Arduino für FunkAmateure

Arduino & graphische Programmiersprachen

- Was will ich machen?
- Aufgabe, Arduino-Sketch, Schaltbild
- Arduino-IDE Editor
- Arduino-IDE „Programming Cheat Sheet“
- Arduino-IDE Vor- & Nachteile
- Alternative: Ardublock-Sketch
- Alternative: Mixly 0.98 von Microduino (ähnlich Scratch)
- Alternative: miniBloq v0.83
- Ardublock Vor- & Nachteile
- Mixly 0.98 von Microduino Vor- & Nachteile
- miniBloq v0.83 Vor- & Nachteile
- Ardublock in der Arduino-IDE starten
- Ardublock als paralleles Fenster zur Arduino-IDE
- Ardublock Oberfläche
- Programm erstellen „ArduBlock_01.adp“
- Informationen zu Ardublock
- Arduino-IDE & Ardublock Kompatibilität
- Ardublock Installation I bis III
- Ardublock Tutorials & Beispiele
- Zusatzmaterial

H39@email.de

Was will ich machen?	
	Eine Alternative zur Arduino-IDE finden
	Vergleich Arduino-IDE zu grafischen Programmierumgebungen
	Am Beispiel „Taster“ Programme mit Ardublock / Mixly 0.98 / miniBloq erstellen
	Ardublock Installation und Funktion darstellen
	Blinkende LED mit Ardublock aufbauen
	Blinkende LED mit Mixly 0.98 aufbauen
	Blinkende LED mit miniBloq aufbauen
	Informationen zu Ardublock

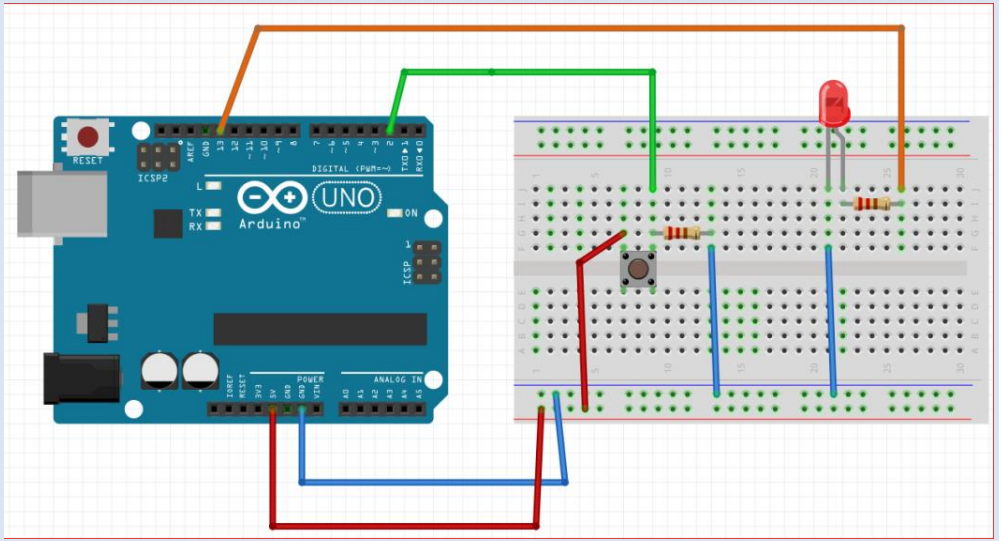
Aufgabe, Arduino-Sketch, Schaltbild

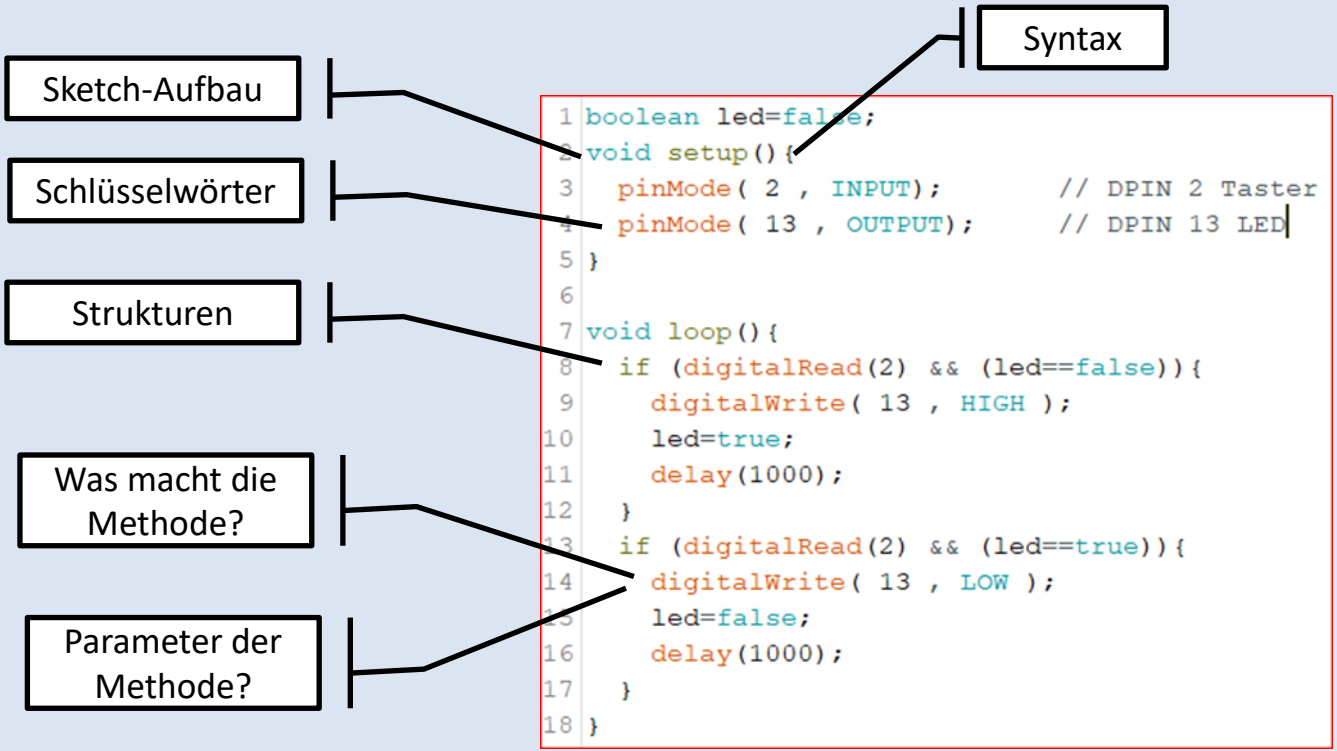
Aufgabe	Taster gedrückt, dann LED an. Taster gedrückt, dann LED aus.

