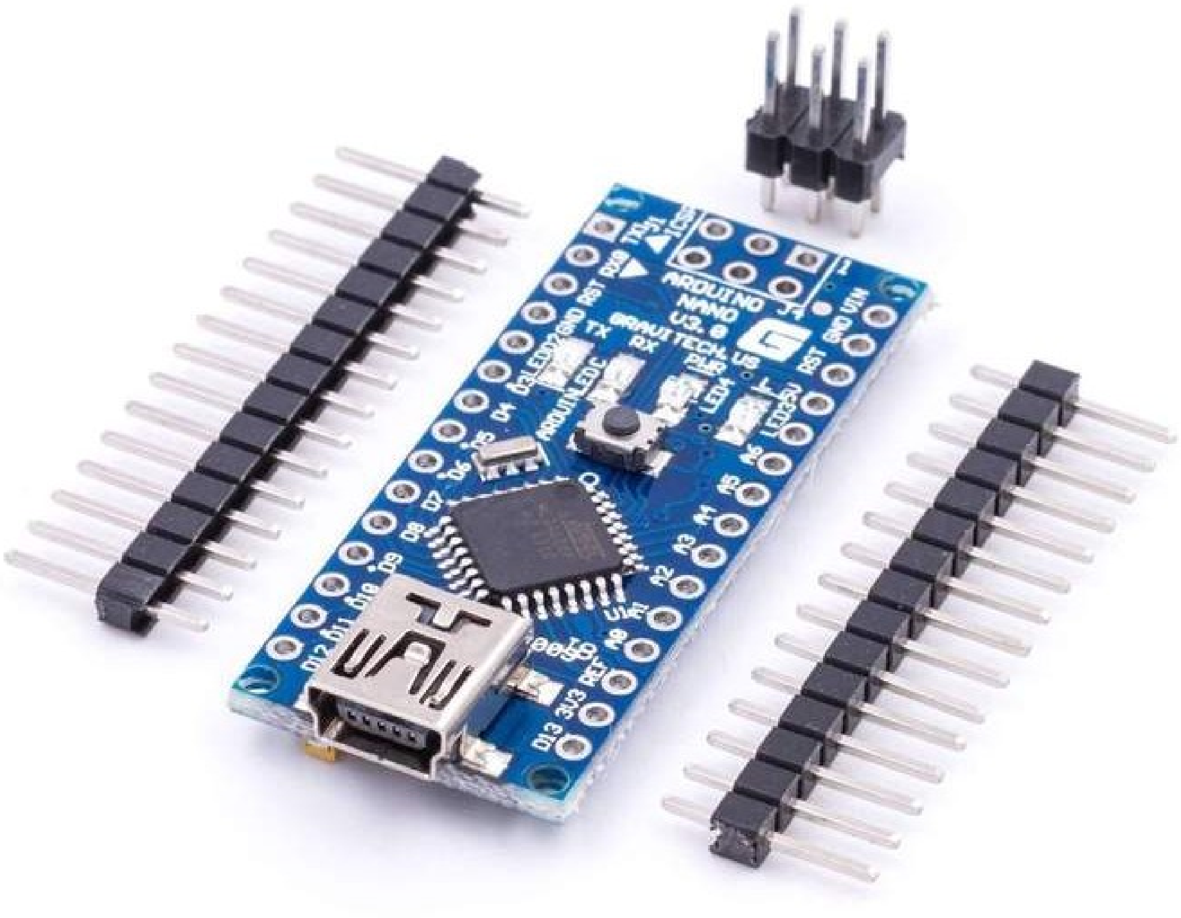
Welcome!

Thank you very much for purchasing our AZ-Delivery Nano V3.0 with FT232RL board. On the following pages, we will introduce you to how to use and setup this handy device.

Have fun!

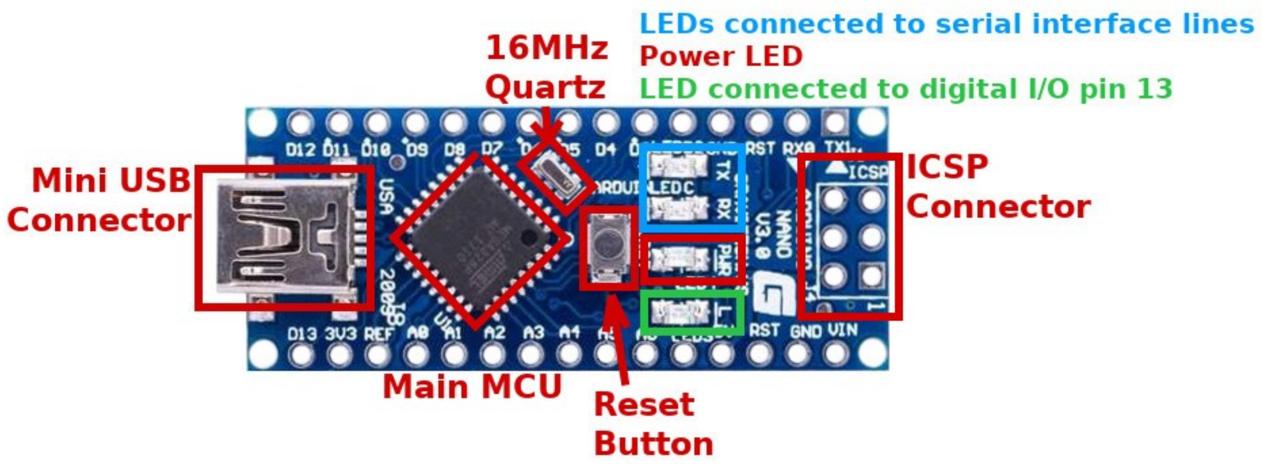


The Nano V3.0 is great device intended for electronics learning or prototyping and learning programming. Nano V3.0 is actually a microcontroller, but assembled in a way that you don’t need extra components with it. If you use single microcontroller, you will need to build stable DC power supply, and external programmer, and reset circuit, and many other things. With an Nano V3.0 you’ve got all of this in one board. And the most powerful thing about Nano V3.0 is that there is Arduino IDE (Integrated Development Environment) with endless number of code examples already written for it, in a way that everyone can understand. There is no need for you to learn internal working of onboard microcontroller in order to programm it. Just connect your Nano V3.0 board, via USB cable to your PC, install and start Arduino IDE, search and upload program that you need, to your board and that’s it. There are endless code and library examples already written online, you just need to search for it. Also there are numerous other Arduino compatible boards, like shields, or many sensors built in a way so you can easily connect them to your Nano V3.0 board. Just search our online store az-deliver.de and you’ll find more than you need.

