

---

# **pyavrutils Documentation**

***Release 0.1.1***

**ponty**

December 02, 2012

# CONTENTS

<b>1</b>	<b>Basic usage</b>	<b>2</b>
<b>2</b>	<b>Installation</b>	<b>3</b>
2.1	General . . . . .	3
2.2	Ubuntu . . . . .	3
2.3	Uninstall . . . . .	3
<b>3</b>	<b>Usage</b>	<b>4</b>
3.1	AVR . . . . .	4
3.2	arduino . . . . .	4
<b>4</b>	<b>Examples</b>	<b>6</b>
4.1	Simple example . . . . .	6
4.2	Test size with unused code . . . . .	7
4.3	Test size with delay.h . . . . .	8
4.4	Test size with program space . . . . .	9
4.5	Test minimum size . . . . .	10
<b>5</b>	<b>Arduino build tests</b>	<b>15</b>
5.1	Results . . . . .	15
<b>6</b>	<b>API</b>	<b>16</b>
<b>7</b>	<b>Development</b>	<b>18</b>
7.1	Tools . . . . .	18
7.2	Install on ubuntu . . . . .	18
7.3	Tasks . . . . .	18
<b>8</b>	<b>Indices and tables</b>	<b>20</b>
	<b>Index</b>	<b>21</b>



## pyavrutils

**Date** December 02, 2012

**PDF** [pyavrutils.pdf](#)

Contents:

pyavrutils can build [AVR](#) and [arduino](#) code from [python](#)

### Links:

- home: <https://github.com/ponty/pyavrutils>
- documentation: <http://ponty.github.com/pyavrutils>

### Features:

- python wrapper for avr-gcc, avr-size, [arscons](#)
- build files or strings (strings are saved as temp files)
- MCU list
- get code size using avr-size
- avr-gcc default is optimized for size

### Known problems:

- Python 3 is not supported
- temp files are not removed
- [arscons](#) has some problems:
  - it builds bigger programs
  - compile error in some cases

### Possible usage:

- experimenting with flags
- building from [paver](#)
- unit tests
- building [arduino](#) code without GUI

# BASIC USAGE

```
>>> from pyavrutils import AvrGcc
>>> cc = AvrGcc()
>>> cc.build('int main(){}')
>>> cc.size().program_bytes
66

>>> from pyavrutils import Arduino
>>> cc = Arduino()
>>> cc.mcu = 'atmega8'
>>> cc.build('void setup(){};void loop(){}')
>>> cc.size().program_bytes
1612
```

# INSTALLATION

## 2.1 General

- `arscons` is already included in the library
- install `pip`
- install `gcc-avr`
- install `scons` (only for `arscons`)
- install `arduino` (only for `arscons`)
- install the program:

if you have `setuptools` installed:

```
# as root
pip install pyavrutils
```

## 2.2 Ubuntu

```
sudo apt-get install python-pip
sudo apt-get install binutils-avr
sudo apt-get install gcc-avr
sudo apt-get install scons
sudo apt-get install arduino
sudo pip install pyavrutils
```

