VICON ENTERTAINMENT

BEYOND MOTION

PYTHON SCRIPTING WITH VICON SHOGUN

WHAT'S INSIDE

Revision 1. For use with Vicon Shogun 1.11.

Vicon Motion Systems is an Oxford Metrics plc company.

Email: support@vicon.com Web: http://www.vicon.com

About Python scripting with Vicon Shogun

This guide contains introductory information about using Python with Vicon Shogun:

Python scripting with Shogun Live

About Python scripting with Shogun Live

These topics provide basic, introductory information to get you started using Python with Shogun Live:

Install the Shogun Live Python module

To use Python and the Shogun Live Python module, you must first install them.

Both Python 2 and Python 3 are supported. Vicon recommends that you use the latest full release of Python 3, unless your project requires you to use a specific version of Python.

> **Note**
> With Shogun 1.4 and later, Python 3.0 and later is fully supported. Legacy support only is provided for Python 2.7.

For information on installation, see:

- Install Python (page 5)
- Install the Python module (page 6)
- Check that the Python module installed correctly (page 10)

Install Python

To install Python 2 or 3:

1. Go to https://www.python.org/downloads/
2. Locate the required version and install Python, ensuring that **Add Python #.# to PATH** is selected:

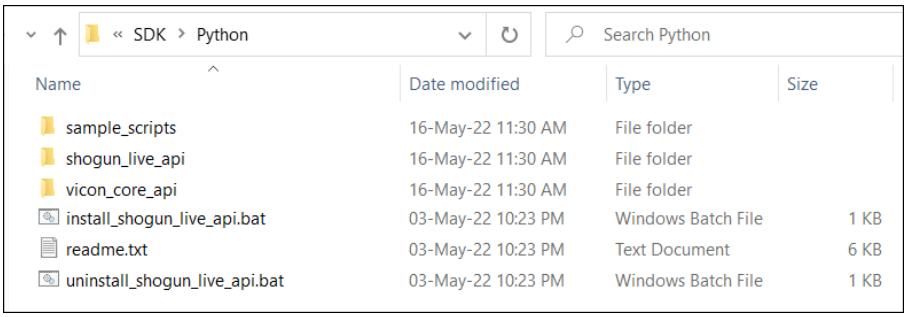
Install the Python module

To install the Python module:

1. Locate the installation files, which, if you installed Shogun Live in the default location, are found in this folder:

   C:\Program Files\Vicon\ShogunLive#.#\SDK\Python

   These files are displayed:



2. You can install the Shogun Live Python module in either of the following ways, depending on your particular installation:

   - The simplest way is to run the batch file (*install_shogun_live_api.bat*) that is included in the Shogun Live installation (as shown in the above image). This usually works well if:
     - Python was installed to the PATH variable; or
     - Multiple versions of Python are installed but you want to install the API to the latest version that you installed; or
     - Only a single version of Python is installed.

     If any of these conditions apply, install the Python module by running the batch file (page 7).

   - In all other cases, install the Python module by using pip. This usually applies if:
     - Multiple versions of Python are installed but you want to install to a specific version; or
     - Multiple different versions of Python are installed and you want to install to all of them (in this case, you must install the module for each version); or
     - Only a single version of Python is installed, but you didn't install to PATH.

