RobotBASIC V4.2.0 Commands & Functions By Category

Note: See the Overview Of The Language section for a discussion on how commands and functions fit within the RobotBASIC language. The commands and functions are listed here in order of functionality. An alphabetical order can be found in another section. Commands and functions ***are not*** case sensitive. ClearScr, clearscr, and *clearSCR* are all the same command also sin(), SIN(), and sIn() are all the same function.

Note: In many commands and functions there are ***optional parameters***. If a parameter is not given RobotBASIC will assume a default value for it. If you need to specify a value for an optional parameter that comes after any preceding optional parameters, you have to put a space (or more or none) in place of any optional parameters that precede the one you wish to specify. For example:

```
      //this is how the command to save the screen is specified
      //SaveScrWH {ne_X1{,ne_Y1{,ne_Width{,ne_Height}}}}
      circlewh 10,10,100,100,red,red
      SaveScrWH ,,70   //in this line we have accepted the default
                       //values for the first 2 parameters then specified
                       //a value for the 3rd and again accepted the default
                       //value for the last parameter
      //this is how the function to obtain a substring from a string is given
      //Substring(se_Text{,ne_StartChar{,ne_NumCharacters}})
      s = Substring(ss)       //useless but allowed start from beginning to the
end
      s = Substring(ss, ,5)   //get 5 characters from the beginning
      s = Substring(ss,3)     //get all the rest of the string from the 3rd
character
      s = Substring(ss,7,2)   //get 2 characters from the 7th
```

Note: The following prefixes are used to describe the type of the parameter to be given to a command or function:

ne_	= An expression resulting in a numeric (integer or float).
se_	= An expression resulting in a string.
e_	= An expression resulting in a numeric **or** string.
vs_	= A simple variable that will be set by the command to a string value.
vn_	= A simple variable that will be set by the command to a numeric value.
v_	= A simple variable that will be set by the command to a numeric or string.
a_	= An array variable that will be used by the command or function as a whole array.
Expr	= An expression that **can** result in a numeric or string but no easy description can be given.
ExprN	= An expression that **must** result in a numeric but no easy description can be given.

{Expr} implies that it is optional and {Expr...} means many can be optionally given.
If a simple variable is expected in any of the commands, then if it exists it will be assigned the result otherwise it would be created and assigned the result.

If an array is expected then it must be a previously dimensioned array, but in some cases where the array is created by the command, it does not have to be previously dimensioned.
The character | means or. So if you see on | off, it means you can use either on, or off. v_Name | a_Name[...] means you can specify a simple variable or an array element.

Flow Control Statements

```
IF ExprN THEN statement

If ExprN1
   statement
   statement
   ...
{ElseIf ExprNn}
   statement
   statement
   ...
   .
   .
   .
{Else}
   statement
   statement
   ...
EndIf

For Var = ExprN1 To ExprN2 {Step ExprN3}
  statement
  statement
  ...
Next

Repeat
   Statement
   Statement
   ...
Until ExprN

While ExprN
   Statement
   Statement
   ...
Wend

Break
Continue
Case Construct
```

Gosub *Label*
Gosub *Expr*
Return

Sub SubName({{&}var1{,{&}var2{,...}})
Return {expr}
Call SubName({expr1{expr2{,...}})

OnError *Label*
OnError *Expr*
OnError

OnAbort *Label* | *SubName*
OnAbort *Expr*
OnAbort

On[CONTROL] *Label* | *SubName*
On[CONTROL] *Expr*
On[CONTROL]

On[CONTROL] can be one of 15 words:

OnKey	occurs when a keyboard key is pressed.
OnMouse	occurs when a mouse button is clicked.
OnSerial	occurs when there are bytes in the serial buffer.
OnTCPC	occurs when there are bytes in the TCP ***client*** buffer.
OnTCPS	occurs when there are bytes in the TCP ***server*** buffer.
OnUDP	occurs when there are bytes in ***any*** activated UDP socket buffer.
OnButton	occurs when a push button is clicked.
OnEdit	occurs when the text in the Edit box has changed.
OnListBox	occurs when an item in the list box is selected.
OnRBGroup	occurs when a radio button from the group is clicked.
OnCheckBox	occurs when the check box is checked.
OnSlider	occurs when the slider position is changed.
OnSpinner	occurs when the spinner is clicked.
OnMemo	occurs when the text in the Memo box is changed.
OnTimer	occurs every time the Timer clicks (every period).

End
Terminate
Exit
Goto *Label*

Flow Control Directives

AllowEvents {on|off}
AbortFlag()
AbortMethod {ne_AbortCode}
Stepping {On|Off}
DebugOn
DebugOff
Debug {Expr1,Expr2;Expr3...}
GetError vn_ErrNo{,vs_ErrMessage{,vn_LineNo{,vn_CharNo}}}
#Include "FileName.Ext"{,...}
Undeclare
Declare v_Name {{=}e_InitialValue} {, ...}

RobotBASIC Constants

Color Constants

Black	= 0
Blue	= 1
Green	= 2
Cyan	= 3
Red	= 4
Magenta	= 5
Brown	= 6
Gray	= 7
Darkgray	= 8
Lightblue	= 9
Lightgreen	=10
Lightcyan	=11
Lightred	=12
Lightmagenta	=13
Yellow	=14
White	=15

Note:
See the discussion about colors in the Screen & Bitmap Graphics section.

Variable & Array Element Type Constants

NoType	= 0
Float	=102
Integer	= 105
String	= 115

Serial Port Settings Constants

Parity:

pNone	= 0
pOdd	= 1
pEven	= 2
pMark	= 3
pSpace	= 4

Stop Bits:

sbOneAndHalf	= 0
sbOne	= 1
sbTwo	= 2

FlowControl:

fcNone	= 0
fcXonXoff	= 1
fcHardware	= 2

BaudRate:

br110	= 0
br300	= 1
br600	= 2
br1200	= 3
br2400	= 4
br4800	= 5
br9600	= 6
br14400	= 7
br19200	= 8
br38400	= 9
br56000	=10
br57600	=11
br115200	=12
br128000	=13
br256000	=14

ErrMsg() Dialog Icon Codes And Return Values

Buttons Codes:

MB_Ok	= 0
MB_OkCancel	= 1
MB_AbortRetryIgnore	= 2
MB_YesNoCancel	= 3
MB_YesNo	= 4
MB_RetryCancel	= 5

Icon Codes:

MB_NoIcon	= 0 there will be no icon.
MB_Error	= 16 will cause a **Stop** icon.
MB_Warning	= 32 will cause a ! icon.

MB_Question	= 48	will cause a **?** icon.
MB_Information	= 64	will cause an **i** icon.

Default Button Codes:

MB_Button1	= 0
MB_Button2	= 256
MB_Button3	= 512

Returned Value Codes:

MB_Ok	= 0
MB_Cancel	= 1
MB_Abort	= 2
MB_Retry	= 3
MB_Ignore	= 4
MB_Yes	= 5
MB_No	= 6

Mouse Cursor Constants

cr_Default	= 0	this is the default system shape (usually a pointer the same as cr_Arrow)
cr_None	= -1	this will effectively remove the cursor. ***Beware of this format!***
cr_Arrow	= -2	
cr_Cross	= -3	
cr_IBeam	= -4	**Note**: -5 is not used and will be the same as -1 if you use it
cr_NESW	= -6	
cr_NS	= -7	
cr_NWSE	= -8	
cr_WE	= -9	
cr_UpArrow	= -10	
cr_HourGlass	= -11	
cr_DragPoint	= -12	
cr_NoDrop	= -13	
cr_HSplit	= -14	
cr_VSplit	= -15	
cr_MultiDrag	= -16	
cr_WaitSQL	= -17	
cr_Stop	= -18	
cr_WaitPoint	= -19	
cr_HelpPoint	= -20	
cr_HandPoint	= -21	

Font Style Constants

fs_Bold	= 1
fs_Italic	= 2
fs_Underlined	= 4

AbortMethod Constants

am_Normal	= 0
am_Exit	= 1
am_Error	= 2
am_NoAbort	= 3
am_Flag	= 4

Some Key Code Constants

These codes should be used only with KeyDown() but they can be used with GetKeyE if the user does not press Shift, Ctrl, or Alt along with the key since GetKeyE will append 1000, 4000, or 2000 correspondingly to the actual code.

