Manipulating Collections, Folders and Files With Indusoft Web Studio Built-in Functions

Implementation Specifications or Requirements

Category	Item	
Software	IWS Version:	6.0 and later
	Service Pack:	N/A
	Windows Version:	WinXP/2000/NT and Windows CE
	Web Thin Client:	Yes
Equipment	Panel Manufacturer:	N/A
	Panel Model:	N/A
	Other Hardware:	N/A
	Comm. Driver:	All
	Controller (e.g.: PLC)	All
	Application Language:	N/A
Software Demo Application	N/A	

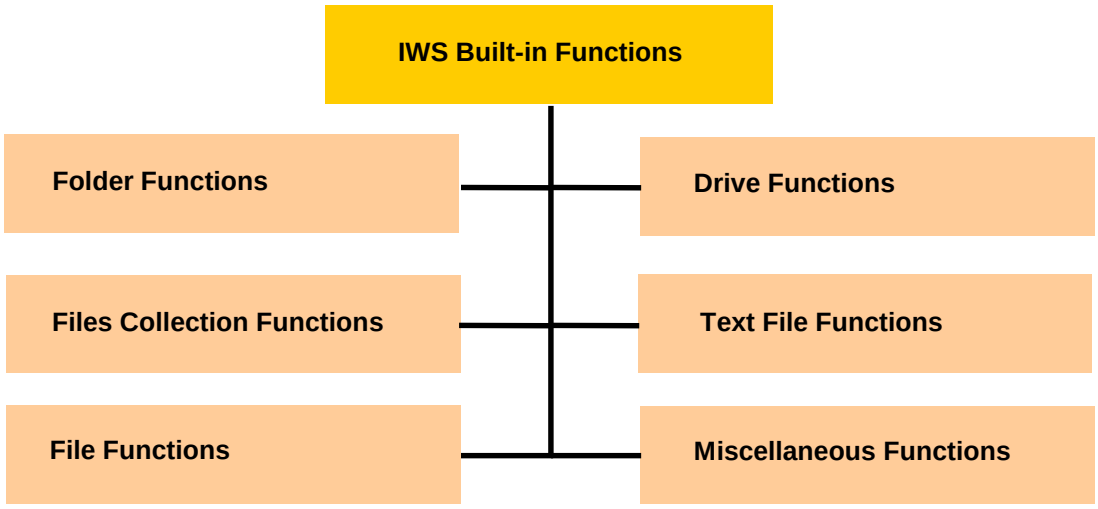
.

Summary

File manipulation is often an important function in an HMI/SCADA application. Loading or storing a Batch or Recipe files, accessing historical data, archiving or purging older files are among the examples where file manipulation is required. As discussed in this Application Note, InduSoft Web Studio (IWS) provides several built-in functions to access Collections (groups of folders or files), Folders (directories) and individual Files. Application Note AN-00-0006 will discuss the use of VBScript's FileSystemObject object, included in VBScript's runtime library, to access Collections, Folders and Files. Either approach can be used, and the selection is often merely the developer's preference.

IWS built-in functions can be called from a Math Worksheet script or from a VBScript code segment. Examples of both will be shown.

.



IWS Built-in Functions for Manipulation of Collections, Folders and Files

IWS Built-in Function Nomenclature

The following notes apply to the IWS built-in function descriptions in this Application Note:

A. Nomenclature/Syntax of Tags
- num[Name] IWS integer tag, expression (value), or VBScript variable
- "num[Name]" IWS integer tag name (cannot be expression or VBScript variable)
- str[Name] IWS string tag, expression (value), or VBScript variable
- "str[Name]" IWS string tag name (cannot be expression or VBScript variable)
- tag[Name] IWS tag name
- optbool[Name] Optional IWS boolean tag, expression (value) or VBScript variable
- optnum[Name] Optional IWS integer tag, expression (value) or VBScript variable
- optstr[Name] Optional IWS string tag, expression (value) or VBScript variable
- optTag[Name] Optional IWS tag name

B. When optional parameters are included in a Command syntax, they can be omitted as long as no additional optional parameters to the right in the Command syntax are specified. If additional optional parameters to the right in the Command syntax are specified, the optional parameter(s) not specified must be a "" (two double quotes). E.g.

GetLine (*strFile, strSeqChar, "TagStore", optNumCase, "optOverFlowTag"*)

Valid

GetLine ("C:\IWS\Readme.txt", "Hello", "StrMyLine")
GetLine ("C:\IWS\Readme.txt", "Hello", "StrMyLine", 1)
GetLine ("C:\IWS\Readme.txt", "Hello", "StrMyLine", "", "NumOverFlow")

Invalid

GetLine ("C:\IWS\Readme.txt", "Hello", "StrMyLine", , "NumOverFlow")
GetLine ("C:\IWS\Readme.txt", "Hello", "StrMyLine", "", "NumOverFlow")

C. If the Command syntax refers to a Tag[Name] or OptTag[Name] parameter, usually enclosed in double quotations, this Tag must be an IWS tag. It cannot be a VBScript variable or an Expression.

D. VBScript variables can only be used in a VBScript code segment, located in a Command Object, Background Script, Graphic Script, Screen Script, or a Global Procedure. VBScript variables are not accessible from an IWS Math Worksheet or other IWS native objects. InduSoft places limitations on the scope (accessibility) of VBScript variables from one code segment to another. Please refer to the InduSoft Web Studio Users Manual for additional details.