Arduino-Sketch

```
boolean led=false;
void setup(){
  pinMode( 2 , INPUT);        // DPIN 2 Taster
  pinMode( 13 , OUTPUT);      // DPIN 13 LED
}

void loop(){
  if (digitalRead(2) && (led==false)){
    digitalWrite( 13 , HIGH );
    led=true;
    delay(1000);
  }
  if (digitalRead(2) && (led==true)){
    digitalWrite( 13 , LOW );
    led=false;
    delay(1000);
  }
}
```

Schaltung





Kaum Unterstützung beim Editieren

Schreibweise der Schlüsselwörter (Groß- und Kleinschreibung).
Keine Vervollständigung der Schlüsselwörter beim Schreiben.
Keine Vorschlagsliste der Methoden/Parameter…

Arduino Programming Cheat Sheet

Primary source: Arduino Language Reference
https://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) { ... }
for (int i = 0; i < 10; i++) { ... }
break;     // Exit a loop immediately
continue;  // Go to next iteration
switch (var) {
  case 1:
    ...
    break;
  case 2:
    ...
    break;
  default:
    ...
}
return x;  // x must match return type
return;    // For void return type
```

Function Definitions

```
<ret. type> <name>(<params>) { ... }
e.g. int double(int x) {return x*2;}
```

Operators

General Operators

=	assignment		
+	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

Pointer Access

&	reference: get a pointer
*	dereference: follow a pointer

Built-in Functions

Pin Input/Output

```
Digital I/O - pins 0-13 A0-A5
  pinMode(pin,
    [INPUT, OUTPUT, INPUT_PULLUP])
  int digitalRead(pin)
  digitalWrite(pin, [HIGH, LOW])

Analog In - pins A0-A5
  int analogRead(pin)
  analogReference(
    [DEFAULT, INTERNAL, EXTERNAL])

PWM Out - pins 3 5 6 9 10 11
  analogWrite(pin, value)
```

Advanced I/O

```
tone(pin, freq_Hz)
tone(pin, freq_Hz, 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) // 0 to max-1
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(val)     byte(val)
int(val)      word(val)
long(val)     float(val)
```

External Interrupts

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

Libraries

Serial - comm. with PC or via RX/TX

```
begin(long speed)  // Up to 115200
end()
int available()  // #bytes available
int read()   // -1 if none available
int peek()   // Read w/o removing
flush()
print(data)    println(data)
write(byte)    write(char * string)
write(byte * data, size)
SerialEvent()  // Called if data rdy
```

SoftwareSerial.h - comm. on any pin

```
SoftwareSerial(rxPin, txPin)
begin(long speed)  // Up to 115200
listen()       // Only 1 can listen
isListening()  // at a time.
read, peek, print, println, write
  // Equivalent to Serial library
```

EEPROM.h - access non-volatile memory

```
byte read(addr)
write(addr, byte)
EEPROM[index]  // Access as array
```

Servo.h - control servo motors

