

# PyWBEM

## Python WBEM CIM/XML Client

### Rapid Overview

[k.schopmeyerwork@gmail.com](mailto:k.schopmeyerwork@gmail.com)

Last update 3 February 2021

Version 0.9, 1 Dec 2016

Version 1.0, 5 Dec 2016

Version 2.0 6 Aug 2018 – Update to current PyWBEM version

Version 2.0.7, Aug 2018- Minor edits

Version 3.0 Update to pywbem version 1.

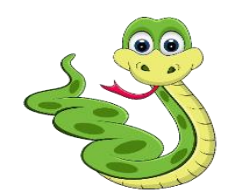
Version 3.1, 3 February 2021 minor cleanup and update



# What is the PyWBEM project?

PyWBEM is a GitHub multi-repository project written in Python that includes several WBEM/CIM components to support the DMTF WBEM/CIM and SNIA SMI-S specifications. It includes:

- **pywbem** – A client for the WBEM infrastructure
- **pywbemtools** – Client-side tools developed to utilize PyWBEM to communicate with WBEM Servers. The core tool is **pywbemcli**, a command line WBEM client
- Other WBEM support components.

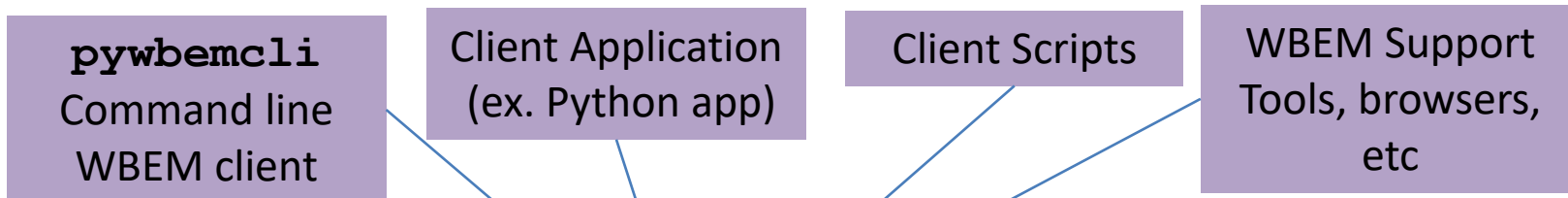


# PyWBEM characteristics

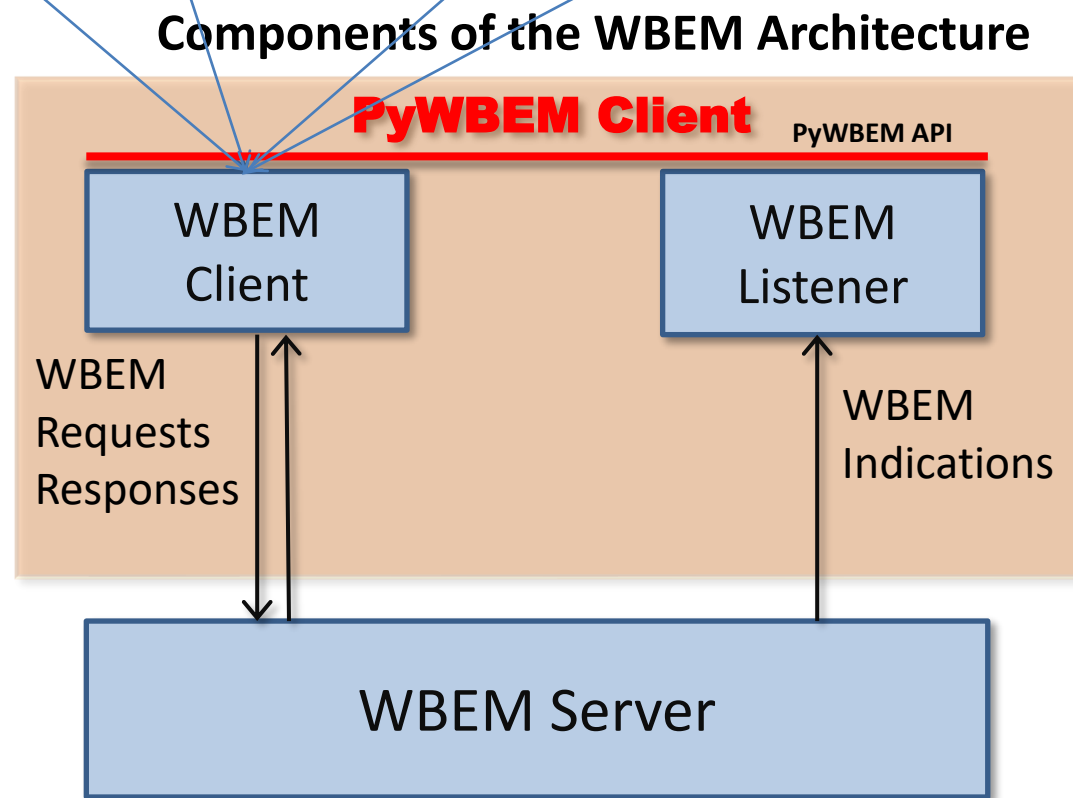
- Compliant with the DMTF CIM/WBEM and SNIA SMI-S specifications
- Open source in GitHub <http://pywbem/pywbem> and well documented on Read The Docs <https://pywbem.readthedocs.io/en/latest/index.html>
- Available as a Python package on GitHub, PyPi, and Linux distributions
- Supports Python 2.7 and Python 3
- Runs on windows (native, Cygwin, etc.), OS-X, Linux
- Simple installation with Python pip

