HORIBA Scientific
Raman Spectroscopy

Title of Document :

LabSpec 5/6 Automation

Abstract :

This document provides the description of LabSpec 5/6 Automation capabilities. This can be done using either VBS Scripts in LabSpec 5/6 application or a LabSpec 5/6 ActiveX in an third party application.

Writer (belonging div.) :

Sebastien LADEN (HORIBA JOBIN YVON - Raman)

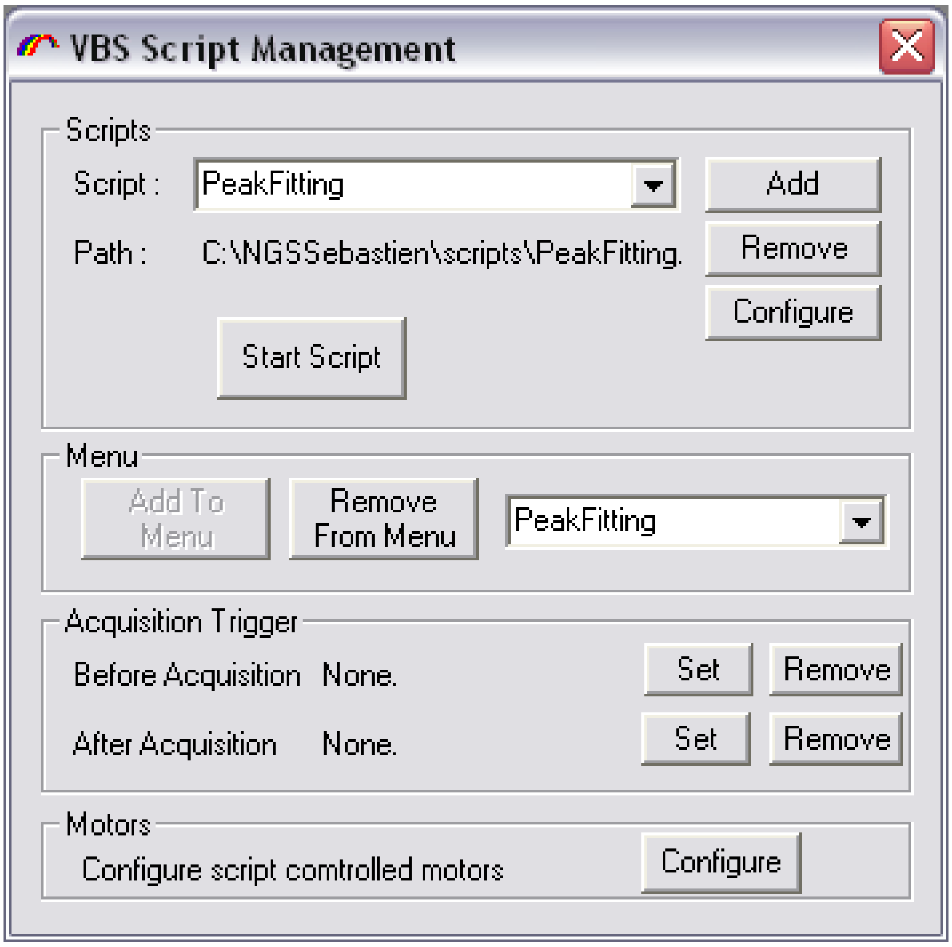
Release date : 03-22-2018

Introduction

LabSpec 5/6 allows developpers to automate its major functionalities, using either a scripting language or even an ActiveX module that can be integrated to any application that supports ActiveX technology. One can choose any of these automation technics, depending on their needs : The scripting capability allows the to write macro commands in Visual Basic Script (VBScript) to drive the motors, acquire data, and use the treatments functions, using the LabSpec interface. This is the easiest way to automate LabSpec. The ActiveX capability allows to fully integrate LabSpec in a 3rd party application. LabSpec interface is not loaded and requires a full graphical interface, giving much more flexibility.

1 – Scripting automation

a – LabSpec interface


VBS Script Management

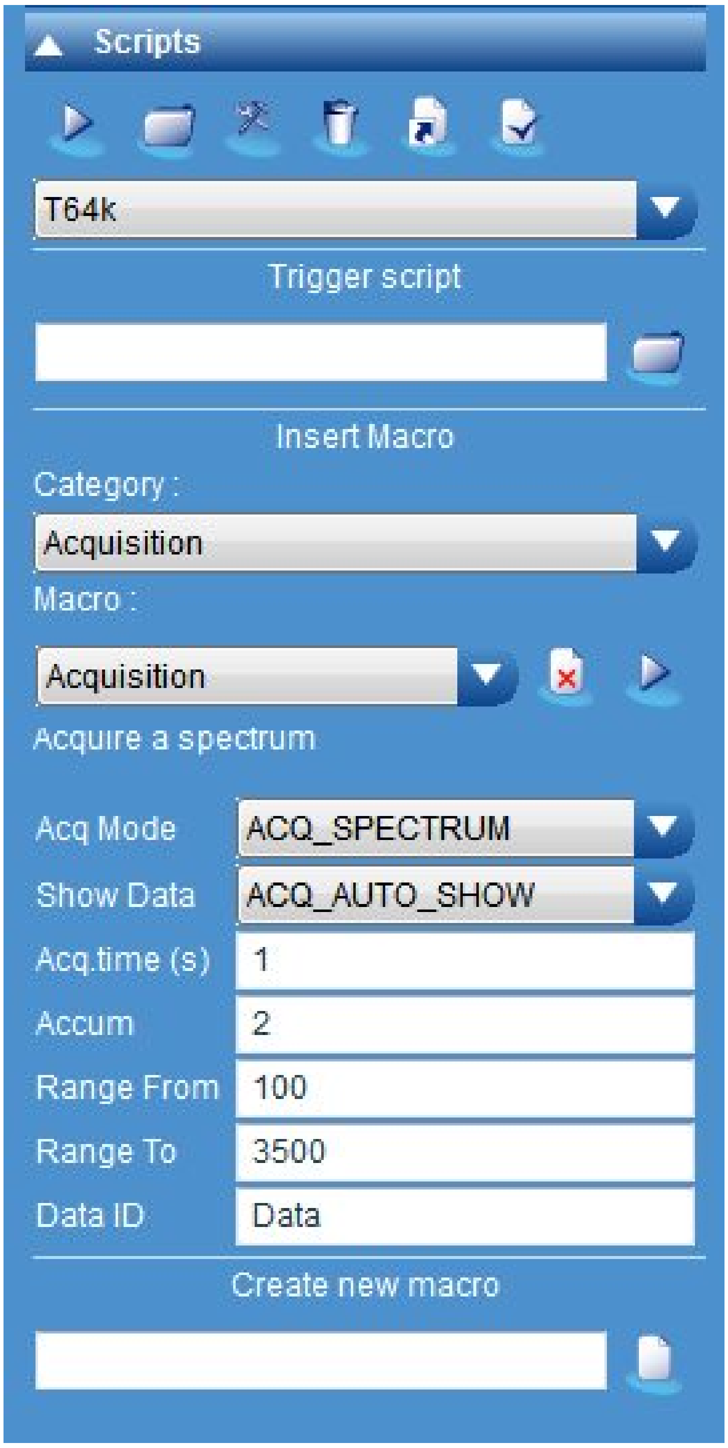
Scripts
Script : PeakFitting
Path : C:\NGSSebastien\scripts\PeakFitting.

Menu
PeakFitting

Acquisition Trigger
Before Acquisition None.
After Acquisition None.

Motors
Configure script comtrolled motors

LabSpec 5 VBS Script Management interface



T64k

Trigger script

Insert Macro

Category : Acquisition

Macro : Acquisition

Acquire a spectrum

Acq Mode	ACQ_SPECTRUM
Show Data	ACQ_AUTO_SHOW
Acq.time (s)	1
Accum	2
Range From	100
Range To	3500
Data ID	Data

Create new macro

LabSpec 6 VBS Script Management interface

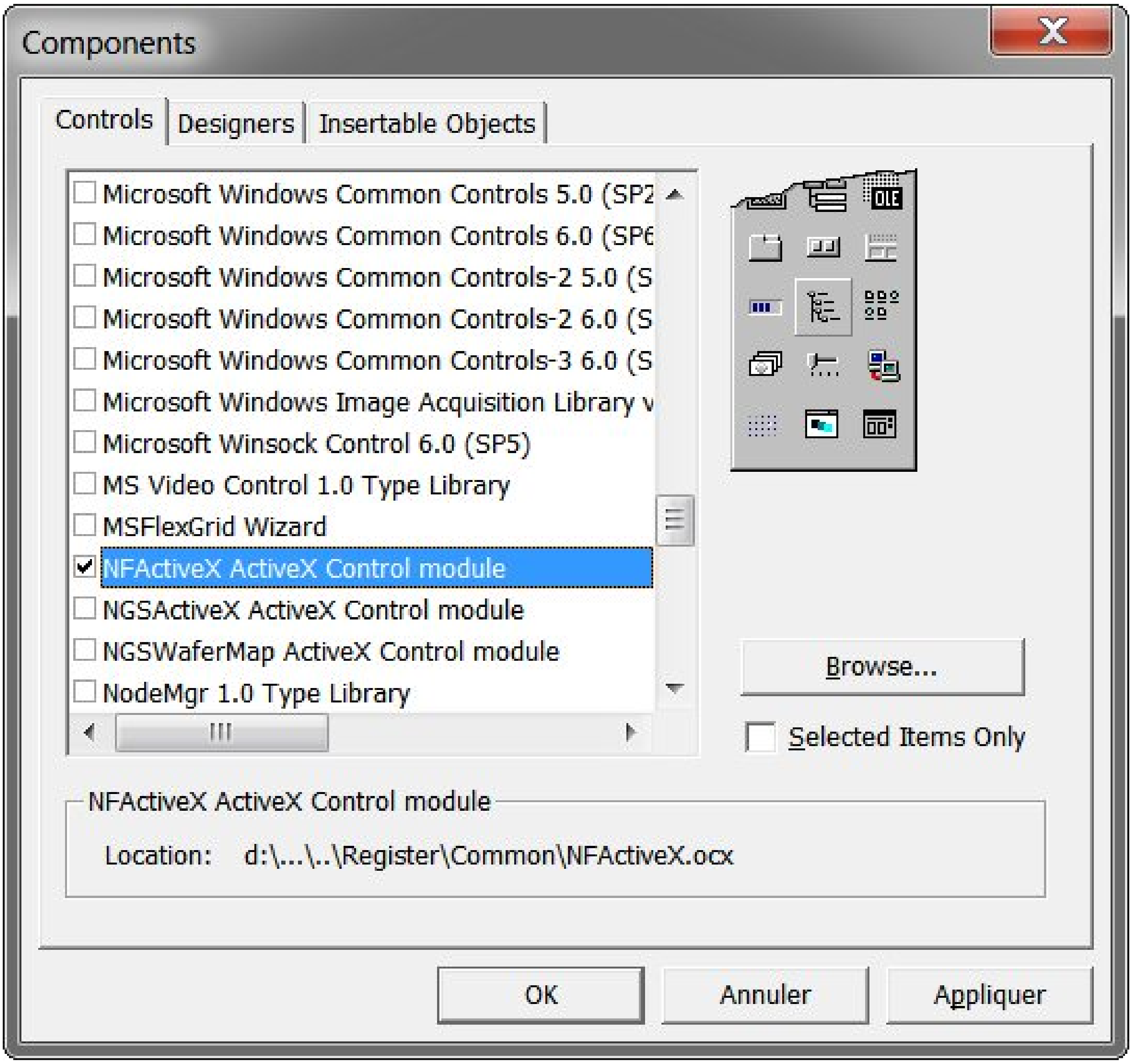
A very simple interface is available in LabSpec 5 (Scripts menu/Options) and labSpec 6 (Processing/Scripts). It is possible to add, remove or configure a script, and to start it. A script can be executed just before and just after an acquisition using the Acquisition Trigger Before and After parameters. It is also possible to add up to 5 script controlled motors. These motors can drive any hardware you want (see GetMotorAction Function)

b – Programming macro commands in VBScript

If a script is started from LabSpec, it automaticaly includes a object called LabSpec. This object manages the main LabSpec functionalities. In order to use one of theese functions, simple add : ReturnedValue = LabSpec.TheFunctionYouNeed(Param1,Param2,...) All the native VBScript functions are also off course available. (exept functions that need a window handeler such has MsgBox for exemple)

2 – ActiveX Automation

a – Integrating the LabSpec ActiveX


Components
Controls
Designers
Insertable Objects
Microsoft Windows Common Controls 5.0 (SP2
Microsoft Windows Common Controls 6.0 (SP6
Microsoft Windows Common Controls-2 5.0 (S
Microsoft Windows Common Controls-2 6.0 (S
Microsoft Windows Common Controls-3 6.0 (S
Microsoft Windows Image Acquisition Library v
Microsoft Winsock Control 6.0 (SP5)
MS Video Control 1.0 Type Library
MSFlexGrid Wizard
NFActiveX ActiveX Control module
NGSActiveX ActiveX Control module
NGSWaferMap ActiveX Control module
NodeMgr 1.0 Type Library
Browse...
Selected Items Only
NFActiveX ActiveX Control module
Location: d:\...\..\Register\Common\NFActiveX.ocx
OK
Annuler
Appliquer

Visual Basic Exemple – Components dialog

See the programming language manual to get how to integrate an ActiveX in your Application. LabSpec 5

ActiveX (NGSActiveX.ocx) or LabSpec 6 ActiveX (NFActiveX.ocx) can be imported in a third party application.

b – Programming an application using LabSpec ActiveX

Programming technics depends on the language you use. In Visual Basic, simply name your activeX « LabSpec » (or any other name, but the following examples use LabSpec as Control Name), and call the associated methods.

Acq

Description : Start an acquisition
Keywords : acquisition integration time accumulation extended range
Type : AutoVBSAct
Category : Collect Data

---

long Acq(long Mode, **double** IntegrationTime, **long** AccumulationNum, **double** From, **double** To)

Start an Acquisition

Mode : Acquisition Mode :
0 : ACQ_SPECTRUM Start a Spectrum Acquisition
1 : ACQ_IMAGE Start a CCD Image Acquisition
2 : ACQ_LABSPEC_PARAM Start Spectrum Acquisition with LabSpec interface parameters
3 : ACQ_SPECTRAL_IMAGE Start Spectral Image Acquisition with LabSpec interface parameters
if IntegrationTime set to -1, Mapping acquisition will start with LabSpec interface parameters
4 : ACQ_GET_TEMPERATURE Get the current Detector Temperature
5 : ACQ_SPECTRUM_RTD Start RTD Acquisition
6 : ACQ_SET_PMT_PARAMETER Set PMT Step (Use IntegrationTime Parameter to specify PMT Step).
Only sets the PMT Step. does not start PMT Acquisition.
7 : ACQ_MACRO_SPOT Start/Stop Macro spot (duo-scan option is required).
IntegrationTime=0 : Macro Spot ON.
IntegrationTime=1 : MacroSpot OFF.
8 : ACQ_CANCEL Cancel current acquisition.
9 : ACQ_PMT_CCD Set current detector
IntegrationTime=1 : PMT
IntegrationTime=2 : CCD
PLEASE NOTE : in PMT mode, you will have to open the shutter before starting the acquisition :
LabSpec.MoveMotor("DetectorShutter",1,"",MOTOR_VALUE), and close it after the acquisition :
LabSpec.MoveMotor("DetectorShutter",0,"",MOTOR_VALUE)

10 : ACQ_AUTO_SHOW Add to any other Acquisition Mode : Automatically show acquired data
100 : ACQ_NO_SPIKE_REMOVING Add this constant disable Spike removing function
200 : ACQ_SINGLE_SPIKE_REMOVING Add this constant to use Single pass Spike removing function
300 : ACQ_DOUBLE_SPIKE_REMOVING Add this constant to use double pass Spike removing function
400 : ACQ_DOUBLE_AUTOADD_SPIKE_REMOVING Add this constant to use double pass Spike removing function with auto add feature (automatically add 1 extra accumulation)

1000 : ACQ_AUTO_SCANNING Add this constant to use AutoScanning
2000 : ACQ_NO_CLOSE_SHUTTER Do not close the shutter after the acquisition
10000 : ACQ_ACCUMULATION_MODE Change accumulation mode (Average/Sum/Detector)
100000 : ACQ_NO_ICS Disable ICS
200000 : ACQ_ICS Enable ICS
1000000 : ACQ_NO_DARK Disable dark correction
2000000 : ACQ_DARK Enable dark correction

IntegrationTime : Spectrum Integration time (in seconds) (ignored if ACQ_LABSPEC_PARAM)
0 : Use autoexposure function
ACQ_ACCUMULATION_MODE : 0=Average ; 1=Sum ; 2=Detector

AccumulationNum : Number of spectrum accumulation (ignored if ACQ_LABSPEC_PARAM)

From, To : Acquisition range (in nm, use ConvertUnit to use cm-1). If From=To, the acquisition width will be the detector width. (ignored if ACQ_LABSPEC_PARAM)

Return Values :

if ACQ_GET_TEMPERATURE : Returns the detector temperature or -1000 if an error has occured
else : always return 0