Nano V3.0 is a small, compatible, flexible and breadboard friendly microcontroller board based on ATmega328P. It comes with almost the same functionality as in Arduino Uno but in small size.

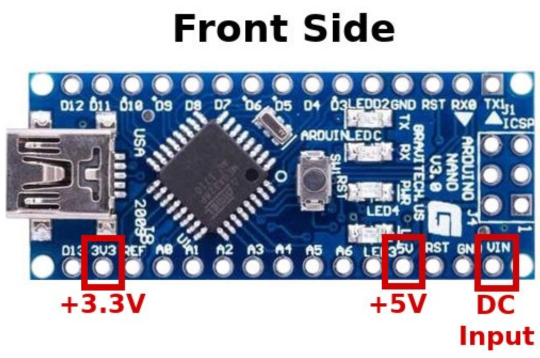
Specifications

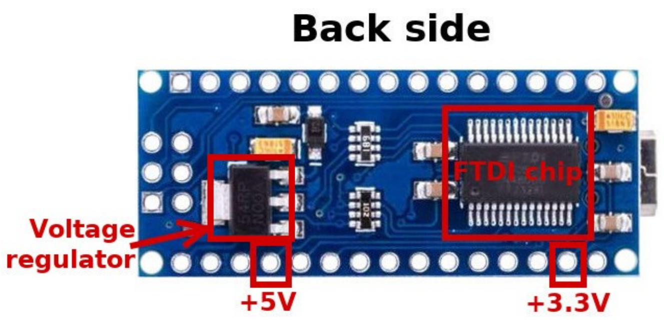
Microcontroller:	ATmega328
Operating Voltage:	5V
Input Voltage (recommended):	7÷12V
Input Voltage (limit):	6÷20V
Digital I/O Pins:	22 (6 of which are PWM)
PWM Output:	6
Analog Input Pins:	8
DC Current per I/O Pin:	20mA
DC Current for 3.3V Pin:	50mA
Flash Memory:	32KB of which 2KB used by bootloader
SRAM:	2KB (ATmega328P)
EEPROM:	1KB (ATmega328P)
Clock Speed:	16MHz
LED_BUILTIN:	connected to digital I/O pin 13
Length:	18mm
Width:	45mm
Weight:	7g


LEDs connected to serial interface lines
Power LED
LED connected to digital I/O pin 13
16MHz Quartz
Mini USB Connector
ICSP Connector
Main MCU
Reset Button

An Nano V3.0 features main microcontroller ATMega328p with 16MHz quartz oscillator. It has mini USB port which can be used to program main microcontroller and to power the board. It also has ICSP connector (In Circuit Serial Programming), if you wish to program main microcontroller externally. It also have 4 SMD LEDs, two which are connected to receive and transmit lines of serial interface, one which is used for power indicator, and one which is connected to the digital input/output pin 13.

It have reset button (image above) and two reset pins (labeled RST).

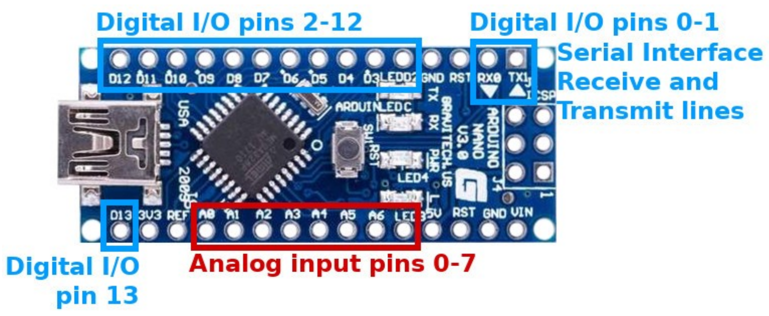




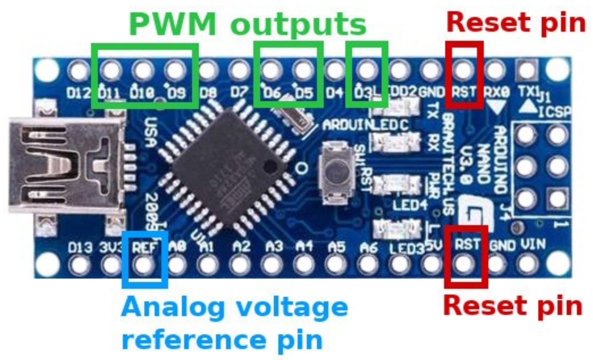
An Nano V3.0 features DC voltage regulator for +5V, **VIN** pin (DC Input on image on the left). You can connect external DC power supply to the **VIN** pin on board with voltage in range from 7V up to 12V, and voltage regulator will lower and stabilize it to the +5V. Output voltage of +3.3V we can get from FT232RL chip, and it is NOT accurate thus NOT stable.

Nano V3.0 uses FT232RL chip to make communication like USART serial interface but via USB. FT232RL chip connect PC's USB port with USART serial interface of microcontroller and thus enable us to program microcontroller via USB port.

Also you can mini USB port to power the Nano V3.0 board!



Nano V3.0 is built in a way that separates digital input/output pins from analog input pins. So there are 8 analog input pins (Arduino Uno does not have pins A6 and A7), and separated 14 digital input/output pins.