```
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()
```

Variables, Arrays, and Data

Arduino-IDE Vor- & Nachteile

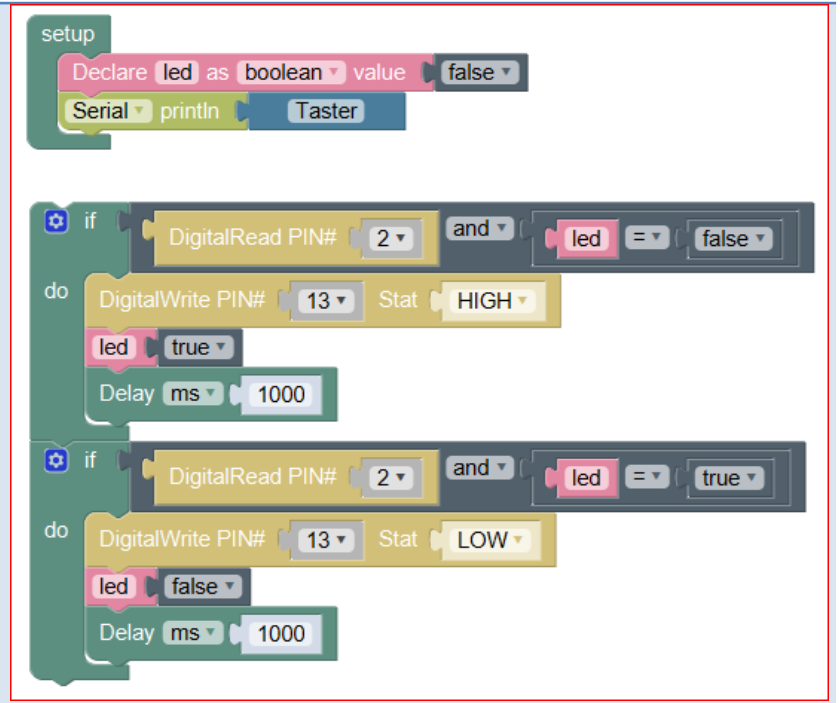
Arduino-IDE Überblick	Gedacht für:	Ausbildung
	Geschrieben in:	Java
	Letzte Version:	1.8.5 (https://www.arduino.cc)
Vorteile	• Editor mit grundlegenden Eigenschaften (Zeilennummerierung) • Übersichtliche Menüführung • Schlüsselwörter werden farblich hervorgehoben • Bleibt ein Schlüsselwort „schwarz“, ist die Schreibweise falsch • ***Da auf unterster Ebene programmiert wird, gibt es keine Einschränkungen beim kodieren***	
Nachteile	• Erlernen der Semantik und Syntax der Arduino-Programmiersprache als Erweiterung von C++. • Keine Code-Vervollständigung • Keine Parametervorgaben	
Editor-Varianten	https://uecide.org/download (einfacher Editor)	
	https://github.com/Sloeber/arduino-eclipse-plugin/releases/download/V4_3/V4.3_win64.2018-08-06_08-21-36.tar.gz (Editor als Plugin zum eclipse-Editor)	

Alternative: Ardublock-Sketch

```
bool _ABVAR_1_led= false ;

void setup()
{
  pinMode( 2 , INPUT);
  pinMode( 13 , OUTPUT);
  _ABVAR_1_led = LOW ;

}

void loop()
{
  if (( digitalRead(2) && ( ( _ABVAR_1_led ) == ( LOW ) ) ))
  {
    digitalWrite( 13 , HIGH );
    _ABVAR_1_led = HIGH ;
    delay( 1000 );
  }
  if (( digitalRead(2) && ( ( _ABVAR_1_led ) == ( HIGH ) ) ))
  {
    digitalWrite( 13 , LOW );
    _ABVAR_1_led = LOW ;
    delay( 1000 );
  }
}
```

Alternative: Mixly 0.98 von Microduino (ähnlich Scratch)

```
boolean led;

void setup()
{
  led = false;
  Serial.begin(9600);
  Serial.println("Taster");
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
}

void loop()
{
  if (digitalRead(2) && led == false) {
    digitalWrite(13,HIGH);
    led = true;
    delay(1000);

  }
  if (digitalRead(2) && led == true) {
    digitalWrite(13,LOW);
    led = false;
    delay(1000);

  }

}
```

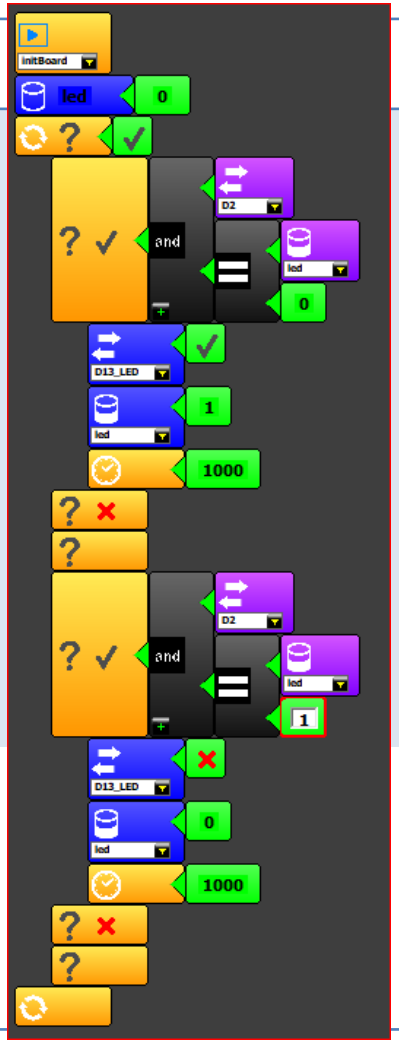


Vorteile:

- Baukasten mit Bausteinen statt Schlüsselwörter
- Syntax automatisch
- Strukturen aus Bausteinen
- Parameter aus Pulldown-Listen

Alternative: miniBloq v0.83

