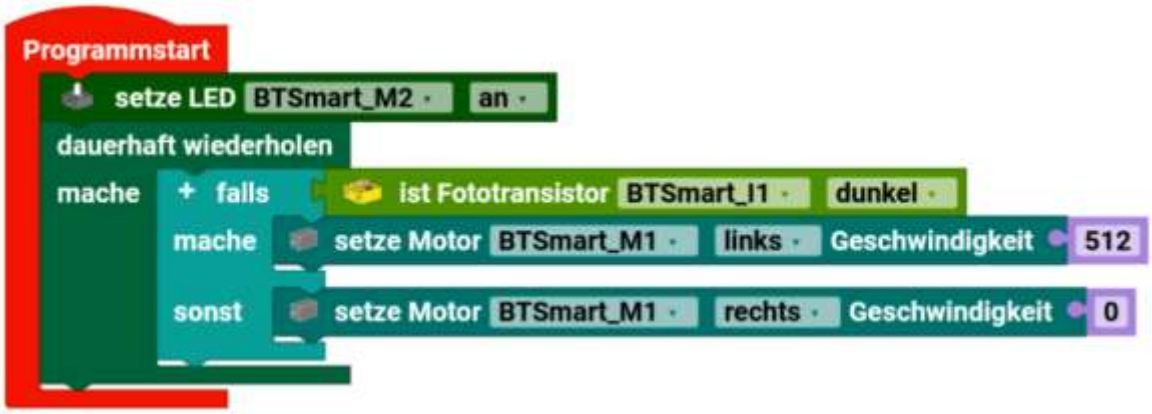
Einstieg in Robo Pro Coding

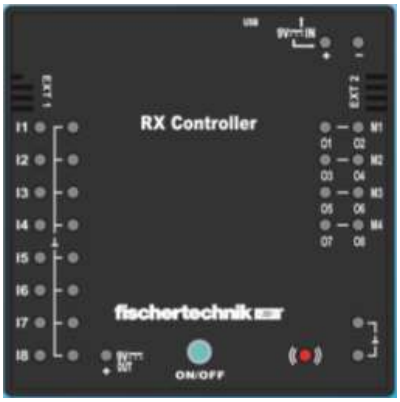
Schritt - für - Schritt

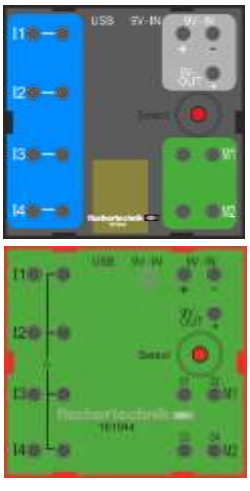
Tutorial / Kompendium für Robo Pro Coding. Von ganz Einfach bis zur künstlichen Intelligenz KI/AI.



Mit ausführlicher Erklärung für die fischertechnik

TXT 4.0 Controller, RX-Controller & BT Smart Controller





+ ftrobopy für Robo Pro und ftSrcratch auf dem TXT 4.0

Von Holger Howey

Version 83

Vorwort

Von ganz Einfach bis zur künstlichen Intelligenz KI/AI.

Schön, dass sie mal reinschauen…

Manchmal scheint es, dass es so viele Möglichkeiten Robo Pro Coding gibt, dass man fast schon die Übersicht verliert.

Dieses Buch soll beim ersten Einstieg von Robo Pro Coding helfen. Auch was man mit den verschiedenen Controllern mit Robo Pro Coding machen kann, soll hier gezeigt werden.

Es ist sehr erstaunlich welche Fähigkeiten Robo Pro Coding hat.
Ob nun einfache Modelle wie einen Motor drehen lassen, bis hin mit Modellen die KI (Künstliche Intelligenz) benutzen, um fehlerhafte Bauteile mit einem Riss zu erkennen und auszusortieren.

-Alle- Befehle/Blöcke habe ich aufgelistet und die meisten mit lauffähigen Beispielen versehen. Alle Informationen, die ich finden konnte, habe ich zu dem Block dazu geschrieben.

Ich möchte mich ganz herzlich bei allen Unterstützern dieses Buchprojektes bedanken.
Insbesondere der Firma fischertechnik, Axel Chobe mit seinen Kurzanleitungen, Horst Günther für das Durcharbeiten des Buches, Torsten Stuehn mit dem Programm „ftrobopy-Server“ und den Infos
und vielen, vielen anderen, für Tipps, Ratschläge und Hinweise. Danke!

Viel Spaß wünscht
Holger Howey

Tipp:

All diese Internetseiten sind kostenlos

Ein großer Freundeskreis der fischertechnik Fans hat hervorragende Seiten zum Thema fischertechnik:

www.ftCommunity.de mit einer Bildersammlung, einem **Forum** für jedermann, Datenbank für Bauteile und Anleitungen älterer Baukästen…

www.fischertechnikclub.nl Der niederländische fischertechnik Fanclub. Gigantische Modelle viel Technik und eine hervorragende Bibliothek…

http://www.chobe.info/34.html …Axel Chobe mit Kurzanleitungen und vielen Infos zu fischertechnik

Und natürlich der Internettauftritt von fischertechnik:

www.fischertechnik.de Viele Informationen, Downloads und zusätzliche Anleitungen zum runterladen…

www.santjohanser.de Einzelteile und Kästen von fischertechnik

Gleiche Programme und anderer Controller

Ich habe für die jeweiligen Controller die Programme geschrieben.
Natürlich kann man die auch für die anderen Controller nehmen.

Zusätzlich gibt es die Möglichkeit andere Bauteile und somit andere Blöcke zu benutzen.
Wer keinen Fototransistor besitzt kann einen Mini-Taster benutzen.

Die Beispiele von fischertechnik aus dem Kasten haben meist im –ersten- Modell, eine genauere Einführung in Robo Pro Coding. Wer die geführten Beispiele in diesem Buch durcharbeitet, hat mit den fischertechnik Beispielen keine Probleme. Sie sind als eine Art Ergänzung zu den aus den fischertechnik Kästen gedacht.

Zusätzlich gibt es, auf der fischertechnik Internetseite, Programmbeispiele und Bauanleitungen von weiteren Kästen.

Bei mir sieht es anders aus!?

Einige Bilder sind mit einer Vorabversion (Betaversion) von Robo Pro Coding entstanden, die mir von fischertechnik zur Verfügung gestellt wurde.

Deswegen kann es sein, dass das endgültige Aussehen etwas anders ist, als hier im Buch dargestellt ist. Die Grundlegenden Funktionen und die Programme sollten dennoch funktionieren.

Hilfe! Ich kriege das nicht hin. Das läuft bei mir nicht…

OK, ich mache auch Fehler. Es kann aber auch was anderes sein. Am einfachsten ist es, einfach im Forum der ftCommunity.de eine Frage zu stellen oder das Problem zu beschreiben. Nicht nur ich bin häufig dort, sondern auch andere kompetente ft-Fans. Ich denke da sollte schnell eine Lösung gefunden werden.
(Momentan) Nehmen sie besser die App-Version von Robo Pro Coding statt der Online-Version.

Richtig im Buch „Suchen“

Tipp: Man kann diese PDF auch durchsuchen. Wenn man die Tasten [Strg]+[f] öffnet sich ein Fenster, wo man den Begriff, den man sucht, eingeben kann. Man kann dann von Fundstelle zu Fundstelle springen.

Die 3 Grundregeln des Programmierens

-Speichern
-Speichern
-Speichern

Speichern sie die Programme.
Speichern sie verschiedene Versionen des Programms.
Speichern sie Sicherheitskopien.

Aufbau im Buch

Zu einem Thema, gibt es erst eine allgemeine Beschreibung, dann für jeden Controller einzeln aufgelistet und mit entsprechenden Bildern/Fenstern für den Controllern. Manche Sachen können auch für mehrere Controller sein.

Die Reihenfolge im Buch:
Thema

- **Allgemein** zum Thema

  - Für den **TXT 4.0** Alle Möglichkeiten, alle Fähigkeiten des Controllers und Robo Pro Coding.

  - Für den **RX** Controller - Die Möglichkeiten die der RX bietet und die die von Robo Pro Coding dazu.

  - Für den **BT Smart** Controller - Die Möglichkeiten die Robo Pro Coding der Controller bietet.

Alle gelben Überschriften sind sehr hilfreich!

Die Reihenfolge der Blöcke im Buch entspricht dem, wie es in Robo Pro Coding ist.

Größtenteils sind die 23 Seiten der Hilfe/Dokumentation von Robo Pro Coding hier integriert worden. Sie ist nun auf dem Stand der Dinge und sehr, sehr stark erweitert worden.

Tipp:
Diese Seite ausdrucken, falten und als Lesezeichen zusammenkleben. Einfach an den Bildschirm oder Ausdruck halten. So kann man die Farbe der Befehle den Reitern zuordnen, unter den sie zu finden sind.

Blau hinterlegte Seitenzahlen sind Querverweise, die man anklicken kann, um zu dieser Seite zu kommen.

Was noch im Buch fehlt:

Einige Beispiele fehlen

Weiterführendes zu Ebenen in Fernbedienung fehlt

Inhaltsverzeichnis
    Reiterfarben im Inhaltsverzeichnis

Map / HTTP nicht ausreichend

Handy App mit dem etwas anderem Aussehen.

Eigene Sounds

Bug Skalierung Charts

Achtung! Einige vorgeschlagene Änderungen oder Zusätze, habe ich eventuell nicht so wie vorgeschlagen übernommen. Entweder sie sind z.B. in den allgemeinen Teil zum Thema eingefügt oder auch (leider) weggelassen worden.

Block“ setze Schalter“ und „ist Schalter“. Unterschied und Zusammenspiel von
    Anzeigenkonfiguration „Aktiviert“ – „Angeklickt“
    Im Block „Aktiviert“ – „Angeklickt“
    „Checked“ und „Enabled“
    Da scheint noch etwas nicht schlüssig zu sein. Bug?

Scheinbar bei Kontrollkästchen (Checkbox) auch. Bug?

Scheinbar bei Schieberegler (Slider) auch. Bug?

Mehr Beispiele zur Anzeige und auch zu den einzelnen Blöcken.

Tipps und Tricks zur Anzeige

Bug im Programm Flussüberquerungsrätsel bisher nicht gefunden. Zu viele Funktionen?

Wie ist deine Meinung?

Zum Buch?

Soll ein „Schnelleinstieg“ gemacht werden?

Soll zu Themen wie z.B. Fernsteuerung, was extra gemacht werden?

Schnellstart... Kurz und knapp

Onlineversion ohne Installation:

Über diesen Link:

https://dev.fischertechnik-cloud.com/de/robopro/

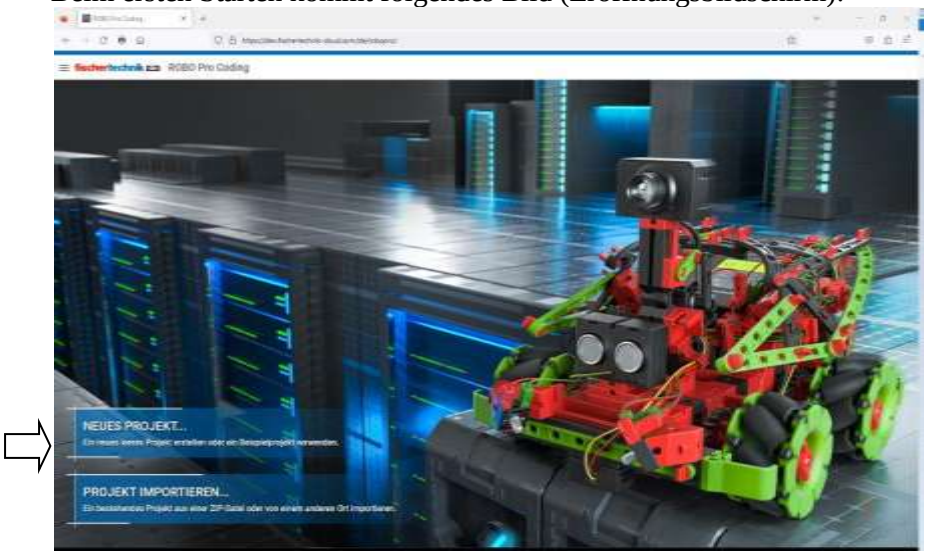
kann man die Onlineversion von Robo Pro Coding ohne Installation aufrufen.
Am aktuellsten ist die Onlineversion von fischertechnik, womit wir hier arbeiten.
Hinweis: Wenn Programme nicht funktionieren sollten, -kann- es sein, dass sie mit der App-Version laufen.

App mit Installation

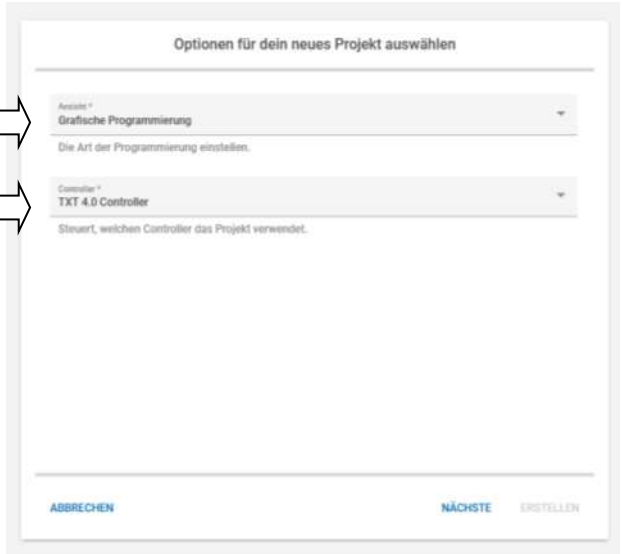
Man kann Robo Pro Coding auch als Programm-App installieren.
Das macht man z.B. für Windows vom MS-Store.

Erstes Mal starten

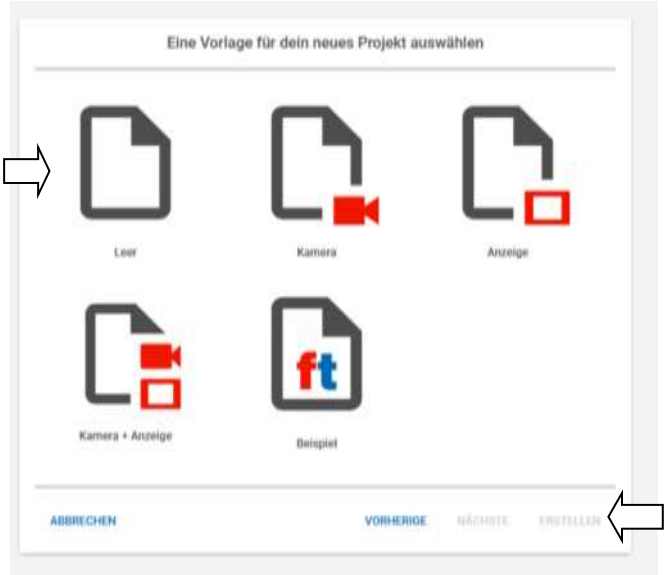
Beim ersten Starten kommt folgendes Bild (Eröffnungsbildschirm):



Hier kann man auswählen, ob man ein neues Projekt starten oder ein vorhandenes laden/importieren möchte. Wenn man auf "Neues Projekt" klickt, kann man die Art der Programmierung, ob Grafische und Python oder nur Python und den Controller auswählen. Wir nehmen die „Grafische Programmierung“.



Hinweis:
Es gibt verschiedene Browser und verschiedene Einstellungen. Es kann sein, dass wenn man den Browser schließt und dann noch mal öffnet, Robo Pro Coding nicht mehr weiß, was eingestellt ist und man bekommt den Eröffnungsbildschirm. Es kann aber auch sein, dass das letzte Projekt direkt geöffnet wird.
Beim Controller nehmen wir den TXT 4.0 .



Die Auswahl ist groß. Möchte man eigene Projekte anlegen, so sind "Leer", "Kamera", "Anzeige" und "Kamera und Anzeige" auszuwählen. Hinter "Beispiele" verbergen sich alle fischertechnik Kästen, Fertigmodelle und alle aktuellen fischertechnik Industriemodelle. Bei den Kästen sind es jeweils alle Programme für die Modelle. Wenn man Modell anderer Kästen ausprobieren möchte, sollte man sich den Kasten selbst oder zumindest die Anleitung kaufen, da man die Verdrahtungspläne braucht. Es ist nicht immer ersichtlich, wozu welcher Motor oder Eingang eingesetzt wird.

Wir nehmen das Projekt „Leer“ und los geht’s mit dem blauen „Erstellen“...

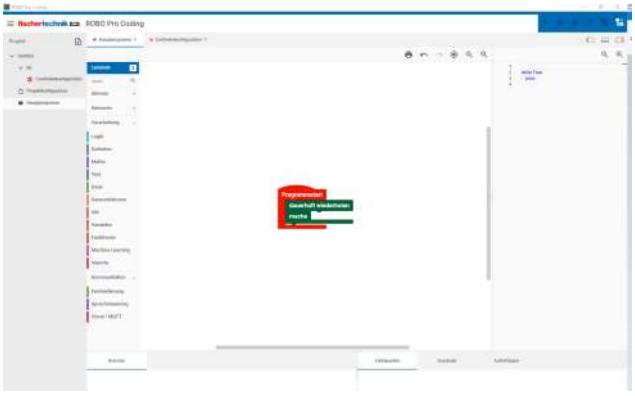
Hinweis: Es gibt eine sehr gute Übersicht über die Robo Pro Coding Programme der fischertechnik Kästen von Axel Chobe unter:

Chobe.info/dokus/ft_modelle.pdf

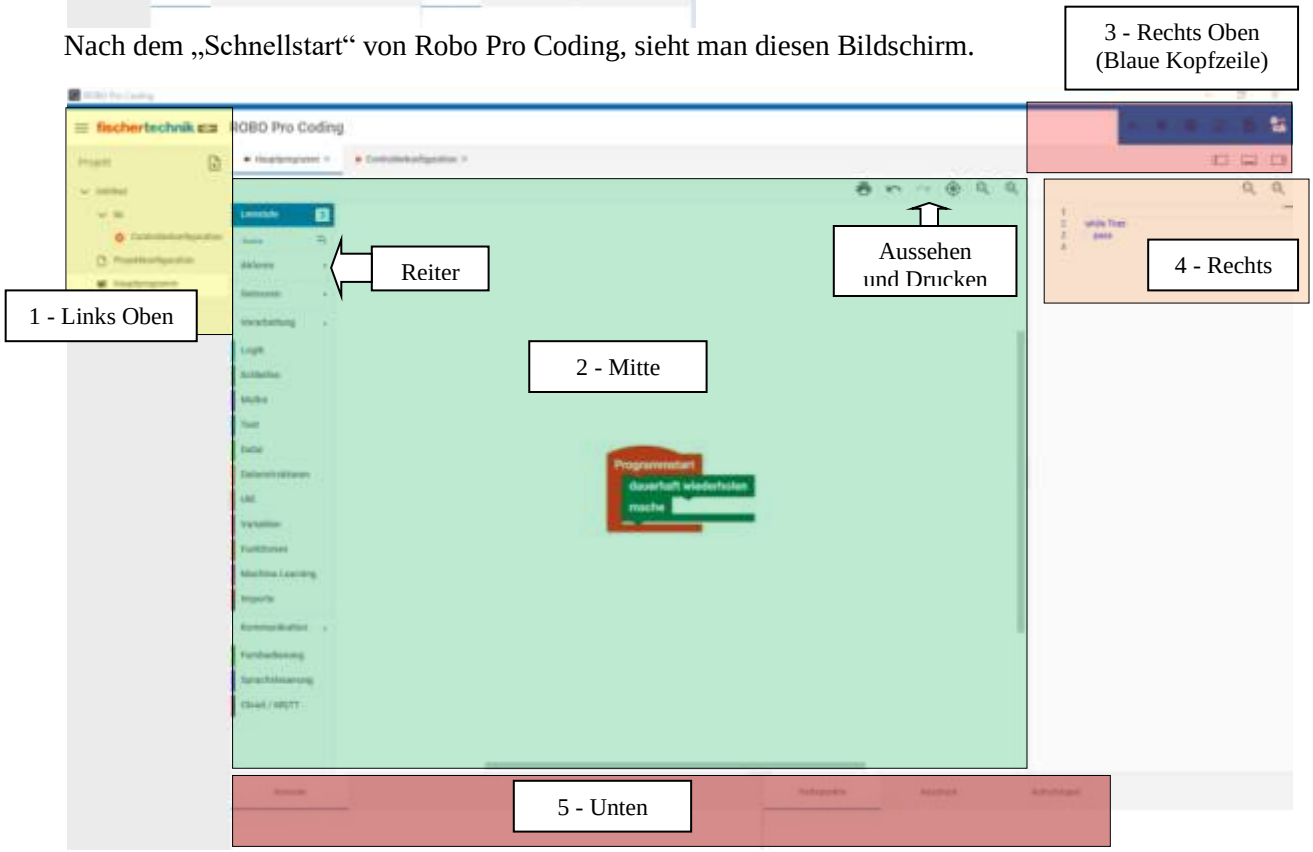
Dort sind alle Beispiele übersetzt und erklärt.

Grundlagen

Die Oberfläche



Nach dem „Schnellstart“ von Robo Pro Coding, sieht man diesen Bildschirm.



Er teilt sich in diese 5 Bereiche auf:

1 - Hier links oben ist die Einstellung und Dateiübersicht von Robo Pro Coding
2 - In der Mitte ist das Programm mit den Reitern an der linken Seite.
3 - Rechts Oben ist die Steuerung vom Programm und dem Controller
4 - Der Bereich rechts, ist der Python-Bereich und Eigenschaften von Bedien-, Anzeige- und Kamerafeldern.
5 - Unten ist der Ausgabebereich vom Programm

Man kann diese Bereiche anders Aufteilen, in dem man den Rand mit der Maus verschiebt. Z.B. ist hier der rechte Bereich (Python) fast nicht sichtbar. Der Python-Bereich (4) wird hier im Buch weniger behandelt. Robo Pro Coding setzt automatisch die Blöcke in Pythoncode um.

Über die Reiter kann man sich die entsprechenden Blöcke anzeigen lassen.
Im unteren Bereich gibt es vier Bereiche:

Konsole: Hier ist die Ausgabe des Programms. Dieses kann über den Textblock „gib aus“.
Haltepunkte: Die werden über den Editor im Pythoncode gesetzt.
Ausdruck: Hier kann man sich Variableninhalte anschauen,
Aufrufstapel: Zeigt die Verschachtelung der Aufrufe.

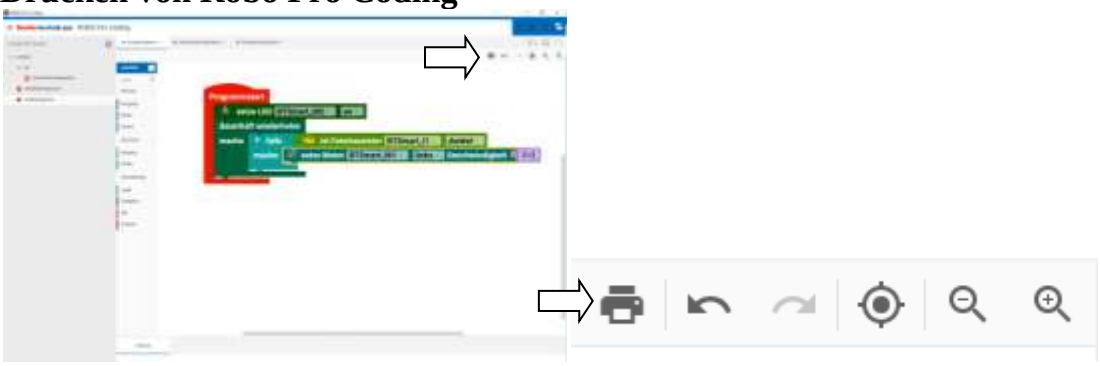
Die Haltepunkte beenden ein Programm. Um überprüfen zu können ob z.B. ein Unterprogramm ein Wert richtig berechnet hat, kann man hier sich den Wert anzeigen lassen. Der Aufrufstapel zeigt an welche Unterprogramme aufgerufen werden.

Aussehen und Drucken von Programmen

Projekt drucken

Projekte und somit Programme, kann man auch in Robo Pro Coding ausdrucken.

Drucken von Robo Pro Coding



Dazu kann man einfach auf das Druckersymbol in der Mitte oben rechts klicken

Es gibt nur die Möglichkeit auf einem DIN-A4-Blatt zu drucken. Dabei entspricht das Blatt, dem mittleren Teil, wo das –gesamte- Hauptprogramm ist. Und das kann ein Problem werden, wenn das Programm groß ist. Entweder muss man das Programm unterteilen, z.B. in verschiedenen Dateien, die man hinzugefügt hat oder man speichert das Projekt erst und löscht alles andere, was man nicht ausdrucken möchte, druckt und lädt dann das Projekt neu. Man kann so mit anderen Teilen weiterverfahren.

Es gibt Druckertreiber, mit denen man Posterdruck machen kann. Somit kann man größer drucken z.B. auf 4 DIN A4 Seiten.

Drucken von Python Programmen.

Direkt aus Robo Pro Coding drucken, kann man die nicht. Ich kopiere den Pythoncode und drucke ihn mit einem anderen Programm. Z.B. Programmers Notepad [Win AVR] und stelle dort Python ein. So wird der Programmcode sofort richtig formatiert und sogar in Farbe gedruckt.

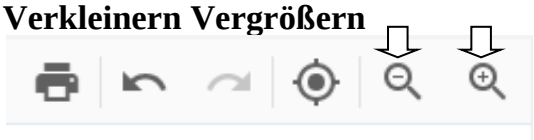
Änderungen Schritt Vor und Zurück

Über diese Symbole kann man Änderungen zurücknehmen oder diese weiterbenutzen.

Blöcke Zentrieren

Über dieses Symbol werden die Blöcke des Programms neu angeordnet. Z.B., wenn sie sich überlagern, kann dies sehr hilfreich sein.

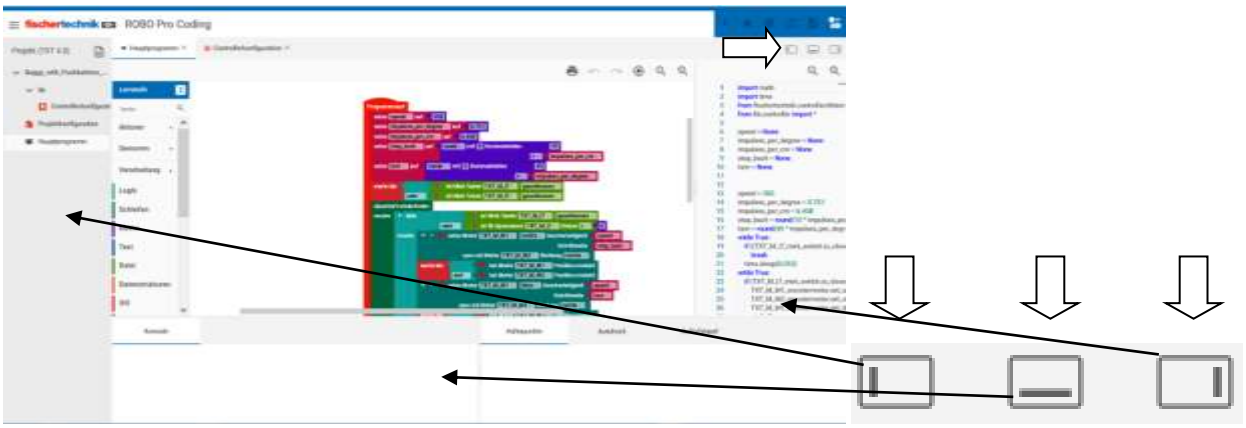
Verkleinern Vergrößern



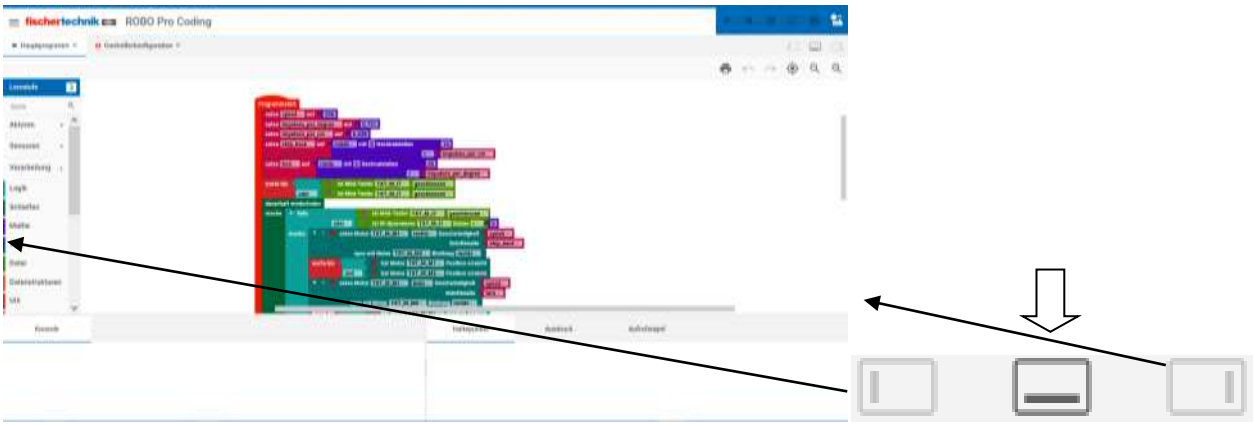
Über diese Symbole kann man die Ansicht aller Blöcke im Programm verkleinern oder vergrößern.

Ein- und Ausblenden von Fenstern

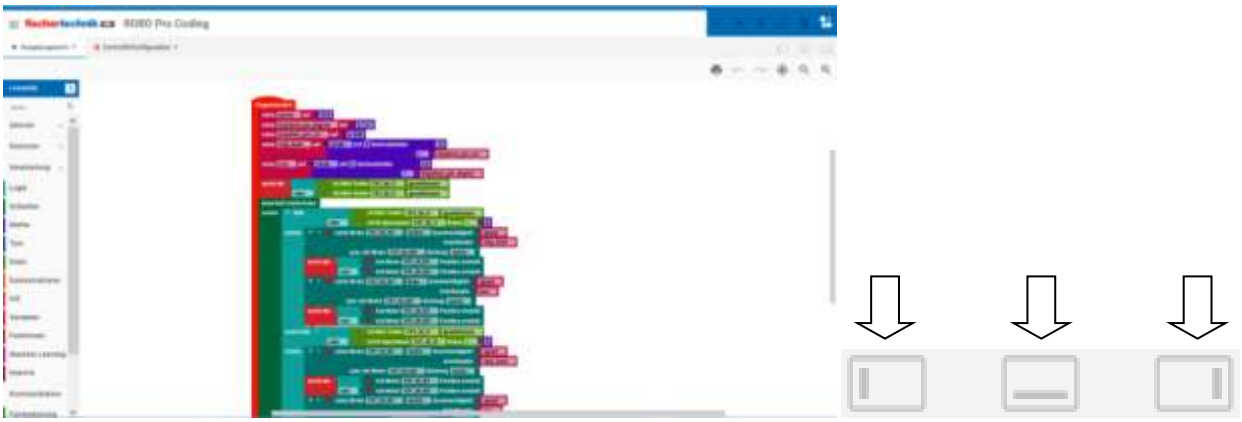
Man kann die „Fenster“ Links, Rechts und Unten aus- und einblenden. Z.B. um sich einen besseren Überblick zu verschaffen.



Normalansicht mit allen eingeblendeten Fenstern.



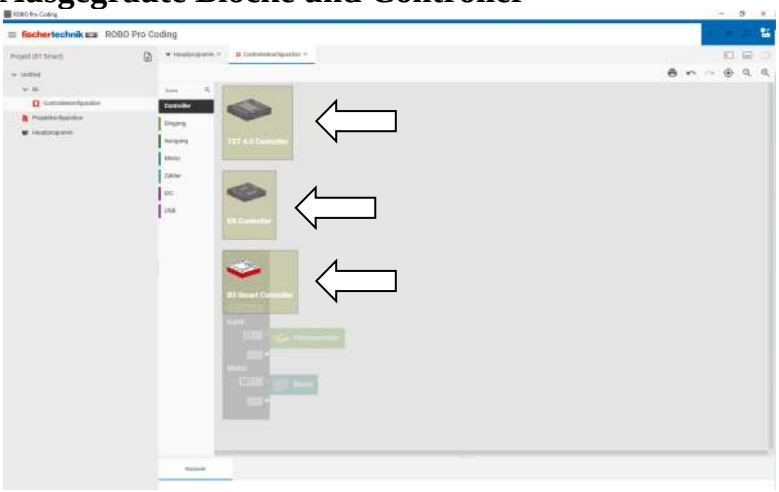
Ansicht nur mit dem unteren eingeblendeten Fenstern.



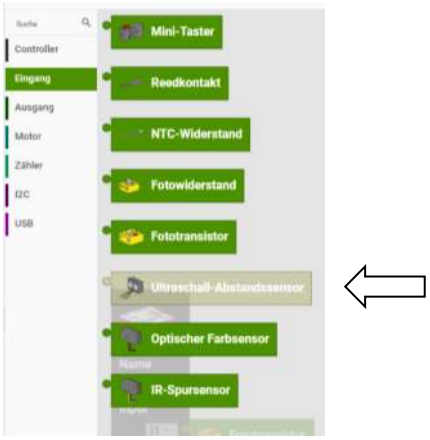
Große-Ansicht mit allen ausgeblendeten Fenstern.

Grundsätzliches bei Robo Pro Coding

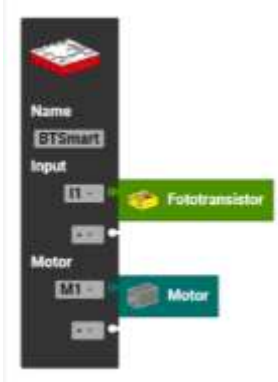
Ausgegraute Blöcke und Controller



Wenn ein Controller ausgewählt wurde, hier der BT Controller, kann kein weiterer Controller ausgewählt werden. Sie sind "ausgegraut".



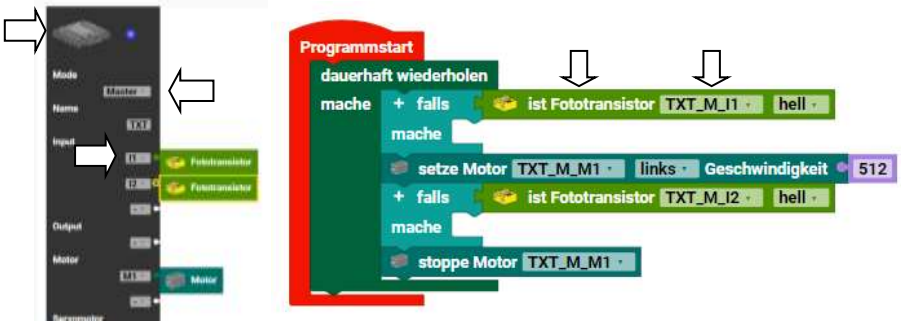
Man kann "ausgegraute" Sensoren nicht an den jeweiligen Controller anschließen, wenn diese dafür nicht geeignet sind. Hier beim BT Smart Controller ist das der Ultraschallsensor.



Im Gegensatz zu TXT 4.0 Controller fehlen dem BT Smart Controller Kamera, Counter… die auch nicht angezeigt werden.

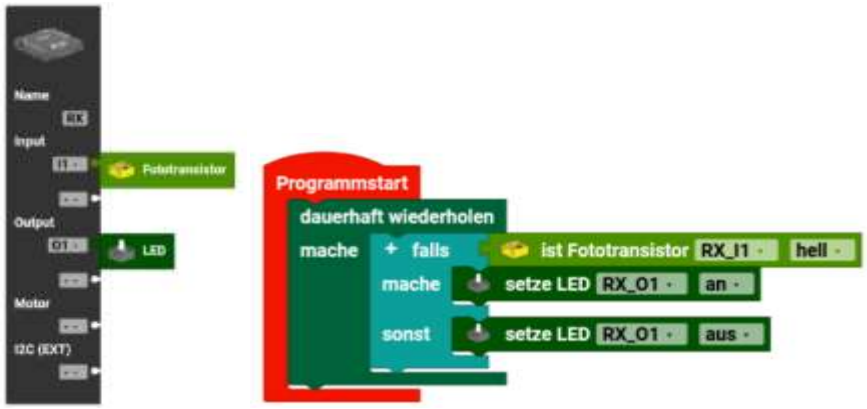
Bezeichnung der Ein- und Ausgänge

Die Bezeichnung der Ein- und Ausgänge ist vom Controller abhängig. Hier der **TXT 4.0 Controller:**



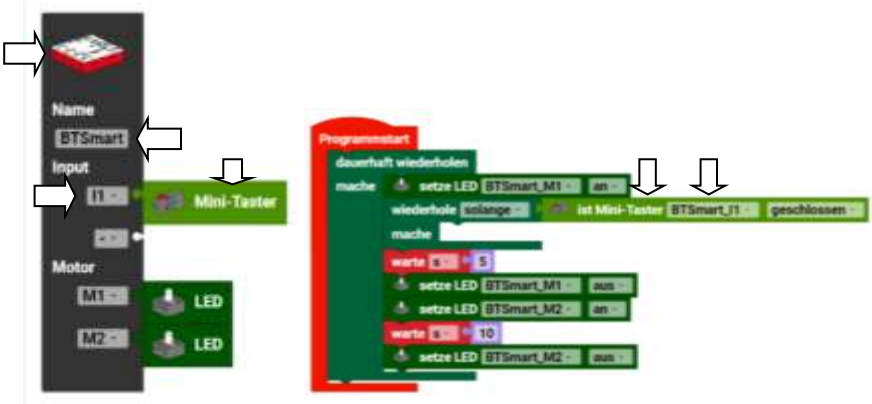
Hier hat der Fototransistor die Bezeichnung TXT_M_I1. Es ist also der **TXT** 4.0 Controller, der **M**aster ist und wo der Fototransistor am Eingang **I1** angeschlossen ist. Wenn der TXT 4.0 Controller die Extension 2 wäre, so hätte er die Bezeichnung TXT_E2_I2. Beim Motor ist es TXT_M_M1. Also **TXT** 4.0 Controller, der **M**aster ist und wo der Motor an **M1** angeschlossen ist.

Die Bezeichnung für den **RX Controller** ist entsprechend.



Die Bezeichnung vom Fototransistor ist RX_I1 und die LED an RX_O1. Somit ist der Fototransistor am RX-Controller und dort am I1 und die LED am Ausgang O1. Momentan kann nur ein RX Controller angeschlossen werden.

Die Bezeichnungen beim **BT Smart Controller**.

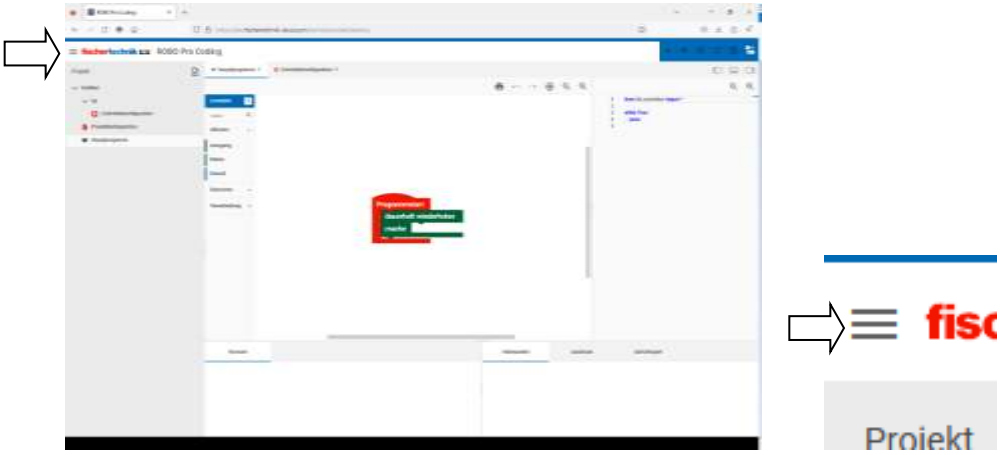


In diesem Beispiel ist ein Taster am BT Smart Controller angeschlossen. Die Bezeichnung ist BTSmart_I1. Es ist also ein Taster, der an einen **BT Smart** Controller an **I1** angeschlossen ist. Der BT Smart Controller hat keine Nummer, da immer nur einer angeschlossen werden kann.

Einstellung und Dateiübersicht von Robo Pro Coding

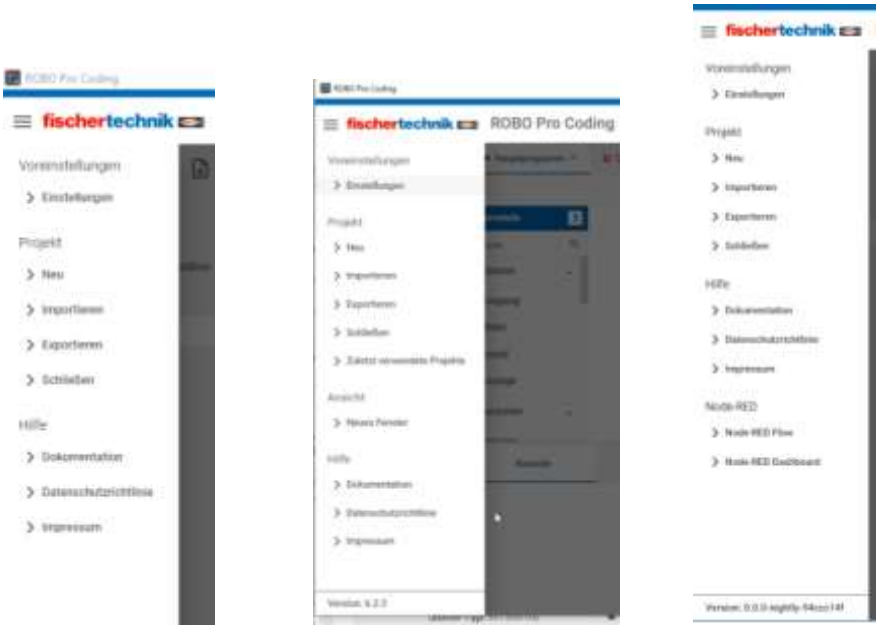
Das Burgermenü.

Hinter den drei Strichen links oben, verbirgt sich ein Menü.
Die drei Striche erinnern an einen Burger. Daher kommt auch der Name: Burgermenü.



Das ist das **Burgermenü,** in einem leeren Projekt.

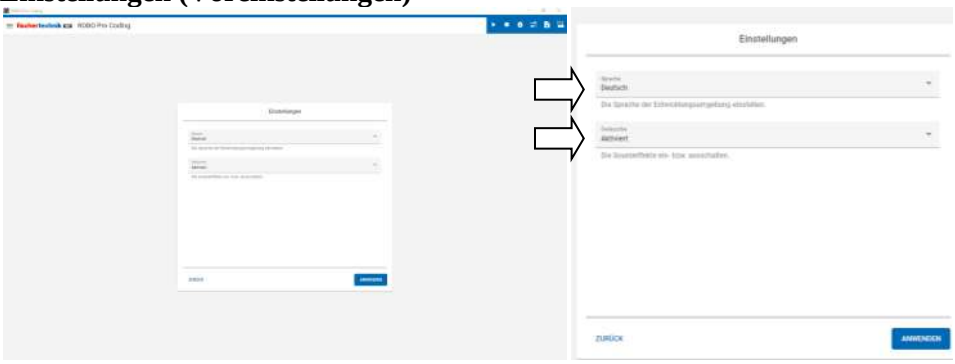
Über das Burgermenü kommt man in die Voreinstellung/Einstellungen von Robo Pro Coding.



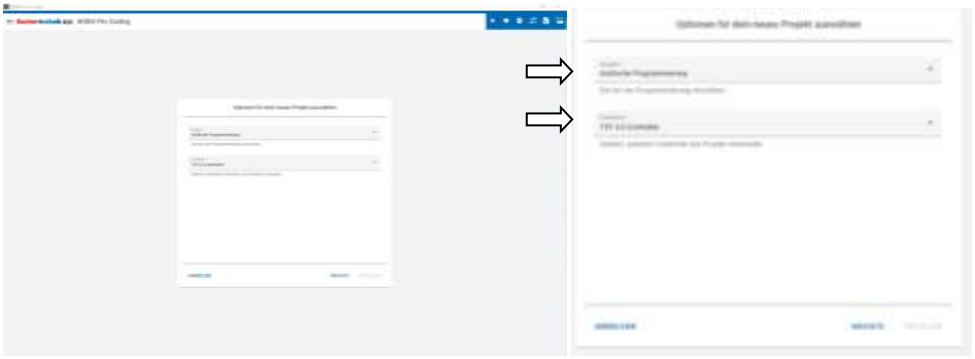
Ja nach Version, gibt es unterschiedliche Menüs.

Beim Klicken auf das Burgermenü klappt sich ein Verzeichnis auf.

Einstellungen (Voreinstellungen)

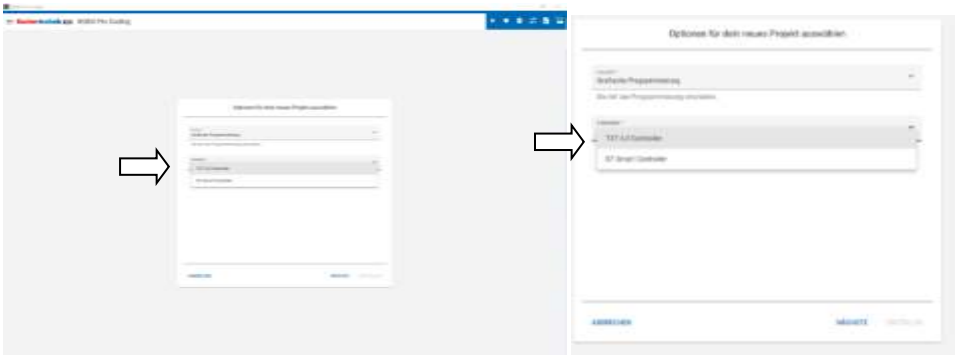


Unter Einstellungen, kann man die Sprache einstellen und Geräusche aktivieren oder deaktivieren. Die Geräusche sind vom PC z.B. beim Anlegen eines Blocks... nicht der Sound vom Controller.
Wenn man die Einstellungen vorgenommen hat, kann man auf Zurück oder Anwenden drücken und man kommt wieder in die normale Ansicht von Robo Pro Coding.

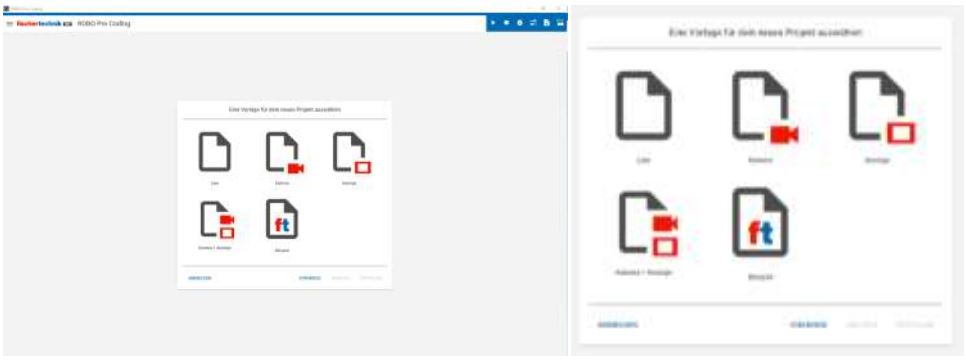


Über Projekt/Neu kann man ein neues Projekt anfangen.
Hier kann man die Ansicht in Grafische-Programmierung oder Python-Programmierung auswählen. In diesem Buch wird die Grafische-Programmierung beschrieben. Manche sagen auch Blocky dazu, da Robo Pro Coding von Blocky gemacht wurde.

Tipp: Bevor man ein neues Projekt anlegt, sollte man das vorhandene speichern, da mit dem Neustart ansonsten das vorherige Projekt im Speicher gelöscht wird.



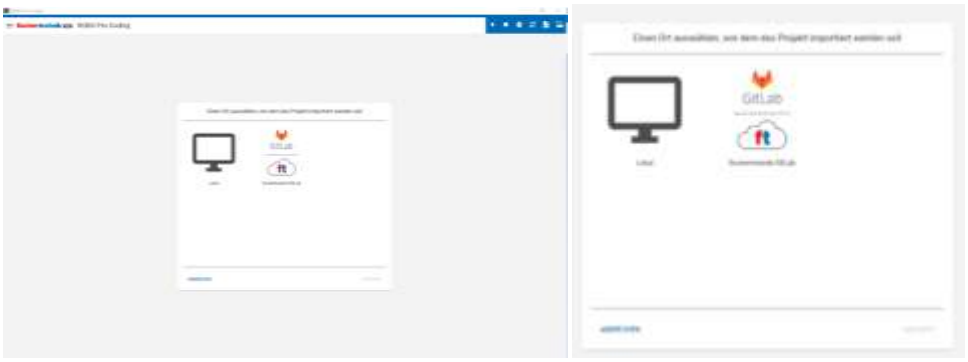
Unter Controller kann man sich den fischertechnik Controller auswählen, den man programmieren möchte. Hier in diesem Beispiel sind es der TXT 4.0 Controller und der BT Smart Controller. Ich benutze auch eine andere Version von Robo Pro Coding, wo auch der RX Controller zur Auswahl steht. Wenn man nun auf "Nächste" klickt, erschein diese Fenster:



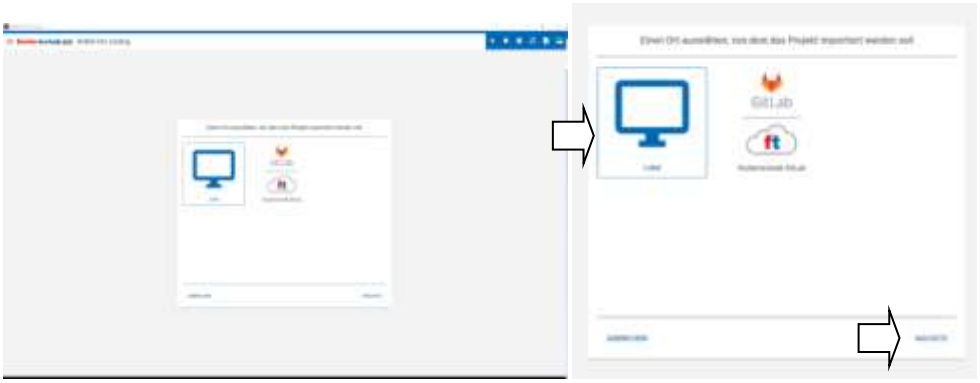
Hier kann man sich nun "leere" Projekte oder Beispielen aussuchen, mit denen man startet. Bei "Kamera", "Anzeige" und "Kamera und Anzeige" sind im leeren Projekt, beim Starten zusätzliche Elemente vorhanden. Die kann man sich aber auch später (über Datei Neu) noch dazu holen.

Wenn wir nun auf Leer und Erstellen klicken, wird ein neues Projekt angezeigt.

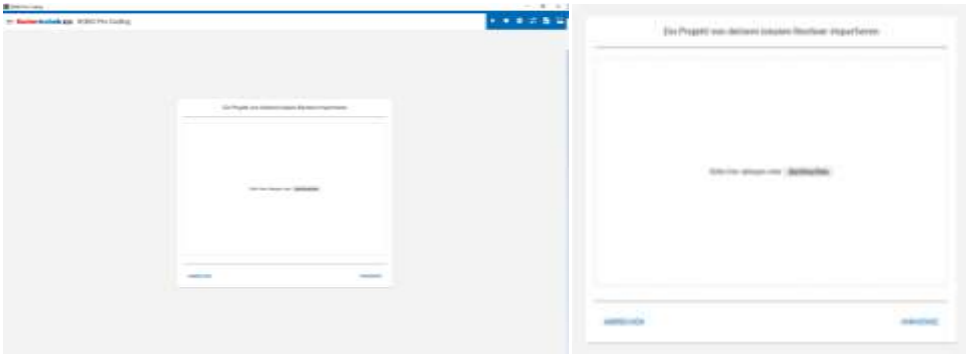
Importieren
Bei Importieren wird ein Projekt geladen und es kommt dieses Fenster:



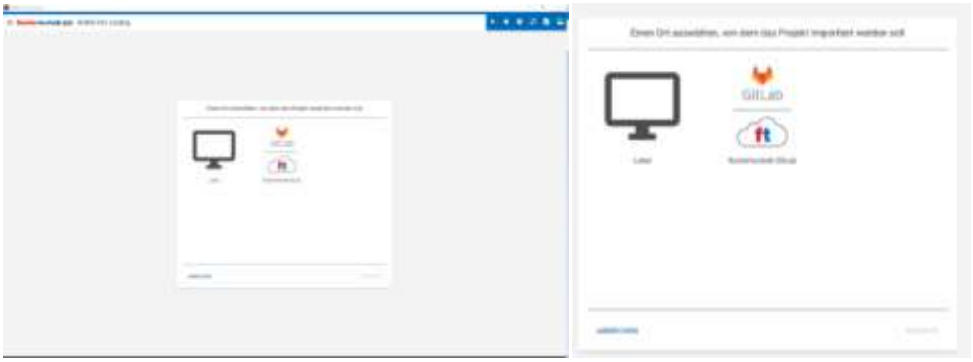
Hier kann man sich entscheiden, ob man "Lokal" also auf dem PC zu Hause oder von GitLab das Programm laden möchte. GitLab ist eine Internetseite, auf der fischertechnik, viele Programme für Robo Pro Coding gespeichert hat. Es kann sein, dass man sich -kostenlos- anmelden muss. Auch das Runterladen ist kostenlos.
Man kann sich mit seinen fischertechnik Fanclubdaten anmelden.



Nehmen wir Lokal (Speicher vom PC) und das blaue "Nächste".



Dann kommt dieses Fenster. Hier kann man den PC "durchsuchen", da öffnet sich der Explorer und man kann das Programm auswählen oder man zieht vom Explorer das Programm rüber.

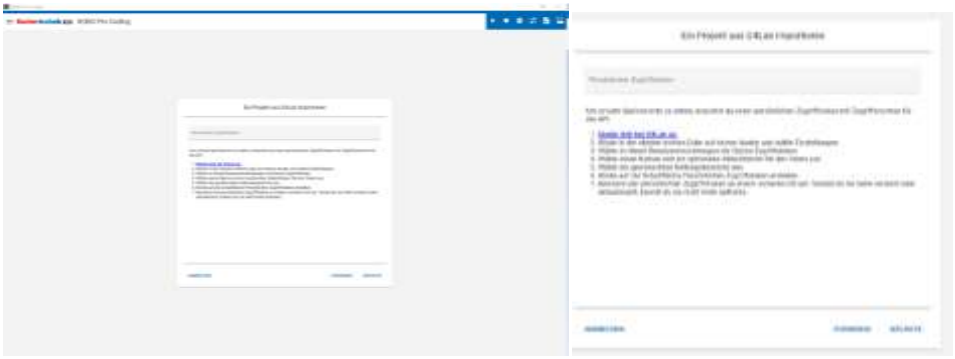


Auch hier kann man sich aussuchen, von wo man das Projekt laden möchte.

Achtung: In seltenen Fällen kann es sein, dass ein Programm mit Fehlermeldungen abbricht. Das kann von einem Übertragungsfehler beim laden der Datei gekommen sein. Einfach das Programm noch mal laden.

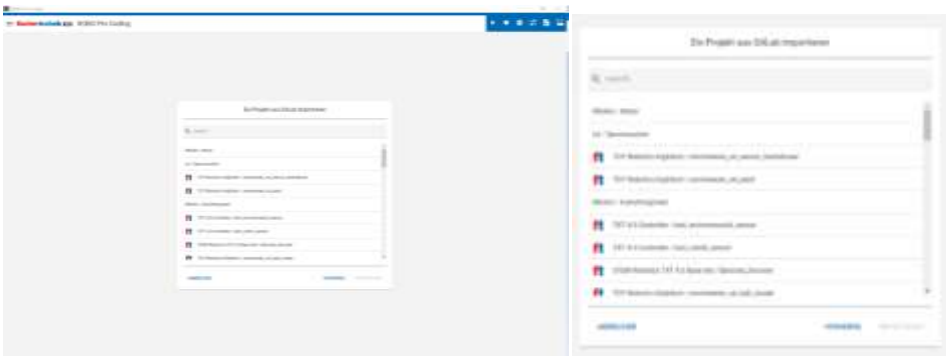
Unterschied von "Projekt" und "Programm"

Tipp: Was ist der Unterschied von "Projekt" und "Programm"? Ein Programm kann nur der Programmcode sein. Beim Projekt sind aber zusätzliche Daten mit dabei. Z.B. Welcher Controller benutzt wird, wie das Aussehen sein soll, was wo beim Display angezeigt werden soll usw. ...



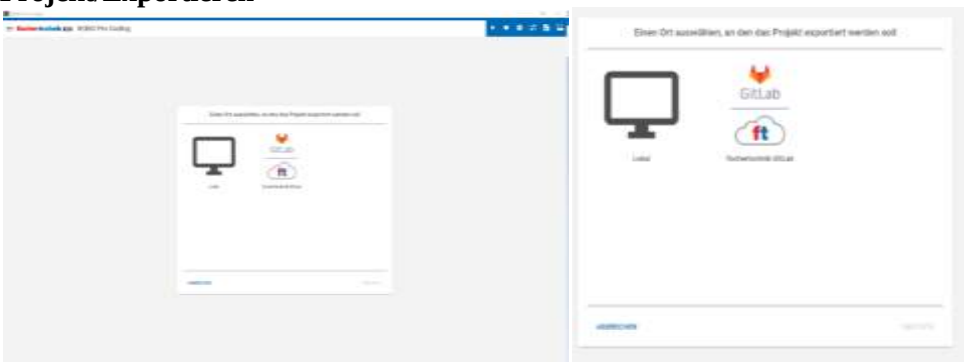
Bei GitLab muss man sich -kostenlos- anmelden, um an die Programme zu kommen.

Wenn man sich schon mal gemacht angemeldet hat, kann man "Nächste" klicken und...



es erscheint die Liste der Projekte und man kann sich eins aussuchen.

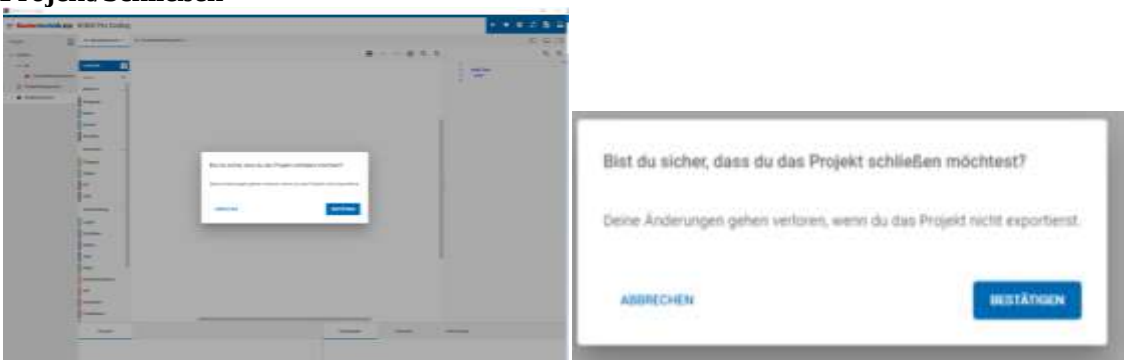
Projekt/Exportieren



Bei Exportieren kann man sein Projekt speichern. Man kann sich aussuchen, ob es Lokal auf dem PC oder im Internet auf GitLab gespeichert werden soll.

Wenn man Lokal und "Nächste" anklickt, gibt es ein Explorerfenster, wo man ein Verzeichnis auswählen soll oder das vorgeschlagene so lässt. Als Dateinamen wird „Untitled.ft“ vorgeschlagen. Hier sollte man seinen eigenen Namen eingeben oder den vom Projekt übernehmen. Wichtig ist die Endung .ft . Wenn man das nicht macht, wird das Projekt eventuell beim Importieren nicht angezeigt. Da muss man bei Dateityp auf "All Files" klicken. Man kann später auch den Namen im Explorer ändern und auch eine Dateiendung anfügen.

Projekt/Schließen



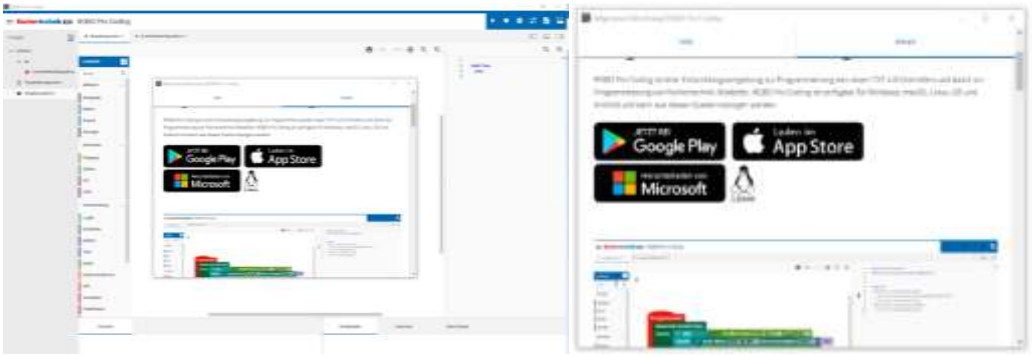
Hier gibt es eine Sicherheitsabfrage, ob man das wirklich machen möchte, da die Änderungen ansonsten verloren gehen - wenn man nicht gespeichert hat.

Zuletzt verwendete Projekte
Dieser Menüpunkt gibt eine Liste der Projekte, die man gespeichert hat.

Hinweis: Es kann sein, das je nach Robo Pro Coding Version, dieser Menüpunkt nicht vorhanden ist.

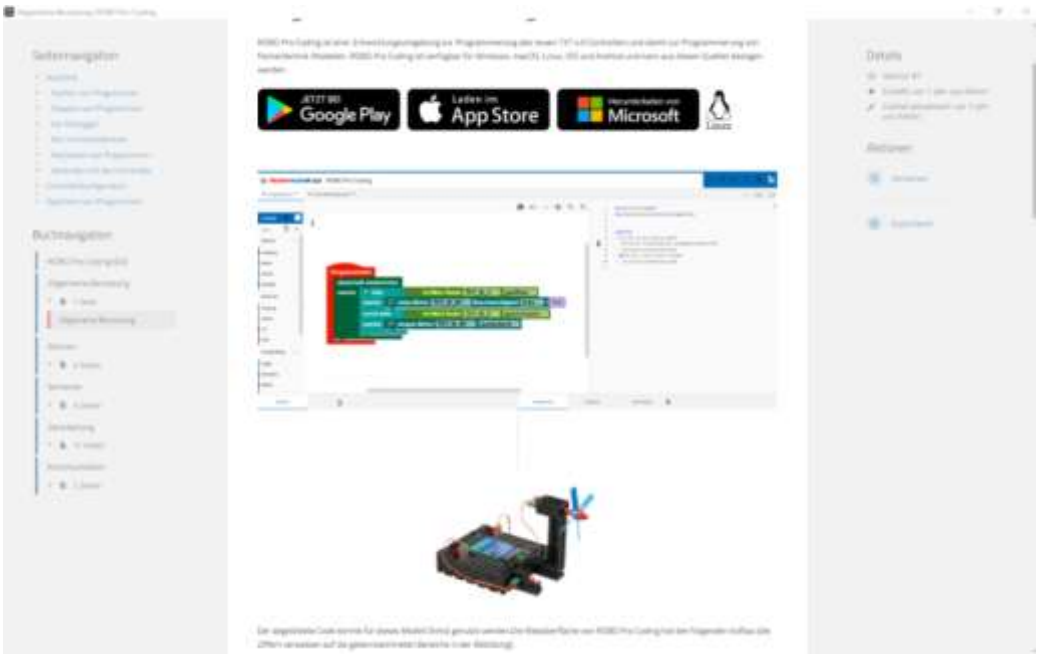
Ansicht/Neues Fenster
Dieser Menüpunkt erzeugt ein weiteres Fenster (eine weitere Instanz) von Robo Pro Coding. Hier kann man Blöcke aus einem anderen Projekt kopieren.

Hinweis: Es kann sein, das je nach Robo Pro Coding Version, dieser Menüpunkt nicht vorhanden ist.

Hilfe/Dokumentation


Hier startet die Dokumentation, wo als erstes die Quellen zum Runterladen angezeigt werden. Wir hier haben das schon gemacht. Wenn man runter scrollt, kommt mehr Hilfe.

Tipp: Fenster vergrößern



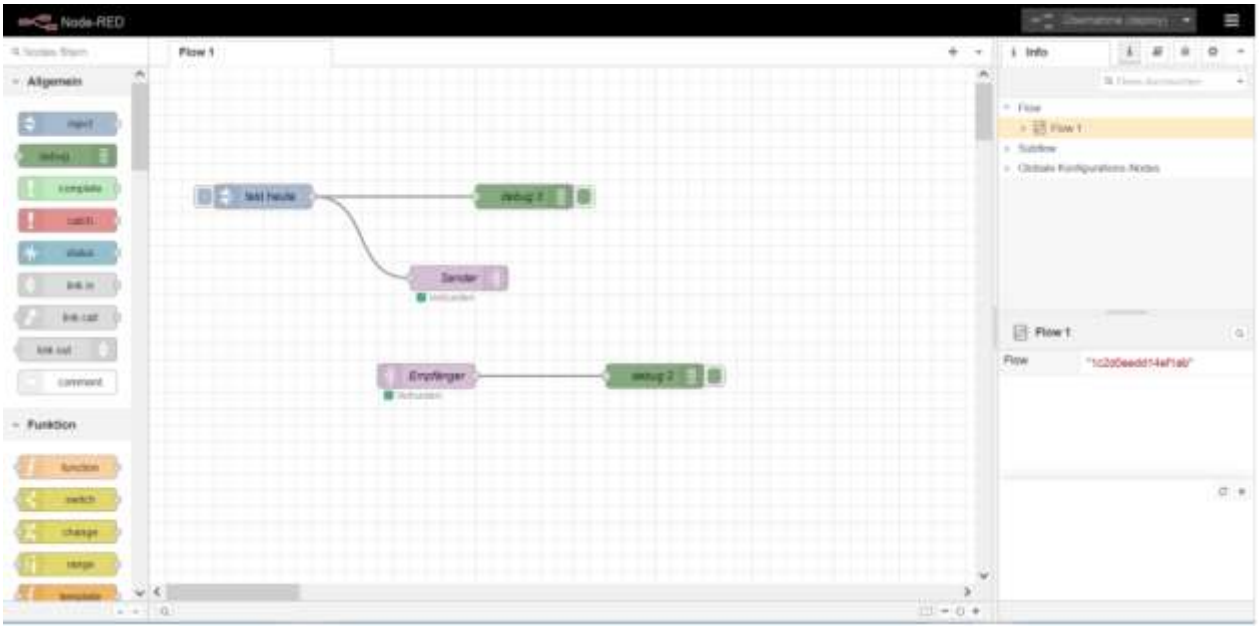
Hier erscheinen noch mehr Infos und Möglichkeiten. Z.B. kann man unter Exportieren eine PDF-Datei machen und diese ausdrucken. Zu manchen Themen ist diese Dokumentation etwas schweigsam.

Hinweis: Das untere abgebildete Modell in der Hilfe, hat einen älterem TXT Controller.

Datenschutzrichtlinie und Impressum
Unter Datenschutzrichtlinie und Impressum sind die entsprechenden Sachen von fischertechnik zu finden.

Node-Red Fow

Dieser Menüpunkt ruft im Browser die Adresse 192.168.7.2:1880 auf. Das ist der TXT 4.0, am USB Anschluss, wo der Port 1880 geöffnet wird. Das ist der Standardport von Red-Node. Es öffnet sich ein neuer Tab und das Programm auf dem TXT 4.0 wird gestartet und die Oberfläche im Browser angezeigt.



Hier ein Beispiel von Node-Red. Das Programm dient zu Empfangen, Verarbeiten und Senden z.B. von MQTT-Daten oder Daten aus dem Internet. Weite Infos siehe MQTT hinten im Buch.

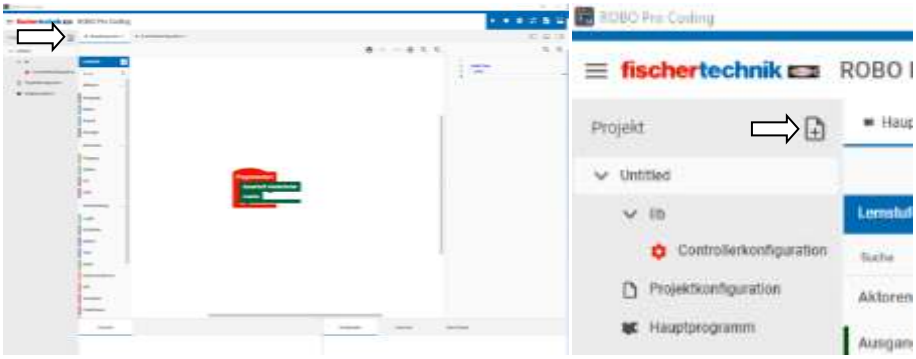
Hinweis: Es kann sein, das je nach Robo Pro Coding Version, dieser Menüpunkt nicht vorhanden ist.

Node-Red Dashboard

Dieser Menüpunkt ruft im Browser die Adresse 192.168.7.2:1880/ui auf. Das ist der TXT 4.0 am USB Anschluss, wo der Port 1880 geöffnet wird. Das ist der Standardport von Red-Node mit der Anwendung ui. Es öffnet sich ein neuer Tab und das Programm auf dem TXT öffnet das Dashboard. Es dient dazu Werte anzuzeigen oder auch Eingaben zu machen. Hier werden MQTT Daten angezeigt. Beim ersten Starten ist der Bildschirm leer.

Hinweis: Es kann sein, das je nach Robo Pro Coding Version, dieser Menüpunkt nicht vorhanden ist.

Neue Datei erstellen



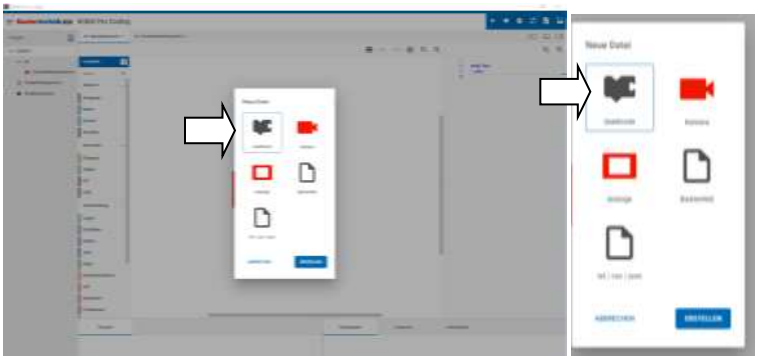
Unterhalb vom Burgermenü ist das Projektmenü. Rechts neben "Projekt" ist ein Symbol mit einem Plus. Das ist das Symbol für neue Datei. Man kann hier eine weitere Datei hinzufügen.

Neue Datei erstellen TXT 4.0/Alle Fähigkeiten
Wenn man daraufklickt

erscheint dieses Fenster (beim TXT 4.0 „alle Fähigkeiten“).

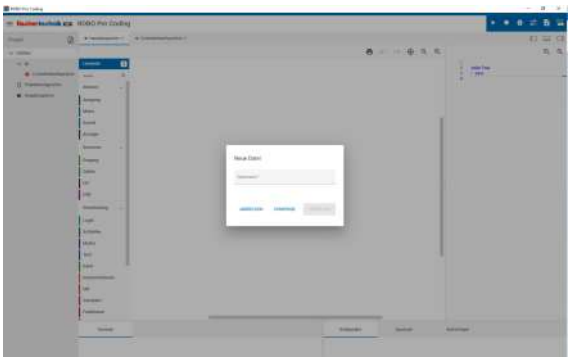
Zur Auswahl stehen:

Quellcode
Kamera
Anzeige
Bedienfeld
txt/cvs/json

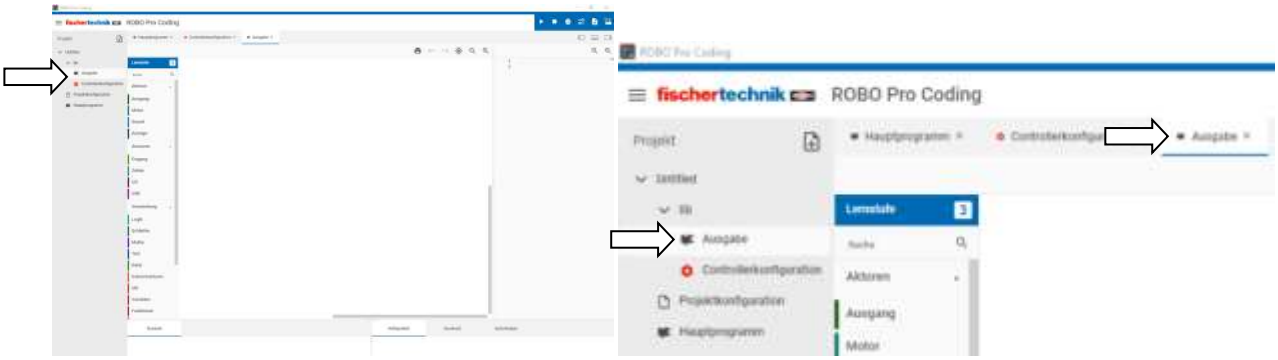


Ein Weiterer Klick auf "Erstellen" erzeugt die Datei und sie wird angezeigt.

Anmerkung. Wenn ein Bedienfeld erschaffen wurde, erscheint es nicht mehr in der Auswahl.

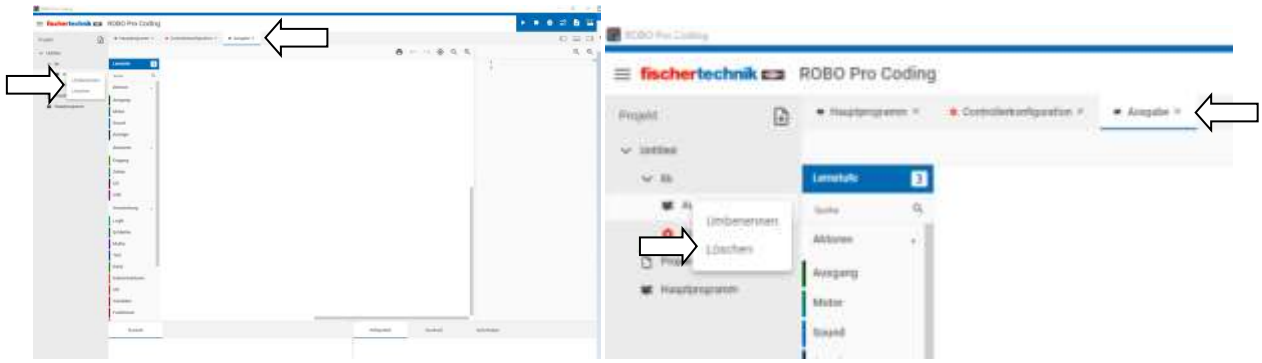


Man wird aufgefordert der Datei einen Namen zu geben.



Der Dateiname wird links oben und zusätzlich oberhalb vom mittleren Bereich angezeigt. Da hier in dem Beispiel ein Quellcode genommen wurde, kann man hier die Blöcke für die Ausgabe rein ziehen die später dem Hauptprogramm hinzugefügt werden. Es können Unterprogramme sein, die dann vom Hauptprogramm aufgerufen werden. Es können aber auch eigenständige Programme die parallel zum Hauptprogramm laufen. Dazu aber noch später mehr.

Alle Dateien werden in einem "LIB"- Ordner gespeichert. Man "kann" diese Lib-Dateien auch in anderen Projekten benutzen.

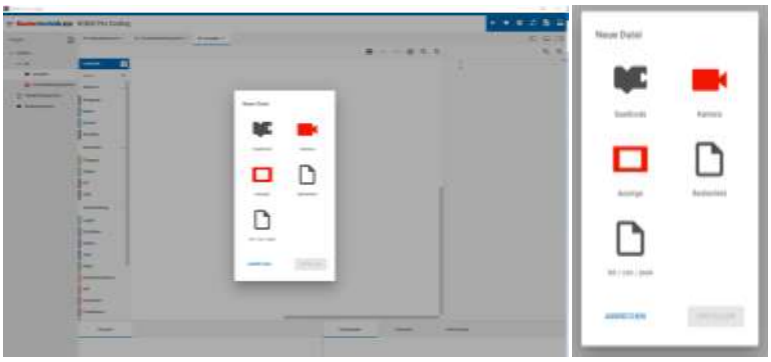


Löschen kann man diese Dateien, in dem man über das Menü mit der linken Maustaste links oben auf den Dateinamen klickt. Das Kreuz oben über der Mitte löscht nicht die Datei, sondern es wird nur die Anzeige in der Mitte gelöscht. Zum erneuten Anzeigen einfach auf den Dateinamen oben links klicken.

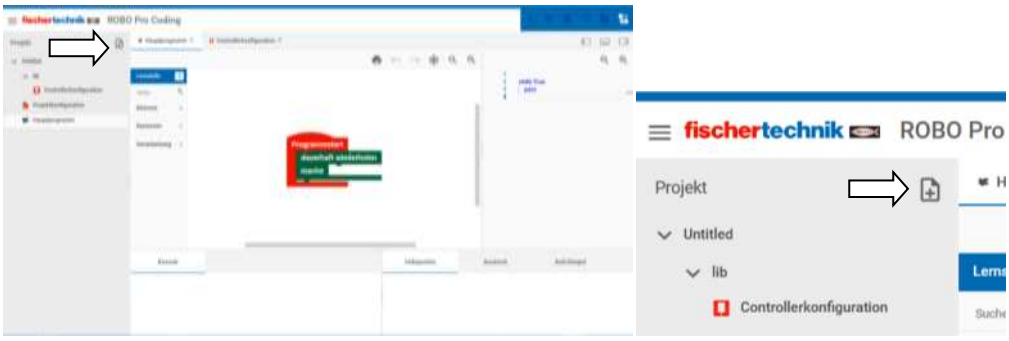
Wenn die Datei gelöscht ist, ist sie weg.

Ansonsten muss man das Projekt **vorher speichern**. Es gibt *kein* "Rückgängig machen".

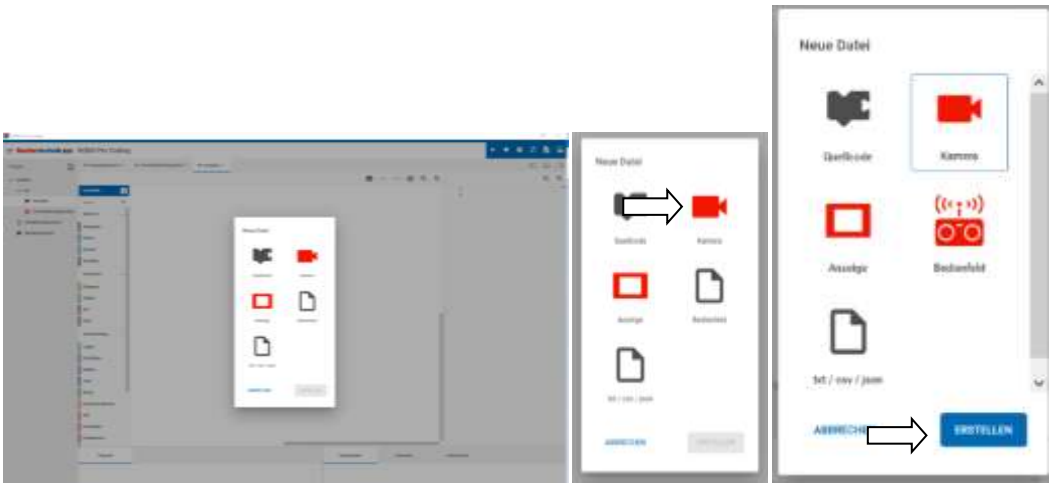
Kamera/ Neue Datei erstellen Kamerakonfiguration



Um eine Kamera mit dem TXT 4.0 zu benutzen, muss man eine neue Datei „Kamera“ hinzufügen. Diese Datei stellt alle notwendigen Bedienelemente für die Kamera bereit.



Dazu auf „Datei Neu“



dann auf „Kamera“ und „Erstellen“ klicken.

Es erscheint Kamerakonfiguration in der Dateiliste links und in der Mitte ein leeres Kamerabild.

In der Mitte links ist dieses Menü:

Das ist der

„Color Detector“ der Farbenfinder, Farberkennung

„Motion Detector“ der Bewegungsfinder, Bewegungserkennung

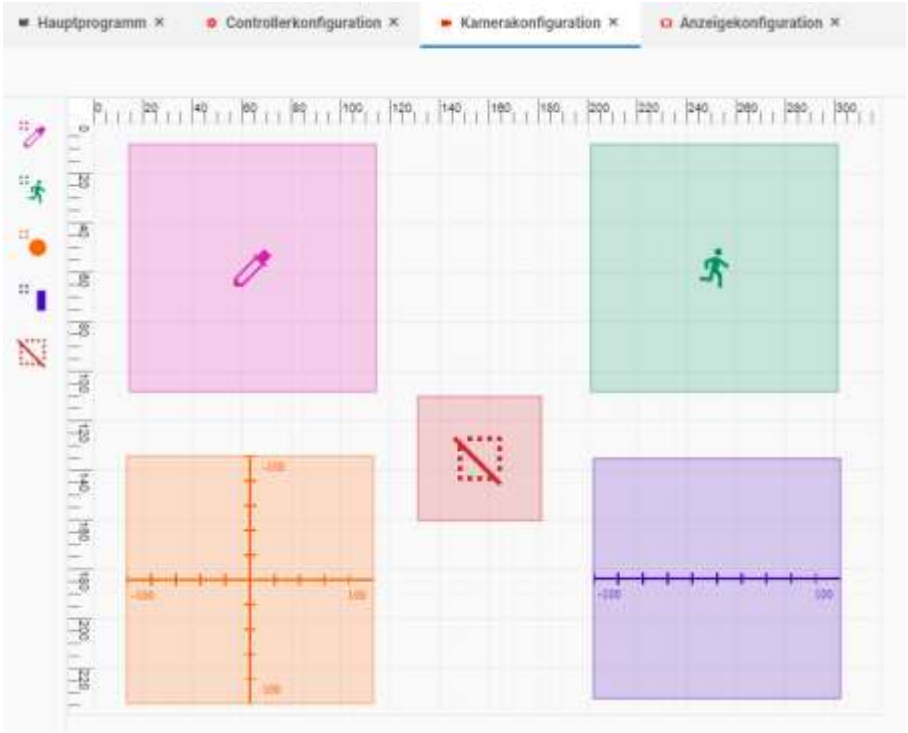
„Ball Detector“ der Ballfinder, Ballerkennung

„Line Detector“ der Linienfinder, Linienerkennung

„Blocked Area“ Sperrfläche

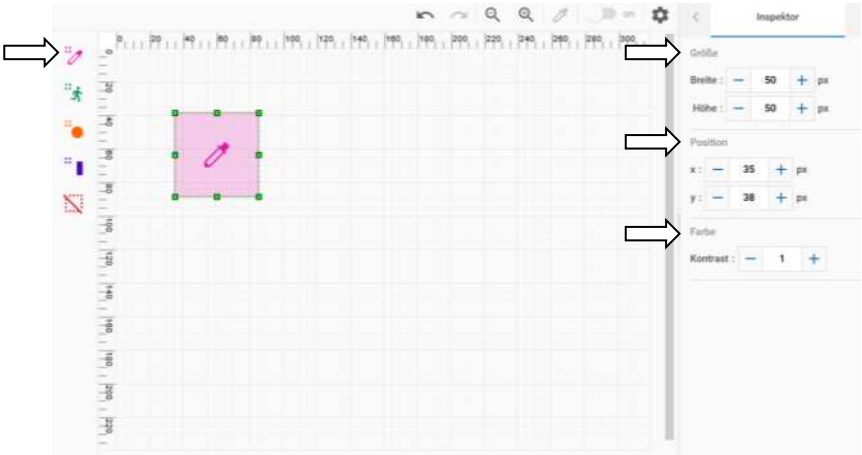
Wie bei einem Malprogramm kann man auf das Kamerabild diese oben genannten Flächen verteilen. Dazu zieht man das gewünschte Element auf die Kamerafläche.

Das kann dann so aussehen:

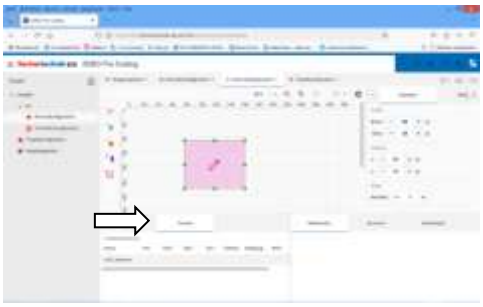


Diese Flächen vom Kamerabild werden also entsprechend untersucht.

Color Detector der Farbenfinder, Farberkennung



Beim „Color Detector“ (Farbenfinder) hat man die Möglichkeit, die Größe, Position und bei der Farbe den Kontrast des eben gemalten Feldes einzustellen. Die Größe und Position kann auch mit der Maus verändert werden, in dem man auf den Rand oder die grünen Quadrate klickt und mit der gedrückten Maustaste zieht. Rechts beim „Inspector“ kann man die Werte sehen. Sie entsprechen denen der Scala oben und links.



Im unteren Bildschirm wird „Kamera“ angezeigt.

FARBERKENNUNG

Name	Rot	Grün	Blau	Hex	Hue	Saturation	Wert
color_detector	118	115	107	76736b	43	9	46

Bei einem Bild kann das dann so aussehen.

Hue = Farbton
Saturation = Farbsättigung

Flächen der Kamerakonfiguration löschen

Um eine Fläche zu löschen kann man diese mit der Entfernen Taste löschen oder über das Menü mit der rechten Maustaste.

Ebenen

Man kann die Flächen auch in Ebenen anordnen. Das hat den Vorteil, dass man dann verschiedene Möglichkeiten hat, ein Bild zu untersuchen.

Mal untersucht man die Ebene 1 dann die Ebene 2…
Dazu kann man Felder auch unsichtbar machen.

Kamera-Bilder scharf stellen

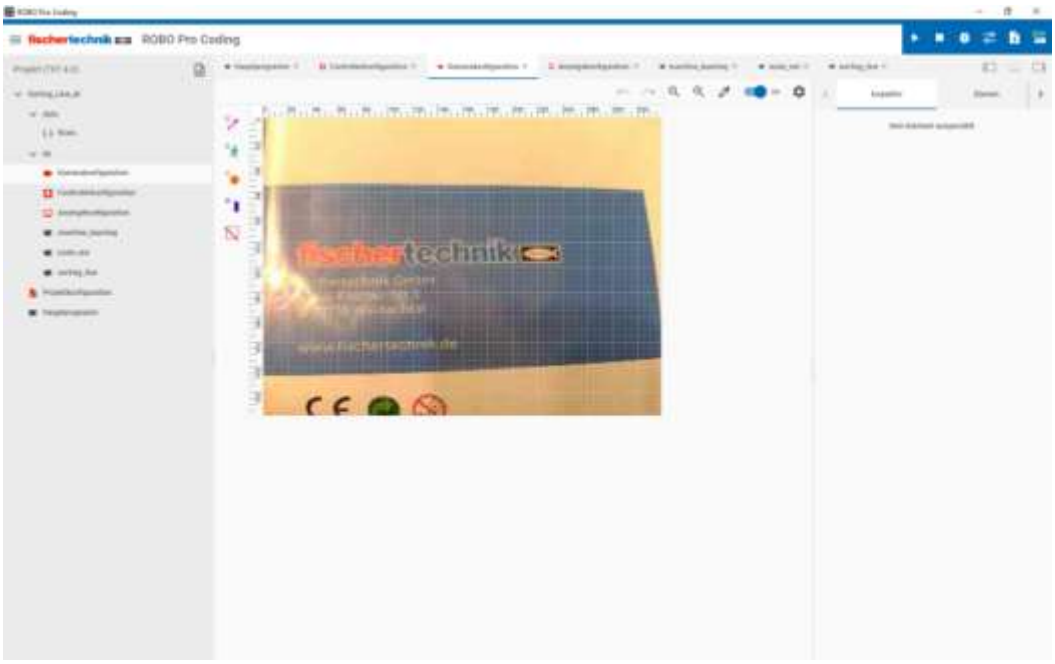
Die Bild-Schärfe stellt man durch das Drehen der Linse an der Kamera ein.

Dazu muss man aber das Kamerabild sehen.

Kamerabild bei Robo Pro Coding ansehen, Liveansicht PC Bildschirm

In der Kamerakonfiguration gibt es einen Schalter (links neben dem Zahnrad). Standartmäßig ist er ausgeschaltet. Wenn man ihn einschaltet, kommt folgende Abfrage:

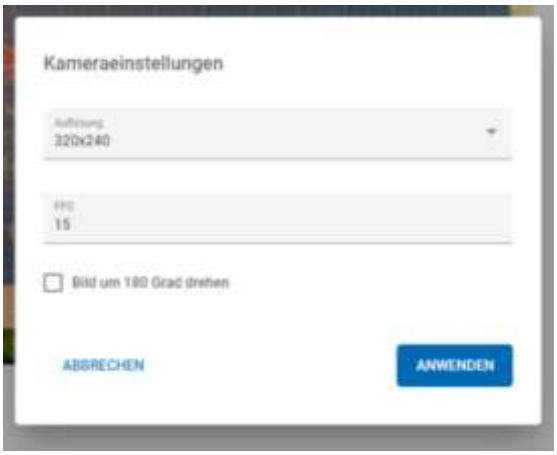
Dadurch, dass viele Daten übertragen werden, bekommt man den Hinweis, dass alle Programme beendet werden. Hier kann man bestätigen bzw. auch „Dialog nicht mehr anzeigen“ anwählen.



Das aktuelle Kamerabild wird nun angezeigt.

Kamera Einstellungen

Hier kann man zwei Einstellungen für die Kamera vornehmen.

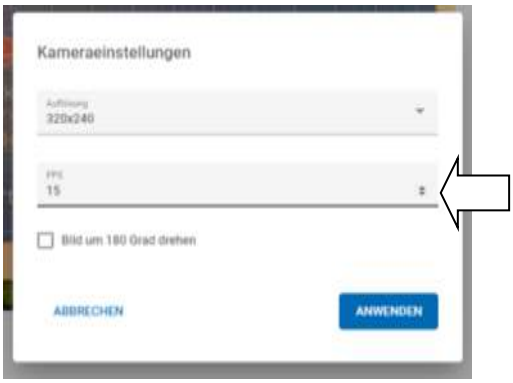


Zum einen ist das die Auflösung der Kamera und FPS. FPS ist „Frames per Second“ also Bilder pro Sekunde. Das ist die Bilderanzahl, die in einem Stream (Datenstrom) gesendet wird. Mehr Bilder, desto größer die Datenmenge.

Bei der fischertechnik- Kamera kann man unter zwei Auflösungen wählen:

Dazu auf den Pfeil rechts klicken und dann kann man auf 320x240 oder 160x120.

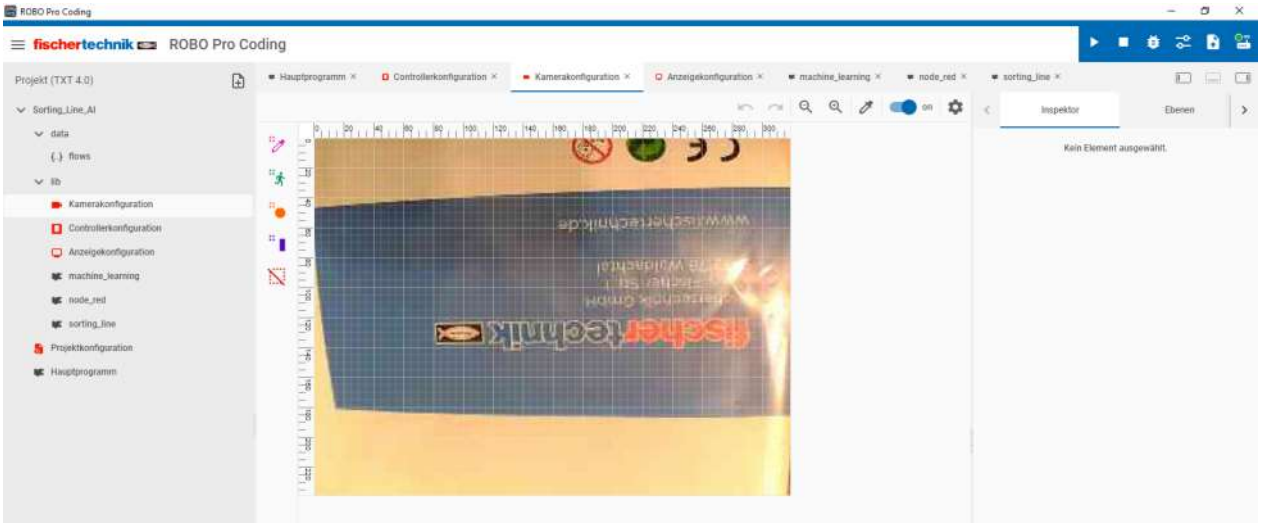
Die Zahlen geben die Anzahl der Pixel (Bildpunkte) in X- und Y-Richtung an.
Man muss bedenken, dass die Verarbeitung großer Datenmengen Zeit kostet. Wenn man z.B. eine Ballerkennung macht, dauert es bei hoher Auflösung länger. Die Daten werden einem Unterprogramm z.B. „Open CV“ übergeben und dort analysiert. Das Ergebnis wird dann dem Programm übergeben, dann kommt ein neues Bild usw.



Die zweite Einstellung ist FPS. Das sind die Bilder pro Sekunde. Wenn man mit dem Mauszeiger auf das Feld geht, kommen zwei Pfeile, womit man die Anzahl erhöhen oder reduzieren kann. Die Standarteinstellung ist 15. Man muss einfach ausprobieren ob es sinnvoll ist die Zahl zu verändern.

Kamerabild drehen
Je nach Einbaulage der Kamera, muss/kann man das Bild drehen.

Hier kann man das Bild um 180° drehen.



Kamerabild Liveansicht TXT 4.0 Display
Statt des PC-Bildschirms kann man das Display des TXT 4.0 nutzen, um das Bild anzuzeigen. Unter Burgermenü/Neu/fischertechnik-Beispiele/ TXT 4.0, gibt es zwei Programme.
Das einfachere Beispiel ist „test_camera_nc“.

Siehe dazu das fischertechnik Beispiel zum TXT 4.0/test_camera_nc oder:
https://git.fischertechnik-cloud.com/fischertechnik-examples/txt-4.0-controller/test_camera_nc

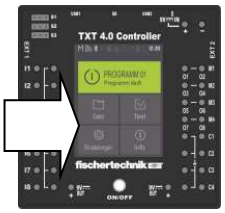
Das Beispielprogramm lädt das Kamerabild auf das Display vom TXT 4.0 . Man kann die Anzeige über einen Schalter am Display auch ausschalten.

Fischertechnik Kamera als PC-USB Kamera
Eine andere Möglichkeit ist, die Kamera als normale USB-Kamera an den PC anschließen und z.B. bei Windows 10 über die Kamera APP das Kamerabild am PC sehen. Dazu muss man die Freigabe der Kamera und des Mikrofons für die Kamera App aktivieren. Die für anderen APPs kann man deaktivieren.

Tipp: Das Testprogramm hochladen und dann am TXT 4.0 starten. Nicht über Robo Pro Coding starten, da die Verbindung z.B. über USB zu langsam ist, da viele Daten hin und her gesendet werden. Es kommt zu Aussetzern in der Bilddarstellung.

Man „kann“ auch andere Kameras benutzen. Diese sollten die Möglichkeit haben die Auflösung und eventuell die FPS runter zusetzen. Man muss es einfach ausprobieren, ob es geht.

Anzeige / Neue Datei erstellen Anzeigenkonfiguration (TXT 4.0-Display)



Das Display des TXT 4.0 ist ein Touchscreen. Das Berühren von Elementen auf dem Display, kann man im eigenen Programm nutzen.
Mit den Blöcken aus dem Reiter Anzeige lässt sich der Bildschirm/Display des TXT 4.0 Controllers gestalten und nutzbar machen. Dies geschieht in zwei Schritten:

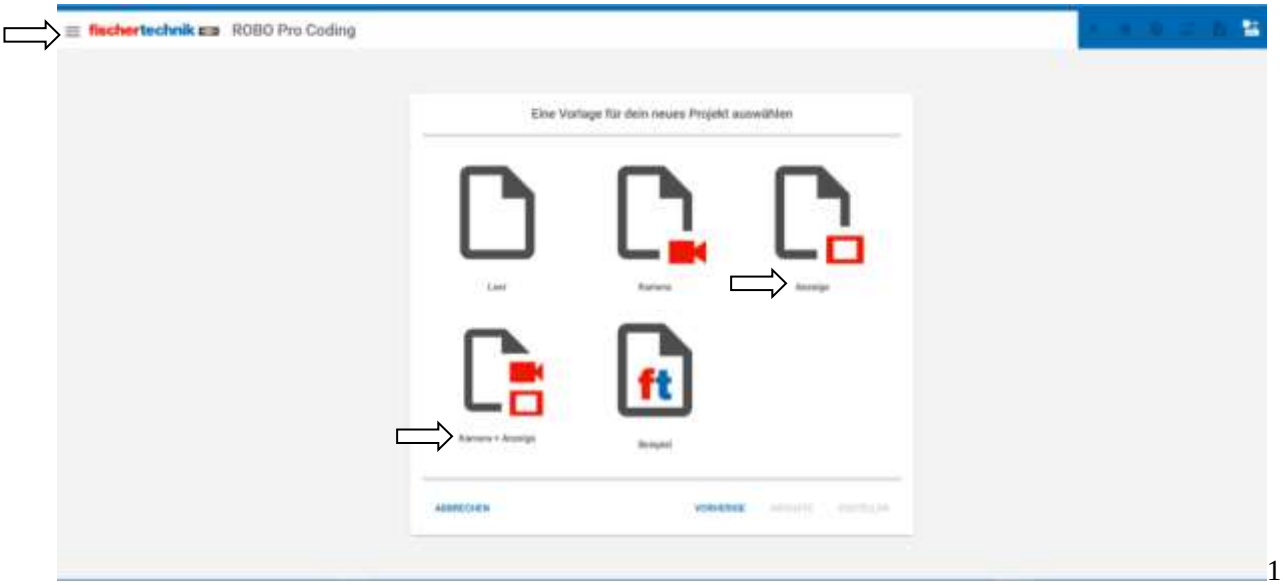
1. Konfigurieren, das heißt:
   - Eine neue Datei der Kategorie Anzeige öffnen, über das Seiten Symbol mit dem Plus oben links (neben Projekt)
   - die gewünschten Elemente auf den gerasterten Bereich ziehen (er stellt den konfigurierbaren Teil des Displays dar)
   - bei Bedarf anpassen.
2. Programmieren, das heißt:
   - Im Hauptprogramm mit den Blöcken der Kategorie Anzeige die Wirkung vom Berühren mit dem Display programmieren.

Das „TXT“ im Namen des Feldes zeigt an, dass es auf dem Display des TXT 4.0 Controller ist. Weitere Informationen sind unter Anzeige (Reiter) im Hauptprogramm.

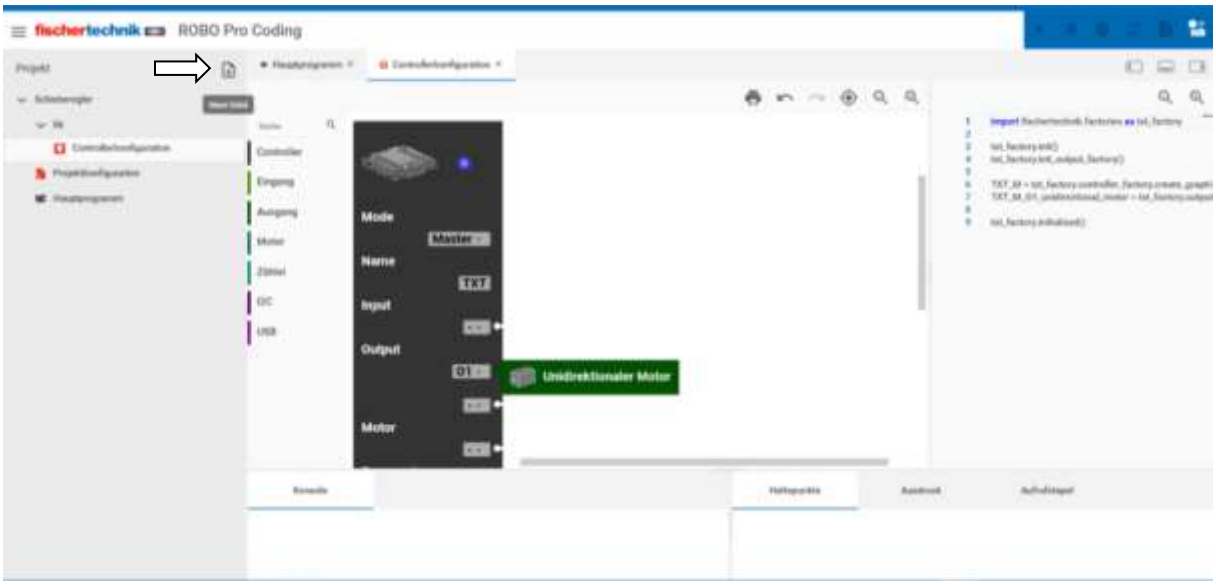
Hinweis/Tipp: Fehlermeldung nach Anzeige löschen

Wenn man einmal eine Anzeige im Programm erschaffen hat und nicht mehr braucht, sollte man sie trotzdem im Programm drin lassen und nicht löschen. Momentan ist es so, dass Teile im Pythoncode nicht gelöscht werden und das Programm z.B. mit der Fehlermeldung „ImportError: No module named 'lib.display'“ abbricht. Da hilft es nur, eine leere Anzeige wieder zu aktivieren.

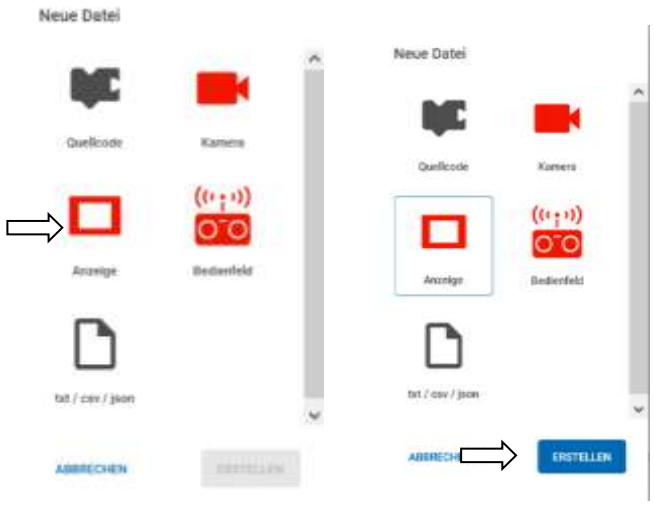
Es gibt zwei Möglichkeiten die neue Datei „Anzeige“ zu erschaffen.



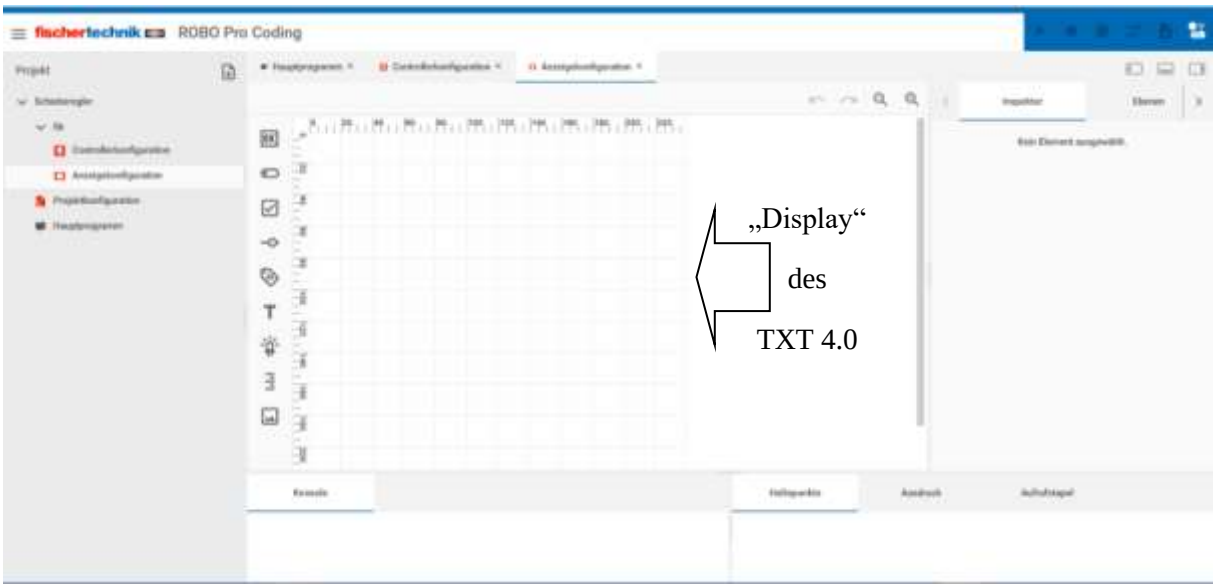
1. Über das Burgermenü/Projekt/Neu/Optionen und dann „Anzeige“ oder „Kamera und Anzeige“ auswählen.



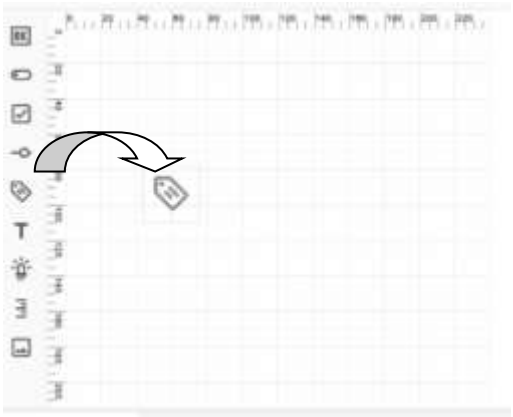
2. Über das Pluszeichen neben Projekt.



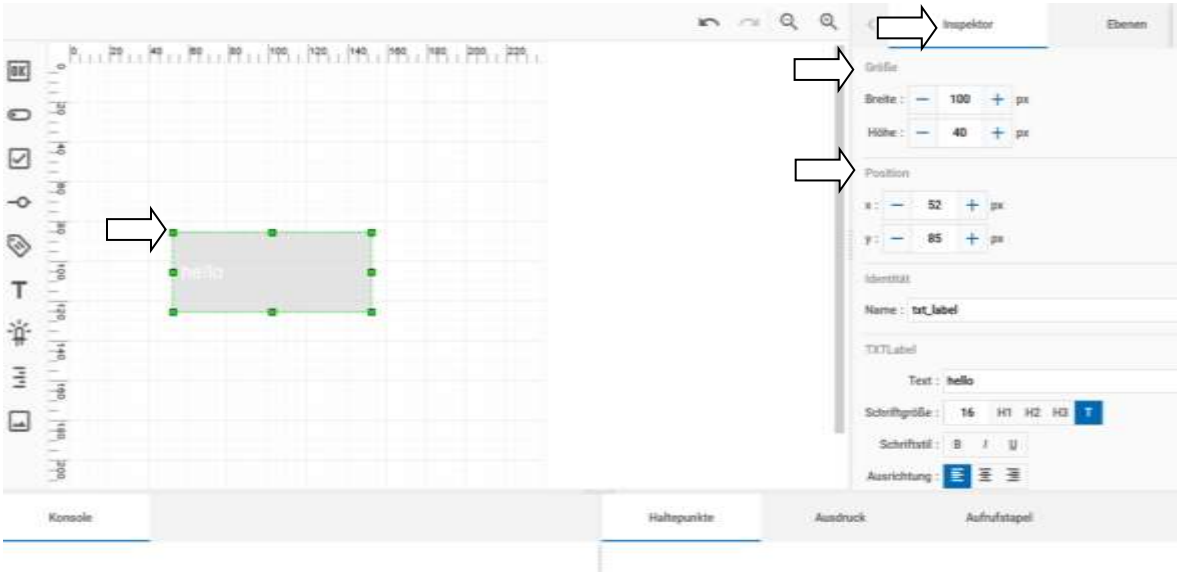
Da kann man dann auf Anzeige klicken und dann auf das blaue „Erstellen“.



Es erscheint die Anzeigenkonfiguration. In der Mitte wird das leere Display vom TXT 4.0 angezeigt. Links sind Felder, die man auf das Display rüber ziehen kann. So kann man auswählen, was wo angezeigt werden soll.



Hier ziehen wir mal ein Beschriftungsfeld auf das „Display“.



Das Beschriftungsfeld wird nach dem Loslassen der Maustaste angezeigt. Man kann jetzt mit der Maus die Position und die Größe an den grünen Punkten des Feldes ändern. Rechts ist ein „Inspektor“. Hier kann man die genauen Werte für die Größe und Position eingeben.

Unter Größe und Position wird der Name des Feldes eingetragen. Robo Pro Coding gibt hier den Namen „txt_label“ schon vor. Als Text wird „hello“ vorgegeben. Das ist der Text, der beim Starten vom Programm auf dem Beschriftungsfeld angezeigt wird. Weiterhin kann man die Schriftgröße, Stil, Ausrichtung und die Farbe ändern.
So kann man nun weitere Felder hinzufügen. Jedes Feld muss einen eigenen Namen haben, um die Eigenschaften wie Größe usw. zuordnen zu können. Jedes weitere Feld vom selben Bereich wird als Name automatisch durchnummeriert. Dennoch sollte man nachvollziehbare Namen wählen.

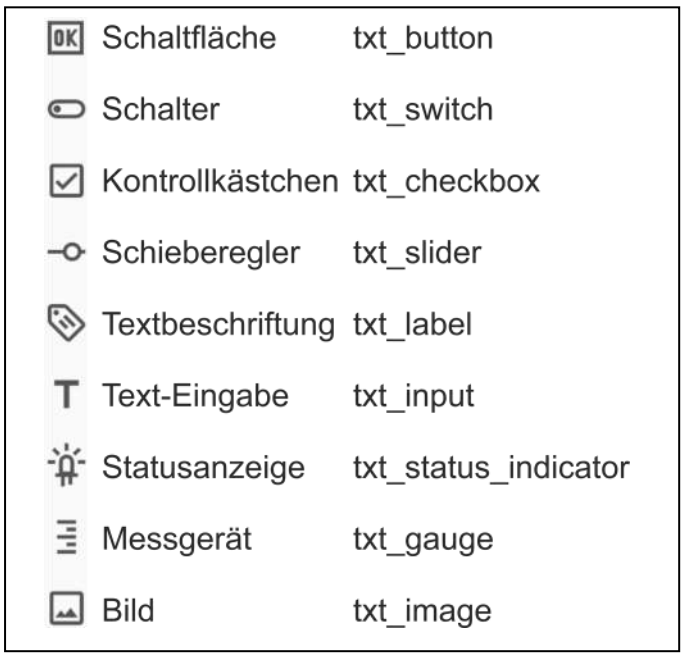
Um mehrere Möglichkeiten von Bildschirmeingaben machen zu können, kann man Felder auch ausblenden = Deaktivieren oder wieder anzeigen = Aktivieren.
Auch kann man Ebenen erstellen um so verschiedene Ansichten, Menüs oder Bedienfelder zu haben.

Farbiger Text mit Formatierung:

Programmausschnitt Add_On_AI
Hier wird ein Text auf dem TXT 40 Display, fett (b) und farbig (font color…) „PASSED“ (grün) angezeigt.
Hier wir ein Block „containInHTML mit“ aus Funktion genommen.

Übersicht der Felder in der Anzeigenkonfiguration

Achtung! Die Reihenfolge der Blöcke im Hauptprogramm und die Reihenfolge der Menüpunkte in der Anzeigenkonfiguration, sind nicht gleich!



Reihenfolge wie in der Anzeigenkonfiguration:

Schaltfläche


TXTButton

Die Schaltfläche ist ein beschriftetes Feld, das gedrückt werden kann. Drückt man die Schaltfläche auf dem Display, läuft das Schaltflächen-Programm ab, sobald sie wieder losgelassen wird. Das zugehörige Symbol für die Schaltfläche ist das Quadrat mit der "OK" Beschriftung. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe der Schaltfläche in Pixeln,
- die Position der Schaltfläche in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Knopfes),
- der Name der Schaltfläche,
- den Text, der auf der Schaltfläche steht und
- die Aktivität der Schaltfläche

festgelegt werden.

Achtung!
Es werden beim „Drücken“ **und** beim „Loslassen“ der virtuellen Schaltfläche, jeweils ein Tastensignal gesendet. Meist funktioniert eine einfache Abfrage trotzdem, weil die Tastenabfrage sehr kurz ist und das Hauptprogramm länger braucht. Wenn beide sehr kurz sind, muss man es berücksichtigen. Am einfachsten ist es, eine Variable in der Tastenabfrage zu setzen und diese im Hauptprogramm abzufragen. Da spielt es weniger die Rolle, ob diese Variable zwei Mal gesetzt wurde.

Inspektor für Schaltfläche

Mit diesen Blöcken werden die Schalter programmiert:

setze Schaltfläche aktiviert

ist die Schaltfläche eingeschaltet

Schalter


TXTSwitch - Schalter

Der Schalter kann zwei Positionen einnehmen und befindet sich immer in genau einer dieser beiden Positionen. Je nach Position gibt er **wahr** oder **falsch** zurück. Das zugehörige Symbol für den Schalter ist das Oval mit dem Punkt. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Schalters in Pixeln,
- die Position des Schalters in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Schalters),
- der Name des Schalters,
- den Text, der neben dem Schalter steht,
- die Aktivität des Schalters und
- den Zustand in dem sich der Schalter bei Start des Programms befinden soll

angepasst werden.

Inspektor für Schalter

Mit diesen Blöcken werden die Schalter programmiert:

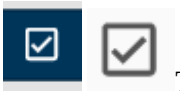
setze Schalter

Markierten oder aktivierten Zustand des Schalters auf wahr oder falsch festlegen.

Der Block übernimmt zwei Funktionen. Man kann entweder die Aktivität (enabled im Dropdown-Menü wählen) oder den Zustand (checked im Dropdown-Menü wählen) auf wahr oder falsch setzen.

ist Schalter

Kontrollkästchen


TXTCheckBox

Das Kontrollkästchen kann zwei Zustände annehmen und befindet sich immer in genau einem dieser beiden. Je nach Zustand gibt es **wahr** oder **falsch** zurück. Das Symbol für das Kontrollkästchen ist das Quadrat mit dem Haken. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Kontrollkästchens in Pixeln,
- die Position des Kontrollkästchens in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Kontrollkästchens),
- der Name des Kontrollkästchens,

- den Text, der neben dem Kontrollkästchen steht,
- die Aktivität des Kontrollkästchens und
- den Zustand in dem sich das Kontrollkästchen bei Start des Programms befinden soll

festgelegt werden.

Inspektor für die Kontrollkästchen.
Die Kontrollkästchen werden mit diesen beiden Blöcken betrieben:

setze Kontrollkästchen

ist Kontrollkästchen

Schieberegler

 TXTSlider-Schieberegler

Der Schieberegler gibt Werte abhängig von seiner Position zurück. Die Position kann dabei über den Touchscreen verändert werden. Der Wert kann über den **Ereignis []**-Block abgerufen werden, sobald der Schieberegler ruht. Der abgerufene Wert ist eine Dezimalzahl. Will man den Wert des Schiebereglers ganzzahlig haben, muss man den **runde**-Block  einsetzen. Das zugehörige Symbol für den Schieberegler ist der Strich mit dem Kreis. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Schiebereglers in Pixeln,
- die Position des Schiebereglers in Pixeln (auf dem angegebenen Punkt liegt dann die obere linke Ecke des Schiebereglers)

- der Name des Schiebereglers,
- die Aktivität des Schiebereglers,
- die Ausrichtung des Schiebereglers,
- den Wertebereich der über den Schieberegler abgedeckt wird und
- der Wert, auf dem der Regler bei Start des Displays steht

festgelegt werden.

Inspektor für Schieberegler

Hinweis: Der Runde-Block ist in der Dokumentation zu Robo Pro Coding. Vermutlich war das in älteren Versionen von Robo Pro Coding nötig. Wenn man sich den Python Code anschaut, sieht man, dass dort ein „Int“ für Integer steht und die gelieferten Zahlen ganzzahlig sind.

Mit diesen Blöcken wird der Schieberegler programmiert:

setze Schieberegler Wert

Setzt den Slider auf den entsprechenden Wert (kann auch der Inhalt einer Variablen sein).

hole Schieberegler Wert

Holt den Wert des Sliders, um ihn im Programm weiterverarbeiten zu können

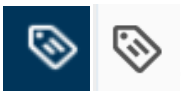
setze Schieberegler aktiviert

Im Grunde noch nicht ausgetestet, was der Befehl genau macht. Vom Namen her, sollte er den Slider bedienbar machen oder eben nicht.

Auch die Kombination mit anderen Blöcken, ist noch nicht ganz klar. Da gibt es noch Bedarf es genau auszutesten, da es sehr viele Möglichkeiten gibt.

ist Schieberegler aktiviert

Beschriftungsfeld


TXTLabel - Textbeschriftung

Mit dem Element Beschriftungsfeld kann man einen Text auf dem Bildschirm platzieren. Das Symbol in der Anzeigekonfiguration ist das Etikett. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Beschriftungsfelds in Pixeln,
- die Position des Beschriftungsfelds in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Textfeldes),
- der Name des Beschriftungsfelds und
- der Inhalt des Beschriftungsfelds (dieser Text wird bei Start des Displays abgebildet)

festgelegt werden.

Inspektor für das Beschriftungsfeld. Hier sind die Sachen aufgelistet, die für das Feld geändert werden können.

Diese Blöcke kann man im Programm mit dem Beschriftungsfeld verwenden:

setze Beschriftungsfeld Text

Je nach Robo Pro Coding Version, kann es sein, dass der Block nicht verfügbar ist.

hole Beschriftungsfeld Text

Texteingabe / Eingaben


TXTInput Text-Eingabe

Das Element **Eingabe** erlaubt es dem Nutzer, dass über den Controller Text eingeben werden kann. Das zugehörige Symbol in der Anzeigenkonfiguration ist das „T" Zeichen. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Eingabefeldes in Pixeln,
- die Position des Eingabefeldes in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Eingabefeldes),
- der Name des Eingabefeldes und
- der Inhalt des Eingabefeldes (dieser Text wird bei Start des Displays abgebildet)

festgelegt werden.

Inspektor für Texteingabe

Mit diesen Blöcken wird das Eingabefeld programmiert:

setze Eingabefeld Text

hole Eingabefeld Text

Statusanzeige

 StatusIndicator Statusanzeige

Der Statusindikator zeigt die Aktivität von etwas an. Je nach Status leuchtet er ("aktiv") oder tut dies nicht ("inaktiv"). Das Symbol in der Anzeigenkonfiguration ist eine leuchtende Diode. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

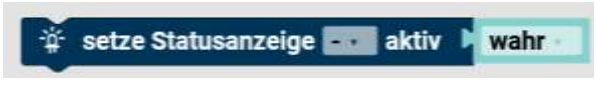
- die Größe der Statusanzeige in Pixeln,
- die Position der Statusanzeige in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Statusindikators),
- der Name der Statusanzeige,
- die Farbe der Statusanzeige und
- ob die Statusanzeige zu Beginn aktiv oder inaktiv sein soll,

festgelegt werden.

Inspektor für Statusanzeige

Mit diesen Blöcken wird die Statusanzeige programmiert:

setze Statusanzeige aktiv



ist die Statusanzeige aktiv

Messinstrument

 Gauge-Messgerät

Die Messinstrument-Funktion kann Werte (keine Werte kleiner 1) darstellen. Das zugehörige Symbol in der Anzeigenkonfiguration ist die Skalierung. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Messinstruments in Pixeln,
- die Position des Messinstruments in Pixeln (auf dem angegebenen Punkt lieg die obere linke Ecke der Messlehre),
- der Name des Messinstruments,
- die Ausrichtung des Messinstruments
- der Wertebereich, den das Messinstrument darstellt, und
- der Wert des Messinstruments, der bei Start des Displays gezeigt wird

festgelegt werden.

Inspektor für das Messgerät

Diese Blöcke können mit dem Messgerät programmiert werden:

setze Messinstrument auf Wert

hole Messinstrument Wert

setze Bild Base64-Bild

 TXTImage

Für ein Bild im Display ein Base64-Bild festlegen. Base64 ist ein Datenformat, wo die Binärzahlen des Bildes, in ASCII-Code (und umgekehrt) umgewandelt werden.
Das Symbol in der Anzeigenkonfiguration ist ein Bild mit Bergen. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Bildes in Pixeln,
- die Position des Bildes in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Statusindikators),
- der Name des Bildes,
- Bild: „Datei hochladen“

festgelegt werden.

Inspektor für Bilder

Wenn man eine Bild-Datei „hochläd“, wird das Bild in das Base64 Format umgewandelt und direkt angezeigt. Wenn man die Größe mit der Maus ändert, bleibt das Seitenverhältnis des Bildes erhalten. Es kann sein, dass dadurch ein freier Bereich unter dem Bild entsteht.

Die unterstützten Formate sind:
*.png, *.jpg, *.jpeg, *.jfif, *.pjpeg und *.pjp

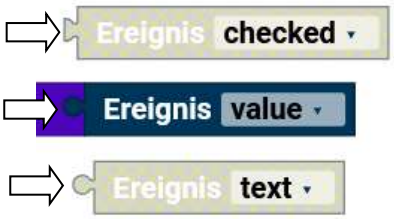
Nachdem die Datei hochgeladen ist, ändert sich der Button in „Löschen“. Damit kann man das hochgeladene Bild löschen und ein anderes Bild hochladen.

Hinweis: Manchmal dauert die Übertragung sehr lange (30 Sekunden).

Ereignis

Der Block **Ereignis []** ruft den Rückgabewert eines Elements ab. Dieser Block kann nur in den Ereignisprogrammen genutzt werden. In diesen Ereignisprogrammen bezieht sich der Block automatisch auf das Ereignis in dessen Programm er verwendet wird. Der geeignete Typ für den Rückgabewert, kann über das Dropdown-Menü (kleines Dreieck) gewählt werden:

Checked = Checkbox (Kontrollbox)
Value = Wert (Zahlenwert)
Text = Buchstaben, Text (Wert)



Erst ändern, dann andocken.

Erst wenn man zuvor beim Block Ereignis, das „checked“ (Logik) durch „value“ oder „text“ (Wert) ändert, kann man den Block andocken. **Es ändert sich vorne der Zapfen**!

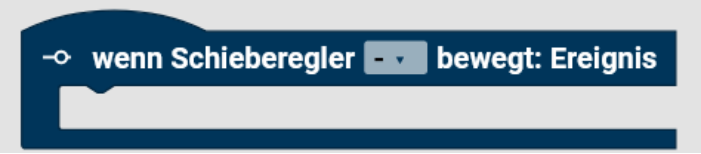
Ereignisprogramme

Diese Unterprogramme werden parallel zum Hauptprogramm ausgeführt. Werte von eingesetzten Variablen sind Global und werden vom Hauptprogramm mit dem Unterprogramm automatisch ausgetauscht.

wenn Schaltfläche angeklickt: Ereignis

Wenn die ausgewählte Schaltfläche angeklickt wird, wird dieses Unterprogramm ausgeführt.

wenn Schieberegler bewegt: Ereignis



Wenn der ausgewählte Schieberegler bewegt wird, wird dieses Unterprogramm ausgeführt.

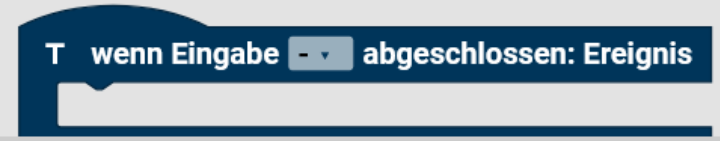
wenn Schalter umgeschaltet: Ereignis

Wenn der ausgewählte Schalter umgeschaltet wird, wird dieses Unterprogramm ausgeführt.

wenn Kontrollkästchen umgeschaltet: Ereignis

Wenn das ausgewählte Kontrollkästchen umgeschaltet wird, wird dieses Unterprogramm ausgeführt.

wenn Eingabe abgeschossen: Ereignis



Wenn die Eingabe am TXT abgeschlossen ist, wird dieses Unterprogramm ausgeführt.

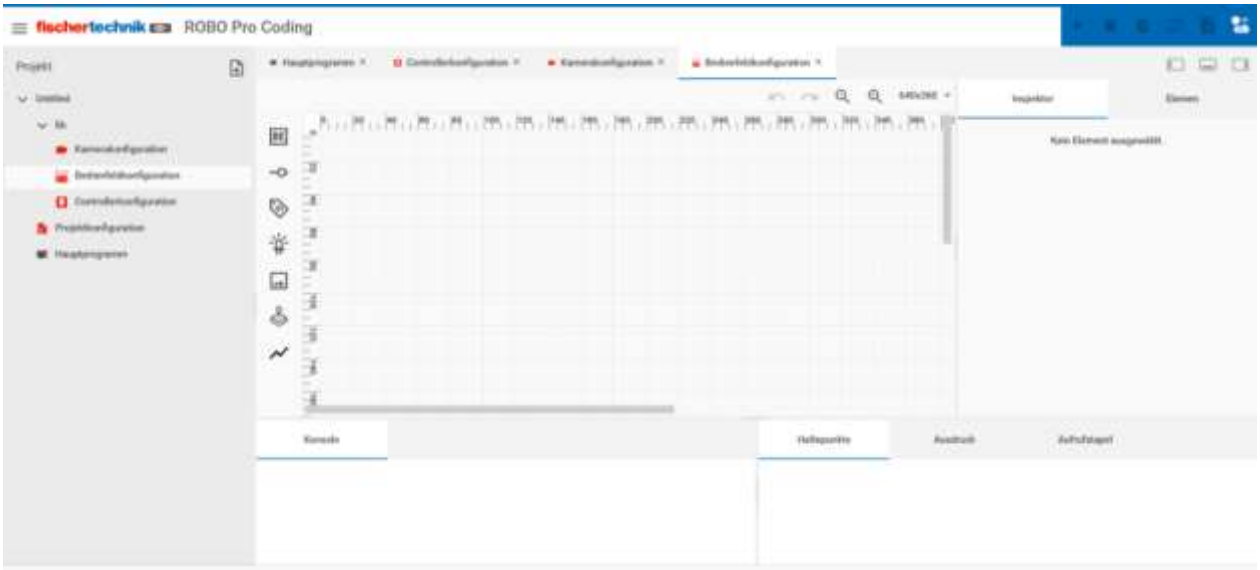
Neue Datei / Bedienfeld – Fernbedienung auf dem PC/Handy-Bildschirm
Die Blöcke der Fernbedienung sind sehr ähnlich denen der Anzeige und der TXT-Anzeige (Aktoren).
Es ist eine Virtuelle Schalttafel, mit der Modelle per Maus vom PC/Notebook oder Table gesteuert werden können. Auf ihr kann man Anzeigen und Bedienelemente zusammenstellen.

RC = Remote Controll = Fernbedienung
Der vorgeschlagene Name, für die Regler und Anzeigen fängt mit „remote“ an. Z.B.: remote_button

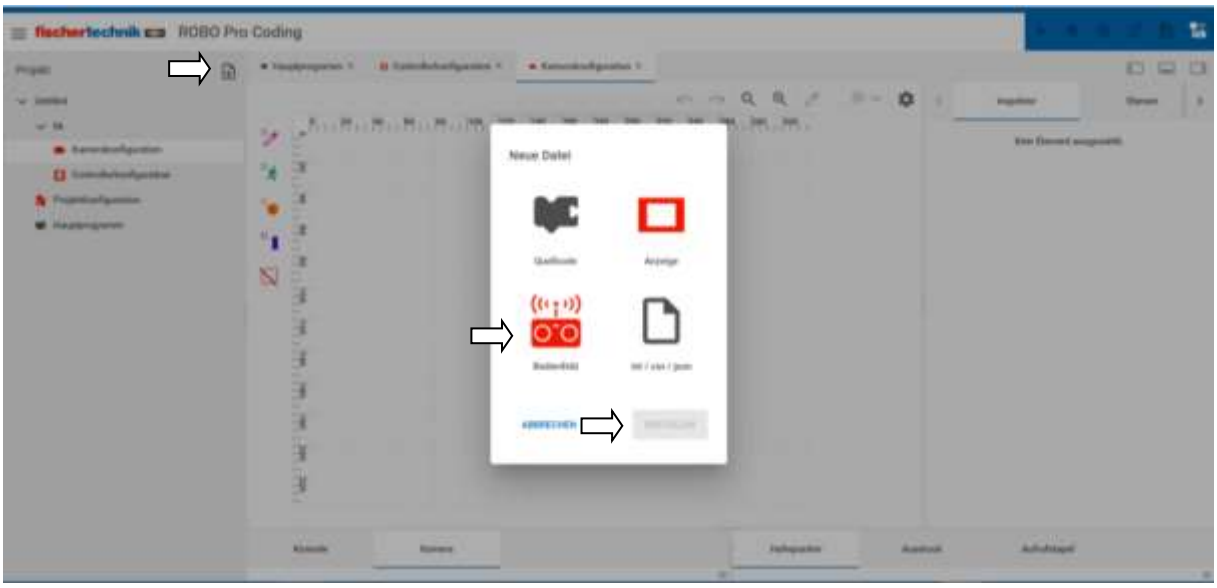
Zwei ausführliche Beispiele zur „BT Smart Fernbedienung 1“ / Bedienfeld befinden sich am Ende des Buches. Das Beispiel kann man auch für andere Controller nutzen.
Wenn man die Fernbedienung zu ersten Mal öffnet, sind die Blöcke ausgegraut und ein zusätzlicher Button erscheint:

„Bedienfeldkonfiguration hinzufügen“

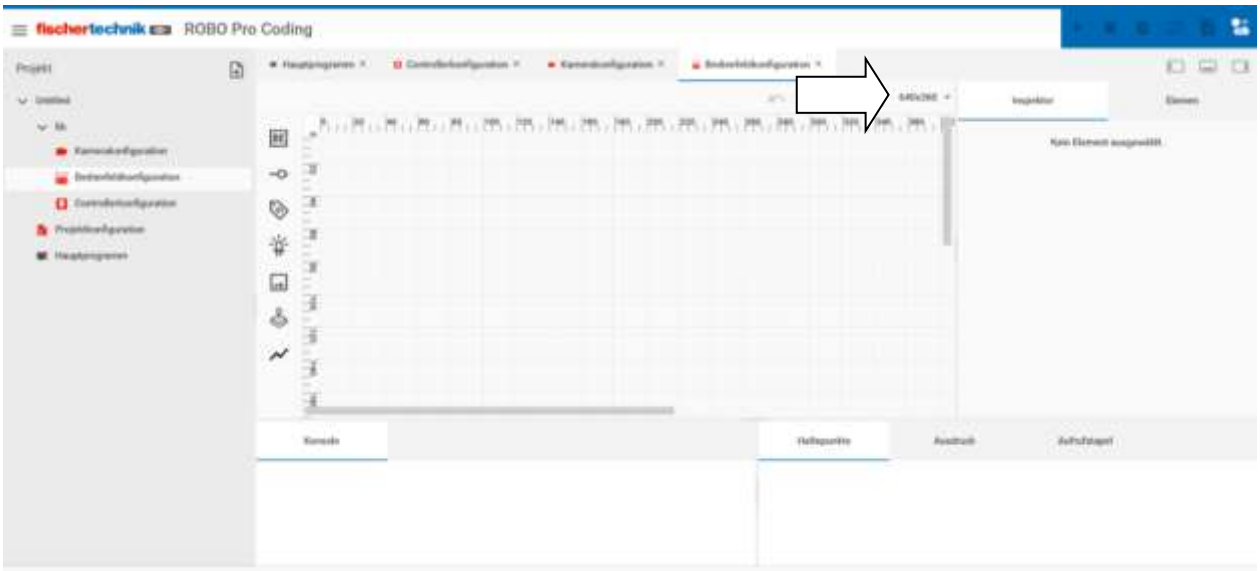
Wenn man auf den Button klickt,



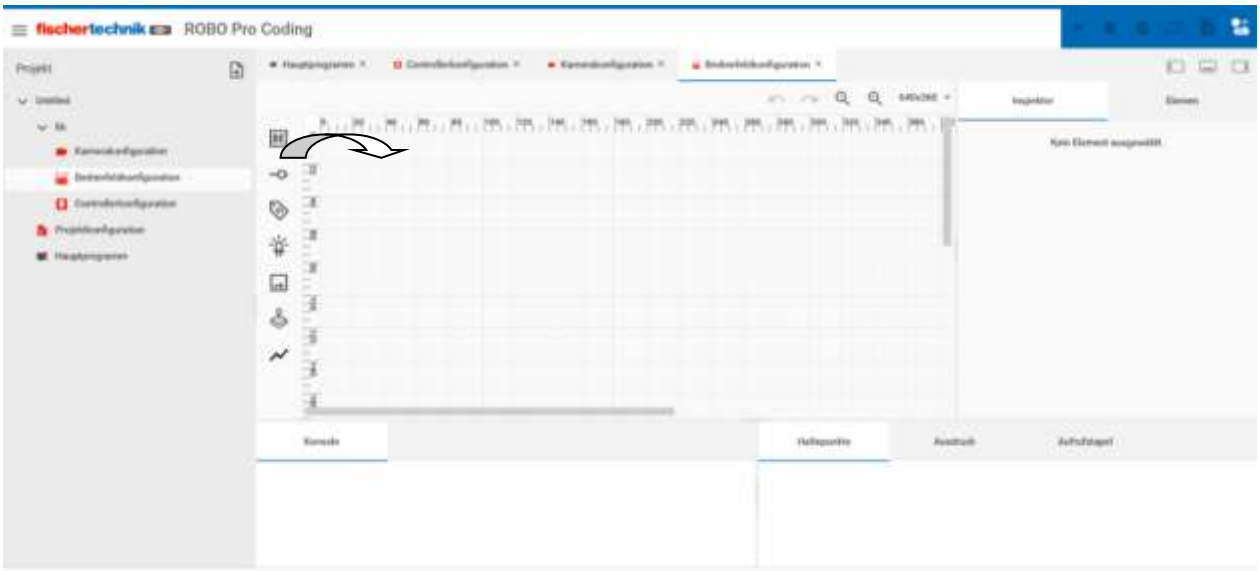
erscheint das leere Bedienfeld. Das Bedienfeld, kann auch über das Plus (Neue Datei) neben „Projekt“ erstellt werden.



Nun auf Bedienfeld und auf das dann blaue Erstellen klicken.



In der Mitte, oben rechts kann man die Größe des Bedienfeldes auswählen.



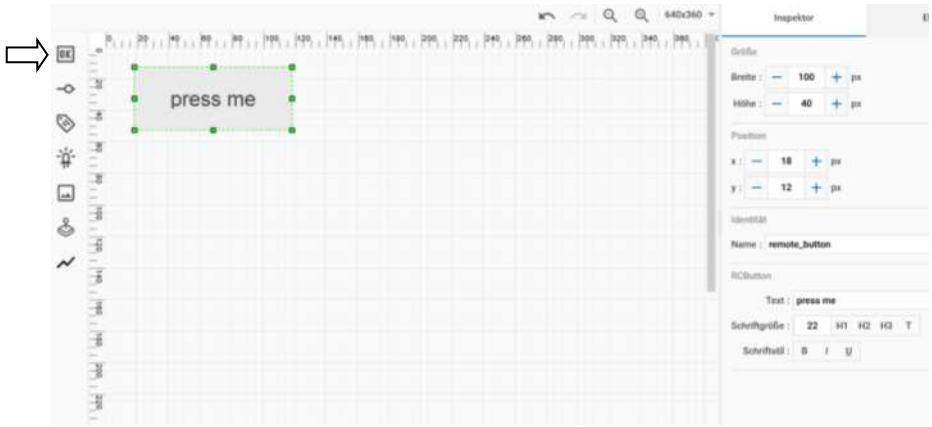
In der Mitte links, kann man die Elemente greifen und rüber ziehen.

