confduino Documentation

Release 0.0.1

ponty

July 12, 2011

CONTENTS

confduino

Date July 12, 2011

PDF confduino.pdf

Contents:

confduino is an arduino library configurator

Links:

- home: https://github.com/ponty/confduino
- documentation: http://ponty.github.com/confduino
- arduino libraries: http://www.arduino.cc/en/Reference/Libraries

Features:

- list, install, remove arduino libraries
- list, install, remove arduino programmers
- list, install, remove arduino boards
- written in python
- crossplatform
- can be used as a python library or as a console program
- unpacker backend: patool
- downloader backend: urllib

Known problems:

- Python 3 is not supported
- tested only on linux
- some libraries with unusual structure can not be installed
- not all commands have console interface

CHAPTER

ONE

BASIC USAGE

install library:

```
>>> from confduino.libinstall import install_lib
>>> install_lib('http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip')
```

or on console:

```
python -m confduino.libinstall http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip
```

CHAPTER

TWO

INSTALLATION

2.1 General

- install arduino
- install python
- install setuptools
- install the program:

```
# as root
easy_install confduino
```

2.2 Ubuntu

```
sudo apt-get install arduino
sudo apt-get install python-setuptools
sudo easy_install confduino
```

2.3 Uninstall

first install pip:

```
# as root
pip uninstall confduino
```

CHAPTER

THREE

USAGE WITH LIBRARIES

3.1 Arduino path

If Arduino can not be found at default path, then `ARDUINO_HOME` environment variable should be set.

on Ubuntu: in `~/.profile`:

```
ARDUINO_HOME=~/opt/arduino
export ARDUINO_HOME
```

Default path:

- Mac: /Applications/Arduino.app/Contents/Resources/Java
- Linux: /usr/share/arduino/

3.2 List installed libraries

From python:

```
>>> from confduino.liblist import libraries
>>> libraries()
['AdvButton', 'Bounce', 'CapSense', 'Charlieplex', 'DallasTemperature', 'DateTime', 'DateTimeStrings'
```

From console:

```
$ python -m confduino.liblist
AdvButton
Bounce
CapSense
Charlieplex
DallasTemperature
DateTime
DateTimeStrings
DigitalToggle
EEPROM
Enerlib
EventFuse
Flash
FreqCounter
LiquidCrystal
Metro
MsTimer2
```

```
NewSoftSerial
OneWire
PID_v1
PS2Keyboard
PString
PinChangeInt
Qtouch1Wire
SSerial2Mobile
SerialIP
SevenSegment
Streaming
TimedAction
TimerOne
Tone
UComms
bassdll
```

3.3 Install new library

Existing library will not be changed.

From python:

```
>>> from confduino.libinstall import install_lib
>>> install_lib('http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip')
```

From console:

```
python -m confduino.libinstall http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip
```

3.4 Upgrade existing library

Same as install with *replace_existing* option.

From python:

```
>>> from confduino.libinstall import install_lib
>>> install_lib('http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip', replace_existing=1)
```

From console:

```
python -m confduino.libinstall http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip --replace
```

3.5 Remove existing library

From python:

```
>>> from confduino.libremove import remove_lib
>>> remove_lib('PS2Keyboard')
```

From console:

```
python -m confduino.libremove PS2Keyboard
```

CHAPTER

FOUR

USAGE WITH BOARDS

4.1 List installed boards

From python:

```
>>> from confduino.boardlist import boards
>>> boards()
AutoBunch(atmega168=AutoBunch(bootloader=AutoBunch(extended_fuses='0x00', file='ATmegaBOOT_168_ng.hex
```

