

# Arduino Programming Cheat Sheet

Primary source: Arduino Language Reference  
<http://arduino.cc/en/Reference/>

## Structure & Flow

### Basic Program Structure

```
void setup() {  
    // runs once when sketch starts  
}  
void loop() {  
    // runs repeatedly  
}
```

### Control Structures

```
if (x < 5) { ... } else { ... }  
while (x < 5) { ... }  
do { ... } while ( x < 5);  
for (int i = 0; i < 10; i++) { ... }  
break; // exit a loop immediately  
continue; // go to next iteration  
switch (myVar) {  
    case 1:  
        ...  
        break;  
    case 2:  
        ...  
        break;  
    default:  
        ...  
}  
return x; // just return; for voids
```

## Operators

### General Operators

```
= (assignment operator)  
+ (add)      - (subtract)  
* (multiply) / (divide)  
% (modulo)  
== (equal to) != (not equal to)  
< (less than) > (greater than)  
<= (less than or equal to)  
>= (greater than or equal to)  
&& (and)    || (or)    ! (not)
```

### Compound Operators

```
++ (increment)  
-- (decrement)  
+= (compound addition)  
-= (compound subtraction)  
*= (compound multiplication)  
/= (compound division)  
&= (compound bitwise and)  
|= (compound bitwise or)
```

### Bitwise Operators

```
& (bitwise and)    | (bitwise or)  
^ (bitwise xor)    ~ (bitwise not)  
<< (shift left)    >> (shift right)
```

## Built-in Functions

### Pin Input/Output

```
Digital I/O (pins: 0-13 A0-A5)  
pinMode(pin, [INPUT, OUTPUT])  
int digitalread(pin)  
digitalWrite(pin, value)  
    // Write HIGH to an input to  
    // enable pull-up resistors  
Analog In (pins: 0-5)  
int analogRead(pin)  
analogReference (  
    [DEFAULT, INTERNAL, EXTERNAL])  
PWM Out (pins: 3 5 6 9 10 11)  
analogWrite(pin, value)
```

### Advanced I/O

```
tone(pin, freqhz)  
tone(pin, freqhz, duration_ms)  
noTone(pin)  
shiftOut(dataPin, clockPin,  
    [MSBFIRST,LSBFIRST], value)  
unsigned long pulseIn(pin,  
    [HIGH,LOW])
```

### Time

```
unsigned long millis()  
    // overflows at 50 days  
unsigned long micros()  
    // overflows at 70 minutes  
delay (msec)  
delayMicroseconds (usec)
```

### Math

```
min(x, y)    max(x, y)    abs(x)  
sin(rad)     cos(rad)     tan(rad)  
sqrt(x)       pow(base, exponent)  
constrain(x, minval, maxval)  
map(val, fromL, fromH, toL, toH)
```

### Random Numbers

```
randomSeed(seed) // long or int  
long random(max)  
long random(min, max)
```

### Bits and Bytes

```
lowByte(x)    highByte(x)  
bitRead(x, bitn)  
bitWrite(x, bitn, bit)  
bitSet(x, bitn)  
bitClear(x, bitn)  
bit(bitn) // bitn: 0=LSB 7=MSB
```

### Type Conversions

```
char()    byte()  
int()     word()  
long()    float()
```

### External Interrupts

```
attachInterrupt(interrupt, func,  
    [LOW, CHANGE, RISING, FALLING])  
detachInterrupt(interrupt)  
interrupts()  
noInterrupts()
```

## Libraries

```
Serial (communicate with PC or via RX/TX)  
begin(long Speed) // up to 115200  
end()  
int available() // #bytes available  
byte read() // -1 if none available  
byte peek()  
flush()  
print(myData)  
println(myData)  
write(myBytes)  
SerialEvent() // called if data rdy
```

```
SoftwareSerial (serial comm. on any pins)  
    (#include <softwareSerial.h>)  
SoftwareSerial(rxPin, txPin)  
begin(long Speed) // up to 115200  
listen() // Only 1 can listen  
isListening() // at a time.  
read, peek, print, println, write  
    // all like in Serial library
```

```
EEPROM (#include <EEPROM.h>)  
byte read(intAddr)  
write(intAddr, myByte)  
  
Servo (#include <Servo.h>)  
attach(pin, [min_uS, max_uS])  
write(angle) // 0 to 180  
writeMicroseconds(uS)  
    // 1000-2000; 1500 is midpoint  
int read() // 0 to 180  
bool attached()  
detach()
```

```
Wire (I2C comm.) (#include <Wire.h>)  
begin() // join a master  
begin(addr) // join a slave @ addr  
requestFrom(address, count)  
beginTransaction(addr) // Step 1  
send(myByte) // Step 2  
send(char * mystring)  
send(byte * data, size)  
endTransmission() // Step 3  
int available() // #bytes available  
byte receive() // get next byte  
onReceive(handler)  
onRequest(handler)
```

## Variables, Arrays, and Data

### Data types

```
void  
boolean (0, 1, true, false)  
char (e.g. 'a' -128 to 127)  
int (-32768 to 32767)  
long (-2147483648 to 2147483647)  
unsigned char (0 to 255)  
byte (0 to 255)  
unsigned int (0 to 65535)  
word (0 to 65535)  
unsigned long (0 to 4294967295)  
float (-3.4028e+38 to 3.4028e+38)  
double (currently same as float)
```

### Qualifiers

```
static (persists between calls)  
volatile (use RAM (nice for ISR))  
const (make read only)  
PROGMEM (Use flash)
```