Constants List :

```
Const ACQ_SPECTRUM = 0
Const ACQ_IMAGE = 1
Const ACQ_LABSPEC_PARAM = 2
Const ACQ_SPECTRAL_IMAGE = 3
Const ACQ_GET_TEMPERATURE = 4
Const ACQ_SPECTRUM_RTD = 5
Const ACQ_SET_PMT_PARAMETER = 6
Const ACQ_MACRO_SPOT = 7
Const ACQ_CANCEL=8
Const ACQ_PMT_CCD=9
Const ACQ_AUTO_SHOW = 10
Const ACQ_LABSPEC_SPIKE_REMOVING = 0
Const ACQ_NO_SPIKE_REMOVING = 100
Const ACQ_SINGLE_SPIKE_REMOVING = 200
Const ACQ_DOUBLE_SPIKE_REMOVING = 300
Const ACQ_DOUBLE_AUTOADD_SPIKE_REMOVING = 400
Const ACQ_AUTO_SCANNING = 1000
Const ACQ_NO_CLOSE_SHUTTER = 2000
Const ACQ_ACCUMULATION_MODE = 10000
Const ACQ_NO_ICS = 100000
Const ACQ_ICS = 200000
Const ACQ_NO_DARK = 1000000
Const ACQ_DARK = 2000000
```

Example

This example starts an acquisition and wait for the data to be ready.

```
' Start 1 sec integration time, 1 accumulation, no multiwindow acquisition
LabSpec.Acq ACQ_SPECTRUM+ACQ_AUTO_SHOW,1,1,0,0
' Wait Until Acquisition is done
do
     SpectrumID=LabSpec.GetAcqID()
Loop Until SpectrumID>0
```

```
Dim SpectrumID
StartAcq()
WaitForAcquisition()
LabSpec.Message "Acquisition done.",0
Private Sub StartAcq()
  Dim Mode
  Dim IntegrationTime
  Dim AccumulationNum
  Dim AcqFrom
  Dim AcqTo
  Mode=ACQ_SPECTRUM
  IntegrationTime=1.5 ' 1.5 sec acquisition
  AccumulationNum=2   ' 2 Accumulations
  AcqFrom=0 ' From=To => No MultiWindows
  AcqTo=0
  LabSpec.Acq Mode,IntegrationTime,AccumulationNum,AcqFrom,AcqTo
End Sub
Private Sub WaitForAcquisition()

  do
     SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is
done)
  Loop Until SpectrumID>0

  LabSpec.Exec SpectrumID , SHOW_SPECTRUM, Param ' Show Spectrum
End Sub
```

See Also GetAcqID

Autofocus

Description : Do an autofocus
Keywords : Z focus
Type : AutoVBSAct
Category : Collect Data

---

long Autofocus(**long** Mode)

Do an Autofocus using the current LabSpec Configuration (including the Raman Offset)

Mode : Autofocus Mode :
0 : START_AUTOFOCUS Start the autofocus
1 : GET_AUTOFOCUS_STATUS Get the autofocus status (1 busy, 0 ready)
2 : STOP_AUTOFOCUS Restore the previous settings
3 : GET_AUTOFOCUS_OFFSET Get the Autofocus offset (in nm)
4 : GET_AUTOFOCUS_STATE Returns 0 when AF set to OFF, 1 when AF is set to ON in LabSpec
5 : AUTOFOCUS_ENABLE Enable the AutoFocus
6 : AUTOFOCUS_DISABLE Disable the AutoFocus

10 : LASER_AUTOFOCUS Activate Laser based autofocus
11 : RAMAN_AUTOFOCUS Activate Raman based autofocus
12 : VIDEO_AUTOFOCUS Activate Video based autofocus

<u>Return Values :</u>

always return 0.

<u>Constants List :</u>

```
Const START_AUTOFOCUS  = 0
Const GET_AUTOFOCUS_STATUS  = 1
Const STOP_AUTOFOCUS  = 2
Const GET_AUTOFOCUS_OFFSET = 3
Const GET_AUTOFOCUS_STATE = 4
Const AUTOFOCUS_ENABLE = 5
Const AUTOFOCUS_DISABLE = 6
Const LASER_AUTOFOCUS  = 10
Const RAMAN_AUTOFOCUS  = 11
Const VIDEO_AUTOFOCUS  = 12
```

<u>Example</u>

```
LabSpec.Autofocus START_AUTOFOCUS

Dim Ret
do
   Ret=LabSpec.Autofocus (GET_AUTOFOCUS_STATUS )
loop until ret=0
Dim SpectrumID
Dim Param

do
   SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
Loop Until SpectrumID>0

LabSpec.Exec SpectrumID , SHOW_DATA, Param  ' Show
LabSpec.Autofocus STOP_AUTOFOCUS ' restore Z motor
```

GetAcqID

Description : Get Acquisition Spectrum ID
Keywords : acquisition acq ID
Type : AutoVBSAct
Category : Collect Data

---

long GetAcqID()

Get last acquisition data ID.
If a new acquisition data is available, GetAcqID will return the new data ID, and will then reset it to -1 (No acquisition in progress).

Return Values :

>0 : Spectrum ID
0 : Acquisition in progress
-1 : No acquisition in progress
-2 : Acquisition has been cancelled

Example

This example starts an acquisition and wait for the data to be ready.

```
Dim SpectrumID
StartAcq()
WaitForAcquisition()
LabSpec.Message "Acquisition done."
Private Sub StartAcq()
   Dim Mode
   Dim IntegrationTime
   Dim AccumulationNum
   Dim AcqFrom
   Dim AcqTo
   IntegrationTime=1.5 //1.5 sec acquisition
   AccumulationNum=2   // 2 Accumulations
   AcqFrom=0 From=To => No MultiWindows
   AcqTo=0
   LabSpec.Acq Mode,IntegrationTime,AccumulationNum,AcqFrom,AcqTo
End Sub
Private Sub WaitForAcquisition()

   do
       SpectrumID=LabSpec.GetAcqID()
   Loop Until SpectrumID>0

End Sub
```

See Also Acq

GetTriggerMode

Description : Get trigger Mode
Keywords : acquition before after dark
Type : AutoVBSAct
Category : Collect Data

---

long GetTriggerMode(**LPCTSTR** Mode)

Get Current Trigger Mode (Trigger scripts only)

Mode :
"BeforeAll" : Returns 1 if before All acquisitions trigger
"BeforeAcq" : Returns 1 if before acquisition trigger
"AfterDark" : (only with Signal-Dark mode) Returns 1 after the dark acq, and before the real acq
"AfterAcq" : Returns 1 if after acq trigger
"AfterAll" : Returns 1 if after All acquisitions trigger

"Dark" : Returns 1 Dark mode is enabled (Valid for all the previous triggers)

Return Values :

1 if Active mode
0 if not Active Mode

Example

```
Const PARAM_SAVE_INTERNAL = 2
Const PARAM_RESTORE_INTERNAL = 3
Const MOTOR_VALUE = 0
Dim BackupSlitPosition
If LabSpec.GetTriggerMode("BeforeAcq")  = 1 Then
    If LabSpec.GetTriggerMode("Dark")  = 1 Then
        BackupSlitPosition = LabSpec.GetMotorPosition ("Slit",MOTOR_VALUE)

        labSpec.Message "Backup Slit Pos : " & BackupSlitPosition,0
        LabSpec.SetSingleScriptParam
"SlitPos","step",BackupSlitPosition,PARAM_SAVE_INTERNAL
        labSpec.Message "Closing Slit",0
        MoveID=LabSpec.MoveMotor ("Slit",0,"",MOTOR_VALUE)
         Do
            Status=LabSpec.GetMotorStatus("Slit",MoveID)
         Loop Until Status=0
        labSpec.Message "Slit Closed",0
    End if
End if
If LabSpec.GetTriggerMode("AfterDark")  = 1 Then
        LabSpec.SetSingleScriptParam
"SlitPos","step",BackupSlitPosition,PARAM_RESTORE_INTERNAL
        labSpec.Message "Restoring Slit : " & BackupSlitPosition,0
        MoveID=LabSpec.MoveMotor ("Slit",BackupSlitPosition,"",MOTOR_VALUE)
         Do
            Status=LabSpec.GetMotorStatus("Slit",MoveID)
         Loop Until Status=0
        labSpec.Message "Slit Restored",0
 End If
```

SetAutoExposure

Description : Set Auto Exposure parameters
Keywords :
Type : AutoVBSAct
Category : Collect Data

---

long SetAutoExposure(**double** TestTime, **double** MinTime, **double** MaxTime, **double** DesiredIntensity)

Set Auto-Exposure parameters.

Auto-Exposure parameters are reset to the current LabSpec parameters at the end of the Acq() function, therefore, this function has to be called before each Acq() command.

TestTime : Exposure time used to test signal intensity

MinTime : Minimum exposure time

MaxTime : Maximum exposure time

DesiredIntensity : Intensity to reach

Example :

```
IntegrationTime=0 ' AutoExposure
  AccumulationNum=1 ' 1 Accumulation
  AcqFrom=0 ' From=To => No MultiWindows
  AcqTo=0
  TestTime=0.1 ' 0.1 sec
  MinTime=5 ' 5 sec minimum
  MaxTime=10 ' 10 sec maximum
  DesiredIntensity=32000 ' try to get 32000 counts
  LabSpec.SetAutoExposure TestTime,MinTime,MaxTime,DesiredIntensity ' Set Auto
Exposure Parameters
  LabSpec.Acq Mode+10,IntegrationTime,AccumulationNum,AcqFrom,AcqTo ' Start
Acquisition
  Do
     SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is
done)
     LabSpec.Message  SpectrumID,6
  Loop Until SpectrumID>0
```

Send

Description : Send Data To an external device
Keywords : AFM Communication network ethernet RS232 GPIB
Type : AutoVBSAct
Category : Communication

VARIANT **Send**(**LPCTSTR** To, **LPCTSTR** Command, **const VARIANT FAR&** Param, **long** Mode, **VARIANT FAR*** Status)

Send data to an external device. Return value is the received value from the external device.

To : Device Name (i.e. "JPKAFM" or "RS232")

Command : Command to send.
RS232_OPEN : Command must be : ComPort ; ComParams ; TimeOut (i.e. "COM1;19200,n,8n1;7000").
COM port number must be in the 1-9 range.
RS232_SEND : Command to send
RS232_RECEIVE : number of bytes to receive

GPIB_OPEN : Command must be : GPIB BoardNum ; Dev DeviceNum (i.e. "GPIB0;Dev1")
GPIB_SEND : Command to send
GPIB_RECEIVE : number of bytes to receive

HTTP_GET : Command must be : host/url (i.e. "www.myserver.com/page1.html" or
"http://192.168.1.100/mypage.php?var1=Myvar1&var2=Myvar2")

HTTP_POST : Command must be : host/url (i.e. "www.myserver.com/page1.php")
Param must include the variables and/or files to post.
The variables must be set before the files.
Variables and files sections are separated by #. i.e. : "var1=Myvar1&var2=Myvar2#MyFile=d:MyFile.jpg"

HTTP_DOWNLOAD : Command must be : host/url (i.e. "www.myserver.com/myfile.pdf" or
"http://192.168.1.100/dir1/MyFile.zip").
Param must include the downloaded file path (i.e. "d:MyDownloadedFile.pdf")
If download is successfull, Status=0 and function returns file size
If download fails, Status<0 and function returns an error message

Param : Optional Parameter (for interpreted commands, see mode)

Mode : Communication Mode :
RAW : Directly send the command to the hardware
INTERPRETED : Command will be interpreted by LabSpec before being sent to the hardware

RS232_OPEN : Open RS232 communication
RS232_SEND : Send RS232 command
RS232_RECEIVE : Receive RS232 communication

GPIB_OPEN : Open GPIB communication

GPIB_SEND : Send GPIB command
GPIB_RECEIVE : Receive GPIB communication

Status : Commad Status (only for INTERPRETED Mode)
SUCCESS if command successfully executed
FAILED if command failed

return Value :

RAW : Received command from the device
INTERPRETED : Interpreted result
HTTP : Requested data.

JPK AFM Interpreted commands Specifications :

Command : "ForceConnection" Try to connect to the AFM if not connected
Param : not used
Return Values : not used

Command : "MoveToPoint" Move to the specified Point
Param : Point Index
Return Values : SUCCESS : Array (2) filled with X and Y coords
FAILED : Error Message

Command : "StartHeightMeasurement" Start AFM Height Measurements
Param : N/A
Return Values : SUCCESS : OK
FAILED : Error Message

Command : "FinishHeightMeasurement" Stop and Get AFM Height Measurements
Param : N/A
Return Values : SUCCESS : Array (3) filled with Average Height, RMS and Accumulation Time
FAILED : Error Message

Command : "GetSampleScannerRange" Get Sample Scanner Range
Param : N/A
Return Values : SUCCES : Array (4) filled with xMin, yMin, xMax, yMax
FAILED : Error Message

Command : "GetListSize" Get Point List Size
Param : N/A
Return Value : Number of Points.

Command : "GetPointCoords" Get Point Coordonates
Param : Point Index
Return Values : SUCCESS : Array (2) filled with X and Y coords
FAILED : Error Message

Command : "GetList" Get Current List
Param : Mode : NATIVE (return either the point list or the point grid) or FORCE_LIST (return a point list)

Return Value : Array (ListSize*2+1) if LIST - (7+1) if GRID : First Value (Value(0)) = LIST or GRID.
if Value(0) = LIST : Value(1) to Value (ListSize*2) : Point List X0, Y0, X1, Y1, ..
if Value(0) = GRID : Value(1) to Value(8) : x0, y0, dU, dV, iLenght, jLenght, theta

---

Constant List :

```
Const INTERPRETED = 0
Const RAW = 1
Const SUCCESS = 0
Const FAILED = 1
Const LIST = 0
Const GRID = 1
Const NATIVE = 0
Const FORCE_LIST = 1
Const RS232_OPEN = 0
Const RS232_SEND = 1
Const RS232_RECEIVE = 2
Const GPIB_OPEN = 0
Const GPIB_SEND = 1
Const GPIB_RECEIVE = 2
```

Example :

```
' Get List Size
ListSize=LabSpec.Send("JPKAFM", "GetListSize", Param, INTERPRETED, Status)
LabSpec.Message "List Size :" & ListSize, ID_OK
' Get Full List or Grid
Values=LabSpec.Send("JPKAFM", "GetList", NATIVE, Param, Status)
LabSpec.Message "List Type:" & Values(0), ID_OK

if Values(0) = LIST then ' LIST
   for i=0 to ListSize-1
      LabSpec.Message "Point " & i & " X:" & Values(i*2+1) & " Y:" & Values(i*2+2), ID_OK
   next
else  ' GRID
   for i=1 to 7
      LabSpec.Message "GRID Param " & i & " :" & Values(i), ID_OK
   next
end if
for i=0 to ListSize-1 ' For Each point in the List
   ' Move to the Point
   Values = LabSpec.Send ("JPKAFM", "MoveToPoint", INTERPRETED, Param, Status)
   if Status = SUCCESS then
      LabSpec.Message "Current AFM Position X:" & Values(0) & " Y:" & Values(1), ID_OK
   end if
   ' Start AFM Measurement during the Raman Measurement
   Values = LabSpec.Send ("JPKAFM", "StartHeightMeasurement", INTERPRETED, Param, Status)

   ' Start Raman Acquisition
   LabSpec.Acq Mode,IntegrationTime,AccumulationNum,AcqFrom,AcqTo
   do
      SpectrumID=LabSpec.GetAcqID() ' Wait untill Spectrum is ready (acquisition is done)
   Loop Until SpectrumID>0
   ' Show Raman Spectrum
   LabSpec.Exec SpectrumID, SHOW_DATA, Param
   ' Get AFM Results
   Values = LabSpec.Send ("JPKAFM", "FinishHeightMeasurement", INTERPRETED, Param, Status)
   if Status=SUCCESS then
      LabSpec.Message "Average Height:" & Values(0) & " RMS:" & Values(1) & " Accumulation Time:" &
Values(2), ID_OK
   ' Set RMS the the Spectrum Parameters Table
      LabSpec.PutDataInfo SpectrumID, "Acq", "RMS", Values(1)
   else
      LabSpec.Message Values, ID_OK
   end if
next
```

RS232 Example (Get Temperature from a linkam stage)

```
Dim Param
Dim Status
Dim ret
' Init RS232 communication
LabSpec.Send "RS232","COM1;19200,n,8,1;7000",Param,GPIB_OPEN,Status
' Send Value to RS232
LabSpec.Send "RS232","T" & vbCr,Param,GPIB_SEND,Status
' Retreive 6 bytes from RS232 (status bytes)
ret=LabSpec.Send ("RS232","6",Param,GPIB_RECEIVE,Status)
' Retreive 4 bytes from RS232 (temperature value)
ret=LabSpec.Send ("RS232","4",Param,GPIB_RECEIVE,Status)
' Convert and display current temperature
LabSpec.Message "Current Temp : " & Hextodec(ret)*0.1 & " C",0
' retreive final data
ret=LabSpec.Send ("RS232","1",Param,GPIB_RECEIVE,Status)
```

GPIB Example (Display SR830 Data in the Status bar)

```
Const GPIB_OPEN = 0
Const GPIB_SEND = 1
Const GPIB_RECEIVE = 2
Dim param
Dim ret
' Initialize PGIB communication With SR830
LabSpec.Send "GPIB","GPIB0;Dev8",param,GPIB_OPEN,param
' Start Acquisition
LabSpec.Send "GPIB","STRT",param,GPIB_SEND,param
' Continuously read data from the Lock-in
Do
   ' Ask For Data
   LabSpec.Send "GPIB","OUTP?3",param,GPIB_SEND,param
   ' Read Data (size=20 bytes)
   ret=LabSpec.Send ("GPIB","20",param,GPIB_RECEIVE,param)
   ' Display Data in the StatusBar
   LabSpec.Message "SR830 Data : " & ret,6
Loop
```

ConvertUnit

Description : Convert unit nm/cm-1
Keywords : conversion wavelenght wavenumber nm cm-1 1/cm
Type : AutoActiveX
Category : Data Manipulation

---

double ConvertUnit(**double** Value, **long** Direction)

Unit conversion (nm, cm-1, eV)

Value : Value to Convert

Direction : 0 : CM1_TO_NM cm-1 to nm
1 : NM_TO_CM1 nm to cm-1
2 : EV_TO_NM eV to nm (LS6 only)
3 : NM_TO_EV nm to eV (LS6 only)

Return Values :

Converted Value

Constants List :

```
Const CM1_TO_NM = 0
Const NM_TO_CM1 = 1
Const EV_TO_NM = 2
Const NM_TO_EV = 3
```

CreateDataObject

Description : Create an empty Spectrum or Image
Keywords : create new data
Type : AutoVBSAct
Category : Data Manipulation

---

long CreateDataObject(**LPCTSTR** Type, **long** Size, **long** Size2, **long** Color)

Type : Data Type. "Spectrum" for a one dimension spectrum, "image" for a two dimension image or "FloatImage" for a two dimension image with float data type.

