PyCIM Documentation

Release 15.13.2

Richard Lincoln

July 02, 2011

CONTENTS

CHAPTER

ONE

INTRODUCTION

PyCIM is a Python implementation of the IEC **Common Information Model**.

Current features include:

- Support for **IEC 61970** 15v13 and **IEC 61968** 11v05,
- Legacy support for IEC 61970 14v15 and IEC 61968 10v31,
- Profiles of the CIM, including: - Common Power Systems Model (**CPSM**) (CIM v14) - Common Distribution Power System Model (**CDPSM**) (CIM v14 and v15) - European Network of Transmission System Operators for Electricity (**ENTSO-E**) (CIM v14),
- Class and attribute documentation integrated as Python doc-strings,
- Transparent bi-directional reference handling using Python properties,
- CIM RDF/XML **parsing** and **serialisation** according to IEC 61970-552.

CHAPTER TWO

LICENSE AND COPYRIGHT

CHAPTER

THREE

INSTALLATION

PyCIM has no dependencies beyond Python 2.5 or later. It can be easy_installed using setuptools:

```
$ easy_install PyCIM
```

Alternatively, download and unpack the tarball and install:

```
$ tar zxf PyCIM-XX.XX.tar.gz
$ python setup.py install
```

On UNIX systems, use sudo for the latter command if you need to install the scripts to a directory that requires root privileges:

```
$ sudo python setup.py install
```

The development Git repository can be cloned from GitHub:

```
$ git clone https://github.com/rwl/PyCIM.git
```

CHAPTER

FOUR

USAGE

To use PyCIM with the Python interpreter, IPython is recommended. For example, to instantiate a ConnecticityNode:

```
In [1]: from CIM14.IEC61970.Core import ConnectivityNode

In [2]: node = ConnectivityNode(name='Node 1')
```

To associate the node with a Terminal:

```
In [3]: from CIM14.IEC61970.Core import Terminal

In [4]: t = Terminal(name='T1', ConnectivityNode=node)

In [5]: node.Terminals[0].name
Out[5]: 'T1'
```

To add a Terminal to a ConnectivityNode:

```
In [6]: t2 = Terminal()

In [7]: node.addTerminals(t2)

In [8]: t2.ConnectivityNode.name
Out[8]: 'Node 1'
```

To view the docstring for an attribute:

```
In [9]: t.connected?
Type:     bool
Base Class: <type 'bool'>
String Form:    False
Namespace:  Interactive
Docstring:
    bool(x) -> bool

    Returns True when the argument x is true, False otherwise.
    The builtins True and False are the only two instances of the class bool.
    The class bool is a subclass of the class int, and cannot be subclassed.
```

CHAPTER

FIVE

INDICES AND TABLES

- *Index*
- *Module Index*
- *Search Page*