6 of 14 digital input/output pins can be used as PWM outputs (Pulse Width Modulation). Those pins are D3, D5, D6, D9, D10, and D11.

To reset main microcontroller you need to press reset button or you connect one of reset pins to the GND (0V).

By default when you are using analog to digital converter, analog voltage reference is used from VCC (=+5V), but you can use any analog voltage reverence you like. You just need to connect that voltage reference to the AREF pin - Analog voltage reference pin (image above). <u>Arduino Uno does not have this pin.</u>

Digital I/O pins D0 and D1 are connected to receive and transmit lines of serial interface. On image above these pins are labeled RX and TX respectively. So we suggest never to use these digital I/O pins as digital inputs or outputs, because serial interface is used every time when you are uploading new program into you microcontroller on board Nano V3.0. Actually wherever you reprogram your Nano V3.0, or whenever you are using serial interface. By using these pins as digital inputs or outputs, you can get many errors while reprogramming, or cause your electronics parts or devices connected to them to work incorrectly.

Communicaton interfaces

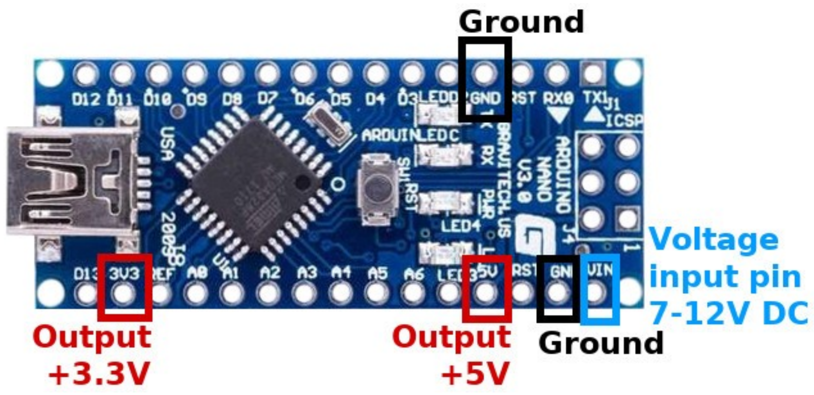
We already wrote that digital I/O pins D0 and D1 have alternative functions. They are connected to receive and transmit lines of serial interface.

There are two more communication interfaces supported by ATMega328P microcontroller, Serial Peripheral Interface - SPI and Inter-Integrated Circuit interface - I2C (or TWI - Two Wire Interface)

For SPI interface digital I/O pins D10, D11, D12 and D13 are used. Their functions are SS, MOSI, MISO and SCK respectively.

For I2C interface analog input pins A4 and A5 are used. Their functions are SDA and SCL respectively.

Power Pins



Nano V3.0 does not have power pin header like on Nano V3.0. It have +3.3V and +5V voltage output pins, two Ground (GND) pins, and VIN pin. We already wrote about VIN pin, which can be used as DC input pin for external power supply. We also wrote that Nano V3.0 can be powered from mini USB port.

Nano V3.0 can output voltages +5V (which is regulated and stable) and +3.3V which is not accurate thus not stable, so be careful when you are using sensitive electronics devices which for power need stable +3.3V.

Nano V3.0 Power, Current, and Voltage Limitations

Voltage Input Limits:

Input power: to power the Nano V3.0, you either plug it in to a USB port, or you input a voltage source to it via its "VIN" and "GND" pins. When powering the Nano V3.0 via the VIN and GND pins, it has the following input voltage limitations:

» **Recommended input voltage limits: 7~12V.**
These input voltages can be sustained indefinitely

» **Absolute voltage limits for powering the Nano V3.0: 6~20V**
- Below 7V may cause the 5V levels on the board to waver, fluctuate, or sag, causing board instability and less accurate analog readings when using `analogRead()`.
- Sustained voltage levels above 12V will cause additional heating on the linear voltage regulator of the Nano V3.0, which could cause it to overheat. Short periods, however, are fine. Feel the voltage regulator with your finger. If it feels too hot to comfortably touch, you need to use a voltage source within the recommended limits in order to reduce heat buildup.

Caution!!!

Before touching any electro-static discharge (ESD) sensitive parts on the Nano V3.0 (which is pretty much all of the Nano V3.0), touch the metal part of the USB plug first to ground yourself out to the board and safely discharge any static voltage you have built up.

» **Voltage limits on input/output pins: -0.5 to +5.5V max.**

If you need to read in a voltage on an Nano V3.0 digital or analog input pin, ensure that it is between 0 and 5V. If it is outside these limits, you can bring down the voltage using a voltage divider. This scales the input voltage to allow for analog or digital readings of voltages otherwise outside the allowed range. If your input signal is digital, and you don't need to take scaled analog readings, another technique is to clip (cut the top off of) the input voltage, rather than scale it. Since AVR microcontrollers have internal clamping diodes, this can be done by simply adding a single resistor in series with the pin. By adding a 10kΩ resistor in series with the input pin (any input pin) permits input voltages as low as -10.5V or as high as +15.5V.

Current Output Limits:

- » **Total maximum current draw from the Nano V3.0 when powered from a USB port: 500mA**
- » **The Nano V3.0 has a “resettable polyfuse that protects your computer’s USB ports from shorts and overcurrent.”**
- » **Total maximum current draw when powered via external power supply: 1A**

Note: If not powered by USB, the total 5V current limit coming out of the Nano V3.0 is limited by the voltage regulator on your particular board, and/or your input power supply, whichever provides less power. Let’s assume your power supply going to the Nano V3.0 can provide 7÷12V and >= 1A. If this is the case, the 5V power is limited strictly by your Nano V3.0 board’s voltage regulator.