E. When used in a VBScript code segment, the **$** symbol is used as a prefix to signify an IWS built-in (native) function. Likewise, the **$** symbol is used as a prefix to signify an IWS tag. A full description of these built-in functions is contained in Appendix A of the InduSoft Web Studio Users Manual,

IWS Collections, Folders and File Built-in Function Summary

Function	Function Group	Execution	Windows XP/2K/NT	Windows CE	Web Thin Client
DeleteOlderFiles()	File	Synchronous	✓	✓	✓
DirCreate()	File	Synchronous	✓	✓	×
DirDelete()	File	Synchronous	✓	✓	×
DirLength()	File	Synchronous	✓	✓	×
DirRename()	File	Synchronous	✓	✓	×
FileCopy()	File	Synchronous	✓	✓	✓
FileDelete()	File	Synchronous	✓	✓	✓
FileLength()	File	Synchronous	✓	✓	✓
FileRename()	File	Synchronous	✓	✓	✓
FileWrite()	File	Synchronous	✓	✓	✓
FindFile()	File	Synchronous	✓	✓	✓
FindPath()	File	Synchronous	✓	✓	✓
GetAppPath()	System Info	Synchronous	✓	✓	×
GetFileAttributes()	File	Synchronous	✓	✓	✓
GetFileTime()	File	Synchronous	✓	✓	✓
GetLine()	File	Synchronous	✓	✓	✓
GetPrivateProfileString()	System Info	Synchronous	✓	✓	✓
GetProductPath()	System Info	Synchronous	✓	✓	✓
Hst2Txt()	File	Asynchronous	✓	×	×
Hst2TxtIsRunning()	File	Synchronous	✓	×	×
InfoAppAlrDir()	System Info	Synchronous	✓	✓	✓
InfoAppDir()*	System Info	Synchronous	✓	✓	×
InfoAppHstDir()	System Info	Synchronous	✓	✓	✓
InfoDiskFree()	System Info	Synchronous	✓	×	✓
Print()	File	Asynchronous	✓	✓	×
PrintSetup()	Graphic	Asynchronous	✓	✓	✓
RdFileN()	File	Synchronous	✓	✓	✓
SaveAlarmFile()	System Info	Synchronous	✓	✓	×
SetAppAlarmPath()	System Info	Synchronous	✓	✓	×
SetAppHstPath()	System Info	Synchronous	✓	✓	✓
SetAppPath()	Module Activity	Synchronous	✓	×	×
SetTranslationFile()	Translation	Synchronous	✓	✓	✓
WebGetFile()	File	Synchronous	✓	✓	✓

Notes:

* Use the function GetAppPath() instead

Synchronous execution means that the function must complete before the next IWS statement will be processed.

Drive Functions

IWS provides very limited Drive functions. The only function provided at this time is to check the free space (bytes) available on a specified disk drive.

Table A: Summary of IWS Built-in Drive Functions

Function	Description
InfoDiskFree	Returns the free space on a specified disk

InfoDiskFree (*strDisk*)

Function
This function returns the free disk space on a specified disk

Parameters
strDisk is an IWS string tag, expression or VBScript variable containing the name of the disk to be checked
Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
Returns the free space on the disk in bytes

Example IWS Usage

Tag	Expression
StatFlag	InfoDiskFree(“C”)

Example VBScript Usage

```
Dim MyDisk, FreeSpace
MyDisk = “C”
FreeSpace = $InfoDiskFree(MyDisk)
```

Folder Functions

IWS uses the term Directory (or DIR) for a folder, although the meaning is interchangeable. The Directory functions

Note that all of the Folder functions work in either a Windows XP/2000/NT or Windows CE runtime environment. However, these functions were not intended to run on Web Thin Client (unless otherwise noted). When a Web Thin Client executes a function that calls a Folder function, the Folder function executes on the Server station.

Folder functions can be used in conjunction with various System Information functions. One example is to create a folder which has the current date built-in to the folder name. This folder can then used to set the *History* path for the current application (see SetAppHSTPath function).

Table B: Summary of IWS Built-in Folder Functions

Function	Description
DirCreate	Creates a directory at the specified file path
DirDelete	Deletes a specified directory
DirLength	Returns the size (bytes) of the specified directory
DirRename	Renames the specified directory
FindPath	Verifies whether a specified directory exists or not

DirCreate (*strDirectory*, *optBoolFullPath*)

Function
This function creates a directory (folder) starting at the specified file path.

Parameters
strDirectory is an IWS string tag, expression or VBScript variable containing the name and file path of the directory.

optBoolFullPath is an optional flag. It can be an IWS boolean tag, expression or VBScript variable. When omitted or set to 0, the directory is created only if the previous directory exists. If a non-zero value, the full path will be created regardless if the previous directory exists.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
The Returned Value has one of 3 possible values:

-1	Invalid Parameters
0	Failure to create the directory (previous directory or drive does not exist)
1	Successful

Example IWS Usage

Tag	Expression
StatFlag	DirCreate (“C:\MyApp\Temp”,1)

Example VBScript Usage

```
Dim StrDir, RetVal
StrDir = “C:\MyApp\Temp”                ‘Specifies the Directory to be created
RetVal = $DirCreate (StrDir, 1)         ‘Call the Directory Create function
If RetVal =1 Then
    MsgBox “Directory Created”
End If
```

DirDelete (*strDirectory*, *optBoolEmptyPath*)

Function
This function is called to delete a specified directory.

Parameters
strDirectory is an IWS string tag, expression or VBScript variable containing the name and file path of the directory.

optBoolEmptyPath is an optional flag. It can be an IWS boolean tag, expression or VBScript variable. When omitted or set to 0, the directory is deleted only when empty. If a non-zero value, the directory is deleted along with any files contained within.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
The Returned Value has one of 4 possible values:

-2	Attempted to delete a non-empty directory when *OptBoolEmptyPath* = 0
-1	Invalid Parameters
0	Failure to delete the directory (directory or drive does not exist)
1	Successful

Example IWS Usage

Tag	Expression
StatFlag	DirDelete (“C:\MyApp\Temp”,1)

Example VBScript Usage

```
Dim StrDir, RetVal
StrDir = “C:\MyApp\Temp”                ‘Specifies the Directory to be deleted
RetVal = $DirDelete (StrDir, 1)         ‘Call the Directory Delete function
If RetVal =1 Then
    MsgBox “Directory Deleted”
End If
```

DirLength (*strDirectory*)

Function

This function returns the size (in bytes) of the Directory specified. This function may take several seconds to return a value, so the use of this function must be done with caution.

