GMGMemSetShort

GMMemory.c

Outline	Set shared memory 16 bit integer
Syntax	void GMGMemSetShort(GMGMEM *gmem,* unsigned long *ulOffset,* short *data*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position (i) short data : setting value
Return value	None
Details	Sets a 16 bit integer value in the ulOffset byte data from the beginning of the specified shared memory.
File name	GMMemory.c

GMGMemGetChar

GMMemory.c

Outline	Get shared memory 8 bit integer
Syntax	char GMGMemGetChar(GMGMEM *gmem,* unsigned long *ulOffset*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position
Return value	8 bit integer value
Details	Gets the ulOffset byte data from the beginning of the specified shared memory as an 8 bit integer value.
File name	GMMemory.c

GMGMemSetChar

GMMemory.c

Outline	Set shared memory 8 bit integer
Syntax	void GMGMemSetChar(GMGMEM *gmem,* unsigned long *ulOffset,* char *data*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position (i) char data : setting value
Return value	None
Details	Sets an 8 bit integer value in the ulOffset byte data from the beginning of the specified shared memory.
File name	GMMemory.c

GMGMemGetLong

GMMemory.c

Outline	Get shared memory 32 bit integer
Syntax	long GMGMemGetLong(GMGMEM *gmem,* unsigned long *ulOffset*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position
Return value	32 bit integer value
Details	Gets the ulOffset byte data from the beginning of the specified shared memory as a 32 bit integer value.
File name	GMMemory.c

GMGMemSetLong

GMMemory.c

Outline	Set shared memory 32 bit integer
Syntax	void GMGMemSetLong(GMGMEM *gmem,* unsigned long *ulOffset,* long *data*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position (i) long data : setting value
Return value	None
Details	Sets a 32 bit integer value in the ulOffset byte data from the beginning of the specified shared memory.
File name	GMMemory.c

GMGMemGetDouble

GMMemory.c

Outline	Get shared memory 64 bit floating point value
Syntax	double GMGMemGetDouble(GMGMEM *gmem,* unsigned long *ulOffset*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position
Return value	64 bit floating point value
Details	Gets the ulOffset byte memory from the beginning of the specified shared memory as a 64 bit floating point value.
File name	GMMemory.c

GMGMemSetDouble

GMMemory.c

Outline Set shared memory 64 bit floating point value

Syntax
```
void GMGMemSetDouble(
  GMGMEM       gmem,
  unsigned long  ulOffset,
  double         data
)
```

Argument
(i) GMGMEM gmem : shared memory area
(i) unsigned long ulOffset : offset position
(i) double data : setting value

Return value None

Details Sets a 64 bit floating point value in the ulOffset byte data from the beginning of the specified shared memory.

File name GMMemory.c

GMGMemGetString

GMMemory.c

Outline Get shared memory area

Syntax
```
GTCHAR *GMGMemGetString(
  GMGMEM       gmem,
  unsigned long  ulOffset
)
```

Argument
(i) GMGMEM gmem : shared memory area
(i) unsigned long ulOffset : offset position
(o) GTCHAR *data : character string acquisition region

Return value None

Details Gets the ulOffset byte data from the beginning of the specified shared memory as a character string.

File name GMMemory.c

GMGMemSetString

GMMemory.c

Outline	Set shared memory character string
Syntax	void GMGMemSetString(GMGMEM *gmem,* unsigned long *ulOffset,* GTCHAR **data*)
Argument	(i) GMGMEM gmem : shared memory area (i) unsigned long ulOffset : offset position (i) GTCHAR *data : setting value
Return value	None
Details	Sets a character string in the ulOffset byte data from the beginning of the specified shared memory.
File name	GMMemory.c

