THORLABS

Application Note

CCS Python OSA Script Programming

This is a brief overview of how to get started making a custom Thorlabs OSA script to communicate with a CCS series Compact CCD Spectrometer. While this application note is written for the CCS175 Spectrometer, the process is similar for the CCS100 and CCS200 Spectrometers. The example script is for reference only; the user is encouraged to extend or modify the script to fit his or her specific needs.

Author: Sean Tanner
Date: 16-06-2015

Table of Contents

Author: Sean Tanner
Date: 16-06-2015

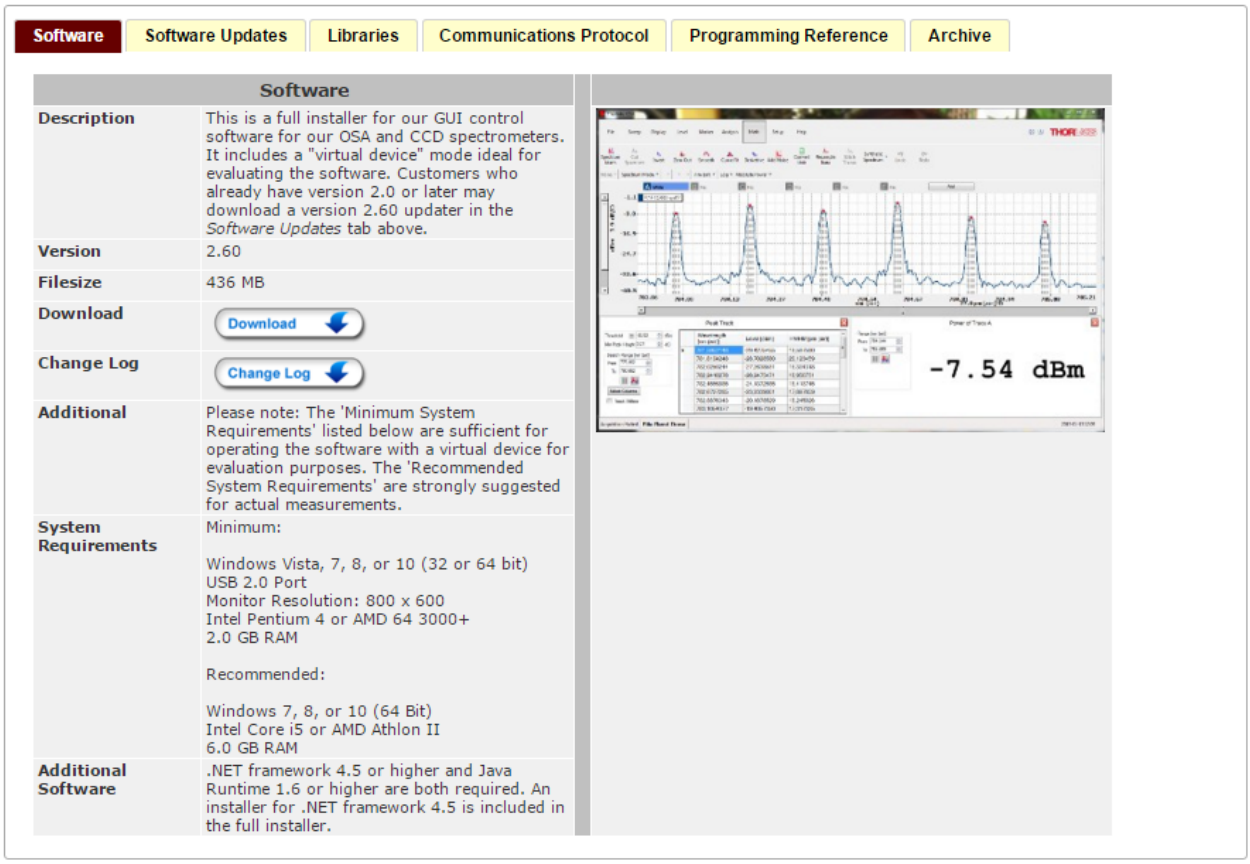
Part 1. Preface

This application note was written for the CCS175 Compact CCD Spectrometer using the firmware and software versions detailed below. Functionality and procedures may vary when using other controllers or firmware/software versions.

