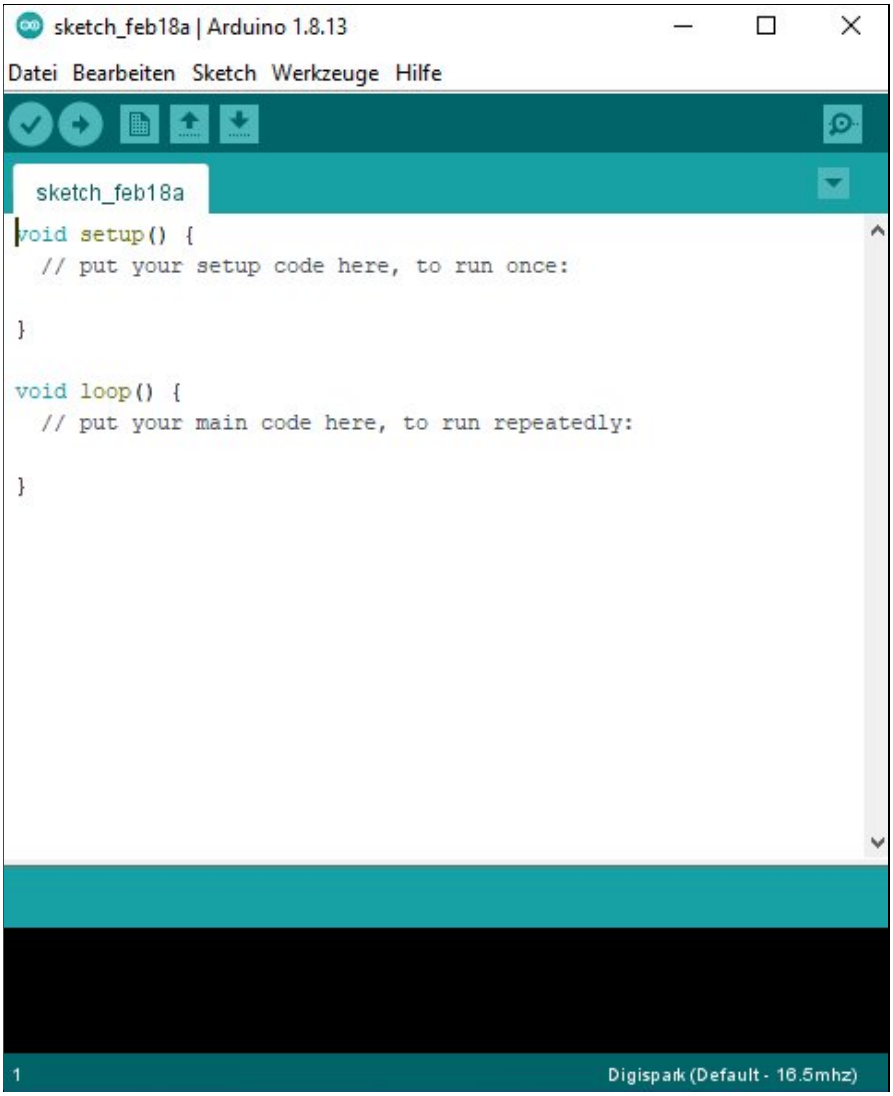
Das erste Programm: "Hallo Welt!"

Programmierung

So – jetzt wird programmiert.
Bitte öffnen Sie die Arduino-Anwendung.

Auf dem Bildschirm wird der "Rumpf" eines neuen Programmes angezeigt.

Ach ja – der Arduino sagt auch nicht "Programm", sondern "Sketch".
Als Namen für den neuen Sketch, den wir erstellen, schlägt er das aktuelle Datum vor (im Beispiel "sketch_feb18a"). Der Name kann aber später beim Speichern geändert werden.



Wichtig:
Bitte überprüfen, ob der "Digispark" als aktuelles Board ausgewählt wurde.
Das aktuelle Board steht unten links in der grünen Zeile.
Gegebenenfalls unter "Werkzeuge - Board - Digistump AVR-Boards" den Digispark auswählen!

