Contents

Chapter 1

References

1.1 WSCballoonHelp

Derived from WSCbase, WSCwindow

At first, register the help string and the instance to the balloon help instance, and if the mouse pointer moves into it,the balloon help is poped up.

Method	Description
long registerClient(WSCbase* client, char* str)	Registers the instance to display the balloon helps.
long unregisterClient(WSCbase* client)	Cancels the registration of the specified.
WSCballoonHelp* WSGIappBalloonHelp()	Returns the default balloon help instance.

1.2 WSCbase

Derived from

This class is base class of another for WideStudio. The WideStudio application usually manages the instances by WSCbase class pointer.

Method	Description
long initialize()	Initializes the instance.
WSCbool getInitialized()	Returns the status of initializing.
char* getInstanceName()	Returns the instance name.
void setInstanceName(char*)	Specifies the instance name.
char* getClassName()	Returns the class name of the instance.
void* cast(char* class_name)	Supplies the function of down cast. Usually C++ language does not allow to cast a abstract pointer to a pointer of child class. So the instance has all child class pointer, cast() method seek for a specified class pointer from contained all the pointer of child classes.

WSCbool setProperty(char* pname, const WSC-variant &)	Sets the value into the property.
WSCvariant getProperty(char* pname)	Returns the value of the specified property
void setVisible(WSCbool fl)	Specifies the status of visibility.
WSCbool getVisible()	Returns the status of visibility.
void setSensitive(WSCbool fl)	Specifies the state of sensibility.
WSCbool getVisible()	Returns the sensibility
long draw()	Draws the instance.
long redraw()	Clears and draws the instance.
long update()	If needed, clears and draws the instance.
WSClistData & getChildren()	Returns the list of child instances. It functions this method of the classes which have a management ability of child instances.
void execProcedure(long trigger)	Executes the event procedures by specified trigger.
void execProcedure(char* pname)	Executes the event procedures by specified procedure name.
long setFocus(WSCbool fl = True)	Changes the state of the keyboard focus.
WSCbool getFocus()	Returns the state of the keyboard focus.
long setSpecialFocus(WSCbool fl = True)	Changes the state of the keyboard special focus.
WSCbool getSpecialFocus()	Returns the state of the keyboard special focus.
long getAllChildren(WSClistData &list)	Returns all of the children.
WSCbase* getParentWindow()	Returns the parent application window of the instance.
WSCbase* getParent()	Returns the parent instance.

1.3 WSCbaseDialog

Derived from WSCbase, WSCwindow

This class is base class of the popup dialogs.

Method	Description
long getStatus()	Returns the state which button is pushed.
long popup()	Pops up the dialog, and wait till that the buttons is pushed. Then returns which the buttons is pushed.

1.4 WSCbaseList

Derived from

This class manages the instance of WSCbase class. It is useful to seek out the instance.

Method	Description
WSDbaseList* WSGIappObjectList()	Returns the global instance of WSCbaseList which is a singleton in an application.

WSCbase* getInstance(char* cname, char* iname)	Returns the instance by a specified class name and instance name.
WSCRbase* getRemoteInstance(char* iname)	Returns the remote instance by a specified class name and instance name.
WSCbool existInstance(WSCbase* instance)	Examines whether the specified instance is valid or deleted.
void execUpdate()	Updates the instances need updating.

1.5 WSCcheckGroup

Derived from WSCradioGroup, WSCform, WSCbase

It is a form that has a check button group. The check button group consists of instances of the WSCvradio class which the WSNunique property is False.

Method	Description

1.6 WSCcolorSet

Derived from

Controls Color class instances. Color related information can be retreived using this class.

Method	Description
WSCcolorSet* WSGIappColorSet()	Obtain a global instance of color controle class existing on an application
WSDcolor* getColor(short cno)	Obtain a color instance from a color number
WSDcolor* getColor(char* cname)	Obtain a color instance from a color name
short getColorNo(char* cname)	Obtain a color number from a color name
short getDefaultColorNo(long kind)	Obtain a color number which is specified as a default color
WSDcolor* getDefaultColor(long kind)	Obtain a color instance of a specified default color
char* getColorName(short cno)	Obtain a color name from a specified color number
WSCbool existColor(short cno)	Checks whether specified color number exists

1.7 WSCcomboBox

Derived from WSCform, WSCbase,

This class is a text input field which has a selection menu. About setting of selection menu, please refer to attention.

Method	Description

1.8 WSCconductor

Derived from

Realize store feature. It serializes a specified instance to output into a file or input a instance from a file. WSGFloadWindow(), WSGFsaveWindow() functions enable you to utilize store feature easily. You can also use global functions

Method	Description
WSCconductor* WSGIconductor()	Obtain a global instance of store controle class existing per application
long saveGUI(WSDserialize* dest, char* name, WSCbase* inst)	Output the specified instance to the named serialize source
long loadGUI(WSDserialize* src, WSCbase** inst, WSCbase* parent)	Read the instance from the specified serialize source
long save(WSDserialize* dest, char* type, char* name, void* ptr)	Output the specified data to named serialize Used when serialize a normal data type not only GUI instase
long load(WSDserialize* dest, char* type, char* name, void* ptr)	Read a named data from the specified name serialized source
WSDserialize* beginTransaction(char* stype, char* sname)	Start store output specifying a serialize name
WSDserialize* endTransaction(WSDserialize* sr)	Complete a store output transaction specifying the serialize instance

1.9 WSCdialog

Derived from WSCbase, WSCwindow, WSCbaseDialog

This class is base class of the popup dialogs, and has a form used as client area.

Method	Description

1.10 WSCdirTree

Derived from WSCbase, WSCform, WSCscrForm, WSClist, WSCtreeList

Show directory tree view based on WSCtreeList class Tree type list. Selected directory can be obtained by the property WSNdirName.

Method	Description
long setSelectedDirName(char* dirname, WSCbool opened = True)	Specify the selected directory
char* getSelectedDirName()	Obtain a directory that is selected status.

1.11 WSCfform

Derived from WSCbase

A form which can be detached to a separated window by grasping a tab. It goes back original form status by moving to the initial position by grasping the tab.

Method	Description
void setFloated(WSCbool fl)	Specify a form status
WSCbool getFloated()	Obtain a form status

1.12 WSCfileSelect

Derived from WSCbase, WSCwindow, WSCbaseDialog

This class is a dialog class to select files. It is possible to move the directories, to mask and select files.