- » **Total max current draw across the Nano V3.0 “5V” pin and “GND”: as specified above.**
- » **Total max current per input/output pin: 40mA**

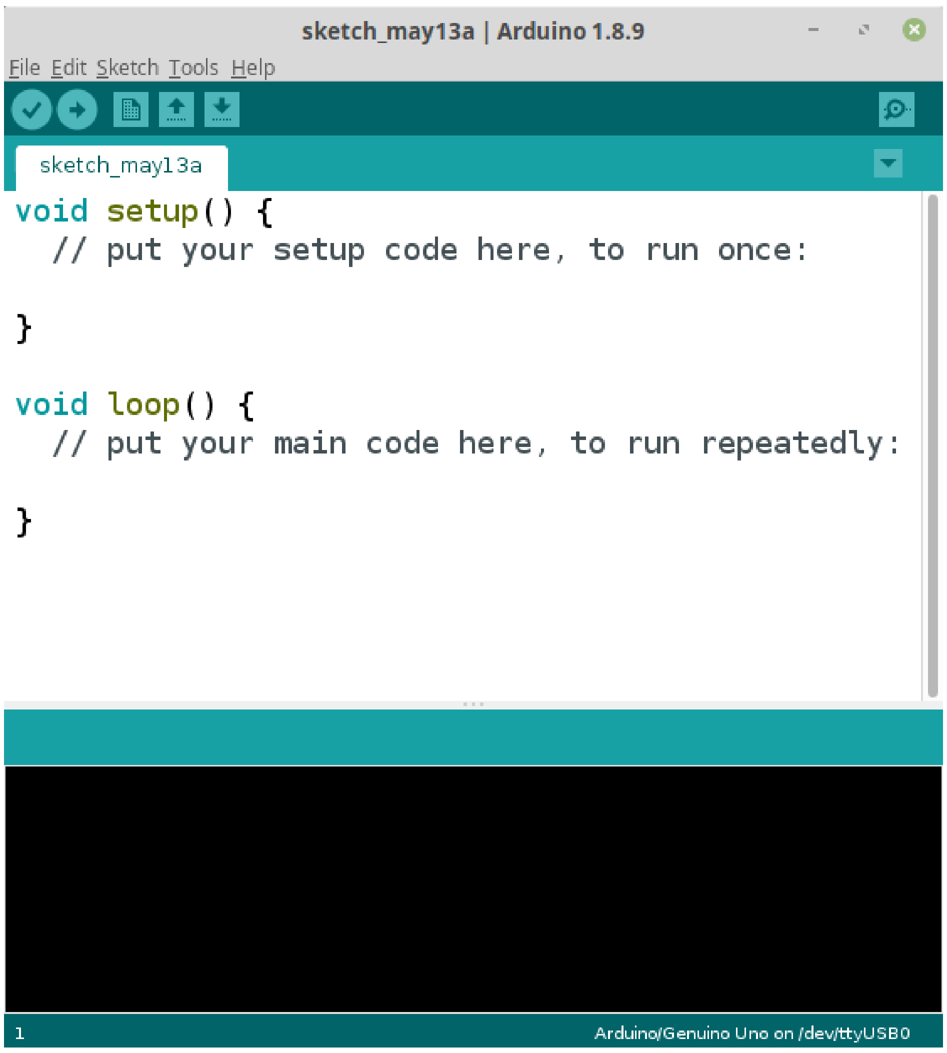
» Sum of currents out of ALL input/output pins combined: 200mA!!!

Note: this is the one that usually gets people, as it may be the least understood! Despite the fact that your voltage regulator on the Nano V3.0 may permit up to 1A draw across the “5V” and “GND” pins, the sum of all currents going into or out of the input/output pins (all Analog and Digital pins combined) of the ATMega328P microcontroller itself **cannot exceed 200mA**. So, if you are powering 10 LEDs at 20mA each, via your Analog or Digital pins, you just hit your limit! Any more than that and you may damage the microcontroller on the Nano V3.0 board. A work-around if you need more current is to use transistors. The Nano V3.0 input/output pins can then use a very low current to activate a transistor, which then turns a higher current on and off from the 5V pin directly (which is connected straight to the output of the on-board linear voltage regulator), to the device you want to control. This way, you keep the sum total output from the Nano V3.0 analog/digital pins below 200mA, while allowing up to the 500mA~1A limit from the 5V pin.

Arduino IDE

To program any of Nano V3.0 board you need to have an IDE app (Integrated Development Environment). Arduino features Arduino IDE, which can be downloaded from https://www.arduino.cc/en/Main/Software
Just find your operating system, download it and install it.

When you install it and open the app, this will be starting window.


sketch_may13a | Arduino 1.8.9
File Edit Sketch Tools Help
sketch_may13a
1
Arduino/Genuino Uno on /dev/ttyUSB0

