

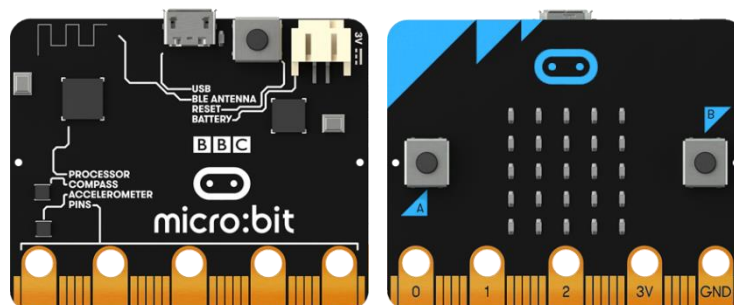
Grundlagen zur Programmierung des micro:bit in Python I

Was ist micro:bit ?

Der **micro:bit** ist ein **Computer** (genauer ein Einplatinencomputer) und sieht aus wie ein großer Chip. Er soll einen **intuitiven und spielerischen Einstieg in die Programmierung** ermöglichen. Er hat einige **Sensoren** (Temperatursensor, Sensoren zur Messung von Bewegung uvm.) und auf der Vorderseite befinden sich **25 LEDs**, die man einzeln ansprechen und rot leuchten lassen kann. Zudem hat er links und rechts (auf der Vorderseite) jeweils eine **Taste, mit A bzw. B bezeichnet**, welche man selbst programmieren kann (was bei Knopfdruck passieren soll – z.B. einen Smiley anzuzeigen). Man kann also unterschiedliche spannende Dinge mit ihm umsetzen!

Um dem micro:bit zu sagen, was er tun soll, programmiert man Code am Computer, steckt den micro:bit an und kopiert das erstellte Programm darauf (wie auf einen USB-Stick). Um den Code am Computer zu erstellen, kann man sich entweder ein Programm herunterladen oder man verwendet den praktischen Webeditor unter: <https://makecode.microbit.org/>! Dort kann man den micro:bit mit Ziehen/Schreiben und Aneinanderreihen von Programmierbefehlen programmieren – entweder mit Blöcken, oder durch textbasierte Programmiersprachen, wie Python oder JavaScript. Man kann ihn aber auch weiteren **unterschiedlichen Programmiersprachen** programmieren, näheres dazu kann man z.B. unter <http://www.microbit.at/> nachlesen.

Der **Vorteil einer textbasierten Programmiersprache** ist, dass man später auch komplexere Programmierprojekte starten kann. Wenn man vorhat, einen Beruf in der Technik oder ein technisches Studium zu ergreifen, erleichtert es den Einstieg, schon davor mit textbasierten Programmiersprachen zu arbeiten. Auch kann man mit textbasierten Programmiersprachen lernen, wie man einen Algorithmus strukturell aufbaut, man lernt, genau zu sein. Man lernt die Fehler nicht nur durch reines Probieren, sondern durch geschicktes Suchen zu finden und zu verbessern. Außerdem tastest du dich an textbasiertes Programmieren ran. Bedenke, dass jede Programmiersprache anders funktioniert, doch kann man einiges von Micro:bit in Python lernen. Es lohnt sich!