Unser Programm (oder wie gesagt eigentlich "Sketch") wird im wesentlichen aus zwei "Blöcken" bestehen, die beide mit dem Schlüsselwort "VOID" anfangen:

VOID setup()
In diesem Block stehen Befehle, die genau einmal beim Start des Contollers ausgeführt werden.

VOID loop()
In diesem Block stehen Befehle, die danach immer und immer wieder ausgeführt werden, solange der Controller angeschaltet ist und mit Strom versorgt wird.

Innerhalb der Blöcke werden zusammenhängende Funktionen durch geschwungene Klammern ("{" und "}") gruppiert, eine Zeile wird meist mit einem Semikolon abgeschlossen und in vielen Fällen kommt es auf die Groß- und Kleinschreibung an! Es gibt also diverse Möglichkeiten, durch falsche Schreibweise, vergessene Semikolons oder falsch gesetzte Klammern Fehler einzubauen!

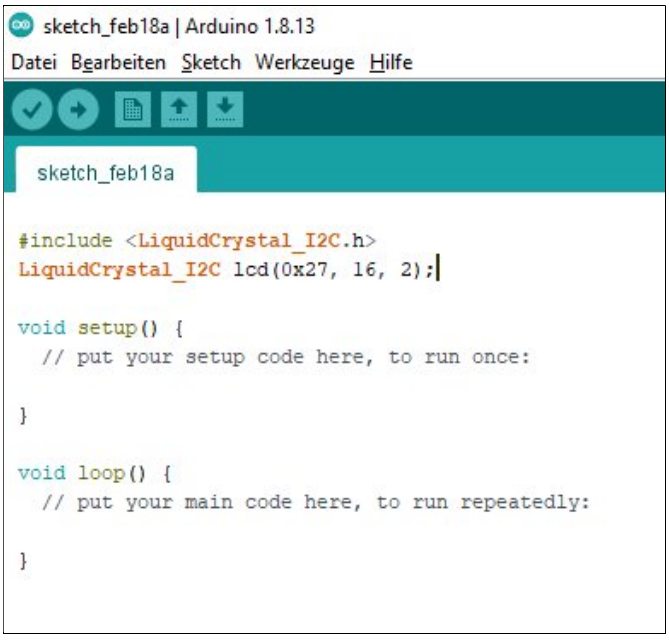
Ganz am Anfang (also noch vor dem "VOID setup"-Block) stehen ein paar Anweisungen, die wir als "vorbereitende Arbeiten" ansehen können.

Schaffen Sie also mal als erstes vor dem Block "VOID setup()" ein bisschen Platz, indem Sie ein paar Leerzeilen einfügen und geben Sie nacheinander folgende Zeilen ein:

```
#include <LiquidCrystal_I2C.h>
```
Hiermit wird sozusagen die "Bedienungsanleitung" des LCD-Displays unserem Sketch hinzugefügt.
Der Digispark "lernt" damit, wie das Display funktioniert.

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```
Hierdurch erfährt der Digispark, dass das Display an der "Adresse" 0x27 zu finden ist*) und dass es sich um ein Display mit zwei Zeilen zu je 16 Zeichen handelt.
*) Der beschriebene Displaytyp ist standardmäßig auf diese Adresse eingestellt

Der Bildschirm sollte jetzt ungefähr so aussehen:



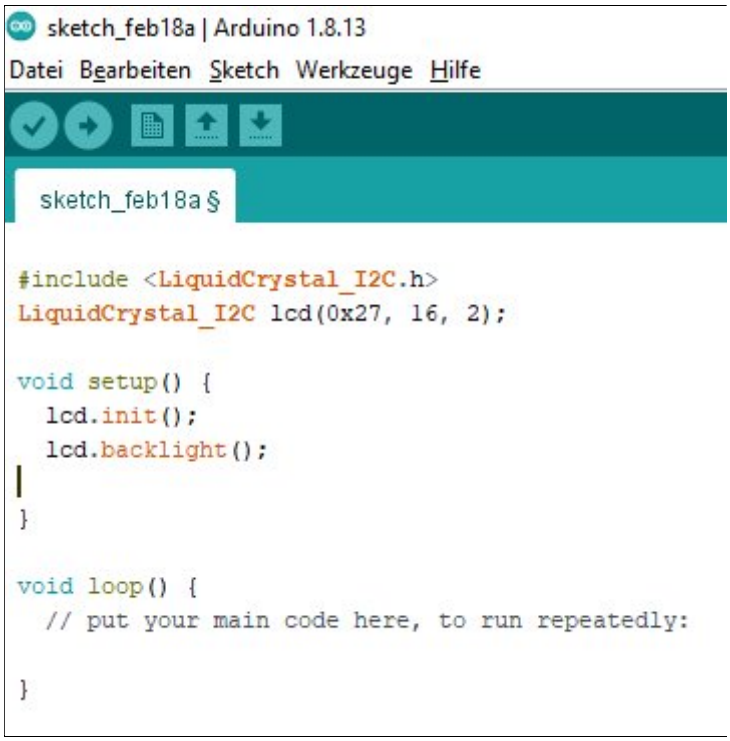
Jetzt kümmern wir uns um den "VOID setup()"-Block.
Das was hier drin steht, wird genau einmal ausgeführt:

lcd.init();
Hierdurch wird das Display "initialisiert", d.h. eingeschaltet und in einen definierten Grundzustand versetzt.

lcd.backlight();
Hierdurch wird die Hintergrundbeleuchtung des Displays eingeschaltet, sodass wir später überhaupt etwas auf dem Display sehen können.

Die Kommentarzeile, die mit "//" anfängt, können wir löschen.

Unser Sketch sollte jetzt so aussehen (das "Einrücken" der Zeilen kann durch die Tastenkombination "STRG-T" automatisiert werden).



Jetzt fehlt noch der "VOID loop()"-Block.
Alles was da drin steht, wird immer und immer wieder ausgeführt.

Fangen wir an:

lcd.setCursor(5, 0);
Damit setzen wir die Startposition, an der als nächstes etwas auf dem Display ausgegeben wird.
Hier: 6. Position in der ersten Zeile.
Irritiert?
Er fängt beim Zählen mit der "0" an, deshalb ist "5" die sechste Stelle und "0" die erste Zeile!

lcd.print("Hallo");
Beginnend an der gerade gesetzten Position soll das Wort "Hallo" geschrieben werden.

lcd.setCursor(5, 1);
Position für die nächste Textausgabe: Position 6 in der zweiten Zeile.

lcd.print("Welt!");
Dort den Text "Welt!" ausgeben.

delay(3000);
3000 Millisekunden (also 3 Sekunden) warten und garnichts tun (damit wir Zeit haben, den Text am Display zu lesen.

lcd.clear();
Dann das Display löschen. Alle Texte verschwinden.

delay(3000);
Wieder 3 Sekunden warten. Wir sehen in dieser Zeit ein leeres Display.

Zusammengefasst:
In der Mitte der ersten und zweiten Zeile wird der Text "Hallo Welt!" ausgegeben und ist drei Sekunden sichtbar. Dann wird das Display gelöscht und wieder drei Sekunden gewartet.
Und dann geht dasselbe wieder von vorne los, und zwar immer und immer wieder.
Wir sehen also den im 3-Sekunden-Takt "blinkenden" Text "Hallo Welt!"

Unser Sketch ist jetzt schon fertig und sieht so aus:
(Ich habe noch ein paar erläuternde Kommentare dazu geschrieben)

```
#include <LiquidCrystal_I2C.h>          // Bibliothek zur Ansteuerung des LCD-Displays
LiquidCrystal_I2C lcd(0x27, 16, 2);     // Modus LCD-Display setzen auf 2 Zeilen zu 16 Spalten

void setup() {
  lcd.init();                           // Initialisierung LCD-Display
  lcd.backlight();                      // Hintergrundbeleuchtung am LCD-Display einschalten
}

void loop() {
  lcd.setCursor(5, 0);                  // Schreibposition: 1. Zeile, 6. Spalte
  lcd.print("Hallo");                   // Den Text "Hallo" ausgeben
  lcd.setCursor(5, 1);                  // Schreibposition: 2. Zeile, 6.Spalte
  lcd.print("Welt!");                   // Den Text "Welt" ausgeben
  delay(3000);                          // 3 Sekunden warten, damit man den Text lesen kann...
  lcd.clear();                          // Display löschen...
  delay(3000);                          // 3 Sekunden lang ein leeres Display anzeigen...
}                                       // bevor es wieder von vorne logeht...
```