- CCS175 Firmware: Version 1.3.0/1.5.0
- Thorlabs OSA Software: Version 2.60
- IronPython: Version 2.7.5

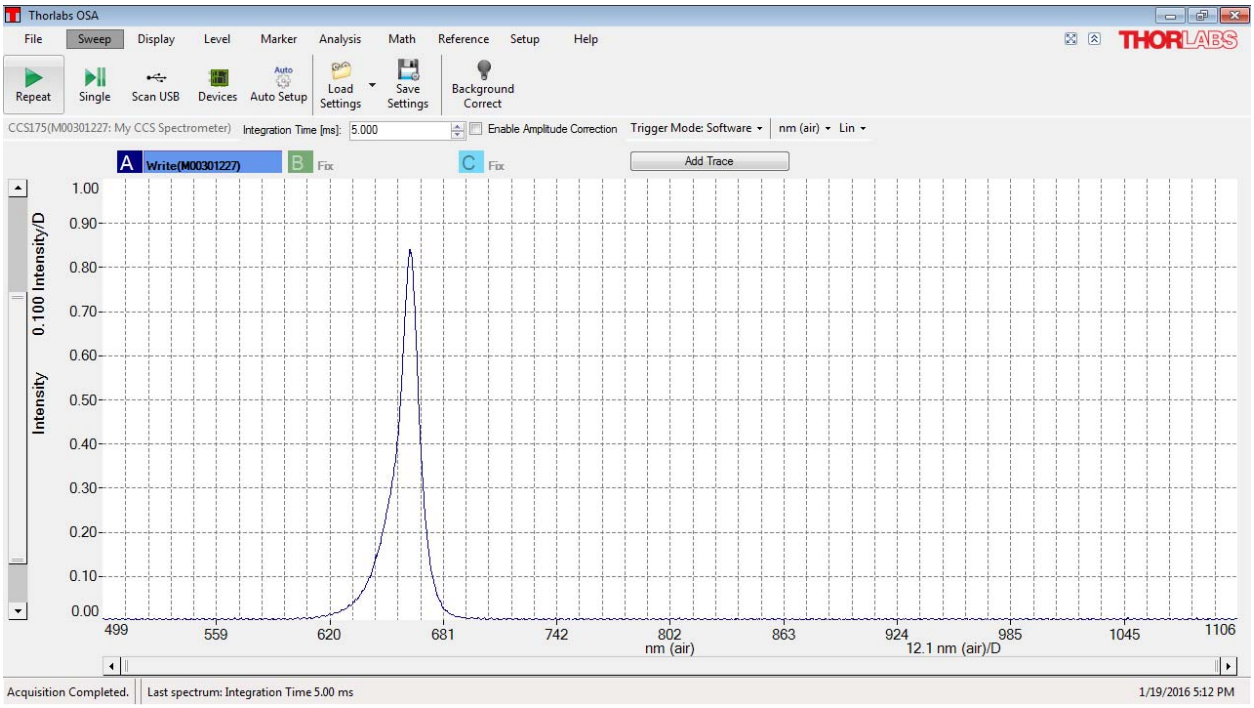
Part 2. Step by Step Instructions

1. Download and install the software for the Optical Spectrum Analyzer and Compact CCD Spectrometers located on the Software tab here: http://www.thorlabs.com/software_pages/viewsoftwarepage.cfm?code=OSA