Zur Verfügung stehen:

RCButton = Taster mit OK
RCSlider = Schieberegler
RCLabel = Beschriftung
RCIndicator = Anzeige („LED“)
RCImage = Bild
RCJoystick = virtueller Joystick
RCLineChat = Graphenanzeige

Fernsteuerung Taster

RCButton



Die Schaltfläche ist ein beschriftetes Feld, das gedrückt werden kann (Taster). Drückt man die Schaltfläche, läuft das Schaltflächen-Programm ab, sobald sie wieder losgelassen wird. Das zugehörige Symbol für die Schaltfläche ist das Quadrat mit der "OK" Beschriftung. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe der Schaltfläche in Pixeln,
- die Position der Schaltfläche in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Knopfes),
- der Name der Schaltfläche,
- den Text, der auf der Schaltfläche steht

festgelegt werden.

Der dazugehörige Block:
wenn Schaltfläche angeklickt

Über den Block „wenn Schaltfläche angeklickt“, kann man ein „Unterprogramm“ aufrufen.

Dieses Programm läuft parallel/unabhängig zum Hauptprogramm. Das Programm wird abgearbeitet, wenn man eine Taste drückt.

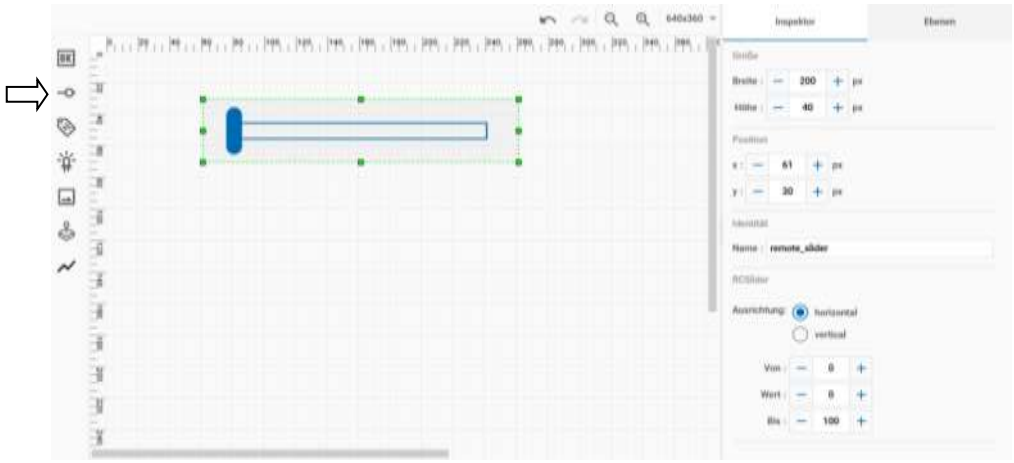
Hinweis: Variablen, die hier genutzt werden, können so auch im Hauptprogramm genutzt werden.

Achtung! Der Block wird **-2- Mal** aufgerufen. Beim **Drücken** der Schaltfläche **und** beim **Loslassen**.

Durch den Ereignis-Block wird das wirkliche drücken des Tasters abgefragt. Bei diesem Beispiel wird jeweils der Zustand auf der Konsole ausgegeben. Man kann das Programm auch um eine Variable erweitern, um so den Zustand zu speichern und eine Art Schalter daraus zu machen.

Fernsteuerung Schieberegler

RCSlider



Der Schieberegler gibt Werte abhängig von seiner Position zurück. Die Position kann dabei über den Touchscreen/Maus verändert werden. Der Wert kann über den Block abgerufen werden, sobald der Schieberegler ruht. Der abgerufene Wert ist eine Dezimalzahl. Will man den Wert des Schiebereglers ganzzahlig haben, muss man den Runde-Block  einsetzen. Das zugehörige Symbol für den Schieberegler ist der Strich mit dem Kreis. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

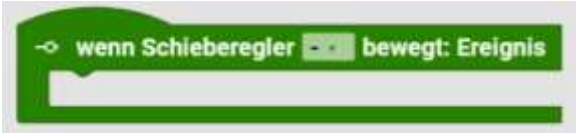
- die Größe des Schiebereglers in Pixeln,
- die Position des Schiebereglers in Pixeln (auf dem angegebenen Punkt liegt dann die obere linke Ecke des Schiebereglers)
- der Name des Schiebereglers,
- die Ausrichtung des Schiebereglers,
- den Wertebereich der über den Schieberegler abgedeckt wird und
- der Wert, auf dem der Regler bei Start des Displays steht

festgelegt werden.

Hinweis: Der Runde-Block ist in der Dokumentation zu Robo Pro Coding. Vermutlich war das in älteren Versionen von Robo Pro Coding nötig. Wenn man sich den Python Code anschaut, sieht man, dass dort ein „Int“ für Integer steht und die gelieferten Zahlen ganzzahlig sind.

Der dazugehörige Block:

wenn Schieberegler…bewegt: Ereignis

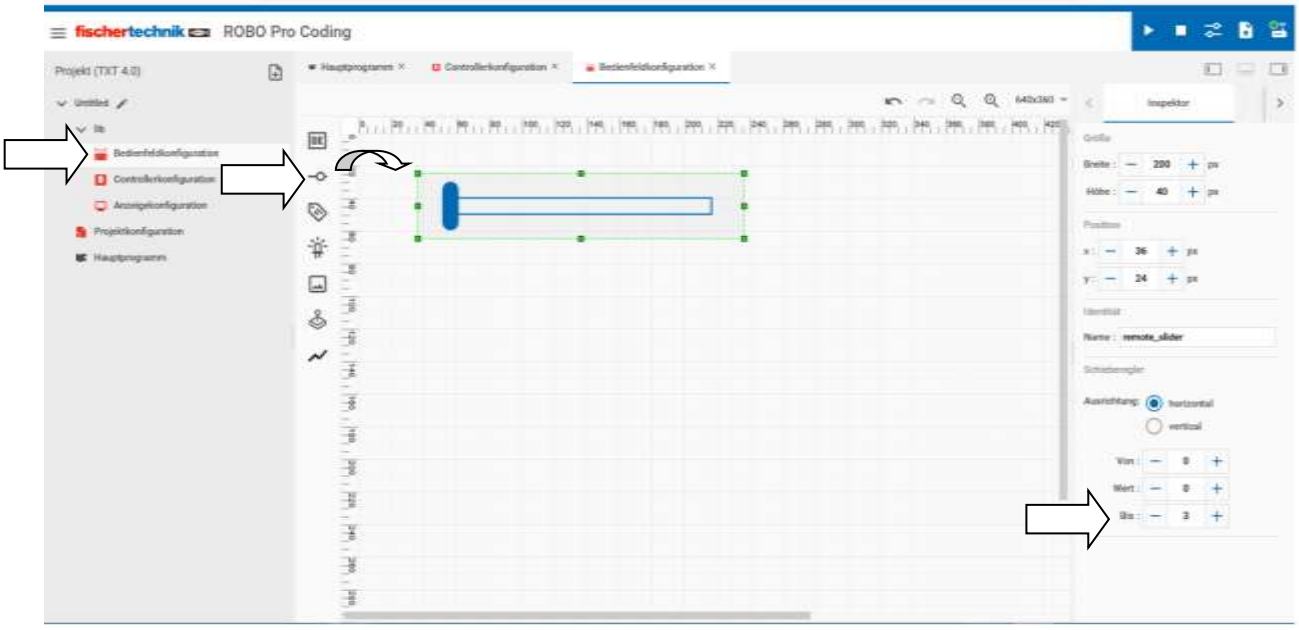


Über den Block kann man ein Unterprogramm aufrufen.

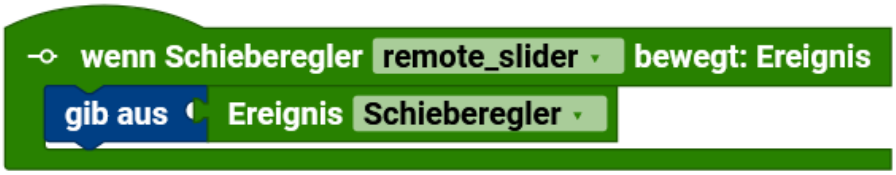
Dieses Programm läuft parallel/unabhängig zum Hauptprogramm. Das Programm wird abgearbeitet, wenn man Schieberegler bewegt.

Hinweis: Variablen, die hier genutzt werden, können so auch im Hauptprogramm genutzt werden.

Beispiel



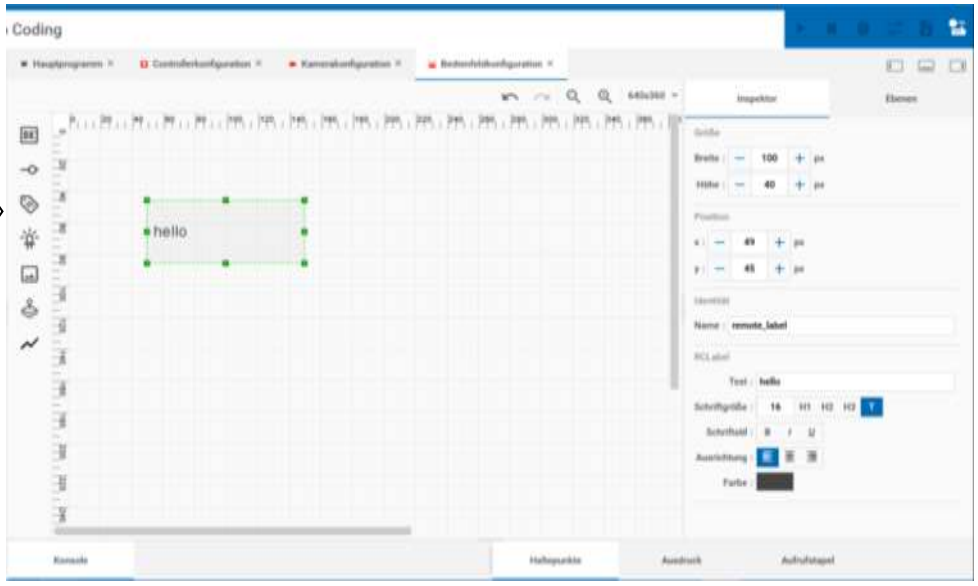
Im Bedienfeld einen Slider rüberziehen. Hier ist dann noch rechts, der „Bis-Wert“ auf 3 gestellt. Somit kann man die Zahlen 0 bis 3 mit dem Slider einstellen.



Das Hauptprogramm ist ein leeres Endlosprogramm. Das Unter-/Parallelprogramm reagiert auf die Bewegung des Schiebereglers. Der Wert wird über den Ereignis-Block geholt und auf der Konsole angezeigt.

Fernsteuerung Beschriftung

RCLabel



Mit dem Element Beschriftungsfeld kann man einen Text auf dem Bildschirm platzieren. Das Symbol in der Anzeigekonfiguration ist das Etikett. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

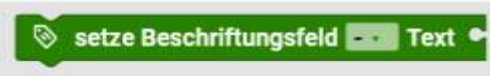
- die Größe des Beschriftungsfelds in Pixeln,
- die Position des Beschriftungsfelds in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Textfeldes),
- der Name des Beschriftungsfelds und
- der Inhalt des Beschriftungsfelds (dieser Text wird bei Start angezeigt)

festgelegt werden.

Wenn ein Text zu lang ist, wird er abgeschnitten. Man kann auch keine zwei Zeilen formatieren.

Der dazugehörige Block:

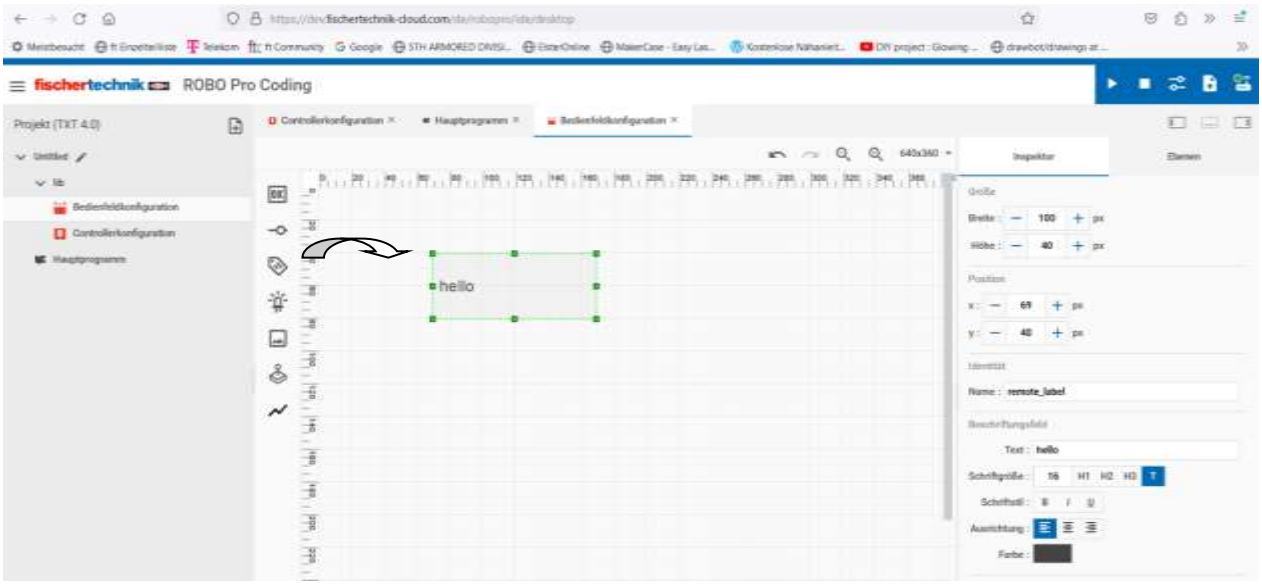
setze Beschriftungsfeld … Text



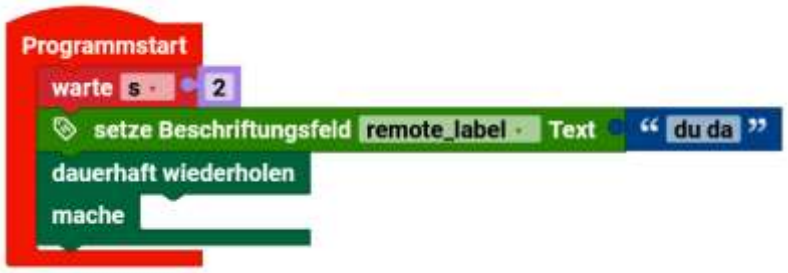
Für das Beschriftungsfeld einen Text ausgeben. Der Text selbst, kann ein fester Text sein (aus dem Reiter Text) oder aus einer Variablen, die einen Text enthält.

Beispiel

Als erstes muss ein Bedienfeld der Fernsteuerung erzeugt werden. Das kann man über den Reiter Kommunikation/Fernbedienung und dann Bedienfeld erzeugen machen. Oder man macht es über (Pulszeichen) Neue Datei/Bedienfeld.



Hier ist ein Beschriftungsfeld rüber gezogen worden und die Standartwerte mit dem Text „Hallo“ sind gleichgeblieben.



Das Hauptprogramm startet und es wird ein Bedienfeld angezeigt. Der ausgegebene Text ist „Hallo“

Bedienfeld mit „Hallo“

Nach 2 Sekunden erscheint dann „du da“.

Die leere Endlos muss in diesem Fall sein, da sonst das Programm endet und das Bedienfeld verschwindet. Man könnte sonst den Text nicht lesen.

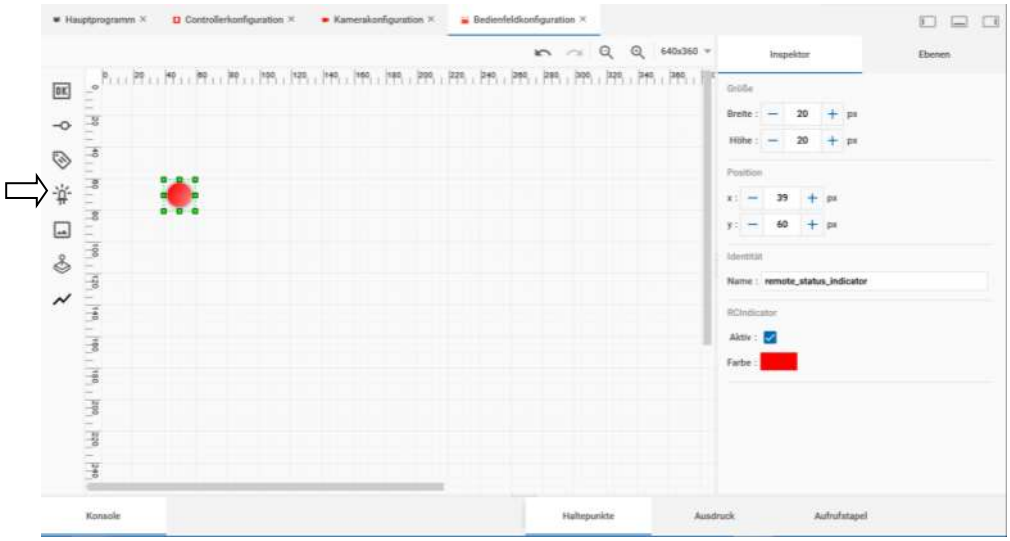
Das Programm wird über das Anklicken des X oben rechts im Bedienfeld beendet.

Man kann in den Text keine weiteren (HTML) Formatierungen wie
 einbringen. Sie werden vom Programm ignoriert. Somit ist es z.B. nicht möglich zwei Zeilen zu machen.

Man muss selber dafür sorgen, dass der Platz auch ausreichend ist, um den Text darstellen zu können. Notfalls die Schriftgröße ändern.

Fernsteuerung Statusindikator /Statusanzeige

RCIndicator



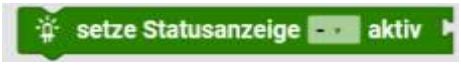
Der Statusindikator zeigt die Aktivität von etwas an. Je nach Status leuchtet er ("aktiv") oder tut dies nicht ("inaktiv"). Das Symbol in der Anzeigenkonfiguration ist eine leuchtende Diode. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe der Statusanzeige in Pixeln,
- die Position der Statusanzeige in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Statusindikators),
- der Name der Statusanzeige,
- die Farbe der Statusanzeige,
- ob die Statusanzeige zu Beginn aktiv oder inaktiv sein soll und
- die Farbe

festgelegt werden.

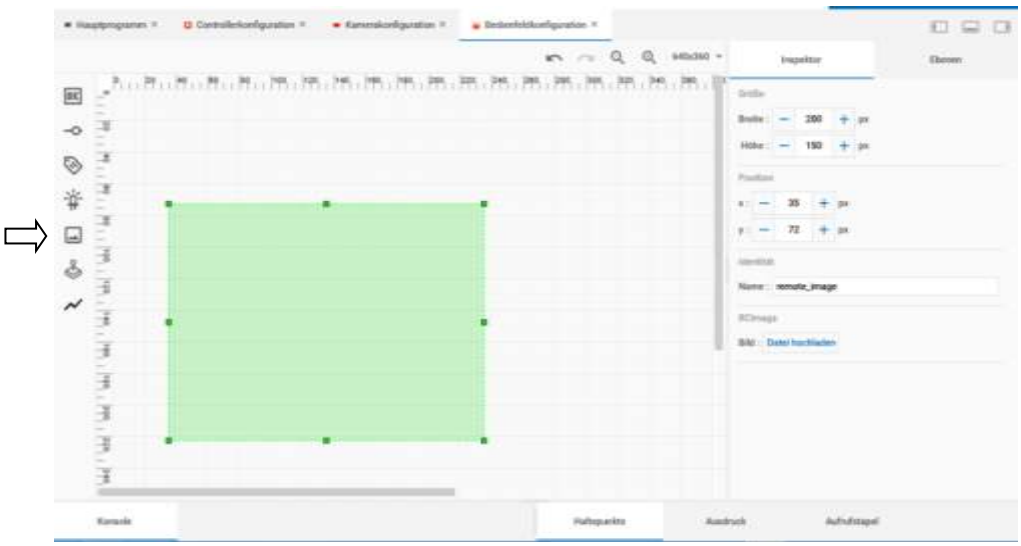
Der dazugehörige Block:

setze Statusanzeige aktiv



Fernsteuerung-Bild

RCImage



Für ein Bild im Display ein Base64-Bild festlegen. Base64 ist ein Datenformat, wo die Binärzahlen des Bildes, in ASCII-Code umgewandelt werden.

Das Symbol in der Anzeigenkonfiguration ist ein Bild mit Bergen. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des Bildes in Pixeln,
- die Position des Bildes in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Statusindikators),
- der Name des Bildes,
- Bild: „Datei hochladen“

festgelegt werden.

Wenn man eine Bild-Datei „hochläd“, wird das Bild in das Base64 Format umgewandelt und direkt angezeigt. Wenn man die Größe mit der Maus ändert, bleibt das Seitenverhältnis des Bildes erhalten. Es kann sein, dass dadurch ein freier Bereich unter dem Bild entsteht.

Die unterstützten Formate sind:
*.png, *.jpg, *.jpeg, *.jfif, *.pjpeg und *.pjp

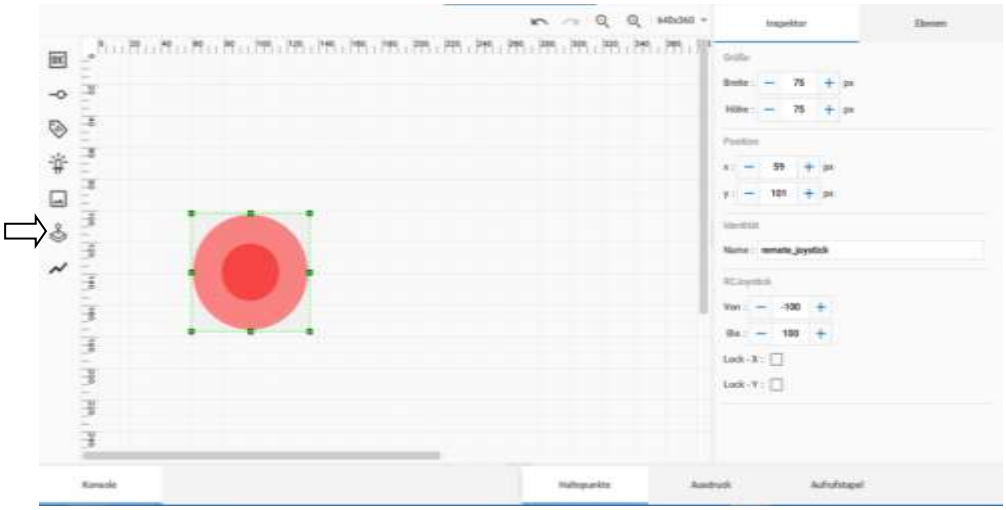
Nachdem die Datei hochgeladen ist, ändert sich der Button in „Löschen“. Damit kann man das hochgeladene Bild löschen und ein anderes Bild hochladen.

Der dazugehörige Block:

setze Bild… Base64-Bild

Fernsteuerung Joystick

RCJoystick



Der RCJoystick dient zur Steuerung. In der Standarteinstellung gibt er Werte zwischen -100 und +100. Das Symbol in der Anzeigenkonfiguration ist ein Joystick. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des RCJoystick in Pixeln,
- die Position des RCJoystick in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Statusindikators),
- der Name des RCJoystick,
- Lock X des RCJoystick und
- Lock Y des RCJoystick

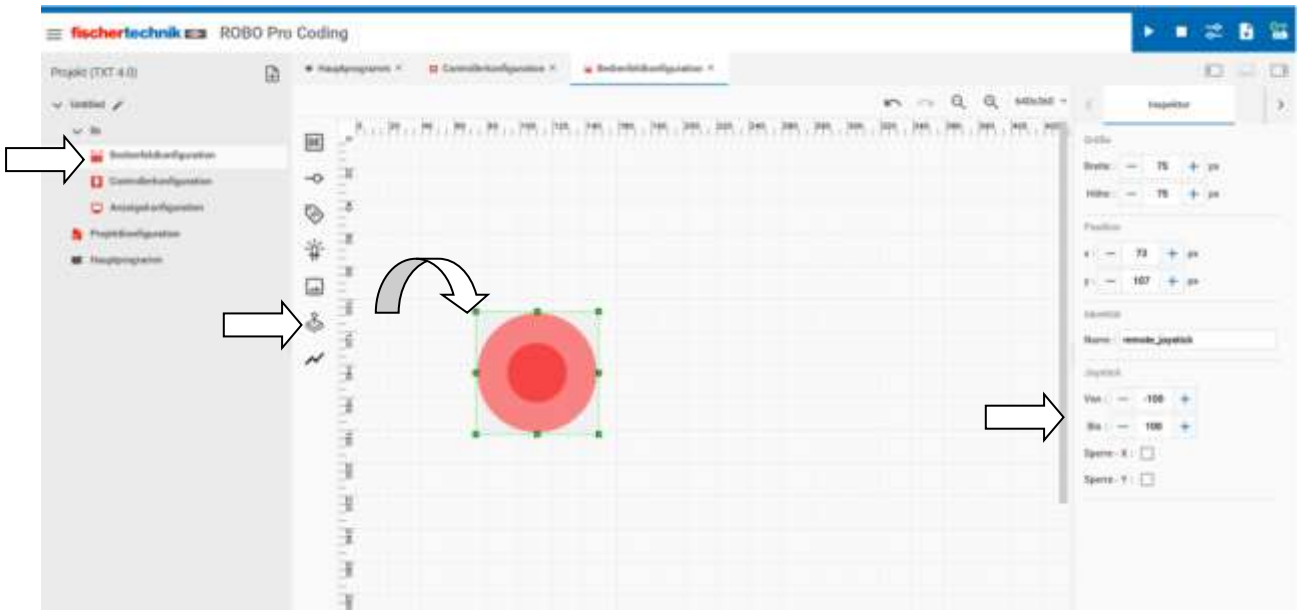
festgelegt werden.

Mit Lock X und Y, kann man die Bewegungsrichtug des Joysticks sperren. Wenn man z.B. Lock Y im Kästchen aktiviert, kann man ihn nur noch in der X-Achse bewegen. Wenn man Lock X aktiviert, kann man ihn nur in Y-Richtung bewegen. Damit kann man Fernsteuerungen machen, die mehrere Joysticks haben, die wie bei einer Funkfernbedienung funktionieren.
Man kann den „Lock“ aber im Programm nicht an- oder abschalten.

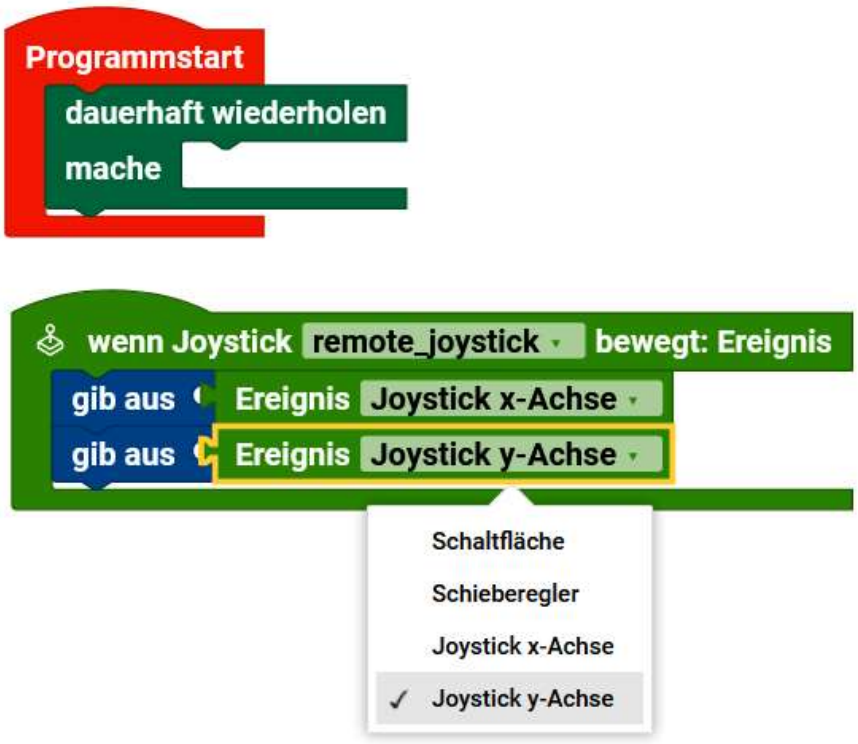
Der dazugehörige Block:

wenn Joystick…bewegt: Ereignis

Beispiel



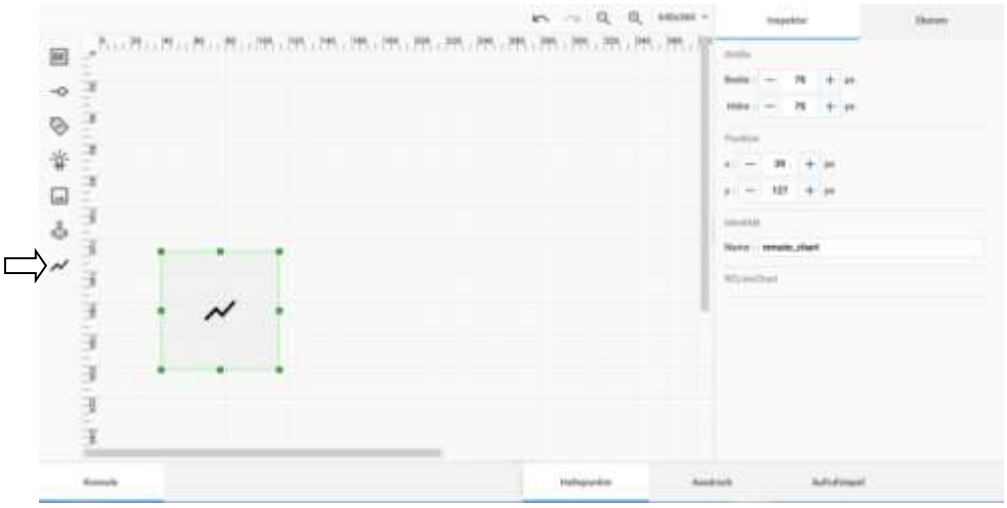
Im Bedienfeld wird der Joystick rüber gezogen. In der Standarteinstellung sind die Werte von -100 bis+100.



Hier ist ein leeres Hauptprogramm. Wenn der Joystick bewegt wird, werden die Werte auf der Konsole ausgegeben. Die Werte werden über den Ereignis-Block übertragen. Man muss auswählen welches Ereignis bzw. welche Richtung genommen werden soll. Hier ist es erst die X-Achse, dann die Y-Achse.

Fernsteuerung Diagramm

RCLineChat



Der RCLineChat dient zur Darstellung von Graphen. Das Symbol in der Anzeigenkonfiguration ist ein Graph. Zieht man dieses Symbol in den gerasterten Bereich, öffnet sich rechts ein Fenster. Hier kann unter Inspektor

- die Größe des RCLineChat in Pixeln und
- die Position des RCLineChat in Pixeln (auf dem angegebenen Punkt liegt die obere linke Ecke des Statusindikators)

festgelegt werden.

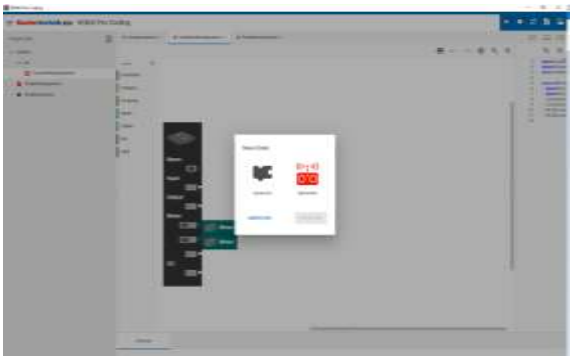
Der dazugehörige Block:

setze Diagramm x;y

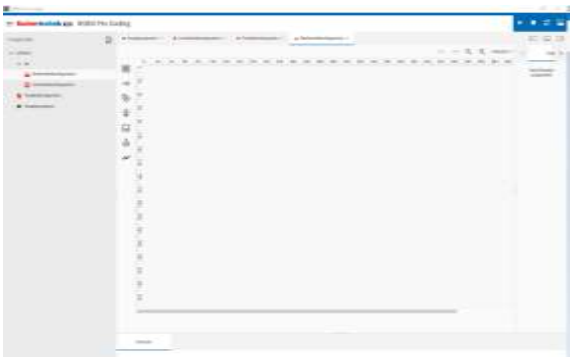
Achtung!

Die Darstellung von Diagrammen, ist nur für „einfache“ Sachen sinnvoll. Insgesamt fehlt dem Block so etwas wie Skalierung. Dadurch kann es sein, dass die Ausgabe, die man erwartet, nicht dem entspricht, wie es denn wirklich angezeigt wird. Manchmal kann man etwas tricksen, in dem man den Zahlenraum erweitert oder die Zahlen mit mehr Stellen darstellen lässt.

Neue Datei RX und BT Smart Controller

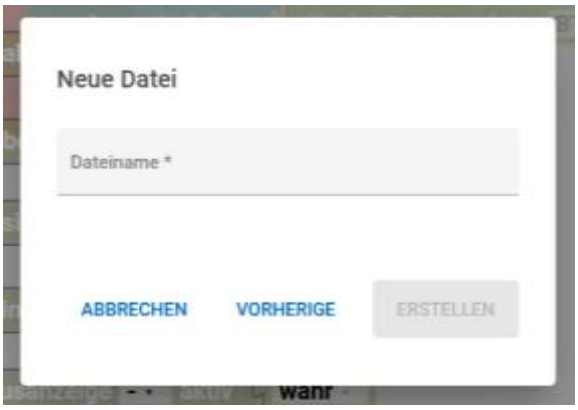


Beim RX und BT Smart Controller kann man als neue Datei entweder Quellcode oder Bedienfeld auswählen.



Hier das Bedienfeld mit der Bedienfeldkonfiguration.

Wenn man Quellcode wählt,



muss man nun der Datei einen Namen geben.

Die Datei wird erzeugt und links oben angezeigt. Die Datei wird automatisch eingefügt. Man kann vom Hauptprogramm die Unterprogramme, Programmteile oder Funktionen … direkt benutzen oder aufrufen. Man sollte aber selbstsprechende Namen sich aussuchen, wie z.B. Ausgabe und nicht nur ein a.

Programm hochladen (Blaue Kopfzeile)

Programm hochladen auf den TXT 4.0 Controller
Programm die mit Robo Pro Coding erstellt wurden, können auf den TXT 4.0 und den RX Controller hochgeladen werden. Das funktioniert wie bei einem USB Stick oder einer Speicherkarte.

Im Gegensatz zum Exportieren, wo das Projekt auf dem PC mit .ft gespeichert wird, wird das Programm als .py, also als Python Programm gespeichert. Weitere Programmteile die z.B. für das Ansprechen vom Display benötigt werden, sind schon auf dem Controller vorhanden. Es wird also nicht das Projekt mit der Controllerkonfiguration usw. gespeichert. Wenn man also das Robo Pro Coding Projekt auf dem PC verloren hat, kann man es sich nicht so einfach vom TXT 4.0 wieder runterladen und damit weiterarbeiten.

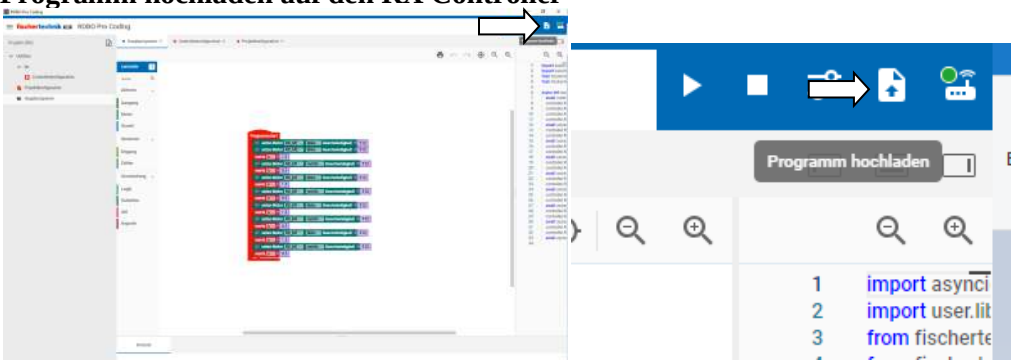
Das auf den Controller hochgeladene Programm ist bei den Python Programmen zu finden. Das ist bei dem TXT 4.0 leichter, da er sie auf dem Display anzeigt. Beim RX Controller sieht man es so nicht, da er kein Display hat.

Der TXT 4.0 kann mit mehreren Programmen umgehen. Man kann über das Display die entsprechende Datei auswählen und starten.

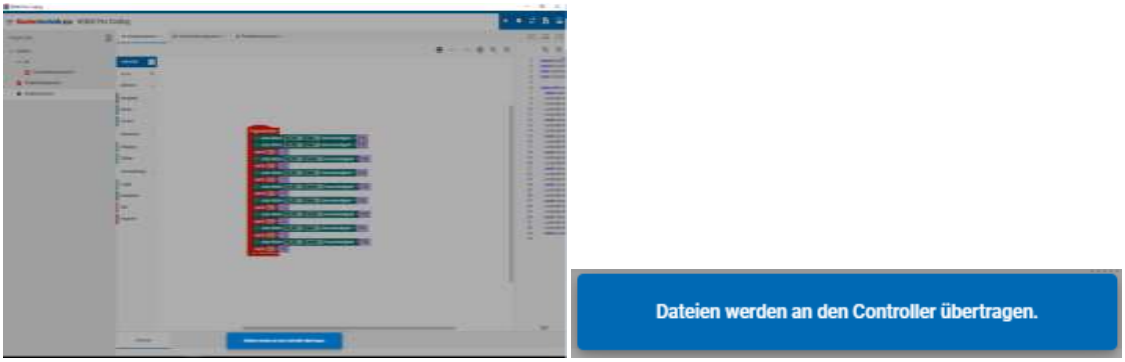
Für Experten

Man kann aber auch per SSH-Zugriff/Browser auf den RX und TXT 4.0 Controller zugreifen. Fast wie auf einen USB Stick.

Programm hochladen auf den RX Controller



Durch Drücken auf „Programm hochladen“ wird das Programm bei verbundenem RX Controller hochgeladen / übertragen.



Es kommt kurz diese Meldung.

Man kann nun das Programm, starten. Hier gibt es zwei Möglichkeiten. Man kann das Programm mit oben

links „Start“ starten. Dabei wird das Programm noch mal übertragen und gestartet.

Was ist der Unterschied?

Einmal ist das Programm im PC und wird zum RX Controller übertragen und auf dem RX Controller gestartet.

Beim Hochladen auf den RX Controller kann man das Programm über die Taste am Controller gestartet werden. Und das immer wieder und das auch ohne PC.

Im Gegensatz zum TXT 4.0 Controller kann der RX Controller nur mit einem Programm umgehen. Er hat ja auch kein Display, wo man unter mehreren Dateien auswählen könnte.

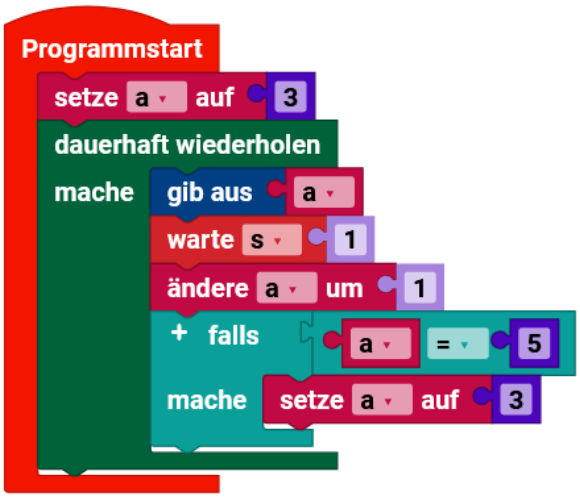
Programm hochladen auf den BT Smart Controller
Es ist nicht möglich auf einen BT Smart Controller ein Programm hochzuladen.

Der Debugger

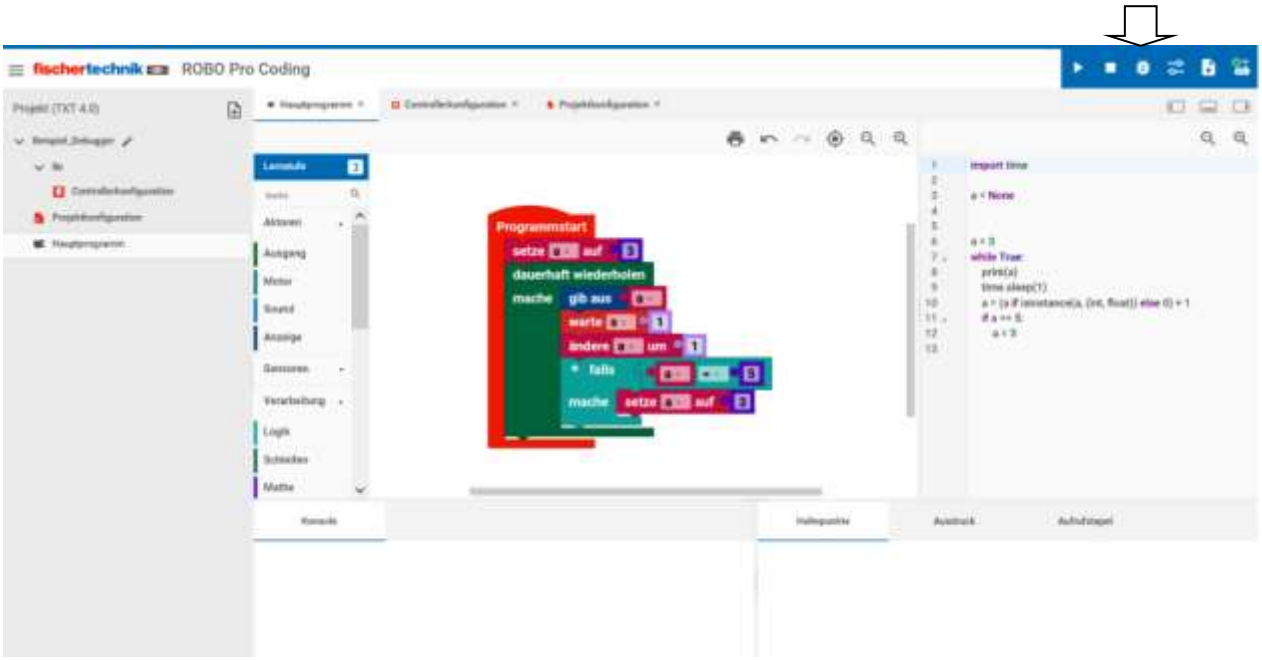
Der Debugger soll das Finden von Fehlern im Programm erleichtern. Über das Menü in der oberen rechten Ecke kann man den Code Schritt für Schritt durchgehen und so einen Programmdurchlauf simulieren. Das Programm reagiert dabei trotzdem auf Interaktion mit Sensoren, wie z.B. das Drücken eines Tasters. Während man das Programm durchläuft, kann man links im großen Feld sehen in welchem Block man sich befinden und rechts an welcher Stelle im Python-Code.

Hinweis: Der Debug-Modus ist nicht verfügbar, wenn das Programm eine Fernbedienung enthält.

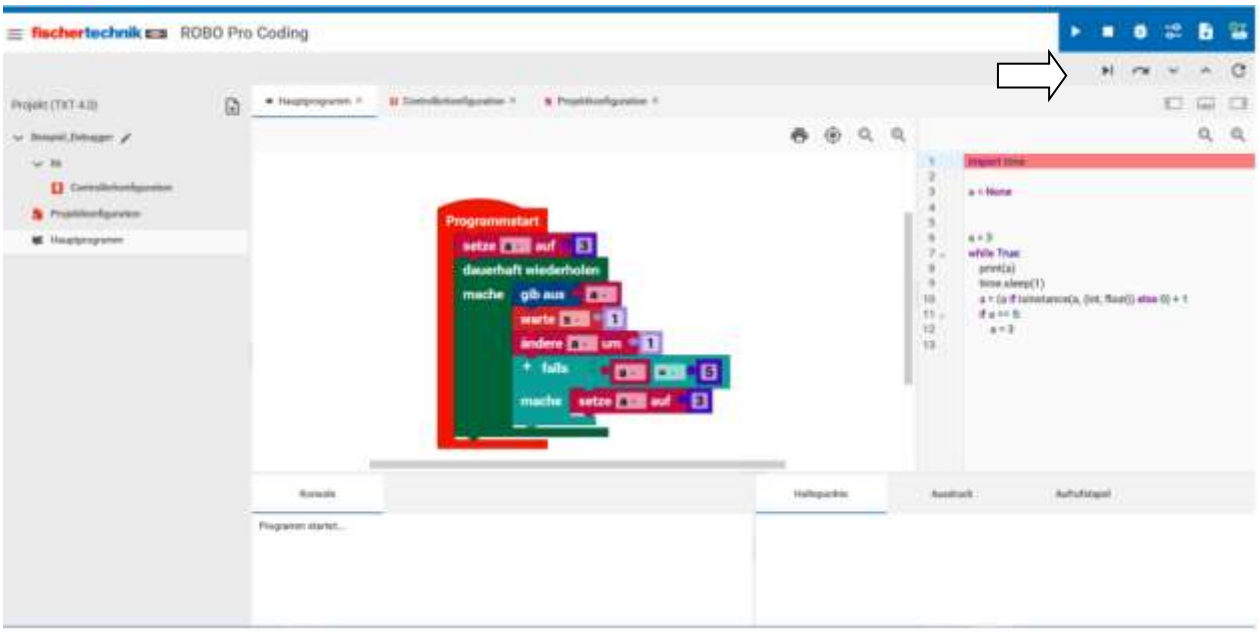
Beispiel



Hier ein Beispielprogramm, das von 3 bis 4 Zählen soll und es soll die Zahlen auf der Konsole ausgeben.

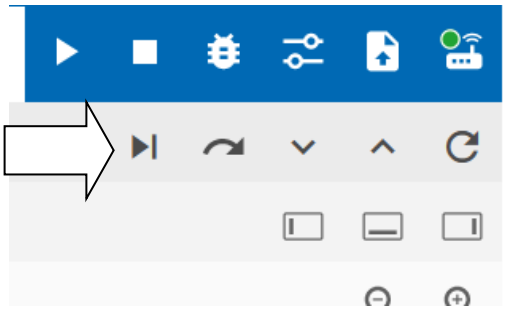


Der Debugger wird über das Symbol „Bug“ gestartet.



Es erscheint eine neue Symbolleiste oben rechts.

Hinweis: Robo Pro Coding versucht 5 Mal mit dem Controller sich zu verbinden. Wenn das nicht klappt, kommt eine Fehlermeldung. Um diese zurückzusetzen, kann man die Verbindung trennen (oben rechts) und den Controller Aus- und wieder Einschalten.



Hier sind:

Fortfahren

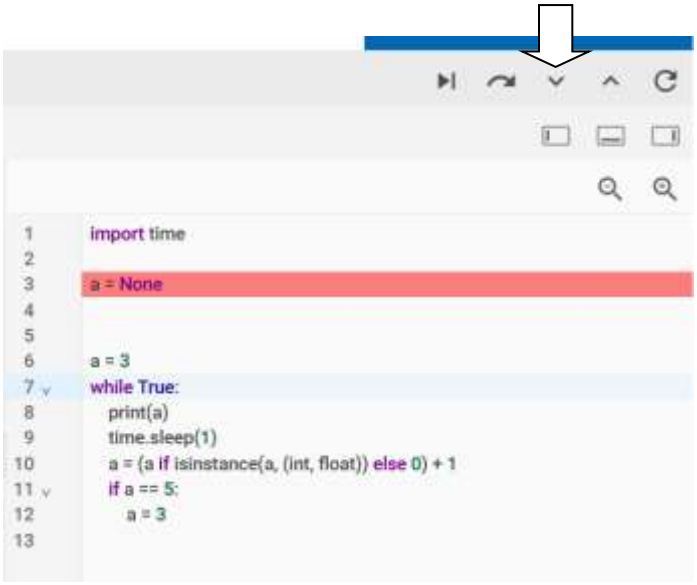
Überspringen

Einzelschritt

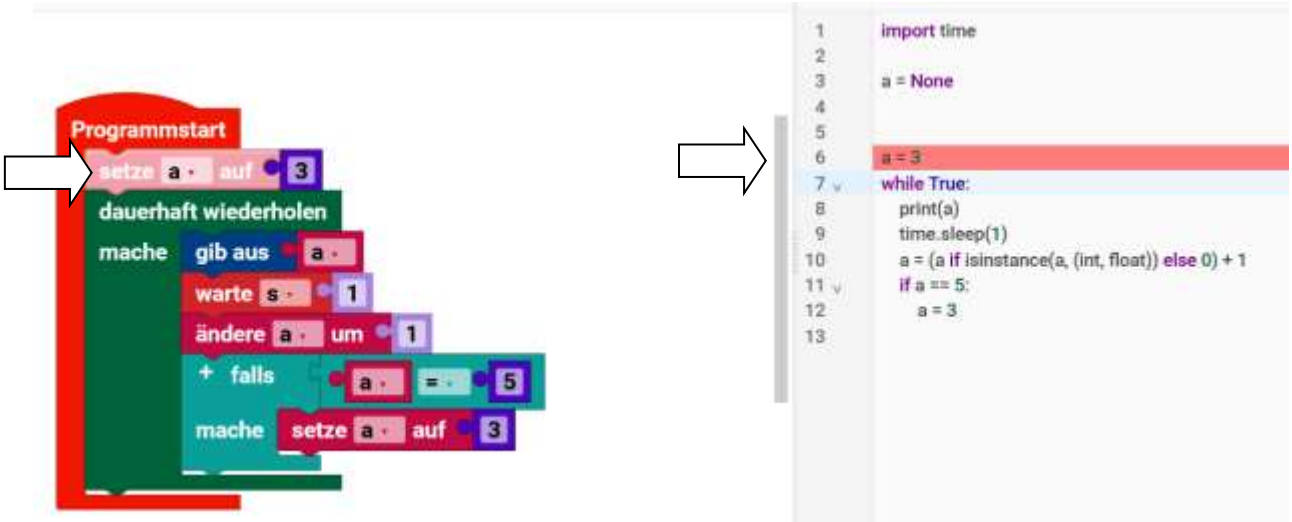
Rückschritt

Neustart

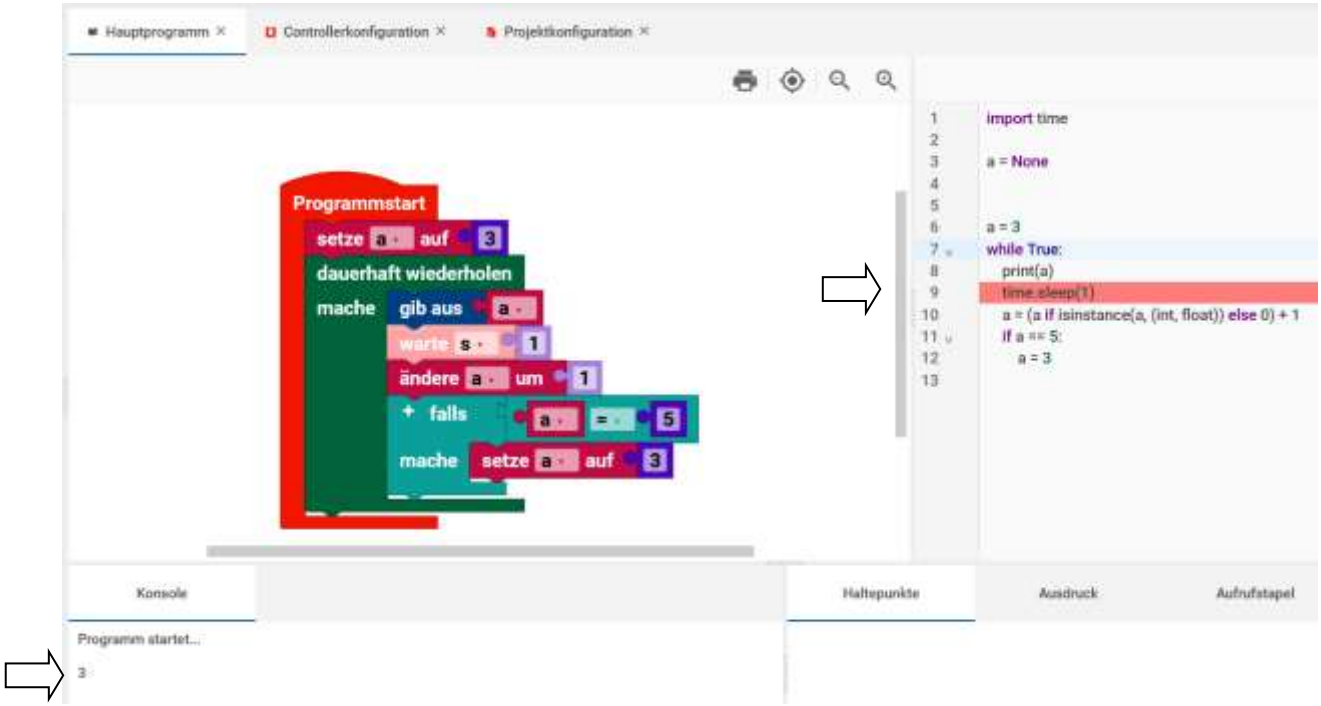
Man sieht hier eine rot hinterlegte Pythoncode-Zeile. Das ist der Programmpointer (Zeiger). Diese Zeile wird momentan noch nicht ausgeführt.



Über „Einzelschritt“ kann man diesen Pointer und somit in die nächste Programmzeile, weiterschalten. Erst jetzt ist hier die erste Programmzeile ausgeführt worden. Die dritte Zeile noch nicht.



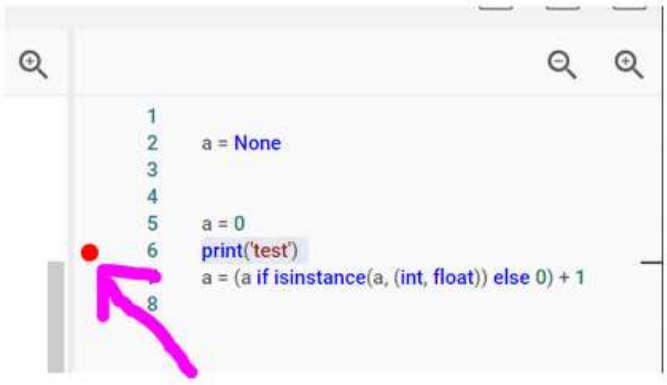
Wenn es im Pythoncode ein ganzer Block ist, wird der Block auch farbig hinterlegt.



Auch hier wird die Ausgabe auf der Konsole, erst in der nächsten Codezeile ausgeführt.

Haltepunkte

Hinweis: Momentan scheint es in der Onlineversion von Robo Pro Coding nicht zu funktionieren. In der aktuellen App-Version schon.



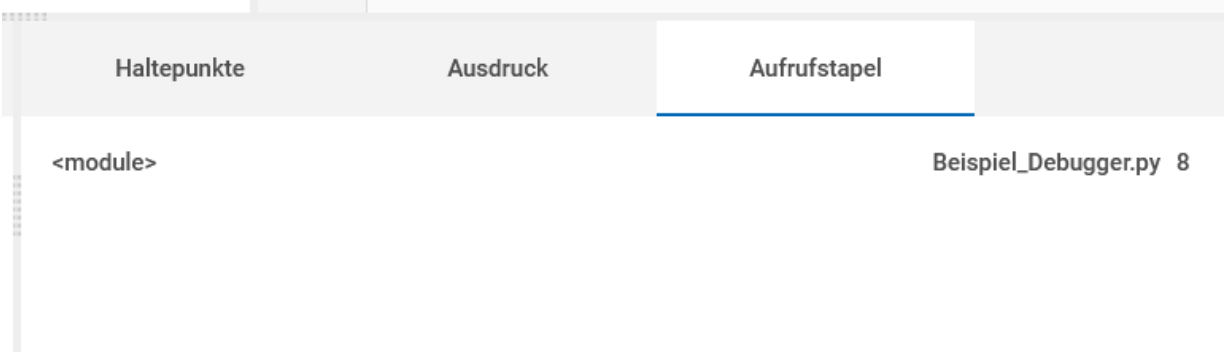
Durch einen Doppelkick am Anfang des Pythoncodes, kann man einen Haltepunkt setzen. Bei einem Programm durchlauf, im Debugger, wird in dieser Zeile das Programm angehalten.

Ausdruck

In Reiter „Ausdruck“, kann man über das Pluszeichen Variable hinzufügen, deren Wert angezeigt werden soll. Hier ist es die Variable „a“. Nach dem Eingeben und drücken der [Enter] Taste…

… erscheint die Variable a und deren Inhalt. Hier ist es a=4. Weiter Variablen können über das Pluszeichen hinzugefügt werden.

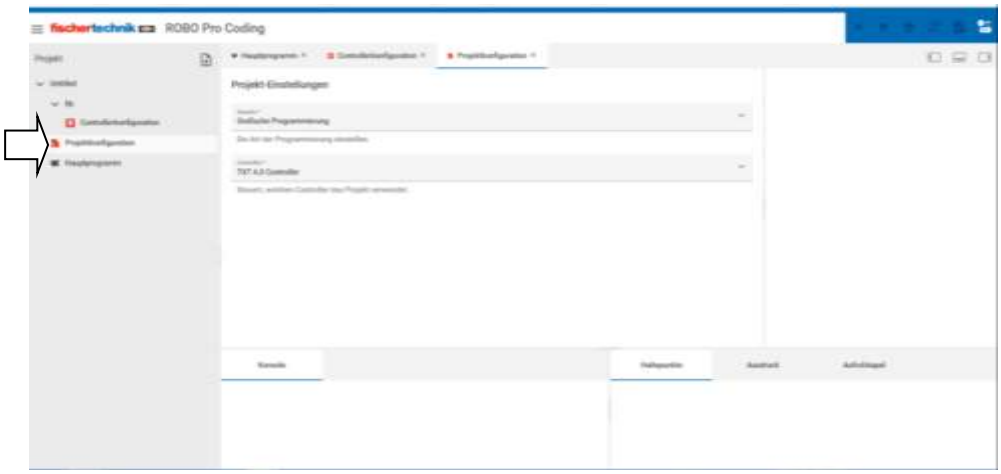
Aufrufstapel



Im Aufrufstapel wird angezeigt in welchem Programmteil wir gerade sind. Da dieses Programm nur ein Hauptprogramm hat, wird hier auch nur der Programmname angezeigt, worunter das Programm gespeichert wurde.

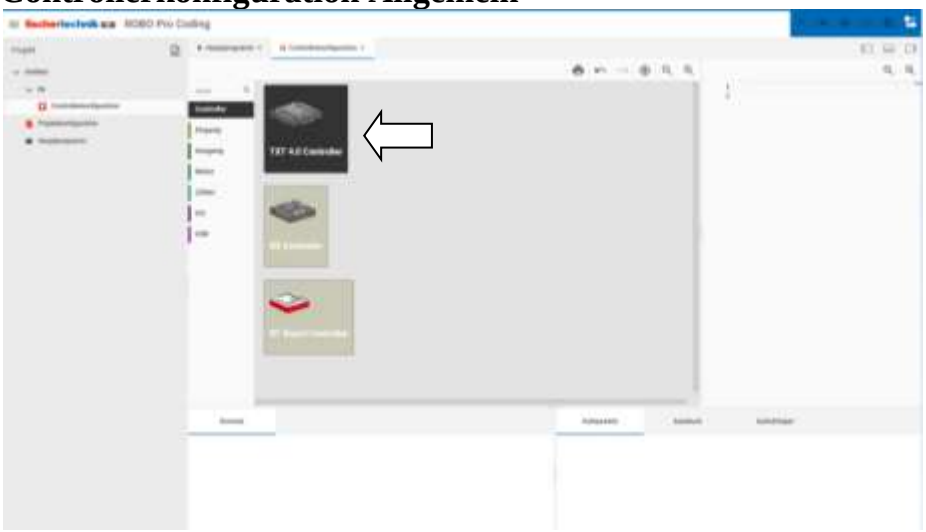
Projektkonfiguration Projekt-Einstellung

Hier kann man zwischen "Grafische Programmierung" (Blocky mit Python) und Python-Programmierung (reiner Python-Textcode) wählen. Unter Controller, kann man zwischen TXT 4.0 ,
RX Controller und BT Smart Controller sich entscheiden.



In diesem Fall braucht man nichts ändern.
Wir lassen es auf „Grafische Programmierung“ und „TXT 4.0 Controller“.

Controllerkonfiguration Allgemein

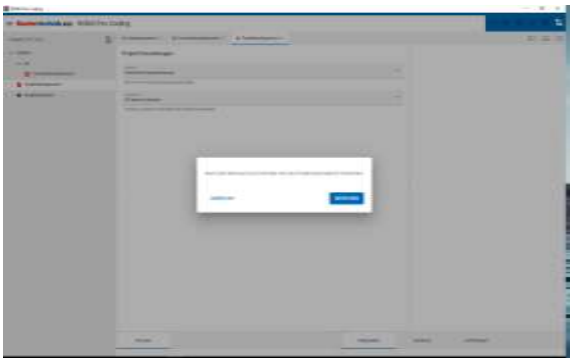


Wie hier z.B. den TXT 4.0 anwählen.
Bei der Controllerkonfiguration, kann man unter "Controller" momentan drei Controller auswählen:

TXT 4.0
RX Controller
BT Smart Controller

Hinweis:
Der TXT 4.0 nicht mit dem (Vorgänger) TXT Controller verwechseln. Den TXT kann man nicht mit Robo Pro Coding programmieren. Die beiden anderen Controller sind „ausgegraut“ und können nicht ausgewählt werden. Wir hatten ja auch vorher den TXT 4.0 ausgewählt.

Achtung beim Wechsel vom Controller im aktuellem Projekt
Beim Wechsel vom Controller z.B. vom TXT 4.0 zum RX Controller kommt folgende Meldung:



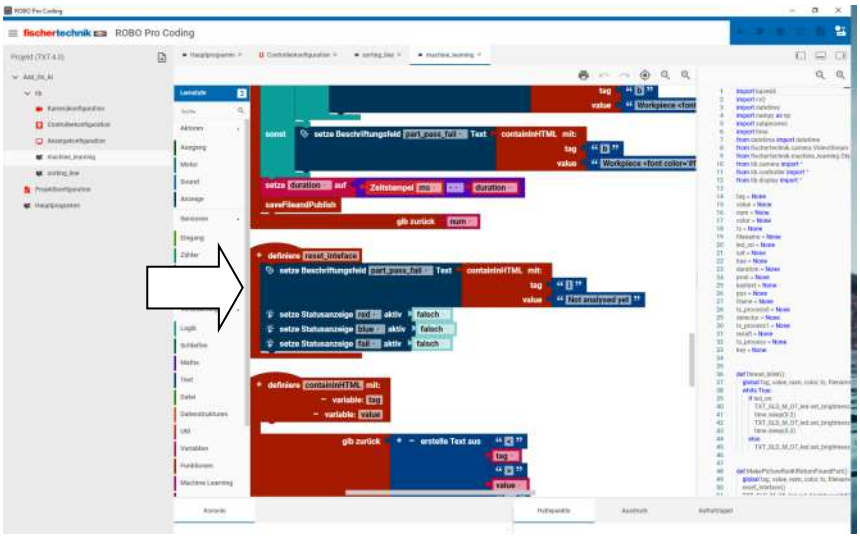
Dies ist eine Sicherheitsabfrage. In diesem Fall ist es so, dass das Projekt an die Fähigkeiten des jeweiligen Controllers angepasst wird. Das kann aber bedeuten, dass der ausgewählte Controller weniger hat.

Achtung beim Wechseln vom Controller!
Diese nicht brauchbaren Programmteile **werden gelöscht**.

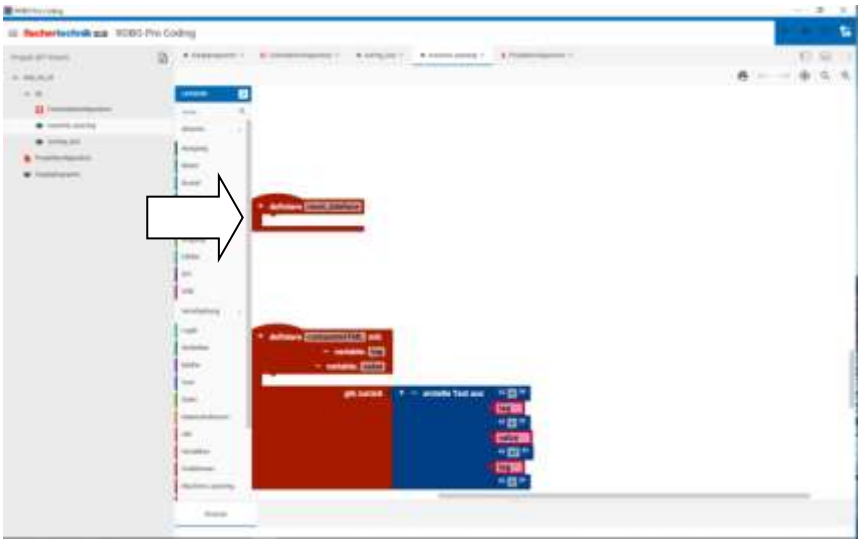
<u>Man hat also hier die letzte Möglichkeit abzubrechen und das Projekt noch zu speichern.</u>

Der TXT 4.0 kann z.B. eine Kamera ansteuern. Der RX-Controller oder der BT Controller nicht.

Beispiel: Hier das Unterprogramm "reset_interface" beim TXT 4.0



Und wenn man vom TXT 4.0 auf den BT Smart Controller wechselt:



Ein leeres Programmteil. Auch andere Teile sind leer. Das Projekt ist nicht mehr lauffähig.
Wenn man den Controller zurückwechselt, bleibt die Änderung erhalten.

Der gelöschte Teil wird nicht wieder hergestellt.

Keine automatische Änderung der Nummerierung.

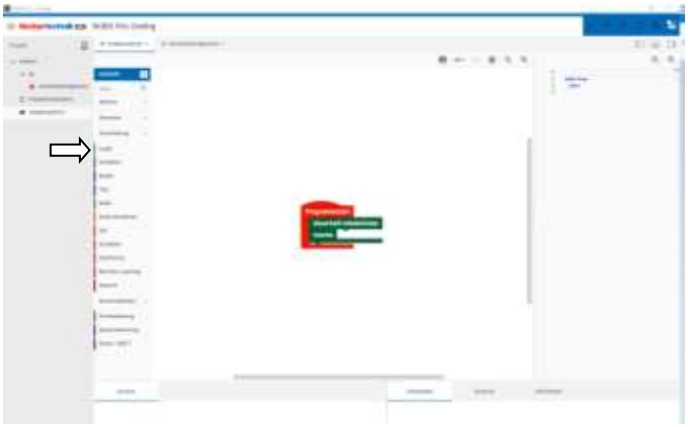
Wenn man z.B. vom RX-Controller zum BT Smart wechselt, werden die Eingänge I5-I8 nicht automatisch in I1-I4 verschoben. Das muss man per Hand machen. Es setzt voraus, dass man die Verkabelung ändern muss.

Allgemeiner Umgang mit Blöcken in Robo Pro Coding

Mausdarstellung

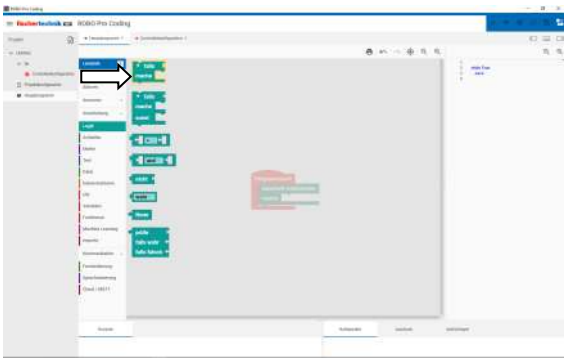
Mauszeiger kleiner Pfeil

Mauszeiger Hand = gegriffener Block



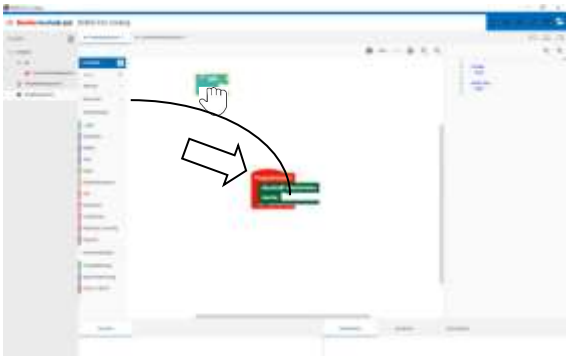
Nach dem Schnellstart. Erst „Neues Projekt“ und dann auf den Reiter „Logik“ klicken

Man sieht die Blöcke, die in dem Reiter vorhanden sind. Es gibt nun zwei Möglichkeiten einen Block davon in das Programm einzufügen.



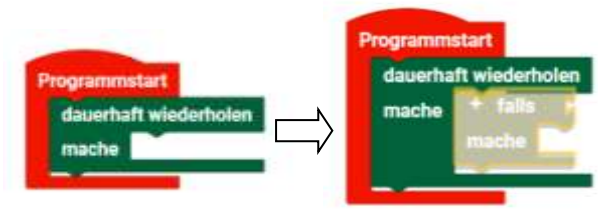
1. Einfaches Anklicken vom Block.

Der Block wird ausgegraut an der Seite im Programm abgelegt

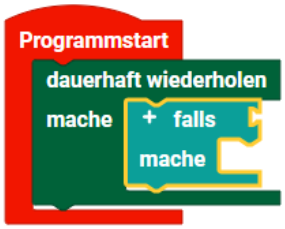


Nun den Block mit der linken Maustaste aus der Ecke rüber ziehen. Man Klickt mit der Maus auf den Block und hält ihn damit fest. Mit gedrückter Maustaste kann man ihn rüber ziehen / bewegen.

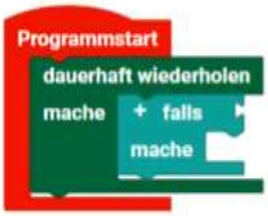
2. Man kann auch den Block, aus dem Menü, direkt rüber ziehen. Dazu den Block mit der Maus anklicken und festhalten und die Maus bewegen.



Wenn man in die Nähe des Ziels kommt, passt sich der Block an und...

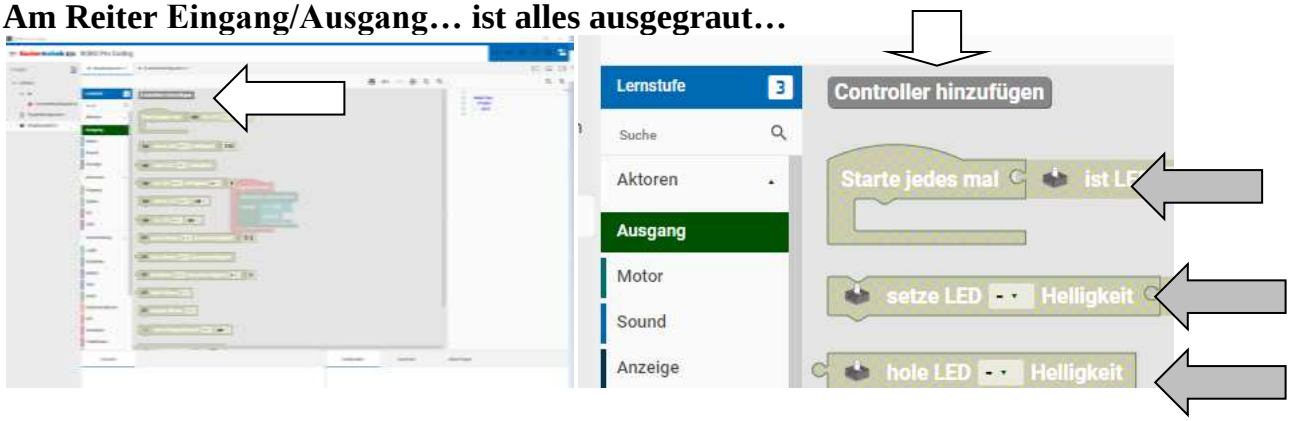


man kann ihn ablegen in dem man die Maustaste loslässt. Dieser Baustein ist mit einem kleinen gelben Rand umgeben. Er ist markiert.

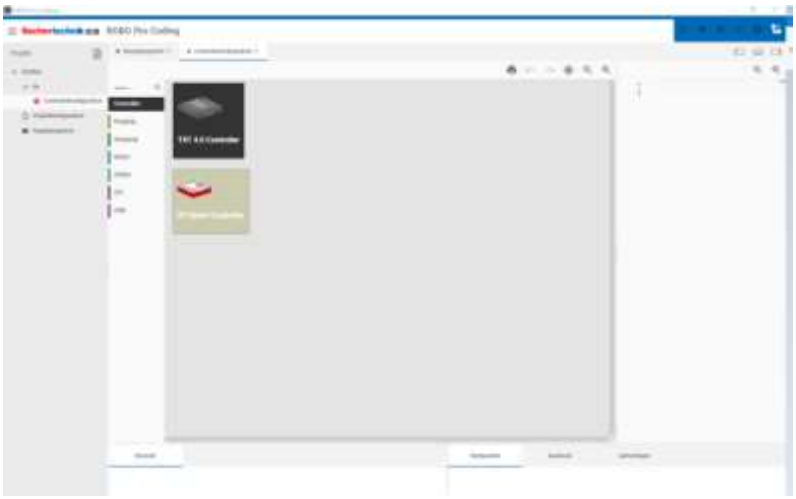


Wenn man in den freien Raum einmal klickt, wird die Markierung aufgehoben.

Am Reiter Eingang/Ausgang… ist alles ausgegraut…



Wenn kein Controller ausgewählt wurde und man z.B. auf Ausgang klickt, wird ein weiteres Feld "Controller hinzufügen" angezeigt. Die anderen Blöcke sind alle ausgegraut und man kann sie nicht in das Hauptprogramm rüber ziehen. Man muss erst einen Controller hinzufügen. Dazu kann man Button "Controller hinzufügen" anklicken oder oben im Menü den Controller auswählen. Wir drücken einfach den Button.



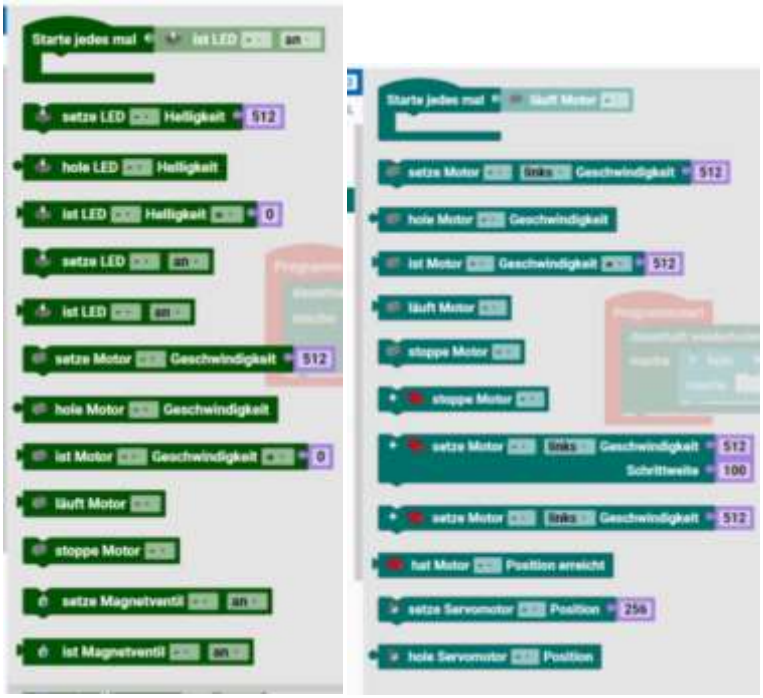
Bei dieser Robo Pro Coding Version, kann man unter zwei verschiedenen Controllern dem TXT 4.0 und dem BT Smart auswählen. Da bei der Erstellung TXT 4.0 angegeben wurde ist auch der BT Smart Controller ausgegraut. Hier kann man nur den TXT 4.0 auswählen. Um auch den BT Smart auswählen zu können sollte man ein neues Projekt starten und dort den BT Smart Controller wählen. Dann wird bei der Controllerkonfiguration der BT Smart Controller sichtbar. Ich habe eine Betaversion von Robo Pro Coding, wo auch der RX Controller mit vorhanden ist.

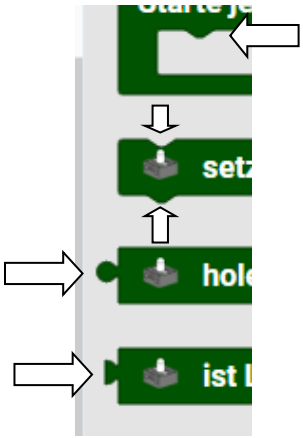
Im Hauptprogramm sind nun alle Blöcke sichtbar und man kann sie rüber ziehen. Der kleine graue Button „Controller hinzufügen ist verschwunden.

Blöcke sind teilweise ausgegraut

Wenn Blöcke teilweise ausgegraut sind, ist wahrscheinlich der gewählte Controller nicht in der Lage diese Blöcke zu verarbeiten. Z.B. kann der BT Smart Controller keine Ultraschall/Abstandssensoren oder auch „weiteren“ Pythoncode verarbeiten. Oder der RX-Controller keine Encodermotoren, weil er keine schnellen Zähler (Counter) hat…
Diese Blöcke sind dann ausgegraut und können nicht mit der Maus „gegriffen“ und rüber gezogen werden.

Andockmöglichkeiten von Blöcken
Vielleicht sind die verschiedenen Andockmöglichkeiten der Blöcke aufgefallen





Es gibt Startblöcke mit einer, manche haben oben und unten eine runde Vertiefung, manche haben vorne eine runde Verbindung und andere haben ein Dreieck.

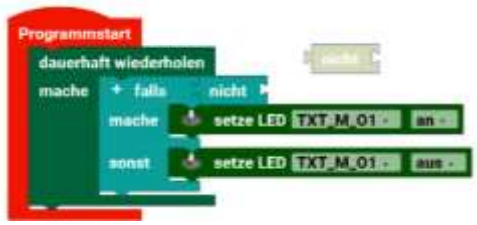
Vertiefung oben = Block kann an einen anderen Block verbunden werden.
Vertiefung unten = Weitere Blöcken können unten folgen
Diese stellen eine Verbindung her und damit den Programmablauf von Oben nach Unten, Block für Block.

Runder Anschluss an der Seite Stecker / Buchse = Werte/Inhalte werden von Rechts nach Links weitergegeben.

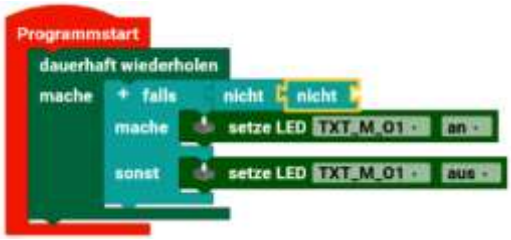
Nichts angeschlossen ist immer Null/Leer.

Eckiger Zapfen = Logisch Wahr/Falsch. Offene Anschlüsse = Logisch 0

Achtung: Es gibt Blöcke, die ihren **Anschluss ändern**, je nachdem welche Auswahl man im Dropdown wählt.



Auch bei –einem- „nicht“-Block
bleibt dieser offene Eingang logisch 0.



Bei zwei „nicht“-Blöcken
wird aus einem leeren Eingang eine logische 1. Hier würde die LED leuchten.

Menü der weiteren Möglichkeiten des Blocks

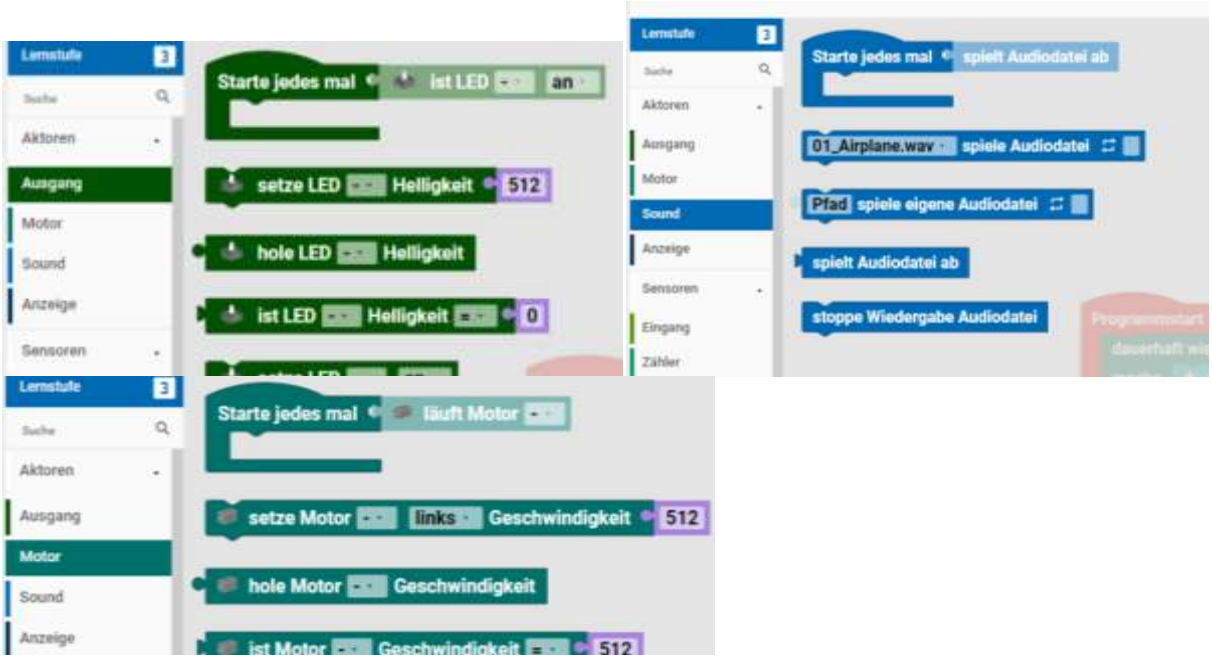
Wenn man mit der rechten Maustaste auf den blauen Block klickt, öffnet sich ein Menü

Man kann Duplizieren, Kopieren, Kommentar hinzufügen, Baustein zusammenfalten Baustein löschen und Hilfe auswählen.

Startblöcke

Startblock

Ein Startblock, ist wie ein eigenes Programm, das parallel zum Hauptprogramm läuft. Startblöcke können eine Startbedingung haben. Hier könnte man z.B. „ist LED an“ anklicken, aber auch andere aus dem Reiter – mit einem Dreieck vorne. Es kann nur innen weitere Blöcke aufnehmen.



Startblöcke in unterschiedlichen Farben aus den unterschiedlichen Registern.

So kann man deren Funktion besser unterscheiden.

Block zum Setzen von Werten

Dieser Block hat oben und unten eine Vertiefung. Er kann nur oben oder unten angeschlossen werden.

Hier in diesem Block, wird ein Wert zugewiesen. In diesem Fall ist es eine LED, wo der Anschluss noch nicht ausgewählt wurde, wo eine Helligkeit mit dem Wert 512 zugewiesen wird. Da die Werte für einen Ausgang von 0=Aus bis 512=Volle Spannung reichen, wird diese LED mit voller Leuchtkraft leuchten.

Block zum Lesen von Werten

Dieser Block hat nur vorne einen runden Anschluss. Er kann nur vorne angeschlossen werden.

Hier in diesem Block, wird ein Wert geholt. In diesem Fall ist es eine LED, wo der Anschluss noch nicht ausgewählt wurde, wo der Wert der Helligkeit geholt wird. Der Werte für einen Ausgang kann von 0=Aus bis 512=Volle Spannung reichen. Dieser Wert wird geholt nach vorne weitergereicht und dort weiterverarbeitet.

Block zum Vergleichen von Werten

Dieser Block hat nur vorne einen „dreieckigen“ Anschluss (fischertechnik-Zapfen) und hinten einen runden. Er kann vorne angeschlossen werden. Hinten kann eine Konstant, Variable oder ein gelesener Wert angeschlossen werden.

Duplizieren

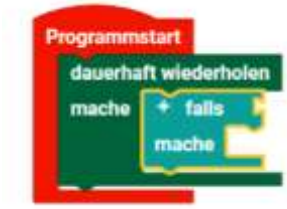
Duplizieren = eine Kopie anfertigen.

Der Block wird automatisch ausgegraut daneben abgelegt. Er wird nicht automatisch angeschlossen.

Man kann es greifen und an die richtige Stelle ziehen...

und ablegen. Für einen weiteren Block muss man ihn wieder duplizieren.

Kopieren

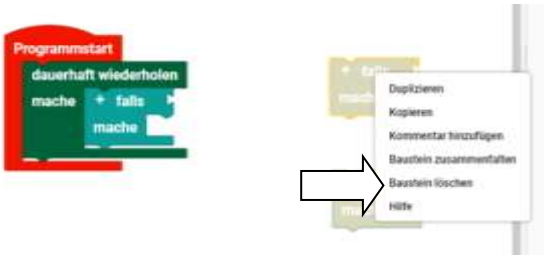


Beim Kopieren wird der Block in die Zwischenablag von Windows kopiert.

Man kann bei einem Rechtsklick in den freien Raum den Block dort einfügen.

Das geht auch mehrfach. Die Blöcke sind noch ausgegraut, da sie noch nicht angeschlossen sind.

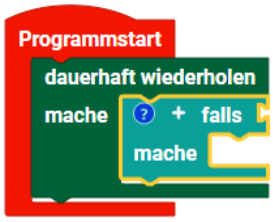
Baustein löschen



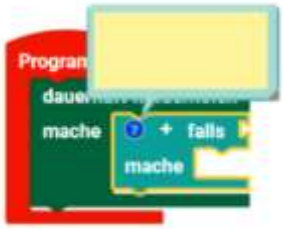
Löschen kann man mit dem Rechtsklick und "Baustein löschen" oder markieren und die Entfernen-Taste drücken. Wenn man ein Unterprogramm oder Funktion löscht, so werden auch die entsprechenden Blöcke in den Programmen gelöscht.

Zweite Möglichkeit: Baustein greifen (also mit gedrückter Maustaste) und nach links auf die farbigen Reiter ziehen. Auch damit wird der Baustein gelöscht.

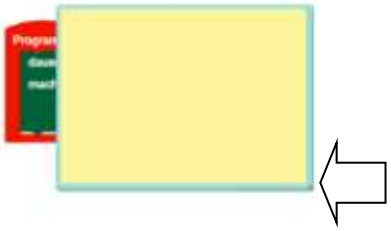
Kommentar



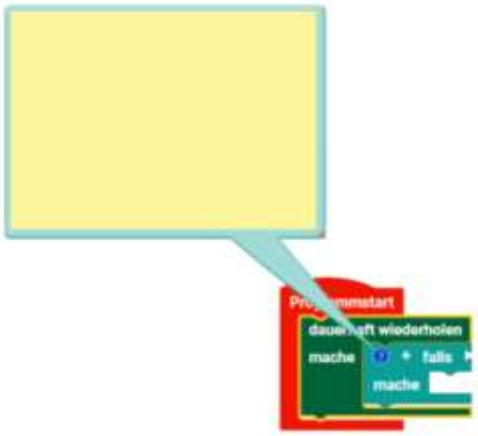
Kommentar hinzufügen. Da entsteht ein „?“ im Symbol. Wenn man auf das Fragezeichen klickt…



entsteht ein Textfeld, in dem man Text eingeben kann. Das kann z.B. für Erklärungen oder Funktion sein.

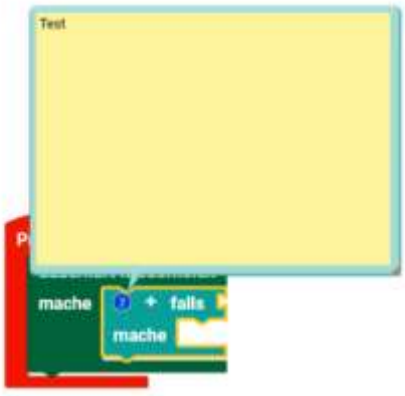


Wenn man es größer brauch, kann man unten rechts das Feld vergrößern.



Wenn man den Rand mit der Maus fasst, kann man den Kommentar verschieben. Es werden dabei aber Sachen verdeckt.

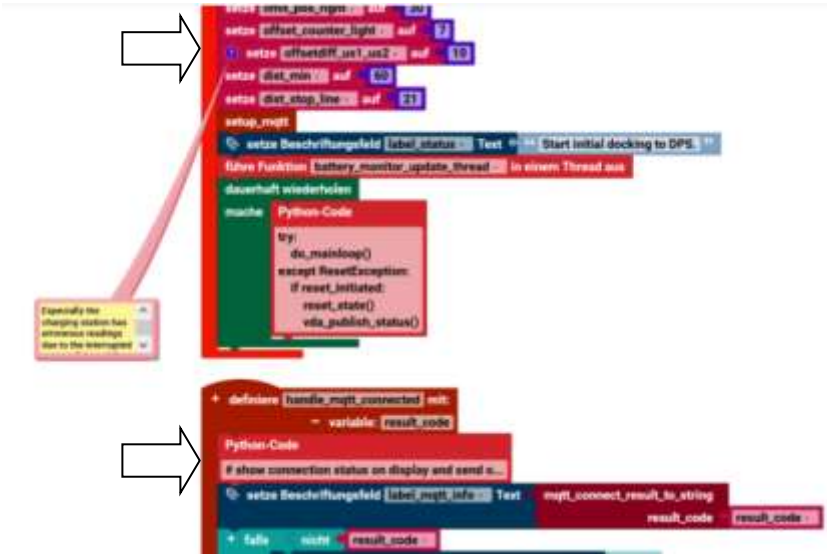
Ein erneutes Klicken auf das Fragezeichen, lässt den Kommentar verschwinden und nur das Fragezeichen ist sichtbar.



Ein Klick auf das Fragezeichen lässt den Kommentar wieder erscheinen. Er erscheint aber nun an einer –anderen- Stelle. Wenn man so Programm erklären möchte, z.B. für Schüleraufgaben und diese Ausdrucken möchte, sollte man die vorher mal entsprechend positionieren. Das hinterher nochmal genauso hinzubekommen, wird schwer werden.

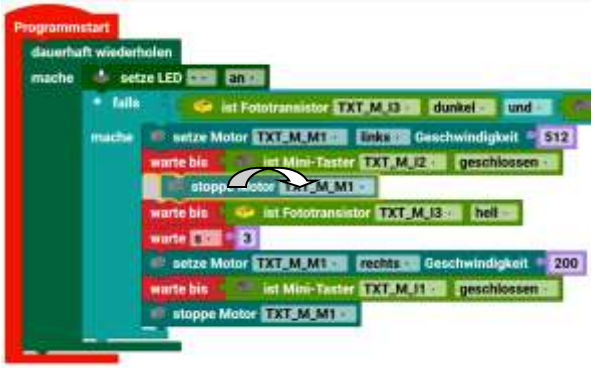
Mit einem Rechtsklick kann man über das Menü den Kommentar wieder entfernen.

Eine Alternative zum Kommentar ist, mit dem Python-Code-Block aus dem Reiter Util, einen Python-Kommentar zu schreiben und ihn mit dem „#“ zu beginnen. fischertechnik macht das in ihren Beispielen auch so.



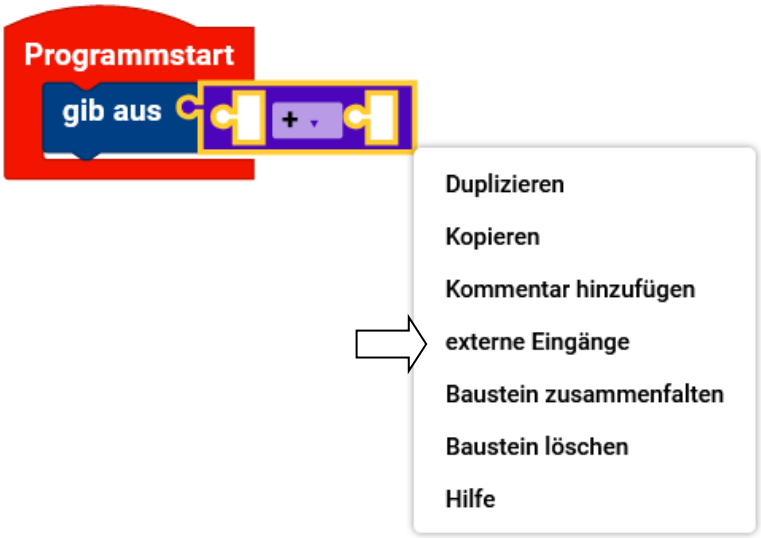
Beispiel eines Kommentars in einem Python-Code-Block (aus fts_main)

Einzelne Bausteine aus einem zusammenhängenden Programm markieren / löschen



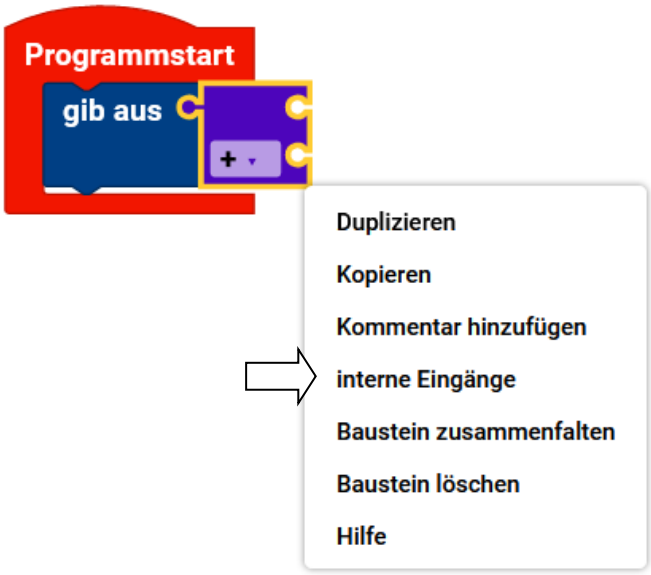
Block mit gedrückter [CRTL]-Taste und linker Maustaste markieren. Entweder man zieht ihn nach rechts rüber oder man kann ihn (markiert) mit der [Entf]-Taste löschen.

Externe Eingänge, Interne Eingänge



Wenn man mit der rechten Maustaste auf einen Block klickt, kann man zwischen zwei Darstellungen wählen. Im Block oben sind die Anschlüsse innenliegend, also intern. Diese Darstellung kann man ändern, in dem man auf „externe Eingänge“ umschaltet.

Hier sind die Anschlüsse nun „extern“.



Um auf die interne Darstellung umzuschalten, muss man nur wieder mit der rechten Maustaste auf den Block klicken und „interne Eingänge“ auswählen.

Baustein zusammenfalten

Baustein zusammenfalten. Hier verkleinert sich die Ansicht des Blocks. Das ist manchmal hilfreich, um eine Übersicht über das Programm zu bekommen.

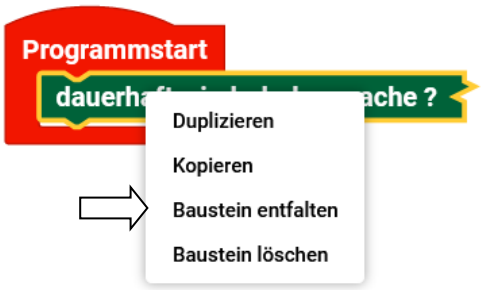
Dazu mit der rechten Maustaste auf den entsprechenden Teil klicken. Hier Programmstart und auf „Baustein zusammenfalten“ klicken.

Nun hat man die verkleinerte Darstellung des Blocks Programmstart.

Es ist auch möglich innere Teile zusammenfalten.

Hier ist es „dauerhaft wiederholen“.

Bausteine entfalten

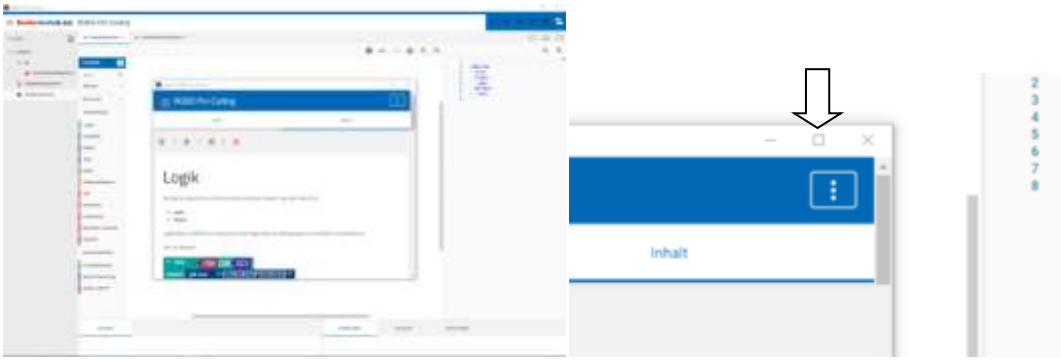


Blöcke kann man auch wieder entfalten. Dazu auf „Baustein entfalten“ klicken.

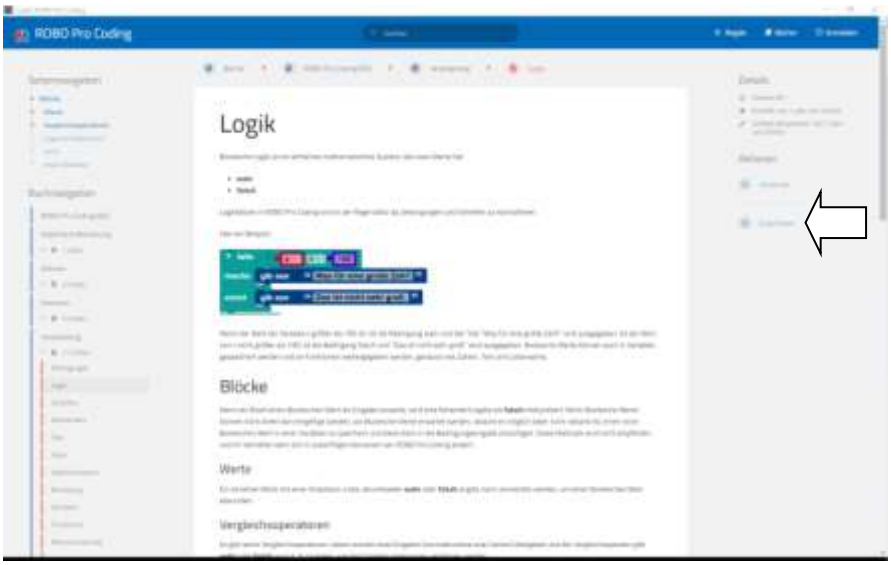
Baustein entfalten

Über Baustein entfalten, klappt sich der Baustein wieder aus.

Baustein Hilfe



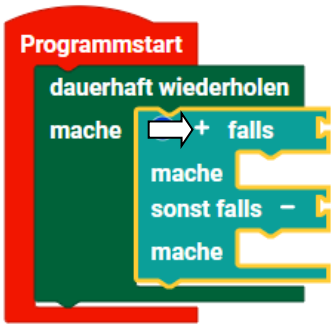
Baustein Hilfe. Da kommt eine Hilfe – was zu erwarten war ☺. Tipp: Auf das Fenster oben rechts klicken und damit das Fenster vergrößern.



Dann kommen viel mehr Möglichkeiten. Unter anderem kann man den Teil des "Buches" z.B. in eine PDF exportieren, die man durchsuchen und ausdrucken kann.

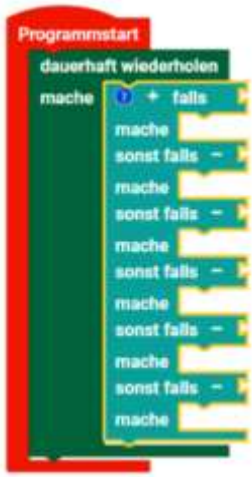
Tipp: Man kann diese PDF auch durchsuchen. Wenn man die Tasten [Strg]+[f] öffnet sich ein Fenster, wo man den Begriff, den man sich eingeben kann.

Das Plus am Block



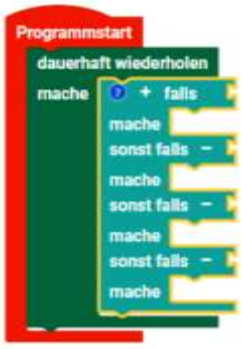
Ein Klick auf das Plus, vor dem "falls", erweitert den Block. Das, was nun zu sehen ist, hängt vom Block

selber ab. Hier hat man eine weitere Abfrage.



Jeder Klick auf das Plus lässt erweitert den Block

Das Minus am Block



Ein Klick auf das Minus löscht den Teil.

Kurzhilfe zum Block (mouseover)

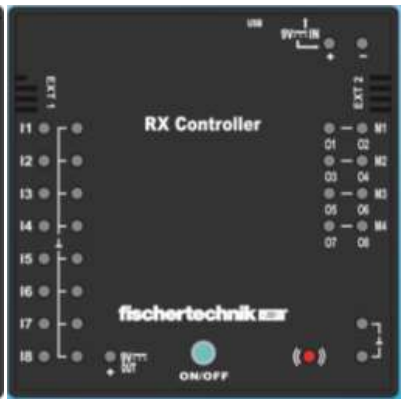
Wenn man mit dem Mauszeiger an einem Block ist, wird eine kleine Hilfe angezeigt.

Geeignete Controller

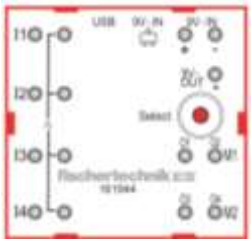
Man muss einen geeigneten fischertechnik Controller haben.
Das kann ein TXT 4.0 , RX Controller oder ein BT Smart Controller sein.

Diese fischertechnik Controller sind geeignet:

TXT Controller



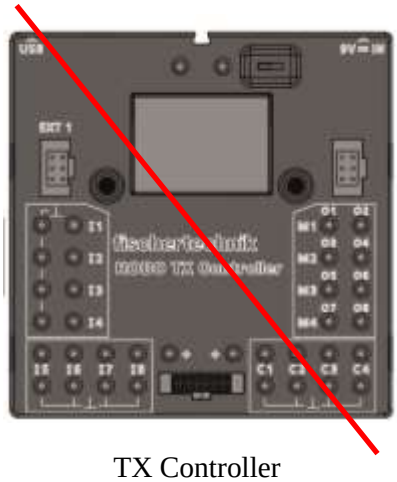
RX Controller



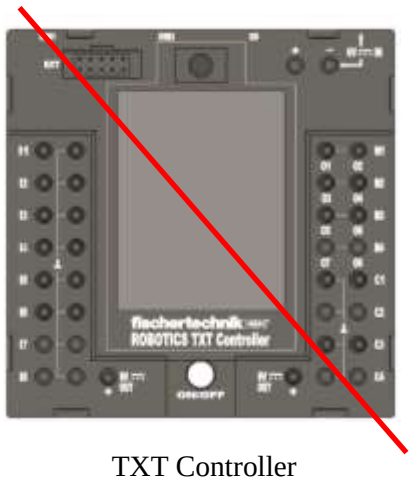
BT Smart Controller

Ältere fischertechnik Controller

Wichtig!
Robo Pro Coding **-läuft nicht-** mit einem der Vorgänger Controller TX oder TXT.
Diese werden mit „**RoboPro**“ programmiert, **nicht** mit „**Robo Pro Coding**“.
Das sind zwei verschiedene Programmierungen.



TX Controller



TXT Controller

Wenn im Robo Pro Coding „TXT“ auftaucht, ist immer der TXT 4.0 gemeint.

Spannungsversorgung

Man muss auch eine Spannungsversorgung haben. Dazu entweder einen (geladenen) 8,4V fischertechnik-Akku, ein neueres fischertechnik-Netzteil mit 2,5A (**nicht** die älteren **mit 1A**) oder den neuen 9V Batteriehalter mit 9 AA Batterien, für den TXT 4.0 und RX Controller. Beim BT Smart Controller ist ein (roter) 9V Block-Batteriehalter ausreichend. Man kann aber auch einen fischertechnik Akku nehmen.
Bei den neuen Kästen ist im blauen Batteriegehäuse ein 1000mAh Akku drin.
Das USB-Kabel vom PC lädt keine Akkus auf und reicht nicht um einen Controller zu betreiben.

Je nachdem wie viele und wie lange, Motoren beim Programmieren benutzt werden, halten die Akkus und Batterien länger oder kürzer. Zum Aufladen vom Akku, den TXT 4.0 ausschalten, vom Akku trennen und dann den Akku laden. Bei RX-Controller mit dem 9V Batteriehalter den Controller ausschalten und die AA Batterien wechseln.
Bei fahrbaren Robotern wird parallel zum Akku ein Netzteil angeschlossen und nur für die Testfahrten das Netzteil rausgezogen. So kann er sich frei bewegen und der Akku wird geschont.
Bitte benutzen sie zum Laden vom fischertechnik Akku ausschließlich das fischertechnik Ladegerät!
Man kann auch den Energiesparmodus vom TXT 4.0 benutzen. Der schränkt allerdings die Handhabung etwas ein. Siehe dazu auch die Bedienungsanleitung zum TXT 4.0 .

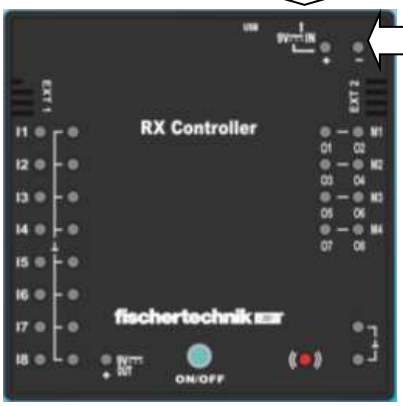
Netzteil (Seite)

Akku

Spannungsversorgung TXT 4.0 Controller

Von der Seite für das Netzteil, von Oben für den Akku.
Lampen oder LEDs vom Modell **nicht** auch noch in die Spannungsversorgung stecken. Wenn man die unteren Plusbuchsen nimmt, werden diese beim Ausschalten vom TXT 4.0 mit ausgeschaltet. Wenn man die oben anschließen würde, hätte das zur Folge, dass die Lampen weiterleuchten und der Akku leer wird.

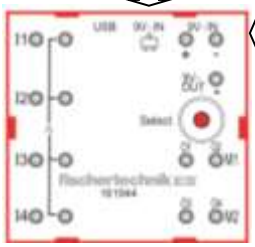
Netzteil (Seite)



Akku Spannungsversorgung RX Controller

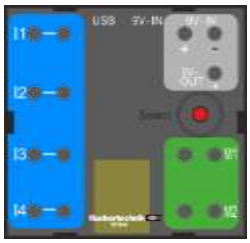
Hinweis: Momentan sieht es so aus, dass die Akkus vom RX-Controller, nicht voll ausgeschöpft werden können. Wir ermitteln noch, an verschiedenen RX, ab welcher Spannung er genau abschaltet. Jetzt sind es ca. 8,7V. Das kann sich aber noch ändern.

Netzteil (Seite)



Spannungsversorgung BT Smart Controller

Hinweis: Im Buch wird er rot dargestellt.



BT Smart Controller
- in neuem Design.

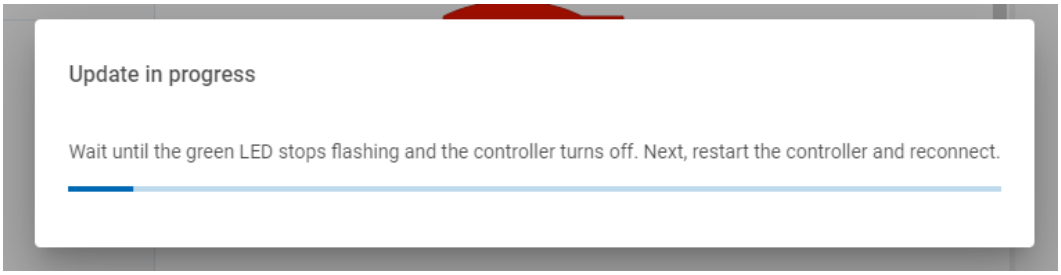
Wichtig! Die Sache mit dem Update der Firmware…
Wenn man von Robo Pro Coding aufgefordert wird ein Update der Firmware von Controller zu machen, muss man einen vollen Akku oder volle Batterien haben. Besser ist es ein Netzteil zu nehmen.

Wenn man nur einen „etwas“ geladenen Akku hat, sollte man **„Abbrechen“ und erst den Akku laden**. Diese Zeit muss man sich nehmen!

Wenn alles OK ist „UPDATE NOW“ drücken

Hier beim **RX-Controller**:

Und nun auf „Update Now…“



und nun abwarten…

Beim **TXT 4.0** ist es fast genauso. Ich hatte beim Testen von Software, drei verschiedene Versionen von Robo Pro Coding auf dem Rechner und der TXT 4.0 wollte auf einmal kein Update mehr akzeptieren. Abhilfe schafft da, die Update-Datei auf den TXT 4.0 zu bringen. Da gibt es die Möglichkeit sie über einen USB-Stick aufzuspielen oder die Datei über einen SSH-Zugang direkt auf den TXT 4.0 zu übertragen.

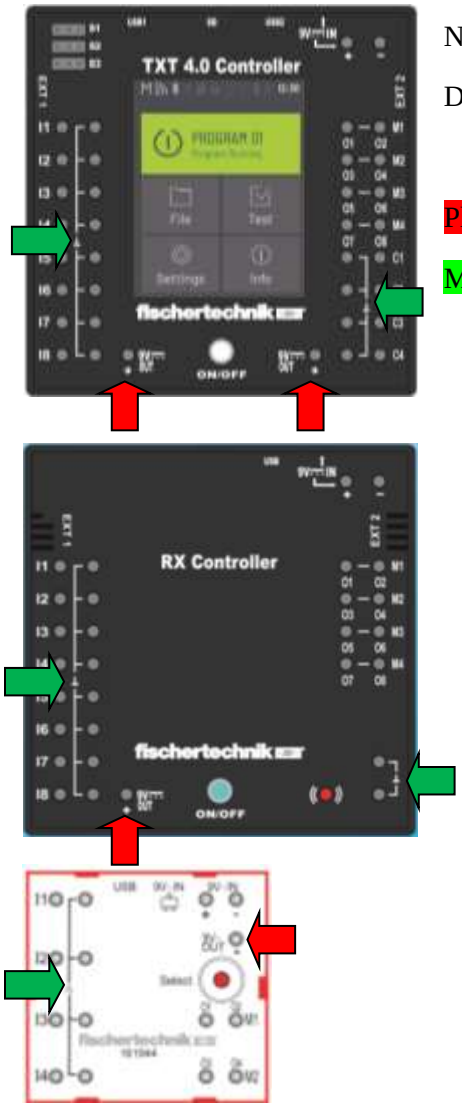
Näheres dazu gibt es auf der ftCommunity.de:
https://forum.ftcommunity.de/viewtopic.php?f=8&t=7978&p=64063

Der **BT Smart Controller** braucht kein Update der Firmware. Er ist nur ein Online Controller – halt mit Kabel oder mit stehender Bluetooth-Verbindung, ohne einen Download der Programme.

Kabelfarben und Steckerfarben:

Kabelfarben
Bei fischertechnik sind in der Regel diese Kabelfarben, mit dieser Funktion:
Rot = Plus +
Grün = Minus - (oder GND Ground = Minus)

Schwarz, Gelb, Dunkelblau… = Signal (z.B. bei Sensoren)



Nutzbare Plus- und Minusbuchsen an den Controllern.

Die Buchsen mit dem GND Zeichen sind alle Minus.

Plus

Minus = GND

Steckerfarben
fischertechnik benutzt heutzutage zwei Steckerfarben:

Rot = Plus und Signal (von den Sensoren)
Grün = Minus

Gerade wenn man Fehler sucht, fällt es mehr auf, wenn man die richtige Farbe benutzt, die in der Anleitung angegeben ist.
Wenn man die neuen Stecker mit den Kabeln verbindet, sollte man darauf achten es richtig zu machen.
Also den grünen Strich am roten Kabel, mit dem grünen Stecker verbinden. Rotes Kabel mit roten Steckern.

Drehrichtung von Motoren

Mit der Drehrichtung von Motoren ist das so eine Sache. In den fischertechnik Bauanleitungen ist die genaue Verkabelung vorgeben. Es kann aber sein, dass trotzdem die Drehrichtung nicht stimmt.

Es kam genau die Fragestellung im Forum auf, worauf wir unsere eigenen Bestände überprüften.

Graue Motoren

Vor allem bei den grauen Motoren stellte sich raus, dass die mal rechts mal links drehten. Es stellte sich raus, dass es die Frage war, wie die Motoren bei der Produktion in den Baustein eingebaut wurden.

Schwarze Motoren

Bei den schwarzen Motoren, habe ich selbst es bisher noch nicht festgestellt, dass die Drehrichtung anders ist.

Es gab von der Firma Knobloch, eine Sonderedition, bei denen eine andere linksdrehende statt einer normalen rechtsdrehenden Schnecke eingebaut war. Ist aber eher ein Sammlerstück.

Kabel

Gerade bei gebrauchtem fischertechnik, sind bei den Kabeln schon die komischsten Sachen vorgekommen. Da würde ich –immer- die Stecker neu anschließen.

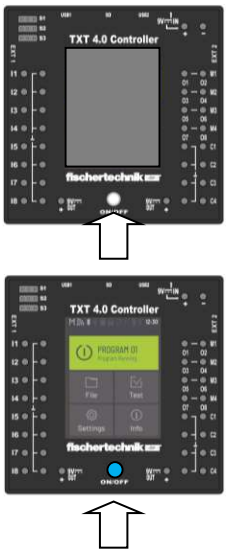
Es kann also sein, dass die Stecker vertauscht sind.

Einbaurichtung

Es kann auch sein, dass der Motor falsch rum eingebaut wurde. Somit dreht sich die Drehrichtung um.

Ansonsten einfach die Stecker am Controller –oder- Motor tauschen.

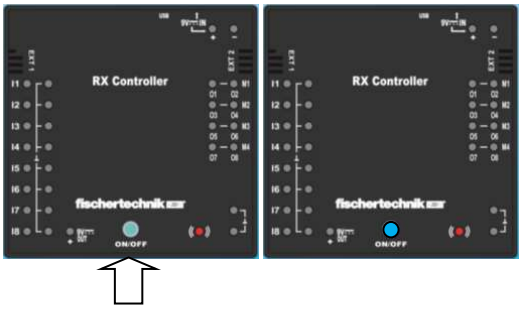
Einschalten der Controller



Einschalten vom TXT 4.0 Controller
Zum Einschalten die ON/OFF Taste ca. 4-5 Sekunden drücken, bis das Display sich einschaltet und die LED in Taster anfängt zu leuchtet **und wieder ausgeht**. Erst dann loslassen. Die LED wird grün und der Controller startet nun.
Wenn das Auswahlfenster erscheint und die die Anzeigenlampe blau leuchtet, ist der Controller bereit. (Weitere Einzelheiten dazu stehen u.a. in der TXT 4.0 Anleitung und in der „RoboProCoding_Schueleranleitung-9.pdf“.)

Man kann nun die Sprache einstellen, da die Standarteinstellung Englisch ist. Auf dem Display Settings/Language und dann auf Deutsch. Wenn man auf den Homebutton drückt, wird die Einstellung übernommen.

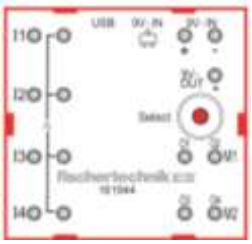
Einschalten vom RX Controller
Zum Einschalten die Taste ca. 4-5 Sekunden drücken. Die LED wird rot, eine weitere Sekunde gedrückt halten und dann kann man loslassen. Der Controller startet nun. Um ein Programm zu starten, die Taste noch mal drücken und die LED wird grün.



Einschalten vom BT Controller
Der BT Controller hat keinen Schalter. Mit verbinden der Spannungsquelle ist er eingeschaltet.

Wenn man den 9V Block nimmt, hat dieser einen Umschalter. Je nachdem wie rum man die Kabel eingesteckt hat, ist der BT Smart Controller an. Um ihn auszuschalten, muss man den Schalter in Mittelstellung bringen.

Tipp: Am besten die Stecker am 9V- Block-Kästchen rausziehen. Das schont die Batterie am besten. Wenn man einen 9V Akku- Block benutzt, immer nur mit dem fischertechnik Batteriehalter benutzen. Nicht die älteren Klips. Der neue Batteriehalter hat eine Kurzschlusssicherung!



Verbinden von Robo Pro Coding mit dem Controller (Blaue Kopfzeile)

Übersicht

TXT 4.0 Controller
Der **TXT 4.0 Controller** kann über vier Möglichkeiten mit dem PC (Tablett, Smartphone) Verbindung aufnehmen:
- USB-Kabel
- WLAN (Netzwerk z.B. vom Haus)
- Access point (Netzwerk vom TXT 40 Controller)
- Bluetooth (Nahfunkverbindung)

RX Controller / BT Controller
Der **RX Controller** und der **BT Controller** können über zwei Möglichkeiten mit dem PC verbunden werden:
- USB-Kabel
- Bluetooth (Nahfunkverbindung)

TXT 4.0 Controller verbinden:

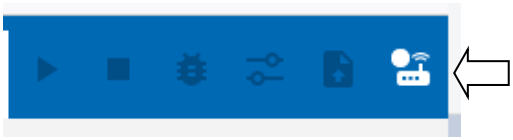
Bemerkung/Tipp:
Es hat sich gezeigt, dass es viele verschiedene Möglichkeiten für Netzwerke gibt. Früher wurden über USB-Kabel sogenannte (Serielle-) COM-Schnittstellen eingerichtet. Man wählte da COM1... usw. aus. Heute gibt man diesen USB-Verbindungen IP Namen. Beim TXT 4.0 wird z.B. 192.168.7.2 genommen und bei Robo Pro Coding vorgegeben.
Für das WLAN wird auch eine andere IP Adresse genommen. Z.B. 192.168.2.12 . Die wird vom Router vorgegeben.

Die von Robo Pro Coding vorgegebene Standard- IP-Adresse -kann- aber falsch sein. Wenn man keine Verbindung aufbauen kann, sollte man das mal kontrollieren. Z.B. WLAN auf dem TXT 4.0 unter "Info"/"WLAN"/"IP".
Wenn man den TXT 4.0 ständig auch bei WLAN neu anmelden muss, sollte man das Häkchen bei "Automatisch verbinden" im Netzwerk vom PC aktivieren.
Bei Bluetooth kann es sein, dass man jedes Mal, wenn der PC ausgeschaltet wurde, die Bluetooth-Verbindung erst wieder aktiviert werden muss. Wenn man z.B. Bilder von der Kamera des TXT 4.0 übertragen möchte, sollte man nicht das etwas langsamere Bluetooth nehmen, sondern WLAN.

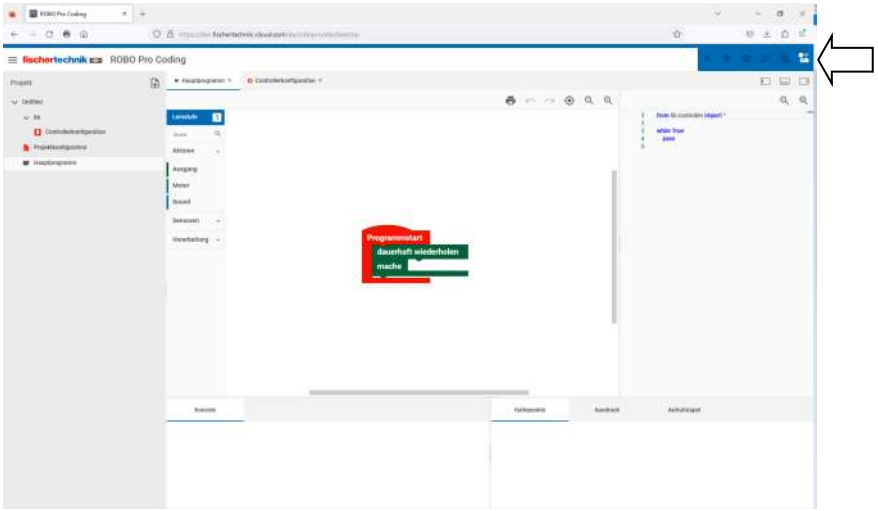
Manche Bildschirmschoner oder der Sperrbildschirm von Windows schließen die Verbindung zum TXT 4.0 . Man muss also diese erst wieder aktivieren, wenn man z.B. das Passwort eingegeben hat.

Die vom Router vorgegebene IP kann auch wechseln. Deshalb kann man im Router diese Adresse als feste Adresse eintragen.

USB Kabel



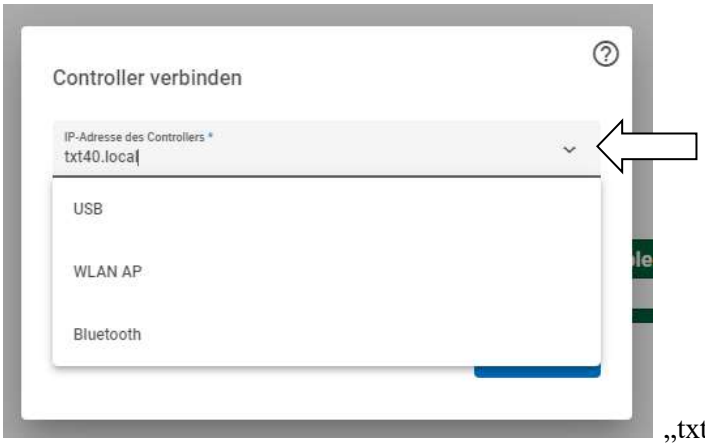
Dazu oben rechts auf "Controller verbinden" klicken.



TXT 4.0 verbinden

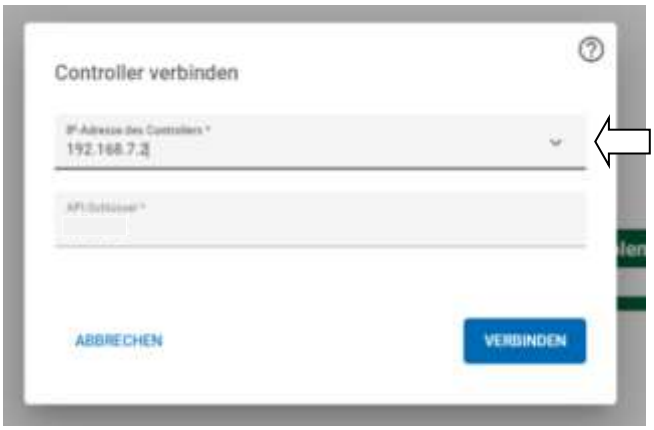
Wir nutzen hier das USB-Kabel

Klick auf „txt40 local“ liefert die Auswahl USB/WLAN AP/Bluetooth

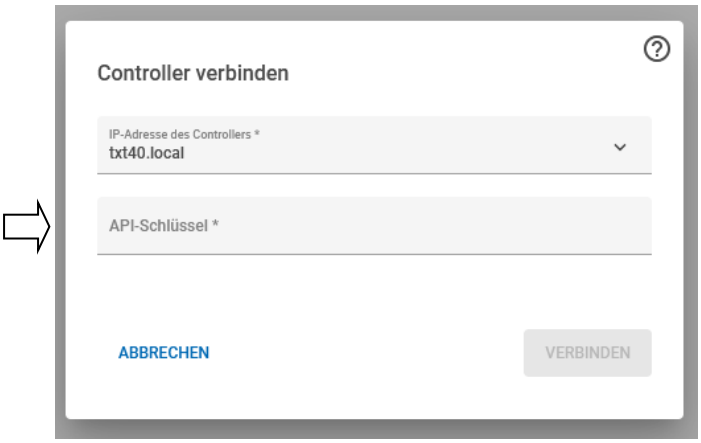


„txt40.local“ ist nur ein Ersatz für die IP Adresse.

Ein Klick auf „USB“ liefert die IP Adresse, die von Robo Pro Coding vorgegeben wird.

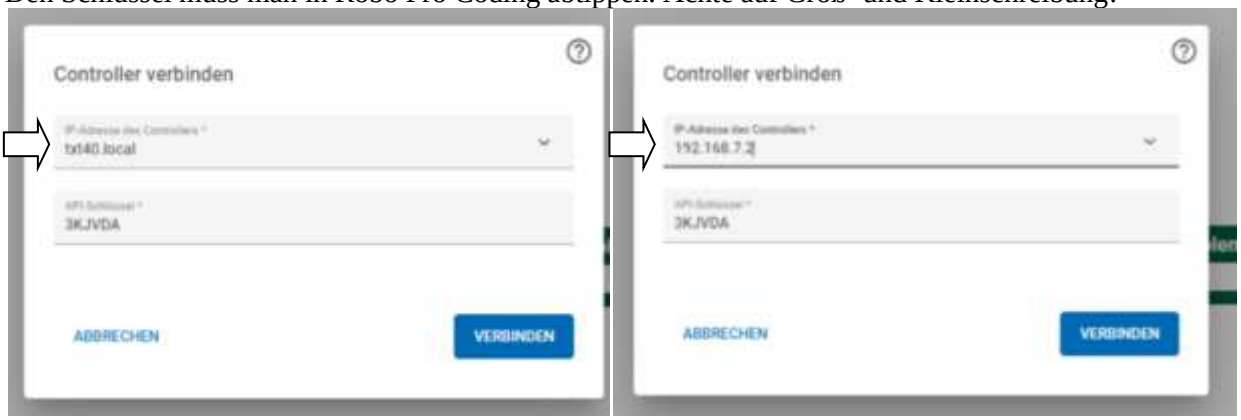


Hier wird die IP Adresse 192.168.7.2 angezeigt.



Hier wird nach einem API-Schlüssel gefragt.

Dazu muss man beim TXT 4.0 Controller, auf "Einstellungen" / "API Schlüssel" gehen.
Den Schlüssel muss man in Robo Pro Coding abtippen. Achte auf Groß- und Kleinschreibung!

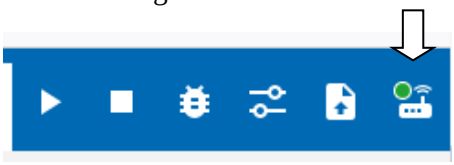


Hier „txt 40 local“ mit dem eingegebenen API Code----Hier die IP Adresse mit dem API Code

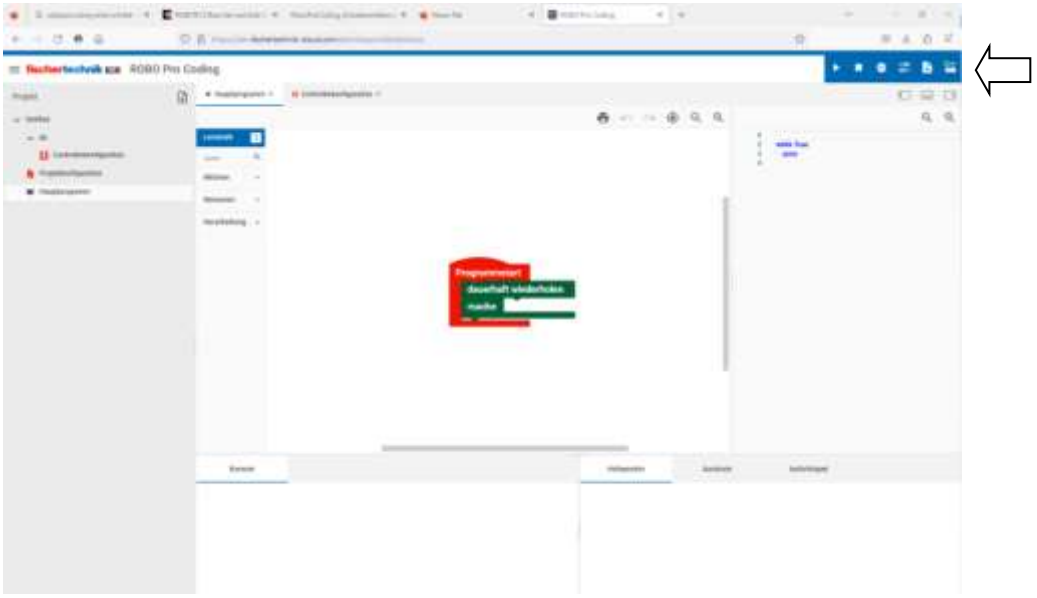
Ist aber das Gleiche.

ACHTUNG! Man **muss** bei TXT 4.0 aus dem API-Schlüssel Menü wieder in das Hauptmenü!
Über Zurück (mehrfach) oder Homesymbol.

Nun auf "Verbinden" in Robo Pro Coding und das Verbinden-Symbol sollte nun grün sein und die weiteren Zeichen nun gut sichtbar sein.



Verbunden. Das rechte Symbol ist grün und die anderen sichtbar.



Der TXT 4.0 ist nun über USB-Kabel mit Robo Pro Coding verbunden

Wenn die Verbindung nicht klappt.
ACHTUNG! Man **muss** bei TXT 4.0 aus dem API-Schlüssel Menü wieder in das Hauptmenü!
Über Zurück (mehrfach) oder Home-Symbol.

Erst dann kann man eine Verbindung aufbauen.

WLAN

Zuerst muss WLAN am TXT 4.0 Controller aktiviert werden. Dazu wird er im Netzwerk angemeldet und vom Router gibt es eine IP Adresse. Die ermöglicht es dem TXT 4.0 Controller auch Updates automatisch aus dem Internet runterzuladen. Der Hauptgrund WLAN zu benutzen ist aber, dass kein störendes Kabel z.B. für Fahrroboter da ist.

Dazu muss man am TXT 4.0 Controller auf Einstellungen /Netzwerk gehen.

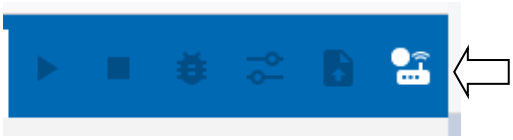
“Access point“ nicht aktivieren.

Man muss das Netzwerk auswählen und das Routerpasswort vom Netzwerk eingeben.

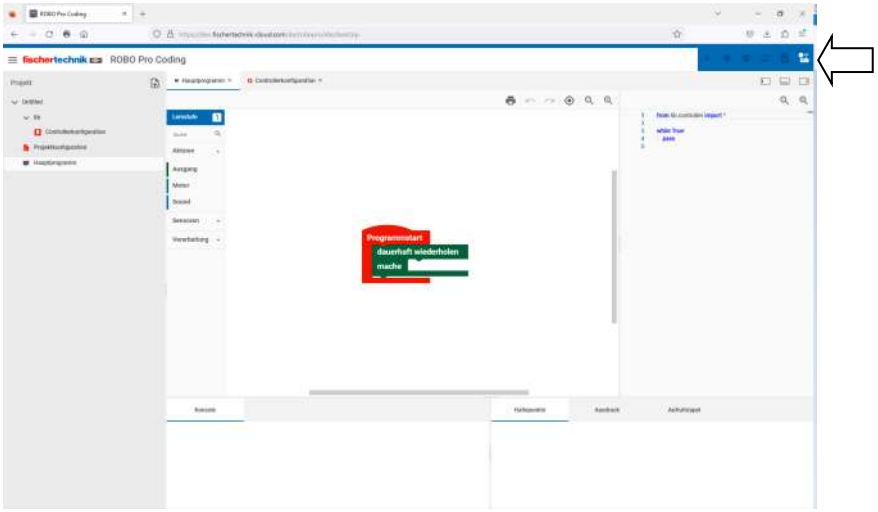
Z.B. steht es bei Fritzbox unter dem Gerät. Je nach Installation kann es auch ein eigenes sein.

Eine funktionierende Verbindung wird durch das WLAN-Symbol in der obersten Zeile des Displays vom TXT 4.0 Controllers angezeigt.

Jetzt kommen wir zu Robo Pro Coding.



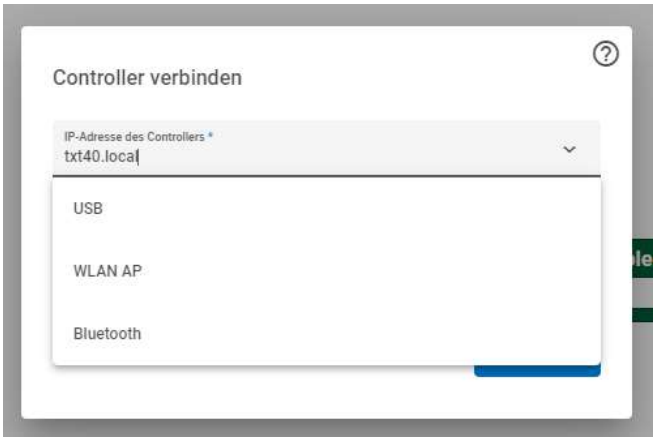
Dazu oben rechts auf "Controller verbinden" klicken.



TXT 4.0 verbinden

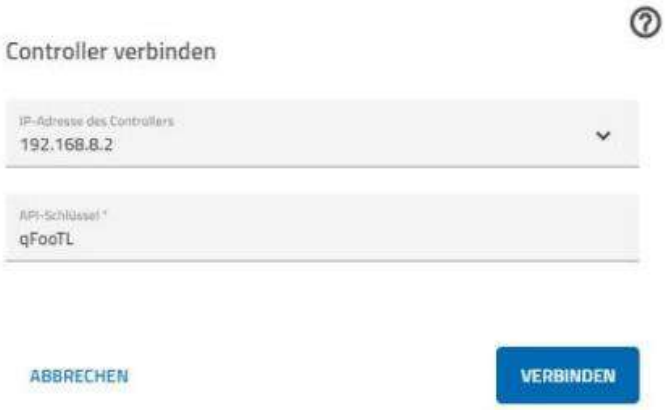
Wir nutzen hier WLAN

Klick auf „txt40 local“ liefert die Auswahl USB/WLAN AP/Bluetooth

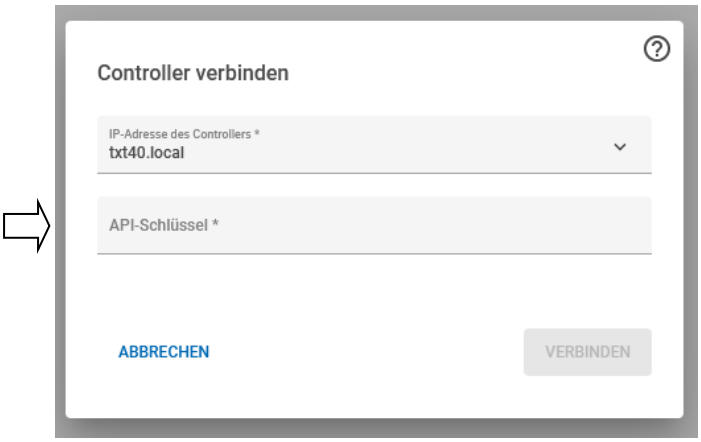


„txt40.local“ ist nur ein Ersatz für die IP Adresse.

Ein Klick auf „WLAN AP“ liefert die IP Adresse, die von Robo Pro Coding vorgegeben wird.
Z.B. 192.168.2.12
Hier muss die IP Adresse vom Router eingegeben werden.



Hier ist schon ein API Schlüssel angegeben, aber der API-Schlüssel den du hast, ist bestimmt anders.