From console:

```
$ python -m confduino.boardlist
{'atmega168': {'bootloader': {'extended_fuses': '0x00',
                              'file': 'ATmegaBOOT_168_ng.hex',
                              'high_fuses': '0xdd',
                              'lock_bits': '0x0F',
                              'low_fuses': '0xff',
                              'path': 'atmega',
                              'unlock_bits': '0x3F'},
               'build': {'core': 'arduino',
                         'f_cpu': '16000000L',
                         'mcu': 'atmega168'},
               'name': 'Arduino NG or older w/ ATmega168',
               'upload': {'maximum_size': '14336',
                          'protocol': 'stk500',
                          'speed': '19200'}},
 'atmega328': {'bootloader': {'extended_fuses': '0x05',
                              'file': 'ATmegaBOOT_168_atmega328.hex',
                              'high_fuses': '0xDA',
                              'lock_bits': '0x0F',
                              'low_fuses': '0xFF',
                              'path': 'atmega',
                              'unlock_bits': '0x3F'},
               'build': {'core': 'arduino',
                         'f_cpu': '16000000L',
                         'mcu': 'atmega328p'},
               'name': 'Arduino Duemilanove or Nano w/ ATmega328',
               'upload': {'maximum_size': '30720',
                          'protocol': 'stk500',
                          'speed': '57600'}},
 'atmega8': {'bootloader': {'file': 'ATmegaBOOT.hex',
                            'high_fuses': '0xca',
                            'lock_bits': '0x0F',
                            'low_fuses': '0xdf',
```

```
                          'path': 'atmega8',
                          'unlock_bits': '0x3F'},
            'build': {'core': 'arduino',
                      'f_cpu': '16000000L',
                      'mcu': 'atmega8'},
            'name': 'Arduino NG or older w/ ATmega8',
            'upload': {'maximum_size': '7168',
                       'protocol': 'stk500',
                       'speed': '19200'}},
 'atmega88p': {'build': {'core': 'arduino',
                         'f_cpu': '16000000L',
                         'mcu': 'atmega88'},
               'name': 'Atmega88 parallel 16MHz',
               'upload': {'using': 'parallel'}},
 'atmega88u': {'build': {'core': 'arduino',
                         'f_cpu': '16000000L',
                         'mcu': 'atmega88'},
               'name': 'Atmega88 usbasp 16MHz',
               'upload': {'using': 'usbasp'}},
 'attiny2313at8': {'build': {'core': 'arduino',
                             'f_cpu': '8000000L',
                             'mcu': 'attiny2313'},
                   'name': 'ATtiny2313 @ 8 MHz',
                   'upload': {'maximum_size': '2048', 'using': 'usbasp'}},
 'bt': {'bootloader': {'extended_fuses': '0x00',
                       'file': 'ATmegaBOOT_168.hex',
                       'high_fuses': '0xdd',
                       'lock_bits': '0x0F',
                       'low_fuses': '0xff',
                       'path': 'bt',
                       'unlock_bits': '0x3F'},
        'build': {'core': 'arduino',
                  'f_cpu': '16000000L',
                  'mcu': 'atmega168'},
        'name': 'Arduino BT w/ ATmega168',
        'upload': {'disable_flushing': 'true',
                   'maximum_size': '14336',
                   'protocol': 'stk500',
                   'speed': '19200'}},
 'bt328': {'bootloader': {'extended_fuses': '0x05',
                          'file': 'ATmegaBOOT_168_atmega328_bt.hex',
                          'high_fuses': '0xd8',
                          'lock_bits': '0x0F',
                          'low_fuses': '0xff',
                          'path': 'bt',
                          'unlock_bits': '0x3F'},
           'build': {'core': 'arduino',
                     'f_cpu': '16000000L',
                     'mcu': 'atmega328p'},
           'name': 'Arduino BT w/ ATmega328',
           'upload': {'disable_flushing': 'true',
                      'maximum_size': '28672',
                      'protocol': 'stk500',
                      'speed': '19200'}},
 'diecimila': {'bootloader': {'extended_fuses': '0x00',
                              'file': 'ATmegaBOOT_168_diecimila.hex',
                              'high_fuses': '0xdd',
                              'lock_bits': '0x0F',
```

```
                            'low_fuses': '0xff',
                            'path': 'atmega',
                            'unlock_bits': '0x3F'},
              'build': {'core': 'arduino',
                        'f_cpu': '16000000L',
                        'mcu': 'atmega168'},
              'name': 'Arduino Diecimila, Duemilanove, or Nano w/ ATmega168',
              'upload': {'maximum_size': '14336',
                         'protocol': 'stk500',
                         'speed': '19200'}},
 'fio': {'bootloader': {'extended_fuses': '0x05',
                        'file': 'ATmegaBOOT_168_atmega328_pro_8MHz.hex',
                        'high_fuses': '0xDA',
                        'lock_bits': '0x0F',
                        'low_fuses': '0xFF',
                        'path': 'arduino:atmega',
                        'unlock_bits': '0x3F'},
         'build': {'core': 'arduino:arduino',
                   'f_cpu': '8000000L',
                   'mcu': 'atmega328p'},
         'name': 'Arduino Fio',
         'upload': {'maximum_size': '30720',
                    'protocol': 'stk500',
                    'speed': '57600'}},
 'lilypad': {'bootloader': {'extended_fuses': '0x00',
                            'file': 'LilyPadBOOT_168.hex',
                            'high_fuses': '0xdd',
                            'lock_bits': '0x0F',
                            'low_fuses': '0xe2',
                            'path': 'lilypad',
                            'unlock_bits': '0x3F'},
             'build': {'core': 'arduino',
                       'f_cpu': '8000000L',
                       'mcu': 'atmega168'},
             'name': 'LilyPad Arduino w/ ATmega168',
             'upload': {'maximum_size': '14336',
