SoundCheck TCP/IP Python Library (soundcheck_tcpip)

The SoundCheck TCP/IP Python library is included for users who want to control SoundCheck with Python scripts. This library greatly simplifies the process of launching SoundCheck, running sequences, and interacting with the Memory List.

Getting Started

Before using soundcheck_tcpip you must install it using **pip**.

From a Terminal (macOS) or DOS shell (Windows), CD to ${SOUNDCHECK_ROOT}/External Control Examples/Python where ${SOUNDCHECK_ROOT} is the location of your SoundCheck installation (e.g., C:\SoundCheck 18).

Run:

```
pip install soundcheck_tcpip-18.0.0-py3-none-any.whl
```

Classes and Methods

SoundCheck Installation (SCInstallation)

The SCInstallation object manages an installed version of SoundCheck.

SCInstallation Methods	
Method	**Description / Example Code**

construct_installation(
`  version,`
`  path`
)
	Returns a `SCInstallation` object for the `version` of SoundCheck located at `path` where `version` is a list containing the major and minor version numbers (e.g., [18, 0])
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18,0], 'C:\SoundCheck 18')`

launcher_path()	**Returns the full path to the SoundCheck executable**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_path = sc_inst.launcher_path()`

Launch Methods	
Method	**Description / Example Code**

launch()	**Launches SoundCheck specified by a `SCInstallation` object**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_inst.launch()`

terminate()	**Terminates the SoundCheck that was launched by a `SCInstallation` object**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_inst.launch()`
	`sc inst.terminate()`

is_running()	**Returns True if SoundCheck specified by a `SCInstallation` object is running**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`running = sc_inst.is_running()`

INI File Methods	
Method	**Description / Example Code**

ini_path()	**Returns the full path to a `SCInstallation` INI file**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_path = sc_inst.ini_path()`

get_ini_option(
`   section,`
`   option,`
`   default`
)
	Attempt to read an `option` **from a** `SCInstallation` **INI file where `section` is a string identifying where in the INI file where the `option` is located. If the `option` is not found in the ini file,** `get_ini_option` **returns the `default` value.**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`option = sc_inst.get_ini_option('Dialogs', 'SHOW SPLASHSCREEN')`

set_ini_option(
`   section,`
`   option,`
`   value`
)
	Attempt to write an `option` **to a** `SCInstallation` **INI file where `section` is a string identifying where in the INI file where the** `option` **is located and** `value` **is written to the INI for option.**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_inst.set_ini_option(`
	`    'Dialogs', 'SHOW SPLASHSCREEN', 'FAlSE'`
	`)`

File Import Methods	
Method	**Description / Example Code**

import_ini(ini_file)	**Import an INI file into an installation where** `ini_file` **is a string containing the path to the INI file**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_inst.import_ini('C:\Users\Me\MySoundCheckINI.ini')`

import_har(har_file)	**Import a hardware file into an installation where** `har_file` **is a string containing the path to the hardware file**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_inst.import_har('C:\Users\Me\MySoundCheckHAR.har')`

import_cal(cal_file)	**Import the calibration file into an installation where** `cal_file` **is a string containing the path to the calibration file**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_inst.import_cal('C:\Users\Me\MySoundCheckCAL.cal')`

Miscellaneous Methods	
Method	**Description / Example Code**

Miscellaneous Methods	
Method	**Description / Example Code**

server_details()	**Returns the IP address and port number for an** `SCInstallation` **TCP/IP server**
	
	`from soundcheck_tcpip.soundcheck.installation import construct_installation`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`ip address, port = sc inst.server details()`

SoundCheck TCP/IP Control (SCControlTCPIP)

The SCControlTCPIP object manages the TCP/IP connection to a SoundCheck installation.

Construction	
Method	**Description / Example Code**

SCControlTCPIP(
`installation`
)
	Creates an SCControlTCPIP object from the `SCInstallation` **object,** `installation`**, and enables its TCP/IP server where:**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`

Launch Methods	
Method	**Description / Example Code**

launch(timeout=30)	**Starts SoundCheck and waits for it to be ready to respond to external commands. If SoundCheck does not respond within** `timeout` **seconds, an exception is raised**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`

close(timeout=5)	**Sends SoundCheck a shutdown command and force exit if SoundCheck does not exit within** `timeout` **seconds**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.close()`