Bildquelle: <https://microbit.org/de/guide/>

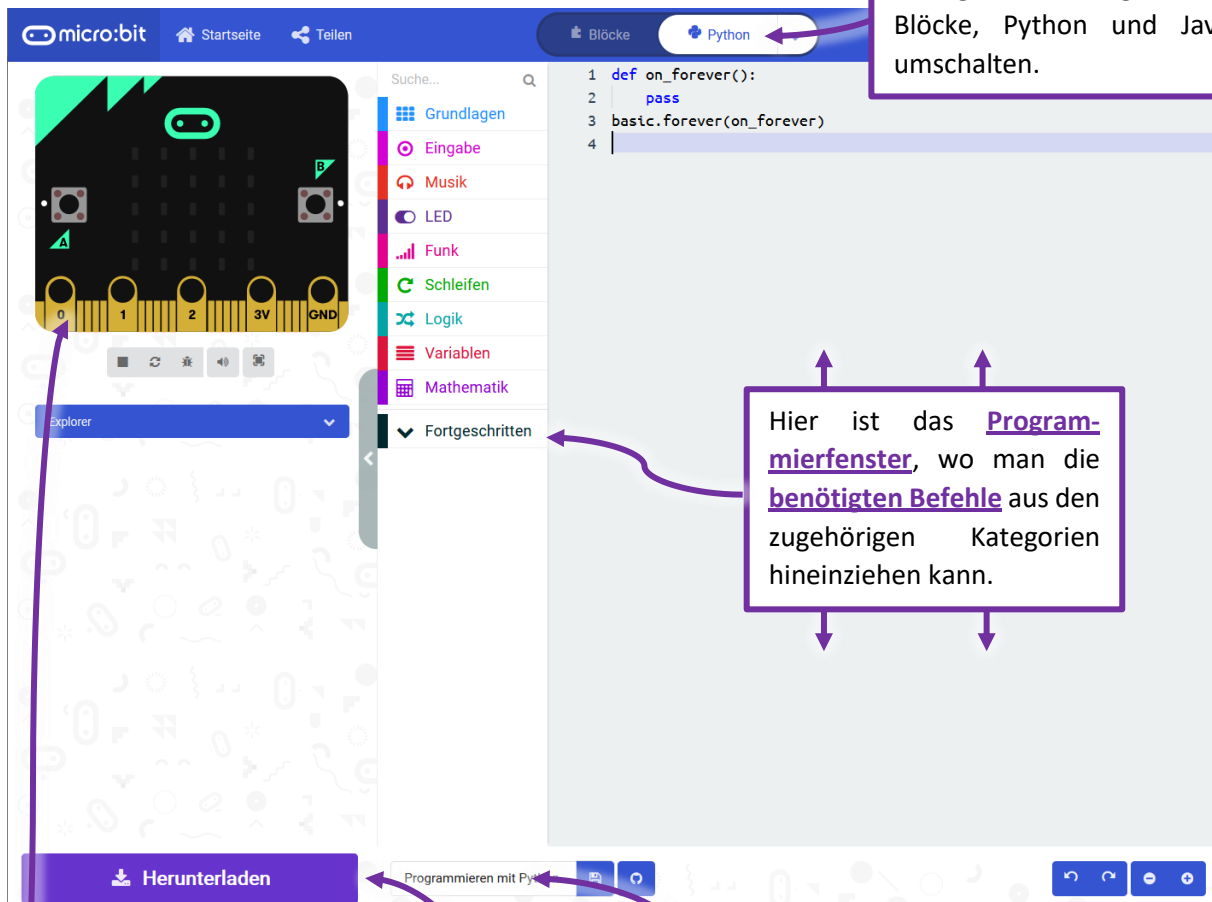
Weitere Informationen zum Arbeiten mit dem micro:bit, findet man z.B. bei folgenden Webseiten:

https://microbit.eeducation.at/wiki/Arbeiten_mit_dem_BBC_micro:bit

<https://microbit.org/de/guide/>

Nach Erlernen der Grundlagen kann man dann z.B. hier vorbeischauchen: <https://microbit.eeducation.at/> oder sich Erweiterungen kaufen, wie den Bit:Bot (<https://shop.pimoroni.de/products/bit-bot>) o. Ä.

Die Oberfläche



Hier kannst du zwischen den drei verfügbaren Programmierarten Blöcke, Python und JavaScript umschalten.

Hier ist das Programmierfenster, wo man die benötigten Befehle aus den zugehörigen Kategorien hineinziehen kann.