Appendix

Appendix 1. Data Type Definitions

The definitions for various structure data types are given in the following.

```
#define   GColor          long                /* color variable                    */
#define   HGFONT          GFontHandle*        /* 2 font handle                     */
#define   HGDRAW          GDraw*              /* drawing handle                    */

/*****************************************************************************
 * Rectangle structure
 *****************************************************************************/
typedef struct _GRect{
        short             nXmin;              /* upper left X coordinate           */
        short             nYmin;              /* upper left Y coordinate           */
        short             nXmax;              /* lower right X coordinate          */
        short             nYmax;              /* lower right Y coordinate          */
}GRect;

/*****************************************************************************
 * Point structure
 *****************************************************************************/
typedef struct _GPoint{
        short             nX;                 /* X coordinate                      */
        short             nY;                 /* Y coordinate                      */
}GPoint;

/*****************************************************************************
 * Line structure
 *****************************************************************************/
typedef struct _GLine{
        short             nX1;                /* X coordinate for starting point   */
        short             nY1;                /* Y coordinate for starting point   */
        short             nX2;                /* X coordinate for ending point     */
        short             nY2;                /* Y coordinate for ending point     */
}GLine;
```

```
/*****************************************************************************
 * Polygon structure
 *****************************************************************************/
typedef struct _GPoly{
        short           nNumPoints;         /* number of vertices              */
        GPoint          *pgptPoints;        /* pointer to vertex data          */
        GRect           grBounds;           /* outline rectangle               */
}GPoly;

/*****************************************************************************
 * Brush structure
 *****************************************************************************/
typedef struct _GBrush{
        short           nFillPattern;       /* fill pattern                    */
        GColor          gcForeColor;        /* fill foreground color           */
        GColor          gcBackColor;        /* fill background color           */
}GBrush;

/*****************************************************************************
 * Border structure
 *****************************************************************************/
typedef struct _GBorder{
        unsigned char   fBorder;            /* 3D border present or absent     */
        GColor          gcULColor;          /* upper left border color         */
        GColor          gcLRColor;          /* lower right border color        */
        GColor          gcLineColor;        /* line color                      */
        short           nSize;              /* 3D border size                  */
}GBorder;
```

```
/*****************************************************************************
 * Caption structure
 *****************************************************************************/
typedef struct _GCaption{
        GColor          gcColor;        /* caption character color          */
        unsigned char   ucHPosition;    /* horizontal display position      */
        unsigned char   ucVPosition;    /* vertical display position        */
        short           nLeftMargin;    /* left margin                      */
        short           nRightMargin;   /* right margin                     */
        short           nTopMargin;     /* top margin                       */
        short           nBottomMargin;  /* bottom margin                    */
}GCaption;

/*****************************************************************************
 * Cursor structure
 *****************************************************************************/
typedef struct _GCursor{
        unsigned char   ucType;         /* cursor type                      */
        GColor          gcColor;        /* cursor color                     */
}GCursor;

/*****************************************************************************
 * Font structure
 *****************************************************************************/
typedef struct _GFontSize{
        short           nAscent;        /* height from baseline to top line */
        short           nDscent;        /* height from baseline to bottom line */
        unsigned short  usWidth;        /* basic character width            */
        unsigned short  usHeight;       /* character height                 */
        unsigned short  usMaxWidth;     /* maximum character width          */
}GFontSize;
```

```
/*****************************************************************************
 * Simple font structure
 *****************************************************************************/
typedef struct _GSimpleFont{
        GTCHAR          szFontName[MAX_FONTNAME_LEN+1];   /* font name          */
        unsigned char   ucSize;                 /* font size                    */
        unsigned char   gptXScale;              /* horizontal scale             */
        unsigned char   gptYScale;              /* vertical scale               */
        unsigned char   ucWeight;               /* thickness                    */
        unsigned char   ucStyle;                /* shape                        */
}GSimpleFont;

typedef struct _GFont{
        short           nID;                    /* font ID                      */
        GTCHAR          szFontName[MAX_FONTNAME_LEN+1];   /* font name          */
        unsigned short  usCharacterSet;         /* character code               */
        short           nFontSizeCount;         /* number of font sizes         */
        GFontSize       *gfsSize;               /* pointer to font size array   */
        short           nFixedWidth;            /* fixed width font information */
        long            lFontSupport;       /* forms supported by the font (italics, bold, etc.) */
}GFont;

typedef struct _GFontAttribute{
        unsigned short  usWidth;                /* character width              */
        unsigned short  usHeight;               /* character height             */
        unsigned short  usWeight;               /* character thickness          */
        unsigned short  usItalic;               /* character italics            */
        unsigned short  usOutline;              /* character border             */
        void            *pData;                 /* additional information       */
