



Turbine Client User Manual (Python Scripts)

Version 2.0.0

Feb 1, 2018



Copyright (c) 2012 - 2018

Copyright Notice

Turbine Client was produced under the DOE Carbon Capture Simulation Initiative (CCSI), and is copyright (c) 2012 - 2018 by the software owners: Oak Ridge Institute for Science and Education (ORISE), Los Alamos National Security, LLC., Lawrence Livermore National Security, LLC., The Regents of the University of California, through Lawrence Berkeley National Laboratory, Battelle Memorial Institute, Pacific Northwest Division through Pacific Northwest National Laboratory, Carnegie Mellon University, West Virginia University, Boston University, the Trustees of Princeton University, The University of Texas at Austin, URS Energy & Construction, Inc., et al.. All rights reserved.

NOTICE. This Software was developed under funding from the U.S. Department of Energy and the U.S. Government consequently retains certain rights. As such, the U.S. Government has been granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable, worldwide license in the Software to reproduce, distribute copies to the public, prepare derivative works, and perform publicly and display publicly, and to permit other to do so.

License Agreement

Turbine Client Copyright (c) 2012 - 2018, by the software owners: Oak Ridge Institute for Science and Education (ORISE), Los Alamos National Security, LLC., Lawrence Livermore National Security, LLC., The Regents of the University of California, through Lawrence Berkeley National Laboratory, Battelle Memorial Institute, Pacific Northwest Division through Pacific Northwest National Laboratory, Carnegie Mellon University, West Virginia University, Boston University, the Trustees of Princeton University, The University of Texas at Austin, URS Energy & Construction, Inc., et al. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the Carbon Capture Simulation Initiative, U.S. Dept. of Energy, the National Energy Technology Laboratory, Oak Ridge Institute for Science and Education (ORISE), Los Alamos National Security, LLC., Lawrence Livermore National Security, LLC., the University of California, Lawrence

Berkeley National Laboratory, Battelle Memorial Institute, Pacific Northwest National Laboratory, Carnegie Mellon University, West Virginia University, Boston University, the Trustees of Princeton University, the University of Texas at Austin, URS Energy & Construction, Inc., nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

You are under no obligation whatsoever to provide any bug fixes, patches, or upgrades to the features, functionality or performance of the source code ("Enhancements") to anyone; however, if you choose to make your Enhancements available either publicly, or directly to Lawrence Berkeley National Laboratory, without imposing a separate written license agreement for such Enhancements, then you hereby grant the following license: a non-exclusive, royalty-free perpetual license to install, use, modify, prepare derivative works, incorporate into other computer software, distribute, and sublicense such enhancements or derivative works thereof, in binary and source code form. This material was produced under the DOE Carbon Capture Simulation Initiative

Revision Log

Version Number	Release Date	Description
Version 2.0.0	2/1/2018	Initial Open Source Release
Version 2014.10.0	10/31/2014	2014 October IAB Release

Table of Contents

1. Introduction	1
2. Prerequisites.....	1
2.1. Operating System	1
2.2. Required Software.....	1
3. Installation.....	2
3.1. UNIX.....	2
3.2. Windows.....	2
4. Configuration	2
5. Basic Operations	3
5.1. UNIX.....	3
5.2. Windows.....	3
5.3. Discover Applications	3
5.4. Application Description	4
5.5. Create Simulation.....	5
6. Advanced Scripting (Bash)	8
6.1. Check Status of Session Every Minute.....	8
6.2. Check All Sessions for Submit Jobs	8
7. Integration Testing	8
8. Support	8
9. Python API Reference	9
9.1. Application Resource	9
9.2. Consumer Resource	9
9.3. Job Resource.....	10
9.4. Session Resource.....	10
9.5. Simulation Resource	13
10. Python API Configuration File.....	14
10.1. Configuration File.....	14
11. Tests.....	15
11.1. Integration.....	15
12. Logging.....	16

To obtain support for this package, please send an email to
ccsi-support@acceleratecarboncapture.org.