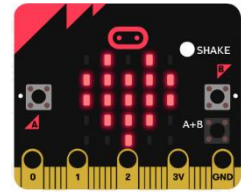
Hier gibt man den gewünschten Programmnamen ein, um das Programm später ggf. wieder zu finden.

Ganz links sieht man eine Vorschau, was das bisher erstellte Programm tut. Mit den Buttons darunter kann diese Vorschau neu gestartet werden, sowie das Tempo der Ausführung eingestellt oder der Vollbildmodus eingeschaltet werden.

Drücke auf diesen Button, um das Programm herunterzuladen. Es wird ein Fenster angezeigt, wo man auf speichern klickt – das zweite sich öffnende Fenster kann geschlossen werden. Das Programm sollte sich nun im Downloadordner am Computer befinden.

Wie funktioniert die Programmiersprache Python?

Python ist eine textbasierte Programmiersprache. Das bedeutet, dass jeder Befehl als Text hingeschrieben werden muss. Die Programmierumgebung für den micro:bit hilft dir jedoch dabei. Es werden dir die Befehle vorgeschlagen, wenn du anfängst zu schreiben. Weiters findest du die verschiedenen Befehle und Strukturen, wie bei der blockbasierten Variante, auf der Seite in den einzelnen Kategorien. Diese kannst du einfach hineinschieben. Wenn du einmal trotz allem nicht mehr weiterkommst, keine Sorge! Je länger du dich damit beschäftigst, desto einfacher wird es!



Wichtig ist: Damit Python-Code ausgeführt werden kann, benötigt es eine spezielle Struktur.

Python-Code hat fast keine Klammern, die genau festlegen, welcher Code wohin gehört. Hier werden **Einrückungen** verwendet. Dazu wird in der vorherigen Zeile mit einem „:“ markiert, dass die nächsten Zeilen, zum (mit dem Doppelpunkt) markierten Teil dazugehören.

Blockbasiert



„dauerhaft“ ist eine **Methode**. Ein kleines **Unterprogramm**, zu dem die beiden „zeige Symbol“-Blöcke gehören.

Python

```
def on_forever():
    basic.show_icon(IconNames.SMALL_HEART)
    basic.show_icon(IconNames.HEART)
basic.forever(on_forever)
```

In Python heißt diese Methode nicht „dauerhaft“, sondern **on_forever()** (kann aber geändert werden). Die beiden **eingerückten Zeilen** entsprechen den „zeige Symbol“-Blöcken, welche zu on_forever() gehören. Das erkennt man an dem „:“

Was ist eine Methode?

Methoden nennt man auch **Prozeduren, Funktionen** oder **Unterprogramme**. Der letzte Begriff sagt am ehesten aus, was eine Methode ist. Ein kleines Programm im Programm. Wenn man einen Code immer zu einem **bestimmten Ereignis** ausführen möchte („Wenn Knopf A gedrückt“, „Wenn geschüttelt“, „Dauerhaft“,...) kann man diese in eine Methode schreiben, die genau dann aufgerufen, also verwendet wird, wenn das Ereignis eintritt. Auch wenn man einen bestimmten Code an verschiedenen Stellen immer wieder verwenden möchte, kann man eine Methode verwenden. Das nennt man im Fachjargon auch „Auslagern“ oder Vermeidung von „duplicated Code“, also von vervielfältigtem Code.

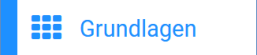
Beim Start

Jener Code, der **beim Start ausgeführt** werden soll, wird einfach **ganz oben** angegeben.