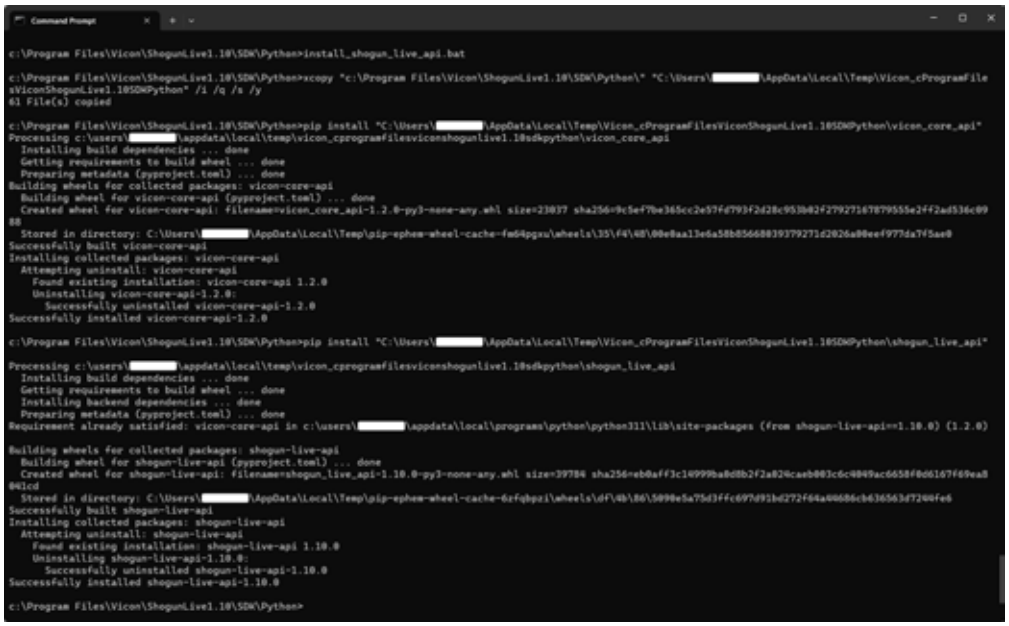
     If any of these conditions apply, install the Python module by running pip (page 8).

Python scripting with Shogun Live

Install the Shogun Live Python module

Install the Python module by running the batch file

To do this, in a cmd window, run the batch file, *install_shogun_live_api.bat*.



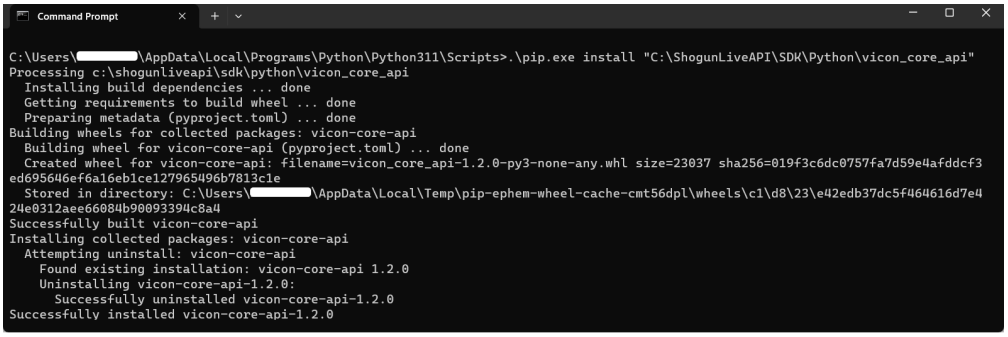
Install the Python module by running pip

> **Important**
> If you are using Python 3, you must copy the *SDK* folder from the *Program Files* directory to a new folder that doesn't require Administrator permission. This is due to a permissions issue with pip versions 21.3 or later.
> By default, you can find the *SDK* folder at *C:\Program Files\Vicon\ShogunLive1.11\SDK* (if you installed Shogun in another location, use this path to find the SDK folder).
> We recommend copying the *SDK* folder into a new folder with an easy-to-access location, for example, "*C:\ShogunLiveAPI*". You can delete this folder after the pip installation is complete.
> The steps in this guide assume you have copied the *SDK* folder to *C:\ShogunLiveAPI*.

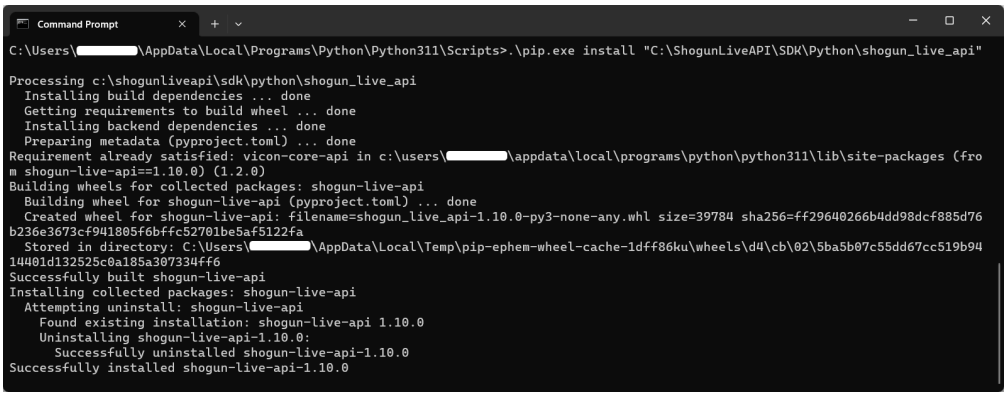
1. Navigate to the *Scripts* folder for the version of Python that you want to use for pip installation.