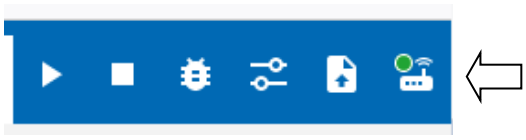


Hier wird nach einem API-Schlüssel gefragt.

Dazu muss man beim TXT 4.0 Controller, auf "Einstellungen" / "API Schlüssel" gehen.
Den Schlüssel muss man in Robo Pro Coding abtippen. Achte auf Groß- und Kleinschreibung!

Wenn etwas Eingegeben wurde erscheint Verbinden in blau.

Nun auf "Verbinden" in Robo Pro Coding klicken und das Verbinden-Symbol sollte grün sein und die weiteren Zeichen nun gut sichtbar sein.



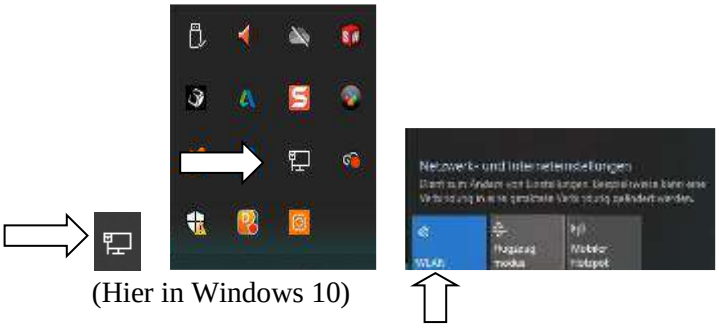
Der Controller ist nun im WLAN.

Access point
Ein Access Point ist in dem Fall ein Netzwerk, was vom TXT 4.0 Controller erzeugt wird.
Hier muss sich der PC oder Smartphone, mit dem TXT 4.0 verbinden. Die Zugangsdaten kommen vom TXT 4.0 Controller. Eine Verbindung zum Internet ist dabei nicht möglich, somit auch keine Updatemöglichkeit der Firmware.

Dazu über Einstellungen / Netzwerk –nur- das „Access point“ aktivieren.

Dazu muss man am TXT 4.0 Controller auf Einstellungen /Netzwerk gehen.

Nun am PC/Smartphone WLAN aktivieren



(Hier in Windows 10)

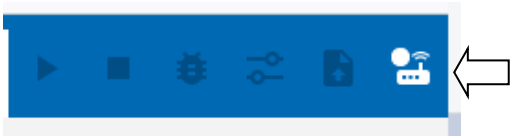
Hier auf das TXT 4.0 Netzwerk klicken.

Es wird nach dem Netzwerkschlüssel gefragt.

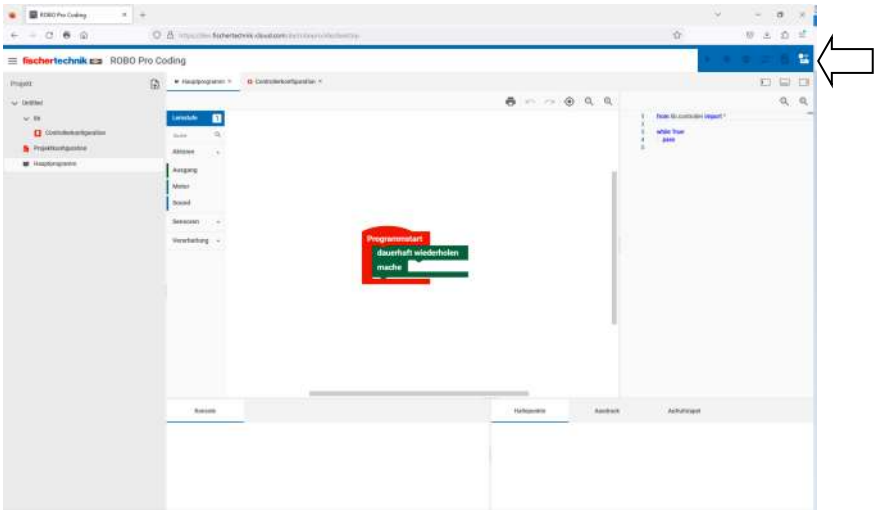
Die ist im TXT 4.0 Controller unter Einstellungen / Netzwerk / Access point / Passwort.
Diesen Schlüssel muss man auf dem PC eintragen.

Es wechselt „Verbinden“ in „Verbunden“.

Nun muss noch in Robo Pro Coding diese Verbindung eingestellt werden.



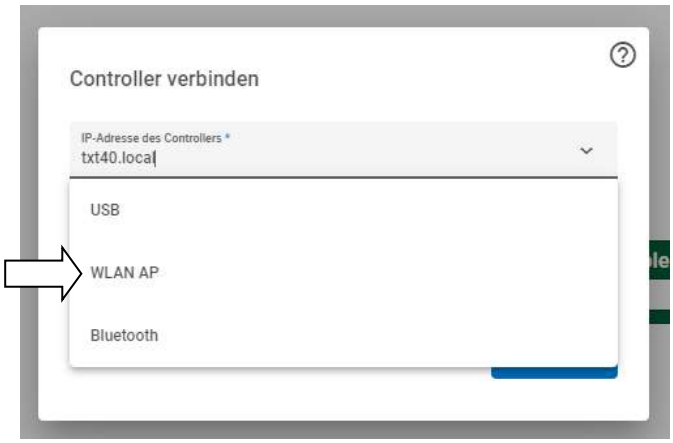
Dazu oben rechts auf "Controller verbinden" klicken.



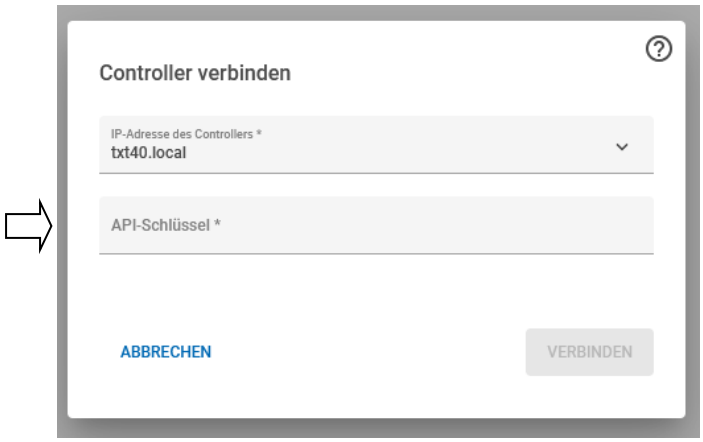
TXT 4.0 verbinden

Wir nutzen hier Access point

Klick auf „txt40 local“ liefert die Auswahl USB/WLAN AP/Bluetooth



„txt40.local“ ist nur ein Ersatz für die IP Adresse (z.B. 192.168.8.2).



Hier wird nach einem API-Schlüssel gefragt.

Dazu muss man beim TXT 4.0 Controller, auf "Einstellungen" / "API Schlüssel" gehen.
Den Schlüssel muss man in Robo Pro Coding abtippen. Achte auf Groß- und Kleinschreibung!

Wenn etwas Eingegeben wurde erscheint Verbinden in blau.

Nun auf "Verbinden" in Robo Pro Coding klicken und das Verbinden-Symbol sollte grün sein und die weiteren Zeichen nun gut sichtbar sein.

Der PC ist nun über den Access point im WLAN des TXT 4.0 Controllers.

Bluetooth Verbindung
Die Bluetooth (BT) Verbindung hat eine etwas eingeschränkte Übertragung. Wenn man Bilder der Kamera übertragen möchte, kommt man schnell an die Grenzen. Im Grunde läuft die Einrichtung genau wie die WLAN ab. Man aktiviert BT und man aktiviert BT auf dem PC und führt ein Peering also eine Verbindung durch.

TXT 4.0 und Bluetooth

Eine Beschreibung siehe auch „RoboProCoding_Schueleranleitung-9.pdf bei fischertechnik zum Runterladen.

Aktivierung von Bluetooth auf dem TXT 4.0

Hinweis: Man sollte vorher die anderen WLAN… deaktivieren, um nur BT zu nutzen.

Dazu muss man auf dem Controller bei Einstellungen/Netzwerk/Bluetooth aktivieren.

Dazu den Schalter auf dem Display nach rechts „schieben“.

Bluetooth ist jetzt am TXT 4.0 aktiviert. Es wird ein Kopplungscode angezeigt hier 604152

Aktivierung von Bluetooth auf dem PC

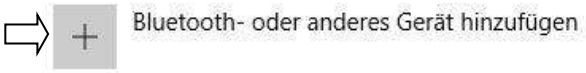
Problem: Verschiedene Betriebssysteme, verschiedene Arten, um Bluetooth einzuschalten. Hier eventuell die Beschreibung zum eigenen Rechner zurate ziehen.

Unten rechts in der Statusleiste am Windows Bildschirm gibt es dieses Zeichen:

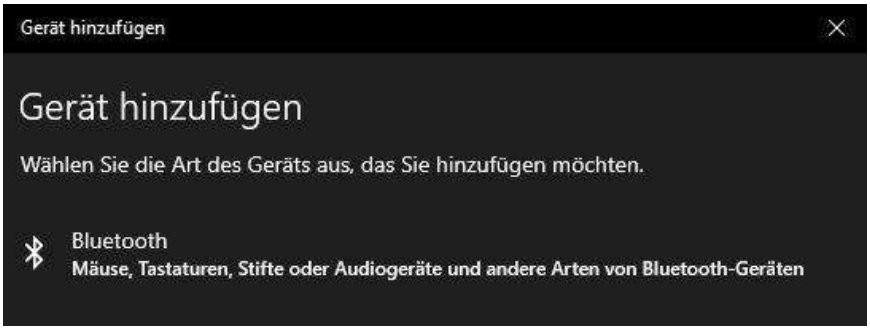
 Damit kann man ausgeblendete Symbole einblenden.

Damit kann man das Bluetooth Symbol  aktivieren.

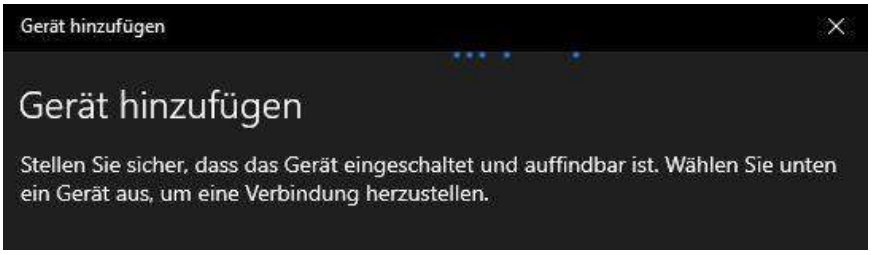
Nun Bluetooth-Gerät hinzufügen.



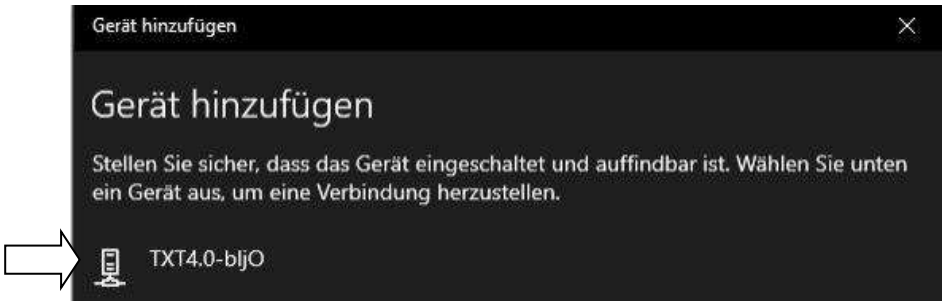
Hier auf das Plus klicken.



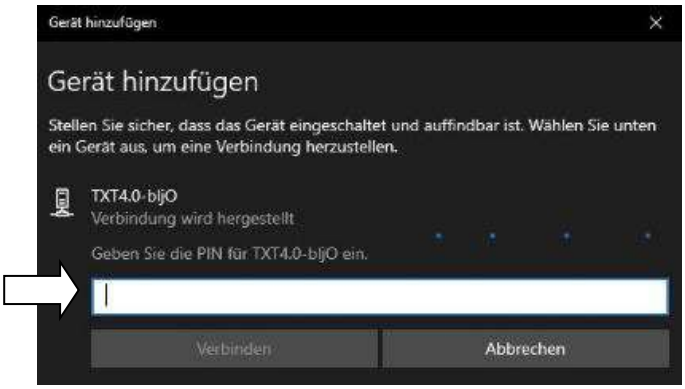
Es kommt ein Fenster, in dem man den Eintrag Bluetooth.



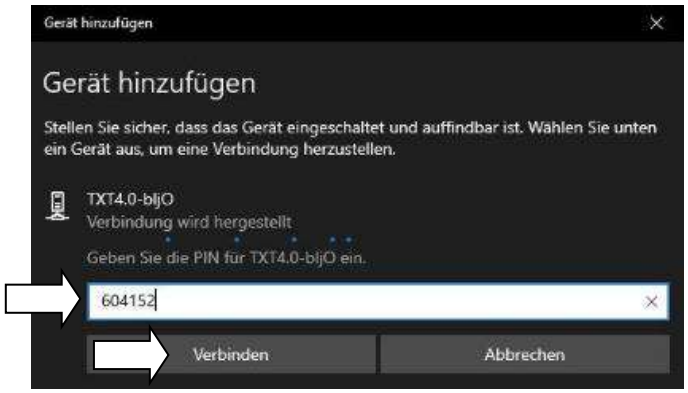
Es wird nach Geräten gesucht.



Wird ein Bluetooth Gerät gefunden wird es angezeigt. Um das Gerät hinzuzufügen, muss man auf den Eintrag kicken.

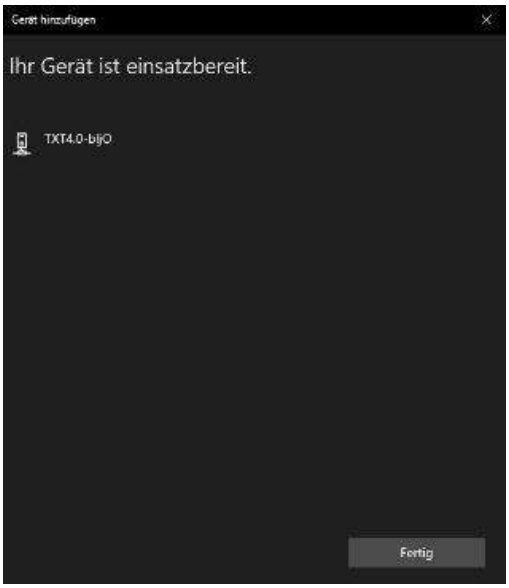


Nun muss der Kopplungscode vom TXT 4.0 Controller eingegeben werden und die Verbindung wird hergestellt.



Hier ist der Kopplungscode ist hier im Beispiel 604152 und nun auf Verbinden klicken.

Nun wird die Verbindung aufgebaut.



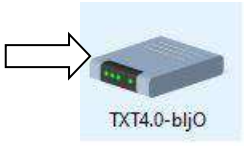
Auf Fertig klicken. Der PC kennt nun den TXT 4.0 Controller.

Im Windows muss man in der Systemsteuerung/

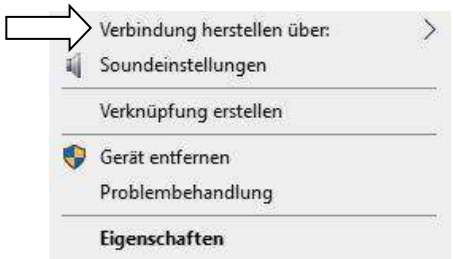
Hardware und Sound …

und dort „Geräte und Drucker“ anzeigen.

Es werden nun alle Geräte angezeigt.

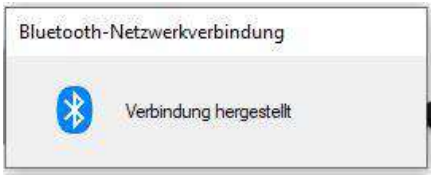


In dieser Liste ist auch der TXT 4.0 Controller. Das „bljO“ ist der Name vom TXT 4.0 und ist von Controller zu Controller unterschiedlich. Auf dieses Symbol mit der rechten Maustaste drücken.



Bei diesem Menü muss man „Verbindung herstellen über“ wählen und dann…

Auf „Zugriffspunkt“ klicken.



Es erscheint ein Statusfenster, dass die Verbindung hergestellt wurde.

Achtung: Diesen Schritt muss man jedes Mal ausführen, wenn der PC ausgeschaltet wurde.
Es besteht keine Verbindung obwohl „gekoppelt“ angezeigt wird.

Nun kann man Robo Pro Coding mit dem TXT 4.0 über BT verbinden.

Dazu auf Verbinden klicken.

Nun muss man auf den rechten Pfeil nach unten klicken.

Es klappt sich dieses Menü auf wo man Bluetooth anklickt.

Es wird eine neue IP-Adresse hier 192.168.9.2 angezeigt.
Die „9“ ist die Bluetooth Verbindung. („7“ die USB, “8“WLAN, eine weitere für die Kamera)

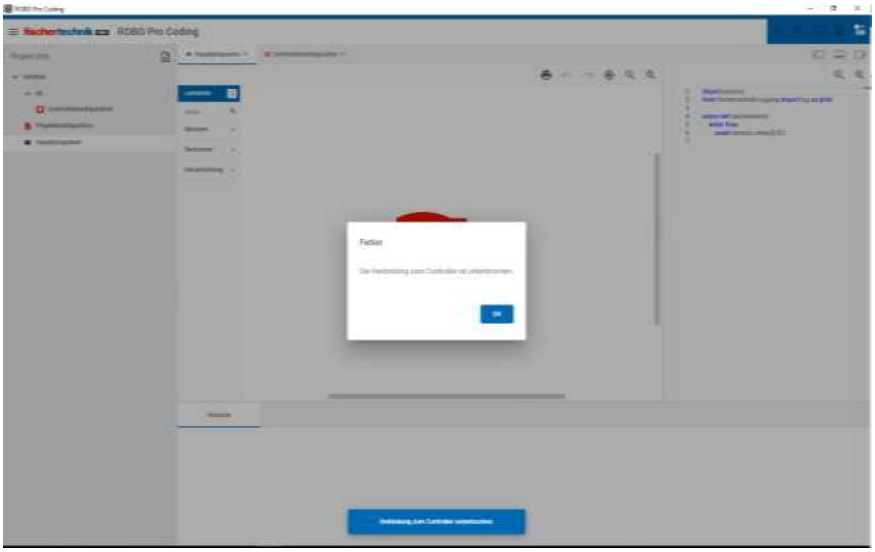
Beim TXT 4.0 Einstellungen /API-Schlüssel ablesen und hier eingeben.
Nach der Eingabe des API-Schlüssels vom TXT 4.0,

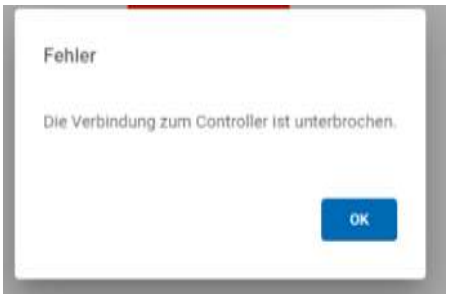
das blau gewordene „Verbinden“ anklicken...
Bei Robo Pro Coding wird oben rechts das Verbindungszeichen kurz gelb und

muss dann grün werden. Auch die anderen Symbole werden nun sichtbar.

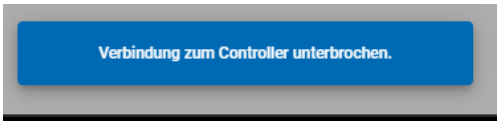
Die Bluetooth Verbindung zwischen PC und TXT 4.0 steht nun.

Wenn keine Verbindung zustande kommt



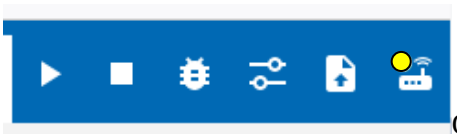


Es erscheint kurz unten dieser Text:



Das Einfachste ist, es einfach noch mal auszuprobieren. Man kann auch die Kabel USB Kabel und die Spannungsversorgung überprüfen. Ursache kann auch der Sperrbildschirm / Bildschirmschoner von Windows gewesen sein, der die Verbindung unterbrochen hat. Es kann auch sein, dass der Aufbau einfach etwas länger dauert.

Dann wird das Symbol gelb:


Gelb = Verbindungsaufbau

Wenn immer noch keine Bluetooth Verbindung zu Stande kommt.

Nicht alle PCs haben Bluetooth. Es kann also sein das es nicht vorhanden ist.

Man kann, gerade auch aus Energiespargründen, Bluetooth im PC abschalten bzw. deaktivieren oder sogar die Treiber deinstallieren. Auch dann kommt keine Verbindung zu Stande. Man muss also Bluetooth aktivieren. Wie das geht, steht meist im Handbuch zum PC.

Bluetooth Verbindung über einen Bluetooth Stick

Ältere Rechner werden meist mit einem USB-Stick um Bluetooth erweitert. Auch da braucht man einen Treiber dazu. Es hat sich in der Vergangenheit gezeigt, dass nicht alle BT-Sticks funktionieren. Windows 7 unterstützt ebenfalls nicht von Hause aus den Bluetooth 4.0 Standard. Man braucht da einen speziellen BT-Stick mit der Bezeichnung: bluegiga, BLED 112-V1.
Auch heutzutage ist es ein Problem, genauso einen Stick zu bekommen. Oft gibt es kompatible Sticks oder man gibt es in die Suchmaschinen ein und trotzdem wird ein anderer BT-Stick angeboten und geliefert. Neuere unterstützen zwar die nun älteren Bluetooth-Standards.

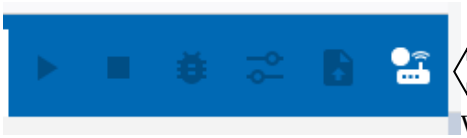
Eine Garantie, dass es auch mit jedem PC läuft, gibt es nicht. Da hilft nur ausprobieren. Ich habe einige Rechner ausprobiert, mit teilweisen unterschiedlichen Windows Versionen.
Die haben alle funktioniert und konnten eine Verbindung aufbauen.

Hanystandort / Tablett Sprachsoftware

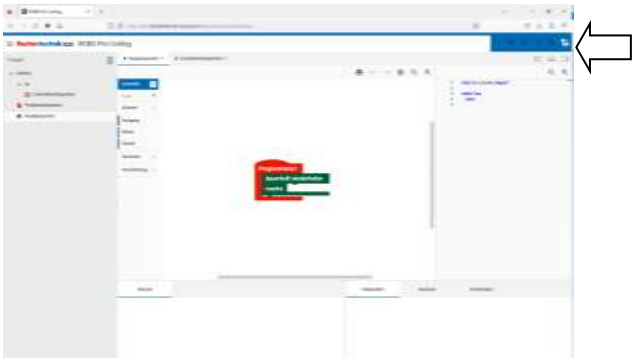
Wenn beim Installieren der "Sprachsoftware" von fischertechnik (nicht Robo Pro Coding) Probleme auftauchen und beim Verbindungsaufbau nicht klappt. Sollte man überprüfen, ob die Funktion Standort im Handy/Tablett aktiviert ist. Die muss beim jeweiligen ersten Starten aktiviert sein. Jedes Mal. Sie wird zwar nicht genutzt und kann danach deaktiviert werden, aber beim Starten der Software muss es aktiv sein. Warum auch immer.

RX Controller verbinden

USB Kabel



Weiß = keine Verbindung



Auf dieses Symbol klicken.

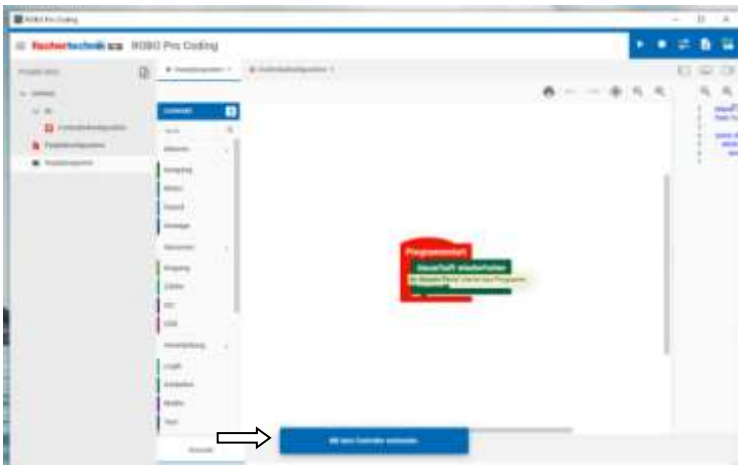
Der fischertechnik RX-Controller meldet sich momentan als „CircuitPython CDC2 control“ an der USB Schnittstelle vom PC an. Es wird aber weiter gescannt.

Nun muss man auf den blau geschriebenen Namen klicken.

Gelb = Verbinden

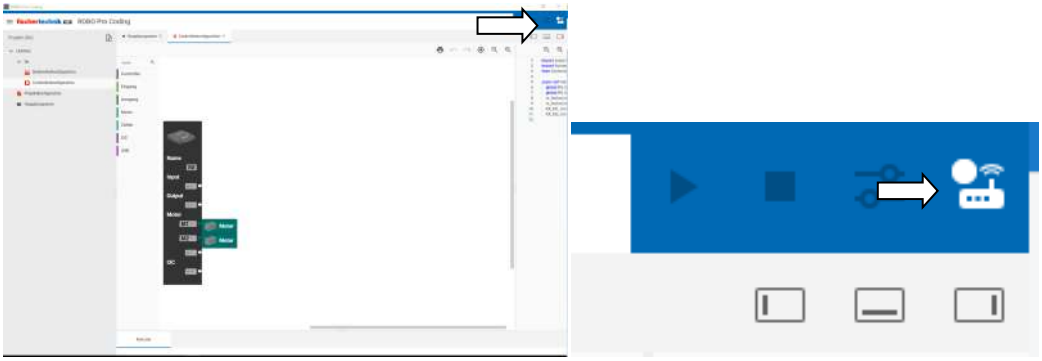
Nach einer kurzen Wartezeit ändern sich die Symbole und der RX Controller ist installiert.


Grün = Verbunden und die anderen Symbole sind sichtbar.

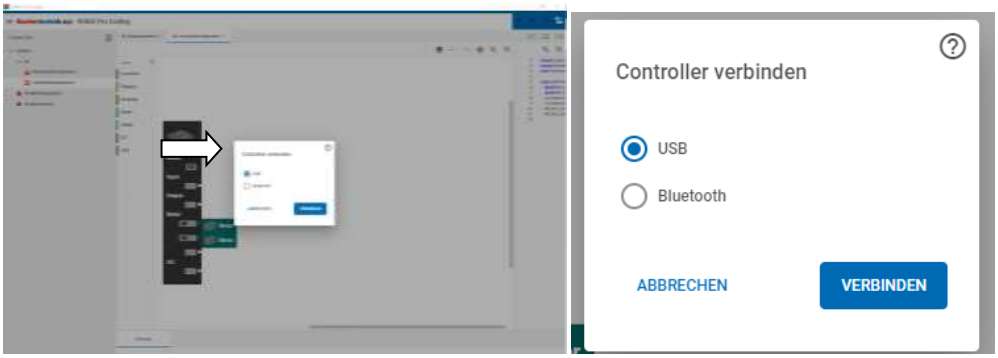


Es wird unten kurz angezeigt, dass der PC nun mit dem Controller verbunden ist.

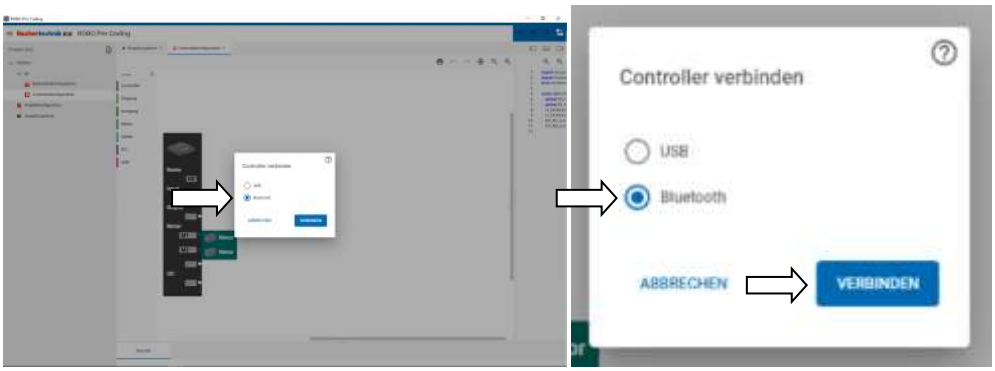
RX mit PC über Bluetooth Verbinden
RX Controller einschalten. LED leuchtet rot.



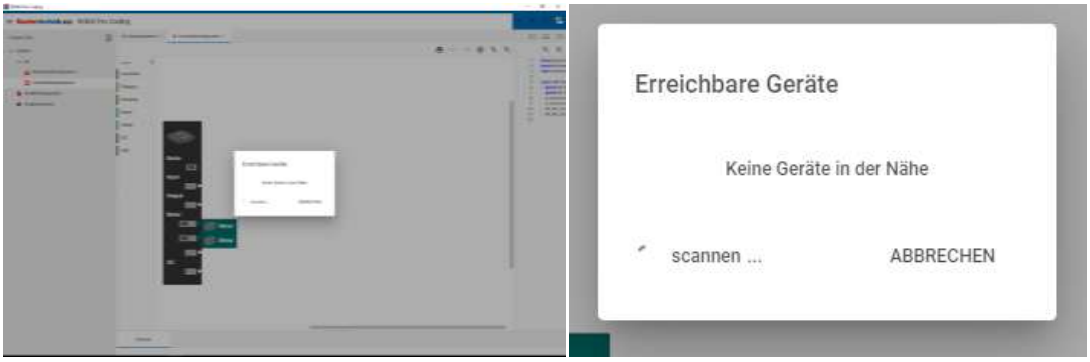
Oben rechts auf Verbinden



Das "Controller verbinden" Fenster erscheint.

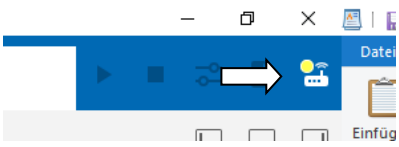


Hier auf Bluetooth klicken und auf das blaue Verbinden.

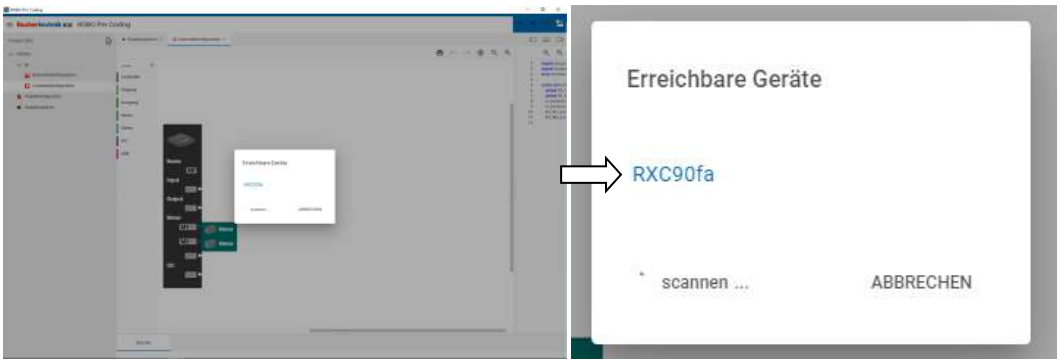


Es wird nach Geräten in der Nähe gescannt.

Die LED am RX Controller leuchtet rot. Nun auf den roten Bluetooth Taster rechts neben dem Einschaltknopf vom RX Controller drücken, bis die LED blau wird. Die blaue Led fängt nun langsam an zu blinken (Verbindungsaufbau).



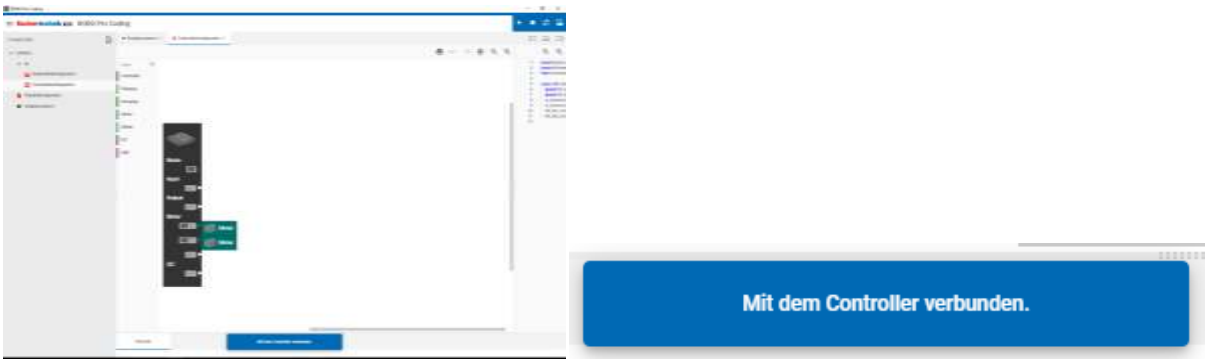
Gelbes Verbindungszeichen für Verbindungsaufbau in Robo Pro Coding erscheint.



Nach kurzer Zeit erscheint dieses Fenster.

Ein "RXC90fa" ist gefunden. oben das Verbindungszeichen ist gelb.

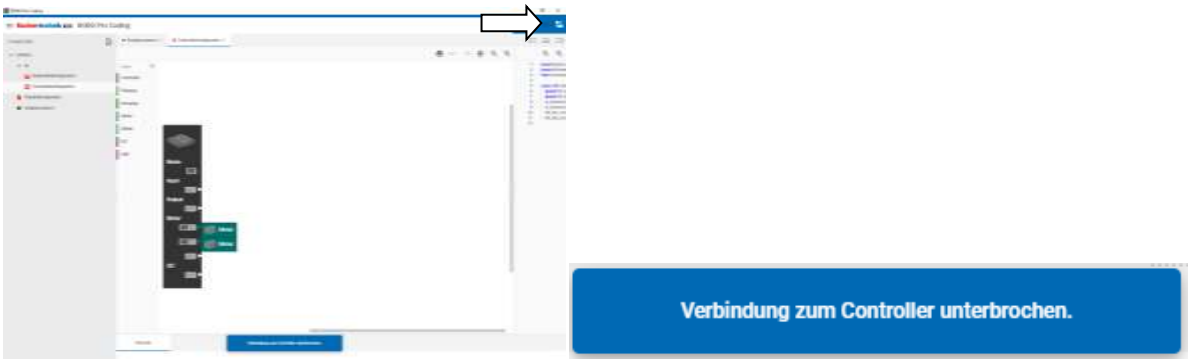
Nun auf das RXC90fa klicken.



Es erscheint kurz unten eine Meldung, dass der RX Controller nun verbunden ist. Die blaue LED am RX Controllerblinkt etwas schneller und zeigt damit an, dass auch er eine Verbindung hat.

Nun kann man mit dem RX Controller arbeiten.

Verbindung trennen
Durch Drücken auf das "Verbindungszeichen" oben rechts kann man eine bestehende Verbindung wieder trennen.



Das Zeichen wird Weiß und die anderen Symbole werden ausgegraut. Es wird kurz diese Meldung unten angezeigt. Auch beim Beenden von Robo Pro Coding wird die Verbindung getrennt und die Ausgänge werden abgeschaltet.

Hinweis: Unterschied vom Verbinden Robo Pro Coding – RX und PC - RX
Die Verbindung zwischen PC und RX Controller wird nicht mit dem Betriebssystem gemacht. Es reicht es nur mit Robo Pro Coding zu machen. Später, wenn man auch mit anderen Programmiersprachen auf den RX Controller zugreifen will muss man den in das Betriebssystem einbinden. Wie den TXT 4.0 Controller.

Achtung!

Wir ein Controller gewechselt z.B. zu einem BT Smart Controller, wird die Verbindung von Robo Pro Coding getrennt! Die Bluetooth-Verbindung bleibt aber bestehen. Siehe auch blinken der LED am RX Controller. Die Verbindung scheint nun nur mit dem Betriebssystem zu sein.

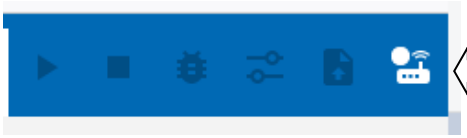
Debugging RX Controller
Momentan funktioniert das Debugging mit dem RX Controller nicht.

Tipp: LED-Farben und Blinken am RX Controller

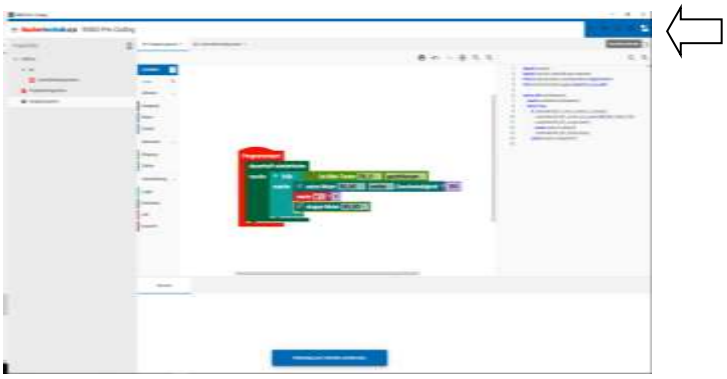
- Keine Verbindung zum PC, kein laufendes Programm = LED am RX Controller leuchtet dauernd rot
- RX-Controller breit = LED blinkt blau
- RX-Controller Verbindung über BT = LED blinkt schneller blau
- RX-Controller Programm läuft = LED leuchtet dauernd
- Geladenes Programm starten (roten Knopf drücken) = Programm läuft LED ist grün

BT Controller verbinden

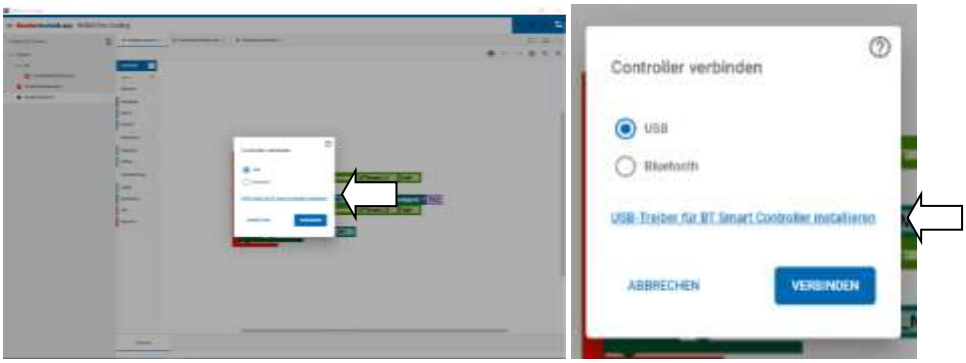
USB Kabel



Man drückt oben rechts auf Verbinden.

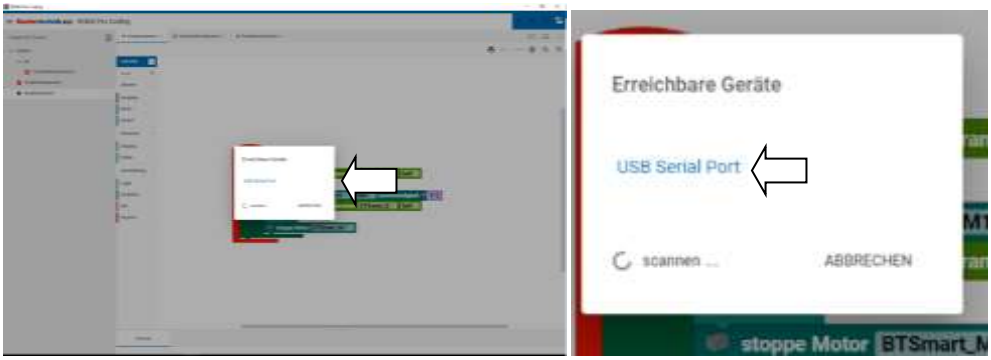


Es kann sein, dass man aufgefordert wird einen Treiber für den BT Smart Controller zu installieren.

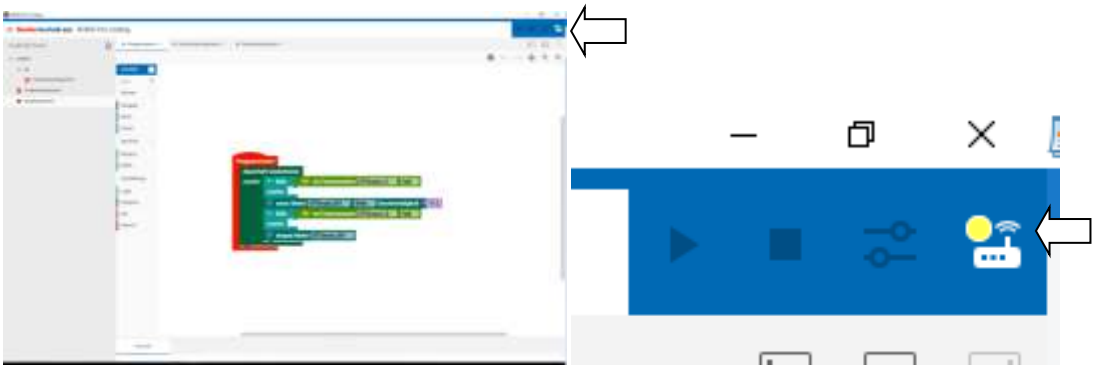


Dazu findet sich in dem Feld ein blauer Text „BT Smart Controller installieren“, den man anklicken muss.

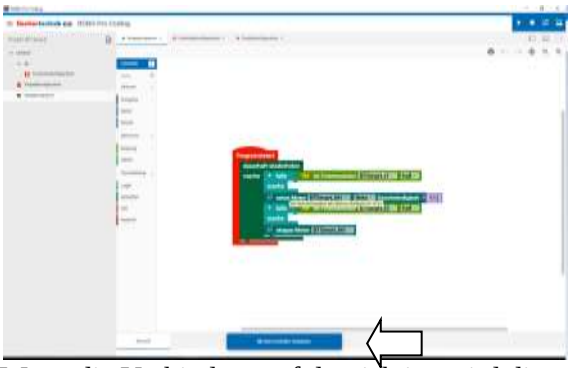
Wenn schon geschehen, kann man auf „Verbinden“ klicken.

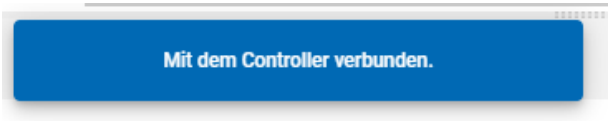


Hier „USB Serial Port“ anklicken. Das ist der Name für den BT Smart Controller



Das Symbol wird beim Verbindungsaufbau gelb.

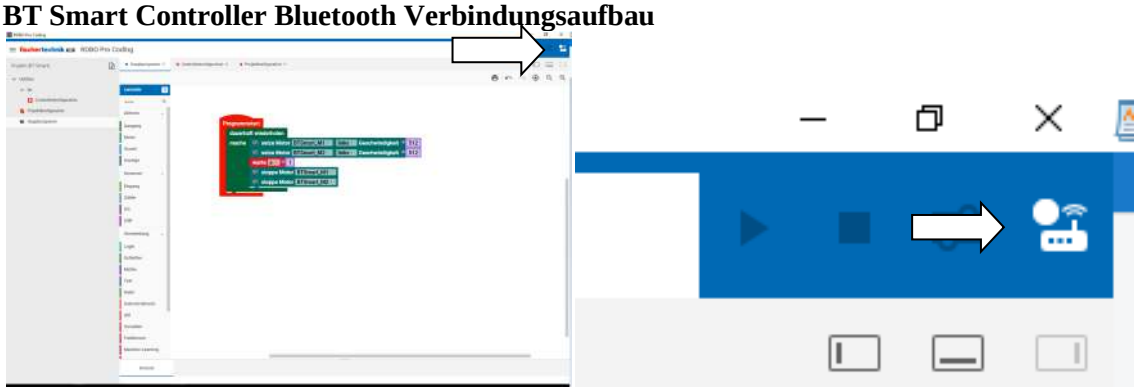




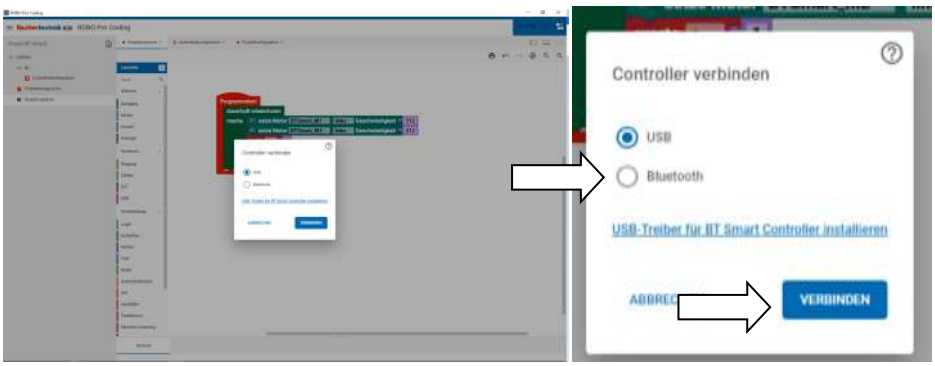
Wenn die Verbindung erfolgreich ist, wird diese blaue Meldung kurz angezeigt und...

das Symbol wird Grün und die anderen Symbole sichtbar.

BT Smart Controller Bluetooth Verbindungsaufbau



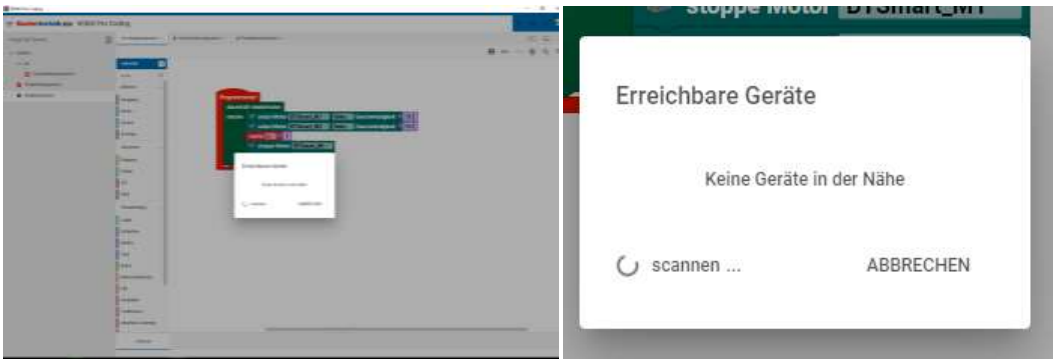
Oben rechts auf das Verbinde Symbol im blauen Feld klicken



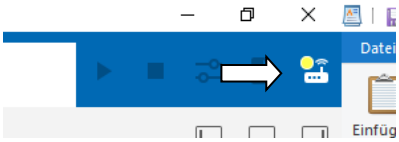
Hier nun auf Bluetooth klicken.

Eventuell, wenn nicht schon geschehen, den Treiber für den BT Smart Controller installieren. Dazu auf die blaue Schrift „USB Treiber für BT Smart Controller installieren" klicken.

Und dann auf das blaue "Verbinden" klicken.

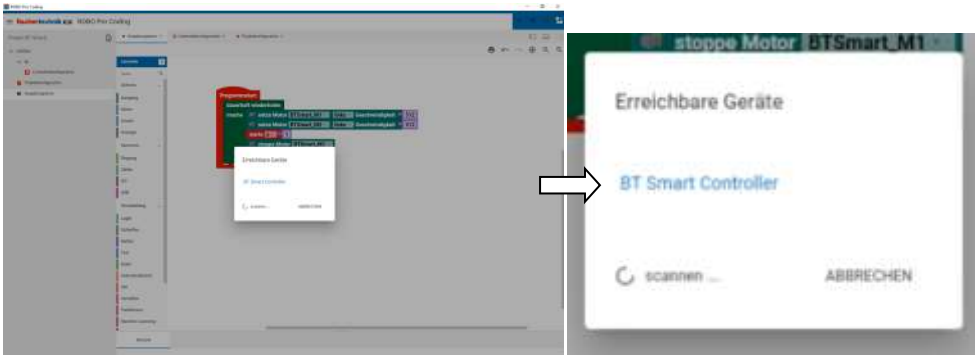


Jetzt scannt der PC nach Bluetooth Geräten in der Umgebung.

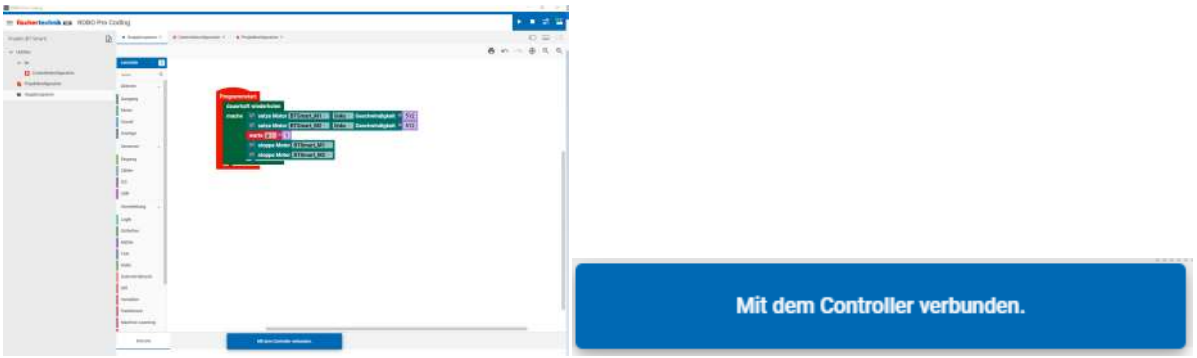


Gelbes Verbindungszeichen für Verbindungsaufbau in Robo Pro Coding erscheint.

Am BT Smart Controller leuchtet die Grüne LED (=Spannungsversorgung ist da) und die blaue LED blinkt. Nun den Select Taster ca. 3 Sekunden drücken, bis die blaue LED schneller blinkt.



Nun erscheint dieses Fenster und man muss auf die blaue Schrift "BT Smart Controller" klicken.



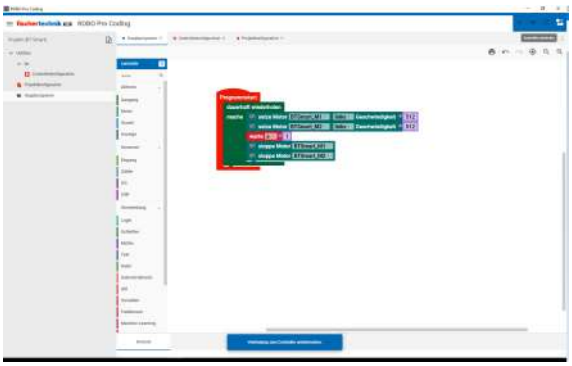
Es erscheint kurz diese Meldung und in Robo Pro Coding wird das Verbindungssymbol oben rechts Grün.

Die blaue LED leuchtet dauernd.

Der BT Smart Controller ist nun mit Robo Pro Coding verbunden.

Bluetooth Verbindung trennen

Um die Verbindung zu trennen, kann man auf das Verbinde Symbol oben rechts im blauen Feld in Robo Pro Coding klicken.



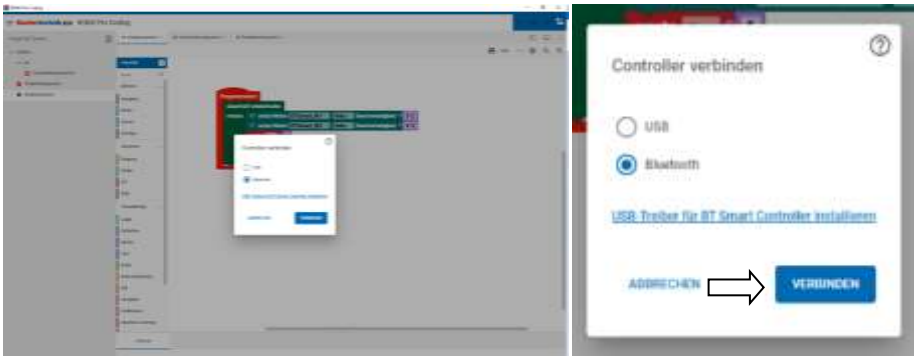
Eine kurze Meldung erscheint und die Verbindung ist unterbrochen.

Die blaue LED am BT Smart Controller blinkt.

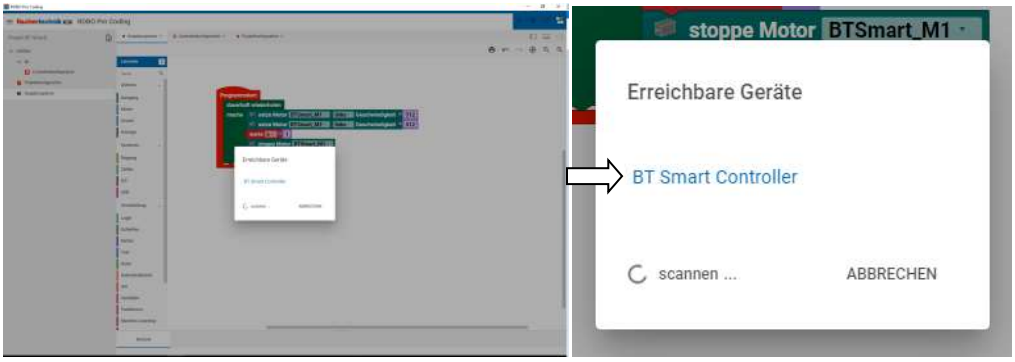
Man kann auch durch Beenden von Robo Pro Coding die Verbindung trennen.

Erneute Bluetooth Verbindung.

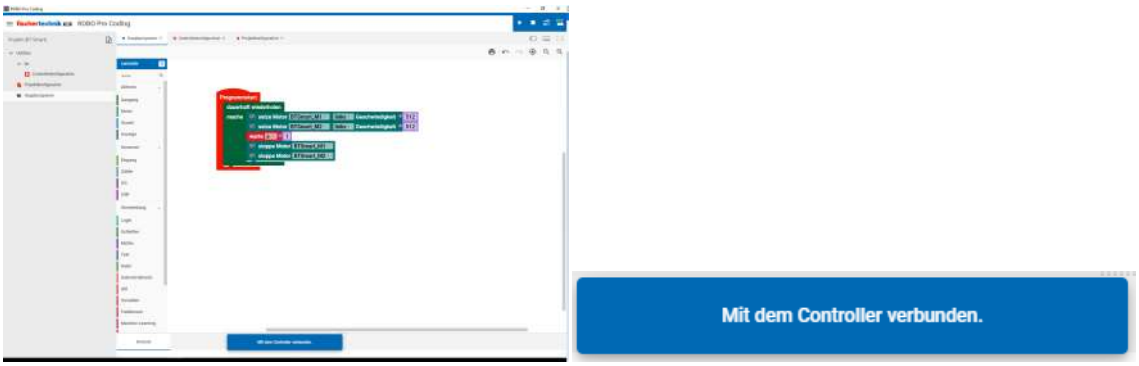
Wenn man nun noch mal bei Robo Pro Coding auf Controller verbinden klickt.



Erscheint wieder dieses Fenster und man klickt auf Verbinden und

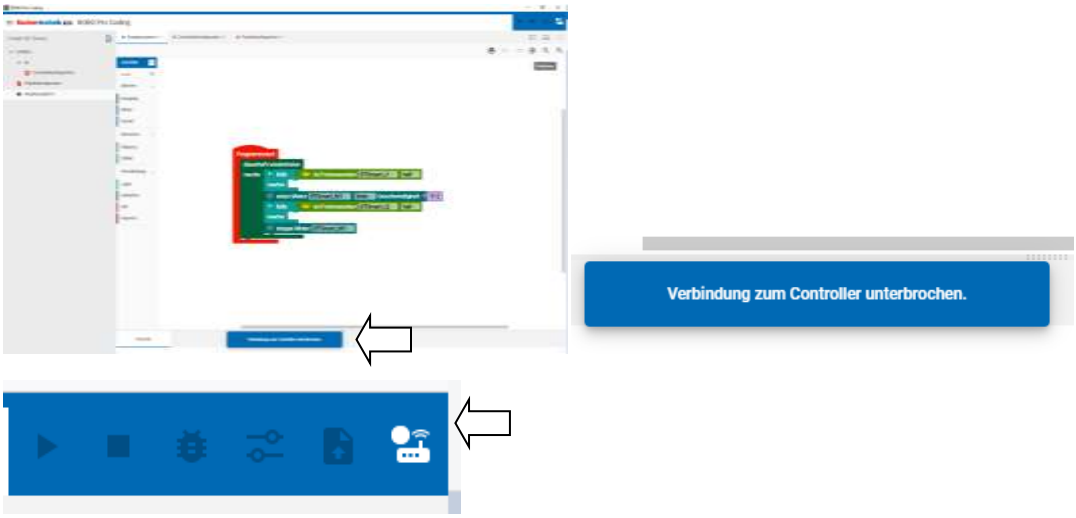


wieder auf die blaue Schrift und nach kurzer Zeit erscheint



diese Meldung und der Controller ist wieder verbunden, ohne noch mal auf den Select Taster am BT Smart Controller zu drücken.

Wenn keine Verbindung zu Stande kommt.



Wenn es ein Problem gibt und die Verbindung nicht zustande kommt, wird der Verbindungsaufbau unterbrochen und das Symbol wird Weiß und die anderen bleiben ausgegraut. Meist hilft es die Spannungsversorgung kurz zu unterbrechen, 21, 22, 23… und dann wieder anschließen. Nun den Verbindungsaufbau wiederholen.

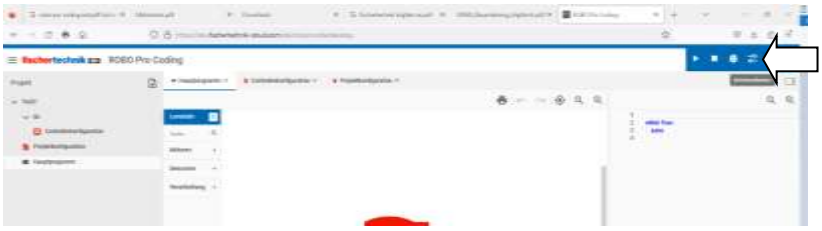
Blinken am BT Smart Controller

Grüne LED	blinkt sehr schnell 4 mal pro Sekunde	Kurzschluss an einem der Ausgänge
Grüne LED	leuchtet	Controller mit Spannung verbunden
Blaue LED	leuchtet	PC mit Controller verbunden
Blaue LED	blinkt schnell	Verbindung über Bluetooth wird aufgebaut (peering)

Schnittstellentest (=Controllertest / Interfacetest) (Blaue Kopfzeile)

Um zu testen, ob ein Eingang oder Ausgang funktioniert, kann man den Schnittstellentest verwenden.

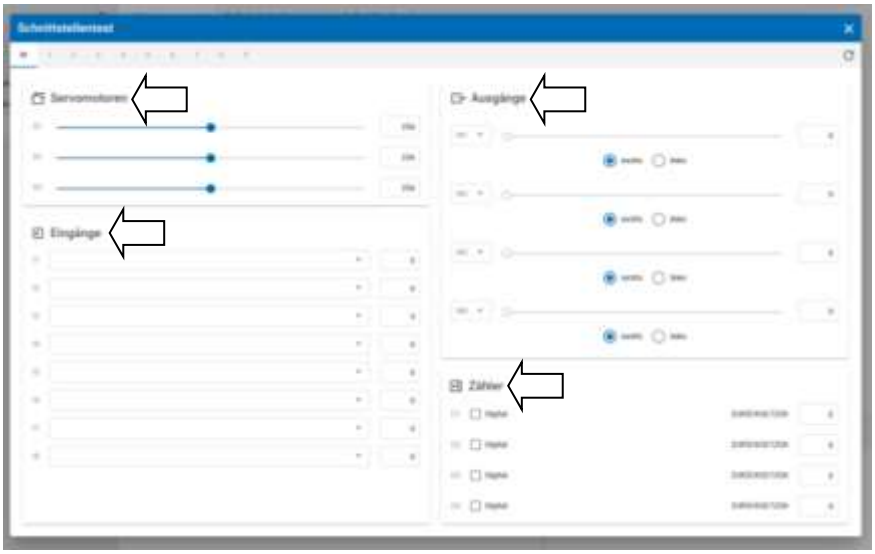

Schnittstellentest



Wenn man ihn anklickt, kommt ein Fenster, dass bei jedem Controller anders aussieht.

Controller

TXT 4.0 Schnittstellentest



Wie beim TXT 4.0 Controller sind im Fenster die Servomotoren, Ein- und Ausgänge und die Zähler angeordnet.

Servomotoren

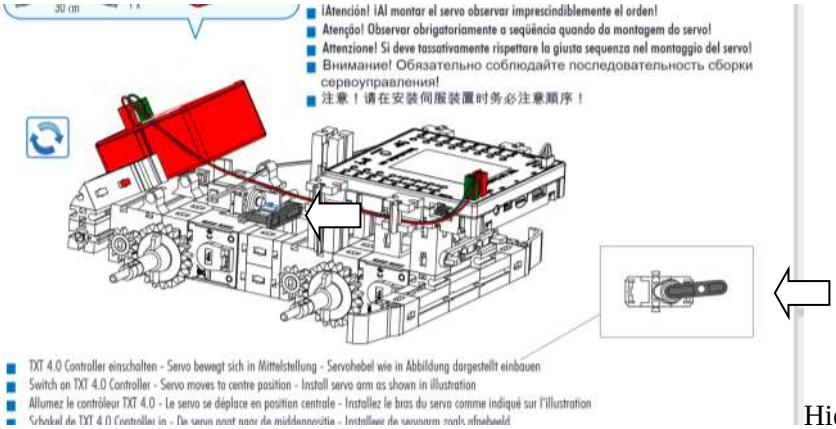
Ausgänge

Eingänge

Counter/Zähler

Servomotoren (Modellbauservos) Schnittstellentest

Die Servomotoren sind in der Mittelstellung. Durch Verschieben der Regler, kann man die Stellung des Servohebels verändern.
Wichtig ist die Stellung des Einbaus des Hebels. Sie ist in jeder Bauanleitung angegeben. Hier beim Malroboter vom fischertechnik Hightech Kasten:



Hier ist der Hebel falsch rum.

Im Regelfall sollte die Position passen. Es kann aber in Ausnahmefällen sein, dass die Nullstellung vom Modell nicht erreicht werden kann, weil im Modell der Hebel an andere Bauteile anschlägt. Da sollte man den Hebel vor dem Schnelltest abziehen. Auch kann es sein, dass der Winkel, den der Hebel maximal drehen kann, kleiner ist als der Regler hergibt. Dann stößt der Hebel an andere Bauteile an. Das sollte man unbedingt vermeiden.

Die „Sache“ mit dem Schnittstellentest und den Servomotoren.

Es gibt den seltenen Fall, das die Servomotoren ungünstig im Modell stehen. Da hat man im Programm es so eingestellt, das genau passt und die Hebel nicht an Hindernisse anstoßen – aber – wenn man den Schnittstellentest aufruft, fahren die Hebel in die Nullstellung und in dem Fall gegen ein Hindernis. Ich habe das einmal bei einem selbstentworfenen Modell gehabt und man muss darauf aufpassen. Am besten ist es halt, wenn keine Hindernisse da sind. Es gibt aber auch Modelle wie z.B. die Beine einer Krabbe, wo dann das Aufstehen „schwer“ ist, weil alle Beine z.B. waagerecht sind. Um zu verhindern, dass die Servomotoren nach dem Beenden vom Schnittstellentest sich auf die Mitte bewegen, hilft nur das Abschalten der Versorgungsspannung der Servos *vor* dem Beenden des Schnittstellentest.

Hinweis: Modellbauservos brauchen manchmal sehr viel Strom. Es ist „möglich“, auch nur die Steuerimpulse zu nutzen, um z.B. größere Servos zu benutzen oder viele gleichzeitig. Die Stromversorgung vom TXT 4.0 ist für so etwas nicht ausgelegt. Sie kann nur 3 Miniservos versorgen.

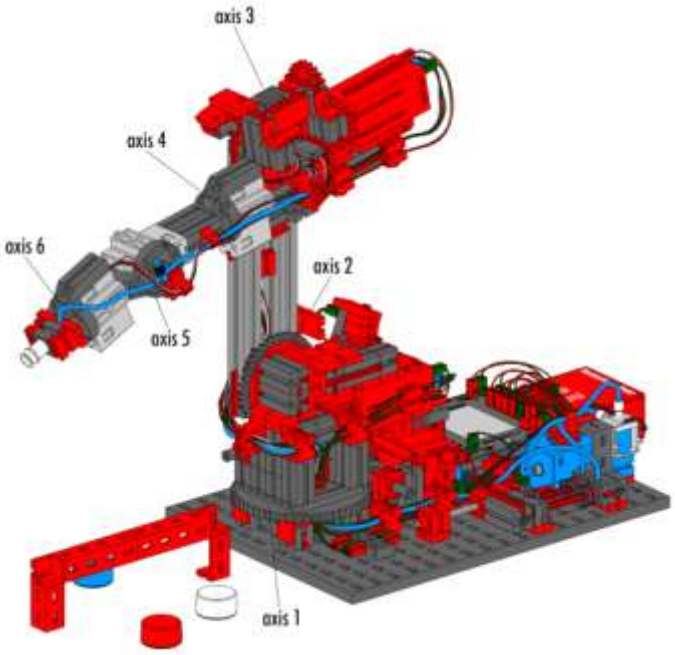
Unterschied Analog- zu Digital-Modellbauservos

Fischertechnik hat zwei verschiedene Servos. Das (kleine) analoge Servo, wird für Lenkungen z.B. bei Fahrzeugen eingesetzt. Bei dem Add ON Industrial Robots sind es (größere) digitale Servos. Die digitalen Servos, behalten die Position bei, wenn sie keine Steuersignale mehr bekommen. Analoge Servos nicht. Die Geschwindigkeit ist bei dem digitalen Servo auch größer. Eine andere Position wird ruckartig angefahren.

Servokalibrierung am TXT 4.0

An dem TXT 4.0 können die maximale und minimale Grenze der Servos 1, 2 und 3 eingestellt werden. Wählen hierzu auf dem Display des TXT 4.0 Controllers / Einstellungen /Servo.

TXT 4.0 Einstellung Servos



6-Achsen Roboter aus AddOn: Industrial Robots

Über + und - kann die maximale und minimale Stellung des Servos in µs eingestellt werden.
Mit R kann der jeweilige Wert auf den Standardwert zurückgestellt werden.

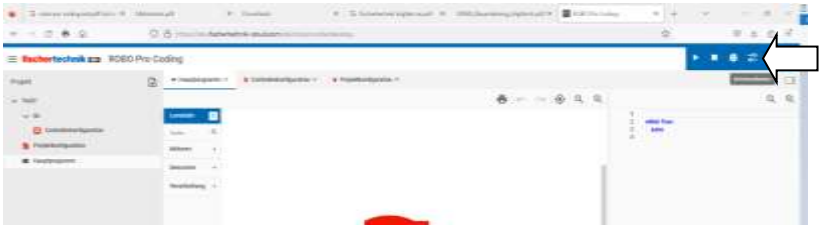
Die Mittelstellung des Servos ist der Mittelwert aus der minimalen und maximalen Grenze. Somit kannst du über das Einstellen der minimalen und maximalen Grenze die Mittelstellung des Servos einstellen. Die eingestellten Werte können über die drei Buttons MIN, MID und MAX angefahren und überprüft werden.

Die eingestellten Werte werden bei Programmen, bei welchen die Servos programmiert sind, angewendet. Beim Ausschalten des TXT 4.0 Controllers bleiben die eingestellten Werte gespeichert.

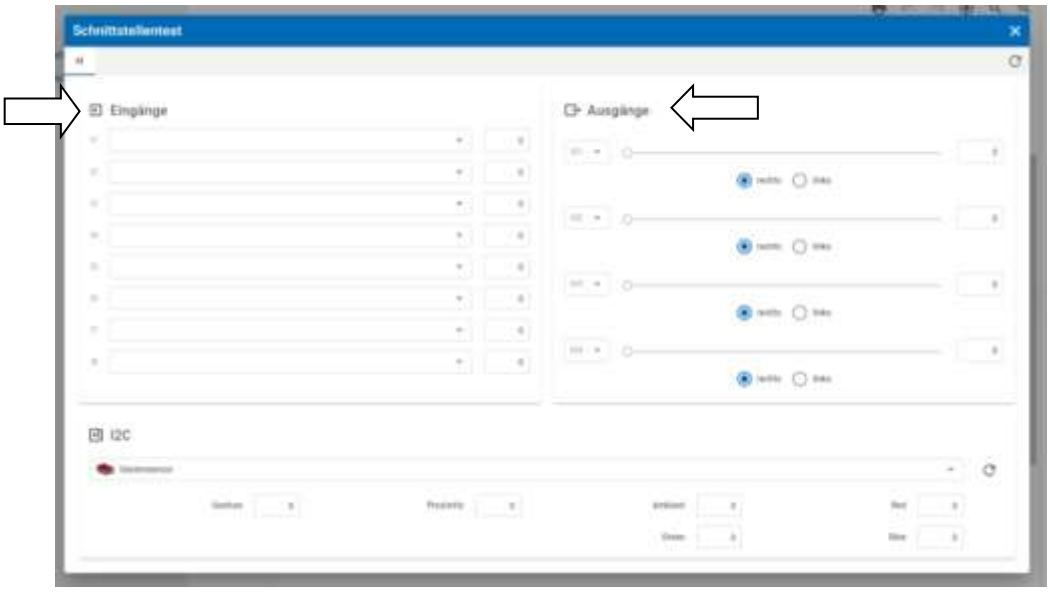
Tipp: Wenn die Servopositionen bei einem neugebauten Modell nicht passen, kann es daran liegen, dass von einem älteren Modell die Einstellzeiten noch da sind. Man muss sie nur einfach mit „R“ wieder zurücksetzen.

Beim 6-Achsen Roboter vom fischertechnik Kasten AddOn: Industrial Robots, kann man die Achsen 4,5 und 6 damit einstellen. Die Achsen 1 bis 3 sind Encodermotoren, mit Taster an der Nullposition.

RX-Controller Schnittstellentest
Um zu testen, ob ein Eingang oder Ausgang funktioniert, kann man den Schnittstellentest verwenden.



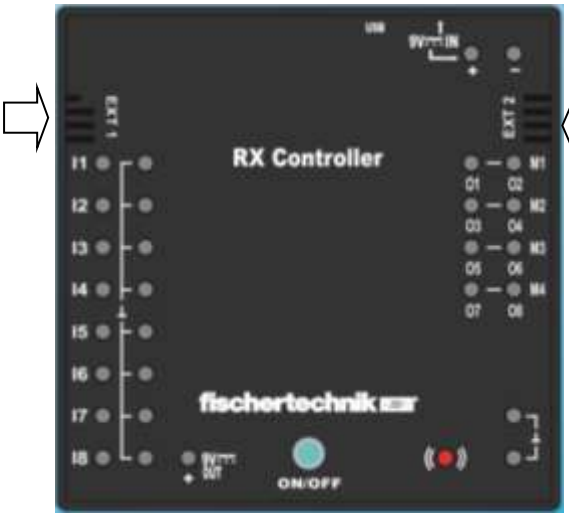
Wenn man den anklickt, kommt folgendes Fenster:



Am I2C ist der Gestensensor voreingestellt und zeigt die umgerechneten Sensorwerte an (erste Ansicht). Wie auch beim TXT 4.0 entspricht die Darstellung der Ein- und Ausgänge auf dem Bildschirm, der Anordnung auf dem RX Controller. Die Eingänge befinden sich links und die Ausgänge auf der rechten Seite.

Eingänge Ausgänge

Beim Schnittstellentest vom RX Controller, kann man wie beim TXT 4.0 die Eingänge abfragen und einstellen. Auch die Motoren und die O-Ausgänge lassen sich einstellen. Nicht vorhanden sind die Servoausgänge und die schnellen Zähler. In der Anzeige unten ist noch der I2C Anschluss zu sehen. Dort kann man den Sensor auswählen und die entsprechenden werden Werte angezeigt.

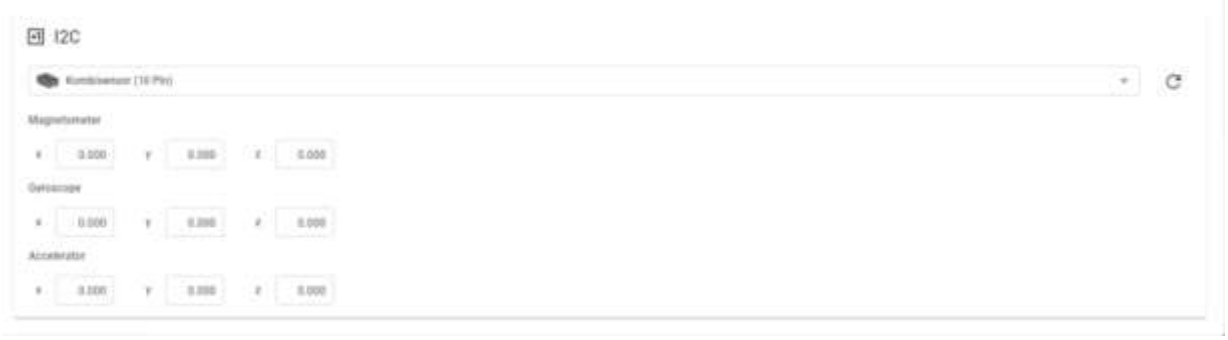


Hier oder auf der gegenüberliegenden Seite kann man den I2C Sensor anschließen.

I2C Ansicht des ausgewählten Sensors:

Gestensensor

Umweltsensor



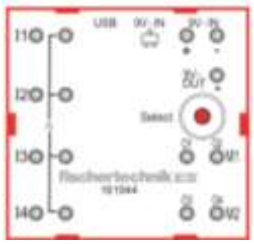
Kombisensor

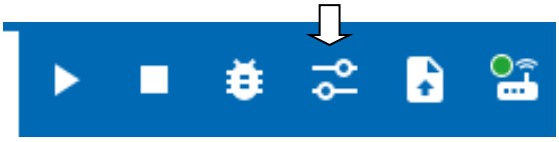
Ergänzung RX Schnittstellentest und Programm starten

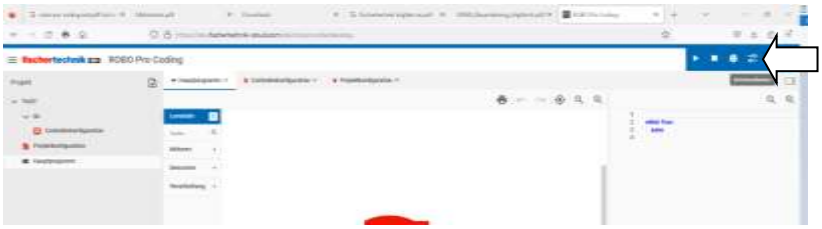
RX Controller. Wenn eine Bluetooth-Verbindung über den Schnittstellentest besteht, kann man trotzdem Programme starten und das Schnittstellentest-Fester bleibt offen. Es erfolgt aber kein weiterer Datenaustausch zwischen PC und RX Controller. Nach der Beendigung des Programms, kann man sofort Motoren ansteuern und Taster einlesen.

BT Smart Controller Schnittstellentest

Um zu testen, ob ein Eingang oder Ausgang funktioniert, kann man den Schnittstellentest verwenden. Der BT Smart Controller hat die vier Eingänge und die zwei Ausgänge.

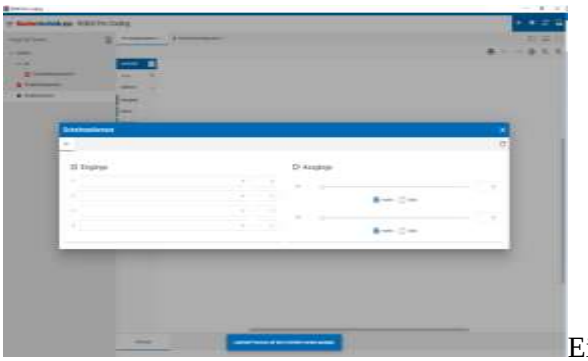


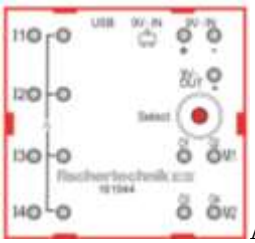




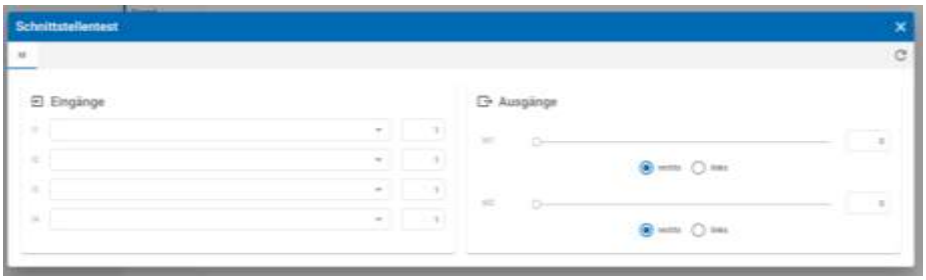
Wenn man den anklickt, kommt folgendes Fenster:

Der Schnittstellentest für den BT Controller



Eingänge 

Ausgänge



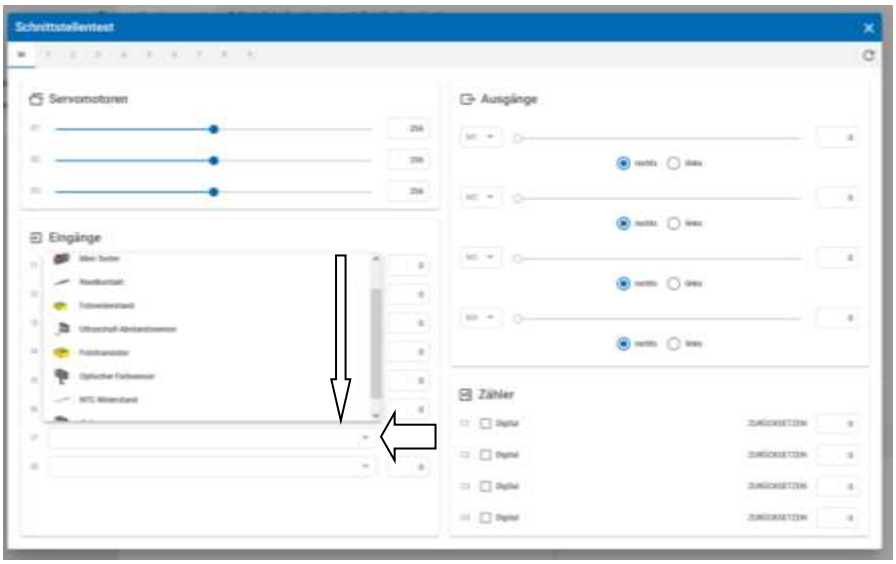
Die Eingänge sind links du die Ausgänge rechts – wie beim BT Smart Controller.

Beim Starten des Schnittstellentests werden laufende Programme gestoppt und dann die Ein- und Ausgänge angezeigt. Diese Meldung wird kurz angezeigt.

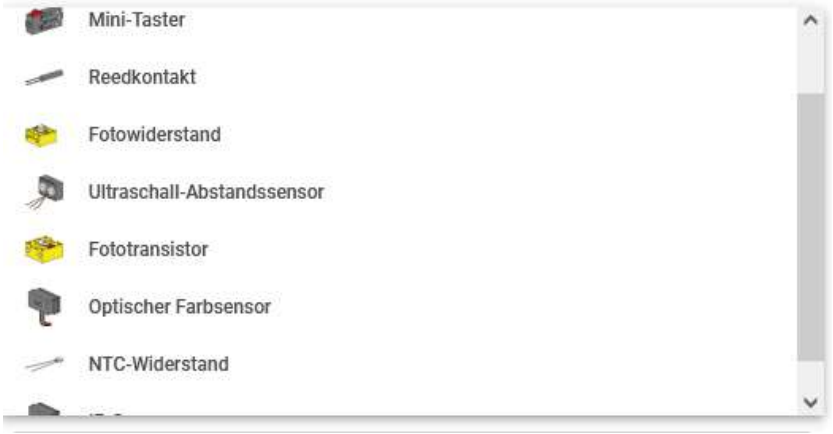
Eingänge (Input-Sensoren) Schnittstellentest

An die Eingänge I1 bis I8 können folgende fischertechnik Sensoren angeschlossen werden:
Mini-Taster
Reedkontakt
Fotowiderstand
Ultraschall-Abstandssensor
Fototransistor
Optischer Farbsensor
NTC-Widerstand
IR-Spurensensor

Die Auswahl erfolgt mit dem kleinen Dreieck neben dem I1...

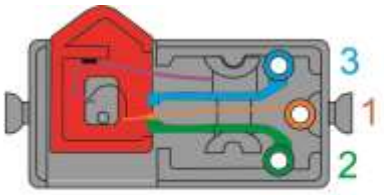


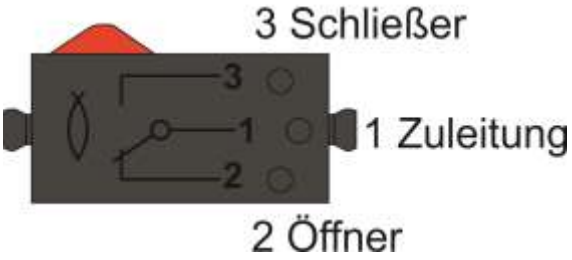
Durch die Auswahl werden ankommenden Daten oder Messwerte vom Controller und Robo Pro Coding in die richtigen Werte umgerechnet und dem Programm übergeben. Die Werte des Sensors werden rechts neben der Auswahl angezeigt.



Mini-Taster
Bei einem Mini-Taster sind das 0 oder 1. Beachte, dass der Taster als Öffner oder als Schließer betrieben werden kann. Je nachdem wie er am Taster angeschlossen ist.

Taster	nicht gedrückt	gedrückt
Schließer	0	1
Öffner	1	0





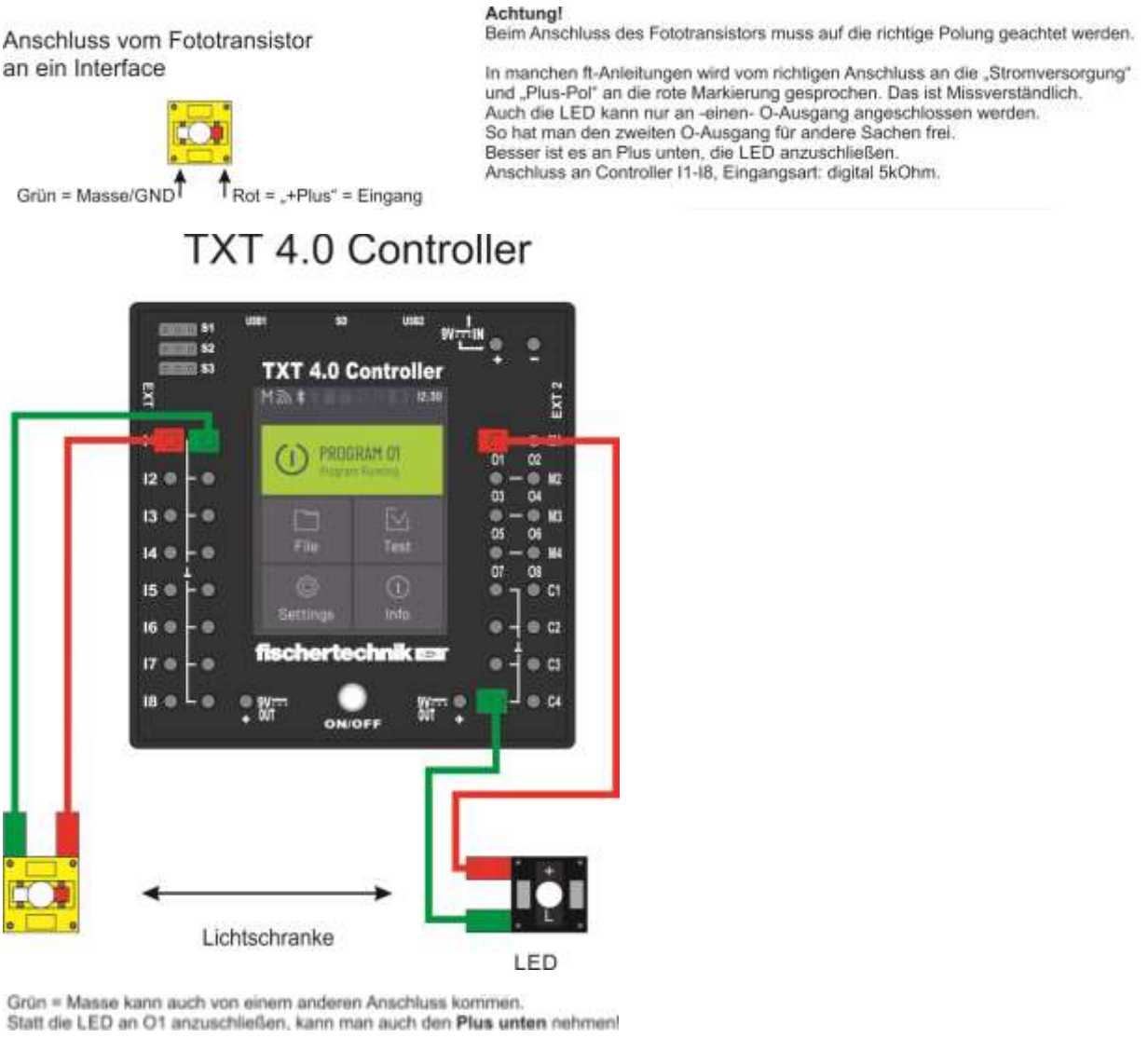
Reedkontakt
Der Reedkontakt ist ein Taster, der auf Magnete reagiert. Ist ein Magnet in der Nähe schließt er den Kontakt und es kommt eine 1 am Eingang.

Fotowiderstand
Der Fotowiderstand, ist ein veränderbarer, analoger Eingang. Je nachdem wie viel Licht auf ihn fällt, ändert er seine Eigenschaft und lässt mehr oder weniger Strom durch. Der umgerechnete Widerstandswert wird angezeigt. Wenn keiner angeschlossen ist, wird der maximale Wert angezeigt, weil kein Strom fließt.

Ultraschall-Abstandssensor
Der Ultraschall-Abstandssensor hat neben der Spannungsversorgung (Rot/+ und Grün/-) ein weiteres Kabel, das an den Eingang angeschlossen wird. Die umgerechneten Abstandswerte werden angezeigt. Die ganz alten Ultraschall-Abstandssensoren vom Robo-Interface funktionieren nicht.

Fototransistor

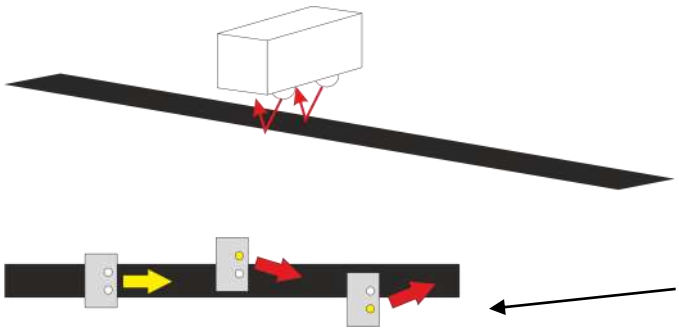
Beim Fotowiderstand muss man darauf achten wie rum man den Eingang anschließt. Der rot markierte Anschluss vom Fototransistor, muss an den linken Anschluss vom Eingang z.B. I1 und der nicht markierte Anschluss vom Fototransistor, an Minus oder dem rechten Anschluss vom z.B. I1 . Sonst funktioniert er nicht. Als Minus kann man sich auch jeden anderen Minus vom TXT 4.0 nehmen z.B. vom Zählereingang oder einem anderen Eingang. Das spart manchmal Kabel.



Optischer Farbsensor
Der Optischer Farbsensor hat auch zwei Kabel für die Spannungsversorgung (Rot/+ und Grün/-) und ein weiteres Kabel, das den Farbwert überträgt. Es wird ein rotes Licht ausgesendet, und je nachdem wie viel reflektiert wird, entspricht das dem Farbwert 0 bis 2000. Das muss man meist ausprobieren und ist sehr stark Fremdlicht abhängig. Man sollte bei solchen Messungen immer den Sensor gut Abdecken.

NTC-Widerstand (Temperaturwiderstand)
Wie beim Fotowiderstand, ist der NTC-Widerstand ein veränderbarer, analoger Eingang. Je nachdem wie hoch die Temperatur ist, verändert er seinen Widerstand und der umgerechnete Wert wird angezeigt.

IR-Spurensensor (Spurenfolger)
Der IR-Spurensensor hat auch zwei Kabel für die Spannungsversorgung (Rot/+ und Grün/-) und zwei weitere Kabel, die an zwei Eingänge gehen. In dem Sensor sind unter anderem zwei infrarot LEDs und zwei Fototransistoren verbaut. Sie funktionieren fast wie Lichtschranken und reagieren auf das Licht was reflektiert wird. So kann man feststellen, ob der Untergrund Schwarz ist oder eine andere Farbe hat. Je nachdem wie breit die schwarze Linie ist, kann man damit messen, ob man sich auf der Linie befindet oder ob die schwarze Linie vorhanden ist, die z.B. einen Bereich umrandet.
Tipp: Man kann auch Bauteile z.B. auf einem Fließband erkennen.



Funktionsweise:

Wie zwei unabhängige (Reflektions-) Lichtschranken. Bei einem schwarzen Untergrund wird kein IR-Licht reflektiert. Somit ist der Eingang 0. Bei weißem Untergrund „schaltet“ die Lichtschranke. Somit ist der Eingang 1.

Gegenlenken, wenn ein Eingang 1 wird.

Ausgänge Schnittstellentest

Die Ausgänge werden als M= Motor oder als O= Out bezeichnet. Ein M-Ausgang hat zwei O Ausgänge.

M1 = O1 und O2
M2 = O3 und O4
M3 = O5 und O6
M4 = O7 und O8

Wenn ein M-Ausgang von einem Motor besetzt ist. Können die entsprechenden O Ausgänge nicht benutzt werden. Über Rechts/Links kann man bei einem Motor die Drehrichtung ändern. Die Geschwindigkeit, mit der sich der Motor drehen soll bzw. die Helligkeit der LED, wird mit dem Zahlenwert 0 bis 512 eingestellt.

Wenn ein Motor schlecht läuft oder eine Lampe noch leuchtet, obwohl sie im Programm aus ist, kann es sein, dass man die beiden O-Ausgänge zu einem Motor benutzt hat.

Hinweis: Obwohl auf dem BT Smart Controller O1-O4 Ausgänge sind, können die nur als M1-M2 Ausgang programmiert werden.

Zähler Schnittstellentest

Die Zähler C1 bis C4 sind schnelle Zähler. Im Gegensatz zu den normalen Eingängen können sie sehr viele Impulse verarbeiten. Sie spielen auch beim Gleichlauf von mehreren Motoren und bei Wegstrecken eine Rolle. Den Zahlenwert kann man wieder auf 0 setzen.

Man kann diese Eingänge aber auch als ganz normale Eingänge benutzen, z.B. für Taster.
Im Programm ist es dann z.B. C1 statt des normalen Eingangs I1.

Man braucht also Encodermotoren mit den zusätzlichen Leitungen oder Motoren und einem Taster, der die Impulse macht mit Leitungen. Die Impulse werden dann z.B. von der Achse eines Bandes über ein Impulszahnrad an den Taster erzeugt.

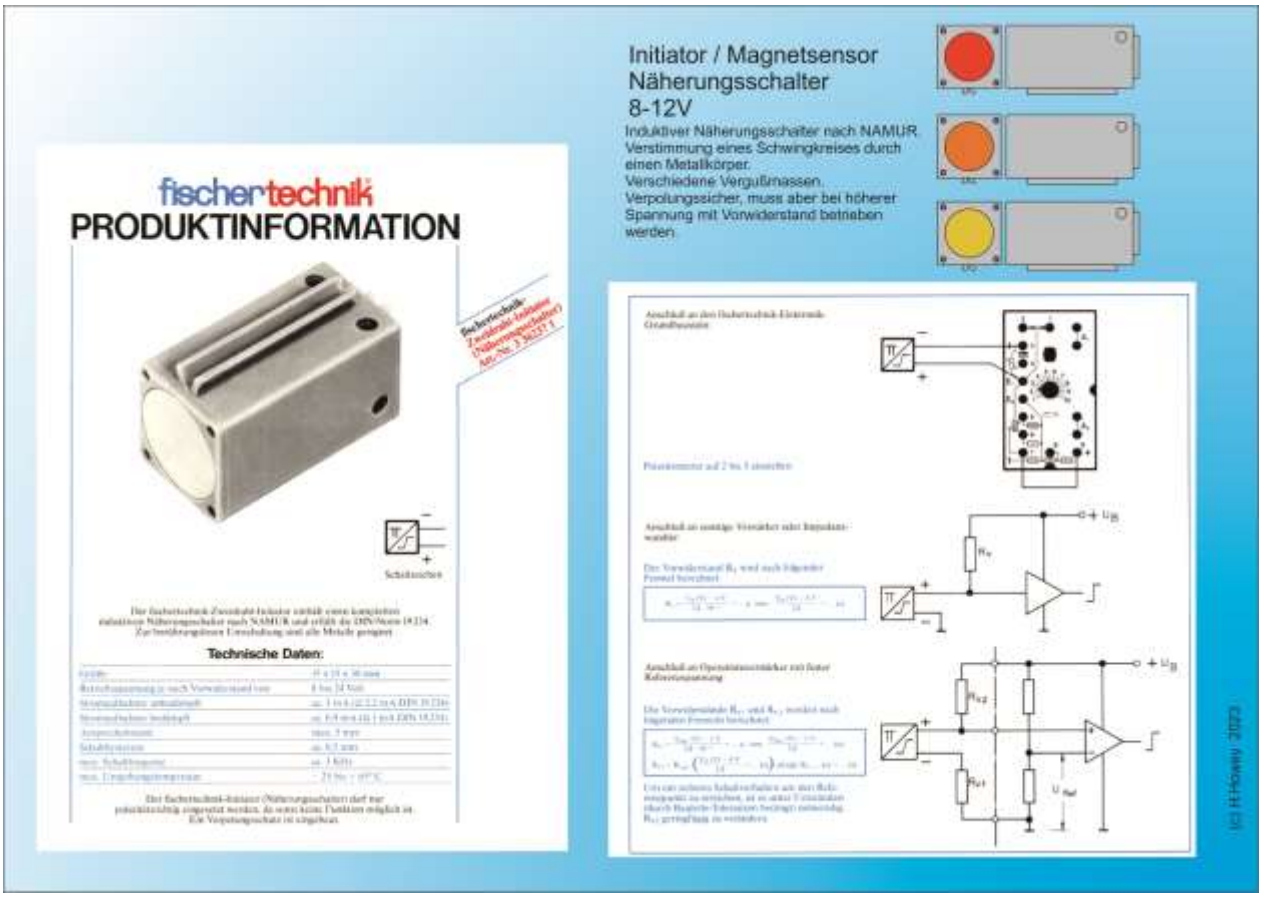
Alte Sensoren und Aktoren aus „grauer“ Vorzeit /Alternativen

Auch alte Sensoren der Kästen, wo die Bausteine noch grau waren, lassen sich an die Controller anschließen.

Initiator (Industriemodelle) 36237

Die Initiatoren sind Sensoren, die auf Eisen reagieren. Sie verändern ihren Widerstand und wurden an den Verstärkerbaustein (Silberlingen) angeschlossen.

Diese Initiatoren können über einen Vorwiderstand an andere Spannungen angeschlossen werden. Um auch etwas die Empfindlichkeit einstellen zu können sollte man ein 10k Ω Potentiometer (Regelbarer Widerstand) nehmen. Der Sensor Plus wird mit dem I-Eingang und der zweite mit dem Minus vom Controller verbunden. Zusätzlich wird das Potentiometer mit dem Schleifer (Mittelabgriff) an Plus 9V vom Controller und einer von beiden weiteren Anschlüssen vom Potentiometer an den Plus vom Sensor angeschlossen. Im Controllertest kann man dann das Potentiometer passend den Abstand einstellen.



Summer / Piezo Signal Signalgeber 36119

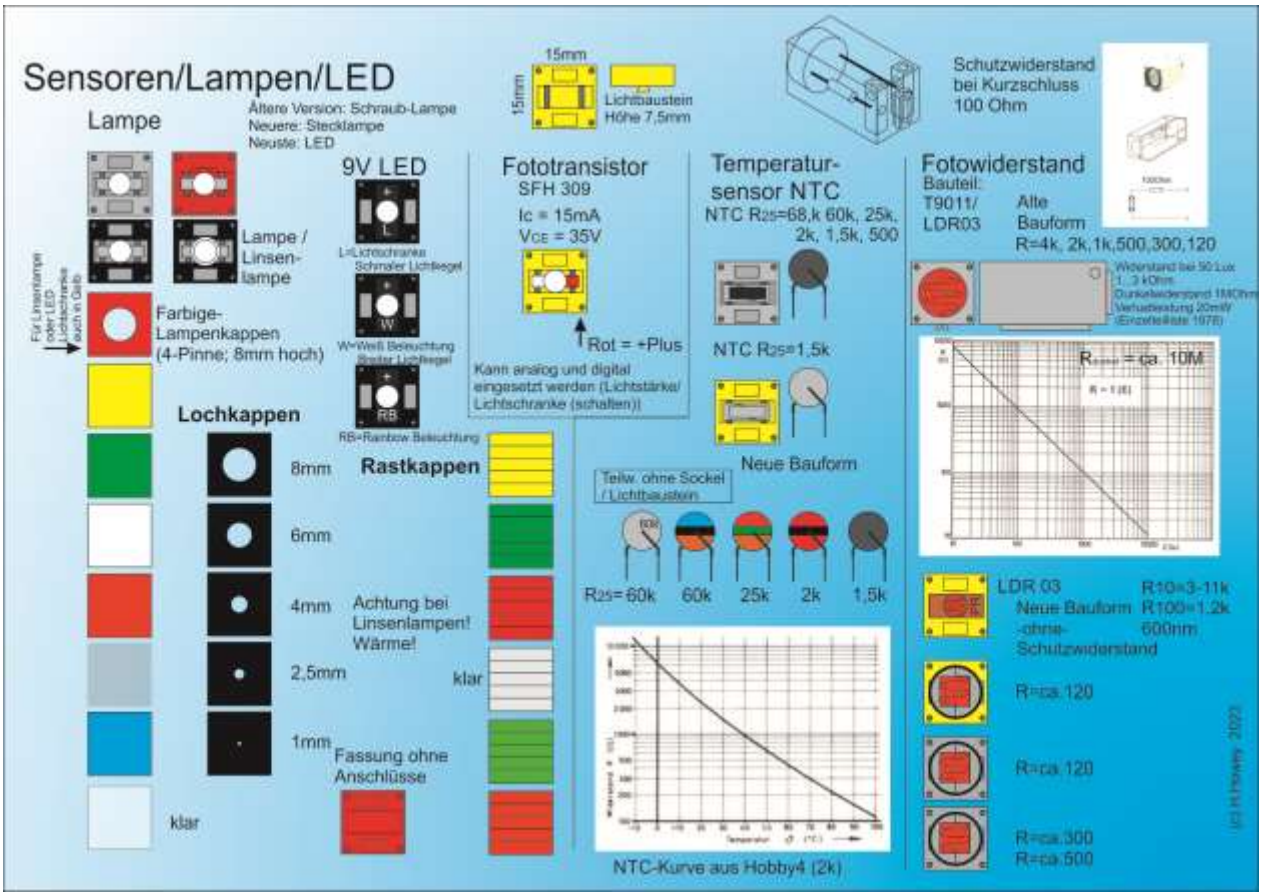
Der Piezosummer (Weiß/Schwarz) ist ein Signalgeber, der eine Piezoplätchen und eine kleine Elektronik enthält. Wenn man ihn mit Spannung versorgt, gibt er ein akustisches Signal ab. Er summt, hupt ...

Ihn kann man an einen O-Ausgang mit Plus (rot) vom Summer und Minus (sw) an Masse anschließen.

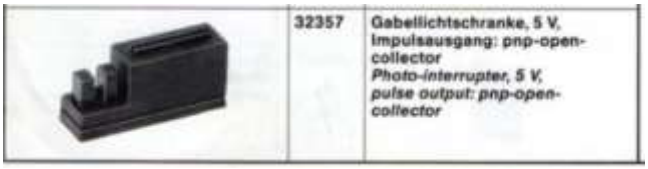
Sensoren Aktuell / Alt

Auch alte Sensoren lassen sich an die neuen Controller anschließen. Es ändern sich aber die Abfragewerte in den Programmen. Die muss man halt selber anpassen.

Hier eine Übersicht von Sensoren und LEDs. Bei den LEDs gibt es bei den Industriemodellen auch noch eine 24V Variante der L/Lichtschranken und W/Weißen LEDs.

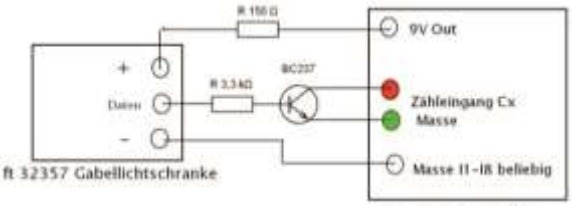


Ältere Gabellichtschranke

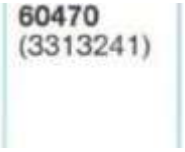


Beim alten Computing Roboter, gab es Gabellichtschranken. Die funktionierten an den „Homecomputern“, weil diese auch weniger mit dem Betriebssystem zu tun hatten und somit mehr Zeit für die Schnittstellen hatten. Bei heutigen PCs und somit auch bei dem fischertechnik TXT 4.0 Controller, sind schnelle Zähler in der Hardware („auf der Platine“) eingebaut. Auch ist von damals zu heute, die Spannung unterschiedlich.

Somit kann man die nur mit folgender Schaltung an den TX/TXT und auch den TXT 4.0 /RX anschließen:

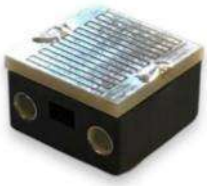


Elektromagnet
Es gibt auch einen „aktuellen“ Elektromagneten und welche aus „grauer“ Vorzeit.

Anschluss von den Polen ist egal. Zum Lösen sollte man kurz die Spannung umkehren. So bleiben die Eisenteile nicht kleben. Man kann auch Tesafilm auf den Magneten kleben. Dann kann man ausprobieren ob nicht auch ein O-Ausgang reicht statt eines M-Ausganges. Es gab auch eine Bauform mit einer Spule.

Feuchtigkeitssensor
Es gibt auch einen „aktuellen“ Feuchtigkeitssensor und welche aus „grauer“ Vorzeit.


70070_fu 
36718

Er kann sowohl Analog über die Widerstandsmessung als auch Digital als Nässesensor verwendet werden.

Reedkontakt

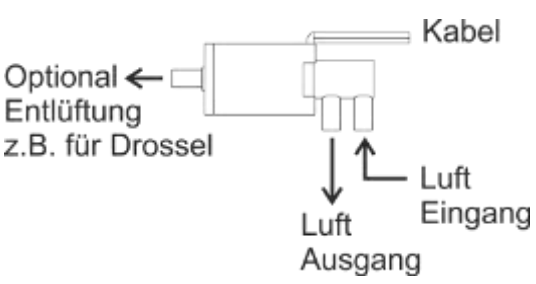

32394

Den Reedkontakt gab es auch in anderen Bauformen. Wird wie ein Taster verwendet.

Magnetventil

Ventil mit Elektromagneten. Ansteuerung wie der Elektromagnet.

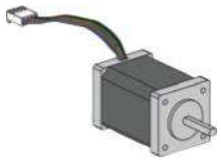
Magnetventil von fischerfriendman (FFM)



FFM 3/2 Magnetventil Es gibt 12V und 9V Ventile. Für Akku-Betrieb, besser die 9V Variante nehmen. Die gibt es nur bei fischerfriendsman.
Mit diesen Ventilen ist es auch möglich Vakuum zu schalten.

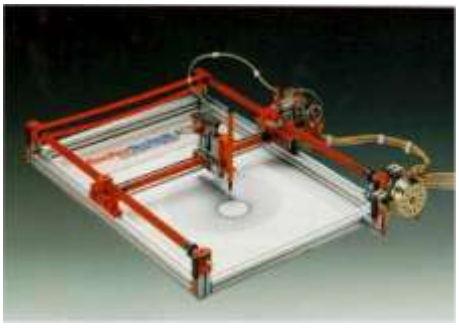
Schrittmotoren

Die alte Bauform der Schrittmotoren wird sehr sehr heiß. Man sollte darauf verzichten diese mit dauer 9V anzusteuern. Besser die in der Programmierung beim Halten abschalten oder mit niedrigem PWM Wert.

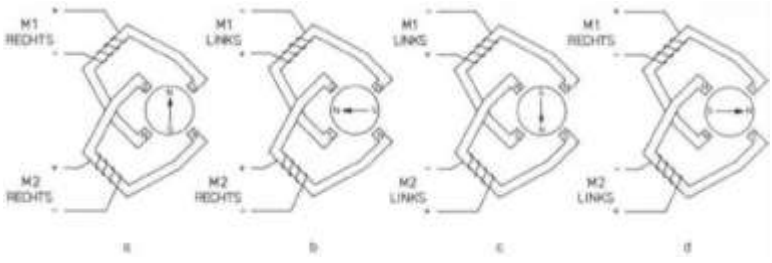


fischertechnik hat beim 3D-Drucker Nema 14 Motoren benutzt. Die kann man ohne weiteres mit 9V ansteuern.

Wenn man (preisgünstig) welche ansteuern möchte, kann man Nema 17 z.B. von Pollin.de nehmen. Da muss man nur einen Achsadapter dazu haben. Normalerweise braucht man 4 Anschlüsse, also 2 M-Ausgänge zur Ansteuerung eines Schrittmotors. Mit einem Trick kann man auch 3 Schrittmotoren mit 3x M-Ausgängen steuern, indem man jeweils zwei der Adern jedes Motors auf einen M-Ausgang legt. Den muss man halt immer auf die gleiche Polung am Ende des Schrittes lassen. Der Nachteil ist aber ein „Zittern“ der nicht angesteuerten Motoren. Im Normalfall ist das ohne Belang, da das nicht auf die weiteren Achsen übertragen wird, weil ein Getriebe dazwischen ist. Man hat so aber noch zwei Ausgänge frei. z.B. für einen Magneten, der einen Stift nach unten zieht. Damit kann man einen Plotter bauen.



Plotter von 1985 mit 4 M-Ausgängen.



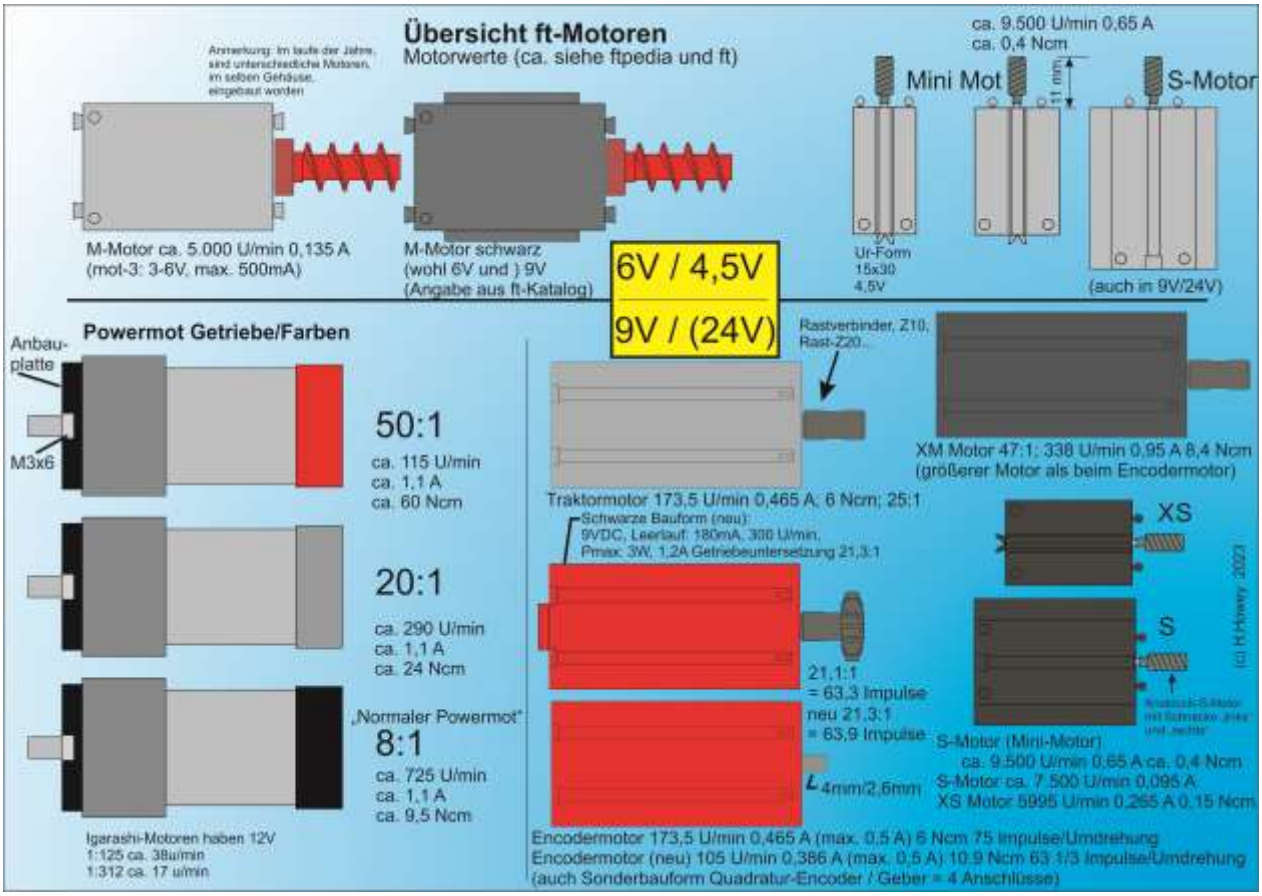
Funktionsweise vom Schrittmotor, wie rum die Spannung anliegt.

Ultraschallsensoren

Von den Ultraschallsensoren gibt es zwei Varianten. Einmal die für das ältere Robo Interface (zwei Kabel) und das neuere für den TX/TXT/TXT 4.0/RX mit drei Kabeln.

Die älteren Ultraschallsensoren für das Robo Interface mit zwei Kabeln, kann man **nicht** an den neueren Controllern anschließen.

Tipp: Übersicht der Motoren von fischertechnik
Eine Übersicht aller aktuellen und bisherigen Motoren von fischertechnik. Es kommen noch wenige spezielle Motoren dazu, die z.B. bei Schaufenster- und Messemodellen, die aber genauso angeschlossen werden können.



Ich selber verwende auch (eher selten) die grauen 6V Motoren an den fischertechnik Controllern. Eine Garantie, dass es auch bei anderen funktioniert, möchte ich aber nicht abgeben. Die 4,5 V Motoren sind Sammlerstücke und werden nicht angeschlossen.

24V Motoren laufen so nicht an den Controllern.
Auch die Quadratur-Encodermotoren (=2x Encoder) sind eher eine seltene Ausnahme bei fischertechnik

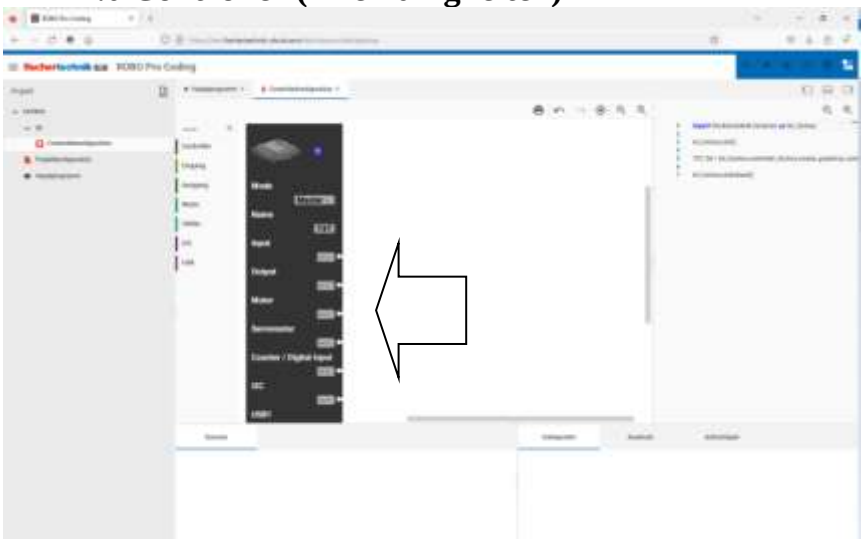
Kompressor Luftpumpen / Vakuum
Von der Firma Pollin gibt es kleine Luftpumpen (330068). Im Gegensatz zu den fischertechnik Kompressoren haben die noch einen zusätzlichen Anschluss. Hier kann man auch Vakuum entnehmen. Ft-Fans haben für diese Pumpe ein Gehäuse gemacht (3D-Drucker), was zu den fischertechnik-Nuten passt.

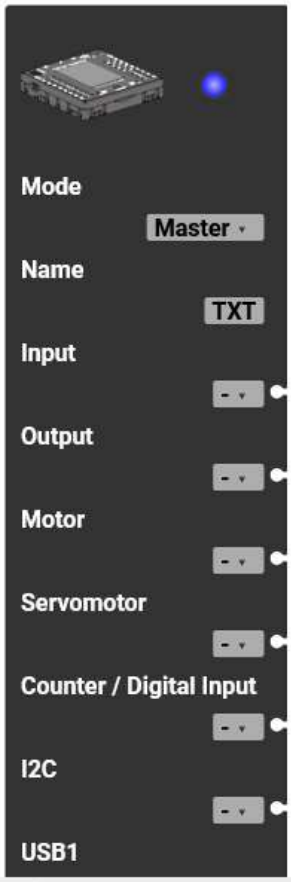
Mit einem Magnetventil von fischerfriendsman.de (+Drossel), kann man dann auch Vakuum schalten. Siehe dazu auch das Forum der ftcommunity.de:

https://forum.ftcommunity.de/viewtopic.php?p=62042

Die Controller und deren Möglichkeiten (Controllerkonfiguration)

TXT 4.0 Controller (Alle Fähigkeiten)





Folgende Anschlussmöglichkeiten hat der TXT 4.0 Controller:

Eingänge -> I1 bis I8 (Input)
Ausgänge -> O1 bis O8 (Output)
Motor -> M1 bis M4 (Motor)
Zähler -> C1 bis C4 (Counter)
EXT1 und EXT2 (Extension) -> I2C (Gestensensor…)
USB1/2 -> (Universeller-Serieller-Bus) Kamera, USB-Stick / PC

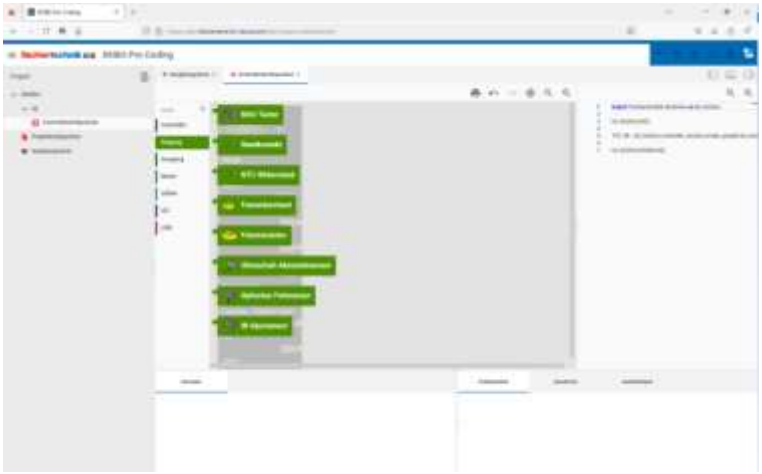
TXT 4.0 Controller

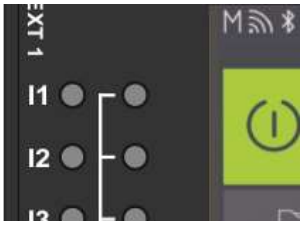
In der Robo Pro Coding Darstellung vom TXT 4.0, ist rechts oben eine blaue LED zu sehen. Die zeigt je nach Farbe an, ob der Controller verbunden ist oder nicht.

Eingänge (Sensoren)
Folgende aktuellen fischertechnik Bauteile kann man an einen Eingang vom TXT 4.0 Controller anschließen:

Mini-Taster (Taster Öffner oder Schließer, Schalter...)
Reedkontakt (Magnetischer Taster)
NTC-Widerstand (Temperaturmesser)
Fotowiderstand (Lichtsensor)
Ultraschall-Abstandssensor (Entfernungsmesser) (nicht BT Smart Controller)
Optischer Farbsensor (Farbenerkennung)
IR-Spursensor (Zweifacher Hell-Dunkel Sensor)

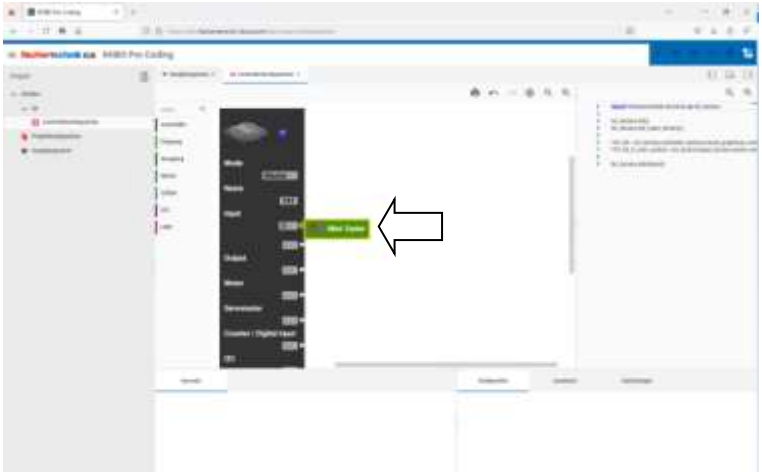
In Robo Pro Coding sind diese fischertechnik Sensoren Bauteile grün dargestellt:





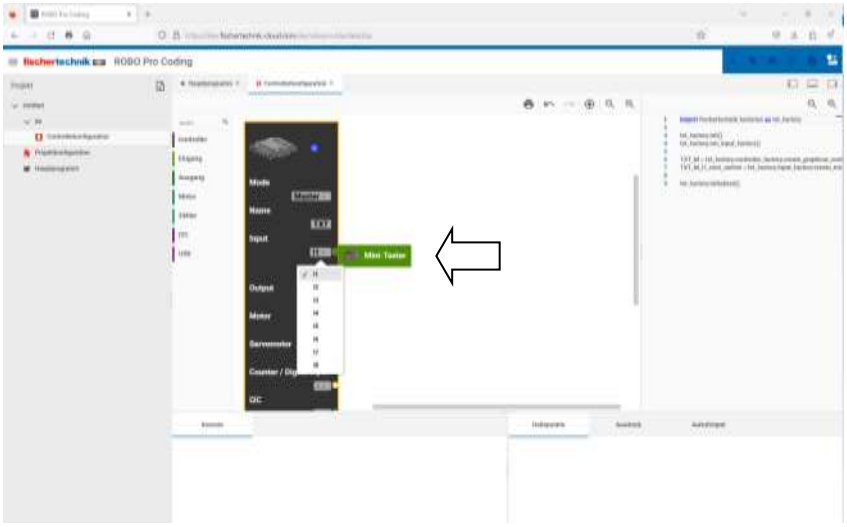
Die Eingänge gehen von I1 bis I8. Ein Eingang z.B. Taster wird an I1 und an dem rechts danebenliegenden Minus angeschlossen. Da Robo Pro Coding nicht wissen kann wo was angeschlossen ist, muss man nun den Taster auswählen (links klicken) und an den Input hinziehen.

I-Eingänge mit Minus rechts davon am TXT 4.0



Den Taster kann man nur an Input oder "Counter/ Digital Input" anschließen. Wo anders, z.B. an den

Motorausgang, geht es nicht. Wenn man mit dem Cursor auf das I1 links klickt, erscheint eine Auswahl der Eingänge I1 bis I8.



Man kann hier auswählen an welchem Eingang der Taster angeschlossen ist. Hier wählen wir I1 aus. Darunter ist ein weiterer neuer leerer Eingang entstanden, wo weitere Taster oder andere Sensoren hingezogen werden können.

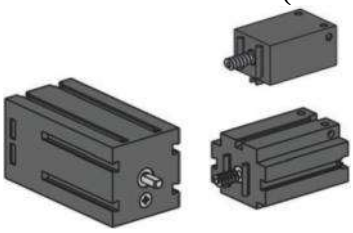
Ausgänge (Aktoren)
Folgende fischertechnik Bauteile kann man an einen Ausgang anschließen:

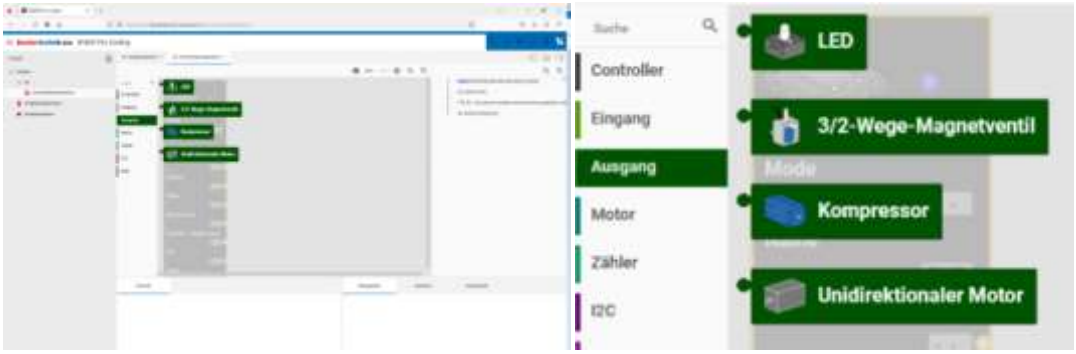
LED (LED, Lampe...)

3/2-Wege-Magnetventil (Druckluftventil An-Aus)

Kompressor (Drucklufterzeuger)

Unidirektionaler Motor (Motor der nur in einer Richtung läuft, An-Aus)

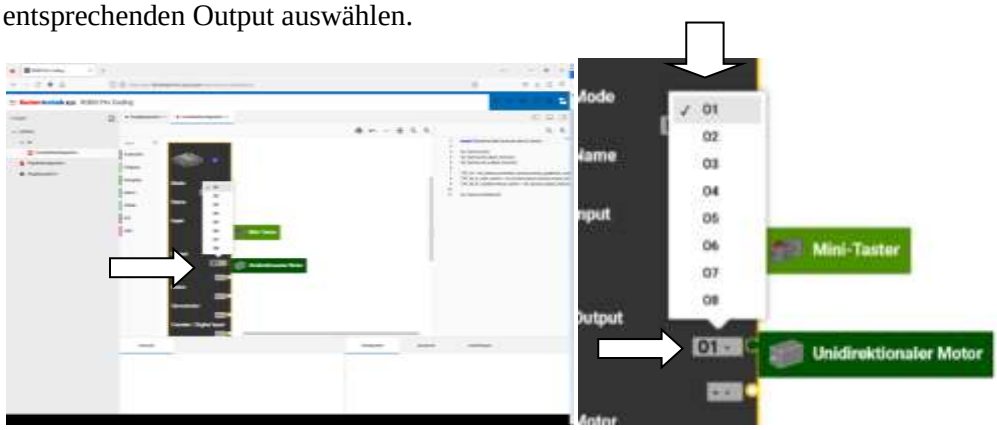


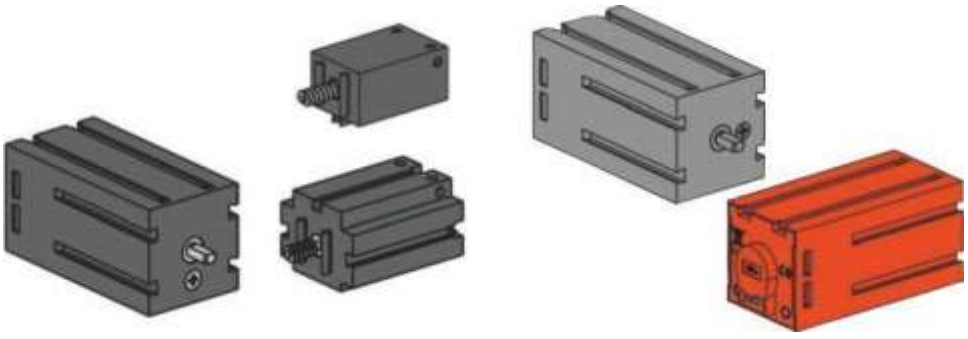


Die Ausgänge gehen von O1 bis O8. Um einen z.B. Motor anzuschließen, muss man den Motor mit O1 und Minus verbinden. Welcher Minus beim TXT 4.0 ist egal. Man kann das Minus links neben den Tastern oder rechts unten, neben den Countern nehmen.

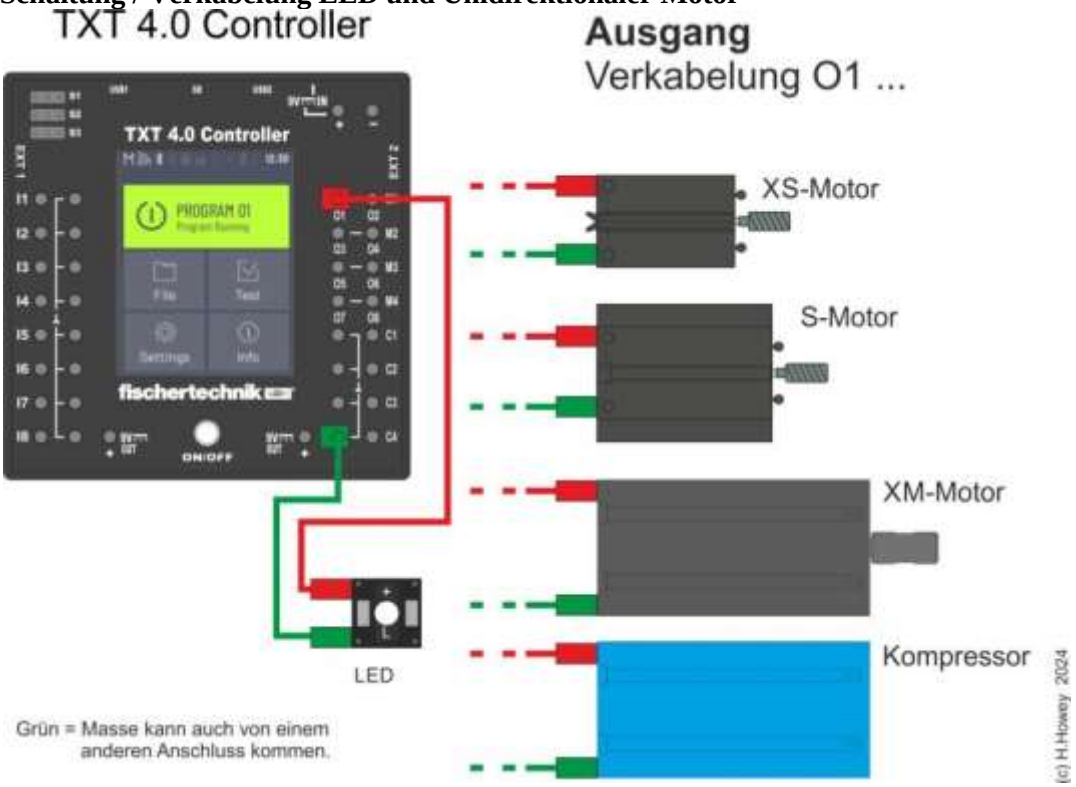
Unidirektionaler Motor

Ein "Unidirektionaler Motor" kann nur an einen Output angeschlossen werden. Um einen anderen Output zu benutzen, muss man auf das O1 links neben den "Unidirektionaler Motor" links klicken und dann den entsprechenden Output auswählen.



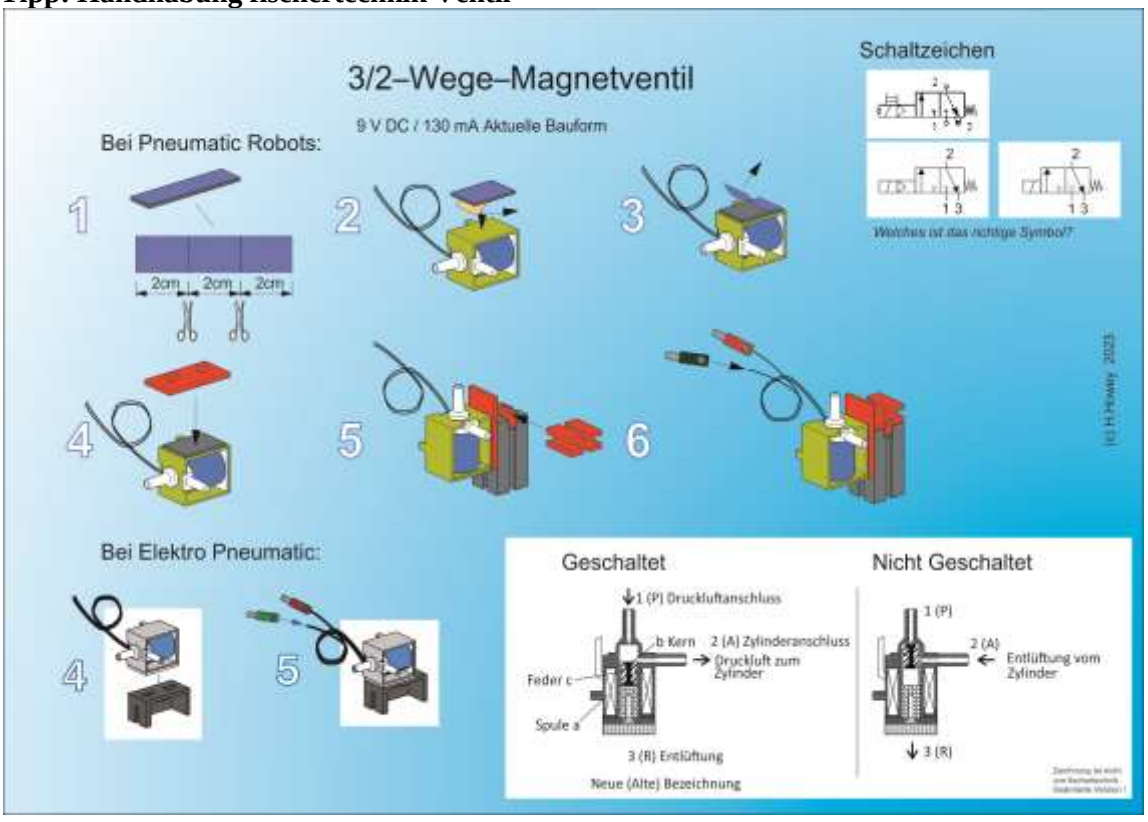


Schaltung / Verkabelung LED und Unidirektionaler Motor



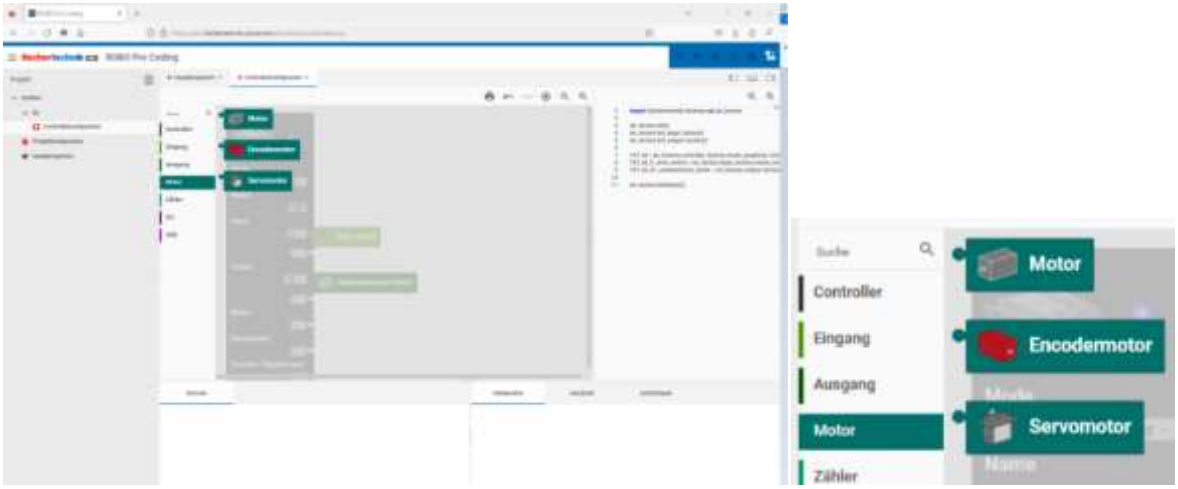
Man kann auch Encodermotoren als „normale“ Motoren benutzen.

Tipp: Handhabung fischertechnik Ventil



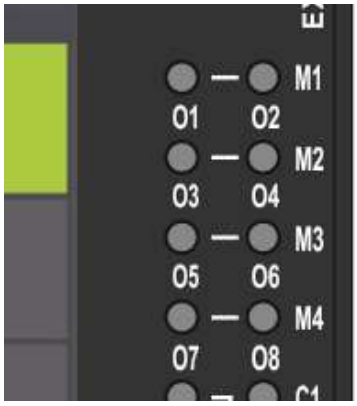
Motor
Bei "Motor" können folgende fischertechnik Bauteile angeschlossen werden:

Motor (fischertechnik Motoren)
Encodermotor (Motor mit einem Impulsgeber (ca. 63 pro Umdrehung))
Servomotor (Motor mit Getriebe, den man um einen bestimmten Winkel drehen kann, z.B. für die Lenkung)



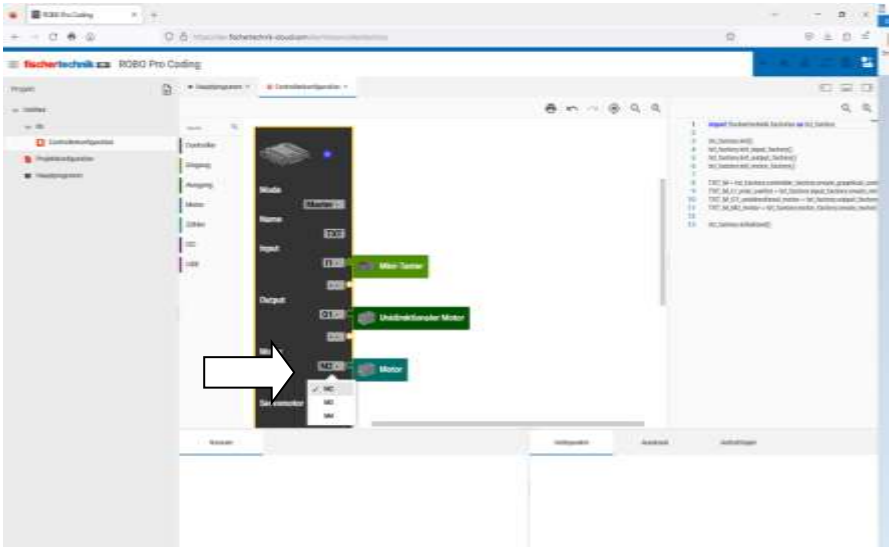
Der Unterschied von Ausgang und Motor ist derjenige, dass jeweils zwei O Outputs ein M-Ausgang sind.