```
void setup()
{
   initBoard();
   float led = 0;
   while(true)
   {
      if((DigitalRead(D2)&&((int)(led)==(int)(0))))
      {
         DigitalWrite(D13_LED, true);
         led = 1;
         delay(1000);
      }
      else
      {
      }
      if((DigitalRead(D2)&&((int)(led)==(int)(1))))
      {
         DigitalWrite(D13_LED, false);
         led = 0;
         delay(1000);
      }
      else
      {
      }
   }
}

void loop()
{
}
```



Vorteile:

- Blöcke statt Schlüsselwörter
- Syntax automatisch
- Strukturen aus Blöcken
- Parameter aus Pickup-Listen

Ardublock Vor- & Nachteile		
Ardublock Überblick	Gedacht für: Geschrieben in: Letzte Version:	Ausbildung Java 04.04.2018 (von arduino-basics)
Vorteile	• Baukasten mit Bausteinen statt Schlüsselwörter • Syntax automatisch • Strukturen aus Bausteinen • Parameter aus Pickup-Listen	
Prinzip	Die Blöcke bieten Anfasser (Noppen & Buchten), die Syntax-Fehler ausschließen.	
Nachteile	Ist kein Modul/Block vorhanden, z.B. für den DHT11-Sensor, ist ein Sketch nur schwierig möglich.	
Varianten	Im Internet sind eine Reihe von Ardublock-Varianten zu finden, die sich in ihrer Ausstattung unterscheiden.	
Community	Eine Vielzahl von unterstützenden Webseiten…	

Mixly 0.98 von Microduino Vor- & Nachteile	
Mixly 0.98 Überblick	Gedacht für: Ausbildung Geschrieben in: Java
Vorteile	• Baukasten mit Bausteinen statt Schlüsselwörter • Syntax automatisch • Strukturen aus Bausteinen • Parameter aus Pickup-Listen
Prinzip	Die Bausteine bieten Anfasser (Noppen & Buchten), die Syntax-Fehler ausschließen.
Nachteile	Unterstützt insbesondere die Microduino-Plattform!
	Microduino wird als Paket zusammen mit der Arduino-IDE 1.6.7 angeboten
Download	http://www.microduinoinc.com/downloads/mdxly.zip
Plattform	http://wiki.microduinoinc.com/Microduino_Modules

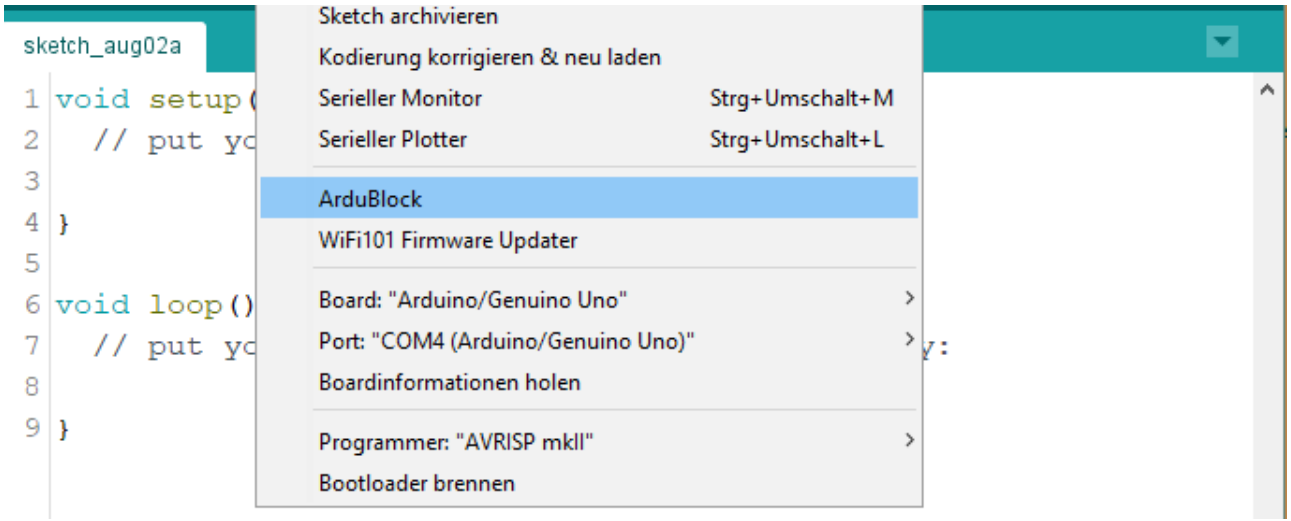
miniBloq v0.83 Vor- & Nachteile	
miniBloq v0.83 Überblick	Gedacht für: Ausbildung Geschrieben in: Java
Vorteile	• Blöcke statt Schlüsselwörter • Syntax automatisch • Strukturen aus Bausteinen • Parameter aus Pickup-Listen • Gute Dokumentation auf der Webseite
Prinzip	Die Bausteine bieten Anfasser (Noppen & Buchten), die Syntax-Fehler ausschließen. Bausteine bieten Pickup-Listen zur schnellen Zuordnung von Parametern
Nachteile	Bausteine sind eng mit Pickup-Listen verbunden! Daher kann einer Variablen kein digitaler PIN zugeordnet werden.
	miniBloq v0.83 wird als Paket angeboten
Download	http://miniblog.td-er.nl/miniBlog.v0.83.exe
Beispiele	http://blog.miniblog.org/p/tutorials-and-examples.html

Ardublock in der Arduino-IDE starten