        unsigned char   ucXScale;               /* horizontal scale             */
        unsigned char   ucYScale;               /* vertical scale               */
}GFontAttribute;

typedef struct _GFontPattern{
        GTCHAR          *pcChar;                /* pointer to character data    */
        GTCHAR          *pcNextChar;            /* pointer to next character data */
        short           nWidth;                 /* width                        */
        short           nHeight;                /* height                       */
        short           nBpp;                   /* number of dots per pixel     */
        char            *pcPattern;             /* pointer to pattern           */
        short           nGetSize;               /* get size flag                */
}GFontPattern;

typedef struct _GFontHandle{
        short           nID;                    /* font ID                      */
        short           nAttributeType;         /* use either pgfaAttribute or  */
                                                /* pnAttributeArray as attribute */
        GFontAttribute  *pgfaAttribute;         /* font attribute               */
        short           *pnAttributeArray;      /* font attribute (array)       */
}GFontHandle;
```

```
/*****************************************************************************
 * Image data structure
 *****************************************************************************/
typedef struct _GRFHeader{
        short           nType;          /* image type                       */
        short           nWidth;         /* width                            */
        short           nHeight;        /* height                           */
        short           nBpp;           /* number of bits per pixel         */
        long            lSize;          /* data size                        */
        unsigned char   *pData;         /* pointer to real data             */
}GRFHeader;
typedef struct _GRFHeaderDIB{
        short           nType;          /* image type                       */
        short           nWidth;         /* width                            */
        short           nHeight;        /* height                           */
        short           nBpp;           /* number of bits per pixel         */
        long            lSize;          /* data size                        */
        unsigned char   *pData;         /* pointer to real data             */
        unsigned char   *pPalette;      /* pointer to palette data          */
}GRFHeaderDIB;

/* image structure */
typedef struct _GImage{
        GRFHeader       *pImage;        /* pointer to image data structure  */
}GImage;


/*****************************************************************************
 * System time structure
 *****************************************************************************/
typedef struct _GSystemTime{
        unsigned long   ulLTime;        /* lower-side 32 bits for system time */
        unsigned long   ulUTime;        /* upper-side 32 bits for system time */
}GSystemTime;
```

```
/*****************************************************************************
 * Drawing environment structure
 *****************************************************************************/
/* GDraw structure */
typedef struct _GDraw{
        GRect               grLocalRect;            /* physical coordinate area                          */
        GPoint              gptLocalOrigin;         /* position of origin on physical coordinates        */
        GRect               grVirtualRect;          /* virtual coordinate area                           */
        GRect               grClipRect;             /* clipping rectangle                                */
        GPoint              gptPenPosition;         /* current position                                  */
        GColor              gcPenColor;             /* line color                                        */
        GPoint              gptPenSize;             /* line thickness                                    */
        short               nPenCap;                /* line end shape                                    */
        short               nPenJoin;               /* line contact shape                                */
        short               nPenDash;               /* line type No.                                     */
        short               nDashOffset;            /* line pattern offset                               */
        short               nTextMode;              /* text mode                                         */
        GColor              gcForeColor;            /* fill foreground color                             */
        GColor              gcBackColor;            /* fill background color                             */
        short               nFillPattern;           /* fill pattern No.                                  */
        short               nDrawCondition;         /* drawing condition                                 */
        short               nDrawingMode;           /* raster operation                                  */
        HGVRAM              hSystemVram;            /* display destination VRAM                          */
        HGVRAM              hDrawVram;              /* drawing destination VRAM                          */
        HGFONT              hFont;                  /* font                                              */
        void                *vgdftDraw;             /* function table for drawing                        */
        char                cDashPatterns[NUM_DASH_PATS][DASH_PAT_SIZE];
                                                    /* line pattern                                      */
        char                cFillPatterns[NUM_FILL_PATS][FILL_PAT_SIZE];
                                                    /* fill pattern                                      */
        GDrawParamPoly      gpPoly;                 /* polygon drawing parameter                         */
        GDrawParamWideLine  gpwWideLine;            /* wide line drawing parameter                       */
        GDrawParamOval      gpwOval;                /* circle, arc and sector drawing parameter          */
        unsigned short      usErrorCode;            /* previous error code                               */
        unsigned char       ucGradationType;        /* gradation type                                    */
                                                    /* (0: up to down, 1: left to right)                 */
        GColor              gcGradationColor1;      /* color1                                            */
        GColor              gcGradationColor2;      /* color2                                            */
        unsigned short      usVertexPos;            /* gradation vertex position (0 to 100)              */
        unsigned short      usGradationLevel;       /* gradation level (0 to 256)                        */
        unsigned char       ucColorMode;            /* actual VRAM color
