

Arduino IDE Toolbar

Verify	Scans your code and reports any errors
Upload	Compiles your code and uploads it to the Arduino board via USB
New	Opens a blank Arduino sketch
Open	Opens a list of your saved sketches in the file browser
Save	Saves your current sketch
Serial Monitor	Opens the serial monitor in a new window

Arduino Program Structure

<code>void setup() { }</code>	Runs once at startup
<code>void loop() { }</code>	Runs continually

Built in Arduino Functions

Pin setup

<code>pinMode(PIN_NUMBER, INPUT/OUTPUT)</code>	Sets the pin at the location PIN_NUMBER to be either an INPUT or an OUTPUT
<code>pinMode(PIN_NUMBER, INPUT_PULLUP)</code>	Sets the pin at the location PIN_NUMBER to be an input using the Arduino board's built-in pull-up resistor
<code>digitalRead(PIN_NUMBER)</code>	Reads the input at PIN_NUMBER and returns a 1 or 0 (HIGH or LOW)
<code>digitalWrite(PIN_NUMBER, VALUE)</code>	Writes a value of 1 or 0 (HIGH or LOW) to digital pin PIN_NUMBER
<code>analogRead(PIN_NUMBER)</code>	Reads the analog pin PIN_NUMBER and returns an integer between 0 and 1023
<code>analogWrite(PIN_NUMBER, VALUE)</code>	Emulates analog output VALUE using PWM on PIN_NUMBER (note: only available on pins 3, 5, 6, 9, 10, and 11)
<code>analogReference(DEFAULT)</code>	Use the default reference voltage (5V or 3.3V depending on board voltage)
<code>analogReference(INTERNAL)</code>	Use an internal reference voltage (1.1v for ATmega168/328p, 2.56 for ATmega 32U4/8)
<code>analogReference(EXTERNAL)</code>	Use a voltage applied to the AREF pin as voltage reference (note: 0-5V only)

Time functions

<code>millis()</code>	Returns the time in milliseconds since the Arduino sketch began running as an unsigned long integer
<code>micros()</code>	Returns the time in microseconds since the Arduino sketch began running as an unsigned long integer
<code>delay(INTEGER)</code>	Delays program execution for INTEGER milliseconds
<code>delayMicroseconds(INTEGER)</code>	Delays program execution for INTEGER microseconds

Mathematical Functions

<code>min(i, j)</code>	Returns the lowest of the two values i and j
<code>max(i,j)</code>	Returns the highest of the two values i and j
<code>abs(i)</code>	Returns the absolute value of i
<code>sin(angle)</code>	Returns the sine of an angle in radians
<code>cos(angle)</code>	Returns the cosine of an angle in radians
<code>tan(angle)</code>	Returns the tangent of an angle in radians
<code>sqrt(i)</code>	Returns the square root of i
<code>pow(base, exponent)</code>	Raises the number base to the number exponent (e.g pow (2 , 3) ==8)
<code>constrain(i, minval, maxval)</code>	Constrains the value i between minval and maxval
<code>map(val, fromL, fromH, toL, toH)</code>	Remaps val from one range to another
<code>random(i)</code>	Returns a random long integer smaller than i
<code>random(i, j)</code>	Returns a random long integer between i and j
<code>randomSeed(k)</code>	Uses the value k to seed the random() function

Casting

(type)variable

Casts the value of variable to a new type

Serial Communication

Serial.begin(speed)	Start serial communication at a specified speed
Serial.end()	Close serial communication
Serial.print(DATA)	Prints DATA to the serial port. DATA can be characters, strings, integers and floating point numbers
Serial.available()	Return the number of characters available to read in the serial buffer
Serial.read()	Read the first character in the serial buffer (returns -1 if no data is available)
Serial.write(DATA)	Write DATA to the serial buffer. DATA can be a character, integer, or array
Serial.flush()	Clears the serial buffer once outgoing communication is complete

Servo (#include <Servo.h>)

Servo myServo	Creates the variable myServo of type Servo
myServo.attach(PIN_NUMBER)	Associated myServo with the pin at location PIN_NUMBER
myServo.write(angle)	Writes an angle between 0 and 180 to the servo attached to myServo
myServo.writeMicroseconds(uS)	Writes a value in microseconds to the servo attached to myServo (typically between 1000 and 2000 with 1500 as the midpoint)
myServo.read()	Returns an integer containing the current angle of the servo between 0 - 180
myServo.attached()	Returns true if the servo is attached to a pin
myServo.detach()	Dissociates myServo with an attached pin