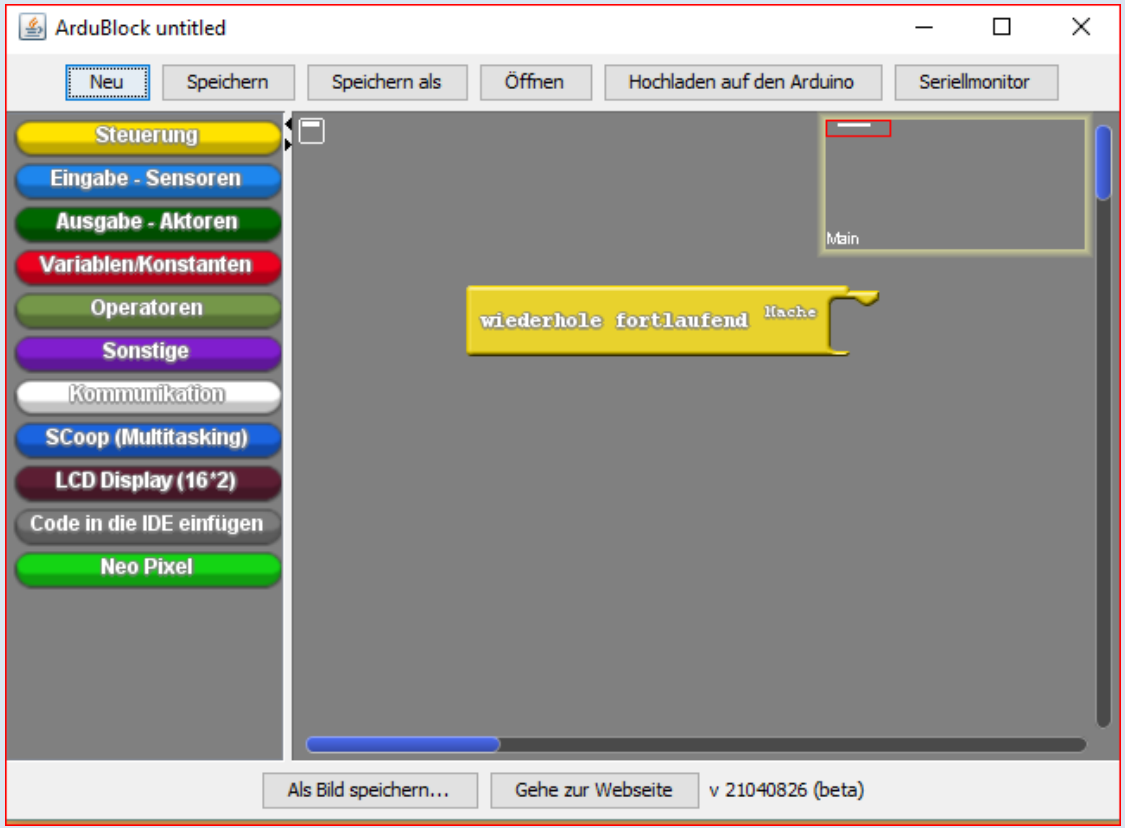
1. Arduino-IDE starten.

2. Innerhalb der Arduino-IDE Bord & Port auswählen.

3. Unter Werkzeuge findet sich der neue Menüpunkt „ArduBlock“.



Ardublock als paralleles Fenster zur Arduino-IDE



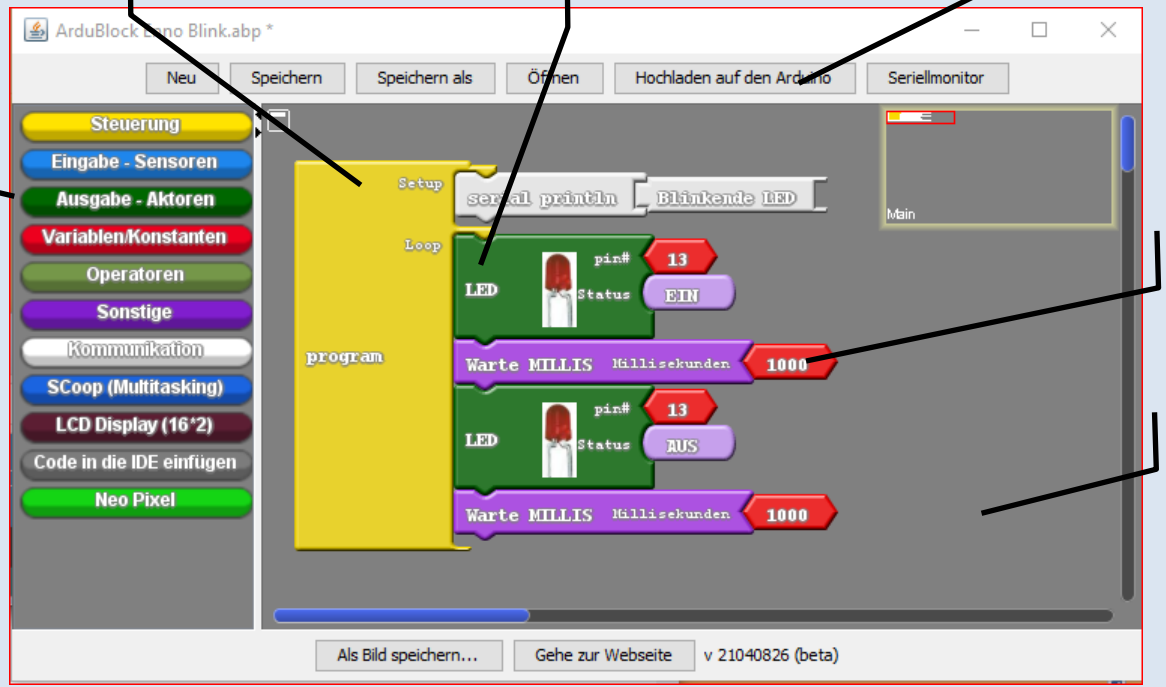
Ardublock Oberfläche

Block „program“ aus „Steuerung“

Block „LED“ aus „Ausgabe Aktoren“

Schaltflächen

Blöcke zur Auswahl



Eingabefelder

Programm-Fenster

Programm erstellen „ArduBlock_01.adp“

Je nach Ardublock-Version weichen die Block-Bezeichnungen ab!

Nach Auswahl der Rubrik, z.B. „Steuerung“, wird der Block „program“ per Drag & Drop in das Programmfenster gezogen.

Passende Blöcke fügen sich an den Anfassern automatisch zusammen (Klickgeräusch).

Durch Klicken auf die Schaltfläche „Hochladen auf den Arduino“ wird:

1. Aus dem Ardublock-Programm der C++ Code in der Arduino-IDE erzeugt.
2. Das Programm temporär als Sketch, z.B. „TemName.ino“, gespeichert.
3. Der Sketch wird automatisch kompiliert und hochgeladen.

Der Sketch wird ausgeführt.

Achtung: Das Ardublock-Programm ist noch nicht gespeichert!

Durch Klicken auf die Schaltfläche „Speichern als“ wird das Ardublock-Programm in einem geeigneten Ordner, z.B. in „…\Dokumente\ArduBlock\“ als ArduBlock_01.adp“ gespeichert.

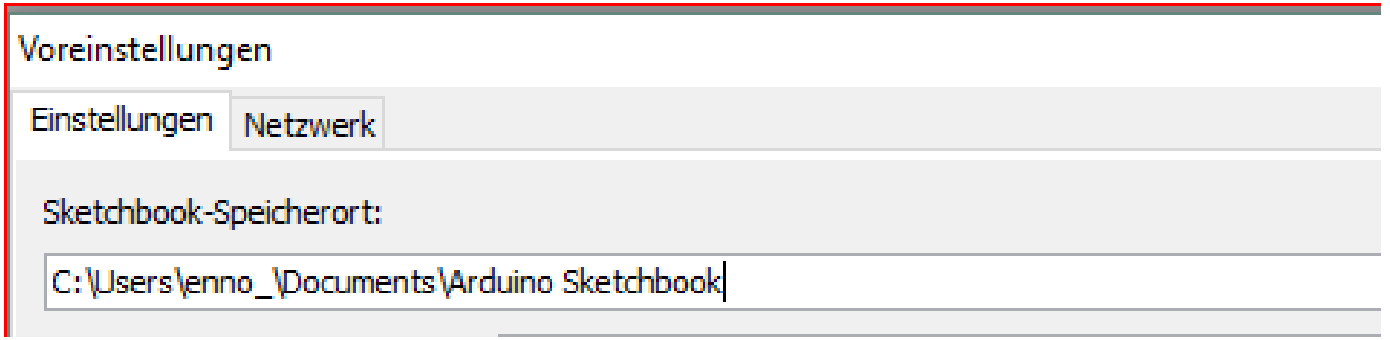
Informationen zu Ardublock	
Empfohlene Webseite: „arduino-basics“	http://arduino-basics.com Hier findet sich der Download für eine zur „Arduino-IDE Version 1.8.5“ kompatible Ardublock Version.
Anleitungen Installation & Einarbeitung	
Stefan Baireuther	www.baireuther.de-page-arduino
Kreativ Kiste	https://www.kreativekiste.de/ardublock-arduino-grafisch-programmieren
RWTH-Aachen	https://schuelerlabor.informatik.rwth-aachen.de/modulmaterialien/ardublock
Ardublock (veraltet)	http://blog.ardublock.com

Arduino-IDE & Ardublock Kompatibilität	
Ardublock für 1.8.5	http://arduino-basics.com/data/documents/ardublock-all-20180404.jar
Ardublock Kompatibilität ☹	Der auf der Webseite „http://blog.ardublock.com“ zu findende Download ist veraltet. Die Version läuft nicht unter der Arduino-IDE Version 1.8.5 (nur bis 1.6.9).
☹	Der auf der Webseite „https://sourceforge.net/projects/ardublock/files/latest/download“ zu findende Download ist veraltet.
☹	Der auf der Webseite „https://github.com/letsgoING/ArduBlock “ zu findende Download ist veraltet.
Arduino-IDE & Ardublock als **Paket**	
1. Alternative (veraltet)	Vollständige Installation von „letsgoING“: Arduino-IDE 1.6.7 & Ardublock Version 5 https://github.com/letsgoING/Arduino