Die wichtigsten Befehle für den Anfang (aus Grundlagen)

Um Code zu schreiben, also den micro:bit zu programmieren, klickt man auf eine der Kategorien (z.B. Grundlagen) und zieht dann den bzw. die benötigten Befehle in das Programmierfenster. Man kann auch selbst die Befehle schreiben, wenn man diese bereits kennt. Ein Befehl ist schon zu Beginn im Programmierfenster:

Befehl	Beschreibung
<p>Code ausführen dauerhaft</p> <pre>def on_forever(): pass basic.forever(on_forever)</pre>	Code, der in diese Methode eingefügt wird (zwischen dem „:“ und „basic“), wird nach dem Start immer wieder ausgeführt. (endlose Wiederholung)
<p>Was bedeutet <code>def on_forever()</code>?</p>	<code>def</code> deutet an, dass hier eine Methode geschrieben wird, also ein kleines Unterprogramm, das nur dann ausgeführt wird, wenn die Methode (<code>on_forever()</code>) aufgerufen, also benötigt und verwendet wird.
<p><code>pass</code></p>	Das ist ein Lückenfüller. Wenn in einer Methode nichts drinstehen hat, wird standardmäßig <code>pass</code> (=weitergeben) geschrieben. Wenn man etwas einfügen möchte, wird die Zeile gelöscht.
<p>Was bedeutet <code>basic.forever(on_forever)</code>?</p>	Hier wird die Methode aufgerufen. Bei der Programmierung des micro:bits wird dies nach der Beschreibung der Methode gemacht. <code>basic.forever()</code> bedeutet, dass alles was in der Definition (<code>def</code>) zwischen „:“ und „basic“ steht, ausgeführt wird, in Dauerschleife!

Um ein erstes Programm zu schreiben, bei dem sich auch etwas Sichtbares tut, können wir folgende Blöcke aus der Kategorie  verwenden:

Block	Beschreibung
<p>zeige Zahl <code>value</code></p> <pre>basic.show_number(0)</pre>	zeigt die angegebene Zahl (hier 0) auf dem micro:bit durch Leuchten der LEDs an
<p>zeige LEDs</p> <pre>basic.show_leds(""" # """)</pre>	Das Zeichen ‚#‘ zeigt die Stellen an, an der die LEDs dann am micro:bit leuchten sollen; Bei den Stellen mit dem Zeichen ‚.‘ bleiben die LEDs ausgeschaltet . (es kann ein eigenes Muster erstellt werden)

<p>zeige Symbol <code>icon</code></p> <p>basic.show_icon(IconNames.HEART)</p>	<p>zeigt das ein Symbol (hier das Herz) am micro:bit durch Leuchten der LEDs an; Wenn du ein anderes Symbol auswählen möchtest, schau einfach in der <i>Symbolsammlung</i> nach!</p>
<p>zeige Text <code>Hello!</code></p> <p>basic.show_string("Hello!")</p>	<p>zeigt den eingegebenen Text (ein Wort oder Satz, hier: Hello!) durch Leuchten der LEDs an.</p>

Ein erstes Beispielprogramm, das am micro:bit ‚Hallo‘ anzeigt (sobald dieser Strom bekommt) und dann einen erfreuten Smiley:

```
basic.show_string("Hallo!")
```

```
def on_forever():
```

```
    basic.show_icon(IconNames.HAPPY)
```

```
basic.forever(on_forever)
```

Sind wir mit dem Code fertig, benennen wir unser Programm und laden es herunter (unten links auf den Button ‚Herunterladen‘ oder auf das Speichersymbol neben dem Titel klicken). Nun können wir das Programm auf den angesteckten micro:bit kopieren und ausprobieren.

Hinweis: Den micro:bit verbindet man mit einem USB-Kabel an den Computer. Ist er korrekt eingesteckt, so leuchtet eine orange LED hinten am micro:bit. Wird ein Programm auf diesen kopiert/verschoben, so blinkt diese LED. Hört das Blinken auf und es leuchtet durchgehend **orange**, so ist das Programm erfolgreich kopiert worden und sollte nun am micro:bit ausgeführt werden.



Du kannst dich nun selbst am Programmieren versuchen, schaue dir dazu das zugehörige **Arbeitsblatt** an! Vergiss nicht dein Programm im Anschluss herunterzuladen und auf den micro:bit zu verschieben um es auszuprobieren!