                                                       environment information                        */
        GColor      gcRedMask;                      /* direct color R value mask                         */
        GColor      gcGreenMask;                    /* direct color G value mask                         */
        GColor      gcBlueMask;                     /* direct color B value mask                         */
        char        cRedShift; /                    * direct color R value shift value                  */
        char        cGreenShift;                    /* direct color G value shift value                  */
        char        cBlueShift;                     /* direct color B value shift value                  */
        GColor      gcForeColorOrg;                 /* foreground color (original)                       */
        GColor      gcBackColorOrg;                 /* background color (original)                       */
        GColor      gcPenColorOrg;                  /* background color (original)                       */

        unsigned long ulExParam;                    /* extension parameter                               */
        short       nExParam;                       /* extension parameter                               */
}GDraw;
```

```
/*****************************************************************************
 * Memory management structure
 *****************************************************************************/
typedef struct _GMemory{
        unsigned char       fUseSpace;      /* shows use (0)/not use (1) status for memory space */
        unsigned char       cReserve[3];    /* reserved space (for 4 byte environment adjustment) */
        unsigned long       ulSize;                     /* memory space size                    */
        struct _GMemory     *pvPrevMemorySpace;
                                            /* pointer to GMemory in previous memory space */
        struct _GMemory     *pvSmallMemorySpace;/* pointer to space GMemory in small        */
                                            /* memory space                                 */
        struct _GMemory     *pvLargeMemorySpace;/* pointer to space GMemory in large        */
                                            /* memory space                                 */
}GMemory;

typedef struct _GMemorySpaceInformation{
        unsigned char       ucType;                     /* memory space type                */
        unsigned char       ucPlane;                    /* memory space plane No.           */
        unsigned char       cReserve[2];    /* reserved space (for 4 byte environment adjustment) */
        char                *pvMemorySpace;         /* pointer to memory space allocated by user */
        unsigned long       ulMemorySpaceSize;
                                    /* size of memory space allocated by user (multiples of 32) */
        GMemory             *pgmNoUseMemoryTree;
                                        /* pointer to unused two-branch memory management */
}GMemorySpaceInformation;
```

```
/*****************************************************************************
 * Control related structures
 *****************************************************************************/
/* Design structure */
typedef struct GDesign{
        GBrush          gbBrush;        /* fill brush                   */
        unsigned short  usImageID;      /* image resource ID            */
}GDesign;

/* Focus movement structure */
typedef struct GFocusObject{
        unsigned short  usKeyCode;      /* virtual key code             */
        unsigned short  usType;         /* focus movement method        */
        unsigned short  usID;           /* ID of object being moved     */
}GFocusObject;

/* Focus movement structure */
typedef struct GFocusInformation{
        unsigned short  usCount;        /* number of focus settings     */
        GFocusObject    *pFocusArray;   /* focus setting (array)        */
}GFocusInformation;

/* Value structure */
typedef union GValue{
        short           nValue;         /* short value                  */
        unsigned short  usValue;        /* unsigned short value         */
        long            lValue;         /* long value                   */
        unsigned long   ulValue;        /* unsigned long value          */
        float           fValue;         /* float value                  */
}GValue;

/* GBaseWindow Export/Import structure */
typedef struct GBaseWindowProperty{
        unsigned short  usType;         /* object type                  */
        unsigned short  usID;           /* object ID                    */
        short           nX;             /* X coordinate                 */
        short           nY;             /* Y coordinate                 */
        short           nWidth;         /* width                        */
        short           nHeight;        /* height                       */
        unsigned long   ulStyle;        /* object shape                 */
        GFocusInformation *pFocusInfo;  /* focus object                 */
}GBaseWindowProperty;
```

```
/*****************************************************************************
 * NC data access related structures
 *****************************************************************************/
/* NC information structure */
typedef struct GNCControl{
        long            lMachine;       /* NC number                        */
                                        /* setting range: 1 to 255          */
        long            lSystem;        /* part system number               */
                                        /* setting range: 0 to 10           */
        long            lGround;        /* ground                           */
                                        /* 0: basic part system / foreground*/
                                        /* 1: basic part system / background*/
                                      /* 2: current part system during cross control /  */
                                      /*                                 foreground */
                                      /* 3: current part system during cross control /  */
                                      /*                                 background */
        unsigned long   ulAxis;         /* axis number                      */
                                        /* setting range: 0 to 16           */
}GNCControl;

/* NC data structure */
typedef union GNCValue{
        char            cValue;         /* char value                       */
        unsigned char   ucValue;        /* unsigned char value              */
        short           nValue;         /* short value                      */
        unsigned short  usValue;        /* unsigned short value             */
        long            lValue;         /* long value                       */
        unsigned long   ulValue;        /* unsigned long value              */
        double          dValue;         /* real number value                */
}GNCValue;
```

