confduino Documentation

Release 0.2

ponty

January 13, 2013

CONTENTS

confduino

Date January 13, 2013

PDF confduino.pdf

Contents:

confduino is an arduino library configurator

Links:

- home: https://github.com/ponty/confduino
- documentation: http://ponty.github.com/confduino

Features:

- **list, install, remove arduino libraries**
  - install libraries from internet or local drive
  - fix `examples` directory name before installing
  - clean library (.*,_*,..) before installing
  - move examples under `examples` directory
  - upgrade library to 1.0: replace `#include "wprogram.h"` with `#include "Arduino.h"`
- list, install, remove arduino programmers
- list, install, remove arduino boards
- written in python
- cross-platform
- can be used as a python library or as a console program
- unpacker back-end: pyunpack
- downloader back-end: urllib
- some functionality is based on arscons
- supported python versions: 2.7
- supported Arduino versions: 0022, 0023, 1.0

Known problems:

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

arduino libraries: http://www.arduino.cc/en/Reference/Libraries

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
```

install a lot of libraries:

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

CHAPTER

TWO

INSTALLATION

2.1 General

- install arduino
- install python
- install pip
- install back-ends for pyunpack (optional)
- install the program:

  ```
  # as root
  pip install confduino
  ```

2.2 Ubuntu

```
sudo apt-get install arduino
sudo apt-get install python-pip
sudo pip install confduino
sudo apt-get install unzip unrar p7zip-full
```

2.3 Uninstall

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

CHAPTER

THREE

ARDUINO PATH

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

on Ubuntu (https://help.ubuntu.com/community/EnvironmentVariables): in `~/.profile`:

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

temporary changes:

```
$ env ARDUINO_HOME=~/opt/arduino-0022 python -m confduino.version
0022

$ env ARDUINO_HOME=~/opt/arduino-1.0 python -m confduino.version
1.0
```

Default path:

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

CHAPTER

FOUR

CHECK ARDUINO VERSION

4.1 From python

```
>>> from confduino.version import version, intversion, sketch_extension
>>> from confduino import set_arduino_path
>>>
>>> version()
'1.0.3'
>>> intversion()
103
>>> sketch_extension()
'.ino'
>>>
>>> set_arduino_path('~/opt/arduino-0022')
>>> version()
'0022'
>>> intversion()
22
>>> sketch_extension()
'.pde'
>>>
>>> set_arduino_path('~/opt/arduino-1.0')
>>> version()
'1.0'
>>> intversion()
100
>>> sketch_extension()
'.ino'
```

4.2 From console

```
$ python -m confduino.version
1.0.3
```

Help:

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

print arduino version

optional arguments:
  -h, --help      show this help message and exit
  -i, --integer
  --debug         set logging level to DEBUG
  --version       show program's version number and exit
```

4.3 Examples

```
$ env ARDUINO_HOME=~/opt/arduino-0022 python -m confduino.version
0022

$ env ARDUINO_HOME=~/opt/arduino-0022 python -m confduino.version --integer
22

$ env ARDUINO_HOME=~/opt/arduino-1.0 python -m confduino.version
1.0

$ env ARDUINO_HOME=~/opt/arduino-1.0 python -m confduino.version --integer
100
```

CHAPTER

FIVE

MENU ITEM "ALL"

5.1 Create menu item "all" for examples

If you have a lot of libraries and low screen resolution then all menu items under "examples" can not be accessed.

Bug report: "Long menus don't scroll" (http://code.google.com/p/arduino/issues/detail?id=426)

My workaround creates a 2 level deep menu structure without changing other menu items. Symbolic links are used if possible.

From python:

```
>>> from confduino.exampallcreate import create_examples_all
>>> create_examples_all()
```

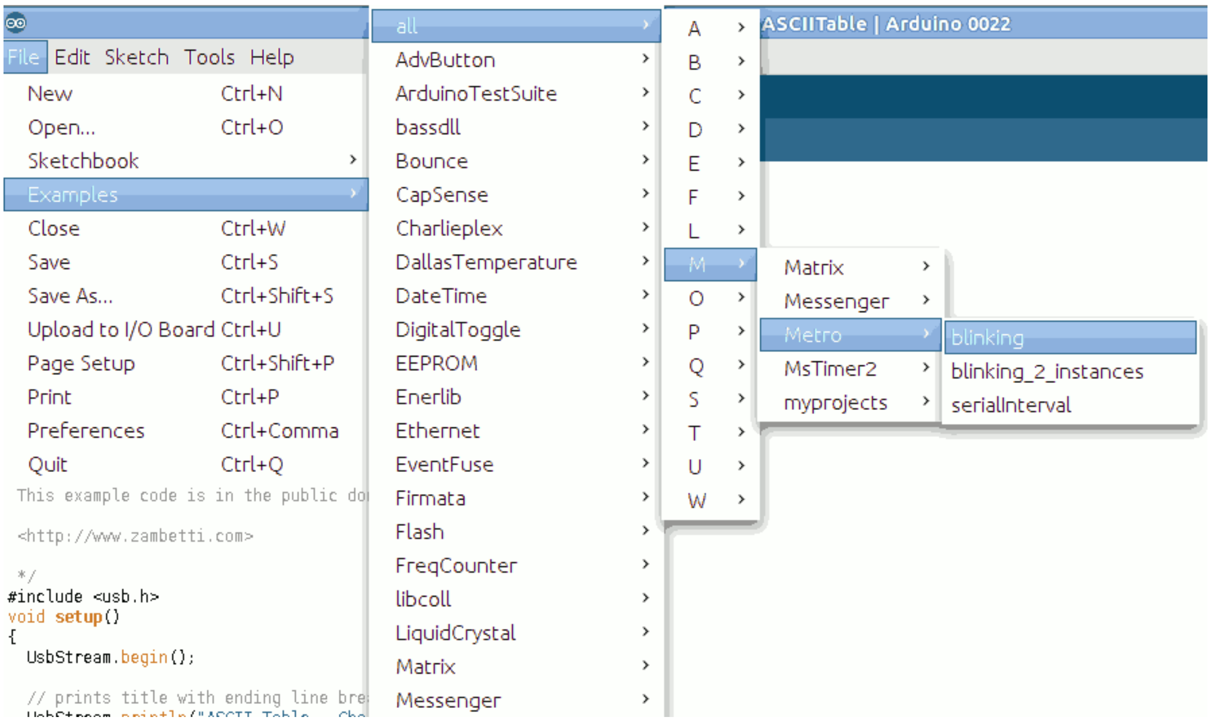
From console:

```
python -m confduino.exampallcreate
```