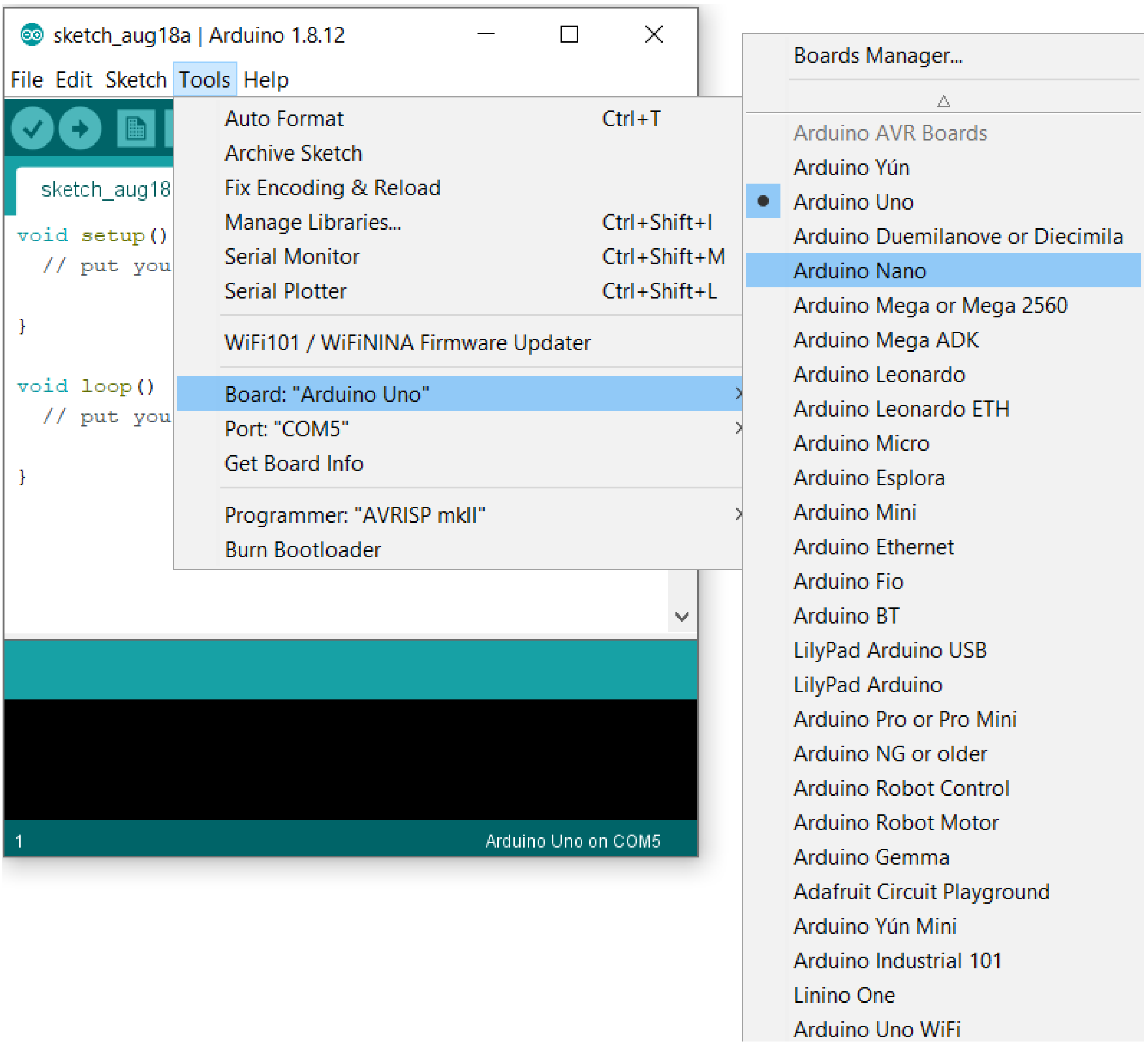
```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Opened program example is called empty sketch. A sketch is program example, where we write our code. It has two essential parts, `setup()` function and `loop()` function, and it can have any number of other functions as well.

`setup()` function runs only once, at the beginning of program execution, when you power up the board, or when you reset the board. In this function we setup all initializations, for example declare the state of digital input/output pins or setting up analog input pins, or setting up serial interface for serial communication, etc.

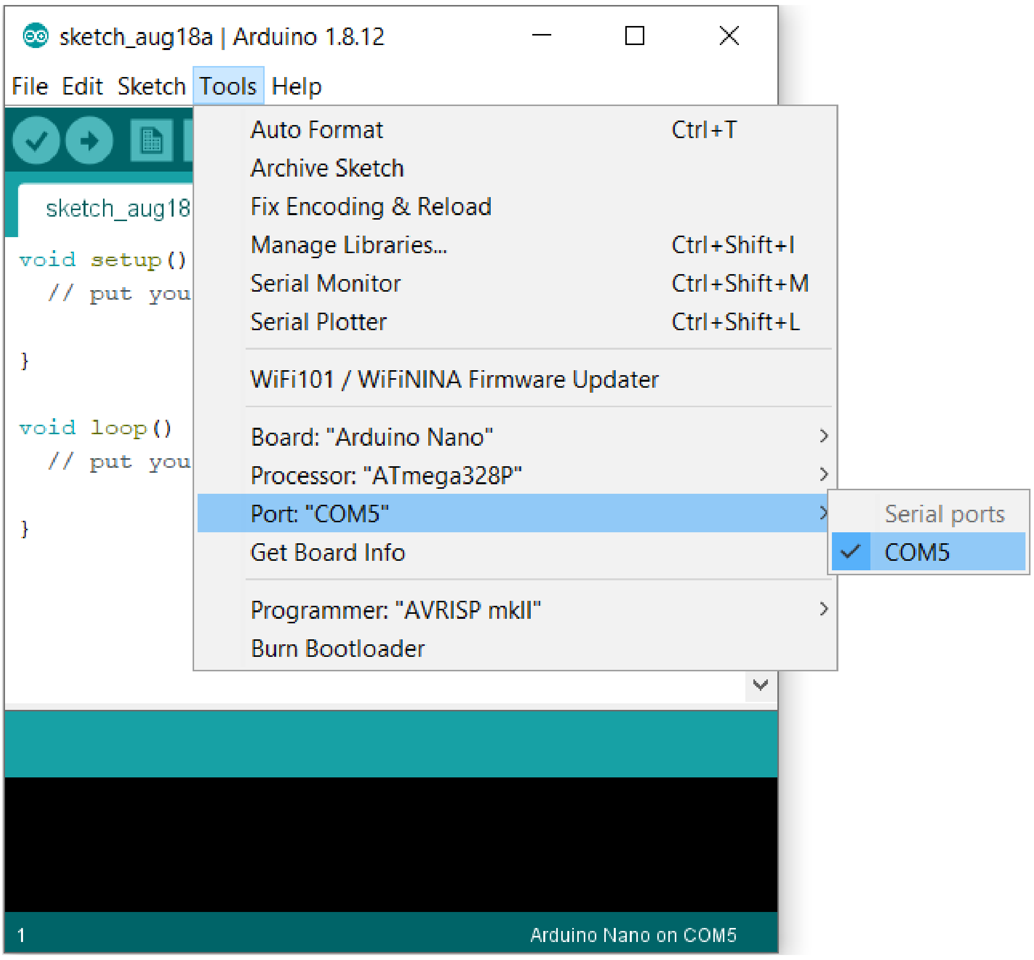
`loop()` function runs after `setup()` and it runs indefinitely, over and over again, so called "endless" loop function. Actually it runs all time while the board is connected to the power. This is because programs in electronics devices should never reach the end, because if that happen that means your device is as good as turned off. Here we write the logic, algorithms on which our application for microcontroller board works.

When we connect our Nano V3.0 via USB cable to the PC, first thing we should do in Arduino IDE is select the Nano V3.0 in the menu unter Tools / Board.