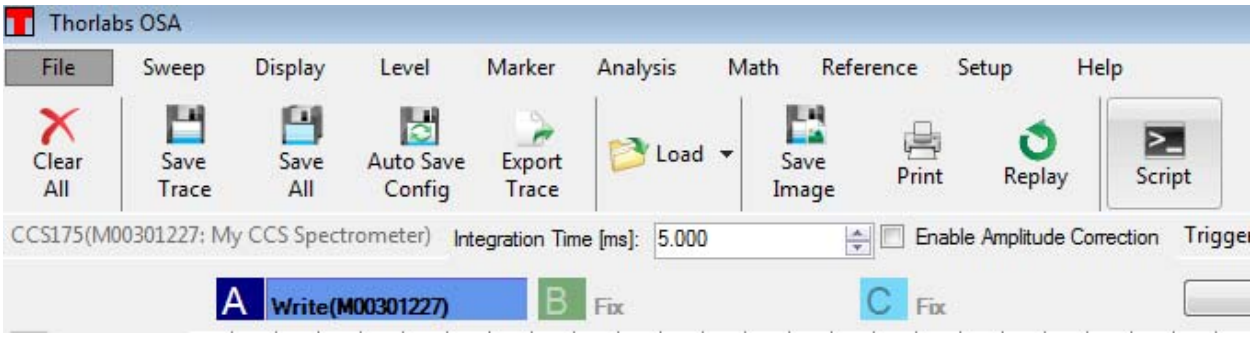


2. Download and install IronPython from http://ironpython.net/.

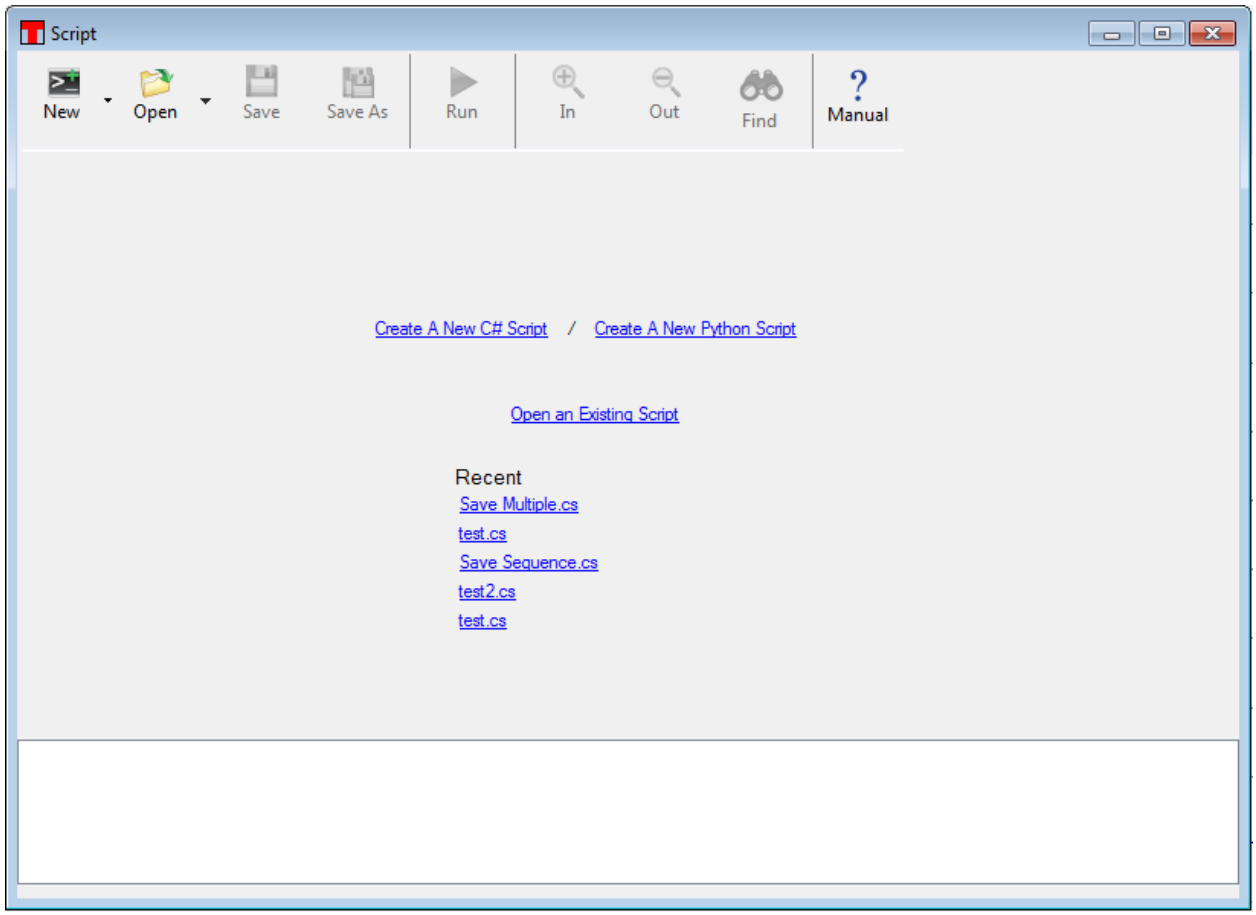
3. Run the Thorlabs OSA software to verify that your instrument is working with the computer correctly and become familiar with the operation of the device. The sample script will save a series of traces from Trace A into a single csv file. The script runs within the OSA software and will use the settings chosen in the OSA software.