Revision History

Date of revision	Manual No.	Revision details
Mar. 2025	IB(NA)1501792-A	First edition created.

Global Service Network

AMERICA

MITSUBISHI ELECTRIC AUTOMATION INC. (AMERICA FA CENTER)
HQ and Central Region Service Center (Chicago)
500 CORPORATE WOODS PARKWAY, VERNON HILLS, ILLINOIS 60061, U.S.A.
TEL: +1-847-478-2500 / FAX: +1-847-478-2650
Minneapolis, MN Service Satellite
Detroit, MI Service Satellite
Grand Rapids, MI Service Satellite
Cleveland, OH Service Satellite
St. Louis, MO Service Satellite
Indianapolis, IN Service Saltellite

South/East Region Service Center (Georgia)
1845 SATELLITE BOULEVARD STE. 450, DULUTH, GEORGIA 30097, U.S.A.
TEL +1-678-258-4529 / FAX +1-678-258-4519
Charleston, SC Service Satellite
Charlotte, NC Service Satellite
Raleigh, NC Service Satellite
Dallas, TX Service Satellite
Houston, TX Service Satellite
Hartford, CT Service Satellite
Knoxville, TN Service Satellite
Nashville, TN Service Satellite
Huntsville, AL Satellite
Pittsburg, PA Service Satellite
Tampa, FL Service Satellite
Miami, FL Satelite
Lafayette, LA Service Satellite
Allentown, PA Satelite

Western Region Service Center (California)
5900-B KATELLA AVE. - 5900-A KATELLA AVE. CYPRESS, CALIFORNIA 90630, U.S.A.
TEL: +1-714-699-2625 / FAX: +1-847-478-2650
San Francisco, CA Satelite
Seattle, WA Service Satellite
Denver, CO Service Satellite

Canada Region Service Center (Toronto)
4299 14TH AVENUE MARKHAM, ONTARIO L3R OJ2, CANADA
TEL: +1-905-475-7728 / FAX: +1-905-475-7935
Edmonton, AB Service Satellite
Montreal, QC Service Satellite

Mexico Region Service Center (Queretaro)
Parque Tecnológico Innovación Querétaro, Lateral Carretera Estatal 431, Km 2+200, Lote 91 Modulos 1 y 2
Hacienda la Machorra, CP 76246, El Marqués, Querétaro, México
TEL: +52-442-153-6050
Monterrey, NL Service Satellite
Mexico City, DF Service Satellite

BRAZIL

MITSUBISHI ELECTRIC DO BRASIL COMÉRCIO E SERVIÇOS LTDA.
Votorantim Office
AV. GISELE CONSTANTINO,1578, PARQUE BELA VISTA, VOTORANTIM-SP, BRAZIL CEP:18.110-650
TEL: +55-15-3023-9000
Blumenau, Santa Catarina Office