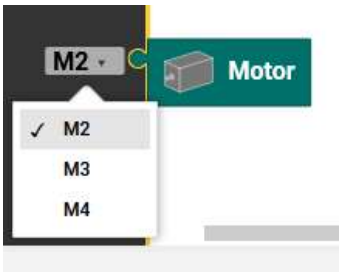
O1 und O2 sind M1
O3 und O4 sind M2
O5 und O6 sind M3
O7 und O8 sind M4



Ausgänge am TXT 4.0 Controller

Wenn man wie im vorherigen Beispiel bei "Ausgang" den O1 belegt hat, kann man nicht mehr M1 auswählen. Andersherum, wenn man z.B. M2 bei "Motor" mit einem Motor belegt, kann man bei "Ausgang" nicht mehr O3 und O4 auswählen.

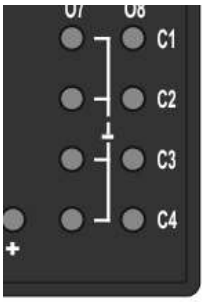




Auswahl des Motorausganges

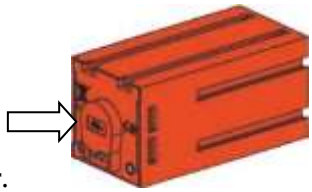
Wenn die Drehrichtung vom Motor nicht mit dem im Programm übereinstimmt, die Stecker am Motor oder die Stecker am Controller tauschen.

Zähler
C1 bis C4

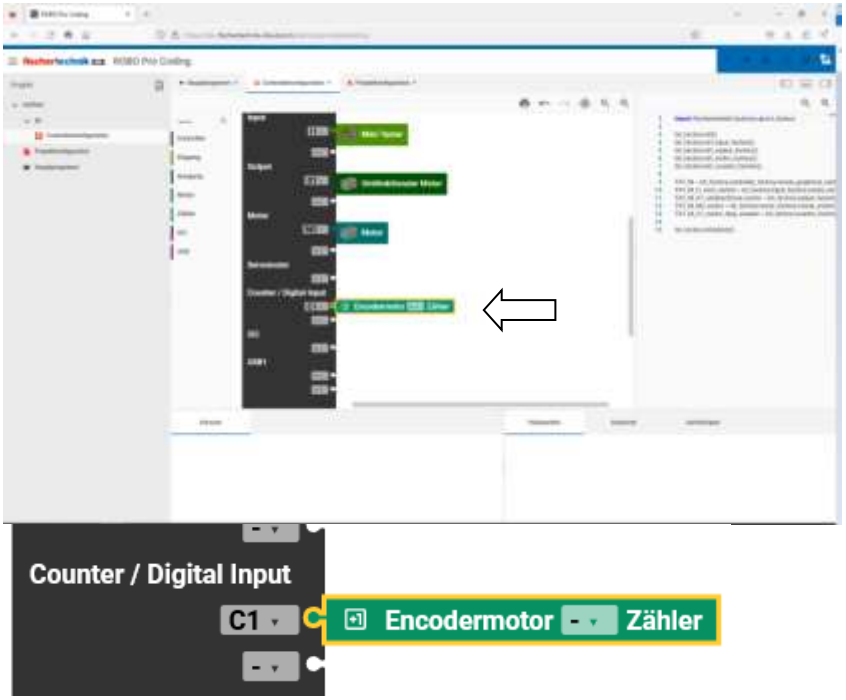


Man kann Impulse auf zwei Arten zählen. 1. man nimmt einen normalen I-Eingang oder 2. vor allem wenn es viele, schnell hintereinanderkommende Impulse sind, den Zähler (C = Counter). Jedem M-Ausgang ist ein Counter zugeordnet. So hat M1 den C1, M2 den C2, M3 den C3 und M4 den C4.

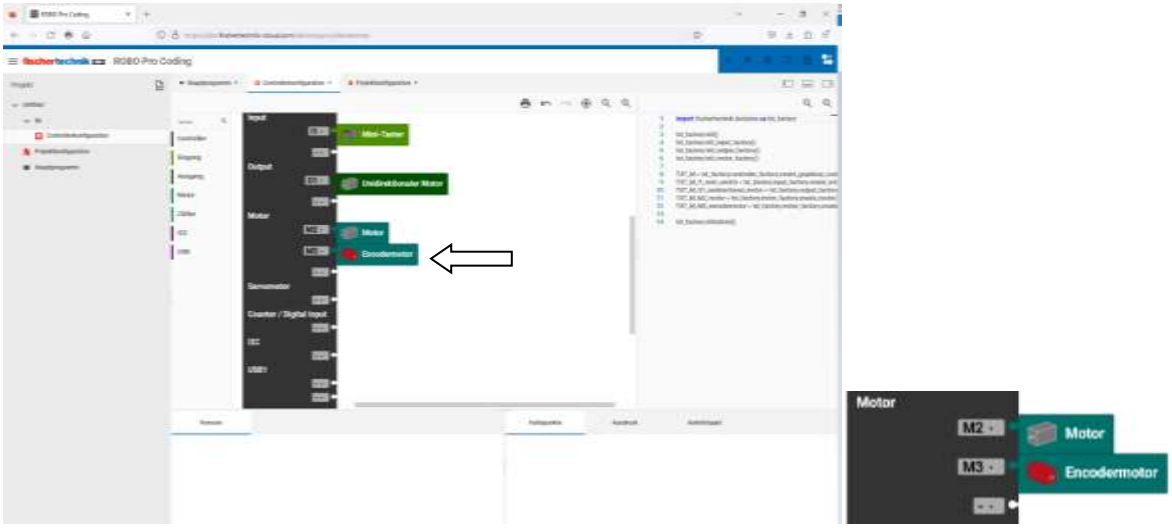
Am Encodermotor sind hinten drei Pins, wo man ein Kabel anschließen kann und mit Plus, Minus und dem Countereingang verbindet. Man kann aber auch beliebige Impulse über ein Impulszahnrad und einem Taster zählen. So kann man z.B. die Umdrehungen an einer Welle zählen, die von einem S-Motor angetrieben wird.



Anschlüsse am Encodermotor der schnellen Zähler.

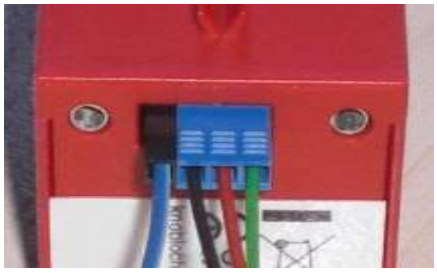


Zähler an C1

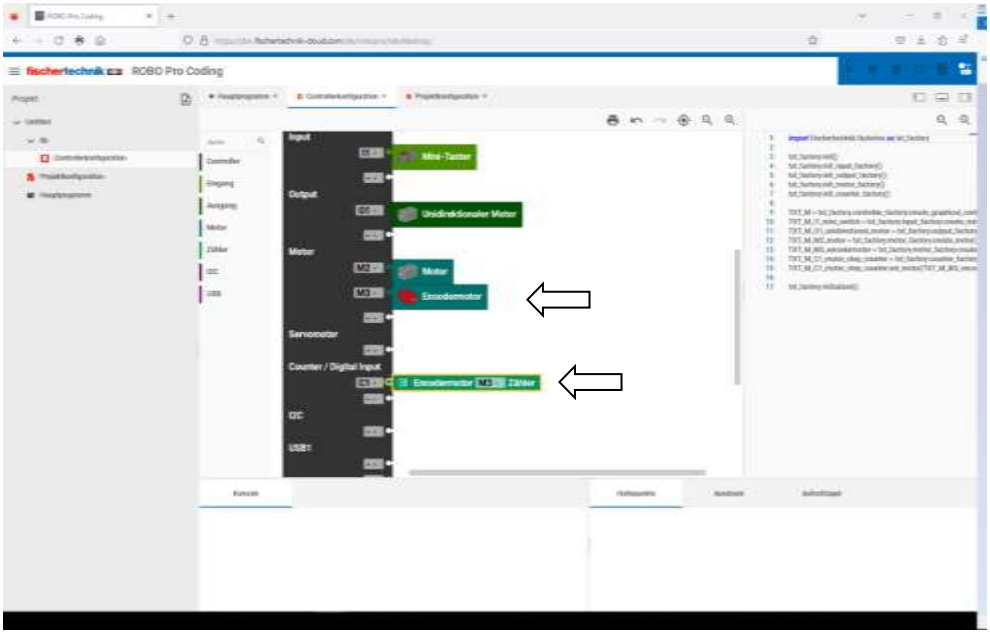


Encodermotor an M3

Ausnahme Quadratur-Encodermotor. Man kann auch diesen Motor an die Zähler anschließen. Sie haben zwei Encoder und belegen somit zwei Zähler. Die Auswertung muss man aber selber programmieren, da sie so in Robo Pro Coding nicht vorgesehen ist. Man kann sich aber damit behelfen, dass man ihn über einen Zähler steuert und die Werte mal zwei nimmt. Die genaue Ermittlung der Drehrichtung muss man auch selber programmieren. Es gibt im Netz und auf der ftCommunity Seite Umbauanleitungen von normalen Encodermotoren zu Quadratur-Encodermotoren. Es ist aber die Frage, ob man das unbedingt braucht, da man normalerweise weis wie rum man den Motor ansteuert bzw. ob man eine so hohe Auflösung braucht.

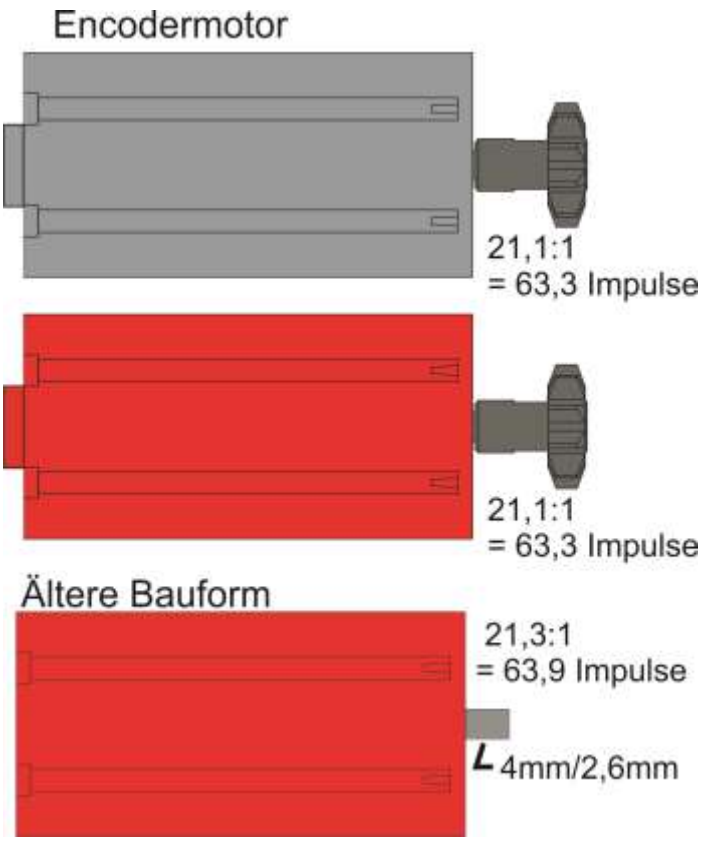


Quadratur-Encodermotor



Encodermotor M3 und Zähler an C3. Es ist wichtig, dass der richtige Motorausgang mit dem richtigen Zählereingang genommen wird. Damit kann man unter anderem mehrere Motoren gleich schnell laufen lassen. Das ist dann wichtig, um z.B. einen Roboter geradeaus laufen zu lassen.

Hinweis Encodermotor:
Man kann einen Countereingang auch als normalen Tastereingang (offen/geschlossen) einsetzen.



I2C
I2C ist ein Bus, an dem man momentan drei fischertechnik Sensoren anschließen kann. (Siehe dazu mehr unter Hinweis)

RGB-Gestensensor (Farbe, Entfernung, Geste)
Umweltsensor 6-Pin oder 10-Pin (Lufttemperatur, Luftfeuchtigkeit, Luftdruck)
Kombisensor 6-Pin oder 10-Pin (Triaxial Gyroskop, Triaxial Beschleunigungssensor, Kompasssensor) (Triaxial = 3-Achsen)

Pinbelegung
fischertechnik
TXT 4.0 Controller
EXT1 + EXT2 -Buchsen

Pin	Belegung
1	+3,3V
2	GND
3	CANL
4	CANH
5	SCL
6	SDA

Es gibt 6-Pin und 10-Pin Sensoren. Um die 10-Poligen Sensoren an den TXT 4.0 anzuschließen muss man einen Adapter oder sich ein Spezialkabel benutzen. Die 10-Poligen sind vorher für den TXT mit 5V gebaut worden. Sie funktionieren, meines Wissens, aber alle auch mit den 3,3V an dem TXT 4.0 .
Man kann sich diese Adapter oder Spezialkabel selber bauen. Anleitungen gibt es im Netz und auf der fischertechnik Communityseite.

Hinweis für sehr weit Fortgeschrittene:

Man –kann- auch andere Sensoren oder Geräte, wie ein Raspberry Pi Pico, an den I2C Bus anschließen. Die müssen aber 3,3V am I2C haben oder man muss einen Levelshifter dazwischenschalten. Der macht dann aus den 5V Signalen 3,3V Signale.

Es gibt zwei Möglichkeiten I2C auf dem TXT zu programmieren:
1. Die ft-Sensoren über Robo Pro Coding. Für diese Sensoren sind spezielle Blöcke vorhanden, die dann mit einer LIB-Datei den entsprechenden Python-Code erzeugen. Diese LIB-Datei ist schon auf dem TXT 4.0 vorhanden und entsprechende Beispiele sind vorhanden, wo man sich anschauen kann, wie fischertechnik das gemacht hat. Auch übergeordnete Befehle wie das Initialisieren vom I2C Bus oder die Startsequenz vom Sensor sind bereits vorhanden und werden automatisch eingefügt, wenn man bei der Controller-Konfiguration, den entsprechenden Sensor an den Controller in Robo Pro Coding anklickt.
2. Eigene Programmierung oder (fischertechnik-)"fremde" Sensoren, können nur mit Python direkt programmiert werden. Dazu werden Python-Code-Blocke eingefügt. Es wird -kein- Sensor bei der Controller-Konfiguration angeklickt. Blocky weiß somit nicht, dass er vorhanden ist.
Man muss sich selber um die Initialisierung vom I2C Bus und den Sensoren kümmern. In der Regel macht man eigene Python Blöcke in Blocky wo man die Initialisiert, Schreibt und Liest. Man kann auch ganz auf die Grafische Programmierung verzichten und nur mit Python programmieren.

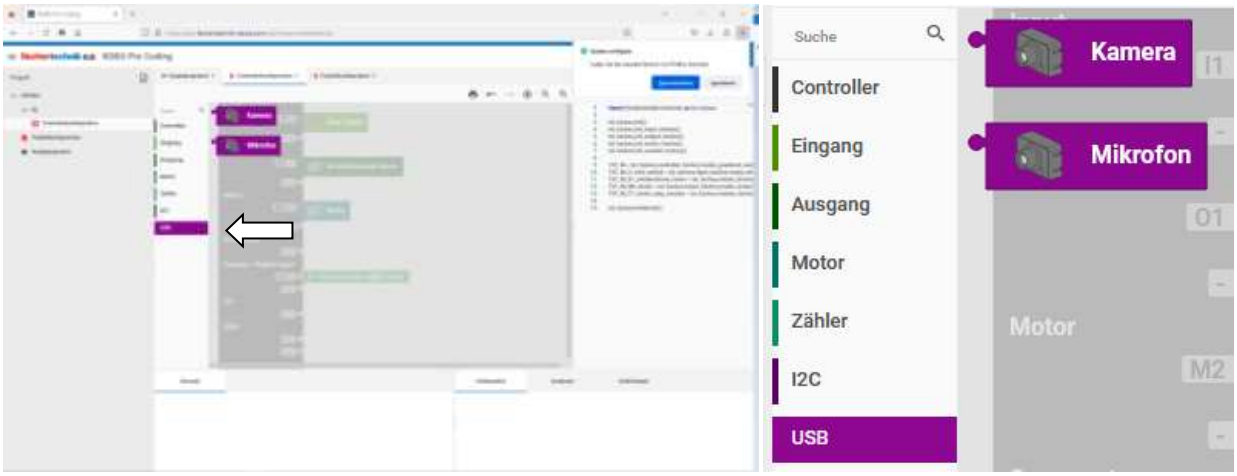
-**Niemals**- den TX-Controller mit 6-poligem Anschluss (5V)
mit dem 6-poligem Anschluss des TXT 4.0 (3,3V) verbinden!!!

Weiterführendes zu I2C weiter hinten im Kapitel „I2C für Experten“ (Seite 405).

Axel Chobe hat weiterführende Informationen zu I2C in einer PDF zusammengefasst:
http://chobe.info/dokus/I2C.pdf

USB

Der TXT 4.0 hat zwei USB-Anschlüsse. USB1 wird für die Verbindung zum PC genutzt. USB2 ist für die USB-Kamera von fischertechnik vorgesehen. Neben der Kamerafunktion kann man auch das Mikrofon der Kamera benutzen.



Es ist auch möglich eine PC Kamera zu nutzen. Es gibt Kameras die fest verbaut sind z.B. in Laptops oder „freie“ die man in den USB-Port vom PC einsteckt. Wenn die fischertechnik Kamera in den TXT 4.0 Controller eingesteckt wird, kann das Programm auch auf den TXT 4.0 Controller geladen werden und dort laufen. Ob nun auch andere Kameras am USB vom TXT 4.0 Controller funktionieren, habe ich selbst noch nicht ausprobiert.

Die Kamera kann in der Verbindung mit Robo Pro Coding sehr viel. Farben erkennen, Linien erkennen und Veränderungen wahrnehmen, Bälle in Größe und Position erkennen und vieles mehr. Auch kann man Spuren folgen, die sogar farbig sein können.

Eine Kamera kann man nicht an den RX- oder BT Controller anschließen.

Für fischertechnik-Profis:
An den USB-Anschluss kann man auch einen USB Stick anschließen, um z.B. Dateien zu übertragen.

Unterschied vom TXT zum RX Controller / Controllerkonfiguration

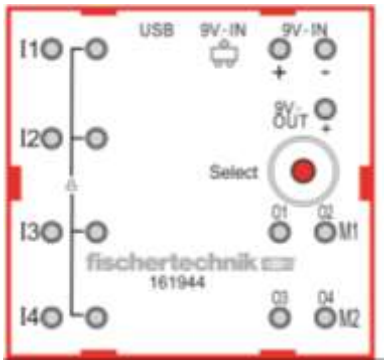
Im Grunde kann man „fast“ alle Sensoren, die am TXT 4.0 Controller laufen auch an dem RX Controller anschließen.
Es gibt jedoch Einschränkungen Z.B:

- Schnelle Zähler
  Es gibt keine schnellen Zählereingänge.

- USB Kamera
  Man kann keine Kamera an den Controller anschließen

.- Weitere Controller
  Es ist noch nicht abschließend geklärt, ob man mehr Ein- und Ausgänge bekommen kann, wenn man mehrere RX Controller koppelt.

Unterschied vom TXT zu BT Smart Controller / Controllerkonfiguration



Im Grunde kann man „fast“ alle Sensoren, die am TXT 4.0 Controller laufen auch an dem BT Smart Controller anschließen.
Es gibt jedoch Einschränkungen z.B.:

- Schnelle Zähler
  Es gibt keine schnellen Zählereingänge.

- USB / I2C
  Man kann keine Kamera und I2C Sensoren an den Controller anschließen

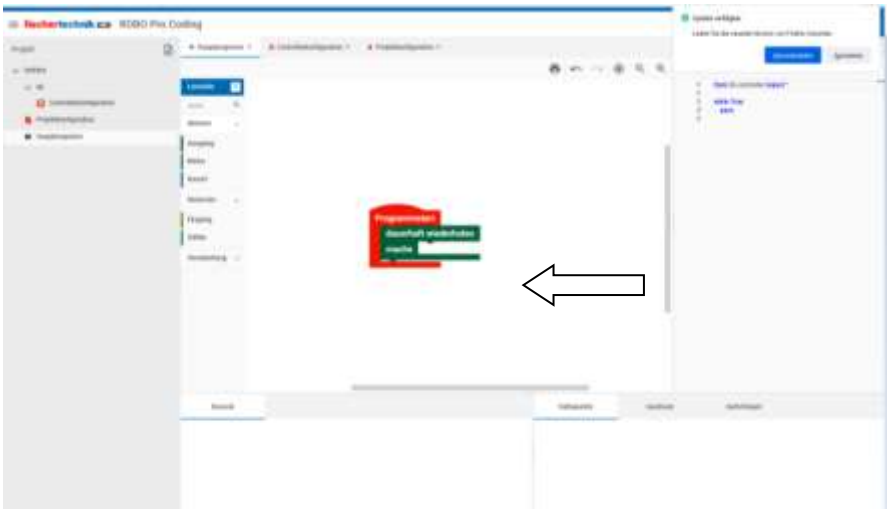
.- Weitere Controller

- Keine Ultraschall/Abstandssensoren

Robo Pro Coding

Programme erstellen. Schritt für Schritt

Man fängt mit einem Hauptprogramm „Programmstart“ an. Das muss immer vorhanden sein, auch wenn es "leer" ist. Hier ist noch automatisch, zusätzlich „dauerhaft wiederholen/mache“ dabei.



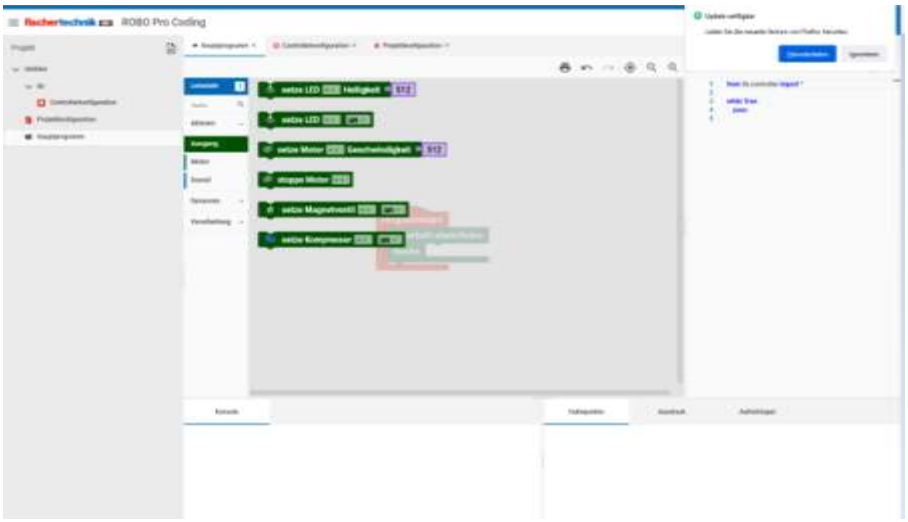
Lernstufe

Je nachdem welche Lernstufe man anwählt, werden mehr oder weniger Möglichkeiten angezeigt.

1 Einsteiger
2 Fortgeschrittene
3 Experten

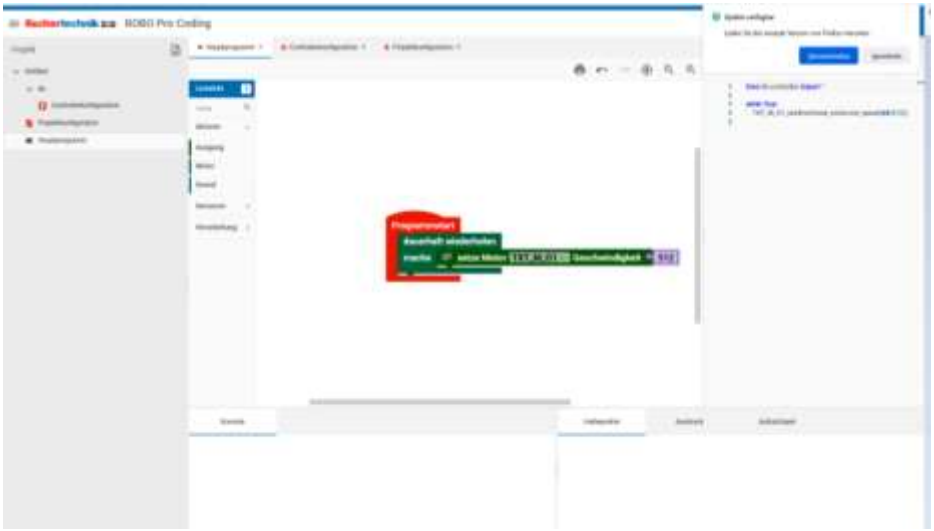
Bei Stufe 1 Einsteiger hat man nur Aktoren, Sensoren und Verarbeitung. Ab Stufe 2 kommt Kommunikation hinzu. Bei den Reitern und Blöcken kommen immer mehr hinzu.

Bei "Ausgang" kann man nun Schalten (An/Aus) oder eine Geschwindigkeit/Helligkeit einstellen.



Hier die verschiedenen Auswahlmöglichkeiten.

Nehmen wir mal "setze Motor" mit Geschwindigkeit und ziehen es in unser Programm.



Hier wird der Motor, den wir in der Controllerkonfiguration an den O1 angeschlossen haben, von Robo Pro Coding automatisch eingesetzt und mit der Geschwindigkeit 512 angesteuert.
Wenn wir das Programm mit an O1 angeschossenem Motor laufen lassen würden, hätte der Motor immer die maximale Geschwindigkeit. Wenn man die 512 durch z.B. 200 ersetzt, läuft der Motor langsamer. Man kann also mit der Angabe "Geschwindigkeit" zwischen 0 und 512 wählen, wie schnell der Motor läuft.

Die Schleife "Programmstart" mit "dauerhaft wiederholen" lässt den Motor dauerhaft mit 512 laufen.
512 ist die maximale Geschwindigkeit.

Herzlichen Glückwunsch!
Sie haben ihr erstes lauffähiges Programm geschrieben!

Programm laufen lassen (Blaue Kopfzeile)

Um ein Programm laufen zu lassen, muss man auf das Playsymbol gehen.

Wenn man den anklickt, wird das Programm übertragen und gestartet.

Programm stoppen

Um ein Programm zu stoppen, muss man auf Stopp drücken. Der Programmablauf wird beendet.

Man kann das laufende Programm auch stoppen in dem man den Controllertest aufruft.

Rechts neben der Blocky Ansicht, kann man sich das gleichwertige Pythonprogramm ansehen. Das, was man mit dem Blocky "gemalt" hat wird zeitgleich in ein Pythonprogramm umgesetzt. Auch wenn man Python nicht kann, erkennt man gleiche Sachen wie O1 oder 512.

Wenn man nur Robo Pro Coding mit Blocky machen möchte, kann man sich die Pythonansicht auch verkleinern, in dem man auf die Trennlinie geht und diese nach rechts zieht.

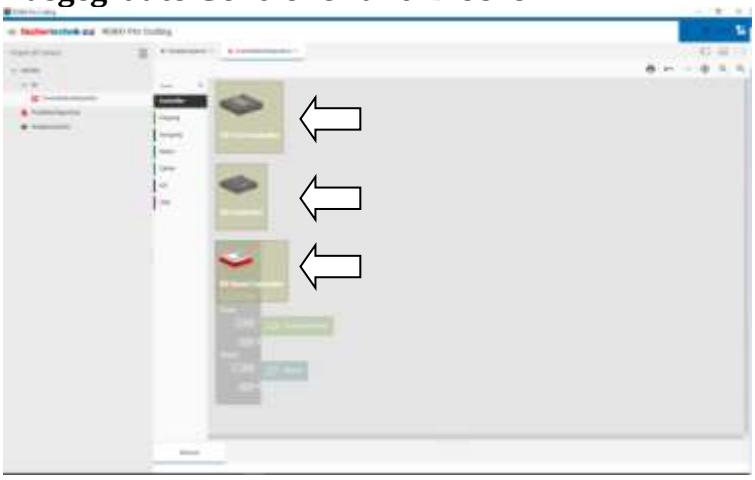
Hinweis zu anderen „Test-“ Programmen

Es gibt vom Controller TXT 4.0 und von fischertechnik Kästen fertige Programme zum Testen und Ausprobieren. Einfach mal bei einem neuen Projekt (Kastenauswahl) Beispiele reinschnuppern.

Ein Programm erstellen – Schritt für Schritt

Die Programme sind auch auf andere fischertechnik Controller übertragbar.

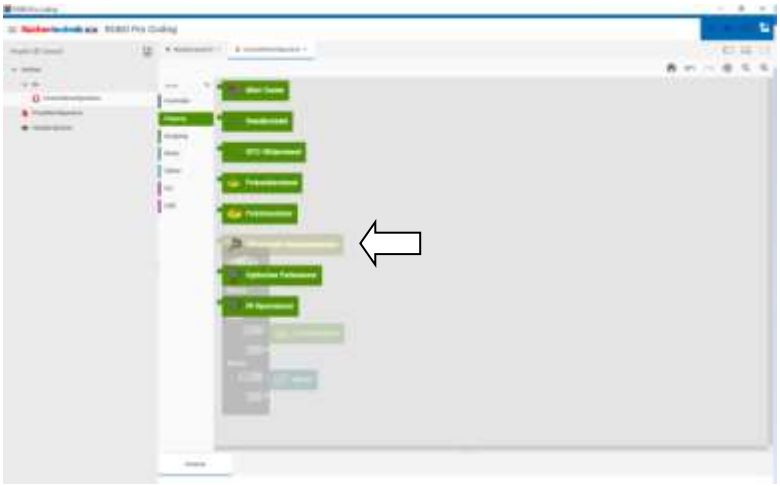
Ausgegraute Controller und Blöcke



Wenn ein Controller ausgewählt wurde, hier der BT Controller, kann kein weiterer Controller ausgewählt werden. Sie sind "ausgegraut".

Meistens liegt es daran, dass in der Controllerkonfiguration kein Controller angewählt wurde.

Wenn dort der passende Controller ausgegraut ist, wird es ein Fehler beim Erstellen des Projektes gewesen sein. Man kann es neu machen oder den Controller wechseln. Schneller wird es sein, mit dem Projekt neu anzufangen.

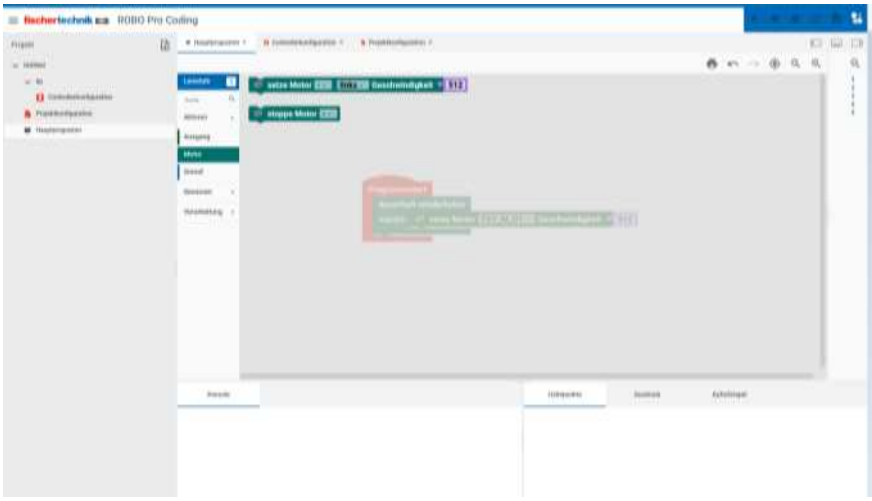


Man kann "ausgegraute" Sensoren nicht an den jeweiligen Controller anschließen, wenn diese dafür nicht geeignet sind. Hier beim BT Smart Controller ist das der Ultraschallsensor. Einzige Abhilfe ist hier, einen anderen Controller zu nehmen.

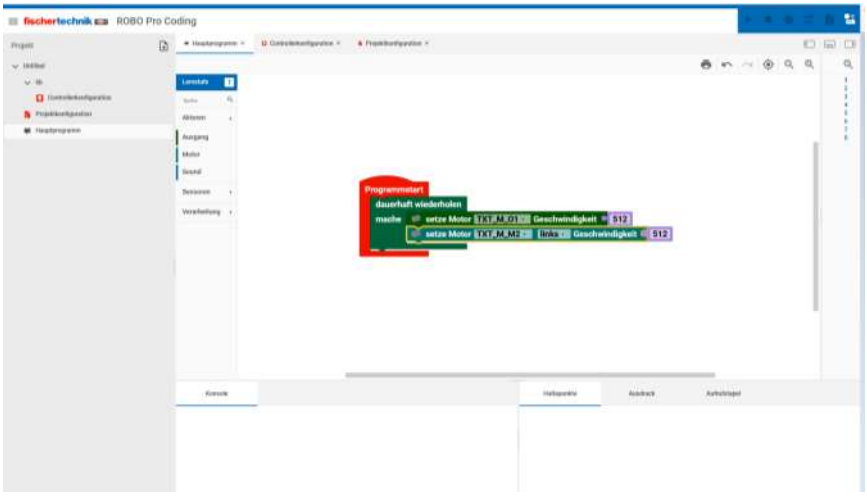
Manchmal sind die Blöcke aber auch ausgegraut, weil noch nichts angelegt wurde. Z.B. erscheinen die Blöcke bei den Variablen nur, wenn eine Variable angelegt/erzeugt wird. Es wird auch ein Feld oberhalb der Blöcke angezeigt, worauf man klicken kann, um das anzulegen. Gleiches gilt für die Fernbedienung.

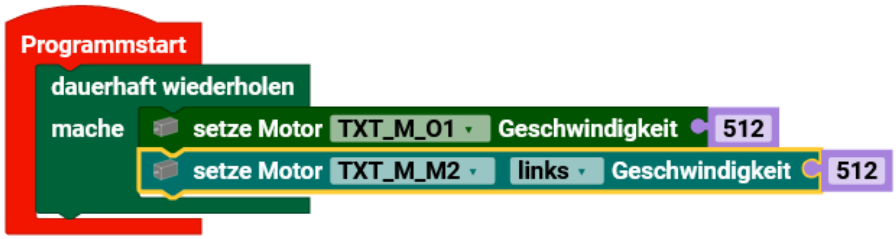
Schritt für Schritt Programm: Motor laufen lassen TXT 4.0

Unter Motor werden die M-Ausgänge gesteuert.



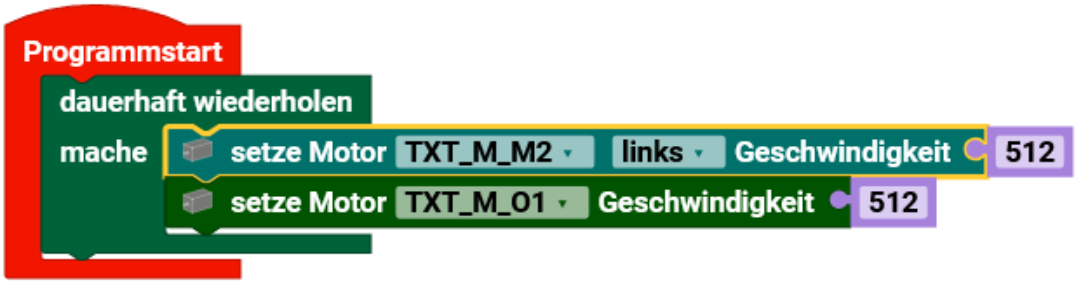
Es wird der Motor M1 bis M4, links/rechts und die Geschwindigkeit 0-512 eingestellt. Hier kann man auch den Motor aus machen, in dem man den Motor mit Geschwindigkeit „0“ ansteuert.



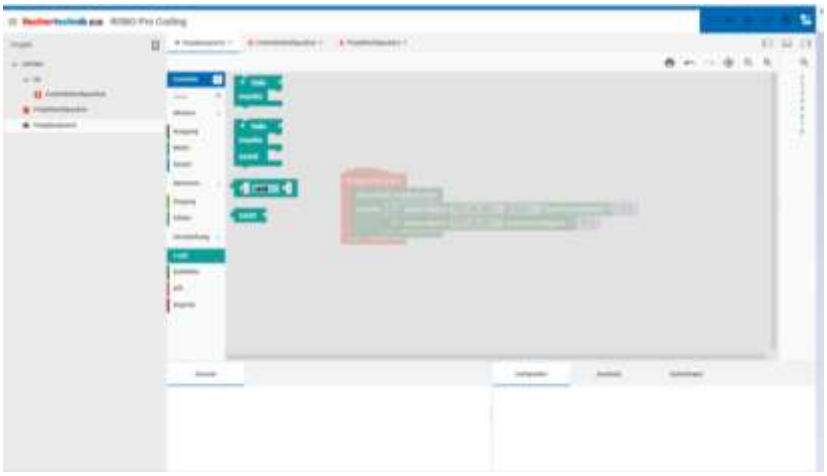


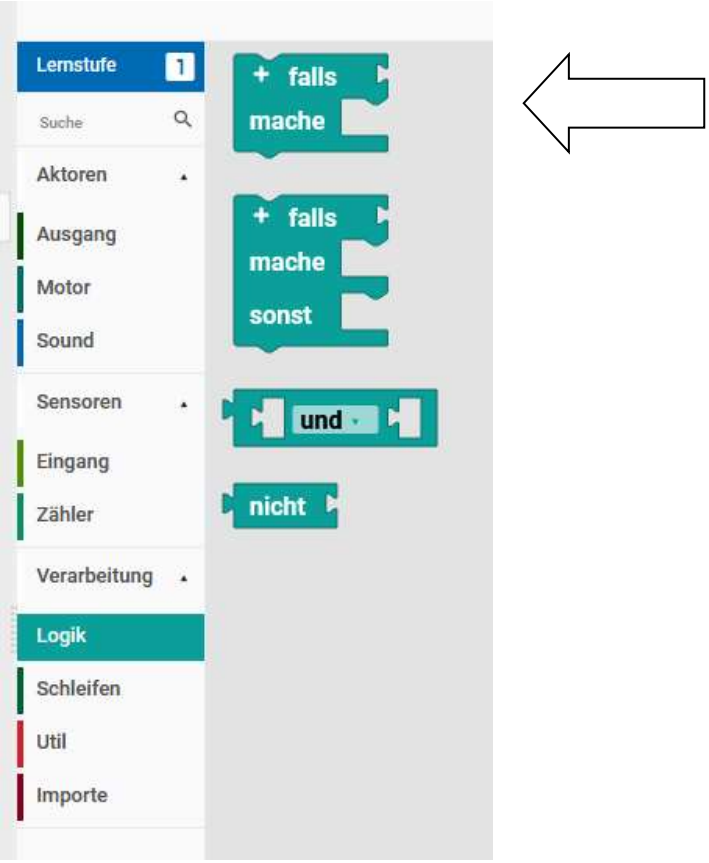
Bei diesem Programm wird der O1 Motor und der Motor am M2 links mit jeweils 512, also der vollen Geschwindigkeit angesteuert.

Auch wenn wir die Reihenfolge der Motoren ändern, läuft das Programm gleich wie das vorherige ab.

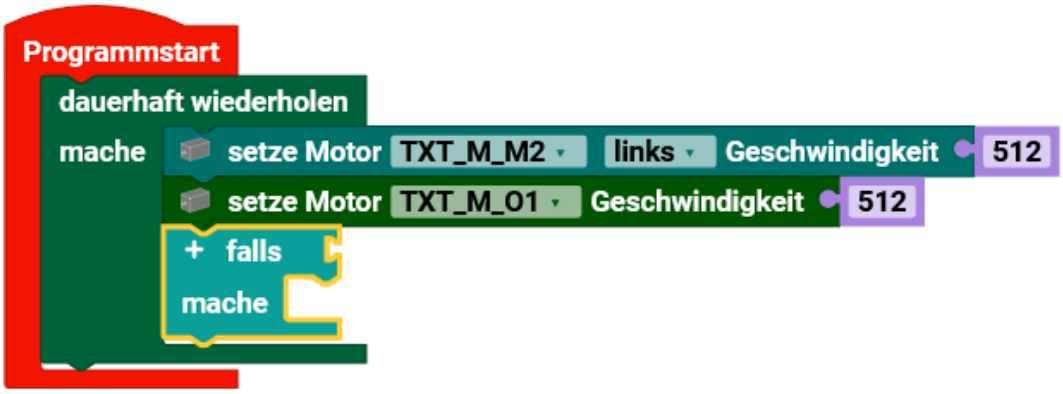


Nehmen wir nun einen Logikblock dazu

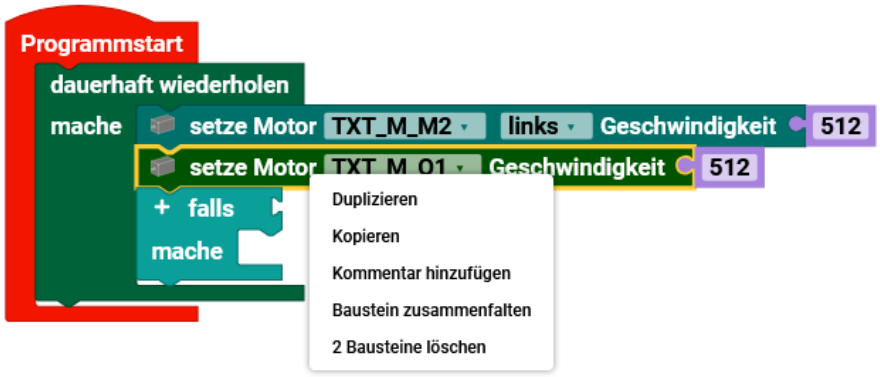




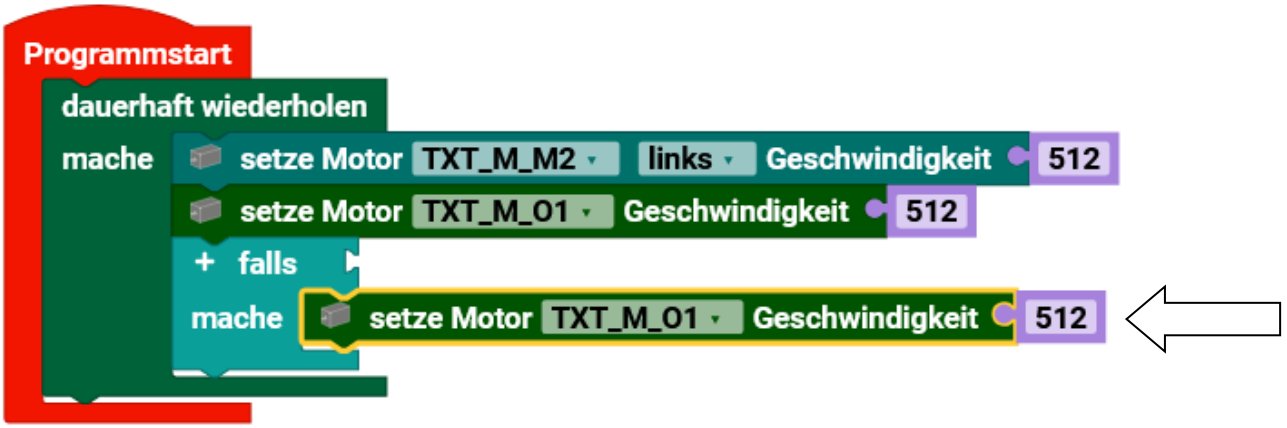
In dem Fall der oberste falls/mache und setzen ihn unter den zweiten Motor, in der Dauerhaft wiederholen Schleife.



Nun klicken wir den Motor am O1 mit der rechten Maustaste an und wählen Duplizieren.

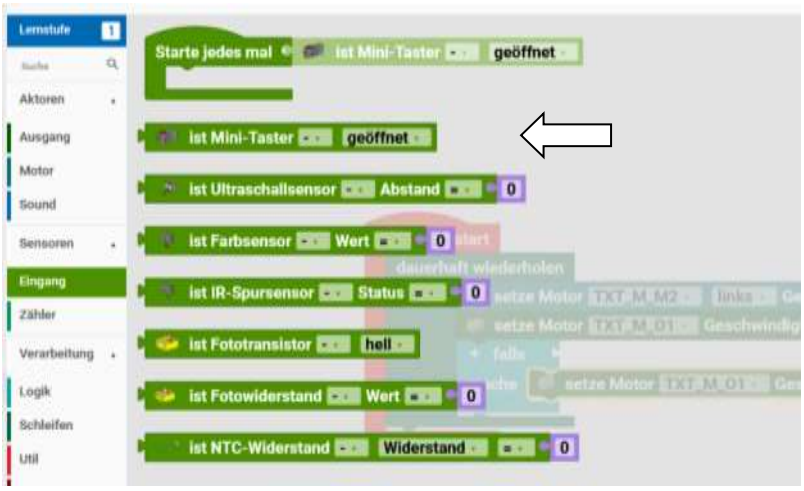


Wir müssen diesen nun etwas in das "falls/mache" verschieben.



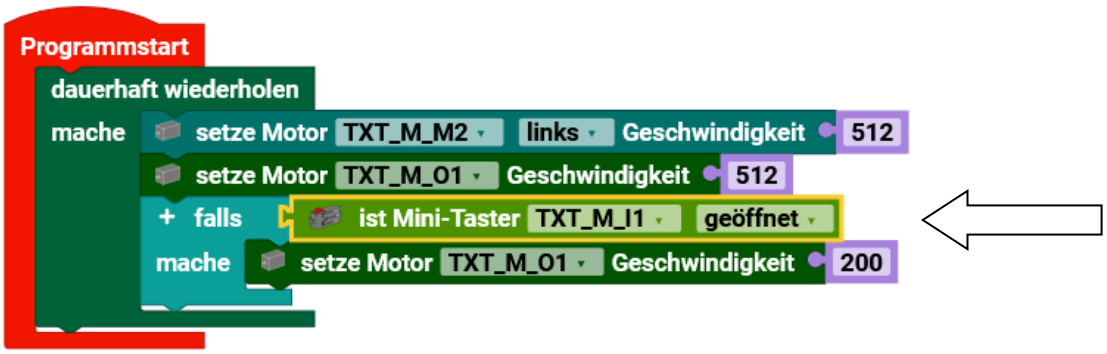
Nun ändern wir den Wert 512 auf 200. Dazu klicken wir auf die 512.

Hier die 200 eingeben.
Das Programm würde noch nicht laufen, da am "falls" noch was fehlt. Dazu gehen wir auf "Eingang"



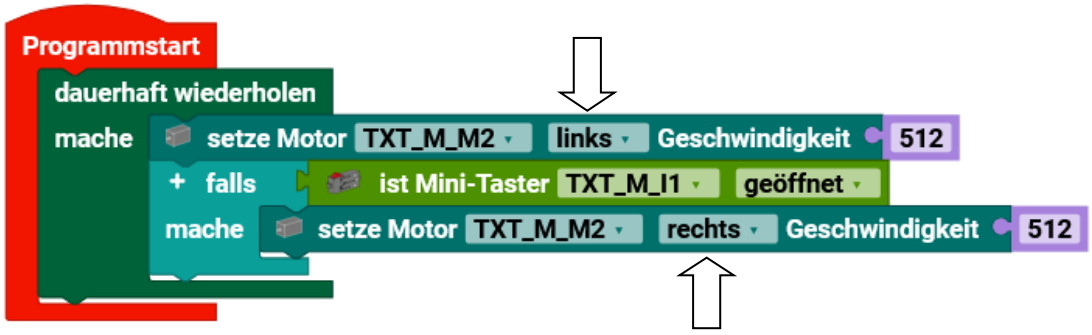
und wählen "ist Minitaster... geöffnet" aus und ziehen es zum "falls".

So sollte das Programm nun aussehen:



Von Robo Pro Coding ist automatisch "I1" ausgewählt worden. Wenn wir das Programm laufen lassen würden. Werden beide Motoren laufen, wobei man mit dem Drücken und Loslassen des Tasters, die Geschwindigkeit des Motors an O1 ändern kann.

Nun kann man mal die beiden Teile von dem O-Motor löschen. Den M-Motor kann man wieder duplizieren und den unteren Eintrag von "links" auf "rechts" ändern. Das Programm ändert nun beim Betätigen des Tasters die Laufrichtung vom Motor M2.
Das Programm sollte nun so aussehen:



Beschreibung siehe unteres Programm.

Frage:

Was würde sich ändern, wenn man statt des Motors eine LED an M2 anschließen würde?

Antwort:

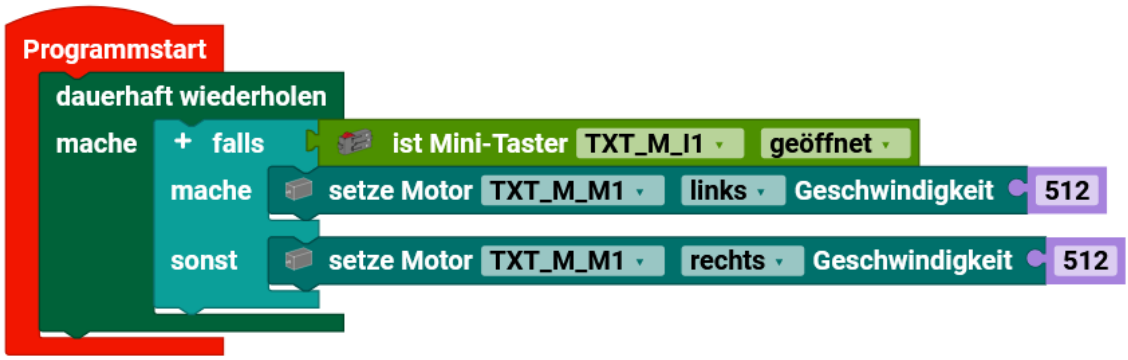
Je nachdem wie rum man die LED angeschlossen hat, leuchtet sie und wird beim Tastendruck ausgeschaltet oder umgekehrt.
Das Kann man z.B. für das Rückfahrlicht verwenden.

Frage:

Was würde sich ändern, wenn man statt des Motors eine Glühlampe an M2 anschließen würde?

Antwort:

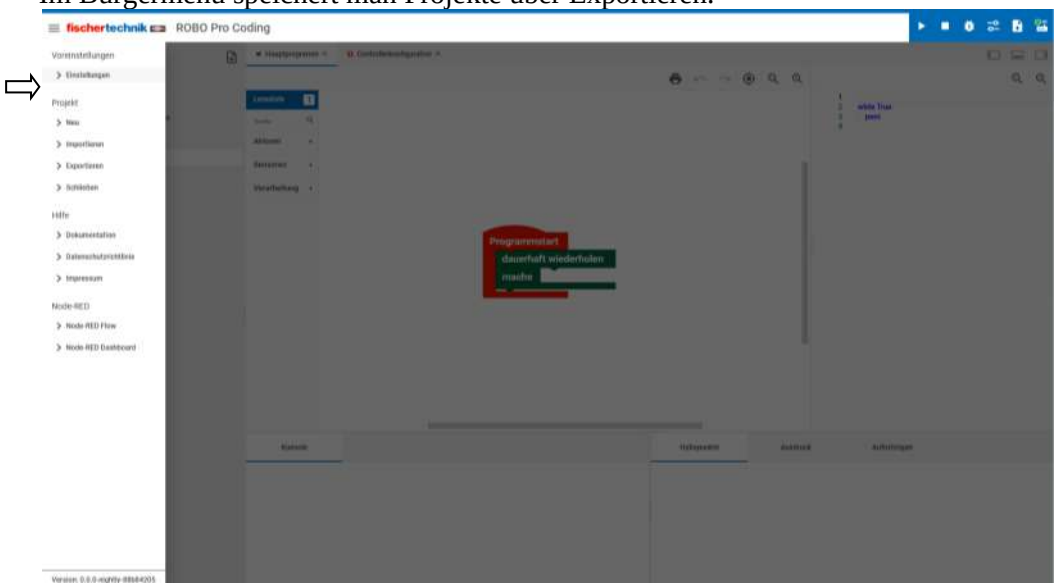
Die Lampe würde immer gleich leuchten. Wenn man einmal die Geschwindigkeit z.B. von 512 auf 200 ändert, würde sie unterschiedlich hell leuchten.



Nicht nur der Unterschied das hier M1 statt M2 ist, sondern auch hier wir der Taster abgefragt und dann erst entschieden, ob er Links oder Rechts drehen soll. Beim oberen Programm wird der Motor Linksrum angesteuert und dann abgefragt, ob der Taster gedrückt ist und dann erst rechts rum laufen gelassen. Der Motor wird also –immer- ganz kurz Linksrum laufen gelassen. Meistens merkt man das aber nicht.
Die bessere Programmierung ist aber die untere.

Programme/Projekte speichern

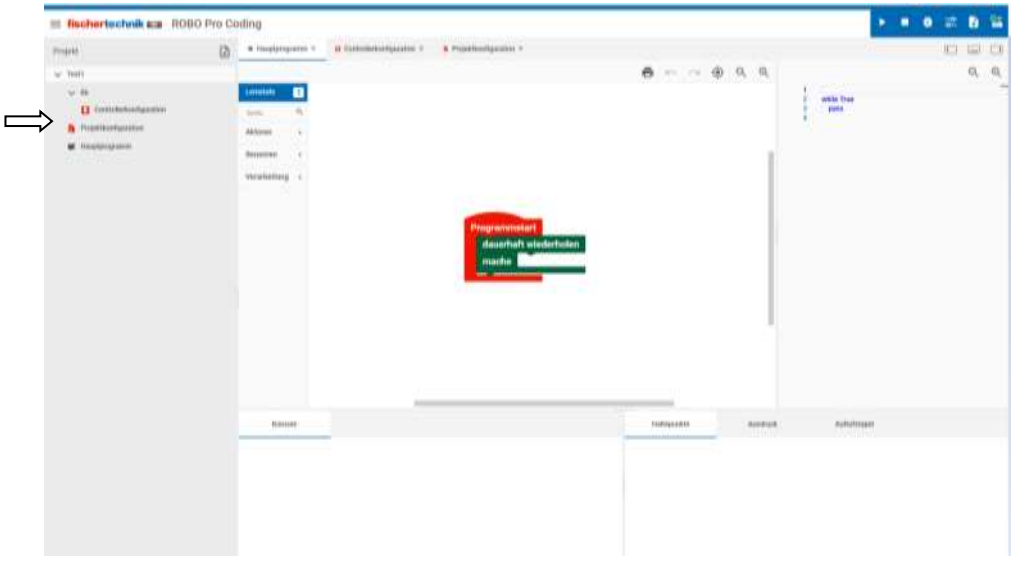
Im Burgermenü speichert man Projekte über Exportieren.



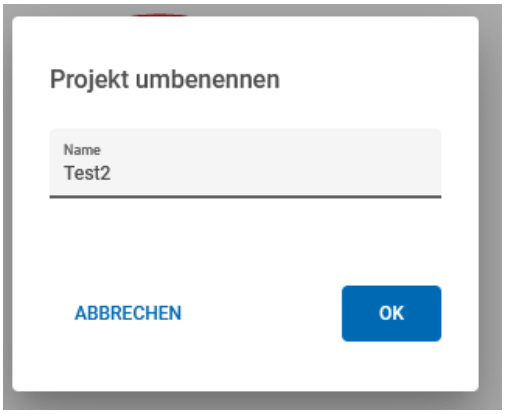
Es erscheinen zwei Möglichkeiten:

Lokal ist auf dem PC und GitLab der fischertechnik Cloud. Wenn man auf Lokal" und "nächste" klickt, kommt bei Dateiname "Untiteld", ohne Namen. Man hier einen Namen z.B. Test1 eingeben und auf "Exportieren" klicken. Das Programm wird in dem Standartordner vom Browser für Downloads gespeichert. Das ist meistens "Downloads", kann aber auch ein anderer sein. Da sollte man drauf achten. Durch die Eingabe des Namens "Test1" hat auch das Projekt den Namen Test1 bekommen.

Hinweis: Für die kostenlose fischertechnik Cloud muss man sich einmalig anmelden. Passwort merken!!



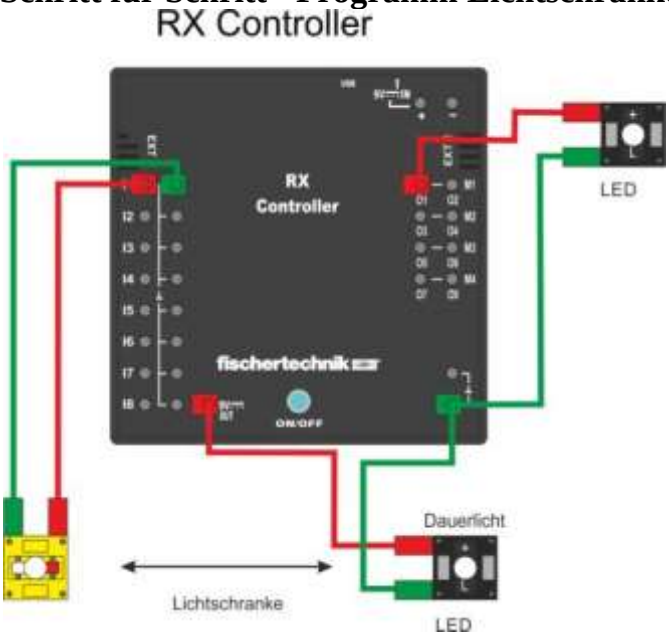
Man kann es auch umbenennen, wenn man auf Test1 mit der rechten Maustaste klickt.
Hier nun Test2 und OK.



Wenn man nun noch mal exportiert, wird Test2 als Name angegeben. Man kann also verschiedene Versionen oder Zwischenstufen, des Programms was man speichert, erzeugen.
Wenn man den gleichen Namen nimmt, wird das vorhandene Programm überschrieben. Das gilt auch für die ft-Beispielprogramme, wenn man diese z.B. sich runtergeladen hatte und diese in einem Unterordner sind. Wenn man also Veränderungen vornimmt, sollte man auch den Namen oder Nummer ändern. Besser ist es sich einen anderen Ordner zu erzeugen und dort die eigenen Programme speichern. Auch sollte man dem Ordner und den Programmen einen aussagekräftigen Namen geben.

Hinweis: Es kann sein, dass Robo Pro Coding beim Speichern, die Leerzeichen in Unterstrich ändert. Das kann auch erst beim zweiten Mal speichern Passieren.

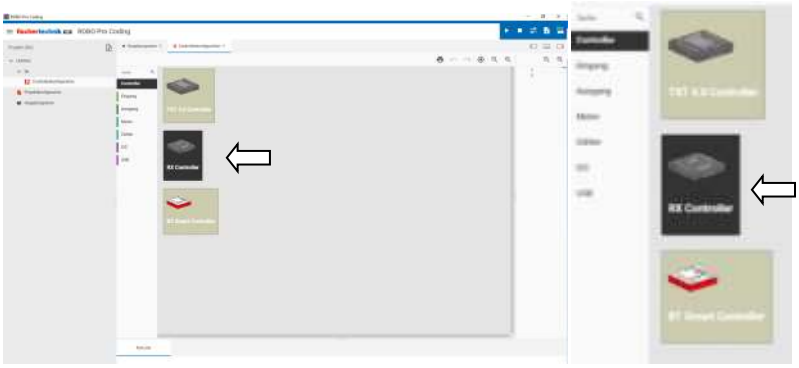
Schritt für Schritt - Programm Lichtschranke mit dem RX Controller



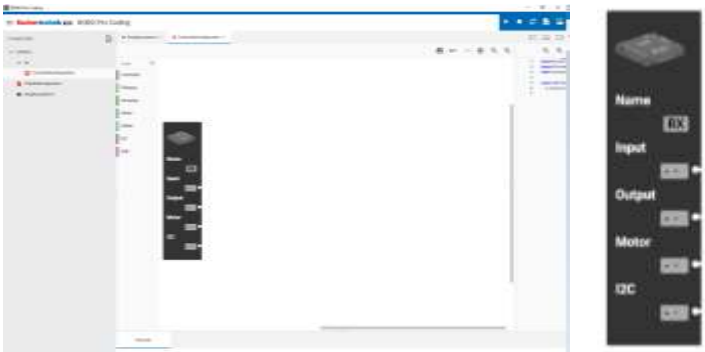
Man kann auch so eine Lichtschranke aufbauen. Besser ist es aber statt dem O1 Ausgang einen von beiden Plus-Buchsen unten zu nehmen. Auch diese werden beim Ausschalten vom Controller mit abgeschaltet.

Starten wir ein neues Projekt.

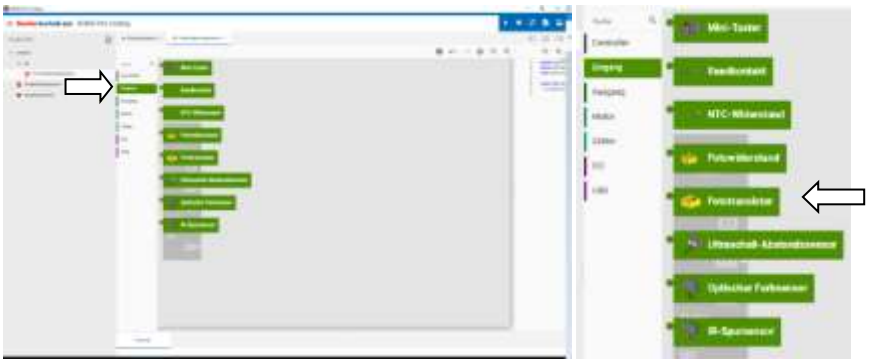
Beispiel hier ist mit Lernstufe 3. Deswegen werden hier mehr Symbole angezeigt. Man kann das aber auch mit Lernstufe 1 machen.



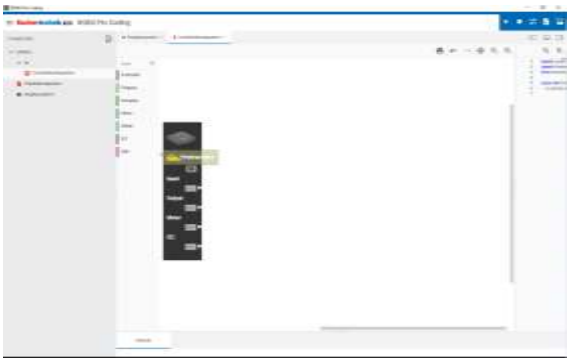
Unter Controllerkonfiguration ist der RX Controller schon farbig, weil er mit Robo Pro Coding verbunden ist. Die beiden anderen Controller sind nicht verbunden und somit ausgegraut. Klicken wir mal auf den RX.



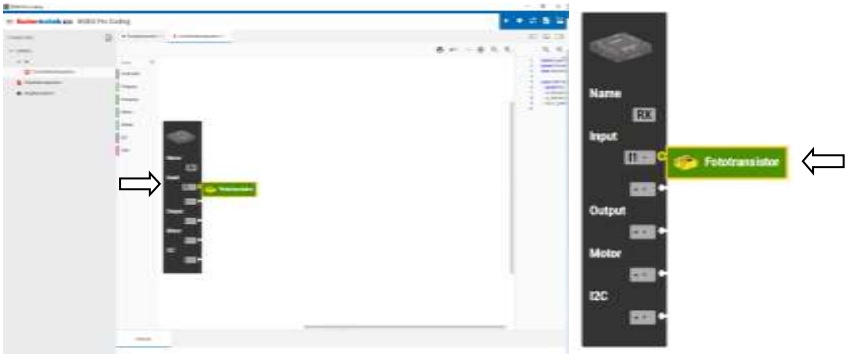
An dem RX Controller sind noch keine Blöcke angeschlossen.



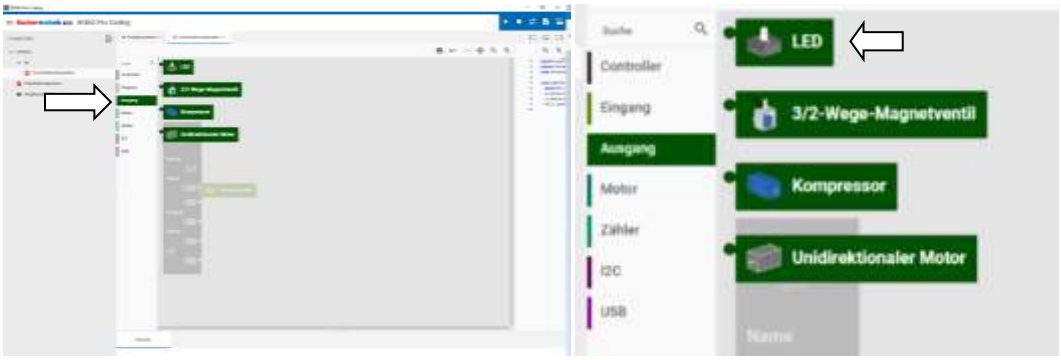
Gehen wir auf Eingang und klicken auf den Fototransistor



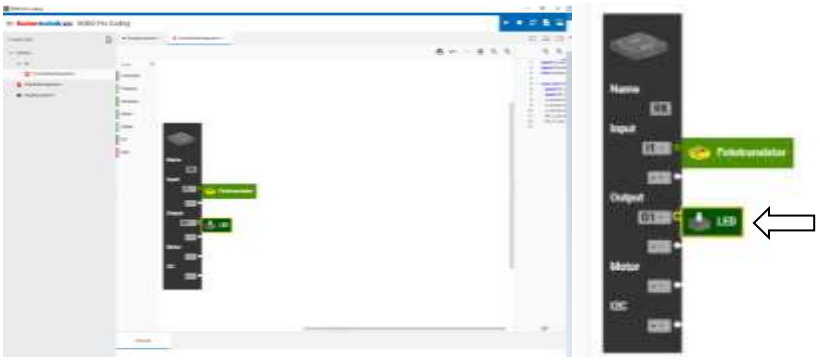
Der Fototransistor wird auf dem RX Controller abgelegt



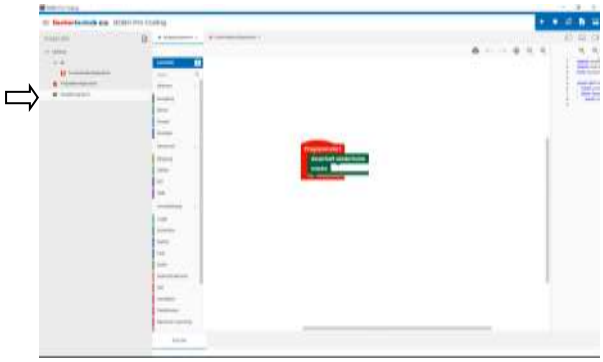
und man muss ihn an den Eingang rüber ziehen.



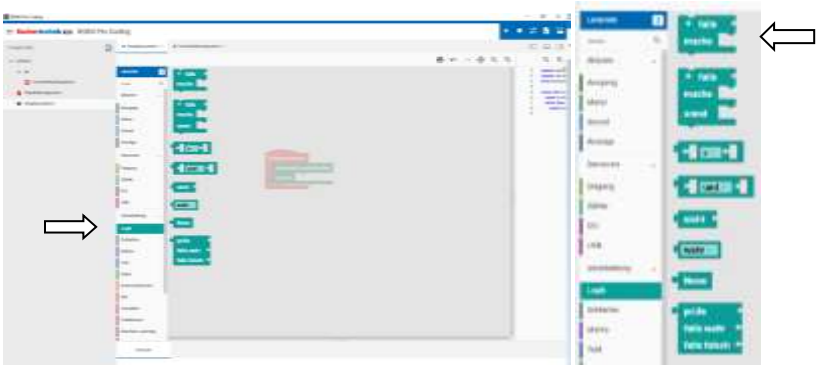
Klicken wir auf Ausgang und wählen die LED.



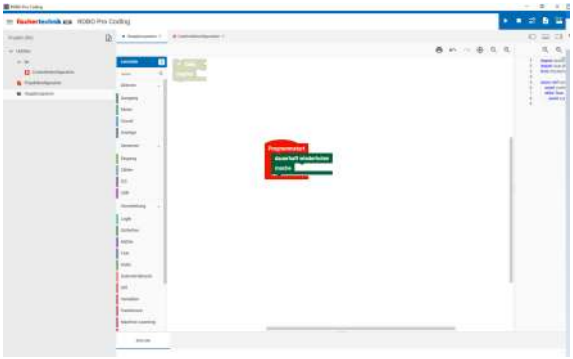
Und ziehen sie an den Ausgang. So sollte die Controllerkonfiguration aussehen.



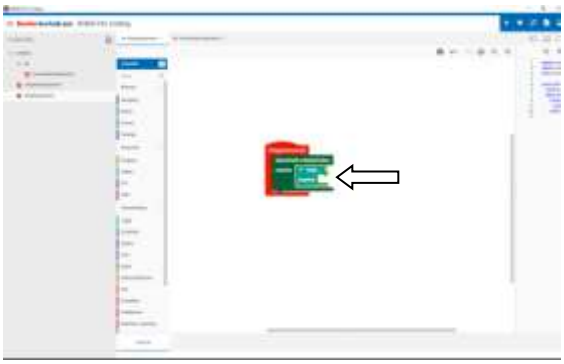
Klicken wir auf Hauptprogramm

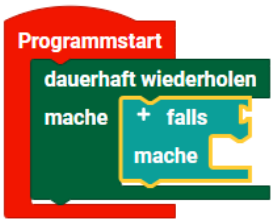


Und dann auf Logik und kicken auf „Falls mache“

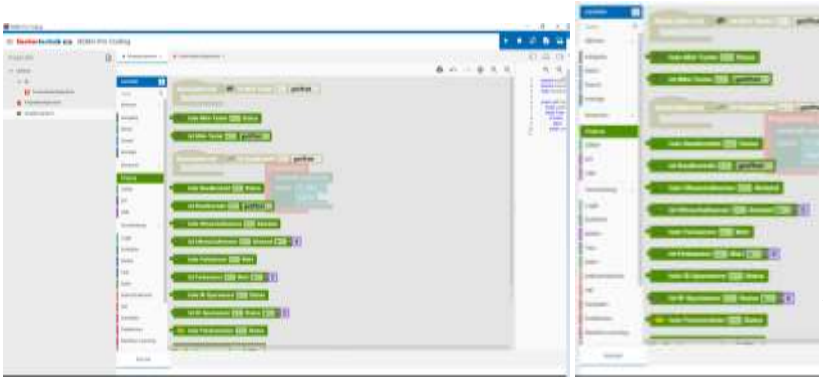


Der Baustein wird ausgegraut abgelegt

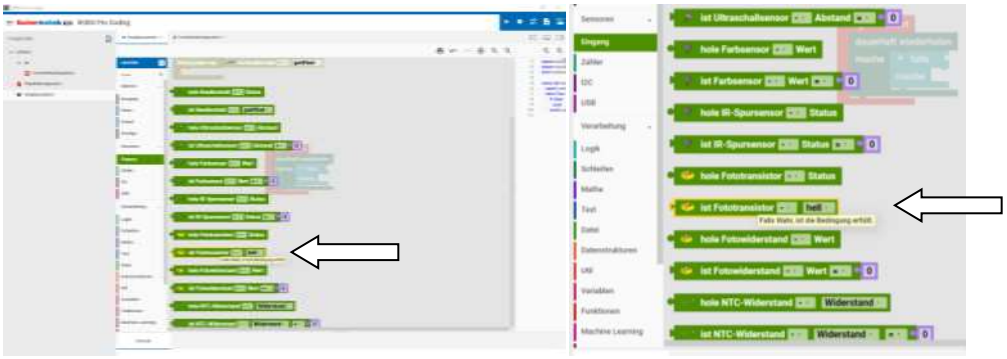




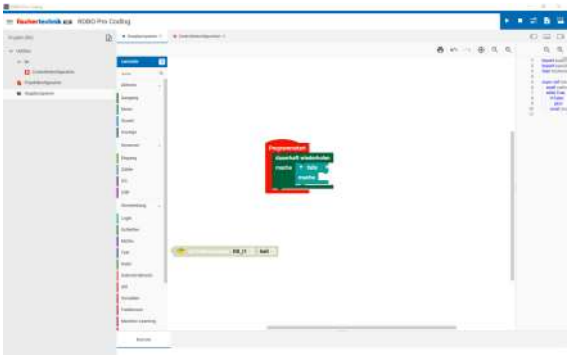
Und muss nun an die richtige Stelle bei „dauerhaft Wiederholen“ abgelegt werden.



Hier klicken wir auf Eingang. Der Fototransistor ist aber noch nicht sichtbar (Lernlevel 3 = mehr Blöcke) und man muss mit dem Mausrad nach unten scrollen.

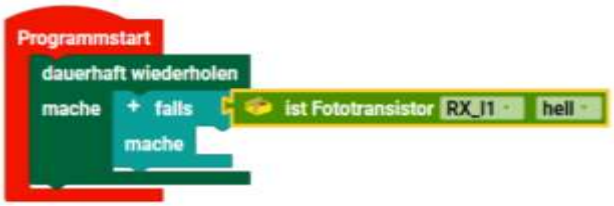


Hier den Eingang „ist Fototransistor…hell“ anklicken.



Der Block wird abgelegt…

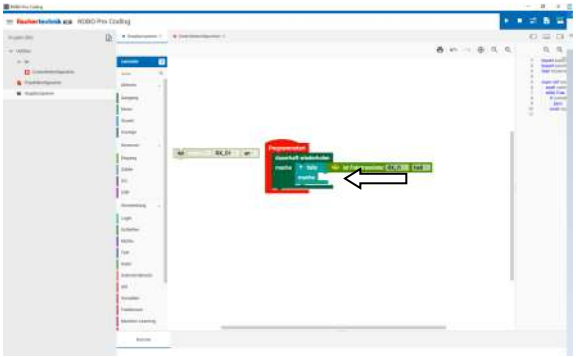
und muss an das „falls“ rüber gezogen werden.



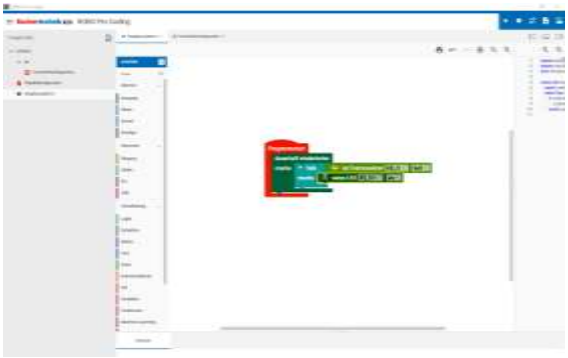
So sollte es nun aussehen

Gehen wir nun auf Ausgang

Ausgang "setze LED an" anklicken



und an das mache rüber ziehen.

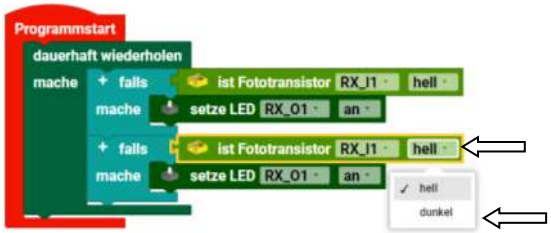


Wenn wir dieses Programm starten würden, geht die LED an, sobald der Fototransistor beleuchtet wird. Aber sie bleibt an, weil sie nie abgeschaltet wird. Jetzt könnte man einen weiteren "falls mache" Block einfügen, mit Fototransistor dunkel und LED aus. Dann würde es so aussehen.

Dazu auf das "falls mache" mit der rechten Maustaste drücken und Duplizieren auswählen. Dabei werden auch die angeschlossenen Blöcke dupliziert und daneben abgelegt.

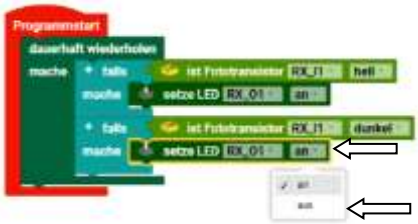
Diese sind noch ausgegraut, weil sie noch nicht richtig angeschlossen sind. Das zweite "falls mache" unter das erste "falls mache" ziehen.

Der rote "Programmstartblock erweitert sich und man kann den zweiten Block ablegen.



Die Blöcke ändern die Farbe und klicken beim zweiten Fototransistor mit der linken Maustaste auf "hell" und wählen "dunkel" aus.

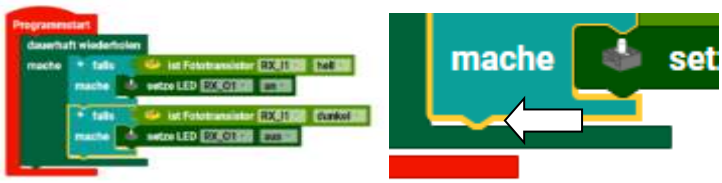
So sollte es nun aussehen



Bei der unteren LED ändern wir „an“ in „aus“.

Herzlichen Glückwunsch, ein lauffähiges Programm.

Das Programm schaltet die LED an, wenn der Fototransistor beleuchtet wird und schaltet die LED aus, wenn er dunkel ist.



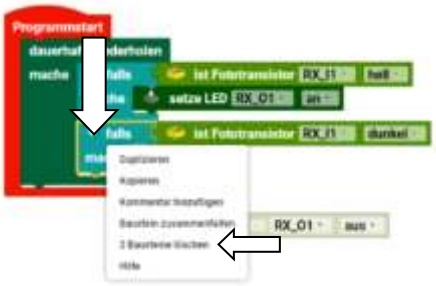
Wenn man sich das Programm mal genau anschaut, sieht man unten eine kleine Nase an dem zweiten "falls mache".

Man kann dort weitere Blöcke anfügen. Und hier läuft das Programm. Es läuft auch dort, wenn die

Bedingung von dem ersten falls nicht erfüllt ist. Man kann also genau dort die LED ausschalten, ohne weiter abzufragen.

Ändern wir das Programm…

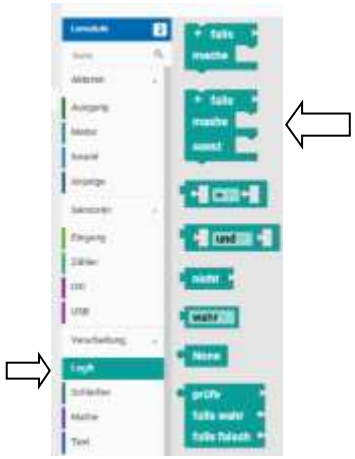
und ziehen die untere LED raus an die Seite.



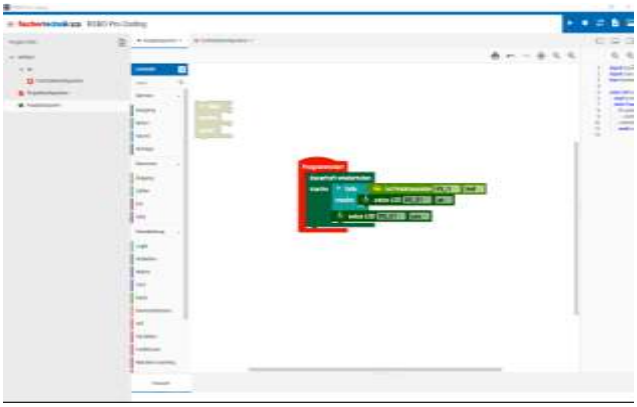
Man könnte nun den zweiten "falls mache" Block mit links anklicken und die Entfernen-Taste drücken oder wie hier, mit der rechten Maustaste klicken und auf 2 Bausteine löschen klicken.

Der zweite "falls mache" Baustein und der "ist Fototransistor..." Baustein sind nun gelöscht.

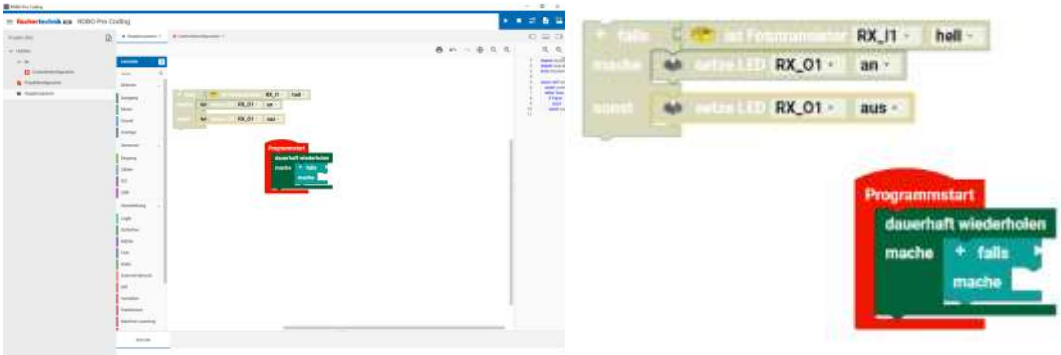
Wir ziehen den LED Baustein unter das "falls mache". Wenn wir das Programm laufen lassen, stellen wir fest, dass die LED nur minimal angesteuert wird, wenn der Taster gedrückt wird. Sie wird zwar eingeschaltet aber sofort danach wieder Ausgeschaltet. Der "falls mache" Block ist also nicht ganz der richtige.



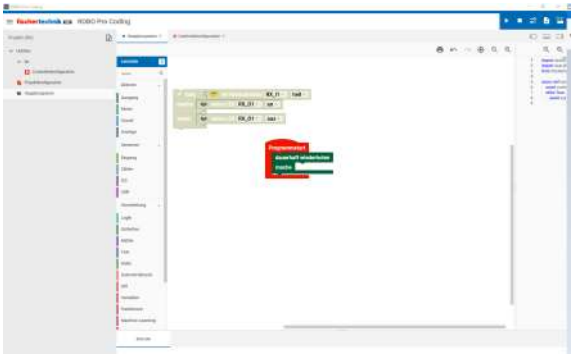
Gehen wir noch mal auf Logik. Dort finden wir unter dem "falls mache" einen weiteren Block. Den "falls mache sonst" Block.



Den klicken wir mal an.

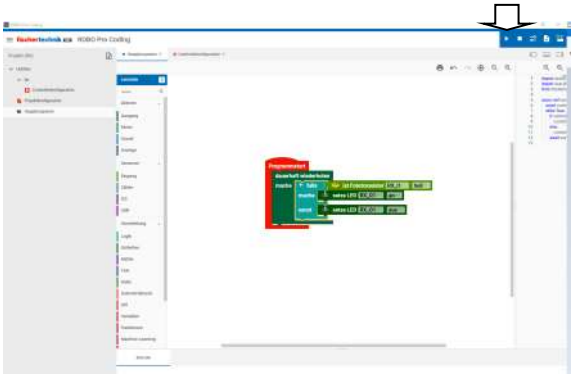


Und ziehen die Blöcke am "falls mache" zum "falls mache sonst" rüber.



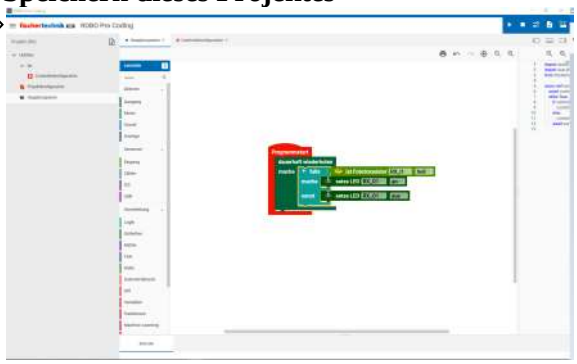
Dann löschen wir den "falls mache" Block und…

ziehen den "falls mache sonst" Block mit den anhängenden Blöcken in das "dauerhaft wiederholen".

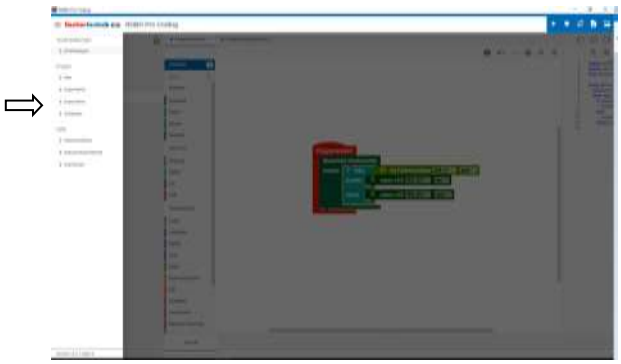


Wenn wir nun das Programm starten (über das Dreieck oben rechts) verhält sich das Programm genauso wie wir es erwartet haben.

Speichern dieses Projektes

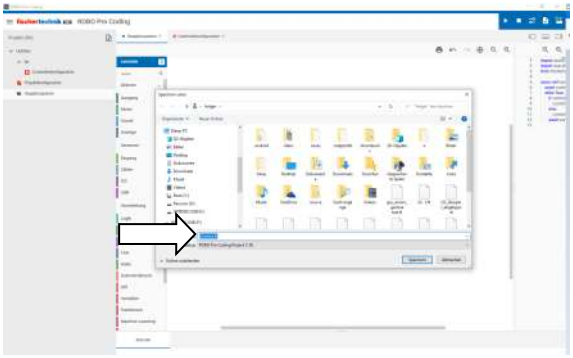


Dazu das Burgermenü oben links anklicken.



Dann auf Exportieren

Auf Lokal und "Nächste"



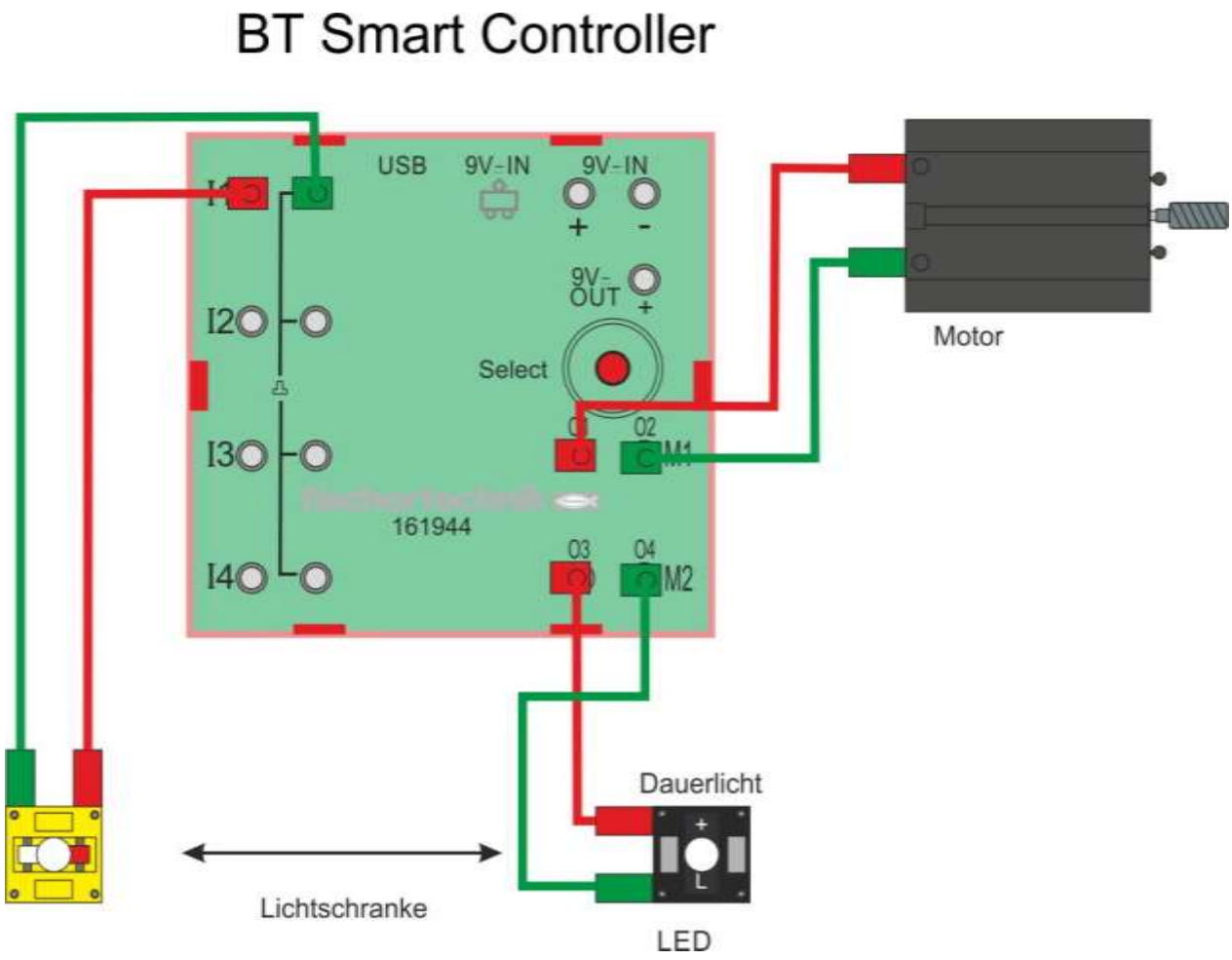
Hier kann man dem Projekt einen Namen geben und speichern.

Und damit ist das Projekt gespeichert.

Herzlichen Glückwunsch, sie haben ein fertiges Programm erschaffen.

Schritt für Schritt - BT Smart Controller mit Programm für Lichtschranke und Motor

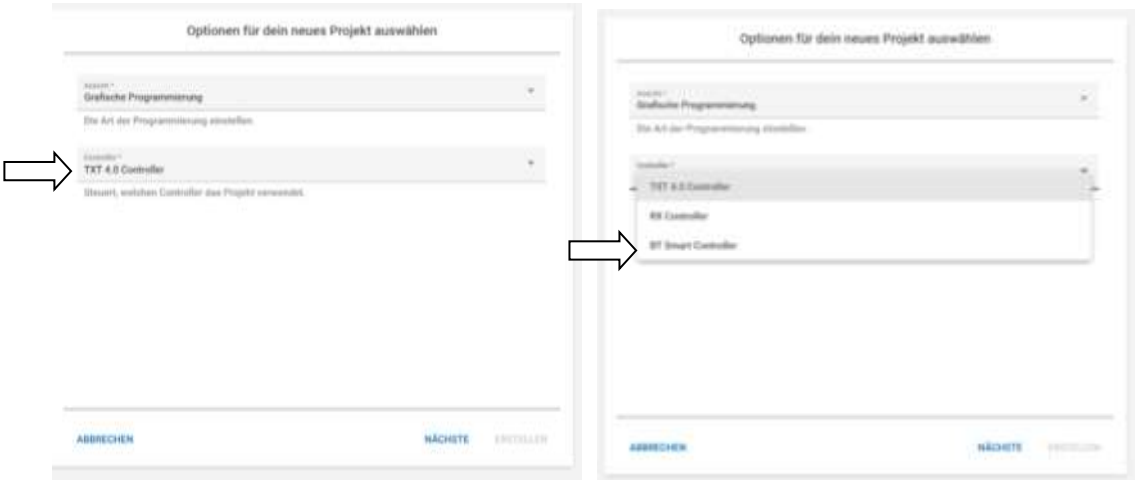
Wir programmieren nun eine Lichtschranke, die einen Motor steuert.



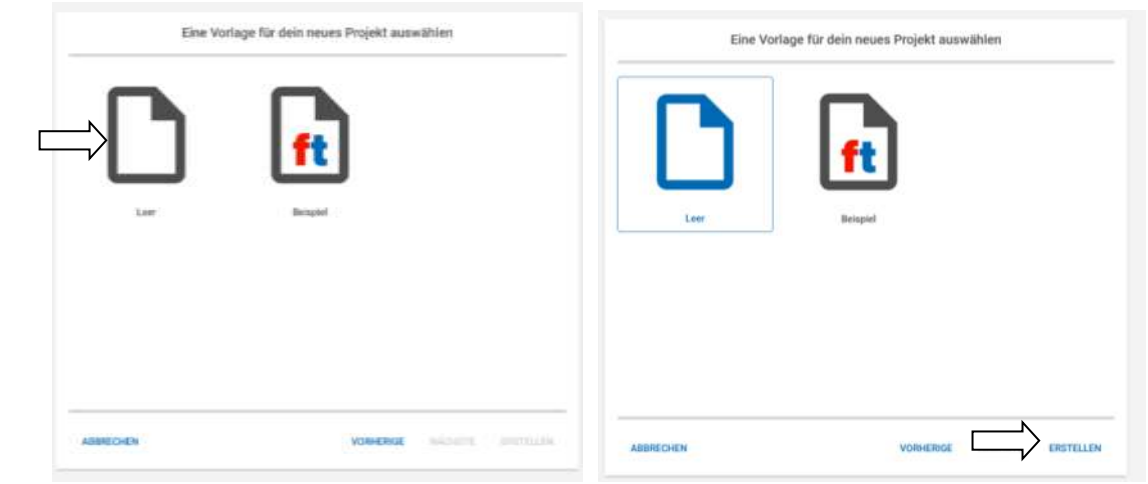
Robo Pro Coding

Wir klicken auf das Burgermenü

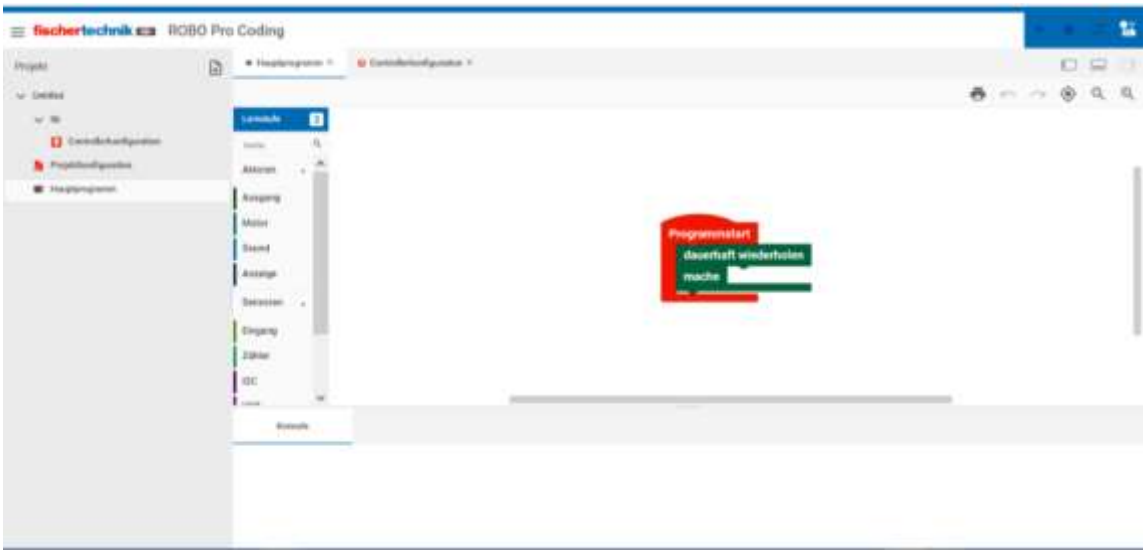
und dann auf Projekt/Neu.



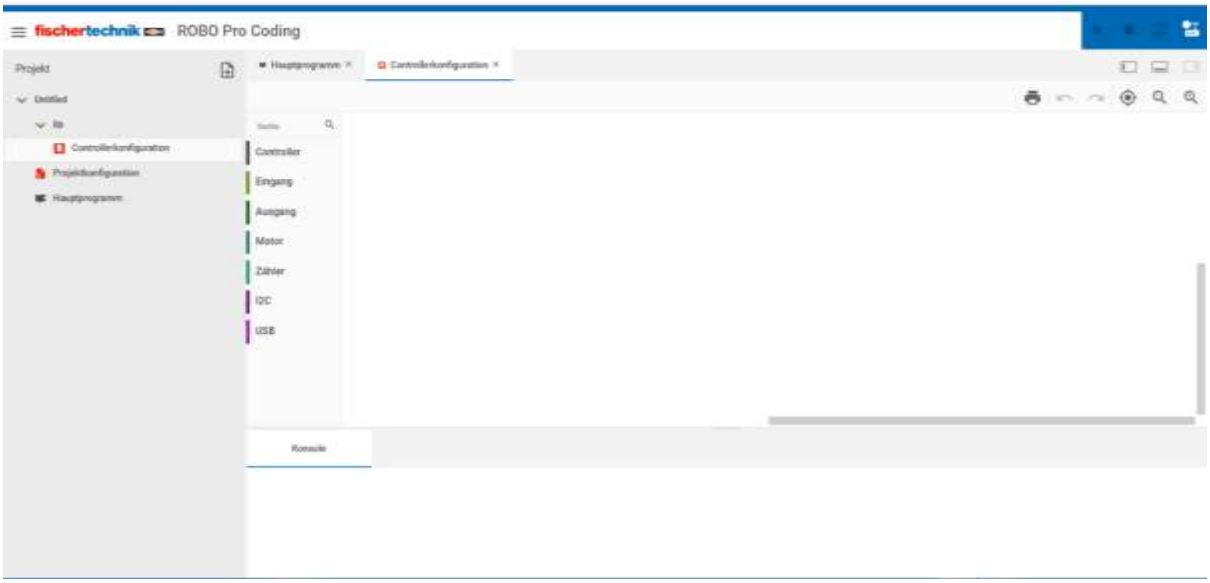
Nun auf Controller klicken und den BT Smart Controller klicken und auf „Nächste“.



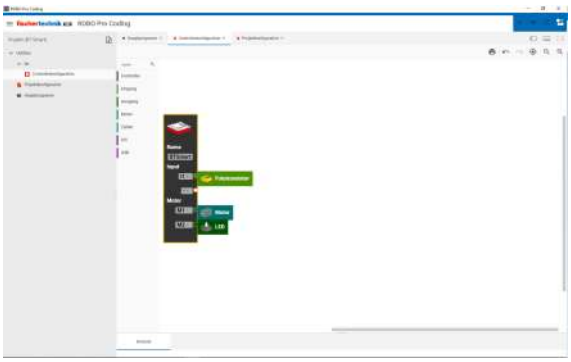
Hier auf die Vorlage „Leer“ klicken und Erstellen.

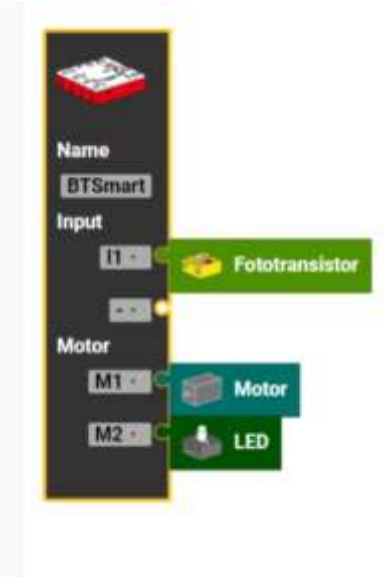


Es kommt das Leere Projekt.

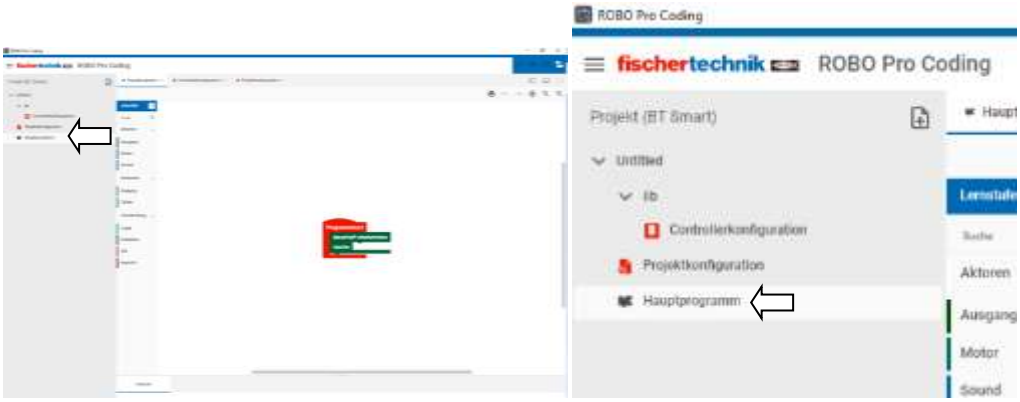


Wir gehen in die Controllerkonfiguration und wählen den BT Smart Controller, LED mit Fototransistor und Motor aus.

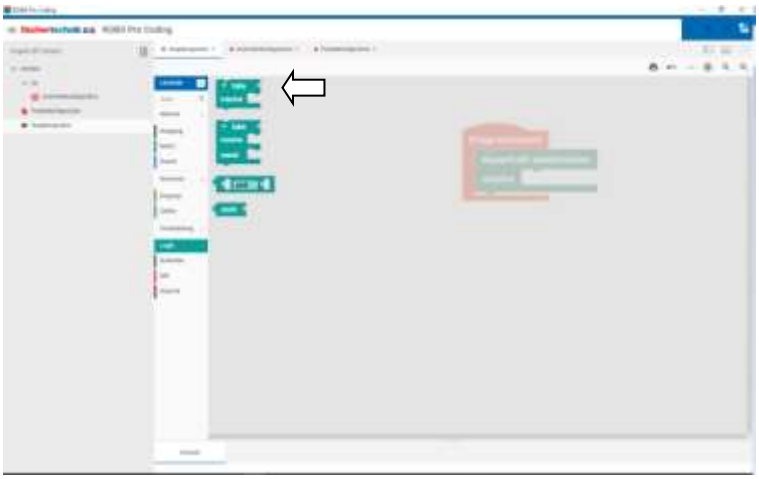




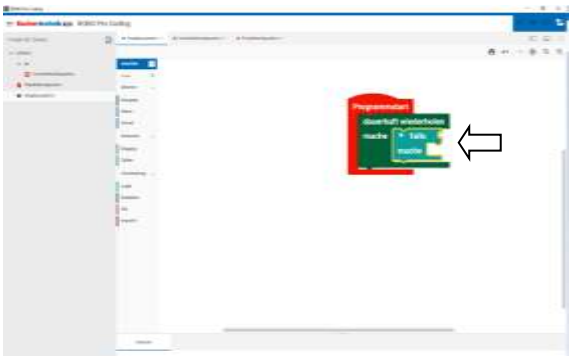
BT Smart Controller mit Fototransistor, Motor und LED



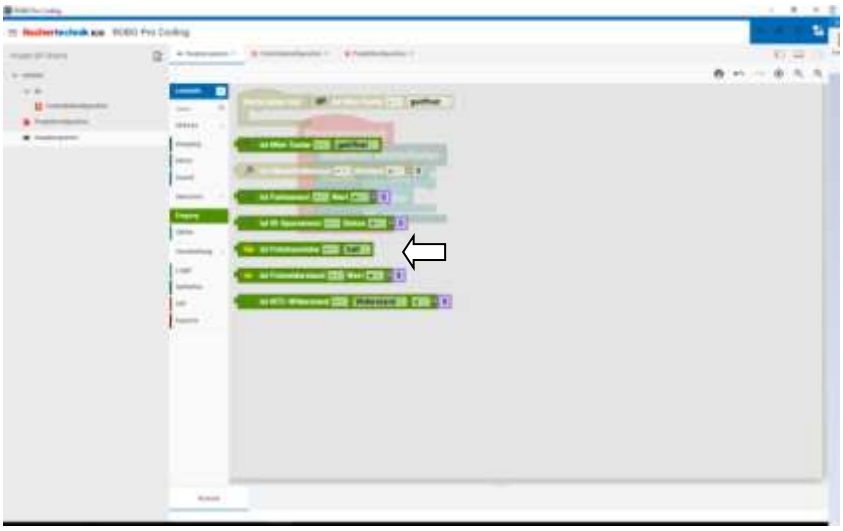
Nun machen wir ein leeres BT Smart Controller-Programm. Es ist schon da. Einfach auf Hauptprogramm gehen.



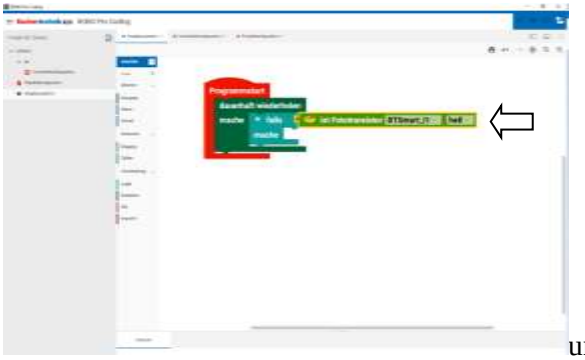
Unter Logik "falls/mache" auswählen



und einfügen.

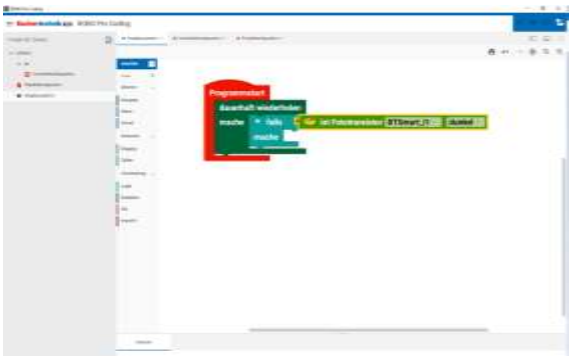


Unter "Eingang" "ist Fototransistor hell" auswählen



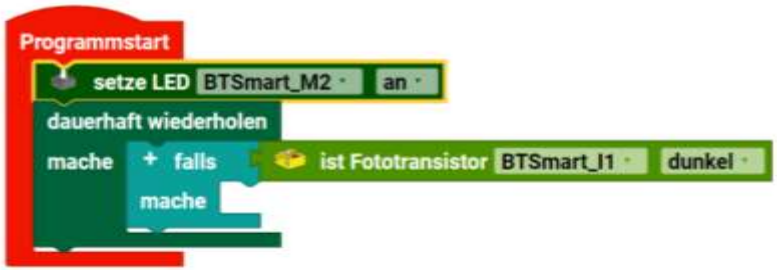
und rüber ziehen.

Hier hell in dunkel ändern.

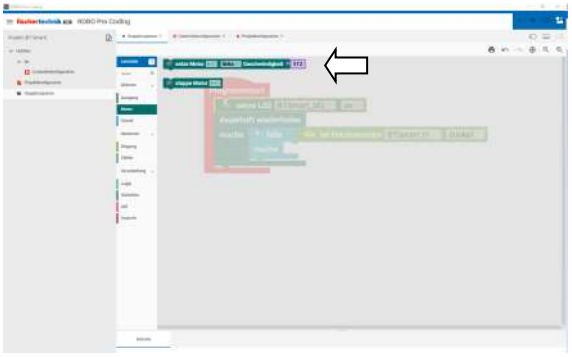


Es fehlt die LED für die Lichtschranke.

Unter Ausgang "setze LED an"

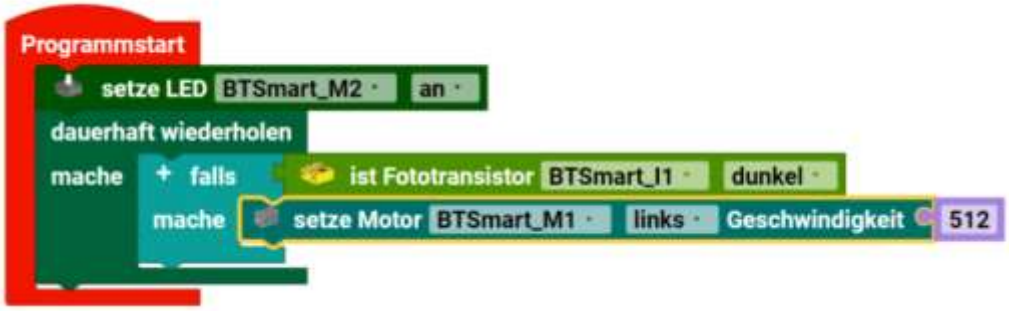


Man könnte sie auch in die Dauerschleife einbinden, was aber unnötig ist. Sie braucht ja nicht immer und immer wieder nur eingeschaltet werden. Die Lichtschranke ist also fertig. Nun soll etwas passieren, wenn die Lichtschranke unterbrochen wird.



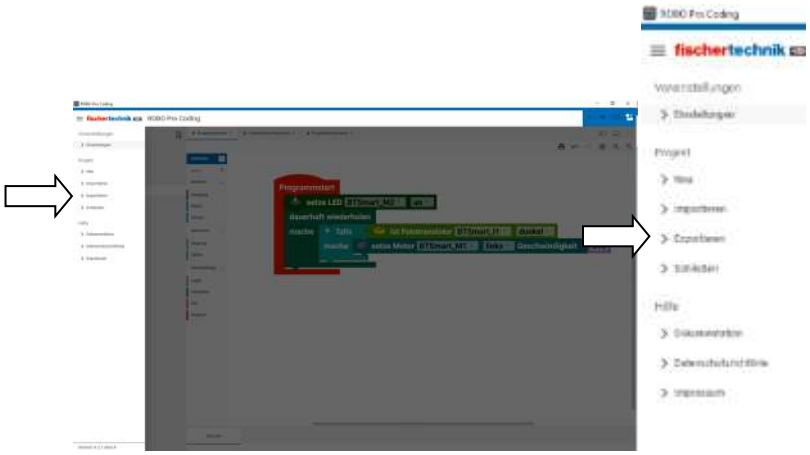
Dazu nehmen wir einen Motor und lassen ihn drehen.

Diesen setzen wir bei "mache" ein. Wenn der Fototransistor dunkel ist, wird der Motor mit voller Geschwindigkeit laufen.



Das ganze Programm der Lichtschranke, die einen Motor steuert.

Projekt Speichern / Exportieren
Das Speichern geht über Projekt Exportieren. Das findet man im Burgermenü:



Datei speichern / exportieren in dem Burgermenü

Danach kann man sich entscheiden, ob man „Lokal“ auf dem PC oder in der kostenlosen fischertechnik Cloud speichern möchte.

Wenn man auf „Lokal“ klickt und „weiter“ kann man hier den Speicherort angeben und speichern.

Herzlichen Glückwunsch! Sie haben ein Programm bzw. Projekt erschaffen.

Ein Programm wird mit der Endung .ft gespeichert. Das entspricht einer Text-Datei. Die Blöcke sind durch entsprechende Einrückungen gekennzeichnet.

Hinweis zu den Endungen bei Projekt-/Programmnamen und anderen Programmiersprachen

Robo Pro Coding = xxxx.ft
Robo Pro Coding Python Programm = xxxx.py

Andere Endungen von anderen älteren Programmiersprachen lassen sich **nicht** öffnen:

Robo Pro Light = xxxx.rpl
Robo Pro Smart = xxxx.xml
Robo Pro = xxxx.rpp
fischertechnik Sratch 3 xxxx.sb3

Wo sind meine Dateien? Die Sache mit der Dateiendung.

Es gibt bei Windows die Möglichkeit, Dateiendungen einzublenden. Scheinbar hat das aber auch die Wirkung, dass die Programme ohne das .ft im Namen gespeichert werden. Man muss es per Hand hinzufügen oder im Explorer den Dateinamen nachträglich ändern. Wenn man ein Projekt importiert, kann es sein, dass nur die Dateien mit .ft angezeigt werden. Man kann nun aber auch alle Dateien anzeigen lassen. Das ist aber unübersichtlicher.

Wenn Dateien in der Cloud gespeichert werden sollen, lege ich mir zusätzlich immer eine Kopie auf meiner Festplatte an.

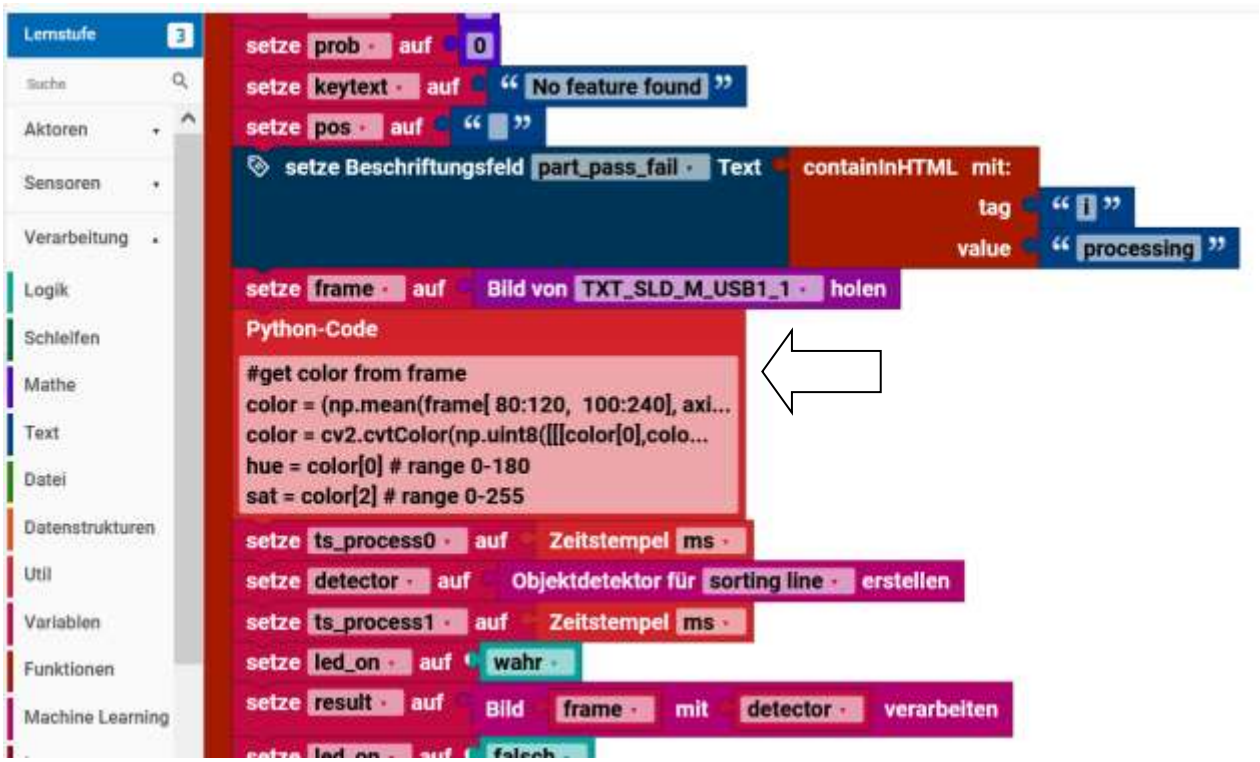
Python Befehle innerhalb von Robo Pro Coding

In den meisten fertigen Robo Pro Coding Programmen, sind Blöcke wo Python-Code steht.

Im Reiter „Util“ gibt es den Block Python-Code. In diesen Block können direkt wie bei einer Textverarbeitung Python-Befehle eingetippt werden.

Tipp: Man kann diesen Block auch dazu nutzen Erklärungen/Kommentare in das Programm zu schreiben, statt den „Kommentar“ des Blocks selbst zu benutzen. (fischertechnik macht das auch so)
Man setzt vor den Text einfach das „Gartenzaun“-Zeichen #. Alles dahinter wird nicht ausgeführt. Man kann also auch vor Python-Befehlen das # setzen, um z.B. testweise Zeilen aus dem Programmablauf raus zu nehmen.

Beispiel AddOn KI:



Wenn man auf den Bock klickt, sieht man den vollständigen Programm-Code. Oben ist er mit drei Punkten abgekürzt.

```
#get color from frame
color = (np.mean(frame[ 80:120,   100:240], axis=(0, 1)))
color = cv2.cvtColor(np.uint8([[[color[0],color[1],color[2]]]]),cv2.COLOR_BGR2HLS)[0][0]
hue = color[0] # range 0-180
sat = color[2] # range 0-255
```

Es steht als erstes ein Kommentar und in jeder neuen Zeile ein Befehl.

Wichtiger Hinweis! OpenCV ermittelt **andere** Zahlenwerte (Farbraum) für die Farben als die Farberkennung der Kamera und des Gestensensors. Am besten die Werte –im- Modell prüfen (Fremdlicht. Fenster…)

Wie kommen die Variablen/Daten von Robo Pro Coding zu Python und umgekehrt?

Ganz einfach. Es gibt keinen Unterschied von den Variablen-Namen, in Robo Pro Coding und Python.

Beispiel:

Hier wird die Variable „color“ auf den Wert 300 gesetzt. Im oberen Python-Code-Block gibt es den Befehl das eine Variable „color“ auf 400 gesetzt wird. Als letzter Befehl wird der Block „gib aus“ und die Variable „color“.

Wenn man das Programm laufen lässt, wird auf der Konsole 400 ausgegeben.

Beide „color“ Variablen sind gleich, da „color“ zuvor in Robo Pro Coding als Variable bereits angelegt wurde. Wenn man andere Variablen benutzen möchte, muss man sie am Programm-Anfang selber eintragen.

Vom Programmablauf her, hat die die Variable „color“ also erst den Wert 300 und nimmt diesen Wert mit in das Pythonprogramm. Dort wird die 400 reingeschrieben und zurück zu Robo Pro Coding mitgenommen.

Im Python Beispiel vom AddOn Ki oben…

```
#get color from frame
color = (np.mean(frame[ 80:120,   100:240], axis=(0, 1)))
color = cv2.cvtColor(np.uint8([[[color[0],color[1],color[2]]]]),cv2.COLOR_BGR2HLS)[0][0]
hue = color[0] # range 0-180
sat = color[2] # range 0-255
```

…wird einer Variablen „color“, aus einem von der Kamera aufgenommenen Bild, die konvertierten RGB-Werte, in HLS-Werte umgewandelt. Hier color[0] = hue (Farbe); color[2] = sat (Sättigung) (und auch color[1] = lightness (Helligkeit)).

Befehle von Robo Pro Coding (Reiter)

Auflistung aller Befehle. Ob nun der Befehl auch mit dem Controller funktioniert, den man hat, kann man in der Befehlsübersicht zu den Controllern am Ende des Buches nachschauen.
Ich habe die Hilfefunktion zu Robo Pro Coding als Grundlage genommen und –alle- Blöcke aufgelistet. Dazu gibt es lauffähige Beispiele zu fast jedem Befehl.

Blöcke Suchen

In Robo Pro Coding kann man auch nach Blöcken suchen.

In diesem Beispiel wird nach „mo“ gesucht, um alle Blöcke, die mit Motor zu tun haben zu finden. In der Mitte ist das Ergebnis zu sehen. Man sollte aber normalerweise mehr Buchstaben benutzen, um die Suche besser einzugrenzen.

Aktoren

Aktoren werden für die Ausgabe benutzt, wie Lampen, Motoren, Kompressor, Sound und Anzeigen.

Ausgang

Ausgang

Der Starte jedes mal-Block

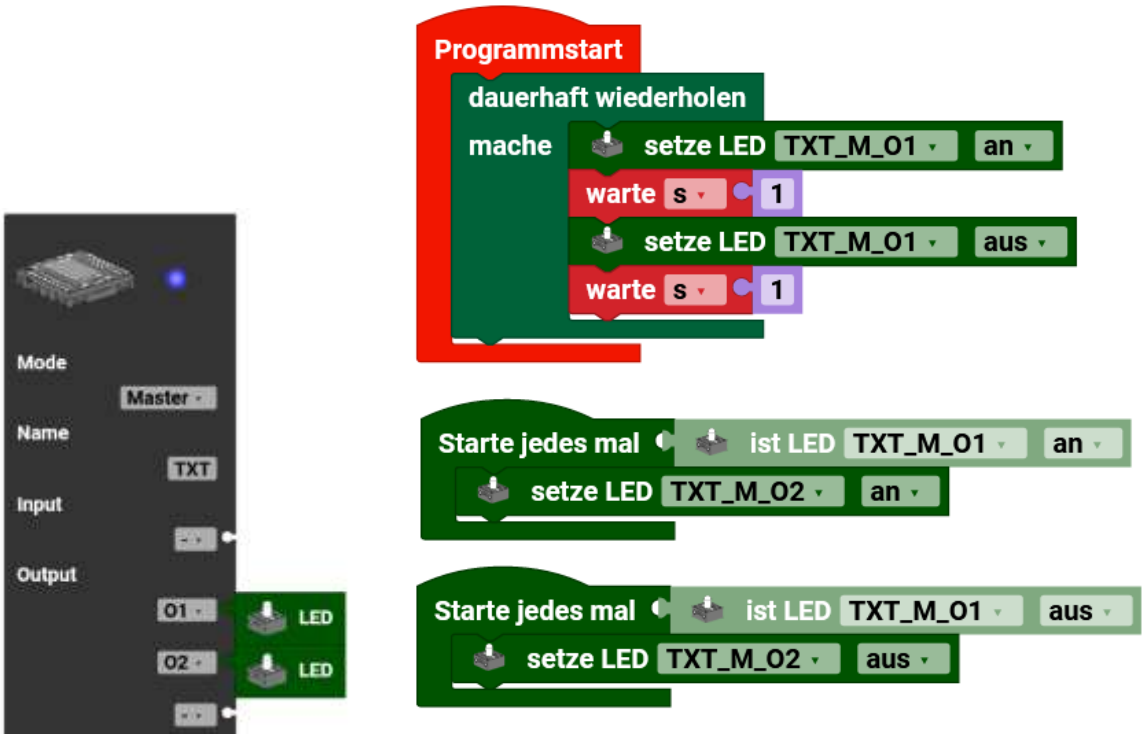
Der **Starte jedes mal-Block** bietet die Möglichkeit ein (Parallel-) Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der **Starte jedes mal-Block**:

Hier wird ein Programm ausgeführt, wenn die LED an ist. Hier fehlen aber noch das Programm und der Ausgang, wo die LED angeschlossen ist. Wenn man vorher keinen Ausgang mit einer LED in der Controllerkonfiguration angeschlossen hat, kann man den Block nicht in das eigene Programm einfügen. Der Rechte Teil ist heller, da es ein Vorschlag von Robo Pro Coding ist. Wenn man nichts ändert, wird er im Programm ausgeführt.

Der Block ist eine Abkürzung für folgendes Konstrukt:

Man kann in den **Starte jedes mal-Block** der Kategorie Ausgänge alle –Bedingungen- (hell rechts vom Block) aus eben nur dieser Kategorie/Reiter einsetzten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.



In diesem Beispiel wird im Hauptprogramm (rot) eine LED am Ausgang O1, jeweils nach einer Sekunde, an und aus geschaltet (Blinker). Je nachdem ob O1 an oder aus ist, wird ein Parallelprogramm ausgeführt. Hier wird eine zweite LED an und aus geschaltet. Wenn man das an/aus bei den Ausgängen O2 tauscht, hat man einen Wechselblinker.

LEDs

Setzen

Mit den Blöcken **setze LED ...** und **setze LED Helligkeit ...** kann man die LED an- und ausstellen beziehungsweise ihre Helligkeit auf einen bestimmten Wert (von 0 bis 512) setzen.

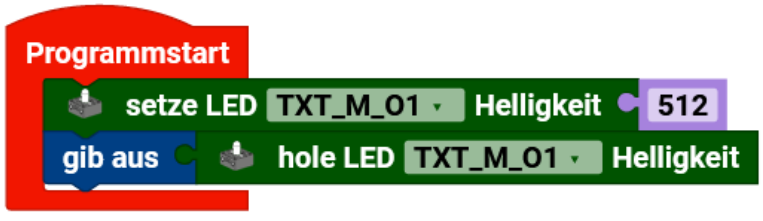
setze LED … Helligkeit

An diesem Block muss rechts ein Wert, Variable oder Berechnung angefügt werden.

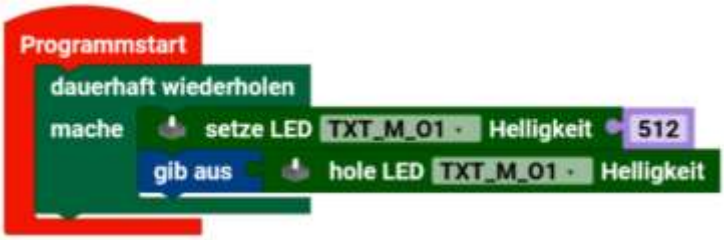
Hier wird eine LED am Ausgang O1 am TXT 4.0 (Master) mit voller Helligkeit eingeschaltet.

hole LED Helligkeit

Dieser Block holt den aktuellen Wert, der Helligkeit einer LED, an einem Ausgang.



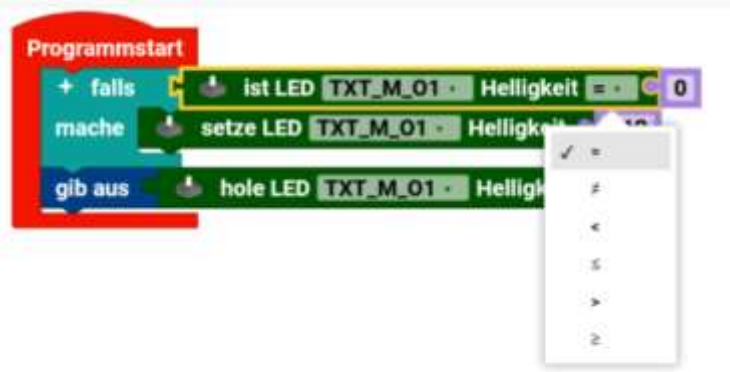
In diesem Beispiel wird die Helligkeit der LED auf 512 gesetzt und der Wert der Helligkeit auf der Konsole ausgegeben. („gib aus“ ist bei Verarbeitung/Text)



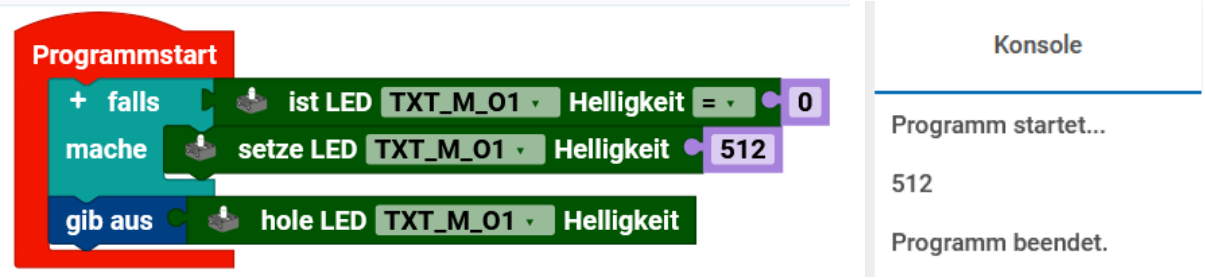
Achtung! Es sieht ähnlich aus, ist aber ein Endlosprogramm. Es kann sein, dass man Schwierigkeiten hat, es zu beenden. Man sollte das Programm vorher speichern!

ist LED Helligkeit

Mit diesem Block wird die Helligkeit abgefragt und dann ein Vergleich gemacht. Das Ergebnis ist wahr oder falsch.



Man kann wahr(gleich), ungleich, kleiner, kleiner-gleich, größer und größer-gleich abfragen. (falls mache ist bei Verarbeitung/Logik)



Das Programm startet und fragt ab, ob die Helligkeit der LED am O1 des TXT 4.0 (Master)
0 Null ist. Wenn sie 0 ist, wird die Helligkeit auf 512 gesetzt. Danach wird der Wert in der Konsole angezeigt.

setze LED an

Dieser Block setzt den O-Ausgang auf „an“ = 512, also volle Stärke. Man kann auch „aus“ anwählen. Dabei wird der Ausgang dann auf 0 gestellt.

Das Programm macht in einer Dauerschleife, die LED am O1 des TXT 4.0 (Master) an, wartet 1 Sekunde, schaltet sie aus, wartet 1 Sekunde und wiederholt die Schleife.
Es ist ein Blinklicht.

ist LED an

Mit diesem Block wird abgefragt, ob die LED an oder aus ist. Das Ergebnis ist wahr oder falsch.

ist Led … Helligkeit …
Mit dem Block **ist LED Helligkeit** lässt sich die Helligkeit einer LED abrufen und als Wert weiterverarbeiten.

Abfragen

Mit den Blöcken **ist LED ...** und **ist LED Helligkeit ...** kann man die Aktivität beziehungsweise die Helligkeit einer LED als Bedingung nutzen. Im Beispiel wir die Helligkeit der LED auf 512 gesetzt, sofern sie nicht schon diese Helligkeit hat.

Motoren (Unidirektionale-Motoren) - Eine Richtung

Das Symbol auf den Motorblöcken steht stellvertretend für alle Unidirektionalen-Motoren, also Motoren, die nur in einer Richtung laufen. Diese sind an einem O-Ausgang und Minus (GND) angeschlossen. Nicht jedoch Encoder- oder Servomotoren oder Motoren die an zwei O-Ausgängen (=M-Ausgang) angeschlossen sind. Diese werden mit dem Reiter „Motor“ angesteuert. Man kann auch Encodermotoren dafür benutzen, wenn man z.B. den Zähler nicht braucht.
Hinweis: Man -könnte- einen Motor mit zwei -einzelnen- O-Motorausgängen ansteuern. Das ist aber schlechte Programmierung und ist sehr unübersichtlich.

setze Motor Geschwindigkeit

Mit den Block **setze Motorgeschwindigkeit auf [] ...** kann man die Geschwindigkeit eines Motors auf einen bestimmten Wert (von 0 bis 512) setzen.

hole Motor Geschwindigkeit

Mit dem Block **hole Motorgeschwindigkeit** lässt sich die Geschwindigkeit eines Motors abrufen und als Wert weiterverarbeiten.

läuft Motor

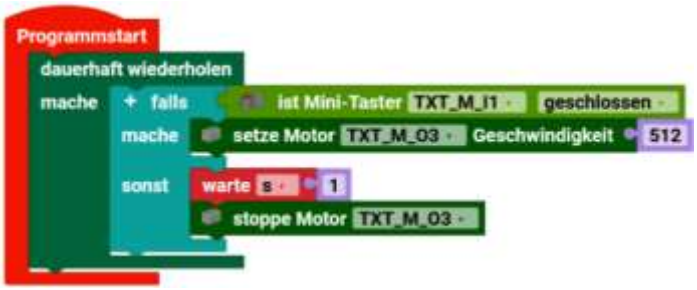
Mit den Blöcken **läuft Motor** und **ist Motorgeschwindigkeit ...** kann man die Aktivität beziehungsweise die Geschwindigkeit eines Motors als Bedingung nutzen.

Man kann das nutzen, um einen Motor nur abzuschalten, wenn ein anderer Motor auch steht. Man benutzt so etwas für zwei Bänder, um keinen Berg von Material am Anfang vom 2. anzuhäufen.

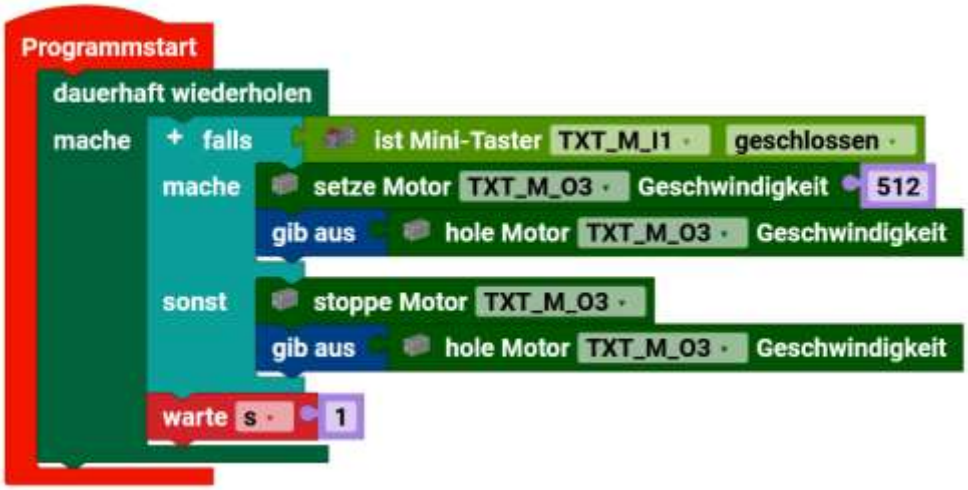
stoppe Motor

Mit dem Block **stoppe Motor …** ist es möglich einen Motor zu stoppen.

In diesem Beispiel läuft der Motor am O-3 so lange, wie der Taster an I1 geschlossen ist.

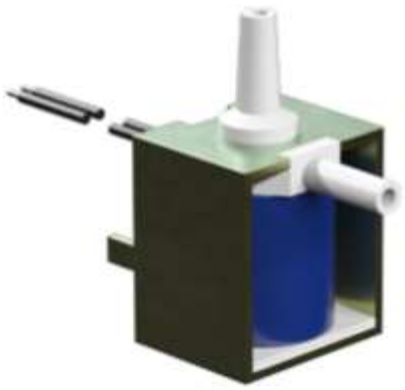


Hier wird der Motor eingeschaltet, wenn der Taster gedrückt wird. Wenn er losgelassen, läuft der Motor noch 1s nach, bevor er ausgeschaltet wird.



In diesem Beispiel wird jede Sekunde abgefragt, ob ein Taster offen oder geschlossen ist. Wenn er geschlossen ist, wird der Motor an O3 mit der Geschwindigkeit 512 angesteuert und dieser Zahlenwert auf der Konsole ausgegeben. Wenn der Taster offen ist, wird der Motor gestoppt und 0 ausgegeben.

Magnetventil



fischertechnik Magnetventil

Das Magnetventil wird an einen O-Ausgang und Minus (GND) angeschlossen. Spannungslos ist das Ventil geschlossen = keine Luft.

setze Magnetventil

Mit den Block **setze Magnetventil []** kann man das Magnetventil ein- oder ausschalten. Hier bedeutet "ein", dass das Ventil offen ist und "aus", dass das Ventil geschlossen ist.

Ansteuerung	Ventil	
aus =	geschlossen =	keine Luft
an =	offen =	Luft strömt durch

ist Magnetventil

Mit den Block **ist Magnetventil []** kann man die Aktivität eines Magnetventils als Bedingung nutzen.

Kompressor

Der Kompressor ist ein Motor mit einer Membran-Luftpumpe. Es ist egal wie rum der Motor läuft. Er wird nur an- oder ausgeschaltet. Man kann nicht durch weniger Geschwindigkeit weniger Druck erzeugen. Im Regelfall ist der Drucklufteil dicht. Man pumpt also in ein geschlossenes System immer mehr Luft rein.

Eventuell kann man es so hinbekommen, dass der Motor stehen bleibt. Das habe ich selbst aber so noch nie gebraucht. Normalerweise macht man das mit extra Regelventilen.

Um z.B. den Akku zu schonen, kann man den Kompressor nur einschalten, wenn man ihn gebraucht. Wenn alle Ventile geschlossen sind, kann man ihn abschalten.

setze Kompressor

Mit den Block **setze Kompressor []** kann man den Kompressor an- oder ausschalten.

ist Kompressor

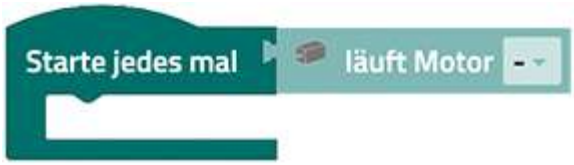
Mit den Block **ist Kompressor []** kann man die Aktivität eines Kompressors (an/aus) als Bedingung nutzen/abfragen.

Motor – Beide Richtungen

Diese Motoren können in beide Richtungen drehen (Links-Aus-Rechts), im Gegensatz zu den Motoren vom Reiter „Ausgang“ (An-Aus).