## 2.3 Uninstall

using `pip`:

```
# as root
pip uninstall pyavrutils
```

# USAGE

## 3.1 AVR

```
>>> from pyavrutils import AvrGcc
>>> cc = AvrGcc(mcu='atmega48')
>>> cc.targets
[u'avr1', u'avr2', u'avr25', u'avr3', u'avr31', u'avr35', u'avr4', u'avr5', u'avr51', u'avr6', u'avr7', u'avr8', u'avr10', u'avr11', u'avr12', u'avr13', u'avr15', u'avr16', u'avr18', u'avr20', u'avr23', u'avr24', u'avr28', u'avr30', u'avr32', u'avr33', u'avr34', u'avr36', u'avr38', u'avr40', u'avr44', u'avr45', u'avr48', u'avr50', u'avr52', u'avr54', u'avr55', u'avr56', u'avr58', u'avr59', u'avr60', u'avr63', u'avr64', u'avr68', u'avr70', u'avr72', u'avr73', u'avr74', u'avr75', u'avr76', u'avr77', u'avr78', u'avr79', u'avr80', u'avr83', u'avr85', u'avr88', u'avr90', u'avr93', u'avr94', u'avr95', u'avr96', u'avr99', u'avr100', u'avr102', u'avr104', u'avr105', u'avr106', u'avr108', u'avr110', u'avr112', u'avr113', u'avr114', u'avr115', u'avr116', u'avr117', u'avr118', u'avr119', u'avr120', u'avr122', u'avr123', u'avr124', u'avr125', u'avr126', u'avr128', u'avr129', u'avr130', u'avr132', u'avr135', u'avr136', u'avr138', u'avr140', u'avr143', u'avr144', u'avr145', u'avr146', u'avr147', u'avr148', u'avr150', u'avr152', u'avr153', u'avr154', u'avr155', u'avr156', u'avr158', u'avr159', u'avr160', u'avr162', u'avr164', u'avr166', u'avr168', u'avr170', u'avr172', u'avr173', u'avr174', u'avr175', u'avr176', u'avr177', u'avr178', u'avr179', u'avr180', u'avr183', u'avr185', u'avr188', u'avr190', u'avr193', u'avr194', u'avr195', u'avr196', u'avr199', u'avr200', u'avr202', u'avr204', u'avr205', u'avr206', u'avr208', u'avr210', u'avr212', u'avr213', u'avr214', u'avr215', u'avr216', u'avr217', u'avr218', u'avr219', u'avr220', u'avr222', u'avr223', u'avr224', u'avr225', u'avr226', u'avr228', u'avr229', u'avr230', u'avr232', u'avr235', u'avr236', u'avr238', u'avr240', u'avr243', u'avr244', u'avr245', u'avr246', u'avr247', u'avr248', u'avr250', u'avr252', u'avr253', u'avr254', u'avr255', u'avr256', u'avr258', u'avr259', u'avr260', u'avr262', u'avr264', u'avr266', u'avr268', u'avr270', u'avr272', u'avr273', u'avr274', u'avr275', u'avr276', u'avr277', u'avr278', u'avr279', u'avr280', u'avr283', u'avr285', u'avr288', u'avr290', u'avr293', u'avr294', u'avr295', u'avr296', u'avr299', u'avr300', u'avr302', u'avr304', u'avr305', u'avr306', u'avr308', u'avr310', u'avr312', u'avr313', u'avr314', u'avr315', u'avr316', u'avr317', u'avr318', u'avr319', u'avr320', u'avr322', u'avr323', u'avr324', u'avr325', u'avr326', u'avr328', u'avr329', u'avr330', u'avr332', u'avr335', u'avr336', u'avr338', u'avr340', u'avr343', u'avr344', u'avr345', u'avr346', u'avr347', u'avr348', u'avr350', u'avr352', u'avr353', u'avr354', u'avr355', u'avr356', u'avr358', u'avr359', u'avr360', u'avr362', u'avr364', u'avr366', u'avr368', u'avr370', u'avr372', u'avr373', u'avr374', u'avr375', u'avr376', u'avr377', u'avr378', u'avr379', u'avr380', u'avr383', u'avr385', u'avr388', u'avr390', u'avr393', u'avr394', u'avr395', u'avr396', u'avr399', u'avr400', u'avr402', u'avr404', u'avr405', u'avr406', u'avr408', u'avr410', u'avr412', u'avr413', u'avr414', u'avr415', u'avr416', u'avr417', u'avr418', u'avr419', u'avr420', u'avr422', u'avr423', u'avr424', u'avr425', u'avr426', u'avr428', u'avr429', u'avr430', u'avr432', u'avr435', u'avr436', u'avr438', u'avr440', u'avr443', u'avr444', u'avr445', u'avr446', u'avr447', u'avr448', u'avr450', u'avr452', u'avr453', u'avr454', u'avr455', u'avr456', u'avr458', u'avr459', u'avr460', u'avr462', u'avr464', u'avr466', u'avr468', u'avr470', u'avr472', u'avr473', u'avr474', u'avr475', u'avr476', u'avr477', u'avr478', u'avr479', u'avr480', u'avr483', u'avr485', u'avr488', u'avr490', u'avr493', u'avr494', u'avr495', u'avr496', u'avr499', u'avr500', u'avr502', u'avr504', u'avr505', u'avr506', u'avr508', u'avr510', u'avr512', u'avr513', u'avr514', u'avr515', u'avr516', u'avr517', u'avr518', u'avr519', u'avr520', u'avr522', u'avr523', u'avr524', u'avr525', u'avr526', u'avr528', u'avr529', u'avr530', u'avr532', u'avr535', u'avr536', u'avr538', u'avr540', u'avr543', u'avr544', u'avr545', u'avr546', u'avr547', u'avr548', u'avr550', u'avr552', u'avr553', u'avr554', u'avr555', u'avr556', u'avr558', u'avr559', u'avr560', u'avr562', u'avr564', u'avr566', u'avr568', u'avr570', u'avr572', u'avr573', u'avr574', u'avr575', u'avr576', u'avr577', u'avr578', u'avr579', u'avr580', u'avr583', u'avr585', u'avr588', u'avr590', u'avr593', u'avr594', u'avr595', u'avr596', u'avr599', u'avr600', u'avr602', u'avr604', u'avr605', u'avr606', u'avr608', u'avr610', u'avr612', u'avr613', u'avr614', u'avr615', u'avr616', u'avr617', u'avr618', u'avr619', u'avr620', u'avr622', u'avr623', u'avr624', u'avr625', u'avr626', u'avr628', u'avr629', u'avr630', u'avr632', u'avr635', u'avr636', u'avr638', u'avr640', u'avr643', u'avr644', u'avr645', u'avr646', u'avr647', u'avr648', u'avr650', u'avr652', u'avr653', u'avr654', u'avr655', u'avr656', u'avr658', u'avr659', u'avr660', u'avr662', u'avr664', u'avr666', u'avr668', u'avr670', u'avr672', u'avr673', u'avr674', u'avr675', u'avr676', u'avr677', u'avr678', u'avr679', u'avr680', u'avr683', u'avr685', u'avr688', u'avr690', u'avr693', u'avr694', u'avr695', u'avr696', u'avr699', u'avr700', u'avr702', u'avr704', u'avr705', u'avr706', u'avr708', u'avr710', u'avr712', u'avr713', u'avr714', u'avr715', u'avr716', u'avr717', u'avr718', u'avr719', u'avr720', u'avr722', u'avr723', u'avr724', u'avr725', u'avr726', u'avr728', u'avr729', u'avr730', u'avr732', u'avr735', u'avr736', u'avr738', u'avr740', u'avr743', u'avr744', u'avr745', u'avr746', u'avr747', u'avr748', u'avr750', u'avr752', u'avr753', u'avr754', u'avr755', u'avr756', u'avr758', u'avr759', u'avr760', u'avr762', u'avr764', u'avr766', u'avr768', u'avr770', u'avr772', u'avr773', u'avr774', u'avr775', u'avr776', u'avr777', u'avr778', u'avr779', u'avr780', u'avr783', u'avr785', u'avr788', u'avr790', u'avr793', u'avr794', u'avr795', u'avr796', u'avr799', u'avr800', u'avr802', u'avr804', u'avr805', u'avr806', u'avr808', u'avr810', u'avr812', u'avr813', u'avr814', u'avr815', u'avr816', u'avr817', u'avr818', u'avr819', u'avr820', u'avr822', u'avr823', u'avr824', u'avr825', u'avr826', u'avr828', u'avr829', u'avr830', u'avr832', u'avr835', u'avr836', u'avr838', u'avr840', u'avr843', u'avr844', u'avr845', u'avr846', u'avr847', u'avr848', u'avr850', u'avr852', u'avr853', u'avr854', u'avr855', u'avr856', u'avr858', u'avr859', u'avr860', u'avr862', u'avr864', u'avr866', u'avr868', u'avr870', u'avr872', u'avr873', u'avr874', u'avr875', u'avr876', u'avr877', u'avr878', u'avr879', u'avr880', u'avr883', u'avr885', u'avr888', u'avr890', u'avr893', u'avr894', u'avr895', u'avr896', u'avr899', u'avr900', u'avr902', u'avr904', u'avr905', u'avr906', u'avr908', u'avr910', u'avr912', u'avr913', u'avr914', u'avr915', u'avr916', u'avr917', u'avr918', u'avr919', u'avr920', u'avr922', u'avr923', u'avr924', u'avr925', u'avr926', u'avr928', u'avr929', u'avr930', u'avr932', u'avr935', u'avr936', u'avr938', u'avr940', u'avr943', u'avr944', u'avr945', u'avr946', u'avr947', u'avr948', u'avr950', u'avr952', u'avr953', u'avr954', u'avr955', u'avr956', u'avr958', u'avr959', u'avr960', u'avr962', u'avr964', u'avr966', u'avr968', u'avr970', u'avr972', u'avr973', u'avr974', u'avr975', u'avr976', u'avr977', u'avr978', u'avr979', u'avr980', u'avr983', u'avr985', u'avr988', u'avr990', u'avr993', u'avr994', u'avr995', u'avr996', u'avr999', u'avr1000']
>>> cc.options_generated()
['avr-gcc', '-Df_cpu=4000000', '-mmcu=atmega48', '--std=gnu99', '-Wl,--relax', '-Wl,--gc-sections']
>>> cc.build('int main(){}')
>>> cc.output
/tmp/pyavrutils_MmQGpR.elf
>>> cc.size()
AvrSize <prog:80 bytes 2.0% mem:0 bytes 0.0% >
>>> cc.size().program_bytes
80
>>> cc.mcu='atmega168'
>>> cc.options_generated()
['avr-gcc', '-Df_cpu=4000000', '-mmcu=atmega168', '--std=gnu99', '-Wl,--relax', '-Wl,--gc-sections']
>>> cc.build('int main(){}')
>>> cc.output
/tmp/pyavrutils_MmQGpR.elf
>>> cc.size().program_bytes
132
```

## 3.2 arduino

```
>>> from pyavrutils import Arduino
>>> cc = Arduino(board='mini')
>>> cc.build('void setup(){};void loop(){}')
>>> cc.output
path('/tmp/pyavrutils_vWgLRa/pyavrutils_mEakND/pyavrutils_mEakND.elf')
>>> cc.size()
AvrSize <prog:430 bytes 2.6% mem:9 bytes 0.9% >
>>> cc.size().program_bytes
430
>>> cc.board='pro'
>>> cc.build('void setup(){};void loop(){}')
>>> cc.output
path('/tmp/pyavrutils_jDtsht/pyavrutils_oT3g4E/pyavrutils_oT3g4E.elf')
>>> cc.size().program_bytes
454
>>> cc.warnings
[u'build/core/IPAddress.h:51:55: warning: dereferencing type-punned pointer will break strict-aliasing']
```

display warnings on console:

```
$ python -m pyavrutils.cli.arduino.build /usr/share/arduino/examples/4.Communication/Dimmer/Dimmer.ino
backend: arcons
MCU: atmega168
avr-gcc: 4.5.3
```

```
=====
SIZE
=====
```

```
program: 2400
data: 192
```

```
=====
WARNINGS
=====
```

```
core
```

```
-----
```

```
build/core/IPAddress.h:51:55: warning: dereferencing type-punned pointer will break strict-aliasing
build/core/IPAddress.h:52:108: warning: dereferencing type-punned pointer will break strict-aliasing
build/core/IPAddress.h:52:75: warning: dereferencing type-punned pointer will break strict-aliasing
build/core/Tone.cpp:108:45: warning: only initialized variables can be placed into program memory
```

```
lib
```

```
-----
```

```
sketch
```

```
-----
```



# EXAMPLES

## 4.1 Simple example

Example program:

```
'''
test minimum program size with different optimizations
'''

from pyavrutils import AvrGcc
from entrypoint2 import entrypoint

cc = AvrGcc()
code = 'int main(){}'

def test():
    print '    compiler option:', ' '.join(cc.options_generated())
    cc.build(code)
    print '    program size =', cc.size().program_bytes

@entrypoint
def main():
    print 'compiler version:', cc.version()
    print 'code:', code
    print
    print 'no optimizations::'
    print
    cc.optimize_no()
    test()
    print
    print 'optimize for size::'
    print
    cc.optimize_for_size()
    test()
```

Output:

```
$ python -m pyavrutils.examples.simple
compiler version: 4.5.3
code: int main(){}

no optimizations::

    compiler option: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99
    program size = 150

optimize for size::
```

```
compiler option: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99 -Wl,--relax -Wl,--gc-sections
program size = 132
```

## 4.2 Test size with unused code

Example program:

```
from pyavrutils.avrgcc import AvrGcc
from entrypoint2 import entrypoint

cc = AvrGcc()

def test_option(sources, optimization, gc_sections=0, ffunction_sections=0):
    print 'optimization =', optimization,
    print 'gc_sections =', gc_sections,
    print 'ffunction_sections =', ffunction_sections,
    print

    cc.optimization = optimization
    cc.gc_sections = gc_sections
    cc.ffunction_sections = ffunction_sections
    try:
        cc.build(sources)
        size = cc.size()
        print 'program, data =', str(size.program_bytes).rjust(8) , ',', str(size.data_bytes).rjust(8)
    except:
        print 'compile error'

def test(sources):
    print 'sources:', sources
    test_option(sources, 0)
    test_option(sources, 's', 0)
    test_option(sources, 's', 1)
    test_option(sources, 's', 1, 1)

@entrypoint
def main():
    cc.optimize_no()
    print 'compiler version:', cc.version()
    print 'compiler options:', ' '.join(cc.options_generated())
    print
    print 'minimum size'
    print 20 * '='
    test(['int main(){}'])

    print
    print 'unused function in separate file'
    print 40 * '='
    test(['int main(){}', 'int f(){return 2;}'])

    print
    print 'unused function in the same file'
    print 40 * '='
    test(['int main(){}; int f(){return 2;}'])
```

Output:

```
$ python -m pyavrutils.examples.deadcode
compiler version: 4.5.3
compiler options: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99
```

```
minimum size
=====
sources: ['int main(){}']
optimization = 0 gc_sections = 0 ffunction_sections = 0
program, data = 150 , 0
optimization = s gc_sections = 0 ffunction_sections = 0
program, data = 138 , 0
optimization = s gc_sections = 1 ffunction_sections = 0
program, data = 138 , 0
optimization = s gc_sections = 1 ffunction_sections = 1
program, data = 138 , 0

unused function in separate file
=====
sources: ['int main(){}', 'int f(){return 2;}']
optimization = 0 gc_sections = 0 ffunction_sections = 0
program, data = 168 , 0
optimization = s gc_sections = 0 ffunction_sections = 0
program, data = 144 , 0
optimization = s gc_sections = 1 ffunction_sections = 0
program, data = 138 , 0
optimization = s gc_sections = 1 ffunction_sections = 1
program, data = 138 , 0

unused function in the same file
=====
sources: ['int main(){}; int f(){return 2;}']
optimization = 0 gc_sections = 0 ffunction_sections = 0
program, data = 168 , 0
optimization = s gc_sections = 0 ffunction_sections = 0
program, data = 144 , 0
optimization = s gc_sections = 1 ffunction_sections = 0
program, data = 144 , 0
optimization = s gc_sections = 1 ffunction_sections = 1
program, data = 138 , 0
```

#### Conclusions:

- both `gc_sections` and `ffunction_sections` should be used

## 4.3 Test size with delay.h

Example program:

```
from entrypoint2 import entrypoint
from pyavrutils.avrgcc import AvrGcc, AvrGccCompileError

templ = '''
#include <avr/io.h>
#include <util/delay.h>
int main()
{
    %s;
    return 0;
}
'''

cc = AvrGcc()
cc.optimize_no()
print 'compiler version:', cc.version()
print
```

```

def test(snippet, option=''):
    print snippet.ljust(33) ,
    cc.options_extra = option.split()
    print 'compiler option:', option, '\t',
    try:
        cc.build([templ % snippet])
        size = cc.size()
        print 'program, data =', str(size.program_bytes).rjust(8) , ',', str(size.data_bytes).rjust(8)
    except AvrGccCompileError as e:
        print 'compile error'

@entrypoint
def main():
    cc.optimization = 0

    test('_delay_ms(4)', '-O0')
    test('_delay_ms(4)', '-O1')
    test('_delay_ms(4)', '-O2')
    test('_delay_ms(4)', '-O3')
    test('_delay_ms(4)', '-Os')

    test('volatile int x=3;_delay_ms(x)', '-Os')

```

Output:

```

$ python -m pyavrutils.examples.delaysize
compiler version: 4.5.3

```

_delay_ms(4)	compiler option: -O0	program, data =	3266 ,
_delay_ms(4)	compiler option: -O1	program, data =	150 ,
_delay_ms(4)	compiler option: -O2	program, data =	150 ,
_delay_ms(4)	compiler option: -O3	program, data =	150 ,
_delay_ms(4)	compiler option: -Os	program, data =	150 ,
volatile int x=3;_delay_ms(x)	compiler option: -Os	compile error	

Conclusions:

- parameter should be constant
- optimization should be 1, 2, 3 or s

## 4.4 Test size with program space

Example program:

```

from pyavrutils.avrgcc import AvrGcc
from entrypoint2 import entrypoint

templ = '''
#include <avr/io.h>
#include <avr/pgmspace.h>
int main()
{
    %s;
    return 0;
}
'''

cc = AvrGcc()
cc.optimization=0
print 'compiler version:', cc.version()

```

```

print 'compiler options:', ' '.join(cc.options_generated())
print

def test(snippet):
    print snippet, '\t\t',
    try:
        cc.build([templ % snippet])
        size = cc.size()
        print 'program, data =', str(size.program_bytes).rjust(8), ',', str(size.data_bytes).rjust(8)
    except:
        print 'compile error'

def test_comb(s):
    words='static const PROGMEM'.split()
    def choice(i):
        return [words[i], ' '*len(words[i])]

    for s0 in choice(0):
        for s1 in choice(1):
            for s2 in choice(2):
#                 for s3 in choice(3):
                test('%s %s char s[] %s = "%s"' % (s0,s1,s2,s))

@entrypoint
def main():
    test_comb("12345")
    test_comb("1234512345")

```

Output:

```

$ python -m pyavrutils.examples.pgmspace
compiler version: 4.5.3
compiler options: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99 -Wl,--relax -Wl,--gc-sections

static const char s[] PROGMEM = "12345"           program, data =      144 ,         0
static const char s[]          = "12345"           program, data =      166 ,         0
static      char s[] PROGMEM = "12345"           program, data =      144 ,         0
static      char s[]          = "12345"           program, data =      166 ,         0
      const char s[] PROGMEM = "12345"           program, data =      220 ,         6
      const char s[]          = "12345"           program, data =      220 ,         6
            char s[] PROGMEM = "12345"           program, data =      220 ,         6
            char s[]          = "12345"           program, data =      220 ,         6
static const char s[] PROGMEM = "1234512345"       program, data =      144 ,         0
static const char s[]          = "1234512345"       program, data =      166 ,         0
static      char s[] PROGMEM = "1234512345"       program, data =      144 ,         0
static      char s[]          = "1234512345"       program, data =      166 ,         0
      const char s[] PROGMEM = "1234512345"       program, data =      232 ,        12
      const char s[]          = "1234512345"       program, data =      232 ,        12
            char s[] PROGMEM = "1234512345"       program, data =      232 ,        12
            char s[]          = "1234512345"       program, data =      232 ,        12

```

### Conclusions:

- constant string should be static or global
- const has no effect on size
- PROGMEM should be used

## 4.5 Test minimum size

Example program:

```
'''
test minimum program size with all MCUs
'''

from entrypoint2 import entrypoint
from pyavrutils.avrgcc import AvrGcc, AvrGccCompileError

def test(cc, mcu):
    print 'MCU =', mcu.ljust(20),
    cc.mcu = mcu
    try:
        cc.build(cc.minprog)
        print '    program/data size =', cc.size().program_bytes, ',', cc.size().data_bytes
    except AvrGccCompileError:
        print '    compile error'

@entrypoint
def main():
    cc = AvrGcc()
    print '-----'
    print 'avr-gcc'
    print '-----'

    print 'compiler version:', cc.version()
    cc.optimize_for_size()
    print 'compiler options:', ' '.join(cc.options_generated())
    print 'code:', cc.minprog
    print
    for mcu in cc.targets:
        test(cc, mcu)
```

Output:

```
$ python -m pyavrutils.examples.minsize
-----
avr-gcc
-----
compiler version: 4.5.3
compiler options: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99 -Wl,--relax -Wl,--gc-sections
code: int main(){};

MCU = avr1                compile error
MCU = avr2                program/data size = 0 , 0
MCU = avr25               program/data size = 0 , 0
MCU = avr3                program/data size = 0 , 0
MCU = avr31               program/data size = 0 , 0
MCU = avr35               program/data size = 0 , 0
MCU = avr4                program/data size = 0 , 0
MCU = avr5                program/data size = 0 , 0
MCU = avr51               program/data size = 0 , 0
MCU = avr6                program/data size = 0 , 0
MCU = avrxmega1           compile error
MCU = avrxmega2           program/data size = 0 , 0
MCU = avrxmega3           compile error
MCU = avrxmega4           program/data size = 0 , 0
MCU = avrxmega5           program/data size = 0 , 0
MCU = avrxmega6           program/data size = 0 , 0
MCU = avrxmega7           program/data size = 0 , 0
MCU = avrtiny10           program/data size = 0 , 0
MCU = at90s1200           compile error
MCU = attiny11            compile error
MCU = attiny12            compile error
MCU = attiny15            compile error
```

MCU = attiny28	compile error
MCU = at90s2313	program/data size = 46 , 0
MCU = at90s2323	program/data size = 30 , 0
MCU = at90s2333	program/data size = 52 , 0
MCU = at90s2343	program/data size = 30 , 0
MCU = attiny22	program/data size = 30 , 0
MCU = attiny26	program/data size = 48 , 0
MCU = at90s4414	program/data size = 54 , 0
MCU = at90s4433	program/data size = 52 , 0
MCU = at90s4434	program/data size = 62 , 0
MCU = at90s8515	program/data size = 54 , 0
MCU = at90c8534	program/data size = 42 , 0
MCU = at90s8535	program/data size = 62 , 0
MCU = attiny13	program/data size = 44 , 0
MCU = attiny13a	program/data size = 44 , 0
MCU = attiny2313	program/data size = 62 , 0
MCU = attiny2313a	program/data size = 66 , 0
MCU = attiny24	program/data size = 58 , 0
MCU = attiny24a	program/data size = 58 , 0
MCU = attiny4313	program/data size = 70 , 0
MCU = attiny44	program/data size = 62 , 0
MCU = attiny44a	program/data size = 62 , 0
MCU = attiny84	program/data size = 62 , 0
MCU = attiny84a	program/data size = 62 , 0
MCU = attiny25	program/data size = 54 , 0
MCU = attiny45	program/data size = 58 , 0
MCU = attiny85	program/data size = 58 , 0
MCU = attiny261	program/data size = 62 , 0
MCU = attiny261a	program/data size = 62 , 0
MCU = attiny461	program/data size = 66 , 0
MCU = attiny461a	program/data size = 66 , 0
MCU = attiny861	program/data size = 66 , 0
MCU = attiny861a	program/data size = 66 , 0
MCU = attiny87	program/data size = 68 , 0
MCU = attiny43u	program/data size = 60 , 0
MCU = attiny48	program/data size = 68 , 0
MCU = attiny88	program/data size = 68 , 0
MCU = at86rf401	program/data size = 40 , 0
MCU = ata6289	program/data size = 82 , 0
MCU = at43usb355	program/data size = 80 , 0
MCU = at76c711	program/data size = 88 , 0
MCU = atmega103	program/data size = 124 , 0
MCU = at43usb320	program/data size = 80 , 0
MCU = attiny167	program/data size = 108 , 0
MCU = at90usb82	program/data size = 144 , 0
MCU = at90usb162	program/data size = 144 , 0
MCU = atmega8u2	program/data size = 180 , 0
MCU = atmega16u2	program/data size = 180 , 0
MCU = atmega32u2	program/data size = 180 , 0
MCU = attiny1634	compile error
MCU = atmega8	program/data size = 66 , 0
MCU = atmega48	program/data size = 80 , 0
MCU = atmega48a	program/data size = 80 , 0
MCU = atmega48pa	compile error
MCU = atmega48p	program/data size = 80 , 0
MCU = atmega88	program/data size = 80 , 0
MCU = atmega88a	program/data size = 80 , 0
MCU = atmega88p	program/data size = 80 , 0
MCU = atmega88pa	program/data size = 80 , 0
MCU = atmega8515	program/data size = 62 , 0
MCU = atmega8535	program/data size = 70 , 0
MCU = atmega8hva	program/data size = 70 , 0
MCU = at90pwm1	program/data size = 92 , 0

MCU = at90pwm2	program/data size = 92 , 0
MCU = at90pwm2b	program/data size = 92 , 0
MCU = at90pwm3	program/data size = 92 , 0
MCU = at90pwm3b	program/data size = 92 , 0
MCU = at90pwm81	program/data size = 68 , 0
MCU = at90pwm161	compile error
MCU = atmega16	program/data size = 112 , 0
MCU = atmega16a	program/data size = 112 , 0
MCU = atmega161	program/data size = 112 , 0
MCU = atmega162	program/data size = 140 , 0
MCU = atmega163	program/data size = 100 , 0
MCU = atmega164a	program/data size = 152 , 0
MCU = atmega164p	program/data size = 152 , 0
MCU = atmega165	program/data size = 116 , 0
MCU = atmega165a	program/data size = 116 , 0
MCU = atmega165p	program/data size = 116 , 0
MCU = atmega168	program/data size = 132 , 0
MCU = atmega168a	program/data size = 132 , 0
MCU = atmega168p	program/data size = 132 , 0
MCU = atmega169	program/data size = 120 , 0
MCU = atmega169a	program/data size = 120 , 0
MCU = atmega169p	program/data size = 120 , 0
MCU = atmega169pa	program/data size = 120 , 0
MCU = atmega32	program/data size = 112 , 0
MCU = atmega323	program/data size = 108 , 0
MCU = atmega324a	program/data size = 152 , 0
MCU = atmega324p	program/data size = 152 , 0
MCU = atmega324pa	program/data size = 152 , 0
MCU = atmega325	program/data size = 120 , 0
MCU = atmega325a	program/data size = 120 , 0
MCU = atmega325p	program/data size = 120 , 0
MCU = atmega325pa	compile error
MCU = atmega3250	program/data size = 128 , 0
MCU = atmega3250a	program/data size = 128 , 0
MCU = atmega3250p	program/data size = 128 , 0
MCU = atmega3250pa	compile error
MCU = atmega328	program/data size = 132 , 0
MCU = atmega328p	program/data size = 132 , 0
MCU = atmega329	program/data size = 120 , 0
MCU = atmega329a	program/data size = 120 , 0
MCU = atmega329p	program/data size = 120 , 0
MCU = atmega329pa	program/data size = 120 , 0
MCU = atmega3290	program/data size = 128 , 0
MCU = atmega3290a	program/data size = 128 , 0
MCU = atmega3290p	program/data size = 128 , 0
MCU = atmega3290pa	compile error
MCU = atmega406	program/data size = 120 , 0
MCU = atmega64	program/data size = 168 , 0
MCU = atmega640	program/data size = 256 , 0
MCU = atmega644	program/data size = 140 , 0
MCU = atmega644a	program/data size = 152 , 0
MCU = atmega644p	program/data size = 152 , 0
MCU = atmega644pa	program/data size = 152 , 0
MCU = atmega645	program/data size = 120 , 0
MCU = atmega645a	program/data size = 120 , 0
MCU = atmega645p	program/data size = 120 , 0
MCU = atmega649	program/data size = 120 , 0
MCU = atmega649p	program/data size = 120 , 0
MCU = atmega649a	program/data size = 120 , 0
MCU = atmega6450	program/data size = 128 , 0
MCU = atmega6450a	program/data size = 128 , 0
MCU = atmega6450p	program/data size = 128 , 0
MCU = atmega6490	program/data size = 128 , 0



MCU = atmega6490a	program/data size = 128 , 0
MCU = atmega6490p	program/data size = 128 , 0
MCU = atmega64hve	program/data size = 128 , 0
MCU = atmega16hva	program/data size = 112 , 0
MCU = atmega16hva2	program/data size = 116 , 0
MCU = atmega16hvb	program/data size = 144 , 0
MCU = atmega16hvbrevb	program/data size = 144 , 0
MCU = atmega32hvb	program/data size = 144 , 0
MCU = atmega32hvbrevb	program/data size = 144 , 0
MCU = at90can32	program/data size = 176 , 0
MCU = at90can64	program/data size = 176 , 0
MCU = at90pwm216	program/data size = 156 , 0
MCU = at90pwm316	program/data size = 156 , 0
MCU = atmega32c1	program/data size = 152 , 0
MCU = atmega64c1	program/data size = 152 , 0
MCU = atmega16m1	program/data size = 152 , 0
MCU = atmega32m1	program/data size = 152 , 0
MCU = atmega64m1	program/data size = 152 , 0
MCU = atmega16u4	program/data size = 200 , 0
MCU = atmega32u4	program/data size = 200 , 0
MCU = atmega32u6	program/data size = 180 , 0
MCU = at90usb646	program/data size = 180 , 0
MCU = at90usb647	program/data size = 180 , 0
MCU = at90scr100	program/data size = 180 , 0
MCU = at94k	program/data size = 172 , 0
MCU = m3000	compile error
MCU = atmega128	program/data size = 168 , 0
MCU = atmega1280	program/data size = 256 , 0
MCU = atmega1281	program/data size = 232 , 0
MCU = atmega1284p	program/data size = 168 , 0
MCU = atmega128rfa1	program/data size = 316 , 0
MCU = at90can128	program/data size = 176 , 0
MCU = at90usb1286	program/data size = 180 , 0
MCU = at90usb1287	program/data size = 180 , 0
MCU = atmega2560	program/data size = 260 , 0
MCU = atmega2561	program/data size = 236 , 0
MCU = atxmega16a4	program/data size = 404 , 0
MCU = atxmega16d4	program/data size = 392 , 0
MCU = atxmega16x1	compile error
MCU = atxmega32a4	program/data size = 404 , 0
MCU = atxmega32d4	program/data size = 392 , 0
MCU = atxmega32x1	compile error
MCU = atxmega64a3	program/data size = 516 , 0
MCU = atxmega64d3	program/data size = 484 , 0
MCU = atxmega64a1	program/data size = 536 , 0
MCU = atxmega64a1u	program/data size = 548 , 0
MCU = atxmega128a3	program/data size = 520 , 0
MCU = atxmega128b1	compile error
MCU = atxmega128d3	program/data size = 488 , 0
MCU = atxmega192a3	program/data size = 520 , 0
MCU = atxmega192d3	program/data size = 488 , 0
MCU = atxmega256a3	program/data size = 520 , 0
MCU = atxmega256a3b	program/data size = 520 , 0
MCU = atxmega256a3bu	compile error
MCU = atxmega256d3	program/data size = 488 , 0
MCU = atxmega128a1	program/data size = 540 , 0
MCU = atxmega128a1u	program/data size = 552 , 0
MCU = attiny4	program/data size = 48 , 0
MCU = attiny5	program/data size = 50 , 0
MCU = attiny9	program/data size = 48 , 0
MCU = attiny10	program/data size = 50 , 0
MCU = attiny20	program/data size = 62 , 0
MCU = attiny40	program/data size = 62 , 0

# ARDUINO BUILD TESTS

Code:

```
void setup()
{
}

void loop()
{
}
```

## 5.1 Results

### 5.1.1 Arduino version 0022

index	board	min
1	atmega8	OK (P:312 D:9)
2	atmega48	OK (P:380 D:9)
3	atmega168	OK (P:440 D:9)
4	atmega328p	OK (P:440 D:9)
5	atmega640	OK (P:642 D:9)
6	atmega1280	OK (P:642 D:9)
7	atmega2560	OK (P:646 D:9)

### 5.1.2 Arduino version 1.0

index	board	min
8	atmega8	OK (P:324 D:9)
9	atmega48	OK (P:392 D:9)
10	atmega168	OK (P:454 D:9)
11	atmega328p	OK (P:454 D:9)
12	atmega640	OK (P:656 D:9)
13	atmega1280	OK (P:656 D:9)
14	atmega2560	OK (P:660 D:9)

# API

```
class pyavrutils.AvrGcc (mcu='atmega168')

    build (sources=None, headers=None)
        sources can be file name or code: sources=['x.c','int main(){}'] or sources='int main(){}'

    command_list (sources, _opt=False)
        command line as list

    error_text

    minprog = 'int main(){};'

    ok

    optimize_for_size ()
        http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=90752
        http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=69813

    optimize_no ()
        all options set to default

    options_generated ()

    size ()

    targets

    version ()
        avr-gcc version

class pyavrutils.AvrSize
    wrapper for avr-size

    ok

    parse_output (s)
        Example output:

        Device: atmega2561

        Program: 4168 bytes (1.6% Full) (.text + .data + .bootloader)

        Data: 72 bytes (0.9% Full) (.data + .bss + .noinit)

    run (objfile, mcu)

class pyavrutils.Arduino (board='pro', hwpack='arduino', mcu=None, f_cpu=None, ex-
                           tra_lib=None, ver=None, backend='arscons')
    wrapper for arscons

    build (sources=None)

    build_arscons (sources=None)
```

**build\_ino** (*sources=None*)  
**command\_list** ()  
**command\_list\_arscons** ()  
    command line as list  
**command\_list\_ino** ()  
**error\_text**  
**guess\_projname** (*allfiles*)  
**mcu\_compiler** ()  
**minprog** = 'void setup(){};void loop(){};'  
**ok**  
**setup\_sources** (*tempdir, sources*)  
**size** ()  
**stderr**  
**warnings**

# DEVELOPMENT

## 7.1 Tools

1. `setuptools`
2. `Paver`
3. `nose`
4. `ghp-import`
5. `pyflakes`
6. `pychecker`
7. `paved fork`
8. `Sphinx`
9. `sphinxcontrib-programsscreenshot`
10. `sphinxcontrib-paverutils`
11. `autorun` from `sphinx-contrib` (there is no simple method, you have to download/unpack/setup)

## 7.2 Install on ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install python-paver
sudo apt-get install python-nose
sudo easy_install ghp-import
sudo apt-get install pyflakes
sudo apt-get install pychecker
sudo easy_install https://github.com/ponty/paved/zipball/master
sudo apt-get install scrot
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install python-imaging
sudo apt-get install python-sphinx
sudo easy_install sphinxcontrib-programsscreenshot
sudo easy_install sphinxcontrib-programoutput
sudo easy_install sphinxcontrib-paverutils
```

## 7.3 Tasks

`Paver` is used for task management, settings are saved in `pavement.py`. `Sphinx` is used to generate documentation.

print [paver](#) settings:

```
paver printoptions
```

clean generated files:

```
paver clean
```

generate documentation under *docs/\_build/html*:

```
paver cog pdf html
```

upload documentation to [github](#):

```
paver ghpages
```

run unit tests:

```
paver nose  
#or  
nosetests --verbose
```

check python code:

```
paver pyflakes  
paver pychecker
```

generate python distribution:

```
paver sdist
```

upload python distribution to [PyPI](#):

```
paver upload
```

# INDICES AND TABLES

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

# INDEX

## A

Arduino (class in pyavrutils), 16  
AvrGcc (class in pyavrutils), 16  
AvrSize (class in pyavrutils), 16

## B

build() (pyavrutils.Arduino method), 16  
build() (pyavrutils.AvrGcc method), 16  
build\_arscons() (pyavrutils.Arduino method), 16  
build\_ino() (pyavrutils.Arduino method), 16

## C

command\_list() (pyavrutils.Arduino method), 17  
command\_list() (pyavrutils.AvrGcc method), 16  
command\_list\_arscons() (pyavrutils.Arduino method), 17  
command\_list\_ino() (pyavrutils.Arduino method), 17

## E

error\_text (pyavrutils.Arduino attribute), 17  
error\_text (pyavrutils.AvrGcc attribute), 16

## G

guess\_projname() (pyavrutils.Arduino method), 17

## M

mcu\_compiler() (pyavrutils.Arduino method), 17  
minprog (pyavrutils.Arduino attribute), 17  
minprog (pyavrutils.AvrGcc attribute), 16

## O

ok (pyavrutils.Arduino attribute), 17  
ok (pyavrutils.AvrGcc attribute), 16  
ok (pyavrutils.AvrSize attribute), 16  
optimize\_for\_size() (pyavrutils.AvrGcc method), 16  
optimize\_no() (pyavrutils.AvrGcc method), 16  
options\_generated() (pyavrutils.AvrGcc method), 16

## P

parse\_output() (pyavrutils.AvrSize method), 16

## R

run() (pyavrutils.AvrSize method), 16

## S

setup\_sources() (pyavrutils.Arduino method), 17  
size() (pyavrutils.Arduino method), 17  
size() (pyavrutils.AvrGcc method), 16  
stderr (pyavrutils.Arduino attribute), 17

## T

targets (pyavrutils.AvrGcc attribute), 16

## V

version() (pyavrutils.AvrGcc method), 16

## W

warnings (pyavrutils.Arduino attribute), 17