EUROPE

MITSUBISHI ELECTRIC EUROPE B.V.
European Service Headquarters (Dusseldorf, GERMANY)
Mitsubishi-Electric-Platz 1 40882 RATINGEN, GERMANY
TEL: +49-2102-486-5000 / FAX: +49-2102-486-5910

South Germany Service Center (Stuttgart)
SCHELMENWASENSTRASSE 16-20, 70567 STUTTGART, GERMANY
TEL: + 49-711-770598-123 / FAX: +49-711-770598-141

France Service Center (Paris)
2 RUE DE L'UNION, 92565 RUEIL-MALMAISON CEDEX, FRANCE
TEL: +33-1-41-02-83-13 / FAX: +33-1-49-01-07-25

France Service Satellite (Lyon)
240, ALLEE JACQUES MONOD 69800 SAINT PRIEST FRANCE
TEL: +33-1-41-02-83-13 / FAX: +33-1-49-01-07-25

Italy Service Center (Milan)
VIA ENERGY PARK 14, VIMERCATE 20871 (MB) ITALY
TEL: +39-039-6053-342 / FAX: +39-039-6053-206

Italy Service Satellite (Padova)
VIA G. SAVELLI, 24 - 35129 PADOVA, ITALY
TEL: +39-039-6053-342 / FAX: +39-039-6053-206

U.K. Service Center
TRAVELLERS LANE, HATFIELD, HERTFORDSHIRE, AL10 8XB, U.K.
TEL: +44-1707-288-780 / FAX: +44-1707-278-695

Spain Service Center
CTRA. RUBI, 76-80 8174 SAINT CUGAT DEL VALLES, BARCELONA, SPAIN
TEL: +34-935-65-2236 / FAX: +34-935-89-1579

Poland Service Center
UL.KRAKOWSKA 50, 32-083 BALICE, POLAND
TEL: +48-12-347-6500 / FAX: +48-12-630-4701

Hungary Service Center
MITSUBISHI ELECTRIC EUROPE B.V. HUNGARIAN BRANCH
BUDAÖRS OFFICE PARK, SZABADSÁG ÚT 117., 2040 BUDAÖRS, HUNGARY
TEL: +36-70-433-2263

Turkey Service Center
MITSUBISHI ELECTRIC TURKEY ELEKTRİK ÜRÜNLERİ A.Ş
SERIFALI MAHALLESI KALE SOKAK. NO.41 34775
UMRANIYE, ISTANBUL, TURKEY
TEL: +90-216-969-2500 / FAX: +90-216-661-44-47

Czech Republic Service Center
MITSUBISHI ELECTRIC EUROPE B.V. PRAGUE OFFICE
Pekařská 621/7, 155 00 PRAHA 5, CZECH REPUBLIC
TEL: +420-734-402-587

Sweden Service Center
HAMMARBACKEN 14, P.O.BOX 750 SE-19127, SOLLENTUNA, SWEDEN
TEL: +46-8-6251200 / FAX: +46-8-6251014

Bulgaria Service Center
AKHNATON Ltd. (Service Partner)
4 ANDREJ LJAPCHEV BLVD. POB 21, BG-1756 SOFIA, BULGARIA
TEL: +359-2-8176009 / FAX: +359-2-9744061

Ukraine Service Center (Kiev)
CSC Automation Ltd. (Service Partner)
4 B, YEVHENA SVERSTYUKA STR., 02002 KIEV, UKRAINE
TEL: +380-44-494-3344 / FAX: +380-44-494-3366

South Africa Service Center
Adroit Technologies (Service Partner)
20 WATERFORD OFFICE PARK, WATERFORD DRIVE, CNR OF WITKOPPEN ROAD,
FOURWAYS JOHANNESBURG SOUTH AFRICA
TEL: +27-11-658-8100 / FAX: +27-11-658-8101

ASEAN

MITSUBISHI ELECTRIC ASIA PTE. LTD. (ASEAN FA CENTER)
Singapore Service Center
307 ALEXANDRA ROAD MITSUBISHI ELECTRIC BUILDING SINGAPORE 159943
TEL: +65-6473-2308 / FAX: +65-6476-7439