The codes for 0 ***to*** 9 are 48 ***to*** 57, the codes for A ***to*** Z are 65 ***to*** 90. There are no codes for lower case letters the same key is lower and upper. You can detect if shift is pressed also to distinguish (if you need to), however in GetKeyE the code returned will have 1000 added to the normal code if shift is pressed.

kc_LMouseB	= 1
kc_RMouseB	= 2
kc_MMouseB	= 4
kc_Esc	= 27
kc_F1 ***to*** kc_F12	= 112 ***to*** 123
kc_LArrow	= 37
kc_UArrow	= 38
kc_RArrow	= 39
kc_DArrow	= 40
kc_Shift	= 16
kc_Ctrl	= 17
kc_Alt	= 18
kc_Ins	= 45
kc_Del	= 46
kc_Home	= 36
kc_End	= 35
kc_PUp	= 33
kc_PDn	= 34
kc_Enter	= 13
kc_BkSpace	= 8
kc_Space	= 32

Spawn Modes

P_WAIT	= 0
P_NOWAIT	= 1

File Low Level I/O Constants

fo_BEGIN	= 0
fo_CURRPOS	= 1
fo_END	= 2
fo_READ	= 0
fo_WRITE	= 1
fo_READWRITE	= 2
fo_EXCLUSIVE	= 16

fo_DENYWRITE	= 32
fo_DENYREAD	= 48
fo_DENYNONE	= 64

Conversion Codes Constants

cc_DCTODF	= 0	Celsius to Fahrenheit
cc_DFTODC	= 1	Fahrenheit to Celsius
cc_DTOR	= 2	Degrees to Radians
cc_RTOD	= 3	
cc_NMTOR	= 4	Nautical miles to Radians
cc_RTONM	= 5	
cc_MITONM	= 6	Miles to Nautical miles
cc_NMTOMI	= 7	
cc_KMTONM	= 8	Kilometers to Nautical miles
cc_NMTOKM	= 9	
cc_KMTOMI	= 10	Kilometers to Miles
cc_MITOKM	= 11	
cc_FTTOMI	= 12	Feet to Miles
cc_MITOFT	= 13	
cc_YRDTOMI	= 14	Yards to Miles
cc_MITOYRD	= 15	
cc_FTTOYRD	= 16	Feet to Yards
cc_YRDTOFT	= 17	
cc_INTOFT	= 18	Inches to Feet
cc_FTTOIN	= 19	
cc_YRDTOM	= 20	Yard to Miles
cc_MTOYRD	= 21	
cc_INTOM	= 22	Inches to Meters
cc_MTOIN	= 23	
cc_M2TOHCT	= 24	Meters^2 to Hectares
cc_HCTTOM2	= 25	
cc_M2TOACR	= 26	Meters^2 to Acres
cc_ACRTOM2	= 27	
cc_CM3TOCUP	= 28	Centimeters^3 to Cups
cc_CUPTOCM3	= 29	
cc_CM3TOGLN	= 30	Centimeters^3 to Gallons
cc_GLNTOCM3	= 31	
cc_CM3TOGLNUK	= 32	Centimeters^3 to Gallons UK
cc_GLNUKTOCM3	= 33	
cc_CM3TOLTR	= 34	Centimeters^3 to Liters
cc_LTRTOCM3	= 35	
cc_CM3TOONC	= 36	Centimeters^3 to Ounces
cc_ONCTOCM3	= 37	
cc_CM3TOONCUK	= 38	Centimeters^3 to Ounces UK
cc_ONCUKTOCM3	= 39	
cc_CM3TOPNT	= 40	Centimeters^3 to Pints
cc_PNTTOCM3	= 41	
cc_CM3TOQRT	= 42	Centimeters^3 to Quarts
cc_QRTTOCM3	= 43	
cc_CM3TOTBLSP	= 44	Centimeters^3 to Table Spoon
cc_TBLSPTOCM3	= 45	

cc_CM3TOTSP	= 46	Centimeters^3 to Teaspoon
cc_TSPTOCM3	= 47	
cc_GTOCART	= 48	Grams to Carrats
cc_CARTTOG	= 49	
cc_GTOONC	= 50	Grams to Ounces
cc_ONCTOG	= 51	
cc_GTOPND	= 52	Grams to Pounds
cc_PNDTOG	= 53	
cc_GTOSTN	= 54	Grams to Stones
cc_STNTOG	= 55	
cc_GTOTON	= 56	Grams to Tones
cc_TONTOG	= 57	
cc_GTOTONL	= 58	Grams to Long Tones
cc_TONLTOG	= 59	
cc_WATTTOHP	= 60	Watts to Horse Power
cc_HPTOWATT	= 61	

Regular Expression Mode Codes

RE_DOTMATCHNEWLINE	= 1
RE_MULTILINE	= 2
RE_IGNORECASE	= 8
RE_RIGHTTOLEFT	= 16
RE_IGNOREWHITESPACE	= 32

Numeric Type Byte Count

BytesCount_I	= 4
BytesCount_F	= 8

Media Player State Codes

ms_NotReady	= 0
ms_Stopped	= 1
ms_Playing	= 2
ms_Recording	= 3
ms_Paused	= 5

DFT/FFT Window Functions Codes

sw_NoWindow	= 0
sw_Hamming	= 1
sw_VonHann	= 2
sw_Blackman	= 3
sw_BHarris	= 4
sw_Bartlet	= 5

Other Constants

True	= 1
False	= 0
On	= 1
Off	= 0
Yes	= 1
No	= 0
Down	= 1
Up	= 0

Standard User Interfacing

Standard Output

Print {Expr,Expr;Expr...}{;|,}
Write {Expr,Expr;Expr...}{;|,}
CommaTab {true|false}
HonorCrLf {true|false}
SetPromptArea {se_Text}
SetInputArea {se_Text}
xyString ne_X,ne_Y,Expr{;expr,expr;...}
xyText {ne_X{,ne_Y{,e_Text{,se_FontName{,ne_FontSize{,ne_FontStyle{,ne_PenColor{,ne_BackgroundColor}}}}}}}}
OverlayText {ne_X{,ne_Y{,e_Text{,se_FontName{,ne_FontSize{,ne_FontStyle{,ne_PenColor{,ne_Quick}}}}}

Standard Input

InlineInputMode {on|off}
Input {e_Prompt,} v_Name | a_Name[...] {,...}
xyInput v_Input{,ne_X{,ne_Y{,e_Title{,e_Default{,ne_BoxLength}}}}}

Keyboard Input

WaitKey {e_Prompt,}vn_KeyCode
GetKey vn_KeyCode
GetKeyE vn_ScanCode
WaitNoKey {ne_MillisWait}
WaitNoKeyE ne_ScanCode{,ne_MillisWait}
LastKey()
KeyDown({ne_ScanCode})

Mouse Input

ReadMouse vn_X,vn_Y{,vn_Buttons}
SetMousePos {ne_X{,ne_Y}}
SetCursor {ne_CursorShapeCode}
GetCursor vn_Code
LastMouse()