Size : Spectrum Size, or First dimension image size

Size2 : Second dimension image size (ignored for spectrum)

Color : RGB Color. NO_SPECIFIED_COLOR = -1 let LabSpec manage the object color.

Return Value :

>0 : Object ID
-1 : Failed to create object

Constant List :

```
const NO_SPECIFIED_COLOR = -1
```

Example :

Creates a 1024 points spectrum

```
Dim Size
Dim Size2
Dim SpectrumID
Size=1024
Size2=0
SpectrumID = LabSpec.CreateDataObject("Spectrum", Size, Size2,
NO_SPECIFIED_COLOR)
```

GetActiveData

Description : Get LabSpec Active Data
Keywords : retreive active data ID
Type : AutoVBS
Category : Data Manipulation

---

long GetActiveData(**LPCTSTR** DataType)

Get LabSpec Active Data. **VBS SCRIPT ONLY**

DataType : Data Type
"" : Get Active Data, not type specified
"Spectrum" : Get Active Spectrum
"SpIm" : Get Active Spectral Profile or Image
"Map" : Get Active Spectral Map
"Video" : Get Active Video Image
"GetFirstData" : Get first active data from the active window
"GetNextData" : Get next active data from the active window
Call GetFisrtData first, and then loop GetNextData until ID<=0 to get all data from the active window

Return Values :

>0 : Data ID
-1 : Failed

Example

```
Private Sub GetActiveSpectrum()
  Dim SpectrumID
  SpectrumID = LabSpec.GetActiveData("Spectrum")
  if SpectrumID <0 then LabSpec.Message "Could not find active Spectrum"
End Sub
```

```
ID=LabSpec.GetActiveData ("GetFirstData") ' Get first data from active window
If ID>0 Then ' If first data OK, try to get other data
   LabSpec.Message "First ID : " & ID, MB_OK
   Do
      ID=LabSpec.GetActiveData ("GetNextData")
      If ID>0 Then LabSpec.Message "Next ID : " & ID, MB_OK
   Loop until ID<=0 ' While data is valid, loop to get next data
End If
```

See Also : Load - Save

GetValue

Description : Get Spectrum Values
Keywords : get value data axis label array intensity frequency pointer cursor
Type : AutoVBSAct
Category : Data Manipulation

---

long GetValue(**long** ID, **LPCTSTR** pName, **VARIANT FAR*** pValue)

Get data Value

ID : Data ID

pName : Type of Value to retreive
"XYData" : Get a 2 dimensions array containing both X (frequency) and Y (intensity) values. **Only works for a single spectrum**.
"Data" : Get an array containing intensity values only
if the data contains several spectra (profile or spectral image), all the data are merged. (See merged data description)
and you have to split them to extract a single spectrum.
To get the size of a single spectrum, see "AxisSize".
"MapPoint:SpectrumIndex" : Get single spectrum from a map. SpectrumIndex is the zero based index of the spectrum inside the map
"AxisLabels" : Get an array containing the Axis Labels
"AxisUnits" : Get an array containing the Axis Units. Typically
X units : "nm", "1/cm"="cm-1", "eV"
Y units : "cnt", "cnt/sec"
Image : "sec","mkm"="?m","kbar"
"Axis" : Get a single dimension array containing the Axis. The axis are merged, use "AxisSize" to get the size of the axis.
"AxisIndex" : Get an array containing the Axis Indexes
"AxisSize" : Get an array containing the Axis Sizes.
Each axis can have a different size. The Intensity Axis has a 0 size.
"AxisType" : Get an array containing the Axis Types
The Axis types depend on the application type. These are the standard types for Raman Applications :
"Intens" : Intensity Axis
"Spectr" : Frequency Axis
"X" : X Axis
"Y" : X Axis
"Z" : X Axis
"Time" : Time Axis
"DoubleCursor" : LS6 Only. Get an array containing the Mouse Pointer Values :
index 0 : Pointer X1
index 1 : Pointer X2
"Pointer" : LS5 Only. Get an array containing the Mouse Pointer Value :
index 0 : Pointer1 X
index 1 : Pointer1 Y
index 2 : Pointer2 X
index 3 : Pointer2 Y

index 4 : Pointer Type :
0 - POINTER_VERTICAL_LINE : Single vertical Line, return Position (Index 0)
1 - POINTER_CROSS : cross pointer return Position (Index 0) and Intensity (Index 1)
2 - POINTER_LEVEL_LINE : Single horizontal Line, return Intensity (Index 1)
3 - POINTER_DOUBLE : dual vertical lines, return Peak 1 Position (Index 0) and Peak 2 Position (Index 2)
4 - POINTER_RECT : rectangle, return upper left corner and lower right corner
5 - POINTER_PEAK : auto find closest peak, return Position (Index 0) Left Limit (Index 1) right Limit (Index 2) and FWHM (Index 3)
"PointerLimits" : Get an array containing the Red, Green and Blue pointer limits :
index 0 : Pointer Red 1 X
index 1 : Pointer Red 1 Y
index 2 : Pointer Red 2 X
index 3 : Pointer Red 2 Y
index 4 : Pointer Green 1 X
index 5 : Pointer Green 1 Y
index 6 : Pointer Green 2 X
index 7 : Pointer Green 2 Y
index 8 : Pointer Blue 1 X
index 9 : Pointer Blue 1 Y
index 10 : Pointer Blue 2 X
index 11 : Pointer Blue 2 Y
"CursorSpectrum" : Get active spectrum cursor (on the Cursor image)
index 0 : Cursor Type
0 - CURSOR_POINT : cross cursor
1 - CURSOR_RECTANGLE : rectangle cursor
2 - CURSOR_ELIPSE : elipse cursor
15 - CURSOR_POLYGON : polygon cursor
index 1 : Surrounding rectangle X1
index 2 : Surrounding rectangle Y1
index 3 : Surrounding rectangle X2 (0 for cross cursor)
index 4 : Surrounding rectangle Y2 (0 for cross cursor)
"CursorSpectrumAverage" : Get active Cursor spectrum: Average
"ActiveCursors" : Get active cursors (Red, Green and blue pointers)
index 0 : Pointer Red
index 1 : Pointer Green
index 2 : Pointer Blue
"NearIndex" : Find Nearest axis index to a specified Value (in nm). pValue must be a single float value (i.e. 100.0). Returned Value is the nearest pixel index.
"AcqInfo" : Returns Acquisition information array.
column 1 : entry name
column 2 : entry value
column 3 : entry unit
"CustomInfo" : Returns Custom information array.
column 1 : entry name
column 2 : entry value
"HistoryInfo" : Returns History information array.
column 1 : entry name
column 2 : entry date
column 3 : entry operation
(use PutDataInfo to add infos)
"DataName" : Get the internal data name (use PutDataInfo to change this name)
"Dispersion" : Get a array containing the dispersion (frequency) axis with current settings (central position, CCD zone...)

If pValue is provided, Dispersion will be calculated with these values,
pValue[0] : CCD Size
pValue[1] : Pixel Size
pValue[2] : Central Pixel
Use DispersionParam to get an empty array of values
"DispersionParams" : Get an empty array (3) of values (see script example)
"LabSpecPath" : Get LabSpec installation folder
"UserLevel" : Get CFR21 User Level
"SystemName" : Get system name (LabramHR, Aramis, XploRA or T64000)
"AutofocusDiode" : Get the current Autofocus diode intensity
"LabSpecExposureTime" : Get LabSpec interface Exposure time
"LabSpecLiveTime" : Get LabSpec interface RTD time
"LabSpecAccumulation" : Get LabSpec interface Number of accumulations
"AlignmentDiode" : Get Aligment diode position (in spectro steps). If ID=-1 : Diode position for the active grating. If ID>=0 : Diode position for the given grating ID
"ShutterMode" : Laser Shutter Mode :
SHUTTER_AUTO=0 : Shutter open during acquisition
SHUTTER_MANUAL=1 : Manual shutter
"PMTValue" : Read current PMT value

pValue : Array of values

Return Values :

0 : Succeeded
-1 : Failed

Constants List :

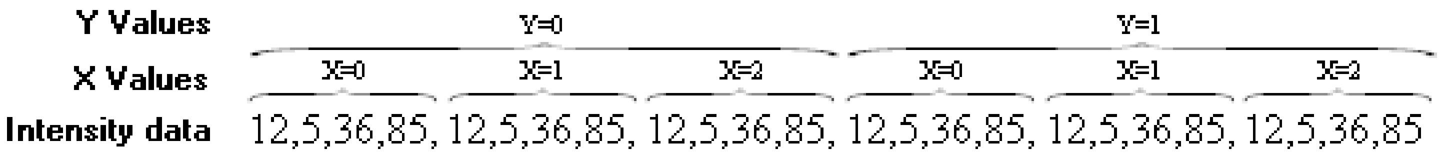
```
Const POINTER_VERTICAL_LINE = 0
Const POINTER_CROSS = 1
Const POINTER_LEVEL_LINE = 2
Const POINTER_DOUBLE = 3
Const POINTER_RECT = 4
Const POINTER_PEAK = 5
Const LEVEL_GUEST = 0
Const LEVEL_OPERATOR = 8
Const LEVEL_ENGINEER = 9
Const LEVEL_ADMINISTRATOR = 10
Const CURSOR_POINT = 0
Const CURSOR_RECTANGLE = 1
Const CURSOR_ELIPSE = 2
Const CURSOR_POLYGON = 15
```

Merged Data Description :

The following example is a spectral image.
- Size of "Y" : 2

- Size of "X" : 3
- Size of "Spectr" : 4

All data are merged in the Axis order, as illustrated bellow :



Example

This Example Opens a spectrum file,
Gets Y (intensity) data from spectrum,
filters the data,
Put the filtered data to the spectrum,
Save the spectrum

```
Dim SpectrumID
Dim SpectrumValues
OpenSpectrum()
GetSpectrum()
FilterSpectrum()
PutSpectrum()
SaveSpectrum()
Private Sub OpenSpectrum()
  Dim FileName
  FileName ="C: est.tsf"
  SpectrumID = LabSpec.Load(FileName)
End Sub
Private Sub GetSpectrum()
  LabSpec.GetValue SpectrumID,"Data",SpectrumValues
End Sub
// Filter example
Private Sub FilterSpectrum()
    For i = 0 To UBound(SpectrumValues)
        sum = 0
        startFilter = i - 10
        If startFilter < 0 Then startFilter = 0
        stopFilter = i + 10
        If stopFilter > UBound(SpectrumValues) Then stopFilter =
UBound(SpectrumValues)

        For j = startFilter To stopFilter
            sum = SpectrumValues(j) + sum
        Next
        SpectrumValues(i) = CDbl(sum / (stopFilter - startFilter + 1))
    Next
End Sub
Private Sub PutSpectrum()
  LabSpec.PutValue ID,"Data",SpectrumValues
End Sub
Private Sub SaveSpectrum()
  Dim FileName
  Dim Format
  Dim Ret
  FileName ="C: est2.tsf"
  Ret = LabSpec.Save(SpectrumID, FileName, Format)
End Sub
```

```
' Get Dispersion Array (depending On input parameters)
Dim Param
LabSpec.GetValue 0,"DispersionParams", Param
Param(0)=CDbl(1376) ' Detector Size
Param(1)=CDbl(13)   ' Pixel Size
Param(2)=CDbl(686)  ' Central Pixel
LabSpec.GetValue 0,"Dispersion",Param
```

```
' Show custom infos
dim CustomInfos
LabSpec.GetValue SpectrumID,"CustomInfo",CustomInfos
for i=1 to Ubound(CustomInfos,2)
 LabSpec.Message CustomInfos(0,i) & "_" & CustomInfos(1,i) ,0
next
```

```
public function FindMax(SpectrumID)
 Dim SpectrumValues
 Dim MaxI
 Dim MaxF
 LabSpec.GetValue SpectrumID, "XYData", SpectrumValues
 For i = 0 To UBound(SpectrumValues, 2)
     If MaxI < SpectrumValues(1, i) Then
         MaxI = SpectrumValues(1, i)
         MaxF = SpectrumValues(0, i)
     End If
 Next
 FindMax=MaxF
end function
```

Cursor spectrum (get position, set position and get average spectrum)

```
Const CURSOR_POINT = 0
Const CURSOR_RECTANGLE = 1
Const CURSOR_ELIPSE = 2
Const CURSOR_POLYGON = 15
ID=LabSpec.GetActiveData("SpIm")
' Get Current SpectrumCursor values
If (LabSpec.GetValue(ID,"CursorSpectrum",CursorSpectrum)<>-1) Then
   CursorType=CursorSpectrum(0)
   x1=CursorSpectrum(1)
   y1=CursorSpectrum(2)
   x2=CursorSpectrum(3)
   y2=CursorSpectrum(4)
   Width=Abs(x2-x1)
   Height=Abs(y2-y1)
   If CursorType=CURSOR_POINT Then CursorTypeStr="Point"
   If CursorType=CURSOR_RECTANGLE Then CursorTypeStr="Rectangle"
   If CursorType=CURSOR_ELIPSE Then CursorTypeStr="Elipse"
   If CursorType=CURSOR_POLYGON Then CursorTypeStr="Polygon"
   LabSpec.Message "Current cursor type is " & CursorTypeStr & vbNewLine & "Width=" & Round(Width,2) &
vbNewLine & "Height=" & Round(Height,2) & vbNewLine & "Surrounding rectangle = (" & Round(x1,2) & ";" &
Round(y1,2) & ") (" & Round(x2,2) & ";" & Round(y2,2) & ")",0
   newX1=-5
   newY1=-5
   newX2= 5
   newY2= 5
   ' Set new SpectrumCursors values
   CursorSpectrum(0)= CURSOR_RECTANGLE
   CursorSpectrum(1)= newX1
   CursorSpectrum(2)= newY1
   CursorSpectrum(3)= newX2
   CursorSpectrum(4)= newY2
   LabSpec.PutValue SpImID,"CursorSpectrum",CursorSpectrum
   ' Get Average Spectrum ID
   LabSpec.GetValue SpImID,"CursorSpectrumAverage",AverageID
   ' Extract average spectrum intensity array
   LabSpec.GetValue AverageID,"Data",AverageData
End if
```

See Also : PutValue - PutDataInfo

GetValueEx

Description : Extended GetValue function
Keywords : get value data axis label array intensity frequency pointer cursor
Type : AutoActiveX
Category : Data Manipulation

VARIANT GetValueEx(**long** ID, **LPCTSTR** pName, **VARIANT FAR*** pValue, **long** VarType)

You should prefere GetValue() function. Only use GetValueEx() if GetValue() doest not work properly for you. If your software does not support VARIANT *, you should use GetValueSimple() LS5 only.
Extended Get data Value.

ID : Data ID

pName : Type of Value to retreive
"XYData" : Get a 2 dimensions array containing both X (frequency) and Y (intensity) values. **Only works for a single spectrum**.
"Data" : Get an array containing intensity values only
if the data contains several spectra (profile or spectral image), all the data are merged. (See merged data description)
and you have to split them to extract a single spectrum.
To get the size of a single spectrum, see "AxisSize".
"AxisLabels" : Get an array containing the Axis Labels
"AxisUnits" : Get an array containing the Axis Units
X units : "nm", "1/cm"="cm-1", "eV"
Y units : "cnt", "cnt/sec"
Image : "sec","mkm"="?m","kbar"
"Axis" : Get a single dimension array containing the Axis. The axis are merged, use "AxisSize" to get the size of the axis.
"AxisIndex" : Get an array containing the Axis Indexes
"AxisSize" : Get an array containing the Axis Sizes.
Each axis can have a different size. The Intensity Axis has a 0 size.
"AxisType" : Get an array containing the Axis Types
The Axis types depend on the application type. These are the standard types for Raman Applications :
"Intens" : Intensity Axis
"Spectr" : Frequency Axis
"X" : X Axis
"Y" : X Axis
"Z" : X Axis
"Time" : Time Axis
"Pointer" : Get an array containing the Mouse Pointer Value :
index 0 : Pointer1 X
index 1 : Pointer1 Y
index 2 : Pointer2 X
index 3 : Pointer2 Y
index 4 : Pointer Type :
0 - POINTER_VERTICAL_LINE : Single vertical Line, return Position (Index 0)

1 - POINTER_CROSS : cross pointer return Position (Index 0) and Intensity (Index 1)
2 - POINTER_LEVEL_LINE : Single horizontal Line, return Intensity (Index 1)
3 - POINTER_DOUBLE : dual vertical lines, return Peak 1 Position (Index 0) and Peak 2 Position (Index 2)
4 - POINTER_RECT : rectangle, return upper left corner and lower right corner
5 - POINTER_PEAK : auto find closest peak, return Position (Index 0) Left Limit (Index 1) right Limit (Index 2) and FWHM (Index 3)
"NearIndex" : Find Nearest axis index to a specified Value (in nm).
"AcqInfo" : Returns Acquisition information array.
column 1 : entry name
column 2 : entry value
column 3 : entry unit
"CustomInfo" : Returns Custom information array.
column 1 : entry name
column 2 : entry value
"HistoryInfo" : Returns History information array.
column 1 : entry name
column 2 : entry date
column 3 : entry operation
"CurrentPositionName:" + Motor Name : Get the current specified motor position name (i.e. : "CurrentPositionName:Objective" will return : "x10" or "x100 LWD"...)

pValue : Array of values. Is Set, GetValueEx() Will ReDim the array and set the data. If NULL, a new array will be created.