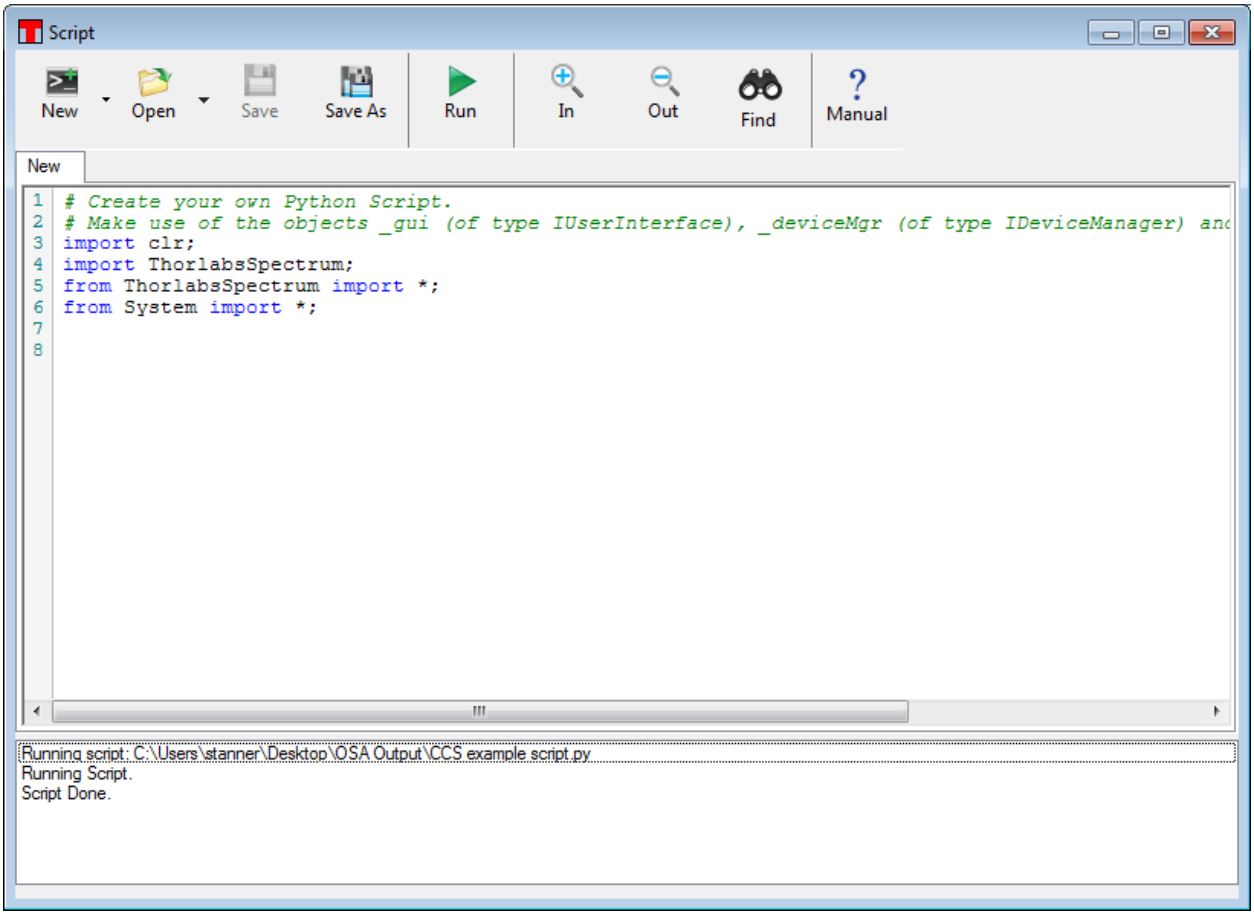
4. From the File menu select Script.



