

Struktur	
void	wird einmal
setup()	ausgeführt
void loop()	wird ewig ausgeführt

Kontrolle	
if (x<5) { Code }	Code wird ausgeführt wenn WAHR
else { Code }	Code wird ausgeführt wenn if FALSCH
for ( int i = 0; i < 255; i++) { Code }	i ist 0, Code wird ausgeführt, i wird um 1 erhöht. Solange wie i kleiner 255.
while ( x < 6 ) { Code }	Code wird solange ausgeführt wie x kleiner 6. x muss im Code geändert werden.

Weitere Syntax	
// Kommentar...	einzeiliger Kommentar
/* Kommentar... */	mehrzeiliger Kommentar
#define PIN 13	definiere Konstante
#include <library.h>	Library einfügen

Operatoren	
=	definieren
+, -	Addition, Subtraktion
*, /	Multiplikation, Division
%	Modulo / Rest
==	ist gleich?
!=	ist nicht gleich?
<	ist kleiner
>	größer
<=	kleiner oder gleich
>=	größer oder gleich

Gemischte Zuweisung	
x ++	Vergrößerung von x um 1
x --	Verkleinerung von x um 1
x += y	Vergrößerung von x um y
x -= y	Verkleinerung von x um y
x *= y	Multiplikation von x mit y
x /= y	Division von x durch y

Konstanten	
true, false	
HIGH, LOW	
INPUT, OUTPUT	

Datentypen	
void	
boolean	0, 1, true oder false
char	z.B. 'a' oder -128 bis 127

Datentypen (cont)	
unsigned char	0 bis 255
int	-32.768 bis 32.767
unsigned int	0 bis 65.535
long	-2.147.483.648 bis 2.147.483.647
float	-3,4028235*10 <sup>38</sup> bis 3,4028235*10 <sup>38</sup>

Arrays	
int meineInts[6];	erzeugt int Array mit 6 Stellen
int meinePins[5];	erzeugt und füllt Array mit 5 Stellen
= 2,4,6,8,10;	
int meineWerte[6];	erzeugt Array mit 6 Stellen und füllt 5 davon
= 2,-4,9,3;	

Strings	
char S1[15];	erzeugt String mit 15 Stellen
char S2[10];	erzeugt String mit 10 Stelle und füllt die ersten 7
= 'A','r','d','u','i','n','o';	
char S3[] = "Arduino";	erzeugt String und füllt ihn mit "Arduino"
char S4[15] = "Arduino";	erzeugt String mit 15 Stelle und füllt die ersten 7 mit "Arduino"

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

Bedingungen	
static	erzeugt Variable die zwischen Funktionsaufrufen nicht gelöscht wird
volatile	erzeugt Variable die von Interrupts verändert werden kann
const	erzeugt unveränderbare Variable

Digital I/O	
pinMode (Pin, mode)	ändert Pin zu INPUT, OUTPUT oder INPUT_PULLUP
digitalWrite(Pin, Wert)	schaltet Output ein (1/HIGH) oder aus (0/LOW)
int digitalRead(Pin)	liest ob Pin ein oder ausgeschaltet ist (0/1)

Analog I/O	
int analogRead(analogPin)	liest Spannung auf analogPin als Wert zwischen 0 und 1023
analogReference(mod-e)	legt Referenzspannung für HIGH fest
DEFAULT, INTERNAL, EXTERNAL	Spannungsreferenzmodi
analogWrite(Wert)	erzeugt PWM-Welle mit einem Wert zwischen 0 und 255



Interrupts	
<b>digitalPinToInterrupt(Pin)</b>	wandelt digitalen <i>Input-Pin</i> in Interrupt-Pin um
<b>LOW, CHANGE, RISING, FALLING</b>	Interrupt-Typen
<b>attachInterrupt(Pin, Function, Typ)</b>	verbindet <i>Interrupt-Pin</i> mit <i>Funktion</i>
<b>detachInterrupt(Pin)</b>	hebt Verbindung zwischen <i>Funktion</i> und <i>Interrupt-Pin</i> auf
<b>interrupts()</b>	aktiviert Interrupts
<b>noInterrupts()</b>	deaktiviert Interrupts

Fortgeschrittene I/O	
<b>tone(Pin, Frequenz)</b>	<i>Frequenz</i> in Hz wird auf <i>Pin</i> ausgegeben
<b>tone(Pin, Frequenz, Dauer)</b>	auf <i>Pin</i> wird eine <i>Frequenz</i> für <i>Dauer</i> Millisekunden ausgegeben
<b>noTone(Pin)</b>	Tonausgabe auf <i>Pin</i> wird beendet
<b>pulseIn(Pin, Wert)</b>	misst die Dauer die <i>Pin</i> <i>*Wert</i> annimmt (HIGH/LOW)

Zeit	
unsigned int	Millisekunden seit Programmstart. ~50
<b>millis()</b>	Tage Overflow

Zeit (cont)	
unsigned int	Microsekunden seit Programmstart. ~70 min bis Overflow
<b>delay(ms)</b>	Pause für <i>ms</i> Millisekunden
<b>delayMicroseconds(us)</b>	Pause für <i>us</i> Microsekunden

Mathematik	
<b>min(x, y), max(x, y), abs(x)</b>	Minimum, Maximum, absoluter Wert
<b>pow(Basis, Exponent)</b>	BasisExponent

<b>sqrt(x)</b>	Wurzel(x)
<b>sin(x), cos(y), tan(z)</b>	Sinus, Cosinus, Tangens
<b>map(Wert, lim1U, lim1O, lim2U, lim2O*)</b>	skaliert <i>Wert</i> zwischen <i>lim1U</i> und <i>lim1O</i> zu Wert zwischen <i>lim2U</i> und <i>lim2O</i>
<b>constrain(Wert, limU, limO)</b>	beschränkt <i>Wert</i> zwischen <i>unterem</i> und <i>oberem Limit</i>

Zufallszahlen	
<b>randomSeed(Wert)</b>	Ausgangswert für Zufallsgenerator
<b>random(-max)</b>	Zufallszahl zwischen 0 und <i>max</i>
<b>random(-min, max)</b>	Zufallszahl zwischen <i>min</i> und <i>max</i>

Serial	
<b>Serial.begin(9600);</b>	startet <b>Serial</b> -verbindung mit Baudrate
<b>Serial.println(Inhalt);</b>	sendet Zeile mit <i>Inhalt</i>
<b>Serial.print(Text);</b>	sendet <i>Text</i>
<b>Serial.write(daten);</b>	sendet <i>Daten</i> als Binärcode
<b>Serial.flush();</b>	wartet bis <i>Daten</i> gesendet sind
<b>Serial.end();</b>	beendet <b>Serial</b> -Verbindung

Servo	
<b>#include &lt;Servo.h&gt;</b>	fügt <b>Servo</b> -Library hinzu
<b>attach(pin)</b>	fügt <i>Servopin</i> hinzu
<b>write(Winkel)</b>	stellt Servomotor auf <i>Winkel</i> (0-180°)
<b>read()</b>	liest Winkel von Servomotor