```
void setup()
  // put you

}

void loop()
  // put you

}
```

Then we need to select the COM port, again under Tools / then Port.


sketch_aug18a | Arduino 1.8.12

File Edit Sketch Tools Help

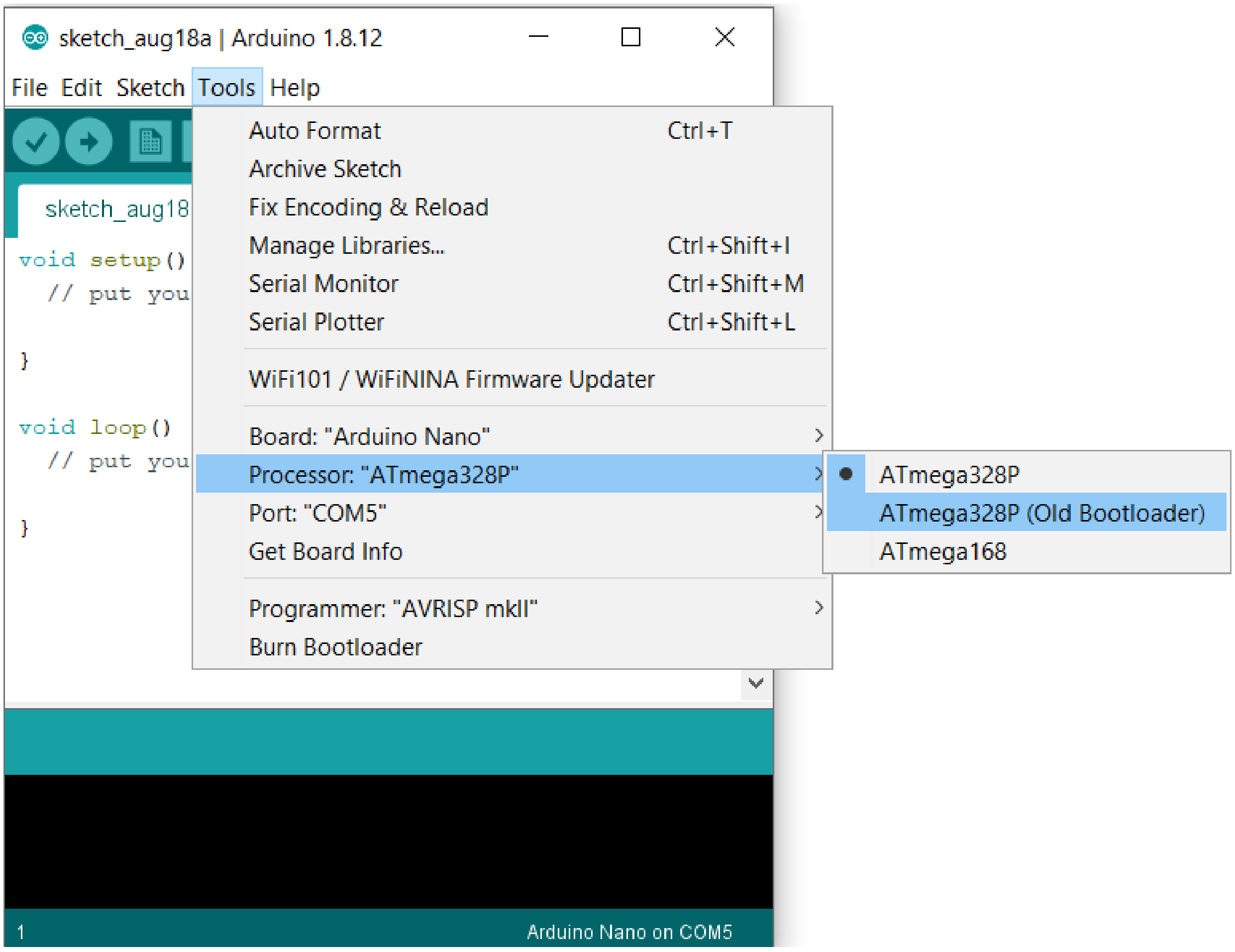
```
void setup() {
  // put you

}

void loop() {
  // put you

}
```

Auto Format	Ctrl+T
Archive Sketch	
Fix Encoding & Reload	
Manage Libraries...	Ctrl+Shift+I
Serial Monitor	Ctrl+Shift+M
Serial Plotter	Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater	
Board: "Arduino Nano"	
Processor: "ATmega328P"	
Port: "COM5"	
Get Board Info	
Programmer: "AVRISP mkII"	
Burn Bootloader	

Serial ports
✓ COM5

1 Arduino Nano on COM5

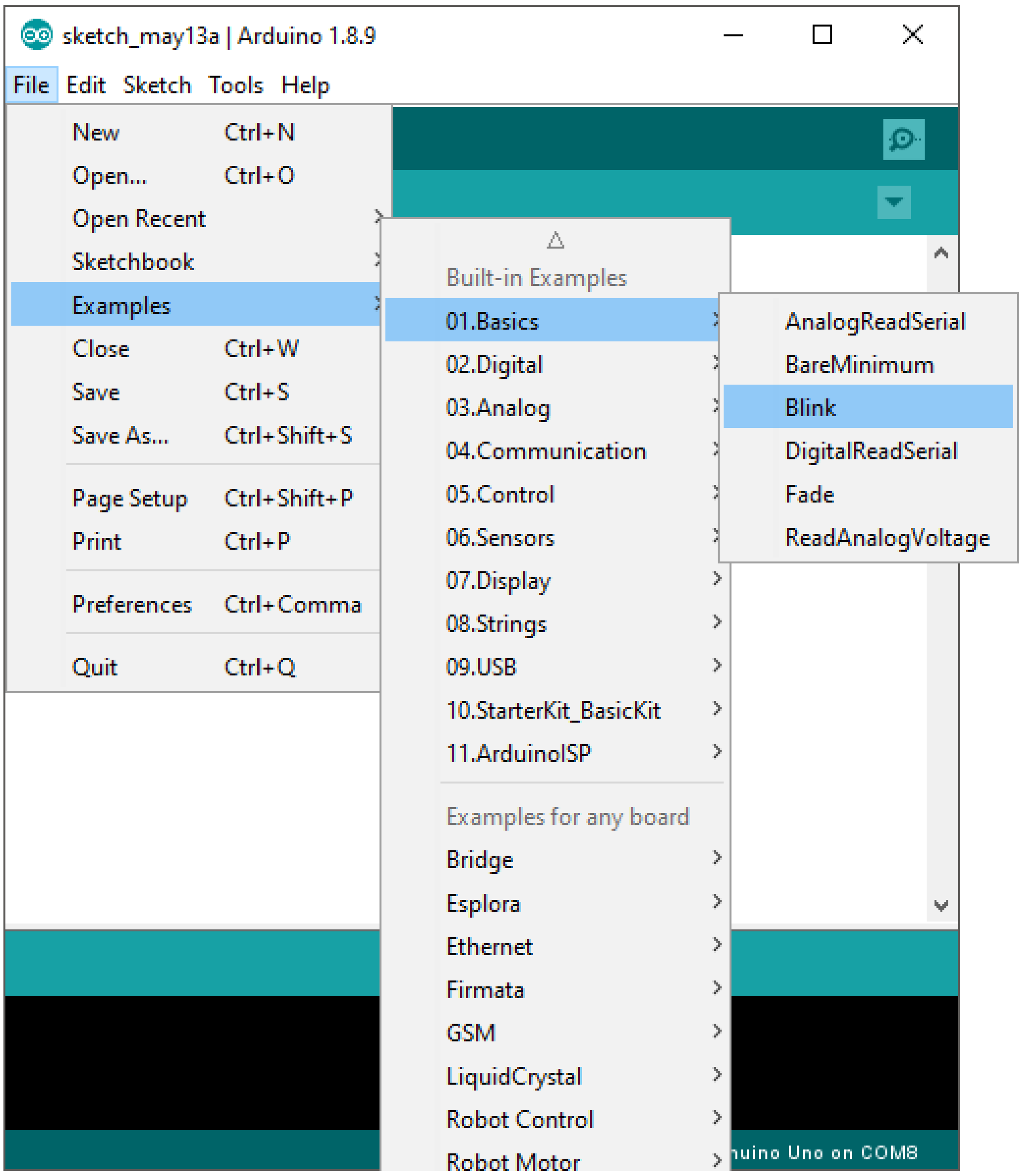
Finally, we select under Tools / Processor the Atmega328P (Old Bootloader).


sketch_aug18a | Arduino 1.8.12
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Nano"
Processor: "ATmega328P"
Port: "COM5"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
● ATmega328P
ATmega328P (Old Bootloader)
ATmega168
Arduino Nano on COM5

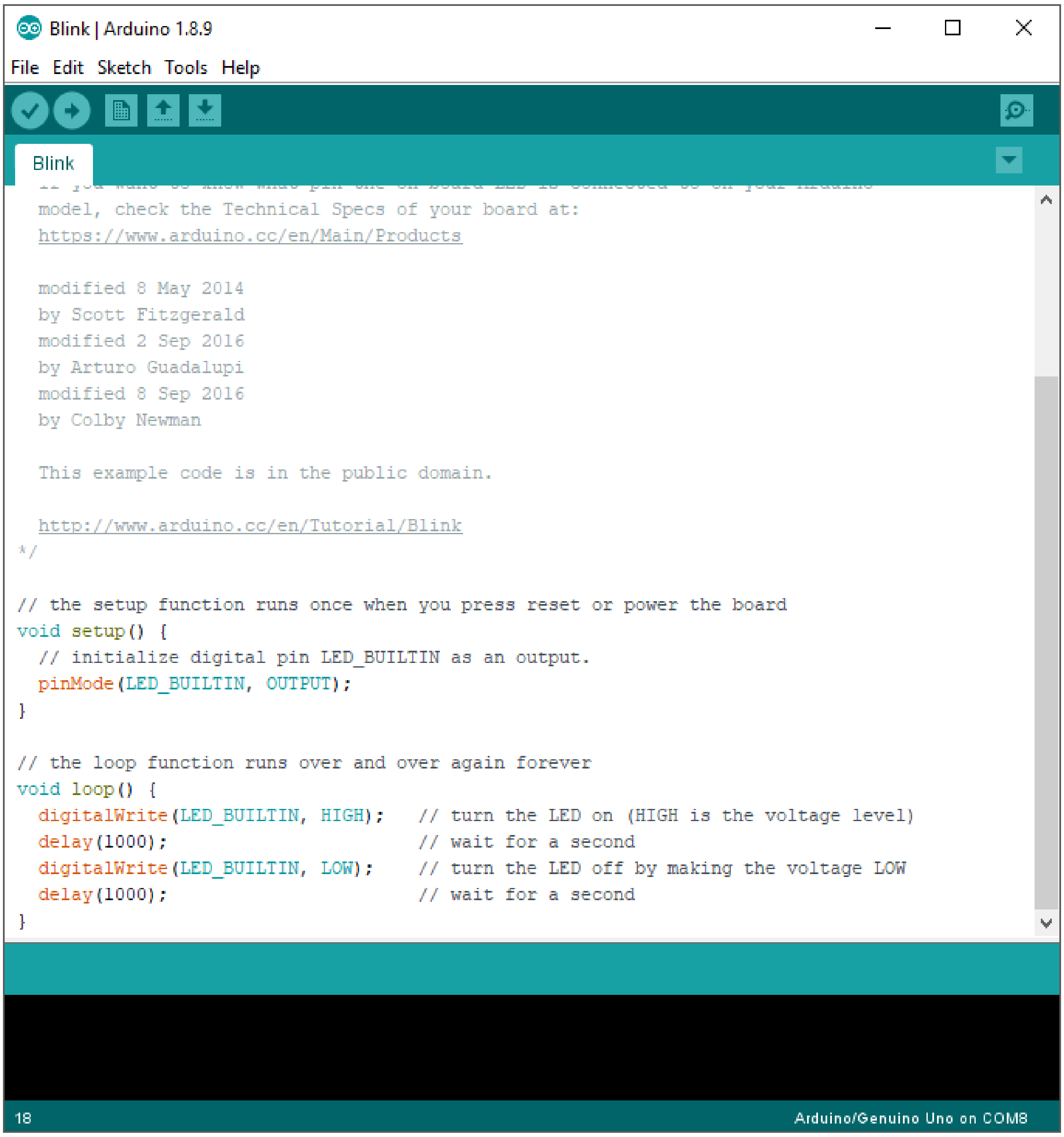
```
void setup()
  // put you
}