VarType : Variant Type. Not all types are valid, see Constant List.

Return Values :

Variant containing the requested array of Values.

Constants List :

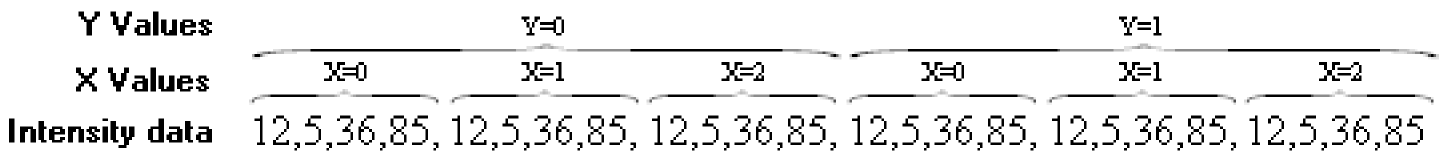
```
Const POINTER_VERTICAL_LINE = 0
Const POINTER_CROSS = 1
Const POINTER_LEVEL_LINE = 2
Const POINTER_DOUBLE = 3
Const POINTER_RECT = 4
Const POINTER_PEAK = 5
Const VT_I4 = 3 // integer
Const VT_R4 = 4 // float
Const VT_R8 = 5 // double
Const VT_UI1 = 17 // byte
```

Merged Data Description :

The following example is a spectral image.
- Size of "Y" : 2
- Size of "X" : 3

- Size of "Spectr" : 4

All data are merged in the Axis order, as illustrated bellow :



Example

This Example Opens a spectrum file,
Gets Y (intensity) data from spectrum,
filters the data,
Put the filtered data to the spectrum,
Save the spectrum

```
Dim SpectrumID
Dim SpectrumValues
OpenSpectrum()
if SpectrumID>0 then
  GetSpectrum()
  FilterSpectrum()
  PutSpectrum()
  SaveSpectrum()
end if
Private Sub OpenSpectrum()
  Dim FileName
  FileName ="C: est.tsf"
  SpectrumID = LabSpec.Load(FileName)
End Sub
Private Sub GetSpectrum()
  Dim EmptyVariant
  SpectrumValues = LabSpec.GetValueEx(SpectrumID,"Data",EmptyVariant,VT_R4)
End Sub
// Filter example
Private Sub FilterSpectrum()
    For i = 0 To UBound(SpectrumValues)
        sum = 0
        startFilter = i - 10
        If startFilter < 0 Then startFilter = 0
        stopFilter = i + 10
        If stopFilter > UBound(SpectrumValues) Then stopFilter =
UBound(SpectrumValues)

        For j = startFilter To stopFilter
            sum = SpectrumValues(j) + sum
        Next
        SpectrumValues(i) = CDbl(sum / (stopFilter - startFilter + 1))
    Next
End Sub
Private Sub PutSpectrum()
  LabSpec.PutValue ID,"Data",SpectrumValues
End Sub
Private Sub SaveSpectrum()
  Dim FileName
  Dim Format
  Dim Ret
  FileName ="C: est2.tsf"
  Ret = LabSpec.Save(SpectrumID, FileName, Format)
End Sub
```

See Also : PutValue - GetValue

GetValueSimple

Description : Simple way to get Values from data
Keywords : get value data axis label array intensity frequency pointer cursor
Type : AutoActiveX
Category : Data Manipulation

VARIANT GetValueSimple(**long** ID, **LPCTSTR** pName, **double** Value, **long** VarType)

You should prefere GetValue() function. Only use GetValueSimple() if GetValue() doest not work properly for you (i.e. VARIANT FAR * not supported).
Simple Get data Value.

ID : Data ID

pName : Type of Value to retreive
"XYData" : Get a 2 dimensions array containing both X (frequency) and Y (intensity) values. **Only works for a single spectrum**.
"Data" : Get an array containing intensity values only
if the data contains several spectra (profile or spectral image), all the data are merged. (See merged data description)
and you have to split them to extract a single spectrum.
To get the size of a single spectrum, see "AxisSize".
"AxisLabels" : Get an array containing the Axis Labels
"AxisUnits" : Get an array containing the Axis Units
X units : "nm", "1/cm"="cm-1", "eV"
Y units : "cnt", "cnt/sec"
Image : "sec","mkm"="?m","kbar"
"Axis" : Get a single dimension array containing the Axis. The axis are merged, use "AxisSize" to get the size of the axis.
"AxisIndex" : Get an array containing the Axis Indexes
"AxisSize" : Get an array containing the Axis Sizes.
Each axis can have a different size. The Intensity Axis has a 0 size.
"AxisType" : Get an array containing the Axis Types
The Axis types depend on the application type. These are the standard types for Raman Applications :
"Intens" : Intensity Axis
"Spectr" : Frequency Axis
"X" : X Axis
"Y" : X Axis
"Z" : X Axis
"Time" : Time Axis
"Pointer" : Get an array containing the Mouse Pointer Value :
index 0 : Pointer1 X
index 1 : Pointer1 Y
index 2 : Pointer2 X
index 3 : Pointer2 Y
index 4 : Pointer Type :

0 - POINTER_VERTICAL_LINE : Single vertical Line, return Position (Index 0)
1 - POINTER_CROSS : cross pointer return Position (Index 0) and Intensity (Index 1)
2 - POINTER_LEVEL_LINE : Single horizontal Line, return Intensity (Index 1)
3 - POINTER_DOUBLE : dual vertical lines, return Peak 1 Position (Index 0) and Peak 2 Position (Index 2)
4 - POINTER_RECT : rectangle, return upper left corner and lower right corner
5 - POINTER_PEAK : auto find closest peak, return Position (Index 0) Left Limit (Index 1) right Limit (Index 2) and FWHM (Index 3)
"NearIndex" : Find Nearest axis index to a specified Value (in nm).
"AcqInfo" : Returns Acquisition information array.
column 1 : entry name
column 2 : entry value
column 3 : entry unit
"CustomInfo" : Returns Custom information array.
column 1 : entry name
column 2 : entry value
"HistoryInfo" : Returns History information array.
column 1 : entry name
column 2 : entry date
column 3 : entry operation

Value : Near index value to find.

VarType : Variant Type. Not all types are valid, see Constant List.

<u>Return Values :</u>

Variant containing the requested array of Values.

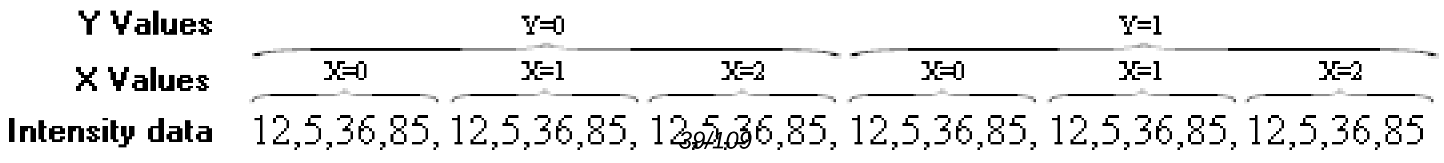
<u>Constants List :</u>

```
Const POINTER_VERTICAL_LINE = 0
Const POINTER_CROSS = 1
Const POINTER_LEVEL_LINE = 2
Const POINTER_DOUBLE = 3
Const POINTER_RECT = 4
Const POINTER_PEAK = 5
Const VT_I4 = 3 // integer
Const VT_R4 = 4 // float
Const VT_R8 = 5 // double
Const VT_UI1 = 17 // byte
```

<u>Merged Data Description :</u>

The following example is a spectral image.
- Size of "Y" : 2
- Size of "X" : 3
- Size of "Spectr" : 4

All data are merged in the Axis order, as illustrated bellow :



Example

This Example Opens a spectrum file,
Gets Y (intensity) data from spectrum,
filters the data,
Put the filtered data to the spectrum,
Save the spectrum

```
Dim SpectrumID
Dim SpectrumValues
OpenSpectrum()
if SpectrumID>0 then
  GetSpectrum()
  FilterSpectrum()
  PutSpectrum()
  SaveSpectrum()
end if
Private Sub OpenSpectrum()
  Dim FileName
  FileName ="C: est.tsf"
  SpectrumID = LabSpec.Load(FileName)
End Sub
Private Sub GetSpectrum()
   SpectrumValues = LabSpec.GetValueSimple(SpectrumID,"Data",0,VT_R4)
End Sub
// Filter example
Private Sub FilterSpectrum()
    For i = 0 To UBound(SpectrumValues)
        sum = 0
        startFilter = i - 10
        If startFilter < 0 Then startFilter = 0
        stopFilter = i + 10
        If stopFilter > UBound(SpectrumValues) Then stopFilter =
UBound(SpectrumValues)

        For j = startFilter To stopFilter
            sum = SpectrumValues(j) + sum
        Next
        SpectrumValues(i) = CDbl(sum / (stopFilter - startFilter + 1))
    Next
End Sub
Private Sub PutSpectrum()
  LabSpec.PutValue ID,"Data",SpectrumValues
End Sub
Private Sub SaveSpectrum()
  Dim FileName
  Dim Format
  Dim Ret
  FileName ="C: est2.tsf"
  Ret = LabSpec.Save(SpectrumID, FileName, Format)
End Sub
```

See Also : PutValue - GetValue

PutDataInfo

Description : Set Extra info to the data table
Keywords : parameter param table acq custom history information
Type : AutoVBSAct
Category : Data Manipulation

PutDataInfo(**long** ID, **LPCTSTR** TableName, **LPCTSTR** TableEntry, **LPCTSTR** TableValue)

Add or modify any data info (see Info Icon in LabSpec).
Use GetValue to read the data

ID : Data ID (Ignored for "MainCustom")

TableName : Name of the Parameter Table ("Acq" or "Custom" for spectrum related info, "MainCustom" for LabSpec info, "AcqValue" for the numerical values of Acq table)

TableEntry : Can be an existing one i.e. :
(Acq : "Exposition", "Accumulation", "Spectro", "Hole", "Slit", "Laser", "Grating", "Filter", "Spec. width", "Detector", "Detector Size")
(Custom : "Operator", "Sample", "Remark", "Power")
(DataName : "DataName") Data internal name (can be different from filename)
(DataName : "SpectrumType") Spectrum type. Spectra will be sorted by type (one window per type)
(DataName : "AllInfos") Insert all current data info (spectral position, etc..) to data object.
Or any other parameter of your choice.

TableValue : Parameter Value.

<u>*Example*</u> :

```
LabSpec.Acq Mode,IntegrationTime,AccumulationNum,AcqFrom,AcqTo
do
   SpectrumID=LabSpec.GetAcqID() ' Wait untill Spectrum is ready (acquisition is done)Loop Until
SpectrumID>0

LabSpec.PutDataInfo SpectrumID, "Acq", "Temperature", CurrentTemperature
```

```
LabSpec.PutDataInfo SpectrumID, "MainCustom", "Date", "08-02-07"
LabSpec.PutDataInfo SpectrumID, "MainCustom", "Operator", "Operator1"
LabSpec.PutDataInfo SpectrumID, "MainCustom", "Sample", "MySample"
LabSpec.PutDataInfo SpectrumID, "MainCustom", "Remark", "Automatic Info"
LabSpec.PutDataInfo SpectrumID, "MainCustom", "Power", "100mW"
```

See Also : GetValue

PutValue

Description : Put Values to a spectrum
Keywords : set value data axis label array intensity frequency pointer cursor
Type : AutoVBSAct
Category : Data Manipulation

---

long PutValue(**long** ID, **LPCTSTR** pName, **VARIANT FAR*** pValue)

Set data Values

ID : Data ID

pName : Type of Value to retreive
"XYData" : Set a 2 dimensions array containing both X (frequency) and Y (intensity) values. **Only works for a single spectrum.**
"Data" : Set an array containing intensity values only
"MapPoint:SpectrumIndex" : Update a single spectrum to a map. SpectrumIndex is the zero based index of the spectrum inside the map
"AxisLabels" : Set an array containing the Axis Labels
"AxisUnits" : Set an array containing the Axis Units
"Axis" : Set a single dimension array containing the Axis. The axis are merged, use "AxisSize" to get the size of the axis.
"AxisLimits" : Set an array containing the Axis Limits
"DisplayUnit" : Set current display unit (use pValue "nm", "1/cm", ...)
"IntensityUnit" : Set current intensity unit (use pValue "cnt", "cnt/sec...)
"CursorType" : LS6 Only. Set current cursor type (CURSOR_LINE, CURSOR_DOUBLE, CURSOR_CROSS, CURSOR_INTEGRAL)
"DoubleCursor" : LS6 Only. Set the Mouse Pointer Values :
index 0 : Pointer X1
index 1 : Pointer X2
"PointerLimits" : Set the Red, Green and Blue pointer limits in nm (Use GetValue first to get the current array) :
index 0 : Pointer Red 1 X
index 1 : Pointer Red 1 Y
index 2 : Pointer Red 2 X
index 3 : Pointer Red 2 Y
index 4 : Pointer Green 1 X
index 5 : Pointer Green 1 Y
index 6 : Pointer Green 2 X
index 7 : Pointer Green 2 Y
index 8 : Pointer Blue 1 X
index 9 : Pointer Blue 1 Y
index 10 : Pointer Blue 2 X
index 11 : Pointer Blue 2 Y
"CursorSpectrum" : Set spectrum cursor (on the Cursor image)
index 0 : Cursor Type
0 - CURSOR_POINT : cross cursor
1 - CURSOR_RECTANGLE : rectangle cursor

2 - CURSOR_ELIPSE : elipse cursor
index 1 : Surrounding rectangle X1
index 2 : Surrounding rectangle Y1
index 3 : Surrounding rectangle X2 (not used for cross cursor)
index 4 : Surrounding rectangle Y2 (not used for cross cursor)
"ActiveCursors" : Activate/Desactivate the Red, Green and blue pointers (Use GetValue first to get the current array) :
Available options are ACTIVATE_CURSOR=1 and/or ACTIVATE_BASELINE=2 (these 2 constants can be added)
index 0 : Pointer Red
index 1 : Pointer Green
index 2 : Pointer Blue
"SpecialScriptType" : Add custom privileges to the current script (HJY scripts only)
"CCDShutter" : (LS6 Only) Change CCD Shutter mode :
SHUTTER_AUTO=0 : CCD Shutter synchronized with the acquisition
SHUTTER_OPEN=1 : CCD Shutter always open
SHUTTER_OPEN_CLOSED=2 : CCD Shutter Auto if expo time>0.5s, Open otherwise
"ShutterMode" : Laser Shutter Mode :
SHUTTER_AUTO=0 : Shutter open during acuquisition
SHUTTER_MANUAL=1 : Manual shutter
"SWIFT" : Change SWIFT mode :
SWIFT_ON=1 : Enable SWIFT
SWIFT_OFF=0 : Disable SWIFT
"LabSpecExposureTime" : Change LabSpec interface Exposure time
"LabSpecAccumulation" : Change LabSpec interface Number of accumulations
"ActiveFrame" : Change LabSpec interface active main frame (Spectrum, Video, 3D..)
"AuxDeviceValue" : Report the Aux. device current value (see example below).
Aux. device script path must be set in the Extra image section.
The Aux. device script is then automatically launched when required by LabSpec, and must update the "AuxDeviceValue" with the current intensity.
"SWIFT" : Check if SWIFT mode is enabled.
pValue=-1 if the current configuration is not SWIFT compatible.
pValue=0 if the current configuration is compatible, but SWIFT is disable.
pValue=1 if SWIFT is enable

pValue : Array of values

Return Values :

0 : Succeeded
-1 : Failed

Constant List

```
Const ACTIVATE_CURSOR = 1
Const ACTIVATE_BASELINE = 2
Const SHUTTER_AUTO = 0
Const SHUTTER_MANUAL = 1
Const SWIFT_OFF = 0
Const SWIFT_ON = 1
Const CURSOR_LINE = 0
Const CURSOR_DOUBLE = 1
Const CURSOR_CROSS = 2
Const CURSOR_INTEGRAL = 3
Const CURSOR_POINT = 0
Const CURSOR_RECTANGLE = 1
Const CURSOR_ELIPSE = 2
```

Example

This Example Opens a spectrum file,
Gets XY (both Frequensy and intensity) data from spectrum,
Inverse the data,
Put the inverted data to the spectrum,
Save the spectrum

```
Dim SpectrumID
Dim SpectrumValues
OpenSpectrum()
GetSpectrum()
InverseSpectrum()
PutXYSpectrum()
SaveSpectrum()
Private Sub OpenSpectrum()
  Dim FileName
  FileName ="C: est.tsf"
  SpectrumID = LabSpec.Load(FileName)
End Sub
Private Sub GetXYSpectrum()
  LabSpec.GetValue ID,"XYData",SpectrumValues
End Sub
// Inverse Spectrum example
Private Sub InverseSpectrum()
  Dim tmpSpectrum()
  Redim tmpSpectrum(UBound(SpectrumValues,2))
    For i = 0 To UBound(SpectrumValues,2)
        tmpSpectrum(i)=SpectrumValues(1,i)
    Next
    For i = 0 To UBound(SpectrumValues,2)
        SpectrumValues(1,i)=tmpSpectrum(UBound(SpectrumValues,2)-i)
    Next
End Sub
Private Sub PutXYSpectrum()
  LabSpec.PutValue ID,"XYData",SpectrumValues
End Sub
Private Sub SaveSpectrum()
  Dim FileName
  Dim Format
  Dim Ret
  FileName ="C: est2.tsf"
  Ret = LabSpec.Save(SpectrumID, FileName, Format)
End Sub
```

Aux. device script.

```
   ' This script will return a random emulation value for the Aux. device.
   ' An hardware script must read the current value from the electronics and update the
AuxDeviceValue
   Randomize
   Val=Rnd(1)
   LabSpec.PutValue 0,"AuxDeviceValue",Val
```

See Also : GetValue

Treat

Description : Treatment function (filtering, peak fitting, baseline removal)
Keywords : filter peak fitting treatment baseline
Type : AutoVBSAct
Category : Data Manipulation

---

long Treat(**long** ID, **LPCTSTR** FunctionName, **long** FunctionMode, **VARIANT FAR*** Param1, **VARIANT FAR*** Param2, **VARIANT FAR*** Param3, **VARIANT FAR*** Param4, **VARIANT FAR*** Param5, **VARIANT FAR*** Param6)

ID : Spectrum ID

FunctionName : Treatment Function Name :

Filter

FunctionName "Filter" : Filtration routines

FunctionMode
0 - FILTER_START : Start filter with specified parameters
return -1 : Unknown filter
return 0 : OK

1 - FILTER_GET_STATE : Get Filter State
return -1 : Filtration in progress
return 0 : Filtration done

Param1 - Filter Type
0 - FILTER_SMOOTH : Smoothing filter
1 - FILTER_DER1 : 1st order Derivative Filter
2 - FILTER_DER2 : 2nd order Derivative Filter
3 - FILTER_MEDIAN : Median Filter
4 - FILTER_FFT : FFT Filter
5 - FILTER_DENOISER : Denoiser Filter
10 - FILTER_INTERPOLATE : Interpolation filter

Param2 - Degree (int)

Param3 - Size (int)

Param4 - Binning (FILTER_INTERPOLATE only) (int) 0:polynomial interpolation - 1:binning interpolation

Param5 - Factor (FILTER_FFT and FILTER_DENOISER only) (double) 0.0 - 100.0
SizeY (int) (FILTER_INTERPOLATE only)

Constants List

```
Const FILTER_START = 0
Const FILTER_GET_STATE = 1
Const FILTER_SMOOTH = 0
Const FILTER_DER1 = 1
Const FILTER_DER2 = 2
Const FILTER_MEDIAN = 3
Const FILTER_FFT = 4
Const FILTER_DENOISER = 5
Const FILTER_INTERPOLATE = 10
```

Example

```
Dim ID
Dim Degree
Dim Size
Dim Status
Dim FilterType
Dim FunctionName
Dim FunctionMode
Dim Param3
Dim Param4
Dim Param5
Dim Param6
ID=LabSpec.GetActiveData("Spectrum")
if (ID>0) then
   FunctionName="Filter"
   FunctionMode=FILTER_START
   FilterType=FILTER_SMOOTH
   Degree=2
   Size=3
   if LabSpec.Treat(ID,FunctionName,FunctionMode,FilterType,Degree,Size,Param4,Param5,Param6)=0 then
         FunctionMode=FILTER_GET_STATE
         do
           Status=LabSpec.Treat
(ID,FunctionName,FunctionMode,Param1,Param2,Param3,Param4,Param5,Param6)
         Loop Until Status=0
         LabSpec.Message "Filter done.",MB_OK
   else
         LabSpec.Message "Filter failed.",MB_OK
   end if
End if
```

Peak Fitting

FunctionName "PeakFitting" : Peak Fitting routine

FunctionMode
0 - PEAKFIT_ADD_PEAK : Add a new Peak to Fit
return -1 : Unknown Spectrum ID
return 0 : OK

Param1 - Peak ID : (int) 0 to NbPeaks

Param2 - Peak Position : (float) display unit

Param3 - Peak Intensity : (float) count

Param4 - Peak Width : (float) nm

Param5 - Gauss/Loren Ratio : (float)

Param6 - Formula : (string) : "Loren()", "Gauss()" or "GaussLoren()"

(LS6 only) If Peak Intensity, width or Ratio is <0, peak pramaters will be approximated

FunctionMode
1 - PEAKFIT_START : Start Fitting
return -1 : Unknown Spectrum ID
return 0 : OK

Param1 - BaseLine correction : (int) PEAKFIT_NO_BASELINE 0 : No - PEAKFIT_BASELINE 1 : Yes - (LS6 only) PEAKFIT_USE_EXISTING_BASELINE : Use already set baseline

Param2 - Number of Iteration: (int) 0 - Default

Param3 - Max Peak Shift: (double) 0 - Default

Param4 - Min Peak Width: (double) 0 - Default

Param5 - Max Peak Width: (double) 0 - Default

Param6 - Baseline Formula : (string) : if not specified, default value is : "b+c*xn+d*xn*xn"

FunctionMode
2 - PEAKFIT_GET_RESULT: Get Fitting State and Results
return -1 : Unknown Spectrum ID
return 0 : OK

Param1 - Peak ID : (int) 0 to NbPeaks

Param2 - Peak Position : (float) nm

Param3 - Peak Intensity : (float) count

Param4 - Peak Width : (float) nm

Param5 - Gauss/Loren Ratio : (float)

Param6 - Peak Area : (float)

FunctionMode
3 - PEAKFIT_FIX: Fix / unFix a specified fitting variable
return -1 : Unknown Spectrum ID
return 0 : OK

Param1 - Peak ID : (int) 0 to NbPeaks

Param2 - Var to Fix / unFix: (str) "P", "A", "W", "G", "B", "C" or "D"

Param3 - Fix value : 1 to Fix, 0 to unFix

FunctionMode
4 - PEAKFIT_ADD_BASELINE: Fit a Baseline
return -1 : Unknown Spectrum ID
return 0 : OK

Param1 - Baseline degree

Param2 - Baseline Max points

Param3 - Substraction (0: fit, 1: fit + substract)

Constants List

```
Const PEAKFIT_ADD_PEAK = 0
Const PEAKFIT_START = 1
Const PEAKFIT_GET_RESULT = 2
Const PEAKFIT_FIX = 3
Const PEAKFIT_ADD_BASELINE = 4
Const PEAKFIT_NO_BASELINE = 0
Const PEAKFIT_BASELINE = 1
Const PEAKFIT_USE_EXISTING_BASELINE = 2
```

Example

```
Dim ID
Dim FunctionName
Dim FunctionMode
Dim PeakNum
Dim PeakPosition
Dim PeakIntensity
Dim PeakWidth
Dim GLRatio
Dim Formula
Dim Param3
Dim Param4
Dim Param5
Dim Param6
Dim BaseLine
Dim Iteration
FunctionName="PeakFitting"
FunctionMode=PEAKFIT_ADD_PEAK
PeakNum=0
PeakPosition=510.08
PeakIntensity=2000.0
PeakWidth=0.0005
GLRatio=0.5
Formula="GaussLoren()"
ID=LabSpec.GetActiveData("Spectrum")
if (ID>0) then
    if LabSpec.Treat(ID,FunctionName,FunctionMode,PeakNum,PeakPosition,PeakIntensity,PeakWidth,GLRatio,For
then
        FunctionMode=PEAKFIT_START ' Start Fitting
        BaseLine=PEAKFIT_BASELINE
        Iteration=25
        LabSpec.Treat ID,FunctionName,FunctionMode,BaseLine,Iteration,Param3,Param4,Param5,Param6

        FunctionMode=PEAKFIT_GET_RESULT
        Do

Status=LabSpec.Treat(ID,FunctionName,FunctionMode,PeakNum,PeakPosition,PeakIntensity,PeakWidth,GLRatio,I
        Loop Until Status=0

        LabSpec.Message "Peak Position : " & PeakPosition,MB_OK
    End if
End if
```

Remove Baseline

FunctionName "RemoveBaseline" : Remove baseline

FunctionMode - Not used

Param1 - Polynomial degree : (int) degree

Param2 - Not used

Param3 - Not used

Param4 - Not used

Param5 - Not used

Param6 - Not used

Example

```
Dim ID
Dim FunctionName
Dim FunctionMode
Dim Degree
Dim Param2
Dim Param3
Dim Param4
Dim Param5
Dim Param6
FunctionName="RemoveBaseline"
Degree=2
ID=LabSpec.GetActiveData("Spectrum")
if (ID>0) then

LabSpec.Treat(ID,FunctionName,FunctionMode,Degree,Param2,Param3,Param4,Param5,Param6)
End if
```

GetDetectorZone

Description : Get Detector Active Zone and Binning
Keywords : acquisition CCD size zone
Type : AutoVBSAct
Category : Detector Control

---

long GetDetectorZone(**VARIANT FAR*** FromX, **VARIANT FAR*** ToX, **VARIANT FAR*** BinningX, **VARIANT FAR*** FromY, **VARIANT FAR*** ToY, **VARIANT FAR*** BinningY)

Get Current Detector Zone

FromX : X Start Position

ToX : X Stop Position

BinningX : X Binning

FromY : Y Start Position

ToY : Y Stop Position

BinningY : Y Binning

Return values :

0 : Succeeded
-1 : Failed

Example :

This example retreives the current zone, and set a new one

```
Dim FromX
Dim ToX
Dim BinningX
Dim FromY
Dim ToY
Dim BinningY

LabSpec.GetDetectorZone FromX, ToX, BinningX, FromY, ToY, BinningY

FromX = 1
ToX = 1024
BinningX = 1
FromY = 1
ToY = 256
BinningY = 1

LabSpec.SetDetectorZone FromX, ToX, BinningX, FromY, ToY, BinningY
```

SetDetectorZone

Description : Set Detector Active Zone and Binning
Keywords : acquisition CCD size zone
Type : AutoActiveX
Category : Detector Control

---

long SetDetectorZone(**long** FromX, **long** ToX, **long** BinningX, **long** FromY, **long** ToY, **long** BinningY)
Set Current Detector Zone

FromX : X Start Position

ToX : X Stop Position

BinningX : X Binning

FromY : Y Start Position

ToY : Y Stop Position

BinningY : Y Binning

Return values :

0 : Succeeded
-1 : Failed

Example :

This example retreives the current zone, and set a new one

```
Dim FromX
Dim ToX
Dim BinningX
Dim FromY
Dim ToY
Dim BinningY

LabSpec.GetDetectorZone FromX, ToX, BinningX, FromY, ToY, BinningY

FromX = 1
ToX = 1024
BinningX = 1
FromY = 1
ToY = 256
BinningY = 1

LabSpec.SetDetectorZone FromX, ToX, BinningX, FromY, ToY, BinningY
```

AddID

Description : Add a data ID to save
Keywords : save multiple file
Type : AutoVBSAct
Category : File Management

long AddID(**long** ID)

Add a data ID to be saved. All data added with this function will be saved in the same file. (see Save())

ID : data ID to add to the save List

Return Values :

>0 : Nb of ID added
-1 : Too many ID added (max : 1000)

Example

```
LabSpec.AddID LabSpec.Load ("C: est1.tsf")
LabSpec.AddID LabSpec.Load ("C: est2.tsf")
LabSpec.AddID LabSpec.Load ("C: est3.tsf")
LabSpec.AddID LabSpec.Load ("C: est4.tsf")
LabSpec.Save 0,"C:saveall.tsf","tsf"
```

See Also : Save()

Load

Description : Load data from file
Keywords : data disk file
Type : AutoVBSAct
Category : File Management

---

long Load(**LPCTSTR** pFileName)

Load a data. Use LoadAll() if more than one data is included in the file.

pFileName : Data FileName

Return Values :

>0 : Data ID
-1 : Failed

Example

```
Private Sub OpenSpectrum()
  Dim FileName
  Dim SpectrumID
  FileName ="C: est.tsf"
  SpectrumID = LabSpec.Load(FileName)
End Sub
```

See Also : Save - GetActiveData - LoadAll

LoadAll

Description : Load all data from a File
Keywords : load multiple data disk file
Type : AutoVBSAct
Category : File Management

VARIANT Load(**LPCTSTR** pFileName)

Load a data

pFileName : Data FileName

Return Values :

Array of Data IDs

Example

```
Private Sub OpenSpectrum()
  Dim FileName
  Dim SpectraID
  Dim i
  FileName ="C: est.tsf"
  SpectraID = LabSpec.LoadAll(FileName)

  for i=0 to Ubound(SpectraID)
    LabSpec.Message SpectraID(i) , MB_OK
  next
End Sub
```

See Also : Save - Load - GetActiveData

Print

Description : Print the active Area
Keywords : print preview page
Type : AutoActiveX
Category : File Management

long Print(**long** Mode)

Print the active area

Mode : Print Mode :
0 : FILE_PRINT_PREVIEW Lauch the Print preview dialog
1 : FILE_PRINT_PAGE_SETUP Lauch the Page Setup preview dialog
2 : FILE_PRINTER_SETUP Lauch the Printer Setup preview dialog
3 : FILE_PRINT Print the active area

Constants List :

```
Const FILE_PRINT_PREVIEW = 0
Const FILE_PRINT_PAGE_SETUP = 1
Const FILE_PRINTER_SETUP = 2
Const FILE_PRINT = 3
```

Save

Description : Save a Spectrum to a file
Keywords : data disk file
Type : AutoVBSAct
Category : File Management

---

long Save(**long** ID, **LPCTSTR** pFileName, **LPCTSTR** pFormat)

Save loaded data.

ID : Data ID. if 0, Save() will save all the data ID added with AddID() in the same file.

pFileName : Data FileName (including full path and extention)

pFormat : Data Format (i.e. "ngc", "tsf" ..) if empty ("") Save will use the FileName extention

Return Values :

0 : Succeeded
-1 : Failed

Example

```
Dim SpectrumID
OpenSpectrum()
Private Sub SaveSpectrum()
  Dim FileName
  Dim Format
  Dim Ret
  FileName = "C: est.tsf"
  Format = "tsf"
  Ret = LabSpec.Save(SpectrumID, FileName, Format)
End Sub
```

See Also : Load - GetActiveData

Template

Description : Load / Save templates
Keywords :
Type : AutoActiveX
Category : Instrument Configuration

---

long Template(**long** Mode, **LPCTSTR** Param, **VARIANT FAR*** Value)

Load / Save a template.

Mode : Template Mode :
0 : LOAD_TEMPLATE Load a template from the template list
1 : SAVE_TEMPLATE Save a template to the template list
2 : ADD_TO_TEMPLATE_LIST Add the template to the custom template list
3 : TEMPLATE_SET_VALUE Set a Value for the specified parameter
4 : TEMPLATE_GET_VALUE Get the parameter current value
5 : APPLY_TEMPLATE Apply the specified template
6 : GET_TEMPLATE_STATUS Check apply status
7 : SAVE_CURRENT_CONFIG Save current config as a template
8 : TEST_TEMPLATE Test if the template plugin is loaded and if the specified template is available on the system
9 : APPLY_TEMPLATE_NO_SPECTRO Apply the specified template without moving the spectrometer

Param : Parameter name :
LOAD_TEMPLATE, SAVE_TEMPLATE, ADD_TO_TEMPLATE_LIST, APPLY_TEMPLATE : Template Name
TEMPLATE_SET_VALUE and TEMPLATE_GET_VALUE :
"Laser"
"Filter"
"Accum"
"Spike"
"AutoExposure"
"AutoFocus"
"Grating"
"RamanPolarizer"
"LaserPolarizer"
"CentralPosition"
"PositionFrom"
"PositionTo"
"Expo"
"Hole"
"Slit"
"AutoExposureMin"
"AutoExposureMax"
"ExtendedRange"
"Objective"

<u>Return Values :</u>
GET_TEMPLATE_STATUS : return 1 while applying

GET_TEMPLATE_STATUS : return 0 when done

Constants List :

```
Const LOAD_TEMPLATE  = 0
Const SAVE_TEMPLATE  = 1
Const ADD_TO_TEMPLATE_LIST  = 2
Const TEMPLATE_SET_VALUE  = 3
Const TEMPLATE_GET_VALUE  = 4
Const APPLY_TEMPLATE  = 5
Const GET_TEMPLATE_STATUS  = 6
Const SAVE_CURRENT_CONFIG  = 7
Const TEST_TEMPLATE  = 8
Const APPLY_TEMPLATE_NO_SPECTRO = 9
```

Example

```
    ' Load Default Values
    LabSpec.Template LOAD_TEMPLATE,"Default",Param ' Load Default Values
    ' Set New Values
    labSpec.Template TEMPLATE_SET_VALUE,"Laser",LaserIndex
    labSpec.Template TEMPLATE_SET_VALUE,"Filter",FilterIndex
    labSpec.Template TEMPLATE_SET_VALUE,"Accum",NbAccumulation
    labSpec.Template TEMPLATE_SET_VALUE,"Spike",SpikeRemoving
    labSpec.Template TEMPLATE_SET_VALUE,"AutoExposure",AutoExposure
    labSpec.Template TEMPLATE_SET_VALUE,"AutoFocus",Autofocus
    labSpec.Template TEMPLATE_SET_VALUE,"Grating",AcqGrating
    labSpec.Template TEMPLATE_SET_VALUE,"RamanPolarizer",Polarizer1Index
    labSpec.Template TEMPLATE_SET_VALUE,"LaserPolarizer",Polarizer2Index
    labSpec.Template TEMPLATE_SET_VALUE,"CentralPosition",CentralPosition
    labSpec.Template TEMPLATE_SET_VALUE,"PositionFrom",AcqFrom
    labSpec.Template TEMPLATE_SET_VALUE,"PositionTo",AcqTo
    labSpec.Template TEMPLATE_SET_VALUE,"Expo",ExposureTime
    labSpec.Template TEMPLATE_SET_VALUE,"Hole",HoleValue
    labSpec.Template TEMPLATE_SET_VALUE,"Slit",SlitValue
    labSpec.Template
TEMPLATE_SET_VALUE,"AutoExposureMin",AutoExposureFrom
    labSpec.Template TEMPLATE_SET_VALUE,"AutoExposureMax",AutoExposureTo
    labSpec.Template TEMPLATE_SET_VALUE,"ExtendedRange",ExtendedRange
    labSpec.Template TEMPLATE_SET_VALUE,"Objective",Objective
    ' Save Template
    LabSpec.Template SAVE_TEMPLATE,"MyTemplate",Param
    ' Apply Template
    LabSpec.Template APPLY_TEMPLATE,"MyTemplate",Param
    do
      Status=LabSpec.Template GET_TEMPLATE_STATUS,"MyTemplate",Param
    Loop until Status=0
    LabSpec.Message "Template applied",MB_OK
```

Dialog

Description : Show a custom dialog
Keywords :
Type : AutoVBSLS6
Category : Messages and Annotation

VARIANT Dialog(**long** Mode, **LPCTSTR** ItemType, **LPCTSTR** ItemName, **LPCSTR** ItemValue, **LPCSTR** Options)

Create and show a custom Dialog. LabSpec 6 only.