5. Create a new Python Script



6. Your empty script will look similar to the empty script below.



7. Create a new CCS object (ccs) and connect to the active deice.

```
# Create a CCS spectrometer object and connect to the active device
myccs = _deviceMgr.GetActive[CCS]()
```

8. Assign the number of traces to save to a variable (num).

```
# Assign the number of traces to save to a variable
num = 10
```

9. Declare a new list which will be used to store acquired spectrum traces

```
# Declare an new list to hold the spectrum traces
traces = []
```

10. Collect spectra in a loop. In each iteration of the loop the CCS will collect new data and a copy of the spectrum data in trace Trace A will be saved in the traces list. Trace A does not need to be the active trace nor does it have to be directly related to the collected CCS data.

```
# Collect spectra
for i in range(0, num):
    # Get  new data from the CCS spectrometer
    myccs.CollectSpectra(1)

    # Store data from Trace A (index 0) in the a list
    traces.append(_gui.GetSpectrumTrace(0).Spectrum.Clone())
```

11. Direct the script output to a file where the data will be saved. Note the double backslash in the string represent only a single \ character in the file path.

```
# Direct the script output to a csv file so we can save the data
_script.DirectOutputToFile("C:\\Users\\stanner\\Desktop\\OSA Output\\test.csv")
```

