

Tech Note 8

Read Data from OPC Server and Store It in a Spreadsheet

Terravic Research, Inc.

Introduction

Tag data from OPC Data Access servers may be stored in any spreadsheet. Our example will show how to read tag information synchronously from an OPC server and dynamically store it in Microsoft Excel spreadsheet. One scripting call will allow us to read tag's value, quality, timestamp, and error code, which we will store in four separate columns of a spreadsheet. The spreadsheet will be updated automatically during script execution with the new tag data.

Platform Requirements

Visual OPCTest® Client 8.0.0000 or later.
Microsoft® Excel® Spreadsheet Application

Examples used in this Tech Note are based on the OPC Foundation sample servers. In order to successfully follow all the procedures outlined in this Tech Note, please download the OPC Foundation's Data Access sample server and the latest Visual OPCTest Client application. If the OPC Foundation sample server is not available, please substitute with another Data Access server and its tag definition. The substitution of server and tag definition in the sample code should be self-explanatory. Additionally, a spreadsheet with an automation interface must also be available to the Visual OPCTest Client, such as Microsoft Excel.

Basic Steps

To setup a script that will automatically read tag data and store it inside a spreadsheet, we need to perform these three steps:

- (1) Connect to the OPC Data Access Server
- (2) Connect to Microsoft Excel
- (3) Setup a FOR Loop to Read and Store the Data

Step 1 - Connect to the OPC Data Access Server

Connect OPC Data Access Server and Add Tag to a Private Group

Run the Client and select *Server | Connect to Server* menu item. The *OPC Server List* dialog will appear. For simplicity, select "OPC Data Access Servers Version 3.0" under LOCAL/ComputerName branch, then select "OPCSample.OpcDaServer.1" server. Click on the *Connect Server* button, then on the *Close* button. Of course, you may select a different server on either local or remote computer. Once we successfully connect to this server, there should be a server listed in the Workspace View with an entry such as "DA1\\LOCAL\\OPCSample.OpcDaServer.1" (this entry will be different based on your server location and name).

Add private group, right-click on "DA1\\LOCAL\\OPCSample.OpcDaServer.1" in a Workspace View, and select *Add Private Group* popup menu item. The *Add Private Group* dialog appears, accept the defaults and click on the *OK* button.

Add tag to the private group, right-click on “Group_” in the Workspace View and select *Add Item* popup menu item. Type “Analog Types/Int” in *Item ID* edit box, click on the *OK* button.

Create Connection Script

After we connect to one tag of the OPC Data Access server, we will create a Connection Script. Connection Script will program all the manual steps we just performed of connecting to the server, adding private group, and getting one tag handle.

Select from the main menu: *Script | Generate Connection Script | VBScript*. New script appears in the Script view. Save the script as “ReadIntoExcel.otx” file.

Step 2 – Connect to Microsoft Excel

We need to modify our automatically generated Connection Script from Step 1, by adding code to create the Microsoft Excel object dynamically.

```
REM create excel object  
Set objXL = CreateObject("Excel.Application")  
objXL.Workbooks.Add
```

This code will create an Excel object and add one workbook to it. Workbook actually holds sheets with cells that we can use to store out tag data. Once we are done with the Excel object we need to release it from memory.

```
REM close the spreadsheet object  
Set objXL = Nothing
```

Step 3 – Setup a For Loop to Read and Store the Data

The For loop will be used to read tag data from OPC Data Access server. Every time we read tag data using the DEVICE parameter, we will receive its value, quality, timestamp, and error code. All that data will be placed in a spreadsheet.

```
REM number of reads  
MaxLoop = 10  
  
REM milliseconds between reads  
REM set to 1 second  
SleepTime = 1000  
  
REM run the For loop MaxLoop times  
For Index = 1 to MaxLoop  
  
    REM display in the Status Pane current read count  
    OPC.StatusPaneMsg "Read Item Count " & Index  
  
    REM wait SleepTime msec between reads  
    OPC.Wait SleepTime  
  
    REM read tag data from device  
    FromDevice = TRUE
```

```
OPC.ReadItemData ItemID1, FromDevice, Value, Quality,  
Timestamp, Error
```

```
REM place read results in a spreadsheet  
objXL.Range("A" & Index).Value = Value  
objXL.Range("B" & Index).Value = Quality  
objXL.Range("C" & Index).Value = Timestamp  
objXL.Range("D" & Index).Value = Error
```

```
REM show the spreadsheet updating  
objXL.Visible = True
```

```
REM update our client window, but also process Ctrl+Break  
REM request to break-out of the script execution  
OPC.UpdateMainWindow
```

Next

The *MaxLoop* variable holds the maximum loop count. The scripting function *UpdateMainWindow* may be used to break the script execution (Client window must have focus), if this variable is very large. The *SleepTime* variable defines the number of milliseconds between read calls (1 second in our example). The *ReadItemData* function accepts two [in] parameters (*ItemID1*, *FromDevice*) and returns four [out] parameters (*Value*, *Quality*, *Timestamp*, *Error*). These four parameters are then stored in the spreadsheet using the *objXL.Range* call. Each read value is stored on a separate line along with the quality, timestamp and error code.

Conclusion

This simple example shows the ease-of-use and the power of scripting functions. Data may be read from a server using different read methods and read intervals. Any automation object may then be used to store the data, not necessarily inside a spreadsheet, but also in a database, or another automation-based OPC Server. More spreadsheet examples are available in the on-line help file, such as an example that reads data into an Excel spreadsheet, and then automatically charts it.

Sample Code File

Complete sample code file: *MyReadIntoExcel.otx*
Sample code file may be found upon installation of the Visual OPCTest Client software.

Contact

Contact Terravic with any questions regarding Visual OPCTest Client. Please let us know your concerns, desired improvements, or current issues with our software. We pride ourselves in making changes quickly based on your input. Please contact us at support@terravic.com.

www.opctest.com
www.terravic.com