   - The default installation folder for Python 3 is: *C:\Users\<user>\AppData\Local\Programs\Python\Python<version>\Scripts*
   - The default installation folder for Python 2.7 is: *C:\Python27\Scripts*

2. Open a command window or powershell in that folder.
3. Run the following command to install the Vicon Core API (assuming you have copied the *SDK* folder to the suggested location):
   `.\pip.exe install "C:\ShogunLiveAPI\SDK\Python\vicon_core_api"`



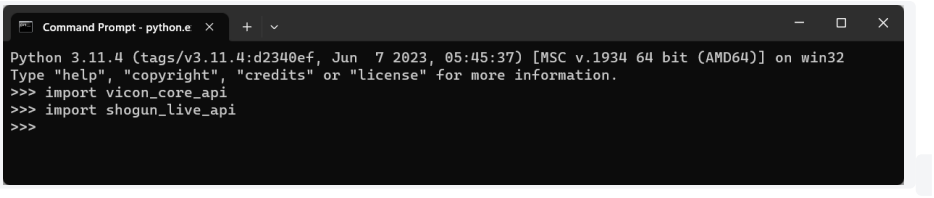
4. Run the following command to install the Shogun Live API (assuming you have copied the *SDK* folder to the suggested location):
   `.\pip.exe install "C:\ShogunLiveAPI\SDK\Python\shogun_live_api"`



Check that the Python module installed correctly

To test that the Shogun Live Python module installed correctly, run Python and try the following import statements:

- `import vicon_core_api`
- `import shogun_live_api`



If these import without any errors then the installation has been successful.

Connect a Python client to Shogun Live

For information related to connecting a Python client to Live, see:

Create a client and check the connection

To connect a Python client to Shogun Live:

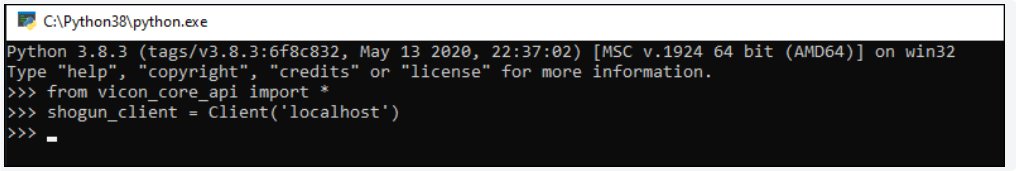
1. Start Shogun Live.
   It is now ready to be interacted with via the Python API.

2. Start Python and begin by importing the Vicon Core API:

   ```
   from vicon_core_api import *
   ```

3. Create a client. In this example, the object is named `shogun_client`:

   ```
   shogun_client = Client('localhost')
   ```



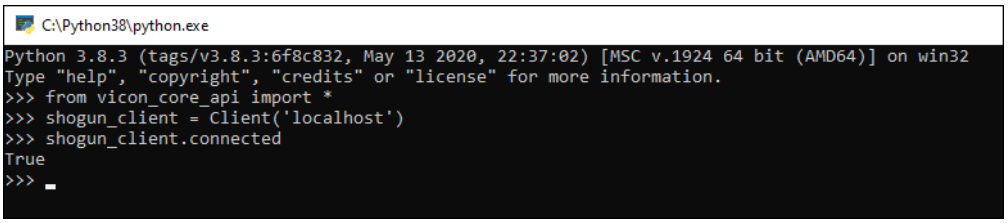
   Where:

   - `shogun_client` is the name of the object, which can be any name you want
   - `localhost` can be replaced with an IP address or hostname running Shogun Live.