2. Alternative	Vollständige Installation von „Duino EDU “: Arduino-IDE 1.8.5 & Ardublock Duino EDU www.duinoedu.com-arduinoaugmente-default.html Dieses Paket ist für fortgeschrittene Anwender ein Muss.

Installation I	
Arduino-IDE	Version 1.8.5 installieren (ohne Ardublock)
Download Arduino-IDE	https://www.arduino.cc/download_handler.php?f=/arduino-1.8.5-windows.zip
Empfohlener Download Ardublock	http://arduino-basics.com/data/documents/ardublock-all-20180404.jar
	Im Download-Verzeichnis findet sich die Datei: ardublock-all-20180404.jar

Installation II

Aduino-IDE	Voraussetzung für die weitere Installation ist eine vorhandene Arduino-IDE (hier 1.8.5)
Variante 1	Arduino-IDE ist unter „C:\Program Files (x86)“ installiert. Arduino-IDE starten und Sketchbook-Speicherort belassen oder ändern. In der Arduino-IDE „Sketchbook-Verzeichnis“ festlegen. Sketchbook-Speicherort merken.



Die Verzeichnisse „…\tools\ArduBlockTool\tool\“ im Sketchbook-Speicherort erstellen.

Die Datei „ardublock-all-20180404.jar“ kopieren nach:

Bei mir: „C:\Users\DeinName\Documents\Arduino Sketchbook\tools\ArduBlockTool\tool\“

Installation III	
Aduino-IDE	Voraussetzung für die weitere Installation ist eine vorhandene Arduino-IDE (hier 1.8.5)
Variante 2	Arduino-IDE ist unter „C:\“ installiert. Zum Beispiel: „C:\Arduino 1.8.5“ Im vorhandenen Verzeichnis „tools“ die Verzeichnisse „…\ArduBlockTool\tool\“ erstellen. Die Datei „ardublock-all-20180404.jar“ kopieren nach:
	„C:\Arduino 1.8.5\tools\ArduBlockTool\tool\“

Tutorials & Beispiele	
„arduino-basics“	Erste Quelle für die Einarbeitung: http://arduino-basics.com
Stefan Baireuther	www.baireuther.de-page-arduino
Kreativ Kiste	https://www.kreativekiste.de/ardublock-arduino-grafisch-programmieren
„dfrobot“	http://image.dfrobot.com/image/data/KIT0017/ardublock_tutorial.zip „ArduBlock_tutorial.pdf“
RWTH Aachen	https://schuelerlabor.informatik.rwth-aachen.de/modulmaterialien/ardublock „Modulhandbuch_ArduBlock.pdf“
„sourceforge“	https://sourceforge.net/projects/ardublock/
„Github“	https://github.com/letsgoING/ArduBlock „Blockreferenz_ArduBlock_letsgoING.pdf“

Variablen	
Ganzzahlen: int	-32768 bis + 32767 2 Byte groß (Arduino UNO)
Wahrheitswerte: boolean (C++ bool)	Zwei mögliche Werte: true oder false auch „1“ oder „0“ 1 Byte groß
Gleitkommazahlen: double	±3,4 * 10^{38} 4 Byte groß (Arduino UNO)
Zeichen: char	-128 bis +127 1 Byte groß
Deklaration	«Datentyp» «Bezeichner»; boolean pinStatus; int zaehler; double messwert; char zeichen;

Initialisierung	pinStatus = false;
	zaehler = 0;
	double = 3.1416;
	char = ´A´

Variablen	