Parameters

strDirectory is an IWS string tag, expression or VBScript variable containing the name and file path of the directory.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

The Returned Value has one of 3 possible values:

-2	Directory does not exist
-1	Invalid Parameters
>=0	Size (in bytes) of the files and subfolders in the directory

Example IWS Usage

Tag	Expression
StatFlag	DirLength (“C:\MyApp\Temp”)

Example VBScript Usage

```
Dim StrDir, RetVal
StrDir = “C:\MyApp\Temp”                    ‘Specifies the Directory
RetVal = $DirLength (StrDir)                ‘Call the Directory Length function
If RetVal >-1 Then MsgBox “Directory Length = ” & RetVal & “ Bytes”
```

DirRename (*strPath, strDirFrom, strDirTo*)

Function
Rename a directory to a new name.

Parameters
strPath is an IWS string tag, expression or VBScript variable containing the name and file path to the directory. Note that the StrPath parameter is one level up from the directory to be renamed.

strDirFrom is an IWS string tag, expression or VBScript variable containing the original name of the directory to be renamed

strDirTo is an IWS string tag, expression or VBScript variable containing the new name of the directory to be renamed

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
The Returned Value has one of 3 possible values:

-1 Invalid Parameters
0 Failed to rename the directory (e.g. StrPath does not exist)
1 Successful

Example IWS Usage

Tag	Expression
StatFlag	DirRename (“C:\MyApp\”, “Temp”, “Old”)

This example renames C:\MyApp\Temp to C:\MyApp\Old

Example VBScript Usage

```
Dim StrPath, StrOld, StrNew, RetVal
StrPath = “C:\MyApp\Temp\”                    ‘Specify the Path
StrOld = “Temp”                               ‘Specify the old Folder name
StrNew = “New”                                ‘Specify the new Folder name
RetVal = $DirRename (StrPath, StrOld, StrNew) ‘Call the Directory Length function
If RetVal = 1 Then
    MsgBox “Successfully renamed”
End If
```

FindPath (*strDir*)

Function
This function verifies whether a directory exists or not.

Parameters
strDir is an IWS string tag, expression or VBScript variable containing the name and file path of the directory to be verified.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
The Returned Value has one of 2 possible values:

0	Path not found
1	Path found

Example IWS Usage

Tag	Expression
StatFlag	FindPath (“C:\MyApp\Temp”)

Example VBScript Usage

```
Dim StrDir, RetVal
StrDir = “C:\MyApp\Temp\”                 ‘Specify the Path
RetVal = $FindPath (StrDir)               ‘Call the FindPath function
If RetVal = 1 Then
    MsgBox “Path Found”
End If
```

Files Collection Functions

The following functions are used to manipulate collections of files.

Table C: Summary of IWS Built-in Files Collection Functions

Function	Description
DeleteOlderFiles	This function deletes files that are older than a specified date. A return value specifies the number of files that were deleted.
FindFile	Finds a file or set of files that match a path and file mask
RdFileN	Opens a dialog box containing a list of files in the specified directory that meet the mask criteria.

DeleteOlderFiles (*strDirectory, strMask, strDate)*

Function

This function deletes files that are older than a specified date. A return value specifies the number of files that were deleted.

Parameters

strDirectory is an IWS string tag, expression or VBScript variable containing the name and file path of the directory.

strMask is an IWS string tag, expression or VBScript variable that contains the mask of the files to be deleted

strDate is an IWS string tag, expression or VBScript variable containing the cut-off date. Files older than this date will be deleted.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

The Returned Value is the number of files deleted.

Example IWS Usage

Tag	Expression
MyPath	GetAppPath()+ “hst”
	DeleteOlderFiles (MyPath, “*.hst”, “06/30/06”)

This example retrieves the application folder of the current application and points to the history (\hst) subfolder. It then deletes any files in the subfolder older than the date specified.

Example VBScript Usage

```
Dim StrPath, StrMask, StrDate
Dim RtnVal
StrPath = $GetAppPath() & “hst”          ‘ Specify hst subfolder in current application directory
StrMask = “*.hst”                        ‘ Specify *.hst files to delete
StrDate = “06/30/06”                     ‘ Specify file date (older files will be deleted)
RtnVal = $DeleteOlderFiles(StrPath, StrMask, StrDate)
MsgBox RtnValue & “ Files Deleted”
```

FindFile (*strFile, "optstrTagFilesFound", optNumTimeOut*)

Function

This function searches for a file (or set of files).

Note: this file function operates in a synchronous manner with other operations; i.e. other operations will not progress until this operation is completed. Therefore, it is highly recommended to use the *OptNumTimeOut* parameter. Slow network transfer rates may also cause a problem. If a timeout occurs, the function returns a value of -1. This does not cancel the copying procedure, but will allow the other IWS operations to proceed and an internal process to finish the file copying will be created.

Parameters

strFile is an IWS string tag, expression or VBScript variable containing the path and file mask for which to search. Note that unless otherwise specified, the path searched will be the current application project folder.