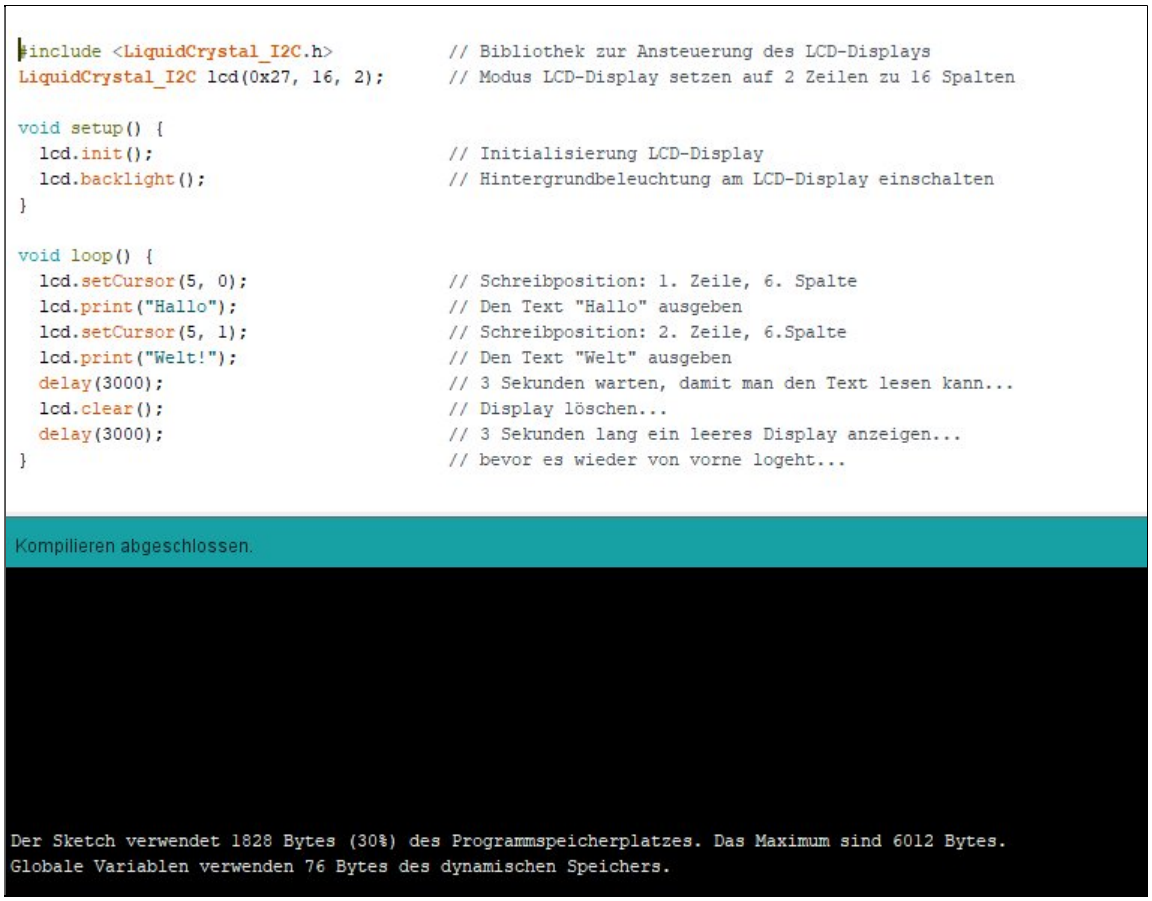
Als nächstes überprüfen wir das Programm auf Fehler.
Hierzu wird der Knopf mit dem Häkchen in der oberen, grünen Bildschirmzeile gedrückt.
Der Sketch wird in einen binären Code übersetzt (kompiliert), den der Digispark verstehen und ausführen kann. Findet er beim Kompilieren Fehler, so werden diese angezeigt. Hätten wir zum Beispiel in einer Zeile das Semikolon am Ende vergessen, würde diese Fehlermeldung ausgegeben:

```
void loop() {
  lcd.clear();                    // Display löschen
  lcd.setCursor(5, 0);            // Schreibposition: 5 nach rechts, 0 nach unten
  lcd.print("Hallo")               // Den Text "Hallo" ausgeben
  lcd.setCursor(5, 1);            // Schreibposition: 5 nach rechts, 1 nach unten
  lcd.print("Welt!");              // Den Text "Welt" ausgeben
  delay(3000);                    // 3 Sekunden warten
}
```

```
expected ';' before 'lcd'
```

```
D:\Daten\PROJEKTE\DigiSpark\05a Fertige Programme LCD\01 - Hello World - Kopie\Code\Code.ino: In function 'void loop()':
Code:26:3: error: expected ';' before 'lcd'
   lcd.setCursor(5, 1);                 // Schreibposition: 5 nach rechts, 1 nach unten
   ^
exit status 1
expected ';' before 'lcd'
```