4. Check that the client is connected:

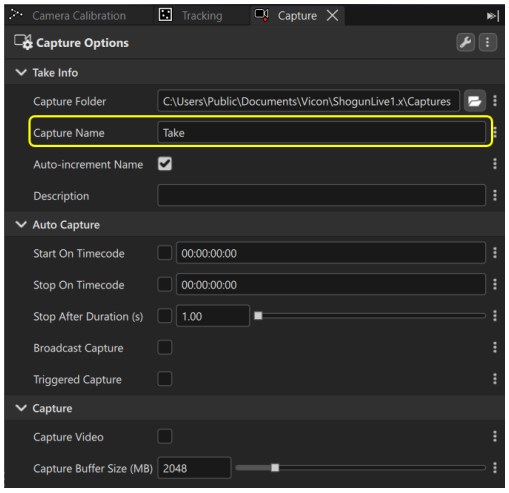
   ```
   shogun_client.connected
   ```

   This returns True if connected, and False if not.



Example: Setting the capture name

In Shogun Live, towards the top of the Capture panel, the default capture name is **Take**.



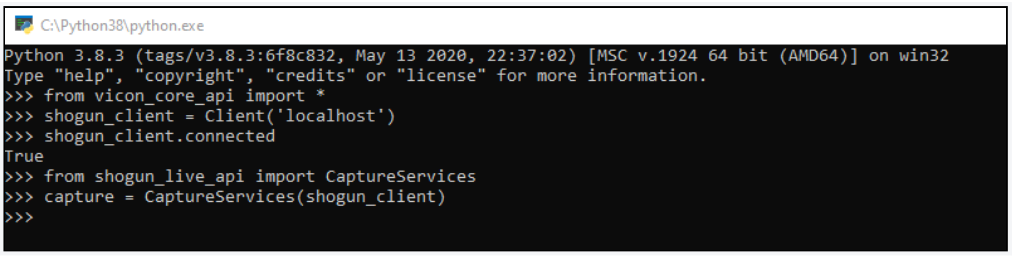
This example shows how you can change the capture name using the Python API.

1. Import the relevant class for interacting with capture:

   ```
   from shogun_live_api import CaptureServices
   ```

2. Create an object for the capture services. In this example, the object is named `capture`, but you can name it as required.
   It references the object name used for the shogun client that was created:

   ```
   capture = CaptureServices(shogun_client)
   ```



3. Use the `set_capture_name` method to set the capture name to `'MyTake001'`:

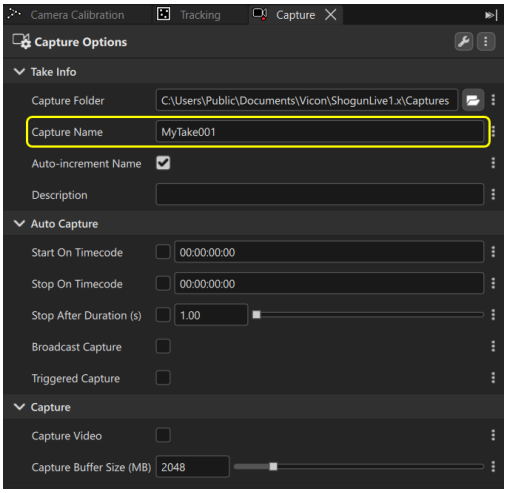
```
result = capture.set_capture_name('MyTake001')
```

```
C:\Python38\python.exe
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from vicon_core_api import *
>>> shogun_client = Client('localhost')
>>> shogun_client.connected
True
>>> from shogun_live_api import CaptureServices
>>> capture = CaptureServices(shogun_client)
>>> result = capture.set_capture_name('MyTake001')
>>>
```

4. Check the return value to make sure that this succeeded:

```
C:\Python38\python.exe
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from vicon_core_api import *
>>> shogun_client = Client('localhost')
>>> shogun_client.connected
True
>>> from shogun_live_api import CaptureServices
>>> capture = CaptureServices(shogun_client)
>>> result = capture.set_capture_name('MyTake001')
>>> result
Result.Ok
>>>
```