Help:

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

create arduino/examples/all directory

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

Result:



```
This example code is in the public do

<http://www.zambetti.com>

*/
#include <usb.h>
void setup()
{
  UsbStream.begin();

  // prints title with ending line bre
```

5.2 Removing menu item 'all'

From python:

```
>>> from confduino.exampallremove import remove_examples_all
>>> remove_examples_all()
```

From console:

```
python -m confduino.exampallremove
```

Help:

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

remove arduino/examples/all directory

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

CHAPTER

SIX

USAGE WITH LIBRARIES

6.1 List installed libraries

From python:

```
>>> from confduino.liblist import libraries
>>> libraries()
['AdvButton', 'ArduinoUnit', 'AtTouch', 'Bounce', 'Button', 'ByteBuffer', 'CapSense', 'Charlieple
```

From console:

```
$ python -m confduino.liblist
AdvButton
ArduinoUnit
AtTouch
Bounce
Button
ByteBuffer
CapSense
Charlieplex
DB
DallasTemperature
DataFlash
DigitalToggle
EDB
EEPROM
EasyTransferI2C
Enerlib
Esplora
Ethernet
EventFuse
FancyLED
Firmata
Flash
FrequencyTimer2
LED
LPM11162
LedControl
LedDisplay
LiquidCrystal
LowPower
MatrixMath
Metro
MorseEnDecoder
MsTimer2
Narcoleptic
NewSoftSerial
NoiseFilter
```

```
OneWire
PID_v1
PS2Keyboard
PS2X_lib
PString
PWMServo
PinChangeInt
Ping
Qtouch1Wire
QueueArray
QueueList
SD
SPI
SSerial2Mobile
SerialIP
SerialManager
Servo
SevSeg
SoftEasyTransfer
SoftUsb
SoftwareSerial
StackArray
StackList
Stepper
Streaming
TVout
TimedAction
TimerOne
TinyGPS
Tone
Tween
Twitter
WebServer
WiFi
WiShield
Wire
arduinode
morse
multiCameraIrControl
rtttl
spline
```

Help:

```
$ 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
```

6.2 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
```

6.3 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=
```

From console:

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

Help:

```
$ python -m confduino.libinstall --help
usage: libinstall.py [-h] [-r] [-f] [--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
  -f, --fix-wprogram
  --debug               set logging level to DEBUG
```

6.4 Remove existing library

From python:

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

From console:

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

Help:

```
$ 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
```

CHAPTER

SEVEN

USAGE WITH BOARDS

7.1 List installed boards

From python:

```
>>> from confduino.boardlist import boards
>>> boards()
AutoBunch(LilyPadUSB=AutoBunch(bootloader=AutoBunch(extended_fuses='0xce', file='Caterina-LilyPadU
>>> boards().diecimila.build.f_cpu
'16000000L'
>>> boards()['diecimila']['build']['f_cpu']
'16000000L'
```

From console:

```
$ python -m confduino.boardlist
LilyPadUSB
atmega168
atmega328
atmega328p_1000000
atmega328p_20000000
atmega328p_8000000
atmega8
atmega88_1000000
atmega88_12000000
atmega88_20000000
atmega88_8000000
atmega8_1000000
atmega8_12000000
bt
bt328
diecimila
esplora
ethernet
fio
leonardo
lilypad
lilypad328
mega
mega2560
micro
mini
mini328
nano
nano328
pro
pro328
pro5v
```

```
pro5v328
uno
```