Method	Description
char* getFileName()	Returns the selected file name.

1.13 WSCfontSet

Derived from

Manage instances of WSDfont font class. This class is used when obtain a information regarding fonts.

Method	Description
WSCfontSet* WSGlappFontSet()	Obtain a global instance of a font management class existing per application
WSDfont* getFont(short fno)	Obtain a font instance by a font number
short getDefaultFontNo()	Obtain a font number of the specified default font.
WSDfont* getDefaultFont()	Obtain a default font instance

1.14 WSCform

Derived from WSCbase,

This class displays and manages the child instances.

Method	Description

1.15 WSCgrid

Derived from WSCbase, WSCform

Disply string data in a table format separated by a delimter

Method	Description
long getCellGeometry(WSCulong cx, WSCulong cy, short* x, short* y, WSCushort* w, WSCushort* h)	Obtain a specified sell size
long getHColumns();	Obtain the number of sell in the horizontal direction
long getVColumns();	Obtainn the number of sell in the vertical direction
long setItem(WSCulong cx, WSCulong, cy, WSCvariant val);	Set a value in a specified sell
long setCellForeColor(WSCulong cx, WSCulong cy, short color);	Set foreground color in a specified sell
long setCellForeColor(WSCulong cx, WSCulong cy, char* cname);	Set foreground color in a specified sell
long setCellBackColor(WSCulong cx, WSCulong cy, short color);	Set background color in a specified sell
long setCellBackColor(WSCulong cx, WSCulong cy, char* cname);	Set foreground color in a specified sell
long getItemAlignment(WSCulong cx, WSCulong cy);	Obtain specified sell alignment
WSCvariant getItem(WSCulong cx, WSCulong cy);	Obtain a specified sell value

1.16 WSChorzForm

Derived from WSCvertForm, WSCform, WSCbase

It is the form that has facility to enumerate child instances sideways. You can appoint the minimum size of child instances by the property WSNminimum.

Method	Description

1.17 WSCimageSet

Derived from

Manages WSDimage drawing class instance This class is used when obtaining image related information

Method	Description

WSCimageSet* WSGlappImageSet()	Obtain a global instance of an image management class existing per application
WSDimage* getImage(short no)	Obtain an image instance from the image number
WSDimage* getImage(char* name)	Obtain an image instance from a image file name
short getImageNo(char* name)	Obtain the image number from a image file name
char* getImageName(short no)	Obtain an image file name from the specified image number's image file
long destroyImage(char* fname)	Destroy a read image specifying a image file name
long destroyImage(short no)	Destroy image specifying the image number

1.18 WSCindexData

Derived from

Generic list dat struct. Since it automatically manages memory, you can easily operate data such as addition. Different from WSClistData, it can attach index on data.

Method	Description
WSCindexData()	Constructor of an indexed list data struct Create an instance of an indexed list data
WSCindexData(WSCindexData& src)	Copy constructor of an indexed list data struct Duplicate the given list
long setData(char* index, void* data)	Store data with a specified index
char* getIndex(long pos)	Obtain an index name of a specified position
void* getData(char* index)	Obtain data corresponding to a specified index
void* getData(long pos)	Obtain a specified position data
long getNum()	Obtain the number of data that have list
long del(char* index);	Delete data corresponding to a specified index
long delPos(long pos);	Delete a specified position data
void clear();	Delete all retaining data
void*& WSCindexData::operator [] (char* index)	Obtain data correspoinding to a specified index
WSCindexData& operator = (WSCindexData& src)	Copy operator

1.19 WSCindexForm

Derived from WSCform, WSCbase

This class has index tabs and changes the screen of its area.

Method	Description

1.20 WSCindexVariantData

Derived from

Generic list data struct. Since memory mangement is automatically done, you can easily operate data such as addtion. Different from WSCindexData, it uses variant type data

Method	Description
WSCindexVariantData()	Constructor of an indexed variant type data list struct Create a instance of an indexed variant type data list
WSCindexVariantData(WSCindexVariantData&)	Copy constructor of an indexed variant type data Duplicate the given list
long setData(char* index, WSCvariant data)	Store a variant type data with specifying the index Since it is variant type, various type data can be specified
char* getIndex(long pos)	Obtain a specified position index
WSCvariant &getData(char* index)	Obtain a data corresponding to a specified index
WSCvariant &getData(long pos)	Obtain a specified position data
long getNum()	Obtain the number of data that have list
long del(char* index);	Delete a data corresponding to a specified indedx
long delPos(long pos);	Delete a specified position data
void clear();	Delete all retaining data
WSCvariant& operator [] (char* index)	Obtain a data corresponding to a specified index
WSCindexVariantData& operator = (WSCindexVariantData& src)	Copy operator

1.21 WSCinputDialog

Derived from WSCbase, WSCwindow, WSCbaseDialog,

This class is a text input dialog by the keyboard. the property: WSNlabelString contains the input string.

Method	Description

1.22 WSCj3wform

Derived from WSCform, WSCbase

Disply J3D file that uses J3W computer graphics libraries. By specifying J3D file made by J3W, displays its graphics.

Method	Description

1.23 WSClist

Derived from WSCbase, WSCform, WSCscrForm

This class shows the list of strings, the tree of strings, the choice of string and so on.