1. INTRODUCTION

The Turbine Science Gateway is a web application and execution environment for running and managing scientific applications and storing and archiving the results. Clients interact with the Turbine resource-oriented architecture through a RESTful web interface, either directly over HTTP or using the higher-level Python Client Application Programming Interface (API). The Python API is designed to be easily scriptable, returning structured JSON output that can be easily consumed by other tools. Turbine is a generic solution that can be extended to process modeling and simulation applications. Currently AspenTech's Aspen Plus®, Aspen Custom Modeler® (ACM), and Microsoft® Excel® applications are supported.

The Turbine Client can be deployed on a single workstation, a cluster, and the Amazon® Web Services EC2® cloud. The server-side software is Windows®-based. The cluster and EC2 deployments allow for parallel application executions, which is shown to scale up to deployments of 400 concurrent instances. This can dramatically increase application throughput and thus decrease the time to solution.

This manual provides detailed usage instructions for using the core Turbine Client scripts with a Turbine Science Gateway. These scripts provide direct access to the Turbine Science Gateway Web API, which is used to upload and change a simulation, execute simulations, and monitor the status of job requests and components.

2. PREREQUISITES

2.1. Operating System

- Windows XP, Windows 2003 Server, Windows 7, Windows 2008 Server
- UNIX® systems (Linux®, Mac® OS)

2.2. Required Software

- Python™ 2.7
 - Other versions may work
- Python dateutil
 - Typically installed automatically

3. INSTALLATION

The Turbine Python module comes with a setup script that installs everything that a user needs to run Turbine. There are multiple options for installation. The user should select the option that matches their needs. See the README file in the distribution for the full instructions.

3.1. UNIX

If the user is installing UNIX for multiple users and has root privileges, the user should use this command:

```
% python setup.py install
```

If the user is only installing UNIX for them self (or does not have root privileges), the user should use this command:

```
% python setup.py install --user
```

3.2. Windows

On Windows a user can either modify the environment to include the Python interpreter in the environment path variable or provide the following full path to the interpreter:

```
> C:\Python2.7\python.exe setup.py install
```

4. CONFIGURATION

The Turbine Client requires a configuration file to retrieve data about the Gateway and the simulation. Because the configuration file contains some simulation specific data, it is convenient to keep a version around for each simulation that is being run. The configuration file is in Python ConfigParser format. A very simple sectioned name=value format. The configuration file is referred to as “config.txt” in the examples in the remainder of this manual (refer to Section 10 for details).

5. BASIC OPERATIONS

All of the examples in this section are written in a UNIX convention. For Windows the user needs to invoke the Python interpreter and provide the path to the command utility. All Turbine Client scripts require a configuration file to be passed as an argument (refer to Section 10 for configuration file details).

5.1. UNIX

```
% CONFIG=config.txt
% turbine_application_list $CONFIG
```

5.2. Windows

```
> C:\Python2.7\python.exe C:\Python2.7\Scripts\turbine_application_list
config.txt
```

5.3. Discover Applications

Several applications are supported by the Turbine Gateway, but not all may be enabled. Use the script below to discover which applications are available.

```
% turbine_application_list $CONFIG
Total Applications: 3
    ACM
    AspenPlus
    excel
```


5.4. Application Description

The application description also defines the set of input and output files that are staged in and out for a single execution run. None of the configured applications below stage files out.

```
% turbine_application_list -v $CONFIG
Total Applications: 3
=====
ACM
  Inputs
    1. configuration: {u'Required': True, u'Type': u'text/plain', u'Name':
u'configuration'}
    2. aspenfile: {u'Required': True, u'Type': u'text/plain', u'Name':
u'aspenfile'}
=====
AspenPlus
  Inputs
    1. configuration: {u'Required': True, u'Type': u'text/plain', u'Name':
u'configuration'}
    2. aspenfile: {u'Required': True, u'Type': u'text/plain', u'Name':
u'aspenfile'}
=====
excel
  Inputs
    1. configuration: {u'Required': True, u'Type': u'text/plain', u'Name':
u'configuration'}
    2. spreadsheet: {u'Required': True, u'Type': u'application/vnd.ms-
excel', u'Name': u'spreadsheet'}
=====
```

5.5. Create Simulation

To run an application such as ACM, the user first needs to create a new simulation resource on the Gateway that references this application and then upload the required staged-input files.

To create an ACM simulation the user needs:

- A Turbine Client configuration file (refer to Section 10)
- A Simulation Name, an informative alpha-numeric string (e.g., MEA_Adsorber_v1.2)
- The Aspen Custom Modeler file (e.g., acmf)
- A Sinter JSON Configuration file (refer to the *SimSinter User Manual*)

For this example it is assumed that the scripts directory is added to the user's path as recommended in the Installation Guide. If not, the user has to use the absolute path to the scripts in the example. It is also assumed that the simulation and configuration files have been copied to the current working directory. Below are the steps that are required for creating a simulation using the Turbine Client scripts.

All of the files in this example are available in the Turbine Client distribution, the paths are listed below:

- test/integration/simulations/Hybrid_split/Hybrid_v0.51_rev1.1_UQ_0809.acmf
- test/integration/simulations/Hybrid_split/Hybrid_v0.51_rev1.1_UQ_0809_sinter.json
- test/integration/files/Hybrid_split_testruns_noreset.txt

5.5.1. Create the Simulation Resource

The simulation resource acts as a holder for all of the simulation files which are to be staged into the working directory of each simulation execution.

```
turbine_simulation_create SIMULATION_NAME APPLICATION_NAME CONFIG
```

- **[SIMULATION_NAME]**
 - ACMHybridSplit is the simulation name that is used in following steps
- **[APPLICATION_NAME]**
 - ACM, this is a reference to the application name
- **[CONFIG]**
 - Turbine Client configuration file

```
% turbine_simulation_create ACMHybridSplit ACM $CONFIG
```

5.5.2. Put the aspenfile Simulation File in the ACMHybridSplit Simulation

```
turbine_simulation_update [options] SIMULATION_NAME FILE_NAME CONFIG
```

- [OPTIONS]
 - **-r RESOURCE, --resource=RESOURCE**
 - Specifies which staged-in file is specified by the application resource (e.g., aspenfile)
- [SIMULATION_NAME]
 - ACMHybridSplit is the simulation name, that was created in the previous step
- [FILE_NAME]
 - Path to file (e.g., acmf)
- [CONFIG]
 - Turbine Client configuration file

```
% turbine_simulation_update -r aspenfile ACMHybridSplit
Hybrid_v0.51_rev1.1_UQ_0809.acmf $CONFIG
```

5.5.3. Put the SimSinter Configuration File in the ACMHybridSplit Simulation

Add the SimSinter configuration file to the ACMHybridSplit simulation resource. To construct a SimSinter configuration file, refer to the *SimSinter User Manual*. The configuration resource is always the placeholder for the SimSinter configuration file.

```
% turbine_simulation_update -r configuration ACMHybridSplit
Hybrid_v0.51_rev1.1_UQ_0809_sinter.json $CONFIG
```

At this point the Gateway has a new simulation that can be run, ACMHybridSplit.

5.5.4. Launch Simulation Executions “Jobs”

A “session” is used to group a collection of simulation runs or “jobs”. To run a set of jobs the user first needs to create the session. This returns a string that is the session GUID (globally unique identifier). The user should keep track of the GUID as it is used to reference this set of jobs.

```
% turbine_session_create $CONFIG
"e1d318af-87fd-4721-8db7-6b616818c642"
```

Each simulation execution is referred to as a “job”. To run a set of jobs the user needs to create a JSON file that contains a list of job descriptions (refer to the *SimSinter User Manual*).

```
% turbine_session_append "e1d318af-87fd-4721-8db7-6b616818c642"
Hybrid_split_testruns_noreset.txt $CONFIG
```

Jobs wait in the create state until they are moved to submit by the user. To move the group of simulations onto the job queue:

```
%turbine_session_start "e1d318af-87fd-4721-8db7-6b616818c642" $CONFIG
```

5.5.5. Check Status of Simulation Executions “Jobs”

To check the status of all the jobs in the session:

```
% turbine_session_status "e1d318af-87fd-4721-8db7-6b616818c642" $CONFIG
{"create":0,"running":3,"setup":1,"submit":4,"pause":0,"finished":0,"error":0,"cancel":0,"success":2,"terminate":0}
```

From this the user can see that:

- 4 Runs are waiting on the submit queue
- 2 Runs have completed successfully
- 3 Runs are currently running
- 1 Run is in setup (staging-in files, initializing AspenEngine)