In Shogun Live the capture name has changed:



For further information, see:

- The readme file:
  C:\Program Files\Vicon\ShogunLive1.11\SDK\Python\readme.txt

- Sample scripts:
  C:\Program Files\Vicon\ShogunLive1.11\SDK\Python\sample_scripts

Explore using Python

Run the Python `dir()` command on objects created, to show the methods that can be called:

For example, `dir(shogun_client)` or `dir(capture)`

```
>>> dir(shogun_client)
['__class__', '__delattr__', '__dict__', '__dir__', '__doc__', '__enter__', '__eq__', '__exit__', '__format__', '__ge__', '__getattribute__', '__gt__',
 '__hash__', '__init__', '__init_subclass__', '__le__', '__lt__', '__module__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__se
tattr__', '__sizeof__', '__str__', '__subclasshook__', '__weakref__', '_connect', '_locked_send_command', '_message_generator', '_read_loop', '_read_m
essage', '_read_server_version', 'add_callback', 'add_schema_callback', 'callback_id_generator', 'callback_map', 'check_schemas', 'check_schemas_schem
a', 'client_failed_callback', 'condition', 'connect_timeout_seconds', 'connected', 'decoder', 'message_id_generator', 'pending_messages', 'remove_call
back', 'send_command', 'send_json_command', 'send_raw_command', 'send_timeout_seconds', 'server_endpoint', 'server_version', 'socket', 'socket_timeout
_seconds', 'stop', 'thread', 'version']
>>> dir(capture)
['EState', '__class__', '__delattr__', '__dict__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__gt__', '__hash__', '
__init__', '__init_subclass__', '__le__', '__lt__', '__module__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__si
zeof__', '__str__', '__subclasshook__', '__weakref__', '_registered_type_name', 'add_auto_capture_options_changed_callback', 'add_capture_options_chan
ged_callback', 'add_latest_capture_changed_callback', 'add_take_info_changed_callback', 'call', 'cancel_capture', 'capture_description', 'capture_fold
er', 'capture_name', 'capture_notes', 'capture_processed_data_enabled', 'capture_video_enabled', 'client', 'duration_limit_in_seconds', 'latest_captur
e_errors', 'latest_capture_file_paths', 'latest_capture_name', 'latest_capture_state', 'latest_capture_timecode', 'limit_capture_duration_enabled', 'r
emove_callback', 'set_capture_description', 'set_capture_folder', 'set_capture_name', 'set_capture_notes', 'set_capture_processed_data_enabled', 'set_
capture_video_enabled', 'set_duration_limit_in_seconds', 'set_limit_capture_duration_enabled', 'set_start_on_timecode_enabled', 'set_start_timecode',
'set_stop_on_timecode_enabled', 'set_stop_timecode', 'start_capture', 'start_on_timecode_enabled', 'start_timecode', 'stop_capture', 'stop_on_timecode
_enabled', 'stop_timecode', 'unsupported']
```

Run `help()` on specific methods or classes.

For example, `help(capture.set_capture_name)`

```
>>> help(capture.set_capture_name)
Help on method set_capture_name in module shogun_live_api.interfaces.capture_services:

set_capture_name(name) method of shogun_live_api.interfaces.capture_services.CaptureServices instance
    Set the name for the next capture.

    Args:
        name < string >: Name suitable for use in file names.

    Return:
        return < Result >: Ok - On success.
            InvalidArgument - If the name includes invalid characters.

>>>
```

Python scripting with Shogun Post

About Python scripting with Shogun Post

For introductory information about various aspects of using Python with Shogun Post, see these topics:

Install the Shogun Post Python module

To use Python and the Shogun Post Python modules, you must first install them.

Both Python 2 and Python 3 are supported. Vicon recommends that you use the latest full release of Python 3, unless your project requires you to use a specific version of Python.

> ⚠ With Shogun 1.4 and later, Python 3.0 and later is fully supported. Legacy support only is provided for Python 2.7.