Joystick Input

Joystick ne_JoystickNo,vn_XAxisPos,vn_YAxisPos,vn_ThrottlePos,vn_Buttons
JoystickE ne_JoystickNo,a_ReturnedData

Sound Output & Input

Beep {ne_Count}
PlayWav {se_FileName{,ne_Mode{,ne_Loop}}}
WavBusy()
MediaPlay ne_DeviceNumber,se_FileName{,ne_Loop}
MediaPause ne_DeviceNumber{,on|off}
MediaStop ne_DeviceNumber
MediaRecord ne_DeviceNumber,se_FileName
MediaSave ne_DeviceNumber
MediaShow ne_DeviceNumber{,true|false}
MediaVideoSize DeviceNumber{,vn_Width{,vn_Height}}
MediaReposition {ne_X{,ne_Y{,ne_Width{,ne_Height}}}}
MediaGetPosition {vn_X{,vn_Y{,vn_Width{,vn_Height}}}}
MediaState(ne_DeviceNumber)
MediaIsVideo(ne_DeviceNumber)
Sound ne_Frequency,ne_Duration{,ne_Mode}
Speaker {on|off}
PlaySong {se_Notes}

Graphical User Interfacing

Dialog Boxes

StrInput({e_Caption{,e_Prompt{,e_Default}}})
TextBox(se_FileName{,e_Title{,ne_X{,ne_Y{,ne_W{,ne_H{,ne_DoWrap}}}}}})
StringBox(se_Text{,e_Title{,ne_X{,ne_Y{,ne_W{,ne_H{,ne_DoWrap}}}}}})
ErrMsg(se_MessageText{,se_BoxTitle{,ne_Style}})
MsgBox(a_TextLines{,e_Title{,ne_X{,ne_Y{,ne_W{,ne_H{,ne_DoWrap}}}}}})

Push Button Components

Push Button Commands

AddButton se_Name,ne_X,ne_Y{,ne_W{,ne_H{,se_Hint}}}
RemoveButton se_Name
FocusButton se_Name
EnableButton se_Name{,true|false}
HideButton se_Name{,true|false}
SetButtonCaption se_Name{,e_Caption}
SetButtonDim se_Name{,ne_X{,ne_Y{,ne_W{,ne_H}}}}
SetButtonFont se_Name{,se_FontType{,ne_FontSize{,ne_FontStyle{,ne_FontColor}}}}
RenameButton se_CurrentName,se_NewName
GetButton vs_Name

Push Button Functions

ButtonEnabled(se_Name)
ButtonHidden(se_Name)
GetButtonCaption(se_Name)
GetButtonX(se_Name)
GetButtonY(se_Name)
GetButtonW(se_Name)
GetButtonH(se_Name)
GetButtonFont(se_Name)
ButtonHasFocus(se_Name)
LastButton()

Edit Box Components

Edit Box Commands

AddEdit se_Name,ne_X,ne_Y{,ne_W{,ne_H{,e_Text{,se_Hint}}}}
RemoveEdit se_Name
FocusEdit se_Name
EnableEdit se_Name{,true|false}
HideEdit se_Name{,true|false}
ReadOnlyEdit se_Name{,true|false}
SetEdit se_Name{,e_Value}
SetEditColor se_Name{,ne_Color}
BorderEdit se_Name{,true|false}
SetEditDim se_Name{,ne_X{,ne_Y{,ne_W{,ne_H}}}}
SetEditFont se_Name{,se_FontType{,ne_FontSize{,ne_FontStyle{,ne_FontColor}}}}
IntegerEdit se_Name{,true|false}
FloatEdit se_Name{,true|false}
SetEditMask se_Name,se_MaskSpecs

Edit Box Functions

EditEnabled(se_Name)
EditHidden(se_Name)
EditReadOnly(se_Name)
EditHasFocus(se_Name)
EditChanged(se_Name)
GetEdit(se_Name)
GetEditUnmasked(se_Name)
GetEditColor(se_Name)
EditBorder(se_Name)
GetEditX(se_Name)
GetEditY(se_Name)
GetEditW(se_Name)
GetEditH(se_Name)
GetEditFont(se_Name)
LastEdit()

List Box Components

List Box Commands

AddListBox se_Name,ne_X,ne_Y{,ne_Width{,se_Items{,se_Hint}}}
RemoveListBox se_Name
FocusListBox se_Name
ClearListBox se_Name
EnableListBox se_Name{,true|false}
HideListBox se_Name{,true|false}
SortListBox se_Name{,true|false}
SetListBox se_Name{,ne_Index}
SetListBoxColor se_Name{,ne_Color}
SetListBoxDim se_Name{,ne_X{,ne_Y{,ne_W}}}
DeleteListBoxItem se_Name{,ne_Index}
AddListBoxItem se_Name{,e_NewItem}
SetListBoxItems se_Name{,e_ItemsList}
SetListBoxFont se_Name{,se_FontType{,ne_FontSize{,ne_FontStyle{,ne_FontColor}}}}

List Box Functions

ListBoxEnabled(se_Name)
ListBoxHidden(se_Name)
ListBoxHasFocus(se_Name)
ListBoxSorted(se_Name)
ListBoxItemsCount(se_Name)
GetListBox(se_Name)
GetListBoxText(se_Name)
GetListBoxItem(se_Name,ne_ItemIndex)
GetListBoxList(se_Name)
GetListBoxColor(se_Name)
GetListBoxX(se_Name)
GetListBoxY(se_Name)
GetListBoxW(se_Name)
GetListBoxFont(se_Name)
LastListBox()

Radio Button Components

Radio Button Commands

AddRBGroup se_Name,ne_X,ne_Y{,ne_Width{,ne_Height{,ne_Columns{,se_Buttons{,se_Caption{,se_Hint}}}}}}
RemoveRBGroup se_Name
FocusRBGroup se_Name
ClearRBGroup se_Name
EnableRBGroup se_Name{,true|false}
HideRBGroup se_Name{,true|false}
SetRBGroup se_Name{,ne_Index}
SetRBGroupColor se_Name{,ne_Color}
SetRBGroupColumns se_Name{,ne_NumColumns}
SetRBGroupDim se_Name{,ne_X{,ne_Y{,ne_W{,ne_H}}}}
AddRBGroupButton se_Name{,e_ButtonCaption}
DeleteRBGroupButton se_Name{,ne_Index}

SetRBGroupButtons se_Name{,e_ButtonsList}
SetRBGroupFont se_Name{,se_FontType{,ne_FontSize{,ne_FontStyle{,ne_FontColor}}}}

Radio Button Functions

RBGroupEnabled(se_Name)
RBGroupHidden(se_Name)
RBGroupHasFocus(se_Name)
GetRBGroupCaption(se_Name)
GetRBGroup(se_Name)
GetRBGroupText(se_Name)
RBGroupItemsCount(se_Name)
RBGroupNumColumns(se_Name)
GetRBGroupX(se_Name)
GetRBGroupY(se_Name)
GetRBGroupW(se_Name)
GetRBGroupH(se_Name)
GetRBGroupButton(se_Name,ne_Index)
GetRBGroupItems(se_Name)
GetRBGroupColor(se_Name)
GetRBGroupFont(se_Name)
LastRBGroup()

Check Box Components

Check Box Commands

AddCheckBox se_Name,ne_X,ne_Y{,se_Caption{,ne_Checked{,ne_LeftOrRight{,se_Hint}}}}
RemoveCheckBox se_Name
FocusCheckBox se_Name
EnableCheckBox se_Name{,true|false}
HideCheckBox se_Name{,true|false}
SetCheckBox se_Name{,true|false}
SetCheckBoxDim se_Name{,ne_X{,ne_Y}}
SetCheckBoxCaption se_Name{,e_NewCaption}
SetCheckBoxColor se_Name{,ne_Color}