"optstrTagFilesFound" is an optional IWS string tag array to receive the path and name of each file found. Array index 0 will be blank. The first file found will be in the array index 1, the next file found will be in array index 2, etc. This must be an IWS string tag array and cannot be an expression or VBScript Array.

optNumTimeOut is an optional IWS integer tag, expression or VBScript variable containing an integer value used to set the timeout period in milliseconds for the operation.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Note: This File Function operates in a synchronous manner with other operations; i.e. other operations will not progress until this operation is completed. Therefore, it is highly recommended to use the *optNumTimeOut* parameter. Slow network transfer rates may also cause a problem. If a timeout occurs, the function returns a value of -1. This does not cancel the copying procedure, but will allow the other IWS operations to proceed and an internal process to finish the file copying will be created.

Return Value

The Returned Value is the number of files deleted.

-1	Timeout occurred
0	No files found
N	Number of files found

Example IWS Usage

Tag	Expression
StatFlag	FindFile ("*.txt")
StatFlag	FindFile ("C:\IWS\MyApp*.txt", "FileArray",1000)
StatFlag	FindFile ("C:\IWS\MyApp\report.txt","FileArray", 500)

The 1st example will look in the current application project folder for files with a .txt file extension. The number of files found will be stored in the StatFlag tag.

The 2nd example looks for .txt files in a specified path. Any files found will be put into the FileArray IWS string tag array, starting with array index 1. A 1000 millisecond timeout is specified.

In the 3rd example, the specific file is searched for. If it exists, StatFlag will be set to a value of 1. The file name will be stored in the FileArray IWS string tag array (index 1). A 500 millisecond timeout value was specified.

Example VBScript Usage

```
Dim StatFlag, StrFile, TimeOut
StrFile = "C:\IWS\MyApp\*.txt"
TimeOut = 1000
StatFlag = $FindFile (StrFile, "$FileArray", TimeOut)
If StatFlag > 0 Then
  MsgBox "Files Found = " & StatFlag
End If
```

RdFileN (“*strSelTagFile”, strDirectory, strMask, numCgDir)*

Function

This is a very useful function that opens a dialog box containing a list of files in a specified directory that meet the mask criteria. The user can then select a file, and the selected file (and path) will be returned in an IWS string tag.

Parameters

“strSelTagFile” is an IWS string tag containing the selected file name (includes the file path). This must be an IWS string tag and cannot be an expression or VBScript Array.

strDirectory is an IWS string tag, expression or VBScript variable containing the name and file path of the directory to be searched.

strMask is an IWS string tag, expression or VBScript variable that contains the mask of the files to be searched

numCgDir is an IWS integer tag, expression or VBScript variable that controls whether to allow changing of directories. If the value is 0, a simple popup window will appear with a list of files in the directory that meet the Mask criteria. If the value is <>0 then a Windows file dialog box will appear that allows the changing of directories. With either window, the selected file (path and file name) will be stored in the SelTagFile tag.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

The Returned Value is the number of files deleted.

0	Successful
1	One of the parameters is not a string
2	The first parameter is not a valid tag name
3	The user cancelled the operation

Example IWS Usage

Tag	Expression
MyDir	GetAppPath()+ “hst”
	RdFileN (“MyFile”, MyDir, “*.hst”, 0)

This example opens the \hst subfolder in the folder where the current application is located. Then, it displays the current historical files (*.hst file extension) contained in this subfolder in a simple dialog box. The result of this function is placed in the IWS string tag MyFile.

Example VBScript Usage

```
Dim StrPath, StrMask, NumCgDir
Dim RtnVal
StrPath = $GetAppPath() & “hst”          ‘ Specify hst subfolder in current application directory
StrMask = “*.hst”                        ‘ Specify *.hst files to return/view
NumCgDir = 1                             ‘ Specify that the user can change directories
RtnVal = $RdFileN(“$MyFile”, strPath, strMask, numCgDir)
MsgBox “Return Value = “ & RtnVal & “  File Name = “ & $MyFile
```

File Functions

IWS has several built-in file functions that allow a developer to manipulate individual files at run-time. These File Functions work for Windows XP/2000/NT and Windows CE runtimes, as well as for Thin Clients, unless otherwise noted.

Table D: Summary of IWS Built-in File Functions

Function	Description
FileCopy	Copies a file to a new path/file
FileDelete	Deletes the specified file
FileLength	Returns the size of a file (bytes)
FileRename	Renames a specified file
GetFileAttributes	Reads the attributes of a specified file
GetFileTime	Reads the date/time the specified file was last modified

FileCopy (*strSource, strTarget, optNumTimeOut*)

Function

This function copies a file (or set of files) pointed to by the StrSource parameter to the path/file configured in the StrTarget parameter.

Note: This file function operates in a synchronous manner with other operations; i.e. other operations will not progress until this operation is completed. Therefore, it is highly recommended to use the *optNumTimeOut* parameter. Slow network transfer rates may also cause a problem. If a timeout occurs, the function returns a value of -1. This does not cancel the copying procedure, but will allow the other IWS operations to proceed and an internal process to finish the file copying will be created.

Parameters

strSource is an IWS string tag, expression or VBScript variable containing the path and file name (or mask) of the file(s) to be copied.

strTarget is an IWS string tag, expression or VBScript variable containing the path where the file(s) are to be copied.

optNumTimeOut is an optional IWS numeric tag, expression or VBScript variable containing an integer to set the timeout period in milliseconds for the operation.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

The Returned Value is the number of files deleted.

-1	Timeout occurred
0	Failed to copy file(s)
1	Successful

Example IWS Usage

Tag	Expression
StatFlag	FileCopy (“C:\IWS\MyApp\HST*.hst”, “C:\Temp\Hst\”,1000)
StatFlag	FileCopy (“C:\IWS\MyApp\report.txt”,”C:\Temp\TuesdayReport.txt”, 500)

Example VBScript Usage

```
Dim StatFlag
Dim StrSource, StrTarget
Dim TimeOut
StrSource = “C:\IWS\MyApp\HST\*.hst”
StrTarget = “C:\Temp\Hst\”
TimeOut = 1000
StatFlag = $FileCopy(StrSource, StrTarget, TimeOut)
If StatFlag = 1 Then
  MsgBox “Copy Successful”
End If
```

FileDelete (*strFile*)

This function deletes the specified file

Parameters

strFile is an IWS string tag, expression or VBScript variable containing the path and file name of the file to delete.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

The Returned Value is the number of files deleted.

0	Failed to delete file
>0	Size (in bytes) of the file deleted. This is an IWS type real

Example IWS Usage

Tag	Expression
DelSize	FileDelete (“C:\IWS\readme.txt”)

Example VBScript Usage

```
Dim StrFile
Dim DelSize
StrFile = “C:\IWS\readme.txt”
DelSize = $FileDelete (StrFile)
If DelSize > 0 Then
  MsgBox “Delete Successful.  “ & DelSize & “  Bytes deleted”
Else If
  MsgBox “Delete Error”
End If
```

FileLength (*strFile*)

Function
This function returns the size of a file in bytes

Parameters
strFile is an IWS string tag, expression or VBScript variable containing the path and file name of the file.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
The Returned Value is the size of the specified file in bytes.

Example IWS Usage

Tag	Expression
FileSize	FileLength (“C:\IWS\readme.txt”)

Example VBScript Usage

```
Dim StrFile, FileSize
StrFile = “C:\IWS\readme.txt”
FileSize = $FileLength (StrFile)
MsgBox “File Size = “ & FileSize & “ Bytes”
```

FileRename (*strOldFileName, strNewFileName*)

Function
This function renames a specified file

Parameters
strOldFileName is an IWS string tag, expression or VBScript variable containing the path and file name of the old file name.

strNewFileName is an IWS string tag, expression or VBScript variable containing the path and file name of the new file name.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
There are no Returned Values.

Example IWS Usage

Tag	Expression
	FileRename (“C:\IWS\readme.txt”, “C:\IWS\readme1.txt”)

Example VBScript Usage

```
Dim StrOldFileName, StrNewFileName
StrOldFileName = “C:\IWS\readme.txt”
StrNewFileName “C:\IWS\readme1.txt”
$FileRename (StrOldFileName, StrNewFileName)
```

GetFileAttributes (*strFile*)

Function
This function reads the attributes of a specified file.

Parameters
strFile is an IWS string tag, expression, or VBScript variable containing the path and file name of the file from which the attributes are read.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
The Returned Value is the number of files deleted.

- 1 Error
  1 Read-only
  2 Hidden
  4 System
  16 Directory
  32 Archive
  128 Normal
  256 Temporary

Example IWS Usage

Tag	Expression
StatFlag	GetFileAttributes (“C:\IWS\readme.txt”)

Example VBScript Usage

```
Dim StrFile
Dim StatFlag
StrFile = ”C:\IWS\readme.txt“
StatFlag = GetFileAttributes (StrFile)
```

GetFileTIme (*strFile, numFormat*)

Function
This function reads the time and/or date the specified file was last modified.

Parameters
strFile is an IWS string tag, expression, or VBScript variable containing the path and file name of the file from which the attributes are read.

numFormat is an IWS integer tag, expression, or VBScript variable which specifies the format of the returned data. Valid values are

0 Returns the date and time the file was last modified (default)
1 Returns the date the file was last modified
2 Returns the time the file was last modified

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
Returns the date and/or time the file was last modified.

Example IWS Usage

Tag	Expression
RtnVal	GetFileTime (“C:\IWS\readme.txt”, 0)

In this example, a return value might be 07/19/2006 15:37:08.

Example VBScript Usage

```
Dim StrFile
Dim RtnVal
StrFile = ”C:\IWS\readme.txt“
RtnVal = $GetFileTime (StrFile, 0)
MsgBox “File date & time = “ & RtnVal
```

WebGetFile (*strURL, strDestLocalFile*)

Function
This function retrieves a file from a Web Server using the HTTP protocol and stores it under the specified file name, creating a new local file.

Parameters
strURL is an IWS string tag or expression that contains a Uniform Resource Locator (URL) for the file to be received.

strDestLocalFile is a fully qualified name of the destination local file

Returned values
0	Success
-1	Invalid number of parameters
-2	Invalid URL
-3	Internal Error
-4	Error connecting to the Web Server
-5	Error creating the destination local file

Example IWS Usage
Tag	Expression
RtnVal	WebGetFile (“http://myfile.txt”, “C:\myfile.txt”)

Example VBScript Usage
```
Dim strURL
Dim destFile
Dim RtnVal
strURL = “www.studio.scada.com/file1.txt”
destFile = “C:\myFile.txt”
RtnVal = $WebGetFile(strUrl, destFile)
```

Text File Functions

The following functions are used to manipulate text files.

Table E: Summary of IWS Built-in Text File Functions (Reading/Writing/Printing)

Function	Description
FileWrite	Writes an ASCII string into a specified file.
GetLine	Searches a text file for a specific string, then returns the whole line
Print	Prints a Text file
PrintSetup	Sets up the default printer

FileWrite (*strFile, strWriteText, numOptAppend*)

Function

This function writes an ASCII sting into a specified file. If file does not exist, it will be created. However, the path must exist.

Parameters

strFile is an IWS string Tag, expression or VBScript variable containing the path and file name of the file to be written into.

strWriteText is an IWS string Tag, expression or VBScript variable containing the text to be written into the specified file.

numOptAppend is an optional IWS integer Tag, expression or VBScript variable. Values are:

Blank or 0	The FileWrite function will replace the contents of the existing file (or new file) with the text specified in the StrWriteText parameter. This line will now be the first line in the file.
Non-zero	The FileWrite function appends the text specified in the StrWriteText parameter to the end of the file. If the file does not exist, this will be the first line in the file.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

The Returned Value is the number of files deleted.

-2	Failed to open file. E.g. disk is write-protected, file name invalid, path does not exist
-1	Invalid parameter(s)
0	Successful

Example IWS Usage

Tag	Expression
StatFlag	FileWrite (“C:\IWS\readme.txt”, TagText)

Example VBScript Usage

```
Dim StrFile, StrText
Dim StatFlag
StrFile =“C:\IWS\readme.txt“
StrText = “This is line 1”
StatFlag = $FileWrite (StrFile, StrText)                    ‘This becomes line 1
StrText = “This is line 2”
StatFlag = $FileWrite (StrFile, StrText,1) ‘This line is appended, becomes line 2
```

GetLine (*strFile, strSeqChar, "strTagStore", optnumCase, "optnumOverFlowTag"*)

Function

This function searches an ASCII (text) file for a specified sequence of characters and stores the contents of the whole line contained in the ASCII file into an IWS string tag specified by the *TagStore* parameter. Note that if more than one line in the ASCII file has the specified sequence of characters, the first line found will be returned in the *TagStore* IWS string Tag.

Parameters

strFile is an IWS string tag, expression, or VBScript variable containing the path and file name of the file from which the attributes are read.

strSeqChar is an IWS string tag, expression, or VBScript variable containing the sequence of characters for which to search

"strTagStore" is the name of an IWS string tag receiving the whole line of characters from the ASCII (text) file from which the sequence was found. This must be an IWS string tag and cannot be an expression or VBScript variable.

optnumCase is an optional IWS integer tag, expression, or VBScript variable that specifies whether the string search is to be case sensitive.

0 Search is not case-sensitive
1 Search is case-sensitive

"optnumOverFlowTag" is an optional name of an IWS integer tag that receives the result of an overflow verification. This must be an IWS numeric tag and cannot be an expression or VBScript variable.

0 Ok
1 Overflow

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

Returns either a status condition or the number of lines matching the search string

-7 Invalid Number of parameters
-6 Invalid *OptNumOverFlowTag* parameter
-5 Invalid *OptNumCase* parameter
-4 Invalid *TagStore* parameter
-3 Invalid *StrSeqChar* parameter
-2 Invalid *StrFile* parameter
-1 ASCII file not found
0 String specified in *StrSeqChar* was not found in the target ASCII file
N Number of lines in which the string sequence was found in the target ASCII file

Example IWS Usage

Tag	Expression
RtnVal	GetLine ("C:\IWS\readme.txt", "Hello", "MyLine", 0, "OverFlowTag")

In this example, a return value indicates how many times the word Hello is in the ASCII file. The first occurrence of a line with the string Hello is stored in the IWS string tag MyLine. The search is not case sensitive

Example VBScript Usage

```
Dim StrFile, StrSearch, RtnVal
StrFile = "C:\IWS\readme.txt"
StrSearch = "Hello"
RtnVal = $GetLine (StrFile, StrSearch, "$MyLine", 0, "$OverFlowTag")
MsgBox "Number of times string found = " & RtnVal
```

Print (*strFile, optNumOrientation*)

Function

This function prints a text file. No graphics can be printed. Unfortunately, this function does not allow selection over the printer port and printer type. This function will print to the default printer only. See the PrintSetup Function for Printer selection.

Parameters

strFile is an IWS string tag, expression, or VBScript variable containing the file name and file path of the file to be printed. .

optNumOrientation is an optional IWS integer tag, expression or VBScript variable that sets the paper orientation. This parameter is not supported on Windows CE systems. Valid values are:

0 Portrait orientation (default)
1 Landscape orientation

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

There is no return value.

Example IWS Usage

Tag	Expression
	Print (“C:\readme.txt”)
	Print (MyFile, 1)

Example VBScript Usage

```
Dim MyFile
MyFile = “C:\readme.txt”
$print(MyFile, 0)
```

PrintSetup()

Function

This function opens the standard printer dialog box from the operating system. A default printer can be selected and configured for print operations.

Parameters

No parameters

Return Value

There is no return value.

Example IWS Usage

Tag	Expression
	PrintSetup()

Example VBScript Usage

$PrintSetup

Miscellaneous Functions

The following functions are miscellaneous functions provided to manage IWS applications and tasks. Many of these functions are special purpose in nature.

Table F: Summary of Miscellaneous IWS Built-in Functions

Function	Description
GetAppPath	Returns the directory of the current application
GetPrivateProfileString	Reads a specified parameter from an .ini file using the standard .ini format
GetProductPath	Returns the path where the IWS Program Files are located
Hst2Txt	Exports an IWS proprietary format historical trend file into a *.txt file
Hst2TxtIsRunning	Returns the status of the Hst2Txt function
InfoAppAlrDir	Returns the Alarm directory for the current application
InfoAppDir	Returns the path to the current application
InfoAppHstDir	Returns the History directory for the current application
SaveAlarmFile	Used to enable/disable the saving feature for Alarm history & specify Alarm history file path
SetAppAlarmPath	Sets Alarm path for the current application
SetAppPath	Sets a new path for an application
SetAppHstPath	Used to set the History path for the current application
SetTranslationFile	Sets the file used for runtime translation

GetAppPath()

Function

This function returns the directory of the current application.

Note: InfoAppDir is a similar function. However, use of the GetAppPath() function is preferred as it always returns a "\" at the end of the path, regardless of the operating system used.

Parameters

None

Return Value

Returns the directory of the current application as a string value. Can be stored in an IWS string tag or a VBScript variable. A "\" is included at the end of the path (directory).

Example IWS Usage

Tag	Expression
StrPath	GetAppPath()

Example VBScript Usage

```
Dim MyAppDir
MyAppDir = $GetAppPath()
MsgBox "My application directory is located at " & MyAppDIr
```

GetPrivateProfileString (strSection, strName, strDefault, strFilename)

Function
This function reads a specified parameter from an .ini file using the standard .ini format.

Parameters
strSection is an IWS string tag, expression, or VBScript variable containing the section name to be read.

strName is an IWS string tag, expression, or VBScript variable containing the parameter name to be read.

strDefault is an IWS string tag, expression, or VBScript variable containing the default setting for this parameter. If the parameter is not found in the .ini file, the function will return this default setting.

strFileName is an IWS string tag, expression, or VBScript variable containing the path and name of the .ini file to be read.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value
Returns the value of the specified parameter.

Example IWS Usage

Tag	Expression
RtnValue	GetPrivateProfileString(“boot loader”, “timeout”, “50”, “C:\boot.ini”

Example VBScript Usage

```
Dim rtnValue
myINIsection = “boot loader”
myINI = “C:\boot.ini”
strTimeDefault = “50”                                    ‘Note this is a string, not a numeric value
rtnValue = $GetPrivateProfileString(myINIsection, “timeout”, strTimeDefault, myINI)
```

GetProductPath()

Function
This function returns the path where the IWS program files are located

Parameters
None

Return Value
Returns the path to the IWS directory as a string value. Can be stored in an IWS string tag or a VBScript variable.

Example IWS Usage

Tag	Expression
StrPath	GetProductPath ()

Example VBScript Usage

```
Dim MyPath
MsgBox $GetProductPath()
MyPath = $GetProductPath()                    'E.g. returns "C:\Program Files\IWS 6.1\
```

Hst2Txt (*strStartDate, strStartTime, numDuration, numGroupNumber, strTargetFile, optStrSeparator, optNumMilliseconds, optStrFormat)*

Function

This function is used to export information from an InduSoft proprietary binary format historical trend file(s) (*.hst) into a text (*.txt) file. The source of the historical trend file is the HST folder contained in the current project application folder. If a comma is used as the optional string separator, a .csv (comma separated variables) file will be created instead of a .txt file. Along with the target file (text or csv), another file with the same file name will be written. This is a header file that indicates the Group (Sheet number) and Tags which values are stored.

Parameters

strStartDate is an IWS string tag, expression, or VBScript variable containing the start date of the data.

strStartTime is an IWS string tag, expression, or VBScript variable containing the start time of the data.

numDuration is an IWS integer tag, expression, or VBScript variable containing the duration of the data in hours.

numGroupNumber is an IWS integer tag, expression, or VBScript variable containing the Trend Group Number (Sheet Number). E.g. for Sheet 1, the value of this parameter should be 1.

strTargetFile is an IWS string tag, expression, or VBScript variable containing the path and name of the file to be written (the .txt file).

optStrSeparator is an optional IWS string tag, expression, or VBScript variable containing the data separator character for the file. If omitted, a Tab character (\t) is used to separate the values in the file.

optNumMilliseconds is an optional IWS integer tag, expression, or VBScript variable used to signify if the text file is to show millisecond precision on the timestamp of each history sample

0	Text file will not show millisecond precision (default)
<0>	Text file will show millisecond precision

optStrFormat is an IWS optional string tag, expression, or VBScript variable which specifies the order of the Month (M), Day (D) and Year (Y) for the timestamp format exported to the text file. If omitted, the function uses the DMY format for the timestamp in the text file. Valid values are:

“DMY”	Day, Month, Year
“MDY”	Month, Day, Year
“YMD”	Year, Month, Day

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

-3	Invalid number of parameters
-2	DLL functions not found
-1	IndHst.dll not found
0	Successful
1	Error. Previous execution of Hst2Txt has not yet been completed

Example IWS Usage

Tag	Expression
RtnVal	Hst2Txt (“04/12/2006”, “07:00:00”, 8, 1, “C:\HistData.txt”, “,” , 0, “MDY”)

Example VBScript Usage

```
Dim StrFile, StrSearch, RtnVal
strDate = “04/12/2006”
strTime = “07:00:00”
numDur = 8
hstFile = “C:\histDate.txt”
rtnVal = $Hst2Txt (strDate, strTime, numDur, 1, hstFile, “,”, 0, “MDY”)
If rtnVal = 0 then MsgBox “Successful conversion“
```

Hst2TxtIsRunning()

Function
This function returns the status of the Hst2Txt function

Parameters
None

Return Value
-30	Cannot access dll function
-20	IndHst.dll was not found
-10	Cannot create Header file (.hdr)
- 9	Invalid number of tag in the header information (0 > nTags > 250)
- 8	Cannot read header information from HST file
- 7	Invalid file type
- 6	Cannot read file information from HST file
- 5	Cannot create/open ASCII file
- 4	Cannot open HST file
- 3	File not found. There are no history files in the configured time interval for the group specified
- 2	Reserved
- 1	Last conversion process was executed properly
0	Hst2Txt is still running

Example IWS Usage
Tag	Expression
RtnVal	Hst2TxtIsRunning()

Example VBScript Usage
```
Dim rtnVal
rtnVal = $Hst2TxtIsRunning()
If rtnVal = 0 then
    MsgBox “Successful conversion“
End If
```

InfoAppAlrDir()

Function
This function returns the Alarm directory for the current application.

Parameters
None

Return Value
Returns the Alarm directory for the current application as a string value. Can be stored in an IWS string tag or a VBScript variable.

Example IWS Usage

Tag	Expression
StrPath	InfoAppAlrDir()

Example VBScript Usage

```
Dim MyAlarmDir
MyAlarmDir = $InfoAppAlrDir()
MsgBox “My application alarm directory is located at “ & MyAlarmDIr
```

InfoAppDir()

Function

This function returns the path to the current application. This function is similar to the GetAppPath() function, with the difference being that the InfoAppDir() function does not return a "\" at the end of the path when used in a Windows XP/2K/NT runtime environment, while returning a "\" at the end of the path when used in a Windows CE environment. The GetAppPath() function is newer and is the preferred function for consistency purposes..

Parameters

None

Return Value

Returns the directory of the current application as a string value. Can be stored in an IWS string tag or a VBScript variable.

Example IWS Usage

Tag	Expression
StrPath	InfoAppDir()

Example VBScript Usage

```
Dim MyAppDir
MyAppDir = $InfoAppDir()
MsgBox "My application directory is located at " & MyAppDIr
```

InfoAppHstDir()

Function
This function returns the History directory for the current application.

Parameters
None

Return Value
Returns the History directory of the current application as a string value. Can be stored in an IWS string tag or a VBScript variable.

Example IWS Usage

Tag	Expression
StrPath	InfoAppHstDir()

Example VBScript Usage

```
Dim MyHistDir
MyHistDir = $InfoAppHstDir()
MsgBox “My application history directory is located at “ & MyHistDIr
```

SaveAlarmFile (*numType, optRemotePath*)

Function

This function is used to enable or disable the saving feature for Alarm history. It is also used to specify the path where the Alarm history files are saved.

Note: this function is not supported on Thin Client environments.

Parameters

numType is an IWS numeric tag, expression, or VBScript variable signifying the type of operation to be performed. Valid values are:

0 Disable saving the alarm file
1 Enable saving the alarm file to the local disk
2 Enable saving the alarm file to the local disk and to the remote path specified in the *optRemotePath* parameter.

optRemotePath is an optional IWS string tag, expression, or VBScript variable containing the name of the remote computer and path where the Alarm file will be saved (simultaneously save to the local computer) when the *numType* parameter contains a value of 2.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

0 Success
1 Second parameter is not a string
2 Second parameter is missing

Example IWS Usage

Tag	Expression
RetVal	SaveAlarmFile (2, “Z:\MyApp\AlarmFiles”)

Example VBScript Usage

```
Dim StrMyPath, RtnVal
StrMyPath = “Z:\MyApp\AlarmFiles”
RtnVal = $SaveAlarmFile (2, StrMyPath)
```

SetAppAlarmPath (*strPath*)

Function

This function is used to set the Alarm path for the current application.

Note: this function is not supported on Thin Client environments.

Parameters

strPath is an IWS string tag, expression, or VBScript variable containing the new Alarm path for the current application.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

There are no return values

Example IWS Usage

Tag	Expression
	SetAlarmPath (“C:\MyApp\Alarm”)

Example VBScript Usage

```
Dim StrMyPath
StrMyPath = “C:\MyApp\Alarm”
$SetAlarmPath (StrMyPath)
```

SetAppPath (*strPath*)

Function

This function sets the new path for an application. After this function is executed, IWS will look for all the configuration files (screens, alarms, trends, database, events, web) in this new path. However, the existing tag database will continue to be used.

Note: this function is only supported on Windows XP/2000/NT runtime environments.

Parameters

strPath is an IWS string tag, expression, or VBScript variable containing the file path to the application. If the computer is on a network, you can use either //<IP address>/<Path> or //<Host Name>/<Path> syntax.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

0 Failed to set path
1 Succeeded in setting path

Example IWS Usage

Tag	Expression
RtnVal	SetAppPath (“C:\IWS\MyApp”)

Example VBScript Usage

```
Dim StrMyNewPath, RtnVal
StrMyNewPath = “C:\IWS\MyApp\”
RtnVal = $SetAppPath (StrMyNewPath)
If RtnVal = 1 Then
  MsgBox “Set new application path”
End If
```

SetAppHstPath (*strPath*)

Function

This function is used to set the History path for the current application.

Note: this function is not supported on Thin Client environments.

Parameters

strPath is an IWS string Tag, expression, or VBScript variable containing the new History path for the current application.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

There are no return values

Example IWS Usage

Tag	Expression
	SetAppHstPath (“C:\MyApp\Hst”)

Example VBScript Usage

```
Dim StrMyPath
StrMyPath = “C:\MyApp\History”
$SetAppHstPath (StrMyPath)
```

SetTranslationFile (*strFile, optStrColumnName*)

Function

This function sets the active translation file and translates all enabled text within the application.

Note: the Translation option in the Project Settings dialog must be enabled for this function to work.

Parameters

strFile is an IWS string tag, expression, or VBScript variable containing the name of the translation file

optStrColumnName is an IWS string tag, expression or VBScript variable containing the name of the column in the translation file that is to be used to translate text in the application. When omitted, the second column from the translation file will be used by default.

Note: You must use this function in VBScript in order to use a VBScript variable as a parameter

Return Value

0	Successful
1	Wrong number of parameters
2	Wrong parameter type
3	Translation file could not be found or opened

Example IWS Usage

Tag	Expression
RtnVal	SetTranslationFile (“German.tra”)
RtnVal	SetTranslationFile (“MyTranslationFile.csv”, “French”)

Example VBScript Usage

```
Dim rtnVal
rtnVal = $SetTranslationFile (“German.tra”)        ‘Use this format when file contains only one language
MyFile = “MyTranslationFile.csv”
rtnVal = $SetTranslationFile(Myfile, “French”)     ‘Use this format when file contains multiple language
```