PHILIPPINES

MELCO FACTORY AUTOMATION PHILIPPINES INC.
Head Office
128 LOPEZ RIZAL STREET, BRGY., HIGHWAY HILLS, MANDALUYONG CITY , MM PHILIPPINES 1550
TEL: +63-2-8256-8042 / FAX: +632-8637-2294

Philippines Service Center
KM.23 WEST SERVICE ROAD SSH, CUPANG ,MUNTINLUPA CITY, PHILIPPINES
TEL: +63-2-8807-0420 / FAX: +63-2-8842-5202

VIETNAM

MITSUBISHI ELECTRIC VIETNAM CO., LTD.
Vietnam Ho Chi Minh Service Center
11TH & 12TH FLOOR, VIETTEL TOWER B, 285 CACH MANG THANG 8 STREET, WARD 12, DISTRICT 10,
HO CHI MINH CITY, VIETNAM
TEL: +84-28-3910-5945 / FAX: +84-28-3910-5947

Vietnam Hanoi Service Center
14TH FLOOR, CAPITAL TOWER, 109 TRAN HUNG DAO STREET, CUA NAM WARD,
HOAN KIEM DISTRICT, HA NOI CITY, VIETNAM
TEL: +84-24-3937-8075 / FAX: +84-24-3937-8076

INDONESIA

PT. MITSUBISHI ELECTRIC INDONESIA
Indonesia Service Center (Cikarang)
JL. KENARI RAYA BLOK G2-07A, DELTA SILICON 5, LIPPO CIKARANG - BEKASI 17550, INDONESIA
TEL: +62-21-2961-7797 / FAX: +62-21-2961-7794

MALAYSIA

MITSUBISHI ELECTRIC SALES MALAYSIA SDN. BHD.
Malaysia Service Center (Kuala Lumpur Service Center)
LOT 11, JALAN 219, P.O BOX 1036, 46860 PETALING JAYA, SELANGOR DARUL EHSAN, MALAYSIA
TEL: +60-3-7626-5032

Johor Bahru Service Center
9, Jalan Perjiranan 4/6, Bandar Dato Onn, 81100 Johor.
TEL: 07-3642250 / 2256

Pulau Pinang Service Center
25-G, Pusat Perniagaan Perdana Jaya, Jalan Permatang Rawa, 14000 Bukit Mertajam, Pulau Pinang
TEL: +60-4-510-1838 / Fax: +60-4-510-1835

THAILAND

MITSUBISHI ELECTRIC FACTORY AUTOMATION (THAILAND) CO., LTD.
Thailand Service Center (Bangkok)
101, TRUE DIGITAL PARK OFFICE, 5TH FLOOR, SUKHUMVIT ROAD, BANGCHAK, PHRA KHANONG,
BANGKOK, 10260 THAILAND
TEL: +66-2-092-8600 / FAX: +66-2-043-1231-33

INDIA

MITSUBISHI ELECTRIC INDIA PVT., LTD.
Service Head office and South India Service Center (Bangalore)
CNC eXPerience Park, PLOT NO. 57/A, PEENYA PHASE 3,
PEENYA INDUSTRIAL AREA, BANGALORE 560058, KARNATAKA, INDIA
TEL : +91-80-4655-2121
Chennai Service Satellite
Coimbatore Service Satellite
Hyderabad Service Satellite

North India Service Center (Gurgaon)
PLOT 517, GROUND FLOOR, UDYOG VIHAR PHASE-III, GURUGRAM 122008, HARYANA, INDIA
TEL : +91-124-463-0300
Ludhiana Service Satellite
Panthnagar Service Satellite
Delhi Service Satellite
Jamshedpur Service Satellite
Manesar Service Satellite

West India Service Center (Pune)
ICC-Devi GAURAV TECHNOLOGY PARK, UNIT NO.402, FOURTH FLOOR, NORTH WING,
SURVEY NUMBER 191-192 (P), NEXT to INDIAN CARD CLOTHING COMPANY Ltd,
OPP. VALLABH NAGAR, PIMPRI, PUNE- 411 018, MAHARASHTRA, INDIA
TEL : +91-20-6819-2274
Kolhapur Service Satellite
Aurangabad Service Satellite
Mumbai Service Satellite

