

PowerMeter daemon settings

Generated by Doxygen 1.7.6.1

Tue Apr 10 2012 15:55:44

Contents

1	Data Structure Documentation	1
1.1	AttachedDevice Class Reference	1
1.1.1	Detailed Description	1
1.1.2	Constructor & Destructor Documentation	2
1.1.2.1	__init__	2
1.1.3	Field Documentation	2
1.1.3.1	computer	2
1.2	Computer Class Reference	2
1.2.1	Detailed Description	3
1.2.2	Constructor & Destructor Documentation	3
1.2.2.1	__init__	3
1.2.3	Member Function Documentation	3
1.2.3.1	__repr__	3
1.2.3.2	add	3
1.2.4	Field Documentation	3
1.2.4.1	devices	4
1.2.4.2	ip	4
1.2.4.3	name	4
1.3	DC2Device Class Reference	4
1.3.1	Detailed Description	4
1.3.2	Constructor & Destructor Documentation	4
1.3.2.1	__init__	5
1.4	DCDevice Class Reference	5
1.4.1	Detailed Description	5
1.4.2	Constructor & Destructor Documentation	6

1.4.2.1	<code>__init__</code>	6
1.5	Device Class Reference	6
1.5.1	Detailed Description	7
1.5.2	Constructor & Destructor Documentation	7
1.5.2.1	<code>__init__</code>	7
1.5.3	Member Function Documentation	7
1.5.3.1	<code>__repr__</code>	7
1.5.3.2	<code>add_line</code>	7
1.5.4	Field Documentation	8
1.5.4.1	<code>lines</code>	8
1.5.4.2	<code>max_frequency</code>	8
1.5.4.3	<code>name</code>	8
1.5.4.4	<code>url</code>	8
1.6	Line Class Reference	8
1.6.1	Detailed Description	9
1.6.2	Constructor & Destructor Documentation	9
1.6.2.1	<code>__init__</code>	9
1.6.3	Member Function Documentation	9
1.6.3.1	<code>__repr__</code>	9
1.6.4	Field Documentation	9
1.6.4.1	<code>description</code>	9
1.6.4.2	<code>name</code>	9
1.6.4.3	<code>voltage</code>	10
1.7	NIDevice Class Reference	10
1.7.1	Detailed Description	10
1.7.2	Constructor & Destructor Documentation	10
1.7.2.1	<code>__init__</code>	10
1.8	PDUDevice Class Reference	11
1.8.1	Detailed Description	11
1.8.2	Constructor & Destructor Documentation	11
1.8.2.1	<code>__init__</code>	11
1.8.3	Member Function Documentation	12
1.8.3.1	<code>add_line</code>	12
1.9	PDULine Class Reference	12

1.9.1	Detailed Description	13
1.9.2	Constructor & Destructor Documentation	13
1.9.2.1	__init__	13
1.9.3	Member Function Documentation	13
1.9.3.1	__repr__	13
1.9.4	Field Documentation	13
1.9.4.1	computer	13
1.9.4.2	description	13
1.9.4.3	name	13
1.9.4.4	voltage	13
1.10	WattsUpDevice Class Reference	14
1.10.1	Detailed Description	14
1.10.2	Constructor & Destructor Documentation	14
1.10.2.1	__init__	14
1.10.3	Member Function Documentation	15
1.10.3.1	add_line	15
2	File Documentation	17
2.1	/home/barrachi/datos/aplicaciones/powermeter/settings-use-example.py File Reference	17
2.2	/home/barrachi/datos/aplicaciones/powermeter/settings-use-example.py	17
2.3	/home/barrachi/datos/aplicaciones/powermeter/settings.py File Reference	18
2.4	/home/barrachi/datos/aplicaciones/powermeter/settings.py	19
2.5	/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py File Reference	20
2.6	/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py . . .	20

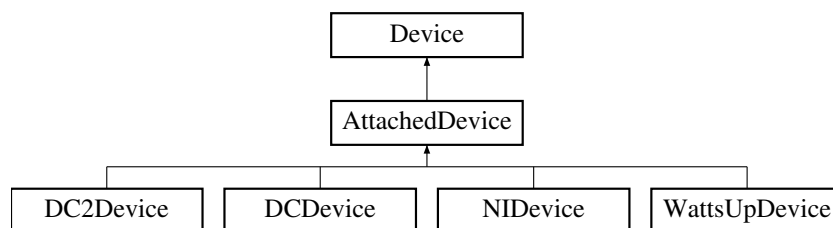
Chapter 1

Data Structure Documentation

1.1 AttachedDevice Class Reference

A device attached to a computer description.