                        'protocol': 'stk500',
                        'speed': '19200'}},
 'lilypad328': {'bootloader': {'extended_fuses': '0x05',
                               'file': 'ATmegaBOOT_168_atmega328_pro_8MHz.hex',
                               'high_fuses': '0xDA',
                               'lock_bits': '0x0F',
                               'low_fuses': '0xFF',
                               'path': 'atmega',
                               'unlock_bits': '0x3F'},
                'build': {'core': 'arduino',
                          'f_cpu': '8000000L',
                          'mcu': 'atmega328p'},
                'name': 'LilyPad Arduino w/ ATmega328',
                'upload': {'maximum_size': '30720',
                           'protocol': 'stk500',
                           'speed': '57600'}},
 'mega': {'bootloader': {'extended_fuses': '0xF5',
                         'file': 'ATmegaBOOT_168_atmega1280.hex',
                         'high_fuses': '0xDA',
                         'lock_bits': '0x0F',
                         'low_fuses': '0xFF',
                         'path': 'atmega',
```

```
                        'unlock_bits': '0x3F'},
          'build': {'core': 'arduino',
                    'f_cpu': '16000000L',
                    'mcu': 'atmega1280'},
          'name': 'Arduino Mega (ATmega1280)',
          'upload': {'maximum_size': '126976',
                     'protocol': 'stk500',
                     'speed': '57600'}},
 'mega2560': {'bootloader': {'extended_fuses': '0xFD',
                             'file': 'stk500boot_v2_mega2560.hex',
                             'high_fuses': '0xD8',
                             'lock_bits': '0x0F',
                             'low_fuses': '0xFF',
                             'path': 'stk500v2',
                             'unlock_bits': '0x3F'},
              'build': {'core': 'arduino',
                        'f_cpu': '16000000L',
                        'mcu': 'atmega2560'},
              'name': 'Arduino Mega 2560',
              'upload': {'maximum_size': '258048',
                         'protocol': 'stk500v2',
                         'speed': '115200'}},
 'mini': {'bootloader': {'extended_fuses': '0x00',
                         'file': 'ATmegaBOOT_168_ng.hex',
                         'high_fuses': '0xdd',
                         'lock_bits': '0x0F',
                         'low_fuses': '0xff',
                         'path': 'atmega',
                         'unlock_bits': '0x3F'},
          'build': {'core': 'arduino',
                    'f_cpu': '16000000L',
                    'mcu': 'atmega168'},
          'name': 'Arduino Mini',
          'upload': {'maximum_size': '14336',
                     'protocol': 'stk500',
                     'speed': '19200'}},
 'pro': {'bootloader': {'extended_fuses': '0x00',
                        'file': 'ATmegaBOOT_168_pro_8MHz.hex',
                        'high_fuses': '0xdd',
                        'lock_bits': '0x0F',
                        'low_fuses': '0xc6',
                        'path': 'atmega',
                        'unlock_bits': '0x3F'},
         'build': {'core': 'arduino',
                   'f_cpu': '8000000L',
                   'mcu': 'atmega168'},
         'name': 'Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168',
         'upload': {'maximum_size': '14336',
                    'protocol': 'stk500',
                    'speed': '19200'}},
 'pro328': {'bootloader': {'extended_fuses': '0x05',
                           'file': 'ATmegaBOOT_168_atmega328_pro_8MHz.hex',
                           'high_fuses': '0xDA',
                           'lock_bits': '0x0F',
                           'low_fuses': '0xFF',
                           'path': 'atmega',
                           'unlock_bits': '0x3F'},
            'build': {'core': 'arduino',
```

```
                    'f_cpu': '8000000L',
                    'mcu': 'atmega328p'},
          'name': 'Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328',
          'upload': {'maximum_size': '30720',
                     'protocol': 'stk500',
                     'speed': '57600'}},
 'pro5v': {'bootloader': {'extended_fuses': '0x00',
                          'file': 'ATmegaBOOT_168_diecimila.hex',
                          'high_fuses': '0xdd',
                          'lock_bits': '0x0F',
                          'low_fuses': '0xff',
                          'path': 'atmega',
                          'unlock_bits': '0x3F'},
           'build': {'core': 'arduino',
                     'f_cpu': '16000000L',
                     'mcu': 'atmega168'},
           'name': 'Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168',
           'upload': {'maximum_size': '14336',
                      'protocol': 'stk500',
                      'speed': '19200'}},
 'pro5v328': {'bootloader': {'extended_fuses': '0x05',
                             'file': 'ATmegaBOOT_168_atmega328.hex',
                             'high_fuses': '0xDA',
                             'lock_bits': '0x0F',
                             'low_fuses': '0xFF',
                             'path': 'atmega',
                             'unlock_bits': '0x3F'},
              'build': {'core': 'arduino',
                        'f_cpu': '16000000L',
                        'mcu': 'atmega328p'},
              'name': 'Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328',
              'upload': {'maximum_size': '30720',
                         'protocol': 'stk500',
                         'speed': '57600'}},
 'uno': {'bootloader': {'extended_fuses': '0x05',
                        'file': 'optiboot_atmega328.hex',
                        'high_fuses': '0xde',
                        'lock_bits': '0x0F',
                        'low_fuses': '0xff',
                        'path': 'optiboot',
                        'unlock_bits': '0x3F'},
         'build': {'core': 'arduino',
                   'f_cpu': '16000000L',
                   'mcu': 'atmega328p'},
         'name': 'Arduino Uno',
         'upload': {'maximum_size': '32256',
                    'protocol': 'stk500',
                    'speed': '115200'}}}
```