For information on installation, see:

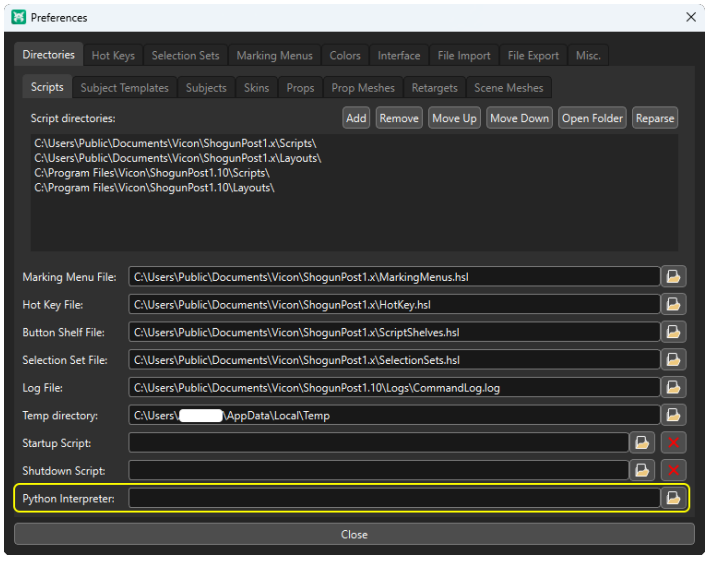
- Install Python (page 18)
- Install the Shogun Post Python modules (page 19)
- Check that the Python modules installed correctly (page 21)

Install Python

To install Python 2 or 3:

1. Go to https://www.python.org/downloads/
2. Locate the required version and install Python, ensuring that **Add Python #.# to PATH** is selected:

To use a Python interpreter with Shogun Post, you must specify it in the Shogun Post **Preferences** dialog box, at the bottom of the **Directories** tab.

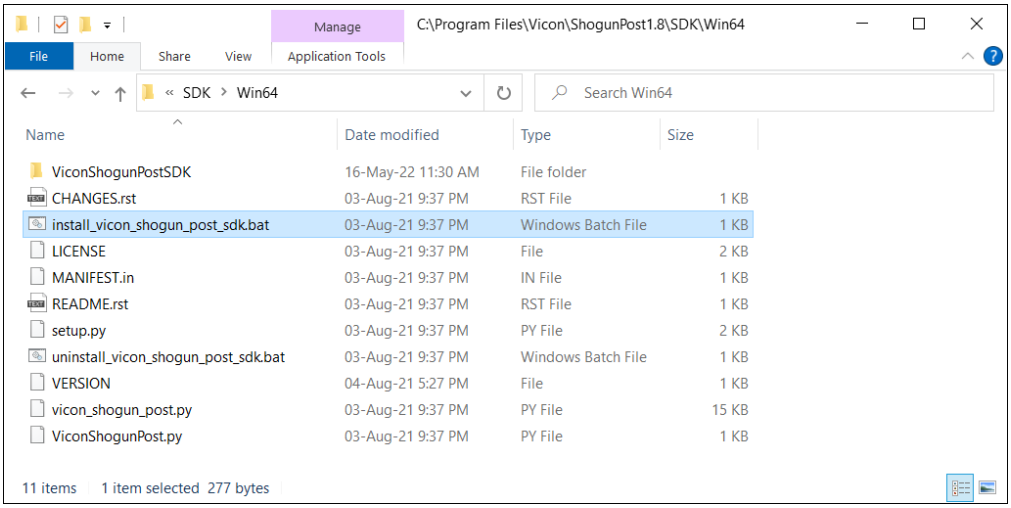


Install the Shogun Post Python modules

> ⚠ This workflow hasn't been tested for Windows 11. For help installing Shogun Post Python modules with Windows 11 or later, contact Vicon Support[1].