Inheritance diagram for AttachedDevice:



Public Member Functions

- `def __init__`

Creates device attached to a computer description and adds it to the devices dictionary.

Data Fields

- `computer`

1.1.1 Detailed Description

A device attached to a computer description.

Definition at line 148 of file `settings_classes.py`.

1.1.2 Constructor & Destructor Documentation

1.1.2.1 `def __init__(self, name, computer, url, max_frequency)`

Creates device attached to a computer description and adds it to the devices dictionary.

Before adding the given device description to the devices dictionary, it checks that the name of the new device has not been used by a previously added device.

Parameters

in	<i>name</i>	The device name (used for identification, must be unique)
in	<i>computer</i>	The computer the device is attached to
in	<i>url</i>	The url of this device
in	<i>max_ - frequency</i>	The maximum sample frequency of the device

Reimplemented in [WattsUpDevice](#), [NIDevice](#), [DC2Device](#), and [DCDevice](#).

Definition at line 162 of file [settings_classes.py](#).

1.1.3 Field Documentation

1.1.3.1 `computer`

Definition at line 162 of file [settings_classes.py](#).

The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.2 Computer Class Reference

A computer description.

Public Member Functions

- `def __init__`
Creates a computer description and adds it to the computers dictionary.
- `def __repr__`
Returns a string representation for this computer.
- `def add`
Adds a device description to the computer.

Data Fields

- [name](#)
- [ip](#)
- [devices](#)

1.2.1 Detailed Description

A computer description.

Definition at line 20 of file [settings_classes.py](#).

1.2.2 Constructor & Destructor Documentation

1.2.2.1 `def __init__(self, name, ip)`

Creates a computer description and adds it to the computers dictionary.

Parameters

in	<i>name</i>	The name of the computer
in	<i>ip</i>	The IP address of the computer

Definition at line 28 of file [settings_classes.py](#).

1.2.3 Member Function Documentation

1.2.3.1 `def __repr__(self)`

Returns a string representation for this computer.

Definition at line 36 of file [settings_classes.py](#).

1.2.3.2 `def add (self, device)`

Adds a device description to the computer.

Parameters

in	<i>device</i>	A device description object
----	---------------	-----------------------------

Definition at line 43 of file [settings_classes.py](#).

1.2.4 Field Documentation

1.2.4.1 devices

Definition at line 28 of file [settings_classes.py](#).

1.2.4.2 ip

Definition at line 28 of file [settings_classes.py](#).

1.2.4.3 name

Definition at line 28 of file [settings_classes.py](#).

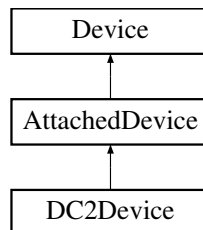
The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.3 DC2Device Class Reference

A DC2Meter device description.

Inheritance diagram for DC2Device:



Public Member Functions

- `def __init__`

Creates a DC2Meter device description and adds it to the devices dictionary.

1.3.1 Detailed Description

A DC2Meter device description.

Definition at line 191 of file [settings_classes.py](#).

1.3.2 Constructor & Destructor Documentation

1.3.2.1 `def __init__(self, name, computer, url, max_frequency)`

Creates a DC2Meter device description and adds it to the devices dictionary.

Parameters

in	<i>name</i>	The device name (used for identification, must be unique)
in	<i>computer</i>	The computer the device is attached to
in	<i>url</i>	The url of this device
in	<i>max_ - frequency</i>	The maximum sample frequency of the device

Reimplemented from [AttachedDevice](#).

Definition at line 201 of file [settings_classes.py](#).

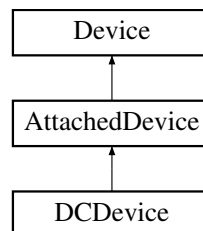
The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.4 DCDevice Class Reference

A DCMeter device description.

Inheritance diagram for DCDevice:



Public Member Functions

- `def __init__`

Creates a DC2Meter device description and adds it to the devices dictionary.

1.4.1 Detailed Description

A DCMeter device description.

Definition at line 174 of file [settings_classes.py](#).

1.4.2 Constructor & Destructor Documentation

1.4.2.1 `def __init__(self, name, computer, url, max_frequency)`

Creates a DC2Meter device description and adds it to the devices dictionary.