verbose (JSON compatible):

```
$ python -m confduino.boardlist --verbose
{
    "LilyPadUSB": {
        "bootloader": {
            "extended_fuses": "0xce",
            "file": "Caterina-LilyPadUSB.hex",
            "high_fuses": "0xd8",
            "lock_bits": "0x2F",
            "low_fuses": "0xff",
            "path": "caterina-LilyPadUSB",
            "unlock_bits": "0x3F"
        },
        "build": {
            "core": "arduino",
            "f_cpu": "8000000L",
            "mcu": "atmega32u4",
            "pid": "0x9208",
            "variant": "leonardo",
            "vid": "0x1B4F"
        },
        "name": "LilyPad Arduino USB",
        "upload": {
            "disable_flushing": "true",
            "maximum_size": "28672",
            "protocol": "avr109",
            "speed": "57600"
        }
    },
    "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",
            "variant": "standard"
        },
        "name": "Arduino NG or older w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "Arduino Duemilanove w/ ATmega328",
        "upload": {
            "maximum_size": "30720",
            "protocol": "arduino",
            "speed": "57600"
        }
    },
    "atmega328p_1000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "1000000L",
            "mcu": "atmega328p",
            "variant": "standard"
        },
        "name": "atmega328p@1MHz",
        "upload": {
            "maximum_size": "32768",
            "using": "usbasp"
        }
    },
    "atmega328p_20000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "20000000L",
            "mcu": "atmega328p",
            "variant": "standard"
        },
        "name": "atmega328p@20MHz",
        "upload": {
            "maximum_size": "32768",
            "using": "usbasp"
        }
    },
    "atmega328p_8000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "8000000L",
            "mcu": "atmega328p",
            "variant": "standard"
        },
        "name": "atmega328p@8MHz",
        "upload": {
            "maximum_size": "32768",
            "using": "usbasp"
        }
    },
    "atmega8": {
        "bootloader": {
            "file": "ATmegaBOOT-prod-firmware-2009-11-07.hex",
            "high_fuses": "0xca",
            "lock_bits": "0x0F",
            "low_fuses": "0xdf",
            "path": "atmega8",
            "unlock_bits": "0x3F"
        },
        "build": {
            "core": "arduino",
            "f_cpu": "16000000L",
            "mcu": "atmega8",
            "variant": "standard"
        },
        "name": "Arduino NG or older w/ ATmega8",
        "upload": {
            "maximum_size": "7168",
            "protocol": "arduino",
            "speed": "19200"
        }
    },
    "atmega88_1000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "1000000L",
            "mcu": "atmega88",
            "variant": "standard"
        },
        "name": "atmega88@1MHz",
        "upload": {
            "maximum_size": "8192",
            "using": "usbasp"
        }
    },
    "atmega88_12000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "12000000L",
            "mcu": "atmega88",
            "variant": "standard"
        },
        "name": "atmega88@12MHz",
        "upload": {
            "maximum_size": "8192",
            "using": "usbasp"
        }
    },
    "atmega88_20000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "20000000L",
            "mcu": "atmega88",
            "variant": "standard"
        },
        "name": "atmega88@20MHz",
        "upload": {
            "maximum_size": "8192",
            "using": "usbasp"
        }
    },
    "atmega88_8000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "8000000L",
            "mcu": "atmega88",
            "variant": "standard"
        },
        "name": "atmega88@8MHz",
        "upload": {
            "maximum_size": "8192",
            "using": "usbasp"
        }
    },
    "atmega8_1000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "1000000L",
            "mcu": "atmega8",
            "variant": "standard"
        },
        "name": "atmega8@1MHz",
        "upload": {
            "maximum_size": "8192",
            "using": "usbasp"
        }
    },
    "atmega8_12000000": {
        "build": {
            "core": "arduino",
            "f_cpu": "12000000L",
            "mcu": "atmega8",
            "variant": "standard"
        },
        "name": "atmega8@12MHz",
        "upload": {
            "maximum_size": "8192",
            "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",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino BT w/ ATmega168",
        "upload": {
            "disable_flushing": "true",
            "maximum_size": "14336",
            "protocol": "arduino",
            "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",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino BT w/ ATmega328",
        "upload": {
            "disable_flushing": "true",
            "maximum_size": "28672",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "Arduino Diecimila or Duemilanove w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "speed": "19200"
        }
    },
    "esplora": {
        "bootloader": {
            "extended_fuses": "0xcb",
            "file": "Caterina-Esplora.hex",
            "high_fuses": "0xd8",
            "lock_bits": "0x2F",
            "low_fuses": "0xff",
            "path": "caterina",
            "unlock_bits": "0x3F"
        },
        "build": {
            "core": "arduino",
            "f_cpu": "16000000L",
            "mcu": "atmega32u4",
            "pid": "0x803C",
            "variant": "leonardo",
            "vid": "0x2341"
        },
        "name": "Arduino Esplora",
        "upload": {
            "disable_flushing": "true",
            "maximum_size": "28672",
            "protocol": "avr109",
            "speed": "57600"
        }
    },
    "ethernet": {
        "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",
            "variant": "standard"
        },
        "name": "Arduino Ethernet",
        "upload": {
            "maximum_size": "32256",
            "protocol": "arduino",
            "speed": "115200"
        }
    },
    "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",
            "f_cpu": "8000000L",
            "mcu": "atmega328p",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino Fio",
        "upload": {
            "maximum_size": "30720",
            "protocol": "arduino",
            "speed": "57600"
        }
    },
    "leonardo": {
        "bootloader": {
            "extended_fuses": "0xcb",
            "file": "Caterina-Leonardo.hex",
            "high_fuses": "0xd8",
            "lock_bits": "0x2F",
            "low_fuses": "0xff",
            "path": "caterina",
            "unlock_bits": "0x3F"
        },
        "build": {
            "core": "arduino",
            "f_cpu": "16000000L",
            "mcu": "atmega32u4",
            "pid": "0x8036",
            "variant": "leonardo",
            "vid": "0x2341"
        },
        "name": "Arduino Leonardo",
        "upload": {
            "disable_flushing": "true",
            "maximum_size": "28672",
            "protocol": "avr109",
            "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",
            "variant": "standard"
        },
        "name": "LilyPad Arduino w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "LilyPad Arduino w/ ATmega328",
        "upload": {
            "maximum_size": "30720",
            "protocol": "arduino",
            "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",
            "variant": "mega"
        },
        "name": "Arduino Mega (ATmega1280)",
        "upload": {
            "maximum_size": "126976",
            "protocol": "arduino",
            "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",
            "variant": "mega"
        },
        "name": "Arduino Mega 2560 or Mega ADK",
        "upload": {
            "maximum_size": "258048",
            "protocol": "wiring",
            "speed": "115200"
        }
    },
    "micro": {
        "bootloader": {
            "extended_fuses": "0xcb",
            "file": "Caterina-Micro.hex",
            "high_fuses": "0xd8",
            "lock_bits": "0x2F",
            "low_fuses": "0xff",
            "path": "caterina",
            "unlock_bits": "0x3F"
        },
        "build": {
            "core": "arduino",
            "f_cpu": "16000000L",
            "mcu": "atmega32u4",
            "pid": "0x8037",
            "variant": "micro",
            "vid": "0x2341"
        },
        "name": "Arduino Micro",
        "upload": {
            "disable_flushing": "true",
            "maximum_size": "28672",
            "protocol": "avr109",
            "speed": "57600"
        }
    },
    "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",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino Mini w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "speed": "19200"
        }
    },
    "mini328": {
        "bootloader": {
            "extended_fuses": "0x05",
            "file": "optiboot_atmega328-Mini.hex",
            "high_fuses": "0xd8",
            "lock_bits": "0x0F",
            "low_fuses": "0xff",
            "path": "optiboot",
            "unlock_bits": "0x3F"
        },
        "build": {
            "core": "arduino",
            "f_cpu": "16000000L",
            "mcu": "atmega328p",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino Mini w/ ATmega328",
        "upload": {
            "maximum_size": "28672",
            "protocol": "arduino",
            "speed": "115200"
        }
    },
    "nano": {
        "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",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino Nano w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "speed": "19200"
        }
    },
    "nano328": {
        "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",
            "variant": "eightanaloginputs"
        },
        "name": "Arduino Nano w/ ATmega328",
        "upload": {
            "maximum_size": "30720",
            "protocol": "arduino",
            "speed": "57600"
        }
    },
    "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",
            "variant": "standard"
        },
        "name": "Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328",
        "upload": {
            "maximum_size": "30720",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168",
        "upload": {
            "maximum_size": "14336",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328",
        "upload": {
            "maximum_size": "30720",
            "protocol": "arduino",
            "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",
            "variant": "standard"
        },
        "name": "Arduino Uno",
        "upload": {
            "maximum_size": "32256",
            "protocol": "arduino",
            "speed": "115200"
        }
    }
}
```