Check Box Functions

CheckBoxEnabled(se_Name)
CheckBoxHidden(se_Name)
CheckBoxHasFocus(se_Name)
GetCheckBox(se_Name)
GetCheckBoxCaption(se_Name)
GetCheckBoxColor(se_Name)
GetCheckBoxX(se_Name)
GetCheckBoxY(se_Name)
LastCheckBox()

Slider/Progress Bar Components

Slider/Progress Bar Commands

AddSlider se_Name,ne_X,ne_Y{,ne_Width{,ne_Min{,ne_Max{,ne_Vertical{,ne_TickFreq{,ne_BigIncr{,se_Hint}}}}}}}
RemoveSlider se_Name
FocusSlider se_Name
EnableSlider se_Name{,true|false}
HideSlider se_Name{,true|false}
SetSliderPos se_Name{,ne_PositionValue}
SetSliderMin se_Name{,ne_NewValue}
SetSliderMax se_Name{,ne_NewValue}
SetSliderBarStart se_Name{,ne_Value}
SetSliderBarEnd se_Name{,ne_Value}
HideSliderDial se_Name{,true|false}
ShowSliderBar se_Name{,true|false}
SetSliderDim se_Name{,ne_X{,ne_Y{,ne_W}}}

Slider/Progress Bar Functions

SliderEnabled(se_Name)
SliderHidden(se_Name)
GetSliderPos(se_Name)
GetSliderMin(se_Name)
GetSliderMax(se_Name)
GetSliderBarStart(se_Name)
GetSliderBarEnd(se_Name)
SliderDialHidden(se_Name)
SliderBarHidden(se_Name)
GetSliderX(se_Name)
GetSliderY(se_Name)
GetSliderW(se_Name)
SliderHasFocus(se_Name)
LastSlider()

Spinner Components

Spinner Commands

AddSpinner se_Name,ne_X,ne_Y{,ne_W{,ne_H{,ne_Min{,ne_Max{,ne_Incr{,ne_Vertical{,ne_Wrap{,se_Hint}}}}}}}}
RemoveSpinner se_Name
FocusSpinner se_Name
EnableSpinner se_Name{,true|false}
HideSpinner se_Name{,true|false}
SetSpinner se_Name{,ne_Position}
SetSpinnerMin se_Name{,ne_NewValue}
SetSpinnerMax se_Name{,ne_NewValue}
SetSpinnerIncr se_Name{,ne_Value}
SetSpinnerWrap se_Name{,true|false}
SetSpinnerDim se_Name{,ne_X{,ne_Y{,ne_W{,ne_H}}}}

Spinner Functions

SpinnerEnabled(se_Name)
SpinnerHidden(se_Name)
SpinnerHasFocus(se_Name)
GetSpinner(se_Name)
GetSpinnerMin(se_Name)
GetSpinnerMax(se_Name)
GetSpinnerIncr(se_Name)
GetSpinnerWrap(se_Name)
GetSpinnerX(se_Name)
GetSpinnerY(se_Name)
GetSpinnerW(se_Name)
GetSpinnerH(se_Name)
LastSpinner()

Memo Box Components

Memo Box Commands

AddMemo se_Name,ne_X,ne_Y{,ne_W{,ne_H{,se_Text{,se_Hint}}}}
RemoveMemo se_Name
FocusMemo se_Name
ClearMemo se_Name
EnableMemo se_Name{,true|false}
HideMemo se_Name{,true|false}
ReadOnlyMemo se_Name{,true|false}
WrapMemo se_Name{,true|false}
BorderMemo se_Name{,true|false}
SetMemoColor se_Name{,ne_Color}
SetMemoDim se_Name{,ne_X{,ne_Y{,ne_W{,ne_H}}}}
DeleteMemoLine se_Name{,ne_LineNumber}
SetMemoScrollBars se_Name{,ne_Value}
AddMemoLine se_Name{,e_Text}
SetMemoText se_Name{,e_Text}
SetMemoFont se_Name{,se_FontType{,ne_FontSize{,ne_FontStyle{,ne_FontColor}}}}
SetMemoSelection se_Name{,ne_LineNumber{,ne_CharacterNumber{,ne_SelectionLength}}}}
SetMemoSelected se_Name{,ne_StartCharPosition{,ne_NumCharacters}}

Memo Box Functions

MemoChanged(se_Name)
MemoEnabled(se_Name)
MemoHidden(se_Name)
MemoWrap(se_Name)
MemoReadonly(se_Name)
MemoBorder(se_Name)
MemoScrollBars(se_Name)
GetMemoColor(se_Name)
GetMemoX(se_Name)
GetMemoY(se_Name)
GetMemoW(se_Name)
GetMemoH(se_Name)

MemoLinesCount(se_Name)
GetMemoText(se_Name)
GetMemoSelection(se_Name)
GetMemoLineNo(se_Name)
GetMemoCharNo(se_Name)
GetMemoCharPos(se_Name)
GetMemoLine(se_Name,ne_LineNumber)
GetMemoFont(se_Name)
MemoHasFocus(se_Name)
LastMemo()

Timer Components

Timer Commands

AddTimer se_Name{,ne_Period}
RemoveTimer se_Name
SetTimer se_Name{,true|false}
SetTimerPeriod se_Name{,ne_Period}
SetTimerTicks se_Name{,ne_Count}

Timer Functions

TimerIsOn(se_Name)
GetTimerPeriod(se_Name)
GetTimerTicks(se_Name)
LastTimer()

Screen & Bitmap Graphics

Color Manipulation

RGB(ne_RedValue,ne_GreenValue,ne_BlueValue)
ConsToClr(ne_ColorConstantValue)
PromptColor({ne_DefaultColor})
RedValue(ne_Color)
GreenValue(ne_Color)
BlueValue(ne_Color)
FactorColor(ne_Color,ne_Factor)
mCombineClr a_RedValues,a_GreenValues,a_BlueValues,a_RGBvalues

Printing The Screen Graphics

PrintScr
PrinterSetup

Screen Output Buffer Control

SetTextBuff {se_Text}
TextBuffToCB
GetTextBuff()

Screen Manipulation

ScrSetMetrics {ne_X{,ne_Y{,ne_Width{,ne_Height{,ne_PanelVisible{,ne_AllowResize}}}}}}

ScrGetMetrics {vn_X{,vn_Y{,vn_Width{,vn_Height{,vn_PanelVisible{,vn_AllowResize}}}}}}

Flip {on|off}

ClearScr {ne_Color}

ScrLimits vn_XLimit,vn_YLimit

SaveScrWH {ne_X1{,ne_Y1{,ne_Width{,ne_Height}}}}

SaveScr {ne_X1{,ne_Y1{,ne_X2{,ne_Y2}}}}

RestoreScr {ne_X{,ne_Y}}

CopyScr {ne_CopyNumber{,ne_X1{,ne_Y1{,ne_Width{,ne_Height}}}}

CopyToScr {ne_CopyNumber{,ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_MapX{,ne_MapY}}}}}}}

CopyFitScr {ne_CopyNumber{,ne_X{,ne_Y{,ne_Width{,ne_Height}}}}}

WriteScr {se_FileName}

ReadScr {se_FileName}

mScrToArray a_Pixels{,ne_X{,ne_Y{,ne_Width{,ne_Height}}}}

mScrFromArray a_Pixels{,ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_ArrayX{,ne_ArrayY}}}}}}

mScrFitArray a_Pixels{,ne_X{,ne_Y{,ne_Width{,ne_Height}}}}}

DeskTopWidth()

DeskTopHeight()

Bitmap Manipulation

Transparent {on|off}

PromptBMP({se_Filter})

ReadBMP {se_FileName{,ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_MapX{,ne_MapY}}}}}}}

WriteBMP {se_FileName{,ne_X{,ne_Y{,ne_Width{,ne_Height}}}}}

mReadBMP a_Pixels{,se_FileName{,ne_ClrCode}}

mWriteBMP a_Pixels{,se_FileName}

RotateBMP{se_FileName{,ne_Angle{,ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_MapX{,ne_MapY}}}}}}}}

FlipBMP {se_FileName{,ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_MapX{,ne__MapY}}}}}}}

MirrorBMP {se_FileName{,ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_MapX{,ne__MapY}}}}}}}

FitBMP {se_FileName{,ne_X{,ne_Y{,ne_Width{,ne_Height}}}}}

SizeBMP se_FileName,vn_Width,vn_Height

ResizeBMP {se_SourceFileName{,ne_Width{,ne_Height{,se_ToFileName}}}}

ToBMP se_SourceFile{,se_ToFile}

BmpToGray se_SourceFileName{,se_ToFileName{,ne_RedRatio{,ne_GreenRatio{,ne_BlueRatio}}}}

BmpNegative se_SourceFile{,se_ToFile}

BmpToBW se_SourceFile{,ne_Threshold{,se_ToFile}}

BmpEdges se_SourceFile{,ne_Threshold{,se_ToFile{,ne_EdgeType}}}

BmpRGB se_SourceFile,ne_Rratio,ne_Gratio,ne_Bratio{,se_ToFile}

BmpContrast se_SourceFile,ne_Ratio{,ne_Threshold{,se_ToFile}}

BmpCompare se_SourceFile,se_CompareFile{,se_ToFile{,ne_Tolerance}}

BmpChangeClr se_SourceFile,ne_FromColor,ne_ToColor{,ne_Tolerance}

BmpFindClr se_SourceFile,ne_Color,vn_Result{,vn_Confidence{,ne_ClrTolerance{,ne_ConfidenceTolerance{,ne_GridSize{a_SectorsCount}}}}}

BmpStats a_Stats{,se_FileName}

Clipboard Manipulation

SetCBText {se_Text}

GetCBText()

ClrCB

SizeCb vn_Width,vn_Height

ScrToCb {ne_X{,ne_Y{,ne_Width{,ne_Height}}}}
ScrFromCb {ne_ScreenX{,ne_ScreenY{,ne_Width{,ne_Height{,ne_MapX{,ne_MapY}}}}}}
FitCb {ne_X{,ne_Y{,ne_Width{,ne_Height}}}}
BmpToCb {se_FileName{,ne_X{,ne_Y{,ne_Width{,ne_Height}}}}}
CbToBMP {se_FileName}
CbFitBMP {se_FileName{,ne_Width{,ne_Height}}
FlipCb
MirrorCb
RotateCb {ne_Angle}

TWAIN Image Capture

CaptureRdy()
CaptureSrc()
CaptureDlg({se_FileName})
CaptureImage({se_FileName})

2D Screen Graphics Drawing

PixelClr(ne_X,ne_Y)
GetXY vn_X,vn_Y
GotoXY {ne_X{,ne_Y}}
SetColor {ne_PenColor{,ne_BackGroundColor}
GetColor vn_PenColor,vn_BkgrndColor
ReadPixel ne_X,ne_Y,vn_Color
SetPixel {ne_X{,ne_Y{,ne_Color}}}
LineWidth {ne_Width}
GetLineWidth vn_Width
LineTo ne_X,ne_Y{,ne_PenWidth{,ne_PenColor}}
Line ne_X1,ne_Y1,ne_X2,ne_Y2{,ne_PenWidth{,ne_PenColor}}
Rectangle ne_X1,ne_Y1,ne_X2,ne_Y2{,ne_PenColor{,ne_FillColor}}
RectangleWH ne_X,ne_Y,ne_Width,ne_Height{,ne_PenColor{,ne_FillColor}}
eRectangle ne_X1,ne_Y1,ne_X2,ne_Y2{,ne_PenWidth{,ne_PenColor}}
eRectangleWH ne_X1,ne_Y1,ne_Width,ne_Height{,ne_PenWidth{,ne_PenColor}}
Circle ne_X1,ne_Y1,ne_X2,ne_Y2{,ne_PenColor{,ne_FillColor}}
CircleWH ne_X1,ne_Y1,ne_Width,ne_Height{,ne_PenColor{,ne_FillColor}}
Arc ne_X1,ne_Y1,ne_X2,ne_Y2{,ne_StartAngle{,ne_ArcLength{,ne_PenWidth{,ne_PenColor}}}}
Pie ne_X1,ne_Y1,ne_X2,ne_Y2{,ne_StartAngle{,ne_ArcLength{,ne_PenColor{,ne_FillColor}}}}
mPolygon a_Vertices{,ne_FillColor}
mBezier a_Vertices{,ne_PenWidth{,ne_PenColor}}
mGraphPaper a_Specs
mPlotXY a_Specs,a_Xvalues,a_Yvalues
mPlotXY a_Specs,a_XYvalues
FloodFill {ne_X{,ne_Y{,ne_NewColor{,ne_OldColor}}}}
FloodFill2 {ne_X{,ne_Y{,ne_NewColor{,ne_BorderColor}}}}
DrawShape se_Shape,ne_X,ne_Y{,ne_Scale,ne_Color}
RotShape(se_ShapeString,ne_Direction)
TextWidth(se_Text{,se_FontName{,ne_FontSize{,ne_FonctStyle}}})
TextHeight(se_Text{,se_FontName{,ne_FontSize{,ne_FonctStyle}}})

3D Screen Graphics Drawing

ge3Dto2DV ne_X,ne_Y,ne_Z,ne_Rho,ne_Theta,ne_Phi,ne_Dist,ne_CenterX,ne_CenterY,vn_ScrX,vn_ScrY
ge3Dto2DVA ne_X,ne_Y,ne_Z,a_CameraSpecs,vn_ScrX,vn_ScrY
ge3Dto2DA a_3DPoints,a_CameraSpecs
geVisibles a_3DPoints,a_CameraSpecs,a_SurfacesSpecs{,a_Edges{,ne_ColorFactor}}
gePlotSurfaces a_3DPoints,a_SurfacesSpecs{,ne_DoFilling{,ne_LineWidth{,ne_OnlyVisible{,ne_CentroidAll}}}}
geCentroids a_3DPoints,a_SurfacesSpecs,a_Centroids
gePlotEdges a_3DPoints,a_Edges{,ne_LineWidth{,ne_LineWidth{,ne_Color}}
geRotVx ne_X,ne_Y,ne_Z,ne_RotAngle,vn_X',vn_Y',vn_Z'
geRotVy ne_X,ne_Y,ne_Z,ne_RotAngle,vn_X',vn_Y',vn_Z'
geRotVz ne_X,ne_Y,ne_Z,ne_RotAngle,vn_X',vn_Y',vn_Z'
geRotateA a_3DPoints,ne_RotAngle,ne_AxisCode{,ne_From{,ne_To}}

Creating & Using Arrays

Arrays Creation & Manipulation

Dim a_Name1[ExprN{,ExprN...}]{ , a_Name2[ExprN{,ExprN...}] {, ...}}}...
Data a_Name;Expr{,Expr....}
mDim(a_Name)
MaxDim(a_Name{,ne_Dimension})
mType(a_Name[...])
mCopy a_Source,a_Destination
mWrite a_Name,se_FileName
mRead a_Name,se_FileName
mTextFW a_Name,se_TextFileName
mTextFR a_Name,se_TextFileName
mFromString a_Name,se_String{,se_Separator}
mToString(a_TextLines{,se_Separator})
mToCommaText(a_TextLines)
ObjectGet a_ObjectArray,ne_ObjectNumber
RecordGet a_DataBaseArray,ne_RecordNumber
ObjectPut a_ObjectArray,ne_ObjectNumber
RecordPut a_DataBaseArray,ne_RecordNumber

Array Math Commands

mAND a_Name,ExprN
mOR a_Name,ExprN
mXOR a_Name,ExprN
mShiftL a_Name,ExprN
mShiftR a_Name,ExprN
mNOT a_Name
mScale a_Name,ExprN
mConstant a_Name,Expr
mDiagonal a_Name,Expr
mmAND a_Source,a_Destination
mmOR a_Source,a_Destination
mmXOR a_Source,a_Destination
mmShiftL a_Source,a_Destination

mmShiftR a_Source,a_Destination
mAdd a_Source,a_Destination
mSub a_Source,a_Destination
mMultiply a_Left,a_Right,a_Result
mInvert a_Source,a_Inverse,vn_Determinant
mDet a_Source,vn_Determinant
mTranspose a_Source,a_Transpose
mRegression a_XYdata,vn_Slope,vn_Intercept
mExpFit a_XYdata,vn_Exponent,vn_Factor
mLogFit a_XYdata,vn_Factor,vn_Translation
mPolyFit a_XYdata,a_Coefficients
mSortR a_Name{,ne_OnRowNumber{,ne_Descending}}
mSortC a_Name{,ne_OnColumnNumber{,ne_Descending}}
mDFT a_Samples{,ne_WindowFunction}
mFFT a_Samples{,ne_WindowFunction}

Arrays Statistical Functions

Sum(a_Data)
mSum(a_Data)
Average(a_Data)
mAverage(a_Data)
Median(a_Data)
Max(a_Data)
mMax(a_Data)
Min(a_Data)
mMin(a_Data)
Range(a_Data)
mRange(a_Data)
Count(a_Data)
mCount(a_Data)
Variance(a_Data)
mVariance(a_Data)
StdDev(a_Data)
mStdDev(a_Data)
CorrCoef(a_Data)

Mathematical Functions

Trigonometric Functions

Pi({ne_Multiplier})
RtoD(ne_Radians)
DtoR(ne_Degrees)
Sin(ne_Radians)
Cos(ne_Radians)
Tan(ne_Radians)
ASin(ExprN)
ACos(ExprN)
ATan(ExprN)
ATan2(ne_X,ne_Y)

Cartesian To Polar Functions

PolarR(ne_X,ne_Y)
PolarA(ne_X,ne_Y)

Polar To Cartesian Functions

CartX(ne_Radius,ne_ThetaRadians)
CartY(ne_Radius,ne_ThetaRadians)

Logarithmic & Exponential Functions

NLog(ExprN)
Log(ExprN)
Log2(ExprN)
LogB(ne_Base,ExprN)
Exp(ExprN)
Exp10(ExprN)
SqRt(ExprN)
CbRt(ExprN)

Hyperbolic Functions

SinH(ExprN)
CosH(ExprN)
TanH(ExprN)
ASinH(ExprN)
ACosH(ExprN)
ATanH(ExprN)

Probability Functions

Random(ExprN)
RandomG(ne_Mean,ne_StdDev)
Factorial(ExprN)
nPr(ne_NumElementsAvailable,ne_NumElementsToSelect)
nCr(ne_NumElementsAvailable,ne_NumElementsToSelect)
ProbG(ne_Element,ne_Mean,ne_StdDev)
ProbGI(ne_Probability,ne_Mean,ne_StdDev)
SeedRandom {ne_Seed}

Financial Functions

ff_FV(PMT,INTR,TERM,TYPE)
ff_FVT(PMT,INTR,FV,TYPE)
ff_FVP(FV,INTR,TERM,TYPE)
ff_PV(PMT,INTR,TERM,BAL,TYPE)
ff_PVT(PMT,INTR,PV,BAL,TYPE)
ff_PVP(PV,INTR,TERM,BAL,TYPE)
ff_CIFV(PV,INTR,TERM)
ff_CIT(PV,INTR,FV)
ff_CII(PV,FV,TERM)
ff_SLN(COST,SALVAGE,LIFE)
ff_SYD(COST,SALVAGE,LIFE,PERIOD)

Great Circle Navigation

Great Circle and Rhumb Line Navigation

ngc_DistanceHeading ne_LatA,ne_LonA,ne_LatB,ne_LonB,vn_Distance,vn_Heading
nrl_DistanceHeading ne_LatA,ne_LonA,ne_LatB,ne_LonB,vn_Distance,vn_Heading
ngc_RadialPoint ne_LatA,ne_LonA,ne_Distance,ne_Heading,vn_Lat,vn_Lon
nrl_RadialPoint ne_LatA,ne_LonA,ne_Distance,ne_Heading,vn_Lat,vn_Lon
ngc_LatFromLonCrossing ne_LatA,ne_LonA,ne_LatB,ne_LonB,ne_Lon,vn_Lat
ngc_LonFromLatCrossing ne_LatA,ne_LonA,ne_LatB,ne_LonB,ne_Lat,vn_Lon1,vn_Lon2
ngc_FractionDistancePoint ne_LatA,ne_LonA,ne_LatB,ne_LonB,ne_FractionalDistance,vn_Lat,vn_Lon
ngc_RadialIntersection ne_LatA,ne_LonA,ne_LatB,ne_LonB,ne_HeadingFromA,ne_HeadingFromB,vn_Lat,vn_Lon
ngc_XTrackError ne_LatA,ne_LonA,ne_LatB,ne_LonB,ne_LatD,ne_LonD,vn_XTrackDistance,vn_AlongTrackDistance
ngc_TrackPointsFromPoint ne_LatA,ne_LonA,ne_LatB,ne_LonB,ne_LatD,ne_LonD,ne_DistanceFromPoint,vn_Lat1, vn_Lon1, vn_Lat2, vn_Lon2

Wind or Current Triangle Navigation

nwt_XWind ne_WindSpeed,ne_WindDirection,ne_RunwayHeading,vn_XWind,vn_HeadWind
nwt_GSpeedHeading ne_WindSpeed,ne_WindDirection,ne_TrueSpeed,ne_CourseHeading,vn_GrndSpeed,vn_Heading
nwt_WSpeedDirection ne_GrndSpeed,ne_CourseHeading,ne_TrueSpeed,ne_Heading,vn_WindSpeed,vn_WindDirection
nwt_GSpeedCourse ne_WindSpeed,ne_WindDirection,ne_TrueSpeed,ne_Heading,vn_GroundSpeed,vn_CourseHeading
nwt_TSpeedHeading ne_WindSpeed,ne_WindDirection,ne_GroundSpeed,ne_CourseHeading,vn_TrueSpeed,vn_Heading
nwt_TSpeedWSpeed ne_V1,ne_V2,ne_V3,vn_TrueAirSpeed,vn_WindSpeed

Navigation Conversion Functions

Degrees({ne_Degrees{,ne_Minutes{,ne_Seconds}}})
Degrees(se_FormattedDegrees)
Lat_DMS(ne_Degrees)
Lat_DM(ne_Degrees)
Lon_DM(ne_Degrees)

String Functions

String Manipulation Functions

CrLf()
Length(se_Text)
Trim(se_Text)
LeftTrim(se_Text)
RightTrim(se_Text)
NoSpaces(se_Text)
Substring(se_Text{,ne_StartChar{,ne_NumCharacters}})
Left(se_Text,ne_NumChars)
Right(se_Text,ne_NumChars)
InString(se_Main,se_Sub{,ne_StartFrom})
Contains(se_Text,se_CharList)
NotContains(se_Text,se_CharList)
Upper(se_Text)

Lower(se_Text)
Proper(se_Text)
Spaces(ne_NumOfSpaces)
sRepeat(se_RepeatChars,ne_NumTimes)
Center(se_Text,se_PadChar,ne_NumChars)
JustifyL(se_Text,se_PadChar,ne_Len)
JustifyR(se_Text,se_PadChar,ne_Len)
Insert(se_Text,se_Insert,ne_CharNum)
Replace(se_OriginalString,se_NewSubString,ne_StartingAt)
Substitute(se_Text,se_TextToReplace,se_ReplaceWith)
ToCommaText(se_Text)
Extract(se_Text,se_SeparatorChars,ne_Part)
NumParts(se_Text,se_Separator)
Encrypt(se_Text,se_Key)
Soundex(se_Text{,ne_Length})

Regular Expression String Functions

re_Setup(se_Template{,ne_Mode})
re_Match(se_Text)
re_Search(se_Text{,ne_SartFromPositionNumber})
re_Start({ne_GroupNumber})
re_End({ne_GroupNumber})
re_GrpNumber(se_GroupName)
re_NumOfGrps()
re_Replace(se_Text{,se_ReplaceWith{,ne_SartFromPositionNumber{,ne_NumberOfMatchesToReplace}}})

Conversion Functions

Scaling And Weight & Measure Conversion

Convert(ne_ValueToConvert, ne_ConversionTypeCode)
Map(ne_FromValue, ne_FromMin, ne_FromMax, ne_ToMin, ne_ToMax)

Sign Conversion

Abs(ExprN)
Sign(ExprN)

Float & Integer Conversion

Round(ExprN)
RoundUP(ExprN{,ne_Type})
RoundDn(ExprN{,ne_Type})
SignExtend8(ExprN)
SignExtend16(ExprN)
Frac(ExprN)
Mod(ne_Numerator,ne_Denominator)
MaxInteger()
MinInteger()
MaxFloat()
MinFloat()

MaxV(ExprN1,ExprN2)
MinV(ExprN1,ExprN2)
Within(ne_Value,ne_LowerLimit,ne_UpperLimit)
Limit(ne_Value,ne_LowerLimit,ne_UpperLimit)
BitSwap(ne_Number{,ne_NumberOfBits})
MakeBit(ne_Number,ne_BitPosition,on|off)
MakeByte(ne_Number,ne_BytePosition,ne_ByteValue)
SetBit(ne_Number,ne_BitPosition)
ClrBit(ne_Number,ne_BitPosition)
SetByte(ne_Number,ne_BytePosition)
ClrByte(ne_Number,ne_BytePosition)
GetBit(ne_Number,ne_BitPosition)
GetByte(ne_Number,ne_BytePosition)

Number & String Conversion

Hex(ExprN{,ne_NumBytes})
HexToInt(e_HexValue)
Bin(ExprN{,ne_NumBits})
BinToInt(e_BinaryValue)
SFtoDF(ExprN)
DFtoSF(ExprN)
ToByte(Expr)
Spell(ExprN)
IsNumber(Expr)
IsString(Expr)
ToNumber(Expr{,ne_Default})
ToString(Expr)
Char(ne_AsciiCode)
Ascii(se_Text)
GetStrByte(se_String,ne_ByteNumber)
PutStrByte(se_String,ne_ByteNum,ne_Val)
StrOfBytes(Expr{,Expr{,....}})
Format(ExprN,se_FormatSpecifier)

File & Directory Functions

Directory Functions

DiskSize(ne_DiskNumber)
DiskFree(ne_DiskNumber)
DirCurrent()
DirSet(se_DirPath)
DirExists(se_DirPath)
DirCreate(se_DirPath)
DirRemove(se_DirPath)
DirCount()
DirList()
DirPrompt()

File Functions

FileExists(se_FileName)
FileSearch(se_FileName,se_DirList)
FileRename(se_OldName,se_NewName)
FileSize(se_Name)
FileDate(se_FileName)
FileCopy(se_SourceFile,se_DestinationFile{,ne_Mode})
FileDelete(se_Name)
FileName(se_Name)
FileExt(se_Name)
FileDrive(se_Name)
FileDir(se_Name)
FilePath(se_Name)
FileChangeExt(se_FileName,se_NewExtension)
FilePrompt({Expr})
FileSave({Expr})
FilesCount({se_Filter})
FilesList({se_Filter})

Byte Buffer Manipulation

BuffPrintT vs_BuffString{,Expr,Expr;Expr...}{;|,}
BuffPrintB vs_BuffString{,Expr,Expr,Expr...}
BuffWrite(se_Buffer,ne_Position,e_Value)
BuffWriteB(se_Buffer,ne_Position,ne_Value)
BuffWriteF32(se_Buffer,ne_Position,ne_Value)
BuffRead(se_Buffer,ne_Position,ne_NumBytes)
BuffReadB(se_Buffer,ne_Position)
BuffReadI(se_Buffer,ne_Position)
BuffReadF(se_Buffer,ne_Position)
BuffReadF32(se_Buffer,ne_Position)

Low Level File I/O

FilePrintT vn_FileHandle{,Expr,Expr;Expr...}{;|,}
FilePrintB vn_FileHandle{,Expr,Expr,Expr...}
FileOpen(se_FileName,ne_Mode)
FileCreate(se_FileName)
FileReadField(ne_FileHandle{,se_Separator})
FileRead(ne_FileHandle{,ne_ByteCount})
FileReadB(ne_FileHandle)
FileReadI(ne_FileHandle)
FileReadF(ne_FileHandle)
FileWrite(ne_FileHandle,e_Value)
FileWriteB(ne_FileHandle,e_Value)
FileSeek(ne_FileHandle,ne_FromWhere,ne_OffsetCount)
FileSize(ne_FileHandle)
FileEnd(ne_FileHandle)
FileClose(ne_FileHandle)

Time, Date, System & Other Functions

Time & Date Functions

Timer( )
Now()
Hour(ne_DateTimeValue)
Minute(ne_DateTimeValue)
Second(ne_DateTimeValue)
Millisecond(ne_DateTimeValue)
Year(ne_DateTimeValue)
Month(ne_DateTimeValue)
Day(ne_DateTimeValue)
DayOfWeek(ne_DateTimeValue)
DateStr(ne_DateTimeValue)
TimeStr(ne_DateTimeValue)
DateTimeStr(ne_DateTimeValue)
DateTimeVal(se_DateTimeString)
DateVal(ne_Year,ne_Month,ne_Day)
TimeVal(ne_Hour,ne_Minute,ne_Second,ne_Milliseconds)
DateTime(ne_DateTimeValue{,se_Format})
Time({ne_Type})
Date({ne_Type})
ToTime(ne_Seconds)

Variables Manipulation & Indirection (Pointers)

vType(v_VarName)
varType(se_VarName)
varValue(se_VarName)
VarSet se_VarName, e_Value
varsList({ne_Global})
Swap v_Left | a_Left[...] , v_Right | a_Right[...]

