# Arduino Grundbefelsübersicht

Alle angegeben Informationen wurden von der Seite http://www.arduino-tutorial.de/ entnommen. Diese Seite ist für den Einstieg in die Arduino Welt sehr zu empfehlen.

## Arduino Software

Die Arduino Software kann auf http://arduino.cc/en/Main/Software runtergeladen werden.

Ganz oben befindet sich das Hauptmenü, darunter einige Symbole:

 Kompilieren (Programm auf Fehler überprüfen)
 Stop (kompilieren abbrechen)
 neuen Sketch erstellen (Programme in Arduino heißen Sketch)
 Sketch öffnen
 Sketch speichern
 Sketch auf Arduino-Board übertragen
 seriellen Monitor öffnen

## Struktur eines Sketches

```
void setup(){

}

 void loop() {

}
```

Ein Programm für Arduino sollte die beiden oberen Methoden immer beinhalten. In der ersten Methode `void setup()` werden Grundeinstellungen vorgenommen. Z.B. ob ein Kanal In-oder Output ist. setup() wird nur bei Programmstart ausgeführt.
Die Methode `void loop()` beinhaltet den eigentlichen Programmablauf. Sie wird immer wieder wiederholt.

# Digital Output

`pinMode(Pinnummer,OUTPUT);` ist eine Funktion, bei der ein digitaler Kanal des Arduino-Boards, der als Output deklariert ist, ein oder ausgeschaltet werden kann. ‚Ein'- oder ‚Aus' ist in diesem Fall eigentlich nicht ganz korrekt, denn der Kanal kann je nach Anweisung entweder ein 5V+ oder ein GND (Minus-Pol) sein.

`pinMode(Pinnummer, OUTPUT);`
Der Befehl `pinMode()` setzt den Port als Ausgang (OUTPUT).
Pinnummer: Nummer des digitalen Pins, dass als Ausgang deklariert(benannt) werden soll.
OUTPUT: Funktion des Kanals, hier Output (z.B. zur Steuerung von LEDs, DC Motoren…)

Ist ein Pin als OUTPUT deklariert, kann man folgende Befehle verwenden, um auf sie Einfluss zu nehmen:

`digitalWrite(Pinnummer, HIGH);`
Dieser Befehl schaltet den Pin mit der angegebenen Nummer auf 5V /HIGH bzw AN;
`digitalWrite(Pinnummer, LOW);`
Dieser Befehl schaltet den Pin mit der angegebenen Nummer auf GND /LOW bzw. AUS.

Beispielprogramm:

```
int ledPin = 13;

void setup() {
     pinMode(ledPin, OUTPUT);
}

void loop(){
     digitalWrite(ledPin, HIGH);
     delay(1000);
     digitalWrite(ledPin, LOW);
     delay(1000);
}
```

Die LED am Pin 13 blinkt im Takt von 2 Sekunden.

# Digital Input

Um digitale Signale auszulesen (Schalter, Taster, usw.) muss man eines der digitalen Pins als Eingang deklarieren, anschließend kann man den momentan Zustand des Pins mithilfe des Befehls `digitalRead()` auslesen.

`pinMode(Pinnummer, INPUT);`
Der Befehl `pinMode()` setzt den Port mit als Eingang(INPUT).
Pinnummer: Nummer des digitalen Pins, dass als Eingang deklariert werden soll.
INPUT: Funktion des Kanals, hier INPUT (z.B. zum Auslesen von Schalter-, Tasterzuständen…)

Ist ein Pin als INPUT deklariert kann man sie mit dem folgenden Befehlen auslesen:

`digitalRead(Pinnummer);`
mit dem Befehl `digitalRead();` lässt sich auslesen, ob an dem angegebenen Pin 5V/ HIGH oder 0V /GND/LOW anliegt. `digitalRead();` kann nur die Werte 1 oder 0 bzw HIGH oder LOW annehmen
[in der Abfrage sind beide Angaben möglich: if (digitalRead(13)==HIGH) ⇔ if (digitalRead(13)==1) ]

Beispielprogramm:

```
const int buttonPin = 2;

const int ledPin = 13;

int buttonState = 0;

void setup() {

pinMode(ledPin, OUTPUT);

pinMode(buttonPin, INPUT);

}

void loop(){

buttonState = digitalRead(buttonPin);

if (buttonState == HIGH) {

digitalWrite(ledPin, HIGH);

}

else {

digitalWrite(ledPin, LOW);

}

}
```

Liest einen Taster aus, der an buttonPin=2 geschaltet ist. Je nachdem ob der Taster gedrückt ist, schaltet er die LED, die an LedPin=13 anschlossen ist, AN (wenn Taster gedrückt) bzw. AUS (wenn Taster nicht gedrückt.)

# Analog Output

Mit den analogen Ausgängen (diese sind auf dem Board mit „PWM“ bzw- „~“ gekennzeichnet) lassen sich auch zwischen Spannungsgrößen (zwischen 0V und 5V) erzeugen. Dabei wird der analoge Ausgangs Pin immer wieder ein- und ausgeschaltet. Dieses Verfahren wird auch Fading genannt. WICHTIG eine Deklaration als analoger Ausgang ist nicht notwendig.

```
analogWrite(Pinnummer, WERT);
```
Pinnummer: sechs der digitalen Pins sind nicht nur digital, sondern analog ansteuerbar (mit Aufdruck „~“ bzw „PWM“ gekennzeichnet)
WERT: zwischen 0 -255 ( 0=0V und 255=5V z.B. für Einstellung der LED Helligkeit)

Beispielprogramm:

```
int ledPin = 9;

void setup() {

}

void loop() {

for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5) {

analogWrite(ledPin, fadeValue);

delay(30);

}

for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {

analogWrite(ledPin, fadeValue);

delay(30);

}

}
```