Wenn keine Fehler gefunden werden und alles (syntaktisch) richtig ist, sieht die Meldung so aus:



```
#include <LiquidCrystal_I2C.h>          // Bibliothek zur Ansteuerung des LCD-Displays
LiquidCrystal_I2C lcd(0x27, 16, 2);     // Modus LCD-Display setzen auf 2 Zeilen zu 16 Spalten

void setup() {
  lcd.init();                           // Initialisierung LCD-Display
  lcd.backlight();                      // Hintergrundbeleuchtung am LCD-Display einschalten
}

void loop() {
  lcd.setCursor(5, 0);                  // Schreibposition: 1. Zeile, 6. Spalte
  lcd.print("Hallo");                   // Den Text "Hallo" ausgeben
  lcd.setCursor(5, 1);                  // Schreibposition: 2. Zeile, 6.Spalte
  lcd.print("Welt!");                   // Den Text "Welt" ausgeben
  delay(3000);                          // 3 Sekunden warten, damit man den Text lesen kann...
  lcd.clear();                          // Display löschen...
  delay(3000);                          // 3 Sekunden lang ein leeres Display anzeigen...
}                                       // bevor es wieder von vorne logeht...
```

```
Kompilieren abgeschlossen.

Der Sketch verwendet 1828 Bytes (30%) des Programmspeicherplatzes. Das Maximum sind 6012 Bytes.
Globale Variablen verwenden 76 Bytes des dynamischen Speichers.
```

Jetzt kann das Programm zum Digispark übertragen werden. Das wird mit dem "Pfeil" in der grünen Symbolleiste gemacht, wobei folgende Schritte in genau dieser Reihenfolge ausgeführt werden müssen:

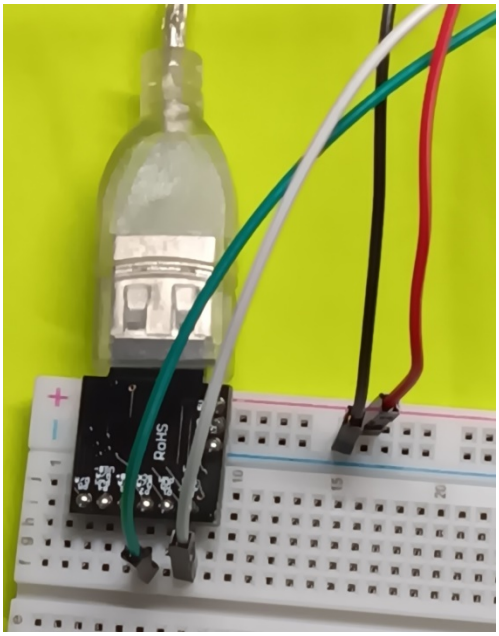
1. Der Digispark darf noch nicht mit dem Computer verbunden sein!
2. Den grünen Pfeil drücken

Der Sketch wird kompiliert und es erscheint - sofern keine Fehler gefunden wurden - diese Meldung:

```
Hochladen...
Der Sketch verwendet 1828 Bytes (30%) des Programmspeicherplatzes. Das Maximum sind 6012 Bytes.
Globale Variablen verwenden 76 Bytes des dynamischen Speichers.
Running Digispark Uploader...
Plug in device now... (will timeout in 60 seconds)
```

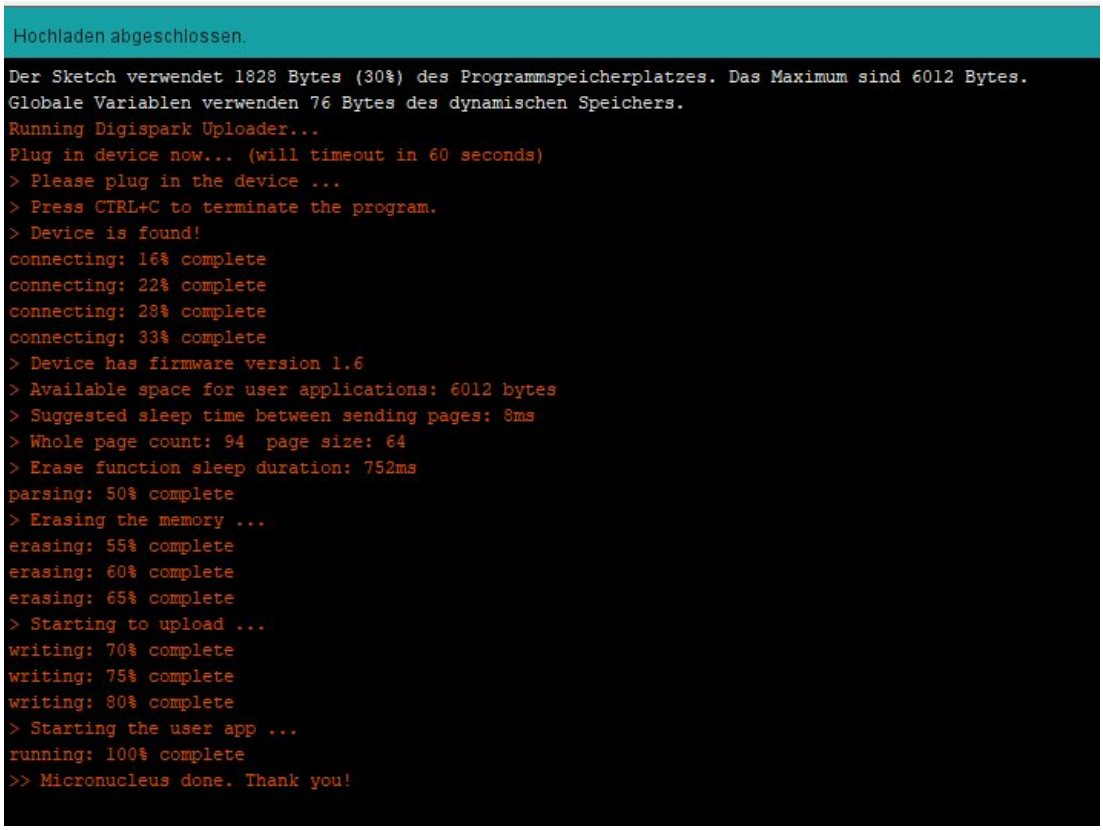
3. Erst jetzt den Digispark mit einer USB-Schnittstelle des Computers verbinden. Wir haben dafür 60 Sekunden Zeit! Benutzen Sie zum Beispiel ein USB-Verlängerungskabel*)

*) Das USB-Kabel muss vollständig verdrahtet sein! Bei USB-Kabeln, die lediglich zur Stromversorgung genutzt werden, sind häufig nur die beiden stromführenden Leitungen vorhanden und die Datenleitungen stehen nicht zur Verfügung. Mit solch einem Kabel würde der Digispark zwar Strom bekommen, es könnten aber keine Daten (also insbesondere nicht unser Sketch) übertragen werden!



Jetzt wird das Programm zum Digispark übertragen.