4.2 Install new board

Existing board will not be changed.

From python:

```
from confduino.boardinstall import install_board
from confduino.util import AutoBunch
from entrypoint2 import entrypoint

@entrypoint
def install():
    'install atmega88 board'
    atmega88 = AutoBunch()
    atmega88.name='Atmega88 usbasp 16MHz'

    atmega88.upload.using='usbasp'

    atmega88.build.mcu='atmega88'
    atmega88.build.f_cpu='16000000L'
    atmega88.build.core='arduino'

    install_board('atmega88', atmega88, replace_existing=0)
```

console is not implemented

4.3 Remove existing board

From python:

```
>>> from confduino.boardremove import remove_board
>>> remove_board('diecimila')
```

From console:

```
python -m confduino.boardremove diecimila
```

CHAPTER

FIVE

USAGE WITH PROGRAMMERS

5.1 List installed programmers

From python:

```
>>> from confduino.proglist import programmers
>>> programmers()
AutoBunch(arduinoisp=AutoBunch(communication='serial', name='Arduino as ISP', protocol='stk500v1', sp
```

From console:

```
$ python -m confduino.proglist
{'arduinoisp': {'communication': 'serial',
                'name': 'Arduino as ISP',
                'protocol': 'stk500v1',
                'speed': '19200'},
 'avrisp': {'communication': 'serial',
            'name': 'AVR ISP',
            'protocol': 'stk500v1'},
 'avrispmkii': {'communication': 'usb',
                'name': 'AVRISP mkII',
                'protocol': 'stk500v2'},
 'parallel': {'force': 'true',
              'name': 'Parallel Programmer',
              'protocol': 'dapa'},
 'stk200': {'force': 'true', 'name': 'STK200', 'protocol': 'dapa'},
 'usbasp': {'communication': 'usb', 'name': 'USBasp', 'protocol': 'usbasp'},
 'usbtinyisp': {'name': 'USBtinyISP', 'protocol': 'usbtiny'}}
```

5.2 Install new programmer

Existing programmer will not be changed.