void loop()
  // put you
}
```

Application example

And now we can start programming. Arduino IDE comes with dosen prewritten sketch examples, which you can use. Here we will use BLINK sketch example. Go to File > Examples > 01.Basics > Blink.



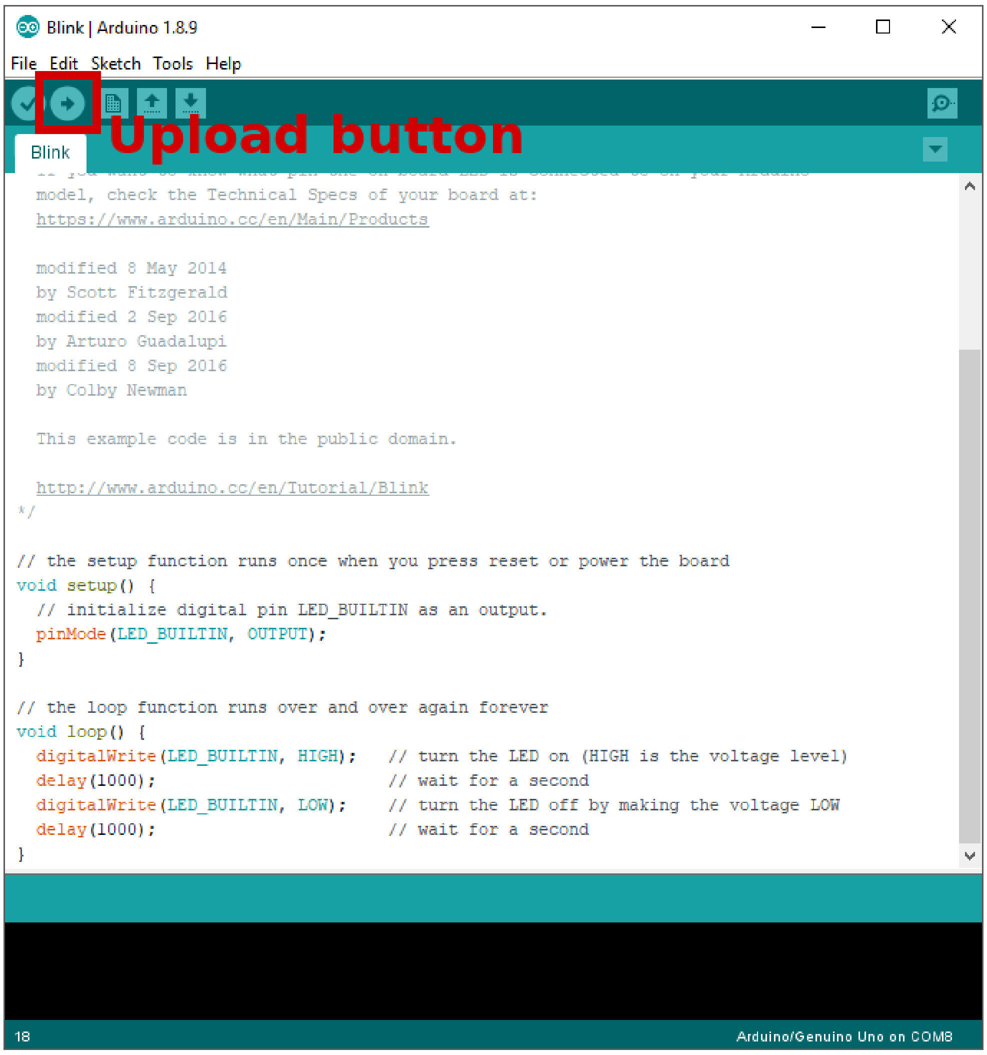
A new window with new sketch example will open:



```
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

What this sketch does is turn ON an LED on board connected to the digital I/O pin 13, for one second, and than turn it OFF for one second. This turning on and off is called blinking, thus this sketch name.

When we are done programing, to compile and upload sketch to your Nano V3.0 board, you press upload button.



```
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

After this, on board LED should start blinking in interval of one second.

You've done it, you can now use your module for your projects.

Now it is time to learn and make the Projects on your own. You can do that with the help of many example scripts and other tutorials, which you can find on the internet.

If you are looking for the high quality products for Arduino and Raspberry Pi, AZ-Delivery Vertriebs GmbH is the right company to get them from. You will be provided with numerous application examples, full installation guides, eBooks, libraries and assistance from our technical experts.