Datentyp Ganzzahlen: **int**	-32768 bis + 32767 2 Byte groß (Arduino UNO)
Bezeichner	frei wählbarer Name (aus erlaubten Zeichen)
Deklaration	«Datentyp» «Bezeichner»;
	int zaehler;
Initialisierung	zaehler = 0;
Deklaration mit Initialisierung	int zaehler = 0;

Wo steht die Anweisung?	vor der setup()-Funktion
	innerhalb von setup(), loop() …
	innerhalb von Blöcken:
	{
	«Anweisung(en)»
	}

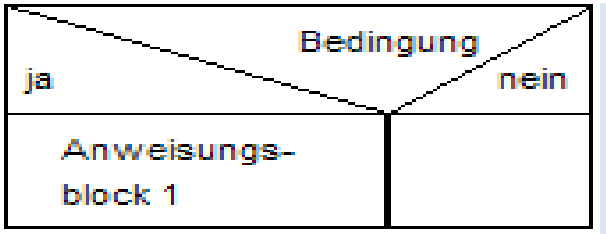
Variablen	
Datentyp Wahrheitswerte: **boolean** (C++ bool)	zwei mögliche Werte: true oder false auch „1“ oder „0“ 1 Byte groß
Bezeichner	frei wählbarer Name (aus erlaubten Zeichen)
Deklaration	«Datentyp» «Bezeichner»;
	boolean tasterStatus;
Initialisierung	tasterStatus = false;
Deklaration mit Initialisierung	boolean tasterStatus = false;

Wo steht die Anweisung?	vor der setup()-Funktion
	innerhalb von setup(), loop() …
	innerhalb von Blöcken:
	{
	«Anweisung(en)»
	}

Bedingungen	true, false if (Bedingung) …
true, false	**Logischer Zustand einer Bedingung** (Wahrheitswerte) „true“ und „false“ sind C++ Schlüsselwörter
Alternative Angaben	true = 1 (Wert 1), false = 0 (Wert 0) (alle Werte ungleich 0 ergeben „true“)
Beispiel: if (Bedingung) …	
Synonyme	HIGH = true = 1 ≙ 5 V; LOW = false = 0 ≙ 0 V; (nur im Kontext anwenden)

Zuweisung

Mathematik Arithmetischer Ausdruck	Programmiersprachen Zuweisung
3 + 4 = x + 2 x = 5	Verboten!
Das Gleichheitszeichen verlangt, das die Ausdrücke links und rechts davon gleich sind!	Das Zeichen „=“ ist als Zuweisungs-Operator zu verstehen! (Pascal „:=“)
	„a = 3 + 4“ bedeutet, das zuerst **rechts** vom „=“ ausgewertet wird, und das Ergebnis dann der Variablen **links** vom „=“ zugewiesen wird.
anzahl = anzahl + 1	anzahl = anzahl + 1
Darf man hinschreiben. Aber es gibt keine Lösung.	Hole den in der Variablen „anzahl“ hinterlegten Wert, addiere dazu „1“, und weise das Ergebnis der Variablen „anzahl“ zu.

Dem Sketch eine Struktur geben		
Verzweigung		
Umgangssprachlich	Falls die «Bedingung» zutrifft, dann führe die «Anweisung(en) » aus	Falls «ich Geld habe» zutrifft, dann «kaufe ich ein»
Informatik	Einfache Auswahl	

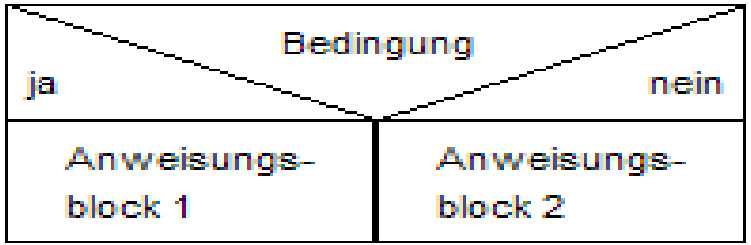
Arduino-Sprache (C++)	if («Bedingung»)
	{
	«Anweisung(en)»
	}	

boolean tasterStatus ;
tasterStatus = digitalRead(2);
if (tasterStatus == true)
{
digitalWrite(9, HIGH)
}
	Der Wert einer Bedingung kann nur „true“ oder „false“ sein.
	Wird der „tasterStatus“ geprüft, dann ergibt sich hier der Wert „true“ oder „false“ .
	Falls „true“ wird der true-Block (ja-Zweig) ausgeführt.
	Achtung: Der logische Operator ist „==“ und nicht „=“!	

Verzweigung

Zweifache Auswahl