Method	Description
void setLabelHeight(WSCushort* height)	Specify the item height.
void delAll()	Deletes all the items.
WSCbase* getLabel(long pos)	Returns the internal instance which displays the item string in the list.
void getNum()	Returns the number of items.
WSCstring getItem(long pos)	Returns the string of the specified position.
void addItem(char* item, long pos = -1)	Inserts the item into the specified position.
void replaceItem(char* item, long pos)	Replaces the item of the specified position.
void delPos(long pos);	Deletes the item of the specified position.
long getSelectedPos();	Returns the position of the selected item.
void setSelectPos(long pos);	Makes the item of the specified position selected.
char* getSelectedItem();	Returns the string of the selected item.
void setTopPos(long pos);	Scrolls the list to display the item of the specfied position most aloft.
void setBottomPos(long pos);	Scrolls the list to display the item of the specfied position most beneath.
void updateList();	Updates the list which is changed by the methods.
void setLabelClass(char* class_name);	Registers the class which is used to display the items.
WSCbool getSelectItemChanged()	Returns the state whether the selection of the items has changed.
void setSelectItemChanged(WSCbool fl);	Sets the value into the flag which indicates whether the selection of the items has changed.
WSClistData* getLabels();	Returns a list of the internal instances which display the items.
void getSelectedLabels(WSClistData& list);	Returns the internal instances which dislpay the selected items with the specified list.
long setItemVisible(long pos, WSCbool fl);	Sets the visiblity of the item of the specified position.
long setItemValue(long pos, long kind, long bal);	Sets the specified value into the specified attributes of the specified item.
long getSortPos();	Returns the number of the sort button on the detail mode.
long getTopPos();	Returns the most top line number. on the view port of the scroll area.
long getBottomPos();	Returns the most bottom line number. on the view port of the scroll area.
void setEnableActivate(WSCbool fl);	Set whether create the ACTIVATE event when the item is selected.
void cancelInput();	Cancels the keyboard input string.
void getInputString();	Returns the keyboard input string.

1.24 WSClistData

Derived from

This class is general-purpose list data structure. it manages the memories by automaticaly. So you can add or remvoe data easily.

Method	Description
WSClistData()	This is the constructor of WSClistData. Creates a instance.
WSClistData(long segment_size)	This is the constructor of WSClistData. Creates a instance which has specified segment size. The segment size is the memory size to add. If you manage a lot of data, add and remove many times, large size is better. The default is 16 which can contain 16 pointers.
long add(void* data, long pos = -1)	Adds the data into the specified position.
void setData(long pos, void* data)	Replaces the data of the specified position.
void* getData(long pos)	Returns the value of the specified position.
long getNum()	Returns the number of data which the list contains.
void** getBuf()	Returns the data buffer which the list contains.
long del(void*)	Removes the value from the list.
long delPos(long pos)	Removes the data of the specified position.
void clear()	Clears the data buffer,
WSClistData & operator = (WSClistData &)	Copies the list.
void* & operator[](long pos)	Return the data of the specified position. It seems like the elements of a array.

1.25 WSClocaleSet

Derived from

Manage language information This class is used when obtaining information related with encoding or locale.

Method	Description
WSClocaleSet* WSGIappLocaleSet()	Obtain a global instance of encoding information management class existing per application
char* getDefaultLocaleName()	Obtain a default locale name
void setDefaultEncoding(long encoding)	Set default encoding
long getDefaultEncoding()	Obtain default encoding
long getSystemLocaleEncoding()	Obtain encoding used in the system I/O
char* getLocaleString(char* locale, char* index, long encode)	Obtain strings of each specified locale language.

1.26 WSCmainWindow

Derived from WSCbase, WSCwindow

This class provide a top level window. Usually used by the application window. It redefined the property WSNexit of WSCwindow True but the other properties are same value.

Method	Description

1.27 WSCmenuArea

Derived from WSCbase, WSCform

Provide a pull down menu space in a rectagle window reagion. The width follows by the size of the parent window. Properties:WSNx, WSNy are fixed at right upper edge.

Method	Description

1.28 WSCmessageDialog

Derived from WSCbase, WSCwindow, WSCbaseDialog

This class is a dialog class to display the messages. It is used by setting the messages to the WSNlabelString property.

Method	Description

1.29 WSCngbase

Derived from WSCnwbase, WSCbase

This class is base class which does not have window resource. You can create new class with this class which draws no picture under running. For example,it is used by the timer class (WSCvtimer).

Method	Description

1.30 WSCnwbase

Derived from WSCbase

This class is a sub class which has no window resource. A class which inherits this class and has no window resource behave whether it has a window resource.

Method	Description
void setBlinkFore(WSCbool fl)	Set the fore state(drawn by fore color) or the opposite state(disappeared or drawn by blink color) of the blink.
void setOutSideMousePress(WSCbool fl)	Sets the flag whether it receives the mouse pressed event of the parent instance which is out of the own area.
void setOutSideMouseMove(WSCbool fl)	Sets the flag whether it receives the mouse moved event of the parent instance which is out of the own area.
void setOutSideMouseRelease(WSCbool fl)	Sets the flag whether it receives the mouse released event of the parent instance which is out of the own area.
WSCbool getOutSideMousePress()	Returns the flag whether it receives the mouse pressed event of the parent instance which is out of the own area.
WSCbool getOutSideMouseMove()	Returns the flag whether it receives the mouse moved event of the parent instance which is out of the own area.
WSCbool getOutSideMouseMove()	Returns the flag whether it receives the mouse released event of the parent instance which is out of the own area.

1.31 WSCopenglForm

Derived from WSCform, WSCbase

A form for drawing/displying an OpenGL computer graphics. Various computer graphics can be displayed by using OpenGL APIs.

Method	Description

1.32 WSCoption

Derived from WSCbase, WSCnwbase, WSCvlabel

This class is used to choice a value by a menu. Clinking the instance, a menu appears.

Method	Description
long setSelectValue(long value, WSCbool flag)	Sets the sensibility of the elemnt of the menu by the specified value.
long setItemSensitive(long no, WSCbool flag)	Sets the sensibility of the element of the menu by the specified position.
long getValue()	Returns the value of the selected element.
long getItems()	Returns the number of the available menus.

WSCbool* getSelectStatus()	Returns the array of the states whether the elements are sensitive. (True: sensitive, False: insensitive).

1.33 WSCpopupMenu

Derived from WSCbase, WSCwindow, WSCpulldownMenuPopup

This class is the popup menu which is poped up by clicking the mouse pointer. A GUI instance which is registered to the popup menu instance, pops up it.

Method	Description
long registerClient(WSCbase* client)	Resisters the instance to display the popup menu.
long unregisterClient(WSCbase*)	Cancels the registration of the specified instance.

1.34 WSCprform

Derived from WSCbase, WSCform

Provide a controle function having other objects internally in a rectagle region being managed as a child object.

Method	Description
long prdraw()	Create a PostScript file

1.35 WSDvariant

Derived from

This is the event procedure class which manages the event procedure,the procedure name,the trigger.