From python:

```
from confduino.proginstall import install_programmer
from confduino.util import AutoBunch
from entrypoint2 import entrypoint

@entrypoint
def install(replace_existing=False):
    'install usbasp programmer'
```

```
    usbasp = AutoBunch()
    usbasp.name = 'USBasp'
    usbasp.communication = 'usb'
    usbasp.protocol = 'usbasp'

    install_programmer('usbasp', usbasp, replace_existing=replace_existing)
```

console is not implemented

5.3 Remove existing programmer

From python:

```
>>> from confduino.progremove import remove_programmer
>>> remove_programmer('parallel')
```

From console:

```
python -m confduino.progremove parallel
```

CHAPTER

SIX

EXAMPLES

Many libraries are upgraded in examples/upgrademany.py, this can be started:

```
python -m confduino.examples.upgrademany
```

Code:

```
from confduino.libinstall import install_lib
from entrypoint2 import entrypoint

UPGRADE = True

def upgrade(url):
    print 'upgrading ' + url
    install_lib(url, UPGRADE)

@entrypoint
def upgrade_many():
    'upgrade many libs'

    # you can set your arduino path if it is not default
    #os.environ['ARDUINO_HOME'] = '/home/...'

    ############################
    # arduino.cc
    ############################
    upgrade('http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/Metro.zip')
    upgrade('http://www.arduino.cc/playground/uploads/Main/MsTimer2.zip')
#     upgrade('http://www.arduino.cc/playground/uploads/Code/Time.zip')
#     upgrade('http://arduino.cc/playground/uploads/Main/LedControl.zip')
#     upgrade('http://www.arduino.cc/playground/uploads/Code/ks0108GLCD.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/Bounce.zip')
    upgrade('http://arduino.cc/playground/uploads/Main/CapacitiveSense003.zip')
    upgrade('http://arduino.cc/playground/uploads/Main/PinChangeInt.zip')
#     upgrade('http://arduino.cc/playground/uploads/Code/TimerThree.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/TimedAction-1_6.zip')
#     upgrade('http://www.arduino.cc/playground/uploads/Code/Time.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/EventFuse.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/Charlieplex.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/DigitalToggle.zip')
    upgrade('http://arduino.cc/playground/uploads/Code/Enerlib.zip')

    upgrade('http://arduino.cc/playground/uploads/Code/AdvButton_11.zip')
    #upgrade('http://arduino.cc/playground/uploads/Code/AdvButton.zip') # old version
```

```
    ############################
    # arduiniana.org
    ############################
    # TODO: how to get latest version??
    upgrade('http://arduiniana.org/PString/PString2.zip')
    upgrade('http://arduiniana.org/Flash/Flash3.zip')
    upgrade('http://arduiniana.org/NewSoftSerial/NewSoftSerial10c.zip')
    upgrade('http://arduiniana.org/Streaming/Streaming4.zip')
#    upgrade('http://arduiniana.org/PWMServo/PWMServo.zip')
#    upgrade('http://arduiniana.org/TinyGPS/TinyGPS10.zip')


    ############################
    # google
    ############################
    upgrade('http://rogue-code.googlecode.com/files/Arduino-Library-Tone.zip')
#    upgrade('http://arduino-playground.googlecode.com/files/LedDisplay03.zip')
    upgrade('http://sserial2mobile.googlecode.com/files/SSerial2Mobile-1.1.0.zip')
#    upgrade('http://webduino.googlecode.com/files/webduino-1.4.1.zip')
    upgrade('http://arduino-pid-library.googlecode.com/files/PID_v1.0.1.zip')
    upgrade('http://ideoarduinolibraries.googlecode.com/files/Qtouch1Wire.zip')
    upgrade('http://arduino-timerone.googlecode.com/files/TimerOne-v2.zip')


    ############################
    # others
    ############################
    upgrade('http://download.milesburton.com/Arduino/MaximTemperature/DallasTemperature_370Beta.zip')
    upgrade('http://interface.khm.de/wp-content/uploads/2009/01/FreqCounter1.zip')
#    upgrade('http://github.com/wimleers/flexitimer2/zipball/v1.0')
#    upgrade('http://www.state-machine.com/arduino/qp_arduino.zip')
#    upgrade('ftp://momjian.us/pub/arduino/TButton.zip') # AdvButton is better
    upgrade('http://johnmchilton.com/media/UComms.zip')
    upgrade('http://www.shikadi.net/files/arduino/SerialIP-1.0.zip')
```

Install USBasp programmer:

```
python -m confduino.examples.usbasp
```

Code:

```
from confduino.proginstall import install_programmer
from confduino.util import AutoBunch
from entrypoint2 import entrypoint

@entrypoint
def install(replace_existing=False):
    'install usbasp programmer'
    usbasp = AutoBunch()
    usbasp.name = 'USBasp'
    usbasp.communication = 'usb'
    usbasp.protocol = 'usbasp'

    install_programmer('usbasp', usbasp, replace_existing=replace_existing)
```