# Analog In

Die analogen Eingangspins sind alle mit einem „A“vor der Nummer gekennzeichnet. WICHTIG eine Deklaration als analoger Eingang ist nicht notwendig.

```
analogRead(sensorPIN);
```
sensorPIN: der analoger Pin sensorPIN wird ausgelesen. Es werden Werte zwischen 0 und 1023 ausgegeben (0= keine Spannung  1023= max. Potential liegt an dem Punkt an 5V)

# Serielle Kommunikation

`Serial.begin(Baudrate)`
Baudrate: Übertragungsrate (Standard Baudrate sind 9600). Im Setup wird die serielle Kommunikation mit dem Befehl Serial.begin(9600); gestartet

Beispielprogramm:

```
int potPin = 0;

int button1 = 3;

int button2 = 4;

int button3 = 5;

void setup(){

Serial.begin(9600);

pinMode(button1,INPUT);

pinMode(button2,INPUT);

pinMode(button3,INPUT);

}

void loop(){

Serial.print(digitalRead(button1));

Serial.print(",");

Serial.print(digitalRead(button2));

Serial.print(",");

Serial.print(digitalRead(button3));

Serial.print(",");
```

```
Serial.println(analogRead(potPin));


delay(10);


}
```

Pro Programmdurchlauf wird eine Zeichenkette der Form 0,0,0,0 + Zeilenumbruch versendet. Man kann sich diese im serielen Monitor anzeigen lassen.

## Variablen

### int

```
int meinWert = 10;
```

### long

```
long meinWert = 1000;
```

### float

```
float meinWert = 2.5;
```

### char

```
char meinBuchstabe = ‘a‘;
```

### Arrays

```
int meineWerte[5] = {10,12,32,46,50};
```

## Abfragen

### if-Abfrage

```
if (digitalRead(btnPin)==HIGH) {
digitalWrite(ledPin,HIGH);                // Anweisungsblock für wahr
}
 else {
digitalWrite(ledPin,LOW);    // Anweisungsblock für falsch
}
```

### switch-case-Abfrage

```
switch (meineVariable) {
case 1:
befehl1;
break;
case 2:
befehl2;
break;
default:
befehl3;
break;
}
```

# Schleifen

for-Schleife

```
for (int i=0; i<10; i++){
// Anweisungen
}
```

do-while Schleife

```
do
{
delay(50);
x = analogRead(3); // prüft den Sensorwert am Pin 3
} while (x < 100);
```

# Methoden

## Beispiel 1

```
void blinken(){
// Anweisungsblock Start
digitalWrite(ledPin, HIGH);
delay(500);
digitalWrite(ledPin, LOW);
delay(500);
// Anweisungsblock Ende
}
void blinken(int thePin, int dauer){
digitalWrite(thePin, HIGH);
delay(500);
digitalWrite(thePin, LOW);
delay(500);
}
```

## Beispiel 2

```
float quadrat(float x){
float ergebnis = x*x;
return ergebnis;
}
```

Der Aufruf wäre z.B.:

```
wert = quadrat(12.3);
```

## *Grundbefehle*

**pinMode()**
pinMode(3,OUTPUT);  // setzt den digitalen Kanal 3 als Ausgang

**digitalWrite()**
digitalWrite(3,HIGH);  // Schaltet 5V+ auf den digitalen Kanal 3

**digitalRead()**
digitalRead(4);  // liefert HIGH oder LOW

**analogWrite()**
analogWrite(3,200);  // am digitalen Kanal 3 wird werden 4V+ angelegt

**analogRead()**
analogRead(1);  // liefert den anliegenden Wert vom analogen Kanal 1

**delay()**
delay(1000);  // der Programmablauf wird eine Sekunde verzögert

**Serial.begin()**
```
void setup(){
Serial.begin(9600);  // Startet die Datenübertragung mit 9600 Baud
}
```

**Serial.println()**
Serial.println(analogRead(1));  // Sendet den analogen Wert am Kanal 1 an den Computer

# Operatoren

| Arithmetische Operatoren | | | |
|---|---|---|---|
| **OPERATOR** | **BEDEUTUNG** | **ANWENDUNG** | **FUNKTION** |
| = | Zuweisung | a=2*b | Weist der linken Seite den Wert auf der rechten Seite zu. |
| + | Addition | a=a+b | |
| - | Subtraktion | a=b-c | |
| ++ | Inkrementieren | a++ | Zählt zur der Variable 1 hinzu (+1) |
| - | Dekrementieren | a- | Zieht von der Variable 1 ab (-1) |
| * | Multiplikation | a=b*c | |
| / | Division | a=b/c | Dabei darf c nie gleich Null sein |
| % | Modulo | a=b%c<br>a=7%5 ; a=2<br>a=10%5 ; a=0 | Liefert den Rest bei der Division von b/c. Ist b durch c teilbar, so ist das Ergebnis = 0. |
| **Vergleichsoperatoren** | | | |
| == | Gleichheit | a==b | Prüft auf Gleichheit. |
| != | Ungleichheit | a!=b | Prüft auf Ungleichheit |
| < | kleiner als | a<b | |
| > | größer als | a>b | |
| <= | kleiner gleich | a<=b | |
| >= | größer gleich | a>=b | |
| **Boolsche Operatoren(können wahr oder falsch sein)** | | | |
| && | UND | (a==2)&&(b==5) | Wenn beide Seiten wahr sind, ist das Ergebnis auch wahr. |
| \|\| | ODER | (a==2)\|\|(b==5) | Wenn eine oder beide Seiten wahr sind, ist das Ergebnis wahr. |
| ! | NICHT | !(a==3) | |