12. Fill a data string with column headers and write the line to the file.

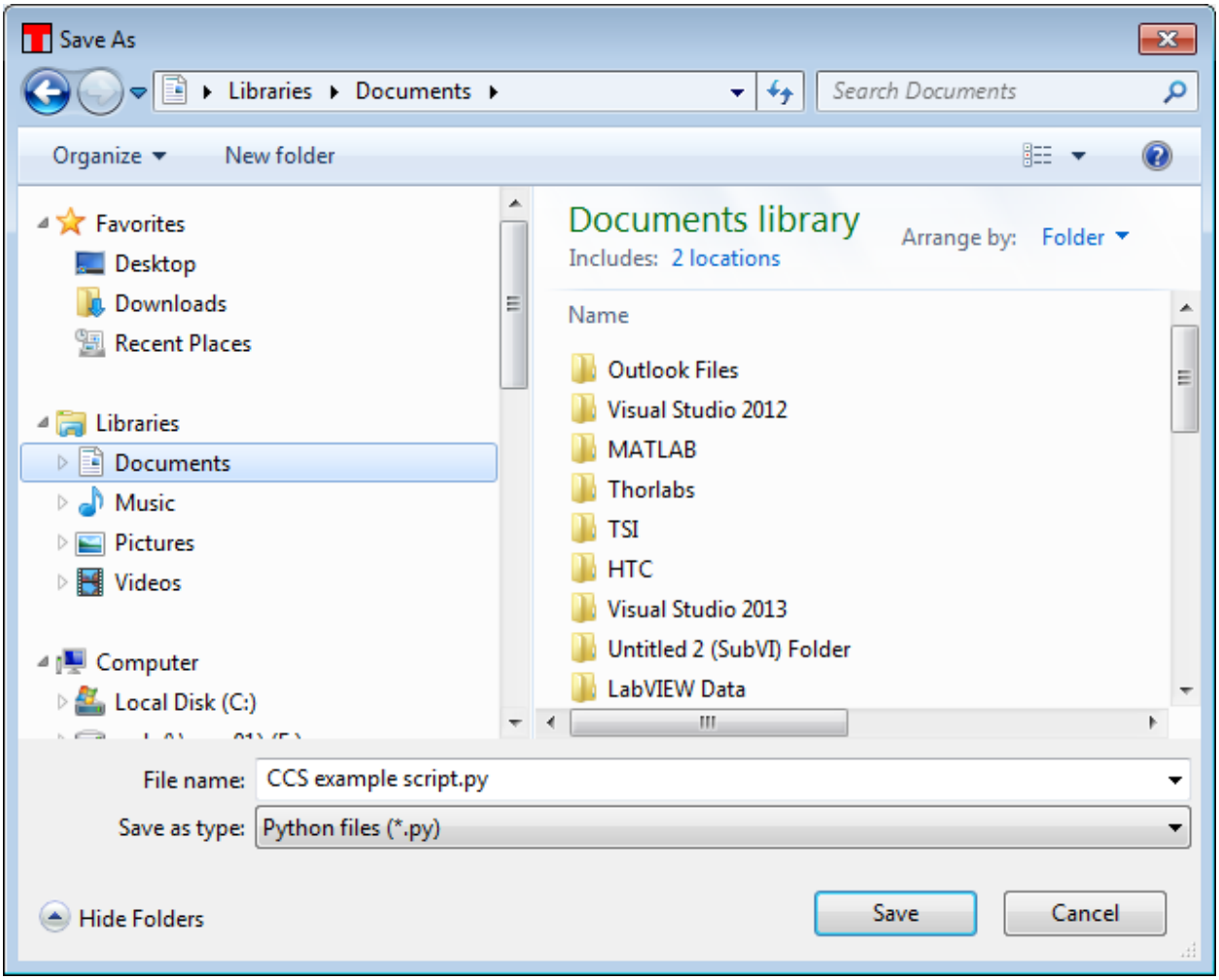
```
# Fill data string with column headers
data = "Wavelength (nm)"
for i in range(0, num):
    data = data + ",Scan " + str(i)

# Write headers to the file
_script.WriteLine(data)
```