Method	Description
WSCprocedure(char* epname, long trigger)	Creates a instance of WSCprocedure class.
long setFunction(void(*func)(WSCbase*), char* fname)	Specifies the event procedure. The type of the event procedure is as follows. void foo(WSCbase*)
long setProcName(char* pname)	Specify the procedure name
long setTrigger(long trigger)	Specify the trigger.
void setInternal(WSCbool fl)	Specify the flag of whether internal instance.
char* getProcName()	Returns the event procedure name.
char* getFunctionName()	Returns the function name.
long getTrigger()	Returns the trigger id.
WSCbool getInternal()	Returns the status of the internal instance flag.

1.36 WSCpulldownMenu

Derived from WSCbase, WSCnwbase, WSCvlabel

This class realizes the pulldown menu. A menu appears by clicking of the mouse pointer.

Method	Description
long getItems()	Returns the number of the menu elements.
long setItemSensitive(short no, WSCbool flag)	Sets the sensibility of the element by the specified position (top:0, 1, 2...).
WSCbool* getItemSensitive()	Returns the array of the states whether the elements are sensitive. (True: sensitive, False: insensitive).
long getValue()	Returns the menu id of the selected element.
long setValueSensitive(short value, WSCbool flag)	Sets the sensibility of the elemnt of the menu by the specified value.
long addItem(char* lb, char* op, char* shortc, long id)	Add a menu element to menu.
long beginCascacde(char* lb)	Declare adding a new sub menu starts.
long endCascacde()	Declare adding the sub menu done.

1.37 WSCpulldownMenuPopup

Derived from WSCwindow, WSCbase

Method	Description

1.38 WSCradioGroup

Derived from WSCform, WSCbase

It is a form that has a radio button group. The radio button group consists of instances of the WSCvradio class which the WSNunique property is True.

Method	Description

1.39 WSCscrForm

Derived from WSCform, WSCbase

This class has a wide area which can be scrolled and vertical,hosizontal scroll bars.

Method	Description

long getVisibleWidth()	Returns the real width of the area.
long getVisibleHeight()	Returns the real height of the area.
WSCbase* getScrForm()	Returns the internal instance of the class: WSCscrFrame which provides the scrolled area.

1.40 WSCscrFrame

Derived from WSCbase, WSCform

This class is used by WSCscrForm internally and supplies the scrolling area. You can get the instance of WSCscrFrame with WSCscrForm::getScrFrame().

Method	Description

1.41 WSCsform

Derived from WSCform, WSCbase

This class has several forms separated by separator, and you can freely chane the width/height of them with the separator by clicking of the mouse pointer. For example: The folloing picture shows a separated form which has 2 scrolled forms.

Method	Description
long getStatus()	Returns the state whether the forms is under resizing by the separator.

1.42 WSCstring

Derived from

Generic strings struct type. You can easily operate strings since memory management is automatically doned internal.

Method	Description
WSCstring()	String class constructor Create a string class instance
WSCstring(char* str, long encoding = WS_EN_DEFAULT)	String class constractor
WSCstring(WSCstring& str)	String class copy constructor Create a string class instance
void setString(char* str, long encode = WS_EN_DEFAULT)	Set strings
char* getString(long encode = WS_EN_DEFAULT)	Obtain strings

long getChars()	Obtain the number of charactor strings currently set
long getEncoding()	Obtain encoding that currently set
long isExist(char* str, long encoding = WS_EN_DEFAULT)	Search for strings
void addString(const WSCstring& str)	Add strings
void addString(const WSCvariant& str, long encoding=WS_EN_DEFAULT)	Add strings
void addString(char* str, long encoding=WS_EN_DEFAULT)	Add strings
void cutString(WSCulong pos);	Delete strings from the position of the specified character.
void insertString(WSCulong pos, WSCstring& str);	Insert strings at a specified character position
void deleteChar(WSCulong pos)	Delete one character at a spacified character position.
void deleteChars(WSCulong pos, WSCulong len)	Delete strings at a specified character position
void clear()	Clear strings
void delLineFeed()	Delete linefeed characters in starings
void delString(char* str, lon num, long encoding = WS_EN_DEFAULT)	Search and delete the specified number of strings from a specified strings When 0 is specified as the number of deletion, it delete all characters that are muched with
void delHeadSpace();	Chop halfsize spaces locating the top of strings
void to_upper();	Convert the alphabets to upper characters
void to_lower();	Convert the alphabets to the lower characters
long replaceString(char* src, char* dest, long num, long encoding= WS_EN_DEFAULT)	Replace specified strings.
long getWords(char* sep, long encoding = WS_EN_DEFAULT)	Obtain the number of words using sep as a separator Obtain the number of words using a whitespace as a separator when sep and encoding are omitted.
long getWordCharPos(long no, char* sep, long encoding = WS_EN_DEFAULT)	Obtain a position of charactors of no-th words with sep as a separator regarding the top as 0 Ottain a position of words with a whitespace as its separator when sep and encoding are omitted
WSCstring getWord(long no, char* sep, long encoding = WS_EN_DEFAULT)	Obtain a no-th word with a specified separator couting the top 0 Obtain a word with a whitespace as a separator when sep and encoding are omitted.
long getLines()	Obtain the number of lines
WSCstring& operator = (const WSCstring& str);	Substitution operator
WSCstring& operator = (char* str);	Substitution operator
WSCstring& operator = (const WSCvariant& str);	Substitution operator
WSCstring& operator += (const WSCstring& str);	Strings concatenating operator
WSCstring& operator += (char* str);	Strings concatenating operator
WSCstring& operator += (const WSCvariant& str);	Strings concatenating operator
WSCstring& operator (const WSCstring& str);	Strings concatenating operator

WSCstring& operator (char* str);	Strings concatenating operator
WSCstring& operator (const WSCvariant& str);	Strings concatenating operator
WSCstring& operator + (const WSCstring& str);	Strings combining operator
WSCstring& operator + (char* str);	appends the specified string to the string.
WSCstring& operator + (const WSCvariant& str);	Strings combining operator

1.43 WSCtextField

Derived from WSCbase, WSCform

This class is a multi-line text input field which has vertical/horisontal scroll bar.

Method	Description
WSCbase* getTextFrame()	Returns the internal instance which provides multi-line text input field (the class:WSCvmifield).
void addString(char* var)	Adds the specified string to the input text.
WSCushort* getBuf()	Returns the internal text input buffer.
void replaceSelectedString(char* str, long encoding = WS_EN_DEFAULT);	Replace the selected string.
WSCstring getSelectedString()	Returns the selected string.
WSCstring getString()	Returns the input string.
void deleteSelectedString()	Deletes the selected string.
long setSelect(long pos, long len)	Make the string selected.
long setSelectedPos()	Returns the position of the selected string.
long setLines()	Returns the lines of the string.
long getTopLine()	Returns the most top line number of the string under displaying.
long setTopLine(long line)	Specify the most top line number.
long getBottomLine()	Returns the most bottom line number of the string under displaying.
long setBottomLine(long line)	Specify the most bottom line number.

1.44 WSCtform

Derived from WSCform, WSCbase

It is a form which has a title and is surrounded with a frame.

Method	Description

1.45 WSCtreeList

Derived from WSCbase, WSCform, WSCscrForm, WSClist

It is a list to indicate a tree,and It is same that the property WSNtype of WSClist class "tree". You can put data to list from specified data source directly.

Method	Description

1.46 WSCvarc

Derived from WSCbase, WSCnwbase, WSCvpolyAttr

This class draws a circle/arc/fan which is filled or not.

Method	Description

1.47 WSDvariant

Derived from

This class can be used for variouse kinds of type.

Method	Description
char getChar()	Returns the value of type: char.
WSCuchar getUnsignedChar()	Returns the value of type: unsigned char.
short getShort()	Returns the value of type: short.
WSCushort getUnsignedShort()	Returns the value of type: unsigned short.
long getLong()	Returns the value of type: long.
WSCulong getUnsignedLong()	Returns the value of type: unsigned long.
int getInt()	Returns the value of type: int.
WSCuint getUnsignedInt()	Returns the value of type: unsigned int.
void* getVoidPtr()	Returns the value of type: void pointer.
char* getCharPtr()	Returns the value of type: char pointer.
float getFloat()	Return the value of type: float.
double getDouble()	Return the value of type: double.
void setValue(TYPE val)	Sets the value. You can pass a value of TYPE: char, WSCuchar, short, WSCushort, long, WSCulong, int, WSCuint, char*, void*, float, double.
WSCvariant(TYPE val)	This is the constructor with an initial value. You can pass a value of TYPE:char, WSCuchar, short, WSCushort, long, WSCulong, int, WSCuint, char*, void*, float, double.
WSCvariant& operator = (TYPE)	Substitutes the specified value of TYPE to the instance. You can specify TYPE: char, WSCuchar, short, WSCushort, long, WSCulong, int, WSCuint, char*, void*, float, double.
long getType()	Returns the identifier of the containd data type.
char* getTypeName()	Returns the type name of the containd data type.
void clear()	clear the contained data.

1.48 WSCvarrow

Derived from WSCbase, WSCnwbase, WSCvlabel, WSCvbtn

This class provides an arrow button. It is same as WSCvbtn except drawing an arrow.

Method	Description

1.49 WSCvballoonHelp

Derived from WSCbase, WSCnwbase, WSCngbase

This class indicates a balloon help string when the mouse pointer moves above the target instance. Set the name of target instance to the property WSNclient, and set the help string to the property WSNlabelString.

Method	Description

1.50 WSCvbarGraph

Derived from WSCbase, WSCnwbase

This class indicates bar graph. You can set data by the proeprty WSCvalue as follows... WSNvalue: 10,20,30,40,50,60,70,... ¿

Method	Description

1.51 WSCvbtn

Derived from WSCbase, WSCnwbase, WSCvlabel

This is a push button class. It dents by pressing of the mouse pointer, and fires an activate event by releaseing.

Method	Description
void setDrawFocusBorder(WSCbool fl)	Specify the attribute whether painting the keyboard focused border under being focused.

1.52 WSCvclock

Derived from WSCbase, WSCnwbase, WSCvlabel

This shows a clock. it indicates the year,month,day,hour,minute,second.

Method	Description

1.53 WSCvcsocket

Derived from WSCbase, WSCnwbase, WSCngbase

Client-side socket class to connect to a server-side socket class and establish a socket communication Set the referral host address in the property: WSNip, and the port number in the WSNport.

Method	Description
long exec()	Connect to the server-side socket specified with the property:WSNip, WSNport and raises an ACTIVATE event when it is succeeded. Actuall communication processes should be implemented in the event procedure started by the ACTIVATE
long read(WSCuchar* buffer, long size)	Read data from a connected socket in the event procedure started by the ACTIVATE
long write(WSCuchar* buffer, long size)	Write data to a connected socket in the event procdure started by the ACTIVATE

1.54 WSCvdb

Derived from WSCbase, WSCnwbase, WSCngbase

Refer to/update a database data through ODBC interface or DB interface. SQL sentences are used for data referencing, data updating. You can select database interface by the property WSNtype

Method	Description
long open(char* hostname, char* username, char* passwd, char* dbname, char* port)	Connect to a database
long open()	Connect to a database
long close()	Disconnect from a database
WSCbool isOpen()	Obtain a status of dabase connection
long sqlExecute(const char* sql)	Issue a SQL sentence against a database
long beginTran()	Start transaction
long commitTran()	Commit transaction
long abortTran()	Abort transaction
long getErrorMsg(char* buffer, long buflen)	Obtain error strings

1.55 WSCvdrawingArea

Derived from WSCbase, WSCnwbase

This class presents a area to painting graphics freely. You can draw some graphics by drawing method on the event procedure by WSEV_EXPOSE.

Method	Description
long setForeColor(char* cname)	This method sets the foreground color by the specified color name.
long setBacKColor(char* cname)	This method sets the background color by the specified color name.
long setLineWidth(short linewidth)	This method sets the line width.
long setLineDashType(char no)	This method sets the dash type of the line.
long setHatchPattern(char no)	This method sets the hatch pattern which is used by drawFillxxx method.
long setRegion(short x, short y, unsigned short w, unsigned short h);	This method limits the area for painting.
long drawArc(short x, short y, unsigned short w, unsigned short h, short a1, short a2);	Draws an arc, circle, oval.
long drawFillArc(short x, short y, unsigned short w, unsigned short h, short a1, short a2, char kind);	Draws an filled arc(pie, chord), filled circle, filled oval.
long drawLine(short x1, short y1, short x2, short y2);	Draws a line.
long drawLines(WSCpoint* pt, long num);	Draws a poly-line.
long drawRect(short x, short y, unsigned short w, unsigned short h);	Draws a rectangle.
long drawFillRect(short x, short y, unsigned short w, unsigned short h);	Draws a filled rectangle.
long drawRects(WSCrect* pt, long num);	Draws rectangles.
long drawFillRects(WSCrect* pt, long num);	Draws filled rectangles.
long drawPoly(WSCpoint* pt, long num);	Draw a poligon.
long drawFillPoly(WSCpoint* pt, long num);	Draw a filled poligon.
long drawGradation(long type, short col1, short col2, short col3, short x, short y, WSCushort w, WSCushort h, WSCuchar grad_margin);	Draws a rectangle which color gradated.
long drawImage(short x, short y, WSCushort w, WSCushort h, WSDimage* img, char align);	Draws an image.
long drawStretchedImage(short x, short y, WSCushort w, WSCushort h, WSDimage* img);	Draws a stretched image.
long drawString(short x, short y, WSCushort w, WSCushort h, char font_no, char align, char* string, long encoding = WS_EN_DEFAULT);	Draws the specified string in the specified area.
long drawFillString(short x, short y, WSCushort w, WSCushort h, char font_no, char align, char* string, long encoding = WS_EN_DEFAULT);	Draws the specified string in the specified filled rectangle.

1.56 WSCverbList

Derived from WSCbase, WSCform, WSCscrForm, WSClist

It is a list to indicate in detail, and It is same that the property WSNtype of WSClist class "detail". You can put data to list from specified data source directly. DATA FORMAT:

[PATH of an icon file],item-1,item-2,item-3,... [PATH of an icon file],item-1,item-2,item-3,... ... ¿You can omit path name of icon,and WSNiconUse is used for default value..

Method	Description

1.57 WSCvertForm

Derived from WSCform, WSCbase

It is the form that has facility to enumerate child instances lengthwise. You can appoint the minimum size of child instances by the property WSNminimum.

Method	Description

1.58 WSCvfbtn

Derived from WSCbase, WSCnwbase, WSCvlabel, WSCvbtn

This button rises if the mouse pointer comes into. It is same as WSCvbtn about the other function,

Method	Description

1.59 WSCvgraphMatrix

Derived from WSCbase, WSCnwbase

This class indicates lattice for a graph.

Method	Description

1.60 WSCvgraphScale

Derived from WSCbase, WSCnwbase

This class indicates a scale bar of a graph. You can change the direction of the bar by the property WSNorientation. Set data to the property WSNvalue as follows... WSNvalue: 0,1,2,3,4,5,6,7,... ¿And, set the bar length of tick on the bar by the property WSNbarThickness as follows... WSNbarThickness: 10,5,5,5,5,10,5,5,5,5,10 ¿

Method	Description

1.61 WSCvifield

Derived from WSCbase, WSCnwbase, WSCvlabel

This is a one-line text input field.

Method	Description
void addString(char* var)	Adds the specified string to the input text.
WSCushort* getBuf()	Returns the internal text input buffer.
void replaceSelectedString(char* str, long encoding = WS_EN_DEFAULT);	Replace the selected string.
WSCstring getSelectedString()	Returns the selected string.
WSCstring getString()	Returns the input string.
void deleteSelectedString()	Deletes the selected string.
long setSelect(long pos, long len)	Make the string selected.
long setSelectedPos()	Returns the position of the selected string.

1.62 WSCvkflabel

Derived from WSCvklabel, WSCvlabel, WSCnwbase, WSCbase

Method	Description

1.63 WSCvkslabel

Derived from WSCvifield, WSCvlabel, WSCnwbase, WSCbase

None..

Method	Description

1.64 WSCvlabel

Derived from WSCbase, WSCnwbase

This shows a text/pixmap.

Method	Description
char getAlignment()	Returns the alignment of the text.

1.65 WSCvline

Derived from WSCbase, WSCnwbase,

This class draws a line.

Method	Description

1.66 WSCvlineGraph

Derived from WSCbase, WSCnwbase

This class indicates poly-line graph. You can set data by the proeprty WSCvalue as follows...

Method	Description

1.67 WSCvmeter

Derived from WSCbase, WSCnwbase, WSCvpolyAttr

This class indicates a value by a bar between 0 and 100. You can set a start and end value. The following picutre shows the left side:start value which is 0, the right side: end value which is 60.

Method	Description

1.68 WSCvmifield

Derived from WSCbase, WSCnwbase, WSCvlabel, WSCvifield

This is a multi-line text input field.

Method	Description

1.69 WSCvodbc

Derived from WSCbase, WSCnwbase, WSCngbase

Refer to/update database data through the ODBC interface. Use SQL to refer to data or update data.

Method	Description
long open(char* dsn, char* username, char* passwd)	Connect to database
long open()	Connect to database
long close()	Cut off a connection to database

WSCbool isOpen()	Obtain status of database connection
long sqlExecute(const char* sql)	Issue SQL syntax against database
long beginTran()	Start transaction
long commitTran()	Commit a transaction
long abortTran()	Abort transaction
long getErrorMsg(char* buffer, long buflen)	Obtain error strings

1.70 WSCvpifield

Derived from WSCvifield, WSCvlabel, WSCnwbase, WSCbase

Method	Description

1.71 WSCvpoly

Derived from WSCbase, WSCnwbase, WSCvpolyAttr

This class indicates a poligon which is filled or not.

Method	Description

1.72 WSCvpolyAttr

Derived from WSCbase, WSCnwbase

This class is a sub class which contains the state of filled or hatched, the line width, the blink type and so on.

Method	Description

1.73 WSCvradio

Derived from WSCbase, WSCnwbase, WSCvlabel, WSCvtoggle

This is a radio button which has a state: True / False. it has same function as WSCvtoggle class but has a default checking image "v" already.

Method	Description

1.74 WSCvrect

Derived from WSCbase, WSCnwbase, WSCvpolyAttr

This draws a rectangle.

Method	Description

1.75 WSCvremoteClient

Derived from WSCbase, WSCnwbase, WSCngbase

Specify a communication port number of an agent in communicating with an agent who controles remote instances

Method	Description

1.76 WSCvremoteServer

Derived from WSCbase, WSCnwbase, WSCngbase

A class for constructing the remote instance server. Pass remote class information existing in the server process to the agent. Deploy one WSCvremoteServer class instance on an application window of the project when constructing a remote instance server.

