

## Structure

void **setup()** void **loop()**

## Control Structures

```
if (x<5){ } else { }  
switch (myvar) {  
  case 1:  
    break;  
  case 2:  
    break;  
  default:  
}  
for (int i=0; i <= 255; i++){ }  
while (x<5){ }  
do { } while (x<5);  
continue; //Go to next in  
do/for/while loop  
return x; // Or 'return;' for voids.  
goto // considered harmful :-)
```

## Further Syntax

```
// (single line comment)  
/* (multi-line comment) */  
#define DOZEN 12 //Not baker's!  
#include <avr/pgmspace.h>
```

## General Operators

= (assignment operator)  
+ (addition) - (subtraction)  
\* (multiplication) / (division)  
% (modulo)  
== (equal to) != (not equal to)  
< (less than) > (greater than)  
<= (less than or equal to)  
>= (greater than or equal to)  
&& (and) || (or) ! (not)

## Pointer Access

& reference operator  
\* dereference operator

## Bitwise Operators

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

## Compound Operators

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

## Constants

HIGH | LOW  
INPUT | OUTPUT  
true | false  
143 // **Decimal** number  
0173 // **Octal** number  
0b11011111 // **Binary**  
0x7B // **Hex** number  
7U // Force unsigned  
10L // Force long  
15UL // Force long unsigned  
10.0 // Forces floating point  
2.4e5 // 240000

## Data Types

void  
boolean (0, 1, false, true)  
char (e.g. 'a' -128 to 127)  
unsigned char (0 to 255)  
byte (0 to 255)  
int (-32,768 to 32,767)  
unsigned int (0 to 65535)  
word (0 to 65535)  
long (-2,147,483,648 to  
2,147,483,647)  
unsigned long (0 to 4,294,967,295)  
float (-3.4028235E+38 to  
3.4028235E+38)  
double (currently same as float)  
sizeof(myint) // returns 2 bytes

## Strings

```
char S1[15];  
char S2[8]='a','r','d','u','i','n','o';  
char S3[8]='a','r','d','u','i','n','o','\0';  
//Included \0 null termination  
char S4[] = "arduino";  
char S5[8] = "arduino";  
char S6[15] = "arduino";
```

## Arrays

```
int myInts[6];  
int myPins[] = {2, 4, 8, 3, 6};  
int mySensVals[6] = {2, 4, -8, 3, 2};
```

## Conversion

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

## Qualifiers

**static** // persists between calls  
**volatile** // use RAM (nice for ISR)  
**const** // make read-only  
**PROGMEM** // use flash

## Digital I/O

**pinMode**(pin, [INPUT,OUTPUT])  
**digitalWrite**(pin, value)  
**int digitalWrite**(pin)  
//Write High to inputs to use pull-up res

## Analog I/O

**analogReference**([DEFAULT,  
INTERNAL,EXTERNAL])  
**int analogRead**(pin) //Call twice if  
switching pins from high Z source.  
**analogWrite**(pin, value) // PWM

## 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**() // 50 days overflow.  
unsigned long **micros**() // 70 min overflow  
**delay**(ms)  
**delayMicroseconds**(us)

## Math

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

## Random Numbers

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

## Bits and Bytes

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

## External Interrupts

**attachInterrupt**(interrupt, function,  
[LOW,CHANGE,RISING,FALLING])  
**detachInterrupt**(interrupt)  
**interrupts**()  
**noInterrupts**()

## Libraries:

### Serial.

**begin**([300, 1200, 2400, 4800,  
9600,14400, 19200, 28800, 38400,  
57600,115200])  
**end**()  
**int available**()  
**int read**()  
**flush**()  
**print**()  
**println**()  
**write**()

**EEPROM** (#include <EEPROM.h>)  
byte **read**(intAddr)  
**write**(intAddr,myByte)

### Servo (#include <Servo.h>)

**attach**(pin, [min\_uS, max\_uS])  
**write**(angle) // 0-180  
**writeMicroseconds**(uS) //1000-  
2000,1500 is midpoint  
**read**() // 0-180  
**attached**() //Returns boolean  
**detach**()

### SoftwareSerial(RxPin,TxPin)

// #include<SoftwareSerial.h>  
**begin**(longSpeed) // up to 9600  
char **read**() // blocks till data  
**print**(myData) or **println**(myData)

### Wire (#include <Wire.h>) // For I2C

**begin**() // Join as master  
**begin**(addr) // Join as slave @ addr  
**requestFrom**(address, count)  
**beginTransmission**(addr) // Step 1  
**send**(mybyte) // Step 2  
**send**(char \* mystring)  
**send**(byte \* data, size)  
**endTransmission**() // Step 3  
byte **available**() // Num of bytes  
byte **receive**() //Return next byte  
**onReceive**(handler)  
**onRequest**(handler)