Parameters

in	<i>name</i>	The device name (used for identification, must be unique)
in	<i>computer</i>	The computer the device is attached to
in	<i>url</i>	The url of this device
in	<i>max_ - frequency</i>	The maximum sample frequency of the device

Reimplemented from [AttachedDevice](#).

Definition at line 184 of file [settings_classes.py](#).

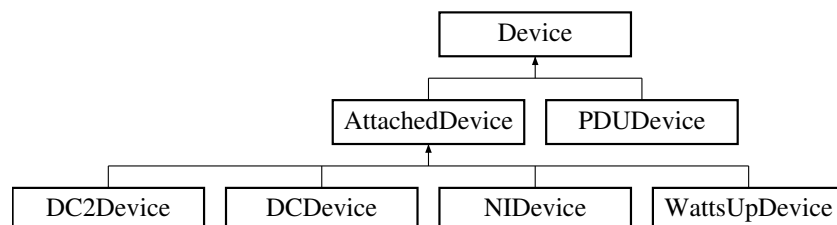
The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.5 Device Class Reference

A device description.

Inheritance diagram for Device:



Public Member Functions

- `def __init__`
Creates a device description and adds it to the devices dictionary.
- `def __repr__`
Returns a string representation for this device.
- `def add_line`
Adds a line description to the device.

Data Fields

- [name](#)
- [url](#)
- [max_frequency](#)
- [lines](#)

1.5.1 Detailed Description

A device description.

Definition at line 100 of file [settings_classes.py](#).

1.5.2 Constructor & Destructor Documentation

1.5.2.1 `def __init__(self, name, url, max_frequency)`

Creates a device description and adds it to the devices dictionary.

Before adding the given device description to the devices dictionary, it checks that the name of the new device has not been used by a previously added device.

Parameters

<code>in</code>	<code>name</code>	The device name (used for identification, must be unique)
<code>in</code>	<code>url</code>	The url of this device
<code>in</code>	<code>max_ - frequency</code>	The maximum sample frequency of the device

Reimplemented in [PDUDevice](#).

Definition at line 113 of file [settings_classes.py](#).

1.5.3 Member Function Documentation

1.5.3.1 `def __repr__(self)`

Returns a string representation for this device.

Definition at line 125 of file [settings_classes.py](#).

1.5.3.2 `def add_line(self, name, voltage, description)`

Adds a line description to the device.

Before adding the given line description to the device, it checks that the name of the new line has not been used by a previously added line.

Parameters

in	<i>name</i>	The line name (used for identification)
in	<i>voltage</i>	The line voltage
in	<i>description</i>	A text description of the line

Reimplemented in [WattsUpDevice](#).

Definition at line 138 of file [settings_classes.py](#).

1.5.4 Field Documentation**1.5.4.1 lines**

Definition at line 113 of file [settings_classes.py](#).

1.5.4.2 max_frequency

Definition at line 113 of file [settings_classes.py](#).

1.5.4.3 name

Definition at line 113 of file [settings_classes.py](#).

1.5.4.4 url

Definition at line 113 of file [settings_classes.py](#).

The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.6 Line Class Reference

A line description.

Public Member Functions

- [def __init__](#)
Creates a line description.
- [def __repr__](#)
Returns a string representation for this line.

Data Fields

- [name](#)
- [voltage](#)
- [description](#)

1.6.1 Detailed Description

A line description.

Definition at line 53 of file [settings_classes.py](#).

1.6.2 Constructor & Destructor Documentation

1.6.2.1 `def __init__(self, name, voltage, description)`

Creates a line description.

Parameters

in	<i>name</i>	The line name (used for identification)
in	<i>voltage</i>	The line voltage
in	<i>description</i>	A text description of the line

Definition at line 61 of file [settings_classes.py](#).

1.6.3 Member Function Documentation

1.6.3.1 `def __repr__(self)`

Returns a string representation for this line.

Definition at line 67 of file [settings_classes.py](#).

1.6.4 Field Documentation

1.6.4.1 `description`

Definition at line 61 of file [settings_classes.py](#).

1.6.4.2 `name`

Definition at line 61 of file [settings_classes.py](#).

1.6.4.3 voltage

Definition at line 61 of file [settings_classes.py](#).

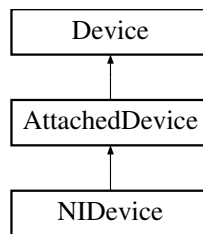
The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.7 NIDevice Class Reference

A National Instruments device description.