### Arrays

```
int myInts[6]; // array of 6 ints  
int myPins[]={2, 4, 8, 3, 6};  
int mySensVals[6]={2, 4, -8, 3, 2};  
myInts[0]=42; // assigning first  
            // index of myInts  
myInts[6]=12; // ERROR! Indexes  
            // are 0 though 5
```

### Constants

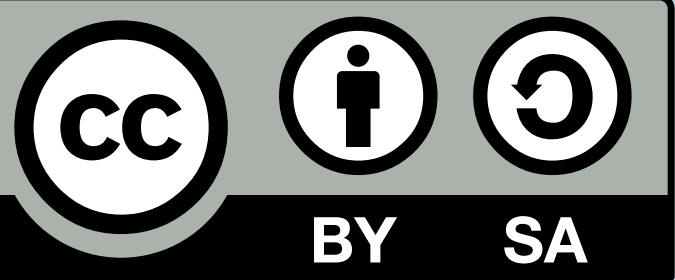
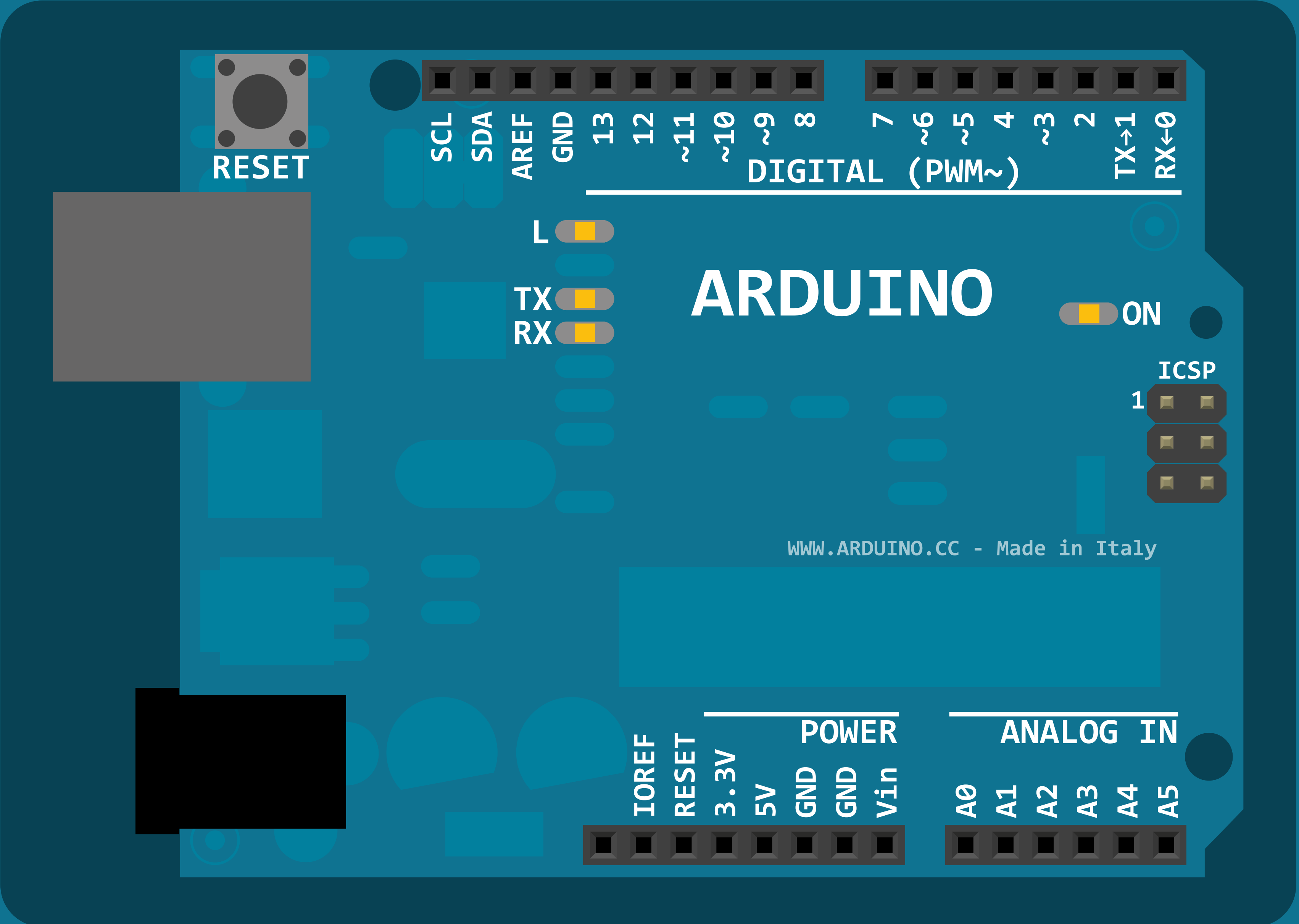
```
HIGH | LOW  
INPUT | OUTPUT  
true | false  
143 (Decimal)  
0173 (Octal - base 8)  
0b11011111 (Binary)  
0x7B (Hexadecimal - base 16)  
7U (force unsigned)  
10L (force long)  
15UL (force long unsigned)  
10.0 (force floating point)  
2.4e5 (2.4*10^5 = 240000)
```

### Pointer Access

```
& (reference: get a pointer)  
* (dereference: follow a pointer)
```

### Strings

```
char S1[8] =  
    {'A','r','d','u','i','n','o'};  
    // unterminated string; may crash  
char S2[8] =  
    {'A','r','d','u','i','n','o','\0'};  
    // includes \0 null termination  
char S3[]="arduino";  
char S4[8]="arduino";
```



by Mark Liffiton

Adapted from:  
- Original by Gavin Smith  
- SVG version by Frederic Dufourg  
- Arduino board drawing  
original by Fritzing.org