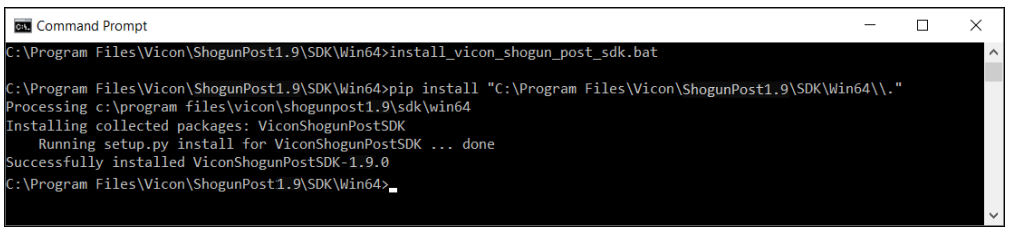
If you set the interpreter, you must also install the Shogun Post Python modules, using the installation utility provided in the following locations:

- For 64-bit installations of the Python interpreter:
  C:\Program Files\Vicon\ShogunPost#.#\SDK\Win64\install_vicon_shogun_post_sdk.bat



- For 32-bit installations of the Python interpreter:
  C:\Program Files\Vicon\ShogunPost#.#\SDK\Win32\install_vicon_shogun_post_sdk.bat

These scripts install the modules using the default *pip* tool.



> ⚠ **Note**
> The same module supports Python 2 and Python 3.

1 mailto:support@vicon.com

Shogun Post Python modules installed

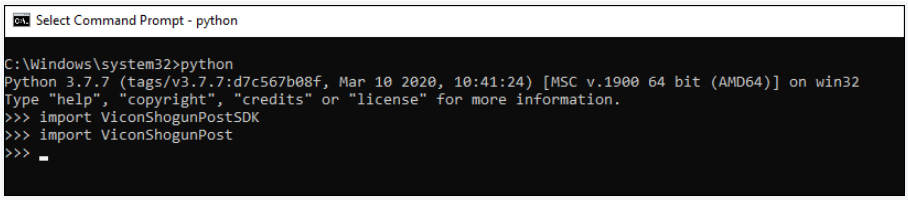
Shogun Post installs two Python modules:

- **ShogunPostSDK**. This folder contains the main Shogun Post SDK, and is the module that is most commonly used.
- **ViconShogunPost**. This module contains the Shogun Post implementation of a common Python SDK, shared by several Vicon products.

Check that the Python modules installed correctly

To test that the Shogun Post Python modules installed correctly, run Python and try the following import statement:

- `import ViconShogunPost`
- `import ViconShogunPostSDK`



If this imports without any errors then the installation has been successful.

Connect a Python client to Shogun Post

When a Shogun Post Python client object is created it automatically tries to connect to a local instance of Shogun Post over TCP/IP via port 803.

```
import ViconShogunPost
shogun = ViconShogunPost.ViconShogunPost()
```

To direct the client to a different instance of Shogun Post, specify the appropriate parameters, or re-direct after creation using the Connect() method.

```
shogun = ViconShogunPost.ViconShogunPost( "192.168.0.0", 804 )
```

You can change the port used by Shogun Post for communication in the **Preferences** dialog box, on the **Misc** tab. The default behavior of Shogun Post is to communicate via port 803, though if multiple instances of Shogun Post are opened on the same machine they will automatically choose different port numbers (searching incrementally upwards). You can specify a fixed port in the **Preferences** dialog box, or from the application command line.

```
ShogunPost.exe -ControlStreamPort 804

ShogunPostCL.exe -controlStreamPort 804
```

> **Tip**
> If you disconnect your Ethernet cable and disable wifi, when you enter a Python command, the following error may be displayed: `Host Application is not connected, unable to retrieve command list` This is because Python connects to Shogun Post over TCP/IP and if you are working entirely offline, Python and Shogun Post cannot connect. To solve this issue, install the Microsoft Loopback Adapter. For instructions on how to do this, see *Adding the MS Loopback Adapter on Windows 7*, on http://blogs.msdn.com.

Use Vicon ShogunPost SDK interfaces

Two kinds of classes are defined in the *ViconShogunPostSDK* module. Classes like *Scene* or *Offline* define interfaces to distinct areas of Shogun Post's functionality. While you can create an instance of one of these interfaces, they can be more easily accessed directly from the Python client.