Wenn das fehlerfrei funktioniert hat, erscheinen folgende Meldungen am Bildschirm:

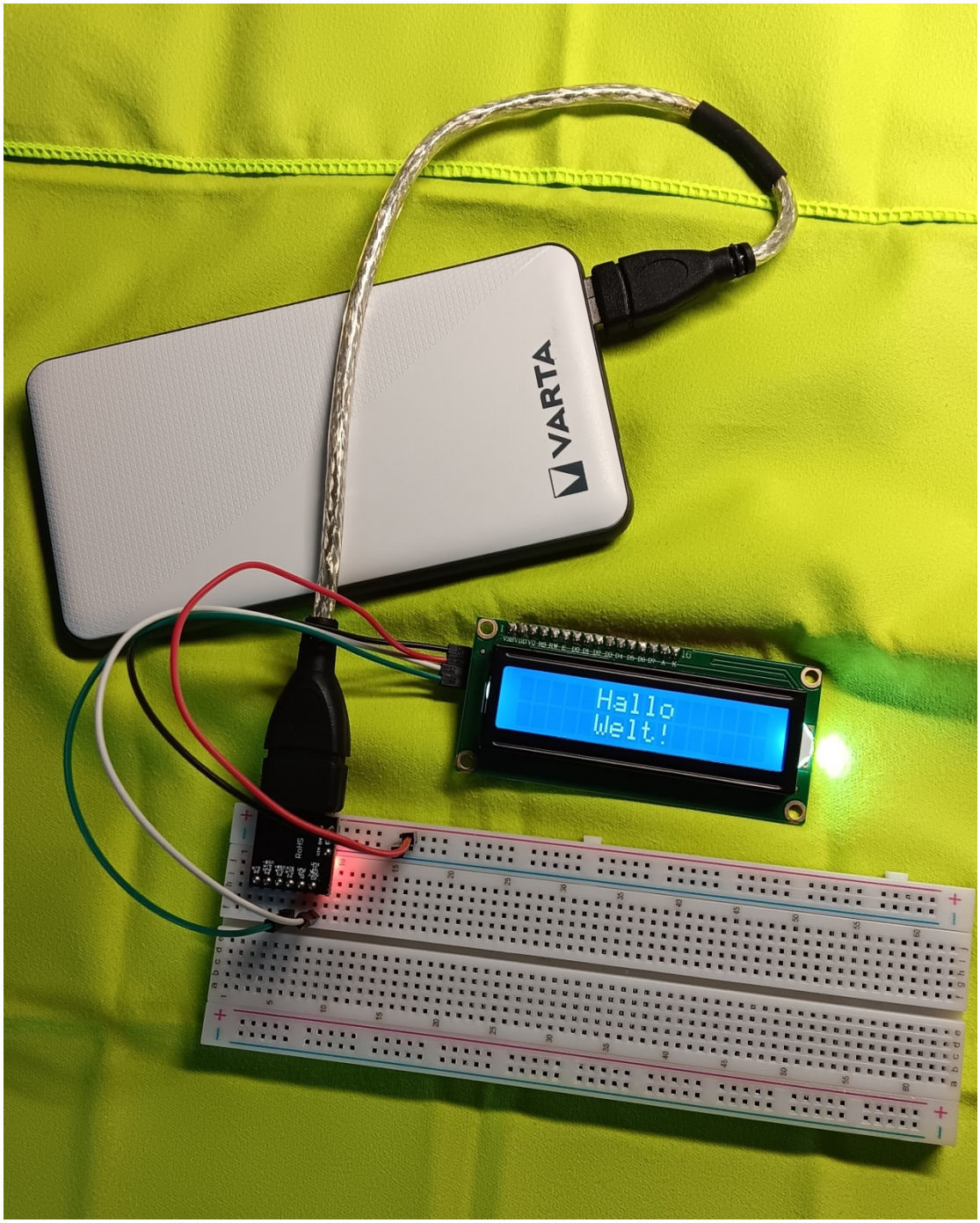


Wichtig sind die letzten zwei Zeilen:

running: 100% complete
>> Micronucleus done. Thank you

Nach der Übertragung kann der Digispark vom Computer getrennt werden und ist nun einsatzbereit.

Wenn Sie ihn jetzt also zum Beispiel mit einer Powerbank verbinden, wird er automatisch gestartet und unser Sketch wird ausgeführt.



Alternative: Anschluss einer Batterie oder eines Akku: