KingView

Version6.53

Quick Reference Manual of Scripts Functions

WellinControl Technology Development Co.,Ltd

Preface

The manual is an assistant of KingView 6.53 User’s Guide of WellinControl technology development CO. LTD, introduces definition and usage of the scripts functions in KingView 6.53. The information in this documentation is subject to change without notice and does not represent a commitment on the part of WellinControl Company. Since deviations cannot be precluded entirely, we cannot guarantee full correction.

This software may be used or copied only in accordance with the terms of these agreements. No part of this documentation shall be reproduced, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of WellinControl Company, except buyer use it.

If you found some problems in the manual, please write it down and contact us. Suggestions for improvement are welcomed.

KingView 6.53 Scripts Functions Quick Reference Manual
2007 WellinControl **All Rights Reserved.**
Edition: 6.5.3.1

Tel: (86)010-82616619
E-Mail:support@WellinControl.com.cn sales@WellinControl.com.cn
http://www.kingview.com

KingView

Version6.53

Quick Reference Manual of Scripts Functions

WellinControl Technology Development Co.,Ltd

Preface

The manual is an assistant of KingView 6.53 User's Guide of WellinControl technology development CO. LTD, introduces definition and usage of the scripts functions in KingView 6.53. The information in this documentation is subject to change without notice and does not represent a commitment on the part of WellinControl Company. Since deviations cannot be precluded entirely, we cannot guarantee full correction.

If you found some problems in the manual, please write it down and contact us. Suggestions for improvement are welcomed.

KingView 6.53 Scripts Functions Quick Reference Manual
2007 WellinControl **All Rights Reserved.**
Edition: 6.5.3.1

Tel: (86)010-82616619
E-Mail:support@WellinControl.com.cn sales@WellinControl.com.cn
http://www.kingview.com

Catalog

Quick Reference Manual of Scripts Functions

KingView supports build-in complicated functions as string functions, math functions, system functions, and control functions, report functions. Next introduce functions alphabetically (the names of functions don't match case).

Abs

Return the absolute value of a specified number or tag. Syntax: Abs (Any number, long or integer tag);
Returns integer or long

Examples:
Abs(14) will return 14
Abs(-7.5) will return 7.5
Abs(Distance) will return the absolute value of the memory analog tag – Distance.

Ack

Confirm tag alarm or tag group alarm. If function parameter is tag name, only confirm tag alarm; if function parameter is tag group name, confirm all tag alarms belonging to this tag group or its subgroup. Function parameter can only be tag name or alarm group name, and can't be string tag. This function used to button script, that is, when raising alarm, confirms alarm using this function, it will create alarm confirmation event. Standard format:
Ack(alarm tag group name); or Ack(alarm tag name);
Examples:

Ack(All Plant) or Ack(Liquid level of reaction tank)

AckByTagName

Confirm tag alarm. Function parameter can be a string tag, also a string tag name. Standard format:

AckByTagName("tag_name");

Parameter:

tag_name: tag name

Example:

AckByTagName ("\\local\liquid depth");

AckByTagName(Varname); Varname is a string tag.

AckByGroupName

Confirm tag group alarm. Standard format:

AckByGroupName("station_name", "group_name");

station_name is IO server name, group_name is alarm tag group name. Function parameter can be a string tag, also a string IO server name or a string alarm tag group name

Example:

AckByGroupName ("PC1","alarm tag group1");

AckByGroupName (PCName, GroupName); //PCName, GroupName is string tag.

ActivateApp

Activate another currently running window application and makes it as current window. May cooperate with the function SendKeys.

Syntax: ActivateApp(“ExeName”);

Parameter	Description
ExekName	The task this function will activate.

Example:
The following statement will activate Microsoft Word.
ActivateApp(“Word.exe”);

You can use the following statement to activate KingView.
ActivateApp(“TouchView.exe”);

ActiveXIsVisible

This function is used to control the hide of form control. Transfer form:
ActiveXIsVisible("CtrlName", nMode);

Paramete	Description
CtrlName	Name of control
nMode	Control mode. When nMode=0, the control hides

Example:
The correct transferring way of hiding control:
ActiveXIsVisible(“Ctrl10”, 0);

ArcCos

Return the arc cosine of the tag. The tag must be within –1 and 1, otherwise the function return invalid result. Syntax:
ArcCos(number or tag);
Returns integer or long

Example:

ArcCos(1); will return 0.

ArcCos(temp);   will return the arc cosine of the tag “temp”.

AcrSin

Return the arc sine of the tag. The tag must be within –1 and 1, otherwise the function will return invalid result. Syntax:

ArcSin(number or tag);

Returns integer or long

Examples:

ArcSin(1); will return 90.

ArcSin(temp);   will return the arc sine of the tag “temp”.

ArcTan

Return the arc tangent of the tag. Syntax:

ArcTan(number or tag);

Returns integer or long

Examples:

ArcTan(1); will return 45.

ArcTan(temp);   will return the arc tangent of the tag “temp”.

Average

Return the average of the cells’ data in specified KingView report or of tags. Syntax:

Average(’a1’,’a2’); or Average('a1:a10');
a1, a2…… are the row number and the column number of the cell or integer tag or long tag. The number of parameters is from 1 to 32.
When averaging the selected cells in the report, the result is displayed in current cell. Syntax: Average(’a1’,’a2’);

Examples:
Averages the selected cells at random:
=Average(‘a1’, ’b2’, ’r10’);
Averages the continual cells:
=Average(‘b1:b10’);.
The following statement is to average the tags.
AverageValue= Average(lVar1,fVar1);

BackUpHistData

Load historical data from IO server to Record server. You may use it in Record server but need to make proper KingView network configuration. Refer to “History” in KingView 6.53 User’s guide for more. Syntax:
BackupStationData (Str chMchinename, Long ftEndtime);

Parameter	Description
chMchinename	string tag representing the name of IO server
ftEndtime	long tag representing ending time of backup

Examples:
//Backup historical data from “IO Acquiring Node” by ending time of current time
endTime=HTConvertTime($Year,$Month,$Day,$Hour,$Minute,0);
BackUpHistData("IO Acquiring Node", endTime);

Bit

Get the value (0 or 1) of a bit of an integer or long tag. Syntax:

OnOff=Bit(Var, bitNo);

Parameter	Description
nOff	discrete tag
Var	integer or long tag
BitNo	the sequence number of a bit between 1 and 16

Return value:

discrete. If the bit bitNo of tag Var is 0, return OnOff 0; if the bit bitNo of tag Var is 1, return OnOff 1;

Examples: Switch=Bit(DDE1, 6);

The status of Switch is equal to the sixth bit of the tag DDE1.

BitSet

Set any bit of an integer or long tag as the specified value (0 or 1). Syntax:

BitSet(Var, bitNo, OnOff);

Parameter	Description
Var	an integer or long tag
BitNo	the sequence number of a bit between 1 and 16
OnOff	specified value

Note: you only may apply the function to write/read tag of IO tag.

Examples:

BitSet(DDE1, 6, 0);   sets the sixth bit of DDE1 as 0.

ChangePassword

Display the Change Password dialog box allowing the logged on operator to change his/her password. Syntax:

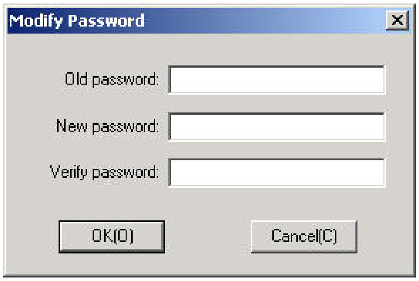
ChangePassword()

Examples:

Set the scripts link of the button on the picture as the following statement:

ChangePassword()

Click the button on running, so the following dialog box shows.



The dialog prompts the user to enter the current password, the new password and verification of the new password. Click OK, so the password becomes new one.

chartAdd

Add a new bar chart in specified bar graph control. Syntax:

chartAdd("ControlName", Value, "label");

Parameters	Description
ControlName	the name of the bar graph control in English or in Chinese
Value	integer or long original value of the new bar chart
Label	the label pf new bar chart. A default is the index value which range is from 1 to 16.

Examples:

chartAdd("XYChart", 1, "L6");

The statement is to add a bar chart with the label “L6” and original value 1 in the bar graph control XYChart.

chartClear

Clear all bar charts in selected bar graph control. Syntax:

chartClear("ControlName")

Parameters	Description
ControlName	the name of the bar graph control in English or in Chinese

Examples: chartClear("XYChart");

The statement is to clear up all bar charts in bar graph control XYChart.

chartSetBarColor

Set the color of the pie chart in selected bar graph control. Cannot be applied to bar chart. Syntax:

chartSetBarColor("ControlName", barIndex, colorIndex);

Parameters	Description
ControlName	the name of the bar graph control in English or in Chinese
BarIndex	integer’s index number of bar chart whose range is from 0 to 15
ColorIndex	integer’s color index number of bar chart whose range is from 0 to 15. The number is corresponding with the following color.

Color index	Color	Color index	Color
0	Default	8	Gray
1	Blue	9	Light Blue
2	Green	10	Light Green
3	Cyan	11	Light Cyan
4	Red	12	Light Red
5	Magenta	13	LightMagenta

6	Yellow	14	Light Yellow
7	White	15	Bright White
16	Black		

Examples:
chartSetBarColor("XYChart", 0, 1); will set the color of the first pie chart of the bar graph control XYChart as blue.
chartSetBarColor("XYChart", 2, 4); will set the color of the third pie chart of the bar graph control XYChart as red.

chartSetValue

Set data for the bar chart with the index as the tag Index in selected the bar graph control. Syntax:
chartSetValue("ControlName", Index, Value);

Parameters	Description
ControlName	the name of the bar graph control in English or in Chinese
Value	integer or long data that you want to set
Index	the label value of the bar chart whose range is from 0 to 15.

KingView sets the label value automatically for added new bar chart from 0 in ascending.

Example: chartSetValue("XYChart",2, 30); sets the value of the bar chart with the index as 2 (third chart) in the bar graph control XYChart as 30.

ClosePicture

Close the picture loaded in the memory and deletes it from the memory. Syntax:
ClosePicture("Picture")

Examples:
ClosePicture("Reaction Workshop"); will close the picture “Reaction Workshop”.

ConfigODBC

It be used to configure ODBC data source, standard format:
ConfigODBC(nDatatbasetype, szAttributes);
Parameter:
nDatatbasetype: database type, now it supports Access, SQL server
When nDatatbasetype is 0, database type is Access; when nDatatbasetype is 1, database type is SQL server.
szAttributes: configure string

Example1:
Configure Access database, DSN name is demo2, database file is E:\Program Files\Kingview\Example\Kingdemo1\database.mdb.

ConfigODBC(0,"DSN=demo2\0Description=E:\Program Files\Kingview\Example\Kingdemo1\database.mdb\0PWD=1234\0UID=shihf");

Example2:
Configure Access database,
ConfigODBC(1, "DSN=MyDSN\0 Description =SQLConfigDSN Sample\0SERVER=MySQL\0ADDRESS=MyServer\0NETWORK=dbmssocn\0DATABASE=pubs\0");

Cos

Return the cosine of the tag. Syntax:
Cos(number or tag)

Examples:
Cos(90); will return 0
Cos(temp); will return the cosine of the tag temp.

Date

Return the data string by given year, month and day. The default format is Year:Month:Day. Syntax:
Date(LONG nYear, LONG nMonth, LONG nDay);

Examples:
The tags Year, Month and Day are respectively "$Year", "$Month" and "$Day", ask you to display the string "$Date" depending on the above three integers, so you may input the following in scripts.
Date=Date(Year, Month, Day).

DisplayMCI

Provide a strong general interface for multimedia devices. The default format is:
DisplayMCI("MCICommand", option); Next introduce its usage on examples.
DisplayMCI("PLAYCD",3);
Play the third song in CD.
DisplayMCI("STOPCD","");
Stop playing CD.
DisplayMCI("PLAYMIDI","c:\midi.mid");
Play the background music of MIDI type "c:\midi.mid".
DisplayMCI("PAUSEMIDI"," c:\midi.mid");
Pause the background music of MIDI type "c:\midi.mid".
DisplayMCI("RESUMMIDI"," c:\midi.mid");

Continue playing the background music of MIDI type “c:\midi.mid”.

DisplayMCI(“CLOSEMIDI”,” c:\midi.mid”);

Stop playing the background music of MIDI type “c:\midi.mid”.

DisplayMCI(“EJECTCD”);

Eject all CD in the CD driver.

Dtext

Dynamically change a string tag based on the value of a discrete tag. Syntax:

Str = Dtext(Discrete_Tag, OnMsg, OffMsg);

Parameters	Description
Discrete_Tag	Discrete tag
OnMsg	String tag
OffMsg	String tag

When Discrete_Tag=1, Str is OnMsg

When Discrete_Tag=0, Str is OffMsg

Examples:

Str = Dtext(Power switch, “Switch on”, “Switch off”);

When Power switch=1, Str is Switch on

When Power switch=0, Str is Switch off.

EditUsers

Configure users when the picture is running. It is often used in the scripts of buttons. Syntax:

EditUsers( );

The access level of current user must be bigger than or equal to 900 for configuring other users.

EnableNavigate

It is used to display/close navigator, standard format:
EnableNavigate(is_enable);
Parameter:
is_enable: integer.
0: when is_enable is 0, close navigator;
1: when is_enable is 1, display navigator.
After using EnableNavigate function to close navigator, navigator can’t be displayed unless using this function.

Example:
EnableNavigate(0);   // close navigator

EnableSaveTagValueToIniValueWhenValueChange

For the tag which setting “Save Value” and “Save Parameter”, after using this function, when tag value and parameter value change, whether KingView TouchView exits normally, tag value and parameter value will be saved to file tagname.db automatically, once KingView is run again, tag value saved and parameter value saved will become initial tag value and initial parameter value.
The meaning of “Save Value” and “Save Parameter” refers to Chapter5 Tags definition and management in KingView User’s Guide.
Standard format:
EnableSaveTagValueToIniValueWhenValueChange(is_enable);
Parameter:
is_enable: integer or value
1: when tag value and parameter value change, tag value and parameter value are saved. KingView TouchView quits, once KingView is run again, tag value saved and parameter value saved will become initial tag value and initial parameter value.
0: function is same to “Save Value” and “Save Parameter”.
Example:

Define tag in Tagname dictionary in KingView WindowMaker, set “Save Value”. Execute function:
EnableSaveTagValueToIniValueWhenValueChange(1);
when tag value changes, system save this tag value. KingView TouchView exits, once running again, tag value saved will become initial value.

Exit

Exit the TouchView. Syntax:
Exit(Option);

Parameters	Description
Option	Any number or integer’s tag, which can be one of the following:

0 - Exit the current application.
1 – Shut down
2 - Restart windows.

Exp

Return the result of e raised to a power.
Exp(number or tag)

Examples:
Exp(1); will return 2.718
Exp(temp); will return the exponential of the tag temp.

FileCopy

Copy a SourceFile to a DestFile, similar to the DOS Copy command or the Copy function in Windows File. Syntax:
FileCopy(SourceFile, DestFile, DoneTag);

Parameters	Description
SOURCEFILE	Source file name (including the full path).
DestFile	Destination file name (including the full path) or directory name (see the following examples).
DONETAG	Invalid temporarily. It is the tag name used to report the *progress* of the copy procedure. The parameter must be a memory long integer or memory analog. Its value changes from o to 100 as the copy procedure.

Return:

success

fail to start

-1- error.

Examples:

Status=FileCopy("C:\*.TXT", "C:\BACKUP", 'DoneTag');

Status is an integer tag to which the 1, -1 or 0 will be written.

FileCopy() function is performed in the background so that it will not interfere with the operation of KingView. Status indicates whether the copy procedure has been initiated successfully. It is set to 1 when the procedure is successfully completed or -1 if there is an error before the procedure could be completed.

Normally, the SourceFile and DestFile will be names of files. However, if a single file is being copied with **FileCopy()** function, the destination can be a directory:

FileCopy("C:\DATA.TXT", "C:\BACKUP", 'DoneTag');

This will copy the file "DATA.TXT" to a directory called "BACKUP" on the "C:\" drive. The tag Monitor will be set to 1 when this is complete.

If the SourceFile contains any wildcards, however, the DestFile MUST be a directory (not a filename) or the function will return an error code:

FileCopy("C:\*.TXT", "C:\BACKUP", 'DoneTag');

This will copy all the .TXT files from the C:\ root directory to the directory C:\BACKUP.

FileDelete

Delete the unnecessary or unwanted files. Syntax:

FileDelete(Filename);

Parameters	Description
Filename	File name to be deleted.

If the file can be found and successfully deleted, the function will return 1. Otherwise, the function will return 0.

Examples:

Status=FileDelete("C:\DATA.TXT");

Status is a 1 if it finds a file called "DATA.TXT" in the C:\ root directory, or a 0 if it does not find the file.

FileMove

Similar to FileCopy() function except that it moves the file from one location to another instead of making a copy. Syntax:

FileMove(SourceFile, DestFile, DoneTag);

Parameter	**Description**
SourceFile	Source filename (including the full path)
DestFile	Destination filename (including the full path)
DoneTag	The name of a tag to report the progress of the move procedure. This parameter must be a memory long integer or a memory analog whose range is from 0 to 100.

Return: 1- success

fail to start

-1- error.

Examples:
Status=FileMove("C:\DATA.TXT","D:\DATA.TXT",Monitor);
Status: an integer to which 1, -1 or 0 will be written.
Monitor: defined memory integer in tagname dictionary.
FileMove() funciton is performed in the background so that it doesn’t interfere with the operation of KingView. The purpose of the DoneTag is to allow the progress of the move operation to be monitored by an application or a user. In this way, the user can be alerted to any errors which might occur after the procedure was initiated. This is different than the Status returned above, which indicates whether the move procedure has been successfully initiated. Once the move procedure has been successfully initiated, the value of Monitor is then assigned. The value is set to 0 while the procedure is still in progress. It is set to 1 when the procedure is successfully completed or -1 if there is an error before the procedure could be completed.
If the SourceFile and DestFile are located on the same drive, the function can simply change the file's directory reference (where the computer keeps the name and location of the file on disk) without actually moving any data. In this case, the move procedure will be very fast, regardless of the size of the file. If the Source File and destination File are located on different drives, the time the move takes will vary with the size of the file. This is because the data must be transferred from one physical disk to another.
FileMove("C:\DATA.TXT","C:\BACKUP\DATA.TXT",Monitor);
This will move the file called "DATA.TXT" from the root directory of the "C" drive to a directory called "BACKUP." The tag Monitor will be set to 1 when this is complete.
This function can also be used to rename files when the Source File and destination File specify the same directory but different filenames.
FileMove ("C:\DATA.TXT","C:\DATA.BAK",Monitor);
This will rename the file "DATA.TXT" to "DATA.BAK" in C:\ root directory. The tag Monitor will be set to 1 when this is complete.

FileReadFields

Read a Comma Separated Variable (CSV) record from a specified file. Syntax:
FileReadFields(Filename, FileOffset, ”StartTag”, NumberOfFields);

Parameters	**Description**
Filename	Specifies the file to read from.
FileOffset	Specifies the location in the file to start reading. If it is 1 it means to read from the beginning.
StartTag	Specifies the name of a KingView tag where the first data item will be written to. The name of this tag must end with a number (such as MyTag1). This parameter must be a string indicating the name of the tag (not the actual tag itself). So, if your tag is called MyTag1, you would provide "MyTag1" or MyTag1.name, not just MyTag1.
NumberOfFields	Specifies the number of fields to read (the number of comma separated fields in each record in the file).

If StartTag is "MyTag1" and NumberOfFields is 3, then 3 fields are read from the file and are stored in MyTag1, MyTag2 and MyTag3. These tags with consecutive names must be first created in KingView and may be of different types (Integer, Message, etc.).

Examples: If the first line of C:\DATA\FILE.CSV is:
“This is text, 3.1416, 5”,
The following script will read this line and store "This is text" into MyTag1, 3.1416 into MyTag2 and 5 into MyTag3:
BytePosition=FileReadFields ("C:\DATA\FILE.CSV", 1, "MyTag1", 3);
The function returns the new byte position after the read. You can use this return value as the FileOffset for the next read.
FileReadFields ("C:\DATA\FILE.CSV", BytePosition, "MyTag1",3);

Note: StartTag must be contained in quotation marks.

FileReadStr

Read a specified number in bytes (or a whole line) from a specified file. Syntax:
FileReadStr(Filename,FileOffset,Str_Tag,CharsToRead);

Parameter	Description
Filename	Specify the file to read from.
FileOffset	Specify the location in the file to start reading. If it is 1, it means to read from the beginning.
Str_Tag	Specify where to store the data read from the file.
CharsToRead	Specify how many bytes to read from the file. For processing text files, CharsToRead can be set to 0. The function will read up to the next LF (linefeed) in the file,

Example:
FileReadStr ("C:\DATA\FILE.TXT", 1, Str_Tag, 0);
The first line will be read from file "C:\DATA\FILE.TXT" and stored into MsgTag. The function returns the new byte position after the read. You can use this return value as the FileOffset for the next read.

FileWriteFields

Write a Comma Separated Variable (CSV) record to a specified file.
Syntax: FileWriteFields(Filename,FileOffset,”StartTag”,NumberOfFields);

Parameter	Description
Filename	Specify the file in which to write. If *Filename* does not exist, it will be created.
FileOffset	Specify the location in the file to start writing. If FileOffset is -1, the function will write to the end of the file.

Starttag	Specify the name of the tag where the first data item will come from. The name of this tag must end with a number (such as MyTag1). This parameter must be a string indicating the name of the tag (not the actual tag itself). So if the tag is called MyTag1, you would provide "MyTag1" or MyTag1.name, not just MyTag1.

NUMBEROFFIELDS SPECIFY THE NUMBER OF FIELDS TO WRITE (THE NUMBER OF COMMA SEPARATED FIELDS IN EACH RECORD IN THE FILE).

The function returns the new byte position after the write. You can use this return value as the *FileOffset* for the next write.

If *StartTag* is "MyTag1" and *NumberOfFields* is 3, then 3 fields are written to the file (from MyTag1, MyTag2 and MyTag3). These tags with consecutive names must first be created in InTouch and may be of different types (Integer, Message, etc.).

The following script will write the line "This is text, 3.1416, 5" to the first line of C:\DATA\FILE.CSV. "This is text" is the current value in MyTag1, 3.1416 is in MyTag2 and 5 is in MyTag3:

FileWriteFields ("C:\DATA\FILE.CSV", 1, "MyTag1", 3);

The following script will write the text string MyTag1 to the end of C:\DATA\FILE.CSV:

FileWriteFields ("C:\DATA\FILE.CSV", 0, "MyTag1", 3);

StartTag must be contained in quotation marks.

FileWriteStr

Write a specified number of bytes (or a whole line) to a specified file. Syntax:

FileWriteStr(Filename,FileOffset,String,LineFeed);

Parameter	Description
Filename	Specify the file to write to. If *Filename* does not exist, it will be

created.

FileOffset	Specify the location in the file to start writing. If FileOffset is -1, the function will write to the end of the file.
String	Specify the characters to write to the file.
LineFeed	Specify whether or not to add a linefeed after the write operation. When writing to a text file, set the LineFeed to 1.

The function returns the new byte position after the write. You can use this return value as the FileOffset() function for the next write.

The following statement writes a string tag called MsgTag to the end of file *C:\DATA\FILE.TXT*:

FileWriteStr ("C:\DATA\FILE.TXT", 0, MsgTag, 1)

GetBackupProgress

Returns percentage of progress bar when KingView combines the backup of network historical data. Syntax:

GetBackupProgress(str szStationName);

Parameter: szStationName: the name of remote node.

Return: an integer’s progress value between 1 and 100.

Examples:

Progress=GetBackupProgress(“IO Acquiring node”);

GetDate

Convert a long integer number of seconds to date value what takes the format of UTC. The time in seconds takes 00:00:00 hours UTC, Jan first, 1970 as reference time. Syntax:

GetDate(DateTime,Year,Month,Day);

Parameter Description

DateTime	the integer’s number converted
Year	Year after conversion, integer
Month	Month after conversion, integer
Day	Day after conversion, integer

Example:

There are 1060301289 seconds between 00:00:00 hours UTC, Jan first, 1970 and 0:8:9 hours UTC, August eighth, 2003, the following statement will get date from the number and store 2003 into Year, 8 into Month and 8 into Day.

GetDate(1060301289, Year, Month, Day);

The function to get time is GetTime().

GetDatelocal

This function converts time from long data type to corresponding date type, output according to year, month, and data separately. Benchmark of Second number of long data type is GMT (Greenwich Mean Time) 01/01/1970 00:00:00. Output date after conversion is native date. Standard format:

GetDatelocal(DateTime,Year,Month,Day);

Parameter:

DateTime: number of needing date conversion, integer, input parameter

Year: year, integer, the number after conversion, output parameter

Month: month, integer, the number after conversion, output parameter

Day: day, integer, the number after conversion, output parameter

Example:

Using function HTConvertTime to convert time from GMT(Greenwich Mean Time) 01/01/1970 00:00:00 to Beijing time 09/26/2006 09:22:04 the second number of long data type is 1159233724, using function GetDatelocal can separate Beijing date to year, month, day.

GetDatelocal (1159233724, year, month, day);

After execution, year value is 2006, month value is 9, and day value is 26.

GetGroupName

Return the name of the alarm group by its ID. Each alarm group has the name as well as ID in KingView. The tag field ".Group" indicates ID of the alarm group.

Syntax:

sGroupName= GetGroupName(StationName,GroupID);

Parameters	Description
StationName	the node name where the alarm group is. (the option is invalid temporarily and may be replaced by null string)
GroupID	ID of the alarm group.

The returned value is a string.

Example:

GroupName=GetGroupName("", \\Local\\Oil tank Level.Group);

GetKey

Return the sequence number of encrypting lock used currently. Syntax:

KeyID=GetKey();

The function has no parameter and returns a string.

GetPictureScrollXPos

This function is used to get the X-value in the upper left corner of the focused window. The format of using grammar:

GetPictureScrollXPos(STRING PictureName);

Parameter	Description
PictureName	Name of the screen

The returned value is X-value in the upper left corner of the focused window.

Example:
xx=GetPictureScrollXPos("ControlCenter");

GetPictureScrollYPos

This function is used to get theY-value in the upper left corner of the focused window. The format of using grammar:
GetPictureScrollYPos(STRING PictureName);

Parameter	Description
PictureName	Name of the screen

The returned value is Y-value in the upper left corner of the focused window.

Example
yy=GetPictureScrollYPos ("ControlCenter");

GetProjectPath

This function is used to get the path of current project.
The format of using grammar:
GetProjectPath();

Example: In the Kingview Tagdictionary, memory string variable is defined as: VarName
VarName=GetProjectPath();

GetRealDBForBool

This function is used to get real bool value of a variable. The format of using grammar:

GetRealDBForBool("VarName");

Parameter	Description
VarName	Name of variable

Note: The variable can only be discrete variable..

Example

bb= GetRealDBForBool("RawoilValve");

GetRealDBForFloat

This function is used to get real analog value of a variable. The format of using grammar:

GetRealDBForFloat ("VarName");

Parameter	Description
VarName	Name of variable, format of character string

Note: The variable can only be analog one.

Example:

ff= GetRealDBForFloat ("RawoilLevel");

GetRealDBForInt

This function is used to get real integar value of a variable. The format of using grammar:

GetRealDBForInt ("VarName");

Parameter	Description
VarName	Name of variable, format of character string

Note: The variable can only be integar one.

Example:
ii= GetRealDBForInt ("Fruit");

GetRealDBForString

This function is used to get real string value of a variable The format of using grammar:
GetRealDBForString ("VarName");

Parameter	Description
VarName	Name of variable, format of character string

Note: The variable can only be string one

Example:
ss= GetRealDBForString ("DateRecord");

GetStationStatus

Return the status of backup when KingView combines the backup of network historical data. Syntax:
BOOL GetStationStatus(str szStationName);

Parameter	Description
szStationName	the name of remote node.

Return: a discrete, it is lager than 0 when in backup or equal to 0 when idle.

Example:
BackupStatus= GetStationStatus(“IO Acquiring node”)

GetTime

Convert the time in seconds (long integer) to the time displayed with the format of year, month, day, hour, minute and second. The time in seconds takes 00:00:00 hours UTC, Jan first, 1970 as reference time. Syntax:

GetTime(DateTime,Hour,Minute,Second);

Parameter	Description
GetTime	the integer’s number converted
Minute	Year after conversion, integer
Month	Month after conversion, integer
Second	Day after conversion, integer

Example:

There are 1060301289 seconds between 00:00:00 hours UTC, Jan first, 1970 and 0:8:9 hours UTC, August eighth, 2003, the following statement will get time from the number and store 0 into hour, 8 into Minute and 9 into Second.

GetTime(1060301289, Hour, Minute, Second);

The function to get time is GetDate().

GetTimelocal

This function converts time from long data type to corresponding time type, output according to hour, minute, second separately. Benchmark of Second number of long data type is GMT (Greenwich Mean Time) 01/01/1970 00:00:00. Output time after conversion is native time. Standard format:

GetTimelocal(DateTime,Hour,Minute,Second);

Parameter:

DateTime: number of needing time conversion, integer, input parameter

Hour: hour, integer, the number after conversion, output parameter

Minute: minute, integer, the number after conversion, output parameter

Second: second, integer, the number after conversion, output parameter

Example:

Using function HTConvertTime to convert time from GMT(Greenwich Mean Time) 01/01/1970 00:00:00 to Beijing time 09/26/2006 09:22:04, second number of long data type is 1159233724, using function GetTimelocal can separate Beijing time to hour, minute, second.
GetTimelocal (1159233724, hour, minute, second);
After execution, hour value is 9, minute value is 22, and second value is 4.

HidePicture

Hide the currently active picture but doesn't delete it from memory. Syntax
HidePicture("PictureName");
Parameter: PictureName a string

Example:
HidePicture("ReactionWorkshop");

HTConvertTime

Convert specified time (displayed with year, month, day, hour, minute, second) to the time in seconds (long integer) based on time reference of 00:00:00 hours UTC, Jan first, 1970. For example, Beijing is in east eighth block, so the time reference is 8:00:00, Jan first, 1970. Syntax:
HTConvertTime(Year,Month,Day,Hour,Minute,Second);

Parameters	Description
Year	Year (integer) that must be between 1970 and 2019
Month	Month (integer) that must be between 1 and 12
Day	Day (integer) that must be between 1 and 31
HOUR	**HOUR (INTEGER) THAT MUST BE BETWEEN 0 AND 23**
Minute	Minute (integer) that must be between 0 and 59

Second	Second (integer) that must be between 0 and 59

Return: integer

Note:

The function converts time (displayed with year, month, day, hour, minute and second) to the time in seconds from 00:00:00 hours UTC, Jan first, 1970 to now. When defining the tag to store the returned result, you should set its maximum as the upper limit of integer type as 2*109. Otherwise it might result in wrong converted time that the returned time exceeds the range.

Example:

The following statement returns 3600.

HTConvertTime(1970,1,1,9,0,0);

HTGetPenName

Return the name of the tag used for the specified pen of the specified trend. Syntax:

MessageResult=HTGetPenName(HistoryName, PenNum);

Parameter	Description
HistoryName	The tag representing the name of the trend.
PenNum	Integer tag or a number representing the pen number (from 1-8). A string tag is returned representing the specified pen's tag.

Example:

The following statement retrieves the tag for Pen2 of the trend with the tag Trend1 and places the result in the message tag TrendPen:

TrendPen=HTGetPenName(Trend1, 2);

HTGetPenRealValue

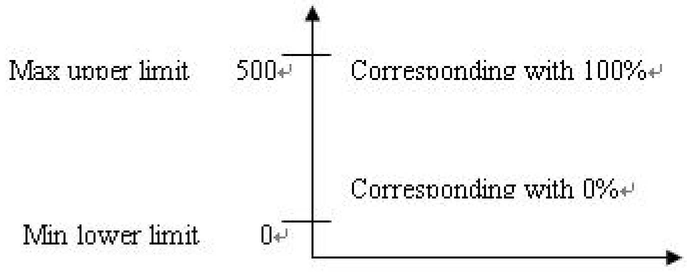
Return the real value of the pen of specified historical trend. Syntax:

HTGetPenRealValue(HistroyName,PenNum,ContentString);

Parameters	Description
HistroyName	the name of the historical trend defined in the Historical trend dialog box
PenNum	the index number of the pen corresponding with the tag in the historical trend
ContentString	a string constant
“start”	the real value of the field ValueStart of the historical trend. ValueStart is percentage to indicate the lower limit of the tag while start is a real value converted from the percentage.
“end”	the real value of the field ValueEnd of the historical trend. ValueStart is percentage to indicate the upper limit of the tag while end is a real value converted from the percentage.

Example:

Given that there is a historical trend of the temperature, its upper limit is 500 and lower limit is 0. it is shown as the following figure.



Displayed data is percentage of the temperature if outputting ValueStart and ValueEnd, as ValueEnd is 50, so it indicates 50%.

The following statement will return the real value (500*50%=250) corresponding with 50%. Here history is the name of historical trend and 1 is the trend pen of the

temperature.
HTGetPenRealValue(histroy,1,“end”);

HTGetTimeAtScooter

Return the time in seconds since 00:00:00 hours GMT, January 1, 1970 for the sample at the scooter location specified by *ScootNumc*.
Syntax:
IntegerResult=HTGetTimeAtScooter(HistoryName,ScootNum);

PARAMETER	**DESCRIPTION**
HistoryName	the tag representing the name of a trend.
ScootNum	Integer representing the left or right scooter (1= Left Scooter, 2= Right Scooter).

ChartStart, ChartLength, ScootNum cause the expression to be evaluated when any of these parameters change.

EXAMPLE:

The following statement retrieves the time in seconds for the value at the current scooter location for the left scooter of the trend labeled *Trend1*:
TimeLength=HTGetTimeAtScooter(Trend1 ,1);

HTGetTimeStringAtScooter

Return the string containing the time/date for the sample at the scooter location specified by ScootNum and ScootLoc. Syntax:
MessageResult=HTGetTimeStringAtScooter(HistoryName,ScootNum,pTextForma
t);
Parameters Description

HistoryName	the tag representing the name of the trend.
ScootNum	Integer representing the left or right scooter (1= Left Scooter, 2= Right Scooter).
TextFormat	String specifying the time/date format to use. The following strings are acceptable:

“Date”	to display date with the format similar to of Windows control panel
“Time”	to display time with the format similar to of Windows control panel
“DateTime”	Display date and time.

ChartStart, ChartLength, ScootNum cause the expression to be evaluated when any of these parameters change. The content of returned value depends on the format of the string.

Example:

The following statement retrieves the date/time for the value at the current scooter location for the right scooter of the trend labeled *Trend1*. The value is stored in the string tag NewRightTimeString and is in "Time" format:

NewRightTimeString=HTGetTimeStringAtScooter (Trend1, 2, "Time");

HTGetValue

Return a value of the requested type for the entire trend's specified pen. Syntax:

RealResult=HTGetValue(HistoryName,PenNum,ContentString);

Parameters	Description
HistoryName	Tag representing the name of the trend.
PenNum	Any integer or tag representing the pen number (from 1-8).
ContentString	String indicating the type of value to return: The following values are acceptable.

“AverageValue”	the average of whole trend
“MaxValue”	the maximum of whole trend
“MinValue”	the minimum of whole value

A memory real tag is returned representing the calculated value of the given type.

Example:
The following statement will return the maximum of data of Pen2 of the trend Trend2.
LeftHemisphereSD=HTGetValue(Trend1,2 , "MaxValue");

HTGetValueAtScooter

Return a value of the requested type for the sample at the specified scooter position, trend and pen number. Syntax:
RealResult=HTGetValueAtScooter(HistoryName, scootNum, PenNum, ContentString);

PARAMETER	DESCRIPTION
HistoryName	Tag representing the name of the trend.
scootNum	Integer representing the left or right scooter (1=Left Scooter, 2=Right Scooter).
PenNum	Any integer or tag or value representing the pen number (from 1-8).
ContentString	String indicating the type of value to return. The following values are acceptable.

“Value”	Value at scooter position.
“Valid”	0 if value is invalid, 1 if valid.

When the parameter ContentString is “Value”, the analog is returned, if “Valid”, the discrete is returned.

Example:
The following statement will return the current value of Pen3 of the trend Trend1 at the right scooter. It stores a 1 into the discrete ValidFlag if the value is valid or a 0 if

the value is invalid.

ValidFlag=HTGetValueAtScooter(Trend1,2,3 , "Valid");

HTGetValueAtZone

Return a value of the requested type for the data contained between the right and left scooter positions for a trend's specified pen. Syntax:

RealResult=HTGetValueAtZone(HistoryName,PenNum, ContentString);

Parameters	Description
HistoryName	Tag representing the name of the trend.
PenNum	Integer tag or value representing the pen number (from 1-8).
ContentString	String indicating the type of value to return: The following values are acceptable.

“AverageValue”	the average for the data contained between the right and left scooter positions
“MaxValue”	the maximum for the data contained between the right and left scooter positions
“MinValue”	the minimum for the data contained between the right and left scooter positions

The function uses directly the trend tags .ScooterPosLeft and .ScooterPosRight of Runtime Database to determine the zone borders and return the result.

Example:

The following statement calculates the average value for the data between the right and left scooters of trend *"Trend1"*, *Pen1*. The value is stored in the memory real tag AvgValue

AvgValue=HTGetValueAtZone(Trend1, 1, "AverageValue");

HTResetValueZone

Value axis of trend curve will recover to initial state if using this function (in KingView WindowMaker, we call start value and maximum value of value axis as initial value in mark definition of historical trend curve). Standard format:

HTResetValueZone(HistoryName);

Parameter:

HistoryName: historical trend tag, represent trend name

Example:

Recover data axis of trend curve Trend1 to initial state.

HTResetValueZone (Trend1);

HTScrollLeft

Set the start time of the trend to a value older than the current start time by a percentage of the trend's width. The effect is to scroll the time axis of chart to the left by a given percentage. Syntax:

HTScrollLeft(HistoryName,Percent);

Parameters	Description
HistoryName	Tag representing the name of the trend.
Percent	Long representing the percent of the chart to scroll (0.0 to 100.0)

The following statement scrolls the time axis for a trend Trend1 to left by 10%:

fvHTScrollLeft(Trend1,10.0);

If the current display starts at 12:00:00 PM and the display width is 60 seconds, then, the new trend will start at 11:59:54 AM (after the function is processed)

HTScrollRight

Set the start time of the trend to a value newer than the current start time by a percentage of the trend's width. The effect is to scroll the time axis of chart to the right by a given percentage. Syntax:

HTScrollRight(HistoryName,Percent)

Parameters	Description
HistoryName	Tag representing the name of the trend.
Percent	Long representing the percent of the chart to scroll (0.0 to 100.0)

Example:

The following statement scrolls the time axis for a trend Trend1 to right by 20%:

HTScrollRight(Trend1,20.0)

If the current display starts at 12:00:00 PM and the display width is 60 seconds, then, the new trend will start at 12:00:12 PM (after the function is processed).

HTSetLeftScooterTime

Set the start time point for historical trend. Syntax:

HTSetLeftScooterTime(HistoryName,LeftScooterTime);

Parameter	Description
HistoryName	The name of historical trend defined in the Historical Trend dialog box
LeftScooterTime	The start time point of historical trend. It is relative time in seconds on the beginning point of 00:00:00 hours GMT, Jan first, 1970

HTSetPenName

Assign a different tag to a trend's pen. Syntax:

HTSetPenName(HistoryName,PenNum,TagNameString);

Parameter	Description
HistoryName	Tag representing the name of the trend.
PenNum	Any integer or integer tag representing the pen number (from 1-

8).

TagnameString	String representing the new tagname to assign to the pen.

Example:

The following statement will set Pen3 of the trend Trend1 to represent the tag OutletPressure.

HTSetPenName(Trend1, 3, ”OutletPressure”);

HTUpdateToCurrentTime

Set the current time as the end time of the trend but doesn’t change the time axis’s width. It is mainly for check the newest data. Syntax:

HTUpdateToCurrentTime(HistoryName);

Parameter	Description
HistoryName	Tag representing the name of the trend.

The following statement will display the newest data of the trend Trend1 at current time.

HTUpdateToCurrentTime(Trend1);

If it is now 3:04 PM and the width of the trend is 60 seconds, the new end time will be 3:04 PM. The new start time will be 3:03 PM.

HTZoomIn

Change the start time and end time of the trend on reducing the time axis’s width to zoom in the part. The new start time depends on AlignPosString. Syntax:

HTZoomIn(HistoryName, AlignPosString);

Parameter	Description
HistoryName	Tag representing the name of the trend.
AlignPosString	String representing the type to zoom in. the following value is acceptable.

“StartTime”	keep the start time equal to before zoom
“Center”	Keep center time equal to before zoom
“EndTime”	Keep end time equal to before zoom

HTZoomIn(HistoryName,AlignPosString);

Note:

The start time and the end time after zoom is relative with the position of the two scooters (value of the fields ScortPosLeft and ScortPosRight) before zoom. If the positions of two scooters are not at the ends of the trend, the width after zoom is equal to between ScortPosLeft and ScortPosRight. In this case, the value of LockString is useless. The minimum width of the chat is 1 second. The scooter positions will be set to .ScooterPosLeft=0.0 and .ScooterPosRight=1.0 after the zoom.

The following statement displays half the width of the trend and maintains the same start time for the trend. Trend1.ScooterPosRight is equal to 0.0. If the start time before zooming was 1:25:00 PM and the chart width was 30 seconds, the start time will still be 1:25:00. But, the chart width will be 15 seconds after zooming.

HTZoomIn(Trend1,"StartTime");

HTZoomOut

Calculate a new chart width and start time. The new chart width is the old chart width multiplied by two. The new start time will be calculated based on the value of AlignPosStrin parameter. Syntax:

HTZoomOut(HistoryName, AlignPosStrin);

Parameter	Description
HistoryName	Tag representing the name of the trend.
AlignPosString	String representing the type of zoom: the following values are acceptable.

"StartTime" Keep the start time equal to before zoom

"Center" Keep center time equal to before zoom

"EndTime" Keep end time equal to before zoom

Example:

The following statement zooms the display by a factor of two and maintains the same center time for the trend. If the start time before zooming was 2:15:00 AM and the chart width was 30 seconds, the start time after zooming will be 2:14:45. The chart width will be 60 seconds and the center of the trend will remain at 2:15:15.

HTZoomOut("Volume","Center");

InfoAppActive

Test whether an application is active. Syntax:

DiscreteResult=InfoAppActive(“ExeName”);

Return: discrete

Parameter	Description
ExeName	the name of the application to test

Example:

InfoAppActive(“Excel.exe”);

The above statement will return 1 when Excel is active or 0 when Excel is inactive.

InfoAppDir

Return the path of current KingView project and store the result into MessageResult. Syntax:

MessageResult= InfoAppDir();

Example:

The following statement returns "C:\Program

Files\Kingview\Example\Kingdemo3”.
DemoPath= InfoAppDir();

InfoAppTitle

Return the Application Title or Windows Task list name of a specified program which is currently running. Syntax:
MessageResult=InfoAppTitle(ProgramEXEName);
Return: String

Parameter	Description
ProgramEXEName	the name of the application to report

Example:
InfoAppTitle;("calc.exe") will return "Calculator"
InfoAppTitle;("excel.exe") will return "Microsoft Excel"

InfoDisk

Returns information about a specific local (or network) disk drive. Syntax:
IntegerResult=InfoDisk(Drive,InfoType,Trigger);
Parameter Description
Drive String or message tag representing drive letter. If the message tag contains more than one character, only the first character of the tag will be used.
InfoType integer representing the message type. The following value is acceptable.

1	Returns the total size (in bytes) of the disk drive.
2	Returns the available space free (in bytes) on the disk drive.

Trigger Execute the InfoDisk function every time the value of *Trigger* changes. *Trigger* can be any tags (not limited to system variables).
Information about the disk drive specified by Drive is returned to IntegerResult.

Example:

The following statement is executed once per minute and returns current value.

InfoDisk("C", 1, $Minute); will return the total size of disk C

InfoDisk("C", 2, $Minute); will return available space free of disk C

InfoFile

Returns information about a specific file or subdirectory. Syntax:

IntegerResult=InfoFile(Filename,InfoType,Trigger);

Parameters	Description
Filename	String representing filename to act on actual string or message tag.
InfoType	Integer representing type of information to retrieve: the following values are acceptable.

1 Does the file exist? Returns 1 if the filename is an actual file or 0 if cannot find the file.

2 File size (in bytes).

3 File Date/Time (in seconds since Jan-1-1970).

4 Number of files that match the filename description. Values larger than 1 will be returned. Only in wildcard searches in which more than one file matching the search criterion is found.

Trigger is Any tag. *Trigger* will execute the InfoFile function every time the value of *Trigger* changes.

Information about the file specified by *Filename* is returned to *IntegerTag*. The *Filename* has to include the full path name to the file, but can include wildcard characters (*, ?).

Example:

The following statement executes every minute and returns the following values:

InfoFile("c:\kingview\touchvew.exe", 1, $minute); will return 1 {file found}.

InfoFile("c:\kingview\touchvew.exe", 2, $minute); will return 634960 (file size)

InfoFile("c:\kingview\touchvew.exe", 3, $minute); will return 736701852 {seconds since 1-1-70}
InfoFile("c:\kingview\*.exe", 4, $minute); will return 4 {found 17 EXE files}

InfoResource

Return various system resource values. Syntax:
IntegerResult=InfoResource(ResourceType,Trigger);

Parameter	Description
ResourceType	Integer representing resources to monitor: the following values are acceptable.

1	Returns the percentage of free space for GDI resources.
2	Returns the percentage of free space for USER resources.
3	Returns the number of bytes of memory currently free
4	Returns the number of tasks currently running.
Trigger	Executes the InfoResources() function every time the value of *Trigger* changes. *Trigger* can be any tag (not limited to system variables).

The particular system resource specified by the integer *ResourceType* is stored in *IntegerResult*.
Example:
InfoResource(1, $minute); will return % available
InfoResource(2, $minute); will return % available
InfoResource(3, $minute); will return available bytes
InfoResource(4, $minute); will return number of tasks
Note:
Return same percentage of free space for GDI and USER resource on Windows NT, which is relative with Windows NT.

Int

Return next integer less than or equal to a specified number. Syntax:
IntegerResult=Int(Number);

Parameter	Description
Number	Any number, long or integer tag.

Example:
Int(4.7); returns 4
Int(-4.7); returns -5

listLoadList

Read items from the list of filename (CSV file) to the specified list control ControlName and replaces all items in it. Only the member' names (string messages) are displayed in the list. Syntax:
listLoadList("ControlName","Filename");

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
Filename	the CSV file for saving the items displayed in the list. You can edit it using Wordpad application.

Example:
The following statement will load the content of the List.csv file to the list called "combo box message".
listLoadList("combo box message", "c:\KingView\list.csv");
Note:
If a full path of the CVS file is not specified, the function will search the file in the path in which KingView is.

listSaveList

Save the items in the list control ControlName to the file FileName. The file is created directly if it doesn’t exist. Syntax:

listSaveList("ControlName","Filename");

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
Filename	the name of the CSV file for saving the items in certain format

Example:

The following statement will save the items in the list “combo box message” to the file "c:\KingView\list.csv".

listSaveList ("combo box message", "c:\KingView\list.csv");

Note:

If a full path of the CVS file is not specified, the file will be created in the path in which KingView is.

listAddItem

Add the string MessageTag to specified list control ControlName. KingView takes the string as a member of the list and defines an index ItemIndex for it. An index ItemIndex is ascending automatically with the step of 1. Syntax:

listAddItem("ControlName","MessageTag");

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
MessageTag	String added to the list control

Example:

The following statement will add the string “Temperature Alarm” to the list control "Alarm Message”.

listAddItem("Alarm Message”, “Temperature Alarm”);

The following statement will add the string “Chocolate Bread” to the list control "Recipe Message”.

listAddItem("Recipe Message”, “Chocolate Bread”);

listClear

Clear all the member items in the specified list control ContorlName. Syntax:

listClear("ControlName");

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.

Example:

The following statement clears all the items in the list "Alarm Message".

listClear("Alarm Message");

listDeleteItem

Delete the item with the index ItemIndex in the specified list control ControlName. Syntax:

listDeleteItem("ControlName",ItemIndex);

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
ItemIndex	a number constant or an integer tag representing the index of the member in the list.

Example:

The following statement deletes the item with the index 1 in the list "Alarm

Message".

listDeleteItem("Alarm Message",1);

The following statement deletes the item with the index 5 in the list "Recipe Message".

listDeleteItem("Recipe Message",5);

listDeleteSelection

Delete the selected item in the list ControlName. Syntax:

listDeleteSelection("ControlName");

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.

Example:

The following statement deletes the selected item in the list "Alarm Message”.

listDeleteSelection("Alarm Message”);

listFindItem

Find the index corresponding with the member string MessageTag and stores the result into the integer tag IndexTag. Syntax:

listFindItem("ControlName","MessageTag",IndexTag);

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
MessageTag	string representing the message that you want to find
IndexTag	integer to save the index

Example:

Take the list items in the file list.csv for example. The following statement stores the index corresponding with the string "Temperature" into the integer tag IndexTag.

listFindItem("combo box message", "Temperature", IndexTag);

listGetItem

Gain the string message of the item with the index ItemIndex and stores the result into the string tag StringTag. Syntax:

listGetItem("ControlName",ItemIndex,StringTag);

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
ItemIndex	any number constant or an integer tag representing the index of the member in the list.
StringTag	string to save the string message

Example:

Take the list items in the file list.csv for example. The following statement stores the item with the index 2 into the string tag StringTag.

listGetItem("combo box message",2, StringTag);

listGetItemCount

This function is used to obtain number of list item of specified control “ ControlName”. Standard format:

listGetItemCount("ControlName");

Parameter:

ControlName: list box control name defined by engineer, which can be Chinese name or English name.

Example:
Count=listGetItemCount("list");
Endow the tag Count with a value of the number of list items in the control “list box”.

ListGetCurSel

This function is used to obtain selected list item ID from specified control “ControlName” (begin from 0), if return value is -1, it means no item is selected in current control. Standard format:
listGetCurSel("ControlName");
Parameter:
ControlName: list box control name defined by engineer, which can be Chinese name or English name.

Example:
ItemIndex=listGetCurSel("list");
Endow tag ItemIndex with a value of ID of selected list item in list box control “list”.

ListSetCurSel

This function is used to setup list item that index number is ItemIndex as current selected item, return value is index number of current selected item of control(begin from 0). If return value is -1, it represents setup isn’t successful (ItemIndex is negative or exceeds number of list item). Standard format:
listSetCurSel("ControlName",ItemIndex);
Parameter:
ControlName: list box control name defined by engineer, which can be Chinese

name or English name.

ItemIndex: numerical value constant or variable, it means index number of list item.

Example:

ID=listSetCurSel("list",1);

Setup current selected item is list item of that index number is 1 in list box control “list”, and endow tag ID with a value of index number of current selected list item.

listGetItemData

Gain the value of the item with the index ItemIndex and stores the result into the integer tag NumberTag. Syntax:

listGetItemData("ControlName",ItemIndex,NumberTag);

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
ItemIndex	any number constant or an integer tag representing the index of the member in the list.
NumberTag	integer to save the gained value

Example:

Take the list items in the file list.csv for example. The following statement stores the value of the item with the index 2 into the tag NumberTag.

listGetItemData("combo box message",2, NumberTag);

listInsertItem

Insert the string StringTag in the list’s position specified via the index ItemIndex. If ItemIndex=1, StringTag it put at the end of the list. Syntax:

listInsertItem("ControlName",ItemIndex, "StringTag");

Parameter Description

ControlName the name of the list control defined by engineer in English or in Chinese.

ItemIndex any number constant or an integer tag representing the index of the member in the list.

StringTag string constant

Example:

The following statement inserts the string “boiler temperature”into the position of the index 2.

listInsertItem("combo box message",2, “boiler temperature”);

listSetItemData

Set the value of the item with the index ItemIndex equal to the tag Number, Syntax:

listSetItemData("ControlName",ItemIndex, Number);

Parameter Description

ControlName the name of the list control defined by engineer in English or in Chinese.

ItemIndex any number constant or an integer tag representing the index of the member in the list.

Number` integer to save the data value

Example:

The following statement sets the value of the item with the index 2 equal to the tag Number.

listSetItemData("combo box message",2, Number);

ListLoadFileName

Displays the file names matched the string "*.ext" in the list. Syntax:

ListLoadFileName("CtrlName","*.ext");

Parameter	Description
ControlName	the name of the list control defined by engineer in English or in Chinese.
ItemIndex	any number constant or an integer tag representing the index of the member in the list.
StringTag	string representing the searched files and supporting the wildcard

Example:

The following statement displays the file name with the suffix .al2 under the path c:\appdir\alarm in the list.

ListLoadFileName(“Alarm file list”, “c:\appdir\alarm\*.al2”);

LoadDriverConfig

Load one or all data of a control field once into the control device according to the driver configuration in the temperature-control table. It is a custom-built function. Syntax:

LoadDriverConfig(sDriverFilePath, nRow);

Parameter	Description
sdriverFilePath	string representing the file name and its path
nRow	integer representing the field number of the parameters loaded. When nRow=9999, load all data of all parameters specified in the driver configuration table.

Return: integer what is 1 when succeeding to load or 0 when failing (failing here means that KingView fails to read data or to check the driver configuration but not to load data into the device).

Example:

1. Load the parameter with the field number 1 specified in the driver configuration.

```
long lRet;
String strDBPath="E:\Database\control.mdb";
long nRow=1;
lRet=LoadDriverConfigure(strDBPath, nRow);
```

2. Load all parameters specified in the driver configuration.

```
long lRet;
String strDBPath="E:\Database\control.mdb";
lRet=LoadDriverConfigure(strDBPath, 9999);
```

LoadText

Load the specified RTF or TXT file into the hypertext control. Syntax:

```
LoadText( "ControlName", "FileName", ".Txt Or .Rtf" );
```

Parameter	Description
ControlName	the name of the hypertext control defined by engineer in English or in Chinese.
Filename	the name of the specified RTF or TXT file which may be edited via Windows Wordpad application.
.Txt Or .Rtf	string representing the file format (.Txt Or .Rtf)

Example:

The following statement loads the file ht1.rtf into the hypertext control called hypertext1.

```
LoadText("hypertext1","D:\Test\recipe\ht1.rtf", ".Rtf");
```

LogE

Return a natural log. Syntax:

```
LogE(tag);
```

Example:
LogE(100); returns 4.605 (loge 100)
LogE(1); returns 0 (loge 1)

LogN

Return the value of the log of x to base n. It is meaningless to take 1 as a base. Syntax:
Result=LogN(Number,Base);

Parameter	Description
Number	any number, integer or long tag
Base	integer to set log base

Example:
LogN(8, 3); returns 1.89279...
LogN(3, 7); returns 0.564...

LogOff

Exit the logon of TouchView. Syntax:
LogOff( );
It has no parameters.

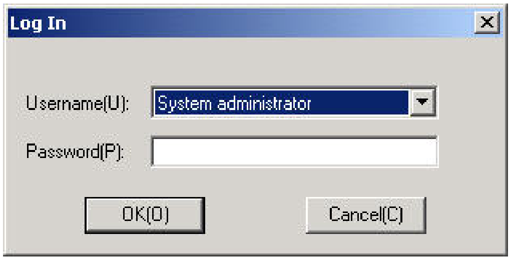
LogOn

Log on TouchView. Syntax:
LogOn( );
It has no parameters.

Example:

Input the following statement in scripts of the button then run the program. Click the button, the Logon dialog box shows.

LogOn( );



You may enter the user name and the password in the box to get operation authority.

LogString

Write user-defined message into KingMessage, which may be implemented by the function Trace(). Syntax:

LogString(String);

Parameter	Description
String	string to write to KingMessage

Example:

LogString("Report Script is Running");

Max

Get the maximum among the given numbers. The acceptable number of parameters

is between 1 and 16. The format is:
Max (Val1, Val2)
Example:
The following statement returns the maximum among var1, var2 and var3.
MaxValue = Max(Max(var1,var2), var3);

Min

Get the minimum among the given numbers. The acceptable number of parameters is between 1 and 16.

Example:
The following statement returns the minimum among var1, var2 and var3.
MinValue=Min(Min(var1,var2),var3);

ModifyTagField

This function is used to modify tag property. Standard format:
ModifyTagField("TagName",Value,Quality,Year,Month,Date,Hour,Minute,Second, millSecond);
Parameter:
TagName: tag name
Value: tag value
Quality: tag quality stamp
Year: year in tag time stamp
Month: month in tag time stamp
Date: date in tag time stamp
Hour: hour in tag time stamp
Minute: minute in tag time stamp
Second: second in tag time stamp

MillSecond: millisecond in tag time stamp

Example
ModifyTagField("R1",999.9,192,2005,10,1,9,0,0,0);

MovePicture

Move the picture to the target position by scripts in runtime. Syntax:
MovePicture(PicName, left, top);

Parameter	Description
PicName	string representing the name of the picture to move
Left	integer representing the coordinate of the left boundary of the target position
Top	integer representing the coordinate of the top boundary of the target position

Example:
The following statement moves the picture to the position of the left boundary=50 and the top boundary=100.
MovePicture(“Prompt Message”,50,100);

PageDown

Turn ahead the page of the messages in KingMessage. Syntax:
PageDown(AlmWin, Lines);
Parameter:
AlmWin: alarm window name
Lines: the number of rows
Example:
The following statement turns ahead the page by the 7 rows of historical alarm

window of the plant. (if there are enough records)
PageDown(historical alarm window of the plant, 7);

PageUp

Turn backward the page of the messages in KingMessage. Syntax:
PageUp(AlmWin, Lines);
Parameter:
AlmWin: alarm window name
Lines: the number of rows

Example:
The following statement turns backward the page by the 7 rows of historical alarm window of the plant. (if there are enough records)
Pageup(historical alarm window of the plant, 7);

PI

Return the value of Pi. Syntax:
RealResult=PI();

Example:
PI(); returns 3.1415926...

PlayAvi

Play the animation file (.avi). Syntax
PlayAvi("CtrlName", filename, option);

Parameter	Description
CtrlName	the name of the control to play the AVI animation.

Filename	string or tag representing the AVI file to play
Option	Option can be one of the following:

a) stop playing
b) play the animation once
c) loop to play the animation until the stop command is called

Example:
The following statement plays the Winner.avi in the control "ctl_avi" once.
PlayAvi("ctl_avi","c:\demo\Winner.avi",1);

PlaySound

Play sound through the computer which has installed audio device driver, sound file is .wav file. Syntax:
PlaySound (SoundName, Flags);

Parameter Description

SoundName Represent string or string tag of sound file. Before sound file name, directory what sound file belonged to can be added or not. The sequence of search for Sound file: current project directory, Windows directory, Windows system directory, directory listed in parameter SoundName. If sound file can’t be found in default directory or directory listed, sound can’t be played

Flags Flags can be one of the following:

0	Stop playing. Note: to use this value can’t stop playing synchronous sound
1	Play sound synchronously
2	Play sound asynchronously
3	Repeat the sound until the next time PlaySound() is called.
4	Buzzer’s alarm
5	Stop playing sound synchronously. Current sound can be played completely, but after that sound, sound playing is stopped.

Example:

1. Play sound file a synchronously, path what file a belongs to is under current KingView project path.
PlaySound("a.wav", 1);
2. Stop playing synchronous sound.
PlaySound("", 1);
Note: when stopping playing sound(include playing synchronously or playing asynchronously), parameter SoundName needn’t be input.

PlaySound2

Play the WAV sound file. Syntax:
PlaySound2(SoundName, DevideId, Flags);

Parameter	Description
SoundName	string representing the file name and its path
DevideId	integer representing the device number to play the sound which is 1 when the device is first sound card or 2 when second sound card
Flags	integer representing the number to play. If it is set 1 when repeat the sound or 0 when play the sound once.

Example:
The statement plays the sound file c:\horns.wav once via the sound card.
PlaySound2(“c:\horns.wav”, 1, 0);

Pow

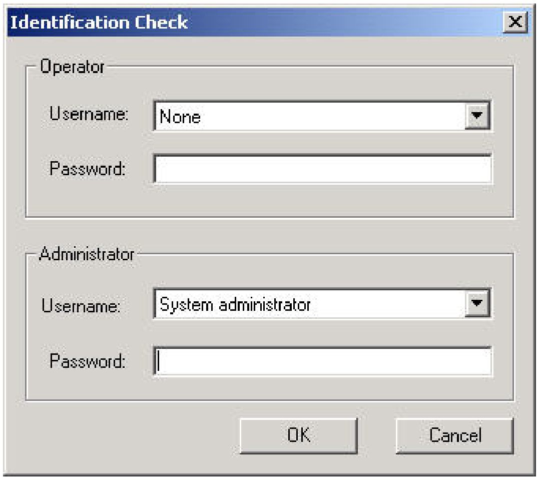
Return the result of any analog number or analog tag raised to a power. Syntax: Result=Pow(x, y);

Parameter	Description
x	base number
y	exponent

Example: Result=Pow(2, 3); returns Result=8.0.

PowerCheckUser

Provide double Identification Check against false operation when user does further operation (as switch on a brake or switch off a brake). You need to enter the username and password of both operator and administrator for further operation in the Identification Check dialog box. The Identification Check dialog box shows as the following figure after executing the function. The user information may be got from the KingView User Config. Syntax:
Result= PowerCheckUser(string OperatorName, string MonitorName);


Identification Check

Operator
Username: None
Password:

Administrator
Username: System administrator
Password:

Parameter	Description
OperatorName	string to store the name of operator
MonitorName	string to store the name of administrator
Result	1-success or 0-fail

Example:
Result= PowerCheckUser(OperatorName, MonitorName);

PreviewWindow

This function is used to realize print preview of KingView picture. Standard format:
PreviewWindow("Window", xScale, yScale , option, xStart, yStart);
Parameter:
Window: KingView window name which needs print preview, that is, KingView picture name, string type.
xScale: the output width (in terms of percentage of the total page width) can be changed using this parameter. Integer or real, if this parameter is 0, parameter option is valid.
yScale: the output height (in terms of percentage of the total page height) can be changed using this parameter. Integer or real, if this parameter is 0, parameter option is valid.
Option: only when xScale and yScale both are 0, it is valid, integer, its value is 0 or 1.
If KingView picture doesn't include bitmap, OCX control, report, or alarm window, when option is 0, keep aspect ratio of picture, thus it can fit the largest print size when printing page(picture hasn't any distortion); when option is 1, zooming picture according to page size(maybe picture has some distortion).
If KingView picture includes one of the following; bitmap, OCX control, report, alarm window, when option is 1, keep aspect ratio of picture, thus it can fit the largest print size when printing page(picture hasn't any distortion); when option is 0, zooming picture according to page size(maybe picture has some distortion).
When picture includes bitmap, OCX control, report, alarm window, parameter option is suggested to set as 1.
xStart: the landscape orientation blank (in terms of percentage of the total page width) can be changed using this parameter. Integer or real, if KingView picture

includes one of the following; bitmap, OCX control, report, alarm window, parameter xStart is invalid.
yStart: the portrait blank (in terms of percentage of the total page height) can be changed using this parameter. Integer or real, if KingView picture includes one of the following; bitmap, OCX control, report, alarm window, parameter yStart is invalid.

Note:
1. After TouchView realizing the function of print preview picture in KingView, if picture is modified in WindowMaker, please preview over again.
2. Can only preview current picture.
3. Print preview must be after TouchView startups and displays picture.
4. If display size is less than picture size (display size and picture size are set in "Picture properties" of WindowMaker), picture preview and picture displayed in TouchView are same. That is, part of a picture is previewed.
5. Suggest setting picture size to be under screen pixel.

Example:
Following button script can be used to preview report print window:
PreviewWindow("report", 0, 0 , 1, 0, 0);
In which, "report" is KingView picture name which includes report

PrintWindow

Print the specified window. Syntax:
PrintWindow("Window", xScale, yScale , option, xStart, yStart);

Parameter	Description
Window	the name of the window to print
XScale	the percentage of the output width to the whole page's one. It can be set to 0 when using default maximum aspect ratio or a specified width.

YScale the percentage of the output height to the whole page’s one. It can be set to 0 when using default maximum aspect ratio or a specified height.

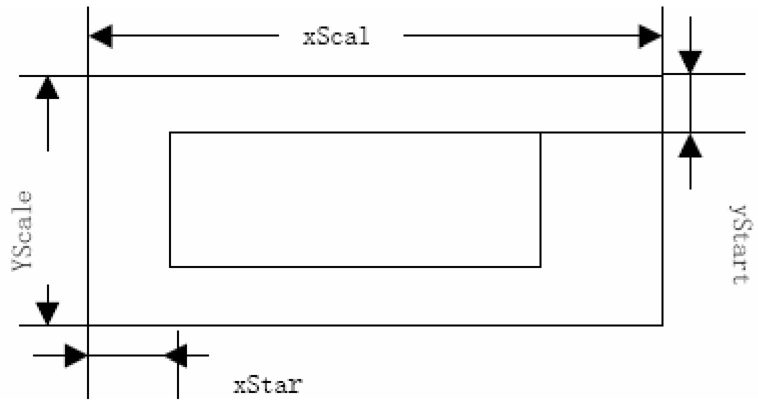
Options discrete (0 or 1) which is valid only when both \Width and Height are 0. It is set as 1 when the window is printed with the original size multiplied under a maximum aspect ratio or 0 when the window is printed with maximum ratio fitting the page. Should set Options as 1 in order to avoid the stretch of the bitmap if there is bitmap in the window.

Xstart the percentage of horizontal blank of the window to print

Ystart the percentage of vertical blank of the window to print

Suggest you set the font of all text fields in the window as True Type in order to print the text correctly. The text on buttons might vanish on printing because the font of the text on buttons is System instead of True Type. In addition, the System font on the screen might be different with printed. If you encounter the case, try to set the larger size of buttons.

The following figure illustrates the relation between xScale, YScale, xStart and yStart.



Example:

PrintWindow("Reaction Workshop",10,10,0,80,80);

pvAddNewRealPt

Add a sample of real time value to the specified temp-control trend. Syntax:
pvAddNewRealPt("ControlName",timeOffset, Value, "commentTag");

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
TimeOffset	the time offset to last sample (interval to last sample). It is 0 for the first sample.
Value	long representing the sample value of temperature. It is often the I/O long tag defined in KingView database.
CommentTag	string or tag for note. Show the prompt message when the cursor moves to the position.

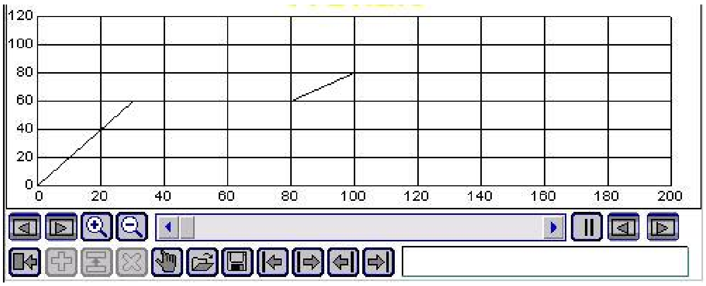
Example1;
The following statement adds a temperature sample of 38 degree to Reaction tank temp-control trend. The sampling time interval is 1 to the last sample. The prompt is “the temperature is 38” when the cursor move over the position.
pvAddNewRealPt("Reaction tank temp-control trend", 1, 38, "the temperature is 38");

Example2:
Given that the Reaction tank real time temperature what receives the temperature values from the lower device, is I/O long tag and TimeString is a string tag. Both they are defined in KingView database,
TimeString=StrFromInt($hour)+ ":"+ StrFromInt($minute)+ ":"+ StrFromInt($second)
The following statement displays the samples of real time temperature in the Reaction tank temp-control trend. The sampling time interval is 10. The prompt in

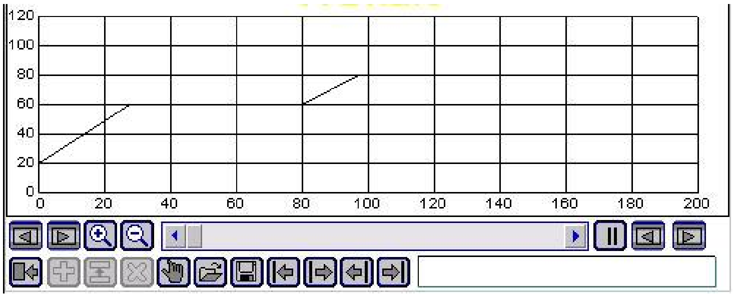
TimeString is displayed when the cursor move over the position.
pvAddNewRealPt(" Reaction tank temp-control trend ", 10, Reaction tank real time temperature, TimeString);

Note:
The set trend is changeable based on the position of the first point of the real time trend.

The first point of a real time trend is shown as the following figure. Add a button and the following statement in its scripts.
pvAddNewRealPt(" Reaction tank temp-control trend ", 10, 20, TimeString);



Then click the button in runtime, the result is shown as the following figure.



The first point of the trend (0, 0) becomes (0, 20).

pvAddNewSetPt

Add a temperature set trend to the specified temp-control trend control, which is applied to free set mode. Syntax:

pvAddNewSetPt("ControlName", TimeOffset, Value);

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
TimeOffset	the time offset to last sample (interval to last sample). It is 0 for the first sample.
Value	Long representing the set value of temperature.

Example:

pvAddNewSetPt(“Reaction tank temp-control trend”, 1, 38);

The following statement adds a temperature set trend to Reaction tank temp-control trend. The set temperature is 38 degree and the sampling time interval is 1 to the last sample.

pvClear

Clear temp-control real time trends or temp-control set trends in the specified temp-control trend control. Syntax:

pvClear("ControlName", IsRealCurve);

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
IsRealCurve	the Boolean tag can be one of the following:
1	Clear temp-control real time trends

0	Clear temp-control set trends

Example:

The following statement clears temp-control set trends in the Reaction tank temp-control trend control.

pvClear("Reaction tank temp-control trend”, 0);

The following statement clears temp-control real time trends in the Reaction tank temp-control trend control.

pvClear("Reaction tank temp-control trend”, 1);

pvGetValue

Return the real time temperature or the set temperature of specified time from the temp-control trend control. If there is no sample at the time, so returns the sample of the time closest to the specified time. Syntax:

pvGetValue("ControlName", timeOffset, TagName, "option");

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
TimeOffset	the time offset to last sample (interval to last sample). It is 0 for the first sample.
TagName	I/O long tag defined in KingView database
Option	string constant can be one of the following:
RealValue	real time temperature
SetValue	set temperature

Example:

The following statement returns a real time temperature of specified time from Reaction tank temp-control trend and stores it into Reaction tank real time temperature. The interval to the origin of coordinates is 5.

pvGetValue(" Reaction tank temp-control trend ",5, Reaction tank real time temperature, "RealValue");

The following statement returns a set temperature of specified time from Reaction tank temp-control trend and stores it into Reaction tank set temperature. The interval to the origin of coordinates is 5.

pvGetValue(" Reaction tank temp-control trend ",5, Reaction tank set temperature, "SetValue");

pvIniPreCuve

Initiate the set trend. Syntax:

pvIniPreCuve("ControlName", "fileName");

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
FileName	string representing the name of the file (.csv). The format of the file content:

SetData

Number of points in the trend

Position of the first point

Speed of rise of temperature, set temperature, heat preservation time for first section

Speed of rise of temperature, set temperature, heat preservation time for second section

Speed of rise of temperature, set temperature, heat preservation time for third section

…

Speed of rise of temperature, set temperature, heat preservation time for n section

Note:

Prohibit appending the blank or other any symbol to the ending of SetData, Number of points in the trend, Position of the first point, and the row of each trend's parameter.

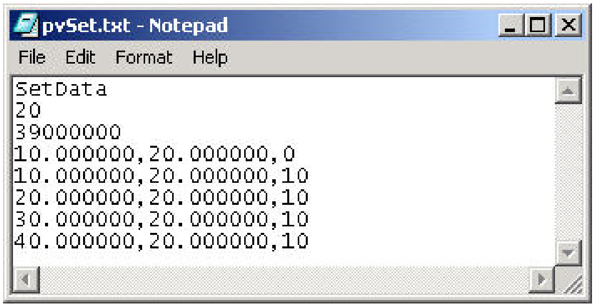
Example:

Given that data in the file pvset.csv are saved in the following format:

SetData
20
39.000000
10.000000,20.000000,0
10.000000,20.000000,10
20.000000,20.000000,10
30.000000,20.000000,10
40.000000,20.000000,10

Edit it with notepad then save as the .csv file, so you may use the function to call the file.

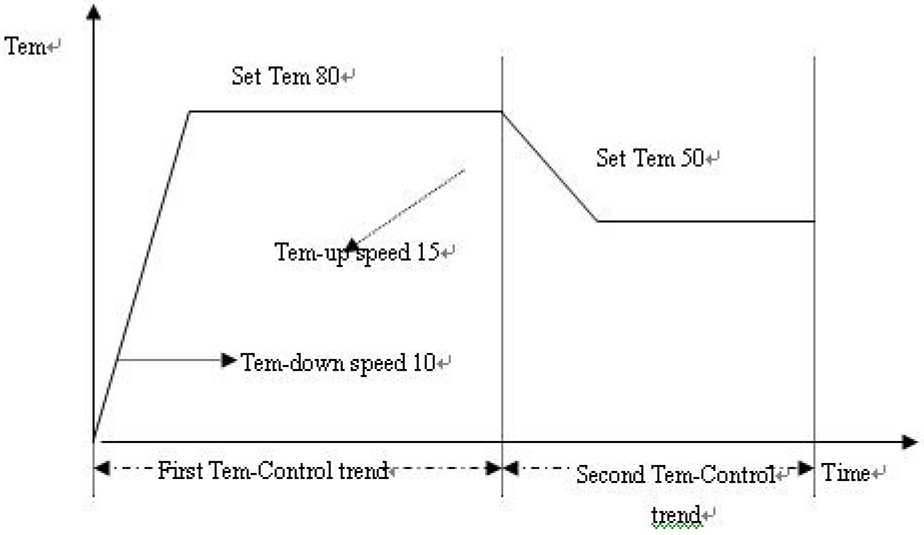
The content of the file edited with notepad is shown as the following figure.



```
SetData
20
39000000
10.000000,20.000000,0
10.000000,20.000000,10
20.000000,20.000000,10
30.000000,20.000000,10
40.000000,20.000000,10
```

The following statement returns the Heater temp-control trend initiated what is shown as the following figure.

pvIniPreCuve("Heater temp-control trend ","c:\pvset.csv");



** Tem stands fior Temperature.

pvLoadData

Read the historical sampling data of the temp-control real time trend or the temp-control set trend from the specified file (.csv).

Syntax:

pvLoadData("ControlName", "FileName", "option");

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
FileName	string representing the name of the file (.csv) what save the historical sampling data of the temp-control real time trend or the temp-control set trend in order: the sequence number of trend, speed of rise of temperature, heat preservation time.

Option	string constant can be one of the following:
RealValue	read historical data of real time trend
SetValue	read historical data of set trend

Example:

Given that data in the file c:\setvalue.csv are saved in the following format:

```
SetData
20
39.000000
10.000000,20.000000,0
10.000000,20.000000,10
20.000000,20.000000,10
30.000000,20.000000,10
40.000000,20.000000,10
```

The following statement displays the temp-control set trend in Reaction tank temp-control trend control.

pvLoadData("Reaction tank temp-control trend", "c:\setvalue.csv", "SetValue");

The following statement displays the historical sampling data of the temp-control real time trend in Reaction tank temp-control trend control.

pvLoadData(" Reaction tank temp-control trend", "fileName", "RealValue");

pvModifyPreValue

Modify a temp-control set trend in the specified temp-control trend control. Syntax:

pvModifyPreValue("ControlName", Index, Tane, SetValue, timeStore);

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
Index	index of temp-control set trend to set
Tane	speed of rise of temperature of temp-control set trend to set

SetValue	temperature of temp-control set trend to set
TimeStore	heat preservation time of temp-control set trend to set

Example:

The following statement set second section of temp-control set trend in Reaction tank temp-control trend as speed of rise=20, set temperature=80, heat preservation=25.

pvModifyPreValue(" Reaction tank temp-control trend", 2, 20,80,25);

pvMoveSlide

Move the slider in specified temp-control trend control to left or right. Syntax:

pvMoveSlide("ControlName", leftORrightSlide, direction, numPt);

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
LeftORrightSlide	1=left slider or 0=right slider
Direction	1 move to left, stop till reaching the limit 0 move to right, stop till reaching the limit
numPt	the point number to move the slider

Example:

The statement moves the right slider of Reaction tank temp-control trend to the left by three sampling points and displays the prompt message at the slider position.

pvMoveSlide("Reaction tank temp-control trend", 0, 1, 3);

pvSaveData

Save the temp-control set trend in specified temp-control trend control to a file and suffixes .csv to the file automatically. Syntax:

pvSaveData("ControlName", "FileName", "option");

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
FileName	string representing the name of the file (.csv) what save the historical sampling data of the temp-control real-time trend or the temp-control set trend in order: the sequence number of trend, speed of rise of temperature, heat preservation time.
Option	string constant can be one of the following:
RealValue	save temp-control real-time curve
SetValue	save temp-control curve which has been set

Example:

The following statement saves the historical sampling data of the temp-control real time trend in Reaction tank temp-control trend control to filename (.csv).

pvSaveData("Reaction tank temp-control trend", "fileName", "RealValue");

The following statement saves the historical sampling data of the temp-control set trend in Reaction tank temp-control trend control to filename (.csv).

pvSaveData("Reaction tank temp-control trend", "fileName", "RealValue");

pvSetLimits

Change the parameters value of specified temp-control trend control, as temperature maximum, temperature minimum, temperature degree, time maximum and time degree. Syntax:

pvSetLimits("CtrlName",TempMax,TempMin,TempScale,TimeMax,TimeScale);

Parameter	Description
ControlName	the name of the temp-control trend control defined by engineer in English or in Chinese.
TempMax	a positive or negative representing a temperature maximum to set

TempMin	a positive or negative representing a temperature minimum to set
TempScale	integer tag representing the temperature degree to set
TimeMax	time maximum to set
TimeScale	integer tag representing the time degree to set

Example:

The statement sets the parameters of temp-control trend control: temperature maximum as value of TempMax, temperature minimum as value of TempMin, temperature degree as value of TempScale, time maximum as value of TimeMax, time degree as value of TimeScale.

pvSetLimits("Reaction tank temp-control trend", TempMax, TempMin, TempScale, TimeMax, TimeScale);

ReadTag

Changes frenquency to acquire IO tag whose RW property is “Read” or “Read/Write” when KingView is running. Syntax:

ReadTag(tagName, freq);

Parameter	Description
TagName	string representing IO tag name defined in Tagname Dictionary.
Freq	integer representing set frequency to acquire from 0 to 3,000,000 in millisecond.

When Freq is 0, to acquire tag once, namely, to acquire tag once per run the function

When Freq is any integer between 1 and 55, to acquire tag each 55 ms.

When Freq is any integer between 56 and 3,000,000, to acquire tag each real set value.

For example:

The following statement set frenquency to acquire “Reactor pot temprature” tag to

0, so to acqure tag data once per run the statement.
ReadTag("Reactor pot temprature", 0);

The following statement set frenquency to acquire "Reactor pot temprature" tag to 1000, so to acquire tag data each 1000 ms.
ReadTag("Reactor pot temprature", 1000);

ReBuildDDE

Rebuilds DDE connection. Syntax:
ReBuildDDE();
The function has no parameter.

ReBuildUnConnectDDE

Rebuild unsuccessful DDE connection. Syntax:
ReBuild UnConnectDDE();
The function has no parameter.

RecipeDelete

Delete the specified recipe in the specified recipe template file. Syntax:
RecipeDelete("filename", "recipeName");

Parameter	description
Filename	string representing the template file name and its path
Recipename	special recipe name in the template file

Note: if the file name and the recipe name are in quotation marks, they represent string constant, if not, they represent string tag.

Example:

The following statement deletes the recipe3 in the recipe template file "Beijing bread plant.csv".

RecipeDelete("C: \recipe\Beijing bread plant.csv","Recipe3");

RecipeInsertRecipe

Insert a new recipe to the specified position of the recipe. A system shows a Select position to insert dialog box after executing the function. you can see all recipes listed in the dialog box then select position for new recipe, click OK. So the new recipe is inserted to the top of selected recipe. Syntax:

RecipeInsertRecipe(filename, InsertRecipeName);

Parameter	description
Filename	string representing the template file name and its path
InsertRecipeName	string representing the name of new recipe

Example:

RecipeInsertRecipe("C:\recipe\ Beijing bread plant.csv","new recipe");

RecipeLoad

Load the specified recipe into the tag in the template file. Syntax:

RecipeLoad("filename","recipeName")

Parameter	description
Filename	string representing the template file name and its path
Recipename	special recipe name in the template file

Note: if the file name and the recipe name are in quotation marks, they represent string constant, if not, they represent string tag of KingView.

Example:

The following statement loads the recipe "fruit bread" of the recipe file "Beijing bread plant.csv" into the tag defined in the template.

RecipeDelete(“C: \recipe\ Beijing bread plant.csv”,“fruit bread”)

RecipeSave

Save the new recipe or changes for the recipe into the recipe template file. Syntax:
RecipeSave(“filename”,“recipeName”)

Parameter	description
Filename	string representing the template file name and its path
Recipename	special recipe name in the template file

Note1: if the file name and the recipe name are in quotation marks, they represent string constant, if not, they represent I/O or memory string tag of KingView.
Note2: the recipe template file must exist. You have to create the file if it doesn’t exist. Or the function will return FALSE.

Example:
The following statement saves changes into the recipe3 of the recipe template file Beijing bread plant.csv. if the file Beijing bread plant.csv didn’t exist, a system would create it automatically.
RecipeSave(“C: \recipe\ Beijing bread plant.csv”,“Recipe3”);

RecipeSelectNextRecipe

Select the next recipe of specified recipe in the recipe template file. Syntax:
RecipeSelectNextRecipe(“filename”, “recipeName”);

Parameter	description
Filename	string representing the template file name and its path
Recipename	string representing specified recipe name

Note: if the file name and the recipe name are in quotation marks, they represent string constant, if not, they represent I/O or memory string tag of KingView.

Example:
The following statement reads the next recipe of recipe3 in the template file.
RecipeSelectNextRecipe(“C: \recipe\ Beijing bread plant.csv”, “recipe3”);
If the string tag RecipeName is empty or not found, returns the first recipe in the file. If RecipeName is last in the file, returns the last recipe.
The recipes are created orderly.

RecipeSelectPreviousRecipe

Select the previous recipe of specified recipe in the recipe template file. Syntax:
RecipeSelectPreviousRecipe(“filename”, “recipeName”);

Parameter	description
Filename	string representing the template file name and its path
Recipename	string representing specified recipe name

Note: if the file name and the recipe name are in quotation marks, they represent string constant, if not, they represent I/O or memory string tag of KingView.

Example:
The following statement reads the previous recipe of recipe3 in the template file.
RecipeSelectPreviousRecipe(“C: \recipe\ Beijing bread plant.csv”, “recipe3”);
If the string tag RecipeName is empty or not found, returns the last recipe in the file. If RecipeName is first in the file, returns the first recipe.
The recipes are created orderly.

RecipeSelectRecipe

Select the recipe typed users in the specified recipe template file. The dialog box shows after executing the function. You can type the specified recipe and store its name into the string tag. Syntax:
RecipeSelectRecipe(“filename”, “recipeNameTag”, “Mess”);

Parameter	description
Filename	string representing the template file name and its path
RecipeNameTag	string tag to store the recipe name
Mess	string prompt message defined by users

Example:

The following statement shows a Select Recipe dialog box and displays the prompt message “Please enter the recipe”. Once you select a recipe in the dialog box, the function stores the recipe name into the tag RecipeName.

RecipeSelectRecipe(“C: \recipe\ Beijing bread plant.csv”, RecipeName, “Please enter the recipe!”;

Report1

Create the real time data report based on the format defined in the source file.

Syntax:

Report1(“SourceFile”, “TargetFile”);

Parameter	description
SourceFile	string constant representing the RTF file to define data report format. It can be edited with wordpad.exe.
TargetFile	string constant representing the file and its path to output the data report

Example:

The following statement creates the real time data report called Real time data.rtf in specified path based on the format defined in the file called report1.rtf.

Report1(“c:\program\files\kingview\example\Kingdemo3\report1.rtf”,“c:\programfiles\kingdemo3\Real time data.rtf”);

Report2

Create the report of historical data in specified time based on the format defined in the source file. Suggest you not use the function because it is created for the edition 5.1. Syntax:

Report2(ST, “SourceFile”, “TargetFile”);

Parameter	description
ST	user-defined start time. You can set it via KingView scripts function HTConvertTime().
SourceFile	string constant representing the RTF file to define data report format. It can be edited with wordpad.exe.
TargetFile	string constant representing the file and its path to output the data report

Example:

ST= HTConvertTime(Year,Month,Day,Hour,Minute,Second);

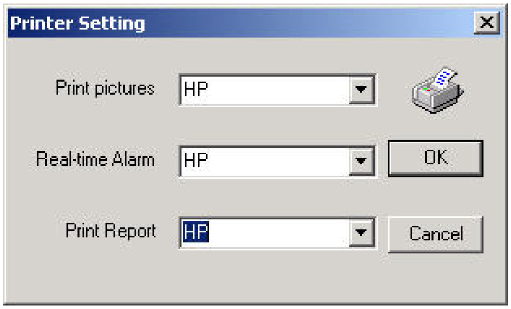
Report2(“c:\program files\kingview\example\ Kingdemo1\report2.rtf”,“c:\program files\kingdemo1\Historical data.rtf”);

The above statement creates the Historical data report called “Historical data.rtf” in specified path based on the format defined in the file called “report2.rtf”.

ReportPrint

Output the data report file to the printer specified in System Config\Printer. Click System Config\Printer in TouchExplorer to show the following dialog box. The printer what the report file is output to is defined in Report Print. Syntax:

ReportPrint(”ReportFile”);



Parameter	Description
ReportFile	the name of the report file to print

Example:
The following statement prints the real time data file “Real time data.rtf”.
ReportPrint(“Real time data.rtf”);

ReportPrint2

Print the report with the printer specified in System Config\Printer. It is the special function for a report. Syntax:
ReportPrint2(String szRptName) or ReportPrint2(String szRptName, EV_LONG|EV_ANALOG|EV_DISC) ;

Parameter	description
SzRptName	the name of the report to print
EV_LONG|EV_ANALOG|EV_DISC	integer or long or discrete. =0: auto-print without showing Print Property dialog box; <>0: show Print Property dialog box

Example:
The following statements print the “Real time data report” automatically.

ReportPrint2(“Real time data report”); or ReportPrint2(“Real time data report”,1);

The following statement shows the Print Property dialog box and prints the “Real time data report” manually.

ReportPrint2(“Real time data report”,0);

ReportPrintSetup

Make the print preview for the specified report. Syntax:

ReportPrintSetup(szRptName);

Parameter	description
SzRptName	the name of the report to preview

Example:

The following statement previews “Real time data report”.

ReportPrintSetup(“Real time data report”);

ReportGetCellString

Get the text in the specified cell of the report. It is special function for a report. Syntax:

ReportGetCellString(ReportName, Row, Col,)

Return string

Parameter	description
ReportName	the name of the report
Row	integer or tag representing row number of cell
Col	integer or tag representing column number of cell

Example:

The following statement stores the text in the cell (across row 2 and column 5) of “Real time data report” into the string tag Text.

Text= ReportGetCellString(“Real time data report”, 2, 5);

ReportGetCellValue

Get the value in the specified cell of the report. It is special function for a report.

Syntax:

ReportGetCellValue(ReportName, Row, Col)

Return long

Parameter	description
ReportName	the name of the report
Row	integer or tag representing row number of cell
Col,	integer or tag representing column number of cell

Example:

The following statement stores the value in the cell (across row 2 and column 4) of “Real time data report” into the long tag Text.

Value= ReportGetCellValue(“Real time data report”, 2, 4);

ReportGetColumns

Return the number of columns in the report. It is the special function for a report.

Syntax:

ReportGetColumns(ReportName);

Parameter	description
ReportName	the name of the report

Example:

The following statement stores the number of columns of “Real time data report” into the tag Number.

Number= ReportGetColumns(“Real time data report”);

ReportGetRows

Return the number of rows in the report. It is the special function for a report.

Syntax:

ReportGetRows(ReportName);

Parameter	description
ReportName	the name of the report

Example:

The following statement stores the number of rows of “Real time data report” into the tag Number.

Number= ReportGetRows(“Real time data report”);

ReportSetRows

This function is special function for report, set specified report row number, standard format:

ReportSetRows(String szRptName, long RowNum);

Parameter:

szRptName: report name

RowNum: row number

Example:

Set report “Real-time Data Report” row number as 1000.

ReportSetRows(“Real-time Data Report”, 1000);

ReportSetColumns

This function is special function for report, set specified report column number, standard format:

ReportSetColumns(String szRptName, long ColumnNum);
Parameter:
szRptName: report name
ColumnNum: column number

Example:
Set report “Real-time Data Report” colunm number as 1000.
ReportSetColumns (“Real-time Data Report”, 1000);

ReportLoad

Read the report file in the specified path into the current report. It is the special function for a report. Syntax:
ReportLoad(ReportName, FileName)
Return integer.
0- success
-3-fali (careful when defining the range of tag to store the return)

Parameter	description
ReportName	the name of the report
FileName	the file name and its path

Example:
The following statement reads the report file called Report.RTL saved in C:\My Documents into the current report and stores the return into the tag ReadFile.
ReadFile= ReportLoad(“Real time data report”,”C:\My Documents\Report.RTL”);

ReportPageSetup

Make the page setup for the report in runtime. Syntax:
ReprotPageSetup(String szRptName);

Parameter	description
SzRptName	the name of the report to setup

Example:
The following statement makes the page setup for “Real time data report”.
ReportPageSetup(“Real time data report”);

ReportSaveAs

Save the report into the specified path with given file name. ReportSaveAs can save report file to the format of .rtl, .xls, .csv, the saved format is decided by the postfix name of the saved file. Syntax:
ReportSaveAs(ReportName,FileName);
Return integer. 0-success

Parameter	description
ReportName	the name of the report
FileName	the file name and its path

Example1:
The following statement saves the report “Real time data report” into the file called Report1.RTL in C:\My Documents and stores the return into the tag SaveFile.
SaveFile=ReportSaveAs(“Real time data report”, "C:\My Documents\Data Report1.RTL”);
The following statement saves the report “Real time data report” into the EXCEL file called Report1.xls in C:\My Documents and stores the return into the tag SaveFile.

Example2:
SaveFile =ReportSaveAs(“Real time data report”, “C:\My Documents\ Data Report 1.xls”);

ReportSetCellString

Set the value of specified cell of the report as the given string. It is the special function for a report. Syntax:

ReportSetCellString(ReportName, Row, Col, Value)

Return	integer
0	success
-1	row or column number is less than zero
-2	wrong report name
-3	fail to set the text

Parameter	Description
ReportName	the name of the report
Row	integer or tag representing row number of cell
Col	integer or tag representing column number of cell
Value	string representing the text to set

Example:

The following statement sets the value of the cell (across row 2 and columns 5) of Real time data report according to the string tag PressurePresentation as the KingView long tag Pressure changes and store the return into the tag SetResult.

SetResult=ReportSetCellString(“Real time data report”, 2, 5, PressurePresentation);

ReportSetCellString2

Set the value of cells in specified block of the report as the given string. It is the special function for a report. Syntax:

ReportSetCellString2(ReportName, StartRow, StartCol, EndRow, EndCol, Value);

Return	integer
0	success

-1	row or column number is less than zero
-2	wrong report name
-3	fail to set the text

Parameter	description
ReportName	the name of the report
StartRow	integer or tag representing start row number of the cells block
StartCol	integer or tag representing start column number of the cells block
EndRow	integer or tag representing end row number of the cells block
EndCol	integer or tag representing end column number of the cells block
Value	string representing the text to set

Example:
The following statement sets the value of the cells in cells block (from row 8 and columns 5 to row 10 and column 7) of Real time data report according to the string tag PressurePresentation as the KingView long tag Pressure changes and store the return into the tag SetResult2.
SetResult2=ReportSetCellString2("Real time data report", 8, 5, , 10, 7, PressurePresentation);

ReportSetCellValue

Set the value of specified cell of the report as the given number. It is the special function for a report. Syntax:
ReportSetCellValue(ReportName, Row, Col,Value)

Return	integer
0	success
-1	row or column number is less than zero
-2	wrong report name
-3	fail to set

Parameter description

ReportName	the name of the report
Row	integer or tag representing row number of the cell
Col	integer or tag representing column number of the cell
Value	string representing the number to set

Example:

The following statement sets the value of the cell (across row 2 and columns 4) of Real time data report according to the tag Pressure as the KingView long tag Pressure changes and store the return into the tag SetResult.

SetResult =ReportSetCellValue(“Real time data report”, 2, 4, Pressure);

ReportSetCellValue2

Set the value of cells in specified block of the report as the given number. It is the special function for a report. Syntax:

ReportSetCellValue2(ReportName, StartRow, StartCol, EndRow, EndCol, Value)

Return	integer
0	success
-1	row or column number is less than zero
-2	wrong report name
-3	fail to set

Parameter	Description
ReportName	the name of the report
StartRow	integer or tag representing start row number of the cells block
StartCol	integer or tag representing start column number of the cells block
EndRow	integer or tag representing end row number of the cell block
EndCol	integer or tag representing end column number of the cells block
Value	string representing the number to set

Example:

The following statement sets the value of the cells in cells block (from row 3 and columns 4 to row 6 and column 7) of Real time data report according to the tag Pressure as the KingView long tag Pressure changes and store the return into the tag SetResult2.

SetResult2=ReportSetCellValue2("Real time data report", 3, 4, 6, 7, Pressure);

ReportSetHistData

Inquire about historical data according to user-defined parameters. It is the special function for a report. Syntax:

ReportSetHistData(ReportName, TagName, StartTime, SepTime, szContent) ;

Parameter	description
ReportName	the name of the report to fill the result of inquiring data
TagName	string in quotation marks representing the tag name to inquire
StartTime	long integer representing start time to inquire. You should convert the StartTime to a long integer before using the function, because it is long integer converted via the function HTConvertTime() on 8:00:00 AM, Jan 1, 1970 as reference time.
SepTime	time interval to inquire with unit of second
SzContent	the range of cells to fill the result

Example:

The following statement inquires about data of the tag Pressure from 8:00:00 hours, May 1, 2001 to now. The interval to inquire is 30 seconds, the range of data to fill is 'a2:a5' (from row 2 and column 1 to row 50 and column 1).

long StartTime; (StartTime is a custom tag)

StartTime=HTConvertTime(2001, 5, 1, 8, 0, 0); ReportSetHistData("historical data report", "pressure", StartTime, 30, "a2:a50");

ReportSetTime

This function is special function for report. Set successive time string for report and cooperate with function ReportSetHisData setting to return historical data time, standard format:

ReportSetTime("ReportName", StartTime, SepTime, "szContent");

Parameter:

ReportName: report name of querying data result.

StartTime: start time of data query, this time need to convert through function HTConvertTime provided by KingView, and it is long data type number using 01/01/1970 08:00:00 as a benchmark, so before using this function to query historical data, firstly start time of query should be converted to long data type number.

SepTime: interval of querying data, second as unit.

szContent: cell area of time character filling in.

Example:

Tine that inserts in report is a time segment starting from 05/01/2001 08:00:00, interval is 30 seconds, filling area of data report is “a2:a100”:

long StartTime; (StartTime is user-defined tag)
StartTime=HTConvertTime(2001, 5, 1, 8, 0, 0);
ReportSetTime (“Historical Data Report”, StartTime, 30, “a2:a100”);

ReportSetHistData2

Inquire about historical data. It is the special function for a report. You only need to set the start row number and the start columns number to fill. A system shows the Historical data inquire dialog box automatically (after the function is processed)

Syntax:

ReportSetHistData2(StartRow,StartCol);

Parameter	description
ReportName	the name of the report

StartRow	integer or tag representing start row number of the cells block
StartCol	integer or tag representing start column number of the cells block

Refer to the chapter Report of the KingView 6.53 User's guide for more.

ReportSetHistData3

This function can control display mode of tags in report when the device related to the tags has fault, quality stamp has error, or TouchView exits. Standard format:

ReportSetHistData3("ReportName", "TagName", StartTime, SepTime, "szContent",bShowInvalidData);

Parameter:

ReportName: report name of querying data result.

TagName: tag name for query

StartTime: start time of data query, this time need to convert through function HTConvertTime provided by KingView, and it is long data type number using 01/01/1970 08:00:00 as a benchmark, so before using this function to query historical data, firstly start time of query should be converted to long data type number.

SepTime: interval of data query, second as unit.

szContent: cell area of result query filling in.

bShowInvalidData: 0, data can't be displayed in report when communication has fault, quality stamp has error, or being shut down computer.; 1, displays the last data record in report.

ReportWebDownload

It's applied in KingView WEB edition. Realizes the following functions:

1. Download the specified report from KingView web server to IE browser.
2. Download the report displayed in IE browser to the local text file.

You only can execute the function via buttons' scripts, because KingView WEB edition in IE browser doesn’t support the back scripts.

Syntax:

ReportWebDownload(ReportName, DownloadType);

Parameter	description
ReportName	string representing the name of the report to download
DownloadType	integer representing the download way can be one among the following values:

0 it is executed in the browser to download the report displayed in IE browser into the .csv file.

1 it is executed in the browser to download the report specified in TouchView of WEB server into the report in IE browser.

2 download directly the report specified in TouchView of WEB server into the report in IE browser. Then save the local report into the .csv file.

Example:

When browsing report in IE client, refresh data in report and make it become updated data in WEB server:

ReportWebDownload(“Real-time Data Report”, 1);

SampleVar

Provide the solution for the I/O tag that you want to acquire intermittently. Must define the acquire frequency of the I/O tag as o Millisecond before calling the function, or it doesn’t work. Execute the function firstly to prepare writing the tag into the KingView data acquiring query then use SampleVarEnd() to acquire data once. Syntax:

SampleVar(TagName);

Parameter	Description
TagName	string representing the name of tag that you want to acquire

Example:

The tag - oxygen content in water in environmental monitoring is acquired upon request, which it is acquired only when it is needed, but not acquired at other time. Define the acquiring frequency of the tag as o Millisecond and use the following statement in the scripts:

SampleVarEnd(“oxygen content in water”);

The above statement prepares writng the tag - oxygen content in water into the KingView data acquire query. Execute SampleVarEnd() to acquire data once..

Note:

1. Both SampleVar() and SampleVarEnd() are used synchronously in scripts. You can use SampleVar()several times then SampleVarEnd() once to acquire.

2. Must set the acquire frequency of the I/O tag as 0 Millisecond to acquire intermittently.

SampleVarEnd

Execute the function after SampleVar() to write the tag into the KingView data acquiring query. Syntax:

SampleVarEnd();

The function has no parameter.

SaveText

Save the text in the hypertext control into the specified RTF or TXT file. Syntax:

SaveText("ControlName", "FileName", ".Txt Or .Rtf");

Parameter	Description
ControlName	the name of the hypertext control defined by engineer in English or in Chinese.

Filename	the RTF or TXT file to save the text You can edit it using Wordpad application.
.Txt Or .Rtf	string to specify the file type

Example:
The following statement saves the text in the hypertext control "hypertext1" into the file calle ht1.rtf.
SaveText("hypertext1","D:\Test\recipe\ht1.rtf", ".Rtf");

SendKeys

You can use the function cooperating with StartApp and ActivateApp to make KingView control remotely other applications. It can start an application (as Excel) then make the application to finish some functions as creating reports trends or recording data. This process can be written by macro language of an application (as Excel). So you can use the function of other applications to enhance the function of KingView.
The function sends keys to application getting the focus. To the receiving application, the keys appear to have been entered from the keyboard. Must make the receiving application getting the focus before calling the function, so you need to call ActivateApp first.

Example:
The following statement sends Control X to Excel. To Excel, it might be command code to inform generating macro commands.
ActivateApp("Excel.exe");
SendKeys("^(X)");
Syntax:
SendKeys(keyT);
Parameter Description

KeyT code of special key whose meaning and usage are similar with of the parameter KeyT in the function Send Keys in Microsoft Excel. Refer to the following table:

Key	Code
{BACKSPACE}or{BS}	Backspace
{BREAK}	BreakCaps
{CAPSLOCK}	aps Lock
{CLEAR}	Clear
{DELETE}or{DEL}	Delete or Del
{DOWN}	Down direction key
{END} End {ENTER}or~	Enter
{ESCAPE}or{ESC}	Esc(Escape)
{HOME}	Home
{INSERT}	Insert
{LEFT}	Left direction key
{NUMLOCK}	Num Lock
{PGDN}	Page Down
	Page up
	Print Screen
	Right direction key
{SCROLLLOCK}	Scroll Lock
	Tab
	Up direction key
{F1}through{F12}	Function keys F1 through F12

You can define a Key command with uppercase and lowercase as well as use it cooperating with the following keys:

Key	Code
+	Shift
^	Ctrl
%	Alt

Example:

The following statement sends a key query to copy selected area.
SendKeys("^{insert}");
For indicating that holding down Shift, Ctrl or Alt while pressing else keys, you can put else keys in the bracket. The following statement first sends the key message: Alt-t. Alt-f and Alt-r, then sends Enter. The mark % represents Alt. Since the letters following Alt is in the bracket, it seems that the key is pressed while Alt is pressed.
SendKeys("%(TFR)~");
The following statement first sends the string secret then presses enter.
SendKeys("secret~");
Since the characters +, ^ and & have special meaning, for getting these characters themselves instead of the special meaning, should put the characters in the {}, for example: SendKeys("A{+}B"). This statement sends the string A+B.

SetNetNodeValid

This function is used to shield network node manually, disconnects for website which is shielded, not to try again, not to exchange data again, it can’t connect unless being activated over again. Standard format:
SetNetNodeValid(szNodeName, bFlag);
Parameter:
szNodeName: node name
bFlag: 0, shield 1, activate When it return 0, it means success

Example:
SetNetNodeValid("king", 0);

SetRealDBForBool

This function is used to create operation event of discrete variables. Except script for buttons, in Kingview script, modifying the values of variables can create

operation event.
transferring form:
SetRealDBForBool("VarName",Value);

Parameter	Description
VarName	Name of variable, discrete format
Value	Value of variable

Example:
In the “While running” of scripts , input:
SetRealDBForBool ("\\local\Valve",1)
Set the analog value of \\local\Valve to be 1.
when the screen displays, the operation event of analog variable \\local\Valve is created.

SetRealDBForFloat

This function is used to create an operation event about real tag. Except button script, it can’t create operation event when modifying tag value using KingView script, while using this function and function SetRealDBForInt below can solve this problem. Standard format:
SetRealDBForFloat("VarName",Value);
VarName is tag name, Value is tag value.

Example:
Input the following in script of “on Startup”
SetRealDBForFloat ("\\local\liquid depth", 1.5)
Set real tag "\\local\liquid depth" as 1.5.
When picture startups, create an operation event of real tag \\local\liquid depth.

SetRealDBForInt

This function is used to create an operation event about integer tag. Except button script, it can't create operation event when modifying tag value using KingView script, while using this function and function SetRealDBForFloat above can solve this problem. Standard format:
SetRealDBForInt("VarName",Value);
VarName is tag name, Value is tag value.

Example:
Input the following in script of "on Startup"
SetRealDBForInt (\\local\row, 10);
Set integer tag "\\local\row" as 10.
When picture startups, create an operation event of integer tag \\local\row.

SetRealDBForString

This function is used to set the real string value of the variable. Tranferring form:
SetRealDBForString ("VarName",”Value”);

Parameter	Description
VarName	Name of variable
Value	string to be set

Note: the variable can only be string one.

Example: Build a button. And the script of popup event connect "Key Up":
SetRealDBForString ("\\local\v ","abc");
Set the value of string variable of "\\local\v“ to be abc.

SetTrendPara

Create a button to show a dialog box in runtime for changing the parameters of the historical trend as start time, data length, start point and end point of y-axis. Syntax:

SetTrendPara(Trend Tag);

Parameter:

Trend Tag historical curve name

Example: SetTrendPara(historical trend);

Sgn

Judge the sign of the number (positive, zero or negative). Syntax:

IntegerResult=Sgn(Number);

Parameter	Description
Number	any number, KingView long or integer tag

Returns 1 when the number is positive or –1 when negative or 0 when zero.

Example: Sgn(425); returns 1

Sgn(0); returns 0

Sgn(-37.3); returns -1

ShowNavigateWindow

This function is used to display and hide the navigate window transferring format:

ShowNavigateWindow(nCmdShow);

Parameter	Description
nCmdShow	Control the displaying and hiding of the navigate window nCmdShow=0, hide the navigetewindow; nCmdShow=1, display the navigetewindow.

Example: Display the navigate window:

ShowNavigateWindow(1);

ShowPicture

Show the picture. Syntax:
ShowPicture("PictureName");
Example: ShowPicture("Reactor Workshop");

Sin

Return the sine of the tag. Syntax:
Sin(tag)
Example: Sin(90); returns 1
Sin(0); returns 0

SQLAppendStatement

Append a statement to the end of the function SQLSetStatement(). Syntax:
[ResultCode=]SQLAppendStatement(DeviceID,"SqlStatement");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
SqlStatement	SQL statement to append

Example;
The additional condition is Age (column) = 24.
SQLAppendStatement(DeviceID, “where Age=24”);
Note: the function deletes the blanks in front of the statements automatically, so the same tag or connector cannot be separated in the two statements.

SQLClearStatement

Release the resources associated with SQLHandle. Syntax:

[ResultCode=]SQLClearStatement(DeviceID,SQLHandle);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
SQLHandle	A memory integer brought by SQLPrepareStatement()

SQLClearTable

Delete all records in the table but keeps the table. Syntax:
[ResultCode=]SQLClearTable(DeviceID, “TableName”);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	The name of the database you want to access

Example:
The following statement deletes all records in the table kingview.
[ResultCode=]SQLClearTable(DeviceID, “kingview”);

SQLCommit

Define the end of a group of transaction commands. The group of commands performed between the SQLTransact() command and the SQLCommit() command is called a transaction set. A transaction set is handled like a single transaction. After the SQLTransact() command is issued, all subsequent operations will not be executed until the SQLCommit() command is issued.Syntax:
SQLCommit(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

Note: the time to process SQLCommit() is long as a number of commands are executed at the same time.

Example:

The following statements implement three inserts.

```
SQLTransact(DeviceID);
SQLInsertPrepare(DeviceID,TableName,BindList,SQLHandle);
SQLInsertExecute(DeviceID,BindList,SQLHandle);
SQLInsertExecute(DeviceID,BindList,SQLHandle);
SQLInsertExecute(DeviceID,BindList,SQLHandle);
SQLCommit(DeviceID);
```

SQLConnect

Connect KingView to the database. Syntax:

```
[ResultCode=]SQLConnect(DeviceID, "dsn=;uid=;pwd=");
```

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
"dsn=;uid=;pwd ="	Connection statement

"dsn=;uid=;pwd=" has the fowling format: “DSN=data source name [;attribute= value[;attribute = value]…”

Example:

The following statement connects KingView to the database Pubs in SQL server called wang with username sa (without password).

```
[ResultCode=]SQLConnect(DeviceID,“DSN=wang;DATABASE=pubs;UID=sa;
PWD=”);
```

Properties	Value
DSN	Data source configured in ODBC
UID	Logon ID

PWD	Password match case
DATABASE	The database you want to access

SQLCreateTable

Create a table in the database based on the table type in the table template. Syntax:
[ResultCode=]SQLCreateTable(DeviceID,"TableName", "TemplateName");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	The table name you want to create
TemplateName	The name of table template

Example:
The following statement creates a table named kingview with the type in table1.
SQLCreateTable(DeviceID, “kingview”, “table1”);

SQLDelete

Delete one record or more records. Syntax:
[ResultCode=]SQLDelete(DeviceID, "TableName", "WhereExpr");
Note: "WhereExpr" cannot be empty.

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	The table name

WhereExpr	Defines a conditions to find valid rows for the funciton Note: if a column name is string type, the expression must be in single quotes. The following example will select all rows whose Name column contains the value Asia: Name='Asia' The following example will select all rows containing Age from 100 to 199: Age>=20 and Age<30

Example: the following statement deletes all records (in the table kingview) whose LogNo column contains a value that is equal to 11.
SQLDelete(DeviceID, “kingview”, “LogNo=11”)

SQLDisconnect

Disconnect the user from the database. Syntax:
[ResultCode=]SQLDisconnect(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

SQLDropTable

Delete a table (including structure). Syntax:
[ResultCode=] SQLDropTable(DeviceID, TableName);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	The table name

SQLEndSelect

It’s used after a SQLSelect() function to free resources that were being used to store the Results Table. Syntax:

[ResultCode=]SQLEndSelect(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

SQLErrorMsg

Return the string representing the error interrelated with ResultCode. Syntax:

SQLErrorMsg(ResultCode,buf);

Parameters	Description
ResultCode	Most of SQL Functions return an integer, which is 0 when the function is successful to be called or negative when failing.
buf	Part of error’s prompt

Refer to the KingView SQL Server User’s Guide for more.

Example;

ErrorMsg = SQLErrorMsg(ResultCode,buf);

Buf is corresponding with an I/O string tag in KingView, it only displays part of error’s prompt, but most of prompt is showen in info window.

SQLExecute

Execute the SQL statements. Syntax:

[ResultCode=]SQLExecute(DeviceID, "BindList" , SQLHandle);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

BindList	Bind list specifying the relation between KingView tags and the columns of the table
SQLHandle	If the function is called after executing SQLPrepareStatement(), the parameter returns an integer. If not available handle, it returns 0.

Note: if there is no available handle, the function only is used once. If the function is called after executing SQLPrepareStatement(), it can be use several times.

SQLFirst

Return the first record in result set gained via SQLSelect(). Syntax:
[ResultCode=] SQLFirst (DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

SQLGetRecord

Return the record of the specified sequence number in selection set. Syntax:
[ResultCode=]SQLGetRecord(DeviceID, RecordNumber);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
RecordNumber	Sequence number

EXAMPLE;

The following statement returns the third record in selection set.
SQLGetRecord(DeviceID, 3);

SQLInsert

Insert a new record in the table according connection defined in bind list. Syntax:
[ResultCode=]SQLInsert(DeviceID, "TableName", "BindList");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	Table name
BindList	Bind list

Example;
The following statement inserts a new record in the table called kingview.
SQLInsert(DeviceID, “kingview”, “bind1”);

Note:

The following functions can replace the standard SQLInsert() to insert quickly: SQLInsetPrepare(), SQLInsertExecute(), SQLInsertEnd(). SQLInsert() is a function containing inserting and ending statements. So in fact the whole process is done again when executing SQLInsert(). But you can execute SQLExecute() several time after, SQLInsertPrepare(), use SQLInsertEnd() finally.

SQLInsertEnd

Release the resource. Syntax:
[ResultCode=]SQLInsertEnd(DeviceID, SQLHandle);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
SQLHandle	Handle assigned by SQLInsertPrepare()

SQLInsertExecute

Execute the inserting statement. Syntax:
[ResultCode=]SQLInsertExecute(DeviceID,"BindList", SQLHandle);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
BindList	Bind list
SQLHandle	Handle assigned by SQLInsertPrpare

SQLInsertPrepare

Generate and prepare an inserting statement but not executes it. The function is processed to generate a handle.
[ResultCode=]SQLInsertPrepare(DeviceID,"TableName", "BindList", SQLHandle);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	Table name
BindList	Bind list
SQLHandle	Handle

SQLLast

Return the last record in result set gained via SQLSelect(). You must execute SQLSelect() first before SQLLast(). Syntax:
[ResultCode=]SQLLast(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

Example:SQLLast(DeviceID);

SQLLoadStatement

Read the statement in the file, similar to SQLSetStatement(). You can also use SQLAppendStatement() to append the statements. Each file only may contain a statement. Syntax:

[ResultCode=]SQLLoadStatement(DeviceID, "OutputFile");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
OutputFile	File name

Example:

SQLLoadStatement(DeviceID, “C:\InTouchAppname\SQL.txt”);

The following is contained in the file SQL.txt:

Select ColumnName from TableName where ColumnName>100;

SQLNext

Select the next record in result set gained via SQLSelect(). Syntax:

[ResultCode=]SQLNext(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

Example:

SQLNEXT(DEVICEID);

SQLNumRows

Return the number of rows in result set gained via SQLSelect(). Syntax:

SQLNumRows(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

Example:
NumRows=SQLNumRows(DeviceID);

SQLPrepareStatement

Return the handle for statements specified SQLSetStatement() or SQLLoadStatement() and SQLAppendStatement(). Syntax:
[ResultCode=]SQLPrepareStatement(DeviceID, SQLHandle);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
SQLHandle	Handle

Example:
The following statement returns the handle and stores it into the memory tag.
SQLPrepareStatement(DeviceID, SQLHandle);

SQLPrev

Select the previous record in result set gained via SQLSelect(). Syntax:
SQLPrev(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

SQLRollback

Undo the last command not submitted following SQLTransact(). Syntax:
[ResultCode=]SQLRollback(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

Example;

SQLTransact(DeviceID);

SQLInsertPrepare(DeviceID, “kingview”, “bind1”, handle);

SQLInsertExecute(DeviceID, “bind1”, handle);

SQLInsertEnd(DeviceID, handle);

/* If SQLCommit(DeviceID) is executed here, the above statements are executed. */

SQLRollback(DeviceID);

/*undo the statements following SQLTransact()/

SQLSelect

Access a database and gets a special selection set. The records in the set may be accessed by SQLFirst(), SQLNext() , etc. Syntax: [ResultCode=]SQLSelect(DeviceID,"TableName","BindList", "WhereExpr","OrderByExpr");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	Table name
BindList	Bind list
WhereExpr	Define conditions to find valid rows for the function Note: if a column name is string type, the expression must be in single quotes. The following example will select all rows whose Name column contains the value Asia: Name=’Asia’ The following example will select all rows containing Age from 100 to 199:

	Age>=20 and Age<30
OrderByExpr	Define the columns to order and directions. Only the column names may be used for order, Expression: ColumnName[ASC|DESC]. Next order the columns "Temperature" by ascending: "Temperature ASC" May use multiple expression in order, as: "Temperature ASC, time DESC"

Examples about WhereExpr:
String example:
"Ser_No='abcd'"
Use the statement like in string:
"Ser_No like ab%"
Note: % represents the character of wide sense.
Use and to connect string and analog:
"Ser_No='abcd' and Number=150"
Note: you should call SQLEndSelect () to release the resource after selection set assigned by SQLSelect().
Example:
The following statement selects the rows whose Ser-No column contains the value abcd ordered in ascending of Temperature. The relation between tags is defined in bind1.
SQLSelect(DeviceID, "kingview", "bind1", "Ser_No='abcd', "Temperature ACS");
Example:
string WhereExpr="Date like+'%"+FindDate+"%'";

SQLSelect(DeviceID, "Data Table", "Daily Report", WhereExpr, "");

Example:

String str1="boiler number="+""+\\Local\test+"";

SQLSelect(DeviceID, "table2", "Bind2", str1, ""); //SQL tag condition search, \\Local\test: string tag

String strtime=StrFromInt(inttime, 10);

//inttime is an integer. You should convert string first. str1="Times="+""+strtime+"";

SQLSelect(DeviceID, "data search", "BIND", str1, "");

Example:

The following statement selects all rows in the table kingview.

SQLSelect(DeviceID, “kingview”, “bind1”, “”, “”);

SQLSelectTop()

This function is used to query n records from the first record which satisfies the query condition in the SQL database, standard format:

SQLSelectTop(DeviceID, "TableName", "BindList", "WhereExpr", "OrderByExpr", "TopExpr");

Parameter:

DeviceID: connection number created by SQLConnct()

TableName: table name

BindList: Bind List name

WhereExpr: condition clause

OrderByExpr: order clause

TopExpr: n records from the first record which satisfies the query condition and order condition.

SQLSetParamChar

Assign the specified parameter to the string. Syntax:
[ResultCode=]SQLSetParamChar(SQLHandle, ParameterNumber, "ParameterValue", MaxLen);

Parameters	Description
SQLHandle	Handle assigned by SQLPrepareStatement()
ParameterNumber	Sequence number of parameters in statements
ParameterValue	Value to assign (may be Kingview tag)
MaxLen	Maximum length of column relative with parameter

SQLSetParamDate

Assign the date value to the specified parameter. Syntax:
[ResultCode=]SQLSetParamDate(SQLHandle,ParameterNumber, ParameterValue);

Parameters	Description
SQLHandle	Handle
ParameterNumber	Sequence number of parameters in statements
ParameterValue	Value to assign (may be Kingview tag)

SQLSetParamDateTime

Assign the date and time values to the specified string parameter. Syntax:
[ResultCode=]SQLSetParamDateTime(SQLHandle, ParameterNumber, ParameterValue, Precision);

Parameters	Description
SQLHandle	Handle
ParameterNumber	Sequence number of parameters in statements
ParameterValue	Value to assign (may be Kingview tag)
Precision	The number of string used byParameterValue

SQLSetParamDecimal

Assign the decimal value to the specified integer parameter. Syntax:
[ResultCode=]SQLSetParamDecimal(SQLHandle, ParameterNumber, "ParameterValue", Precision, Scale);

Parameters	Description
SQLHandle	Handle
ParameterNumber	Sequence number of parameters in statements
ParameterValue	Value to assign (may be Kingview tag)
Precision	The number of string used byParameterValue

SQLSetParamTime

Assign the time value to the specified parameter. Syntax:
[ResultCode=]SQLSetParamTime(SQLHandle,ParameterNumber, ParameterValue);

Parameters	Description
SQLHandle	Handle
ParameterNumber	Sequence number of parameters in statements
ParameterValue	Value to assign (may be Kingview tag)

SQLSetParamFloat

Assign the value to the specified floating-point parameter. Syntax:
[ResultCode=]SQLSetParamFloat(SQLHandle,ParameterNumber, ParameterValue);

Parameters	Description
SQLHandle	Handle
ParameterNumber	Sequence number of parameters in statements

ParameterValue	Value to assign (may be KingView tag)

Example;

SQLSetStatement(ConnectionID, “select * from kingview where highth=?”)

SQLPrepareStatement(ConnectionID, handle);

SqlSetParamFloat(handle, 1, var1) /*1 represents the first ?, var1 is the tag of KingView.*/

SQLSetParamInt

Assign the value to the specified integer parameter. Syntax:

[ResultCode=]SQLSetParamInt(SQLHandle,ParameterNumber, ParameterValue);

Parameters	Description
SQLHandle	Handle
ParameterNumber	Sequence number of parameters in statements
ParameterValue	Value to assign (may be KingView tag)

Example:

SQLSetStatement(ConnectionID, “select * from kingview where agg=?”);

SQLPrepareStatement(ConnectionID, handle);

SqlSetParamInt(handle, 1, var2); /*1 represents the first ?, var2 is the tag of KingView.*/

SQLSetParamNull

Assign Null to the specified parameter. Syntax:

[ResultCode=]SQLSetParamNull(SQLHandle, ParmeterNumber, ParameterType, Precision, Sclae);

SQLSetStatement

Start a SQL statement buffer. Syntax:

[ResultCode=]SQLSetStatement(DeviceID, "SQLStatement");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
SQLStatement	SQL statement

Example:

SQLSetStatement(DeviceID, “Select LotNo, LotName from LotInfo”);

Example:

SQLSetStatement(DeviceID, “select Speed from kingview”);

SQLExecute(DeviceID, “BIND”, 0);

The above example doesn’t use SQLPrepareStatement() to prepare a handle, so sets 0 to the handle.

Example:

SQLSetStatement(DeviceID, “select Speed from kingview”);

SQLPrepareStatement(DeviceID, handle);

SQLExecute(DeviceID, “BIND”, handle);

SQLClearStatement(DeviceID, handle);

SQLTransact

Define a group of statements. The group is not executed until SQLCommit(). Syntax:

[ResultCode=]SQLTransact(DeviceID);

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()

SQLUpdate

Modify the record in database using the KingView tag value. Syntax:

[ResultCode=]SQLUpdate(DeviceID, "TableName", "BindList", "WhereExpr");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
TableName	Table
BindList	Bind list
WhereExpr	Defines a conditions to find valid rows for the function Note: if a column name is string type, the expression must be in single quotes. The following example will select all rows whose Name column contains the value Asia: Name='Asia' The following example will select all rows containing Age from 100 to 199: Age>=20 and Age<30

Example:

The following statement updates all row whose age column contains the value equal to 20 using the tag kingview.

SQLUPDATE(DEVICEID, "KINGVIEW", "BIND1", "AGE=20");

SQLUpdateCurrent

Update the current record in database using the KingView tag. Syntax:

[ResultCode=]SQLUpdateCurrent(DeviceID, "TableName");

Parameters	Description
DeviceID	The connection number assigned by SQLConnct()
"TableName"	Bind list

Sqrt

Return the square root of the tag. Syntax:

Sqrt(tag name or number);

The type of the tag may be integer, analog, discrete. The returned value is valid when the tag is positive, or invalid when the tag is negative.

StartApp

Start an application in other windows. Enter a full path of the application for ensuring starting application successfully.

StartApp("parameter of command row"); or StartApp("Application");

The following statement starts Excel and opens the spreadsheet report.xls automatically.

StartApp(“c:\programfiles\microsoftoffice\office\excel report.xls”);

If you don’t want to open the file, you may use the following statement.

StartApp("c:\program files\Microsoft office\office\excel ");

StrASCII

Return the ASCII value of the first letter of specified string tag. Syntax:

IntegerResult=StrASCII(Char);

Parameter	Description
Char	a character in the alphabet or String tag

The function stores the ASCII value of the first letter in Char into IntegerResult. Only single character is tested or affected. If the character in the string tag is more than one, only the first character is tested.

Example:

StrASCII("A"); retunrs 65

StrASCII("A Mixer is Running"); returns 65

StrASCII("a mixer is running"); returns 97

StrChar

Return the character represented by specified ASCII. Syntax:

MessageResult=StrChar(ASCII);

PARAMETER	DESCRIPTION
ASCII	ASCII code or string tag of KingView

The function stores the character into MessageResult. The advantage of the function is to add the character into a string tag without keyboard.

Example:

The following statement append [CR] and [LF] to the end of MessageTag and stores the result into ControlString.

ControlString=MessageTag+StrChar(13)+StrChar(10);

It is used to create the control code of peripheral equipment (as printer or modem) to insert characters with ASCII code beyond 32 and 126.

StrFromInt

Convert an integer value to a string in specified system (as decimal, octal). Syntax:

MessageResult=StrFromInt(Integer,Base);

Parameter	Description
Integer	Number or a integer tag to convert
Base	Number or an integer tag representing specified system you want to convert the number to

The result is stored into MessageResult

Example:

StrFromInt(26, 2); returns "11010"

StrFromInt(26, 8); returns "32"

StrFromInt(26, 16); returns"1A”

StrFromReal

Convert a long value to a string in floating-point system or exponential system.
Syntax:
MessageResult=StrFromReal(Real, Precision,Type);

Parameter	Description
Real	number to convert based on Precision and Type

The result is stored into MessageResult.

Precision	Decimal digits to display
Type	display way can be one of the following:

“f”	floating-point
“e”	exponential system by e
“E”	exponential system by E

Example:
StrFromReal(263.355, 2,"f"); returns "263.36"
StrFromReal(263.355, 2,"e"); returns "2.63e2"
StrFromReal(263.55, 3,"E"); returns "2.636E2"
Refer to StrToReal().

StrFromTime

Convert the time value (in seconds from 16:00 hours, Dec 31, 1969) to a string.
Syntax:
MessageResult=StrFromTime(SecsSince1_1_70, StringType);

Parameter	Description
SecsSince1_1_70	the type specified by StringType to convert to

The result is stored into MessageResult

StringType	display way can be one of the following:
1	to display date with the format similar to of Windows control

panel

2 to display time with the format similar to of Windows control panel

3 to display date and time

StrFromTime(86400, 1); returns "1/2/70"

StrFromTime(86400, 2); returns "12:00:00 AM"

StrFromTime(86400, 3); returns " 1/2/70 12:00:00 AM "

StrInStr

Return the position when SearchFor appears in Text first time. Syntax:

IntegerResult=StrInStr(Text,SearchFor,StartPos,CaseSens);

Parameter	Description
Text	text to find SearchFor in. if it contains several SearchFor, the first position is returned to IntegerResult.
SearchFor	the text for search
StartPos	integer representing the position to begin to find
CaseSens	can be one of 0 or 1 representing if match case

Example:

StrInStr("The mixer is running", "mix", 1, 0); //returns 5

StrInStr("Today is Thursday", "day", 1, 0); //returns 3

StrInStr("Today is Thursday", "day", 10, 0); //returns 15

StrInStr("Today is Veteran's Day", "Day", 1, 1); //returns 20

StrInStr("Today is Veteran's Day", "Night", 1, 1); //returns 0.

StrLeft

Return the some characters at the most left of the specified string tag. Syntax:

MessageResult=StrLeft(Text,Chars);

Parameter Description

TEXT	STRING OR STRING TAG
Chars	string or integer tag representing the number of the characters to return. If Char is set as 0, all character in the string will be returned.

Example:

StrLeft("The Control Pump is On", 3); //returns "The"

StrLeft("Pump 01 is On", 3); //returns "Pump"

StrLeft("Pump 01 is On", 96); //returns "Pump 01 is On"

StrLeft("The Control Pump is On", 0); //returns "The Control Pump is On".

StrLen

Return the length of specified string. Syntax:

IntegerResult=StrLen(Text);

Parameter	Description
Text	string or string tag

The length of the string is stored into IntegerResult. All characters in the string including these that cannot be displayed on screen, are counted

Example:

StrLen("Twelve percent"); //returns 14

StrLen("12%"); //returns 3

StrLen("The end. [CR]"); //returns10, [CR] represents Enter key- ASCII 13.

StrLower

Convert all uppercase specified characters into lowercase but don’t affect lowercase characters, tabs, digits and other special characters in it. Syntax:

MessageResult=StrLower(Text);

Parameter	Description
Text	string or string tag

Example:

StrLower("TURBINE"); //returns "turbine"
StrLower("22.2 Is The Value"); //returns "22.2 is the value."

StrMid

Extract a substring in the string beginning from specified position. It is different with StrLeft() and StrRight(), because it allows you to specify the beginning and ending position to extract. Syntax:

MessageResult=StrMid(Text,StartChar,Chars);

Parameter	Description
Text	string or string tag
StartChar	beginning position to extract
Chars	the number of substring

Example:

StrMid("The Furnace is Overheating",5,7,); //returns "Furnace"
StrMid("The Furnace is Overheating",13,3); //returns "is "
StrMid("The Furnace is Overheating",16,50); //returns "Overheating"

StrReplace

Replace the specified substring in the string. You can use it to get the string tag and replace character, word or phrase. Syntax:

MessageResult = StrReplace(Text,SearchFor, ReplaceWith, CaseSens, NumToReplace, MatchWholeWords);

Parameter	Description
Text	string that you want to modify
SearchFor	replaced substring to find in Text
ReplaceWith	replacing substring
CaseSens	0 or 1 to specified if match case
NumToReplace	number to replace, 0 = replace all

MatchWholeWords	0 or 1 to specified if match whole word only

Example:

StrReplace("In From Within","In","Out",0,1,0) //returns "Out From Within" (only replace one)

StrReplace("In From Within","In","Out",0,0,0) //returns "Out From WithOut" (replace all)

StrReplace("In From Within","In","Out",1,0,0) //returns "Out From Within" (replace all and match case)

StrReplace("In From Within","In","Out",0,0,1) //returns "Out From Within" (replace all and match whole word only)

StrReplace() can not recongnize specila characters as @#$%&*(), and considers them as separator. For example, if StrReplace (abc#,abc#,1234,0,1,1) is processed, no replace happens. # is considered as separator instead of character.

StrRight

Return some characters at the most right of the specified string tag. Syntax:

MessageResult=StrRight(Text,Chars);

Parameter	Description
Text	string or string tag
Chars	string or integer tag representing the number of the characters to return. If Chars is set as 0, all character in the string will be returned.

Example:

StrRight("The Pump is On", 2); returns "On"

StrRight("The Pump is On", 5); returns "is On"

StrRight("The Pump is On", 87); returns "The Pump is On"

StrRight("The Pump is On", 0); returns "The Pump is On".

StrSpace

Create some blanks in string tag or expression. Syntax:

MessageResult=StrSpace(NumSpaces);

Parameter	Description
NumSpaces	number or integer tag representing the number of blanks

Example:

X represents blank.

StrSpace(4); returns "XXXX"

"Pump" + StrSpace(1) + "Station" returns "PumpXStation".

StrToInt

Convert the string consisting of digits to an integer that can be used for calculation.

Syntax:

IntegerResult=StrToInt (Text);

Parameter	Description
Text	string or string tag to proccess

The function gets the number represented by the first character in the string. If the first character is not digit (blank is ignored), returns 0. If is, the function continues to read the following characters until encountering non-digit character.

Example:

If Text="ABCD",then IntegerTag=0.

If Text="22.2 is the Value", then IntegerTag=22

If Text="22 is the Value", then IntegerTag=0

StrToReal

Convert the string consisting of digits to a long that can be used for calculation.

Syntax:

RealResult=StrToReal(Text);

Parameter	Description
Text	string or string tag to proccess

The function gets the number represented by the first character in the string. If the first character is not digit (blank is ignored), returns 0. If is, the function continues to read the following characters until encountering non-digit character.

Example:

If Text="ABCD", then RealTag=0.

If Text="22.261 is the Value", then RealTag=22.261.

If Text="The Value is 22", then RealTag=0.

StrTrim

Remove unnecessary blanks in string tag. Syntax:

MessageResult=StrTrim(Text,TrimType);

Parameter	Description
Text	string or string tag to proccess
TrimType	way to delete can be one of the following:

1	to remove all leading blanks from the string
2	to remove all trailing blanks from the string
3	to remove unnecessary blanks between words

Find the blank in Text (ASCII is ox9-ox01 or ox20).

Example:

X represents the blank.

StrTrim("xxxxxThisxisxaxxtestxxxxx", 1); returns "Thisxisxaxxtestxxxxx"

StrTrim("xxxxxThisxisxaxxtestxxxxx", 2); returns "xxxxxThisxisxaxxtest"

StrTrim("xxxxxThisxisxaxxtestxxxxx", 3); returns "Thisxisxaxtest"

StrReplace()

StrReplace() may remove all blanks in the string, that is to replace all blanks with Null.

StrType

Detect the first character in the string tag for judging its type. Syntax:

DiscreteResult=StrType(Text,TestType);

Parameter	Description
Text	string or string tag to proccess
TrimType	string type can be one of the following:

1	leter or digit (‘A’-’Z’, ‘a’-’z’, ‘0’-’9’)
2	digit (‘0’-’9’)
3	letter (‘A’-’Z’, ‘a’-’z’)
4	uppercase (‘A’-’Z’)
5	lowercase (‘a’-’z’)
6	punctuation mark (ox21-ox2F)
7	ASCII (ox00-ox7F)
8	hex character (‘A’-’F’, ‘a’-’f’, ‘0’-’9’)
9	printable character (ox20-ox7e)
10	control character (ox00-ox1F or ox7F)
11	blank (ASCII is ox9-ox0D or ox20)

If the type of the first character in Text has the same type as one specified by TestType, StrType() returns a positive to DiscreteResult. If there are more than one character in Text, StrType() only detects the first character.

Example:

StrType("ACB123",1); //returns 1.

StrType("ABC123",5); //returns 0.

StructVarRefAddress

Make the reference between structure tags. Structure tag may refer to other structure tags when they have the same number and type of numbers. Generally, the function is used in defining several groups of same IO tags. You can define a group of memory tags for configure the picture display. Syntax:

StructVarRefAddress(RefStructTagname, RefedStructTagname);

Parameter	Description
RefStructTagname	string representing the name of structure tag referring other
RefedStructTagname	string representing the name of structure tag to refer

Example:

There is an e_power monitoring system consisting of several transformers. Ask you to create a picture for displaying voltage, electric current and power of each transformer at different time. Ypu can use the structure tag to do it.

Define a structure Transformer including three number tags: I, V, P. Then define two structure tags: Transformer1 and Transformer2, their numbers I, V, P as IO tag. Define the structure tag Transformer3, its number I, V, P as memory tag. When defining picture animation link, you can use Transformer3 in scripts to refer to the two IO tags.

The following statement displays data of Transformer1.

StructVarRefAddress(“Transformer3”, “Transformer1”);

The following statement displays data of Transformer2.

StructVarRefAddress(“Transformer3”, “Transformer2”);

StrUpper

Convert all lowercase characters in specified into uppercase but doesn’t affect uppercase characters, tabs, digits and other special characters in it. Syntax:

MessageResult=StrUpper(Text);

Parameter	Description
Text	string or string tag

Example:

StrUpper("abcd");//returns "ABCD."

StrUpper("22.2 is the value"); //returns "22.2 IS THE VALUE"

StopBackupStation

Interrupt the backup manually when KingView makes the historical data backup and combination. Syntax:

BOOL StopBackupStation(str szStationName);

Parameter	Description
SzStationName	string representing the name of remote node

Example:

StopBackupStation(“IO Acquiring node”);

Sum

Return the sum of several tags. Syntax:

Sum(’a1’, ’a2’);

Parameter	Description
a1, a2	integer or long tag.

The number of parameters may be between 1 and 32.

When making the sum of the values in specified cells of the report, the result is displayed in current cell. The blanks or string in cells don’t affect the result. Syntax:

Sum(’Cells area’);

Example:

=Sum(‘a1’, ‘b2’, ‘r10’); returns the sum of any cells.

=Sum(‘b1:b10’); returns the sum of continual cells.

Tan

Return the tangent of the tag. Syntax:

Tan(tag);

Example:

Tan(45); //returns 1

Tan(0); //returns 0.

Text

Display an analog tag (integer or long) with the specified format. Syntax:

MessageResult=Text(Analog_Tag, Format_Text);

Parameter	Description
Analog_Tag	analog tag to convert
Format_Text	specified format

Example:

MessageTag=Text(Analog_Tag,"#.00");

If Analog_Tag=66, MessageTag=66.00.

If Analog_Tag=22.269, MessageTag=22.27.

If Analog_Tag=9.999, MessageTag=10.00

Time

Return the time string of default format (hour:minute:second) from given integer hour, minute and second. Syntax:

Time(LONG nHour, LONG nMinute, LONG nSecond);

Example:

The tags representing hour, minute and second are respectively “$hour”, “$Minute” and “$Second”, so the following statement stores the time string depending on three tags into the tag Time.

Time=Time(hour, minute, second);

Trace

Output the value with specified format in the Info window. It is for debug. Syntax: Trace(‘test = %2d’, Express)

That is to output the value of Express with decimal system into Info window. if Express=100, “text=100” is displayed in the Info window. you may specify the string “test”. Refer to the following table for output format:

Character	Type	Output format
D	Integer	Decimal integer distinguishing between positive and negative
x(lowercase)	Integer	Hexadecimal integer without distinguishing between positive and negative. Output the lowercase as ‘abcdef.’
X(uppercase)	Integer	Hexadecimal integer without distinguishing between positive and negative. Output the uppercase as ‘ABCDEF.’
E	Double	The number distinguishing between positive and negative, with the format:[–]*d***.***dddd***e**[*sign*]*ddd* *d*-decimal number, *dddd*-one or more decimal number, *ddd*-decimal number consisting of three digits, *sign*- + or -. As -1.000000e+002
E	Double	Same with the character e. only difference is exponent as E instead e, as -1.000000E+002
F	Double	The number distinguishing between positive and negative, with the forma: [–]*dddd***.***dddd*. *Dddd*-one or more decimal number. Digits before decimal point lies in data, after decimal point lies in precision
S(not match case)	String	Output the string tag on string format

Trunc

Truncate fractional part of the number. Syntax:

ResultNumericTag=Trunc(Number);

Parameter	Description
Number	any number, long tag of KingView

The result is similar to putting the long tag into the integer tag.

Example:

Trunc(4.3); returns 4

Trunc(-4.3); returns –4

VarRefAddress

Make the reference between general tags. General tag may refer to other general tags when they have the same number and type of numbers. Generally, the function is used in defining several groups of same IO tags. You can define a group of memory tags for configure the picture display. Syntax:

VarRefAddress(RefTagName, RefedTagName);

Parameter	Description
RefTagName	string representing the name of tag referring other
RefedTagName	string representing the name of tag to refer

Example:

There is an e-power monitoring system consisting of several transformers. Ask you to create a picture for displaying voltage, electric current and power of each transformer at different time. Ypu can use reference tag to do it.

Define all the IO tags: Transformer1_I, Transformer1_V, Transformer1_P, Transformer2_I, Transformer2_V, Transformer2_P. Define memory tags: Transformer3_I, Transformer3_V, Transformer3_P.Using the following tags when defining picture animation link: Transformer3_I, Transformer3_V,

Transformer3_P. You may use the tag reference function to refer to the IO tags of two devices in scripts.
The following statement displays data of Transformer1.
VarRefAddress (“Transformer3_I”, “Transformer1_I”);
VarRefAddress (“Transformer3_V”, “Transformer1_V”);
VarRefAddress (“Transformer3_P”, “Transformer1_P”);
The following statement displays data of Transformer2.
VarRefAddress (“Transformer3_I”, “Transformer2_I”);
VarRefAddress (“Transformer3_V”, “Transformer2_V”);
VarRefAddress (“Transformer3_P”, “Transformer2_P”);

WindowSize

This function is used to maximize or minimize the running screen. It avoids the blue directory (not match the screen color) existing in the running screen when maximizing and minimizing buttons are used. In this way the screen looks better. When this function is used, assisted by selecting “max/min button” in the setup of running systerm, the maximization and minimization of the window can switched.
Transferring format:
WindowSize(nFlag);
Note for parameter:
nFlag: control of Window max/min. nFlag =0, maximization; nFlag =1, minimization
No returned value

Example: Minimization of the running window:
WindowSize(1);

xyAddNewPoint

Add a point into specified trend of X-Y trend control. Syntax:

xyAddNewPoint ("ControlName", X, Y, Index);

Parameter Description

ControlName the name of the X-Y trend control defined by engineer in English or in Chinese

X: x- coordinate pf the point

Y: y- coordinate pf the point

Index the index of the trend in the control whose range is from 1 to 7.

Example:

The following statement adds a data point into the trend with index 1 in Reactor tank liquid pressure-level control. The point has the coordinate (30, 20) representing that the liquid pressure is 30 and the liquid level is 20.

xyAddNewPoint ("Reactor tank liquid pressure-level",30, 20,1);

xyClear

Clear the specified trend in the X-Y trend control. Syntax:

xyClear("ControlName",Index);

Parameter Description

ControlName the name of the X-Y trend control defined by engineer in English or in Chinese

Index the index of the trend in the control whose range is from 1 to 7. When it is set as –1, clears all trends.

Example:

The following statement clears the trend with the index 1 in Reactor tank liquid pressure-level control.

xyClear("Reactor tank liquid pressure-level",1);