Method	Description

1.77 WSCvscrBar

Derived from WSCbase, WSCnwbase

This is a scroll bar.

Method	Description

1.78 WSCvslider

Derived from WSCbase, WSCnwbase

This is a slider that you can choice a value between the minimum value and the maximum by the mouse pointer.

Method	Description

1.79 WSCvspace

Derived from WSCbase, WSCnwbase, WSCngbase

This class secures space on the form. It makes the form as WSCvertForm/WSChorzForm when the property WSNextension is True and moves other instaces like as spring. It does nothing when the property WSNextension False, but secures space when with other WSCvspace which WSNextention is True.

Method	Description

1.80 WSCvssocket

Derived from WSCbase, WSCnwbase, WSCngbase

Server side TCP socket class. Used in accepting a connection from a clien-side socket class to enable a socket communication Specify a bind port number in the property WSNport. Also usually, property WSNip is not used but when you want to confine a bind IP address of the machine which have multiple IP addresses, WSNip can be specified to bind the IP address.

Method	Description
long read(WSCuchar* buffer, long size)	Read data from a connected socket in the event procedure staring by the ACTIVATE
long write(WSCuchar* buffer, long size)	Write data to a connected socket in the event procedure staring by the ACTIVATE

1.81 WSCvtimer

Derived from WSCbase, WSCnwbase, WSCngbase

This class generates the WSEV_ACTIVATE event by appointed time interval. You can specify the time interval to the property WSNinterval, and consecutive trigger to the property WSNcont. Set True to the property WSNrunning to run the timer and set False to stop it. It will stop if the property WSNcont is False and it fires once.

Method	Description

1.82 WSCvtoggle

Derived from WSCbase, WSCnwbase, WSCvlabel

This is a toggle button which has a state: True / False. You can make a toggle group and variouse forms of value selection by several toggles.

Method	Description

WSCbool getStatus()	Returns the state of the toggle.
long setStatus(WSCbool state, WSCbool create_event=True)	Set the state of the toggle.
long getGroupValue()	Returns the property: WSNid of the selected instance which is grouped by unique-selection, or returns the bit or-ed value of the properties: WSNid of the selected instances which is grouped by multi-selection.
long setGroupValue(long value)	Sets the selected toggle of a group of toggles by the specified group value.
long clearGroupValue()	Clear the selected state.

1.83 WSCvudpsocket

Derived from WSCbase, WSCnwbase, WSCngbase

Broadcasting class using UDP sockets Set a broadcast address in the property:WSNip, and the port number in WSNport. When there is broadcasting from another UDP socket, the ACTIVATE envent arises. Data can be exchaged in the procedure which starts with the ACTIVATE event.

Method	Description
long read(WSCuchar* buffer, long size)	Read the broadcasted data from a UDP socket in a procedure which starts by the ACTIVATE event.
long write(WSCuchar* buffer, long size)	Write broadcast data into a socket to send

1.84 WSCwindow

Derived from WSCbase

This class provide a top level window. Usually used by the application window.

Method	Description
long setMapStatus(long)	Changes the window stacks.
virtual void adjustForm()	Calls this method when resized.

1.85 WSCwizardDialog

Derived from WSCbase, WSCwindow, WSCbaseDialog,

This class provides a wizard dialog. It has the button:"Back","Next" and "Finish". You can changes the scene of the dialog by the button:"Back","Next", and finishs the scene by the button:"Finish". It has the internal instance of indexed form:WSCindexForm which changes the scene. The property: WSNmenuItems desides the number of the scenes.

Method	Description

1.86 WSCworkingDialog

Derived from WSCbase, WSCwindow, WSCbaseDialog

A dialog with the progressing indicator bar for showing computing process Set a value raging 0-100 in the property: WSNvalue

Method	Description
void setValue(WSCushort value)	Set a value raging from 0 through 100

1.87 WSDappDev

Derived from

Method	Description
WSDappDev* WSGlappDev()	Returns the global instance of the class: WSDappDev.
int getArgc()	Returns the argc of main(int argc, char** argv).
char** getArgv();	Returns the argv of main(int argc, char** argv).
char* getInstanceName()	Returns the project name.
unsigned short getWidth()	Returns the width of the display.
unsigned short getHeight()	Returns the height of the dislpay.

1.88 WSDcolor

Derived from

This is a color class. It holds color names, RBG and pixel values.

Method	Description
char* getColorName()	Obtain a color name
long getRGB(long* r, long* g, long* b)	Obtain RBG values
long setColorName(char* cname)	Set a color name

1.89 WSDdev

Derived from

Class for managing window system dependent resources

Method	Description

void setScalePtr(double* ptr)	Set a pointer to a valiable stored a magnify rate.
void setXOffset(short* ptr)	Set a pointer to a valiable that stored X offset of displaying position X
void setYOffset(short* ptr)	Specify a pointer to a valiable that stored Y offset of displaying position Y
double getScale()	Obtain a magnify rate
short getXOffset()	Display position X offset
short getYOffset()	Display position Y offset
long setForeColor(short cno)	Specify drawing color by the color number
long setBacKColor(short cno)	Specify a background color by the color number
long setLineWidth(short linewidth)	Specify a line girth to draw
long setLineDashType(char no)	Specify dotted line attribute to a drawing line
long setHatchPattern(char no)	Specify daub attribute
long setRegion(short x, short y, unsigned short w, unsigned short h);	Specify a drawing area
long beginDraw(short x, short y, WSCushort w, WSCushort h, WSCbool absolute = False, WSCbool scaling = True)	Start drawing on a specified area
long drawArc(short x, short y, unsigned short w, unsigned short h, short a1, short a2);	Draw circular arc, orval arc
long drawFillArc(short x, short y, unsigned short w, unsigned short h, short a1, short a2, char kind);	Draw a daub fill arc/oval arc
long drawLine(short x1, short y1, short x2, short y2);	Draw a line
long drawLines(WSCpoint* pt, long num);	Draw a polygonal line
long drawRect(short x, short y, unsigned short w, unsigned short h);	Draw a rectagle
long drawFillRect(short x, short y, unsigned short w, unsigned short h);	Draw a daub fill rectagle
long drawRects(WSCrect* pt, long num);	Draw a multiple ractalges
long drawFillRects(WSCrect* pt, long num);	Draw multiple daub fill rectagles
long drawPoly(WSCpoint* pt, long num);	Draw a polygon
long drawFillPoly(WSCpoint* pt, long num);	Draw a daub fill polygon
long drawGradation(long type, short col1, short col2, short col3, short x, short y, WSCushort w, WSCushort h, WSCuchar grad_margin);	Draw a gradation rectagle
long drawImage(short x, short y, WSCushort w, WSCushort h, WSDimage* img, char align);	Display an image
long drawStretchedImage(short x, short y, WSCushort w, WSCushort h, WSDimage* img);	Display a stretched image.
long drawString(short x, short y, WSCushort w, WSCushort h, char font_no, char align, char* string, long encoding = WS_EN_DEFAULT);	Display specified strings within a specified rectagle area
long drawFillString(short x, short y, WSCushort w, WSCushort h, char font_no, char align, char* string, long encoding = WS_EN_DEFAULT);	Display a specified strings with background color within a rectagle area
long getDeviceResource()	Obtain window system resources
long getWindowResource()	Obtain window resources
long getContextResource()	Obtain context resources
long getSpecialResource()	Obtain window resources