```
if («Bedingung» )
 {
  «Anweisung(en)»
}
else
 {
  «Anweisung(en)»
}
```




```
int a, b, c;
a = 3;
b = 5;
if ( a == b)
{
  c = a * 2 * b;
}
else
{
  c = a;
}
```

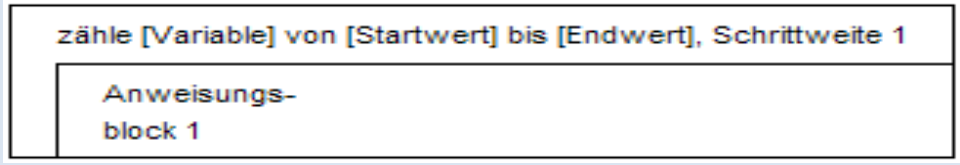
Der Wert einer Bedingung kann nur „true“ oder „false“ sein.
Werden die Variablen a und b auf Gleichheit geprüft, dann ergibt sich hier der Wert „false“.

Der else-Block (nein/false-Zweig) wird ausgeführt.

Achtung:
Der logische Operator ist „==“ und nicht „=“!

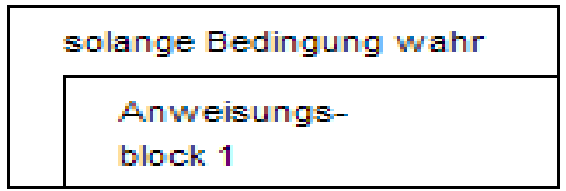
Dem Sketch eine Struktur geben

Zählergesteuerte Schleife	for («Start»;
	«Bedingung»;
	«Weiter»)
	{
	«Anweisung(en)»
	}




```
if ( int  zaehler = 1;  zaehler  <=  5;  zaehler = zaehler + 1 )
{
     digitalWrite (  5 + zaehler,  HIGH ) ;
}
```

Abweisende Schleife	while («Bedingung»)
	{
	«Anweisung(en)»
	}




```
int zaehler = 0;
while ( zaehler  <=  5 )  {
     //  «Anweisung(en)»
     zaehler = zaehler + 1;
 }
```