Inheritance diagram for NIDevice:



Public Member Functions

- [def __init__](#)

Creates a National Instruments device description and adds it to the devices dictionary.

1.7.1 Detailed Description

A National Instruments device description.

Definition at line 208 of file [settings_classes.py](#).

1.7.2 Constructor & Destructor Documentation

1.7.2.1 `def __init__(self, name, computer, url, max_frequency)`

Creates a National Instruments device description and adds it to the devices dictionary.

Parameters

in	<i>name</i>	The device name (used for identification, must be unique)
in	<i>computer</i>	The computer the device is attached to
in	<i>url</i>	The url of this device
in	<i>max_-frequency</i>	The maximum sample frequency of the device

Reimplemented from [AttachedDevice](#).

Definition at line 218 of file [settings_classes.py](#).

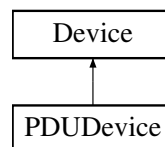
The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.8 PDUDevice Class Reference

A PDU device description.

Inheritance diagram for PDUDevice:



Public Member Functions

- `def __init__`
Creates a PDU device description and adds it to the devices dictionary.
- `def add_line`
Adds a pdu line description to the device.

1.8.1 Detailed Description

A PDU device description.

Definition at line 255 of file [settings_classes.py](#).

1.8.2 Constructor & Destructor Documentation

1.8.2.1 `def __init__(self, name, url, max_frequency)`

Creates a PDU device description and adds it to the devices dictionary.

Parameters

in	<i>name</i>	The device name (used for identification, must be unique)
in	<i>url</i>	The url of this device
in	<i>max-frequency</i>	The maximum sample frequency of the device

Reimplemented from [Device](#).

Definition at line 264 of file [settings_classes.py](#).

1.8.3 Member Function Documentation

1.8.3.1 `def add_line (self, name, computer, voltage, description = " ")`

Adds a pdu line description to the device.

Before adding the given line description to the device, it checks that the name of the new line has not been used by a previously added line.

Parameters

<code>in</code>	<code>name</code>	The line name (used for identification)
<code>in</code>	<code>computer</code>	The computer the line is attached to
<code>in</code>	<code>voltage</code>	The line voltage
<code>in</code>	<code>description</code>	An optional text description of the line

Definition at line 279 of file [settings_classes.py](#).

The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

1.9 PDULine Class Reference

A PDU line description.

Public Member Functions

- `def __init__`
Creates a PDU line description.
- `def __repr__`
Returns a string representation for this line.

Data Fields

- [name](#)
- [computer](#)
- [voltage](#)
- [description](#)

1.9.1 Detailed Description

A PDU line description.

Definition at line 74 of file [settings_classes.py](#).

1.9.2 Constructor & Destructor Documentation

1.9.2.1 `def __init__(self, name, computer, voltage, description = " ")`

Creates a PDU line description.

Parameters

in	<i>name</i>	The line name (used for identification)
in	<i>computer</i>	The computer this PDU line is attached to
in	<i>voltage</i>	The line voltage
in	<i>description</i>	An optional text description of the line

Definition at line 83 of file [settings_classes.py](#).

1.9.3 Member Function Documentation

1.9.3.1 `def __repr__(self)`

Returns a string representation for this line.

Definition at line 93 of file [settings_classes.py](#).

1.9.4 Field Documentation

1.9.4.1 computer

Definition at line 83 of file [settings_classes.py](#).

1.9.4.2 description

Definition at line 83 of file [settings_classes.py](#).

1.9.4.3 name

Definition at line 83 of file [settings_classes.py](#).

1.9.4.4 voltage

Definition at line 83 of file [settings_classes.py](#).

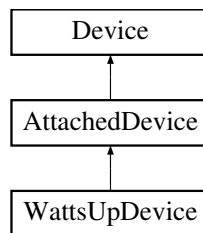
The documentation for this class was generated from the following file:

- /home/barrachi/datos/aplicaciones/powermeter/settings_classes.py

1.10 WattsUpDevice Class Reference

A WattsUp device description.

Inheritance diagram for WattsUpDevice:



Public Member Functions

- `def __init__`
Creates a WattsUp device description and adds it to the devices dictionary.
- `def add_line`
Fake adding of line description to the device.

1.10.1 Detailed Description

A WattsUp device description.

Definition at line 225 of file [settings_classes.py](#).

1.10.2 Constructor & Destructor Documentation

1.10.2.1 `def __init__(self, name, computer, url, max_frequency)`

Creates a WattsUp device description and adds it to the devices dictionary.

Parameters

in	<i>name</i>	The device name (used for identification, must be unique)
in	<i>computer</i>	The computer the device is attached to
in	<i>url</i>	The url of this device
in	<i>max_ frequency</i>	The maximum sample frequency of the device

Reimplemented from [AttachedDevice](#).

Definition at line 235 of file [settings_classes.py](#).

1.10.3 Member Function Documentation

1.10.3.1 `def add_line(self, name, voltage, description)`

Fake adding of line description to the device.

The WattsUp [Device](#) does not have lines. This method avoids the base class method silently been called.

Parameters

<i>in</i>	<i>name</i>	The line name (used for identification)
<i>in</i>	<i>voltage</i>	The line voltage
<i>in</i>	<i>description</i>	A text description of the line

Reimplemented from [Device](#).

Definition at line 247 of file [settings_classes.py](#).

The documentation for this class was generated from the following file:

- [/home/barrachi/datos/aplicaciones/powermeter/settings_classes.py](#)

Chapter 2

File Documentation

2.1 /home/barrachi/datos/aplicaciones/powermeter/settings-use-example.py File Reference

Functions

- def **header**
- def **main**

Shows the information given by the settings file.

2.2 /home/barrachi/datos/aplicaciones/powermeter/settings-use-example.py

```
00001 #!/usr/bin/env python2
00002 # -*- coding: utf-8 -*-
00003
00004 #=====
00005 # PowerMeter daemon settings use example
00006 #=====
00007
00008 ## Read settings
00009 import settings
00010
00011 def header(txt):
00012     print "-----"
00013     print "{0:^60}".format(txt)
00014     print "-----"
00015
00016
00017 ## Shows the information given by the settings file
00018 def main():
00019
00020     # Show all the computers
00021     header("Computers")
00022     for computer in settings.computers.values():
00023         print computer
00024     print
00025     print
00026
```

```

00027     # Show all the devices
00028     header("Devices")
00029     for device in settings.devices.values():
00030         print device
00031     print
00032     print
00033
00034     # Access the devices of a given computer
00035     header("Devices in computer 'lorca'")
00036     for device in settings.computers['lorca'].devices.values():
00037         print device
00038     print
00039     print
00040
00041     header("Devices in computer 'matserve'")
00042     for device in settings.computers['matserve'].devices.values():
00043         print device
00044     print
00045     print
00046
00047     # Access to which computer a given device is attached to
00048     header("Computer to which device 'DCMeter' is attached")
00049     print settings.devices["DCMeter"].computer
00050     print
00051     print
00052
00053     # Access the lines of a given device
00054     header("Lines in device 'DCMeter'")
00055     for line in settings.devices["DCMeter"].lines.values():
00056         print line
00057     print
00058     print
00059
00060     header("Lines in device 'PDU'")
00061     for line in settings.devices["PDU"].lines.values():
00062         print line
00063     print
00064     print
00065
00066     # Access individual properties of a given line of a given device
00067     header("Properties of line 'DC 12V' of device 'DCMeter'")
00068     line=settings.devices["DCMeter"].lines["DC 12V"]
00069     print "Line name: {}".format(line.name)
00070     print "Line voltage: {}".format(line.voltage)
00071     print "Line description: '{}'".format(line.description)
00072     print
00073
00074
00075 if __name__=="__main__":
00076     main()

```

2.3 /home/barrachi/datos/aplicaciones/powermeter/settings.py File Reference

Variables

- int **PORT** = 6526
- string **LOGFILENAME** = "/var/log/powermeter.log"
- tuple **comp1** = Computer(name="lorca", ip="150.128.83.25")
- tuple **comp2** = Computer(name="matserve", ip="150.128.83.35")
- tuple **dev1** = DCDevice(name="DCMeter", computer=comp1, url="file:///dev/usb0", max_frequency=25)
- tuple **dev2** = DC2Device(name="DCMeter2", computer=comp1, url="file:///dev/usb1", max_frequency=1000)

- tuple **dev3** = WattsUpDevice(name="WattsUp", computer=comp2, url="file:///dev/usb3", max_frequency=50)
- tuple **dev4** = PDUDevice(name="PDU", url="ssh://user:pass@matsserv.uji.es", max_frequency=1000)

2.4 /home/barrachi/datos/aplicaciones/powermeter/settings.py

```

00001 #!/usr/bin/env python2
00002 # -*- coding: utf-8 -*-
00003
00004 #=====
00005 # PowerMeter daemon settings
00006 #=====
00007
00008 from settings_classes import computers, devices, Computer
00009 from settings_classes import DCDevice, DC2Device, NIDevice, WattsUpDevice,
00010 PDUDevice
00011
00012 #-----
00013 # General section
00014 #-----
00015
00016 # Port in which the daemon will be listening (default: 6526)
00017 PORT=6526
00018
00019 # Log file name (default: "/var/log/powermeter.log")
00020 LOGFILENAME="/var/log/powermeter.log"
00021
00022 #-----
00023 # Computers section
00024 #-----
00025
00026 comp1=Computer(name="lorca", ip="150.128.83.25")
00027 comp2=Computer(name="matsserv", ip="150.128.83.35")
00028
00029 #-----
00030 # Devices section
00031 #-----
00032
00033 dev1=DCDevice(name="DCMeter", computer=comp1, url="file:///dev/usb0",
00034 max_frequency=25)
00035 dev1.add_line(name="DC 12V", voltage=12, description="A 12V power line")
00036 dev1.add_line(name="DC 3V", voltage=3, description="A 3V power line")
00037
00038 dev2=DC2Device(name="DCMeter2", computer=comp1, url="file:///dev/usb1",
00039 max_frequency=1000)
00040 dev2.add_line(name="DC2 12V", voltage=12, description="A 12V power line")
00041 dev2.add_line(name="DC2 3V", voltage=3, description="A 3V power line")
00042
00043 dev2=NIDevice(name="National Instruments", computer=comp1, url="file:///dev/usb2",
00044 max_frequency=1000)
00045 dev2.add_line(name="DC2 12V", voltage=12, description="A 12V power line")
00046 dev2.add_line(name="DC2 3V", voltage=3, description="A 3V power line")
00047
00048 dev3=WattsUpDevice(name="WattsUp", computer=comp2, url="file:///dev/usb3",
00049 max_frequency=50)
00050
00051 dev4=PDUDevice(name="PDU", url="ssh://user:pass@matsserv.uji.es", max_frequency=
00052 1000)
00053 dev4.add_line(name="PDU lorca", computer=comp1, voltage=220, description="lorca
00054 watts")
00055 dev4.add_line(name="PDU matsserv", computer=comp2, voltage=220, description="
00056 matsserv watts")

```

2.5 /home/barrachi/datos/aplicaciones/powermeter/settings_classes.py File Reference

Data Structures

- class [Computer](#)
A computer description.
- class [Line](#)
A line description.
- class [PDULine](#)
A PDU line description.
- class [Device](#)
A device description.
- class [AttachedDevice](#)
A device attached to a computer description.
- class [DCDevice](#)
A DCMeter device description.
- class [DC2Device](#)
A DC2Meter device description.
- class [NIDevice](#)
A National Instruments device description.
- class [WattsUpDevice](#)
A WattsUp device description.
- class [PDUDevice](#)
A PDU device description.

Variables

- dictionary **devices** = {}
Dictionary of devices.
- dictionary **computers** = {}
Dictionary of computers.

2.6 /home/barrachi/datos/aplicaciones/powermeter/settings_classes.py

```

00001 #!/usr/bin/env python2
00002 # -*- coding: utf-8 -*-
00003
00004 #=====
00005 # PowerMeter daemon setting classes
00006 #
00007 # This module defines the classes and dictionaries used by settings.py
00008 #=====
00009

```

```

00010
00011 ## Dictionary of devices
00012 devices={}
00013
00014 ## Dictionary of computers
00015 computers={}
00016
00017
00018 ## A computer description
00019 #
00020 class Computer(object):
00021
00022     ## Creates a computer description and adds it to the computers
00023     ## dictionary
00024     #
00025     # @param [in] name    The name of the computer
00026     # @param [in] ip      The IP address of the computer
00027     #
00028     def __init__(self, name, ip):
00029         self.name=name
00030         self.ip=ip
00031         self.devices={}
00032         # Register the computer
00033         computers[name]=self
00034
00035     ## Returns a string representation for this computer
00036     def __repr__(self):
00037         return "Computer {0} ({1}): {2} device(s)".format(self.name, self.ip,
len(self.devices))
00038
00039     ## Adds a device description to the computer
00040     #
00041     # @param [in] device  A device description object
00042     #
00043     def add(self, device):
00044         if not isinstance(device, Device):
00045             msg="the given device parameter is not a Device object"
00046             raise SyntaxError, msg
00047             self.devices[device.name]=device
00048
00049
00050
00051 ## A line description
00052 #
00053 class Line(object):
00054
00055     ## Creates a line description
00056     #
00057     # @param [in] name      The line name (used for identification)
00058     # @param [in] voltage   The line voltage
00059     # @param [in] description A text description of the line
00060     #
00061     def __init__(self, name, voltage, description):
00062         self.name=name
00063         self.voltage=voltage
00064         self.description=description
00065
00066     ## Returns a string representation for this line
00067     def __repr__(self):
00068         return "Line {0} (voltage: {1}, description: '{2}')"
00069         .format(self.name,
self.voltage, self.description)
00070
00071
00072 ## A PDU line description
00073 #
00074 class PDULine(object):
00075
00076     ## Creates a PDU line description
00077     #
00078     # @param [in] name      The line name (used for identification)
00079     # @param [in] computer  The computer this PDU line is attached to
00080     # @param [in] voltage   The line voltage
00081     # @param [in] description An optional text description of the line

```

```

00082     #
00083     def __init__(self, name, computer, voltage, description=""):
00084         if not isinstance(computer, Computer):
00085             msg="the given computer parameter is not a Computer object"
00086             raise SyntaxError, msg
00087         self.name=name
00088         self.computer=computer
00089         self.voltage=voltage
00090         self.description=description
00091
00092     ## Returns a string representation for this line
00093     def __repr__(self):
00094         return "Line {0} (computer: '{1}', voltage: {2}, description: '{3}')."
00095         format(self.name, self.computer.name, self.voltage, self.description)
00096
00097
00098     ## A device description
00099     #
00100     class Device(object):
00101
00102         ## Creates a device description and adds it to the devices
00103         ## dictionary
00104         #
00105         # Before adding the given device description to the devices
00106         # dictionary, it checks that the name of the new device has not
00107         # been used by a previously added device.
00108         #
00109         # @param [in] name          The device name (used for identification, must
00110         #                           be unique)
00111         # @param [in] url           The url of this device
00112         # @param [in] max_frequency The maximum sample frequency of the device
00113         #
00114         def __init__(self, name, url, max_frequency):
00115             self.name=name
00116             self.url=url
00117             self.max_frequency=max_frequency
00118             self.lines={}
00119             if devices.has_key(name):
00120                 msg="there are at least two devices with the same name ({0})."
00121                 format(self.name)
00122                 raise SyntaxError, msg
00123             # Register the device
00124             devices[name]=self
00125
00126         ## Returns a string representation for this device
00127         def __repr__(self):
00128             return "Device {0} (url: '{1}', max frequency: {2}, lines: {3})."
00129             format(self.name, self.url, self.max_frequency, len(self.lines))
00130
00131         ## Adds a line description to the device
00132         #
00133         # Before adding the given line description to the device, it
00134         # checks that the name of the new line has not been used by a
00135         # previously added line.
00136         #
00137         # @param [in] name          The line name (used for identification)
00138         # @param [in] voltage       The line voltage
00139         # @param [in] description  A text description of the line
00140         #
00141         def add_line(self, name, voltage, description):
00142             if self.lines.has_key(name):
00143                 msg="there are at least two lines with the same name, '{0}', in
00144                 device '{1}'."
00145                 format(name, self.name)
00146                 raise SyntaxError, msg
00147             self.lines[name]=Line(name, voltage, description)
00148
00149
00150     ## A device attached to a computer description
00151     #
00152     class AttachedDevice(Device):
00153
00154         ## Creates device attached to a computer description and adds it

```

```

00151     ## to the devices dictionary
00152     #
00153     # Before adding the given device description to the devices
00154     # dictionary, it checks that the name of the new device has not
00155     # been used by a previously added device.
00156     #
00157     # @param [in] name          The device name (used for identification, must
be unique)
00158     # @param [in] computer      The computer the device is attached to
00159     # @param [in] url           The url of this device
00160     # @param [in] max_frequency The maximum sample frequency of the device
00161     #
00162     def __init__(self, name, computer, url, max_frequency):
00163         if not isinstance(computer, Computer):
00164             msg="the given computer parameter is not a Computer object"
00165             raise SyntaxError, msg
00166         self.computer=computer
00167         super(AttachedDevice, self).__init__(name, url, max_frequency)
00168         # Register the device in the computer it is attached to
00169         self.computer.add(self)
00170
00171
00172     ## A DCMeter device description
00173     #
00174     class DCDevice(AttachedDevice):
00175
00176         ## Creates a DC2Meter device description and adds it to the
00177         ## devices dictionary
00178         #
00179         # @param [in] name          The device name (used for identification, must
be unique)
00180         # @param [in] computer      The computer the device is attached to
00181         # @param [in] url           The url of this device
00182         # @param [in] max_frequency The maximum sample frequency of the device
00183         #
00184         def __init__(self, name, computer, url, max_frequency):
00185             super(DCDevice, self).__init__(name, computer, url, max_frequency)
00186
00187
00188     ## A DC2Meter device description
00189     #
00190     class DC2Device(AttachedDevice):
00191
00192         ## Creates a DC2Meter device description and adds it to the
00193         ## devices dictionary
00194         #
00195         # @param [in] name          The device name (used for identification, must
be unique)
00196         # @param [in] computer      The computer the device is attached to
00197         # @param [in] url           The url of this device
00198         # @param [in] max_frequency The maximum sample frequency of the device
00199         #
00200         def __init__(self, name, computer, url, max_frequency):
00201             super(DC2Device, self).__init__(name, computer, url, max_frequency)
00202
00203
00204     ## A National Instruments device description
00205     #
00206     class NIDevice(AttachedDevice):
00207
00208         ## Creates a National Instruments device description and adds it
00209         ## to the devices dictionary
00210         #
00211         # @param [in] name          The device name (used for identification, must
be unique)
00212         # @param [in] computer      The computer the device is attached to
00213         # @param [in] url           The url of this device
00214         # @param [in] max_frequency The maximum sample frequency of the device
00215         #
00216         def __init__(self, name, computer, url, max_frequency):
00217             super(NIDevice, self).__init__(name, computer, url, max_frequency)
00218
00219
00220

```

```

00221
00222
00223 ## A WattsUp device description
00224 #
00225 class WattsUpDevice(AttachedDevice):
00226
00227     ## Creates a WattsUp device description and adds it to the devices
00228     ## dictionary
00229     #
00230     # @param [in] name            The device name (used for identification, must
be unique)
00231     # @param [in] computer        The computer the device is attached to
00232     # @param [in] url             The url of this device
00233     # @param [in] max_frequency  The maximum sample frequency of the device
00234     #
00235     def __init__(self, name, computer, url, max_frequency):
00236         super(WattsUpDevice, self).__init__(name, computer, url, max_frequency)
00237
00238     ## Fake adding of line description to the device
00239     #
00240     # The WattsUp Device does not have lines. This method avoids the
00241     # base class method silently been called.
00242     #
00243     # @param [in] name            The line name (used for identification)
00244     # @param [in] voltage         The line voltage
00245     # @param [in] description    A text description of the line
00246     #
00247     def add_line(self, name, voltage, description):
00248         msg="a WattsUp Device can not have lines"
00249         raise SyntaxError, msg
00250
00251
00252
00253 ## A PDU device description
00254 #
00255 class PDUDevice(Device):
00256
00257     ## Creates a PDU device description and adds it to the devices
00258     ## dictionary
00259     #
00260     # @param [in] name            The device name (used for identification, must
be unique)
00261     # @param [in] url             The url of this device
00262     # @param [in] max_frequency  The maximum sample frequency of the device
00263     #
00264     def __init__(self, name, url, max_frequency):
00265         super(PDUDevice, self).__init__(name, url, max_frequency)
00266
00267
00268     ## Adds a pdu line description to the device
00269     #
00270     # Before adding the given line description to the device, it
00271     # checks that the name of the new line has not been used by a
00272     # previously added line.
00273     #
00274     # @param [in] name            The line name (used for identification)
00275     # @param [in] computer        The computer the line is attached to
00276     # @param [in] voltage         The line voltage
00277     # @param [in] description    An optional text description of the line
00278     #
00279     def add_line(self, name, computer, voltage, description=""):
00280         if self.lines.has_key(name):
00281             msg="there are at least two lines with the same name, '{0}', in
device '{1}'".format(name, self.name)
00282             raise SyntaxError, msg
00283             self.lines[name]=PDULine(name, computer, voltage, description)
00284             computer.add(self)

```