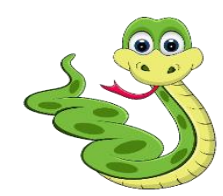


# What is a PyWBEM client?



Python client for WBEM servers using the DMTF CIM-XML protocol



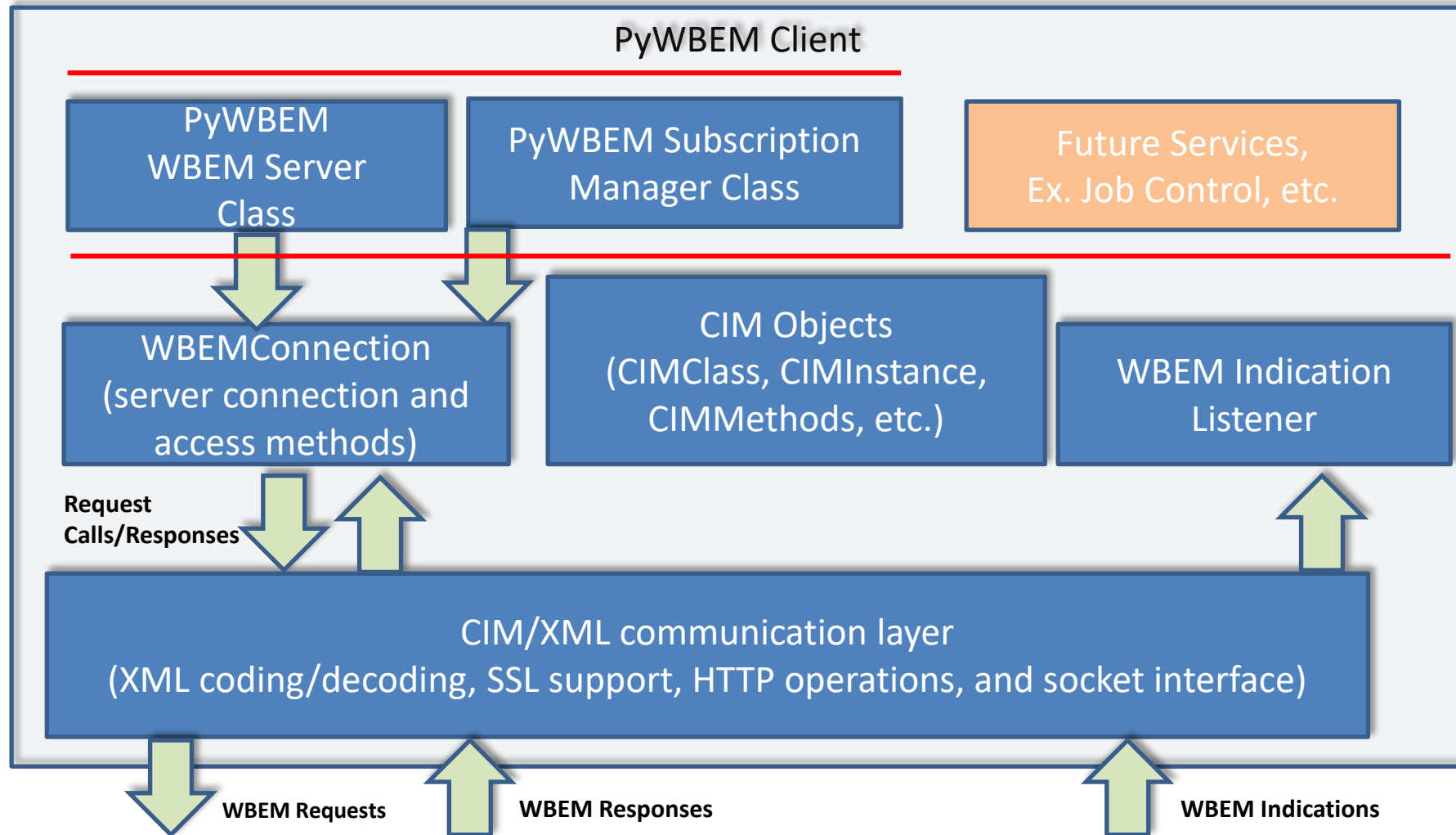


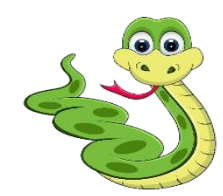
# PyWBEM Client: Overview

- Pure Python code:
  - Python versions 2.7, 3.4, 3.5 – 3.9
- Supports DMTF CIM-XML protocol
  - WBEM Client library with a Pythonic API
  - Python classes for all CIM model defined objects (CIMClass, CIMInstance, etc.)
  - Indication listener/subscription manager
  - Server class to access common objects (ex. Registered Profiles, Namespaces) in WBEM server
  - Includes support for SSL
  - CIM-XML protocol for communication with WBEM server
- Well tested, well documented
- Compliant with DMTF WBEM Specification and SNIA SMI-S specification
- Utilities:
  - MOF compiler
  - `Pywbem_mock` – Mock of a WBEM server that allows testing `pywbem` and `pywbemtools` with no WBEM server
  - Test tools
- Open source, LGPL 2.1 license
  - Available on GitHub and Python PyPi: <https://github.com/pywbem/pywbem>



# PyWBEM Client Architecture





# The PyWBEM Client APIs

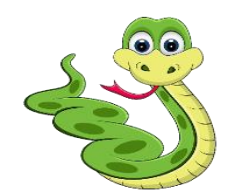
- Construction and manipulation of Python CIM objects
  - CIMClass, CIMInstance, CIMMethod, CIMProperty, etc.
- WBEMConnection API
  - Define connections to WBEM server
  - Execute WBEM operations against the WBEM server
    - Get, create, enumerate, etc. of CIM objects on WBEM server
    - Request execution of CIM Methods on WBEM server
- Higher level WBEM client functions
  - Indication subscription management
    - Create, delete, subscriptions for indications on WBEM server
  - WBEM server discovery
    - Discover basic characteristics of WBEM Servers
      - CIM Namespaces, basic server information, Registered Profiles, etc.
- Indication Listener API
  - Listen for indications from the WBEM indication exporter (i.e. WBEM server that sends indications)



# WBEMConnection, Client API

- Defines connection and request/response operations on CIM Objects
- CIMObjects are:
  - CIMClasses
  - CIMInstances
  - CIMQualifierDeclarations
  - CIMMethods
- Operations:
  - Get, enumerate, create, delete, modify CIMObjects in WBEM server
  - Get Associations
  - Invoke Methods defined in the model
  - Query model resources in WBEM server

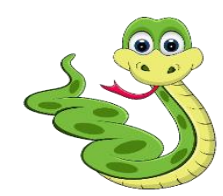




# A simple PyWBEM code example

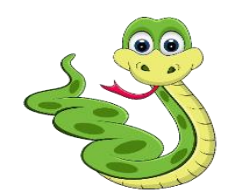
```
import pywbem
# Global variables used by all examples:
server = 'http://localhost'
username = 'user'
password = 'password'
namespace = 'root/cimv2'
classname = 'CIM_ComputerSystem'
max_obj_cnt = 100
conn = pywbem.WBEMConnection(server, (username, password),
                             default_namespace=namespace,
                             no_verification=True)

try:
    inst_iterator = conn.IterEnumerateInstances(classname, MaxObjectCount=max_obj_cnt)
    for inst in inst_iterator:
        print('path=%s' % inst.path)
        print(inst.tomof())
except pywbem.Error as exc:
    print('Operation failed: %s' % exc)
```



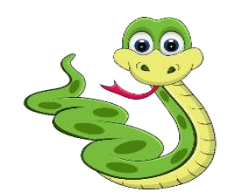
# PyWBEM Developer Aids

- MOF Compiler
  - Compiles DMTF MOF into repositories as CIM objects
- Usage support
  - Operation statistics
    - Statistics on execution time of WBEM Server operations
  - Operation Logging
    - Python logging interface for operation requests/responses
  - Operations recording
    - Record details of operations for tests generation
- Testing Support
  - PyWBEM WBEM Server mock/simulator
    - Simulates a WBEM Server within the pywbem client to allow testing without a running WBEM Server
  - Pywbemcli (see the pywbemtools repository)
    - Interactive REPL command line tool for accessing WBEM servers



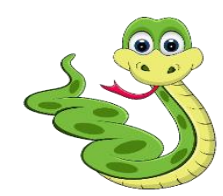
# PyWBEM Installation

- pip package installation
  - Within your Python environment get from pip
    - `pip install pywbem`
- Install complete GitHub package
  - `git clone https://github.com/pywbem/pywbem`
  - Installation instructions are part of the documentation downloaded



# PyWBEMTools characteristics

- Command line client (pywbemcli) that provides commands to access and modify a WBEM server
- Open source in GitHub <http://pywbem/pywbemtools> and well documented on Read The Docs  
<https://pywbemtools.readthedocs.io/en/latest/index.html>
- Available as a Python package on GitHub, PyPi.
- Supported on Python 2 and Python 3



# Pywbemcli commands

- **pywbemcli** includes multiple commands in command groups. The groups include: class, qualifier, instance, connection, server, profile
- Commands to enumerate/get/create/delete CIM qualifier declarations, CIM classes, and CIM instances. These are command line implementations of the WBEM operations
- Higher level commands
  - Extensions for CIM Object visualization (class trees, association trees, etc.)
  - **connection** – Manage a persistent definition of a wbem server
  - **server** – Inspect a wbem server
    - Namespaces, and other information about a WBEM server
  - **profile**
    - Inspect WBEM server profiles
- Multiple output display formats (MOF, tables, trees to show relationships, etc.)
- Common usage support:
  - Interactive mode (multiple commands within pywbemcli shell)
  - Autocompletion of command syntax and some command variables
  - Extensive help with all commands through – **help** option

# Command line examples

- Get the class CIM\_ManagedElement from the server

```
< Pywbemcli -s http://localhost class get
TST_Person
class TST_Person {

    [Key ( true ),
     Description ( "This is key prop" )]
    string name;

    string extraProperty = "defaultvalue";

    [ValueMap { "1", "2" },
     Values { "female", "male" }]
    uint16 gender;

    [ValueMap { "1", "2" },
     Values { "books", "movies" }]
    uint16 likes[];

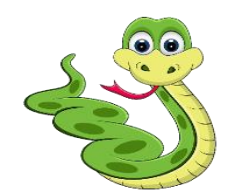
};
```

- Class Tree

```
< Pywbemcli -m mockassoc class tree
root
+-- TST_FamilyCollection
+-- TST_Lineage
+-- TST_MemberOfFamilyCollection
+-- TST_Person
    +-- TST_Personsub
```

- Association Tree

```
< Pywbemcli -m mock assoc instance shrub TST_Person.?
Pick Instance name to process
0: root/cimv2:TST_Person.name="Gabi"
1: root/cimv2:TST_Person.name="Mike"
2: root/cimv2:TST_Person.name="Saara"
Input integer between 0 and 7 or Ctrl-C to exit selection: 0
TST_Person.name="Gabi"
+-- child(Role)
| +-- TST_Lineage(AssocClass)
|   +-- parent(ResultRole)
|     +-- TST_Person(ResultClass)(1 insts)
|       +-- /:TST_Person.name="Mike"
+-- member(Role)
    +-- TST_MemberOfFamilyCollection(AssocClass)
        +-- family(ResultRole)
            +-- TST_FamilyCollection(ResultClass)(1 insts)
                +-- /:TST_FamilyCollection.name="family1"
```



# Pywbemtools installation

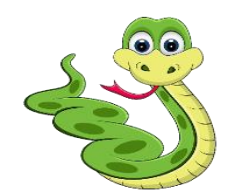
- pip package installation
  - Within your Python environment get from pip
    - Setup a python virtual environment (not required but helps)
    - `pip install pywbemtools` (Installs pywbem and pywbemtools)
- Install complete GitHub package
  - Setup a python virtual environment (not required but helps)
  - `git clone https://github.com/pywbem/pywbemtools.git`
  - Installation instructions are part of the documentation downloaded



# Status

- Active Development
  - Pywbem: release 1.1, Released 31 October 2020
  - Pywbemtools: release 0.8, released 13 October 2020
  - Next release ~ Q1 2021 (pywbem 1.2 and pywbemtools 0.9)
- Extensively Tested:
  - Mock server implementations in continuous integration
  - OpenPegasus WEB server before each release
  - A variety of SMI servers as part of the SNIA SM Lab/Plugfests





# Resources and more information

- PyWBEM Project – Project encompassing PyWBEM and tools
  - Pywbem Project github: <https://github.com/pywbem>
  - PyWBEM Client
    - PyWBEM Client github repository : <https://github.com/pywbem/pywbem>
    - PyWBEM Client Documentation
      - <https://pywbem.github.io> - General documentation
      - <https://pywbem.readthedocs.io/en/latest/index.html> - Detailed API Reference, installation, tutorial, development information, and change log
      - <https://pywbem.readthedocs.io/en/latest/tutorial.html> - Jupyter notebooks with examples of pywbem client usage
  - PyWBEM Tools
    - Github repository: <https://github.com/pywbem/pywbemtools>
    - Documentation: <https://pywbemtools.readthedocs.io/en/latest/index.html>
- Other Resources
  - OpenPegasus
    - <https://collaboration.opengroup.org/pegasus/>
  - SNIA pywbem page
    - <https://www.snia.org/pywbem>



# CIM/WBEM Specification References

- DMTF Specifications:
  - See: [https://www.dmtf.org/standards/published\\_documents](https://www.dmtf.org/standards/published_documents)
  - Common Information Model (CIM) Schema releases
    - [CIM \(Common Information model specifications and schemas\)](#)
  - [Common Information Model, DSP0004](#)
  - [CIM Operations over XML - DSP0200](#)
  - [Representation of CIM in XML - DSP0201](#)
  - [CIM Query Language Specification –DSP0202](#)
  - [Filter Query Language\(FQL\) – DSP0212](#)
- SMI Specifications:
  - SNIA SMI-S specification
    - [http://www.snia.org/tech\\_activities/standards/curr\\_standards/smi](http://www.snia.org/tech_activities/standards/curr_standards/smi)