Checking again later, more jobs have completed:

```
% turbine_session_status "e1d318af-87fd-4721-8db7-6b616818c642" $CONFIG
{"create":0,"running":0,"setup":0,"submit":0,"pause":0,"finished":0,"error":0,"cancel":0,"success":10,"terminate":0}
```

Once the jobs completed, the user can pull the results down from the Gateway. The script `turbine_session_get_results` prints the resulting JSON file to standard out, to capture the results in a file redirect standard out.

```
% turbine_session_get_results "e1d318af-87fd-4721-8db7-6b616818c642"
$CONFIG > results.json
```

The results of the run are in the file `results.json` and a user can easily load the results in Python.

```
% python -c "import json; l = json.load(open('results.json')); print l"
```

6. ADVANCED SCRIPTING (BASH)

This section contains a few advanced examples of using the Turbine scripts to monitor the Gateway. The variable “SESSION” is set to the identifier of the session the user is interested in. The variable “CONFIG” is set to the absolute path of the Turbine Client configuration file.

6.1. Check Status of Session Every Minute

```
% I=0; while true; do I=$((I+1)) ; echo -n "${I})  " ;
turbine_session_status $SESSION $CONFIG ; sleep 60; done
```

6.2. Check All Sessions for Submit Jobs

```
% echo "=== CHECK SESSIONS FOR SUBMIT JOBS ==="; INCR=0; for SESSION in
`turbine_session_list -j $CONFIG | python -c "import sys,json; print
'\n'.join(json.load(sys.stdin))"`; do INCR=$((INCR+1)); echo -n "$INCR)
$SESSION -- "; (turbine_session_status $SESSION $CONFIG | python -c "import
sys,json; d = json.load(sys.stdin); x = (d if d['submit']>0 else None);
print x") ; done
```

7. INTEGRATION TESTING

There are several test suites that are available to verify that the Turbine Client and Gateway are working correctly.

- ws_test_suite.py
 - Basic Read operations on web resources
- simulation_test_suite.py
 - SimulationWriteTest creates a simulation on the Gateway
- acm_test_suite.py
 - Executes the ACM simulation on the Gateway
- aspenplus_test_suite.py
 - Executes the Aspen Plus simulations on the Gateway
- excel_test_suite.py
 - Executes an Excel simulation on the Gateway

To execute a test the user should be in the “test/integration” directory. The user needs to copy over the template “integration_test.cfg.in” file to “integration_test.cfg”, and then add the information from their Turbine Client configuration file (refer to Section 10).

```
% cp integration_test.cfg.in integration_test.cfg
% python suites/ws_test_suite.py
```

8. SUPPORT

To obtain support for this package, send an email to ccsi-support@acceleratecarboncapture.org.

9. PYTHON API REFERENCE

9.1. Application Resource

9.1.1. turbine_application_list

Usage: turbine_application_list [options] CONFIG_FILE

List all application resources, by default print in human readable format.

Options:

-h, --help	show this help message and exit
-p PAGE, --page=PAGE	page number
-r RPP, --rpp=RPP	results per page
-v, --verbose	verbose output
-j, --json	print results as json to stdout

9.2. Consumer Resource

9.2.1. turbine_consumer_list

Usage: turbine_consumer_list [options] CONFIG_FILE

List all Consumer resources, by default print in human readable format.

Options:

-h, --help	show this help message and exit
-v, --verbose	verbose output
-s STATUS, --status=STATUS	query on status ['up' 'down' 'error']
-j, --json	print results as json to stdout

9.2.2. turbine_consumer_log

Usage: turbine_consumer_log [options] CONFIG_FILE

Retrieves logging messages from compute resource running the specified Consumer. Log messages are printed to screen in order. This functionality is not available in all deployments.

Options:

-h, --help	show this help message and exit
------------	---------------------------------

9.3. Job Resource

9.3.1. turbine_job_script

Usage: turbine_job_script [options] CONFIG_FILE

Queries for job resources based on select criteria, by default prints JSON array of jobs.

Options:

```
-h, --help          show this help message and exit
-j SUBRESOURCE, --jobid=SUBRESOURCE
                    JOB ID
-n SIMULATION, --sim=SIMULATION
                    Simulation Name
-x STATE, --state=STATE
                    Job Status to query: ['pause', 'success', 'create',
                    'terminate', 'submit', 'running', 'warning', 'error',
                    'cancel', 'expired', 'setup']
-c CONSUMER, --consumer=CONSUMER
                    Consumer GUID to query
-b, --basic         Print Basic Information About Job(s)
-p PAGE, --page=PAGE page number
-r RPP, --rpp=RPP   results per page
-v, --verbose       verbose output
-s SESSION, --session=SESSION
                    session identifier (guid)
```

9.4. Session Resource

9.4.1. turbine_session_append

Usage: turbine_session_append SESSIONID JOBS_FILE CONFIG_FILE

Appends job descriptions to a session, prints the number of jobs added.

Options:

```
-h, --help  show this help message and exit
```

9.4.2. turbine_session_create

Usage: turbine_session_create CONFIG_FILE

Creates new session resource and prints GUID of created session.

Options:

```
-h, --help  show this help message and exit
```

9.4.3. turbine_session_delete

Usage: turbine_session_delete SESSIONID CONFIG_FILE

Delete session and all jobs it contains, prints number of jobs deleted as result of session delete.

Options:

```
-h, --help  show this help message and exit
```

9.4.4. turbine_session_get_results

Usage: turbine_session_get_results SESSIONID CONFIG_FILE

Gets the results of all the completed jobs in this session

Options:

-h, --help	show this help message and exit
-p PAGE, --page=PAGE	page number
-r RPP, --rpp=RPP	results per page
-v, --verbose	verbose output

9.4.5. turbine_session_graphs

Usage: turbine_session_graphs SESSIONID CONFIG_FILE

session resource utility, prints basic session statistics

Options:

-h, --help	show this help message and exit
-p PAGE, --page=PAGE	page number
-r RPP, --rpp=RPP	results per page
-v, --verbose	verbose output
--plot=PLOT	Plot type (see --list). Any unique prefix is OK. (default=density)
--list	List plot types and metrics.
--metric=OPT_METRIC	Metric to show in plot (see --list).

9.4.6. turbine_session_kill

Usage: turbine_session_kill SESSIONID CONFIG_FILE

Terminate jobs in session in state setup, running. Print number of jobs terminated.

Options:

-h, --help	show this help message and exit
------------	---------------------------------

9.4.7. turbine_session_list

Usage: turbine_session_list [options] CONFIG_FILE

Prints human readable listing of all session GUIDs.

Options:

-h, --help	show this help message and exit
-p PAGE, --page=PAGE	page number
-r RPP, --rpp=RPP	results per page
-v, --verbose	verbose output
-j, --json	print results as json to stdout

9.4.8. turbine_session_start

Usage: turbine_session_start SESSIONID CONFIG_FILE

Move all jobs in states pause and create to submit. Prints the number of jobs moved to state submit.

Options:

-h, --help show this help message and exit

9.4.9. turbine_session_stats

Usage: turbine_session_stats SESSIONID CONFIG_FILE

session resource utility, prints basic session statistics

Options:

-h, --help show this help message and exit

-p PAGE, --page=PAGE page number

-r RPP, --rpp=RPP results per page

-v, --verbose verbose output

9.4.10. turbine_session_status

Usage: turbine_session_status [options] SESSIONID CONFIG_FILE

session resource utility, lists all session resources

Options:

-h, --help show this help message and exit

-p PAGE, --page=PAGE page number

-r RPP, --rpp=RPP results per page

-v, --verbose verbose output

9.4.11. turbine_session_stop

Usage: turbine_session_stop SESSIONID CONFIG_FILE

Move all jobs in state submit to pause. Prints the number of jobs moved to state pause.

Options:

-h, --help show this help message and exit

9.5. Simulation Resource

9.5.1. turbine_simulation_create

Usage: turbine_simulation_create [options] SIMULATION_NAME APPLICATION_NAME CONFIG_FILE

Create an empty Simulation Resource

Options:

-h, --help show this help message and exit

9.5.2. turbine_simulation_get

Usage: turbine_simulation_get [options] SIMULATION_NAME CONFIG_FILE

Retrieves the Simulation resource, by default prints as JSON.