West India Service Center (Ahmedabad)
204-209, 2ND FLOOR, 31FIVE, CORPORATE ROAD PRAHLADNAGAR,
AHMEDABAD -380015, GUJARAT, INDIA
TEL : + 91-79-6777-7888
Rajkot Service Satellite

CHINA

MITSUBISHI ELECTRIC AUTOMATION (CHINA) LTD. (CHINA FA CENTER)
CNC Call Center
TEL: +86-400-921-5130

Shanghai Service Center
NO.1386 HONG QIAO ROAD, CHANG NING QU, SHANGHAI 200336, CHINA
TEL: +86-21-2322-3030 / FAX: +86-21-2322-3000*8422
Qingdao Service Center
Suzhou Service Center
Wuhan Service Center
Ningbo Service Center
Hefei Service Center
Beijing Service Center
Tianjin Service Center
Xian Service Center
Dalian Service Center
Chengdu Service Center

Shenzhen Service Center
LEVEL8, GALAXY WORLD TOWER B, 1 YABAO ROAD, LONGGANG DISTRICT,
SHENZHEN 518129, CHINA
TEL: +86-755-2399-8272 / FAX: +86-755-8229-3686
Dongguan Service Center
Xiamen Service Center

KOREA

MITSUBISHI ELECTRIC AUTOMATION KOREA CO., LTD. (KOREA FA CENTER)
Korea Seoul Service Center
8F GANGSEO HANGANG XI-TOWER A, 401 YANGCHEON-RO, GANGSEO-GU,
SEOUL 07528 KOREA
TEL: +82-2-3660-9631 / FAX: +82-2-3664-8668

Korea Daegu Service Center
2F KT BLDG. 8, HOGUK-RO, BUK-GU, DAEGU 41518, KOREA
TEL: +82-53-382-7401 / FAX: +82-53-382-7412

Korea Changwon Technical Center
#205, 888, CHANGWON-DAERO, SEONGSAN-GU, CHANGWON-SI,
GYEONGSANGNAM-DO, 51532, KOREA
TEL: +82-55-261-5559 / FAX: +82-55-261-5558

TAIWAN

MITSUBISHI ELECTRIC TAIWAN CO., LTD. (TAIWAN FA CENTER)
Taiwan Taichung Service Center
NO. 8-1, GONGYEQU 16TH RD., XITUN DIST., TAICHUNG CITY 40768 , TAIWAN
TEL: +886-4-2359-0688 / FAX: +886-4-2359-0689

Taiwan Taipei Service Center
11F, NO.88, SEC.6, ZHONGSHAN N. RD., SHILIN DIST., TAIPEI CITY 11155, TAIWAN
TEL: +886-2-2833-5430 / FAX: +886-2-2833-5433

Taiwan Tainan Service Center
11F-1, NO.30, ZHONGZHENG S. RD., YONGKANG DIST., TAINAN CITY 71067, TAIWAN
TEL: +886-6-252-5030 / FAX: +886-6-252-5031

OCEANIA

MITSUBISHI ELECTRIC AUSTRALIA PTY. LTD.
Oceania Service Center
348 VICTORIA ROAD, RYDALMERE, N.S.W. 2116 AUSTRALIA
TEL: +61-2-9684-7269/ FAX: +61-2-9684-7245

Notice

Every effort has been made to keep up with software and hardware revisions in the contents described in this manual. However, please understand that in some unavoidable cases simultaneous revision is not possible.

Please contact your Mitsubishi Electric dealer with any questions or comments regarding the use of this product.

Duplication Prohibited

This manual may not be reproduced in any form, in part or in whole, without written permission from Mitsubishi Electric Corporation.

© 2025 MITSUBISHI ELECTRIC CORPORATION
ALL RIGHTS RESERVED.

MITSUBISHI ELECTRIC CORPORATION
HEAD OFFICE : TOKYO BLDG.,2-7-3 MARUNOUCHI,CHIYODA-KU,TOKYO 100-8310,JAPAN

MODEL	NC Designer2
MODEL CODE	-
Manual No.	IB-1501792

Made in JAPAN

Specifications are subject to change without notice.