Help:

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

print boards from boards.txt

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

7.2 List installed MCUs

From python:

```
>>> from confduino.mculist import mcus
>>> mcus()
['atmega1280', 'atmega168', 'atmega2560', 'atmega328p', 'atmega32u4', 'atmega8', 'atmega88']
```

From console:

```
$ python -m confduino.mculist
atmega1280
atmega168
atmega2560
atmega328p
atmega32u4
atmega8
atmega88
```

Help:

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

print boards from boards.txt

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

7.3 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

TEMPL = '{mcu}@{f_cpu} programmer:{upload}'


@entrypoint
def install(
    board_id='atmega88',
    mcu='atmega88',
    f_cpu=20000000,
    upload='usbasp',
    core='arduino',
    replace_existing=True,
):
    'install atmega88 board'

    board = AutoBunch()
    board.name = TEMPL.format(mcu=mcu, f_cpu=f_cpu, upload=upload)

    board.upload.using = upload
    board.upload.maximum_size = 8 * 1024

    board.build.mcu = mcu
    board.build.f_cpu = str(f_cpu) + 'L'
    board.build.core = core

    # for 1.0
    board.build.variant = 'standard'

    install_board(board_id, board, replace_existing=replace_existing)
```

console is not implemented

7.4 Remove existing board

From python:

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

From console:

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

Help:

```
$ 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
```

CHAPTER

EIGHT

USAGE WITH PROGRAMMERS

8.1 List installed programmers

From python:

```
>>> from confduino.proglist import programmers
>>> programmers()
AutoBunch(arduinoisp=AutoBunch(communication='serial', name='Arduino as ISP', protocol='stk500v1',
>>> programmers().arduinoisp.speed
'19200'
>>> programmers()['arduinoisp']['speed']
'19200'
```

From console:

```
$ python -m confduino.proglist
arduinoisp
avrisp
avrispmkii
dapa
parallel
stk200
usbasp
usbtinyisp
```

verbose (JSON compatible):

```
$ python -m confduino.proglist --verbose
{
    "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"
    },
    "dapa": {
        "force": "true",
        "name": "DAPA",
        "protocol": "dapa"
    },
    "parallel": {
        "force": "true",
        "name": "Parallel Programmer",
        "protocol": "dapa"
    },
    "stk200": {
        "name": "STK200",
        "protocol": "stk200"
    },
    "usbasp": {
        "communication": "usb",
        "name": "USBasp",
        "protocol": "usbasp"
    },
    "usbtinyisp": {
        "name": "USBtinyISP",
        "protocol": "usbtiny"
    }
}
```

Help:

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

print programmers from programmers.txt

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

8.2 Install new programmer

From python:

```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

8.3 Remove existing programmer

From python:

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

From console:

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

Help:

```
$ 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

NINE

EXAMPLES

9.1 Install libraries

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

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

Code:

```
from confduino import exampallcreate
from confduino.libinstall import install_lib
from confduino.util import ConfduinoError
from entrypoint2 import entrypoint


@entrypoint
def upgrade_many(upgrade=True, create_examples_all=True):
    '''upgrade many libs

    source: http://arduino.cc/playground/Main/LibraryList

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

    def inst(url):
        print 'upgrading ' + url
        assert url not in urls
        urls.add(url)
        try:
            lib = install_lib(url, upgrade)
            print ' -> ', lib
        except Exception as e:
            print e

    # inst('http://nootropicdesign.com/hackvision/downloads/Controllers.zip')

    ############################
    # github.com
    ############################
    inst('https://github.com/madsci1016/Arduino-EasyTransfer/zipball/master')
    inst('https://github.com/sparkfun/SevSeg/zipball/master')
    inst('https://github.com/madsci1016/Arduino-SoftEasyTransfer/zipball/master')
    inst('https://github.com/madsci1016/Arduino-PS2X/zipball/master')
#     inst('http://github.com/wimleers/flexitimer2/zipball/v1.0')# can't install
    inst('https://github.com/kerinin/arduino-splines/zipball/master')
    inst('https://github.com/asynclabs/WiShield/zipball/master')
    inst('https://github.com/asynclabs/dataflash/zipball/master')
    inst('https://github.com/slugmobile/AtTouch/zipball/master')
    inst('https://github.com/carlynorama/Arduino-Library-Button/zipball/master')
    inst('https://github.com/carlynorama/Arduino-Library-FancyLED/zipball/master')
    inst('https://github.com/markfickett/arduinomorse/zipball/master')
    inst('https://github.com/rocketscream/Low-Power/zipball/master')

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

    ############################
    # google
    ############################
    # TODO: how to get latest version??
    # parse http://code.google.com/p/arduino-pinchangeint/downloads/list
    inst('http://rogue-code.googlecode.com/files/Arduino-Library-Tone.zip')
         # simplified version in core
    inst('http://arduino-playground.googlecode.com/files/LedDisplay03.zip')
    inst('http://sserial2mobile.googlecode.com/files/SSerial2Mobile-1.1.0.zip')
    inst('http://webduino.googlecode.com/files/webduino-1.4.1.zip')
         # can't install
    inst('http://arduino-pid-library.googlecode.com/files/PID_v1.0.1.zip')
    inst('http://ideoarduinolibraries.googlecode.com/files/Qtouch1Wire.zip')
    inst('http://arduino-timerone.googlecode.com/files/TimerOne-v8.zip')
    inst('http://arduinounit.googlecode.com/files/arduinounit-1.4.2.zip')
    inst('http://arduinode.googlecode.com/files/arduinode_0.1.zip')
    inst('http://arduino-edb.googlecode.com/files/EDB_r7.zip')
    inst('http://arduino-dblib.googlecode.com/files/DB.zip')
    inst('http://morse-endecoder.googlecode.com/files/Morse_EnDecoder_2010.12.06.tar.gz')
    inst('http://arduino-pinchangeint.googlecode.com/files/PinChangeInt.zip')
    inst('http://arduino-tvout.googlecode.com/files/TVout_R5.91.zip')
    inst('http://narcoleptic.googlecode.com/files/Narcoleptic_v1a.zip')

    ############################
    # others
    ############################
    inst('http://download.milesburton.com/Arduino/MaximTemperature/DallasTemperature_370Beta.zip'
    inst('http://www.pjrc.com/teensy/arduino_libraries/OneWire.zip')
#    inst('http://www.state-machine.com/arduino/qp_arduino.zip') # too big
    inst('http://www.shikadi.net/files/arduino/SerialIP-1.0.zip')
    inst('http://siggiorn.com/wp-content/uploads/libraries/ArduinoByteBuffer.zip')
    inst('http://siggiorn.com/wp-content/uploads/libraries/ArduinoSerialManager.zip')
    inst('http://arduino-tweet.appspot.com/Library-Twitter-1.2.2.zip')
# inst('http://gkaindl.com/php/download.php?key=ArduinoEthernet')# can't
# install
    inst('http://sebastian.setz.name/wp-content/uploads/2011/01/multiCameraIrControl_1-5.zip')
    inst('http://www.pjrc.com/teensy/arduino_libraries/FrequencyTimer2.zip')
    inst('http://alexandre.quessy.net/static/avr/Tween_01.zip')
    inst('http://www.lpelettronica.it/images/stories/LPM11162_images/Arduino/LPM11162_ArduinoLib_

    ############################
    # arduino.cc
    ############################
    inst('http://arduino.cc/playground/uploads/Main/PS2Keyboard002.zip')
    inst('http://arduino.cc/playground/uploads/Code/Metro.zip')
    inst('http://www.arduino.cc/playground/uploads/Main/MsTimer2.zip')
# inst('http://www.arduino.cc/playground/uploads/Code/Time.zip')# can't
# install
    inst('http://arduino.cc/playground/uploads/Main/LedControl.zip')
# inst('http://www.arduino.cc/playground/uploads/Code/ks0108GLCD.zip')#
# can't install
    inst('http://arduino.cc/playground/uploads/Code/Bounce.zip')
    inst('http://arduino.cc/playground/uploads/Main/CapacitiveSense003.zip')
    inst('http://arduino.cc/playground/uploads/Main/PinChangeInt.zip')
# inst('http://arduino.cc/playground/uploads/Code/TimerThree.zip')# can't
# install
    inst('http://arduino.cc/playground/uploads/Code/TimedAction-1_6.zip')
# inst('http://www.arduino.cc/playground/uploads/Code/Time.zip')# can't
# install
    inst('http://arduino.cc/playground/uploads/Code/EventFuse.zip')
    inst('http://arduino.cc/playground/uploads/Code/Charlieplex.zip')
    inst('http://arduino.cc/playground/uploads/Code/DigitalToggle.zip')
    inst('http://arduino.cc/playground/uploads/Code/Enerlib.zip')

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

# inst('http://arduino.cc/playground/uploads/Code/SerialDebugger.zip') #
# can't install
    inst('http://arduino.cc/playground/uploads/Code/MatrixMath.zip')

    inst('http://arduino.cc/playground/uploads/Code/StackArray.zip')
    inst('http://arduino.cc/playground/uploads/Code/StackList.zip')
    inst('http://arduino.cc/playground/uploads/Code/QueueArray.zip')
    inst('http://arduino.cc/playground/uploads/Code/QueueList.zip')
    inst('http://arduino.cc/playground/uploads/Code/Ping-1_3.zip')
    inst('http://www.arduino.cc/playground/uploads/Code/LED.zip')

#     inst('')
    if create_examples_all:
        print 'create "all" menu item'
        exampallcreate.create_examples_all()
    print 'install finished'
```

9.2 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)
```

9.3 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 = 'stk200'
    # bunch.force = 'true'
    # bunch.delay=200

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

9.4 Install atmega88 board

```
python -m confduino.examples.atmega88
```

Code:

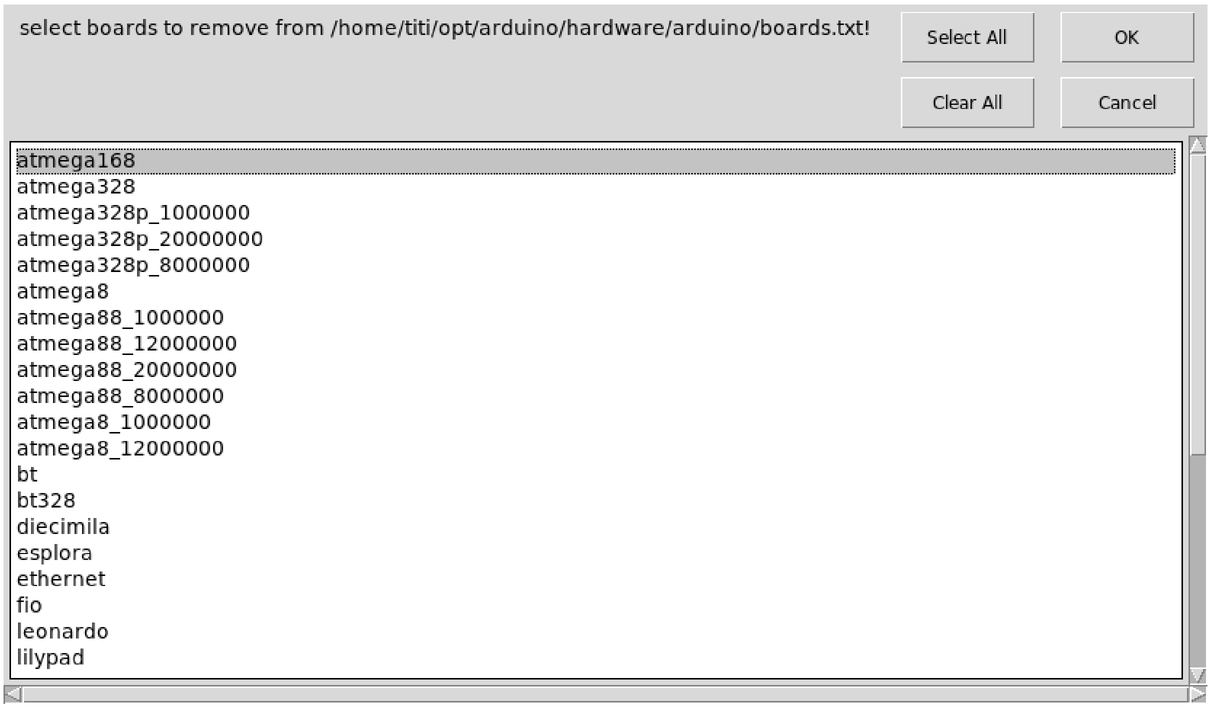
```
from confduino.boardinstall import install_board
from confduino.util import AutoBunch
from entrypoint2 import entrypoint

TEMPL = '{mcu}@{f_cpu} programmer:{upload}'


@entrypoint
def install(
    board_id='atmega88',
    mcu='atmega88',
    f_cpu=20000000,
    upload='usbasp',
    core='arduino',
    replace_existing=True,
):
    'install atmega88 board'

    board = AutoBunch()
    board.name = TEMPL.format(mcu=mcu, f_cpu=f_cpu, upload=upload)

    board.upload.using = upload
    board.upload.maximum_size = 8 * 1024

    board.build.mcu = mcu
    board.build.f_cpu = str(f_cpu) + 'L'
    board.build.core = core

    # for 1.0
    board.build.variant = 'standard'

    install_board(board_id, board, replace_existing=replace_existing)
```

options:

```
$ python -m confduino.examples.atmega88 --help
usage: atmega88.py [-h] [-b BOARD_ID] [-m MCU] [-f F_CPU] [-u UPLOAD]
                   [-c CORE] [-r] [--debug]

install atmega88 board

optional arguments:
  -h, --help            show this help message and exit
  -b BOARD_ID, --board-id BOARD_ID
  -m MCU, --mcu MCU
  -f F_CPU, --f-cpu F_CPU
  -u UPLOAD, --upload UPLOAD
  -c CORE, --core CORE
  -r, --replace-existing
  --debug               set logging level to DEBUG
```

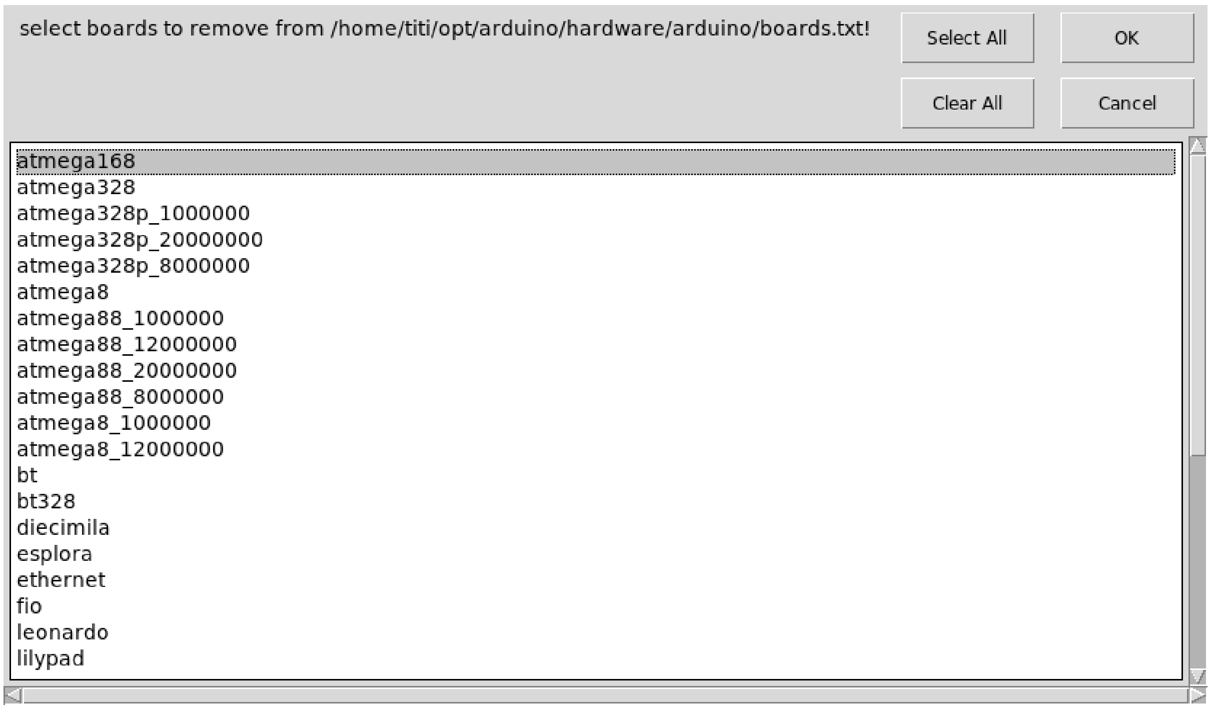
9.5 remove boards

```
$ python -m confduino.examples.remove_boards
```


select boards to remove from /home/titi/opt/arduino/hardware/arduino/boards.txt!

```
atmega168
atmega328
atmega328p_1000000
atmega328p_20000000
atmega328p_8000000
atmega8
atmega88_1000000
atmega88_12000000
atmega88_20000000
atmega88_8000000
atmega8_1000000
atmega8_12000000
bt
bt328
diecimila
esplora
ethernet
fio
leonardo
lilypad
```

```
$ python -m confduino.examples.remove_boards --hwpack arduino
```



select boards to remove from /home/titi/opt/arduino/hardware/arduino/boards.txt!

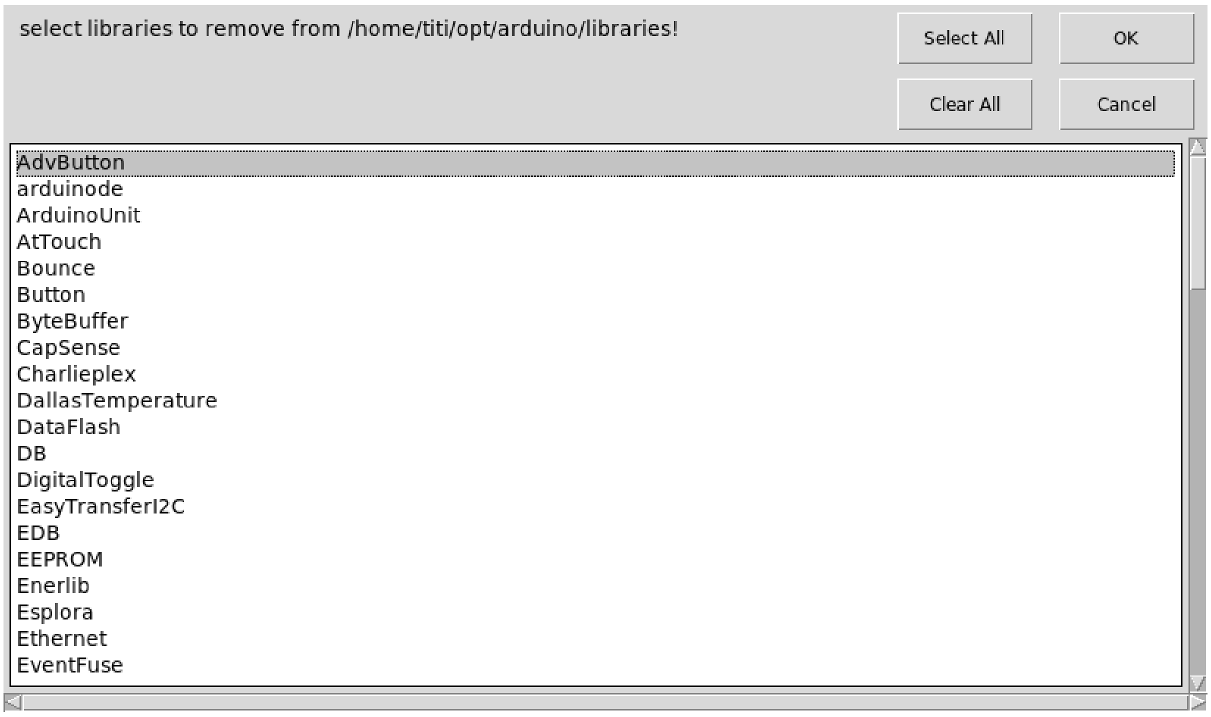
```
atmega168
atmega328
atmega328p_1000000
atmega328p_20000000
atmega328p_8000000
atmega8
atmega88_1000000
atmega88_12000000
atmega88_20000000
atmega88_8000000
atmega8_1000000
atmega8_12000000
bt
bt328
diecimila
esplora
ethernet
fio
leonardo
lilypad
```

Code:

```python
from confduino.boardlist import boards, boards_txt, board_names
from confduino.boardremove import remove_board
from confduino.hwpacklist import hwpack_names
from entrypoint2 import entrypoint
import psidialogs


@entrypoint
def remove_boards_gui(hwpack=''):
    'remove boards by GUI'
    if not hwpack:
        if len(hwpack_names()) > 1:
            hwpack = psidialogs.choice(hwpack_names(),
                                       'select hardware package to select board from!',
                                       title='select')
        else:
            hwpack = hwpack_names()[0]
    print hwpack, 'selected'

    if hwpack:
        sel = psidialogs.multi_choice(board_names(hwpack),
                                      'select boards to remove from %s!' % boards_txt(hwpack),
                                      title='remove boards')
        print sel, 'selected'

        if sel:
            for x in sel:
                remove_board(x)
                print x + ' was removed'
```

9.6 remove libraries

```
$ python -m confduino.examples.remove_libraries
```


select libraries to remove from /home/titi/opt/arduino/libraries!

```
AdvButton
arduinode
ArduinoUnit
AtTouch
Bounce
Button
ByteBuffer
CapSense
Charlieplex
DallasTemperature
DataFlash
DB
DigitalToggle
EasyTransferI2C
EDB
EEPROM
Enerlib
Esplora
Ethernet
EventFuse
```

Code:

```
from confduino.liblist import libraries, libraries_dir
from confduino.libremove import remove_lib
from entrypoint2 import entrypoint
import psidialogs


@entrypoint
def gui():
    'remove libraries by GUI'

    sel = psidialogs.multi_choice(libraries(),
                                  'select libraries to remove from %s!' % libraries_dir(),
                                  title='remove boards')
    print sel, 'selected'

    if sel:
        if psidialogs.ask_yes_no('Do you really want to remove selected libraries?\n' + '\n'.join
            for x in sel:
                remove_lib(x)
                print x + ' was removed'
```

CHAPTER

TEN

API

10.1 lib

`confduino.liblist.`**`lib_dir`**`(`*lib*`)`
return library directory

$ARDUINO/libraries/$LIB

`confduino.liblist.`**`lib_example_dir`**`(`*lib*, *example*`)`
return library example directory

$ARDUINO/libraries/$LIB/examples/$EXAMPLE

`confduino.liblist.`**`lib_examples`**`(`*lib*`)`
return library examples

EXAMPLE1,EXAMPLE2,..

`confduino.liblist.`**`lib_examples_dir`**`(`*lib*`)`
return library examples directory

$ARDUINO/libraries/$LIB/examples

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

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

$ARDUINO/libraries

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

`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*, *fix_wprogram=True*`)`
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

10.2 board

`confduino.boardlist.`**`board_names`**(*hwpack='arduino'*)
return installed board names

`confduino.boardlist.`**`boards`**(*hwpack='arduino'*)
read boards from boards.txt

Parameters **core_package** – 'all,'arduino',..

`confduino.boardlist.`**`boards_txt`**(*hwpack='arduino'*)
path of boards.txt

`confduino.boardlist.`**`print_boards`**(*hwpack='arduino'*, *verbose=False*)
print boards from boards.txt

`confduino.boardinstall.`**`install_board`**(*board_id*, *board_options*, *hwpack='arduino'*, *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

10.3 programmer

`confduino.proglist.`**`print_programmers`**(*verbose=False*)
print programmers from programmers.txt

`confduino.proglist.`**`programmer_names`**(*hwpack='arduino'*)
return installed board names

`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

10.4 version

`confduino.version.`**`all_sketch_extensions`**()
['.pde','.ino']

`confduino.version.`**`intversion`**(*text=None*)
return version as int

0022 -> 22 0022ubuntu0.1 -> 22 0023 -> 23 1.0 -> 100 1.0.3 -> 103

`confduino.version.`**`print_version`**(*integer=False*)
print arduino version

example: 0022

`confduino.version.`**`sketch_extension`**()
.pde or .ino

`confduino.version.`**`version`**()
return version as string

example: 0022

`confduino.version.`**`version_txt`**()
return version.txt path

$ARDUINO/lib/version.txt

CHAPTER

ELEVEN

INDICES AND TABLES

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

PYTHON MODULE INDEX

INDEX