13. Loop through each line in the array. Add each element to the data string and write the string to the file.

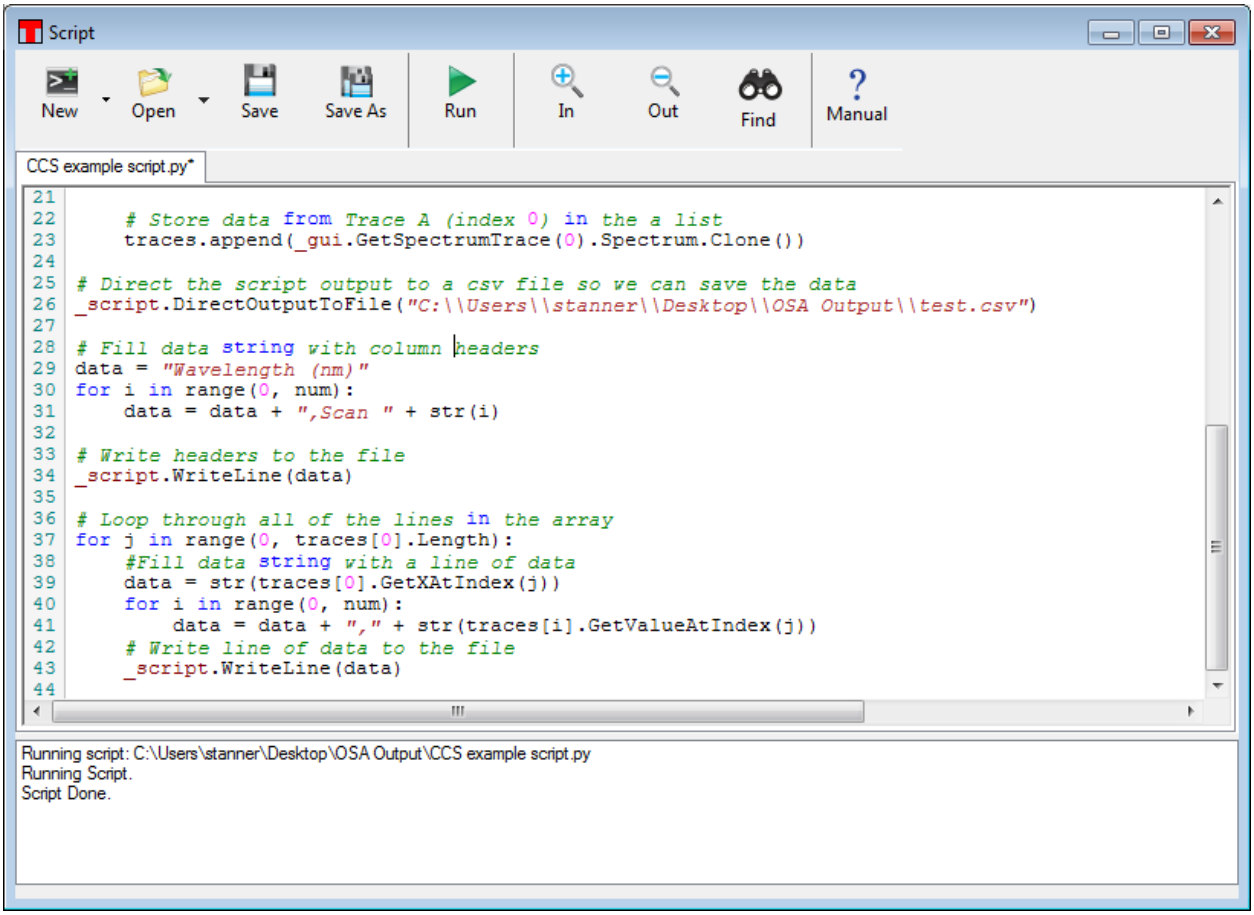
```
# Loop through all of the lines in the array
for j in range(0, traces[0].Length):
    #Fill data string with a line of data
    data = str(traces[0].GetXAtIndex(j))
    for i in range(0, num):
        data = data + "," + str(traces[i].GetValueAtIndex(j))
    # Write line of data to the file
    _script.WriteLine(data)
```

14. Press the Run button at the top of the script editor.

15. You will be prompted to save the script (if you have not already done so).



16. The script will compile and run. Any messages or errors will appear in the bottom window of the script editor.


Script
New
Open
Save
Save As
Run
In
Out
Find
Manual
CCS example script.py*
21
22 # Store data from Trace A (index 0) in the a list
23 traces.append(_gui.GetSpectrumTrace(0).Spectrum.Clone())
24
25 # Direct the script output to a csv file so we can save the data
26 _script.DirectOutputToFile("C:\\Users\\stanner\\Desktop\\OSA Output\\test.csv")
27
28 # Fill data string with column headers
29 data = "Wavelength (nm)"
30 for i in range(0, num):
31 data = data + ",Scan " + str(i)
32
33 # Write headers to the file
34 _script.WriteLine(data)
35
36 # Loop through all of the lines in the array
37 for j in range(0, traces[0].Length):
38 #Fill data string with a line of data
39 data = str(traces[0].GetXAtIndex(j))
40 for i in range(0, num):
41 data = data + "," + str(traces[i].GetValueAtIndex(j))
42 # Write line of data to the file
43 _script.WriteLine(data)
44
Running script: C:\Users\stanner\Desktop\OSA Output\CCS example script.py
Running Script.
Script Done.

Part 3. Methods Used

T ThorlabsSpectrum.IDeviceManager.GetActive[T]> ()

Summary:
This function returns the active spectrum device case as the specified type (CCS or OSA)

Parameters:
None

Type Constraints:
T: The generic type (class name within []) can be any type of interface that the device can implement, e.g. OSA, CSS, or IEnvironmentMeasuringDevice.