System Information

ProgName()
CommandsList()
FunctionsList()
StatementsList()
ConstantsList()

Miscellaneous Functions

Delay {ne_Milliseconds}
MicroDelay {ne_Amount}
Evaluate(se_Expression)
Spawn(se_ProgramName,se_Parameters,ne_Mode)

Robot Simulator

Simulator Commands

rLocate ne_X,ne_Y{,ne_Heading{,ne_Size{,ne_BorderColor{,ne_InsideColor{,ne_ObeyFlip}}}}}
rRelocate {ne_X{,ne_Y{,ne_Heading}}}
rInvisible ne_Color1 {,ne_Color2...}
rFloorColor {ne_Color}
rForward {ne_Pixels}
rTurn {ne_Degrees}
rHeading {ne_Degrees}
rSpeed {ne_Speed}
rGps vn_X,vn_Y
rPen ne_State {,ne_Color}
rCharge {ne_Value}
rIgnoreCharge {true|false}
rSensor ne_SensorNo,ne_Range,vn_Color,vn_Distance,vn_Found
rSensorA ne_Angle,ne_Range,vn_Color,vn_Distance,vn_Found
rSlip {ne_PercentageLevel}
rSenseType {ne_NumSensors}
rInstError {ne_PercentageLevel}

Simulator Functions

rChargeLevel()
rPoints()
rRange({ne_Angle})
rBeacon(ne_Color)
rFeel()
rDFeel({ne_Color})
rBumper()
rDBumper({ne_Color})
rSense({ne_Color})
rLook({ne_Angle})
rGround(ne_SensorNo)
rGroundA(ne_Angle)
rCompass()
rGpsX()
rGpsY()

Simulator Serial I/O Protocol

rCommPort ne_PortNum {,ne_BaudRate {,ne_NumBits {,ne_Parity {,ne_StopBits {,ne_Protocol}}}}}
rCommand(ne_Command,ne_Data)

rLocate ne_X,ne_Y	(code 3)
rForward ne_Amount	(code 6 or 7)
rTurn ne_Amount	(code 12 or 13)
rCompass ()	(code 24)
rSpeed ne_Speed	(code 36)
rLook ({ne_Angle})	(code 48 or 49)
rGPS vn_X,vn_Y	(code 66)

rBeacon (ne_Color)	**(code 96)**
rChargeLevel ()	**(code 108)**
rPen ne_State	**(code 129)**
rRange ({ne_Angle})	**(code 192 or 193)**

Ports & Serial I/O

Serial I/O Commands

SetCommPort ne_PortNum {,ne_BaudRate {,ne_NumBits {,ne_Parity {,ne_StopBits {,ne_Protocol}}}}}
SerPorts vs_PortsList
SerOut Expr {,Expr {; Expr ...}}
SerialOut Expr {,Expr {, Expr ...}}
SerIn vs_Bytes
SerBytesIn ne_NumOfBytesToRead,vs_BytesRead,vn_ActualNumberRead
SetTimeOut {ne_MilliSeconds}
GetTimeOut vn_TimeOutValue
CheckSerBuffer vn_NumOfBytes
ClearSerBuffer {ne_Which}
ReadSerSignals vn_Flags
SetSerDTR {on|off}
SetSerRTS {on|off}

Parallel Ports I/O Commands

PPortOut {ne_ByteValue}
PPortIn vn_ByteValue
SetPPortNumber {ne_PortNumber}

Virtual Parallel Port I/O Protocol

VPPortOut ne_VirtualPortNo,ne_ByteValue
VPPortIn ne_VirtualPortNo,vn_ByteValue

General Ports I/O Commands

OutPort ne_PortNumber,ne_ByteValue
InPort ne_PortNumber,vn_ByteValue

USBmicro U4x1 Functions

DLL Specific Functions

usbm_DllSpecs()
usbm_ErrorSpecs()
usbm_ClearRecentError()
usbm_FindDevices()
usbm_NumberOfDevices()
usbm_SetReadTimeout(ne_Time)

Device Information Functions

usbm_DeviceSpecs(ne_DeviceNumber)
usbm_DeviceValid(ne_DeviceNumber)
usbm_CloseDevice(ne_DeviceNumber)

Device I/O Functions

usbm_InitPorts(ne_DeviceNumber)
usbm_InitPortsU401(ne_DeviceNumber)
usbm_InitPortsU421(ne_DeviceNumber)
usbm_InitPortsU451(ne_DeviceNumber)
usbm_DirectionA(ne_DeviceNumber,ne_PinsDirection,ne_PinsFormat)
usbm_DirectionB(ne_DeviceNumber,ne_PinsDirection,ne_PinsFormat)
usbm_DirectionAIn(ne_DeviceNumber)
usbm_DirectionAOut(ne_DeviceNumber)
usbm_DirectionAInPullUp(ne_DeviceNumber)
usbm_DirectionBIn(ne_DeviceNumber)
usbm_DirectionBOut(ne_DeviceNumber)
usbm_DirectionBInPullUp(ne_DeviceNumber)
usbm_WriteA(ne_DeviceNumber,ne_ByteValue)
usbm_WriteB(ne_DeviceNumber,ne_ByteValue)
usbm_ReadA(ne_DeviceNumber)
usbm_ReadB(ne_DeviceNumber)
usbm_ReadLatches(ne_DeviceNumber)
usbm_SetBit(ne_DeviceNumber,ne_PinNumber)
usbm_ResetBit(ne_DeviceNumber,ne_PinNumber)
usbm_WriteABit(ne_DeviceNumber,ne_AndingMask, ne_OringMask)
usbm_WriteBBit(ne_DeviceNumber,ne_AndingMask, ne_OringMask)

LCD Related Functions

usbm_InitLCD(ne_DeviceNumber,ne_Sel, ne_Port)
usbm_LCDCmd(ne_DeviceNumber,ne_CommandByte)
usbm_LCDData(ne_DeviceNumber,ne_DataByte)

1-Wire Related Functions

usbm_Reset1Wire(ne_DeviceNumber,ne_Specs)
usbm_Write1Wire(ne_DeviceNumber,ne_Data)
usbm_Read1Wire(ne_DeviceNumber)
usbm_Write1WireBit(ne_DeviceNumber,ne_BitValue)
usbm_Read1WireBit(ne_DeviceNumber)

2-Wire (e.g. I2C) Related Functions

usbm_Wire2Control(ne_DeviceNumber,ne_Signal)
usbm_Wire2Data(ne_DeviceNumber,se_DataBytes)

SPI Related Functions

usbm_InitSPI(ne_DeviceNumber,ne_Specs)
usbm_SPISlaveRead(ne_DeviceNumber)
usbm_SPISlaveWrite(ne_DeviceNumber,se_DataBytes)
usbm_SPIMaster(ne_DeviceNumber,se_DataBytes)

Stepper Motor Related Functions

usbm_Stepper(ne_DeviceNumber,se_DataSpecs)

Strobe I/O Functions

usbm_StrobeWrite(ne_DeviceNumber,se_ByteData)
usbm_StrobeRead(ne_DeviceNumber,se_ByteData)
usbm_StrobeWrites(ne_DeviceNumber,se_ByteData)
usbm_StrobeReads(ne_DeviceNumber,se_ByteData)

A General *HOOK* Function to future functionalities in the U4x1

usbm_DeviceCmd(ne_DeviceNumber,se_Data)

Internet Commands & Functions

Email (SMTP) Commands

SendEMail a_MessageSpecs,a_MessageBody {,ne_ShowProgress}

TCP Sockets Functions

General Functions

TCP_LocalIP()

Server Functions

TCPS_Serve({ne_Port})
TCPS_Header({on|off})
TCPS_BuffCount()
TCPS_Read()
TCPS_Peek()
TCPS_Send(se_Data)
TCPS_Close()
TCPS_Status()

Client Functions

TCPC_Connect(se_ServerIPaddress{,ne_ServerPort})
TCPC_ConnectHost(se_ServerName{,ne_ServerPort})
TCPC_BuffCount()
TCPC_Read()
TCPC_Peek()
TCPC_Send(se_Data)
TCPC_Close()
TCPC_Status()

UDP Sockets Functions

UDP_Start(se_Name{,ne_ListenPort})
UDP_BuffCount(se_Name)

UDP_Read(se_Name)
UDP_Peek(se_Name)
UDP_Send(se_Name,se_Data,se_TargetIP{,ne_TargetPort})
UDP_Header(se_Name{,on|off})
UDP_Status(se_Name)