

Spickzettel C/C++ Programmierung

Grundstruktur eines Arduino-Programms

```
void setup() {  
  // Wird einmal beim Start ausgeführt  
  // Hier initialisieren wir Pins, Serial, etc.  
}  
  
void loop() {  
  // Wird endlos wiederholt  
  // Hier kommt der Hauptcode hin  
}
```

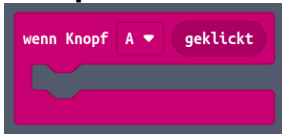


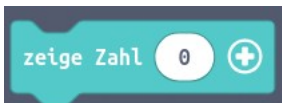
Variablen und Datentypen

Calliope	C/C++	Erklärung
Zahl	<code>int zahl = 42;</code>	Ganze Zahlen, typischerweise -32768 bis 32767
Kommazahl	<code>float komma = 3.14;</code>	Zahlen mit Nachkommastellen
Wahr/Falsch	<code>bool wahr = true;</code>	Logische Werte (true oder false)
Text	<code>String text = "Hallo";</code>	Zeichenketten

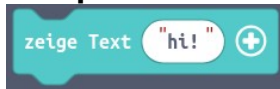
Zusätzliche wichtige Datentypen:

- `long`: Für größere ganze Zahlen
- `double`: Für präzisere Kommazahlen
- `char`: Für einzelne Zeichen

Ein- und Ausgabe

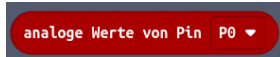
Calliope	C/C++ (Arduino)	Erklärung
	<code>digitalRead(BUTTON_A) == LOW</code>	Prüft, ob ein Knopf gedrückt ist
	<code>digitalWrite(LED_BUILTIN, LOW);</code>	Schaltet die eingebaute LED ein (invertiert)
	<code>digitalWrite(LED_BUILTIN, HIGH);</code>	Schaltet die eingebaute LED aus (invertiert)
	<code>Serial.println(zahl);</code>	Gibt eine Zahl über Serial aus

Calliope



Pin 0 als Eingang

Pin 1 als Ausgang



C/C++ (Arduino)

```
Serial.println("hi!");
```

```
pinMode(0, INPUT);
```

```
pinMode(1, OUTPUT);
```

```
int wert = analogRead(A0);
```

Erklärung

Gibt Text über Serial aus

Konfiguriert Pin 0 als Eingang

Konfiguriert Pin 1 als Ausgang

Liest einen analogen Wert (0-1023)

Wichtig für Serial:

```
void setup() {  
  Serial.begin(9600); // Startet die serielle Kommunikation  
}
```

Kontrollstrukturen

Wenn-Dann (If-Else)

Calliope



C/C++

```
if (Bedingung) {  
  // Anweisungen wenn Bedingung wahr  
} else {  
  // Anweisungen wenn Bedingung falsch  
}
```

Beispiel

```
if (temperatur > 25) {  
    digitalWrite(VENTILATOR_PIN, HIGH);  
} else {  
    digitalWrite(VENTILATOR_PIN, LOW);  
}
```

Schleifen

Calliope “x-Mal wiederholen”



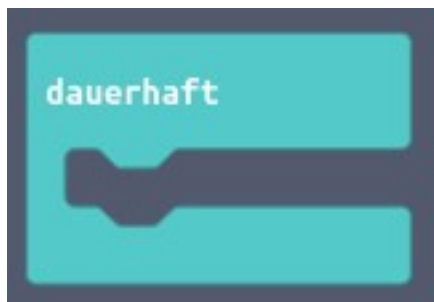
C/C++ “x-Mal wiederholen”

```
for (int i = 0; i < 5; i++) {  
    // Anweisungen  
}
```

Beispiel

```
for (int i = 0; i < 5; i++) {  
    digitalWrite(LED_PIN, HIGH);  
    delay(100);  
    digitalWrite(LED_PIN, LOW);  
    delay(100);  
}
```

Calliope “dauerhaft”



C/C++ “dauerhaft”

```
while (true) {  
    // Anweisungen  
}
```

Beispiel

```
while (digitalRead(BUTTON_PIN) == HIGH) {  
    // Führe Aktionen aus, solange der Knopf gedrückt ist  
}
```

Funktionen

Calliope



C/C++

```
void meineFunktion() {  
    // Anweisungen  
}
```

Beispiel

Funktion mit Rückgabewert und Parametern:

```
int addiere(int a, int b) {  
    return a + b;  
}
```

Operatoren

Operation

Gleich
Ungleich
Größer
Größer gleich
Kleiner
Kleiner gleich

C/C++

==
!=
>
>=
<
<=

Beispiel

```
if (a == b) { ... }  
if (a != b) { ... }  
if (a > b) { ... }  
if (a >= b) { ... }  
if (a < b) { ... }  
if (a <= b) { ... }
```

Operation	C/C++	Beispiel
Und	&&	if (a && b) { ... }
Oder		if (a b) { ... }
Nicht	!	if (!a) { ... }

Wichtige Arduino-Funktionen

- `delay(ms)`: Pausiert das Programm für die angegebene Zeit in Millisekunden
`delay(1000); // Wartet 1 Sekunde`
- `random(max)` oder `random(min, max)`: Erzeugt eine Zufallszahl
`int zufallszahl = random(1, 101); // Zufallszahl zwischen 1 und 100`

Arrays und Strings

Arrays

```
int zahlen[5] = {1, 2, 3, 4, 5};
int ersteZahl = zahlen[0]; // Zugriff auf das erste Element (Index 0)
```

Strings

```
String name = "Alice";
int laenge = name.length(); // Länge des Strings
```

Tipps für C/C++ Programmierung

- Jede Anweisung endet mit einem Semikolon: `;`
- Blöcke werden mit geschweiften Klammern `{ }` umschlossen
- Kommentare
 - Einzeilig: `// Kommentar`
 - Mehrzeilig: `/* mehrzeiliges
Kommentar */`
- Variablen müssen vor der Verwendung deklariert werden
- Groß- und Kleinschreibung ist wichtig (case-sensitive)
- Verwende aussagekräftige Variablen- und Funktionsnamen
- Einrückung verbessert die Lesbarkeit (wird vom Compiler ignoriert)
- Nutze Konstanten für unveränderliche Werte: `const int LED_PIN = 13;`
- Debuggen mit `Serial.print()`:

```
Serial.print("Debug: ");
Serial.println(variable);
```

- Bei Fehlern im Code zeigt die Arduino IDE Fehlermeldungen an die, mit ein wenig Erfahrung, durchaus hilfreich sind. Leider sind sie auf Englisch.

```
36 delay(1000) // wait for a second
37 }
38
```

Ausgabe

```
/tmp/.arduinoIDE-unsaved2024824-18370-3ty3ie.03qn2/sketch_sep24a/sketch_sep24a.ino: In function 'void loop()':
/tmp/.arduinoIDE-unsaved2024824-18370-3ty3ie.03qn2/sketch_sep24a/sketch_sep24a.ino:36:14: error: expected ';' before '}' token
36 |     delay(1000)
    |     ^
37 | }
    | ~
exit status 1
Compilation error: expected ';' before '}' token
```

Hier wurde das ";" hinter dem delay(1000) vergessen und die Arduino IDE weist darauf hin

- Das Internet ist voll von Arduino Tutorials, sowohl als Video als auch als Text und auch in der Fabmobil-Bibliothek gibt es einige Bücher zum Thema.