Returns:
The active spectrum device or null if the active spectrum device cannot be cast to the specified type.

ISpectrum

ISpectrum is a general interface that all spectrum containers implement. This provides the common elements necessary to handle the spectrum correctly in the OSA Software.

---

void ThorlabsSpectrum.ISpectrumDevice.CollectSpectra(int number)

Summary:
Collect the specified number of spectra from the device. The function will not return until the specified number of spectra has been collected.

Parameters:
number: The number of spectra to collect

Returns:
void

ITrace ThorlabsSpectrum.IUserInterface.GetSpectrumTrace(Int32 index)

Summary:
Retrieves the spectrum trace with the given index.

Parameters:
index: Index of trace to be retrieved. The index starts at zero for the first trace and goes up to the number of traces minus one.

Returns:
The specified spectrum trace.

ISpectrum ThorlabsSpectrum.ITrace.Spectrum

The Spectrum is the spectral contents of a trace.

---

abstract TraceData ThorlabsSpectrum.Data.TraceData.Clone()

Creates and returns a clone of the trace.

---

void ThorlabsSpectrum.IScriptEngine.DirectOutputToFile(string fullFileName)

Summary:
Redirects the script output to the file with the given file name. If the file does not exist it will be created.

Parameters:
fullFileName: Full path and file name for the output file.

Returns:
void

---

void ThorlabsSpectrum.IScriptEngine.WriteLine (string message)

Summary:
Writes one message string to the output.

Parameter:
message: string which will be written to the output.

Double ThorlabsSpectrum.Data.TraceData.GetXAtIndex(Double index)

Summary:
Gets the x value (wavelength) at a given index in a trace

Parameter:
index: The index to get the x value for. The index starts at zero and goes to the number of points in the trace minus one.

Returns:
The x value at the given index. The units will depend on the settings of the OSA software.

Double ThorlabsSpectrum.Data.TraceData. GetValueAtX(Double index)

Summary:
Gets the y value (intensity) at a given index in a trace

Parameter:
index: The index to get the y value for. The index starts at zero and goes to the number of points in the trace minus one.

Returns:
The y value at the given index. The unit will depend on the settings of the OSA software.

Part 4. Full Program

```
# Create your own Python Script.
# Make use of the objects _gui (of type IUserInterface), _deviceMgr (of type
IDeviceManager) and _script (of type IScriptEngine)
import clr;
import ThorlabsSpectrum;
from ThorlabsSpectrum import *;
from System import *;

# Create a CCS spectrometer object and connect to the active device
myccs = _deviceMgr.GetActive[CCS]()

# Assign the number of traces to save to a variable
num = 10

# Declare an new list to hold the spectrum traces
traces = []

# Collect spectra
for i in range(0, num):
    # Get  new data from the CCS spectrometer
    myccs.CollectSpectra(1)

    # Store data from Trace A (index 0) in the a list
    traces.append(_gui.GetSpectrumTrace(0).Spectrum.Clone())

# Direct the script output to a csv file so we can save the data
_script.DirectOutputToFile("C:\\Users\\stanner\\Desktop\\OSA
Output\\test.csv")

# Fill data string with column headers
data = "Wavelength (nm)"for i in range(0, num):
    data = data + ",Scan " + str(i)

# Write headers to the file
_script.WriteLine(data)

# Loop through all of the lines in the array
for j in range(0, traces[0].Length):
    #Fill data string with a line of data
    data = str(traces[0].GetXAtIndex(j))
    for i in range(0, num):
        data = data + "," + str(traces[i].GetValueAtIndex(j))
    # Write line of data to the file
    _script.WriteLine(data)
```

Part 5. Other Resources

More information about the commands available to OSA scripts can be found in the Scripting Thorlabs OSA manual. The manual can be found in the Windows start menu Thorlabs>Thorlabs OSA>OSA>GUI folder.

The manual can also be opened by clicking on the Manual button in the OSA software: