(/)

Search ...

EPSON
EXCEED YOUR VISION

EPSON HOME THEATRE PROJECTORS
FOR A BIG SCREEN **SPORTS** EXPERIENCE.

plesk powered by amazon web services

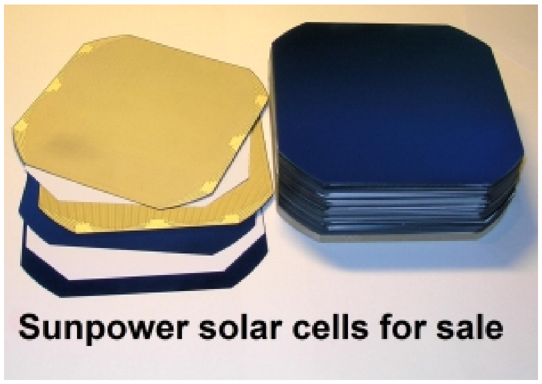
Get up to $500
Build with **Plesk**
Scale on **AWS**

Try now

Most recent articles

- OsmAnd getest (/gps/osmand-getest.html)
- Fietsnavigatie getest (/gps/fietsnavigatie.html)
- How to safely test the Panasonic microwave oven inverter HV power supply (/tech-tips/panasonic-hv-psu.html)
- Tektronix 465, 475, 475A spare parts for repair (/div/tektronix-465-parts.html)
- Cheerson CX-10WD mini RC quadcopter teardown (/div/quadcopter.html)
- Heartbeat with diagnostic blink LED error codes (/arduino/hardware-interfacing/heartbeat.html)

- Spot welder controller for the Kende DN-100E (Harbor Freight) prevents blown fuses (/arduino/hardware-interfacing/kende-spot-welder.html)
- Changing Arduino IDE theme colors (/arduino/arduino-ide.html)
- Fast PWM-DAC library for the SAM15x15 and Arduino Zero (/arduino/hardware-interfacing/samd21-pwm-dac.html)
- SAMD21 Timer library for the SAM15x15 and Arduino Zero (/arduino/libraries/samd21-timer.html)



(/component/banners/click/19.html)

See my new video of the solarbike
Please share it with your friends!

(/component/banners/click/25.html)

Arduino tips and troubleshooting

Details
C Last Updated: 25 February 2017

Missing libraries

Most problems arise because of missing libraries. For instance when the library "Timer1" is missing, we receive the compiler message: "'Timer1' was not declared in this scope". I use many libraries, those of others and of myself, and it's a lot of work to install them all. Therefore I have collected all those libraries in a zip file that you can download HERE (/Article_files/Arduino/Tips/libraries.zip). Sometimes I forget to update the zip file so it can happen that something is missing, please let me know that.

Troubleshooting if Arduino applications don't work

My Arduino applications and libraries are tested with the Arduino software version used at the time of development. After installation of a new Arduino software version (http://arduino.cc/en/Main/Software), problems may arise with existing projects. I don't have time to regularly test all my software on newer Arduino versions, so let me know if problems occur.

To solve a problem, read this page, and first check if the Arduino works well with some simple sample projects. Normally, by building up the application step by step, you will find out what the problem is.
Don't forget to select the right board: Tools > Board > and the serial port: Tools > Serial Port.

How to install Arduino libraries

Look at a library in the folder \libraries\ and see that a library is just a folder with the c++ code file (.cpp) and a header (.h) file. A library will be included in a project with #include <libraryBlaBla.h>. If the library is included in the project folder itself, in a tab, than use #include "libraryBlaBla.h". To convert an Arduino tab into a library file, change the extension from .ino to .cpp.
Arduino doesn't accept file and folder names with a dash **(-)**, use the underscore **(_)** instead.

I started adding my libraries on GitHub to make downloading and installation easier.

Make a copy of the preference.txt file.

Sometimes the Arduino software doesn't work anymore, at least on Windows XP; the preference.txt file may be corrupt. Put the old preference.txt file back and try again. If this doesn't work, the computer has to be start up again. For Windows XP `c:\Documents and Settings\\Application Data\Arduino\preferences.txt.`

Package installation

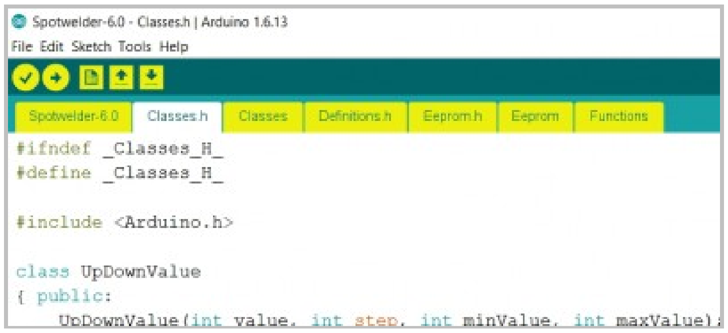
To use new Arduino boards, such as the Arduino Zero, we have to install an appropriate package, which describes the processor and pins. See here the package installation procedure (/arduino/samd21/sam-15x15-installation.html).

Adding a board to the boards manager list

If a boards is not already in the boards manager list, it should be added.
Look for the board in the Unofficial list of 3rd party boards support urls (https://github.com/arduino/Arduino/wiki/Unofficial-list-of-3rd-party-boards-support-urls) and copy the URL json.
File > Preferences > paste the URL in the field "Additional Boards Manager URLS".

Changing the Arduino IDE colors

Some colors of the Arduino development environment are awkward and make text almost unreadable. See here how you (/arduino/arduino-ide.html) can change that.


Spotwelder-6.0 - Classes.h | Arduino 1.6.13
File Edit Sketch Tools Help
Spotwelder-6.0
Classes.h
Classes
Definitions.h
Eeprom.h
Eeprom
Functions

```
#ifndef _Classes_H_
#define _Classes_H_

#include <Arduino.h>

class UpDownValue
{ public:
    UpDownValue(int value, int step, int minValue, int maxValue);
```

Improved Arduino colors for better visibility (/Article_files/Arduino/Tips/Arduino-tabs-ebef0b.jpg)

Troubleshooting with installing new Arduino software versions

After the installation of a new Arduino software version, problems arise often with existing projects.
Compiler differences and missing or incompatible libraries may cause compiler errors like these:

```
'blabla' was not declared in this scope
'blabla' does not name a type
expected ',' or '...' before 'blabla'
ISO C++ forbids declaration of 'blabla' with no type
```

See here what to do:

- Libraries can't be found
  Installed additional libraries are not moved to the new Arduino environment, they have to be installed again.
- Incompatible libraries
  Older additional libraries may be not compatible with newer Arduino software versions, so update the libraries too.
- Compiler differences
  See further.

Arduino problems on Windows XP

Running Arduino on Windows XP may give compiler messages like this:

```
collect2.exe: error: ld returned 53 exit status
```

or this:

```
collect2.exe: error: ld returned 5 exit status
exit status 1
```

This problem can be solved by replacing ld.exe by ld.exe of version 1.0.5, download ld.exe (/Article_files/Arduino/Tips/ld.exe).

```
Path c:\Arduino\arduino-1.6.3\hardware\tools\avr\avr\bin\
```

Arduino.h and WProgram.h

For recent Arduino versions (since 1.0) we have to use:

```
#include "Arduino.h"
```

instead of

```
#include "WProgram.h"
```

This change was not handy because all old software doesn't work anymore without modifing these lines.

So, if a program doesn't run, install the latest Arduino version and change the above lines in the program and / or libraries.

How to include libraries

There are two ways to include libraries:

- With greater and less than sign
  `#include <libname.h>`
  This is used normally to include libraries that are located in the library folder:
  C:\Program Files (x86)\Arduino\libraries
- With quotation marks
  `#include "libname.h"`
  This is to include libraries that are located in the sketch folder or in the the library folder. The first situation is needed if the libraries that are used by the sketch, also need to be edited.

Compiler differences between Arduino software versions

After the installation of a new Arduino software version, problems arise often due to compiler differences. See here what to do:

- Tabsheet ino extention
  Tab sheets should have the extension *.ino now, otherwise global variables from other tab sheets are not recognized.
- Inline public member functions
  Public member functions can't be inline anymore.

Downloading doesn't work

It often happens that downloading doesn't work. The download application is not very stable, sometimes it helps to close the program and connect the USB cable again.

Missing boot loader

I recently purchased from different suppliers in China the ATmega328P-PU with UNO boot loader; however, none had the bootloader programmed. You can check this; the LED has to blink several times after power up, else there is no bootloader. So I had to figure out how to program the boot loader myself. I use this ISP programmer.

Upload speed problems

Arduino boards have all different serial baud rates, from 9600 to 115200, these are set in the file C:\Program Files\Arduino\hardware\arduino\boards.txt.

For instance: `uno.upload.speed=115200`.

But too often, the speed isn't correct if your Arduino board is outdated, than you must edit the boards.txt file. Since the baud rate is set in the bootloader, swapping an ATmega chip can also lead to speed problems if the bootloader is outdated.