Options:

-h, --help show this help message and exit

-r RESOURCE, --resource=RESOURCE

return only the specified input subresource

-s, --save Save the resource as a file

9.5.3. turbine_simulation_list

Usage: turbine_simulation_list [options] CONFIG_FILE

Retrieves list of all simulations, by default prints in human readable format.

Options:

-h, --help show this help message and exit

-p PAGE, --page=PAGE page number

-r RPP, --rpp=RPP results per page

-v, --verbose verbose output

-j, --json print results as json to stdout

9.5.4. turbine_simulation_update

Usage: turbine_simulation_update [options] SIMULATION_NAME FILE_NAME CONFIG_FILE

Update simulation by essentially doing a PUT with the specified file to the resource or optionally sub-resource.

Options:

-h, --help show this help message and exit

-r RESOURCE, --resource=RESOURCE

select the staged input file name (matches Application)

10. PYTHON API CONFIGURATION FILE

10.1. Configuration File

The Turbine Python script configuration file is a parameter to every Turbine Python script, it is in “ConfigParser” format. The file contains a separate section for each web resource Application, Consumer, Job, Session, and Simulation. These sections each contain a “url” key, which specifies the URL of the web resource. The “Logging” section is optional. In this section a user can specify a logging configuration file. There is a sample “logging.conf” included in the distribution. The “Authentication” section is for specifying user’s credentials.

```
[Logging]
fileConfig=logging.conf
[Consumer]
url=https://SERVER:PORT/Turbine/consumer/
[Session]
url=https://localhost:8080/Turbine/session/
[Job]
url=https://localhost:8080/Turbine/job/
[Simulation]
url=https://localhost:8080/Turbine/simulation/
[Application]
url=https://localhost:8080/Turbine/application/
[Authentication]
username=USERNAME
password=PASSWORD
[Security]
TrustedCertificateAuthorities=CACERTS
```

- **SERVER: PORT**
 - The IP Address/FQDN and port of the Gateway (for example: *localhost:8080*)
- **USERNAME**
 - Turbine Gateway account username
- **PASSWORD**
 - Turbine Gateway account password
- **CACERTS**
 - If the user provides the “Security” section the Client validates the server’s certificate against the provided certificate authority list
 - This is a text file containing CA certificates
 - To turn validation off, remove the “Security” section

11. TESTS

11.1. Integration

A template integration test configuration file is located in the “test/integration” directory, and is named “integration_test.cfg.in”. The file contains all of the sections contained in the Python API Configuration File (Section 9), and these sections should match. There are additional sections for several of the tests, but these require no modification. Not all of these tests may be relevant for the Turbine Gateway Deployment the user is using, namely some of the applications may not be supported (e.g., ACM, Aspen Plus, Excel, gPROMS).

- **WS Tests Suite**

```
% python suites/ws_test_suite.py
....
```

```
-----
Ran 4 tests in 0.405s
```

```
OK
```

- **Simulation**

```
% python suites/simulation_test_suite.py
.
```

```
-----
Ran 1 test in 1.474s
```

```
OK
```

- **AspenTech ACM**

```
% python suites/acm_test_suite.py
.
```

```
-----
Ran 1 test in 132.458s
```

```
OK
```

- **AspenTech Aspen Plus**

```
% python suites/aspenplus_test_suite.py
.
```

```
-----
Ran 1 test in 115.863s
```

```
OK
```

12. LOGGING

The Turbine Client uses the standard Python logging module. To use logging for Turbine Client, add a python logging file with the name “logging.conf” into the working directory. Below is an example file.

```
[loggers]
keys=root,commands,session

[handlers]
keys=consoleHandler

[formatters]
keys=simpleFormatter

[logger_root]
level=DEBUG
handlers=consoleHandler

[logger_commands]
level=DEBUG
handlers=consoleHandler
qualname=turbine.commands
propagate=0

[logger_session]
level=DEBUG
handlers=consoleHandler
qualname=turbine.commands.turbine_session_script
propagate=0

[handler_consoleHandler]
class=StreamHandler
formatter=simpleFormatter
args=(sys.stdout,)

[formatter_simpleFormatter]
format=%(asctime)s - %(name)s - %(levelname)s - %(message)s
datefmt=
```