long getReady()	Check whether drawable or not

1.90 WSDenv

Derived from

This class contains the environment variables.

Method	Description
WSDenv* WSGIappEnvironment()	Returns the global instance of the class: WSDenv.
char* getEnv(char* str);	Returns the value of the specified environment variable. returns "" if does not exist.
char* getPlaneString(char* str)	Converts the string which contains some environment variables, and returns the plane string converted.

1.91 WSDfont

Derived from

Font class. It manages font name, height, width.

Method	Description
char* getFontName()	Obtain a font name
long getFontHeight()	Obtain font height
long getStringWidth(WSCstring* str)	Obtain width of specified strings
long getStringHeight(WSCstring* str)	Obtain height of specified strings
long getStringWidthUCS2(WSCushort* str)	Obtain width of specified strings
long getStringHeightUCS2(WSCushort* str)	Obtain height of specified strings

1.92 WSDimage

Derived from

This is an Image class. It holds a image file name and image handle.

Method	Description
long destroyImage()	Destruct a read image file
long setImageName(char* iname)	Set an image file name
long getImageWidth()	Obtain an image width
long getImageHeight()	Obtain an image height

1.93 WSDkeyboard

Derived from

This contains the key which is pressed.

Method	Description
WSDkeyboard* WSGlappKeyboard()	Returns the global instance of the class: WSDkeyboard.
char* getText()	Returns the pressed key by the string.
void setText(char*)	Replaces the input string.
long getKey()	Returns the pressed key.
void setKey(long key)	Replaces the key code.
WSCbool withShift()	Examines whether the shift key is pressed at the same time.
WSCbool withLock()	Examines whether the CapsLock key is pressed at the same time.
WSCbool withCntl()	Examines whether the control key is pressed at the same time.
WSCbool withAlt()	Examines whether the alt key is pressed at the same time.
WSCbool isCursorKey()	Examines whether it is the cursor key.
WSCbool isFuncKey()	Examines whether it is the function key.

1.94 WSDmessage

Derived from

Generic message. Shorter data messages can be exchanged between processes.

Method	Description
WSDmessage* getNewInstance()	Create new message class instance
void setupMessage(char* index, void(*proc)(char*, void*), void*)	Set a message ID and a message hander
int sendMessageEx(char* disp, char* index, char* data)	Send a message with a specified message identification string

1.95 WSDmouse

Derived from

Mouse class. It manages the mouse status and the position

Method	Description
WSDmouse* WSGlappMouse();	Obtain a mouse instance existing per system
WSCbool getMousePosition(short* x, short* y, WSCbase* inst);	Obtain mouse position

long getMouseStatus()	Set mouse status
void setMousePosition(short x, short y)	Obtain mouse position
long getTargetBtn()	Obtain a mouse button which most recently pushed.

1.96 WSDmwindowDev

Derived from WSDdev

Memory drawing class. Different from usual window system resource class (WSDdev), it is a virtual resource on memory. On a memory buffer, you can transfer an image data, read image, or operate an image data.

Method	Description
void getGeometry(WSCushort* w, WSCushort* y)	Obtain height and width of the current drawing are
long copyToWindow(WSDdev* dev, short x, short y, WSCushort w, WSCushort y, short dx, short dy)	Copy a specified region image on a specified window system resource instance
long copyFromWindow(WSDdev* dev, short x, short y, WSCushort w, WSCushort y, short dx, short dy)	Copy a specified region image to a specified window system resource instance.
long copyToWindowWithMask(WSDdev* dev, short x, short y, WSCushort w, WSCushort y, short dx, short dy, WSDimage* mask)	Copy a specified region image of specified window system resource instance
long copyToWindowWithMask(WSDdev* dev, short x, short y, WSCushort w, WSCushort y, short dx, short dy, WSDmwindowDev* mask)	Copy a specified region image of a specified window system resource instance
long initBuffer()	Allocate a buffer which can input/output an WSDmindowDev image
long setBufferRGB(WSCushort x, WSCushort y, WSCuchar r, WSCuchar g, WSCuchar b);	Set RGB value to a specified coordinate dot
long getBufferRGB(WSCushort x, WSCushort y, WSCuchar* r, WSCuchar* g, WSCuchar* b);	Obtain RGB value of a specified coordinate dot
long putBufferToPixmap()	Effect the contents of a operated I/O-enabled buffer to a image buffer

1.97 WSDtimer

Derived from

This is a timer which fires at every 250 ms. You can access the global instance of timer by the global function: WSGIappTimer().

Method	Description
WSDtimer* WSGIappTimer()	Returns the global instance of the class: WSDtimer. It is an invarid instance which is created by WSDtimer().

long addTriggerProc(WStimerProc proc, WSCuchar rate, void* data)	Executes once the specified procedure after the specified period.
long addTimerProc(WStimerProc proc, WSCuchar rate, void* data)	Fires the specified procedure at every the specified cycles.
void delTriggerProc(long id)	Unregisters the procedure by the specified id which is returned by addTriggerProc().
void delTimerProc(long id)	Unregisters the procedure by the specified id which is returned by addTimerProc().