Example scripts showing how to run operations in Post

To import libraries:

```
import ViconShogunPostSDK
import ViconShogunPost
```

To connect to a local copy of Shogun Post:

```
shogun = ViconShogunPost.ViconShogunPost()
```

To load a file:

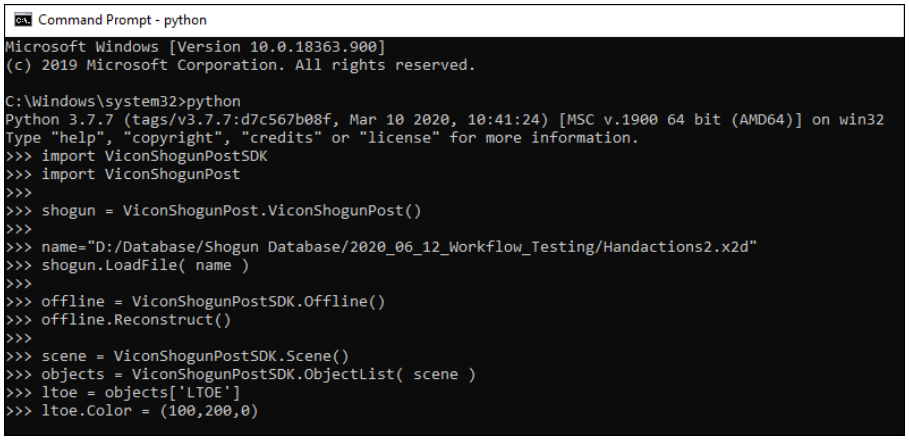
```
name="path and name of file to be loaded using /"
shogun.LoadFile( name )
```

To reconstruct a loaded file:

```
offline = ViconShogunPostSDK.Offline()
offline.Reconstruct()
```

To change the color of a marker:

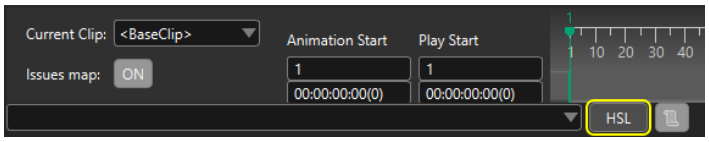
```
scene = ViconShogunPostSDK.Scene()
objects = ViconShogunPostSDK.ObjectList( scene )
ltoe = objects['LTOE']
ltoe.Color(231,101,0)
ltoe.Color = (100,200,0)
```



Switch the command line between HSL and Python

To switch the command line between HSL and Python:

1. Go to the bottom of the Shogun Post window.
2. In the center, under the time bar, click the Python/HSL button:



The Python command line is essentially a mini script editor, not a direct interface to a Python interpreter, so variables will not persist from one call to the next. However, you don't need to create a client object when using the Shogun Post command line: default clients for the two Python modules are always available: `shogunPost` for the ViconShogunPost module, and `sdk` for the ViconShogunPostSDK module:

- `shogunPost.Command()` This is the ViconShogunPost module, which is the simple Vicon SDK.
- `sdk.Command()` This is the ViconShogunPostSDK module, which is the expanded Shogun Post-specific SDK.

Run Python scripts in Shogun Post

You can use Python scripts from anywhere in Shogun Post that currently accepts HSL scripts:

- **General** menu > **Preferences** dialog box > **Directories** tab.
- As a stage in a pipeline
- Attached to a shelf button
  Note that when you add Python to a shelf button, a default shogunPost client is always available for use, as with the Shogun Post command line.
- Python command line (see Switch the command line between HSL and Python (page 25))

Python scripts have an identifying icon in the Script Viewer. You can assign them to hot keys in the same way as HSL commands.

Python / HSL interaction

A command in HSL runs a Python command string, which works in the same way as the Shogun Post command line.

```
python "shogunPost.Command()";
```

You can call a Python script from HSL in the same way as a native one:

```
MyPythonScript;
```

It can also be given arguments:

```
MyPythonScript( "Arg1", "Arg2");
```

> **Note**
> Returning values back from Python to HSL is not supported. You also cannot step into a Python script in the debugger.

You can call HSL from Python, using the following HSL command.

```
Result = shogunPost.HSL('hslString;')
```

The command returns a Python string containing the result of the HSL operation.

Run Python from ShogunPostCL

You can switch the ShogunPostCL command line between HSL and Python using the command `setLanguage`.

From HSL, the command is:

```
setLanguage "python";
```

From Python, the command is:

```
shogunPost.SetLanguage( "hsl" )
```