Install STK200 programmer:

```
python -m confduino.examples.stk200
```

Code:

```
from confduino.proginstall import install_programmer
from confduino.util import AutoBunch
from entrypoint2 import entrypoint

@entrypoint
def install(replace_existing=False):
    'install stk200 programmer'
    bunch = AutoBunch()
    bunch.name = 'STK200'
    bunch.protocol = 'dapa'
    bunch.force = 'true'
    # bunch.delay=200

    install_programmer('stk200', bunch, replace_existing=replace_existing)
```

CHAPTER

SEVEN

COMMAND LINE HELP

7.1 lib

7.1.1 list

```
$ python -m confduino.liblist --help
usage: liblist.py [-h] [--debug]

print installed arduino libraries

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG
```

7.1.2 install

```
$ python -m confduino.libinstall --help
usage: libinstall.py [-h] [-r] [--debug] url

install library from web or local files system

positional arguments:
  url                   web address or file path

optional arguments:
  -h, --help            show this help message and exit
  -r, --replace-existing
                        bool
  --debug               set logging level to DEBUG
```

7.1.3 remove

```
$ python -m confduino.libremove --help
usage: libremove.py [-h] [--debug] lib_name

remove library

positional arguments:
  lib_name    library name (e.g. 'PS2Keyboard')
```

```
optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG
```

7.2 board

7.2.1 list

```
$ python -m confduino.boardlist --help
usage: boardlist.py [-h] [--debug]

print boards from boards.txt

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG
```

7.2.2 install

not implemented

7.2.3 remove

```
$ python -m confduino.boardremove --help
usage: boardremove.py [-h] [--debug] board_id

remove board

positional arguments:
  board_id    board id (e.g. 'diecimila')

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG
```

7.3 programmer

7.3.1 list

```
$ python -m confduino.proglist --help
usage: proglist.py [-h] [--debug]

print programmers from programmers.txt

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG
```

7.3.2 install

not implemented

7.3.3 remove

```
$ python -m confduino.progremove --help
usage: progremove.py [-h] [--debug] programmer_id

remove programmer

positional arguments:
  programmer_id  programmer id (e.g. 'avrisp')

optional arguments:
  -h, --help     show this help message and exit
  --debug        set logging level to DEBUG
```

CHAPTER

EIGHT

API

8.1 lib

confduino.liblist.**libraries**()
return installed library names

confduino.liblist.**libraries_dir**()
return library root path

confduino.liblist.**print_libraries**()
print installed arduino libraries

confduino.libinstall.**clean_lib_dir**(*root*)
remove .* and _* files and directories under root

confduino.libinstall.**find_lib_dir**(*root*)
search for lib dir under root

confduino.libinstall.**fix_examples_dir**(*lib_dir*)
rename examples dir to `examples`

confduino.libinstall.**install_lib**(*url*, *replace_existing=False*)
install library from web or local files system

Parameters

- **url** – web address or file path
- **replace_existing** – bool

Return type None

confduino.libinstall.**move_examples**(*root*, *lib_dir*)
find examples not under lib dir, and move into `examples`

confduino.libremove.**remove_lib**(*lib_name*)
remove library

Parameters **lib_name** – library name (e.g. 'PS2Keyboard')

Return type None

8.2 board

confduino.boardlist.**boards**()
read boards from boards.txt

confduino.boardlist.**boards_txt**()
path of boards.txt

confduino.boardlist.**print_boards**()
print boards from boards.txt

confduino.boardinstall.**install_board**(*board_id*, *board_options*, *replace_existing=False*)
install board in boards.txt

Parameters

- **board_id** – string identifier
- **board_options** – dict like
- **replace_existing** – bool

Return type None

confduino.boardremove.**remove_board**(*board_id*)
remove board

Parameters **board_id** – board id (e.g. 'diecimila')

Return type None

8.3 programmer

confduino.proglist.**print_programmers**()
print programmers from programmers.txt

confduino.proglist.**programmers**()
read programmers from programmers.txt

confduino.proglist.**programmers_txt**()
path of programmers.txt

confduino.proginstall.**install_programmer**(*programmer_id*, *programmer_options*, *replace_existing=False*)
install programmer in programmers.txt

Parameters

- **programmer_id** – string identifier
- **programmer_options** – dict like
- **replace_existing** – bool

Return type None

confduino.progremove.**remove_programmer**(*programmer_id*)
remove programmer

Parameters **programmer_id** – programmer id (e.g. 'avrisp')

Return type None

CHAPTER

NINE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

C

INDEX