General Arduino tips

- One c++ code file name (.ino) must be equal to the project folder name. A copy of a project shall not run anymore because the project map name is changed to `"Copy of ..."`. To avoid this, place the project map into another map, this can further be copied without problems.
- Older Atmega boards, before 2010, may have a slower baudrate of 19200 instead of 57600 baud for recent boards. During uploading an error may occur like: `avrdude: stk500_getsync(): not in sync: resp=0x00`. To fix upload problems, edit the file hardware\arduino\boards.txt: Change `atmega328.upload.speed=57600` to `atmega328.upload.speed=19200`
- Turn on the internal 20k pull up resistors: `pinMode(pinBlabla, INPUT); digitalWrite (pinBlabla, HIGH);`
- Variables used inside interrupt service routines and also outside them have to be volatile. This prevents that the compiler removes the variable during optimization. For instance: **`volatile int i`.**
- The analog pins can be used identically to input/output digital pins, using the aliases A0, A1 etc.

Arduino C++ anomalies

Although Arduino is programmable in C++, there are a few exceptions.

- Instead of a mandatory function `main()`, the Arduino need to have these two functions:
- `void setup(void)` and `void loop(void)`.
- The Arduino uses for a c++ code file the extension `.ino`, instead of `.cpp`. The extension `.cpp` may be used too, but at least one file must have the extension `.ino`.
- Dynamic memory allocation (`new en delete`) is not allowed.
- Library `inline` functions must be placed in the header file. `Inline` member functions can be placed in the cpp file.
- At the Arduino, we can use `explicit` only at declarations of constructors.
- Function templates must be placed in a header instead of a cpp file (`template <class T>`).

Arduino peculiarities

While I was reducing the Switch library program size, weird things came to light:

- Variables of type bool instead of byte, increases the program size with 6 byte.
- Initialization of member variables with parenthesis in a constructor, consumes 6 bytes more when using for instance `pin(pin) instead of pin(_pin).`

```
Switch::Switch(byte _pin):

pin(_pin) // _ is not required


Switch::Switch(byte pin):

pin(pin) // consumes 6 bytes more
```

- Reading a public member variable directly (`classBlabla.x)` consumes 54 bytes more than using a getter `(classBlabla.read()).`
- Initialization of member variables to 0 costs 2 bytes, although they are set to 0 by default.

Saving ATmega program memory and RAM

- In contrast to the usual practice, don't initialize variables; this takes extra program space. Variables will be initialized to zero automatically.
- Use the `Flash` library, see below.

- Place constant data in program memory instead of RAM:
  `const PROGMEM char str[] = "blabla";`
- Comment out unused and debug code with `//` or `/* */`. Unused member functions however, take no extra code.
- Use byte instead of int if possible:
  `for(byte i=0; i<3; i++)` instead of `int` saves about 12 byte program memory.
- Avoid floating point operations; the large floating point library uses a lot of program memory.
- Saving ATmega memory with the Flash library:
  `Serial.println("blabla")` uses 7 byte RAM **and** 7 byte program memory. Avoid this by using the `Flash` library:
  `Serial.println(F("blabla"))` uses only 7 byte program memory. Or use:
  `Serial << F("blabla").`
- Don't use endl, **`<< endl;`** takes 60 byte more program space than **`<< "\n";`**

Easy printing with the overload stream operator <<

- Using the `Streaming.h` library makes printing to the PC or LCD simpler. It generates no extra program memory:
  Serial << F("blabla") << data << F("blabla");
  Streaming is such an important c++ feature that it should be integrated standardly in the Arduino environment. However, this is not the case; download the latest streaming library from Mikal Hart here (http://arduiniana.org/libraries/streaming/).

Inline functions with the Arduino

- About 8 ... 50 byte program memory can be saved when functions are made `inline`. It depends on the number of times the function is called and on the size of the function, whether it makes sense. By using refactoring, code is split into small member functions, which is part of Extreme Programming. When these functions are `inline`, no extra code will be generated.
- C++ compilers may sometimes determine itself, which functions become `inline` and which not. In the past, member functions were made implicit inline by the Arduino compiler, but I have experienced that this is not the case anymore. So, to be sure, add `inline` if you need that.
- Inline **public** class-member functions are not allowed.
- Inline library functions must be placed in the header instead of the cpp file.

My Arduino wish list

- Remember the window size.
- The serial monitor needs a stop button.
- Solve incompatibility, using just #include "Arduino.h".

Disclaimer (/disclaimer.html)
Contact (/contact.html)

Do you have any comments? Please let me know (http://www.avdweb.nl/contact.html).

Back to Top