Sequence Methods	
Method	**Description / Example Code**

open_sequence(
`path,`
`timeout=10`
)
	Sends a command for SoundCheck to open the sequence located at `path`**. If the sequence does not open within** `timeout` **seconds, an exception is raised**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`

current_sequence()	**Returns the name of the currently loaded sequence**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`

Sequence Methods	
Method	**Description / Example Code**

	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`sqc_name = sc_tcpip.current_sequence()`

get_sequence_path()	**Returns the full path name of the currently loaded sequence**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`sqc path = sc tcpip.get sequence path()`

get_sequence_steps_list()	**Returns a list of dicts containing the name, type, input channels, and output channels of the currently loaded sequence. The format of the response is:**
	
	`[`
	`  {`
	`  'Name': step1_name,`
	`  'Type': step1_type,`
	`  'InputChannelNames': [list of input channel names],`
	`  'OutputChannelNames': [list of output channel names]`
	`  },`
	`  .`
	`  .`
	`  .`
	`]`
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`sqc_steps = sc_tcpip.get_sequence_steps_list()`

run_sequence(
`  path=None,`
`  timeout=300,`
`  iterations=1,`
`  respond=True`
)
	Sends a command for SoundCheck to open and run the sequence located at `path` **for the specified number of** `iterations`**. If the sequence does not open and run within** `timeout` **seconds, an exception is raised. If** `respond` **is** `True`**, a list of responses from each sequence iteration is returned, otherwise** `None` **is returned.**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`sc_tcpip.run_sequence()`

Sequence Methods	
Method	**Description / Example Code**

save_sequence()	**Saves the currently loaded sequence to disk.**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`    import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`sc tcpip.save sequence()`

get_sequence_duration()	**Returns the time (in seconds) that it took for the last sequence run.**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`    import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`    import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.open_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`sc_tcpip.run_sequence()`
	`duration = sc_tcpip.get_sequence_duration()`

Memory List Methods	
Method	**Description / Example Code**

get_memlist_names(
`data_type=None`
)
	Returns a dict of name lists keyed by data_type where `data_type` **can be one of** **`'Curves'`**, **`'Values'`**, **`'Results'`**, **or** **`'Waveforms'`**. **If** `data_type` **is not specified,** `get_memlist_names` **returns the names of all items of all data types as a list of strings.**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.run_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`ml names = sc tcpip.get memlist names()`

get_result(name)	**Gets the result from the Memory List identified by** **`name`**
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.run_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`result = sc tcpip.get result('My Result')`

get_value(name)	**Returns a dict representing the value in the Memory List identified by** **`name`**. **The format of the response is:**
	
	`{`
	`  'Name': name,`
	`  'XData': X data,`
	`  'YData': Y data,`
	`  'ZData': Z data,`
	`  'XUnit': X units,`
	`  'YUnit': Y units,`
	`  'ZUnit': Z units,`
	`  'XDataScale': X data scale,`
	`  'YDataScale': Y data scale,`
	`  'ZDataScale': Z data scale,`
	`  'XdBRef': X dB reference,`
	`  'YdBRef': Y dB reference,`
	`  'ZdBRef': Z dB reference,`
	`  'XAxisScale': X axis scale,`
	`  'YAxisScale': Y axis scale,`
	`  'ZAxisScale': Z axis scale,`
	`  'Protected': protected status`
	`}`
	
	`from soundcheck_tcpip.soundcheck.installation`
	`  import construct_installation`
	`from soundcheck_tcpip.soundcheck.controller`
	`  import SCControlTCPIP`
	
	`sc_inst = construct_installation([18, 0], 'C:\SoundCheck 18')`
	`sc_tcpip = SCControlTCPIP(sc_inst)`
	`sc_tcpip.launch()`
	`sc_tcpip.run_sequence('C:\Users\Me\Sequences\My Seq.sqc')`
	`value = sc_tcpip.get_value('My Value')`

get_curve(name)	**Returns a dict representing the curve from the Memory List identified by** `name`**. The format of the response is:**
	
	`{`
	`  'Name': name,`
	`  'XData': X data,`
	`  'YData': Y data,`
	`  'ZData': Z data,`
	`  'XUnit': X units,`
	`  'YUnit': Y units,`
	`  'ZUnit': Z units,`
	`  'XDataScale': X data scale,`
	`  'YDataScale': Y data scale,`
	`  'ZDataScale': Z data scale,`
	`  'XdBRef': X dB reference,`
	`  'YdBRef': Y dB reference,`
	`  'ZdBRef': Z dB reference,`