Mode : Mode
DIALOG_ADD_ITEM=0 : Add a new item to the custom dialog
DIALOG_SHOW=1 : Show the custom dialog
DIALOG_GET_VAR=2 : Get the current value of a specific item inside the dialog
DIALOG_GET_STATUS=3 : For dynamic dialog only : Check if the dialog has been closed
DIALOG_UPDATE_VAR=4 : For dynamic dialog only : Update a item value

ItemType : Item Type
"Dialog" : Main dialog, this has to be the first item
"Label" : Non editable text
"Edit" : Editable text (Edit box)
"ComboBox" : Drop-down list
"Checkbox" : Check box
"Radio" : Radio button
"Button" : Button
"BeginTable" : Invisible table with a fixed number of colums
"EndTable" : End of table
"BeginFrame" : Frame with a fixed number of colums
"EndFrame" : End of Frame

ItemName : Item Name. This name must be unique, and is required to get or update the item value
ItemName must be identical for each group of radio buttons.

ItemValue : Item Value.
For Edit, Label and Button : Displayed text
For CheckBox : 1=checked, 0=unchecked
For Radio: Variable name. Must be identical of a given group of radio buttons.
For ComboBox : List of selectable items, separated by ";"

Options : Item options, separated by ";"
For Dialog : Width:xx = minimum dialog width; Heigth:xx minimum dialog height. If not specified, the dialog size will be adjusted automatically; NonModal:1 = Non modal dialog
For BeginTable and BeginFrame: Cols:xx = number of columms
For BeginFrame: Style:Regular (minimizing frame width header) - Small (minimizing frame without header)

or Simple (fixed frame)
For BeginFrame: Minimized:Yes/No (default minimizing style)
For Edit: Width:xx = Edit box width; Disable:1 = Read only
For ComboBox: Selected:xx = default selection (0 based index)
For Radio: Index:xx = Radio button index (0 based) ; Selected:xx index of the selected radio by default
For Button: CloseDialog:Yes/No ; Disable:1 = greyed out
If Yes, a button click will close the dialog and return the associated Return Value (see below)
If No, a button click will generate an event without closing the dialog. (Dynamic dialog only). See DIALOG_GET_VAR to monitor the event.
For Button: ReturnValue:xx = returned value when the button is pressed. (if CloseDialog option is set to "Yes" only).

Returned Value :
if DIALOG_SHOW : Return -1 while the dialog is open, and the ReturnValue set for the buttons used to clse the dialog. A click to the upper right dialog close icon will return 0.

if DIALOG_GET_VAR : Return the current value of the specified item.

```
Const DIALOG_ADD_ITEM = 0
Const DIALOG_SHOW = 1
Const DIALOG_GET_STATUS = 3
Const DIALOG_GET_VAR = 2
Const DIALOG_UPDATE_VAR = 4
```

A simple input dialog (non dynamic)

```
Const DIALOG_ADD_ITEM = 0
Const DIALOG_SHOW = 1
Const DIALOG_GET_VAR = 2
Const MB_OK = 0
Const MB_ICONERROR = 60
' Create a new custom dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "Dialog", "My Dialog", "","Width:0"
' Insert a table To the dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "BeginTable", "", "","Cols:2"
' Insert your items here
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "prompt", "Please enter your value : ",""
' Insert a label
LabSpec.Dialog DIALOG_ADD_ITEM, "Edit", "MyEdit", "0","" ' Insert an edit box
' Close the table
LabSpec.Dialog DIALOG_ADD_ITEM, "EndTable", "", "",""
LabSpec.Dialog DIALOG_ADD_ITEM, "Button", "OK", "OK","Close:Yes;ReturnValue:1"
' Insert a button
' Show the static dialog. ReturnedValue = button pressed
ReturnedValue=LabSpec.Dialog(DIALOG_SHOW, "", "", "", "")
MyValue=LabSpec.Dialog(DIALOG_GET_VAR, "", "MyEdit", "","") ' Get an item value
If ReturnedValue=1 Then LabSpec.Message "User Value : " & MyValue,MB_OK
If ReturnedValue=0 Then LabSpec.Message "Operation canceled",MB_ICONERROR
```

A dynamic calculator dialog

```
Const MB_OK = 0
Const MB_ICONERROR = 60
Const DIALOG_ADD_ITEM = 0
Const DIALOG_SHOW = 1
Const DIALOG_GET_VAR = 2
Const DIALOG_GET_STATUS = 3
Const DIALOG_UPDATE_VAR = 4
' Create a new custom dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "Dialog", "Dynamic calculator", "","Width:30;Height:0"
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "MyLabel", "Select your data","" ' Insert a label
' Insert a table To the dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "BeginTable", "", "","Cols:3"
' Insert your items here
LabSpec.Dialog DIALOG_ADD_ITEM, "Edit", "Operator", "0","" ' Insert an edit box
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "times", "x","" ' Insert a label
LabSpec.Dialog DIALOG_ADD_ITEM, "ComboBox", "TimesCombo", "0;1;2;3;4;5","Selected:1" ' Insert a
combobox
' Close the table
LabSpec.Dialog DIALOG_ADD_ITEM, "EndTable", "", "",""
' Insert a frame To the dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "BeginFrame", "More Options",
"","Cols:1;Style:Regular:Minimized:Yes"
' Insert a table To the dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "BeginTable", "", "","Cols:2"
' Insert your items here
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "Add100", "Add 100","" ' Insert a label
LabSpec.Dialog DIALOG_ADD_ITEM, "CheckBox", "CheckAdd100", "0","" ' Insert a checkbox
' Close the table
LabSpec.Dialog DIALOG_ADD_ITEM, "EndTable", "", "",""
' Insert a table To the dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "BeginTable", "", "","Cols:6"
' Insert your items here
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "MyLabel2", "- 0","0" ' Insert a label
LabSpec.Dialog DIALOG_ADD_ITEM, "Radio", "SubRadio1", "SubRadio","Index:0" ' Insert a radio button
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "MyLabel3", "- 10","0" ' Insert a label
LabSpec.Dialog DIALOG_ADD_ITEM, "Radio", "SubRadio2", "SubRadio","Index:1" ' Insert a radio button
LabSpec.Dialog DIALOG_ADD_ITEM, "Label", "MyLabel4", "- 20","0" ' Insert a label
LabSpec.Dialog DIALOG_ADD_ITEM, "Radio", "SubRadio3", "SubRadio","Index:2" ' Insert a radio button
' Close the table
LabSpec.Dialog DIALOG_ADD_ITEM, "EndTable", "", "",""
' Close the frame
LabSpec.Dialog DIALOG_ADD_ITEM, "EndFrame", "More Options", "",""
LabSpec.Dialog DIALOG_ADD_ITEM, "Edit", "Result", "0","" ' Insert an edit box
' Insert a table To the dialog
LabSpec.Dialog DIALOG_ADD_ITEM, "BeginTable", "", "","Cols:2"
LabSpec.Dialog DIALOG_ADD_ITEM, "Button", "Reset", "Reset","CloseDialog:No;ReturnValue:1" ' Insert a
button
LabSpec.Dialog DIALOG_ADD_ITEM, "Button", "OK", "OK","CloseDialog:Yes;ReturnValue:1" ' Insert a
```

```
button
' Close the table
LabSpec.Dialog DIALOG_ADD_ITEM, "EndTable", "", "",""
' Show the dynamic dialog. ReturnedValue = button pressed
DialogStatus=LabSpec.Dialog(DIALOG_SHOW, "", "", "","Dynamic:Yes")
Do
        'Check and update the dialog items here
        Operator=LabSpec.Dialog(DIALOG_GET_VAR, "", "Operator", "", "") ' Get an item value
        TimesCombo=LabSpec.Dialog(DIALOG_GET_VAR, "", "TimesCombo", "", "") ' Get an item value
        CheckAdd100=LabSpec.Dialog(DIALOG_GET_VAR, "", "CheckAdd100", "", "") ' Get an item value
        SubRadio=LabSpec.Dialog(DIALOG_GET_VAR, "", "SubRadio", "", "") ' Get an item value
        Result=CDbl(Operator)*CDbl(TimesCombo)+100*CDbl(CheckAdd100)-10*CDbl(SubRadio)
        Reset=LabSpec.Dialog(DIALOG_GET_VAR, "", "Reset", "", "") ' Get an item value
        LabSpec.Dialog DIALOG_UPDATE_VAR, "", "Result", Result, "" ' Update an item
        If Reset=1 Then ' Reset button clicked => Reset the values
            LabSpec.Dialog DIALOG_UPDATE_VAR, "", "Operator", "0", "" ' Update an item
            LabSpec.Dialog DIALOG_UPDATE_VAR, "", "TimesCombo", "1", "" ' Update an item
            LabSpec.Dialog DIALOG_UPDATE_VAR, "", "CheckAdd100", "0", "" ' Update an item
            LabSpec.Dialog DIALOG_UPDATE_VAR, "", "SubRadio", "0", "" ' Update an item
    End If

        DialogStatus=LabSpec.Dialog(DIALOG_GET_STATUS, "","","", "")
Loop until DialogStatus>=0
FinalResult=LabSpec.Dialog(DIALOG_GET_VAR, "", "Result", "","") ' Get an item value
If DialogStatus=1 Then LabSpec.Message "Final result : " & FinalResult,MB_OK
If DialogStatus=0 Then LabSpec.Message "Operation canceled",MB_ICONERROR
```

Message

Description : Display a Message Box
Keywords : messageBox MsgBox show statusbar
Type : AutoVBS
Category : Messages and Annotation

---

long Message(**LPCTSTR** Message, **long** Type)

Display a message in a Message Box or in the Status Bar
For advanced messaging functions (Input box, Open/Save file) Please see MessageEx()

pMessage : Message to display

Type : MessageBox Type :

Button :
MB_OK=0 - Single OK Button
MB_OKCANCEL=1 - OK/Cancel Buttons
MB_YESNO=2 - Yes/No Buttons
MB_YESNOCANCEL=3 - Yes/No/Cancel Buttons
MB_RETRYCANCEL=4 - Retry/Cancel Buttons
MB_ABORTRETRYIGNORE=5 - Abort/Retry/Ignore Buttons
MB_STATUS_BAR=6 - Display message in the status bar instead of a MessageBox
MB_WAIT_FOR_EVENT=7 - Popup a non Modal Message Box (allow LabSpec to run background operations i.e live video)
MB_NON_BLOCKING=8 - Popup a non Modal Message Box and returns immediatly.
Call MB_NON_BLOCKING again with Message="" to test if popup is still open
Call MB_NON_BLOCKING again with Message="#" to close the popup window
MB_MULTI_BUTTONS=9 - Popup a Modal Message Box with up to 10 buttons. Message text and Button texts are separated by # (i.e. MyMessage#Button1#Button2#Button3)

Icon :
MB_ICONEXCLAMATION=10 - exclamation-point icon
MB_ICONINFORMATION=20 - lowercase letter i in a circle
MB_ICONQUESTION=30 - question-mark icon
MB_ICONSTOP=40 - stop-sign icon
MB_ICONWARNING=50 - warning icon
MB_ICONERROR=60 - error icon
MB_ICONHAND=70 - hand icon

<u>Return Values :</u>

0 : ID_OK
1 : ID_YES
2 : ID_NO
3 : ID_CANCEL

4 : ID_ABORT
5 : ID_IGNORE
6 : ID_RETRY

if MB_NON_BLOCKING :
0 : Popup closed
-1 : Popup open

if MB_MULTI_BUTTONS :
Pressed Button
0 if Canceled

Constants List :

```
Const MB_OK = 0
Const MB_OKCANCEL = 1
Const MB_YESNO = 2
Const MB_YESNOCANCEL = 3
Const MB_RETRYCANCEL = 4
Const MB_ABORTRETRYIGNORE = 5
Const MB_STATUS_BAR = 6
Const MB_WAIT_FOR_EVENT = 7
Const MB_NON_BLOCKING = 8
Const MB_MULTI_BUTTONS= 9
Const MB_ICONEXCLAMATION = 10
Const MB_ICONINFORMATION = 20
Const MB_ICONQUESTION = 30
Const MB_ICONSTOP = 40
Const MB_ICONWARNING = 50
Const MB_ICONERROR = 60
Const MB_ICONHAND = 70
Const IDOK = 0
Const IDYES = 1
Const IDNO = 2
Const IDCANCEL = 3
Const IDABORT = 4
Const IDIGNORE = 5
Const IDRETRY = 6
```

Example

```
Dim Button
Button = LabSpec.Message ("Do You Want To Continue
?",MB_YESNO+MB_ICONQUESTION)
if Button=IDYES then LabSpec.Message "Continue !",MB_ICONEXCLAMATION
if Button=IDNO then LabSpec.Message "STOP !",MB_ICONWARNING
```

```
ret=LabSpec.Message("Click to stop RTD",MB_NON_BLOCKING) ' Show message. Function will return
immediatly
Do
   LabSpec.Acq ACQ_SPECTRUM_RTD+ACQ_AUTO_SHOW, 1, 1, 0, 0
   Do
      SpectrumID = LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
   Loop Until SpectrumID > 0
   ret=LabSpec.Message("",MB_NON_BLOCKING) ' Test is message is still open (-1) or closed by tne
operator (0)
Loop Until ret = 0
```

MessageEx

Description : Extended Message function
Keywords : open save inputbox dialog
Type : AutoVBS
Category : Messages and Annotation

BSTR MessageEx(**LPCTSTR** Message, **long** Type)

Open an advanced message dialog (input box, open/save file).
For regular messaging function, please see Message()

pMessage : MB_INPUTBOX=0 : Message to display (default value can be set after "#" i.e: "Enter Test Band :#520.7")
MB_OPEN_FILE=1 - Open File Extention (i.e. "txt")
MB_SAVE_FILE=2 - Save File Extention (i.e. "txt")
MB_FROM_TO_INPUT=4 : Message to display (default value can be set after "#" i.e: "Enter limits#100#200#Security limits")
MB_BROWSE_FOR_FOLDER=5 - Browse dialog message
MB_TRACE_DEBUG=13 - DebugFileName;DebugText (i.e : "c:MyDebugFile.txt;MyDebugInformation")

Type : MessageBox Type :

Button :
MB_INPUTBOX=0 - InputBox, ask the user to enter some informations
MB_OPEN_FILE=1 - Display windows standard open file box
MB_SAVE_FILE=2 - Display windows standard save file box
MB_FROM_TO_INPUT=4 - From To InputBox, ask the user to enter some informations
MB_BROWSE_FOR_FOLDER=5 - Display windows standard browse for folder box
MB_TRACE_DEBUG=13 - Store debug information into a text file
MB_STOP=14 - Is stop button clicked ?

Return Values :

MB_INPUTBOX=0 : User Message
MB_OPEN_FILE=1 : File path
MB_SAVE_FILE=2 : File path
MB_FROM_TO_INPUT=4 : From#To
MB_BROWSE_FOR_FOLDER=5 : Folder path
MB_TRACE_DEBUG=13 :
STOP=14 : 1 if stop button clicked, 0 otherwise

Constants List :

```
Const MB_INPUTBOX = 0
Const MB_OPEN_FILE = 1
Const MB_SAVE_FILE = 2
Const MB_FROM_TO_INPUT = 4
Const MB_BROWSE_FOR_FOLDER = 5
Const MB_TRACE_DEBUG = 13
Const MB_STOP = 14
```

Example

```
Dim FilePath
FilePath=LabSpec.MessageEx ("txt",MB_OPEN_FILE)
LabSpec.MessageEx "File to open : " & FilePath,MB_OK
```

Paint

Description : Draw and export to WMF
Keywords : picture area box message
Type : AutoVBSAct
Category : Messages and Annotation

---

long Paint(**long** Mode, **long** SpectrumID, **float** Value, **double** PosX, **double** PosY, **double** SizeX, **double** SizeY, **LPCTSTR** Text)

Paint can draw a text box on a window (up to 10), linked by an arrow to a spectrum.
Export to Windows Meta File (WMF) also available.
LS5 only.

Mode : Draw Mode
0 - ADD_BOX : Add a Box to the area
1 - REMOVE_BOXES : Remove all boxes
2 - EXPORT_WMF : If Value=0, Export the active window to a WMF file. If Value=1, export SpectrumID to a WMF file. Destination path must exist.
3 - ACTIVATE_WINDOW : Active a present window by type (ie "Map", "Point", "SpIm",..)

SpectrumID : Spectrum ID (Only for ADD_BOX)

Value :
ADD_BOX : Frequency value (destination of the arrow). Intensity level is automatically detected
EXPORT_WMF :
SHOW_SINGLE = Only show the active spectrum
SHOW_OVERLAY = Show all spectra on the window
SHOW_CURSORS = Show cursors. (optionnally add to SHOW_SINGLE or SHOW_OVERLAY)

PosX : Box Upper corners position (in % of the full window)

PosY : Box Left corners position (in % of the full window)

SizeX : Box X Size (in % of the full window)

SizeY : Box Y Size (in % of the full window)

Text :
ADD_BOX : Text to be displayed in the box
EXPORT_WMF : WMF Full path and file name (i.e : "c:ExportsMyWMF.wmf")

Return Values :
0 : OK
-1 : An error has occured

if ACTIVATE_WINDOW : Returns Active DataID

Constant List

```
Const ADD_BOX = 0
Const REMOVE_BOXES  = 1
Const EXPORT_WMF = 2
Const ACTIVATE_WINDOW = 3
Const SHOW_SINGLE = 0
Const SHOW_OVERLAY = 1
Const SHOW_CURSORS = 10
```

TickCount

Description : Get a tick count
Keywords : GetTickCount
Type : AutoActiveX
Category : Messages and Annotation

---

long TickCount()

Retrieves the number of milliseconds that have elapsed since the system was started.
Equivalent to the C++ GetTickCount() function.

Return Values :

Elapsed time (ms)

GetMotorPosition

Description : Get Motor Position
Keywords : value step index motor position
Type : AutoVBS
Category : Motor Control

VARIANT GetMotorPosition(**LPCTSTR** MotorName, **long** Mode)

Get a current motor position

MotorName : Motor Name (see motor name list in MoveMotor())

Mode : Motor Mode.
0 : MOTOR_VALUE return motor value in its own unit (i.e. nm)
1 : MOTOR_STEP return motor value in step
2 : MOTOR_INDEX return motor index
3 : MOTOR_SIZE return Number of available positions
4 : MOTOR_FULL_SIZE return Number of positions (even if not physically present) LS6 only
10 : MOTOR_INDEXTOVALUE + Index return motor value according to the specified index
20 : MOTOR_INDEXTOSTRING + Index return motor string according to the specified index
30 : MOTOR_FULL_INDEXTOVALUE + Index return motor value according to the specified full index (return all potential values, even if not physically present) LS6 only
40 : MOTOR_FULL_INDEXTOSTRING + Index return motor string according to the specified index (return all potential values, even if not physically present) LS6 only
50 : MOTOR_INDEXTOSTEP + Index return motor step according to the specified index
Return Values :

Motor Position.

MOTOR_SIZE: -1 if motor not present. 0 for continuous motors (spectro..). NbOfPosition for discrete motors (laser, filter..)

Constants List :

```
Const MOTOR_VALUE = 0
Const MOTOR_STEP = 1
Const MOTOR_INDEX = 2
Const MOTOR_SIZE = 3
Const MOTOR_FULL_SIZE = 4
Const MOTOR_INDEXTOVALUE = 10
Const MOTOR_INDEXTOSTRING = 20
Const MOTOR_FULL_INDEXTOVALUE = 30
Const MOTOR_FULL_INDEXTOSTRING = 40
Const MOTOR_INDEXTOSTEP = 50
```

Example

```
Const MOTOR_VALUE=0
Dim MotorPosition
MotorPosition=LabSpec.GetMotorPosition("Spectro",MOTOR_VALUE)
LabSpec.Message "Spectro Position : " & MotorPosition & " nm",0
NbLaser=LabSpec.GetMotorPosition ("Laser", MOTOR_SIZE)
for i=0 to NbLaser-1
   Laser=LabSpec.GetMotorPosition ("Laser", MOTOR_INDEXTOVALUE+i)
   LabSpec.Message "Laser 1/" & NbLaser & " : " & Laser & " nm" , 0
next
```

See Also : GetMotorStatus - MoveMotor
To get a position name (i.e. microscope objective), use GetValueEx with CurrentPositionName param.

ManageTemperature

Description : Manage Linkam parameters
Keywords : cooling heating speed linkam temperature
Type : AutoVBSAct
Category : Motor Control

---

long ManageTemperature(**long** Mode, **double** HeatingSpeed, **double** HeatingTime, **double** CoolingSpeed, **double** CoolingTime, **double** HoldingTime)

Mode : Heating and Cooling Mode
CONSTANT_SPEED : Use constant speed (?C/min) during heating/cooling
CONSTANT_TIME : Use constant time (sec) for heating/cooling
FREE_TEMPERATURE : Free the cooling stage (other parameters are ignored)
BACKUP_SETTINGS : Backup the current settings (other parameters are ignored)
RESTORE_SETTINGS : Restore the backup settings (other parameters are ignored)

HeatingSpeed : Heating Speed (in ?C/min)

HeatingTime : Heating Time (in sec)

CoolingSpeed : Cooling Speed (in ?C/min)

CoolingTime : Cooling Time (in sec)

HoldingTime : Holding Time (in sec)

Return Values :

-1 : No linkam table found
0 : Succeeded

Constants List :

```
Const CONSTANT_SPEED = 0
Const CONSTANT_TIME = 1
Const FREE_TEMPERATURE = 2
Const BACKUP_SETTINGS = 3
Const RESTORE_SETTINGS= 4
```

MoveMotor

Description : Moving a specified motor
Keywords : script step value string index move motor
Type : AutoVBSAct
Category : Motor Control

long MoveMotor(**LPCTSTR** MotorName, **double** PositionValue, **LPCTSTR** PositionName, **long** Mode)

MotorName : Motor Name (see motor name list).

PositionValue : Value to reach.

PositionName : Position Name. (only for named position motors (i.e. microscope etc..)

Mode : Motor Mode
0 : MOTOR_VALUE set value in its own unit (i.e. nm) (PositionName is ignored)
1 : MOTOR_STEP set value in step (PositionName is ignored)
2 : MOTOR_INDEX set motor index (PositionName is ignored)
3 : MOTOR_STRING set motor using the string (PositionValue is ignored)
4 : MOTOR_CALIBRATE Calibrate Motor (position is ignored)
5 : MOTOR_ORIGIN Set current position as origin position
6 : MOTOR_STOP Stop the specified motor
7 : VALUE_TO_STEP Convert Value (PositionValue) to Step (Return Value). Does not move the motor
10: MOTOR_NO_WAIT if added to the value, the motor will start, and the command will return immedialty.
100: MOTOR_NO_MESSAGE if added to the value, a manual motor will not prompt to change the position.

Return Values :

>0 : MoveID (see GetMotorStatus)
-1 : Failed

if VALUE_TO_STEP : return Motor Step

Constants List :

```
Const MOTOR_VALUE = 0
Const MOTOR_STEP = 1
Const MOTOR_INDEX = 2
Const MOTOR_STRING = 3
Const MOTOR_CALIBRATE = 4
Const MOTOR_ORIGIN = 5
Const MOTOR_STOP = 6
Const VALUE_TO_STEP = 7
Const MOTOR_NO_WAIT = 10
Const MOTOR_NO_MESSAGE = 100
```

Motor Name List :

```
Motor Name : Description (value unit) (internal unit)
"Spectro" : Spectrometer motor (nm) (step)
"Premono"    : ForeMonochromator (nm) (step)
"Grating" : Grating motor (gr/mm) (index)
"Slit"    : Slit motor (um) (step)
"Hole"    : Hole motor (um) (step)
"X"       : X Stage direction motor (um) (step) Active device
"Y"       : Y Stage direction motor (um) (step) Active device
"XT"      : X Stage direction motor (um) (step) Active Stage (LS6 only)
"YT"      : Y Stage direction motor (um) (step) Active Stage (LS6 only)
"XL"      : X Stage direction motor (um) (step) Active Scanning device (LS6 only)
"YL"      : Y Stage direction motor (um) (step) Active Scanning device(LS6 only)
"Z"       : Z motor (um) (step)
"Laser"   : Laser motor (nm) (index)
Other specific motors can be present depending on hardware configuration
```

```
Aramis Specific Motor Names (for advanced automation) :
"Aramis:Video"  : Video beam splitters (2 positions)
"Aramis:Micro90/Macro180" : Switch between Micro90 and Macro90 (2 positions)
"Aramis:Micro/Macro" : Switch between Micro and Macro (2 positions)
"Aramis:Raman Polarization" : Select Raman Polarization (3 positions)
"Aramis:Scrambler" : Select Scrambler On/Off or fiber entrance (3 positions)
"Aramis:Fiber Exit" : Select fiber exit (2 positions)
"Aramis:Microscope" : Select Micro or Macro (2 positions)
"Aramis:PinHole" : Move PinHole (2 positions)
"Aramis:Laser Polarization" : Select Laser Polarization (3 positions)
"Aramis:Laser Polarization2" : 2nd Laser Polarizer (3 positions)
"Aramis:Laser Polarization3" : 3rd Laser Polarizer (3 positions)
"Aramis:CCD/PMT" : Switch between 1st detector and PMT or 2nd detector (2
positions)
' These are direct access to laser motors. You should use the "Laser" motor instead of
these commutations
"Aramis:Laser HeNe-Other" : Switch between HeNe laser or other (2 positions)
"Aramis:Laser1/Other" : Switch between 1st laser or other (2 positions)
"Aramis:Laser3" : Select 3rd laser (2 positions)
T64000 Specific Motor Names :
"T64000:Slit1" : First int slit
"T64000:Slit2" : Second int slit
"T64000:First Stage Lateral Entrance" : Switch between Lateral/Axial
"T64000:Analysis mode" : Switch between Micro/Macro
"T64000:Spectrograph" : Switch between Single/Triple
"T64000:Premonochromator" : Switch between Subtractive/Additive
"T64000:Second/Third Stage" : Switch between Direct/Adaptation
"T64000:Detection System" : Switch between Multichannel/PMT-IGA
Other Motors :

"DetectorShutter" : Open/Close detector shutter. Instant motor, no need to use
GetMotorStatus
```

Example

```
Dim MoveID
' Move Spectro motor to 500 nm
Dim MoveID = LabSpec.MoveMotor("Spectro",500,"",MOTOR_VALUE)
Dim Status
' Wait Until Spectro Motor reached its position
do
   Status=LabSpec.GetMotorStatus("Spectro",MoveID)
Loop Until Status=0
```

```
MoveMotor()
WaitForAllMotors()
Dim MotorPosition
Dim MOTOR_VALUE=0
MotorPosition=LabSpec.GetMotorPosition("Spectro",MOTOR_VALUE)
LabSpec.Message "Spectro Position : " & MotorPosition
Private Sub MoveMotor()
  Dim MotorName
  Dim PositionValue
  Dim PositionName
  Dim MOTOR_VALUE
  MotorName="Spectro"
  PositionValue=500
  MOTOR_VALUE=0
  LabSpec.MoveMotor MotorName,PositionValue,PositionName,MOTOR_VALUE
End Sub
Private Sub WaitForAllMotors()
   Dim Status
   do
      Status=LabSpec.GetMotorStatus("",0)
   Loop Until Status=0
End Sub
```

See Also : GetMotorPosition - GetMotorStatus

MoveXY

Description : Move stage X and Y
Keywords : table simultaneous
Type : AutoVBSLS6
Category : Motor Control

---

long MoveXY(**double** X, **double** Y, **long** Mode)

Move X and Y simultaneously

X : X position (in um)
Y : Y position (in um)
Mode : Mode
STAGE_REL = 0 : Move to the XY position relative to the stage calibration
LASER_REL = 1 : Move to the XY position relative to the laser position (Green spot)

Constants List :

```
Const STAGE_REL = 0
Const LASER_REL = 1
```

SetScriptParamOptions

Description : Set configuration Options
Keywords : parameter validate show configuration
Type : AutoVBS
Category : Script Configuration

VBS Script only function

---

long SetScriptParamOptions(**LPCTSTR** Description, **long** Mode)

Validate parameters list, set Script description and config show options.

Note : Use SetSingleScriptParam before calling SetConfigOptions to set all the parameters.

Description : Script Short Description, to be displayed on the config page

ShowMode : Config dialog show options
0 : SHOW_ONCE Show the config dialog once (the first time the the script is launched)
1 : SHOW_ALWAYS Show config dialog each time the script is launched
2 : SHOW_NEVER Never show the config dialog

<u>Return Values :</u>

0 : First Time the script is launched (or Description has been modified)
1 : Script has already been launched before

<u>Constants List :</u>

```
Const SHOW_ONCE = 0
Const SHOW_ALWAYS = 1
Const SHOW_NEVER = 2
```

See SetSingleScriptParam to set parameters and to get an example

SetSingleScriptParam

Description : Set a single parameter
Keywords : configuration parameter
Type : AutoVBS
Category : Script Configuration

VBS Script only function

---

long SetSingleScriptParam(**LPCTSTR** Name, **LPCTSTR** Unit, **const VARIANT FAR&** Value, long Mode)

Set GUI accessible script parameters (up to 10 per script).
Set Internal (non GUI accessible) script parameters (up to 10 per script).

Name : Parameter Name

Unit : Parameter Unit or browse button
To add a Browse for folder button : Unit="BrowseForFolder;Dialog title msg"
To add a Browse to save file button : Unit="BrowseSaveFile;Dialog title msg;File Extention"
To add a Browse to open file button : Unit="BrowseOpenFile;Dialog title msg;File Extention"

Value : Parameters Default values. Parameter type will be checked on config.

Mode : Parameter Mode
0 - PARAM_DEFAULT : Set a param as default, and retreive the value from the config page
1 - PARAM_OVERWRITE : overwrite the parameter, even if it has already been saved in the config page
2 - PARAM_SAVE_INTERNAL : Save Internal Parameter (Use PARAM_SAVE_TO_FILE once all parameters are saved).
Internal parameter will not be accessible from GUI, but can be restored later from the script (for data backup).
3 - PARAM_RESTORE_INTERNAL : Restore Internal Parameter (Use PARAM_LOAD_FROM_FILE once before restoring values).
Parameters must be saved before beeing restored.
4 - PARAM_LOAD_FROM_FILE : Load Internal parameters list from file
5 - PARAM_SAVE_TO_FILE : Save Internal parameters list to file
6 - PARAM_REMOVE_INTERNAL : Remove parameter from the list

Constants List :

```
Const PARAM_DEFAULT = 0
Const PARAM_OVERWRITE = 1
Const PARAM_SAVE_INTERNAL = 2
Const PARAM_RESTORE_INTERNAL = 3
Const PARAM_LOAD_FROM_FILE = 4
Const PARAM_SAVE_TO_FILE = 5
Const PARAM_REMOVE_INTERNAL = 6
```

Return Values :

0 : First Time the parameter is set
1 : Parameter has already been set before

Example

```
Dim Param1
Param1=10.5  ' Will be set as float
Dim Param2
Param2=10    ' Will be set as integer
Dim Param3
Param3="test" ' Will be set as string
' Set default Values
LabSpec.SetSingleScriptParam "Param1","mm",Param1,PARAM_DEFAULT
LabSpec.SetSingleScriptParam "Param2","cm",Param2,PARAM_DEFAULT
LabSpec.SetSingleScriptParam "Param3","Label",Param3,PARAM_DEFAULT
' Set config Options and launch config
LabSpec.SetScriptParamOptions "Single Param (x3) Config Example", SHOW_ONCE
' Get parameters values after config has been launched (if not set, params will still be set to default previous value)
LabSpec.SetSingleScriptParam "Param1","mm",Param1,PARAM_DEFAULT
LabSpec.SetSingleScriptParam "Param2","cm",Param2,PARAM_DEFAULT
LabSpec.SetSingleScriptParam "Param3","Label",Param3,PARAM_DEFAULT
' Show Param3 modified value
LabSpec.Message Param3,MB_OK
```

See SetScriptParamOptions to validate parameters.

Pause

Description : Pause script execution
Keywords : wait sleep
Type : AutoVBS
Category : Scripting Options

---

long Pause(**double** Time)

Pause Script execution

Time : Pause Time (ms)

Example :

```
LabSpec.MoveMotor "Grating", 2400, "", 0
LabSpec.Pause 5000 ' 5 seconds pause
```

Exec

Description : Execute LabSpec Command
Keywords : show hide remove unload
Type : AutoVBSAct
Category : Spectral Display

---

long Exec(**long** ID, **long** Command, **VARIANT*** pParam)

Execute a LabSpec command

ID : Data ID

Command : Command Code to execute.
0 : SHOW_DATA Show the data
1 : HIDE_DATA Hide the data
2 : REMOVE_DATA Remove the data from memory
3 : CLONE_DATA Clone the current data
4 : SHOW_STYLE Change the display Style
5 : SHOW_MODE Change the display Mode
6 : SHOW_AXIS Display Spectrum/Video Axis or not.
7 : HIDE_ALL Hide all spectra in the active window
8 : GET_COLOR Get data color from active area (ID=0 based index of data in area)
9 : SET_COLOR Set data color in active area (ID=0 based index of data in area)
10 : STOP_PROCESSES Stop all active Video and acquisition processes
11 : CLONE_TABLE Clone the data tables (Acq, History, Custom..) from ID to pParam
12 : START_EXE Start an executable

pParam : Parameter for the SHOW_* commands :
SHOW_DATA parameters :
1 : SHOW_ACTIVATE show and activate the data
SHOW_STYLE parameters :
0 : SHOW_SINGLE single data per view
1 : SHOW_OVERLAY multiple spectra in the same view
2 : SHOW_TILE one view per spectrum
3 : SHOW_1D 1D display
4 : SHOW_2D 2D display
5 : SHOW_2D 3D display
6 : SHOW_SMOOTH Enable smoothing
SHOW_MODE parameters (you can add them for multiple options):
10000 : SHOW_SCALE_NORMA_X normalize X scale
20000 : SHOW_SCALE_NORMA_Y normalize Y scale
1000 : SHOW_SCALE_FIX_X fixed X scale
2000 : SHOW_SCALE_FIX_Y fixed Y scale
100 : SHOW_SCALE_AUTO_X auto X scale
200 : SHOW_SCALE_AUTO_Y auto Y scale
10 : SHOW_SCALE_SEP_X separate X scale
20 : SHOW_SCALE_SEP_Y separate Y scale
1 : SHOW_SCALE_LOG_X log X scale
2 : SHOW_SCALE_LOG_Y log Y scale

SHOW_AXIS parameters :
0 : AXIS_OFF Do not show Axis
1 : AXIS_ON Show Axis
START_EXE Executable absolute path and arguments. If executable path contains spaces, path must be between quotes.

Return Values :

>0 : Clone ID
0 : Succeeded
-1 : Failed

Constants List :

```
Const SHOW_DATA = 0
Const HIDE_DATA = 1
Const REMOVE_DATA = 2
Const CLONE_DATA = 3
Const SHOW_STYLE = 4
Const SHOW_MODE = 5
Const SHOW_AXIS = 6
Const HIDE_ALL = 7
Const GET_COLOR = 8
Const SET_COLOR = 9
Const STOP_PROCESSES = 10
Const CLONE_TABLE = 11
Const START_EXE = 12
Const SHOW_ACTIVATE = 1
Const SHOW_SINGLE = 0
Const SHOW_OVERLAY = 1
Const SHOW_TILE = 2
Const SHOW_1D = 3
Const SHOW_2D = 4
Const SHOW_3D = 5
Const SHOW_SMOOTH = 6
Const SHOW_SCALE_NORMA_X = 10000
Const SHOW_SCALE_NORMA_Y = 20000
Const SHOW_SCALE_FIX_X = 1000
Const SHOW_SCALE_FIX_Y = 2000
Const SHOW_SCALE_AUTO_X = 100
Const SHOW_SCALE_AUTO_Y = 200
Const SHOW_SCALE_SEP_X = 10
Const SHOW_SCALE_SEP_Y = 20
Const SHOW_SCALE_LOG_X = 1
Const SHOW_SCALE_LOG_Y = 2
Const AXIS_OFF = 0
Const AXIS_ON = 1
```

Example

This Example Loads a spectrum from a file and show it.

```
Dim SpectrumID
OpenSpectrum()
ShowSpectrum()
Private Sub OpenSpectrum()
  Dim FileName
  FileName ="C:Test.tsf"
  SpectrumID = LabSpec.Load(FileName)
End Sub
Private Sub ShowSpectrum()
   Dim Param
   LabSpec.Exec SpectrumID , SHOW_DATA, Param
End Sub
```

Start Notepad++ and open a file

```
Path="""C:Program Files (x86)Notepad++Notepad++.exe"" d:myFile.txt"
LabSpec.Exec 0,START_EXE,Path
```

SetScale

Description : Set Window Scale
Keywords : X Y intensity frequency scale
Type : AutoVBSAct
Category : Spectral Display

---

long SetScale(**double** FromX, **double** ToX, **double** FromY, **double** ToY)

Set Active window X and Y scale

FromX : X From limit (view frequency Unit)

ToX : X To limit (view frequency Unit)

FromY : Y From limit (view intensity Unit)

ToY : Y To limit (view intensity Unit)

Example :

```
LabSpec.SetScale 250, 275, 0, 100 ' Set X scale from 250 to 275, and Y scale from 0
to 100
```

SetScaleEx

Description : Set Window Scale
Keywords : X Y intensity frequency scale
Type : AutoVBSLS6
Category : Spectral Display

---

long SetScaleEx(**long** ID, **double** FromX, **double** ToX, **double** FromY, **double** ToY)

Set Active window X and Y scale. LS6 Only.

ID : Data ID

FromX : X From limit (view frequency Unit)

ToX : X To limit (view frequency Unit)

FromY : Y From limit (view intensity Unit)

ToY : Y To limit (view intensity Unit)

Example :

```
LabSpec.SetScale SpectrumID, 250, 275, 0, 100 ' Set X scale from 250 to 275, and Y scale from 0 to 100
```

GetMappingParams

Description : get labspec mapping parameters
Keywords : properties
Type : AutoActiveX
Category : Spectral Mapping

---

long GetMappingParams(**LPCTSTR** Axis, **VARIANT FAR***From, **VARIANT FAR*** To, **VARIANT FAR*** Step, **VARIANT FAR*** Mode, **VARIANT FAR*** Use)

Get Mapping parameters for specified axis

Axis : Motor Name (e.g. "X", "Y", ...)

From : Start position

To : Stop position

Step : Step size (if Mode=INCREMENT_STEP) or Number of point (if Mode=INCREMENT_SIZE)

Mode : INCREMENT_SIZE=0 : Get Number of points
INCREMENT_STEP=1 : Get step size

Use : DISABLE_AXIS=0 : Disabled axis
ENABLE_AXIS=1 : Enabled axis

Constants List :

```
Const INCREMENT_SIZE = 0
Const INCREMENT_STEP = 1
Const DISABLE_AXIS = 0
Const ENABLE_AXIS = 1
```

Map

Description : Create a Multi Dimension Profile
Keywords : mapping 2D
Type : AutoVBSAct
Category : Spectral Mapping

long Map(**long** Mode, **VARIANT FAR*** DataID, **VARIANT FAR*** MapID, **long** SpectrumID, **const VARIANT FAR&** Values, **const VARIANT FAR&** Labels, **const VARIANT FAR&** Units, **const VARIANT FAR&** Display, **float** From, **float** To)

Create a Map from specified spectra, colored by a averaged intensity.
For faster 2D maps, with known size and axis values, please use MapEx() function

Mode : Map Creation Mode
0 - CREATE_MAP : Create a Map
1 - ADD_TO_MAP : Add a spectrum to the Map

DataID : Full Data ID. This ID is defined if CREATE_MAP, and has to be set if ADD_TO_MAP

MapID : Final Map ID. This ID can be modified by the Map() function. Do not store it.

SpectrumID : Spectrum to add to the Map

Values : Array of values for the current Spectrum, for each axis

Labels : Array of Labels for each axis

Units : Array of Units for each axis

Display : Array of axes indexes. Set which axes will be displayed in the map.
Display(n) = X Axis on map (with n=number of dimensions to display)
Display(n-1) = Y Axis on map
Display(n-2) = Extra axes ...

From : Frequency limit for average intensity.

To : Frequency limit for average intensity.

Constant List :

```
Const CREATE_MAP = 0
Const ADD_TO_MAP = 1
```

Example :

```
Dim SpectrumID
Dim Param
Dim Labels(2)
Dim Units(2)
Dim Values(2)
Dim Display(1)
Dim DataID
Dim MapID
Dim ret
Dim TestFrom
Dim TestTo
Labels(0)="X"
Labels(1)="Y"
Labels(2)="Temperature"
Units(0)="mm"
Units(1)="mm"
Units(1)="degrees"
Display(0)=1 ' Set axis number 1 for the Y axis
Display(1)=0 ' Set axis number 0 for the X axis
TestFrom=112.3
TestTo=114.5
' Start First Acquisition
SpectrumID=0
LabSpec.Acq 0,1,1,0,0
do
   SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
Loop Until SpectrumID>0
Values(0)=10.0 ' Define X, Y and Temperature Values for the current spectrum
Values(1)=15.0
Values(2)=32.6
' Create a map and store the current spectrum ans parameters
ret=LabSpec.Map(CREATE_MAP,DataID,MapID,SpectrumID,Values,Labels, Units, Display, TestFrom,
TestTo)
' Start Acquisition 2
SpectrumID=0
LabSpec.Acq 0,1,1,0,0
do
   SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
Loop Until SpectrumID>0
Values(0)=12.0
Values(1)=15.0
Values(2)=35.2
' Add a spectrum to the previously generated map and dataID
ret=LabSpec.Map(ADD_TO_MAP,DataID,MapID,SpectrumID,Values,Labels, Units, Display, TestFrom,
TestTo)
' Start Acquisition 3
SpectrumID=0
LabSpec.Acq 0,1,1,0,0
```

```
do
   SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
Loop Until SpectrumID>0
Values(0)=26.0
Values(1)=64.0
Values(2)=62.2
ret=LabSpec.Map(ADD_TO_MAP,DataID,MapID,SpectrumID,Values,Labels, Units, Display, TestFrom,
TestTo)
' Start Acquisition 4
SpectrumID=0
LabSpec.Acq 0,1,1,0,0
do
   SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
Loop Until SpectrumID>0
Values(0)=96.0
Values(1)=23.0
Values(2)=14.2
ret=LabSpec.Map(ADD_TO_MAP,DataID,MapID,SpectrumID,Values,Labels, Units, Display, TestFrom,
TestTo)
' Show the Map
LabSpec.Exec MapID , SHOW_DATA, Param ' Show
```

MapEx

Description : Extended mapping
Keywords : Fast maps
Type : AutoActiveX
Category : Spectral Mapping

---

long MapEx(long Mode, **long** MapID, **long** SpectrumID, **long** SpectrumIndex, **const VARIANT FAR&** AxisX, **const VARIANT FAR&** AxisY, **const VARIANT FAR&** Labels, **const VARIANT FAR&** Units)

Create a Map from specified spectra, colored by a averaged intensity.
Use this function if you know in advance your 2D map size and axis values.
Otherwise, please use the Map() function.

Mode : Map Creation Mode
0 - CREATE_MAP : Create a Map
1 - ADD_TO_MAP : Add a spectrum to the Map
2 - EXTRACT_FROM_MAP : Extract a spectrum from a map (return Value=SpectrumID)

DataID : Full Data ID. This ID is defined if CREATE_MAP, and has to be set if ADD_TO_MAP

MapID : Set Current Map ID (ignored with CREATE_MAP)

SpectrumID : Spectrum to add to the Map (ignored with EXTRACT_FROM_MAP)

SpectrumIndex : Index of spectrum to add (Y*SizeX + X)

AxisX : Array of values for X Axis (ignored with ADD_TO_MAP,EXTRACT_FROM_MAP)

AxisY : Array of values for Y Axis (ignored with ADD_TO_MAP,EXTRACT_FROM_MAP)

Labels : Array of Labels for each axis (ignored with ADD_TO_MAP,EXTRACT_FROM_MAP)

Units : Array of Units for each axis (ignored with ADD_TO_MAP,EXTRACT_FROM_MAP)

Constant List :

```
Const CREATE_MAP = 0
Const ADD_TO_MAP = 1
Const EXTRACT_FROM_MAP  = 1
```

Example :

```
Dim SpectrumID
Dim Param
Dim Labels(1)
Dim Units(1)
Dim AxisX(25)
Dim AxisY(25)
Dim DataID
Dim MapID
For i=0 to UBound(AxisX)
    AxisX(i)=CDbl(i*2)
Next
For i=0 to UBound(AxisY)
    AxisY(i)=CDbl(i+10)
Next
Units(0)="mm"
Units(1)="um"
Labels(0)="X"
Labels(1)="Y"
LabSpec.Acq 0,0.01,1,0,0
SpectrumID=0
Do
   SpectrumID=labSpec.GetAcqID ()
Loop until SpectrumID>0
MapID=LabSpec.MapEx(0,0,SpectrumID,0,AxisX,AxisY,Labels,Units)
LabSpec.Exec MapID,0,Param
For i=1 to (UBound(AxisX)+1)*(UBound(AxisY)+1)
    LabSpec.Acq 0,0.01,1,0,0
    SpectrumID=0
    Do
       SpectrumID=labSpec.GetAcqID ()
    Loop until SpectrumID>0
    MapID=LabSpec.MapEx(1,MapID,SpectrumID,i,AxisX,AxisY,Labels,Units)
Next
```

Profile

Description : Create Spectral Image from spectra
Keywords : map mapping
Type : AutoVBSAct
Category : Spectral Mapping

long Profile(**long** Mode, **long** ProfileID, **long** SpectrumID, **double** Value, **LPCTSTR** Unit, **LPCTSTR** Label)

Create a profile from single spectra

Mode : Profile Mode :
0 : CREATE_PROFILE Create a profile with one spectrum (ProfileID is ignored)
1 : ADD_TO_PROFILE Add a Spectrum to the profile
2 : CHANGE_PROFILE_TYPE Profile type can be changed to time value
3 : EXTRACT_FROM_PROFILE Extract spectrum from Profile (return new SpectrumID)
4 : GET_PROFILE_SIZE Get Profile Size (return Profile Size)
5 : ADD_TO_PROFILE_INDEX Update a spectrum to the profile (Value=Spectrum Index)

ProfileID : only for ADD_TO_PROFILE : ID of previously created profile

SpectrumID : ID of spectrum to add to profile

Value : Extra dimension value value for the current spectrum
if EXTRACT_FROM_PROFILE : Index of the spectrum to extract

Unit : Profile extra dimension Unit (if "" with ADD_TO_PROFILE, keep the previously set Unit)
if CHANGE_PROFILE_TYPE, use "time" to set extra axis to time values.
when extra value type is time, use "AbsHour" as Profile Unit to get a 00:00:00 representation
extra value for time should be : hours from 00:00:00 (i.e. 08:30:00 will be 8.5)

Label : Profile extra dimension Label (if "" with ADD_TO_PROFILE, keep the previously set Label)

Return Values :

ProfileID if succeeded, -1 if failed
if EXTRACT_FROM_PROFILE : New Spectrum ID
if GET_PROFILE_SIZE : Profile Size

Constant List

```
Const CREATE_PROFILE = 0
Const ADD_TO_PROFILE = 1
Const CHANGE_PROFILE_TYPE = 2
Const EXTRACT_FROM_PROFILE = 3
Const GET_PROFILE_SIZE = 4
Const ADD_TO_PROFILE_INDEX = 5
```

Example

```
   Dim SpectrumID
   Dim Param
   LabSpec.Acq Mode,IntegrationTime,AccumulationNum,AcqFrom,AcqTo
   do
      SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
   Loop Until SpectrumID>0
   ProfileID=LabSpec.Profile(CREATE_PROFILE,0,SpectrumID, CurrentTemp, "Celsius deg", "Temperature")
' Create a profile with the first spectrum

   LabSpec.Exec ProfileID, SHOW_DATA, Param ' Show Profile
   ' Aquire a second spectrum and add it to the profile
   SpectrumID=0
   LabSpec.Acq Mode,IntegrationTime,AccumulationNum,AcqFrom,AcqTo
   do
      SpectrumID=LabSpec.GetAcqID() ' Wait until Spectrum is ready (acquisition is done)
   Loop Until SpectrumID>0

   LabSpec.Profile ADD_TO_PROFILE,ProfileID,SpectrumID, CurrentTemp,"", ""

   LabSpec.Exec ProfileID, SHOW_DATA, Param ' Show modified Profile
```

SetMappingParams

Description : set mapping parameters
Keywords : properties
Type : AutoVBSAct
Category : Spectral Mapping

---

long SetMappingParams(**LPCTSTR** Axis, **double** From, **double** To, **double** Step, **long** Mode, **long** Use)

Set Mapping parameters for specified axis

Axis : Motor Name (e.g. "X", "Y", ...)

From : Start position

To : Stop position

Step : Step size (if Mode=INCREMENT_STEP) or Number of point (if Mode=INCREMENT_SIZE)

Mode : INCREMENT_SIZE=0 : Set Number of points
INCREMENT_STEP=1 : Set step size
USE_ONLY=2 : Enable/Disable the axis only (From, To and Step are ignored)
DISABLE_ALL_MOTORS=3 : Disable all axis (Axis, From, To and Step are ignored)

Use : DISABLE_AXIS=0 : Disable the axis
ENABLE_AXIS=1 : Enable the axis

Constants List :

```
Const INCREMENT_SIZE = 0
Const INCREMENT_STEP = 1
Const USE_ONLY = 2
Const DISABLE_ALL_MOTORS = 3
Const DISABLE_AXIS = 0
Const ENABLE_AXIS = 1
```

Video

Description : Display video image
Keywords : camera extended
Type : AutoVBSAct
Category : Video Control

---

long Video(**long** Mode)

Mode : Start/Stop video
0 : START_VIDEO Start Video
1 : STOP_VIDEO Stop Video
2 : GET_VIDEO_ID Get Video ID (ActiveX mode : You need to call that function during the entire video process)
3 : START_EXTENDED_VIDEO Start an Extended Video (see SetExtendedVideo() to define the parameters). An Extended Video Image is not a live image.
4 : GET_ACTIVE_CAMERA Get the active video CameraID
10+CameraID : SET_ACTIVE_CAMERA Set the active video camera ID

Return Values :
>0 : VideoID
0 : OK
-1 : Error

Constant List :

```
Const START_VIDEO = 0
Const STOP_VIDEO = 1
Const GET_VIDEO_ID = 2
Const START_EXTENDED_VIDEO = 3
Const GET_ACTIVE_CAMERA = 4
Const SET_ACTIVE_CAMERA = 10
```

Example :

```
Dim Ret
Dim VideoID
Dim StopVideo
Ret = LabSpec.Video(START_VIDEO)
' Only for ActiveX use
Do
   IDVideo = LabSpec.Video(GET_VIDEO_ID)
   DoEvents
Loop until StopVideo=1 ' WARNING : You have to add a button to stop the video
(StopVideo=1)
' End Only for ActiveX use
Ret = LabSpec.Video(STOP_VIDEO)
```