Der Starte jedes mal-Block

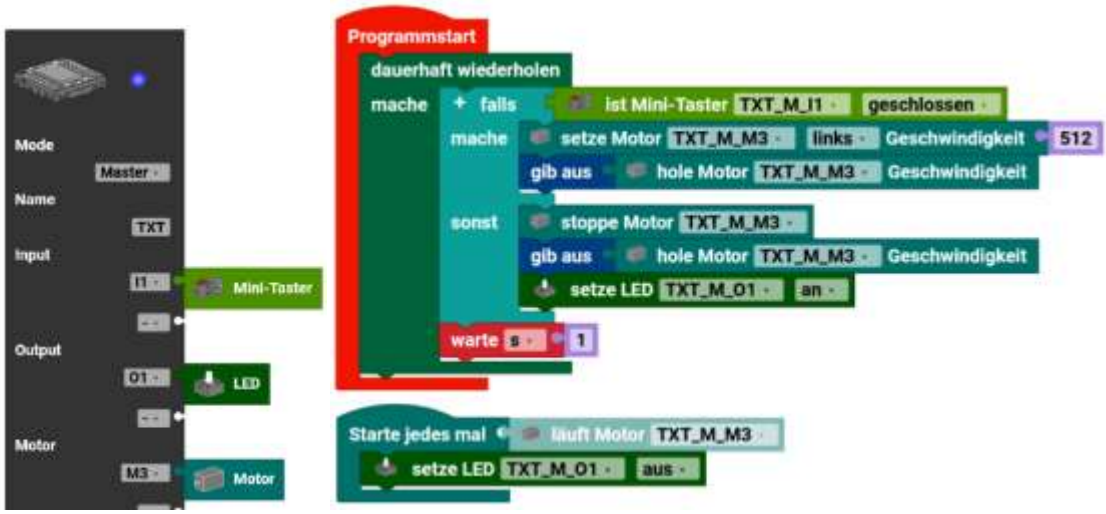
Der **Starte jedes mal-Block** bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der **Starte jedes mal-Block**:



Ist eine Abkürzung für folgendes Konstrukt:

Man kann in dem **Starte jedes mal-Block** der Kategorie Motor alle Bedingungen aus eben dieser Kategorie einsetzten. Achtung: Der „nicht“ Block aus dem Reiter Logik, funktioniert hier nicht!
Man kann also nicht abfragen, ob der Motor steht, um ein Parallelprogram zu starten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.



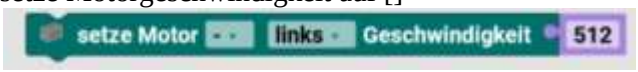
Bei diesem Beispiel wird eine LED (Bremslicht) ausgeschaltet, wenn der Motor läuft. Auch wird die Motorgeschwindigkeit auf der Konsole ausgegeben. Hier ist der Motor an M3 angeschlossen und die LED an O1 !

Motor

Das Symbol auf den Motorblöcken steht stellvertretend für alle Motoren, die nicht Encoder- oder Servomotoren sind. Man kann aber auch Encodermotoren damit betreiben, wenn man z.B. den Zähler nicht braucht.

Setzen

setze Motorgeschwindigkeit auf []



Mit dem Block **setze Motorgeschwindigkeit auf [] ...** kann man die Geschwindigkeit eines Motors auf einen bestimmten Wert (von 0 bis 512) setzen. Über das Dropdown-Menü (kleines Dreieck) kann die Drehrichtung gewählt werden.

Abrufen

hole Motor Geschwindigkeit

Mit dem Block **hole Motorgeschwindigkeit** lässt sich die Geschwindigkeit eines Motors abrufen und als Wert weiterverarbeiten.

Abfragen

ist Motorgeschwindigkeit

Mit dem Block **ist Motorgeschwindigkeit ...** kann man die Geschwindigkeit eines Motors als Bedingung nutzen.

läuft Motor

Mit den Blöcken **läuft Motor** kann man die Aktivität eines Motors als Bedingung nutzen.

Stoppen

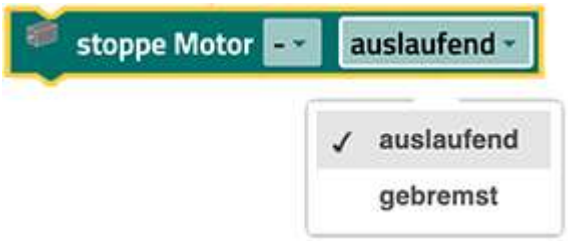
stoppe Motor []

Mit dem Block **stoppe Motor []** ist es möglich einen Motor zu stoppen.

Ältere Version des Blocks „stoppe Motor“:

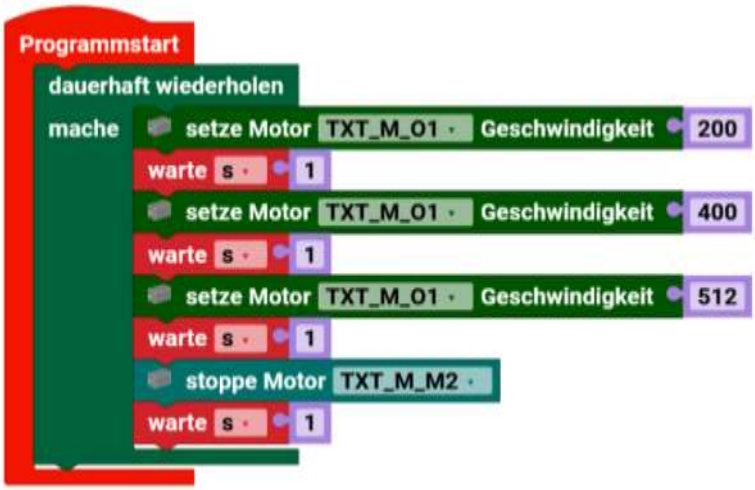
> Mit dem Block **stoppe Motor []** ist es möglich einen Motor zu stoppen. Dabei bietet der Block **stoppe Motor []** die Optionen, einen Motor direkt oder auslaufend zu stoppen.

Die gewünschte Option kann über das Dropdown-Menü (kleines Dreieck) ausgewählt werden:



Diese Option gibt es in den neueren Versionen von Robo Pro Coding nicht mehr!

Programm Beispiel Motor



In einer Endlosschleife, wird ein Motor wird mit verschiedenen Geschwindigkeiten angesteuert und dann kurz gestoppt.

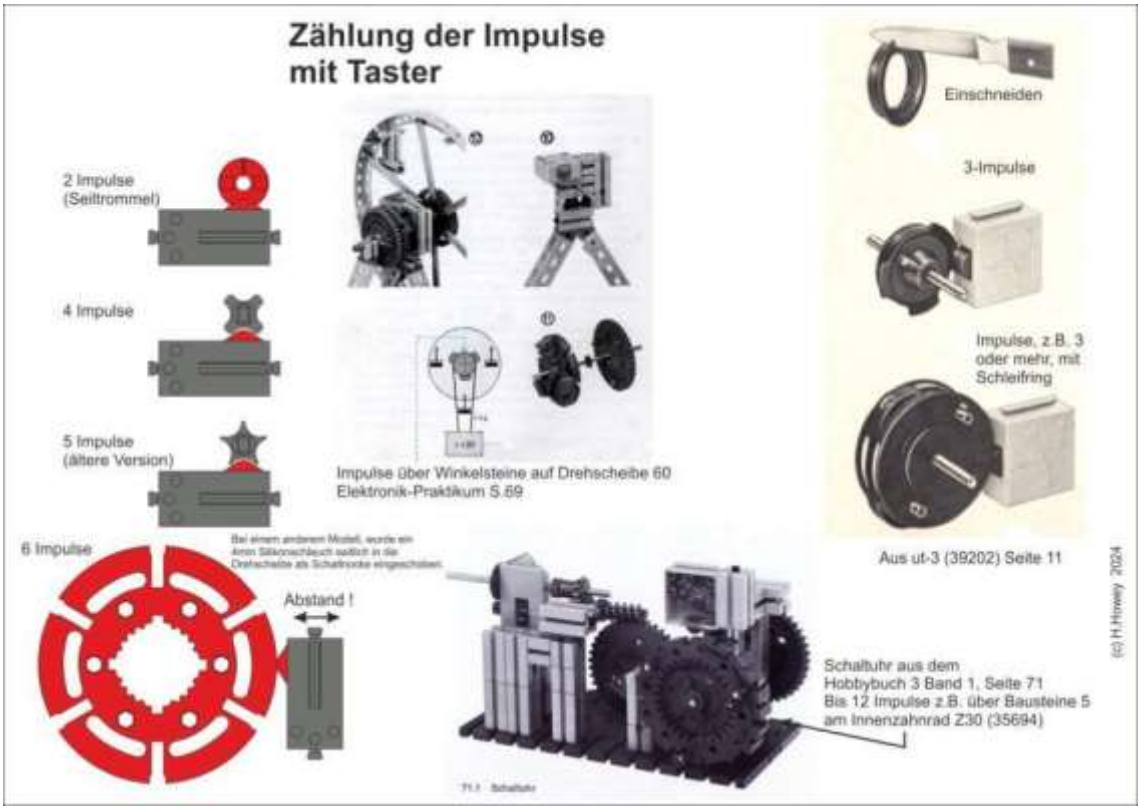
Encodermotor

Der Encodermotor hat die gleichen Funktionen wie ein normaler Motor, bietet aber zusätzlich die Möglichkeit, die Motor-Umdrehungen zu zählen und mehrere Motoren synchron anzusteuern.
Hinweis: Das synchrone Laufen von Motoren, funktioniert nur auf dem gleichen Controller. Nicht ein Motor auf dem TXT 4.0 Nr.1 und der andere Motor auf dem TXT 4.0 Nr.2 …
Diese Funktion ist rechenintensiv, deswegen kann sie nur auf dem jeweiligen TXT 4.0 gemacht werden. Es können aber 4 Encodermotoren auf dem Ersten, 4 Encodermotoren auf dem Zweiten… TXT 4.0 angeschlossen werden.
Eine Umdrehung der Welle, wird dabei in ~64 (63,9) Schritte unterteilt.

Impuls-Zahnrad 4 (37157)

Als Alternative zum Encodermotor, kann man auch einen Taster und z.B. ein Impuls-Zahnrad nehmen. Damit kann man z.B. an einem Fließband, die Entfernung zu Bearbeitung festlegen. Das kann man auch mit dem XM- oder XS-Motor (min-mot) machen.

Zählung der Impulse eines Tasters



Beispiele von älteren fischertechnik Modellen, wo Impulse gezählt werden.

Setzen

setze Motor … Geschwindigkeit/Schrittweite

Mit dem Block „setze Motor … Geschwindigkeit/Schrittweite“

kann man die Geschwindigkeit eines Motors auf einen bestimmten Wert (von 0-512) setzen. Über das Dropdown-Menü (kleines Dreieck) kann die Drehrichtung gewählt werden. Zusätzlich kann man die Anzahl an Schritten eingeben, die der Motor zurücklegen soll. In diesem Beispiel dreht sich der Motor 100

Schritte, also eine volle und ein Drittel Umdrehungen. Wie am Bespiel zu sehen hat dieser Block ein Pluszeichen, mit Hilfe dessen sich mehrere Motoren synchron ansteuern lassen. Es ist möglich Motoren am Master *oder* an einer Extension miteinander zu *synchronisieren*, eine übergreifende Synchronisierung bspw. zwischen Motoren des Master *und* einer Extension ist nicht möglich. Somit kann jeder Controller für sich 4 Encodermotoren synchronisieren.

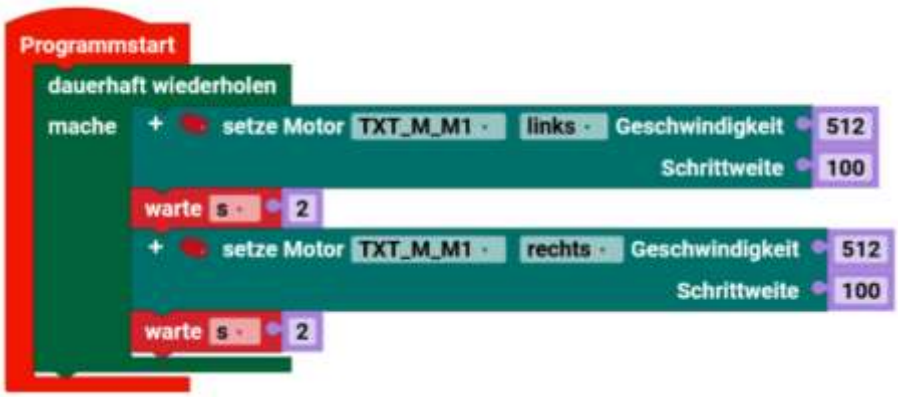
Counter-Zähler

Wo zählt der TXT 4.0 Controller?
Die Nummer vom Encodermotor und die Nummer vom Counter hängen zusammen.
Motor 1 (M1) ist am Counter 1 (C1)
Motor 2 (M2) ist am Counter 2 (C2)
Motor 3 (M3) ist am Counter 3 (C3)
Motor 4 (M4) ist am Counter 4 (C4)

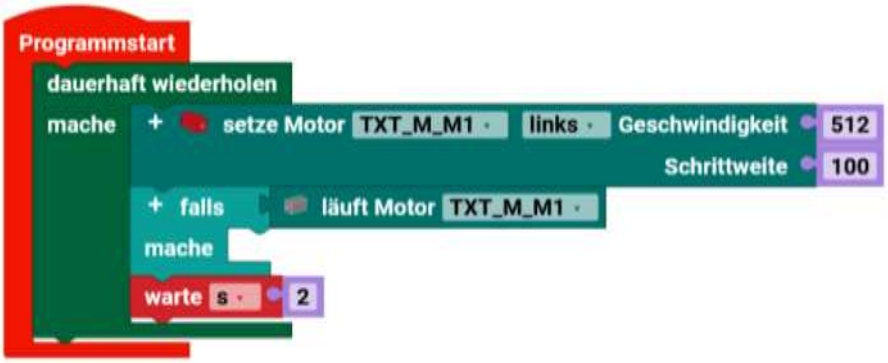
Es ist in Robo Pro Coding nicht möglich einen Encodermotor an einen O-Ausgang anzuschließen (nur eine Richtung), um den zweiten O-Ausgang für was anderes zu benutzen. Man kann nichts in der Controllerkonfiguration anschließen.

Hinweis: Schnell aufeinanderfolgende Synchronisierungsaufrufe, wie sie z.B. durch eine Schleife möglich sind (siehe Beispiel), können die Synchronität beeinträchtigen oder sogar komplett verhindern.

Schlechte Programmierung, da häufiger Aufruf. Der Controller und der Motor müssen ja auch Zeit haben, die Schritte zu machen.



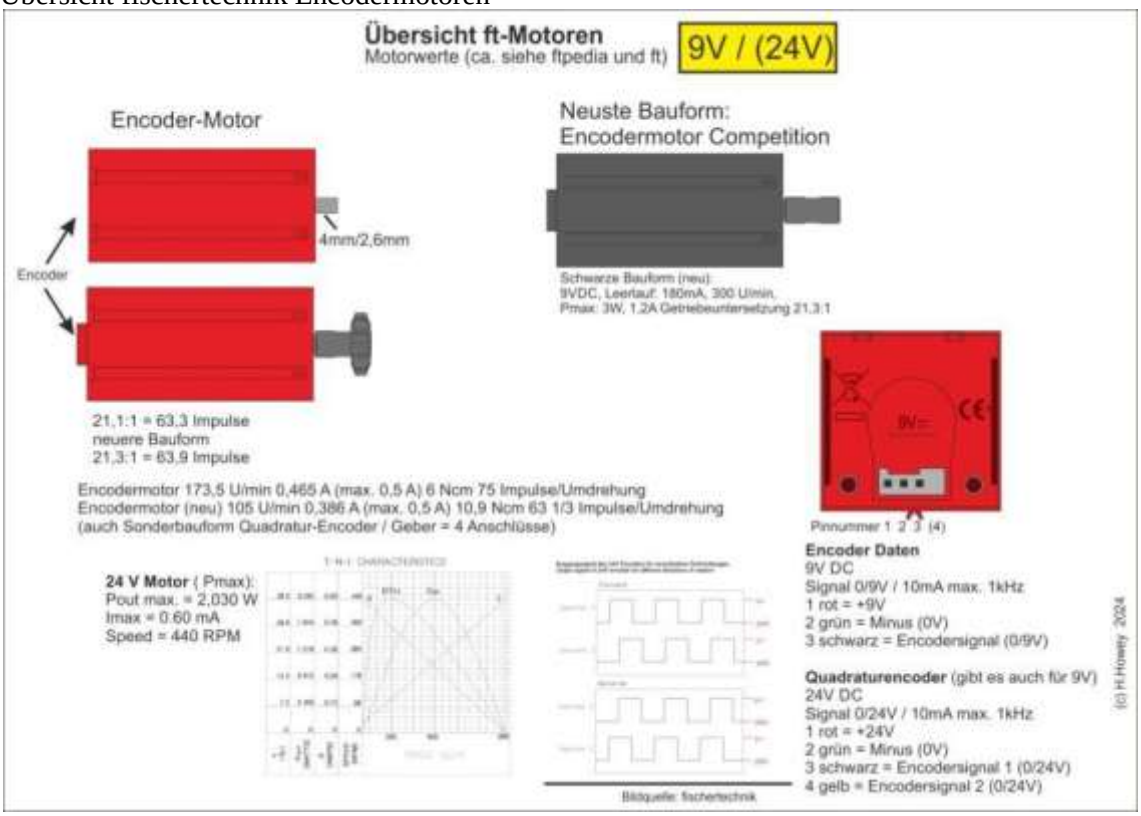
Beispiel mit Wartezeit nach dem „setze Motor“. Hier kann der Motor zu Ende drehen.



Eine andere Möglichkeit ist, abzufragen, ob der Motor noch läuft. Wenn man nur eine Wartezeit hätte, könnte der Motor ja schon fertig sein und das Programm wartet umsonst und bleibt einfach unnötig stehen. Hier wird also wirklich 2 Sekunden gewartet.

Im Laufe der Zeit gab es unterschiedliche Encodermotoren. Die unterscheiden sich im Getriebe und in der Anzahl der Impulse pro Umdrehung. Eine Sonderbauform war der Quadraturmotor mit zwei versetzten Impulsausgängen. Neben der höheren Auflösung konnte man auch die Drehrichtung messen. In „normalen“ fischertechnik Modellen wird das nie gebraucht, da das Programm ja weiss, in welcher Richtung der Motor angesteuert wird.

Übersicht fischertechnik Encodermotoren



Stoppen

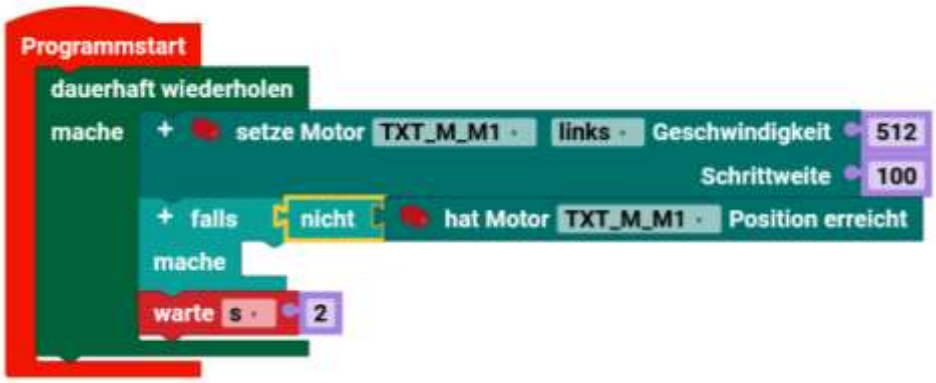
stoppe Motor

Mit dem Block **stoppe Motor …** stoppt man einen Motor. Möchte man mehrere Motoren gleichzeitig stoppen, kann man über das Plus, links am Block, bis zu drei weitere Motoren hinzufügen.

Abfragen

hat Position erreicht

Der Block **hat Position erreicht** wird genutzt, um das Erreichen der Position als Bedingung zu nutzen. Mit Position ist hier die Endposition eines Encodermotors nach vollendeter Schrittweite gemeint.



Hier wird in einer Endlosschleife der Motor M1 100 Schritte weit gefahren. Es wird gewartet bis der Motor die Position erreicht hat und dann 2 Sekunden gewartet, bis ein neuer Durchgang startet. Man beachte den „Nicht“-Block (aus Logik). Er negiert die Bedingung „hat Motor Position erreicht“ in „falls der Motor noch nicht erreicht, hat“ – tue nichts.

Hinweis: Man –sollte- „normale“ Motorbefehle und Encoderbefehle nicht mischen. Ist halt ein schlechter Programmierstyle. Die Encoderbefehle werden intern vom Controller gesteuert. Ein Befehl von außen, vom Programm oder PC, hat dann normalerweise keine weitere Wirkung.

Servomotor Modellbauservo

Setzen

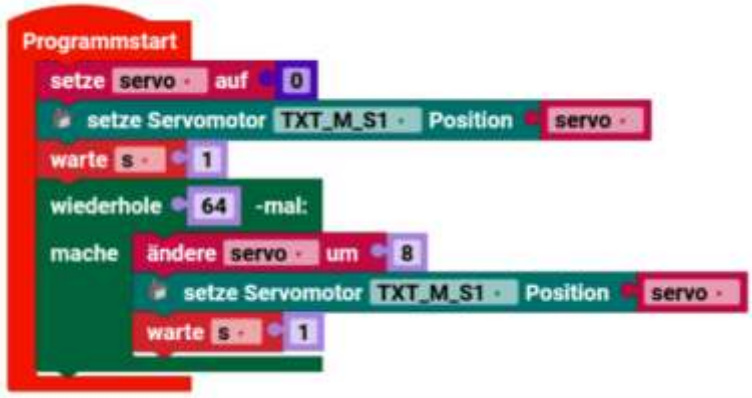
setze Servomotor Position

Mit dem Block **setze Servomotors Position auf ...** kann man die Position eines Servomotors auf einen bestimmten Wert (von 0-512) setzen. 0 und 512 sind die Werte für die maximale Auslenkung rechts und links (ca. +-90° = Gesamt 180°). Bei dem Wert 256 steht der Servomotor dementsprechend in der Mitte. Es sind also ca. 0.35° pro Schritt.

Abrufen

hole Servomotor Position

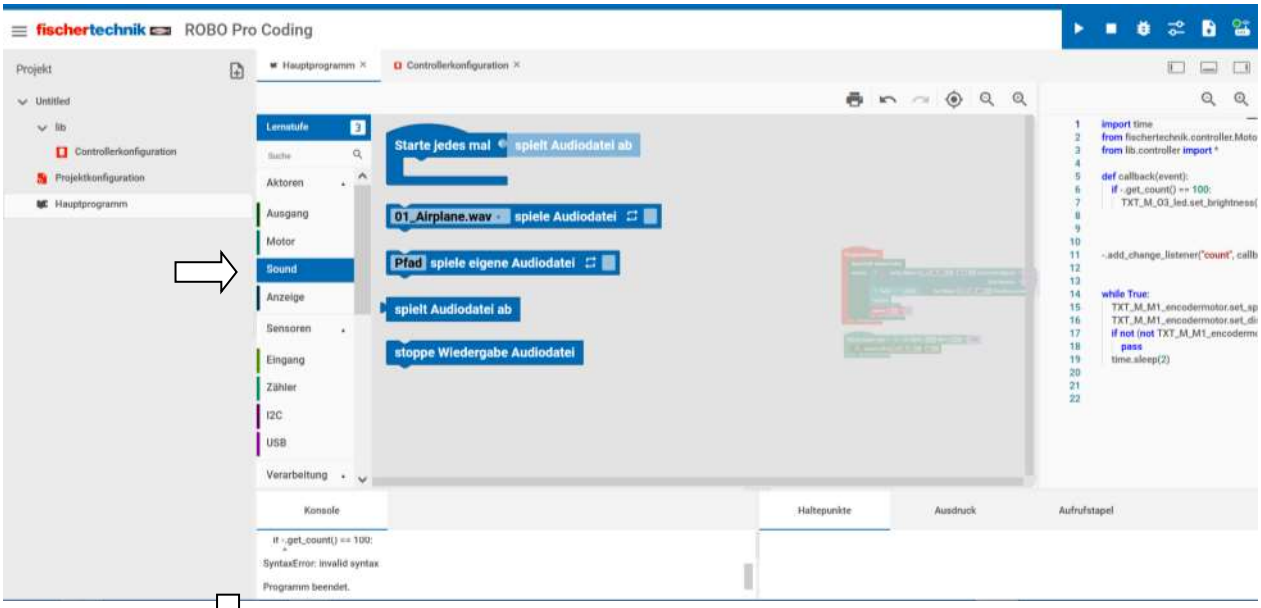
Mit dem Block **rufe Position ab** lässt sich die Position eines Servomotors abrufen und als Wert weiterverarbeiten.

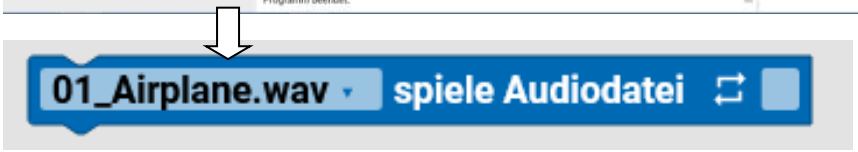


Das Programm setzt das Servo auf 0 und verändert 64-mal die Position vom Servo, mit einer Pausenzeit von einer Sekunde.

Sound

Der TXT 4.0 Controller hat einen eingebauten Lautsprecher und bietet somit die Möglichkeit Sounds abzuspielen.
Bei Sound können 29 Geräusche/Dateien abgespielt werden, die sich auf dem TXT 4.0 befinden.
Diese Dateien kann man sich auch als Test über das Menü auf dem Display des TXT 4.0 abspielen lassen.





Der Sound, der abgespielt werden soll, wird über das linke Feld ausgewählt.

Starte jedes mal-Block

Der **Starte jedes mal-Block** bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der **Starte jedes mal-Block i**st eine Abkürzung für folgendes Konstrukt:

Man kann in den **Starte jedes mal-Block** der Kategorie Sound alle Bedingungen aus eben dieser Kategorie einsetzten.

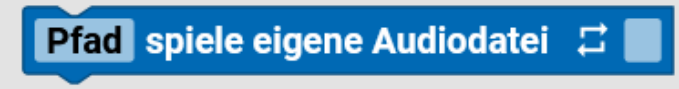
Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.

Abspielen

Vorinstallierte Audiodateien

Mit dem folgenden Block kann man einen von 29 vorinstallierten Sounds abspielen. Die gewünschte Audiodatei kann über das Dropdown Menü (kleines Dreieck) ausgewählt werden. Außerdem ist es möglich, den Ton in Dauerschleife abzuspielen. Dafür muss man das Kästchen hinter dem Dauerschleife-Symbol ankreuzen.

Eigene Audiodateien



Möchte man einen eigenen Sound abspielen, kann man den Block nutzen.
Um seinen eigenen Sound in den Block einzubetten, muss man:

1. Mit dem Controller verbunden sein
2. Die IP-Adresse des Controllers in den Browser eingeben (hierbei muss die IP gewählt werden, die auch zum Verbinden mit dem Controller genutzt wurde)
3. Auf der aufgerufenen Seite USER: ft, PASSWORD: fischertechnik eingeben
4. Ordner „sounds“ öffnen und dort über das Plus die gewünschte Audiodatei auf den Controller laden (wichtig: die Audiodatei muss im wav-Format vorliegen)
5. Im ROBO Pro Coding-Block unter Pfad "./dateiname.wav" angeben

Auch hier gibt es die Option, den Sound in Dauerschleife abzuspielen.

Abfragen

spielt Audiodatei ab

Um abzufragen, ob eine Audiodatei abgespielt wird, nutzt man den Block **„spielt Audiodatei ab“.** Dieser kann als Bedingung im Programm genutzt werden.

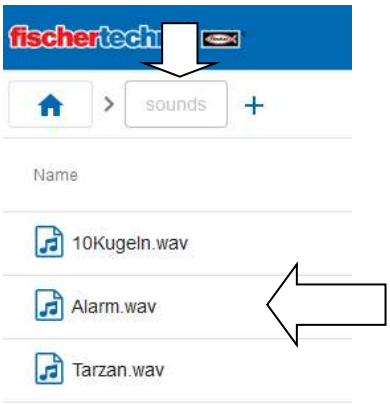
Stoppen

stoppe Wiedergabe Audiodatei

Um einen Ton zu stoppen, wird einfach der Block **stoppe Wiedergabe Audiodatei** im Programm verwendet.

Beispiel:

Wenn der Taster gedrückt ist, wird der Sound „Alarm.wav“ einmalig, also nicht in dauerschleife, abgespielt.



Die Datei „Alarm.wav“ befindet sich hier im Unterverzeichnis „sounds“. Über die Pfadangabe kann man die auch in ein anderes Verzeichnis setzen.

Anzeige (Display vom TXT 4.0)

Das Display des TXT 4.0 ist ein Touchscreen. Das Berühren von Elementen auf dem Display, kann man im eigenen Programm nutzen.
Mit den Blöcken aus dem Reiter Anzeige lässt sich der Bildschirm/Display des TXT 4.0 Controllers gestalten und nutzbar machen. Dies geschieht in zwei Schritten:

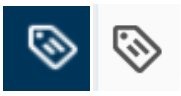
1. **Konfigurieren**, das heißt:
   - Eine „neue Datei“ der Kategorie *Anzeige* öffnen, über das Seiten Symbol mit dem Plus oben links (neben Projekt)
   - die gewünschten Elemente auf den gerasterten Bereich ziehen (er stellt den konfigurierbaren Teil des Displays dar)
   - bei Bedarf anpassen. Weitere Beschreibung unter „neue Datei/Anzeige“ / Anzeigenkonfiguration.
2. **Programmieren**, das heißt:
   - Im Hauptprogramm mit den Blöcken der Kategorie Anzeige die Wirkung vom Berühren mit dem Display programmieren.

Das „TXT“ im Namen des Feldes zeigt an, dass es auf dem Display des TXT 4.0 Controller ist. Weitere Informationen sind unter Anzeigenkonfiguration.

Achtung, kleine Stolperfalle! Die Reihenfolge der Blöcke im Hauptprogramm und die Reihenfolge der Menüpunkte in der Anzeigenkonfiguration, sind nicht gleich!

Reihenfolge Reiter Anzeige:

Beschriftungsfeld


TXTLabel - Textbeschriftung

Mit dem Element Beschriftungsfeld kann man einen Text auf dem Bildschirm platzieren. Das Symbol im Reiter Anzeige ist das Etikett.

setze Beschriftungsfeld Text

Mit dem Block **setze Beschriftungsfeld Text ...** lässt sich der Text auf dem Display, durch den Text „abc“ ersetzen oder auch durch den Inhalt einer Variablen, im laufenden Programm ändern.

hole Beschriftungsfeld Text

Mit diesem Block wird der Text vom Beschriftungsfeld geholt und kann im Programm weiterverarbeitet werden.

Eingaben

Texteingabe


TXTInput - Text-Eingabe

Das Element **Eingabe** erlaubt es dem Nutzer, dass über den TXT 4.0 Controller Text eingeben werden kann. Das zugehörige Symbol im Reiter Anzeige ist das "T" Zeichen.

setze Eingabefeld Text

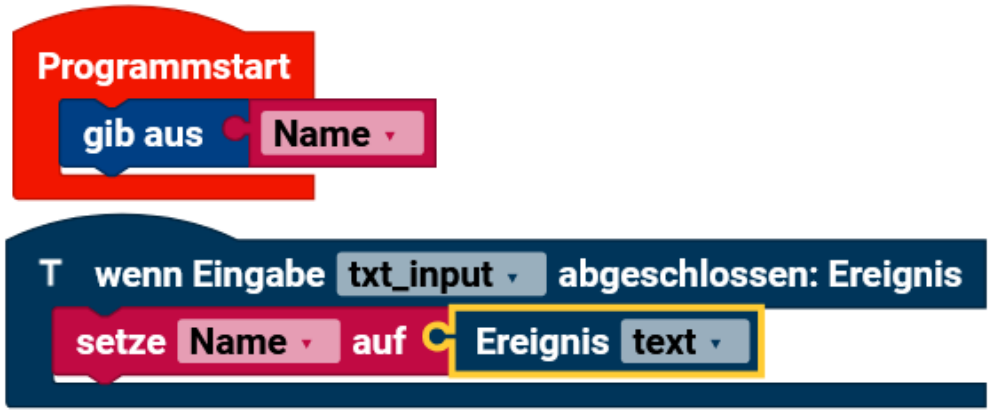
Mit dem Block **setze Eingabefeld Text ...** lässt sich der abgebildete Text im laufenden Programm ändern.

hole Eingabefeld Text

Über hole Eingabefeld Text, wird der Text in das Programm übernommen und kann weiterverarbeitet werden.

Eingabe-Programm

Das Eingabe-Programm läuft ab, wenn eine Eingabe abgeschlossen wurde. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Eingabe-Programm läuft im Block **wenn Eingabe abgeschlossen** ab. Der **Ereignis []**-Block wird im Eingabe-Programm auf "text" gesetzt. In diesem Beispiel wird die Variable „**Name“** auf den eingegebenen Text gesetzt, sie wird dann im Hauptprogramm genutzt, um den eingegebenen Text auf der Konsole auszugebenauszugeben:



Der Name des Feldes ist in der Anzeigenkonfiguration „txt_input“. Der Standarttext für den TXTInput ist „hello“. Der wird auch auf dem Display angezeigt, wenn man ihn nicht ändert.
Hinweis: Warum auch immer, läuft das Programm nicht mit der momentanen Onlineversion von Robo Pro Coding. Bitte die App-Version von Robo Pro Coding zum Runterladen benutzen.

Messinstrument


Gauge-Messgerät

Die Messinstrument-Funktion kann Werte (keine Werte kleiner 1) darstellen. Das zugehörige Symbol in der Anzeigenkonfiguration ist die Skalierung.

setze Messinstrument auf Wert

Hier wird die Anzeige des Messinstrumentes auf den vorgegebenen Wert gesetzt.

Dieser Wert sollte im vorher definierten Wertebereich liegen. Liegt der Wert außerhalb des Wertebereichs, wird, je nachdem ob der Wert zu groß oder zu klein ist, eine der Grenzen des Wertebereichs angezeigt. (min oder max)

hole Messinstrument Wert

Hier wird der Wert des Messinstrumentes geholt, den es gerade anzeigt und weitergegeben.

Statusanzeige


StatusIndicator

Der Statusindikator zeigt die Aktivität von etwas an. Je nach Status leuchtet er ("aktiv") oder tut dies nicht ("inaktiv"). Das Symbol in der Anzeigenkonfiguration ist eine leuchtende Diode.

setze Statusanzeige aktiv

Mit dem Block **setze Statusanzeige aktiv []** lässt sich die Statusanzeige aktivieren bzw. deaktivieren. Im Dropdown-Menü (kleines Dreieck) lässt sich wählen, ob die Statusanzeige auf aktiv oder inaktiv gesetzt werden soll.

ist die Statusanzeige aktiv

Mit diesem Block lässt sich abfragen, ob die Statusanzeige aktiviert oder deaktiviert ist.

Schieberegler

 TXTSlider-Schieberegler

Der Schieberegler gibt Werte abhängig von seiner Position zurück. Die Position kann dabei über den Touchscreen verändert werden. Der Wert kann über den **Ereignis []**-Block abgerufen werden, sobald der Schieberegler ruht. Der abgerufene Wert ist eine Dezimalzahl. Will man den Wert des Schiebereglers ganzzahlig haben, muss man den **runde**-Block  einsetzen. Das zugehörige Symbol für den Schieberegler ist der Strich mit dem Kreis.

setze Schieberegler Wert

Mit dem Block **setze Schieberegler Wert ...** kann man den Schieberegler auf einen anderen Wert setzen.

hole Schieberegler Wert

Mit hole Schieberegler Wert, liest man den aktuellen Wert des Schiebereglers.

setze Schieberegler aktiviert

Mit **setze Schieberegler aktiviert []** kann man den aktivierten Zustand des Schiebereglers auf wahr oder falsch setzen.

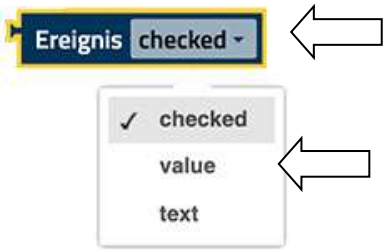
ist Schieberegler aktiviert

Hier wird der Zustand (wahr/falsch) vom Schieberegler zurückgegeben.

Schieberegler-Programm

Das Schieberegler-Programm läuft ab, nachdem der Schieberegler verschoben wurde. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Schieberegler-Programm läuft im Block, **wenn Schieberegler bewegt** ab. Der **Ereignis []**-Block wird im Schieberegler-Programm auf "value" gesetzt. In diesem Beispiel wird die Geschwindigkeit des Motors über den Schieberegler gesteuert. Der Wert des Schiebereglers muss gerundet werden, da der Motor nur ganze Zahlen als Drehzahl akzeptiert:

In der Anzeigenkonfiguration unten rechts, sollte man „Bis“ in 512 ändern. So kann man den vollen Geschwindigkeitsbereich vom Motor nutzen.



Achtung! **Erst** wenn man beim Block Ereignis, das „checked“ durch „value“ ändert, kann man den Block erst andocken.

Hinweis: Dieses Programm funktioniert nicht mit der momentanen Onlineversion von Robo Pro Coding. Bitte die App zum Runterladen nehmen.

Schaltfläche

 TXTButton

Die Schaltfläche ist ein beschriftetes Feld, das gedrückt werden kann. Drückt man die Schaltfläche, läuft das Schaltflächen-Programm ab, sobald sie wieder losgelassen wird. Das zugehörige Symbol für die Schaltfläche ist das Quadrat mit der "OK" Beschriftung

setze Schaltfläche aktiviert

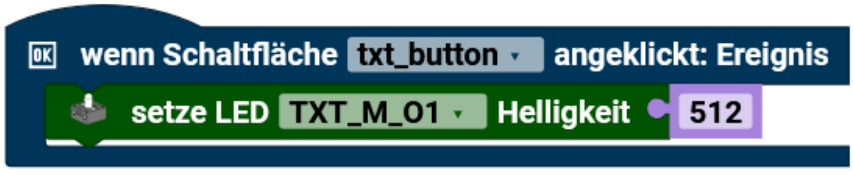
Mit dem Block **setze Schaltfläche aktiviert []** kann man den aktivierten Zustand auf wahr oder falsch setzen.

ist die Schaltfläche eingeschaltet

Hier die Abfrage des Status der Schaltfläche.

Schaltflächen-Programme

wenn Schaltfläche angeklickt
Das Schaltflächen-Programm läuft ab, sobald die Schaltfläche nicht mehr gedrückt ist. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Schaltflächen-Programm läuft im Block **wenn Schaltfläche angeklickt** ab. Der **Ereignis []**-Block kann im Schaltflächen-Programm nicht verwendet werden, da die Schaltfläche keinen Rückgabewert hat. In diesem Beispiel wird die LED aktiviert, wenn die Schaltfläche gedrückt wurde.



Schalter

 TXTSwitch - Schalter

Der Schalter kann zwei Positionen einnehmen und befindet sich immer in genau einer dieser beiden Positionen. Je nach Position gibt er **wahr** oder **falsch** zurück. Das zugehörige Symbol für den Schalter ist das Oval mit dem Punkt.

setze Schalter

Markierten oder aktivierten Zustand des Schalters auf wahr oder falsch festlegen.

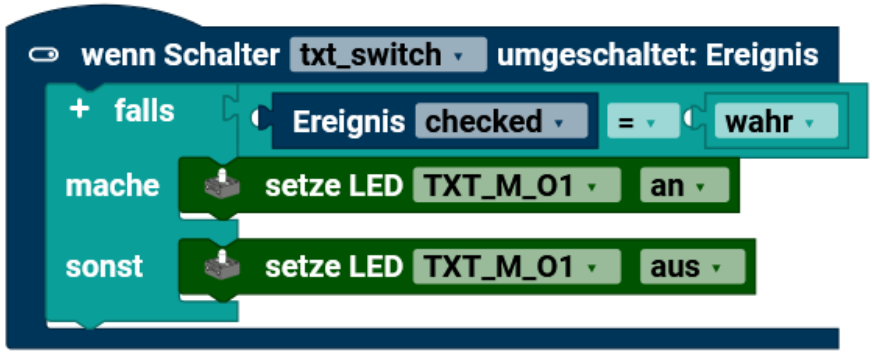
Der Block übernimmt zwei Funktionen. Man kann entweder die Aktivität (enabled im Dropdown-Menü wählen) oder den Zustand (checked im Dropdown-Menü wählen) auf **wahr** oder **falsch** setzen.

ist Schalter

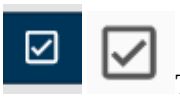
Holen des markierten (angeklickten) oder aktivierten Zustandes des Schalters.

wenn Schalter umgeschaltet

Das Schalter-Programm läuft jedes Mal ab, wenn der Schalter umgelegt wird. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Schalter-Programm läuft im Block **wenn Schalter umgeschaltet** ab. Der **Ereignis []**-Block wird im Schalter-Programm auf "checked" gesetzt, er gibt **wahr** oder **falsch** zurück. Dieses Beispielprogramm schaltet die LED ein, wenn der Schalter umgelegt ist, andernfalls wird die LED ausgeschaltet:



Kontrollkästchen



TXTCheckBox

Das Kontrollkästchen kann zwei Zustände annehmen und befindet sich immer in genau einem dieser

beiden. Je nach Zustand gibt es **wahr** oder **falsch** zurück. Das Symbol für das Kontrollkästchen ist das Quadrat mit dem Haken.

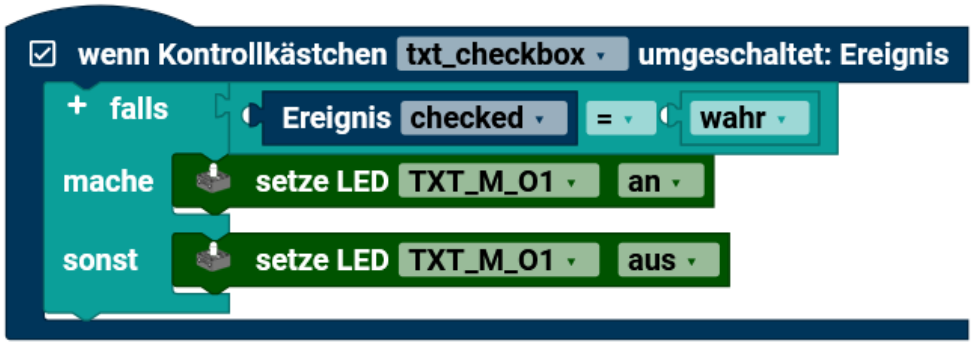
setze Kontrollkästchen

Der Block übernimmt zwei Funktionen. Über das Dropdown-Menü (kleines Dreieck) kann gewählt werden, welche man nutzt. Man kann entweder die Aktivität (enabled im Dropdown-Menü wählen) oder den Zustand (checked im Dropdown-Menü wählen) auf **wahr** oder **falsch** setzen.

ist Kontrollkästchen

Hier ist die Abfrage zum Status vom Kontrollkästchen.

wenn Kontrollkästchen umgeschaltet
Das Kontrollkästchen-Programm läuft jedes Mal ab, wenn das Kontrollkästchen gedrückt wird. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren über beide Programme hinweg. Das Kontrollkästchen-Programm läuft im Block **wenn Kontrollkästchen umgeschaltet** ab. Der **Ereignis []**-Block wird im Schalter-Programm auf "checked" gesetzt, er gibt **wahr** oder **falsch** zurück. Dieses Beispielprogramm schaltet die LED ein, wenn das Kontrollkästchen angehakt ist, andernfalls wird die LED ausgeschaltet.



setze Bild Base64-Bild

Für ein Bild im Display ein neues Base64-Bild festlegen. Base64 ist ein Datenformat, wo die Binärzahlen des Bildes, in ASCII-Code (und umgekehrt) umgewandelt werden. Links kann ein Textblock mit dem Pfad der Datei stehen.

Hier ist das Bild vom Baustein 163439 15x30 rund rot hochgeladen. Dieses Bild wird als erstes angezeigt. Mit dem „setze Bild…“- Block kann man ein neues Bild anzeigen lassen. Es muss ein Base64 Bild sein oder umgewandelt werden.

Aus Sorting_Line_AI:

```
with open(filename, "rb") as img_file:
    my_string = base64.b64encode(img_file.read())
imgb64 = "" + (my_string.decode('utf-8'))
time.sleep(0.2)
```

In der Variablen my_string ist das Bild im Base64 Format.

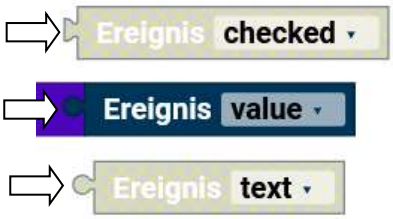
Man kann es auch etwas anders machen:

Beispiel aus Add_On_AI. Hier wird ein (ausgeschnittenes) Bild, das auf dem TXT 4.0 gespeichert ist, auf dem Display vom TXT 4.0 umgerechnet und dargestellt. Man beachte die Wartezeit!

Ereignis

Der Block **Ereignis []** ruft den Rückgabewert eines Elements ab. Dieser Block kann nur in den Ereignisprogrammen genutzt werden. In diesen Ereignisprogrammen bezieht sich der Block automatisch auf das Ereignis in dessen Programm er verwendet wird. Der geeignete Typ für den Rückgabewert, kann über das Dropdown-Menü (kleines Dreieck) gewählt werden:

Checked = Checkbox (Kontrollbox)
Value = Wert (Zahlenwert)
Text = Buchstaben, Text (Wert)



Erst wenn man zuvor beim Block Ereignis, das „checked“ (Logik) durch „value“ oder „text“ (Wert) ändert, kann man den Block andocken. **Es ändert sich vorne der Zapfen**!

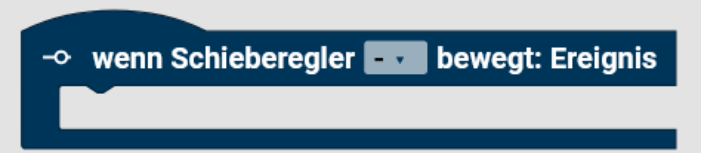
Ereignisprogramme

Diese Unterprogramme werden parallel zum Hauptprogramm ausgeführt. Werte von eingesetzten Variablen sind Global und werden vom Hauptprogramm mit dem Unterprogramm automatisch ausgetauscht.

wenn Schaltfläche angeklickt: Ereignis

Wenn die ausgewählte Schaltfläche angeklickt wird, wird dieses Unterprogramm ausgeführt.

wenn Schieberegler bewegt: Ereignis



Wenn der ausgewählte Schieberegler bewegt wird, wird dieses Unterprogramm ausgeführt.

wenn Schalter umgeschaltet: Ereignis

Wenn der ausgewählte Schalter umgeschaltet wird, wird dieses Unterprogramm ausgeführt.

wenn Kontrollkästchen umgeschaltet: Ereignis

Wenn das ausgewählte Kontrollkästchen umgeschaltet wird, wird dieses Unterprogramm ausgeführt.

wenn Eingabe abgeschossen: Ereignis

Wenn die Eingabe am TXT abgeschlossen ist, wird dieses Unterprogramm ausgeführt.

Sensoren

Sensoren reagieren auf Eingänge, Taster, Sensoren, Kamera…

Eingang

Starte jedes mal (Mini-Taster)

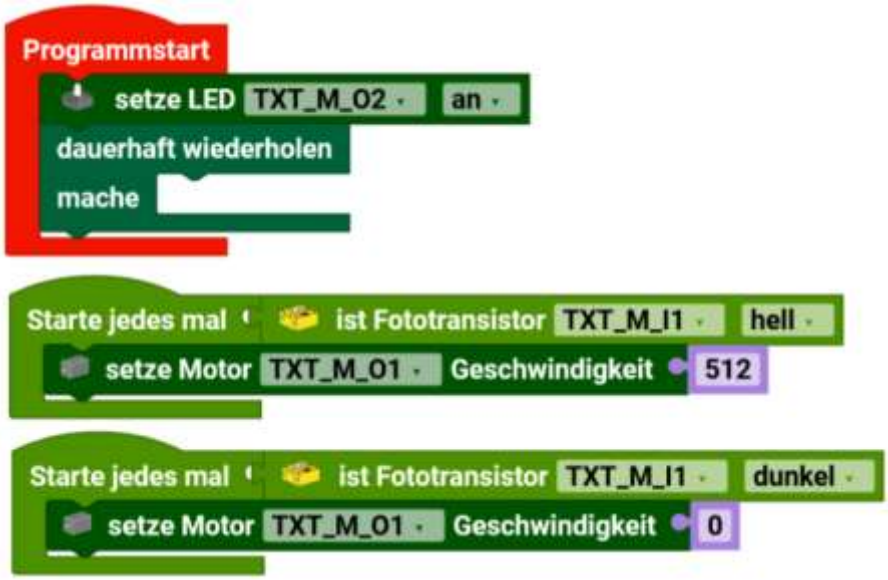
Der **Starte jedes mal-Block** bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der **Starte jedes mal-Block**:

Ist eine Abkürzung für folgendes Konstrukt:

Man kann in den **Starte jedes mal-Block** der Kategorie Eingaben alle Bedingungen aus eben dieser Kategorie einsetzten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurz gehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.

Programm mit „Starte jedes mal“ Block Unterprogrammen



Das Hauptprogramm schaltet die LED am O2 an, danach startet eine leere Endlosschleife. Die beiden

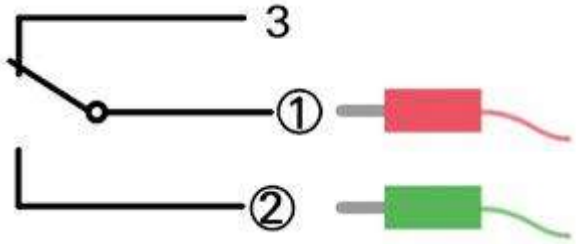
Unterprogramme fragen jeweils den Fototransistor ab, ob er hell oder dunkel ist. Wenn er „hell“ ist, startet das Unterprogramm und setzt die Geschwindigkeit vom Motor an O1 auf 512 = volle Spannung. Wenn er dunkel ist, wird die Geschwindigkeit auf 0 = Aus gesetzt.

Taster Mini-Taster

Der Taster ist ein digitaler Sensor, der "Es fließt Strom" von "Es fließt kein Strom" unterscheidet. Ob Strom fließt, hängt dabei sowohl von der Verkabelung, als auch davon ab, ob der Taster gedrückt ist. Man kann den Taster also auf zwei verschieden Arte verwenden:

Als „Schließer“:

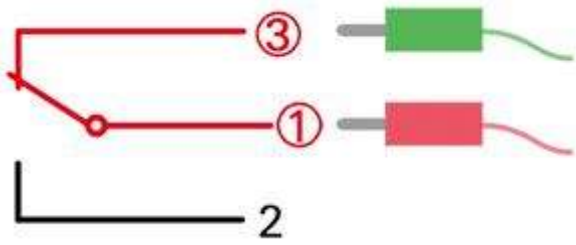
Kontakte 1 und 2 werden angeschlossen.



Taster gedrückt: Es fließt Strom.
Taster nicht gedrückt: Es fließt kein Strom

Als „Öffner“:

Kontakte 1 und 3 werden angeschlossen.



Taster gedrückt: Es fließt kein Strom.
Taster nicht gedrückt: Es fließt Strom.

Abrufen

hole Mini-Taster Status

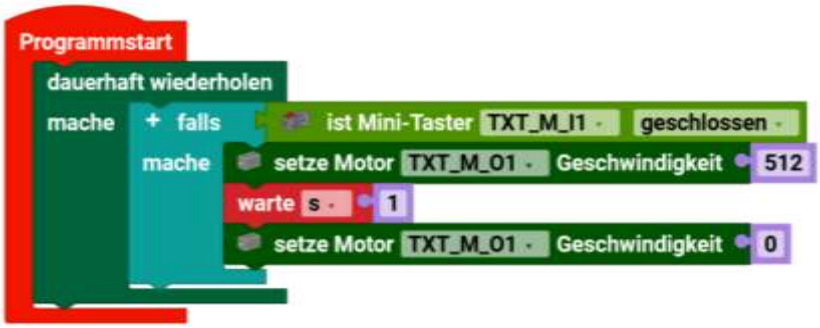
Mit hole Mini-Taster Status erhält man Information darüber, ob durch den Taster Strom fließt oder nicht. Fließt Strom, wird 1 zurückgegeben fließt kein Strom, 0.

Abfragen

ist Mini-Taster

Um abzufragen, ob der Taster in einem bestimmten Zustand ist, wird der Block **ist Taster…** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt werden nach welchem Zustand (geöffnet/geschlossen) gefragt wird. Dieser Block kann als Bedingung genutzt werden.

Beispiel Programm Taster



Dieses Programm startet eine Endlosschleife. In der wird abgefragt, ob der Taster geschlossen ist. Wenn ja, wird der Motor 1 Sekunde lang eingeschaltet und dann wieder ausgeschaltet. Danach wird der Taster erneut abgefragt. Ein Drücken des Tasters, während der Motor läuft, hat keine Wirkung.

Hinweis: Warum auch immer, läuft das Programm nicht immer richtig mit der momentanen Onlineversion von Robo Pro Coding. Bitte die App-Version von Robo Pro Coding zum Runterladen benutzen.

Starte jedes mal (Reedkontakt)

Der **Starte jedes mal-Block** bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der **Starte jedes mal-Block**:

Ist eine Abkürzung für folgendes Konstrukt:

Man kann in den **Starte jedes mal-Block** der Kategorie Eingaben alle Bedingungen aus eben dieser Kategorie einsetzten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.

Hinweis 2: Ich habe bisher keinen Unterschied zum ersten „Starte jedes mal“ Block gefunden. Auch der Pythoncode ist identisch.

Reedkontakt

Der Reedkontakt ist ein digitaler Sensor, der "Es fließt Strom" von "Es fließt kein Strom" unterscheidet. Von der Funktion her ist er wie ein Taster. Der Reedkontakt reagiert auf Magnete. Ist ein Magnet nahe dem Reedkontakt schaltet er. Er schließt seinen Kontakt.

Abrufen

hole Reedkontakt Status

Mit hole Reedkontakt Status erhält man Information darüber, ob durch den Reedkontakt Strom fließt oder nicht. Fließt Strom, wird 1 zurückgegeben fließt kein Strom, 0.

Abfragen

ist Reedkontakt

Um abzufragen, ob der Reedkontakt in einem bestimmten Zustand ist, wird der Block **ist Reedkontakt …** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt werden nach welchem Zustand (geöffnet/geschlossen) gefragt wird. Dieser Block kann als Bedingung genutzt werden.

Ultraschallsensor

Der Ultraschallsensor wird genutzt, um Entfernungen zu messen.

Abrufen

hole Ultraschallsensor Abstand

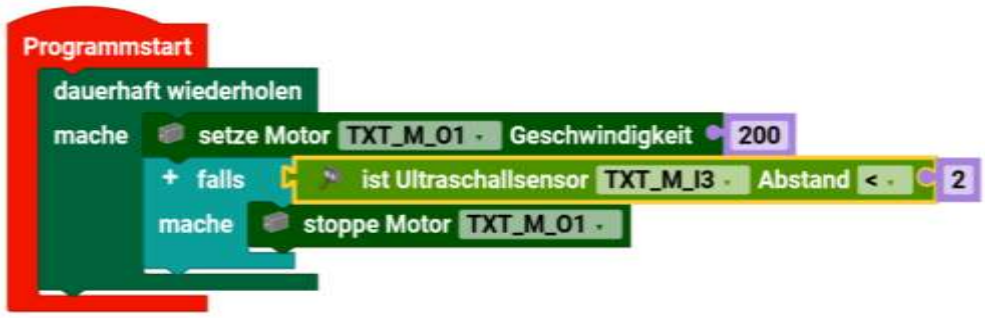
Mit **hole Ultraschallsensor Abstand** erhält man die Information, wie weit der Sensor vom nächsten Gegenstand entfernt ist. Der Abstand wird in cm zurückgegeben.

Abfragen

ist Ultraschallsensor Abstand…

Der Wert rechts am Ende, ist die Entfernung in cm.

Um abzufragen, ob der Sensor einen bestimmten Abstand zum nächsten Gegenstand hat, wird der Block **ist Ultraschallsensor Abstand…** genutzt.
Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt werden, wie der gemessene Abstand mit einem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >) . Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der gemessen Abstand kleiner als 2 cm ist.



Farbsensor

Der Farbsensor sendet rotes Licht aus und misst, wie viel davon zurückreflektiert wird. Je nachdem, wie stark die Reflexion ist, gibt der Farbsensor Werte von 0 bis 2000 zurück. Er eignet sich gut, um vorher kalibrierte Farben zu erkennen.

Abrufen

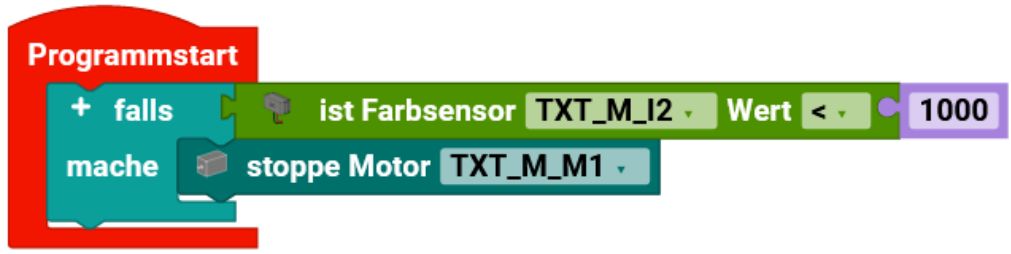
hole Farbsensor Wert

Mit **hole Farbsensor Wert** erhält man die Information, wie stark eine Oberfläche das Licht reflektiert.

Abfragen

ist Farbsensor Wert …

Um abzufragen, ob der Sensor eine bestimmte Farbe vor sich hat, wird der Block **ist Farbsensor Wert …** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt werden, wie der gemessene Farbwert mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >) . Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der gemessen Farbwert kleiner als 1000 ist.



IR-Spursensor

Der Infrarot-Spursensor ist ein digitaler Sensor zur Erkennung einer schwarzen Spur auf weißem Untergrund, der bei einem Abstand von 5-30 mm von Sensor zu Untergrund arbeitet.
Der richtige Abstand ist sehr wichtig! Auch Fremdlicht sollte so wenig wie möglich um den Sensor herum sein.

Abrufen

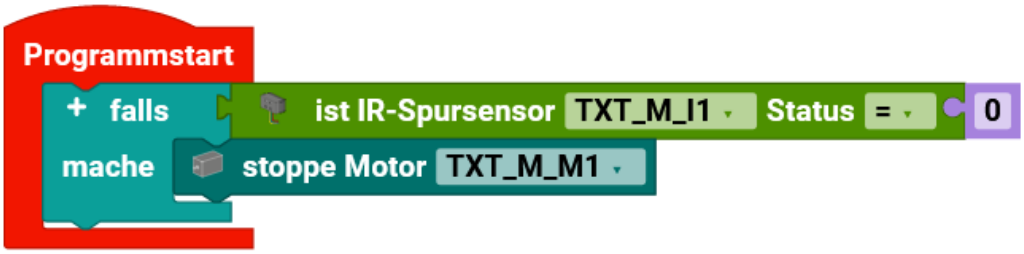
IR-Spursensor Status

Mit **hole IR-Spursensor Status** erhält man 0, wenn der Sensor keine Spur erkennt. Erkennt der Sensor eine Spur, wird 1 zurückgegeben

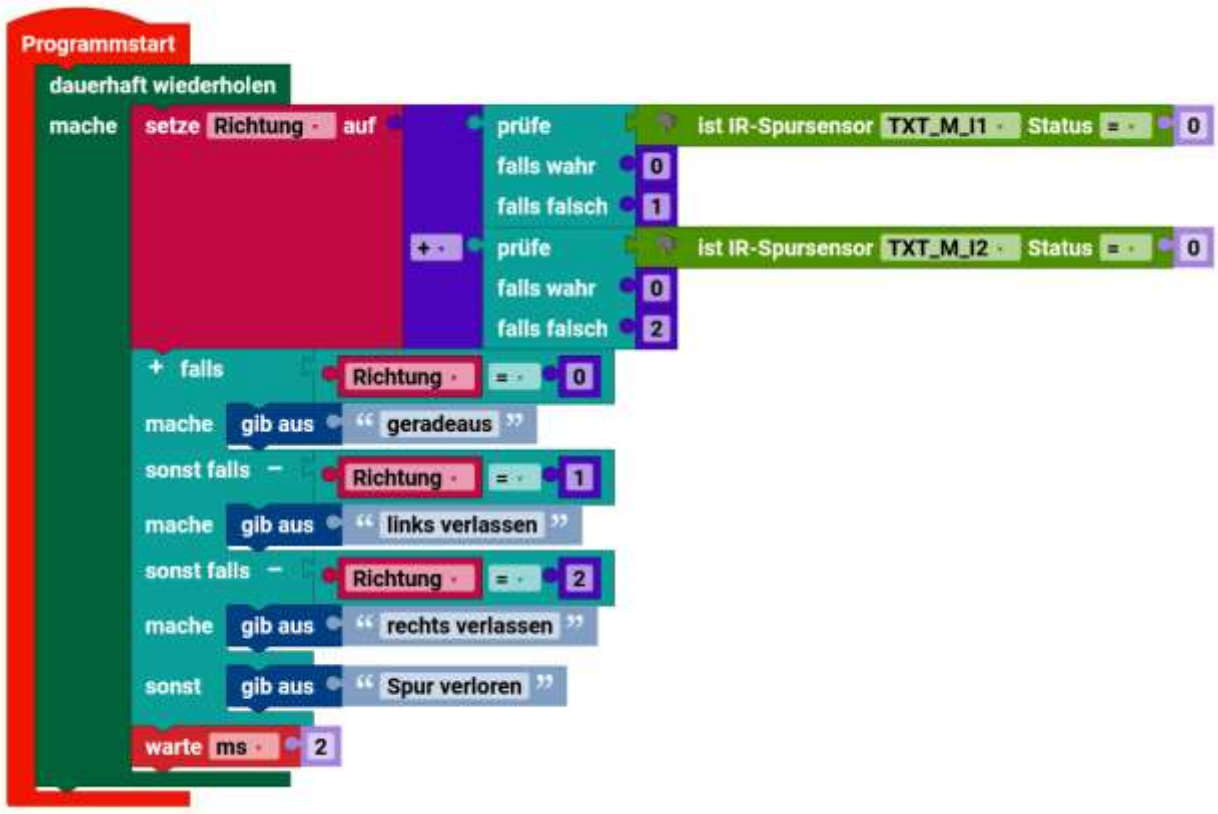
Abfragen

IR-Spursensor Status [] ... .

Um abzufragen, ob der IR-Spursensor eine Spur erkennt, vergleicht man den aktuellen Spurstatus mit 0 oder 1. Hierzu eignet sich der Block **ist IR-Spursensor Status [] ...** . Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt werden, wie der Spurstatus mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der Spurstatus 0 ist.



Beispiel



Das Programm fragt in einer Dauerschleife die Spurensensoren ab. Wenn der erste Sensor 0 ist, dann ist er auf der schwarzen Linie. Wenn er 1 ist, wird er beleuchtet. Genau so wird der zweite Sensor abgefragt, nur das 0 und 2 zurückgegeben werden. Die Werte vom Sensor 1 und 2 werden addiert und in der Variablen „Richtung“ gespeichert.

Die Variable „Richtung“ wird nun ausgewertet.
Wenn sie 0 ist, sind beide Sensoren nicht beleuchtet und es wird der Text „geradeaus“ auf der Konsole ausgegeben.
Wenn sie 1 ist, ist der Sensor 1 beleuchtet und der Sensor 2 nicht beleuchtet. Es wird der Text „links verlassen“ ausgegeben.
Wenn sie 2 ist, ist der Sensor 1 unbeleuchtet und der Sensor 2 beleuchtet. Es wird der Text „rechts verlassen“ ausgegeben.
Als letzte Möglichkeit muss die Variable „Richtung“ gleich 3 sein. Somit sind beide Sensoren beleuchtet und es wird der Text „Spur verloren“ auf der Konsole ausgegeben.

Anschließend wird 2ms gewartet und das Programm fängt wieder von vorne an.

Fototransistor

Der Fototransistor ist ein digitaler Sensor, der hell von dunkel unterscheidet.

Abrufen

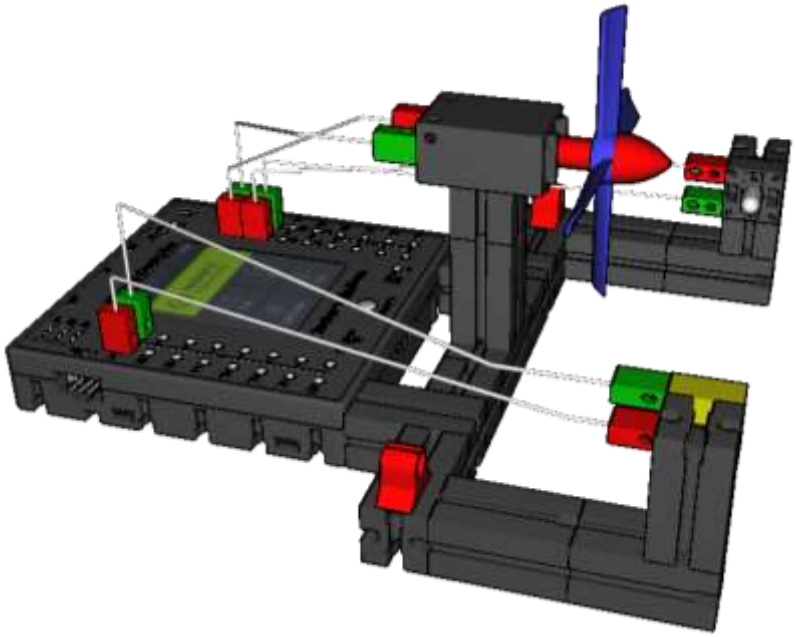
hole Fototransistor Status

Mit **hole Fototransistor Status** erhält man 0, wenn der Sensor kein Licht erkennt. Erkennt der Sensor ausreichend Licht, wird 1 zurückgegeben.

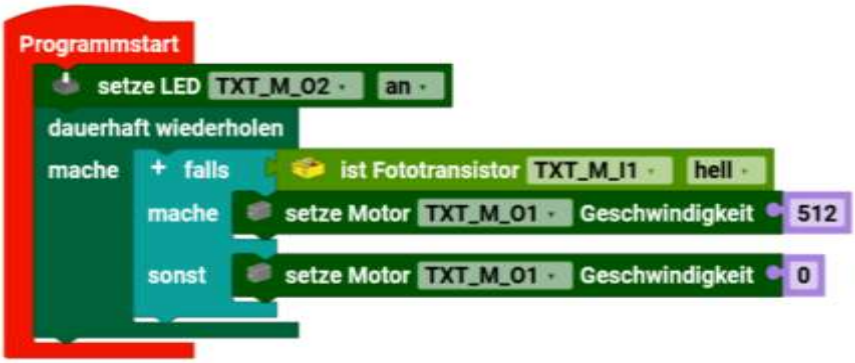
Abfragen

ist Fototransistor Status []

Um abzufragen, ob der Fototransistor hell oder dunkel erkennt, vergleicht man den Helligkeitsstatus mit 0 oder 1. Hierzu eignet sich der Block **ist Fototransistor Status []** . Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt werden, ob hell oder dunkel abgefragt werden soll.



Eine mögliche Verwendung für einen Fototransistor ist in einer Lichtschranke, wie diesem Modell.



Lichtschranke - Sicherheitslichtschranke.

Die LED an O2 wird eingeschaltet. Dann wird in einer Endlosschleife abgefragt, ob der Fototransistor beleuchtet ist oder nicht. Wenn er hell ist, wird der Motor eingeschaltet. Die Lichtschranke ist somit nicht unterbrochen. Sobald sie unterbrochen wird, schaltet der Motor ab.

Man kann aus der Sicherheitslichtschranke einen Händetrockner machen, in dem man den Motor erst laufen lässt, wenn die Lichtschranke dunkel ist. Das kann man auch erreichen, in dem man die Geschwindigkeiten genau andersherum einträgt. Probiere es aus.

Fotowiderstand

Der Widerstand des Fotowiderstands sinkt, wenn er mehr Helligkeit ausgesetzt ist. Der ausgegebene Wert des Fotowiderstands ist also ein Maß für Helligkeit.

Abrufen

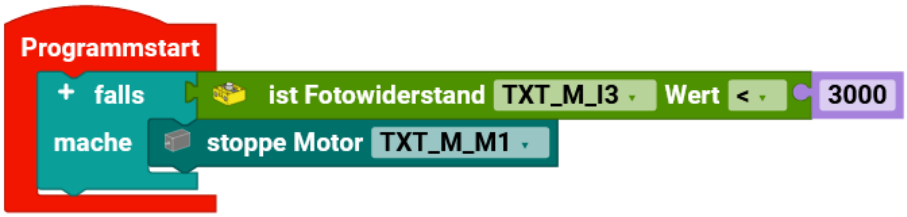
hole Fotowiderstand Wert

Mit **hole Fotowiderstand Wert** erhält man die Information, wie hell es ist. Je kleiner der ausgegebene Wert, desto heller ist es.

Abfragen

ist Fotowiderstand Wert []

Um abzufragen, ob der Fotowiderstand einen bestimmten Helligkeitswert misst, wird der Block **ist Fotowiderstand Wert [] ...** genutzt Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt, wie der Helligkeitswert mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der Helligkeitswert kleiner als 3000 ist.



NTC-Widerstand (Heißleiter)

Der NTC-Widerstand ist ein nichtbinärer Temperatursensor. Sein elektrischer Widerstand sinkt, wenn die Temperatur steigt, und ist damit ein Maß für die Temperatur.

Abrufen

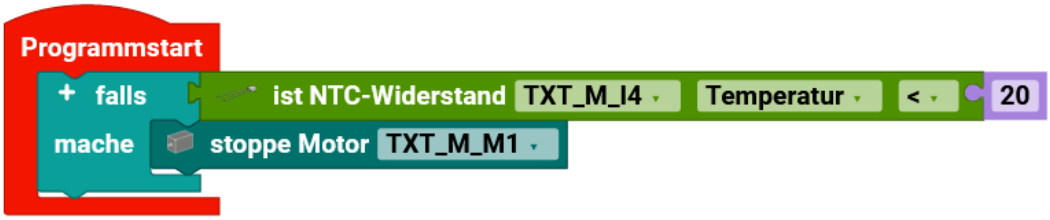
hole NTC-Widerstand []

Mit **hole NTC-Widerstand []** erhält man entweder einen Widerstandswert oder die daraus errechnete Temperatur. Was zurückgegeben werden soll kann über das Dropdown-Menü (kleines Dreieck) gewählt werden.

Abfragen

Block ist NTC-Widerstand [] []

Um abzufragen, ob der NTC-Widerstand einen bestimmten Wert misst, wird der Block **ist NTC-Widerstand [] [] ...** Über die Dropdown-Menüs (kleines Dreieck) kann gewählt werden, was und mit welchem Vergleichsoperator verglichen werden soll. Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Temperatur kleiner als 20 ist.



Zähler


Symbol vom Zähler [+1]

Der Encodermotor kann als Zähler eingesetzt werden. Gezählt wird die Anzahl seiner Umdrehungen, und zwar nicht nur, wenn er sich selber als Motor dreht, sondern auch, wenn er von außen mechanisch angetrieben wird.

Hinweis: Man muss in der Controllerkonfiguration einen Zähler an den Counter/Digitaler Input anschließen. Nur den Encodermotor bei „Motor“ auswählen reicht nicht. Ansonsten ist der Encodermotor ein ganz normaler Motor ohne Zähler.

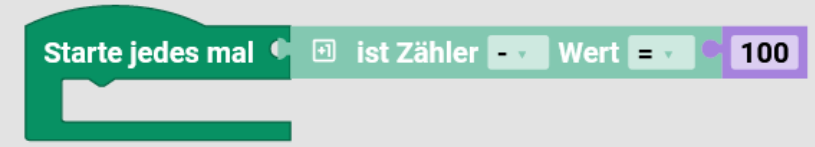

Impulsrad

Alternative Möglichkeiten:

Wie beim Zähler/Counter beim Encodermotor, kann man auch hier einen fischertechnik Taster+Impulsrad und einen XM- oder XS-Motor nehmen. Z.B. um die Entfernung eines Roboters durch Umdrehungen oder Entfernung auf einem Band, zu setzen.

Starte jedes mal-Block

Der Starte jedes mal-Block bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der Starte jedes mal-Block:



Ist eine Abkürzung für folgendes Konstrukt:

Man kann in den Starte jedes mal-Block der Kategorie Zähler alle Bedingungen aus eben dieser Kategorie einsetzten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.

Abrufen

hole Zähler Wert

Mit hole Zähler Wert erhält man den vom Zähler gezählten Wert.

Abfragen

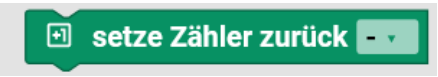
ist Zähler Wert

Um abzufragen, ob der Zähler einen bestimmten Wert gezählt hat, wird dieser Block

genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt, wie der gezählte Wert mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >).

Zurücksetzen

setze Zähler zurück

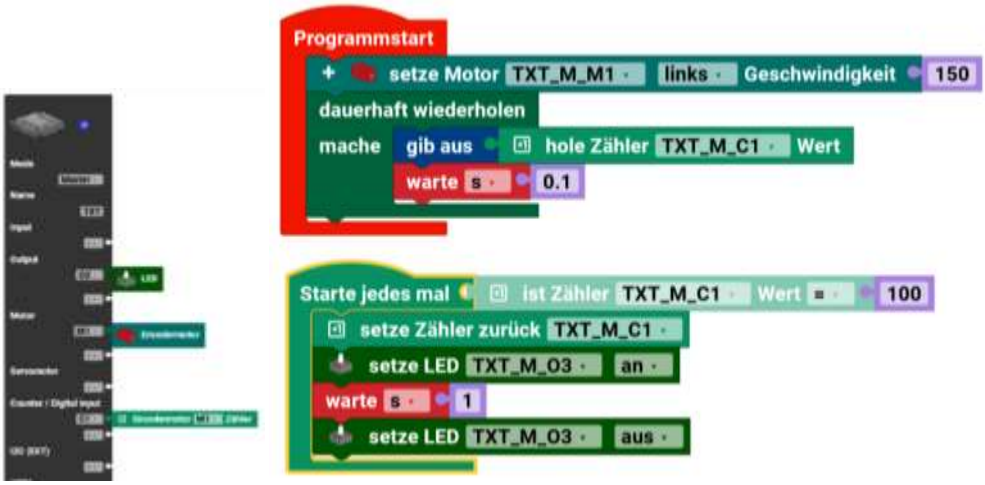


Der Zähler beginnt wieder bei 0, wenn der Block setze Zähler zurück ausgeführt wird.

Programm Zähler

Dieses Unterprogramm startet, wenn der Zähler = 100 ist. Das Programm läuft nur ein Mal, wenn man nicht den Zähler zurücksetzt. Das rote Hauptprogramm muss auch vorhanden sein.

Programm Zähler mit Blinken



Hier sind eine LED am Ausgang O3, ein Encodermotor an M1 und ein Zähler an C1 angeschlossen. Das Hauptprogramm setzt den Encodermotor M1 auf die Drehrichtung links und die Geschwindigkeit auf 150. In einer Endlosschleife wird der Zählerwert C1 geholt und auf der Konsole am Bildschirm ausgegeben. Um nicht zu viele Werte anzuzeigen wird 0,1 Sekunden gewartet. Zwischenzeitlich zählt der Zähler weiter.
Im Unterprogramm wird jedes Mal der Zähler, wenn er 100 erreicht, hat auf 0 gesetzt. Dann wird die LED eingeschaltet, 1 Sekunde gewartet und die LED wieder ausgeschaltet.

I2C

Die in diesem Kapitel beschriebenen I2C-Sensoren werden über ein geeignetes Flachbandkabel mit dem TXT 4.0 und RX-Controller verbunden. Es ist egal ob man den Sensor links oder rechts am Controller einsteckt.

Es gibt die Sensoren in zwei Bauarten: Einmal mit 10-poligem Buchse/Stecker (anfangs für den TXT) und die neueren mit 6-poligem Buchse/Stecker. Es gibt Adapter oder man baut sich ein eigenes Adapterkabel, um die älteren Sensoren an den TXT 4.0 oder den RX Controller anzuschießen. Beide Controller haben 6-polige Buchsen. Für Versuche „kann" man auch es mit Verbindungskabeln machen. Dabei sollte man aber genauestens auf die Steckerbelegung achten. Die in der Hilfe/Dokumentation von Robo Pro Coding abgebildeten Sensoren (und auch die meisten hier dargestellten) sind die, mit 10-poligem Buchsen.

Achtung! Die Sensoren laufen auf 3,3V, haben aber teilweise eine zusätzliche 9V Spannungsversorgung (Plus 9V) die intern auf 3,3V runter geregelt wird.
So den Sensor oder den TXT 4.0 /RX *niemals* an einem 5V I2C-Bus betreiben!

Man „kann" auch per Python-Block auf I2C und somit auf die fischertechnik Sensoren zugreifen. Siehe dazu auch die anderen Kapitel über I2C. Mit den „normalen" I2C-Blöcken, von Robo Pro Coding, ist nur möglich auf die fischertechnik Sensoren zuzugreifen. Nicht auf andere „fremde" Sensoren. Dazu siehe auch die Beispiele zu I2C in Python hinten im Buch. Da ist es dann möglich, weil man z.B. die Adressen frei eingeben kann.

Starte jedes mal-Block

Der **Starte jedes mal-Block** bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms.

Der **Starte jedes mal-Block** ist eine Abkürzung für folgendes Konstrukt:

Man kann in den **Starte jedes mal-Block** der Kategorie I2C alle Bedingungen aus eben dieser Kategorie einsetzten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.

Gestensensor

Der RGB Gestensensor hat folgende Messgrößen: Farbe (RGB), Umgebungshelligkeit, Abstände bis 15cm, Gestenerkennung in 6 Richtungen.

Der Sensor ist ein APDS9960. Für weitere Programmierung, z.B. in Python, findet sehr viel über diesen Sensor im Internet. Auch zur Vorgänger-Programmiersprache Robo Pro gibt es Beispiele z.B. im Kasten „Robotics Smarttech“, die man in Robo Pro Coding übertragen kann.

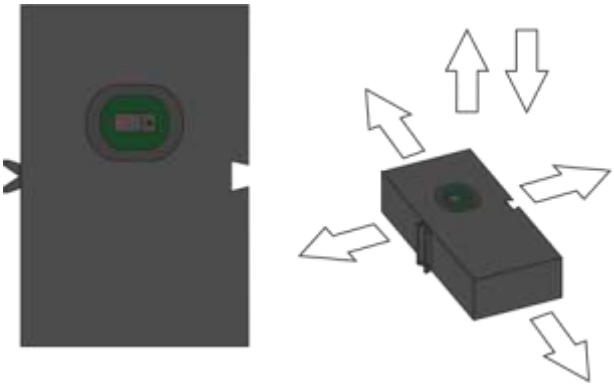
Gesten sind Bewegungen nach Oben, Unten, Links und Rechts. Zusätzlich kommen noch von unten nach oben, Entfernen und von oben nach unten, Annäherung.

10-polig

Alte Bauform

6-polig

Neue Bauform



Gestensensor von der Frontseite und Bewegungsrichtungen die Erkannt werden. Wichtig ist die Einbaurichtung des Sensors, denn sonst stimmen die Werte nicht.

Der Sensor hat eine Art von Kamera/Sensor, mit dem gemessen wird.

Gestensensor aktiviere Licht

Dieser Block dient zu Starten (Initialisieren) der einzelnen Sensoren des Gestensensors.

Folgende Sachen kann man am Gestensensor in Robo Pro Coding aktivieren oder deaktivieren:

Licht, Entfernung und Geste.

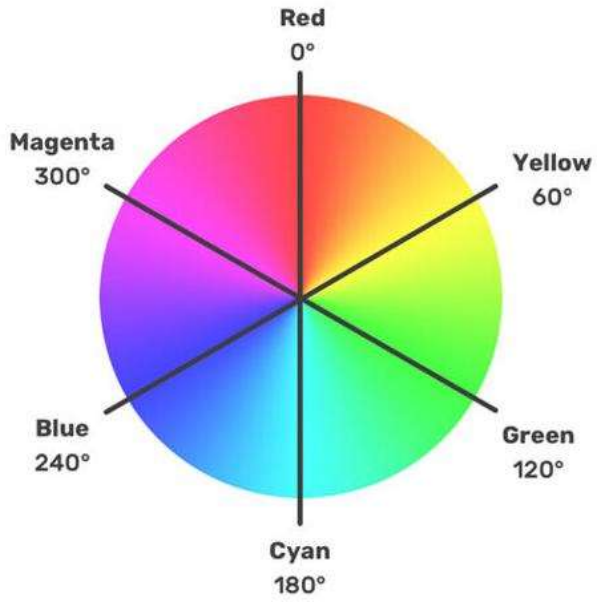
hole Gestensensor

Folgende Daten kann man vom Gestensensor holen:
HEX = Hexadezimalwert der Messung
RGB = Rot Grün Blau –Wert
RGB red = Rotanteil (0-255)
RGB green = Grünanteil (0-255)
RGB blue = Blauanteil (0-255)
HSV = HSV Farbwert Farbwinkel
HSV hue (°) = Farbton
HSV Saturation (%) = Farbsättigung
HSV value (%) HSV Farb-Wert in % (0%=dunkel, 100%= volle Helligkeit)
Umgebungslicht
Entfernung = 0(weit) bis 255(nah)
Geste

HSV-Farbraum
Achtung! Diese Beschreibung ist –nur- für den Gestensensor. Die Kamera hat zwei (!) andere Farbräume. Sie funktionieren aber mit anderen Werten. Am besten –im- Modell ausprobieren (Fremdlicht, Fenster…).

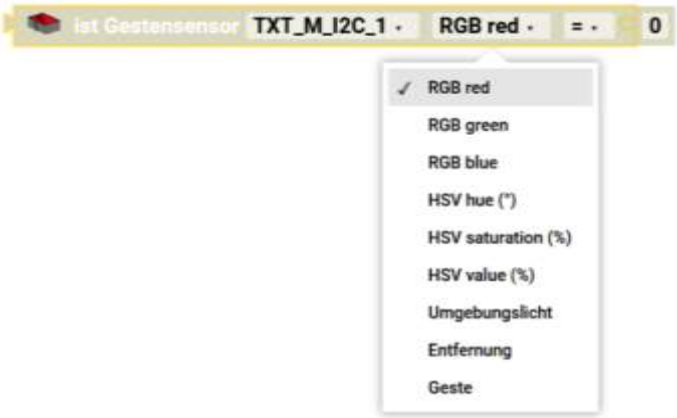
Siehe auch:
https://de.wikipedia.org/wiki/HSV-Farbraum
https://www.fischertechnik.de/de-de/service/elearning/lehren/base-set-und-add-ons

Auf der Seite steht u.a. beschrieben, wie aus den RGB Werten der HSV Wert (Winkel) berechnet wird.



HSV Farbraum: Winkel = Farbe

ist Gestensensor RGB



Über die Dropdown-Menüs (kleines Dreieck rechts) kann ausgewählt, wie der Wert mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Erklärung der Möglichkeiten siehe Block vorher.

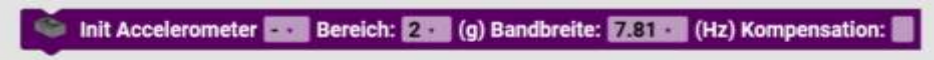
Kombisensor

Der Kombisensor vereint die drei Funktion Beschleunigungssensor (Accelerometer), Gyroskop und Kompasssensor (Magnetometer) in einem Bauteil. Der Sensor muss vor Gebrauch initialisiert werden. Intern ist ein BMX055 Sensor von Bosch verbaut.

Die 6-Polige neuere Variante vom Sensor läuft direkt mit den 3,3V aus dem Übertragungskabel und braucht keinen 9V Anschluss mehr. Aktuell verkauft fischertechnik allerdings den 10-Poligen mit dabei liegendem Adapter.

Beschleunigungssensor (Accelerometer)
Initialisieren

Init Accelerometer Bereich Bandbreite Kompensation



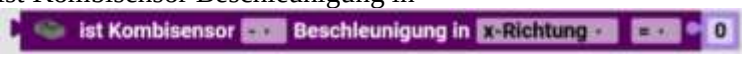
Abrufen

hole Kombisensor Beschleunigung in []

Mit **hole Kombisensor Beschleunigung in []** erhält man die Beschleunigung in einer Raumrichtungen. Die gewünschte Raumrichtung kann über das Dropdown-Menü (kleines Dreieck) gewählt werden. Die Beschleunigung wird in g angegeben.

Abfragen

ist Kombisensor Beschleunigung in



Um abzufragen, ob man eine bestimmte Beschleunigung misst, wird der Block **ist Kombisensor Beschleunigung in [] [] ...** genutzt. Über die Dropdown-Menüs (kleines Dreieck) kann ausgewählt, wie die Beschleunigung mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >) und welche Raumrichtung abgefragt werden soll. Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Beschleunigung in x-Richtung größer als 10 ist.

Gyroskop
Abrufen

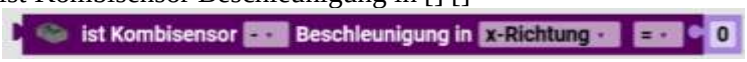
hole Kombisensor Beschleunigung in []

Mit **hole Kombisensor Beschleunigung in []** erhält man die Beschleunigung in einer Raumrichtungen. Die gewünschte Raumrichtung kann über das Dropdown-Menü (kleines Dreieck) gewählt werden. Die Werte vom Block **ist Kombisensor Beschleunigung in,** werden in °/s angegeben.

Insgesamt ist es nicht so einfach diesen Sensor nur mit Python anzusteuern. Man sollte auf fertige Libs zurückgreifen. Unterandrem ist der Sensor mit der Robo Pro Sprache gut beschrieben. Es gibt auch Beispiele mit Python für den Raspberry Pi die man (bis auf den I2C Bus) auch für den TXT 4.0 nutzen kann.

Abfragen

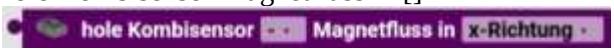
ist Kombisensor Beschleunigung in [] []



Um abzufragen, ob man eine bestimmte Winkelgeschwindigkeit misst, wird der Block **ist Kombisensor Beschleunigung in [] [] ...** genutzt. Über die Dropdown-Menüs (kleines Dreieck) kann ausgewählt, wie die Rotation mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >) und welche Raumrichtung abgefragt werden soll. Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Rotation in x-Richtung größer als 10 ist.

Kompasssensor
Abrufen

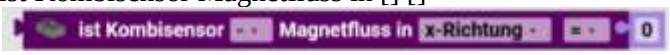
hole Kombisensor Magnetfluss in []



Mit **hole Kombisensor Magnetfluss in []** erhält man den magnetischen Fluss in einer Raumrichtungen. Die gewünschte Raumrichtung kann über das Dropdown-Menü (kleines Dreieck) gewählt werden. Der magnetische Fluss wird in µT angegeben.

Abfragen

ist Kombisensor Magnetfluss in [] []



Um abzufragen, ob man einen bestimmten magnetischen Fluss misst, wird der Block **ist Kombisensor Magnetfluss in [] [] ...** genutzt. Über die Dropdown-Menüs (kleines Dreieck) kann ausgewählt, wie der magnetische Fluss mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >) und welche Raumrichtung abgefragt werden soll. Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der magnetische Fluss in x-Richtung größer als 10 ist.

Umweltsensor

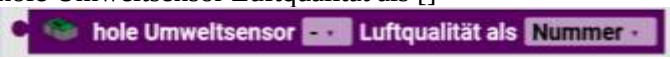
Der Umweltsensor vereint die vier Funktionen Luftqualitätssensor, Luftfeuchtigkeitssensor, Barometer und Thermometer in einem Bauteil. Intern ist ein BME680 Sensor verbaut.

Luftqualitätssensor
Kalibrieren/Initialisieren

Vor der ersten Messung muss der Sensor kalibriert werden. Wenn man längere Zeit Messungen macht, kann man den Sensor abfragen, ob eine weitere Kalibrierung vorgenommen werden muss.

Abrufen

hole Umweltsensor Luftqualität als []



Mit dem Block **hole Umweltsensor Luftqualität als []** kann man die Luftqualität messen. Über das Dropdown-Menü (kleines Dreieck) kann gewählt werden, ob die Luftqualität als Zahlenwert (von 0 bis 500) oder als Text zurückgegeben werden soll.

Abfragen

ist Umweltsensor Luftqualität

Um abzufragen, ob man eine bestimmte Luftqualität misst, wird der Block **ist Umweltsensor Luftqualität [] ...** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt, wie die Luftqualität mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Luftqualität größer als 10 ist.

Barometer
Abrufen

hole Umweltsensor Luftdruck

Mit dem Block **hole Umweltsensor Luftdruck** kann man den Luftdruck messen.

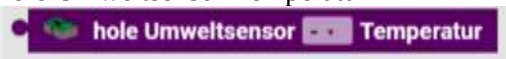
Abfragen

ist Umweltsensor Luftdruck

Um abzufragen, ob man einen bestimmten Luftdruck misst, wird der Block **ist Umweltsensor Luftdruck [] ...** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt, wie der Luftdruck mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der Luftdruck größer als 10 ist.

Thermometer
Abrufen

hole Umweltsensor Temperatur



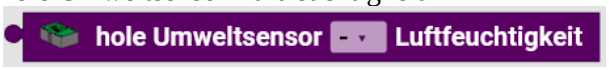
Mit dem Block **hole Umweltsensor Temperatur** kann man die Temperatur messen.

ist Umweltsensor Temperatur []
Abfragen

Um abzufragen, ob man eine bestimmte Temperatur misst, wird der Block **ist Umweltsensor Temperatur [] ...** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt, wie die Temperatur mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Temperatur größer als 10 ist.

Luftfeuchtigkeitssensor
Abrufen

hole Umweltsensor Luftfeuchtigkeit



Mit dem Block **hole Umweltsensor Luftfeuchtigkeit** kann man die Luftfeuchtigkeit messen.

ist Umweltsensor Luftfeuchtigkeit []
Abfragen

Um abzufragen, ob man eine bestimmte Luftfeuchtigkeit misst, wird der Block **ist Umweltsensor Luftfeuchtigkeit [] ...** genutzt. Über das Dropdown-Menü (kleines Dreieck) kann ausgewählt, wie die Luftfeuchtigkeit mit dem eingegebenen Wert verglichen werden soll (<, ≤, =, ≠, ≥, >). Der Vergleichswert wird in das Zahlenfeld am Ende des Blocks eingegeben. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Luftfeuchtigkeit größer als 10 ist.

Programmbeispiele für die Sensoren von fischertechnik

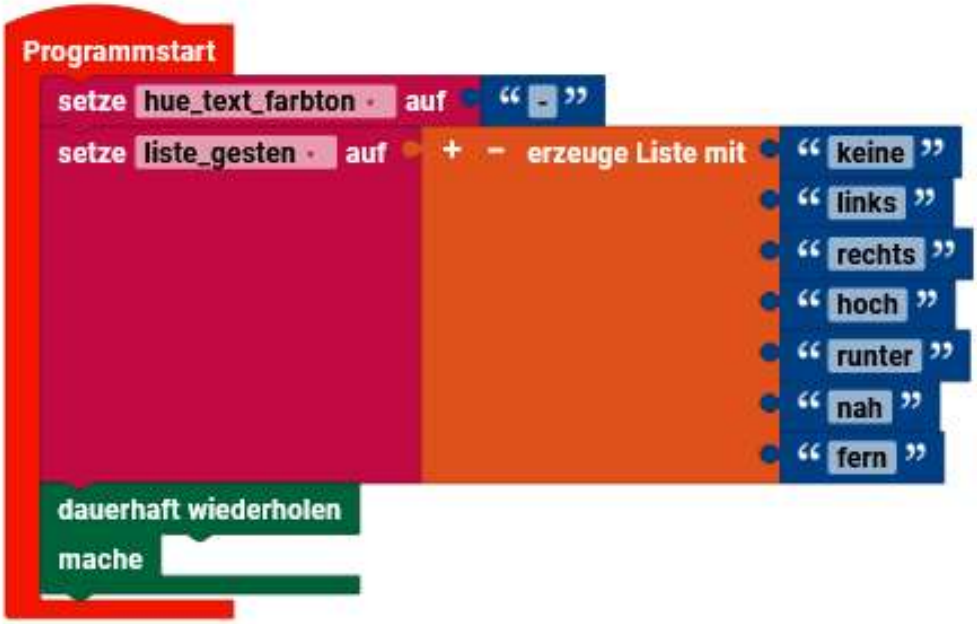
für den Sensor	**gibt es beim „Kasten“**	**Programmname:**
Kombisensor	TXT 4.0	test_combi_sensor
	Robotics Add On: Competition	gui_sensor_combi
Gestensensor	TXT 4.0	test_gesture_sensor
	Robotics Add On: Competition	gui_sensor_gesture
Umweltsensor	TXT 4.0	test_environmental_sensor
	Robotics Add On: IoT	IoT_Barometer
		IoT_CO2_Signal_Light
		IoT_Indoor_Air_Quality
		oT_Indoor_Climate
		Sensorstation

Sie sind bei Robo Pro Coding zu finden unter:
Burgermenü/Neu/Optionen/ft Beispiele/TXT 4.0 Controller/
Burgermenü/Neu/Optionen/ft Beispiele/ Robotics Add On: Competition /
Burgermenü/Neu/Optionen/ft Beispiele/ Robotics Add On: IoT /

Das Programm „gui_sensor_gesture“ eingedeutscht, mit Erklärungen

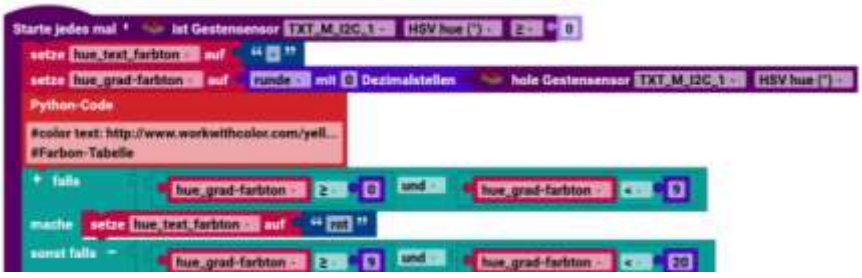
Das Originalprogramm benutzt englische Begriffe. Entweder wurden sie ersetzt oder der deutsche Begriff dahinter geschrieben.

Hauptprogramm



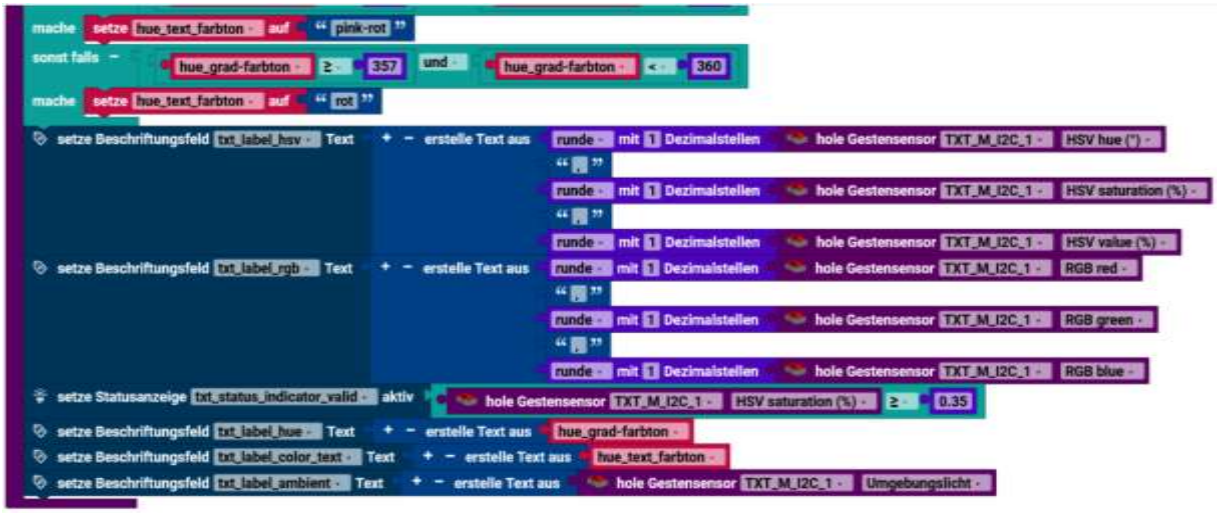
Nach dem Starten wird in die Variable „hue_text_farbton“ ein Minus für Leer gegeben. Dann wird eine Liste erzeugt mit 7 Begriffen erzeugt. Sie sind für die erkannte Bewegungsrichtung (Geste). Dann wir eine leere Endlosschleife durchlaufen. Dadurch läuft das Programm weiter, bis man Stopp auf dem Controller oder in Robo Pro Coding drückt. Es werden also nur die Variablen gesetzt.

Unterprogramm Farbton >= 0



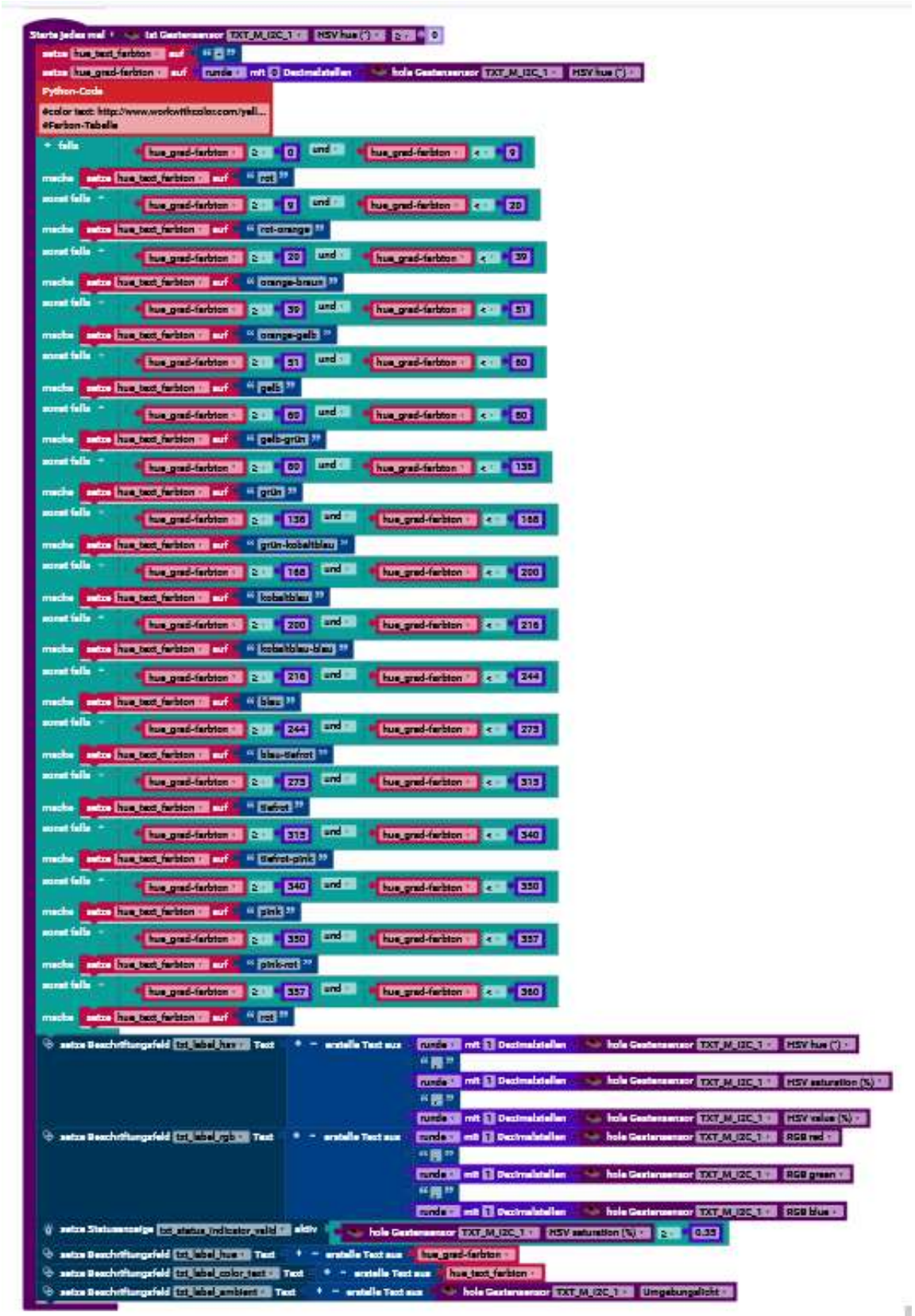
Das Unterprogramm startet nur wenn der Farbton größer gleich 0 ist. Als erstes wird die Variable hue_text_farbton auf „-“ gesetzt. Dann wird vom Sensor der Farbtonwert geholt und in die Variable „hue_grad-Farbton“ geschrieben. Der Python-Code ist nur ein Hinweis auf eine Hue-Farbtabelle.

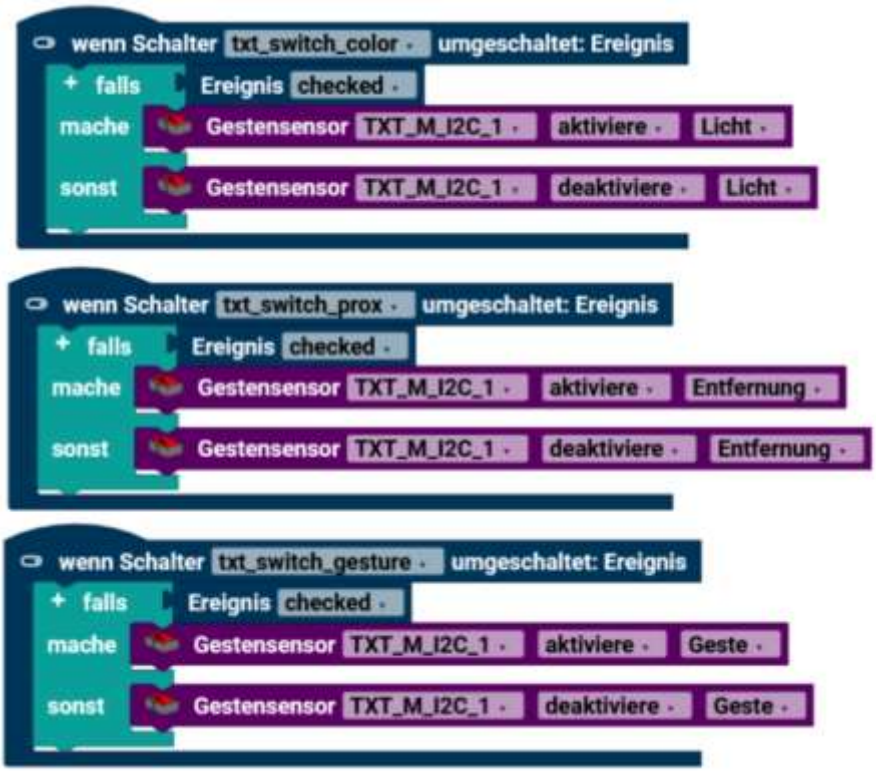
Nun wird der geholte Sensorwert mit einer Tabelle verglichen und der dazugehörige Text (Farbname) in die Variable „hue_text_farbton“ geschrieben.



Am Ende wird im Beschriftungsfeld „txt_label_hsv“ der Text, aus den Zahlenwerten vom Sensor, geschrieben. In das Beschriftungsfeld „txt_label_rgb“ werden die RGB-Werte vom Sensor als Text geschrieben. Wenn die Sättigung größer als 0,35 ist, wird die Statusanzeige gesetzt. Der Farbton-Sensorwert wird auf dem Beschriftungsfeld „txt_label_hue“ als Text und anschließend der Name der Farbe aus der Variablen „hue_text_farbton“ dargestellt. Zuletzt wird der Sensorwert „Umgebungslicht“ vom Sensor in das Beschriftungsfeld „txt_label_ambient“ ausgegeben.

Hier die Übersicht über das gesamte Unterprogramm.





Die drei Unterprogramme reagieren auf jeweils einen virtuellen Schalter am Bedienfeld. Sie aktivieren oder aktivieren Licht, Entfernung oder Geste.

USB

Über den USB-Anschluss kann die Kamera mit integriertem Mikrophon angeschlossen werden. Kamera und Mikrophon werden hier getrennt betrachtet. Um die Funktionen der Kamera zu nutzen, muss man sie erst in der Kamerakonfiguration einstellen. Man kommt am schnellsten in die Kamerakonfiguration, wenn noch nichts konfiguriert wurde, über „Kamerakonfiguration hinzufügen“.

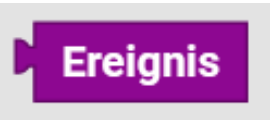
Bis auf die Blöcke vom Mikrofon sind die restlichen Blöcke ausgegraut.

Kamera

fischertechnik Kamera

Die Kamera kann man als Bewegungsdetektor, als Farbdetektor, als Balldetektor und als Liniendetektor nutzen. Auch gleichzeitig. Man kann auf ein leeres Kamerabild Felder einzeichnen die auf bestimme Sachen reagieren, wie Bewegung, Farbe, Ball oder Linie. Auch hier sind mehrere möglich.

Ereignis



Die Callback-Antwort. Ist ein Ereignis eingetroffen. Kann als Abfrage genutzt werden.

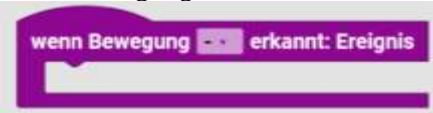
Bewegungsdetektor
Um die Kamera als Bewegungsdetektor zu nutzen, muss man in der Kamerakonfiguration das Männchen-Symbol in die gerasterte Fläche ziehen, dann öffnet sich rechts ein Fenster in dem man unter Inspektor

- den Pixelbereich, in dem auf Bewegung überprüft wird,
- die Position dieses Bereichs (auf dem angegebenen Punkt liegt die obere linke Ecke des Bereichs),
- den Namen des Bewegungsdetektors und
- die Toleranz

festlegen kann.

Bewegungsdetektor Programm

wenn Bewegung erkannt



Das Bewegungsdetektor-Programm läuft ab, wenn eine Bewegung erkannt wurde. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Eingabe-Programm läuft im Block **wenn Bewegung erkannt** ab.

Farbdetektor

Achtung! Farbraum siehe auch Gestensensor. Farbdetektor hat auch andere Werte als OpenCV!

Um die Kamera als Farbdetektor zu nutzen, muss man in der Kamerakonfiguration das Pipetten-Symbol in die gerasterte Fläche ziehen, dann öffnet sich rechts ein Fenster in dem man unter Inspektor

- den Pixelbereich, in dem auf Farbe überprüft wird,
- die Position dieses Bereichs (auf dem angegebenen Punkt liegt die obere linke Ecke des Bereichs),
- den Namen des Farbdetektors und
- den Kontrast

festlegen kann.

Abrufen

hole Farbe als []

Mit **hole Farbe als []** erhält man die erkannte Farbe im Hexadezimal oder im RGB Format. Das Format kann über das Dropdown-Menü (kleines Dreieck) eingestellt werden.

Abfragen

ist Farbe detektiert

Um abzufragen, ob der Detektor eine Farbe wahrgenommen hat, wird der Block **ist Farbe detektiert** genutzt. Dieser Block kann als Bedingung genutzt werden.

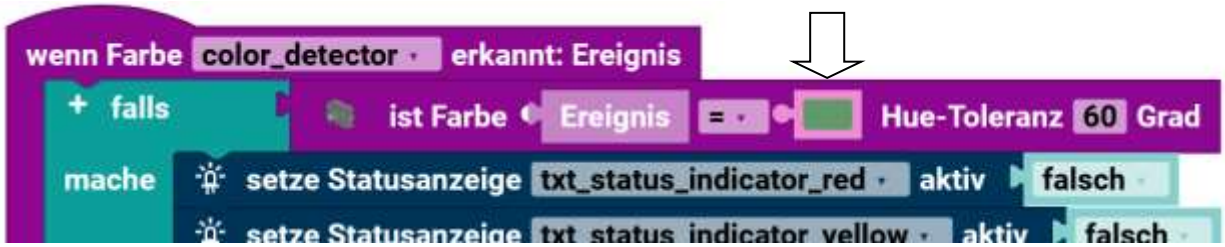
Um abzufragen, ob der Detektor eine bestimmte Farbe wahrnimmt, wird dieser Block

Aktueller Block

Alte Darstellung

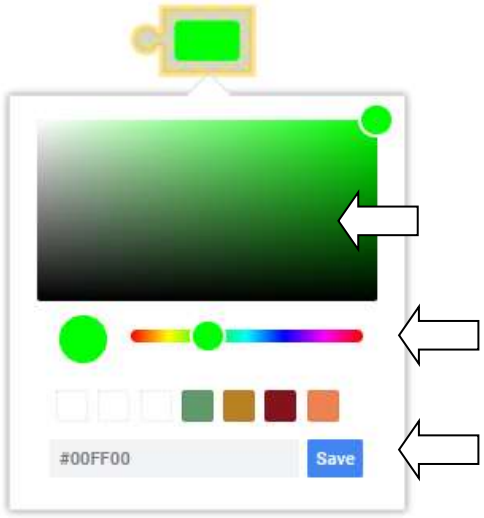
genutzt. Mit dem Block kann man die aufgenommene Farbe mit einer eingegeben vergleichen. Über das Dropdown-Menü (kleines Dreieck) kann gewählt werden, ob die eingestellte Farbe gleich oder ungleich der gefilmten Farbe seien soll. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die aufgenommene Farbe Rot ist.

Alte Darstellung



Aktuelle Darstellung der Blöcke

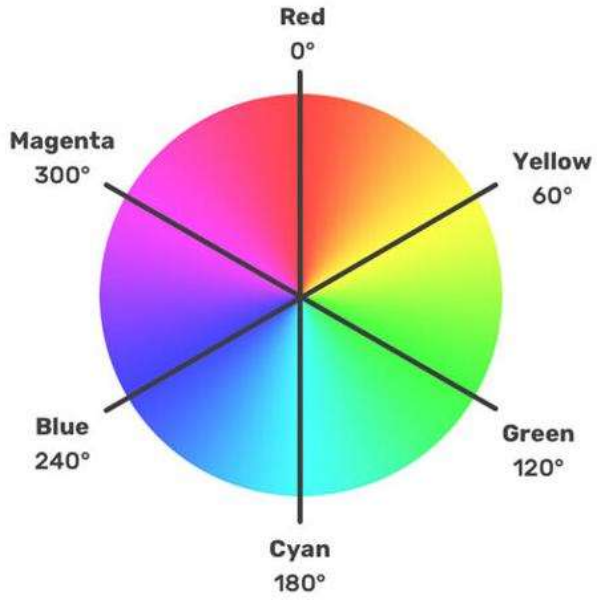
Die Farbauswahl, die erkannt werden soll, geht über das farbige kleine Rechteck. Das ist im Block schon vorhanden oder man nimmt den aus dem Reiter Util.



Durch einen Klick in das farbige Rechteck erscheint ein weißer Ring, den man auch verschieben kann. So wählt man die Farbsätigung (heller-dunkler) aus. Mit dem Regler wird die Grundfarbe gewählt. Man kann die Farbe auch als RGB-Wert eingeben.
Mit Save wird die Einstellung für den Block gespeichert.

Achtung: Die Zahlenangaben sind hexadezimal, aber die Abfrage selbst ist der HVS Farbraum (Hue-Winkel). Deswegen ist die Toleranz auch in Grad angegeben.

HVS Farbraum



Winkel der Farben

Umrechnung

Die RGB (Rot, Grün, Blau) Werte, können jeweils zwischen 0 und 255 liegen. Wenn man diese Dezimalzahlen ohne Trennzeichen hintereinanderschreibt, kann man nicht genau sagen, wo die einzelnen Werte anfangen und aufhören. Man nimmt deswegen Hexadezimalzahlen.
Die gehen von 0 bis F. 0=0, 1=1, 2=2…9=9, 10=A, 11=B, 12=C, 13=D…15=F, 16=10, 17=11… 255=FF
Man kann z.B. mit dem Windowsrechner die Dezimalzahlen auf Hexadezimalzahlen umrechnen.

Erweiterung des Programms mit der „Hexadezimalen Anzeige“ und der „RGB Anzeige“ der ermittelten Farbe.

Das System der **hexadezimalen Farbdefinition** findest du in vielen Bereichen des computergestützten Designs. Bei diesem System wird eine Farbe durch drei aufeinander folgende Hexadezimalzahlen dargestellt, die jeweils für eine Farbe des RGB-Farbraums stehen.

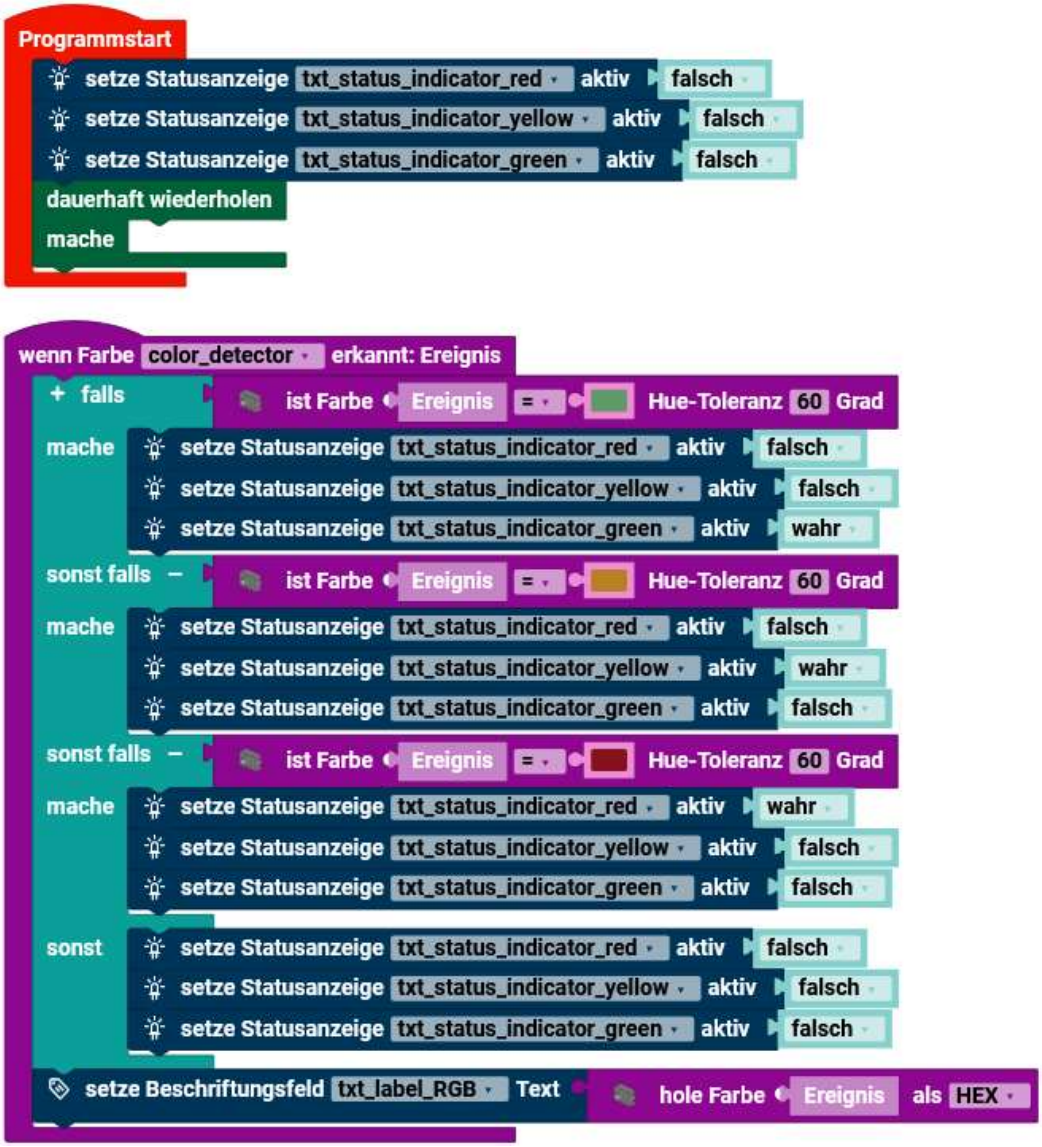
RGB steht dabei als Abkürzung für die Anteile der Basisfarben Rot, Grün und Blau an der Mischfarbe.

Grundfarben

Farbe	Hex-Wert	R	G	B	Komplementär zu
Schwarz	#000000	0	0	0	Weiß
Grau	#888888	136	136	136	#777777
Weiß	#FFFFFF	255	255	255	Schwarz
Rot	#FF0000	255	0	0	Cyan
Dunkelrot	#880000	136	0	0	#77FFFF
Grün	#00FF00	0	255	0	Magenta
Dunkelgrün	#008800	0	136	0	#FF77FF
Blau	#0000FF	0	0	255	Gelb
Dunkelblau	#000088	0	0	136	#FFFF77

Üblich ist die hexadezimale Farbdefinition in sechsstelliger Form, das heißt, als eine Aneinanderreihung von drei jeweils zweistellig geschriebenen Hexadezimalzahlen, nach dem Schema: #RRGGBB, die es erlaubt, pro Objekt (= 256) verschiedene Zustände zu definieren – von 0 für eine komplett „ausgeschaltete“ Farbe und 255 (FF) für 100 % Farbsättigung. Es wird die additive Farbmischung zum Mischen der Farben benutzt; der Code #FFFF00 ergibt die Farbe Gelb, weil Rot mit Grün gemischt wird.

Beispiel Farberkennung, aus „cameraman_detect_color“



Das Hauptprogramm „löscht“ die Statusanzeigen und macht eine leere Endlosschleife.
Die Farberkennung läuft parallel zum Hauptprogramm. Wenn eine Farbe erkannt wird, wird überprüft, ob die Farbe mit einer der programmierten übereinstimmt. Das ist nicht der genaue Wert, sondern ein Farbbreich der über die Toleranz angegeben wird. Je nachdem welche übereinstimmt, wir die Statusanzeige aktiviert und zeigt somit die erkannte Farbe an. Zuletzt wird der RGB Wert angezeigt.

Farbdetektor Programm

wenn Farbe erkannt

Das Farbdetektor-Programm läuft ab, wenn eine Farbe erkannt wurde. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Eingabe-Programm läuft im Block **wenn Farbe erkannt** ab.

Balldetektor

Um die Kamera als Balldetektor zu nutzen, muss man in der Kamerakonfiguration das Bälle-Symbol in die gerasterte Fläche ziehen, dann öffnet sich rechts ein Fenster in dem man unter Inspektor

- den Pixelbereich, in dem auf Bälle überprüft wird,
- die Position dieses Bereichs (auf dem angegebenen Punkt liegt die obere linke Ecke des Bereichs),
- den Namen des Balldetektors,
- den Bereich, in dem der Durchmesser des Balls liegt,
- den Bereich der x-Achse,
- die Farbe des Balles und
- die Farbtoleranz

festlegen kann.

Abrufen

hole [] des Balls

Mit **hole [] des Balls** erhält man x-Position, y-Position, Radius oder Durchmesser, des Balles.

Abfragen

ist Ball detektiert

Um abzufragen, ob der Detektor einen Ball wahrgenommen hat, wird der Block **ist Ball detektiert** genutzt. Dieser Block kann als Bedingung genutzt werden.

Um abzufragen, ob der Detektor einen Ball mit einer bestimmten x-Position, y-Position, Radius oder Durchmesser wahrnimmt wird dieser Block

genutzt. Mit dem Block kann man Spezifikationen, des aufgenommenen Balles, mit einem eingegebenen Wert vergleichen. Über die Dropdown-Menüs (kleines Dreieck) kann gewählt werden, was und mit welchem Vergleichsoperator verglichen werden soll. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn der Durchmesser, des aufgenommenen Balles, 5 ist.

Balldetektor Programm

Das Balldetektor-Programm läuft ab, wenn eine Farbe erkannt wurde. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Eingabe-Programm läuft im Block **wenn Ball erkannt** ab.

Liniendetektor

Um die Kamera als Liniendetektor zu nutzen, muss man in der Kamerakonfiguration das Symbol mit den Punkten auf einer Linie in die gerasterte Fläche ziehen, dann öffnet sich rechts ein Fenster in dem man unter Inspektor

- den Pixelbereich, in dem auf Linien überprüft wird,
- die Position dieses Bereichs (auf dem angegebenen Punkt liegt die obere linke Ecke des Bereichs),
- den Namen des Liniendetektors,
- die Anzahl an Linien, die erkannt werden sollen, und
- den Bereich, in dem die Breite der Linie(n) liegt

festlegen kann.

Abrufen

hole [] der Linie []

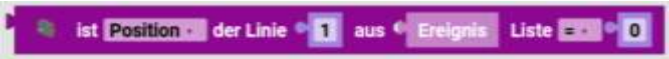
Mit **hole [] der Linie []** erhält man Position oder Breite einer der maximal fünf Linien ab.

hole Farbe der Linie [] als []

Mit **hole Farbe der Linie [] als []** erhält die Farbe einer Linie im Hexadezimal oder im RGB Format ausgeben lassen. Das Format kann über das Dropdown-Menü (kleines Dreieck) eingestellt werden.

Abfragen

ist … der Linie



Um abzufragen, ob der Detektor eine Linie wahrgenommen hat, wird der Block **ist Linie detektiert** genutzt. Dieser Block kann als Bedingung genutzt werden.

Um abzufragen, ob der Detektor eine Linie mit einer bestimmten Position oder Breite wahrnimmt, wird dieser Block

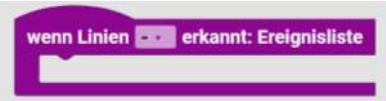
genutzt. Mit dem Block kann man Spezifikationen, der aufgenommenen Linie(n), mit einem eingegebenen Wert vergleichen. Über die Dropdown-Menüs (kleines Dreieck) kann gewählt werden, was und mit welchem Vergleichsoperator verglichen werden soll. Dieser Block kann als Bedingung genutzt werden. Im Beispiel wird der Motor gestoppt, wenn die Breite, der aufgenommenen Linie, kleiner als 2 ist.

Um abzufragen, ob der Detektor eine Linie mit einer bestimmten Farbe wahrnimmt, wird dieser Block

genutzt. Mit dem Block kann man die aufgenommene Linienfarbe mit einer eingegeben vergleichen. Über das Dropdown-Menü (kleines Dreieck) kann gewählt werden, ob die eingestellte Farbe gleich oder ungleich der gefilmten Farbe seien soll. Dieser Block kann als Bedingung genutzt werden.

Liniendetektor Programm

wenn Linien erkannt



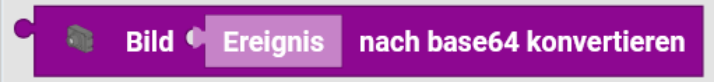
Das Liniendetektor-Programm läuft ab, wenn eine oder mehrere Linien erkannt wurden. Es wird separat vom Hauptprogramm geschrieben. Variablen funktionieren global über beide Programme hinweg. Das Eingabe-Programm läuft im Block **wenn Linien erkannt** ab.

Bild

wenn Bild geändert

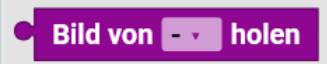
Das Ereignis tritt ein, wenn ein neues Bild verfügbar ist

Bild … nach base64 konvertieren



Gibt das Bild als base64 konvertierte Zeichenkette zurück.

Bild von … holen



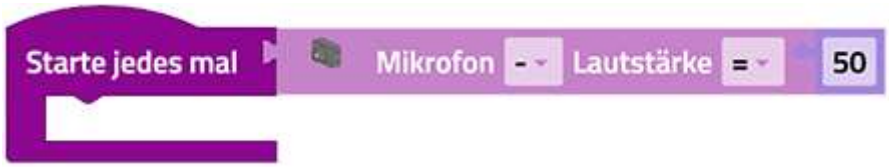
Liefert das aktuelle Bild aus der ausgewählten Kamera

Mikrofon

Das in der Kamera integrierte Mikrofon kann als Lautstärkedetektor genutzt werden.

Starte jedes mal

Der **Starte jedes mal-Block** bietet die Möglichkeit ein Programm ablaufen zulassen, wenn eine Bedingung erfüllt ist. Er funktioniert also ähnlich wie eine Fallunterscheidung wird aber nicht nur einmal durchlaufen, sondern jedes Mal, wenn die Bedingung erfüllt ist, während des gesamten Ablaufs des Programms. Der **Starte jedes mal-Block**:



Ist eine Abkürzung für folgendes Konstrukt:

Man kann in den **Starte jedes mal-Block** der Kategorie Mikrofron alle Bedingungen aus eben dieser Kategorie einsetzten.

Hinweis: Der Programmabschnitt innerhalb des Starte jedes mal-Block sollte kurzgehalten werden und keine blockierenden Aufrufe oder Endlosschleifen enthalten, so dass dieser Teil des Programms schnell abgearbeitet werden kann.

Lautstärkedetektor – Mikrofon der USB Kamera

Abrufen

Mikrofon Lautstärke 1

Mit **Mikrofon Lautstärke** erhält man die Lautstärke in Dezibel.

Mikrofon Lautstärke 2

Um abzufragen, ob der Lautstärkedetektor eine bestimmte Lautstärke wahrnimmt, wird dieser Block

genutzt. Mit dem Block kann man die aufgenommene Lautstärke mit einer eingegebenen vergleichen. Über das Dropdown-Menü (kleines Dreieck) kann gewählt werden, mit welchem Vergleichsoperator verglichen werden soll. Dieser Block kann als Bedingung genutzt werden.

Verarbeitung

Die Verarbeitung kann vergleichen, berechnen…

Logik

Boolesche Logik ist ein einfaches mathematisches System, das zwei Werte hat:

- **wahr**
- **falsch**

Logikblöcke in ROBO Pro Coding sind in der Regel dafür da, Bedingungen und Schleifen zu kontrollieren.

Hier ein Beispiel:

Wenn der Wert der Variable x größer als 100 ist, ist die Bedingung wahr und der Text "Was für eine große Zahl!" wird ausgegeben. Ist der Wert von x nicht größer als 100, ist die Bedingung falsch und "Das ist nicht sehr groß." wird ausgegeben. Boolesche Werte können auch in Variablen gespeichert werden und an Funktionen weitergegeben werden, genauso wie Zahlen, Text und Listenwerte.

Blöcke

Wenn ein Block einen Booleschen Wert als Eingabe erwartet, wird eine fehlende Eingabe als **falsch** interpretiert. Nicht-Boolesche Werte können nicht direkt dort eingefügt werden, wo Boolesche Werte erwartet werden, obwohl es möglich (aber nicht ratsam) ist, einen nicht-Booleschen Wert in einer Variablen zu speichern und diese dann in die Bedingungseingabe einzufügen. Diese Methode wird nicht empfohlen, und ihr Verhalten kann sich in zukünftigen Versionen von ROBO Pro Coding ändern.

Werte - wahr / falsch Block

Ein einzelner Block mit einer Dropdown-Liste, die entweder **wahr** oder **falsch** angibt, kann verwendet werden, um einen Booleschen Wert abzurufen:

wahr / falsch Block

Am Ausgang links, liegt der ausgewählte Wert an.

Falls mache Block

Der „falls mache“ Block prüft, ob eine Bedingung wahr oder falsch ist. Wenn sie wahr ist werden die weiteren Blöcke innerhalb ausgeführt. Wenn die Bedingung falsch ist, wird zum nächsten Block weitergegangen. Durch das Klicken auf das + wird eine weitere Prüfung „sonst falls“ mit einer anderen Bedingung eingefügt.

Falls mache sonst Block

Der „falls mache“ Block prüft, ob eine Bedingung wahr oder falsch ist. Wenn sie wahr ist werden die weiteren Blöcke innerhalb ausgeführt. Wenn die Bedingung falsch ist, werden die weiteren Blöcke im unteren Teil bei „sonst“ ausgeführt.

Automatisches Fahrtlicht. Falls es dunkel ist, mache das Licht an, sonst schalte das Licht aus.

Vergleichsoperatoren

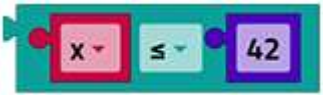
Es gibt sechs Vergleichsoperatoren. Jedem werden zwei Eingaben (normalerweise zwei Zahlen) übergeben und der Vergleichsoperator gibt **wahr** oder **falsch** zurück, je nachdem, wie die Eingaben miteinander verglichen werden.

gleich

nicht gleich

kleiner als

größer als



kleiner als oder gleich

größer als oder gleich

Die sechs Operatoren sind: gleich, nicht gleich, kleiner als, größer als, kleiner als oder gleich, größer als oder gleich.

Logische Operatoren

und / oder Block

und / oder Block

Der **und**-Block gibt dann und nur dann **wahr** zurück, wenn seine beiden Eingangswerte wahr sind.

Der **oder**-Block gibt **wahr** zurück, wenn mindestens einer seiner beiden Eingangswerte wahr ist.

Nicht Block

Der **nicht**-Block wandelt eine boolesche Eingabe in ihr Gegenteil um. Zum Beispiel ist das Ergebnis von:

Es liegt links am Ausgang falsch.

Wenn keine Eingabe rechts von einem Block erfolgt, wird der Wert **wahr** angenommen, so dass der folgende Block den Wert **falsch** erzeugt:

Es liegt hier ein „falsch“ links am Ausgang an.

Es wird jedoch nicht empfohlen, eine Eingabe leer zu lassen.

dreier Operator prüfe falls wahr / falls falsch

Der dreier Operator verhält sich wie ein Miniatur-**wenn-sonst**-Block. Er nimmt drei Eingangswerte entgegen. Der erste Eingangswert ist die zu testende boolesche Bedingung, der zweite Eingangswert ist der Wert, der zurückgegeben werden soll, wenn der Test **wahr** ergibt, der dritte Eingangswert ist der Wert, der zurückgegeben werden soll, wenn der Test falsch ergibt. Im folgenden Beispiel wird die Variable **Farbe** auf Rot gesetzt, wenn die Variable **x** kleiner als 10 ist, andernfalls wird die Variable **Farbe** auf grün gesetzt.

Ein dreier Block kann immer durch einen **wenn-sonst**-Block ersetzt werden. Die folgenden zwei Beispiele sind genau gleich.

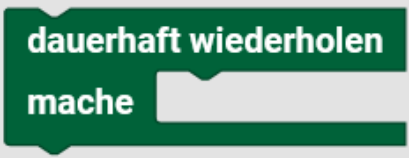
Schleifen

Der Bereich "Steuerung" enthält Blöcke, die steuern, ob andere Blöcke, die in ihrem Inneren platziert sind, ausgeführt werden. Es gibt zwei Arten von Steuerungsblöcken: **wenn-sonst**-Blöcke (die auf einer eigenen Seite beschrieben werden) und Blöcke, die steuern, wie oft ihr Inneres ausgeführt wird. Letztere werden Schleifen genannt, da ihr Inneres, auch als Schleifenkörper oder Körper bezeichnet, (möglicherweise) mehrfach wiederholt wird. Jeder Durchlauf einer Schleife wird auch als Iteration bezeichnet.

Blöcke zur Erstellung von Schleifen

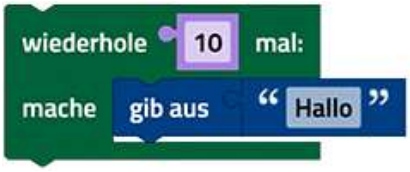
dauerhaft wiederholen

Der „**dauerhaft wiederholen“**-Block führt den Code in seinem Körper so lange aus, bis das Programm beendet wird.



Wiederhole x mal

Der **wiederhole**-Block führt den Code in seinem Körper, so häufig wie angegeben aus. Der folgende Block gibt zum Beispiel zehnmal "Hallo!" aus:



wiederhole-solange

Stelle dir ein Spiel vor, bei dem ein Spieler einen Würfel wirft und alle geworfenen Werte addiert, solange die Summe kleiner als 30 ist. Die folgenden Blöcke implementieren dieses Spiel:

1. Eine Variable namens „insgesamt**“** erhält einen Anfangswert von 0.
2. Die Schleife beginnt mit einer Überprüfung, ob insgesamt kleiner als 30 ist. Wenn ja, werden die Blöcke im Körper durchlaufen.
3. Eine Zufallszahl im Bereich von 1 bis 6 wird erzeugt (um einen Würfelwurf zu simulieren) und in einer Variablen namens „gewürfelt“ gespeichert.
4. Die gewürfelte Zahl wird ausgegeben.

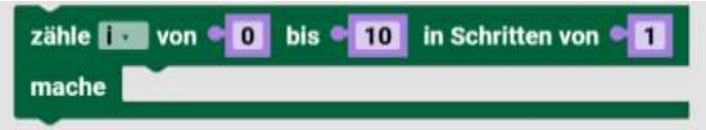
5. Die Variable „insgesamt“ wird um „gewürfelt“ erhöht.
6. Wenn das Ende der Schleife erreicht ist, geht die Steuerung zurück zu Schritt 2.

Nach Beendigung der Schleife werden alle nachfolgenden Blöcke (nicht dargestellt) durchlaufen. Im Beispiel endet der Schleifendurchlauf, nachdem eine gewisse Anzahl von Zufallszahlen im Bereich von 1 bis 6 ausgegeben wurde, und die Variable **insgesamt** hat dann als Wert die Summe dieser Zahlen, die mindestens 30 beträgt.

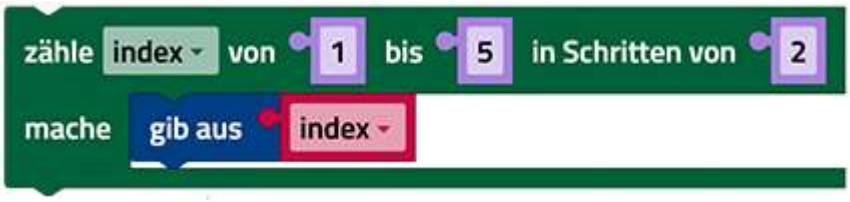
wiederhole-bis

wiederhole solange-Schleifen wiederholen ihren Körper, **solange** eine Bedingung erfüllt ist. **wiederhole bis**-Schleifen sind ähnlich, mit dem Unterschied, dass sie ihren Körper so lange wiederholen, **bis** eine bestimmte Bedingung erfüllt ist. Die folgenden Blöcke sind Äquivalent zum vorherigen Beispiel, weil die Schleife läuft, bis **insgesamt** größer oder gleich 30 ist.

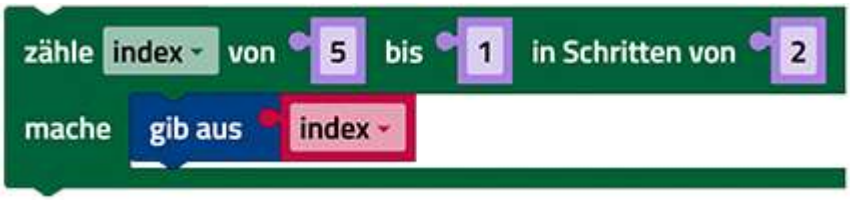
zählen-von-bis

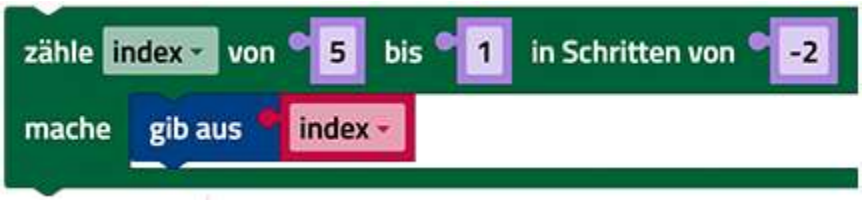


Die **zählen-von-bis**-Schleife erhöht einer Variable den Wert, beginnend mit einem ersten Wert, endend mit einem zweiten Wert und in Schritten von einem dritten Wert, wobei der Körper für jeden Wert der Variable einmal ausgeführt wird. Das folgende Programm gibt zum Beispiel die Zahlen 1, 3 und 5 aus.



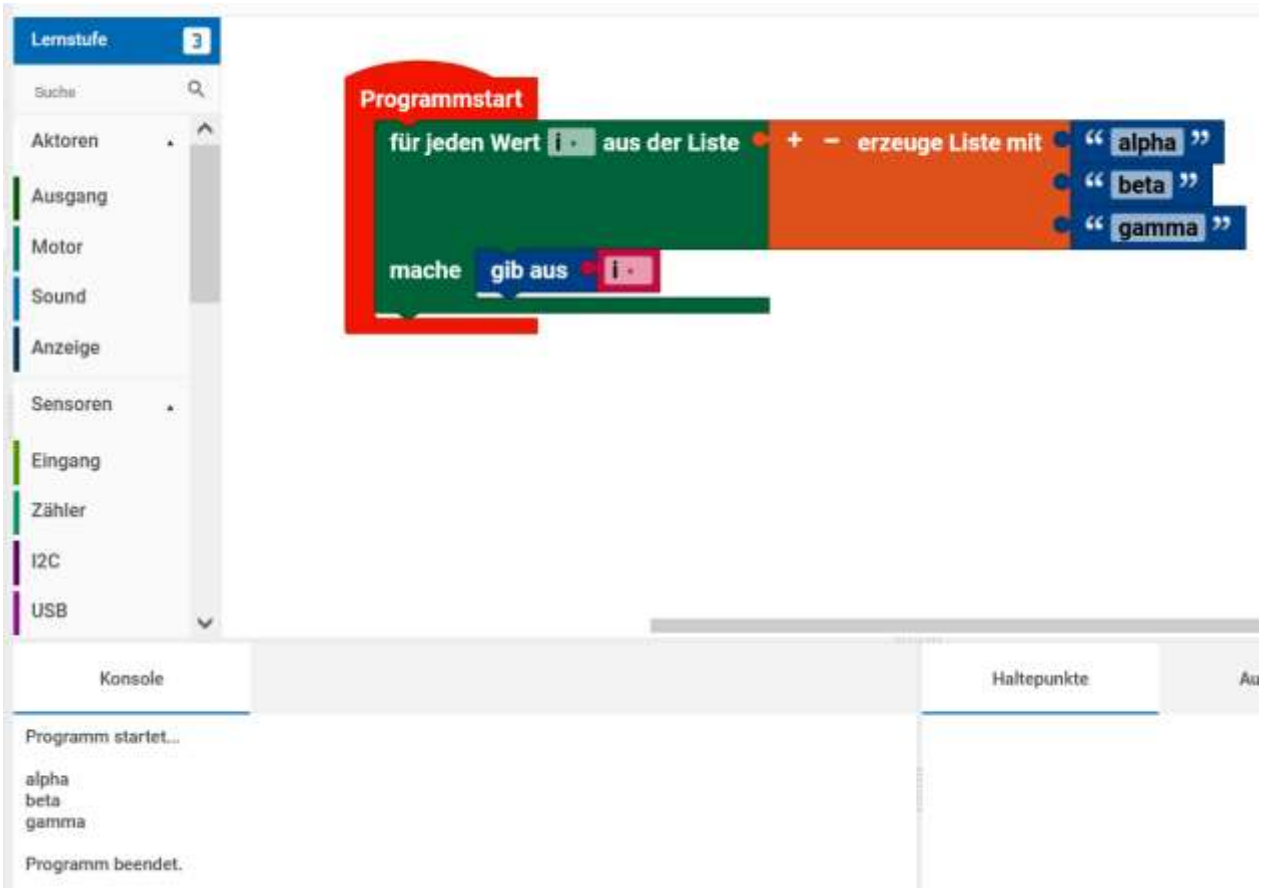
Wie die beiden folgenden Schleifen zeigen, werden jeweils die Zahlen 5, 3 und 1 ausgeben. Dieser erste Wert kann größer sein als der zweite. Das Verhalten ist das gleiche, egal ob der Zunahmebetrag (Inkrementbetrag) (dritter Wert) positiv oder negativ ist.





für jeden Wert

Der **für jeden**-Block ist ähnlich, wie die **zählen-von-bis**-Schleife, nur dass er statt der Schleifenvariable in einer numerischen Reihenfolge die Werte aus einer Liste der Reihe nach verwendet. Das folgende Programm gibt jedes Element der Liste "alpha", "beta", "gamma" aus:

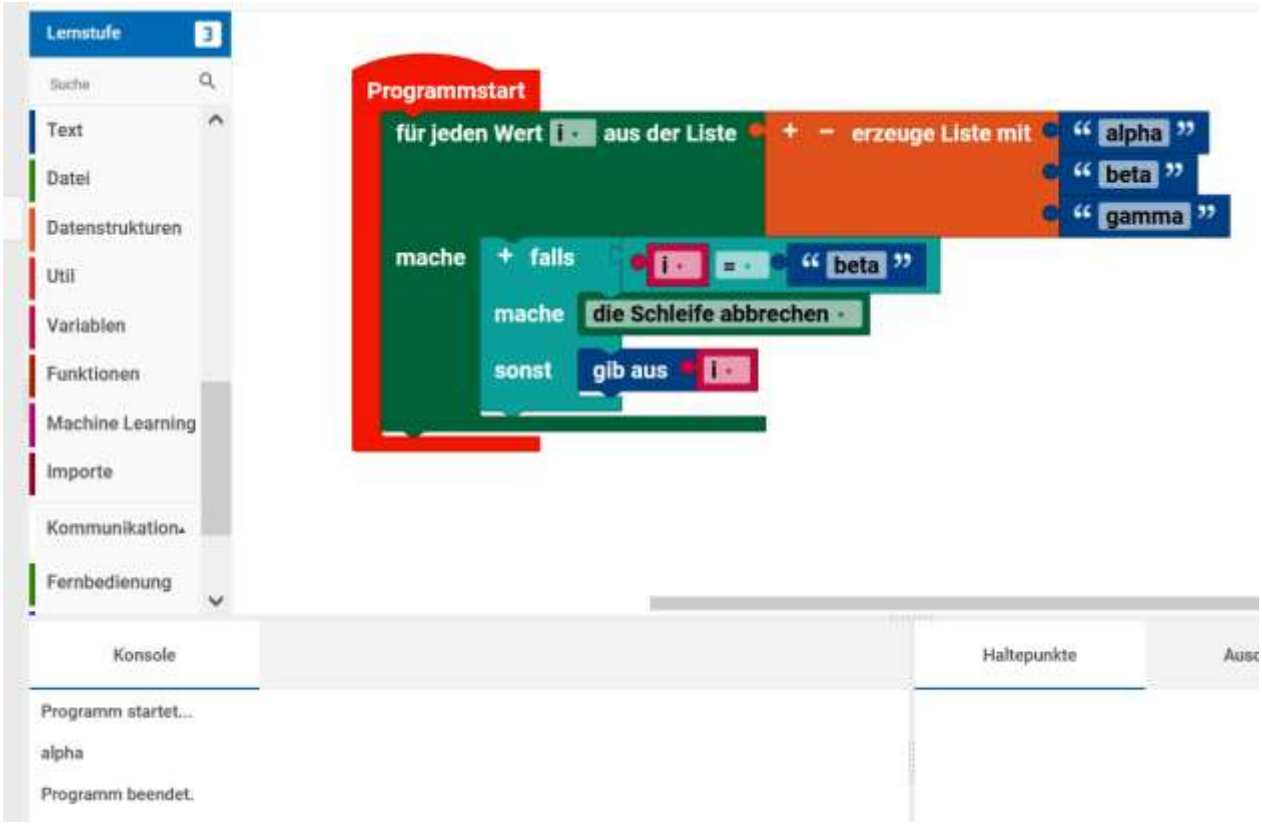


Schleifenabbruchblöcke

Die meisten Schleifen werden so lange durchlaufen, bis die Abbruchbedingung (bei **wiederhole**-Blöcken) erfüllt ist oder bis alle Werte der Schleifenvariable angenommen wurden (bei **zählen mit**- und **für jeden**-Schleifen). Zwei selten benötigte, aber gelegentlich nützliche Blöcke bieten zusätzliche Möglichkeiten zur Steuerung des Schleifenverhaltens. Sie können bei jeder Art von Schleife verwendet werden, auch wenn die folgenden Beispiele ihre Verwendung bei der **für jeden**-Schleife zeigen.

die Schleife abbrechen

Der „die Schleife abbrechen“-Block ermöglicht einen vorzeitigen Ausstieg aus einer Schleife. Das folgende Programm gibt bei der ersten Iteration (Durchlauf) "alpha" und bricht bei dem zweiten Durchlauf die Schleife ab, wenn die Schleifenvariable gleich "beta" ist. Der dritte Punkt in der Liste wird nie erreicht.



Beim ersten Aufruf erscheint der Block mit einem Warnsymbol.

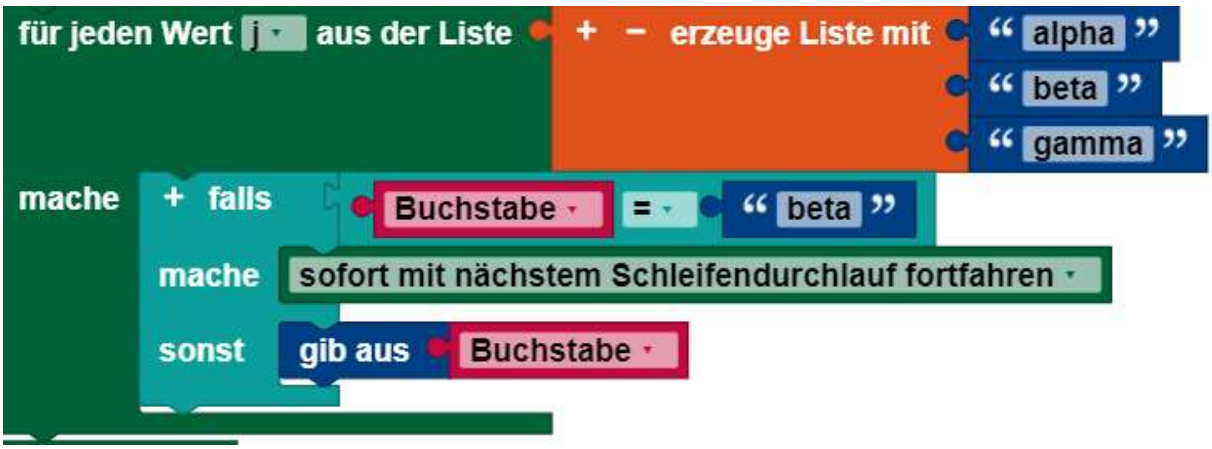
Sobald man ihn in einer Schleife hat, verschwindet das Warnsymbol.

sofort mit nächstem Schleifendurchlauf fortfahren

Den Block erhält man, wenn man das kleine Dreieck rechts anklickt und den unteren Eintrag auswählt.

„sofort mit nächstem Schleifendurchlauf fortfahren“ bewirkt, dass die restlichen Blöcke im Schleifenkörper übersprungen werden und die nächste Iteration der Schleife beginnt.
An den Block können keine weiteren Blöcke unten angeschlossen werden.

Das folgende Programm gibt bei der ersten Iteration der Schleife "alpha" aus. Bei dem zweiten Durchlauf wird der Block „sofort mit nächstem Schleifendurchlauf fortfahren“ ausgeführt, wodurch die Ausgabe von "beta" übersprungen wird. Bei der letzten Iteration wird "gamma" gedruckt.



Mathe

Die Blöcke, der Kategorie Mathematik werden genutzt, um Berechnungen anzustellen. Die Ergebnisse der Berechnungen können zum Beispiel als Werte für eine Variable verwendet werden. Die meisten Mathematik-Blöcke beziehen sich auf allgemeine mathematische Berechnungen und sollten selbsterklärend sein.

Eine Zahl

Nutze den Zahl-Block, um eine beliebige Zahl in dein Programm hinzufügen oder einer Variable diese Zahl als Wert zuzuweisen. Dieses Programm weist der Variablen „**Alter“** die Zahl 12 zu:

Ist die Summe zweier Zahlen - Einfache Rechnungen

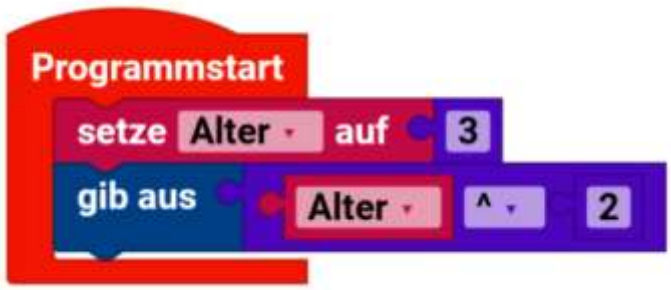
Dieser Block hat die Struktur:

Wert-Operator-Wert.

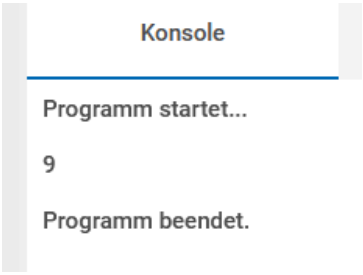
Als Operatoren stehen die Rechenarten
+ (Plus),
- (Minus),
× (Mal),
÷ (Geteilt), und
^ (hoch) zu Verfügung.

Der Operator kann über das Dropdown-Menü ausgewählt werden. Er kann unmittelbar auf Zahlen oder auch auf Werte von Variablen angewendet werden. Beispiel:

Dieser Block gibt Ergebnis 144 (12^2) aus.



Dieses Programm setzt die Variable „Alter“ auf 3. Dann wird 3 hoch 2 gerechnet. Das Ergebnis wird auf der Konsole ausgegeben.



Operatoren auf Bitebene - Bitweise Operatoren, Vergleiche auf Bitebene

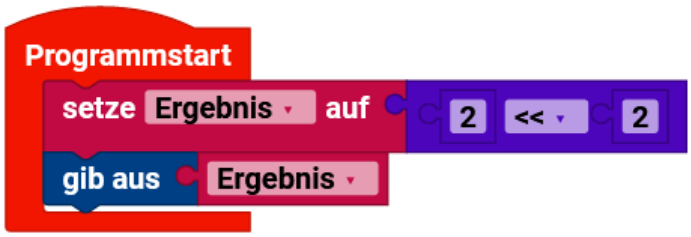
Die „Bitweisen Operatoren“ sind für Ganzzahlen, die in das Binärformat umgewandelt werden und dann auf Bit Ebene je nach Operator „behandelt“ werden.
In Robo Pro Coding sind das:

Shift links – Bitzahl nach links schieben
Shift rechts – Bitzahl nach rechts schieben
AND – Zwei Bitzahlen mit Logisch UND vergleichen
OR – Zwei Bitzahlen mit Logisch ODER vergleichen
XOR – Zwei Bitzahlen mit Logisch EXKLUSIVODER vergleichen
NOT – Zwei Bitzahlen mit Logisch NICHT vergleichen

Das Ergebnis (Zahl) wird nach links übergeben. Aus dem Zahlenbereich rausfallende Einsen oder Nullen verschwinden. Aufgefüllt wird immer mit Nullen.

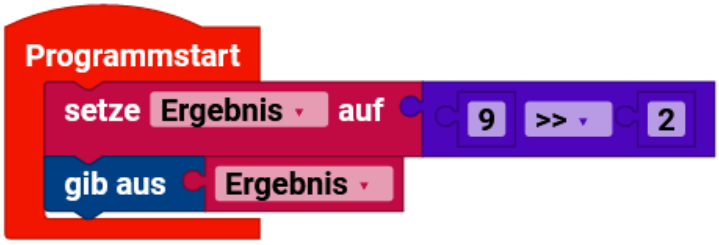
Shift

Shift-Beispiele, Lauffähige Programme, die das Ergebnis auf der Konsole ausgeben**:**



2 << 2 = 8

Die Zahl 2 (0010) wird zweimal nach links verschoben. Das ergibt 1000 und entspricht der Zahl 8.

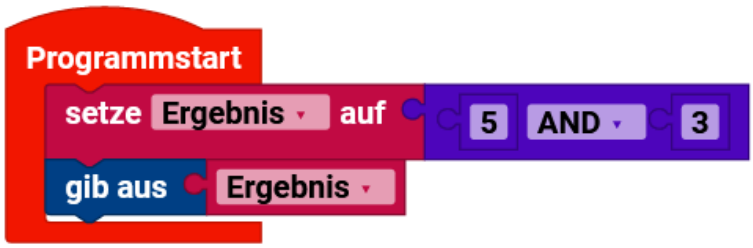


9 >> 2 = 2

Die Zahl 9 (1001) wird zweimal nach rechts verschoben. Das Ergibt 0010 und entspricht der Zahl 2.
Die linke Zahl ist diejenige die verschoben werden soll, rechte Zahl ist die Anzahl der Verschiebungen.

Logische-Beispiele, Lauffähige Programme, die das Ergebnis auf der Konsole ausgeben**:**

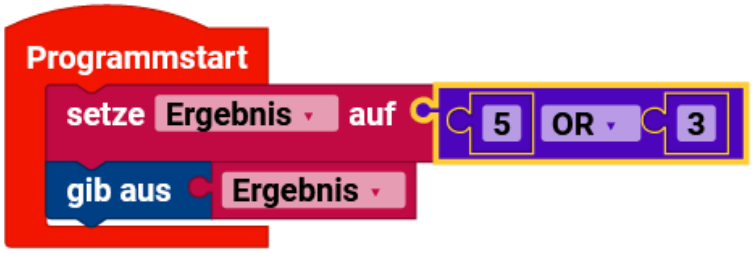
AND



5 AND 3 = 1

Die Zahl 5 (0101) mit Logisch UND 3 (0011) ergibt 0001, dass der Zahl 1 entspricht. Nur die letzte Stelle, der Bitzahlen, hat bei beiden Zahlen eine 1. Ansonsten ist es 0-0. 1-0 oder 0-1 was eine 0 ergibt. Die letzte Stelle hat 1-1 und somit eine 1.

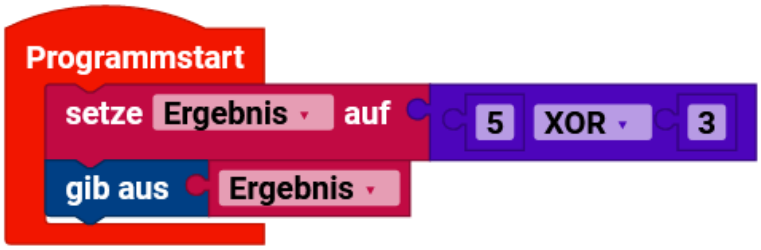
OR



5 OR 3 = 7

Die Zahl 5 (0101) mit Logisch ODER 3 (0011) ergibt 0111, dass der Zahl 7 entspricht. Im Ergebnis ist entweder die Eins der ersten oder die der zweiten Zahl.

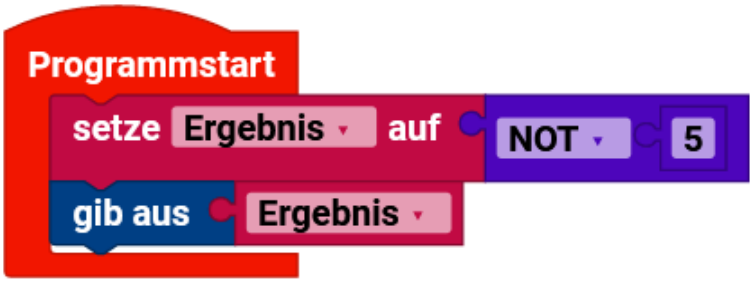
XOR



5 XOR 3 = 6

Die Zahl 5 (0101) mit Logisch Exklusiv ODER 3 (0011) ergibt 0110, dass der Zahl 6 entspricht. 0-0 und 1-1 ergeben 0. Wobei 0-1 und 1-0 eine 1 ergeben.

NOT



NOT 5 = -6

Die Bitweise Negation der Zahl = Einerkomplement = -(x+1).

Spezielle Rechnungen

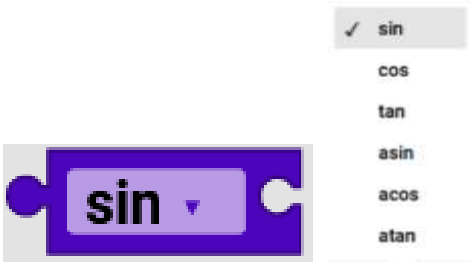
Dieser Block verwendet die, über das Dropdown-Menü ausgewählte Rechenart auf die dahinter platzierte Zahl oder auf den Wert der dahinter platzierten Variablen an. Die zur Verfügung stehenden Rechenoperationen sind:

- Quadrieren,
- Quadratwurzel,
- Betrag,
- Vorzeichenwechsel (Multiplikation mit -1).
- natürlicher Logarithmus,
- dekadischer Logarithmus,
- Exponentialfunktion mit der Basis e (e^1, e^2,...),
- Exponentialfunktion mit der Basis 10 (10^1, 10^2,...)

e ist hierbei die Euler'sche Zahl.

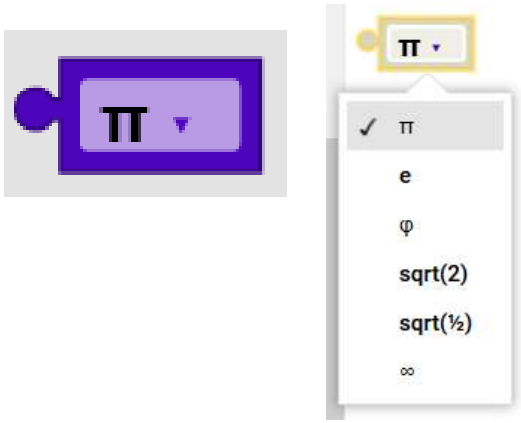
Dieser Block quadriert 16 und setzt die Variable „**i**“ auf das Ergebnis.

sin Block - Trigonometrische Funktionen



Dieser Block funktioniert ähnlich wie der zuvor beschriebene Block, mit dem Unterschied, dass als Rechenoperationen die trigonometrischen Funktionen Sinus, Cosinus, Tangens und ihre Umkehrfunktionen Arkussinus, Arkuscosinus und Arkustangens genutzt werden. Die angegebene Zahl oder der Wert der angegebenen Variablen wird also in die im Dropdown-Menü gewählte Funktion eingesetzt und das Ergebnis kann dann im Programm weiterverarbeitet werden.

Häufig verwendete Konstanten



Dieser Block funktioniert genauso wie der Zahl-Block, jedoch gibt man hier den Zahlenwert nicht selber an. Stattdessen sind es häufig verwendete Konstanten:

π = Pi = 3,141…
e = Euler'sche Zahl = 2,718…
Phi = 1,618…
sqrt(2) = Quadratwurzel von Quadratwurzel von ½ = 0,707…∞ = Unendlich

Die Konstante kann über das Dropdown-Menü ausgewählt werden.

ist gerade, ungerade … Block

Der Block überprüft welche Eigenschaft eine Zahl hat oder nicht hat. Dabei kann man die zu untersuchende Eigenschaft auswählen:

Gerade
Ungerade
eine Priemzahl
eine ganze Zahl
positiv
negativ
teilbar durch
keine Nummer

Der Block gibt wahr oder falsch zurück,

konvertiere zu Block

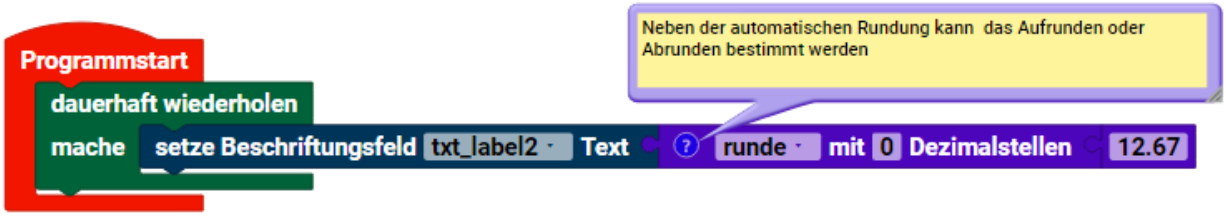
Konvertiert eine Zahl zur angegebenen Basis.

Runden

Mit dem **runde ...**- Block lässt sich eine angegebene Dezimalzahl oder der Wert einer angegebenen Variablen auf eine ganze Zahl runden. Dabei kann man im Dropdown-Menü drei Optionen wählen:

- mit "runde" kaufmännisch gerundet (z.B. wird 4,5 zu 5)
- mit "runde auf" wird aufgerundet (z.B. wird 5,1 zu 6)
- mit "runde ab" wird abgerundet (z.B. wird 5,9 zu 5).

Beispiel:

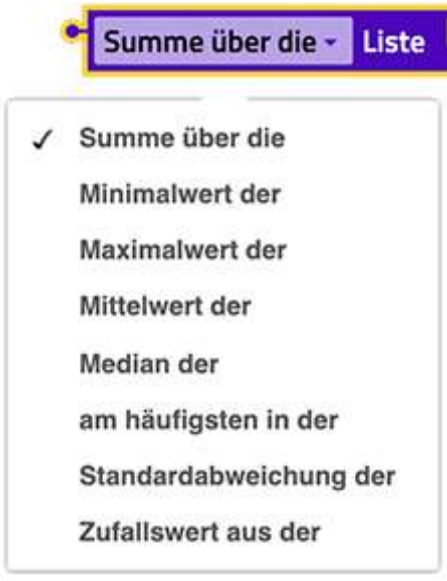


Summe über die Liste - Auswertung von Listen

Mit dem **... der Liste**-Block kann man sich

- mit "Summe" die Summe aller Werte einer Liste,
- mit "min" den kleinsten Wert einer Liste,
- mit "max" den größten Wert einer Liste,
- mit "Mittelwert" den Mittelwert aller Werte einer Liste,
- mit "Median" den Median einer Liste,
- mit "Modalwert" den/die am häufigsten vorkommenden Werte einer Liste,
- mit "Standardabweichung" die Standardabweichung aller Werte einer Liste,
- mit "Zufallswert" einen zufälligen Wert aus einer Liste

ausgeben lassen. Alle diese Optionen können über das Dropdown-Menü des Blocks ausgewählt werden:



Beispiel:

Hier wird der Maximalwert einer Liste auf der Konsole ausgegeben.

Rest von einer Division Block

Der **Rest von ...**-Block wird genutzt, um den Rest einer Division auszugeben. Dieses Programm weist der Variablen „**Rest“** den Rest der Division von 3:2, also 1, zu:

Beispiel:

In einem Beschriftungsfeld wird der Rest der Division 6/4 angezeigt.

begrenze ... zwischen ... und Block Eingabewerte einschränken

Der **begrenze ... zwischen ... und ...**-Block erlaubt es, Eingabewerte auf ein bestimmtes Intervall zu beschränken. Bevor ein Eingabewert weiterverarbeitet wird, wird getestet, ob er im festgelegten Intervall liegt. Es gibt drei Optionen, wie mit einem eingeben Wert verfahren wird:

- Der Wert liegt im Intervall, also wird er unverändert weitergegeben.
- Der Wert liegt unter der unteren Grenze des Intervalls, also wird diese untere Grenze weitergegeben.
- Der Wert liegt über der oberen Grenze des Intervalls, also wird diese obere Grenze weitergegeben.

In diesem Beispiel wird der Block genutzt, um den Wert der Variable **Geschwindigkeit** auf die vom Motor unterstützten Drehzahlen einzuschränken:

Wenn man z.B. bei einer vorherigen Rechnung für die Geschwindigkeit 1230 herausbekommen würde, wird der Wert von Geschwindigkeit auf 512 geändert. Ein Wert von 341 würde nicht geändert werden.

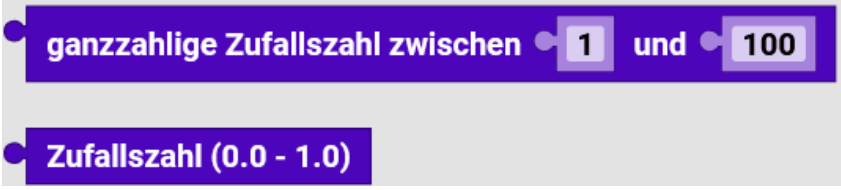
Beispiel:

Der Schieberegler liefert Werte von 0 bis 100, es werden aber die Werte zwischen 20 und 80 genutzt. Die Werte werden auf einem Beschriftungsfeld ausgegeben.

ganzzahlige Zufallszahl zwischen …

Zufallszahl(0.0-1.0)

Zufällige Werte generieren



Die beiden Blöcke **ganzzahlige Zufallszahl zwischen …** und **Zufallszahl(0.0-1.0)** geben einen zufälligen Wert aus. Dabei gibt der **ganzzahlige Zufallszahl zwischen … Block** eine Zahl aus dem definierten Intervall aus. Der Block **Zufallszahl(0.0-1.0)** gibt hingegen einen Wert zwischen 0,0 (eingeschlossen) und 1,0(ausgeschlossen) aus.

Beispiel:

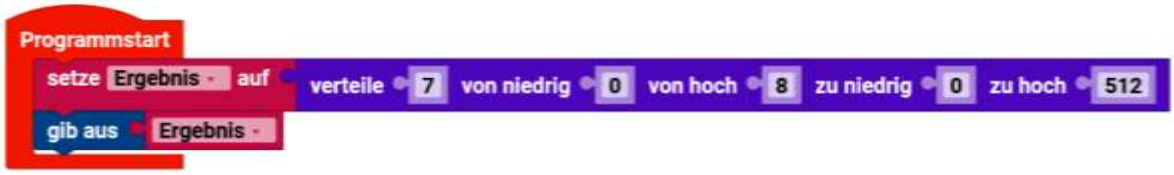
Wenn man zwei Beschriftungsfelder erzeugt hat, kann man dort die Zufallszahlen ausgeben.

arctan2 von X: ... Y

Zusätzlich zum Sinus Block, gibt es noch den Block **arctan2 von X: ... Y: ...**, der es erlaubt, sich mit Hilfe von zwei reellen Zahlen (einzusetzen als X und Y) einen Funktionswert des arctan2 im Bereich von 360° ausgeben zu lassen.

verteile von niedrig von hoch… Block

Dieser Block wirkt wie eine Veränderung des Maßstabes einer Strecke. Frühere fischertechnik Interfaces hatten nur Geschwindigkeiten von 0-8. Die neueren Controller haben eine feinere Einteilung von 0-512. Um z.B. diese Geschwindigkeit von 7 auf einen Wert von 0-512 umzurechnen, muss man als ersten Wert 7 eingeben.



Wenn man das Programm startet, kommt als Ergebnis 448 heraus. Die Geschwindigkeit 7 von früher, entspricht der heutigen von 448.

Beispiel:

Die Werte vom Schieberegler 0-100 werden als Werte zwischen 0 und 20 ausgegeben,

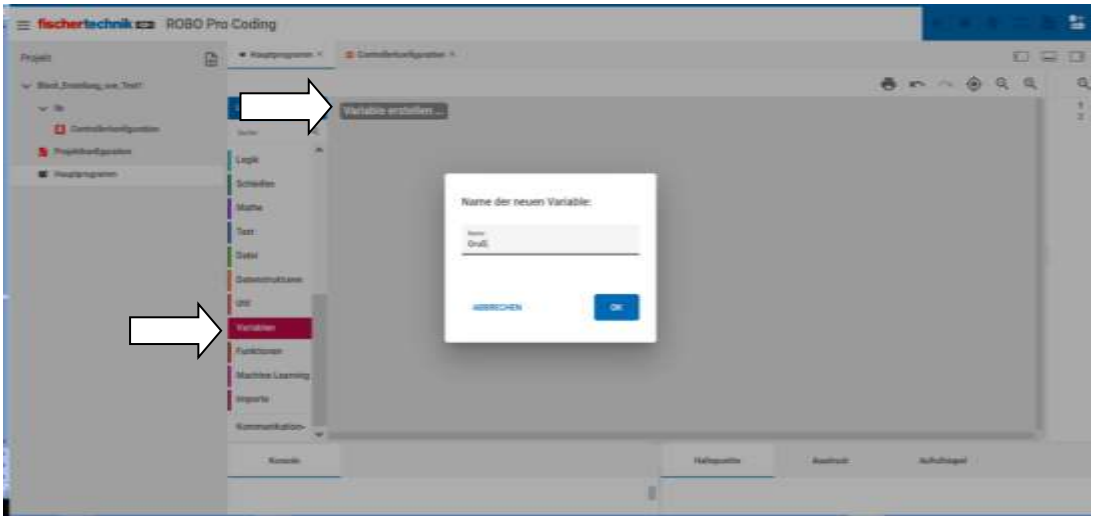
Text

Beispiele für Texte sind:

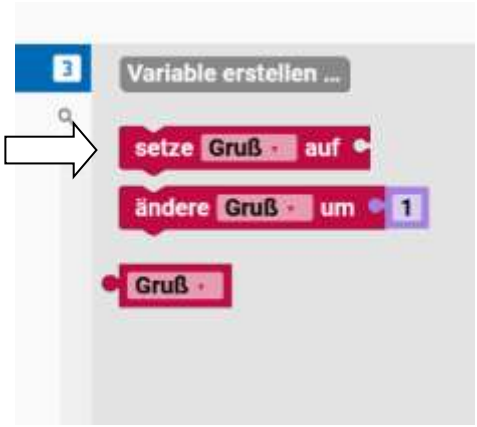
"Ding 1"
"12. März 2024"
"" (leerer Text)
Text kann Buchstaben (klein oder groß geschrieben), Zahlen, Satzzeichen, andere Symbole und Leerzeichen enthalten.

Erstellung von Text

Der folgende Block erzeugt den Text "Hallo" und speichert ihn in der Variablen namens **Gruß:**



Variablen klicken– Variablen erstellen klicken. Name der neuen Variablen: „Gruß“



Es erscheinen die Blöcke für die Variable Gruß. Wir klicken auf den Block „setze…auf“.

Und ziehen ihn in das Programm.

Ein Buchstabe, Text oder Satz

Dann auf den Text-Reiter klicken und den obersten Block „Ein Buchstabe, Text oder Satz“ klicken und in das Programm ziehen.

Nun mit der linken Maustaste hier klicken und

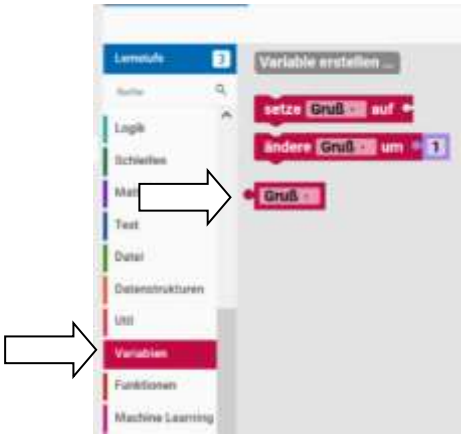
„Hallo“ eingeben und auf OK klicken.

So sollte es nun aussehen. Wenn wir das Programm laufen lassen, sieht man nichts. Wir lassen uns deswegen den Inhalt von Gruß mal ausgeben.

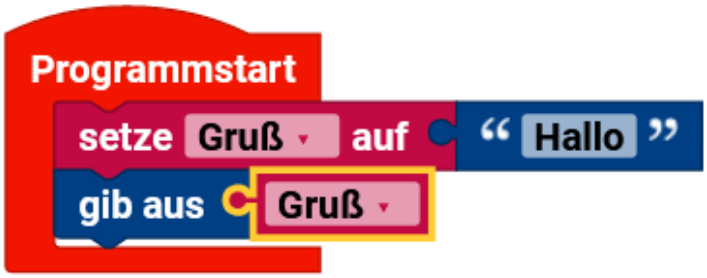
gib aus

Dazu nehmen wir den Block „gib aus“ Das ist der zweite Block beim Reiter Text

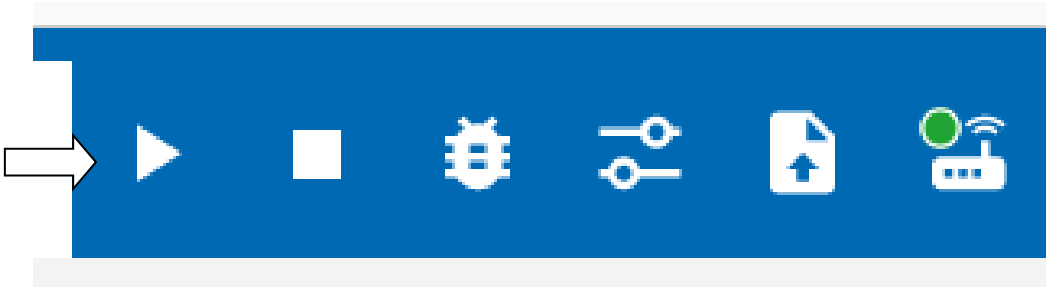
und ziehen ihn in das Programm.



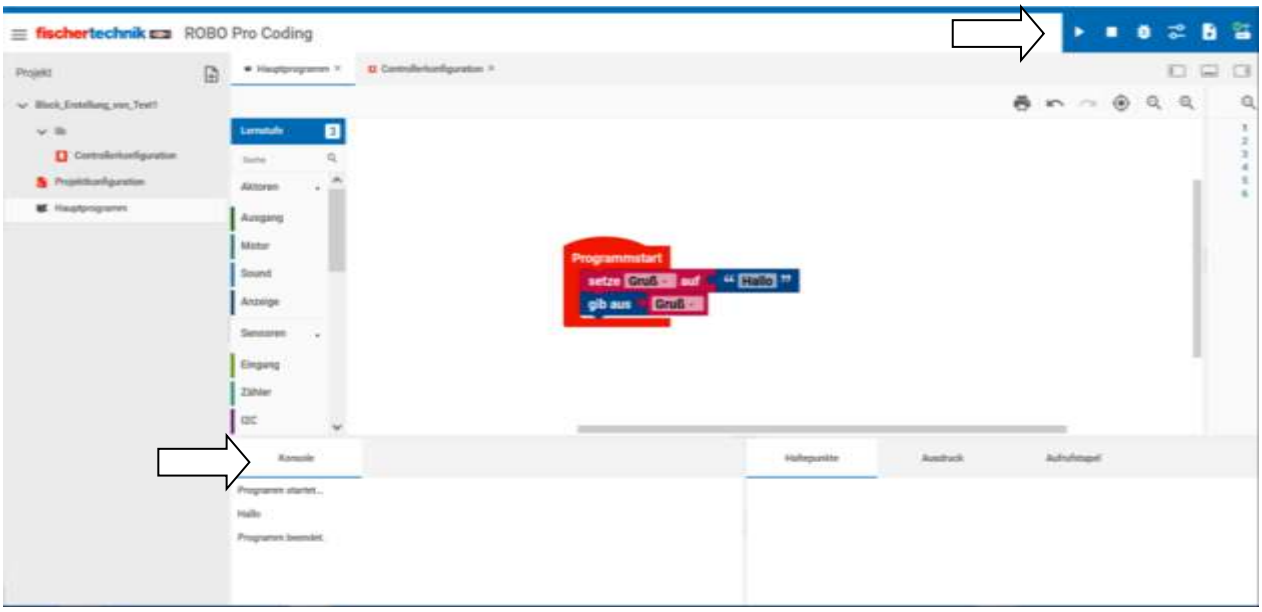
Nun klicken wir auf Variablen und ziehen den unteren Block hinter den „gib aus“ Block.



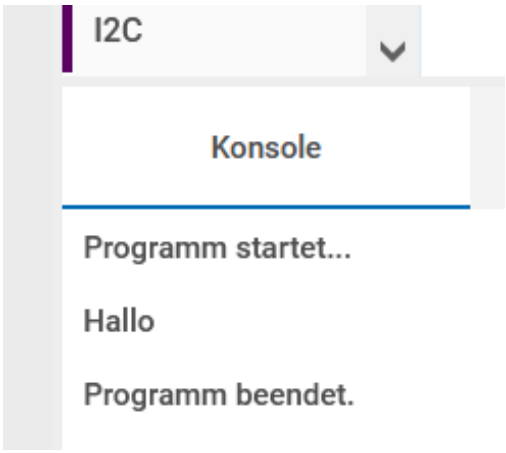
Wir lassen das Programm mal laufen, indem wir auf „Programm starten“ oben rechts im blauen Feld klicken.



Programm starten



Unten in der Konsole wird die Ausgabe des Programms angezeigt.



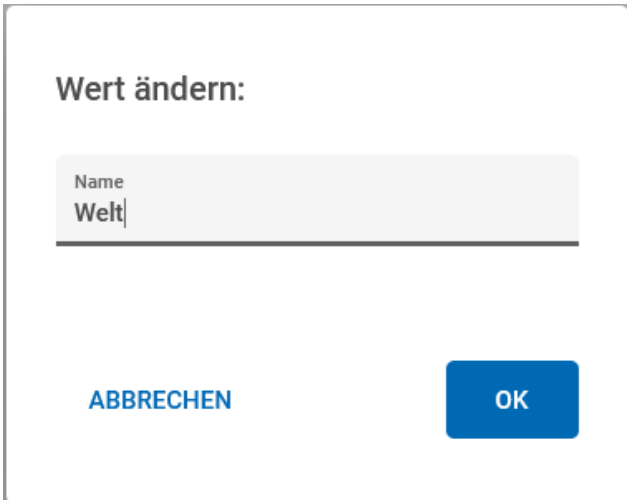
Das Programm ist gestartet, hat „Hallo“ ausgegeben und ist wieder beendet worden.

erstelle Text aus

Der Block **erstelle Text aus** kombiniert den Wert der Variable **Gruß** und den neuen Text "Welt" zu dem Text "Hallo Welt". Beachte, dass zwischen beiden Texten kein Leerzeichen steht, da in den beiden ursprünglichen Texten keines vorhanden war.

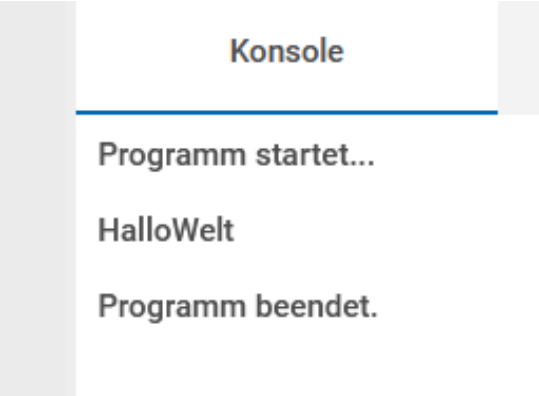
Um die Anzahl der Texteingaben zu erhöhen, klicke auf das (+) Symbol. Um die letzte Ausgabe zu entfernen, klicke auf das (-) Symbol.

Ändern wir das Programm, indem wir den „erstelle Text aus“ Block und den Block „Ein Buchstabe, Text oder Satz“ aus dem Text Reiter in das Programm ziehen.



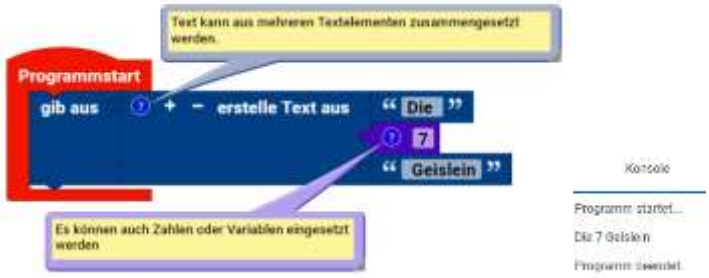
Ändern wir den Block unten rechts in „Welt“.

Wenn wir das Programm starten,



Kommt in der Konsole „HalloWelt“ ohne Leerzeichen.

Beispiel:



zu Text anhängen

Änderung von Text

Der Block **zu Text ... anhängen** fügt den angegebenen Text an die angegebene Variable an. In diesem Beispiel ändert er den Wert der Variable **Gruß** von "Hallo" in "Hallo, da!":

Beispiel:

Nach dem Starten wird die Variable „Element“ mit dem Inhalt „Hello World“ gefüllt. Zu dem Inhalt, wird „alles OK.“ angehängt. Anschließend wird der neue Inhalt auf der Konsole ausgegeben.

Länge von

Textlänge

Der **Länge von**-Blockzählt die Anzahl der Zeichen (Buchstaben, Zahlen usw.), die in einem Text enthalten sind. Die Länge von "Wir sind #1!" ist 12, und die Länge des leeren Textes ist 0.

Beispiel:

Es werden auch Leerzeichen und Satzzeichen gezählt.

ist leer

Prüfen auf leeren Text

Der Baustein **ist leer** prüft, ob der angegebene Text leer ist (die Länge 0 hat). Das Ergebnis ist im ersten Beispiel **wahr** und im zweiten Beispiel **falsch**.

im Text suche Auftreten des Begriffs

Suchen von Text

Diese Blöcke können verwendet werden, um zu prüfen, ob ein Text in einem anderen Text vorkommt und wenn ja, wo er vorkommt. Zum Beispiel wird hier nach dem ersten Vorkommen von "a" in "Hallo" gefragt, das Ergebnis ist 2:

Dies fragt nach dem letzten Vorkommen von "a" in "Hallo", was ebenfalls 2 ergibt:

Unabhängig davon, ob das erste oder letzte Vorkommen ausgewählt wird, liefert dieser Block das Ergebnis 0, da "Hallo" kein "z" enthält.

Beispiel:

Ab dem 9 Zeichen, ist der Begriff „der“.

Extrahieren von Text

im Text Buchstabe
Extrahieren eines einzelnen Zeichens

Dies ergibt "b", den zweiten Buchstaben in "abcde":

Beispiel:

Dies liefert "d", den vorletzten Buchstaben in "abcde":

Dies liefert "a", den ersten Buchstaben in "abcde":

Dies erhält "e", den letzten Buchstaben in "abcde":

Dies erhält jeden der 5 Buchstaben in "abcde" mit gleicher Wahrscheinlichkeit:

Keiner von ihnen verändert den Text, aus dem extrahiert wird.

Extrahieren eines Textbereichs

im Text … Buchstaben

Mit dem **im Text ... Buchstaben** -Block kann ein Textbereich extrahiert werden, der entweder mit:

- Buchstabe
- Buchstabe vom Ende
- erster Buchstabe

startet und mit:

- Buchstabe
- Buchstabe vom Ende
- letzter Buchstabe

endet.

Im folgenden Beispiel wird "abc" extrahiert:

Wie mit einer Schere wird der Text von 5 bis 7 ausgeschnitten.

wandel um in …

Anpassen der Groß-/Kleinschreibung des Textes

Dieser Block erzeugt eine Version des Eingabetextes, die entweder in

- GROßSCHREIBUNG (alle Buchstaben in Großbuchstaben) oder
- kleinschreibung (alle Buchstaben sind Kleinbuchstaben) oder
- Substantive (erste Buchstaben Großbuchstaben, andere Buchstaben Kleinbuchstaben).

Das Ergebnis des folgenden Blocks ist "HALLO":

Nicht-alphabetische Zeichen sind davon nicht betroffen. Beachte, dass dieser Block auf Text in Sprachen ohne Groß- und Kleinschreibung, wie z. B. Chinesisch, nicht wirkt.

Beispiel:

Umwandlung der Kleinbuchstaben in Großbuchstaben.

entferne von Leerzeichen vom…

Trimmen (Entfernen) von Leerzeichen

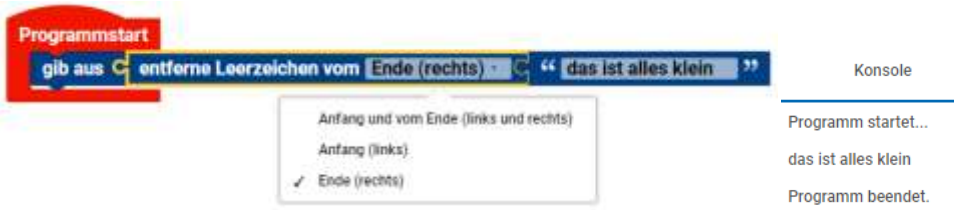
Der folgende Block entfernt, je nachdem, was im Dropdown-Menü (kleines Dreieck) eingestellt wird, Leerzeichen:

- am Anfang des Textes
- am Ende des Textes
- an beiden Seiten des Textes

Das Ergebnis des folgenden Blocks mit „ Hi du “ ist "Hi du".

Leerzeichen in der Mitte des Textes sind nicht betroffen.

Beispiel:



Das Leerzeichen am Ende wird entfernt.

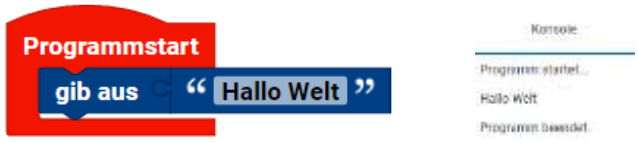
gib aus

Text ausgeben

Der **gib aus**-Block bewirkt, dass der Eingabewert im **Konsole-Fenster** ausgegeben wird:

Auf keinen Fall wird er an den Drucker geschickt, wie der Name vielleicht vermuten lässt.

Beispiel:



Nach dem Starten, gibt das Programm „Halle Welt“ auf der Konsole aus.

Schriftgröße ändern

Wie bei der Syntax in HTML, können Texte formatiert ausgegeben werden. Die Formatierung kann entsprechend der Anwendung in der Anzeigekonfiguration oder im Programm bei der Textausgabe durchgeführt werden. Die Größe wird mit <h1> bis <h6> eingeleitet und mit </h1> bis </h6> abgeschlossen.

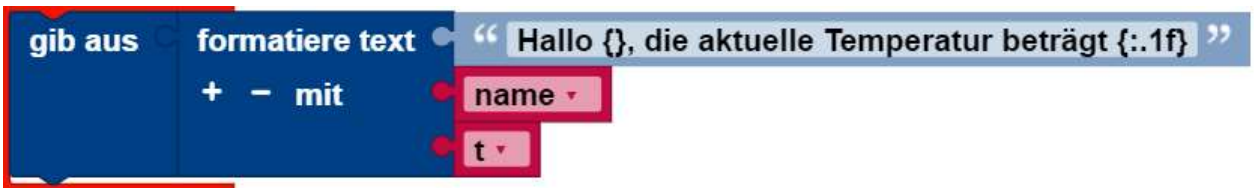
Weitere Formatierungen wären fett mit <b>, kursiv mit <i>und unterstrichen mit<u>.

Die Größe entspricht der Hierarchie der Überschriften.

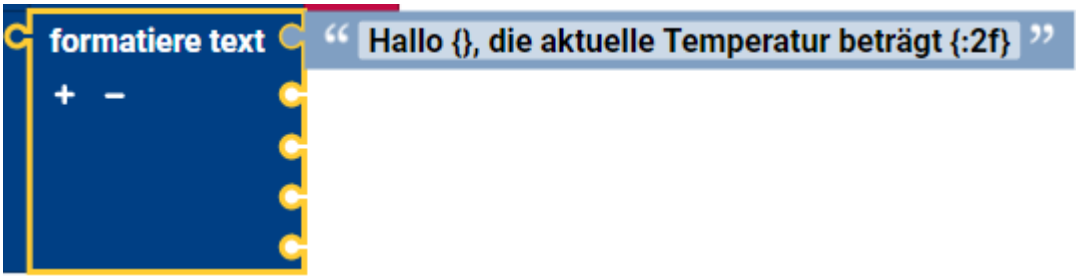
formatiere text …

Text ausgeben mit Formatierung

Mit dem **formatiere text**-Block können Textausgaben mit Variableninhalt formatiert ausgegeben werden. Dabei werden alle Platzhalter **{}** im Text durch den Inhalt der nach dem Text angehängten Variablen ersetzt. In den geschweiften Klammern kann eine Formatierung angegeben werden. Die Formatierung **{:.1f}** gibt z.B. nur die erste Nachkommastelle der Kommazahl in der Variablen **t** aus.



formatiere Text, weitere Erklärungen



Formatierungen werden für Ausgaben genutzt.
Das können

Listen von Datenstrukturen,
Ausgaben auf dem Bedienfeld,
Print-Befehl in die Konsole oder
Daten die in eine Datei geschrieben werden sein.

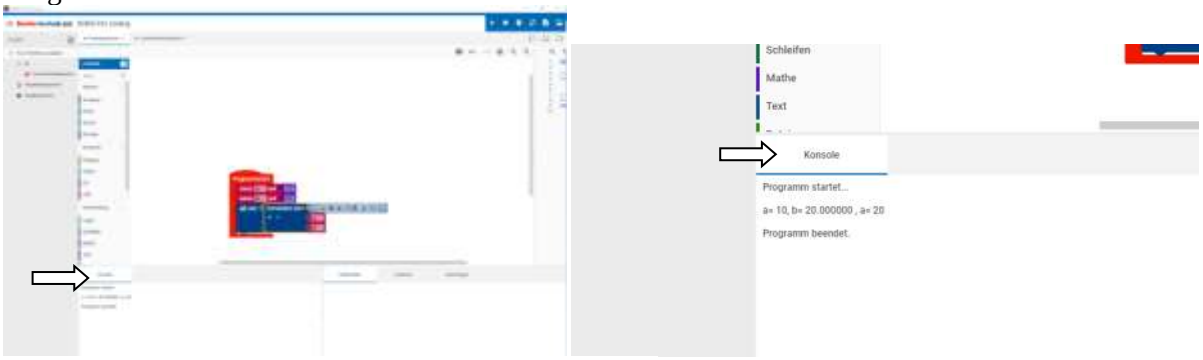
Über + kann man noch mehr Variablen andocken lassen. Über - wird der letzte Andockpunkt gelöscht.

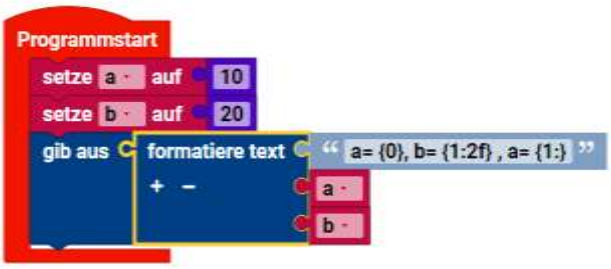
Formatieren von Text funktioniert auf zwei Arten:

1. Automatische Variablennummerierung {} leeres Feld oder {:2f} nach dem Doppelpunkt.
   Hier geht es in der Reihenfolge, wie die Reihenfolge der Variablen ist.

2. Manuelle Variablennummerierung {0} {1} {0:2f}
   Jede Variable, im unteren Teil des Blocks "formatiere Text", hat eine Nummer, von 0 = oberste beginnend, die in der nächsten Zeile 1 usw...

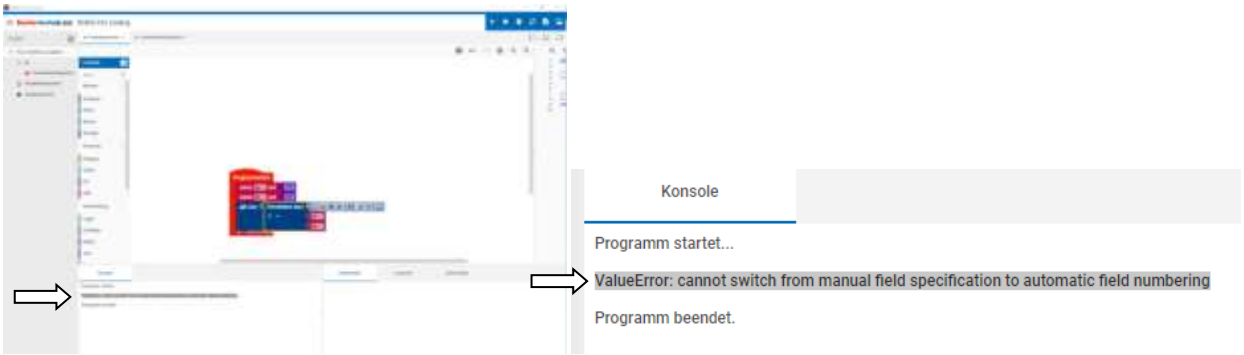
Ausgabe auf der Konsole



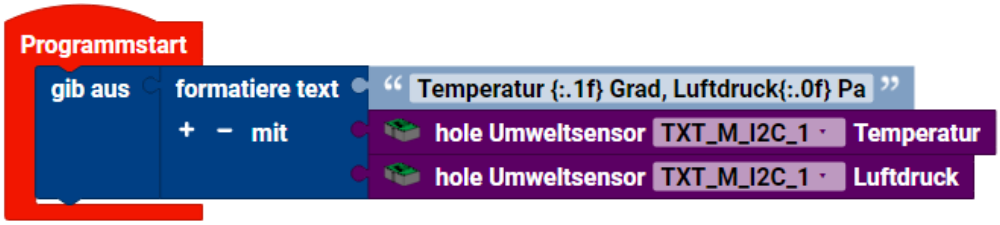


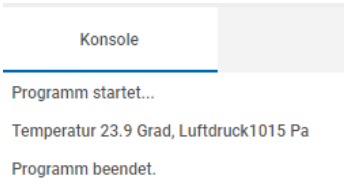
Hier sind zwei Variablen a und b. a ist die Nummer 0 und b die Nummer 1.

Die **Angabe im ersten Feld entscheidet** welche Bezeichnung alle anderen haben **-müssen-**. Man kann nicht mischen. Ansonsten kommt die Fehlermeldung "ValueError: cannot switch from manual field specification to automatic field numbering" in der Konsole.



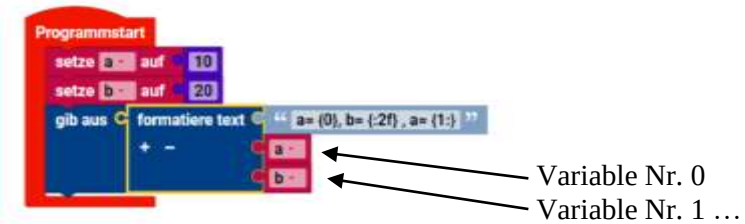
Beispiel:





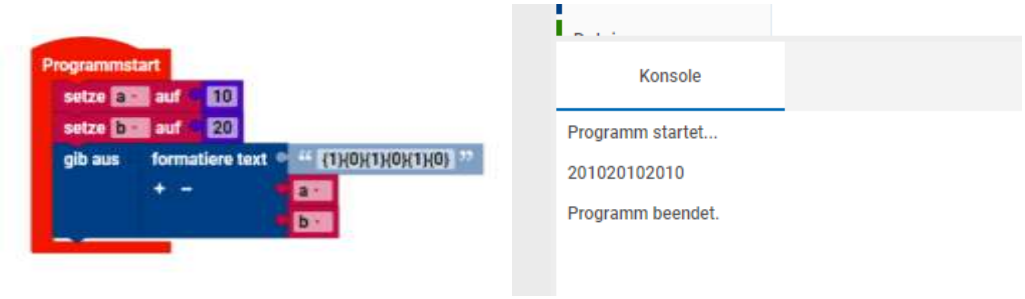
In diesem Beispiel werden die Daten vom Umweltsensor geholt. Es können aber auch Variablen oder feste Zahlenwerte sein.

Nummer der Variablen
Wichtig: Die Nummerierung beginnt mit 0. Eins ist die zweite Variable!
Die Nummerierung im Block ist von oben nach unten.



Man kann auch eine Mischung in einer anderen Reihenfolge oder Mehrfach der Variablen machen. Auch Text zwischen den Variablen ist erlaubt.

Beispiele (a=10 und b=20):
a= {0}, b= {1:2f} , b= {1:} Ergibt: "a= 10, b= 20.000000 , b= 20"
{}{} Ergibt: 1020
{1}{0} Ergibt: 2010
{1}{0}{1}{0}{1}{0} Ergibt: 201020102010



Programm und Ausgabe in der Konsole

Aufbau der Formatierung
Eventuell sind sie oben über das Ergebnis von b= {1:2f} mit b= 20.000000 überrascht gewesen. Das liegt am Aufbau der Formatierung.

{[Nummer der Variablen] : [Dezimalstellen] [.] [Nachkommastellen] [Zahlentyp]

Nummer der Variablen haben wir weiter oben schon kennengelernt. Beginnend mit 0 und dann fortlaufend 1, 2, 3...

: Der Doppelpunkt trennt die Nummer von der nachfolgenden Formatierung.

Dezimalstellen Anzahl der Vorkommastellen. Z.B. 2

. Punkt als Trenner zu den Nachkommastellen

Nachkommastellen Anzahl der Nachkommastellen Z.B. 3 (Rundung der folgenden Stellen beachten!)

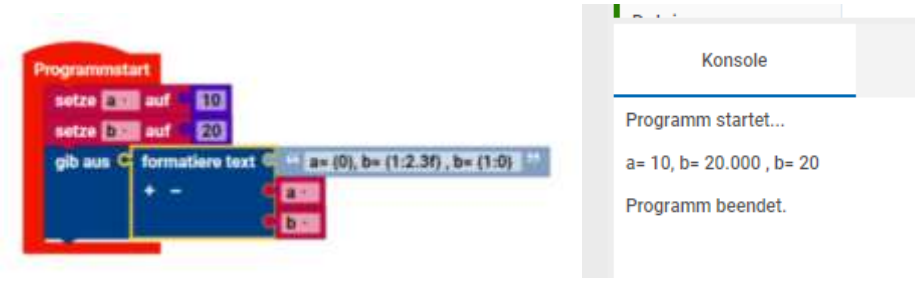
Somit ist die Ausgabe beim Beispiel wie oben:

Bei b= {1:2f} mit b= 20.000000 wird die Variable Nummer 1 mit zwei Vorkommastellen (2) und den unveränderten Nachkommastellen der Floatzahl (f), hier 6 Stellen ausgegeben.

Zahlentyp

Tipp: Zahlen Typen in Python (Auswahl) / Robo Pro Coding

none = Kein Zahlentyp (Standarteinstellung in Robo Pro Coding = kann reine Dezimalzahl oder Kommazahl sein.)
d = integer (Wird in Robo Pro Coding nicht gesondert eingestellt)
f = float
0x = hexadezimal (Bei der Eingabe von Hexadezimalzahlen in Variablen, werden diese sofort in Dezimalzahlen umgewandelt.)
0 = oktal (Führende Null wird von Robo Pro Coding ignoriert, nicht nutzbar)

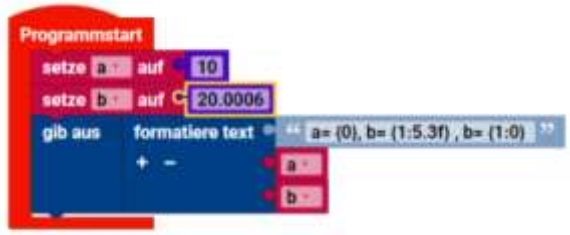


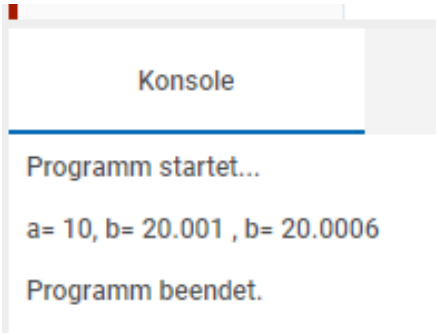
Hier sind bei b= {1:2.3f} die Ausgabe 20.000 . Also Variable Nummer 1, 2 Vorkommastellen und 3 Stellen der Floatzahl.

Anzahl von Vor- und Nachkommastellen

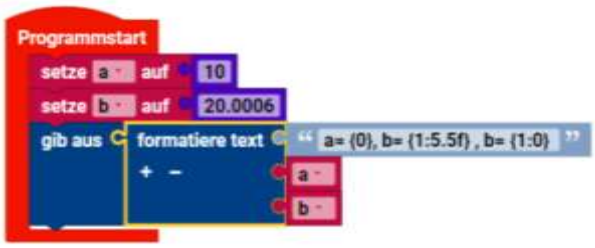
Anzahl von Vorkommastellen
Wird eine größere Anzahl an Vorkommastellen angegeben werden die fehlenden bei Python mit 0 aufgefüllt. Bei Robo Pro Coding werden diese ignoriert. Man muss also bei einer formatierten Ausgabe aufpassen, dass sich ein Text nicht verschiebt. Z.B. beim Bedienfeld.

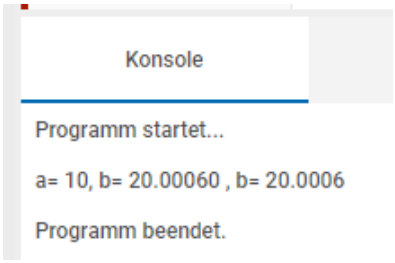
Anzahl von Nachkommastellen





Hier wird aus der 20,0006 aufgerundet eine 20,001 , da hier im Programm 3 Nachkommastellen angegeben sind.

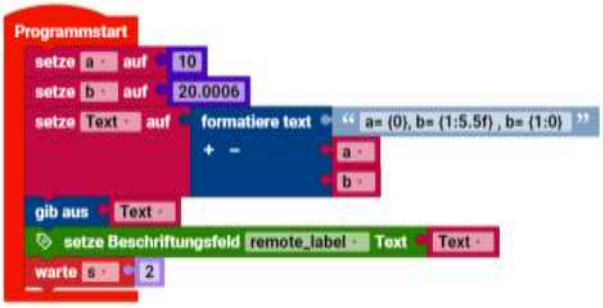




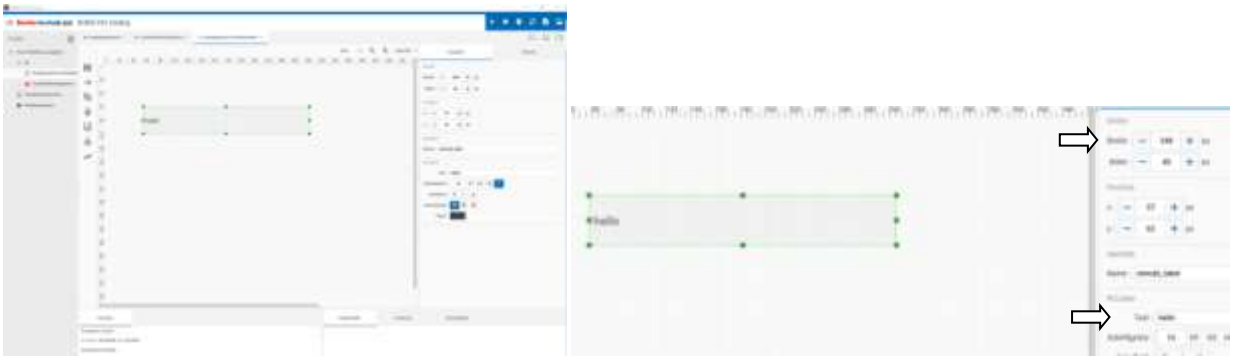
Hier wird aus der 20,0006 eine 20,00060 , da hier 5 Stellen angeben sind. Es werden also keine Nullen abgeschnitten, sondern eine angehängt.

Formatierungen von Texten, die in eine Datei geschrieben werden sollen sind unter „Datei“ zu finden.

Ausgabe auf dem Bedienfeld



Im Programm wird der formatierte Text in eine Variable mit Namen "Text" gespeichert und auf der Konsole und danach auf dem Bedienfeld ausgegeben.
Achtung! Die *Wartezeit* von 2 Sekunden *muss sein*, da sonst das Bedienfeld beendet wird, bevor man den Text lesen kann.



Beim Bedienfeld muss man darauf achten, dass das „RCLabel“ groß genug ist, um den Text aufnehmen zu können. Ansonsten wird nur ein Teil angezeigt und der Rest mit ... dargestellt. Hier wird „Hello“ belassen. Nach dem Starten vom Programm wird kurz „Hello“ angezeigt.

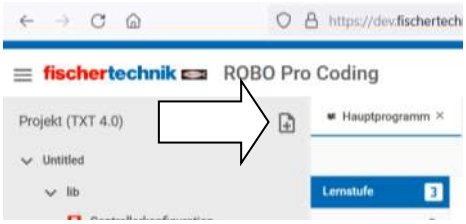
Dann wird der Inhalt der Variablen Text wird kurz auf dem Bedienfeld angezeigt und das Programm beendet.

Datei

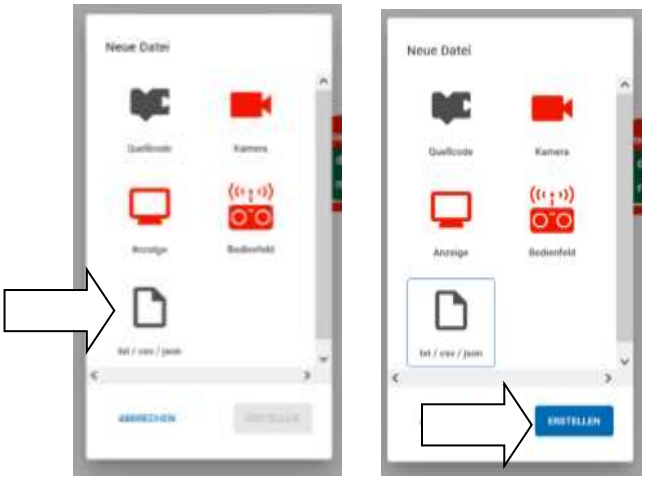
Die Blöcke im Reiter "Datei" können mit Dateien "umgehen". Diese Dateien können nur auf dem TXT Controller sein, nicht auf dem PC.
Der RX und der BT Smart Controller können diese Befehle nicht verarbeiten.

Datei … mit Modus… öffnen

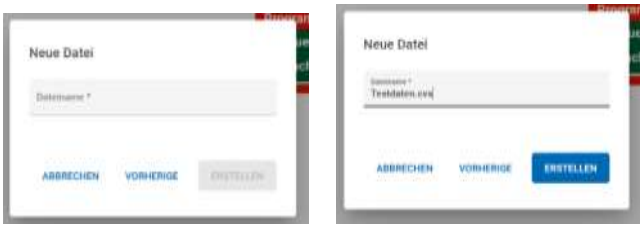
Dieser Befehl greift auf Dateien zu, die man mit dem Plussymbol (Neue Datei) links oben erzeugt.
Man kann damit auf selbst erstellte Textdateien zuzugreifen. Das können txt-Dateien für Text, cvs-Dateien meist für Zahlen oder Daten oder json-Dateien für Java sein.
Diese Daten können auch von anderen Programmen wie z.B. Excel eingelesen werden.



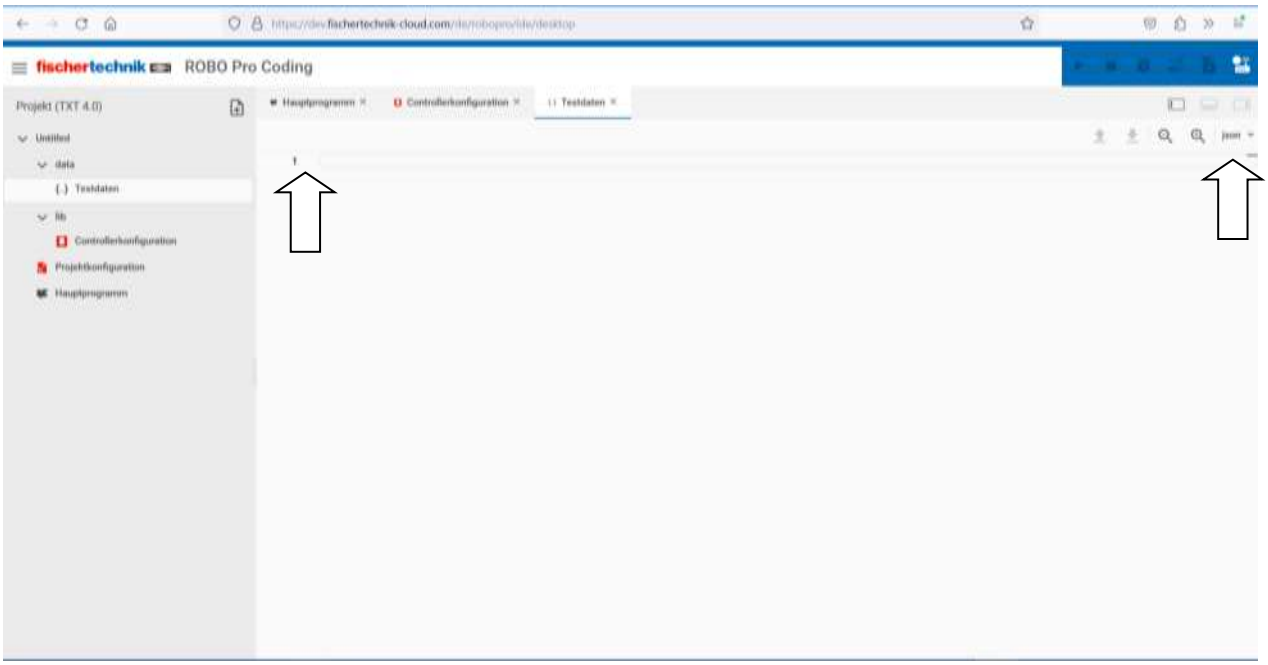
Hier auf das Plus (Neue Datei) klicken.



Anschließend auf txt/cvs/json und dann auf Erstellen.
Man hat hier die Möglichkeit die Endung festzulegen.

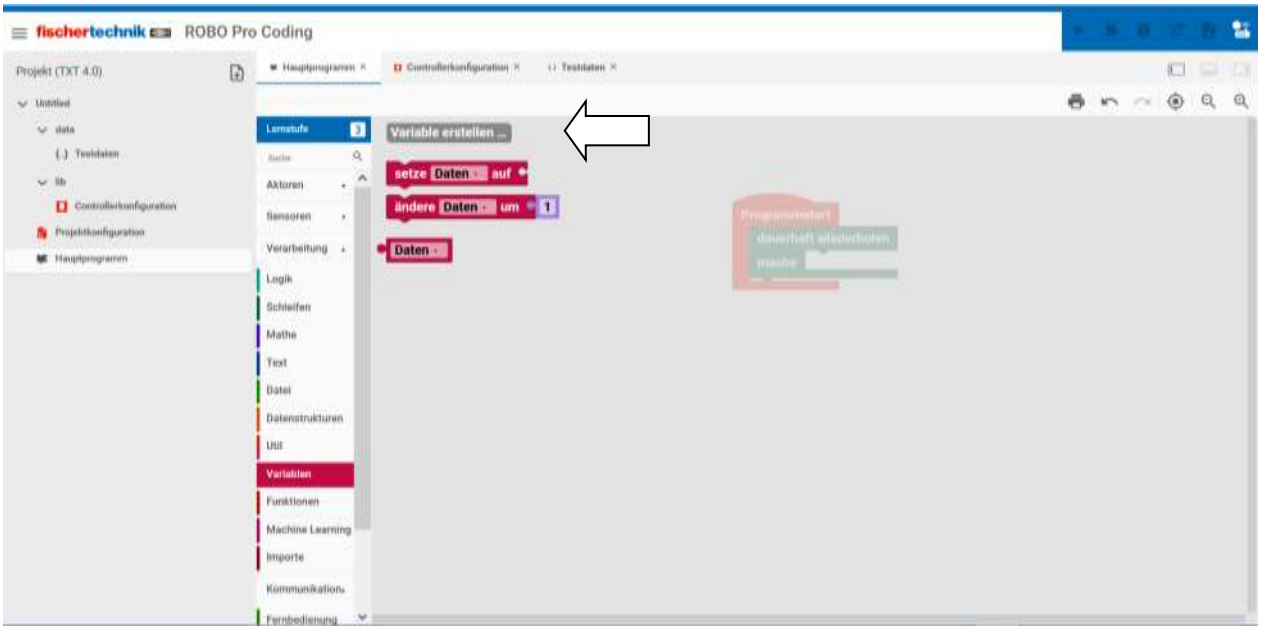


Nun muss ein Name festgelegt werden. Hier „Testdaten.cvs“ und Erstellen klicken.



Es öffnet sich eine leere Textdatei. Hier könnte man Daten… eintragen.
(Ganz rechts oben kann man die Endung auf „.cvs“ setzen.)

Im Hauptprogramm, wird dann z.B. mit einer Variablen auf die Datei zugegriffen.



Hier wird über „Variable erstellen“ eine mit dem Namen „Daten“ erzeugt.

Und dann mit „setze … auf“ in das Hauptprogramm eingefügt.

Dann wird über den Reiter „Datei“ der erste Block in das Hauptprogramm eingefügt.

Wenn man auf den kleinen Pfeil klickt, werden die möglichen Dateien angezeigt. Hier sieht man, dass obwohl wir vorher .cvs angegeben haben, hier diese durch .json ersetzt wurde. (Endung rechts oben)

Hinweis: Wenn man nun eine .cvs-Datei haben möchte, kann man diese Datei nur herunterladen und den Namen ändern. Bei Windows kann es sein, dass man erst die Möglichkeit zur Änderung der Dateinamenendung freischalten muss. In Excel kann man auch Dateien einlesen, die nicht die Endung cvs haben.

Mit Klick auf den Pfeil, kann man auswählen, ob man Daten lesen, schreiben oder anhängen möchte.

Datei path mit Modus… öffnen

Im Grunde derselbe Block wie oben, nur das hier die Variable „path“ steht. Die kann man aber auch überschreiben. Das ist der Zentrale Block, um eine Datei zu öffnen.

Man gibt dabei den Speicherort und den Modus an. Es wird in eine Variable geöffnet.
Bemerkung: Hier fehlt noch das Schließen der Datei (siehe weiter unten).

Der Speicherort und der Dateiname werden im ersten Feld angegeben. Dabei ist zu beachten, dass man auf dem TXT 4.0 von fischertechnik ein Arbeitsverzeichnis hat. Das dient auch dazu, zu verhindern, dass Dateien vom Betriebssystem überschrieben werden. Der Pfad ist z.B. „/opt/ft/workspaces/log.txt“. Die

Datei log.txt steht in der obersten Ebene. Man kann sie über SSH im Browser vom PC auslesen (siehe dazu weiter unten im Beispiel). Man kann auch in einen Projektordner auf dem TXT speichern. Dazu muss man dann den Pfad entsprechend ändern.
Wenn der Dateiname noch nicht existiert, wird die Datei automatisch erzeugt.

Als Modus kann man wählen:

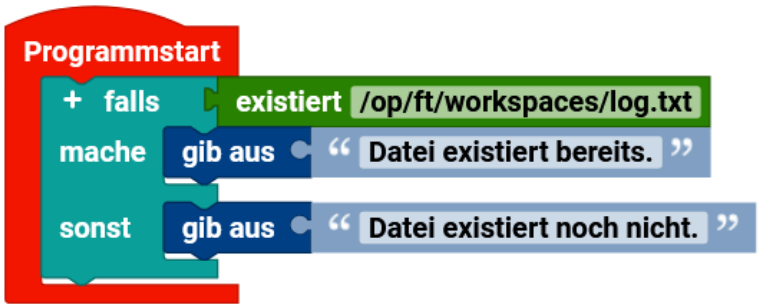
read = lesen
write = schreiben
append = anhängen

Eventuell hilft die Vorstellung von einem leeren Blatt Papier, wo man mit Kugelschreiber etwas darauf schreibt. Man kann auf dem Blatt etwas schreiben (=write), man kann das geschriebene lesen (=read) und man kann am Ende etwas dazuschreiben (=append).

Man kann aber die Zeilen einzeln abschneiden (=Liste), untereinanderlegen und auch dazwischen etwas einfügen. Diese Liste kann man dann wieder scannen und ausdrucken und man hätte ein neues Blatt Papier.

existiert …

Hiermit kann man Überprüfen, ob der angegebene Pfad oder Dateiname schon existiert.



Hier wird Überprüft, ob die Datei „log.txt“ bereits existiert oder nicht. Wenn ja wir der Text „Datei existiert bereits,“ auf der Konsole ausgegeben. Wenn nicht, wird der Text „Datei existiert noch nicht.“ Ausgegeben.

Man kann den Befehl dazu benutzen, Dateien nicht zu überschreiben, um damit keine Daten zu verlieren.

Der Block funktioniert auch mit Verzeichnissen.

Datei… als Text lesen

„Datei als Text lesen“ ist eine der Möglichkeiten an den Inhalt der Datei zu kommen. Da der Inhalt in eine Variable eingelesen wird, kann man mit den Text-Blöcken diese bearbeiten.

Datei … zeilenweise als Liste lesen

Hier wird der Inhalt der Datei in eine Liste eingelesen. Man kann diesen Inhalt dann mit den Listen-Blöcken bearbeiten. Damit kann man auch Sachen dazwischen einfügen.

schreibe… in Datei… neue Zeile am Ende

Bei diesem Block wird an den vorhandenen Inhalt, am Ende etwas hinzugefügt. Mit dem Hacken am Ende, kann man ein „neue Zeile“-Zeichen („Enter“) anfügen. Das kann man auch als Trennzeichen benutzen, wenn man z.B. cvs-Dateien damit macht, um Messwerte später mit Excel auszuwerten. Wenn man nur Text anhängen möchte, kann man auch den Hacken deaktivieren.

Liste… zeilenweise in die Datei … schreiben

Der Inhalt einer Liste, wird hier in eine Datei geschrieben. Zusätzlich wird an jeden Eintrag am Ende ein „neue Zeile“-Zeichen („Enter“) anfügt. Somit ist es einfacher die Datei später wieder als Liste einzulesen.

Änderungen in Datei … übernehmen

Ich vermute, dass bei diesem Block eine veränderte Datei endgültig geändert wird und vorher nur eine Kopie der Datei bearbeitet wurde.

Datei … schließen

Wichtiger Block. Man muss *immer* eine geöffnete Datei am Ende wieder schließen, da sonst Daten verloren gehen können.

Verzeichnis path erstellen

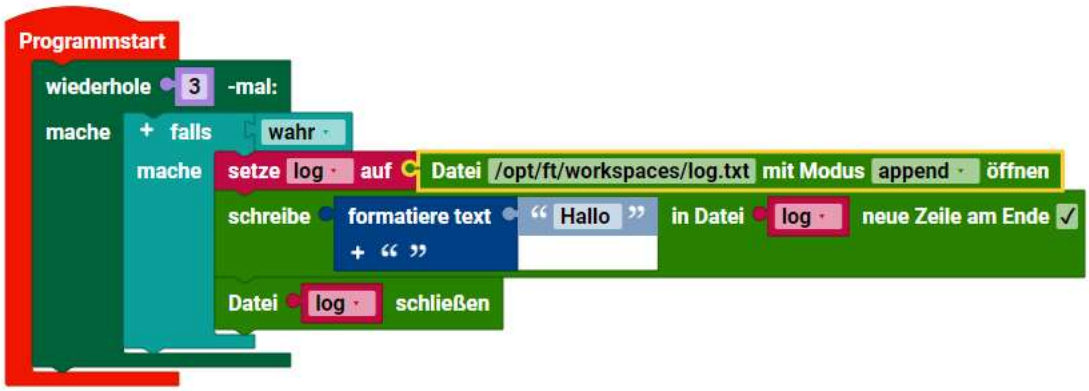
Hiermit kann man ein neues Verzeichnis erstellen, um z.B. Messwerte nicht in das Programmverzeichnis speichern zu müssen. Es wird hierbei die Variable „path“ genommen, Diese kann aber überschrieben werden.

Beispielprogramme

Etwas in eine Datei schreiben.

In diesem Beispiel wir das Wort „Test“ in eine Datei geschrieben. Nur einmal, auch wenn man das Programm mehrfach aufruft, steht in der Datei log.txt nur einmal das Wort Test.

Programm „3xHello“
Hier mal ein Beispiel um etwas, hier “Hallo“, an eine Datei zu anzuhängen:



Hier wird in die Datei "log.txt" der Text "Hallo" 3x reingeschrieben (angehängt).

Der Block "falls mache" mit dem wahr dient zum an und ausschalten der Funktion. Damit kann man beim Testen eines Programmes das Schreiben abstellen, um keine Endlosdatei zu machen. (Kann man auch weglassen.)

Es gibt eine Variable "log". In diese wird der Pfad mit dem realen Namen der Datei angegeben. Bei Modus geben wir append (angehängt) an. Das bedeutet, dass die Sachen, die wir an die Datei senden am Ende der Datei angehängt werden. Diese Datei wird nun geöffnet. Wenn sie noch nicht existierte, wird sie nun automatisch erstellt.

Vor dem Namen der Datei ist ein Pfad angegeben. In diesem Fall von: /opt/ft/workspaces/log.txt
ist also im Verzeichnis /opt/ft/workspaces/ die Datei log.txt geöffnet. Dies ist die Verzeichnisstruktur vom TXT 4.0, also das Arbeits-/Programmverzeichnis des TXT 4.0 , -Nicht- die des PCs.

Das Schreiben vom Text passiert mit dem Block "Schreibe...in Datei ... neue Zeile am Ende".

Es wird ein Formatierter Text, hier "Hallo" in die Datei log.txt am Ende angehängt und danach ein Enter oder wie bei einer Textverarbeitung eine neue Zeile angefügt.

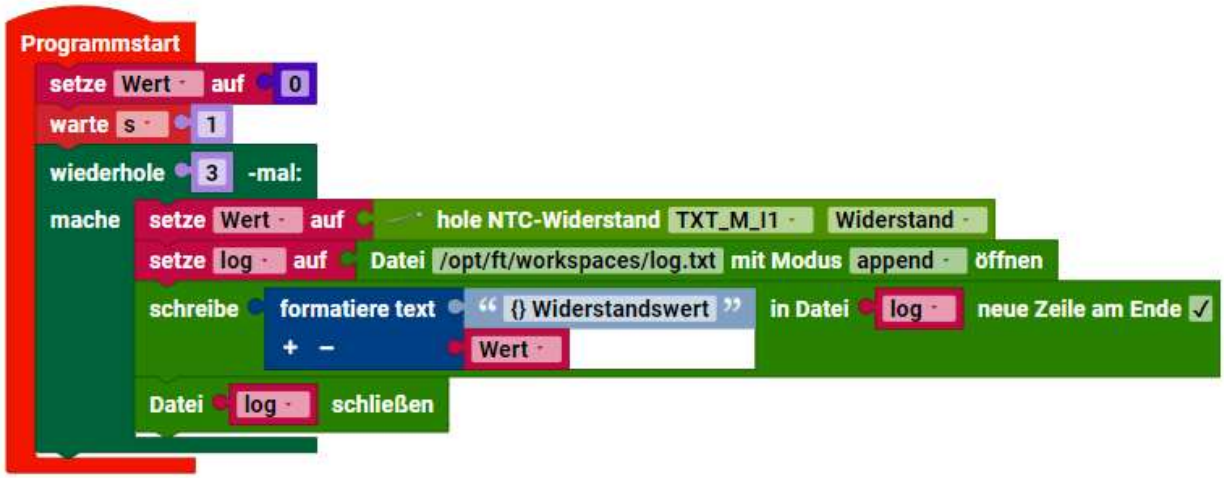
Die Datei wird nun wieder geschlossen.

Das Alles wird dreimal gemacht.

Wenn man den TXT 4.0 über USB angeschlossen hat, kann man die Datei sich anschauen. Dazu die Adresse 192.168.7.2 in den Browser z.B. Firefox eingeben. Man muss noch ft und fischertechnik als Benutzer und Kennwort eingeben.

Man sieht unten die Dateien und Projekte, die auf dem TXT 4.0 sind.

Messwerte speichern
Hier ein Beispiel, wie man Messwerte in eine Datei speichern kann.



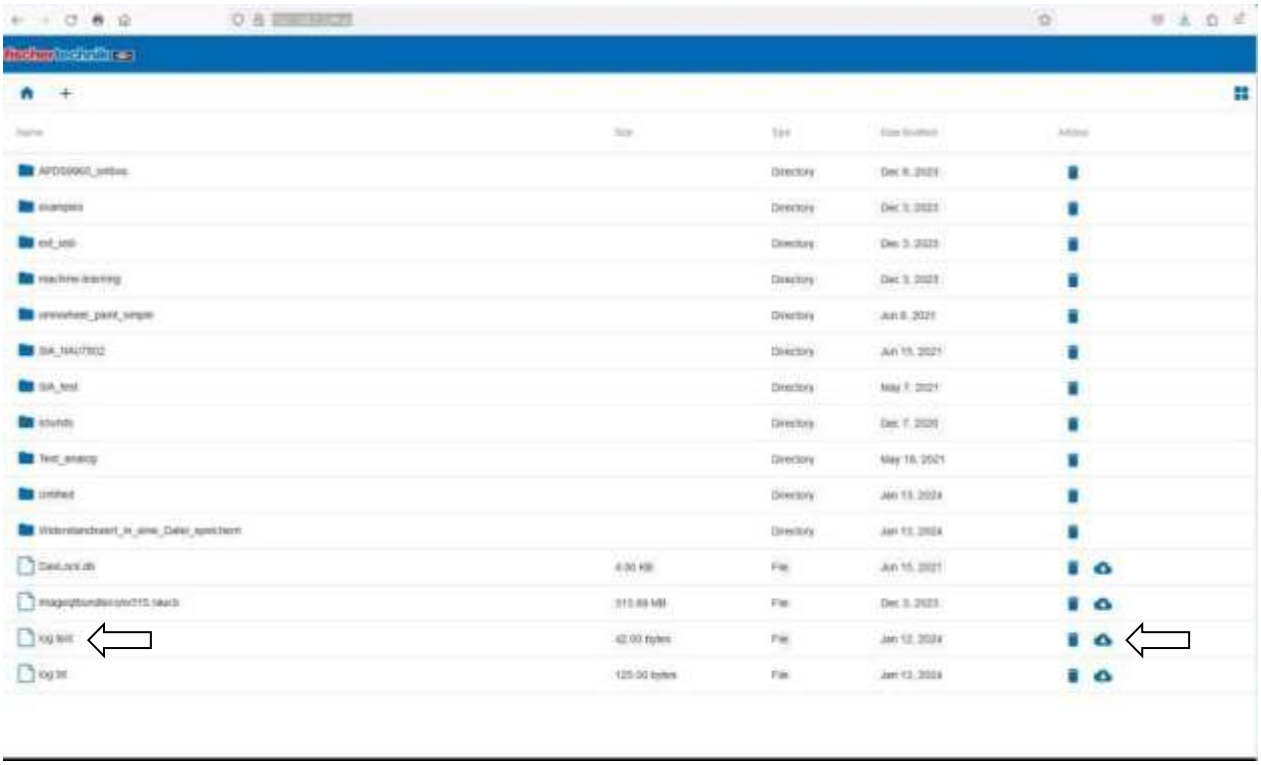
Das Programm nimmt vom I1 den Widerstandswert oder die Temperatur (kann man umstellen) und schreibt sie in eine Datei mit dem Namen „log.txt“.
Die Datei wird entweder erzeugt, wenn sie noch nicht existierte oder geöffnet.
Es werden immer drei Werte gemessen und an die Datei angehängt. Die Daten werden in der Variablen "Wert" gespeichert, in der geschweiften Klammer eingesetzt und der Text Widerstandswert" dahinter geschrieben. Anschließend wird eine neue Zeile der Textdatei erzeugt.
Dann wird die Datei geschlossen.

Achtung! Die Wartezeit am Anfang -muss- dastehen, weil es sonst zu Fehlmessungen kommt bzw. die erste Messung muss verworfen werden.

Warum der Block „formatierter Text“?
Man kann auch nur die Werte der Variablen speichern. Es gibt manchmal das Problem, dass man nicht genau weis, wo ein Wert anfängt oder aufhört. Wenn man die Daten mit anderen Programmen bearbeitet, muss manchmal ein Trennzeichen z.B. Komma eingefügt werden.

Ein weiterer Grund ist, dass die Werte in den Variablen vom Typ Float (Kommazahl) sind und nicht in einen String umgewandelt werden können.

Dateien vom TXT 4.0 lesen, kopieren und löschen (über SSH Zugriff).
Die Datei ist auf dem TXT 4.0, nicht auf dem PC, auch wenn man das Programm auf dem PC startet.
Das "/opt/ft/workspaces" ist das Verzeichnis, was man über den Browser sieht, wenn man darauf zugreift.
Da macht man über die Eingabe im Browser von 192.168.7.2 (Hier USB-Kabel).
Benutzer ist ft und Kennwort fischertechnik (Das ist ein SSH Zugriff auf den TXT 4.0).
Man kann über das Wölkchen ganz rechts, hinter dem Namen, die Datei auf den PC runterladen. Entweder man speichert die dann oder man lässt sich die Datei z.B. im Windows Editor anzeigen. Man kann sie auch mit Excel, Editor, Word usw. öffnen.



Man kann das Ganze auch noch steigern, in dem man ein Autostartprogramm daraus macht und den TXT 4.0 z.B. über eine Zeitschaltuhr kurz einschaltet. Das Programm macht seine Messung und "schaltet" sich dann ab, bzw. die Spannungsversorgung wird dann unterbrochen.

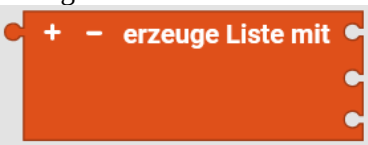
Man kann auch auf einen Stick die Datei schreiben, den man in den TXT 4.0 steckt. Da ist die Path-Angabe dann aber anders. Beachte, dass das Schreiben auf den Stick Zeit kostet.
Vor dem Abschalten sollte besser ca. 1 Minute warten.

Datenstrukturen Listen

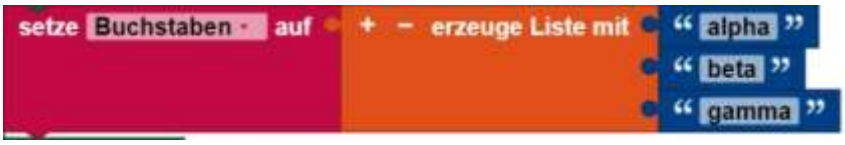
Wie in der Alltagssprache ist auch in ROBO Pro Coding eine Liste eine geordnete Sammlung von Elementen, wie z. B. eine "To-Do"-Liste oder eine Einkaufsliste. Elemente in einer Liste können von beliebigem Typ sein, und derselbe Wert kann mehrmals in einer Liste erscheinen.

Erstellen einer Liste

erzeuge Liste mit



Mit dem Block **erzeuge Liste mit** kann man die Anfangswerte in einer neuen Liste angeben. In diesem Beispiel wird eine Liste von Wörtern erstellt und in einer Variablen namens **Buchstaben** abgelegt:



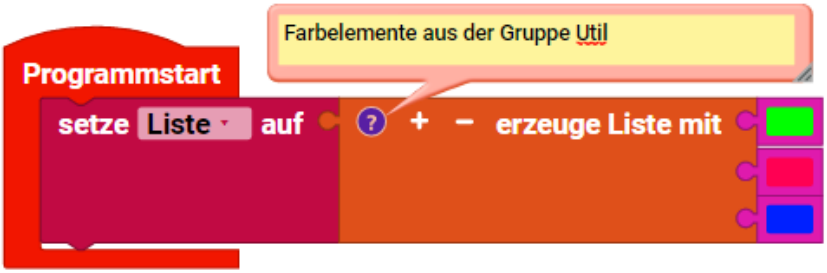
Wir bezeichnen diese Liste als ["alpha", "beta", "gamma"].

Dies zeigt die Erstellung einer Liste von **Zahlen**:

So wird eine Liste von **Farben** erstellt:

Es ist weniger üblich, aber möglich, eine Liste mit Werten unterschiedlichen Typs zu erstellen:

Beispiele:



Anzahl der Eingänge ändern
Um die Anzahl der Eingänge zu ändern, klicke bzw. tippe auf das Plussymbol. Dadurch wird ein neues Fenster geöffnet. Du kannst Elementunterblöcke von der linken Seite des Fensters in den Listenblock auf der rechten Seite ziehen, um einen neuen Eingang hinzuzufügen.
Während das neue Element in diesem Beispiel unten hinzugefügt wurde, kann es überall hinzugefügt werden. In ähnlicher Weise können unerwünschte Elementunterblöcke aus dem Listenblock nach links gezogen werden.

Beispiel:

erzeuge Liste mit … –mal dem Element

Liste mit Element erstellen
Mit dem Block **erzeuge Liste mit Element** kannst du eine Liste erstellen, die die angegebene Anzahl von Kopien eines Elements enthält. Die folgenden Blöcke setzen zum Beispiel die Variable **Wörter** auf die Liste ["sehr", "sehr", "sehr"].

Prüfen der Länge einer Liste

Länge von

Der Wert des **Länge von**-Blocks ist die Anzahl der Elemente, die sich in der als Eingabe verwendeten Liste, befinden. Der Wert des folgenden Blocks wäre z. B. 3, da **Farbe** drei Elemente hat:

Beachte, dass der **Länge von**-Block angibt, wie viele Elemente in der Liste enthalten sind, und nicht, wie viele verschiedene Elemente in ihr enthalten sind. Zum Beispiel hat das Folgende den Wert 3, obwohl **Wörter** aus drei Kopien desselben Textes „sehr“ besteht:

Beachte die Ähnlichkeit mit dem Block **Länge von** für Text.

Beispiel:

ist leer

Der Wert eines **ist leer**-Blocks ist **wahr**, wenn seine Eingabe die leere Liste ist, und **falsch**, wenn es irgendetwas anderes ist. Ist diese Eingabe **wahr**? Der Wert des folgenden Blocks wäre **falsch**, weil die Variable Farben nicht leer ist: Sie hat drei Elemente.

Beachte die Ähnlichkeit mit dem **ist leer**-Block für Text.

in der Liste - Suchen von Elementen in einer Liste

Diese Blöcke finden die Position eines Elements in einer Liste. Das folgende Beispiel hat den Wert 1, weil das erste Auftreten von "sehr" am Anfang der Wortliste steht (["sehr", "sehr", "sehr"]).

Das Ergebnis des Folgenden ist 3, weil das letzte Auftreten von "sehr" in **Wörter** an Position 3 ist.

Wenn das Element nirgendwo in der Liste vorkommt, ist das Ergebnis der Wert 0, wie in diesem Beispiel:

Diese Blöcke verhalten sich analog zu den Blöcken für das Finden von Buchstaben im Text.

Beispiel:

Abrufen von Elementen aus einer Liste

in der Liste … Element „nimm“

Abrufen eines einzelnen Elements

Erinnere dich an die Definition der Liste Farben:

Der folgende Block erhält die Farbe Blau, weil es das zweite Element in der Liste ist (von links beginnend gezählt):

Dieser erhält Grün, weil es das zweite Element ist (vom rechten Ende ausgezählt):

Dieser erhält das erste Element, Rot:

Dies erhält das letzte Element, Gelb:

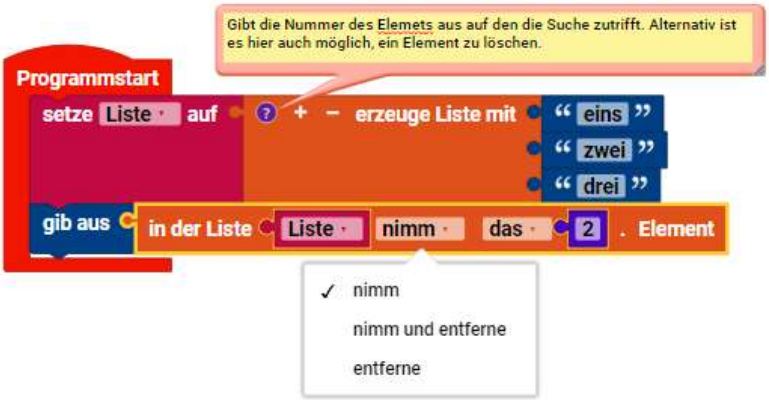
Dies wählt zufällig ein Element aus der Liste aus, wobei mit gleicher Wahrscheinlichkeit eines der Elemente Rot, Blau, Grün oder Gelb zurückgegeben wird.

Abrufen und Entfernen eines Elements

Mit dem Dropdown-Menü wird der Block aus Liste ... abrufen in den Block aus Liste ... abrufen und entfernen geändert, der die gleiche Ausgabe liefert, aber auch die Liste verändert:

Dieses Beispiel setzt die Variable erster Buchstabe auf "alpha" und lässt die restlichen Buchstaben (["beta", "gamma"]) in der Liste.

Beispiel:



in der Liste … Element „entferne“
Entfernen eines Eintrags

Achtung: Wenn du im Dropdown-Menü „entfernen“ wählst, verschwindet die Nase links vom Block:

Damit wird das erste Element aus Buchstaben entfernt.

in der Liste … „setze für“

Hinzufügen von Elementen an eine Liste

Elemente in einer Liste ersetzen

Der Block in Liste ... ersetze ersetzt das Element an einer bestimmten Stelle einer Liste durch ein anderes Element.

Die Bedeutung der einzelnen Dropdown-Optionen findest du im vorherigen Abschnitt.

Das folgende Beispiel bewirkt zwei Dinge:

Die Liste Wörter wird mit 3 Elementen erstellt: ["sehr", "sehr", "sehr"].

Das dritte Element in der Liste wird durch "gut" ersetzt. Der neue Wert von „Wörter“ ist ["sehr", "sehr", "gut"]

Beispiel:

in der Liste … Element „nimm Teilliste ab“

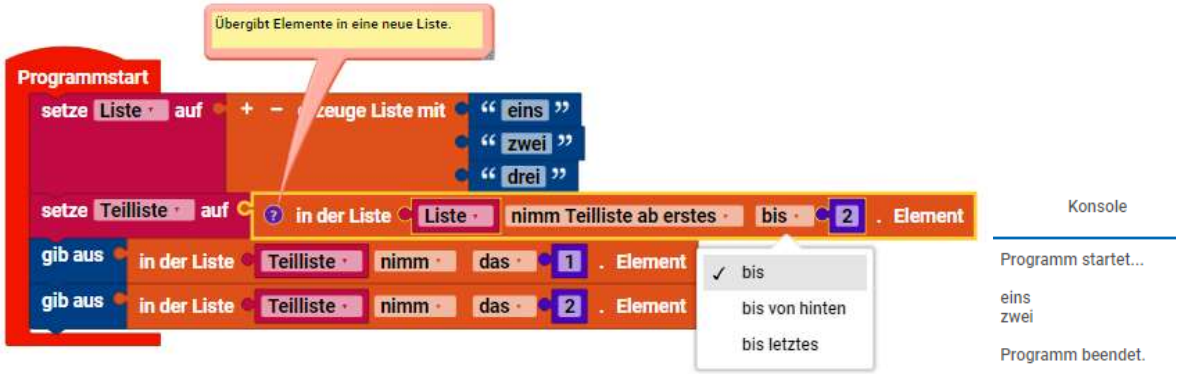
Eine Subliste abrufen

Der Block aus Liste ... Subliste abrufen ähnelt dem Block in aus Liste ... abrufen mit dem Unterschied, dass er eine Subliste extrahiert und nicht ein einzelnes Element. Es gibt mehrere Optionen, den Anfang und das Ende der Subliste anzugeben:

In diesem Beispiel wird eine neue Liste erster Buchstabe erstellt. Diese neue Liste hat zwei Elemente: ["alpha", "beta"].

Beachte, dass dieser Block die ursprüngliche Liste nicht verändert.

Beispiel:



in der Liste … ein „füge als“
Elemente an einer bestimmten Stelle in eine Liste einfügen

Der in Liste ... einfügen bei-Block wird über das Dropdown-Menü des in Liste ... ersetze-Blocks aufgerufen:

Er fügt ein neues Element an der angegebenen Stelle in die Liste ein, und zwar vor dem Element, das sich zuvor an dieser Stelle befand. Das folgende Beispiel (das auf einem früheren Beispiel aufbaut) tut drei Dinge:

Die Liste Wörter wird mit 3 Elementen erstellt: ["sehr", "sehr", "sehr"].

Das dritte Element in der Liste wird durch "gut" ersetzt. Der neue Wert von „Wörter“ ist somit ["sehr", "sehr", "gut"].

Das Wort "Sei" wird am Anfang der Liste eingefügt. Der endgültige Wert von „Wörter“ ist somit ["Sei", "sehr", "sehr", "gut"].

… erstellen mit Trennzeichen

Zeichenketten aufteilen und Listen zusammenfügen

Eine Liste aus einem Text erstellen

Der Baustein erstelle Liste aus Text zerlegt den angegebenen Text mit Hilfe eines Begrenzungszeichens in Teile:

Im obigen Beispiel wird eine neue Liste zurückgegeben, die drei Textstücke enthält: "311", "555" und "2368".

Beispiel:

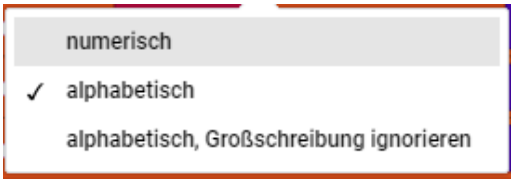
Text aus Liste erstellen

Der Baustein **erstelle Text aus Liste** fügt eine Liste mit Hilfe eines Trennzeichens zu einem einzigen Text zusammen:

Im obigen Beispiel wird ein neuer Text mit dem Wert zurückgegeben: "311-555-2368".

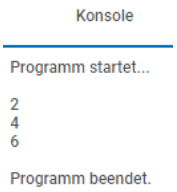
numerisch aufsteigend … sortieren

Beispiel:



Sortiert werden kann:

nummerisch,
alphabetisch und
alphabetisch, Großschreibung ignorieren



Ausgabe des Programms

Verwandte Blöcke

Drucken einer Liste

Der **drucken**-Baustein in der Kategorie Text kann Listen ausgeben. Das Ergebnis des folgenden Programms ist die abgebildete Konsolenausgabe:

Beispiel:

Etwas für jedes Element in einer Liste durchführen

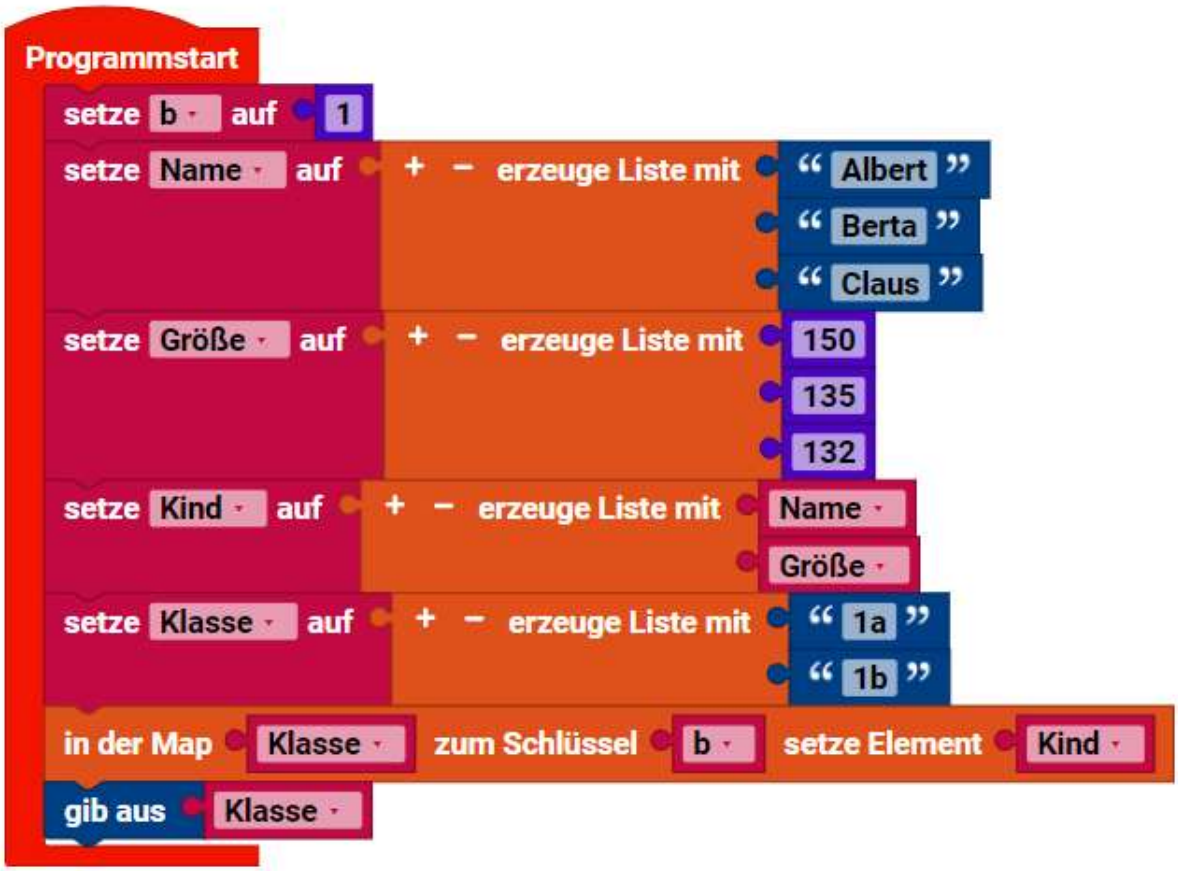
Der **für-jeden**-Block in der Kategorie Steuerung führt eine Operation für jedes Element in einer Liste aus. Dieser Block druckt zum Beispiel jedes Element in der Liste einzeln aus:

Dadurch werden die Elemente nicht aus der ursprünglichen Liste entfernt.

Siehe auch die Beispiele für die Schleifenabbruchblöcke.

Map

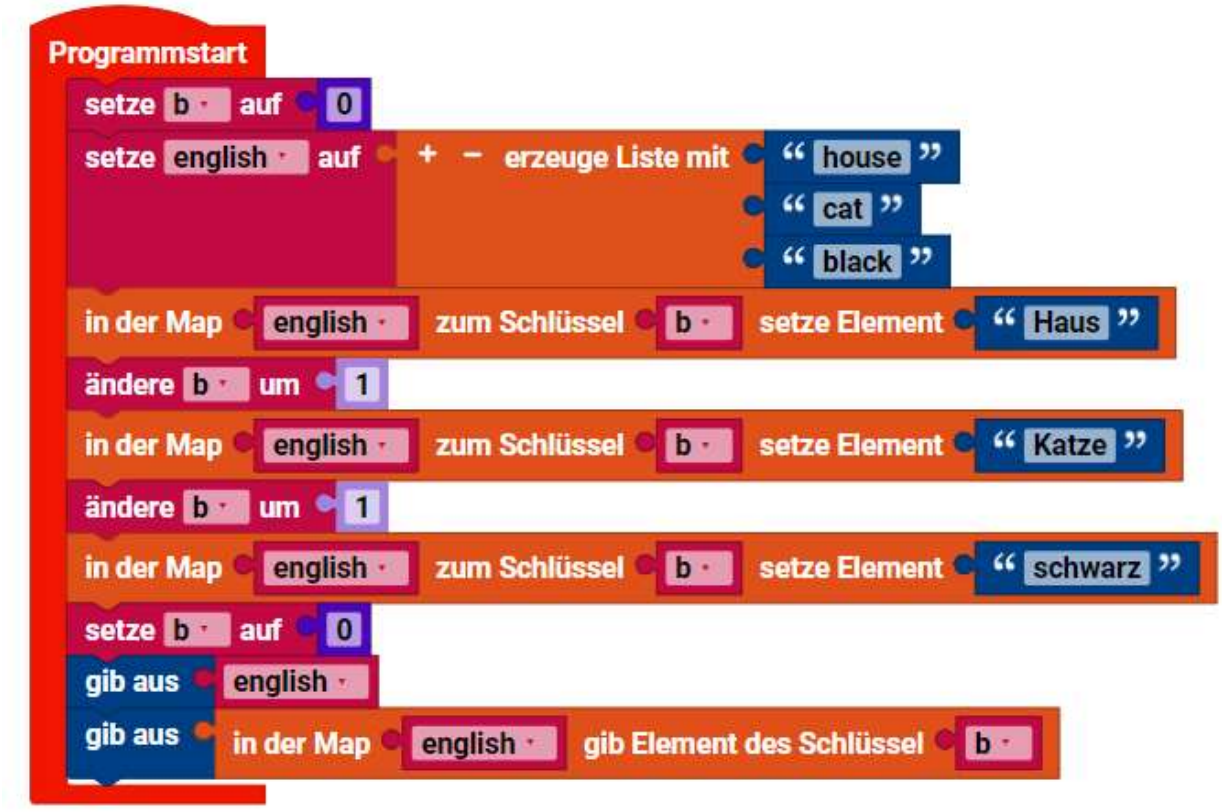
Der Map-Block funktioniert anders als das map in Python. Ich habe es nur so weit nur durch ausprobieren rausbekommen.
Eigentlich suche ich ein Beispiel für die Map-Blöcke, was man besser verstehen kann und wo man den Sinn von Map besser sieht. Leider gibt es dazu keinen Eintrag in der Hilfe vom Robo Pro Coding. Nach längerem hin und her habe ich lauffähige Beispiele hinbekommen.
Man lern ja dazu...
Hier mal ein Beispiel:



Programmablauf:
b ist nur eine durchlaufende Zähler-Variable, hier halt 1 .
Es werden Listen mit Name und Größe erzeugt.
Das Kind setzt sich aus Name und Größe zusammen.
Eine Liste der Klassen ein Schule
Eine Map für die erste Klasse also 1a wird mit den "Kindern" gefüllt.
Ausgegeben wird dann:

```
Konsole

Programm startet...

c
d
['1a', [['Albert', 'Berta', 'Claus'], [150, 135, 132]]]

Programm beendet.
```

OK es läuft schon mal und ich bin stolz es ohne Anleitung hin bekommen zu haben.
Aber zufrieden bin ich damit nicht ganz.
OK, jetzt könnte man die Sache mit einer ganzen Liste aller Namen machen und dann ausschneiden usw.
Das zeigt aber noch nicht ganz den Einsatz von map. Finde ich.
Ich habe dann mal ein Beispiel für ein Dictionary (Standard-Beispiel in Python) versucht:



Ausgabe:

```
Konsole

Programm startet...

['Haus', 'Katze', 'schwarz']
Haus

Programm beendet.
```

Das Dumme ist, dass der Map-Block nur Zahlen als Schlüssel will, keine str-Variable. Somit kann man nicht „cat“ für Katze setzen.

in der Map… gib alle Schlüssel

Hier werden alle Schlüssel für die Elemente ausgegeben.

in der Map … gib Element des Schlüssels …

Hier wird das Element über den Schlüssel ausgegeben.

in der Map…zum Schlüssel… setze Element

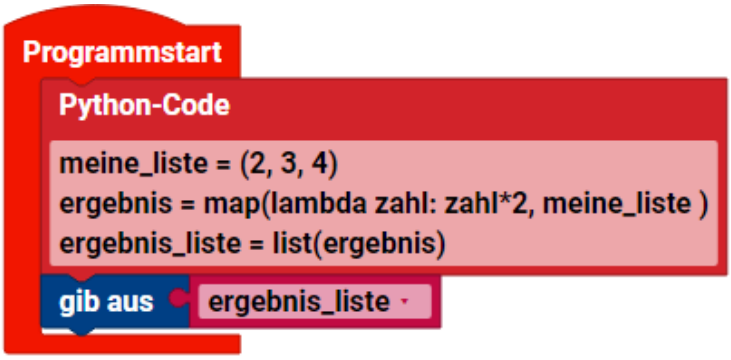
Ich bin mir nicht sicher. Das Hauptproblem, was ich hatte, war der Schlüssel. Warum auch immer kam ein Fehler, dass der Typ nicht stimmte. In Python kann man auch Text als Schlüssel nehmen. Hier nicht – oder ich mache was falsch. Im Grunde wird nur der Index einer Liste benutzt. Das Mouseover zeigt aber „dictionary“ an… ich kann es nicht mit Bestimmtheit sagen.

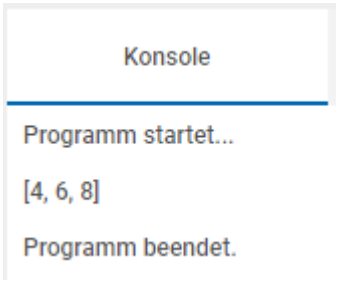
Die Map-Blocks arbeiten mit einem Dictionary, das aus einem Schlüssel und dem dazugehörigen Element besteht. Man kann sich das wie ein Wörterbuch für Englisch vorstellen.

„House“ : „Haus“
„cat“ : „Katze“
„black“: „schwarz“

Map in Python

Die map()-Funktion in Python wird verwendet, um eine Funktion auf jedes Element einer Sequenz (wie einer Liste) anzuwenden und eine neue Sequenz mit den Ergebnissen zu erstellen. Im Beispiel werden die Werte der Liste quadriert und in einer weiteren Liste ausgegeben. Dabei ist „lambda“ eine Funktion ohne Namen, die es ermöglicht, die Transformation ohne Schleife zu erledigen.





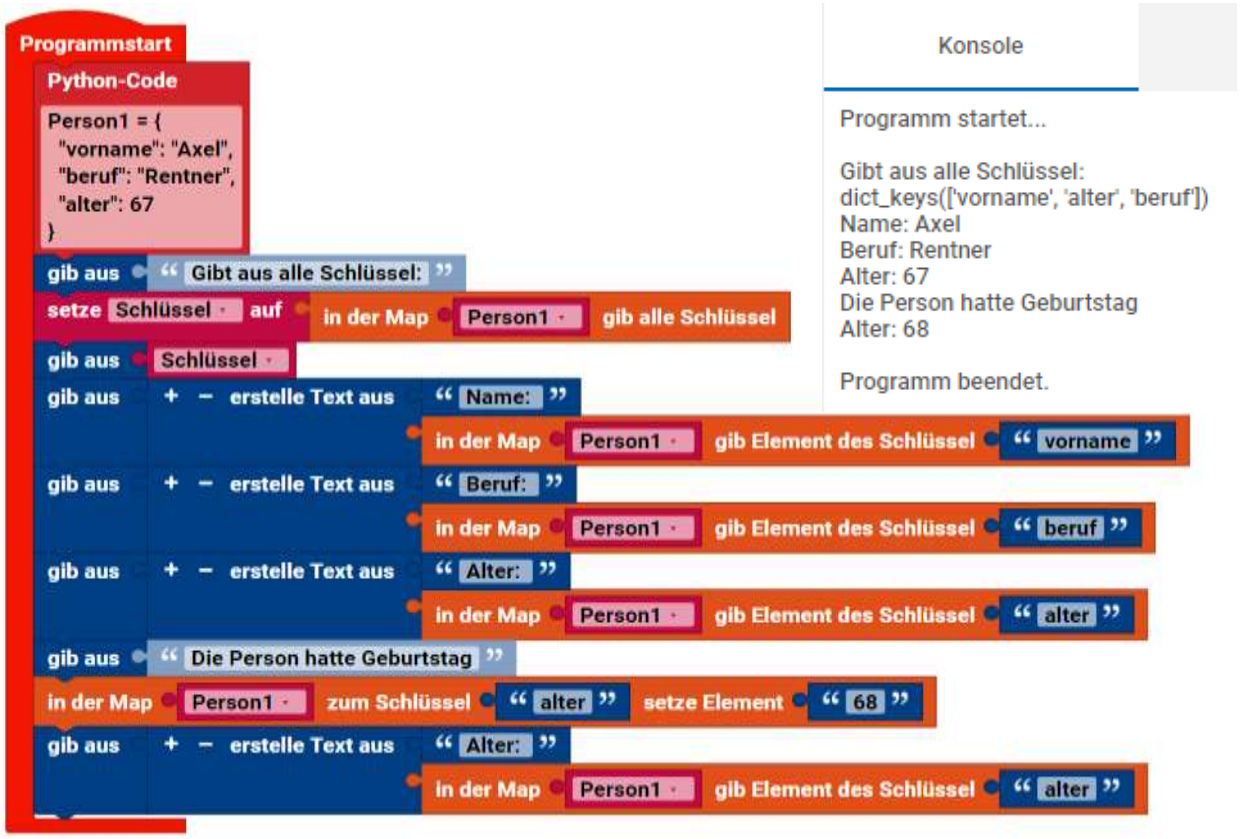
Zeile 1: Erzeugung einer Liste mit den entsprechenden Ziffern.

Zeile 2: Die Funktion Map beginnt mit dem Lamda-Aufruf, der dann einen Paramater benötigt. Nach dem Doppelpunkt wird die eigentliche Funktion ausgeführt. Als letztes wird nach dem Komma der Name der zu bearbeitende Liste definiert.

Zeile 3: Das Map-Objekt wird in eine Liste zurückgewandelt und anschließend auf der Konsole ausgegeben.

Map in fischertechnik Robo Pro Coding

Eine Map in Robo Pro C, auch bekannt als Dictionary oder Assoziatives Array, ist eine Sammlung von Schlüssel-Wert-Paaren, bei der jeder eindeutige Schlüssel direkt auf einen Wert verweist. Die Integration von Dictionaries in Robo Pro Coding ist begrenzt. Die Erstellung ist nicht direkt möglich, sondern über den Block JSON (Siehe JSON zu Map) oder direkt mit Python.



Zuerst wird die Map „Person1“ mit Python erstellt. Dann werden die Schlüssel angefordert, um zu Testzwecken die Korrektheit zu dokumentieren.

Um nun mehr über die Person zu erfahren, muss man nicht die Position der Information kennen, sondern man fordert die Information direkt über den Schlüssel an.

Nach dem Geburtstag ist die Person ein Jahr älter. Dazu wird das Programm aktualisiert, indem das Alter verändert wird. Zum Abschluss wird nun auch das neue Alter ausgegeben.

JSON

JSON = „JavaScript Object Notation“. Es ist ein Datenaustauschformat, das nicht nur auf Java beschränkt ist. Vereinfacht gesagt, wird eine Python-Liste, mit den JSON Klammern, Trennern und Hochkommata unterteilt und auch der Text eingerückt. In sehr vielen Programmiersprachen gibt es die Möglichkeit JSON-Daten einzulesen.

JSON (JavaScript Object Notation) ist ein textbasiertes Datenformat, das für den Datenaustausch im Web genutzt wird. Es ist leicht lesbar und einfach zu handhaben. JSON strukturiert die Daten als Sammlung von Schlüssel-Wert-Paaren, ähnlich wie Dictionaries oder Maps.

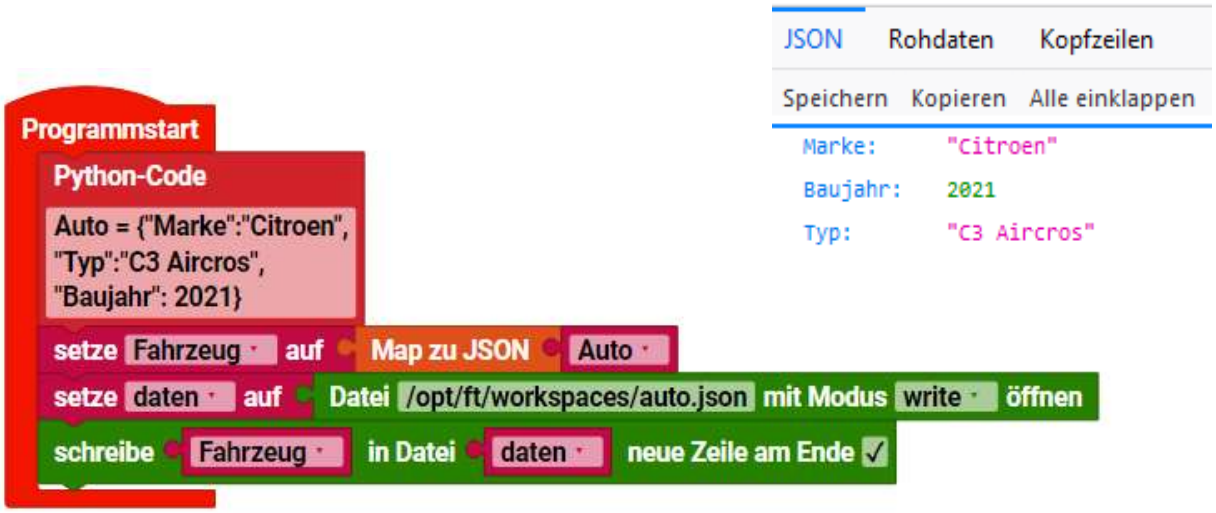
Konvertierung Map zu JSON und umgekehrt. Wird auch u.a. bei HTML verwendet.

Map zu JSON

Map zu JSON Konvertierung

Manchmal sagt man auch JSON Dump. Hier wird also die Liste formatiert ausgegeben, so dass es von anderen Programmiersprachen gelesen werden kann.

Der Map zu JSON-Block ermöglicht die Umwandlung von Map-Datenstrukturen in einen JSON-String. Dieser Vorgang ist besonders nützlich, wenn die strukturierten Daten einer Map für eine Datenübertragung oder als Konfigurationsdatei in einem standardisierten Format benötigt werden.



Im Beispiel wird eine Map „Auto“ über Python erzeugt. Diese wird nun zu einem JSON-String umgewandelt.

Im folgenden werden die Daten in die Datei „auto.json“ übertragen. Das Ergebnis nach dem öffnen der Datei ist oben zu sehen.

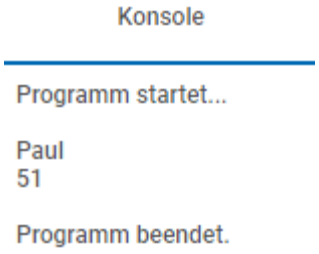
Hinweis: Alle in den Beispielen vorkommenden Variablen, müssen vorher angelegt werden!

JSON zu Map

JSON zu Map Konvertierung

Hier wird die Python-Notation umgewandelt. {} in dict,[] in list, Zahlen in int float… usw.

Dieser Block nimmt einen JSON-String als Eingabe und konvertiert diesen in eine Map. Jedes Schlüssel-Wert-Paar im JSON-String wird zu einem Eintrag in der Map und der Inhalt kann unter Verwendung des Schlüssels abgerufen werden.



Util

Util = Nützlich
Die Kategorie Benutzung beinhaltet bei ROBO Pro Coding Blöcke folgender Art:

- Farbauswahl
- Warten
- Python Code
- Starten
- Funktionsausführung

Farbauswahl

Dieser Block dient als Eingabewert, wenn nach einer Farbe gefragt wird (z.B. beim Farbabgleich durch die Kamera). Durch Klicken bzw. Tippen auf die Farbe kann aus einer Farbpalette eine von 70 Farben ausgewählt werden.

Beispiel:

warte

Warten bis die Zeit abgelaufen ist

Der Block **warte [] ...** hindert das Programm für die angegebene Wartezeit daran, weiterzulaufen. Dabei kann im Dropdown-Menü (kleines Dreieck) die Zeiteinheit und im Eingabefeld dahinter die gewünschte Länge der Pause gewählt werden.

warten bis

Beim **warte bis**-Block ist die Pause nicht an die Zeit, sondern an die Erfüllung einer Bedingung (z.B. ob ein Taster gedrückt ist) geknüpft. Die Bedingung wird an den **warte bis**-Block angehängt.

Python-Code in Robo Pro Programmen

Python Importe

Unterprogramm

Eine andere Position/Möglichkeit, wo man den Python-Code lassen kann, ist ihn in einem separaten Teil zu speichern. Das kann man mit den Unterprogramm Python Importe machen. Er wird unabhängig vom Hauptprogramm gestartet. Als Anfänger braucht man diesen Befehl nie. Mir fällt spontan auch kein einfaches Beispiel ein.

Python Code

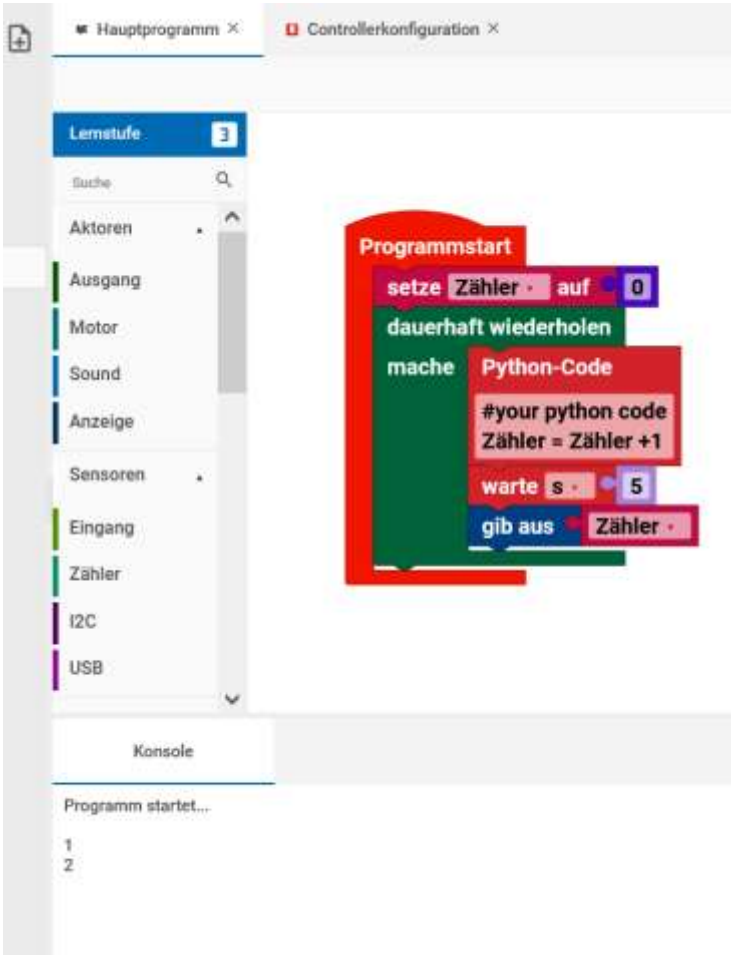
Möchte man bestehenden Python-Code in ROBO Pro Coding (Hauptprogramm) integrieren, so kann man ihn in den **Python Code**-Block einfügen. Das Programm führt dann alles aus, was in dem Block in Python geschrieben wurde.
Die Darstellung im Block ist ein gekürzter Text. Man muss sich den ganzen Text bzw. Pythoncode anschauen, in dem man daraufklickt. Das kann sonst manchmal irritieren.

Man kann mit dem #-Zeichen auch zusätzliche Erklärungen einbringen. So ist ein Pythoncode besser lesbar. fischertechnik selbst macht das auch so, in ihren Programmen.

Ein Klick auf „#your python code“ öffnet ein neues Fenster.

Und hier kann man seinen Pythoncode eingeben.

Beispiel:



Hier wird eine Variable „Zähler“ von Robo Pro Coding erzeugt und auf Null gesetzt. In einer Dauerschleife wird die Variable Zähler um 1 hochgezählt. Nun wird 5 Sekunden gewartet und die Zahl auf der Konsole ausgegeben. Der Programmabbruch erfolgt durch Stopp, oben rechts oder am Controller.
Hier im Programm sieht man den ganzen Pythoncode, weil er sehr kurz ist. Ansonsten muss man darauf klicken, um ihn ganz sehen zu können.

Älterer Block:

Starten

Auch der **starte wenn**-Block ist an eine Bedingung geknüpft. Erst wenn diese Bedingung erfüllt ist, startet das im Blockkörper stehende Programm.

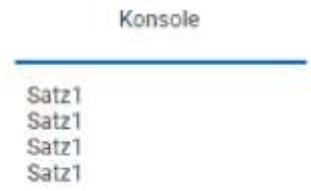
Dieser Block ist nicht mehr Bestandteil von Robo Pro Coding.

Funktionsausführung

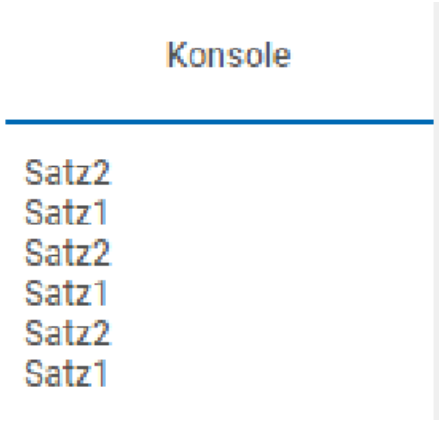
führe Funktion ... in einem Thread aus

Mit dem **führe Funktion ... in einem Thread aus** lässt sich die ausgewählte Funktion in einem separaten Thread ausführen. Diese Maßnahme kann in manchen Fällen ermöglichen, dass ein Programm weiterhin auf Eingaben reagieren kann und schneller ausgeführt wird. Es wird aus einer Funktion ein eigenständiges Unterprogramm.

Im Normalfall werden Programme nacheinander ausgeführt. So wartet das Programm auch mit der Weiterführung, wenn ein Unterprogramm abgearbeitet wird. Da hier das Unterprogramm unendlich ist, läuft das Hauptprogramm nicht weiter.



Deshalb wird hier das Unterprogramm 2 angewiesen, als Thread (also parallel) zu laufen.



Zeitstempel

Hiermit kann man einen Zeitstempel erstellen und einer Variablen zuweisen. Es wir die Systemzeit genommen. Hier kann man noch unter ms und s wählen.

Hinweis: Man kann es anders/besser machen mit einem Unterprogramm:

Unterprogramm mit einem Python-Code-Block
Python Code:

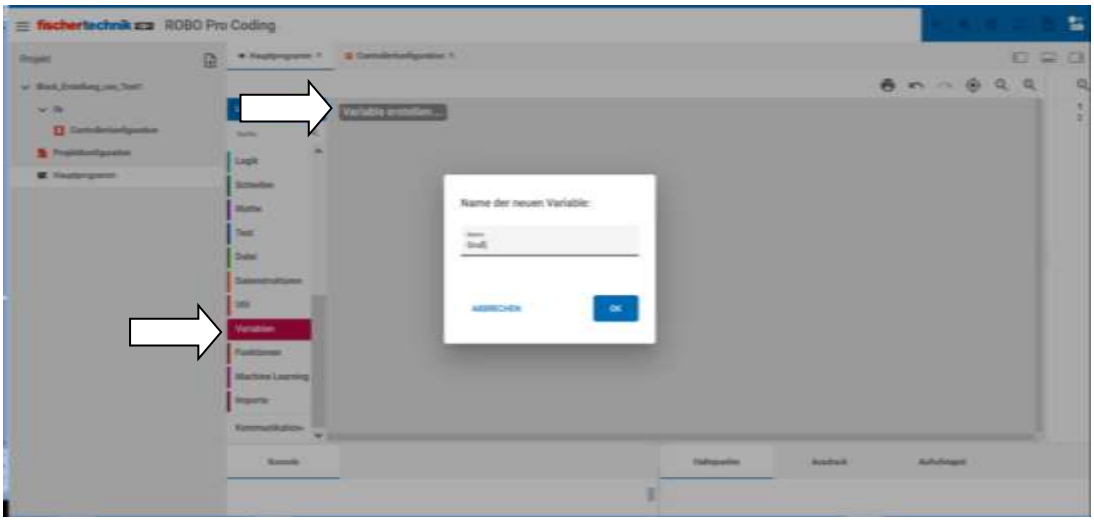
```
ts = datetime.now().strftime("%Y-%m-%dT%H:%M:%S.%f")[:-3] + "Z"
```

Variablen

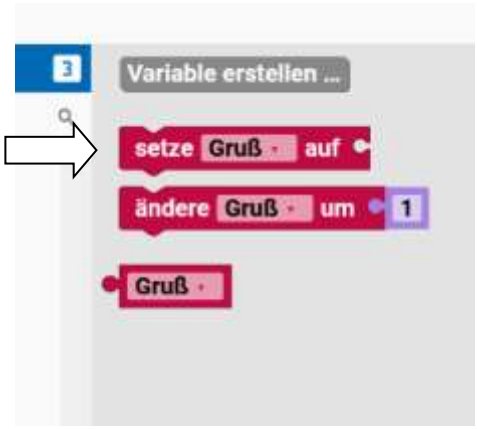
Wir verwenden den Begriff Variable so, wie er in der Mathematik und in anderen Programmiersprachen verwendet wird: ein benannter Wert, der verändert (variiert) werden kann. Variablen können auf verschiedene Arten erstellt werden.

Eine andere Art der Vorstellung ist Variable = Schuhkarton. Man kann verschiedene Sachen reintun und den Karton mit einem Namen beschriften.

Erstellen einer Variablen über den Button „Variable erstellen“ im Reiter Variablen.



Variablen klicken– Variable erstellen klicken. Name der neuen Variablen eingeben. In diesem Fall hat die Variable den Namen „Gruß“.



Es erscheinen Blöcke, wo der vorher eingetippte Name automatisch eingetragen wird.

- Einige Blöcke wie **zähle mit** und **für jeden** verwenden eine Variable und definieren ihre Werte. Ein traditioneller Informatik-Begriff für solche Variablen lautet Schleifenvariablen.
- Benutzerdefinierte Funktionen (auch als "Prozeduren" bezeichnet) können Eingaben definieren, wodurch Variablen erzeugt werden, die nur innerhalb dieser Funktion verwendet werden können. Solche Variablen werden traditionell als "Parameter" oder "Argumente" bezeichnet.
- Benutzer können jederzeit Variablen über den **setze**-Block verändern. Diese werden traditionell als "globale Variablen" bezeichnet. Sie sind überall im Code von ROBO Pro Coding verwendbar.

Dropdown-Menü

Wenn du auf das Dropdown-Symbol (kleines Dreieck) einer Variablen klickst, erscheint das folgende Menü:

Das Menü bietet die folgenden Optionen.

- die Anzeige der Namen aller vorhandenen, im Programm definierten Variablen.
- "Variable umbenennen...", d.h. die Änderung des Namens dieser Variable, wo immer sie im Programm erscheint (die Auswahl dieser Option öffnet eine Abfrage für den neuen Namen)
- "Variable löschen...", d.h. das Löschen aller Blöcke, die auf diese Variable verweisen, wo immer sie im Programm vorkommt.

Blöcke

setze auf

Festlegen

Der **setze**-Block weist einer Variablen einen Wert zu und legt die Variable an, falls sie noch nicht existiert. Zum Beispiel wird so der Wert der Variable **Alter** auf 12 gesetzt:

Beispiele:

Hier werden die Variablen mit den entsprechenden Inhalten gefüllt.

rufe ab

Abrufen

Der **rufe ab**-Block liefert den in einer Variablen gespeicherten Wert, ohne ihn zu verändern:

Es ist möglich, aber eine schlechte Idee, ein Programm zu schreiben, in dem ein **rufe ab**-Block ohne einen entsprechenden vorherigen setze-Blockvorkommt.

ändere um

Ändern

Der **ändere**-Block fügt eine Zahl zu einer Variablen hinzu.

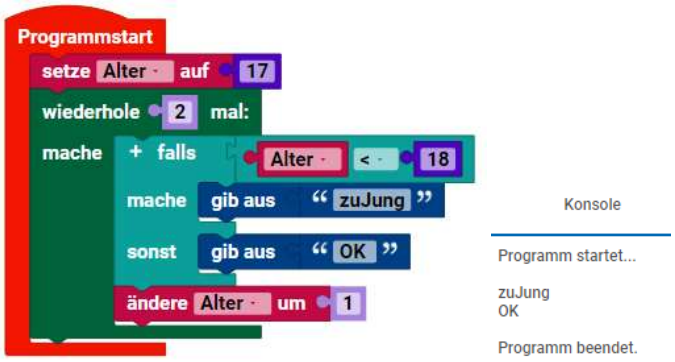
Der **ändere**-Block ist eine Abkürzung für das folgende Konstrukt:

Beispiel

Betrachte den folgenden Beispielcode:

Die erste Reihe von Blöcken erzeugt eine Variable namens „**Alter“** und setzt ihren Anfangswert auf die Zahl 12. Die zweite Reihe von Blöcken **ruft** den Wert 12 **ab**, addiert 1 dazu und speichert die Summe (13) in der Variablen „Alter“. In der letzten Zeile wird die Meldung ausgegeben: "Herzlichen Glückwunsch! Du bist jetzt 13".

Beispiel:



Funktionen

Funktionen dienen dazu, Teile des Codes wiederverwendbar zu machen und dadurch den Code insgesamt zu strukturieren. Füllt man einen Funktionsblock, so erscheint im Funktionen-Menü ein neuer Block, der den gleichen Namen trägt wie eben dieser Funktionsblock. Es ist nun möglich, in das Hauptprogramm nur noch den Block mit dem Namen der Funktion einzusetzen. Wenn das Programm durchlaufen wird, leitet dieser Block zum Code in der gleichnamigen Funktion weiter und arbeitet diesen ab.

Einfache Funktion ohne Rückgabe

Mit dem einfachen Funktionsblock lässt sich eine Funktion erstellen, die den im Textfeld eingegebenen Namen trägt. Die Funktion kann beliebig viele Variablen enthalten, die über das Plussymbol hinzugefügt werden können. Diese Funktion „**Altern“** addiert 1 zu der Variable **Alter**:

Hinweis: Obwohl diese Funktion keinen Rückgabewert hat, ändert sie eine Variable. Und Variablen sind in Robo Pro Coding „Global“. Sie können von allen Unterprogrammen gelesen und geändert werden. Es gibt keine neuen globalen Variablen mit genau demselben Namen. Wenn man Funktionen aus anderen Programmen nutzen möchte, sollte man auf die Namen achten. Negatives Beispiel ist „Zähler“. Wenn er in mehreren Funktionen verwendet wird, kann es zu Verwirrungen führen.

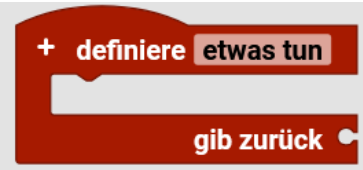
Die Funktion kann dann im Hauptprogramm genutzt werden:

Das Programm setzt die Variable „Alter“ auf 12. In der Funktion „Altern“ wird 10 mal 1 dazu addiert und dann auf der Konsole ausgegeben.

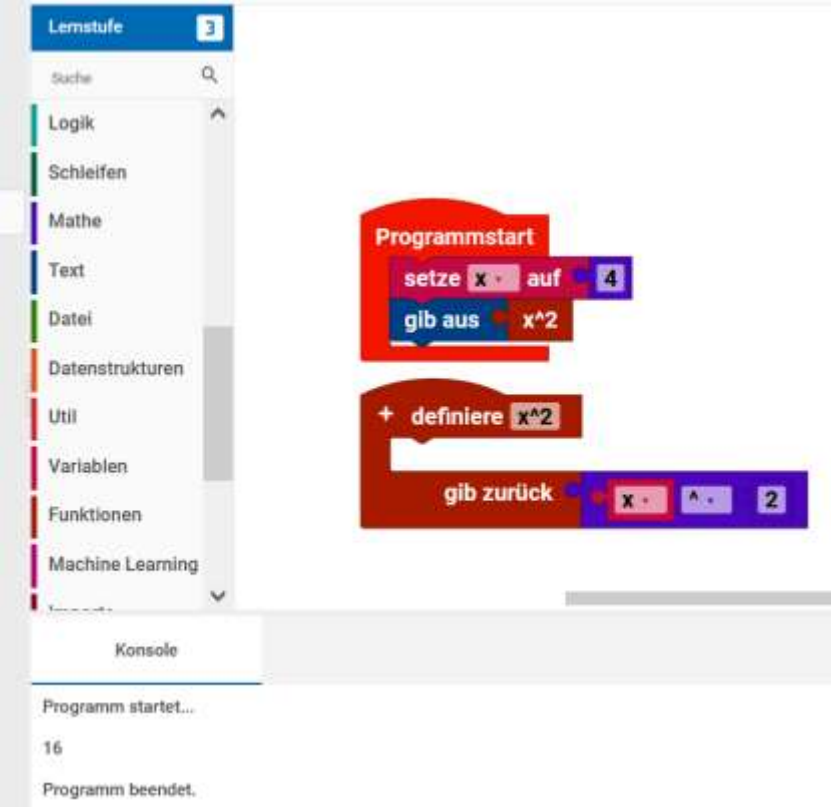
Hintergrund:

Wenn man die Funktion „Altern“ erzeugt, wird automatisch ein Block mit dem Namen „Altern“ unter dem Reiter Funktionen erzeugt.

Funktion mit Rückgabewert



„definiere etwas tun“ – Block. Dieser Block erlaubt es eine Funktion mit Rückgabewert zu erstellen. Dieser Rückgabewert kann dann im Hauptprogramm weiterverwendet werden. Hier ein Beispiel:



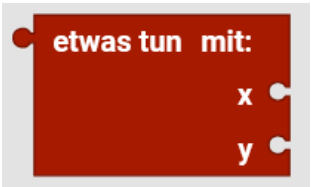
Hie eine Funktion, die im mittleren Teil etwas berechnet.

Hintergrund:

Wenn man die Funktion „x^2“ erzeugt, wird automatisch ein Block mit dem Namen „x^2“ unter dem Reiter Funktionen erzeugt.

Dieser Block  kann dann wie eine Zahl oder Variable genutzt werden.

Jeder Klick auf das Plus erzeugt eine weitere Eingangsvariable. Hier die Standartvorgabe x und dann y. Als nächste würde z kommen und dann mit a weitergehen. Man kann die jederzeit umbenennen.



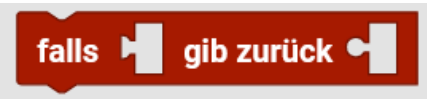
Im Reiter „Funktionen“ erscheint dieses Unterprogramm als Block mit zwei Eingängen. Wir haben ja vorher auch nur zwei erschaffen.

Hier hätte man vier Eingänge. Mit einem rechtsklick auf den Block kann man die Funktion als Block in das Programm einsetzen oder die Variablen (x-a) in dem Unterprogramm verwenden.

Funktion als Block Variable

Interessanterweise wird beim Löschen des Unterprogramms nur der Funktionsblock gelöscht, nicht die Variable.

falls gib zurück



Dieser Block gibt den zweiten Wert zurück und verlässt die Funktion, falls der erste Wert wahr ist.

Im Hauptprogramm wird A auf 5 gesetzt und die Funktion „mathe“ ausgegeben. Die Funktion „mathe“ prüft, ob das Quadrat kleine 10 ist. Wenn dem so ist wird „zu groß“ ausgegeben. Bei 5 gibt sie „zu klein“ aus.

Machine Learning

Die Gruppe "Machine Learning" enthält Blöcke zur Verwendung mit dem TensorFlow-Projekt (https://git.fischertechnik-cloud.com/ml/machine-learning) und der USB-Kamera.

Ein einfaches Beispiel zu machen ist schwer. Ich habe aber das Programm der KI/AI-Sortierung hinten im Buch, mit vielen zusätzlichen Erklärungen versehen.

Bildklassifikator erstellen…

Erstellt eine Bildanalyse für ein Model im Pfad „path“ auf dem TXT 4.0 Controller.

Objektdetektor für … erstellen

Erstellt eine Objekterkennung mit einem Standardmodell "Sortierstrecke mit KI". Man kann momentan auch keine andere auswählen.

Objektdetektor erstellen…

Erstellt eine Objekterkennung für ein Model im Pfad „path“ auf dem TXT 4.0 Controller.

Bild … mit … verarbeiten

Analysiert ein Bild mit einer Objekt- oder Bilderkennung. Ausgegeben werden die erkannten Eigenschaften, deren Wahrscheinlichkeit und bei einer Objekterkennung deren Position. Das Ergebnis dieses Blocks kann in eine Variable geschrieben werden, um dieses später im „get value of result“ Block auszuwerten.

Erhalte … der Ergebniseigenschaft …

Gibt einen einzelnen Wert einer Eigenschaft des x-ten Ergebnisses einer Bild- oder Objektanalyse aus. kann dabei entweder direkt der „process image“ Block oder dessen Ergebnisse aus einer Variablen sein.

Probability = Wahrscheinlichkeit
Position = Position
Label = Beschriftung

Allgemeine Vorgehensweise.

Die Vorgehensweise mit AI/KI ist, eine KI zu erstellen und diese anzulernen.
Die KI TensorFlow von Goggle, kann zwei Grundsätzliche Modis verarbeiten.

1. Aus einer Punktwolke eine Tendenz erstellen und
2. Bildanalyse Bildbewertung.

Das Ergebnis ist dann eine Wahrscheinlichkeit. Zusammen mit der Bildanalyse von OpenCV, wird aus dem Videostream ein Bild gewonnen, das nach Objekten untersucht wird. Das Bild selbst, Bildwerte wie Farbe, Helligkeit… und die Position werden als (ausgeschnittenes) Objekt übergeben. TensorFlow untersucht das ausgeschnittene Bild. Auch die anderen so gewonnenen Bildwerte wie Farben kann man im eigenen Programm nutzen, wobei der Farbwert erst umgewandelt werden muss

Wenn man eine eigene KI anlernt, sieht man erst kein richtiges Ergebnis. Es gibt von fischertechnik eine Kurzanleitung, wo das Vorgehen beschrieben ist. Als Ergebnis, das etwas gemacht wurde, erscheint ein Dreieck. Erst im Modell zeigt sich, ob es geklappt hat. Z.B. kann man mal andere Aufkleber machen mit Quadraten, so dass nur diese statt der Bohrlöcher als OK erkannt werden.

Tipp: Die Farberkennung im AI/KI AddOn Modell muss man im Modell ausprobieren. Es hat sich gezeigt, dass der Farbraum anders ist, da die Farben von RGB nach HSV umgewandelt werden.

Das Anlernen selber funktioniert leider noch nicht automatisch mit einem vorgegebenen Programm.
Leider ließ sich bisher die von fischertechnik vorgefertigte ZIP-Datei zu einem Programmstart überreden.
Man muss also sich selber ein Programm schreiben, das die Bilder erfasst.
Laut Anleitung braucht man so ca. 200 Bilder pro Objekt.

Importe

Importe enthält alle Funktionen aus selbst definierten Modulen in "lib".

Funktionen dienen dazu, Teile des Codes wieder verwendbar zu machen und dadurch den Code insgesamt zu strukturieren.

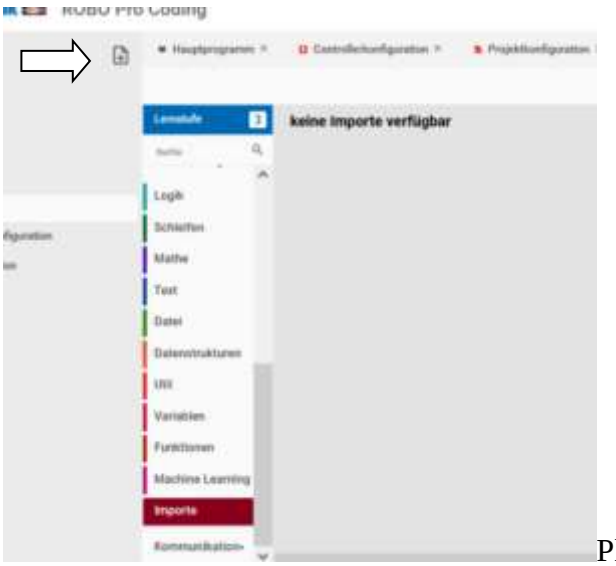
Siehe auch "Funktionen"

Wenn man das erste Mal Importe aufruft, sind keine Blöcke vorhanden und es steht da „keine Importe verfügbar“.

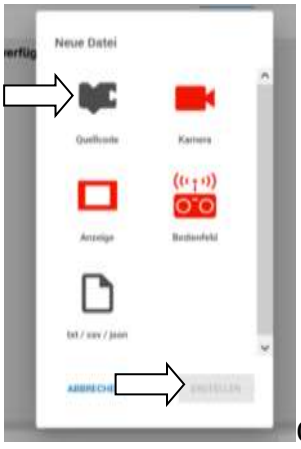
Man kann über das Hinzufügen einer Quelldatei, in der Funktionen sind, diese Funktionen in das Hauptprogramm Importieren. Der Vorteil ist z.B., wenn man viele Programme macht, die immer dieselbe Ausgabe haben, so braucht man nur einmal eine Ausgabe programmieren und verwendet sie in allen anderen Programmen.

Neue Datei „Quellcode“

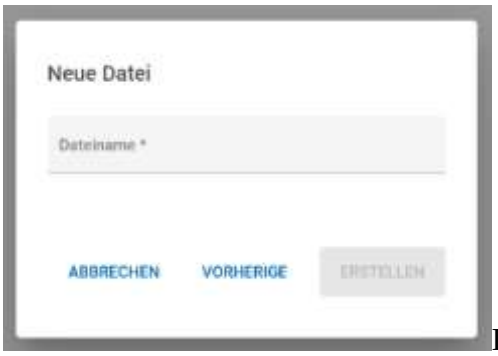
Erstellen wir eine neue Datei „Quellcode“ über das Plus neben Projekt.



Plus anklicken



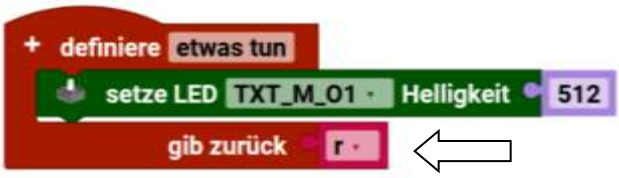
Quellcode anklicken und dann das blaue Erstellen

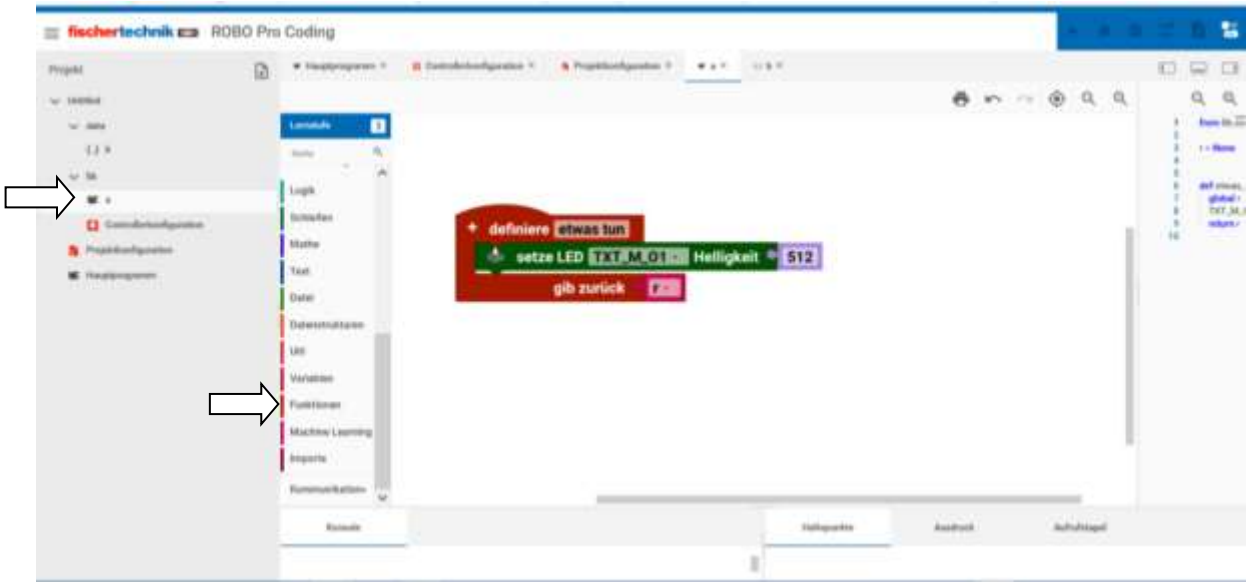


Dann wird man nach dem Dateinamen gefragt. Wir sagen mal „a“.

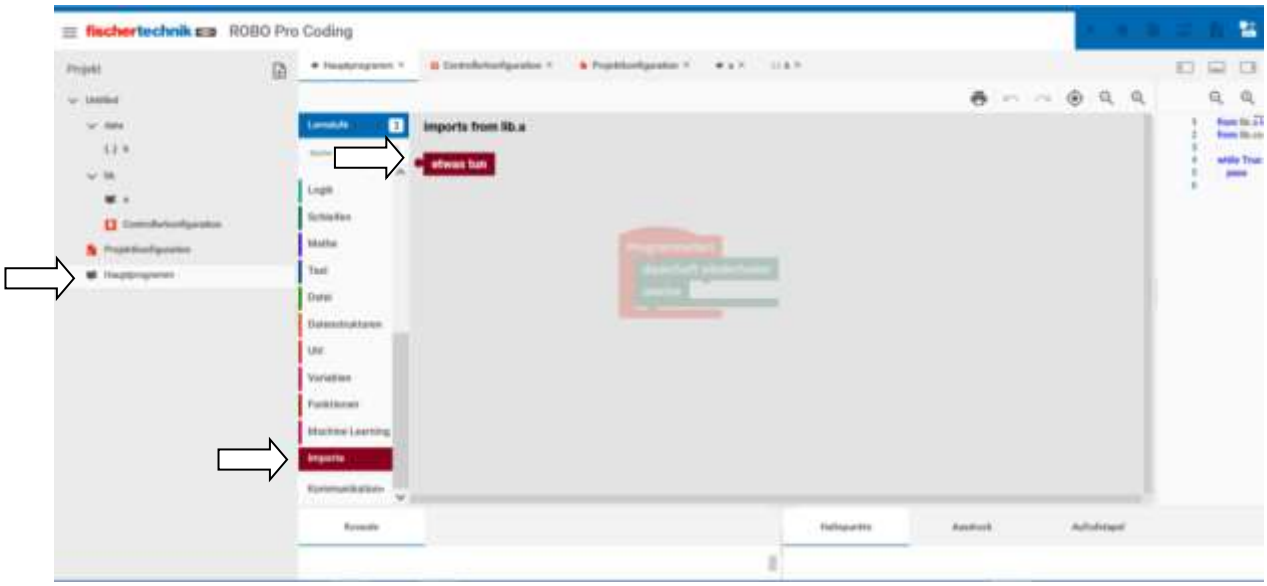
Hinweis: Wenn eine Datei mit demselben Namen schon vorhanden ist, wird automatisch eine Durchnummerierung vorgenommen. So wird z. B. aus „a“ „a2“

Funktion mit Rückgabewert importieren





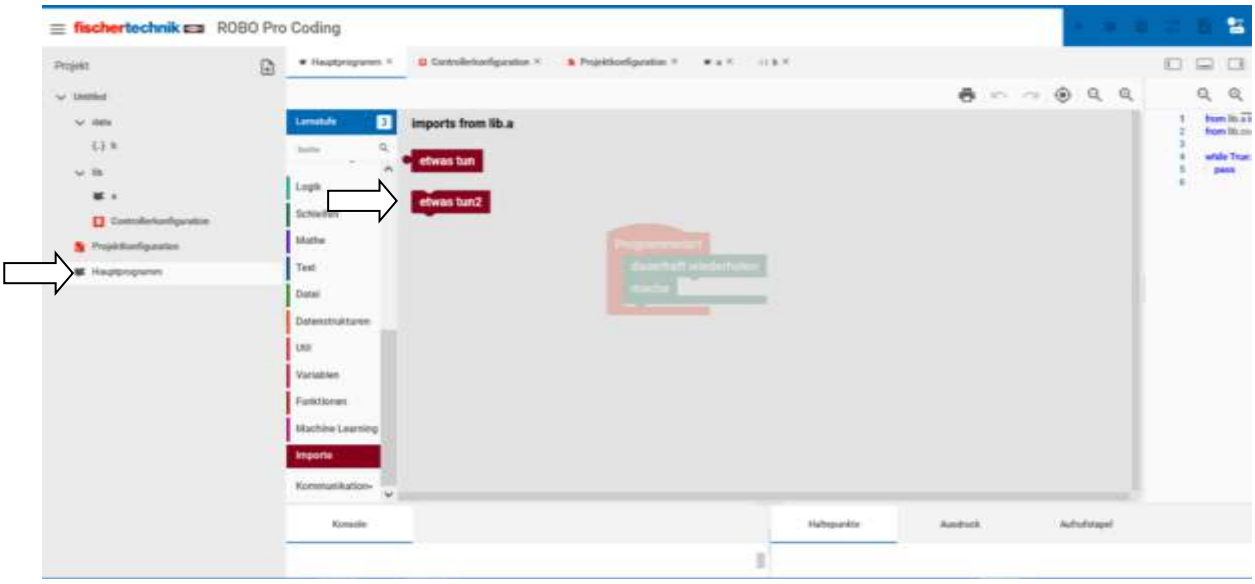
In die Quellcodedatei „a“ fügen wir eine Funktion ein. Hier ist es eine Funktion die einen Wert (Inhalt der Variablen „r“) zurückgibt. Die Funktion setzt nun einfach die Helligkeit einer LED. In diesem Beispiel hat die Variable aber nicht mit dem Steuern der LED zu tun.



Im Hauptprogramm steht nun bei Importe die Funktion „etwas tun“ und sie gibt den Inhalt der Variablen r weiter. In diesem Beispiel geht es nicht um einen sinnvollen Einsatz, sondern nur um das Zeigen wie man einen Import machen kann, da man hier einfach die Variable „r“ nehmen könnte, da die Variablen global sind, also aus allen Unterprogrammen gelesen werden können.

Funktion ohne Rückgabewert importieren

Wenn man im Quellcode „a“ eine Funktion ohne Rückgabewert einsetzt.



Stehen im Hauptprogramm zwei Unterprogramme zur Verfügung.
Das „etwas tun2“ hat keinen Rückgabewert und ist ein „normaler“ Baustein.
Man kann sich also so „eigene“ Bausteine erschaffen.

Kommunikation

Bedienfeld / Fernbedienung (auf dem PC/Handy-Bildschirm)

Die Blöcke des Bedienfelds / der Fernbedienung sind sehr ähnlich denen der Anzeige und der TXT-Anzeige (Aktoren).
Es ist eine Virtuelle Schalttafel, mit der Modelle per Maus vom PC/Notebook oder Tablet gesteuert werden können. Auf ihr kann man Anzeigen und Bedienelemente aufbringen.

RC = Remote Controll = Fernbedienung
Der vorgeschlagene Name, für die Regler und Anzeigen fängt mit remote an.

Zwei ausführliche Beispiele zur „BT Smart Fernbedienung“ / Bedienfeld befinden sich am Ende des Buches. Das Beispiel kann man auch für andere Controller nutzen.

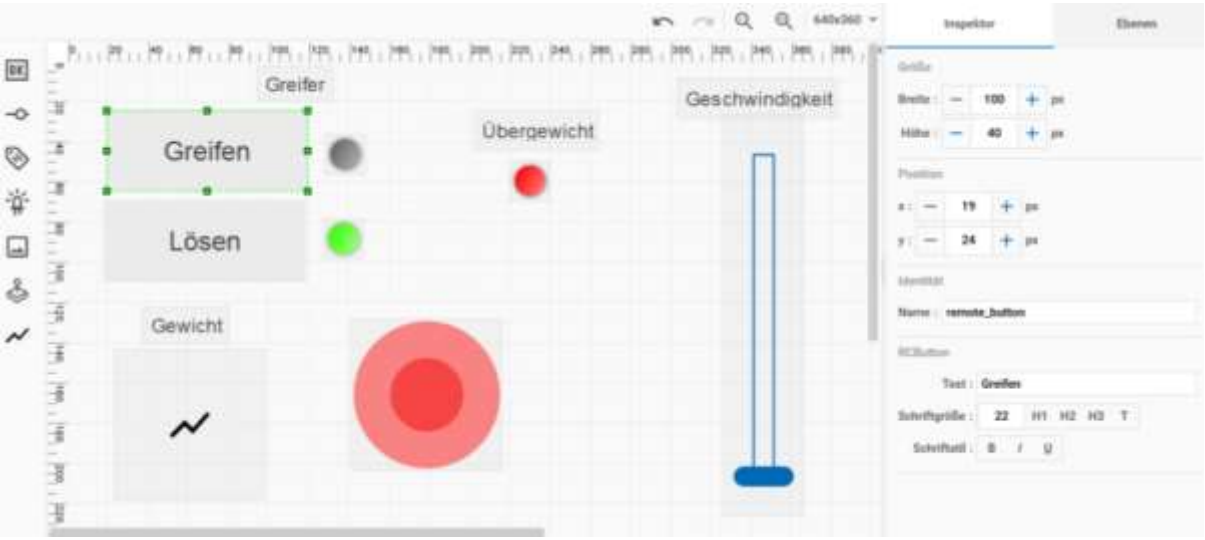
1. Man muss erst in der Bedienfeldkonfiguration ein Bedienfeld erstellen.

2. Programmieren, in dem man im Programm auf die Bedienköpfe und Regler reagiert und Anzeigen bedient.

Wenn man die Fernbedienung zu ersten Mal öffnet, sind die Blöcke ausgegraut und ein zusätzlicher Button erscheint:

„Bedienfeldkonfiguration hinzufügen“

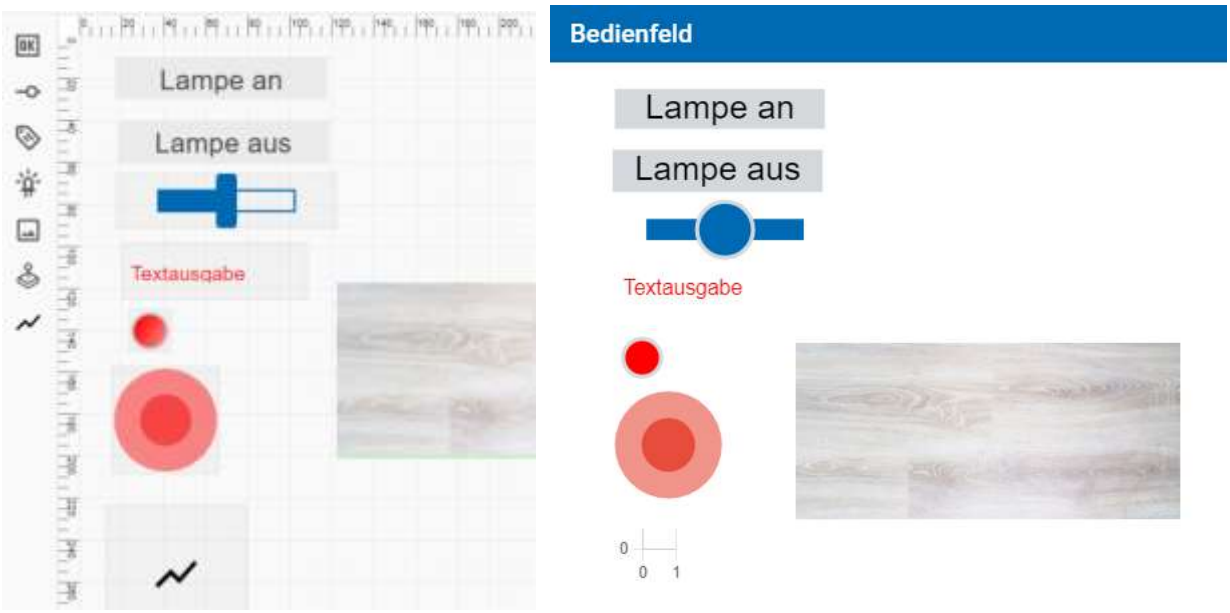
Beispiel eines Bedienfelds / Fernbedienung



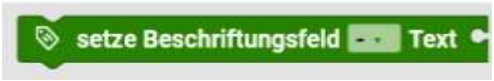
Hinweis: Der Debug-Modus ist nicht verfügbar, wenn das Programm eine Fernbedienung enthält.

Alle mit „Beispiel“ gekennzeichneten Teile gehören zusammen.
Über eine Benutzeroberfläche, können Fahrzeuge oder Modelle ferngesteuert werden.
Bevor Befehle ausgeführt werden können, dauert es derzeit bis zu 20 Sekunden, bis Aktionen möglich sind.

Beispiel:



setze Beschriftungsfeld … Text



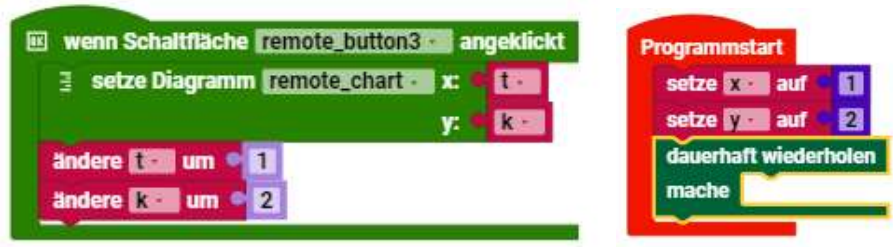
Für das Beschriftungsfeld einen Text ausgeben.

Beispiel:

Einerseits kann hier eine Beschriftung der Fernbedienung erfolgen. Andererseits können auch Rückmeldungen als Text erfolgen. (Siehe Beispiel). Zusätzlich zur Textgröße und Schriftstil, kann auch die Ausrichtung und Farbe im Inspektor voreingestellt werden.

setze Diagramm x;y

Beispiel:



Für das Diagramm wird bei jedem Druck auf „Zähler“ der aktuelle Wert dargestellt.

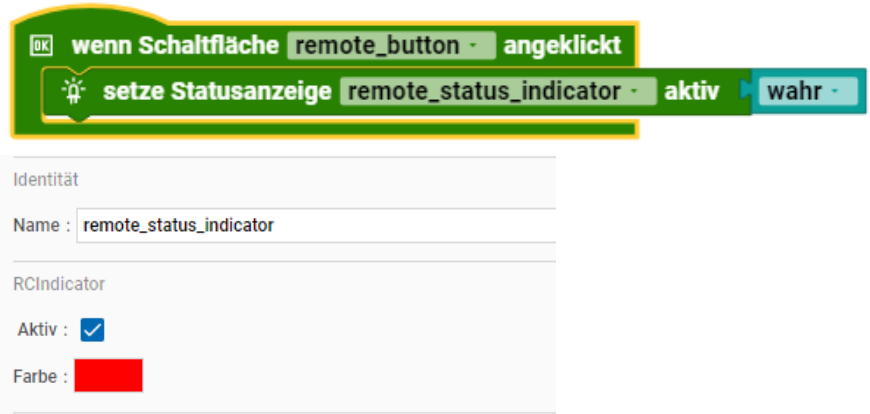
setze Bild… Base64-Bild

Eine zweite Möglichkeit, ist die Darstellung des Kamerabildes.
Abhängig von der Größe des Bildes,
kann es sehr lange dauern, bis das Bild dargestellt wird!!

Hiermit lässt sich ein Bild auf die Bedienoberfläche legen. Entweder als Detailbild zur besseren Erkennung oder sogar als Hintergrundbild des ganzen Bedienfeldes. Über das Kontextmenü kann dann die Ebene Hintergrund ausgewählt werden.

setze Statusanzeige aktiv

Beispiel:



Wie bei dem remote_label kann hier die Rückmeldung einer Aktion angezeigt werden. Oder man nutzt die Anzeige für Fehlermeldungen oder das Ende einer Aktion.

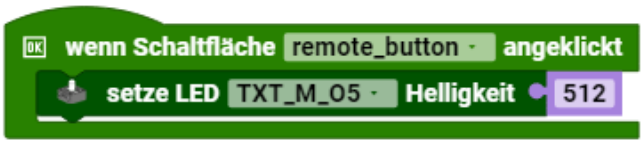
Ereignis Schaltfläche

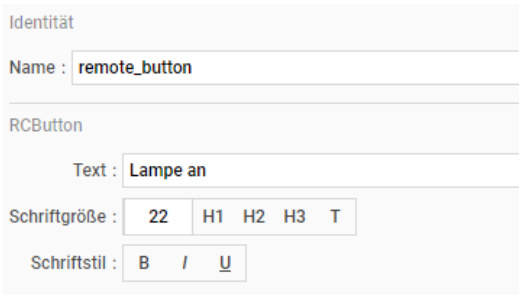
Man kann zwischen; Schaltfläche, Schieberegler und Joystick x/y Achse wählen.
Bemerkung:Eigentlich sollte der Anschluss links rund sein, da er auch die entsprechenden Werte weitergibt.

wenn Schaltfläche angeklickt

Unterprogramm wird ausgeführt, wenn Schaltfläche… angeklickt wird.

Beispiel:

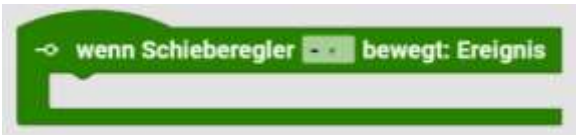




Durch Anklicken der Schaltfläche wird eine Aktion ausgeführt

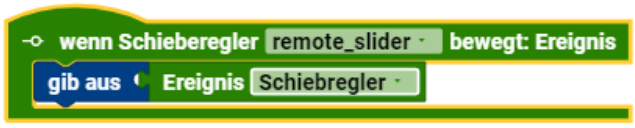
Schriftgröße und Schriftstil können im Inspektor festgelegt werden.

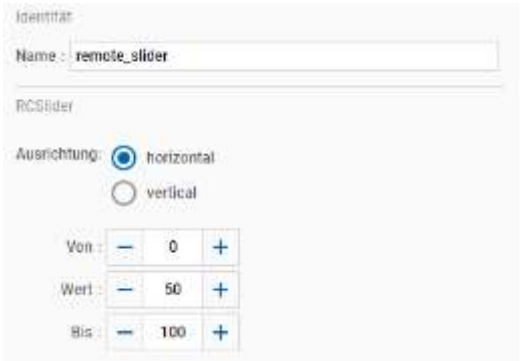
wenn Schieberegler…bewegt: Ereignis



Unterprogramm wird ausgeführt, wenn Schieberegler … bewegt wird.

Beispiel:

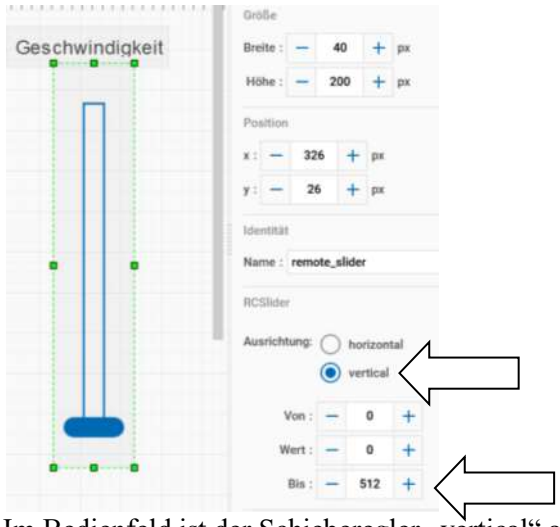




Die Bewegung des Joysticks kann in X und Y-Position ausgewertet werden. Im Inspektor können Anfangs und Endwert festgelegt werden. Im Beispiel erfolgt die Ausgabe auf der Konsole.

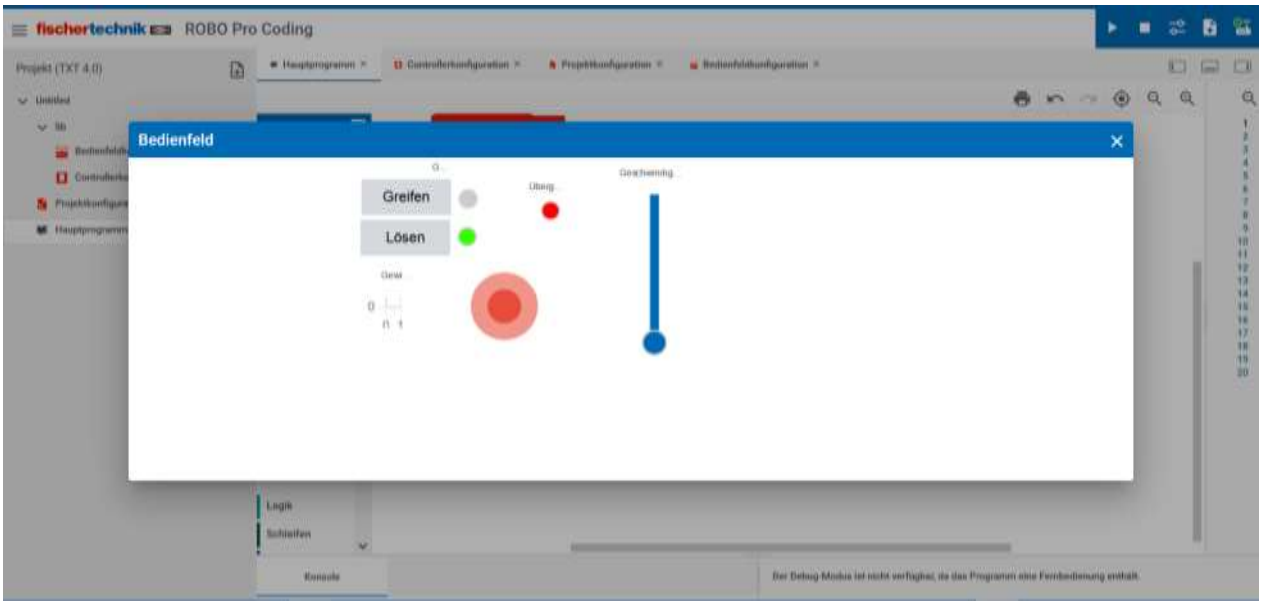
Die Bewegung des Sliders kann zur Steuerung von Motoren genutzt werden. Vorher werden der Wertebereich und der Anfangswert festgelegt. (hier Ausgabe auf Konsole)

Beispielprogramm:



Im Bedienfeld ist der Schieberegler „vertical“ angeordnet und „Bis“, also der obere Wert, ist auf 512 gesetzt. Somit kann man den gesamten Bereich der Motorgeschwindigkeit nutzen.

Das Unterprogramm reagiert auf eine Änderung vom Schieberegler und gibt den Wert als Geschwindigkeit an den Encodermotor.



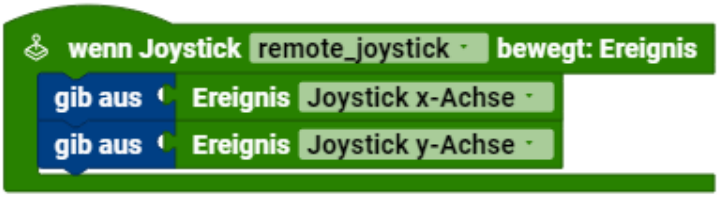
Beim Starten von Programm, wird hier nur der Schieberegler benutzt. In dieser Stellung ist der Encodermotor aus, da er den Wert 0 bekommt.

Hier hat der ca. den Wert 250. Somit dreht sich der Encodermotor mit „halber“ Geschwindigkeit. Ein Beenden des Bedienfeldes, beendet auch das laufende Programm.

wenn Joystick…bewegt: Ereignis

Unterprogramm wird ausgeführt, wenn Joystick… bewegt wird.

Beispiel:



Sprachsteuerung

Blöcke für die die Kommunikation mit der App „Voice Control“

Diese App kann kostenlos unter:
https://www.fischertechnik.de/de-de/spielzeug/apps-and-software
runtergeladen werden. Es sind Versionen für Android und iOS verfügbar.

Das Handy, Tablett… hat eine Spracheingabe, die genutzt wird, um über eine Verbindung zum TXT 4.0 Controller, den erkannten Text an den Controller zu senden. Dieser Vergleicht den Text nun im Programm und wenn er übereinstimmt, wird das weitere Programm ausgeführt.

Beispiele sind: Vorwärts, Rückwärts, Stopp, Links, Rechts, Drehen, Hoch, Runter, Widerhole, Tanze…

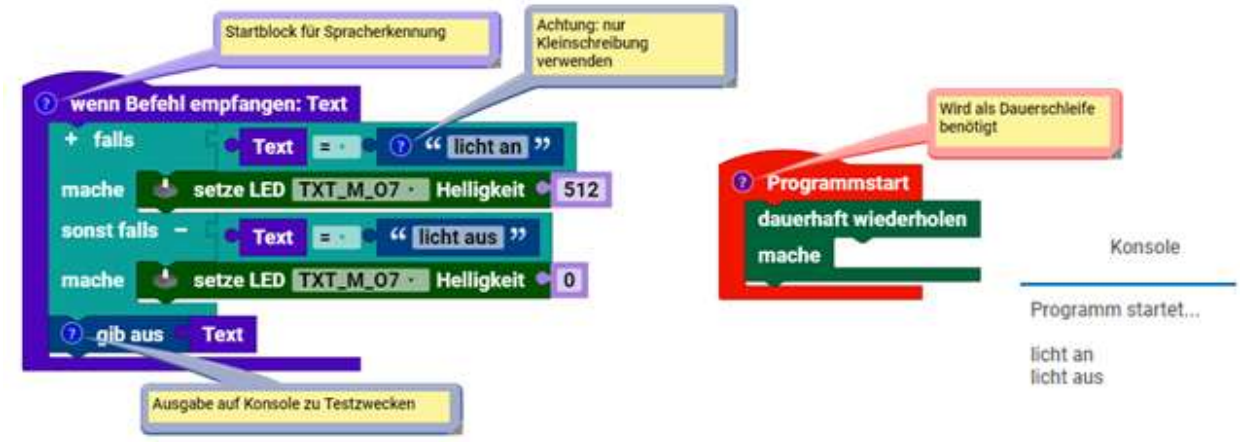
Symbol der fischertechnik App „Voice Control“

wenn Befehl empfangen: Text

Wird ausgeführt, wenn ein neuer Text von App „Voice Control“ empfangen wurde.

Text

Text ist der erkannte Sprachbefehl.



Um die Sprachsteuerung zu nutzen, ist die App „Voice Control fischertechnik“ auf dem Handy erforderlich. Anschließend meldet man sich auf dem TXT-Controller über WLAN an.

Der gesprochene Befehl wird wie im Beispiel ausgewertet und es erfolgt die entsprechende Aktion.

Cloud / MQTT

fischertechnik Cloud

Blöcke für die Kommunikation mit der fischertechnik Cloud.

Diese Blöcke werden für "Sensorstation IoT" und "Lernfabrik 4.0" verwendet. Die Daten werden an Unterverzeichnisse der Cloud gesendet und von dort aus z.B. an das Dashboard gesendet.

fischertechnik Cloud verbinden

Hier wird die Verbindung mit der fischertechnik Cloud aufgebaut.

mit fischertechnik Cloud verbunden

Wenn wahr, besteht eine Verbindung mit der fischertechnik Cloud.

fischertechnik Cloud trennen

Verbindung mit der fischertechnik Cloud trennen.

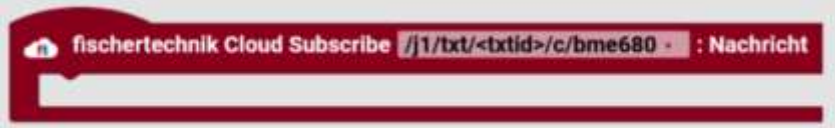
fischertechnik Cloud Publish Text

Textnachricht an die fischertechnik Cloud senden.

Man kann hier die Unterverzeichnisse auswählen. U.a. die fischertechnik Industrieanlage SLD (Sortieranlage) oder den Stock/Order Ordner, von wo aus die Daten an das Dashboard gesendet werden können.

Die „txtid“ siehe MQTT weiter unten.

fischertechnik Cloud Subscibe

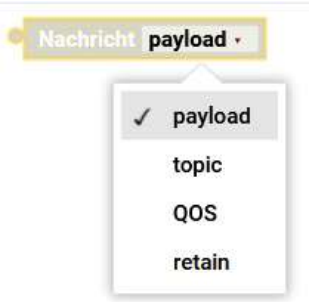


Topic abonnieren/Empfangen. Wieder mit der Pfadangabe wie oben.

MQTT Client
MQTT steht für "Message Queuing Telemetry Transport" Protokoll und wird oft bei IoT (Internet of Things) Anwendungen eingesetzt.

Nachricht

Eine MQTT Nachricht



Payload = Nachricht / Inhalt
Topic = „Variablennamen“ / kann auch einer Art Verzeichnisstruktur entsprechen
QOS = Quality of Service Sicherheitsstufe/ Art der Zustellung (0, 1, 2) = Einmalige Sendung (Schnell nicht garantierter Empfang), Synchron (mittlere Geschwindigkeit, evtl. mehrfache Sendung), garantierter Empfang (langsamste aber sicherste Verbindung)
Retain = gespeicherte Meldung, manchmal letzter gespeicherter Wert (z.B. von einer Raumtemperatur…)

MQTT Client erstellen: Websockets

Erstellt einen MQTT Client, mit dem Nachrichten empfangen und gesendet werden können. Es wird empfohlen die Ausgabe des Blocks in eine Variable zu schreiben, um den Client später in anderen Blöcken mehrfach verwenden zu können.

MQTT client ... connect

Verbindet einen MQTT Client mit einem MQTT Broker mit den angegebenen Einstellungen. Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Host gibt die Adresse des MQTT Brokers an. Um den lokalen MQTT Broker zu verwenden, wird als Wert localhost oder 127.0.0.1 eingetragen. Um externe MQTT Broker zu verwenden, muss dessen IP-Adresse oder Hostname verwendet werden. Port gibt den Port an, an welchem der MQTT Broker verfügbar ist. Standard ist 1883 für MQTT Broker (fischertechnik Cloud) oder 2883 (GUI Applikation). Bei Verwendung externer MQTT Broker muss dessen Port eingetragen werden. Username und Password sind standardmäßig leer, bei externen Servern müssen dessen Anmeldeinformationen hier eingetragen werden.

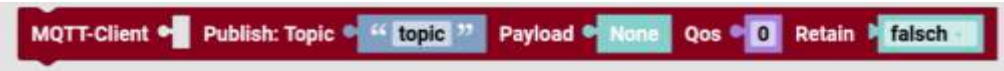
MQTT Client ... ist verbunden

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Gibt „true“ aus, wenn der angegebene MQTT Client mit einem MQTT Broker verbunden ist. Wenn der MQTT Client nicht verbunden ist, wird „false“ zurückgegeben.

MQTT Client ... trennt die Verbindung

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Trennt den angegebenen MQTT Client vom MQTT Broker.

MQTT Client ... Publish



MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Veröffentlicht mit dem angegebenen MQTT Client eine Nachricht „payload“ in einen angegebenen Kanal „topic“. Zusätzlich kann "qos" und "retain" angegeben werden, also ob neu verbundene Clients die Nachricht nach dem Senden empfangen sollen und mit welchem Sicherheitslevel die Nachricht gesendet werden soll.

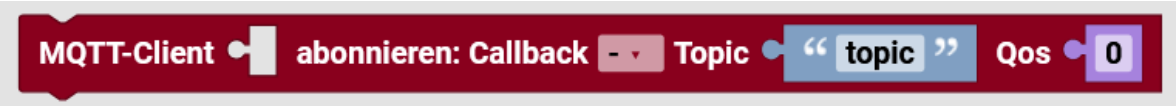
MQTT Client ... Publish (mit Rückgabewert)

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Veröffentlicht mit dem angegebenen MQTT Client eine Nachricht „payload“ in einen angegebenen Kanal „topic“. Zusätzlich kann "qos" und "retain" angegeben werden, also ob neu verbundene Clients die Nachricht nach dem Senden empfangen sollen und mit welchem Sicherheitslevel die Nachricht gesendet werden soll. Beim erfolgreichen Versenden wird „true“ ansonsten „false“ zurückgegeben.

MQTT Client ... Will Set

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Setzt die Nachricht „payload“, welche nach dem Trennen des MQTT Clients in einem angegebenen Kanal „topic“ versandt werden soll, und mit welchem Sicherheitslevel die Nachricht gesendet werden soll.

MQTT Client ... abonnieren



Abonniert mit einem MQTT Client einen Kanal. im Callback Argument wird die Funktion angegeben, welche beim Empfangen einer Nachricht ausgeführt werden soll. "Qos" gibt an, mit welchem Sicherheitslevel die Nachricht versendet werden soll.

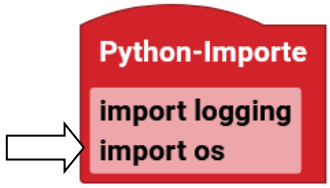
Callback abonnieren... : Nachricht

subscribe callback

Definiert eine Funktion, welche durch Empfangen einer Nachricht ausgeführt werden soll. Message enthält die empfangene Nachricht.

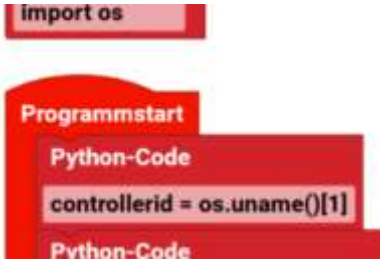
txtid Controller-ID

Die txtid ist eine Identifikationsnummer vom TXT 4.0 und wird für Zugriffe u.a. für die Cloud gebraucht. Man nimmt aus dem Reiter Util „Python-Importe und importiert dort „os“.



Import der LIB „os“

Im Programm wird die Controller-ID über den Python-Code gelesen.



Hier wird der Variablen „controllerid“ der Wert vom Betriebssystem übertragen.

Das Beschriftungsfeld erhält sein Text aus dem oberen Text<h4> (Überschriftenlevel 4) und „UI at:http://“ dem Inhalt der Variablen „controllerid“ und dem Text „.local:“ 1880 (Portnummer) „/ui und <h4> .

Die Programmausschnitte sind von Sorting_Line_AI.

Beispiel:

Die Cloud ist unter „https://www.fischertechnik-cloud.com“ zu erreichen. Bevor man mit diesem Dashboard arbeiten kann, muss eine Anmeldung erfolgen.

Dann bekommt man den Benutzernamen und Passwort. (Am besten aufschreiben / ausdrucken)

Achtung: Wenn die Seite nicht geladen wird, muss man die Tastenkombination [Strg]+[F5] drücken.

Anschließend muss der Controller mit der Cloud gekoppelt werden.
Am TXT 4.0 unter: Einstellungen / fischertechnik-Cloud / Neue Kopplung

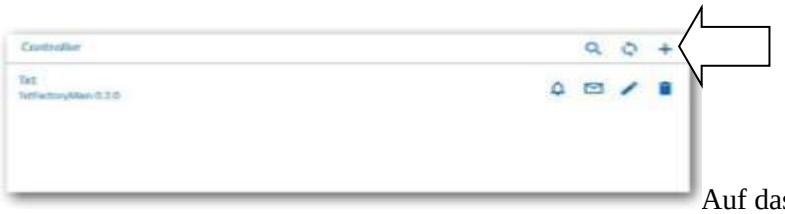
Es kommt eine Meldung auf dem TXT 4.0:

Keine Kopplungsdaten für die Cloud! Erstelle ein fischertechnik-Cloud Konto und starte Kopplung erneut. Kopplung Zeitüberschreitung Neue Kopplung

Wenn man auf „neue Kopplung drückt kommt folgende Meldung auf dem TXT 4.0

Man bekommt einen Kopplungscode, den man bei der Anmeldung bei der fischertechnik Cloud eingeben muss. Dieser Kopplungscode hat eine Gültigkeit von 30 Minuten. Danach muss man einen neuen Kopplungsvorgang starten.

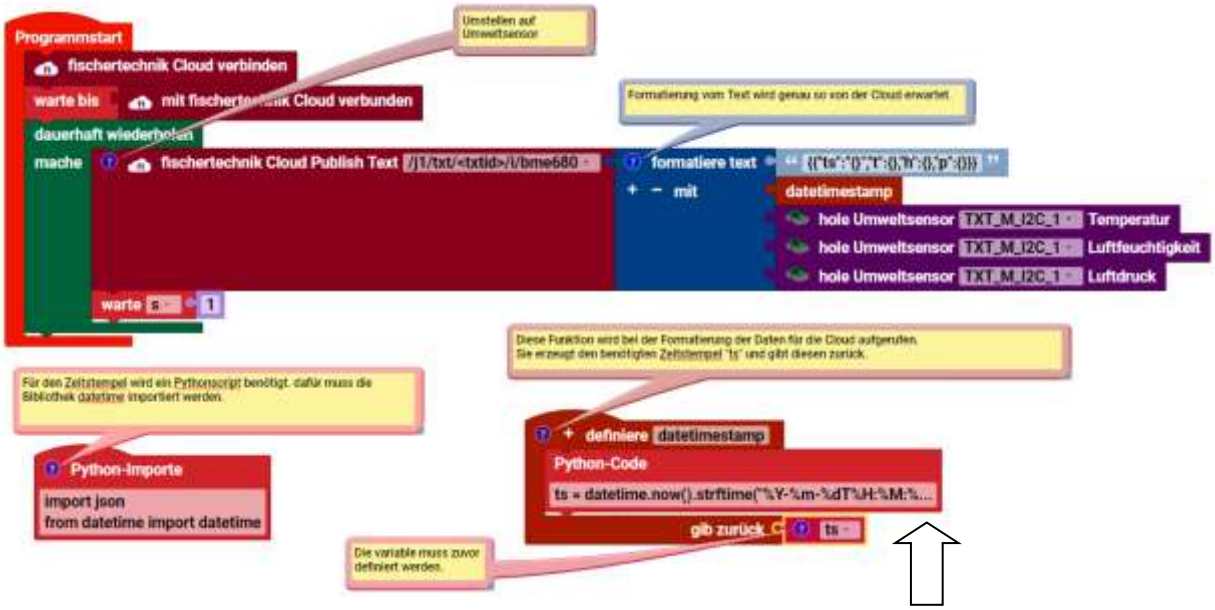
Man kann sich über den QR-Code in der Cloud anmelden oder auf der fischertechnik Cloudseite auf Einstellung / Controller hinzufügen gehen und den Kopplungscode dort eingeben.



Auf das Plus klicken

Weiter unten auf dem Display, hat man die Möglichkeit eine Kopplung wieder zu löschen.

Anmeldung auf der fischertechnik Cloud. Hier muss man den Kopplungscode eingeben.



Der vollständige Eintrag: ts=datetime.now().strftime("%Y-%m-%dT%H:%M:%S.%f")[:-3] + "Z"

Die Daten von „ts“ haben folgende Notation: YYYY-MM-DDThh:mm:ss.fffZ (fffz=Rundung)

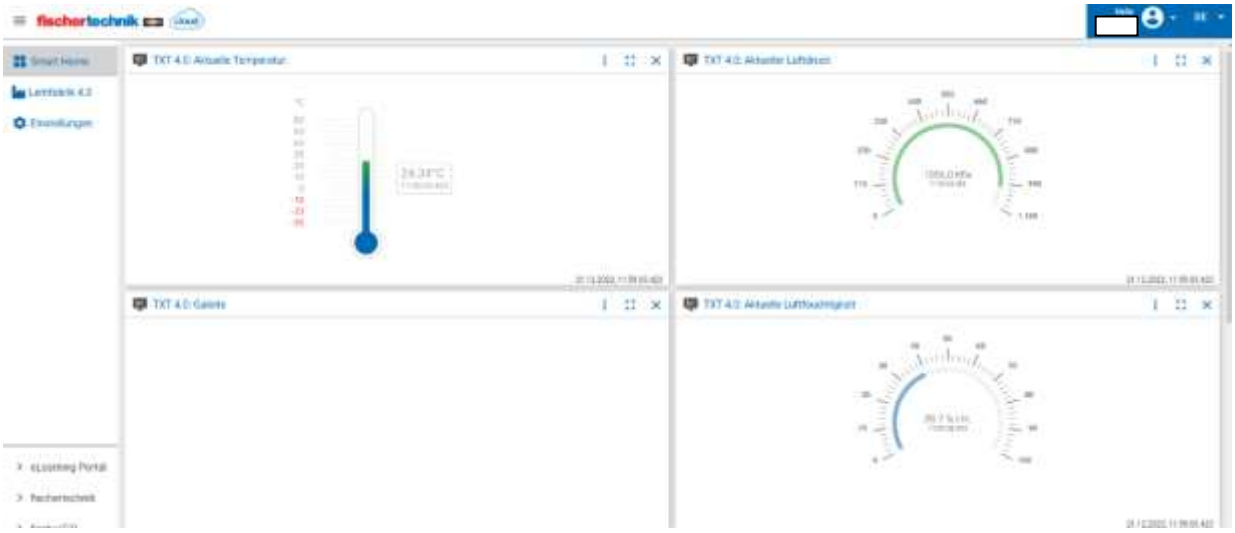
Die Daten werden an ein Unterverzeichnis /ibme680 in der Cloud gesendet (Publish) und dann wieder von Dashboard dort ausgelesen (Subscribe). Es können auch von mehreren TXT 4.0 Daten an die Cloud gesendet und empfangen werden. Die Unterscheidung ist dann mit der Controller ID.

Programmablauf:

Das Programm auf dem TXT 4.0 startet mit dem Initialisieren des Zugangs zur fischertechnik Cloud. Es wir gewartet, bis eine Verbindung aufgebaut ist. In einer Dauerschleife werden 4 Daten eingelesen:
1x „datetimestamp“ (Zeitstempel) und 3x Umweltsensor.
Die Daten werden formatiert und mit weiterem Text versehen. Die Reihenfolge ist von oben nach unten. Dieser Datenstring wir in die fischertechnik Cloud in das Unterverzeichnis „/i/bme680“ gesendet. Es wird eine Sekunde gewartet, bevor ein weiterer Durchgang startet.

Über Python-Importe werden die LIBs „json“ und „datetime“ importiert. Im Unterprogramm „datetimestamp“, wird aus dem Datenstring, eine formatierte Datum-Zeit Variable „ts“.

Wenn alles klappt, sollten die 3 Werte (Temperatur, Luftfeuchtigkeit und Luftdruck) in den jeweiligen Fenstern der Cloud im Sekundentakt angezeigt werden.



Anzeige auf dem Dashboard, hier die von Smart Home.

Soweit ich das sehe, wird beim TXT 4.0 der gleiche Datenaustausch vorgenommen, wie bei den Modellen mit dem Vorgänger TXT-Controller. Somit kann man die MQTT-Beschreibungen / Erklärungen übernehmen.
Wie z.B.: https://github.com/fischertechnik/txt_training_factory/blob/master/README_de.md

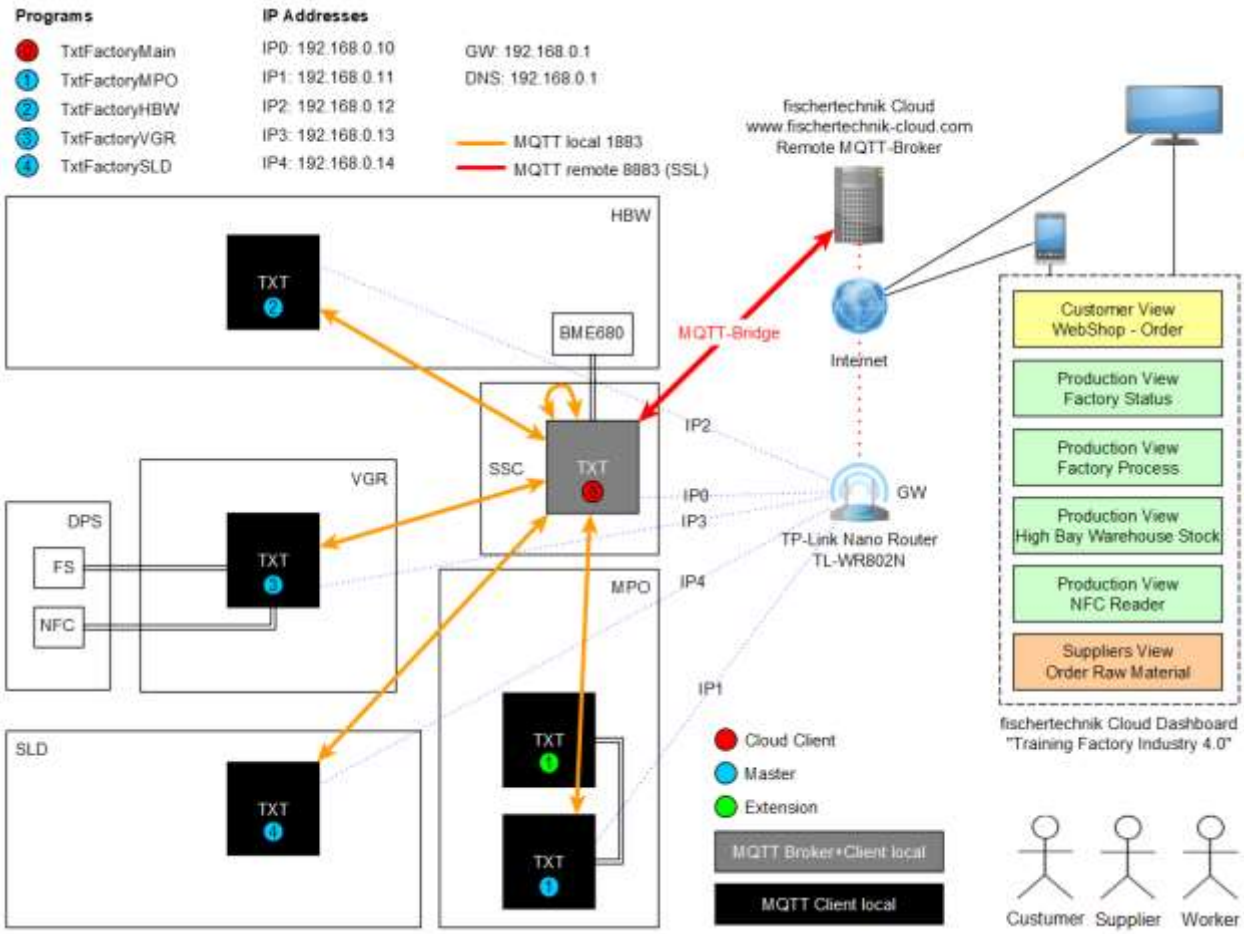
MQTT Interface Local Clients - Zugangsdaten

host: 192.168.0.10
port: 1883
user: txt
password: xtx

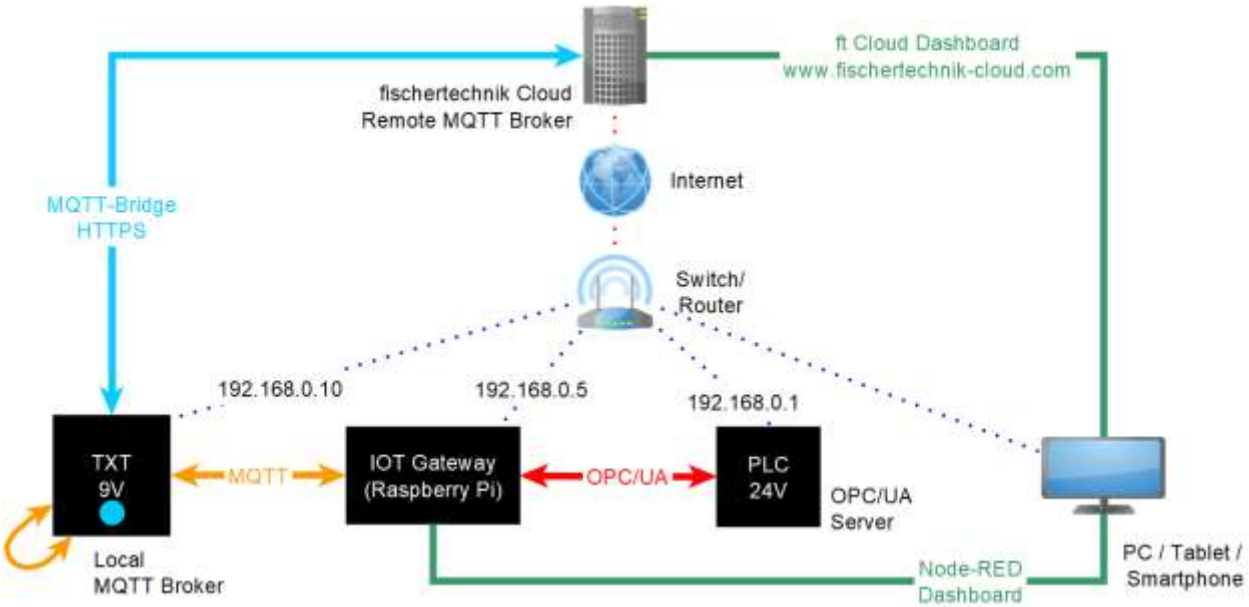
IOT Gateway (Raspberry Pi 4) (mit fischertechnik SD Card Image)

User: pi
Password: ft-IOTpi2

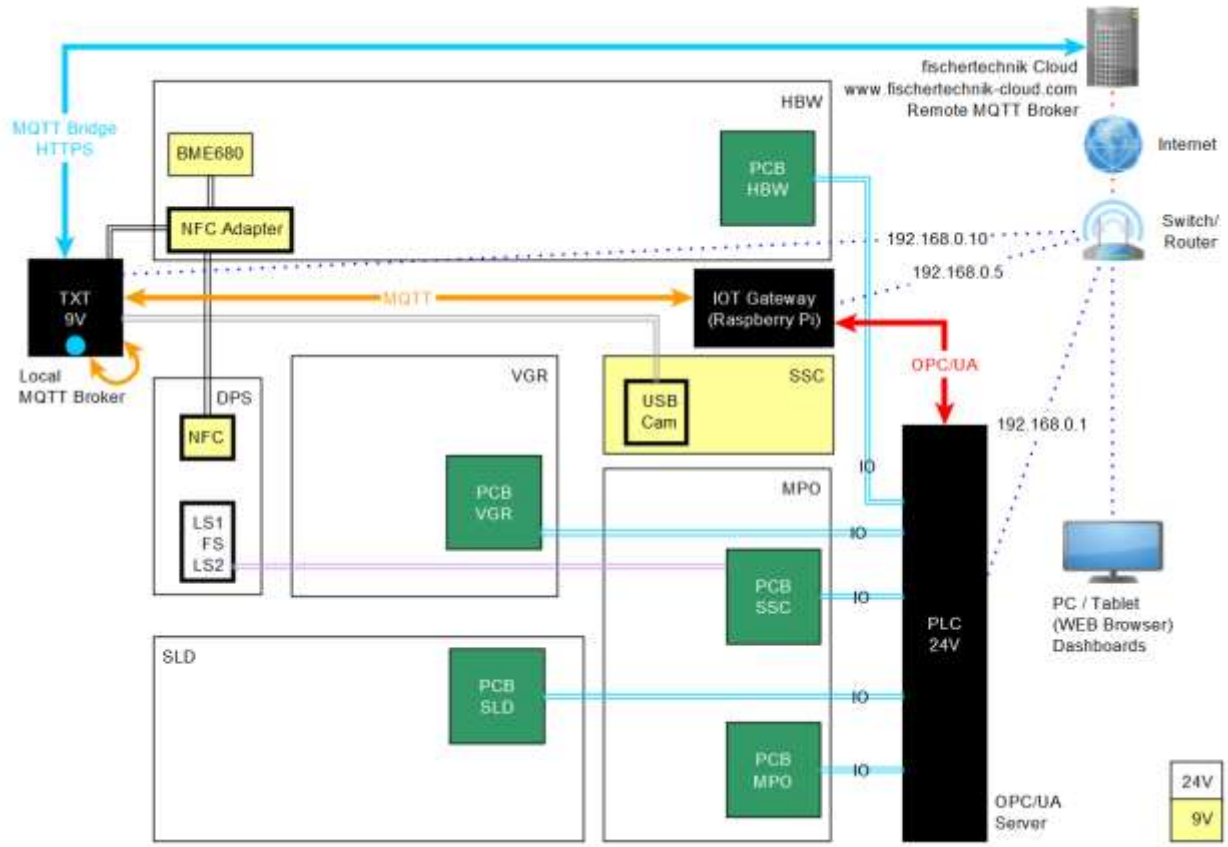
MQTT Übertragungen in fischertechnik Modellen



Übersicht der Industrieanlage 9V mit den Übertragungswegen und TXT (!) Controllern.



Die Übersicht der Kommunikation des TXT 4.0 in der 24V SPS Industrie 4.0 fischertechnik Anlage.



Hier die 24V Anlage mit SPS und einem TXT 4.0 in der Übersicht.

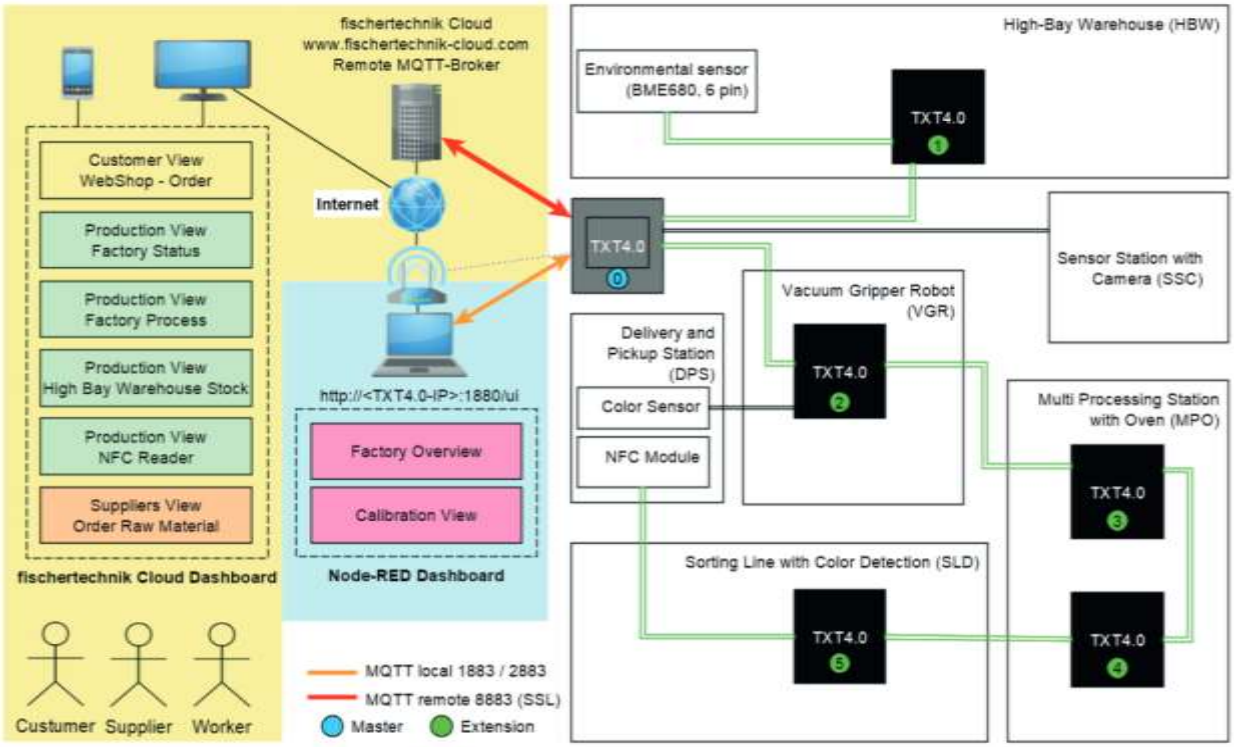
Komponenten:

SSC: Sensor Station with Camera
HBW: High-Bay Warehouse
VGR: Vacuum Gripper Robot
DPS: Delivery and Pickup Station
MPO: Multi-Processing Station with Oven
SLD: Sorting Line with Color Detection

Die gelben 9V Teile, sind über USB mit dem TXT 4.0 verbunden. Die MQTT-Kommunikation verläuft über den Raspberry Pi und dann über den Router. Die blaue ist nur eine virtuelle Darstellung derselben Verbindung.

Welche Station was sendet und wie genau der Payload aufgebaut ist, kann man hier einsehen: https://github.com/fischertechnik/txt_training_factory/blob/master/TxtSmartFactoryLib/doc/MqttInterface.md

Es gibt eine zweite neuere Version der Fabrik nur mit TXT 4.0 Controllern.



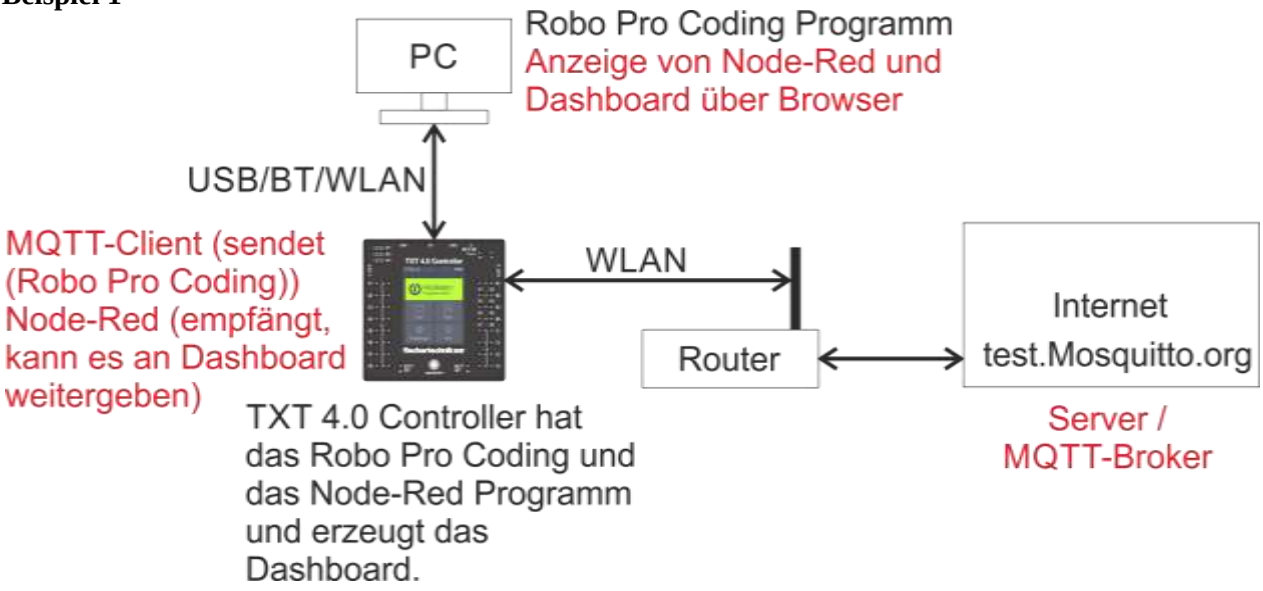
Lernfabrik TXT 4.0 9V

Es gibt nur einen TXT 4.0, wo alle Programme darauf laufen. Der rote und der orange Pfeil geben den MQTT Datenaustausch an.

Beim fischertechnik Industriemodell KI Qualitätssicherung, werden auch MQTT Befehle für den Datenaustausch genutzt. (Sie Modell und Programmbeschreibung weiter hinten im Buch)

MQTT Beispiele

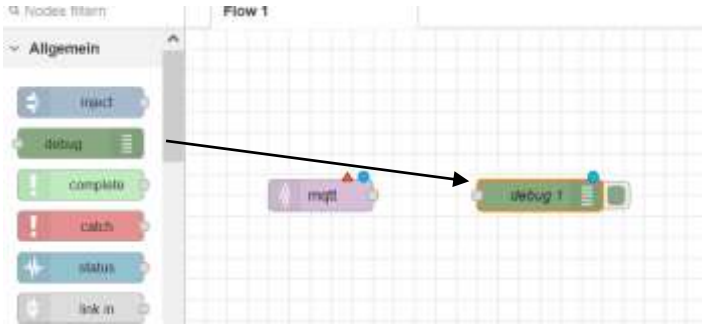
Beispiel 1



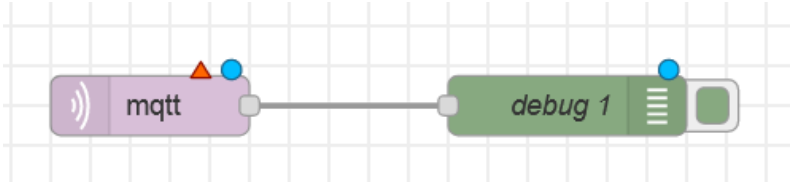
Node-Red Einstellungen am TXT 4.0 / PC um MQTT zu senden und zu empfangen
Der TXT 4.0 muss mit dem WLAN verbunden sein (s. TXT 4.0 Anleitung WLAN).
Man muss über Info / WLAN / IP die IP Adresse ermitteln. Beispielsweise 192.168.0.100
Jetzt muss man noch unter Einstellungen / Services / Node-Red den Schalter am Display einschalten.
Nun im Browser diese Adresse und den Port eingeben: Hier z.B. 192.168.0.100:1880
Man muss den Benutzernamen: ft und das Passwort: fischertechnik eingeben.
Dann erscheint der Node-Red Willkommensbildschirm. Den wir schließen.

Links, unter Netzwerk, gibt es „mqtt in“. Den ziehen wir in die Mitte. Das ist der „Flow“. Wir haben nun schon mal einen Eingang. An dem Symbol sind zwei Zeichen, die uns sagen, dass da noch Angaben fehlen und es noch nicht gestartet ist.

Oben rechts unter Allgemein ist das zweite Zeichen „debug“, dass wir rechts neben das erste rüber ziehen.

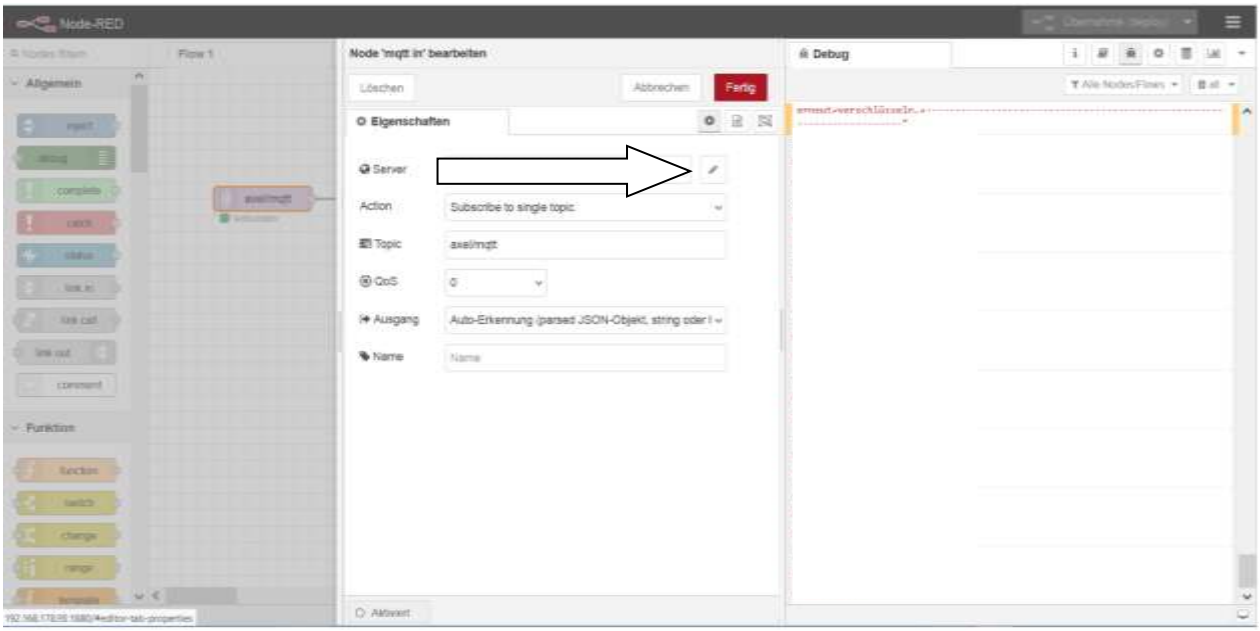


Nun verbinden wir die beiden Symbole



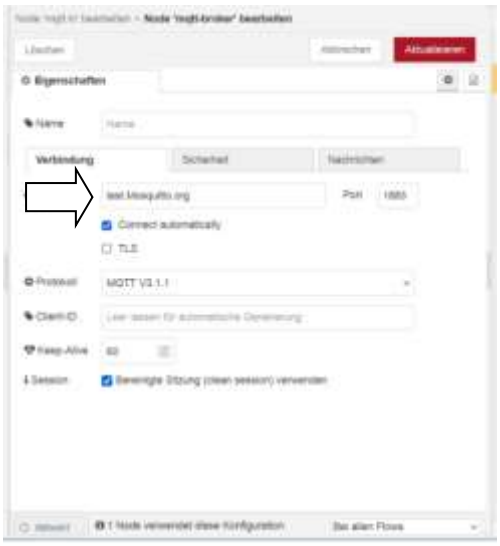
Dazu auf den linken Kreis klicken und zum rechten rüber ziehen.

Nun machen wir einen Doppelklick auf das mqtt-Symbol und man kann nun Einstellungen vornehmen

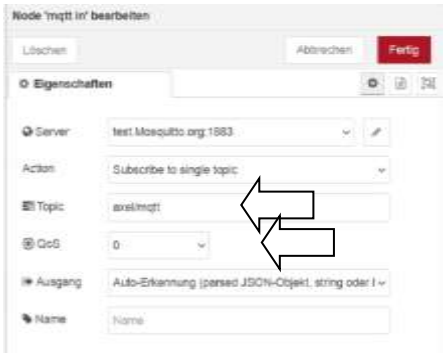


Nun rechts neben Server auf den Bleistift, für Bearbeiten, klicken.

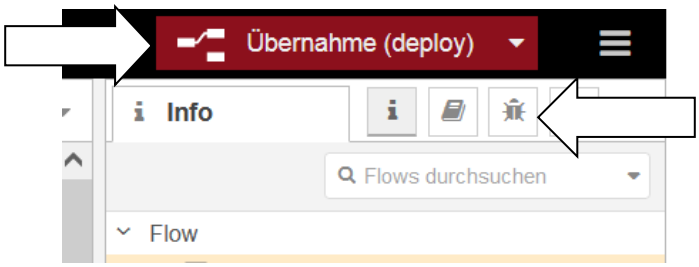
Hier muss man **test.Mosquitto.org** eingeben. Das ist nur der Name. Das „:1883“ ist rechts davon und steht unter Port 1883. Nun auf das rote Hinzufügen klicken.



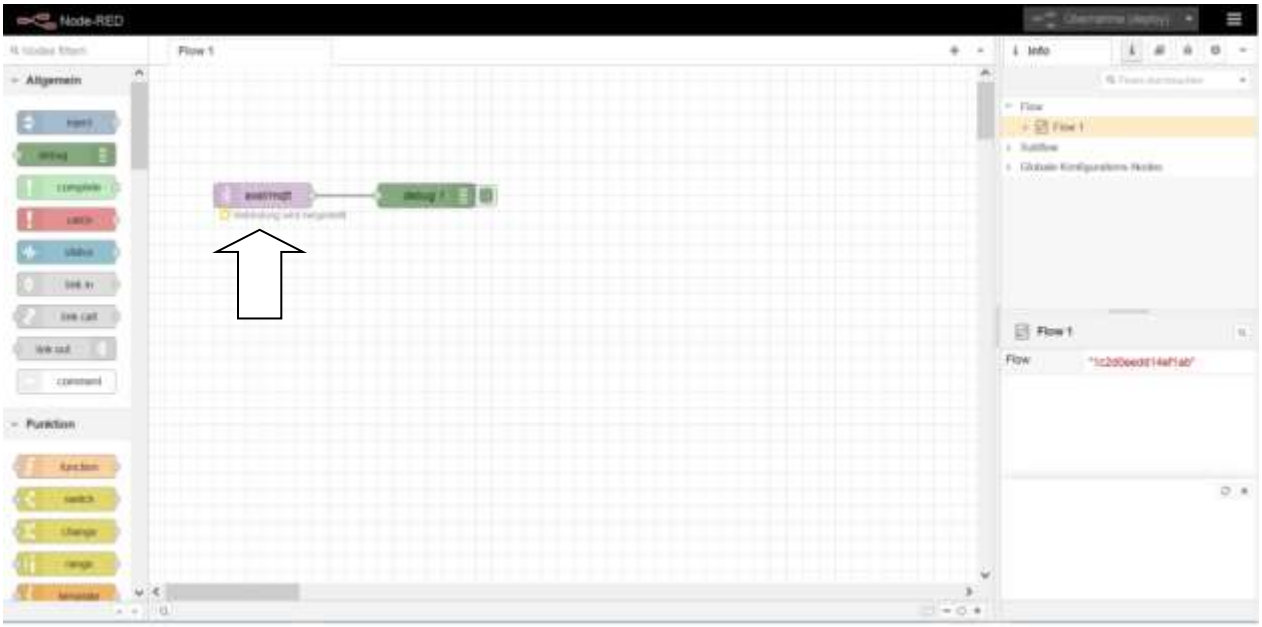
Und auf das rote Aktualisieren klicken.



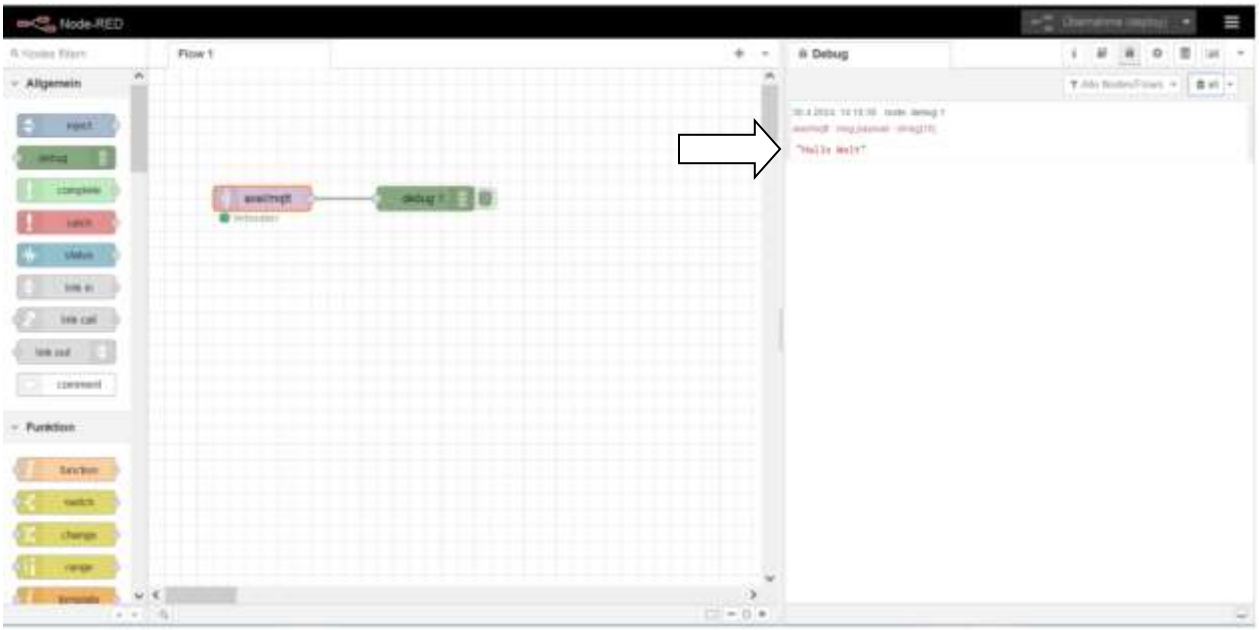
Unter Topic tragen wir axel/mqtt ein. Das ist die Nachricht, die wir empfangen wollen.
Und bei Qos wählen wir die Null aus. Dann klicken wir auf das rote Fertig.



Nun auf Übernahme und oder auf den Pfeil und dann „Flows neustarten“ anklicken. Dann auf das grüne Debug-Element klicken und oben auf den kleinen Käfer (Debug) klicken, um den Inhalt sich anzeigen zu lassen.



Wenn hier nur „Verbindung wird hergestellt“ liegt ein Schreibfehler vor.



Hier ist das mqtt-Element mit dem Server verbunden. Wenn man das Programm auf dem TXT 4.0 oder PC startet, erscheint rechts ein Zeitstempel mit dem Payload „Hallo Welt“.

Immer wieder, wenn man das Programm startet, erscheint die Meldung,

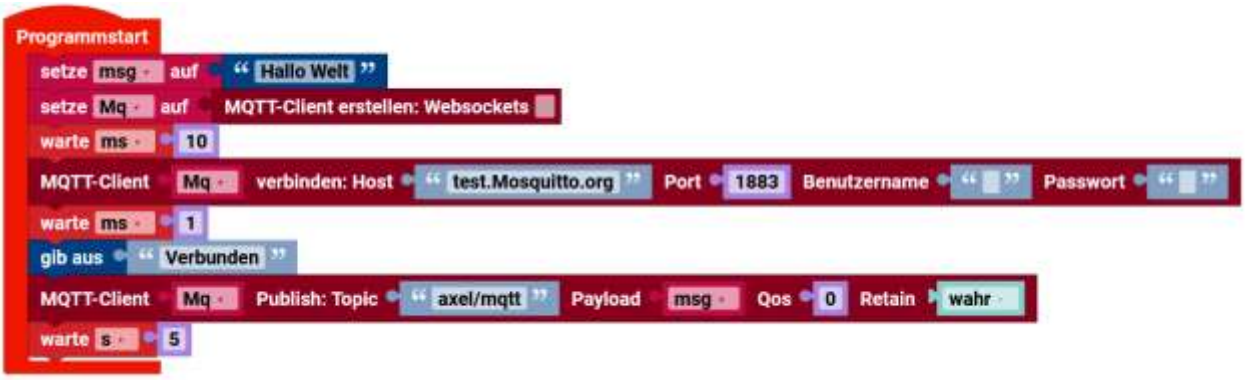
Statt nun wie in den vorangegangenen Beispielen nur Text zu übertragen, kann man auch Messwerte wie Temperatur oder auch Roboter-Positionen übertragen.

MQTT Das Robo Pro Coding Programm
Grundlagen

Ein Programm (Client) sendet Daten an einen Server(Broker). Das passiert mit Überschriften (Topic) und den Daten(Payload). Ein anderes Programm (Client) abonniert nun dieses Topic und bekommt dadurch die Daten.

Beispiel 1

Hier ist der Client ein TXT 4.0, der zu einem Testserver (Mosquitto) ein Topic (axel/mqtt) mit dem Payload (Hallo Welt) sendet. Die Daten werden dann vom Testserver abgerufen. Dieser sendet daraufhin das Topic mit dem Payload wieder zurück an den TXT 4.0. Node-Red auf dem TXT 4.0 reagiert auf das MQTT-Topic und gibt es an das Element Debug weiter. Die Konsole vom Debug zeigt an, welche Daten ankommen.



Man muss zur Erstellung des Programmes, zwei Variablen erzeugen: msg für Masage und Mq als Hilfsvariable für die MQTT Übertragung.
Dann muss ein Text ausgesucht werden der übertragen werden soll. Hier „Hallo Welt“.
Im Reiter Cloud/MQTT ist dann „MQTT-Client erstellen: Websockets“ Man erzeugt einen Client.
Dieser MQTT Client muss nun Verbindung aufnehmen. Das macht er, in dem man ihm sagt, dass es eine Web-Anwendung ist und im ersten Teil die Variable Mq einfügt.
Statt dem vorgegebenen „localhost“ tippen wir „test.Mosquitto.org“ ein. Das ist eine Test-Webseite vom Mosquitto-Server. Sie gibt den Topic zurück, den man da hin sendet. Sie braucht keinen Benutzernamen oder Passwort.
Wichtig ist hier eine kleine Zeitverzögerung einzugeben. Bei Tests hat sich rausgestellt, das sonst nicht oder selten gesendet wird.
Wenn das geklappt hat, soll Verbunden auf der Konsole ausgegeben werden.
Nun senden wir unseren Topic an den Server. Unser Topic hat den Namen axel/mqtt und den Inhalt (Payload) der Variablen msg, also „Hallo Welt“. Qos gibt die Wichtigkeit der Sendung an. Bei 0 kann die Sendung auch mal verloren gehen. Retain ist hier wahr. Die Nachricht wird auf dem Server gespeichert. (Ist hier aber ohne Belang.)

Wenn man dieses Programm startet, Sieht man erst mal wenig auf dem PC. Es erscheint:

Konsole
Programm startet…
Verbunden
Programm beendet.

Die Ausgabe erfolgt im oberen/vorherigen Node-Red Fenster.

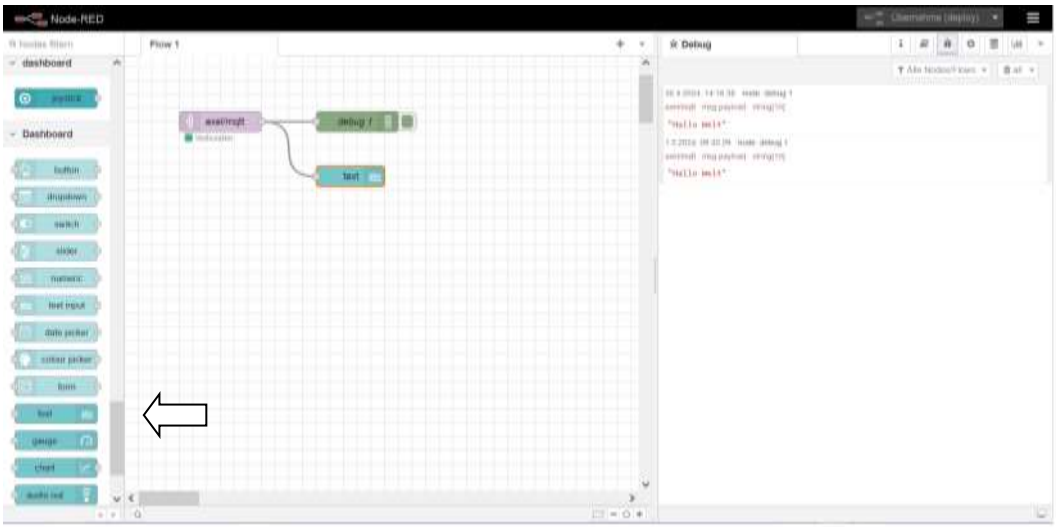
Beispiel 2

Kleine Abwandlung vom Programm Beispiel 1

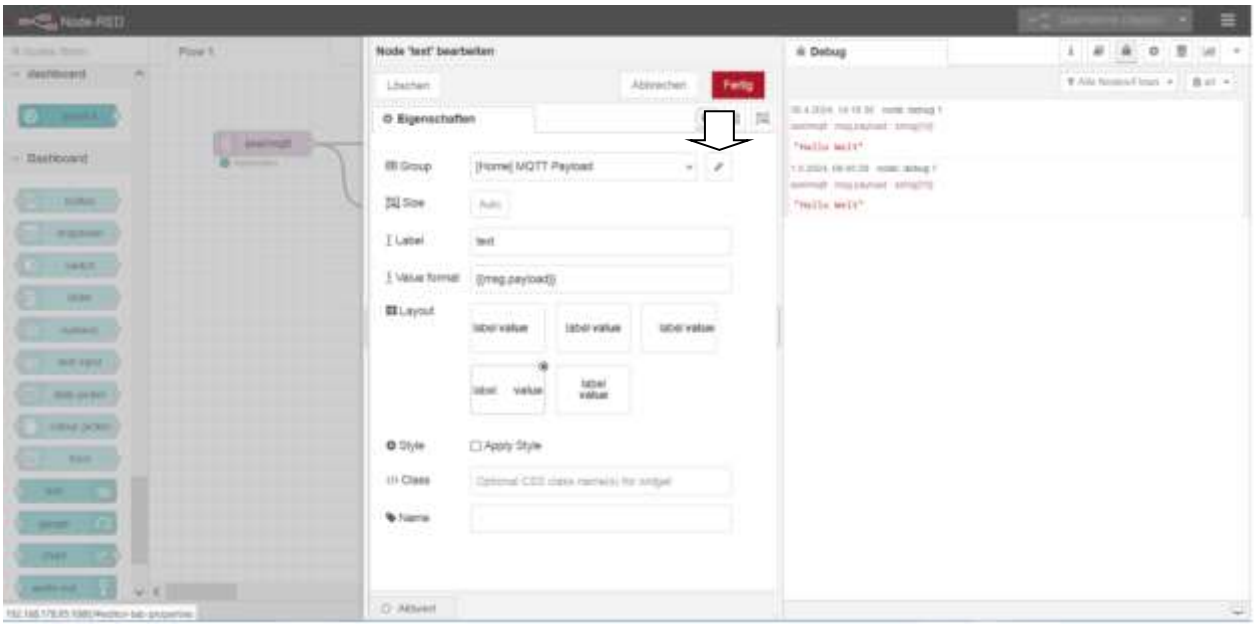
Hier wird nun 10 Mal alle 5 Sekunden das Payload gesendet.

Entsprechend steht die Meldung „Hello Welt“ nun mehrfach in der Debug-Konsole.

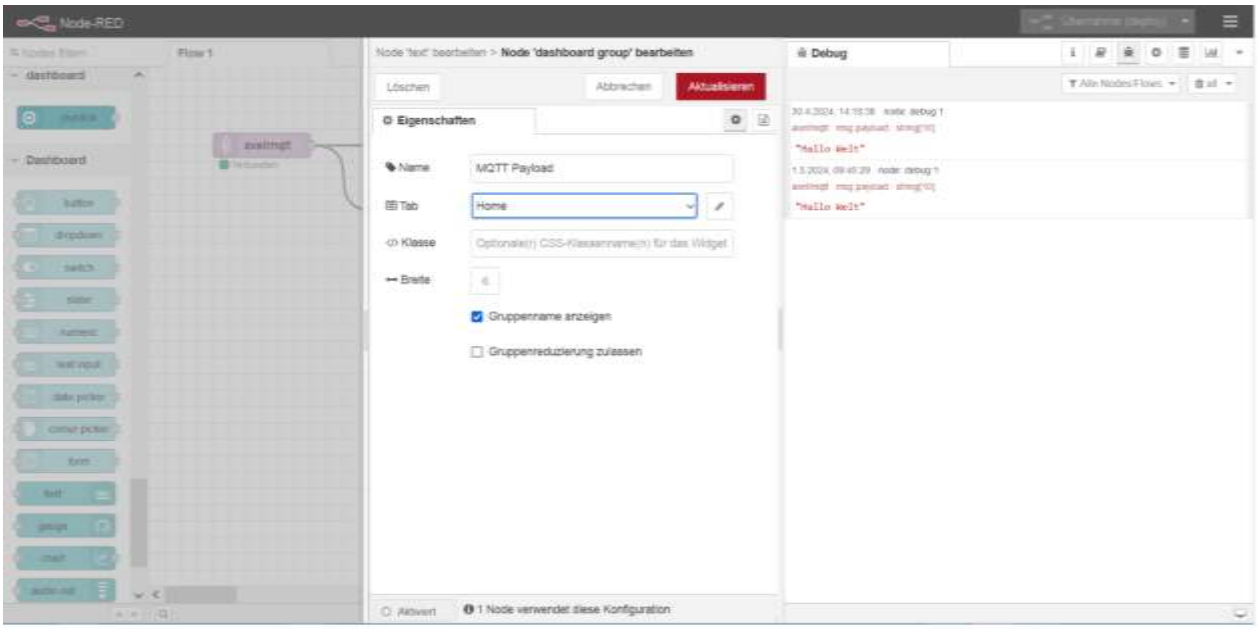
Dashboard Anzeige



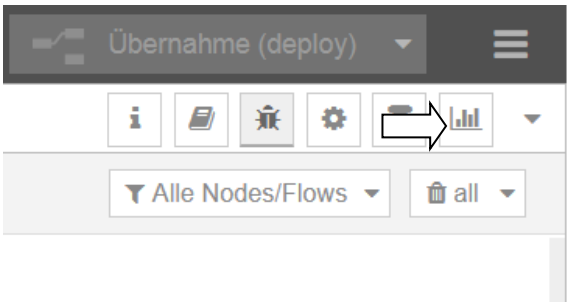
Hier ist zusätzlich zum ersten Flow, ein Abzweig zu einem Dashboard-Element „text“. Er wird einfach mit dem linken Element verbunden.



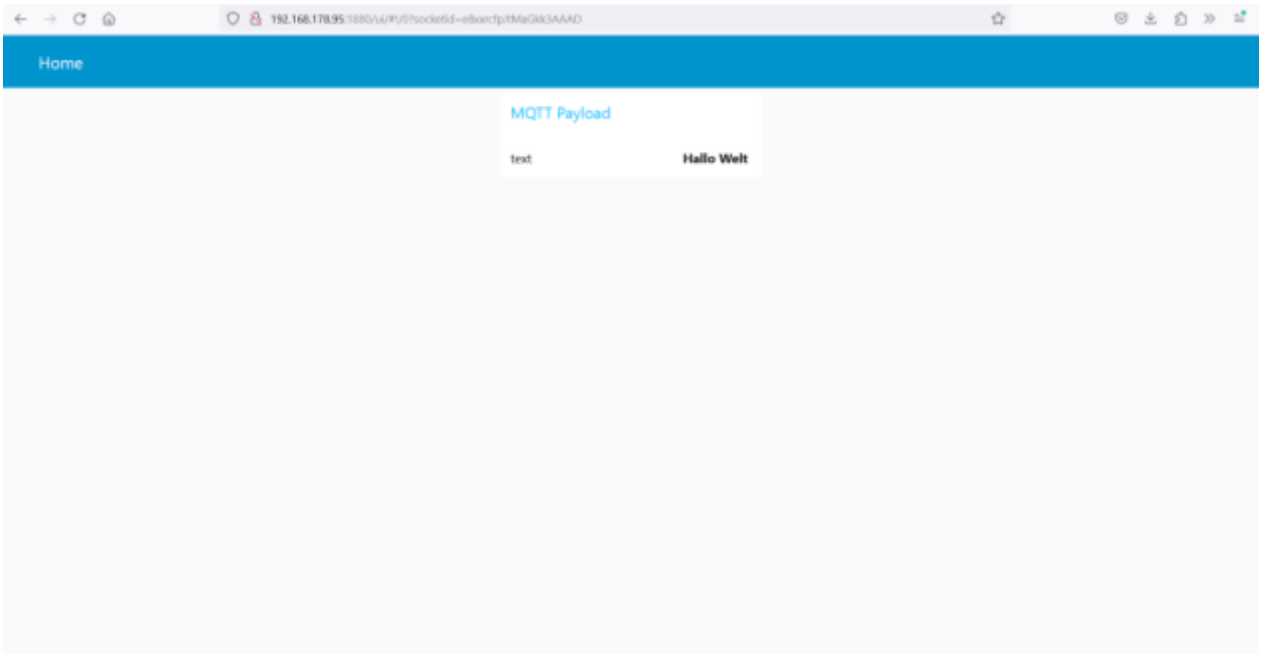
Durch einen Doppelklick auf das Element „text“ kann man die Eigenschaften ändern. Hier nun auf den Bleistift klicken.



Als Name habe ich hier „MQTT Payload“ und bei Tab „Home“ genommen. Nun das rote Aktualisieren und Fertig anklicken und danach Übernehmen oben rechts.



Mit einem Klick auf das Symbol Dashboard (oder den kleinen Pfeil rechts) wechselt man auf das Dashboard.



Der Payload vom MQTT „Hallo Welt“ wird in der Textanzeige vom Dashboard angezeigt. Man kann auch text und MQTT Payload ändern, um so z.B. Wetterdaten anzeigen zu lassen, die vom TXT 4.0 gemessen werden.

Es gibt auch andere Elemente wie ein Messinstrument… womit man die Temperatur schöner anzeigen lassen kann.

Auf YouTube gibt es Videos zu dem Thema. Unter anderem:
https://www.youtube.com/watch?v=_jMyAnPjkLQ **NodeRed-Tutorial Part 2** (Auch die anderen Teile).
Sehr viel kann man direkt mit dem TXT 4.0 ausprobieren und nachvollziehen.

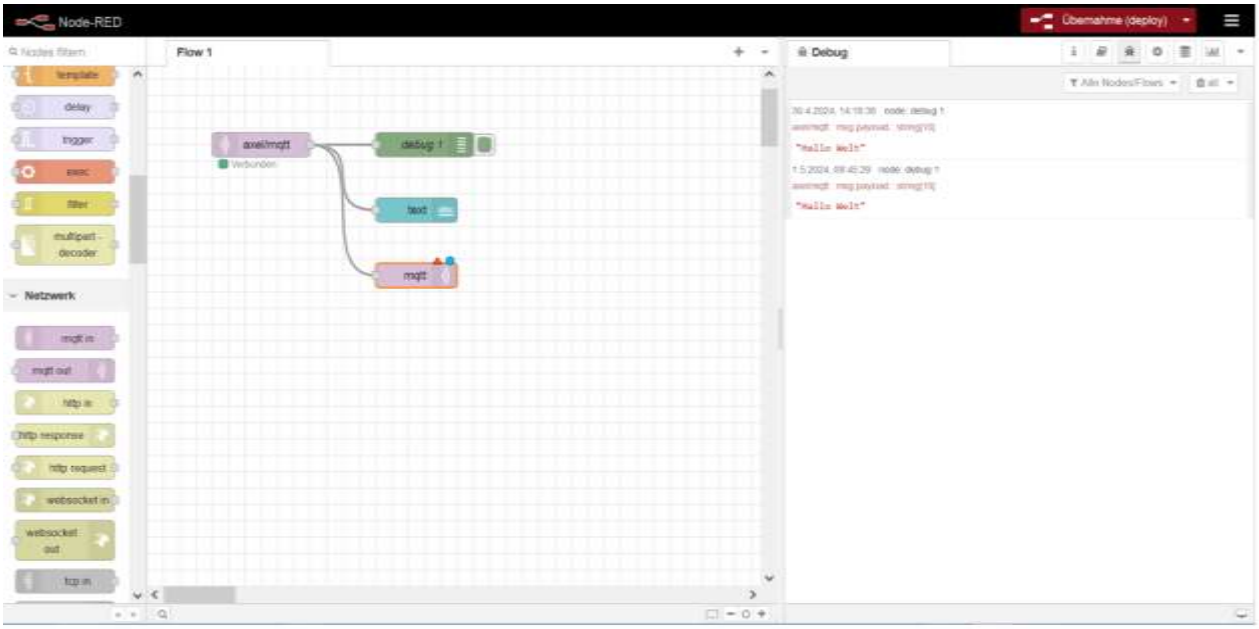
Übrigens.
Wenn man den TXT 4.0 ausschaltet, versucht Node-Red jede Minute wieder Verbindung mit dem Server aufzunehmen. Wenn man den TXT 4.0 wieder einschaltet und ihn in Robo Pro Coding verbindet, ist die Meldung in Node-Red weg und Node-Red hat die Verbindung zum Server/Broker wieder.

MQTT-Broker auf dem TXT 4.0

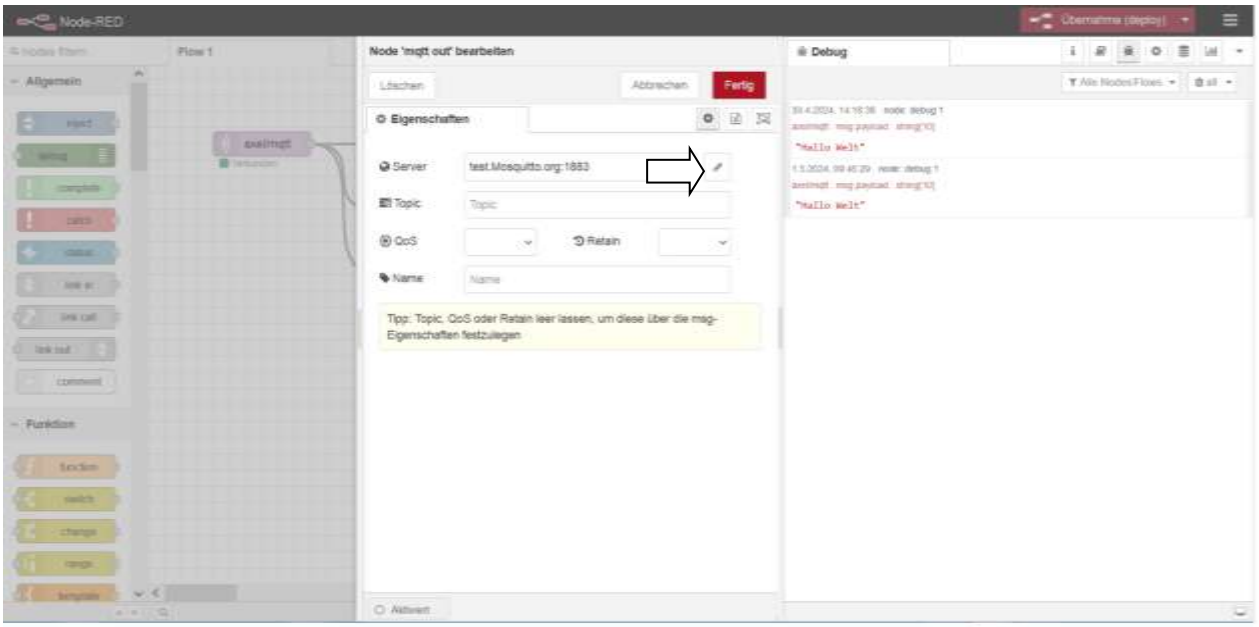
Es gibt einen bisher undokumentierten MQTT-Broker auf dem TXT 4.0. In den vorherigen Beispielen wurde auf den Mosquitto Server im Internet zugegriffen um Daten schreiben und lesen zu können.
Das Robo Pro Programm hat das Schreiben übernommen und das Node-Red Programm das Lesen.

Aus Node-Red heraus, ist aber auch möglich zu senden. Dieser Broker hat die Adresse 127.0.0.1:1883 .

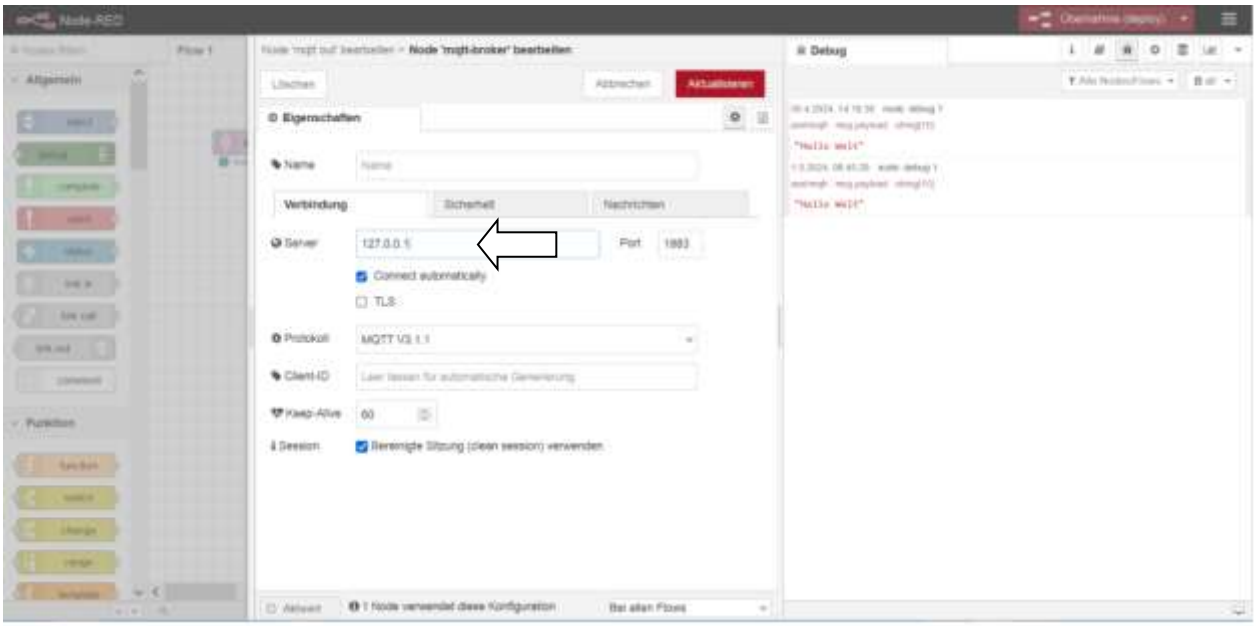
Wir Erweitern das Beispiel 2 von oben, indem wir unter Netzwerk das Symbol „mqqt out“ rüber zieht und mit dem linken Element verbindet.



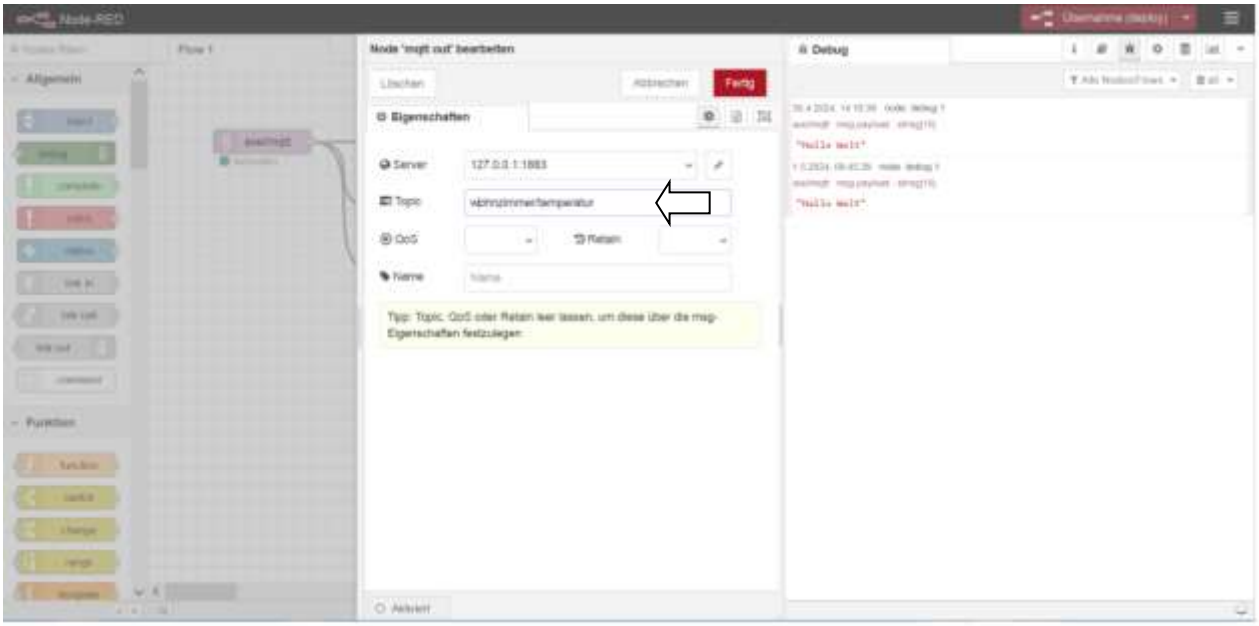
Ein Doppelkick auf das neue mqtt-Element (Sender).



Und dann auf den Bleistift, kommt man zu:

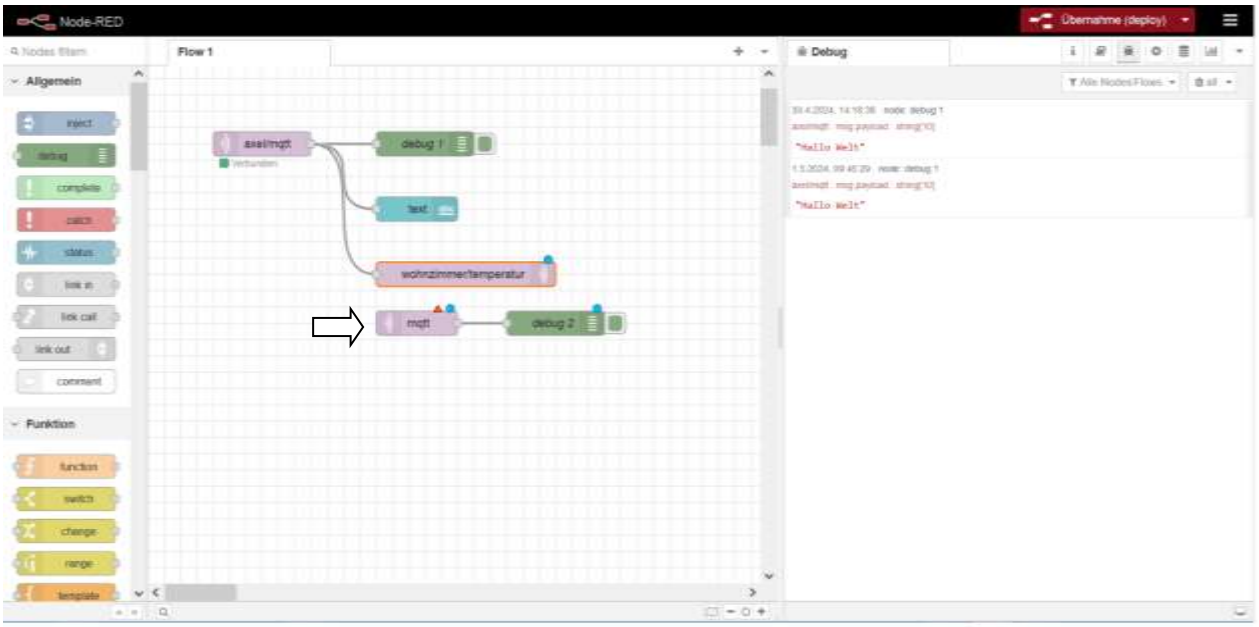


Statt der test.Mosquitto.org-Adresse, muss man 127.0.0.1 eingeben. Der Port 1883 bleibt derselbe. Dann wieder auf Aktualisieren klicken.

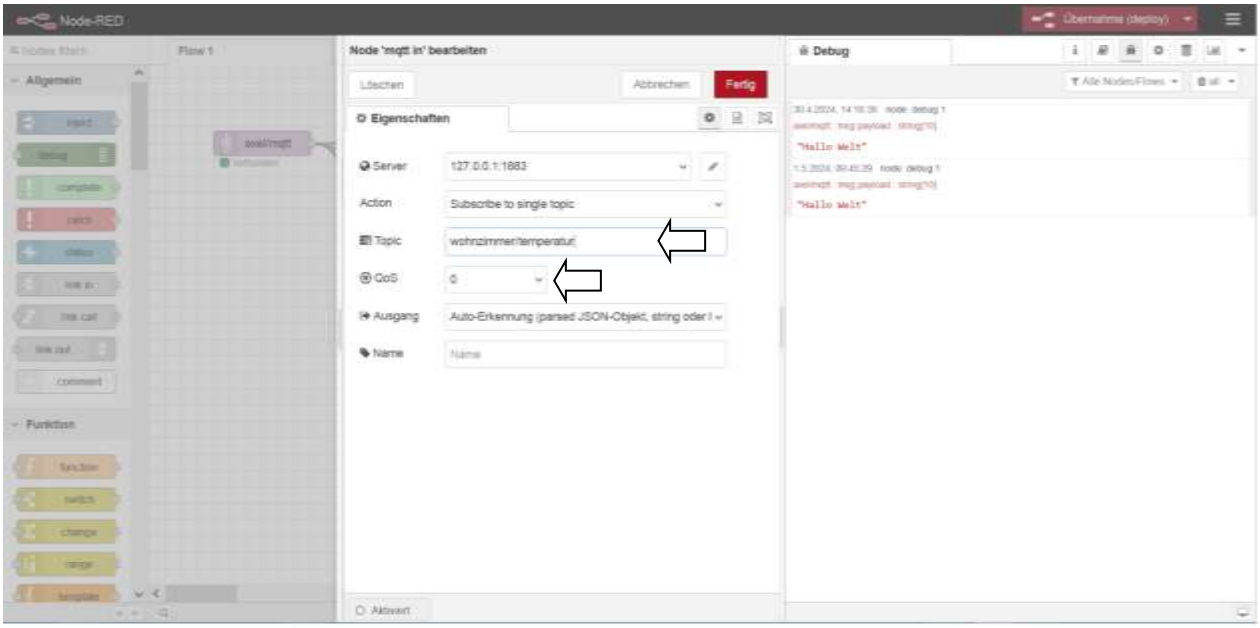


Unter Topic geben wir „wohnzimmer/temperatur“ ein und klicken auf das rote Fertig.

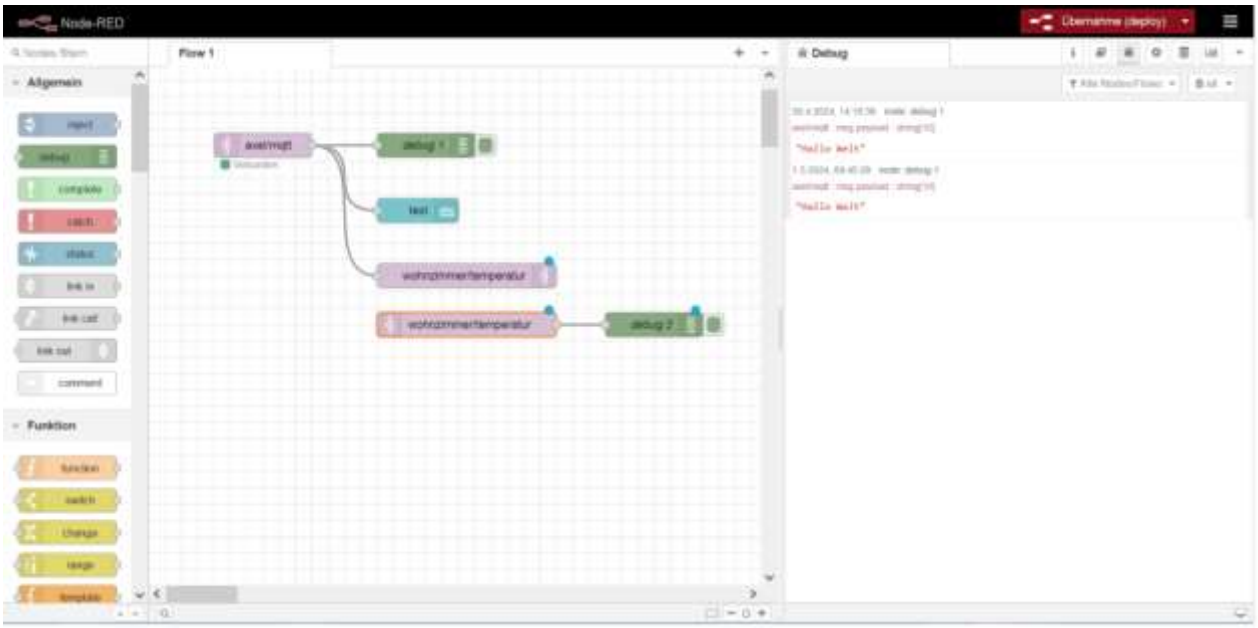
Dann ziehen wir noch zwei Elemente (Empfangen und Debug) rüber und verbinden sie.



Ein Doppelklick auf mqtt-Element (empfangen) und hier kann man das Topic eingeben.



Hier das Topic „wohnzimmer/temperatur“ und Qos (Wichtigkeit der Übertragung) auf 0 (kann verloren gehen) stellen.



So sollte es nun aussehen. Das debug2-Element ist etwas weiter rechts verschoben worden. Es sind noch die blauen Kreise zu sehen, weil die Änderungen noch nicht in den aktuellen Flow übernommen worden sind. Dazu muss man auf das rote Übernahme (deploy) klicken

Und wenn es nicht laufen sollte?

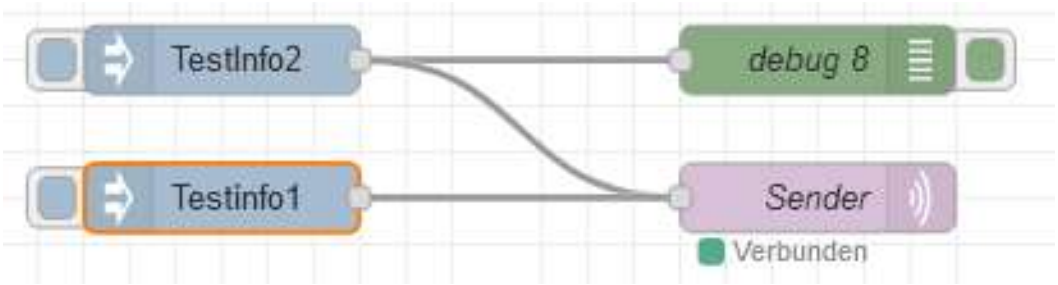
Ich hatte das Problem, das das Beispiel bei mir nicht lief. Nach sehr sehr langem Such und Probieren, habe ich die Lösung gefunden. Statt dem Port 1883 habe ich den Port 2883 genommen und es lief.

Warum dem so ist, kann ich nicht erklären. Ich habe aber die Vermutung, da ich mit zwei Rechnern auf den TXT 4.0 zugegriffen habe, das der 1883 Port dadurch „blockiert“ wurde. (so als Kurzerklärung)
Eine Nachfrage, im Forum ergab, dass man eigentlich die MQTT Verbindung beenden muss. Ich kann auch nur vermuten, dass auch im Industriemodell, genau das nicht vorkommt. Zumindest habe ich keine Art von MQTT Beenden gefunden.

MQTT Daten an sich selber (Broker vom TXT 4.0) senden und empfangen

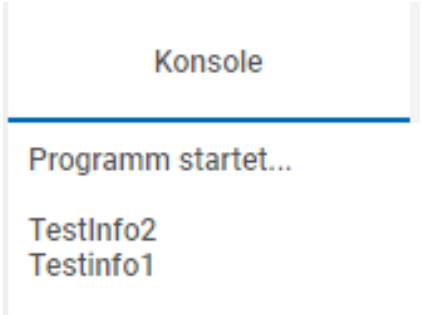
Beispiel

Hier wird der MQTT Broker auf dem TXT 4.0, mit 127.0.0.1 (Lokal) ohne Passwort und Benutzername geöffnet.



In NodeRed werden zwei Injektion mit dem Inhalt Testifo1 und Testinfo2 erstellt, die beide an den Sender die Daten senden. Zusätzlich wird beim Debug 8 die gesendeten Daten ausgegeben.

Jedes Mal, wenn nun die Daten, durch drücken der rechten Seite der Injektion, gesendet werden, wird durch das Parallelprogramm mqtt-callback die Nachricht empfangen und auf der Konsole in Robo Pro Coding ausgegeben.



HTTP
HTTP = Hypertext Transfer Protocol

HTTP ist ein Datenübertragungsprotokoll für Internetseiten und Daten.

GET - Anfrage an … : Header

HTTP Anforderungsmethode. Hierbei wird eine Anfrage an einen Server gestartet und nach (bestimmten) Informationen oder Ressourcen gefragt.

Anfrage an …: Header … Payload …

Zusätzlich wird hier ein Payload mit an die aufgerufene Seite mitgegeben.

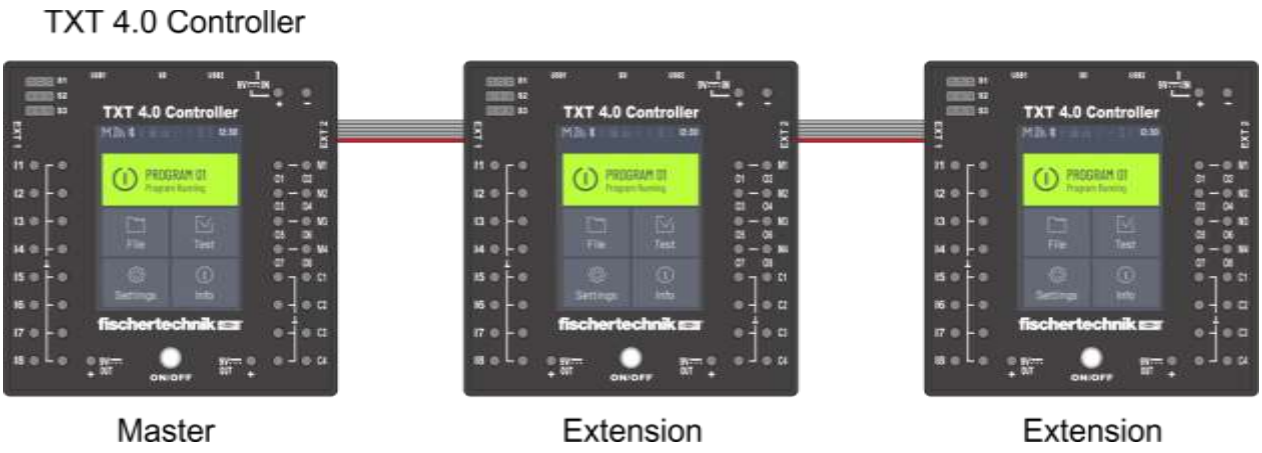
Beispiel einer Google Suchanfrage: https://www.google.com/search?q=fischertechnik

Header ist: „q“ und Payload „fischertechnik“.

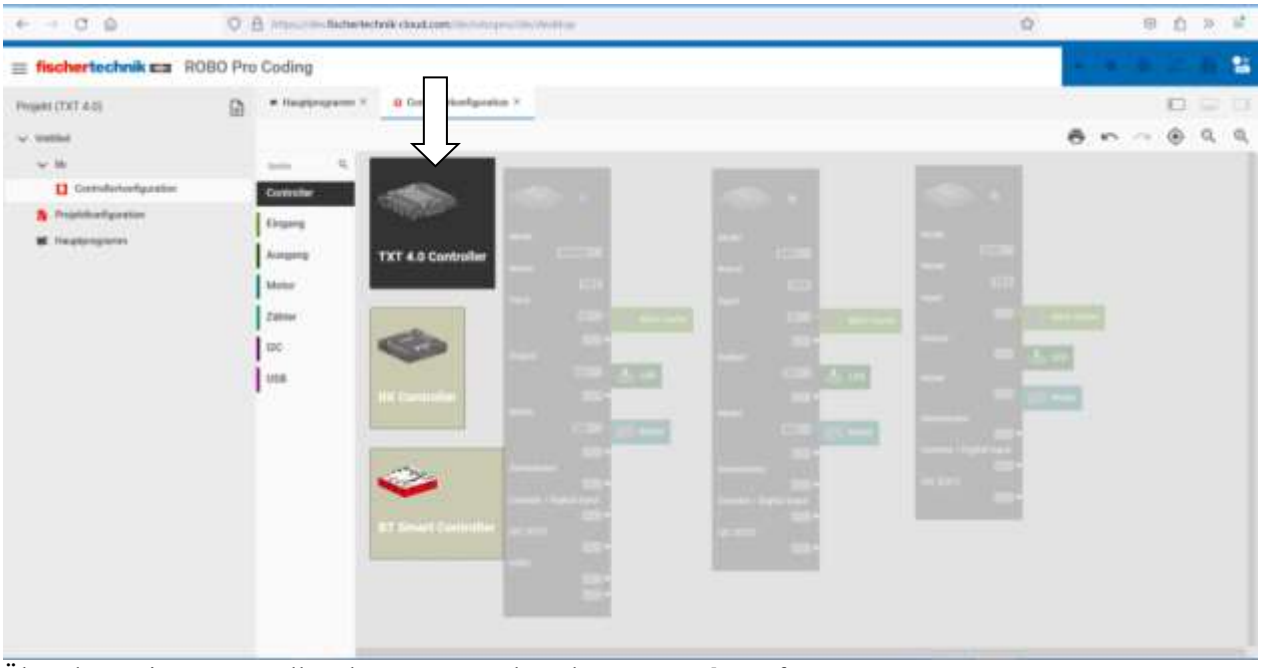
Die Antwort, die kommt, also die ganze Internetseite wird nach links weitergegeben.
Das können sehr, sehr viele Daten sein.
Man „kann“ auch per JSON abfragen.

Mehrere Controller

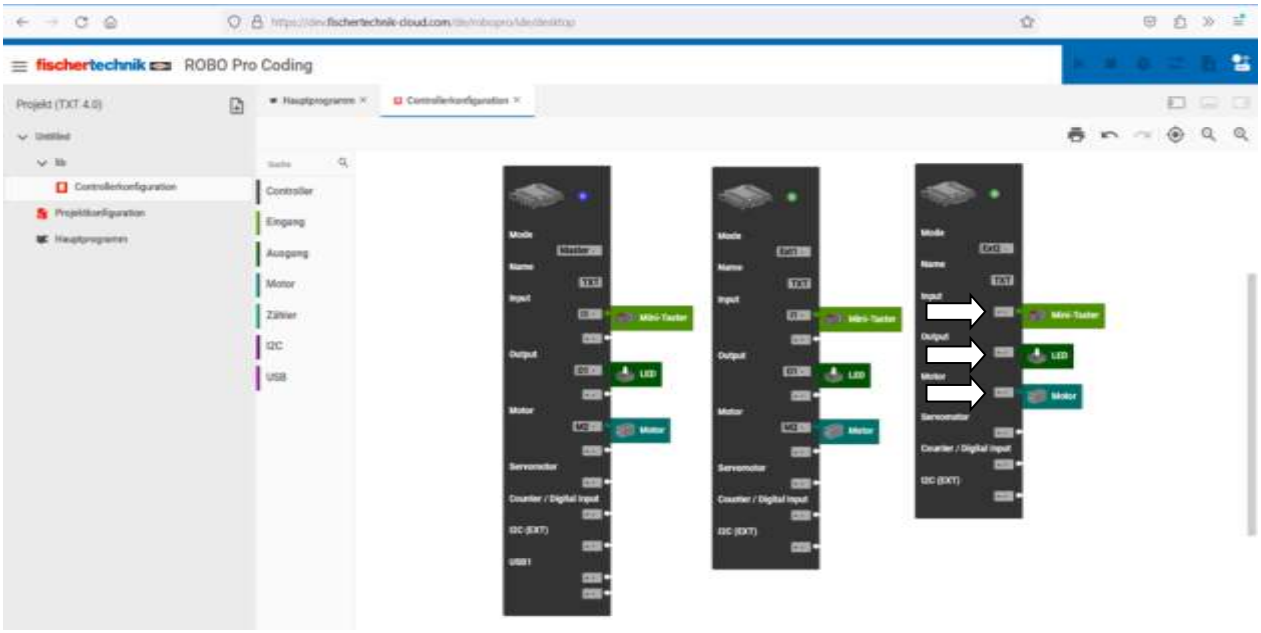
Es gibt die Möglichkeit bis zu 10 TXT 4.0 Controller zu verbinden. Das kann man über ein 6-Poliges Kabel machen. Es **muss** der erste der Master sein und bei den anderen **muss** Extension eingestellt sein.



Das macht man unter Einstellungen/Eigenschaft/Extension am Controller. Der erste (Master) Controller macht den Kontakt zum PC und gibt die Daten an die Extensions 1 bis 9 weiter. Auch bekommt der Master z.B. die Zustände der Eingänge und der Extensions und gibt die Daten an den PC bzw. an das laufende Programm im PC oder Master TXT 4.0 weiter. In Robo Pro Coding kann man weitere TXT 4.0 Controller aufrufen.



Über den Reiter „Controller“ kann man noch mehr TXT 4.0 hinzufügen.

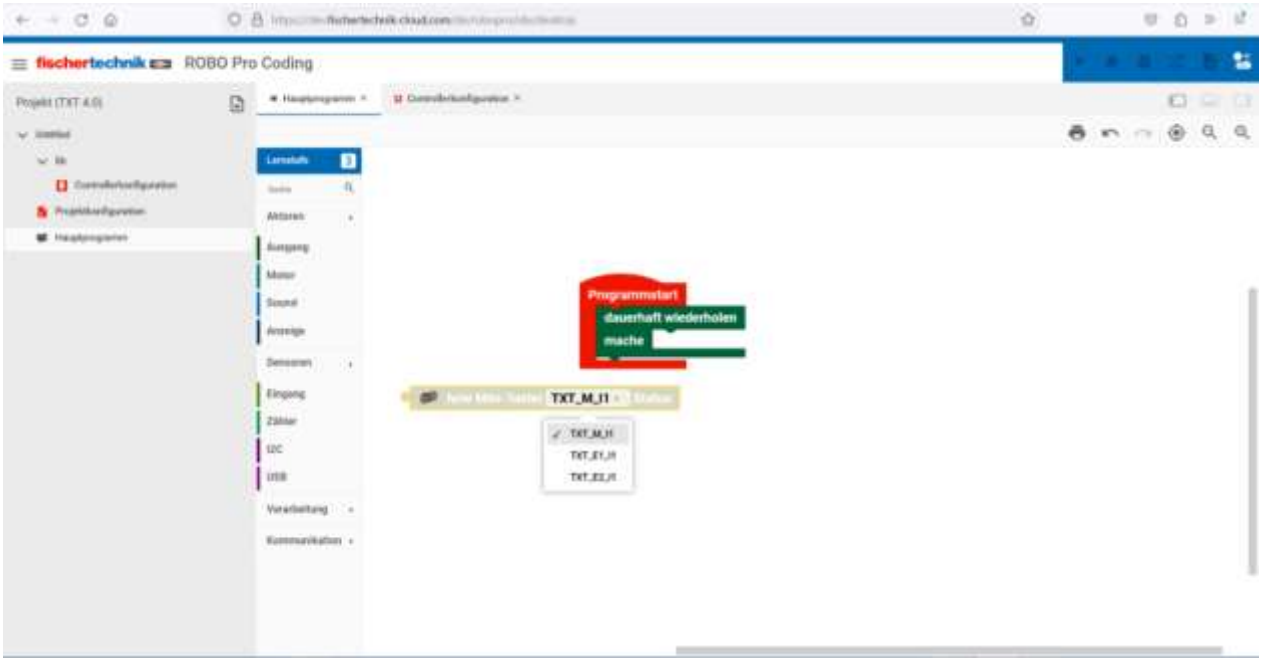


Der erste TXT 4.0 ist der Master, dann kommen Ext1 (Extension1), Ext2 (Extension2)…

Der mittlere Controller wurde über einen Klick auf einen Controller erzeugt. Der Taster, die LED und der Motor wurden über die Reiter eingefügt Was auffällt ist, dass er kleiner ist als der Master. Der Master hat zusätzlich die nutzbaren USB Anschlüsse – die Extensions nicht.
Der rechte wurde über Duplizieren des mittleren Controllers erzeugt. Auch dieser ist kleiner. Das liegt aber daran, dass die Ein und Ausgänge noch nicht z.B. als I1, O1 und M2 definiert sind. Die muss man noch auswählen, was beim einzelnen hinzufügen, automatisch vorgeschlagen wird.

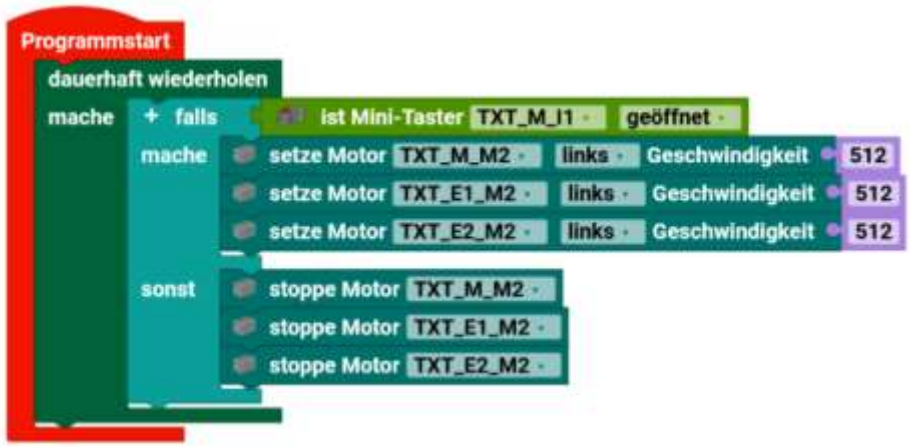
Wenn man die Ausgänge definiert hat, ist das Symbol genau so lang.

Wenn man den O-Ausgang definiert, wird der M2 Ausgang automatisch gesetzt.



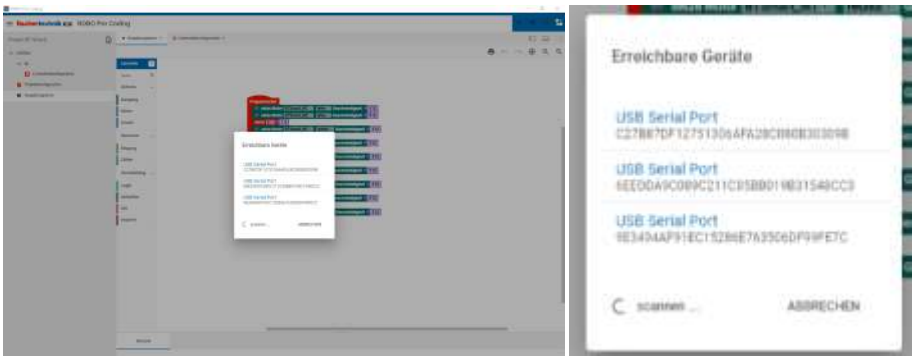
Die Auswahl der Taster erweitert sich um die entsprechenden Taster der Extensions. Hier kann man sich zwischen dem Taster am Master oder der Extensions 1 oder 2 entscheiden.

Oder hier ein Motor M2 am Master oder einer der zwei Extensions.



Man kann somit über den Taster am Master z.B. alle M-Motoren auch die an den Extensions Ein- und Ausschalten.

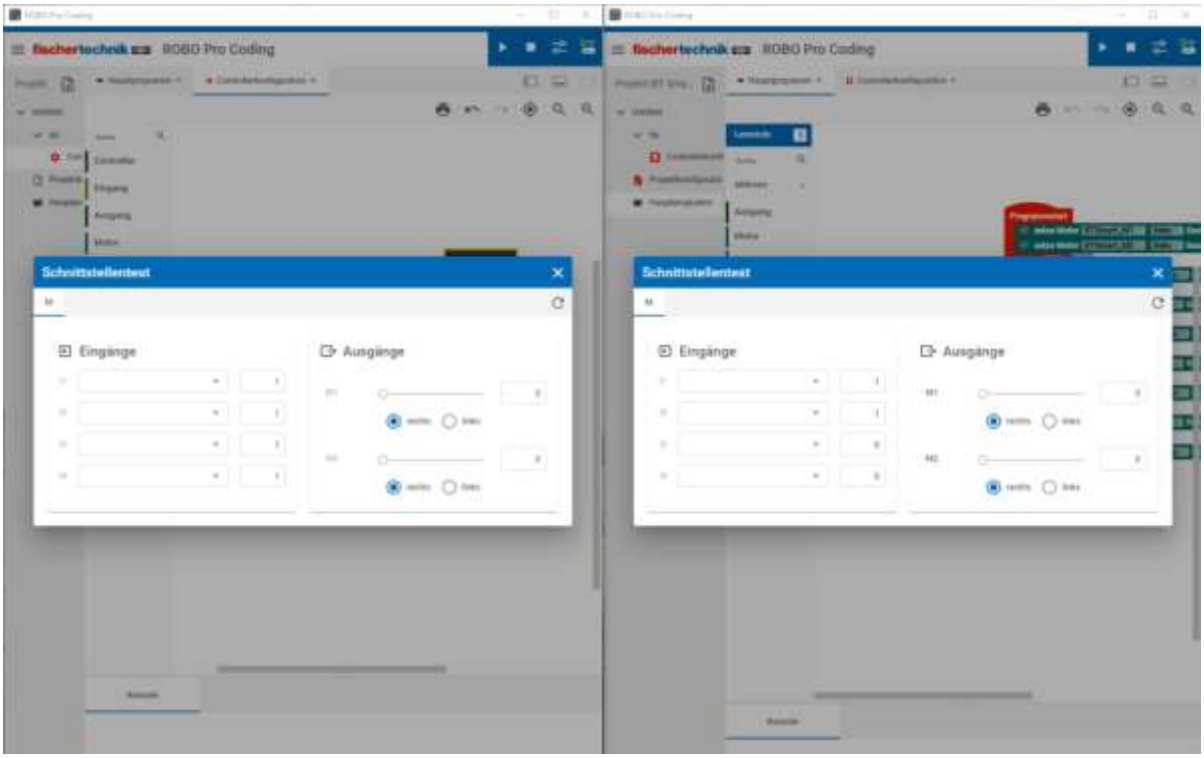
Mehrere BT Smart Controller
Die Qual der Wahl...



Hier mal 3 BT Smart Controller

Man sieht, dass sich die Controller nur über die darunter stehende Nummer unterscheiden. Das ist vermutlich die Nummer des USB Anschlusses. Die drei BT Smart Controller werden einmal als USB Gerät geführt und mit einer COM-Schnittstelle, hier COM3, COM4 und COM5. Bislang habe ich noch nicht herausbekommen, was es mit den Zahlen unter dem Namen des USB Portanschlusses auf sich hat. Es ist so nicht zuzuordnen welcher USB Port denn ist. Auf den neusten BT Smart Controllern ist ein Aufkleber mit der BT-Mac-Adresse. (Bei meinen nicht.)

Dennoch kann man nur einen in Robo Pro Coding auswählen.



Was aber geht, ist mehrere Instanzen von Robo Pro Coding starten und jeweils einen anderen Controller auswählen. Hier sind zwei Robo Pro Coding Versionen gleichzeitig mit funktionierendem Schnittstellentest.

Man kann aber keinen Datenaustausch zwischen den Controllern über die zwei Robo Pro Coding machen,

um damit große Modelle anzusteuern. Man "könnte" versuchen über einen Eingang des BT Smart Controller eine Verbindung mit dem TXT 4.0 aufzubauen. Der Aufwand ist hoch, der Nutzen gering.

Was ich nicht ausprobiert habe, ist Robo Pro Coding mehrfach zu installieren. Ansonsten kann man Robo Pro Coding nur einmal starten und somit nur einen Controller ansteuern.

Mischungen von Controllern

BT Smart Controller
Was man machen kann, ist auf einen RX oder TXT 4.0 Controller ein Programm runterladen und zu starten, um dann ein neues Projekt für den BT Smart Controller importieren und das Programm auf dem laufen zu lassen.

RX und TXT 4.0 Controller
Momentan sehe nur die Möglichkeit RX und TXT 4.0 Controller über I2C zu verbinden.

RX und TXT 4.0 Controller und ältere Controller

TX/TXT
Auch hier sehe nur die Möglichkeit die Controller über I2C zu verbinden. Achtung bei der Spannung vom TX (5V) zu den anderen (3V)!
Auch sind diese älteren Controller dann mit Robo Pro zu programmieren. Meines Wissens hat das bisher noch keiner gemacht.

Verbindung über die Ein- und Ausgänge

Bei Industrieanlagen kann man auch einen Trick machen. Man baut sich eine Lichtschranke an der Übergabe von einem zum anderen Modell. Der eine Fototransistor der Lichtschranke, kann nun von beiden Controllern als Eingang genutzt werden. Man muss dazu auch die Masse/Minus beider Controller verbinden. Das geht bei allen fischertechnik Interfaces.

Ältere Robo-Interface, Serielles Interface, Parallel Interface
Ja, über die 9V Ein- und Ausgänge kann man sich was programmieren. Wobei die Frage ist, wie sinnvoll so etwas ist, da man halt die Ein- und Ausgänge nicht für das Modell nutzen kann.

Robo Connect Box und Serielles Interface / Parallel Interface
Die Firma Knobloch bietet eine Robo Connect Box an, mit deren Hilfe man ältere Serielles Interface und Parallel Interface über USB an die heutigen Rechner anschließen kann. Man kann die Interfaces dennoch nicht mit Robo Pro Coding programmieren, da sie nicht damit verbunden werden können.

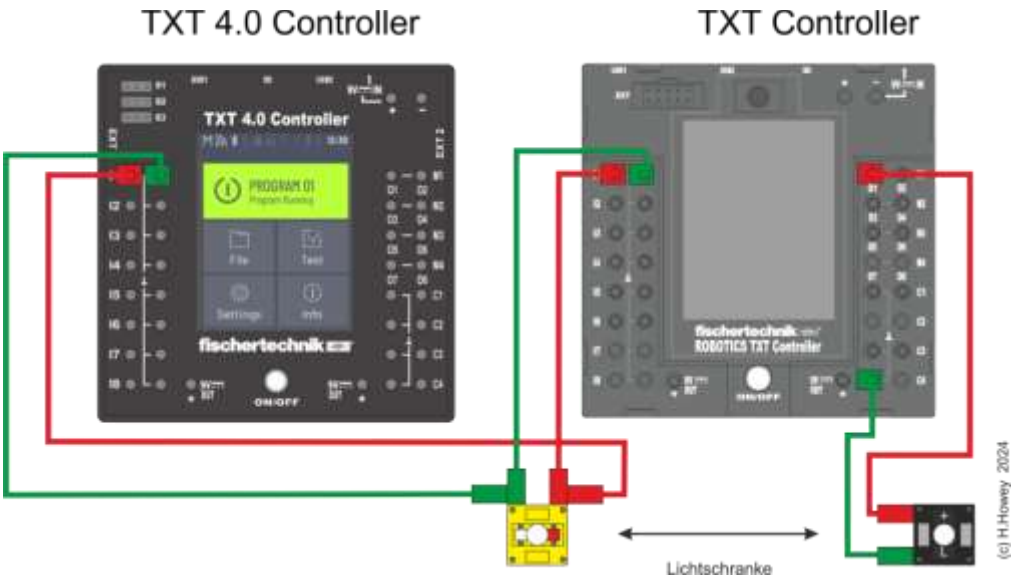
Knobloch: Multiface Interface, Böing und Kallenbach: USB-Interface und andere...
Für das Multiface Interface von Knobloch, dem USB-Interface von Böing und Kallenbach oder anderen, gilt dasselbe wie bei den Robo Interface, man kann die Interfaces nicht mit Robo Pro Coding programmieren, da sie nicht damit verbunden werden können.

Arduino und Co, ftDuino
Im Grunde muss man das Übertragungsprotokoll zwischen den Controllern und dem PC kennen. Dann muss man ein Programm scheiben, das dieses Protokoll benutzt, um dem Robo Pro Coding vorzugaukeln, das da ein fischertechnik Controller am PC ist.

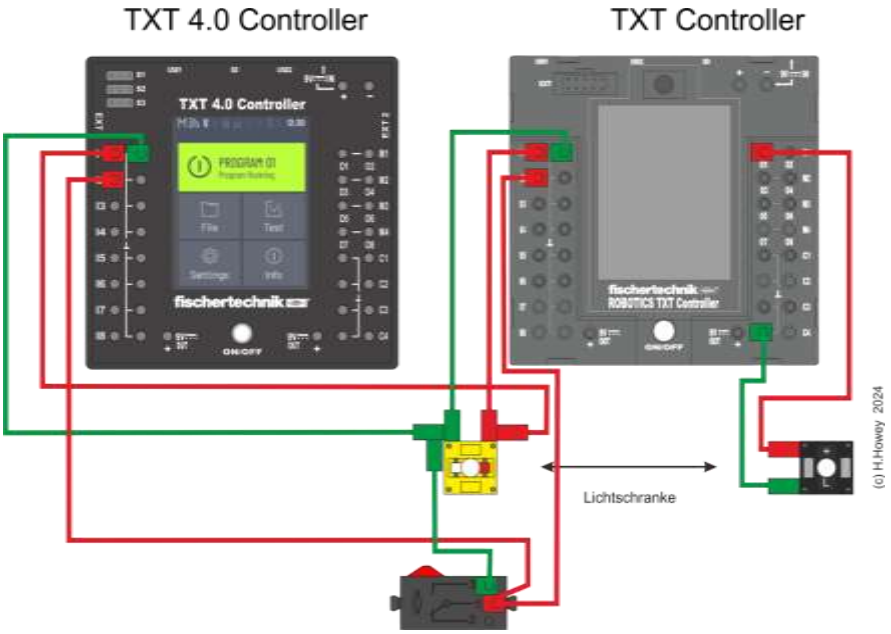
Für den Arduino Uno/Mega und den ftDuino mit dem älteren Robo Pro habe ich das mal gemacht. Meiner Einschätzung nach sollte es auch mit dem Robo Pro Coding gehen. Bislang hat es meines Wissens noch keiner gemacht. Den ftDuino könnte man über den I2C Bus ansprechen.

Gemeinsame Nutzung von Eingängen bei verschiedenen Controllern.

Beispiel TXT 4.0 mit einem TXT Controller



Wenn man beispielsweise zwei Modelle miteinander verbinden möchte, die eine Gemeinsame Lichtschranke für eine Übergabe haben, kann man einfach den Fototransistor an beide Controller anschließen. Man könnte auch die LED an den TXT 4.0 anschließen.
Wenn man weitere Taster, Fototransistoren usw. gemeinsam nutzen möchte, braucht man nur einmal das Minus zu verbinden. Es würde nur jeweils ein weiteres Plus, z.B. am Taster, genügen.

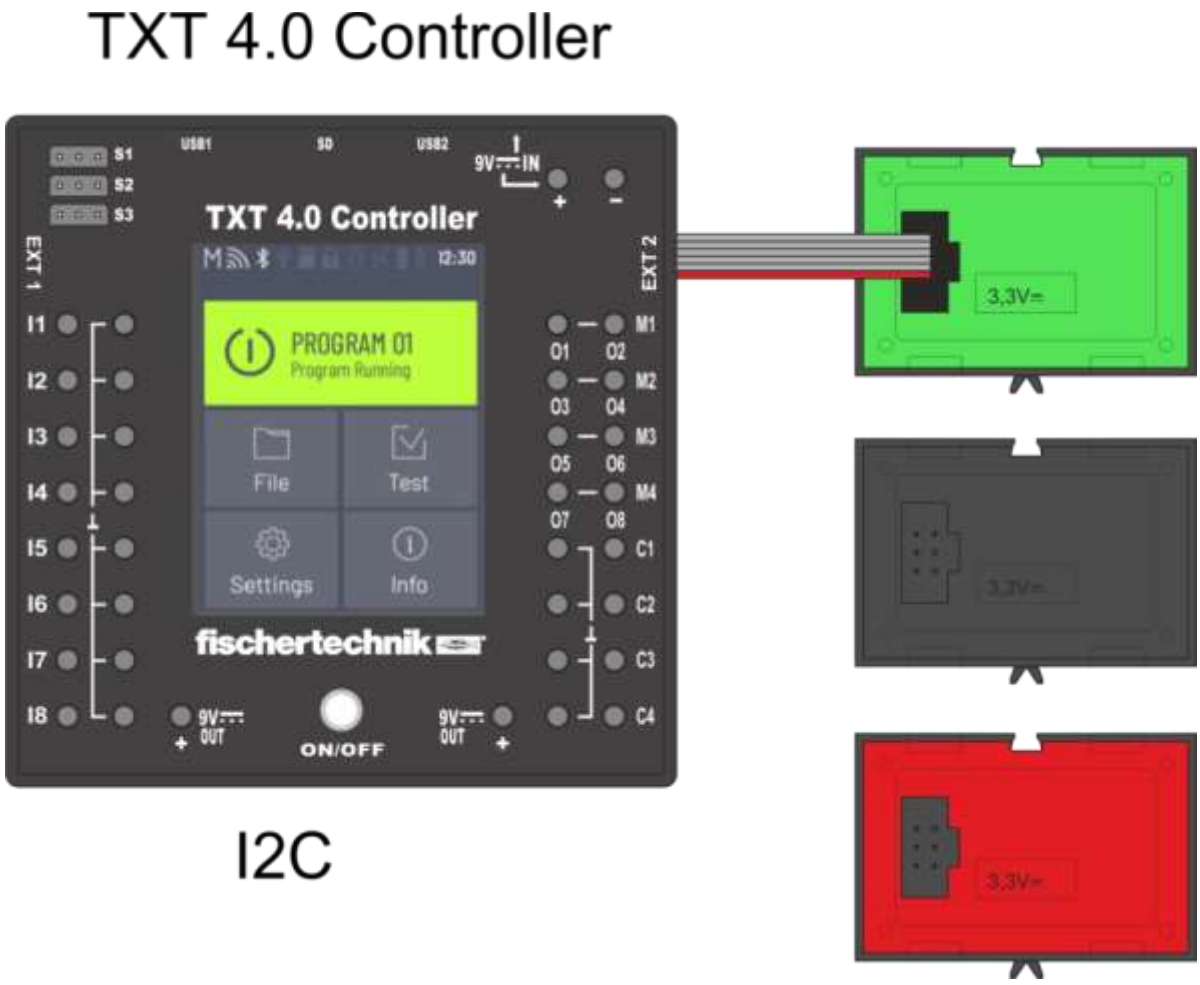


Das funktioniert auch mit allen anderen fischertechnik Controllern. Beim oberen Beispielbild könnte man den TXT 4.0 mit Robo Pro Coding programmieren und den TXT Controller mit RoboPro.

Die Nutzung von gemeinsamen -Ausgängen- ist *nicht* möglich.

I2C für Experten

Wenn es noch nicht gelesen wurde, bitte erst den Teil zu I2C weiter vorne unter Controllerkonfiguration lesen (Seite 173).



TXT 4.0 Controller mit den fischertechnik Sensoren für den I2C Bus. Es spielt keine Rolle ob die Sensoren am EXT 1 oder 2 angeschlossen sind, da beide intern verbunden sind.
Es gibt sehr viele Beispiele für die älteren TX und TXT Controller. Diese können auch für den TXT 4.0 und den RX Controller genommen werden. Die Busnummer ist aber nicht 1 sondern es ist Bus Nr. 3.

Wichtiger Hinweis zum I2C:

Man sollte schon wissen, was man da macht. Man darf nicht Sachen einfach mischen. Z.B. ist die Spannung, die man verwendet sehr wichtig. Man kann nicht einfach 3,3V mit 5V mischen. Das führt zur unwiderruflichen Zerstörung der Ein- und Ausgänge am TXT 4.0. Das betrifft z.B. die Kopplung vom TX/TXT zum TXT 4.0 oder vom Arduino UNO und dem TXT 4.0. Es gibt "auch" 3,3V Versionen vom UNO aber viele der I2C-Sensoren laufen mit 5V und damit auch die I2C Schnittstelle auf 5V. Ansonsten Levelshifter 3,3V <-> 5V benutzen.

Als Bus wird der I2C Bus Nr.3 benutzt der an der Seite vom TXT 4.0 zugänglich ist. Niemals einen anderen Bus benutzen! Man könnte Werte z.B. vom Speicher überschreiben und den TXT 4.0 dauerhaft blockieren. Man darf auch keinen Kurzschluss auf dem Bus erzeugen. Immer erst prüfen, dann einschalten. Auch sollte man die Sensoren nicht im laufenden Betrieb anschließen.

Ich habe sehr lange gebraucht, bis ich eine I2C Verbindung vom TXT 4.0, zu einem anderen I2C Gerät ohne Blocky und ohne den fischertechnik Liberys, zum Laufen gebracht habe. Schlussendlich hat mir fischertechnik auf meine Nachfrage hin, ein lauffähiges Beispiel zugesendet.

Das Beispiel bezieht sich auf den 20.Teil vom I2C Workshop, der Zeitschrift Raspberry Pi Geek:
https://www.raspberry-pi-geek.de/ausgab ... apds-9960/
Hier wird das APDS-9960 Sensor-Modul über I2C angesprochen.
Genau dieses IC ist in dem fischertechnik-Gestensensor drin.

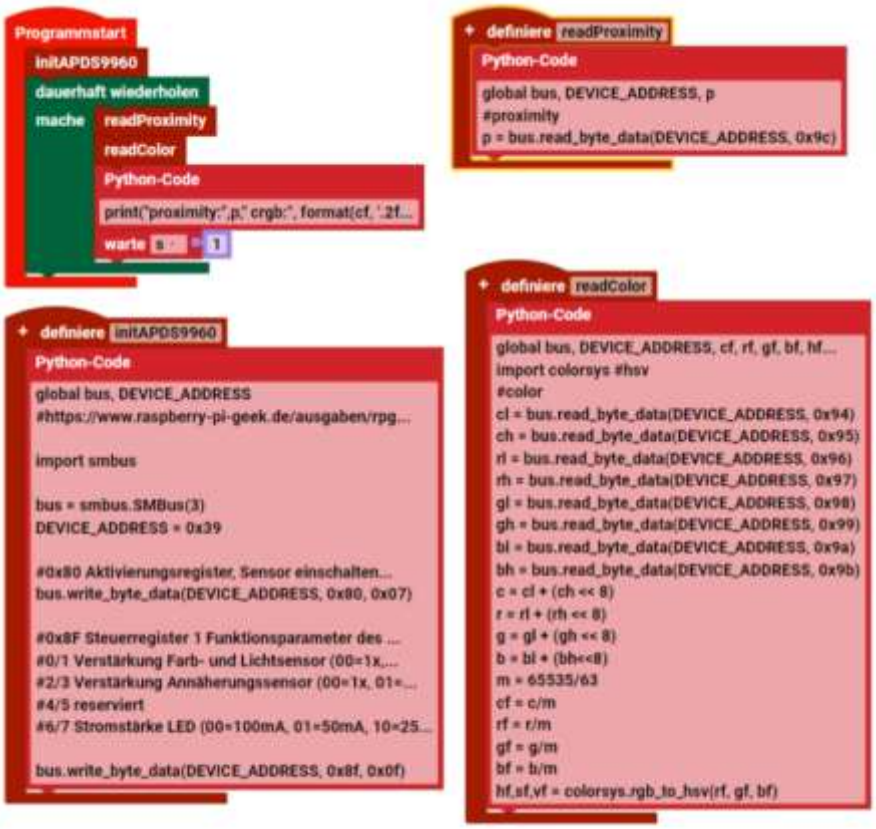
Im RoboPro Coding - Programm ist ein Hauptprogramm, wo der Sensor initialisiert wird und dann in einer Dauerschleife die Übertragenen Werte jede Sekunde auf dem Bildschirm ausgegeben werden. Es werden die Werte der Entfernung (Proximity) und der Farbe (Color) übertragen. Als Bus wird der I2C Bus "Nr.3" benutzt der an der Seite vom TXT 4.0 zugänglich ist.

Tipp, Schreibweise von Zahlen und Daten:
Die Zahlen z.B. der Adresse, vom Device am I2C Bus, werden Hexadezimal im Programm angegeben. Hier ist sie 0x39 Hexadezimal = Dezimal 57.
Die Farbdaten werden in zwei Bytes übertragen. Einmal dem High Byte und dem Low Byte. Zuerst wird das High Byte 8-mal nach links geschiftet `r = rl + (rh << 8)` und dann mit dem Low Byte zu einer Zahl addiert. Die 8 kommt von einem Byte = 8 Bit.

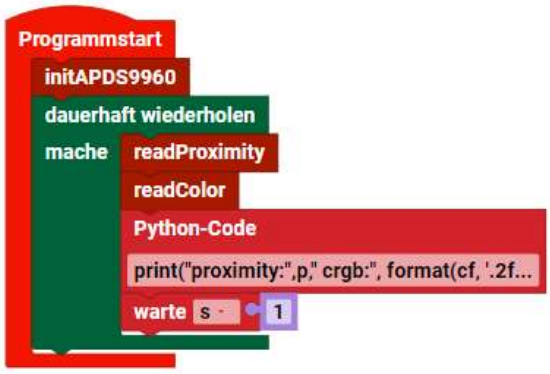
Die Python-Variante für I2C:

```
import time

def initAPDS9960():
    global bus, DEVICE_ADDRESS
    #https://www.raspberry-pi-geek.de/ausgaben/rpg/2018/08/i2c-workshop-teil-20-apds-9960/

    import smbus

    bus = smbus.SMBus(3)
    DEVICE_ADDRESS = 0x39

    #0x80 Aktivierungsregister, Sensor einschalten, Funktionen aktivieren
    bus.write_byte_data(DEVICE_ADDRESS, 0x80, 0x07)

    #0x8F Steuerregister 1 Funktionsparameter des ICs einstellen.
    #0/1 Verstärkung Farb- und Lichtsensor (00=1x, 01=4x, 10=16x, 11=64x)
    #2/3 Verstärkung Annäherungssensor (00=1x, 01=2x, 10=4x, 11=8x)
    #4/5 reserviert
    #6/7 Stromstärke LED (00=100mA, 01=50mA, 10=25mA, 11=12,5mA)

    bus.write_byte_data(DEVICE_ADDRESS, 0x8f, 0x0f)


def readProximity():
    global bus, DEVICE_ADDRESS, p
    #proximity
    p = bus.read_byte_data(DEVICE_ADDRESS, 0x9c)


def readColor():
    global bus, DEVICE_ADDRESS, cf, rf, gf, bf, hf, sf, vf
    import colorsys #hsv
    #color
    cl = bus.read_byte_data(DEVICE_ADDRESS, 0x94)
    ch = bus.read_byte_data(DEVICE_ADDRESS, 0x95)
    rl = bus.read_byte_data(DEVICE_ADDRESS, 0x96)
    rh = bus.read_byte_data(DEVICE_ADDRESS, 0x97)
    gl = bus.read_byte_data(DEVICE_ADDRESS, 0x98)
    gh = bus.read_byte_data(DEVICE_ADDRESS, 0x99)
    bl = bus.read_byte_data(DEVICE_ADDRESS, 0x9a)
    bh = bus.read_byte_data(DEVICE_ADDRESS, 0x9b)
    c = cl + (ch << 8)
    r = rl + (rh << 8)
    g = gl + (gh << 8)
    b = bl + (bh<<8)
    m = 65535/63
    cf = c/m
    rf = r/m
    gf = g/m
    bf = b/m
    hf,sf,vf = colorsys.rgb_to_hsv(rf, gf, bf)


initAPDS9960()
while True:
    readProximity()
    readColor()
    print("proximity:",p," crgb:", format(cf, '.2f'),format(rf, '.2f'),format(gf, '.2f'), format(bf,
'.2f')," hsv:",format(hf, '.2f'),format(sf, '.2f'),format(vf, '.2f'))
    time.sleep(1)
```

I2C in Blocky



Übersicht der vier Python-Codeblöcke



Das Hauptprogramm initialisiert den I2C Bus und das Device. Danach wird in einer Dauerschleife die Werte vom Device gelesen und dann ausgegeben.

Python-Code:

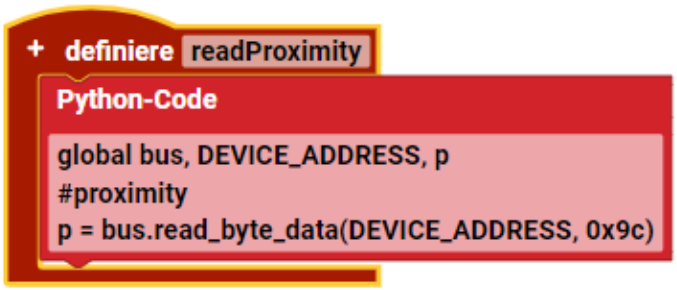
```
print("proximity:",p," crgb:", format(cf, '.2f'),format(rf,
  '.2f'),format(gf, '.2f'), format(bf, '.2f')," hsv:",format(hf,
  '.2f'),format(sf, '.2f'),format(vf, '.2f'))
```

Hier ist die Ausgabe der Daten.

Python-Code:

```
global bus, DEVICE_ADDRESS
#https://www.raspberry-pi-geek.de/ausgaben/rpg/2018/08/i2c-workshop-tei
l-20-apds-9960/

import smbus

bus = smbus.SMBus(3)
DEVICE_ADDRESS = 0x39

#0x80 Aktivierungsregister, Sensor einschalten, Funktionen aktivieren
bus.write_byte_data(DEVICE_ADDRESS, 0x80, 0x07)

#0x8F Steuerregister 1 Funktionsparameter des ICs einstellen.
#0/1 Verstärkung Farb- und Lichtsensor (00=1x, 01=4x, 10=16x, 11=64x)
#2/3 Verstärkung Annäherungssensor (00=1x, 01=2x, 10=4x, 11=8x)
#4/5 reserviert
#6/7 Stromstärke LED (00=100mA, 01=50mA, 10=25mA, 11=12,5mA)

bus.write_byte_data(DEVICE_ADDRESS, 0x8f, 0x0f)
```

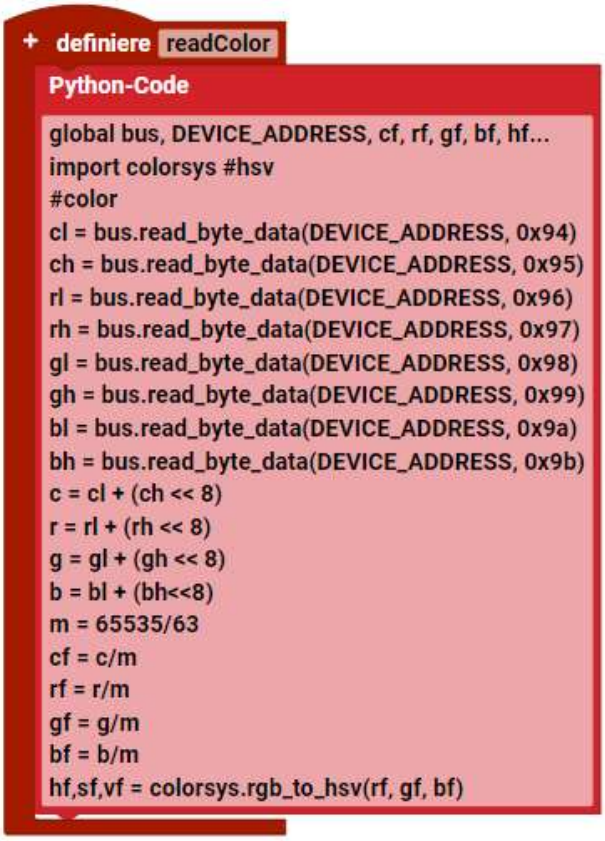
Hier ist die Initialisierung des Gestensensors. Es ist der SMBus 3 mit der Deviceadresse 0x39. Mit dem Write-Befehl werden nun Daten hintereinander an das Device gesendet.



Python-Code:

```
global bus, DEVICE_ADDRESS, p
#proximity
p = bus.read_byte_data(DEVICE_ADDRESS, 0x9c)
```

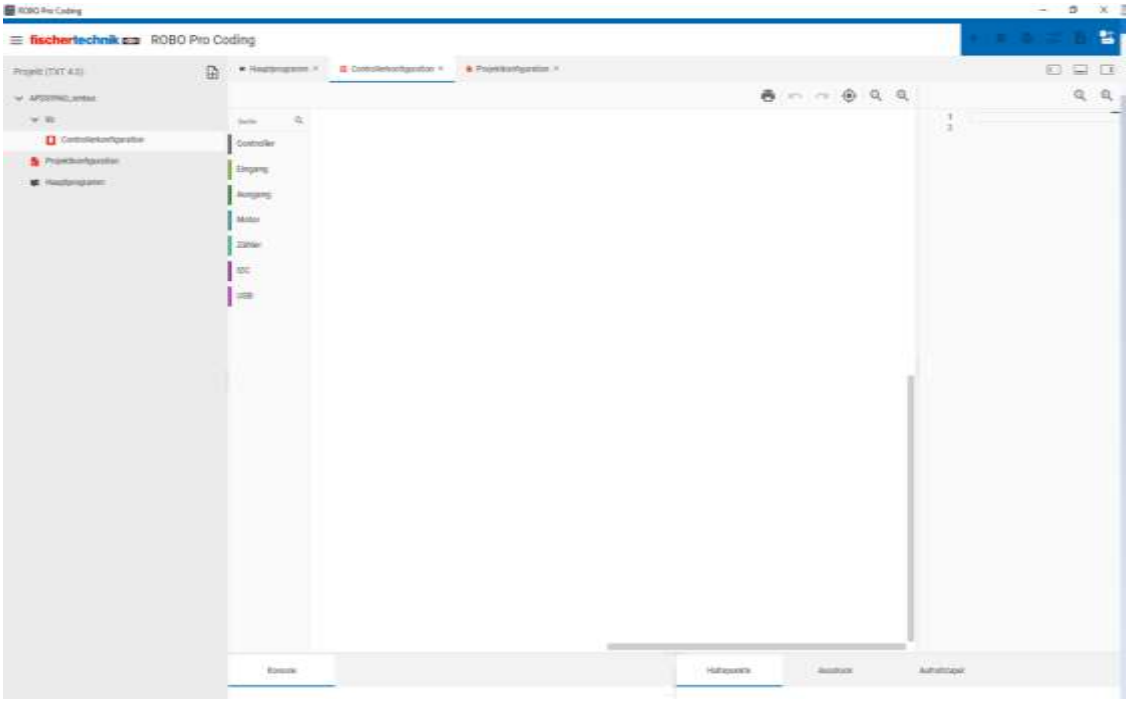
Hier werden Daten (proximity = Entfernung) vom Device am I2C-Bus gelesen.



Python-Code:

```
global bus, DEVICE_ADDRESS, cf, rf, gf, bf, hf, sf, vf
import colorsys #hsv
#color
cl = bus.read_byte_data(DEVICE_ADDRESS, 0x94)
ch = bus.read_byte_data(DEVICE_ADDRESS, 0x95)
rl = bus.read_byte_data(DEVICE_ADDRESS, 0x96)
rh = bus.read_byte_data(DEVICE_ADDRESS, 0x97)
gl = bus.read_byte_data(DEVICE_ADDRESS, 0x98)
gh = bus.read_byte_data(DEVICE_ADDRESS, 0x99)
bl = bus.read_byte_data(DEVICE_ADDRESS, 0x9a)
bh = bus.read_byte_data(DEVICE_ADDRESS, 0x9b)
c = cl + (ch << 8)
r = rl + (rh << 8)
g = gl + (gh << 8)
b = bl + (bh<<8)
m = 65535/63
cf = c/m
rf = r/m
gf = g/m
bf = b/m
hf,sf,vf = colorsys.rgb_to_hsv(rf, gf, bf)
```

Hier werden Daten (Color = Farben) vom Device am I2C-Bus gelesen. Auch werden die RGB-Daten (Rot/Grün/Blau) umgerechnet und die Helligkeit bestimmt.



Die Controllerkonfiguration bei diesem Projekt ist –leer-. Es ist kein I2C Device in Blocky angeschlossen.

Alte Robo Pro, ftSrcratch und ftrobopy (statt Robo Pro Coding) Programme auf dem TXT 4.0 laufen lassen.

Hinweis:
Das Programm „ftrobopy-Server“ 0.9.8. ist momentan „Beta“ und hat ein paar Einschränkungen aber auch zusätzliche Möglichkeiten. Hier mit Windows 10 (8 geht so nicht).

Torsten Stuehn hat das Programm „ftrobopy-Server“ geschrieben, womit man Robo Pro, ftSrcratch und ftroboy Programme auf dem TXT 4.0 laufen lassen kann. Siehe dazu auch die ftpedia 2-2022 von der ftcommunity.de . Auf dem TXT 4.0 wird ein Programm gestartet, das die Kommunikation zwischen einem älteren TXT und einem PC emuliert. Man kann damit z.B. Robo Pro 4.7.0. Programme laufen lassen. Die Nummer 4.7.0. kann aber auch für ältere Versionen geändert werden. In dieser Version werden keine Extensions unterstützt. Man muss SSH am TXT 4.0 Controller aktivieren. (Einstellungen/Fernzugriff/SSH)

1.- Download

Download der ZIP-Datei von: https://github.com/ftrobopy/ftrobopy_server/archive/refs/heads/main.zip

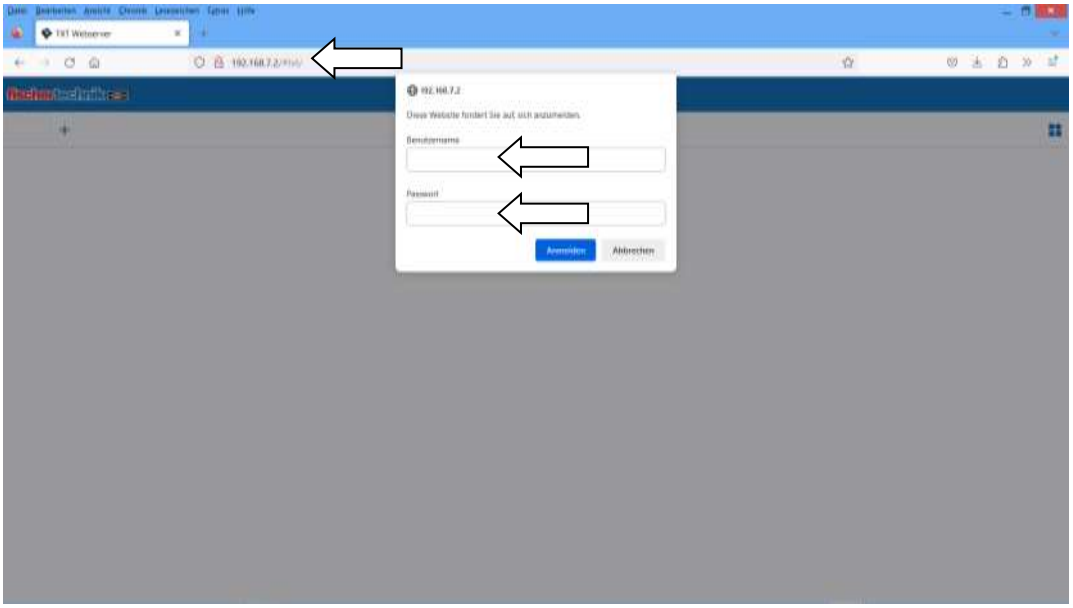
2.- Entpacken

Entpacken der ZIP-Datei (ftrobopy_server-main.zip) auf dem PC (rechte Maustaste und „Alles Entpacken“ und dann Extrahieren).

3.- Dateien kopieren

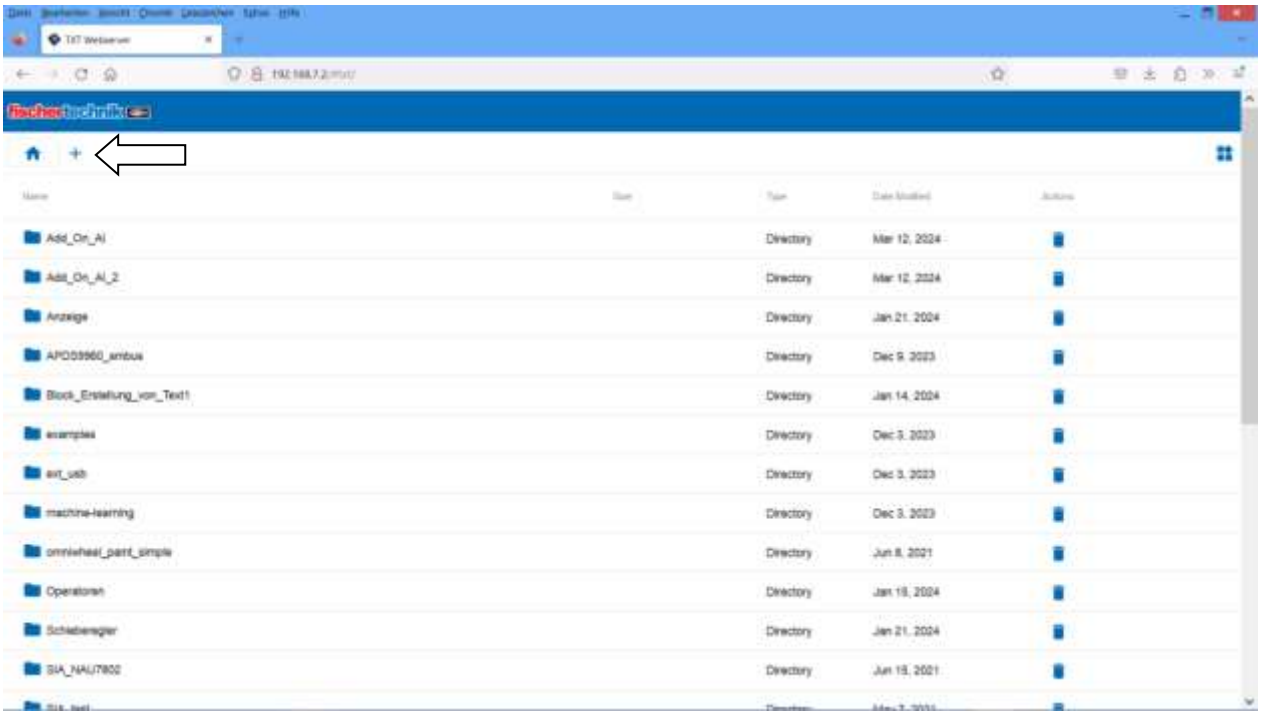
Im Ordner und Unterordner /scr befinden sich jeweils eine Datei, die auf den TXT 4.0 kopiert werden müssen.

- TXT 4.0 Einschalten (Voller Akku! / Netzteil)
- TXT 4.0 mit USB-Kabel mit dem PC verbinden.
- Internetbrowser auf PC starten und die Nummern 192.168.7.2 eingeben.



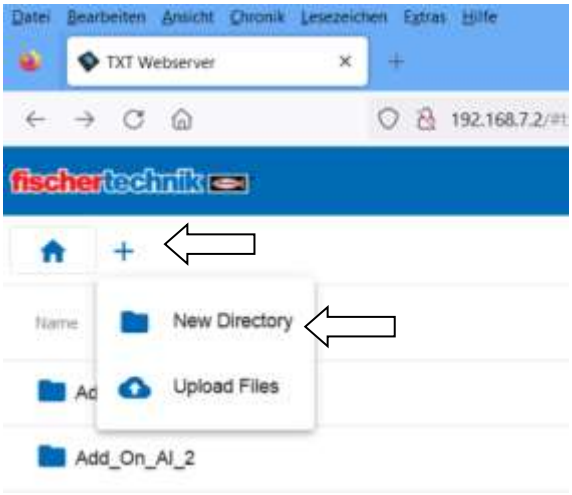
- Unter Benutzername: ft und das Passwort ist fischertechnik eingeben.
  Dann auf Anmelden klicken.

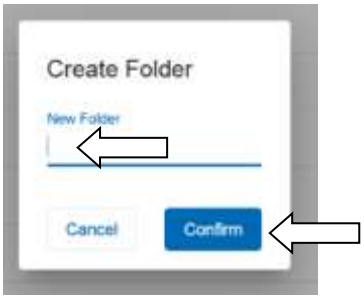
Nun werden alle (zugänglichen) Programme/Ordner angezeigt, die auf dem TXT sind.



Je nach dem, was man vorher schon gemacht hat, ist die Liste länger oder kürzer. Sie kann auch leer sein. Wenn man z.B. in einen Ordner reingeht, um zu schauen was drin ist und wieder zurück geht, dauert das Laden und Übertragen auf den PC etwas Zeit.

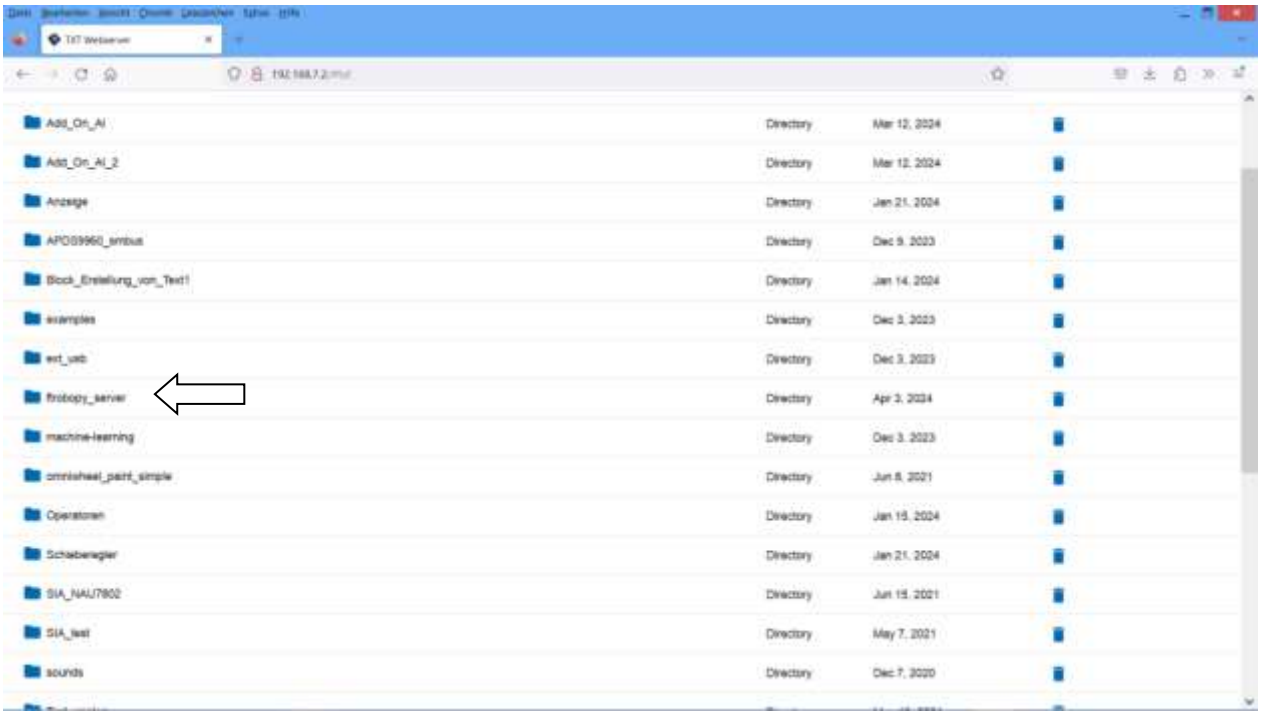
- Über das Plus einen neuen Ordner (New Directory) mit den Namen ftrobopy_server anlegen.



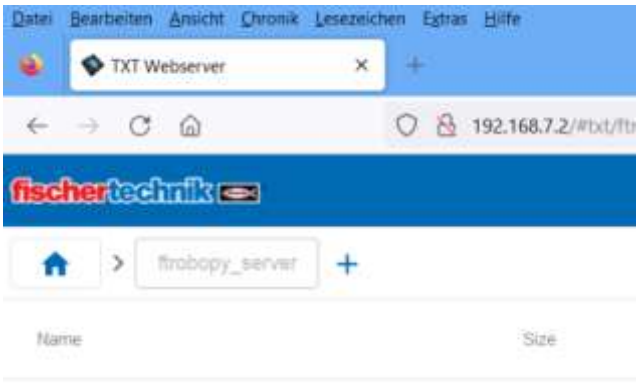


Hier ftrobopy_server eingeben und dann auf „Confirm“ klicken.

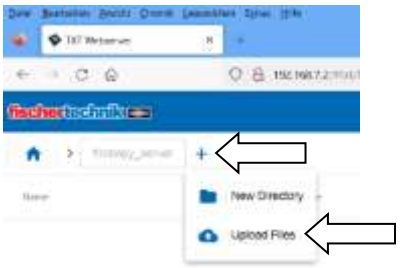
Den neuen Ordner anklicken.



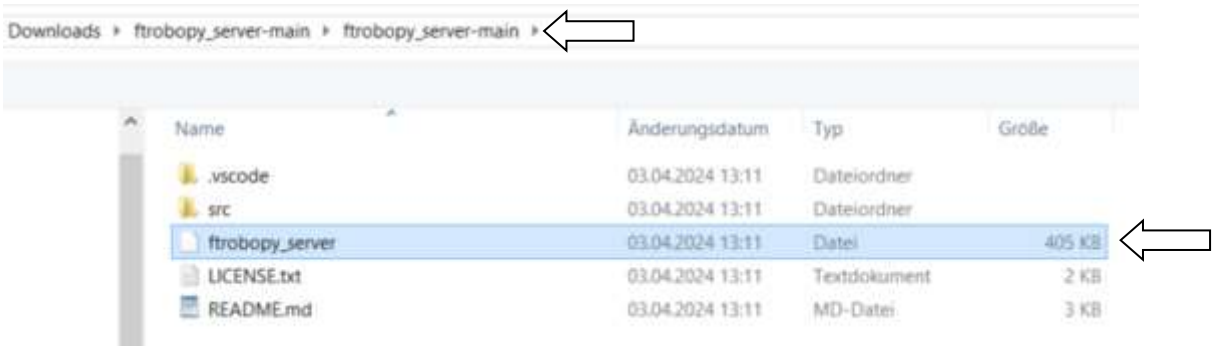
Und man kommt in diesen Ordner.



- Über das Plus Dateien (mit Upload Files) auf den TXT 4.0 kopieren.

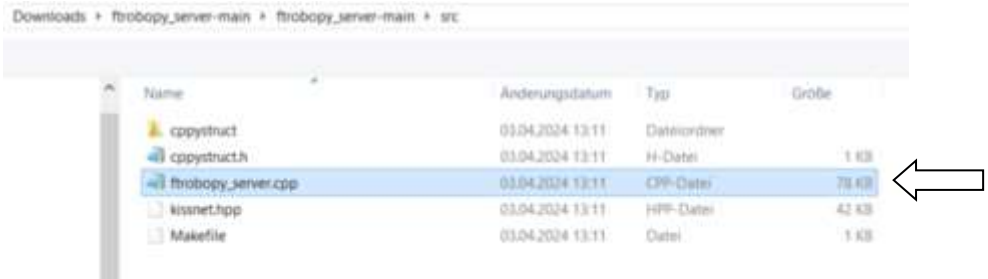


Nun auf „Add Files“ klicken.



Im entpackten Ordner auf dem PC, die Datei ftrobopy_server anklicken und auf „Öffnen“ klicken.
Hier ist die Datei unter „Downloads/ftrobopy_server-main/ftrobopy_server-main“ zu finden.

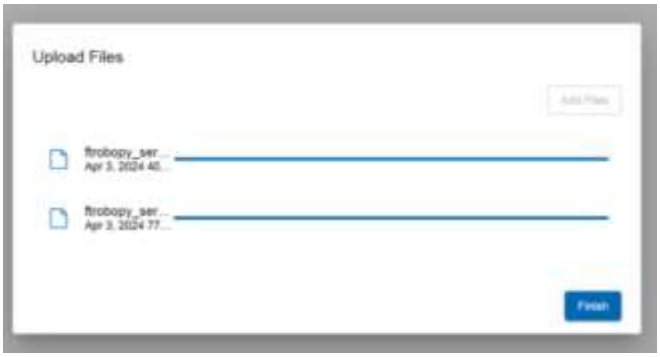
Die Datei erscheint in der Liste der zu übertragenen Dateien. Es muss aber noch eine weiter Datei dazu.
Diese findet sich im Ordner src .



Hier muss die Datei ftrobopy_server.cpp angeklickt werden und dann auf Öffnen klicken.

Das sind die Dateien, die wir brauchen und wir können „Upload“ anklicken.

Es wird der Ladefortschritt angezeigt, was sehr schnell geht.

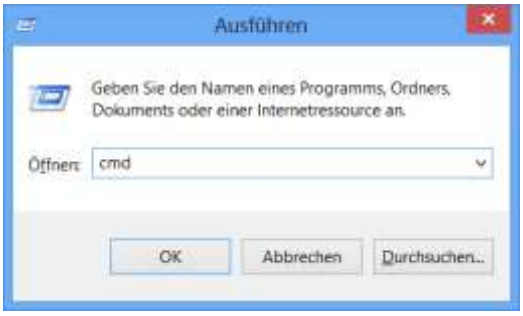


Anders als vermutet, führt ein Klicken auf „Finish“ nicht zum Schließen des Fensters.
Man muss die [Esc]-Taste drücken, um es zu schließen.

Die beiden hochgeladenen Dateien sind nun auf dem TXT 4.0.

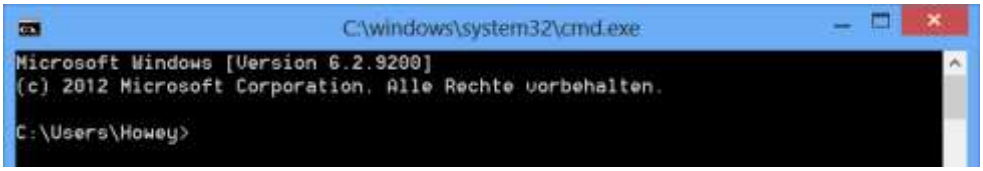
- Die Dateien als ausführbare Programme markieren (Execute-Flag), um sie für alle Nutzer ausführbar zu machen.

Dazu auf dem PC die Windows+R Tasten drücken und es erscheint folgendes Fenster:

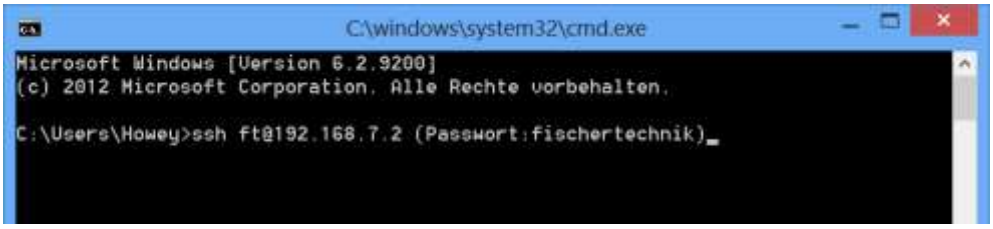


Hier cmd eingeben und OK drücken.

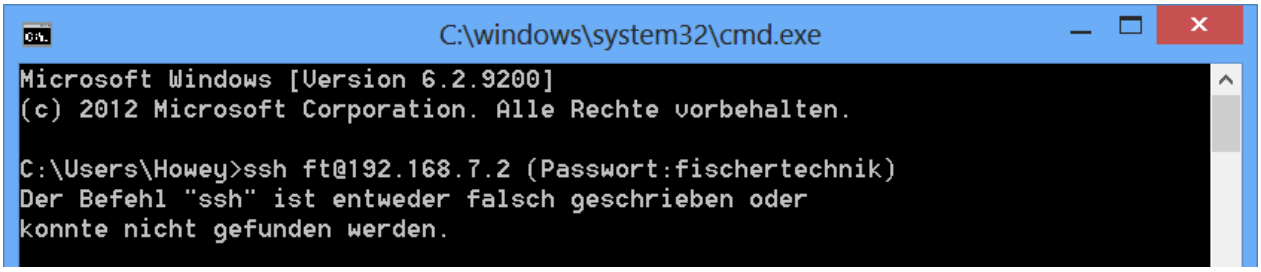
Hinweis: In Windows 8 geht es so nicht.



Hier muss man
ssh ft@192.168.7.2 (Passwort:fischertechnik)
eintippen. Zusätzlicher Fehler: Es ist falsch die Klammer & Text mit einzutippen!

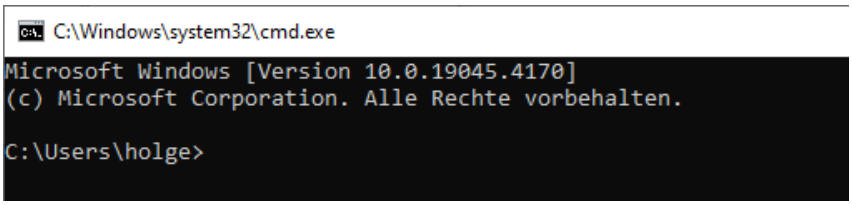


Und es kommt folgende Fehlermeldung:

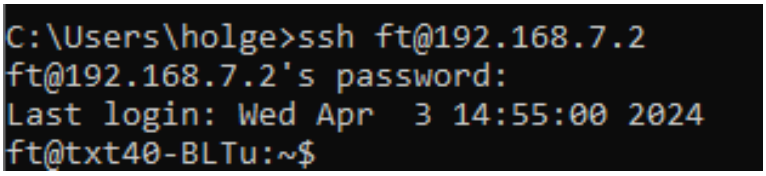


So wie es aussieht, muss bei Windows 8 erst Openssh installiert werden, um den ssh Befel nutzen zu können. Anleitungen dazu, gibt es von Microsoft selbst und auch Videos mit Schritt für Schritt.

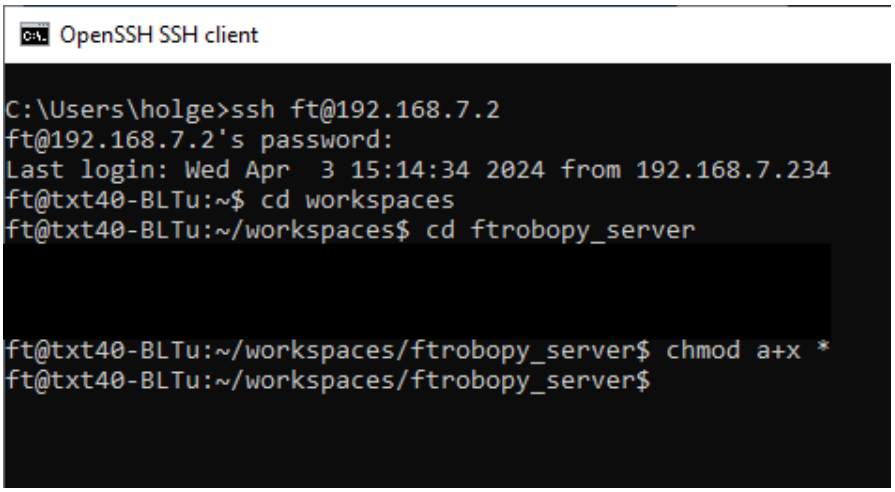
In Windows 10 kommt folgendes Fenster:



Hier muss man
ssh ft@192.168.7.2
eintippen. C:\\User\... wird bei jedem PC anders aussehen.



Man wird aufgefordert, das Passwort einzugeben. Hier muss man
fischertechnik
eintippen und die [Enter] Taste drücken. (Die Buchstaben sind nicht sichtbar)



Nun diese Befehle eingeben und jeweils die [Enter]-Taste danach drücken:
cd workspaces [Enter] Taste
cd ftrobopy_server [Enter] Taste
chmod a+x * [Enter] Taste

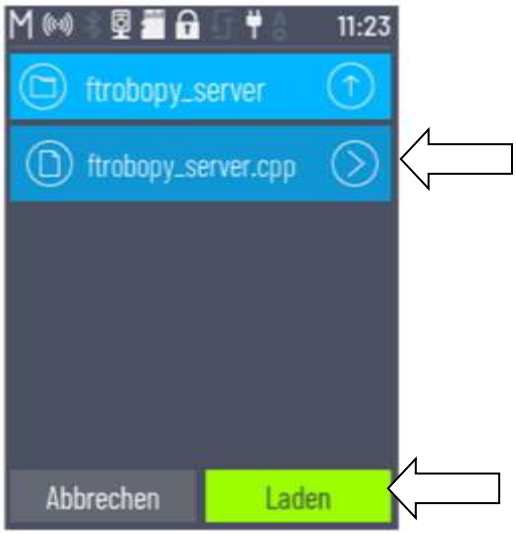
(beachte die Leerzeichen)

Das ftrobopy_server Programm kann nun, wie man es möchte, nach Bedarf oder automatisch beim Hochfahren des TXT 4.0 gestartet werden. Im Online-Betrieb kann man nun den TXT 40 mit Robo Pro und ftScratch benutzen. ftrobopy-Programme kann man Online und sogar Offline benutzen.

Hinweis: Man –kann- auch die Dateien, statt mit einem USB Kabel, über WLAN übertragen. Dann muss aber die Adresse entsprechend angepasst werden.

4. - Starten von ftrobopy_server auf dem TXT 4.0.

Im Verzeichnis Datei/ftrobopy_server, findet man nun die Datei ftrobopy_server.cpp. Wenn man diese antippt, färbt sich „Laden“ nach grün und man kann es laden und somit starten.



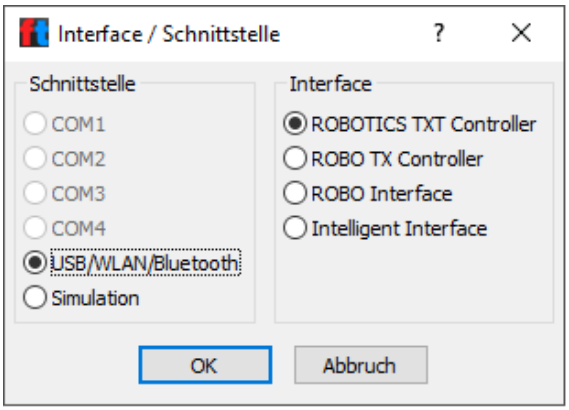
Nun wird der Startbildschirm beim TXT 4.0 mit dem roten Feld und dem Namen ftrobopy_server.cpp angezeigt. Diesen muss man anklicken und das Feld ändert seine Farbe in grün.

Es steht zwar dort, dass das Programm initialisiert wird, aber das Programm selber läuft und wartet auf Daten, um diese auszuführen.

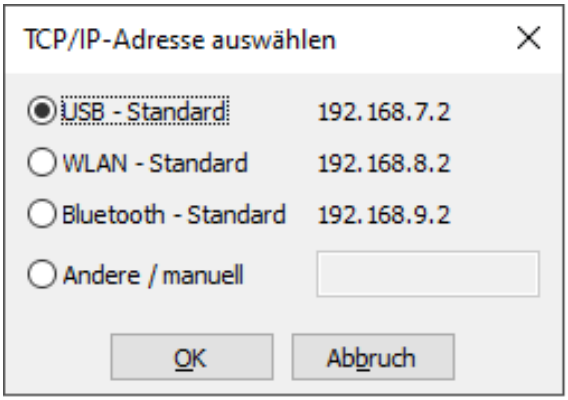
Zum Stoppen des Programms, einfach das grüne Feld antippen und das Feld wechselt die Farbe zu rot.

Nachdem Robo Pro 4.7.0 gestartet wurde, muss im Programm eingestellt werden wo der Controller zu finden ist.
ftrobopy muss auf dem TXT 4.0 laufen (grünes Feld wie oben angezeigt).

Dazu auf COM/USB klicken.

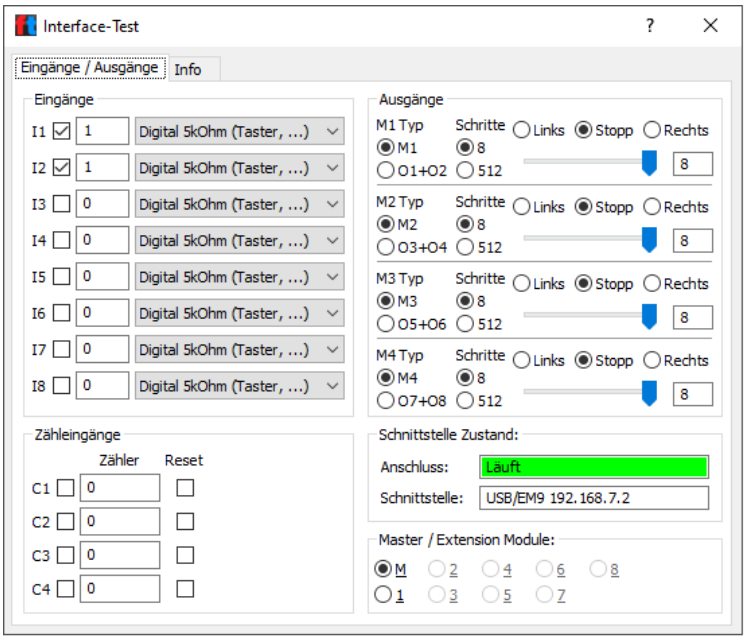


Nun links USB/WLAN/Bluetooth und rechts Robotics TXT Controller anklicken.

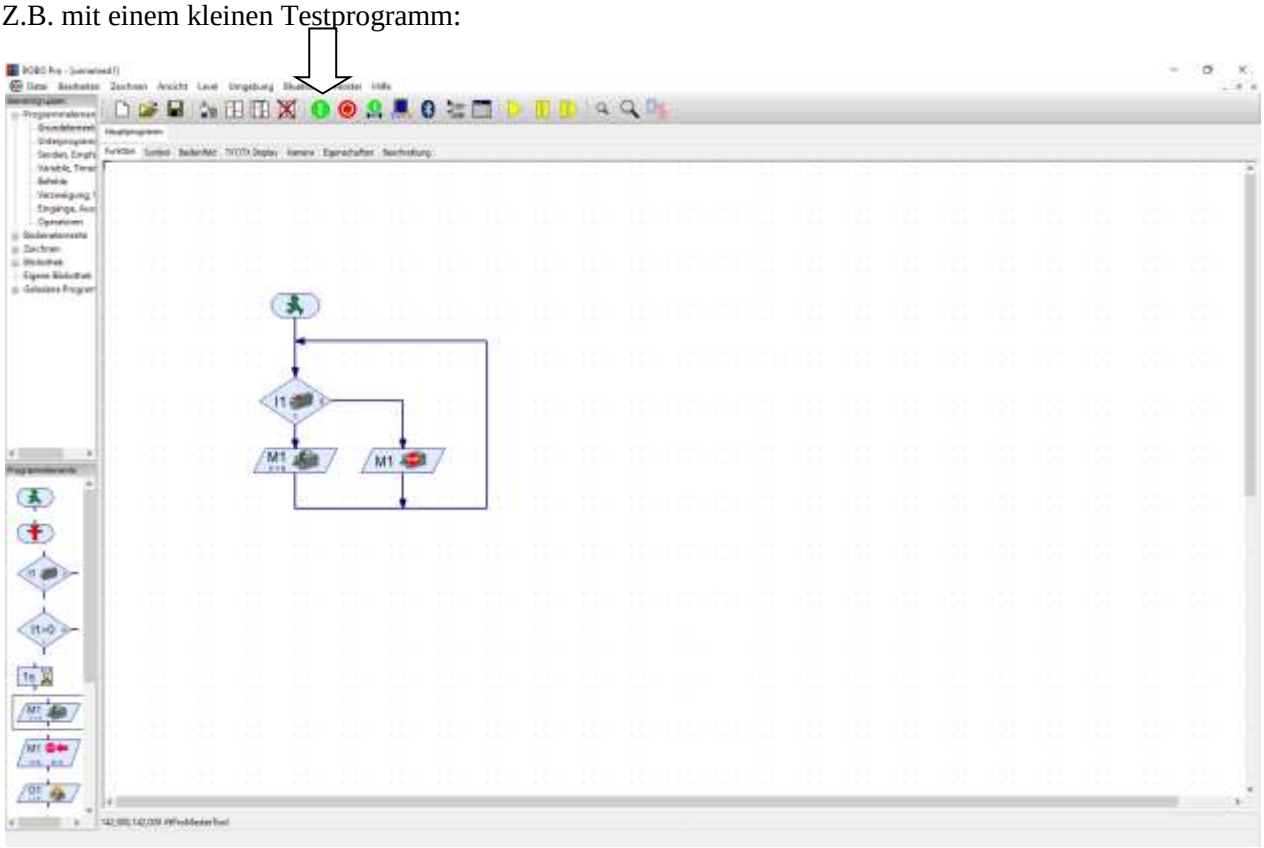


Hier den USB Anschluss auswählen und OK drücken.

Jetzt kann man den Interface Test anklicken.

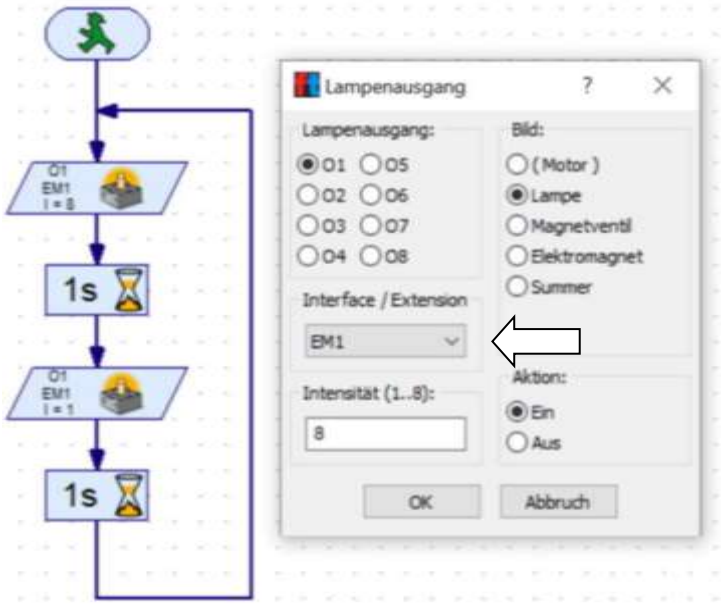


Und ein TXT 4.0 erscheint als TXT in Robo Pro. Nun kann man Motoren und Eingänge Testen.
Z.B. mit einem kleinen Testprogramm:



Zum Starten einfach auf den grünen Button drücken.
Das hier gezeigte Programm schaltet einen Motor M1 über den Taster an I1 aus und an.

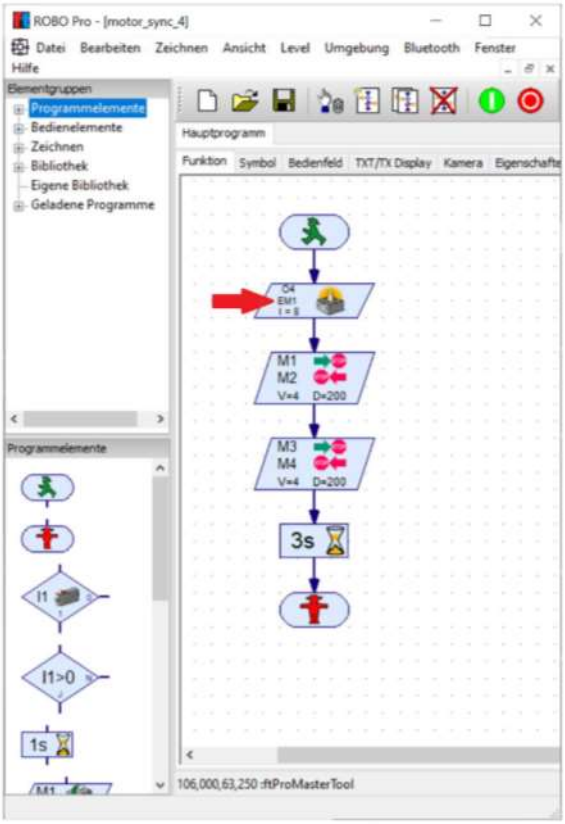
Um die drei Servos vom TXT 4.0 anzusteuern, werden die drei O-Ausgänge der ersten Extension EM1 benutzt.



Die Intensität der „Lampe“ (0-8) entspricht der Stellung des Servohebels. Man kann auch 0-512 nehmen, um mehr Stellungen zu bekommen. Die Werte 4 bzw. 256 sind die Mittelstellungen vom Servo.

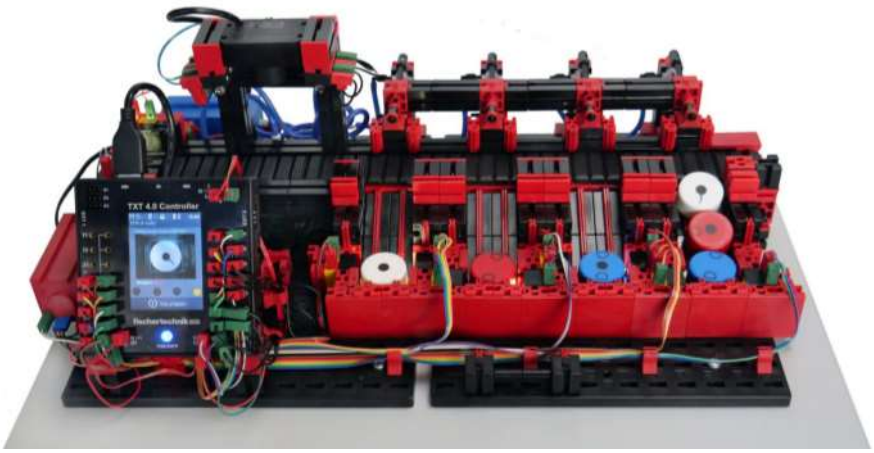
Synchronisation von vier Motoren

In Robo Pro können nur zwei Motoren synchron laufen. Mit ftrobopy ist es möglich alle vier Motoren synchron laufen zu lassen. Dazu wir der Ausgang O4 der ersten Extension EM1 auf 8 bzw. 512 gesetzt.



Hinweis:
Es ist noch nicht alles getestet worden. Es sollte mit ftrobopy möglich sein, den TXT 4.0 wie den TXT auch von C/C++-Programmen, Python und anderen Programmiersprachen anzusteuern.

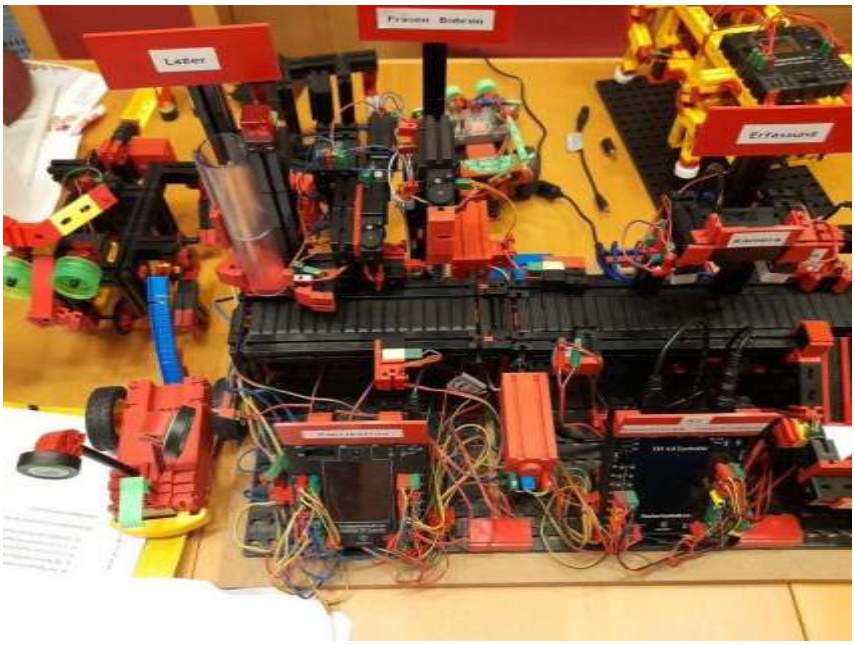
Qualitätssicherung mit KI(fischertechnik Industriemodell) mit dem TXT 4.0



Original Industriemodell von fischertechnik

Ich habe das Modell nachgebaut und auf einem neuen TXT 4.0 die ganzen Programme installiert. (Siehe dazu auch ftcommunity-Forum „Künstlich Intelligenz und fischertechnik“).

Am Ende ist das Modell von mir um eine Fabrik, die die Bauteile produziert und einen Farbsortierer, der den Ausschuss sortiert, erweitert worden.



Rechter Teil der Anlage mit Lager/Fabrik/KI (Auf der Makerfair 2023 in Hannover, mit anderen Modellen)

Linker Teil der Anlage mit Sortierung und Recycling (Farbsortierung)

Hier noch ohne die „Fabrik“.

Es gibt zwei PDF Dateien von fischertechnik mit „kurzen“ Erklärungen:

"Qualitätssicherung mit KI 9 V"
https://www.fischertechnik.de/de-de/produkte/industrie-und-hochschulen/simulationsmodelle/568416-qualitaetssicherung-mit-ki-9-v

Unter e-Learning gibt es Begleitmaterial
https://www.fischertechnik.de/de-de/industrie-und-hochschulen/technische-dokumente/simulieren/qualitaetssicherung-mit-ki

Es gibt 4 Speicherorte von Programmen und Daten:
1. PC mit Robo Pro Coding
2. TXT 4.0
3. fischertechnik Cloud
4. fischertechnik GitLab

Mit folgenden Programmen und Betriebssystemen bekommt man es zu tun:
Windows, Linux, Python, Microsoft Build Tools for C++, Tensorflow, Node-RED, Putty und die TXT 4.0 Anleitung.

Tensorflow, Node-RED… und co. sind schon auf dem TXT vorhanden. Man braucht sie nicht neu zu installieren. Python 3.9.13 als Windows Installer (64-bit) (!) und Microsoft Build Tools for C++ müssen installiert werden. Als Problem stellte sich die Path-Angabe raus. Wenn es Fehler über nicht gefundene Programme gibt, ist dies die Ursache. Auch sollte man dabei die Beschränkung auf 256 Zeichen aufheben.

Achtung: Node-RED muss auf dem TXT 4.0 aktiviert werden! Unter Einstellungen / Node-RED

Python 3.9.13 gibt es hier (ganz unten):
https://www.python.org/downloads/release/python-3913/

Microsoft Build Tools für C++
https://visualstudio.microsoft.com/de/visual-cpp-build-tools/

Man muss den TXT 4.0 auf die aktuelle Firmware über WLAN updaten. Dabei gibt es einen neuen API-Schlüssel für Robo Pro Coding. Man muss sich also neu in Rob Pro Coding anmelden.

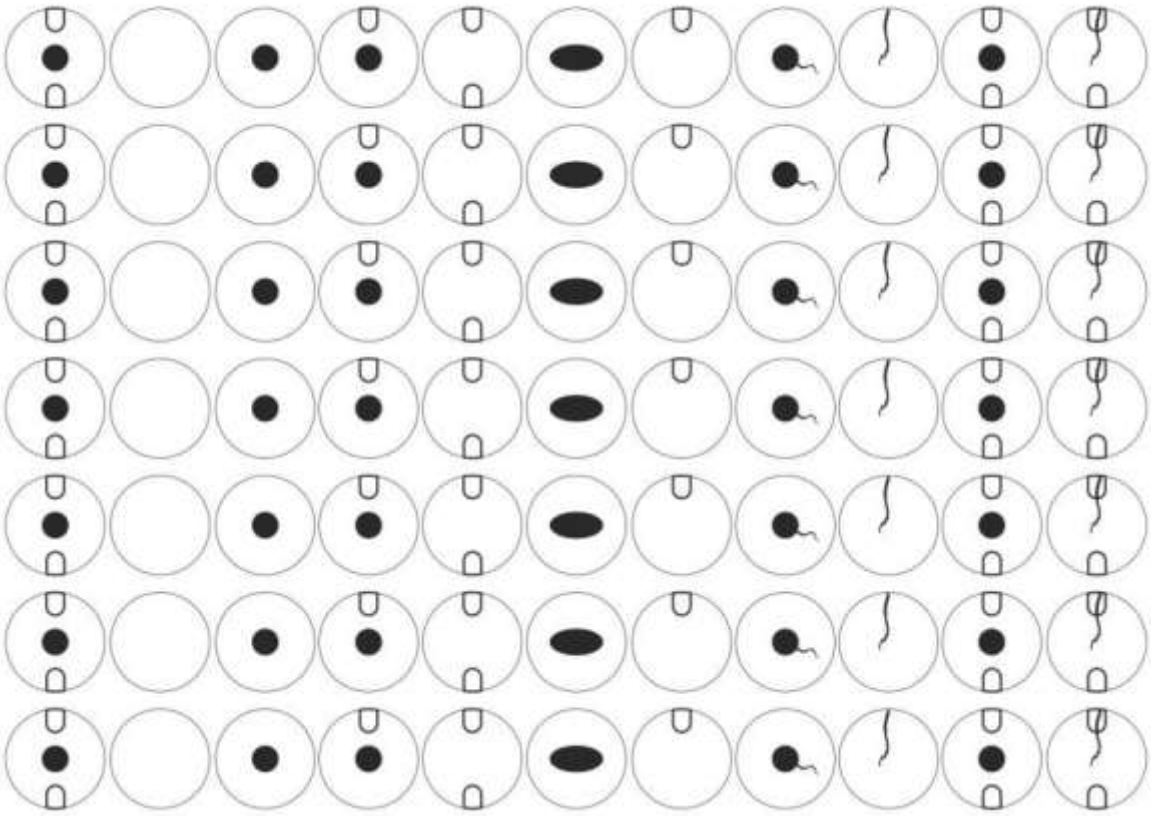
fischertechnik GitLab
https://git.fischertechnik-cloud.com/ml/machine-learning#tensorflow-example-project

Um mit Node-Red zu arbeiten, muss das Programm auf dem TXT 4.0 laufen, ansonsten kommt man nur auf die Hauptseite. Man muss Node-Red auf dem PC auch starten!

Tipps:
Was Hilfreich ist, wenn man etwas Gewicht in die Plastikkörper gibt. Entweder Unterlegscheiben oder Muttern. Der Übergang von einem zum anderen Band geht leichter. Wenn man etwas mit NFC machen möchte, kann man sich kleine Aufkleber in die Dosen kleben oder einfach nur reinlegen.

Für die Aufkleber gibt es von fischertechnik einen Aufkleberbögen. Ich habe mir selbst einen durchsichtigen Bogen gemacht. Sinnvoll ist, diesen eigenen Ausdruck noch mal mit einem leeren durchsichtigen Aufkleber zu überkleben. Im Dauerbetrieb gibt es einen Abrieb der unteren Aufkleber und die Bauteile werden fehlerhaft erkannt.

Tipp: Ich habe mir Vorlagen für den 3D-Drucker gemacht und die Bauteile farbig ausgedruckt.
Ist für die Zuschauer besser verstehbar und man hat keinen Abrieb.



Dadurch, dass die Bauteile Farbig sind, schimmert die Farbe durch die Aufkleber. Auch kann man auf beide Seiten des Bauteils verschiedene Aufkleber aufkleben.

Werte / Einheiten

Auflösung des Bandes/Steuerung (bei meinem Modell)
13,833 Punkte je cm (bei meiner Bandgeschwindigkeit 400)
Damit kann man die Werte der Buchten genau einstellen. Einfach die Entfernung des Bauteils zur Bucht messen, wo es liegengeblieben ist und mit 13.833 multiplizieren. Den Wert im Programm dazu addieren oder abziehen, je nachdem wo das Bauteil lag.

Abkürzungen/Begriffe im Programm

AI Artificial intelligence Kunstliche Intelligenz
API Application Programming Interfaces Anwendung /Programm
API-Schlüssel Sicherheitscode z.B. um PC-Programm Zugiff auf den TXT zu erlauben
BLANK Leer Teil ohne Eigenschaften
BOHO rundes Bohrloch (borehole)
BOHOEL Bohrloch Ellipse /ellyptisch (borehole ellyptical)
BOHOMIPO1 Bohrloch und einer Taschenfräsung (borehole milling pocket)
cd change direktory Wechsle Inhaltsverzeichnis(-ebene) / Ordner
Controller fischertechnik TXT 4.0 Interface
Confidence Wahrscheinlichkeit
Crack Riss
Duration Dauer
Encodermotor Motor mit Umdrehungszähler
Except Exception Ausnahmebehandlung Was machen bei einem Programmfehler.

Fototransistor Lichtabhängiger Schalter
GitLab Cloud-Speicher wo Programme und Daten geladen und gespeichert werden
GitHub Speicher wo Programme und Daten geladen und gespeichert werden
Hue Farbton
http Hypertext Transfer Protokoll
https Hypertext Transfer Protokoll Secure
IP-Adresse z.B. 179.168.178.123
IO PASSED In Ordnung
key Label Beschriftung
KI Künstliche Intelligenz
Klassifizieren Alle Eigenschaften einordnen / einteilen
Microsoft Build Tools for C++ Hiervon werden wohl Programmteile der Programmiersprache C++ benötigt. Wird sonst nicht weiter benötigt.
Log Ereignissdatei Logbuch ist in log.txt auf dem TXT 4.0 gespeichert
Logging Logdatei/Überwachung von Python zur Fehleranalyse z.B. nach einem unerwarteten Programmabbruch (Schreibfehler...)
MIPO1 Ausfräsung einer Tasche (milling pocket)
ML Machine Learning
Modell In dem Fall kein fischertechnik-Modell sondern ein KI-Modell
Ms Millisekunden Wird dann mit 1000 mal genommen um auf die Zeit in sec zu kommen.
NIO FAILED Nicht in Ordnung
Node-RED Grafische Programmieroberfläche z.B. für Daten auf einem Display...
Objekt Ding/Teil im Bild von Interesse. Nicht Objekt im Programm.
Pascal/VOC Datei um Eigenschaften eines Objekten zu speichern. Nicht die Programmiersprache Pascal.
PIP Python Package Index Installationsprogramm für Python und Programme, Daten...
Pfad Verzeichnisstruktur
Pos position Position
Powershell Windows Kommandozeileneingabe Windows/Suchen: Powershell
Prob probability Wahrscheinlichkeit
Python Programmiersprache
Python 3.9.13 Version der Programmiersprache Python
Sat Saturation Sättigung des Bildes
SLD Sorting Line (Demomodell, Demonstration, Dauerschleife ??)
SSH Secure Shell Gesichertes Netzwerkprotokoll z.B. um Programme zu übertragen.
STRG+C auch CRTL-C Taste[STRG] gedrückt halten und Buchstabentaste C drücken
Tensorflow 2.5.0. Die KI selbst und die Version
Ts Tensorflow
Terminal Eingabeaufforderung in Windows Start/Windows-System/Eingabeaufforderung
Timestamp Zeitstempel
TXT 4.0 fischertechnik Interface/Controller
Ventil Elektromagnetisches Ventil das Druckluft an/ausschaltet (2/3 Wegeventil)
WEB-Browser Browser Firefox, Chrom, Edge...
WEB Server Rechner der ein eigenes Netzwerk erschafft. Kann auch der TXT 4.0 sein.
Windows 10 64bit PC-Betriebssystem
WinSCP Programm, um Daten im Netzwerk zu übertragen

Zu erkennende Bauteile (Eigenschaften):

Crack Riss
MIPO1 Ausfräsung einer Tasche (milling pocket)
MIPO2 Ausfräsung zweier Taschen
BOHO rundes Bohrloch (borehole)
BOHOEL Bohrloch Elypse/ellyptisch (borehole ellyptical)
BOHOMIPO1 Bohrloch und einer Taschenfräsung (borehole milling pocket)
BOHOMIPO2 Bohrloch und zweier Taschenfräsungen
BLANK Leer Teil ohne Eigenschaften
NIO FAILED Nicht in Ordnung
num Nummer

Funktion des Programms (Zusammenfassung mit Erklärungen)

Hauptprogramm

thread_SLD = sorting_line
 thread_SLD Unterprogramm in sorting_line starten
 mqtt_client_forever MQTT Client wird gestartet (Node-RED)

sorting_line
  thread_SLD
   Variablen Initialisierung/Zuweisung
   mainSLDexternal_th() macht Endlosschleife mit mainSLDexternal_th()
    Ist ein Programmfehler?
     Fehler ausgeben
     clean_exit Abstellen (Motoren/LEDs) der Anlage und Programmabbruch

     mainSLDexternal_th
      Initialisierung von Variablen
      PartInGoodsReceipt Ist ein Teil in der Eingangslichtschranke?
      reset-interface
      Teil reinfahren
      Teil muss die Lichtschranke wieder freigeben ansonsten Programmabbruch
      Warte 10 ms --- warum auch immer... Zeit für openCV schinden?
      Fahre bis Kameraposition
      MakePictureRunKiReturnFoundPart() Mache vom Bild eine KI-Analyse (machine_learning)
      Wenn Teil = num
        1 fahre zur Bucht 1 und auswerfen (white)
        2 fahre zur Bucht 2 und auswerfen (red)
        3 fahre zur Bucht 3 und auswerfen (blue)
        4 fahre zur Bucht 4 und auswerfen (fail)
            Funktionen/Befehle
             AwaitBeltToReachPosition() hat der Motor die Position erreicht ansonsten warte
             SetBeltSpeedSteps(BeltSpeed, BeltSteps) Setze/fahre das Band mit den Schritten
             ejektWhite bei Weis auswerfen
             ejektRed bei Rot auswerfen
             ejektBlue bei Blau auswerfen

ejektFail bei Gelb auswerfen
istWhite ist das Bauteil in Weis angekommen?
istRed ist das Bauteil in Rot angekommen?
istWhite ist das Bauteil in Blau angekommen?
istFail ist das Bauteil in Gelb angekommen?
PartInGoodsReceipt Eingangslichtschranke belegt?
clean_exit ausstellen (Motoren/LEDs) der Anlage und Programmabbruch

machine_learning vom Bild eine KI-Analyse machen
MakePictureRunKiReturnFoundPart
Initialiesieren
reset_interface Beschriftungsfed löschen bzw "Not analysed" anzeigen
Foto-LEDs steuern
Zeitstempel
Variable auf Fail setzen
Foto holen
Mittelwert vom Foto holen
Farbe Mittelwert vom OpenCV holen
hue color[0] vom OpenCV holen
sat color[1] vom OpenCV holen
Motorcounter Zähler auf 0 setzen
Teil 200 vorlaufen lassen
ts_process0 Tensorflow Zeitstempel setzen für Timestamp?
detector für Objekt
ts_process1 Tensorflow Zeitstempel setzen
result Resultat auf Bild verarbeiten
ts_process Tensorflow Zeitstempel setzen gesamte Processingtime
get_color Unterfunktion Farbe auswerten
Foto-LEDs ausschalten
Ausgabe der Ergebnisse in log-Datei
Ergebnisse in die Variablen übergeben
Ausgeben der Ergebnisse über HTML
duration Dauer ermitteln
Unterfunktionen/ Befehle
reset_interface Beschriftungsfed löschen bzw. "Not analysed" anzeigen
containHTML Text über HTML ausgeben übertragen
get_color Unterfunktion Farbe auswerten hue=Farbton
Wenn (hue>=85 und hue<130) und sat>=40 dann color=3
sonst ((hue>=130 und hue<=180) oder (hue>=0 und hue<15)) und sat>=40 dann color=3
sonst color=1
timestamp Zeitstempel / Zeit holen
saveFileandPublish Name der Bilddatei vom letzen Bild (Foto) holen.
Rechteck aufspannen
logging.debug Fehler speichern
Foto speichern
holen vom ("chmod","777",filename)
Bildnamen... holen
Ergebnisse als Daten speichern, bzw. Display

Open CV2 Befehle - - Programm „machine-learning“
cv2.cvtColor
color = cv2.cvtColor(np.uint8([[[color[0],color[1],color[2]]]]),cv2.COLOR_BGR2HLS)[0][0]
Konvertiert ein Bild z.B. in Graustufen

cv2.rectangle
image = cv2.rectangle(frame, (pos[0], pos[1]), (pos[2], pos[3]), (180,105,0), 2)
Zeichne Rechteck im Bild

cv2.imwrite
cv2.imwrite(filename, frame)
Bild speichern/schreiben

Programm node_red
keine cv2 Befehle

Programm sorting-line
keine cv2 Befehle

Python Befehle
np.mean numpy.mean Mittelwert einer Reihe (vom Bild)

```
color = (np.mean(frame[ 80:120,   100:240], axis=(0, 1)))
```

MQTT
Unterprogramm mqtt_client_forever
Adresse IP Adresse des TXT 4.0 Zugang Port 1880 - Abgang Port 2883
http://txt.lokal:1880/ui Funktioniert nur bei einem TXT 4.0 im Netzwerk ansonsten IP Adresse des TXT 4.0 z.B. http://192.168.178.123:1880/ui

Initialisierung Unterprogramm node_red

```
def mqtt_client_forever():
    global imagedata, description, color, bay, confidence, duration, client, string, ts
    client = mqtt.Client("mqtt_client")
    client.connect('127.0.0.1',2883,60)
    #client.on_message=on_message
    #client.subscribe("#")
    client.loop_forever()
    return client
```

Übertragene MQTT Daten in puplish (mqtt_client_forever) Alles wird als String gesendet.
 imagedata
 description
 color
 bay
 confidence
 duration

Data IO
```
def publish(imagedata, description, color, bay, confidence, duration):
    global client, string, ts
    string = '{{ "ts":"{}", "description":"{}", "color":"{}", "bay":"{}", "confidence":"{}", "data":"{}",
"duration":"{}" }}'.format(timestamp(), description, color, bay, confidence, imagedata, duration)
    logging.debug("sending...")
    ret = client.publish("i/cam",string)
```

Add On: Künstliche Intelligenz

Deckblatt der Bauanleitung vom Kasten „Add On: Künstliche Intelligenz“ von fischertechnik

Vieles vom Industriemodell „Qualitätssicherung mit KI“ steckt in diesem Kasten. So kann man auch die ganzen Abkürzungen übernehmen. Der größte Unterschied liegt in dem Auswurf der Bausteine.

Hinweis: Wenn man versucht das Modell nachzubauen, - muss – man LEDs als Beleuchtung einsetzen. Mit ft-Lämpchen geht es nicht bzw. führt zu massenhaften Erkennungsfehlern. Man kann auch versuchen weiße Abdeckungen über die LEDs zu machen. Was sehr störend ist, sind Lampen oder Spiegelungen von Vitrinen, vor allem am Fototransistor am Eingang. (War auf einer Ausstellung bei meinem Modell so.)

Add On Industrial Robots

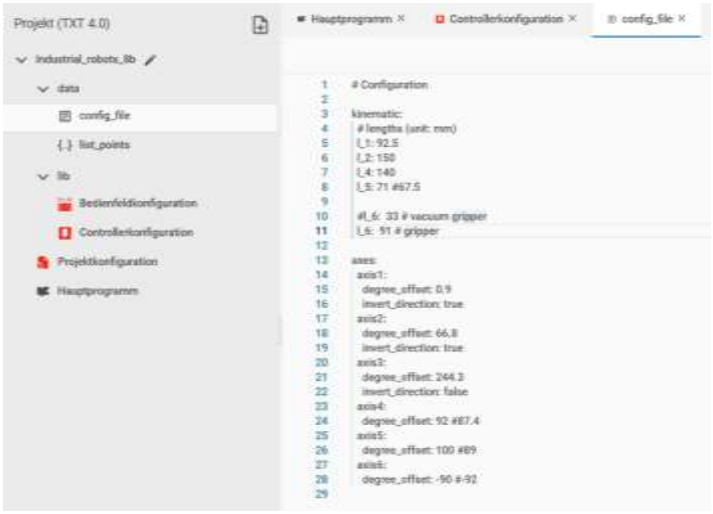
Dieser fischertechnik Zusatzkasten Kasten zum Robotics TXT 4.0 Base Set, ermöglicht es zwei Modelle von Robotern zu bauen.

Das Hauptprogramm für den Greifroboter ist Industrial_robots_8b.ft . Im Grunde kann man das Programm auch für selbstgebaute Roboter nehmen.

In der Datei „config_file“ sind die Daten und Maße des Roboters gespeichert.

Die Datei „list_points“ ist die Liste der gespeicherten Punkte, die der Roboter anfahren soll. Sie ist beim ersten Programmstart leer. Man kann die Werte in der Liste, vom Programm speichern oder laden.

Tipp zum Ausdrucken des Programms: Bei manchen Druckern, kann man Posterdruck einstellen. Das Programm ist sehr umfangreich. Bei diesem Programm sollte man mindesten 3x3 Din A4 (besser noch 4x4) wählen, um es noch lesen zu können.

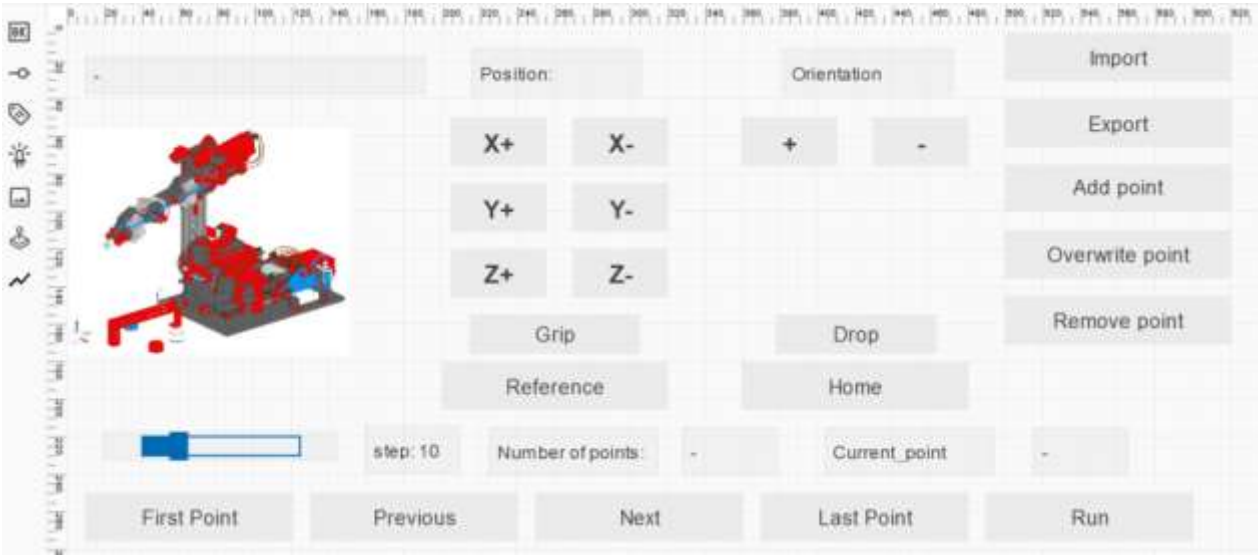


Configuration

kinematic: Hier sind die mechanischen Längen gespeichert
  # lengths (unit: mm)
  l_1: 92.5
  l_2: 150
  l_4: 140
  l_5: 71 #67.5

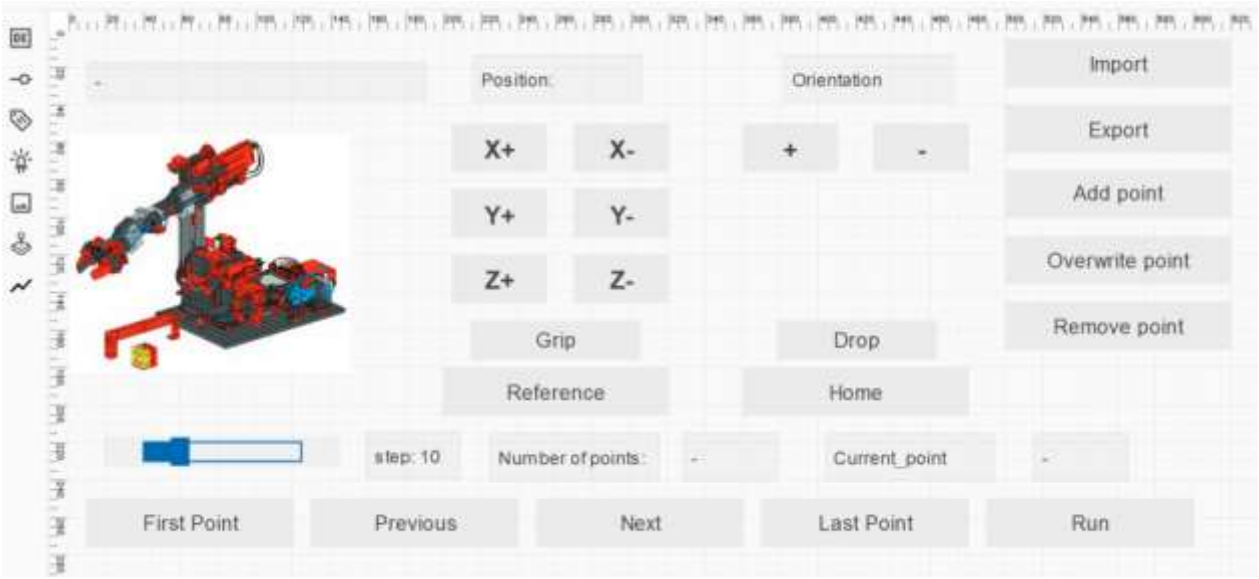
  #l_6: 33 # vacuum gripper
  l_6: 91 # gripper

axes: Hier werden die Startwerte und Drehrichtung vorgegeben
  axis1: Die Achsen 1-3 sind Encodermotoren
    degree_offset: 0.9
    invert_direction: true
  axis2:
    degree_offset: 66.8
    invert_direction: true
  axis3:
    degree_offset: 244.3
    invert_direction: false
  axis4: Die Achsen 4-6 sind Servomotoren
    degree_offset: 92 #87.4
  axis5:
    degree_offset: 100 #89
  axis6:
    degree_offset: -90 #-92



Interessanterweise ist im Bedienfeld der falsche Roboter als Bild.

Man kann das Bild löschen und durch ein eigenes Ersetzen.



Ich habe die PDF der Add ON Bauanleitung mir anzeigen lassen und dann ein Bildschirmfoto gemacht (Tasten [Alt] + [Druck]). Das habe ich in Paint eingefügt ([CTRL]+[V]), die Zeichnung beschnitten, Beschriftung wegradiert und als PNG gespeichert. Das Bild im Bedienfeld löschen und dann Datei hochladen und die Datei, die mit Paint gespeichert wurde, einfügen.
Das Projekt dann lokal speichern z.B. als „Industrial_robots_8b_neu“ – Fertig.

Programm / Funktionsübersicht vom Programm Industrial_robots_8b

zan_rot	Funktion (Zangenrotation?) Servo 3 Bereichsprüfung und setzen der Drehung
bt_rot_plus	Fernbedienung Servo 3 Drehung+step
bt_rot_minus	Fernbedienung Servo 3 Drehung-step
remote_slider_step	Fernbedienung Schrittweite (1 bis 40)
Programmstart	Hauptprogramm (grauer Textblock „zm“ kann gelöscht werden)
	Loggin einschalten
	robot_config laden
	Konfiguration der Achsen
	Initialisieren der Variablen
	Statusanzeige setzen (ist aber nicht im Bedienfeld vorhanden)
	Werte im Bedienfeld anzeigen
	Dauerschleife Abfragen des Kommandos cmd und ausführen
	Python Abfrage: Ist der Roboterarm im Bereich?
	Anfahren der Position
	cmd auf None setzen
Python-Importe	Util LIBs importieren, Roboter Unterprogramme laden
remote_button_grip	Fernbedienung Button-Abfrage Greifen
	Wenn gedrückt
	Ausgeben „grip“
	Ausführen der Funktion grip
remote_button_drop	Fernbedienung Button-Abfrage Lösen dito
remote_button_home	Fernbedienung Button-Abfrage Homeposition anfahren
remote_button_xp	Fernbedienung bt_Add - Ohne Funktion
remote_button_xp	Fernbedienung reset – Ohne Funktion
remote_button_xp	Fernbedienung Button-Abfrage x+
remote_button_zp	Fernbedienung Button-Abfrage z+
remote_button_xm	Fernbedienung Button-Abfrage x-
remote_button_ym	Fernbedienung Button-Abfrage y-
remote_button_zm	Fernbedienung Button-Abfrage z-
bt_reference	Fernbedienung Roboter referenzieren
bt_Add	Fernbedienung Button-Abfrage Liste Punkt hinzufügen
bt_export	Fernbedienung Button-Abfrage Liste exportieren
bt_import	Fernbedienung Button-Abfrage Liste importieren
first point	Funktion Ersten Punkt der Liste anfahren
	Wenn ein Eintrag in der Map/Liste vorhanden ist…
	Liste sortieren
	Zur ersten Position fahren
	Wenn greifen in der Liste steht, dann greifen
	Aktuellen Pointer auf k (=1) setzen
	Meldungen anzeigen

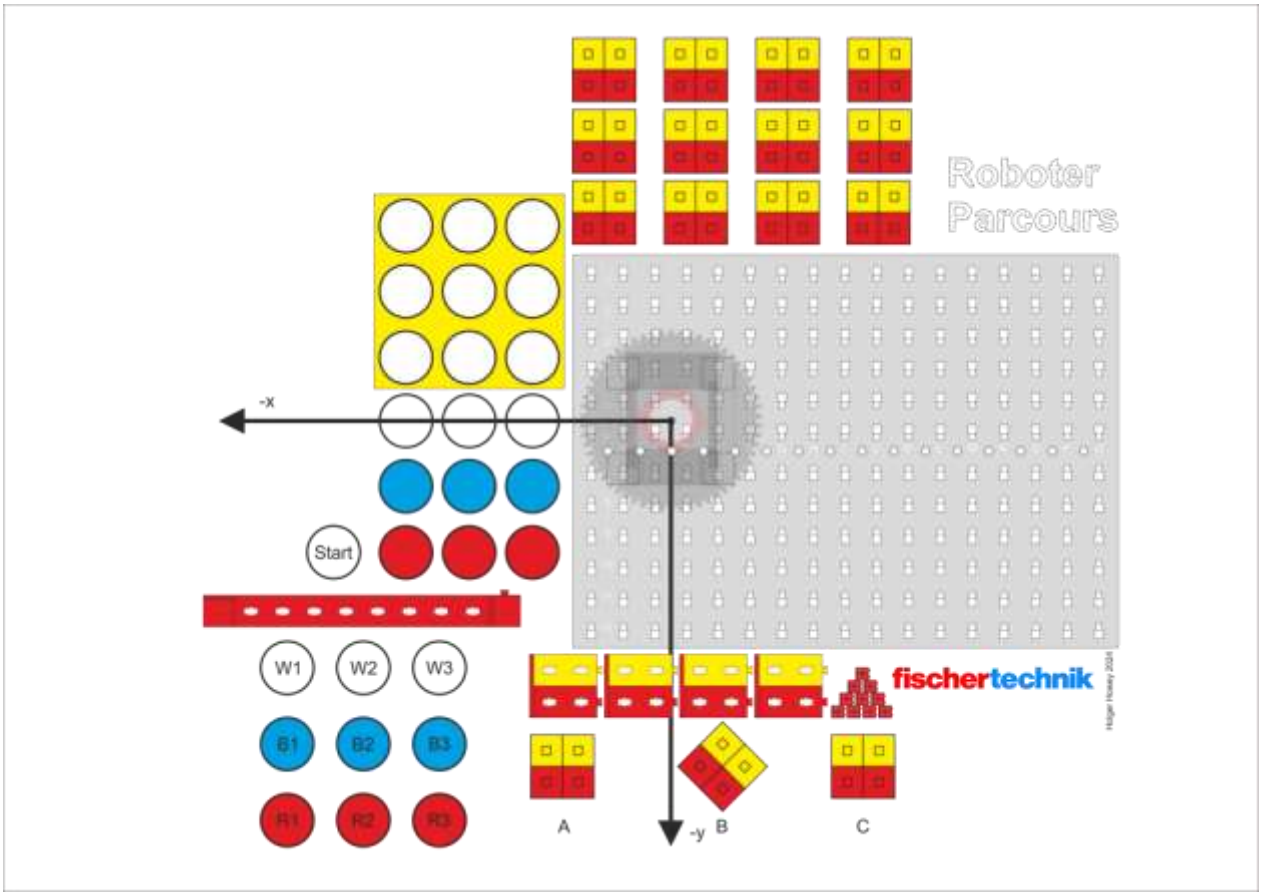
	Wenn die Liste leer sein sollte Fehlermeldung ausgeben.
last_point	Funktion Letzten Punkt der Liste anfahren
	dito mit first point, mur halt mit dem letzten Element der Map/Liste
new_position	Funktion Neue Position
	Wenn altuelle Position ungleich der neuen Position ist
	Variable return = wahr
	wenn nicht return = falsch
	return zurückgeben
grip	Funktion Greifen
	„Grip“ ausgeben
	Kompressor einschalten
	Variable grip = wahr
	1s warten
	Ventil einschalten
	1s warten
drop	Funktion Lösen
	„Drop“ ausgeben
	Kompressor und Ventil ausschalten
	Variable grip = falsch
	1s warten
move_ptp	Funktion Punkt zu Punkt (anfahren)
	Übergabe der neuen Position und der Wartezeit
	Python-Code Umrechnung der Werte
	Ausführung der Bewegung
	Ausgabe der Position und vom Status
bt_overwrite	Fernbedienung Button-AbfragePunkt überschreiben
	Wenn Schaltfläche gedrückt
	Punkt in die Map/Liste eintragen und ausgeben
	Meldungen ausgeben
bt_button_run	Fernbedienung Button-Abfrage Punkte automatisch abfahren
	Wenn Taste gedrückt ist
	Anfahren vom ersten Punkt
	Alle anderen Punkte nacheinander abfahren
bt_First	Fernbedienung Button-Abfrage Erster Punkt
	Wenn Taste gedrückt ist
	Zum ersten Punkt der Liste hinfahren
bt_last	Fernbedienung Button-Abfrage Letzter Punkt dito
bt_pre	Fernbedienung Button-Abfrage Punkt zurück dito
bt_next	Fernbedienung Button-Abfrage Nächster Punkt dito
next_point	Funktion Nächsten Punkt anfahren
	Sind mehrere Einträge vorhanden?
	Sortiere Einträge nach aufsteigender Nummer
	Ist letzter Eintrag erreicht?
	Dann Meldung ausgeben

	Ansonsten nächsten Punkt k holen
	Neue Position und Rotation anfahren
	und greifen oder lösen.
	Aktuelle Punktnummer = k
	Meldungen ausgeben
previous_point	Funktion Davorliegenden Punkt anfahren dito
bt_remove	Fernbedienung Button-Abfrage Punkt löschen
	Ist ein Eintrag vorhanden?
	Wenn ja, Punkt löschen
	Meldung ausgeben Punkt löschen
	Ersten Punkt anfahren
	Ansonsten Meldung ausgeben, keine Punkte

Tipp zum Greifroboter:
Wenn man die Bauteile hat, kann man den Greifer breiter bauen, in dem man links und rechts jeweils einen Baustein 163439 (15x 30 rund rot) mit Federnocken 31982 anbaut. Man muss aber ein Antirutsch Gummi (Baumarkt) aufkleben.
Man kann auch normale 15x30er Bausteine nehmen. Bei meinem Modell habe ich nicht rote, sondern grüne 163202 verbaut. Das fällt mehr auf.

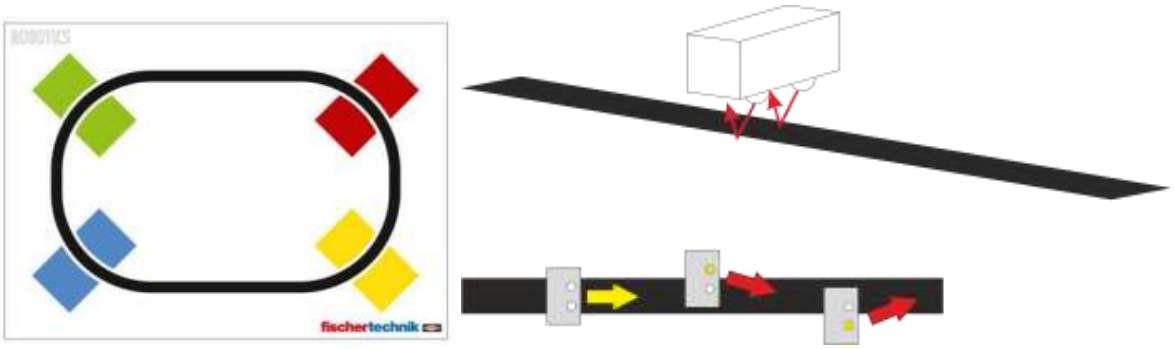
Das Programm greift auf eine LIB zurück, die man unter:
https://github.com/dmholtz/ft_robot/tree/ft_fw_txt40/robotic_arm
findet. Auf dem TXT 4.0 sind die Dateien unter: /usr/lib/python3.5/site-packages/robotic_arm
In der Datei robot.py werden die einzelnen Achsen definiert. Also welche einen Encodermotor oder Servo hat und welche Übersetzung Zahnkranz (58 oder 40) die haben.

Ich habe mal einen eigenen Roboter Parcours, für den Greifroboter gezeichnet. Wenn man mehrere rot/gelbe Bausteine hat, kann man die auch stapeln oder auf einen Lagerplatz ablegen. Die Grafik muss in 2x2 Din A4 Seiten ausgedruckt werden. Für ganz geübte Programmierer ist dann noch das Spiel tic tac to vorgesehen. Und für diejenigen die einen Zweiten Roboterarm haben: Die können dann auch noch gegeneinander spielen. Nur so als Ideen…



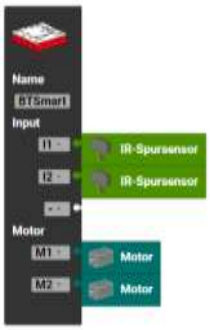
Beispiel Spurensucher Funktionsweise

Dieses Programm soll einem Roboter mit zwei Antriebsrädern, einem Stützrad und einem IR-Spurensensor(128598) ermöglichen, eine Spur entlang zu fahren. Wenn er die Spur verlieren sollte, soll er stehen bleiben.

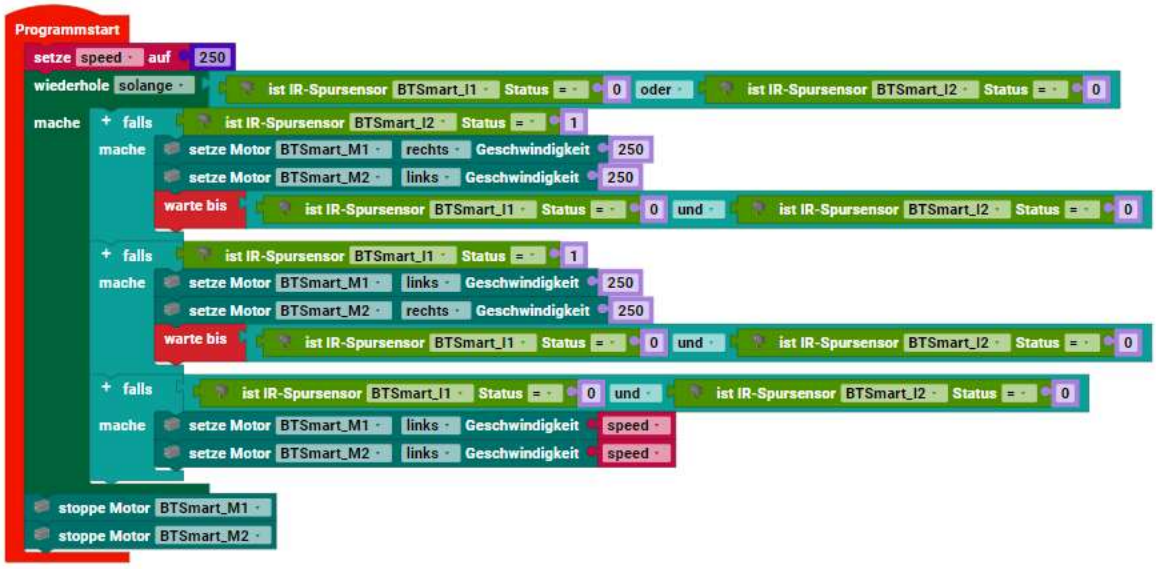


Beispiel eines Parcours von fischertechnik. Hier aus dem Kasten Smart Tech. Man kann auch schwarzes Isolierband nehmen und auf eine weiße Holzplatte kleben. Durch Gegenlenken werden beide Sensoren 0.

BT Smart Controller Spurensucher



BT Smart Controllerkonfiguration

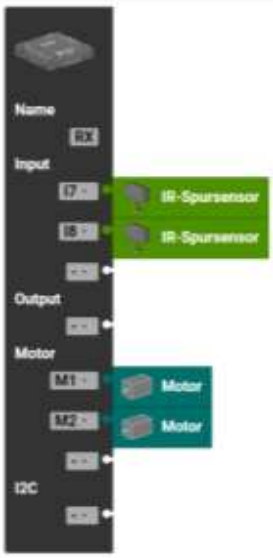


Das Programm setzt die Variable „speed“ auf 250. Solange die Bedingung vom Spurensucher erfüllt ist, dass entweder der erste oder der zweite IF Sensor 0 ist (und somit eine Spur hat) wird abgefragt ob der zweite IR Sensor 1 ist (keine Spur hat). Wenn ja wird so lange gedreht bis die Spur wieder da ist (beide 0).

Nun dasselbe in der anderen Richtung.
Wenn beide die Spur haben (beide 0) vorwärtsfahren. Wenn beide keine Spur haben, werden die Motoren abgeschaltet und das Programm beendet.

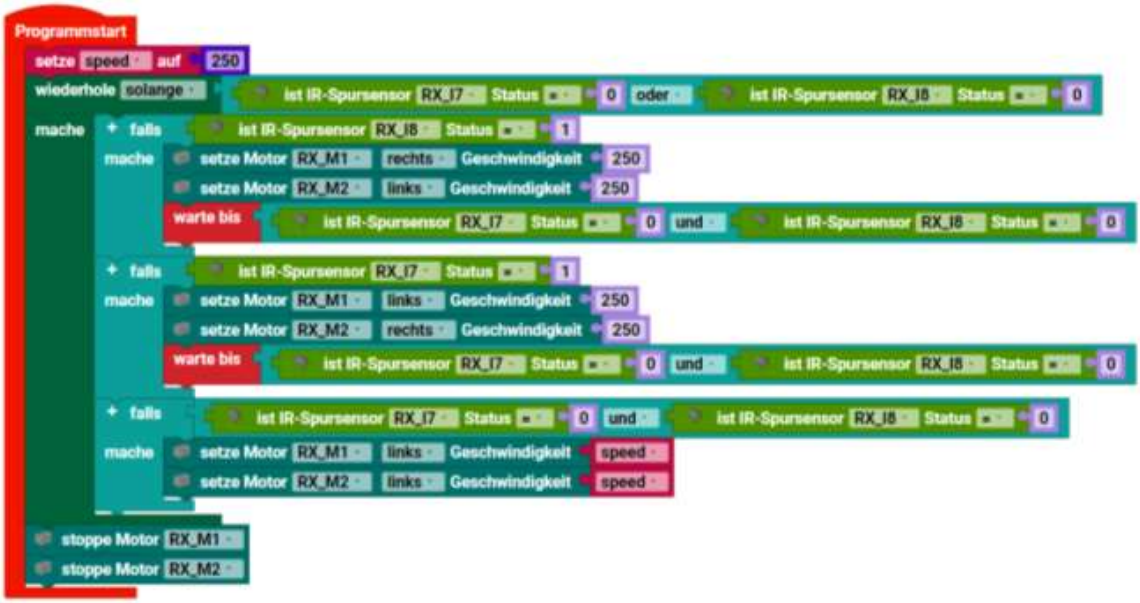
RX Controller Spurensucher

Dasselbe Programm für den RX-Controller.

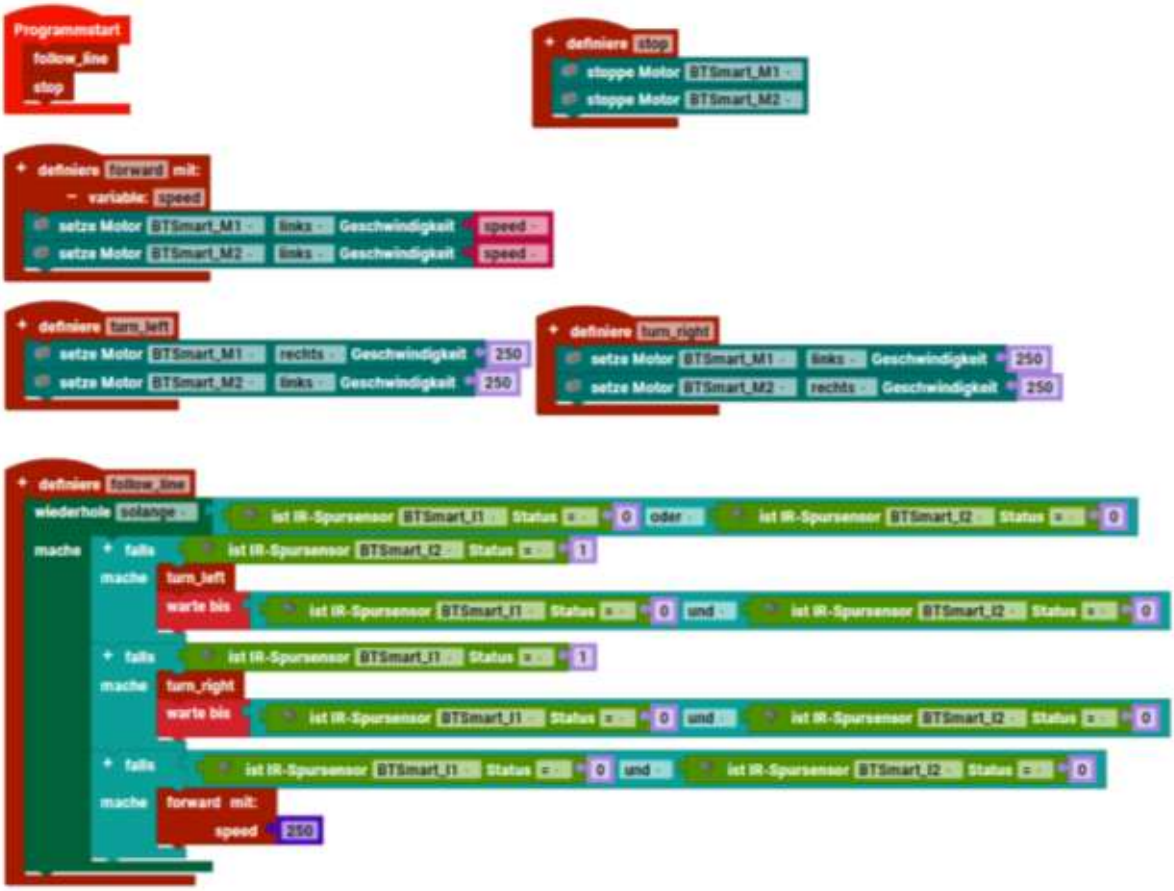


RX Controllerkonfiguration

Der Unterschied ist nur in den Eingängen I7 / I8 statt I1 und I2 beim BT Smart.



Man kann nun das Programm zerlegen und aus den Bereichen „Rechts, Links, Geradeaus und Stopp“ Unterprogramme machen. Selbst das Hauptprogramm kann man noch in ein Unterprogramm „fahre Spur“ machen. So hätte man nur noch „fahre Spur“ und „Stopp“ im Hauptprogramm.


Programmstart
follow_line
stop
definiere stop
stoppe Motor BTSmart_M1
stoppe Motor BTSmart_M2
definiere forward mit:
variable: speed
setze Motor BTSmart_M1 links Geschwindigkeit speed
setze Motor BTSmart_M2 links Geschwindigkeit speed
definiere turn_left
setze Motor BTSmart_M1 rechts Geschwindigkeit 250
setze Motor BTSmart_M2 links Geschwindigkeit 250
definiere turn_right
setze Motor BTSmart_M1 links Geschwindigkeit 250
setze Motor BTSmart_M2 rechts Geschwindigkeit 250
definiere follow_line
wiederhole solange
ist IR-Spursensor BTSmart_I1 Status = 0 oder ist IR-Spursensor BTSmart_I2 Status = 0
mache
falls ist IR-Spursensor BTSmart_I2 Status = 1
mache turn_left
warte bis ist IR-Spursensor BTSmart_I1 Status = 0 und ist IR-Spursensor BTSmart_I2 Status = 0
falls ist IR-Spursensor BTSmart_I1 Status = 1
mache turn_right
warte bis ist IR-Spursensor BTSmart_I1 Status = 0 und ist IR-Spursensor BTSmart_I2 Status = 0
falls ist IR-Spursensor BTSmart_I1 Status = 0 und ist IR-Spursensor BTSmart_I2 Status = 0
mache forward mit:
speed 250

Omniwheels mit Robo Pro Coding

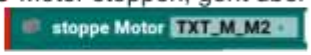
Grundlagen zu den fischertechnik Omniwheels

XS-Motor (Minimot) (Achte auf das kleine ***schwarze*** Symbol links im Block)

Wenn du mit der ROBO Pro Coding Software arbeitest und Motorantriebe programmierst, benutzt du für XS-Motoren den Block „setze Motor ... Geschwindigkeit".

Die Drehrichtung kann im Programm durch die Auswahl „links" und „rechts" geändert werden.

Den XS-Motor stoppen, geht über den Block „stoppe Motor"



Encodermotor (Achte auf das kleine ***rote*** Symbol links im Block)

Wenn du mit der ROBO Pro Coding Software arbeitest und Motorantriebe programmierst, benutzt du für Encodermotoren den Block „setze Motor ... Geschwindigkeit".

Die Drehrichtung kann im Programm durch die Auswahl „links" und „rechts" geändert werden.

Den Encodermotor stoppen, geht über den Block „stoppe Motor"

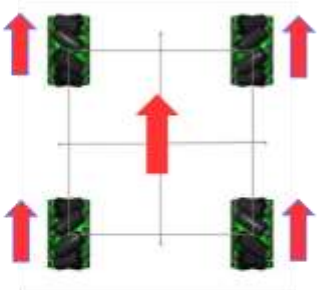
Nachfolgende Darstellungen zeigen dir die verschiedenen Antriebe der Mecanum-Räder. Die äußeren farbigen Pfeile entsprechen der Antriebsrichtung des Rades, der rote Pfeil in der Mitte die Bewegungsrichtung deines Fahrzeugs.

Motordrehrichtung:

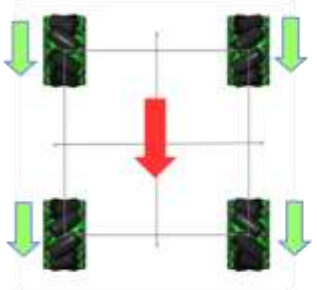
Links (rot)  Rechts (grün) 

Tipp:
Schreib dir an die Zeichnung, deine M-Ausgangs-Nummer dran.

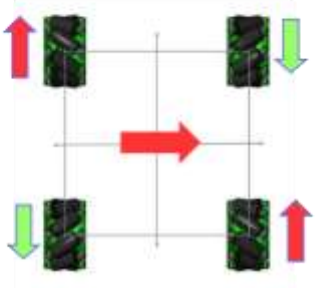
Fahrtrichtung: Vorwärts



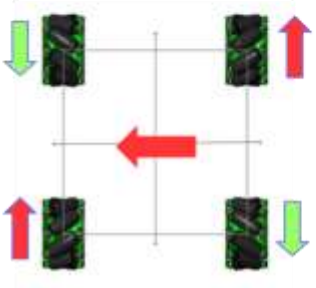
Rückwärts



Fahrtrichtung: Seitwärts rechts



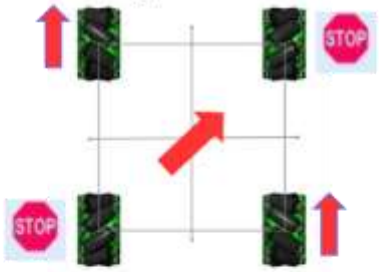
Seitwärts links



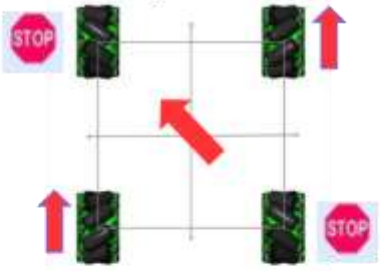
H. Howey 2024

Quellen Grundlage, fischertechnik: https://www.fischertechnik.de/-/media/fischertechnik/fite/service/elearning/spielen/smartech/aufgaben/grundlagenomniwheels.ash

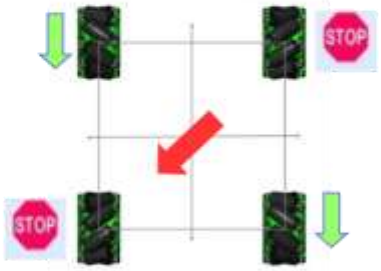
Fahrtrichtung: Diagonal rechts vor



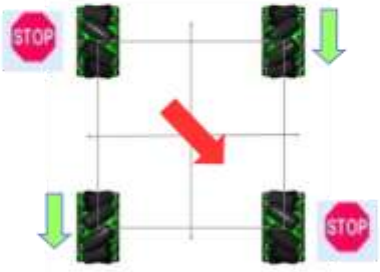
Diagonal links vor



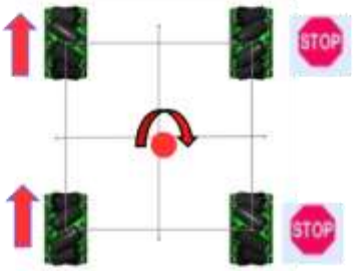
Diagonal rechts rückwärts



Diagonal links rückwärts



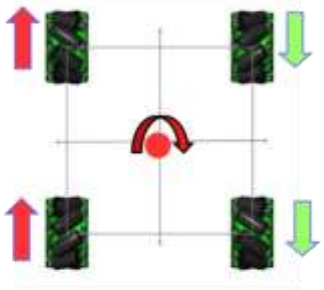
Kurvenfahrt: Rechts



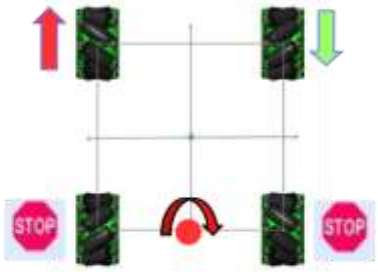
Links

Für größere Kurven, statt Stop die Motoren langsam laufen lassen.

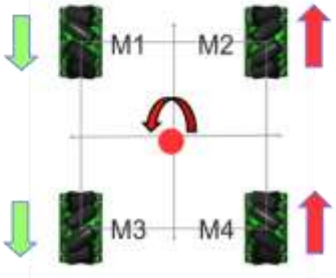
Drehung: Rechts



Drehung um den Mittelpunkt einer Achse



Benötigst du noch weitere Bewegungsrichtungen, musst du nur die Drehrichtung der Motoren ändern. z.B. Drehung nach Links



H. Howey 2024

Einige Beispiel-Programme anderer fischertechnik Kästen von anderen Programmiersprachen in Robo Pro Coding umgesetzt

Robotics BT Beginner

Dieser Kasten ist der überarbeitete BT Smart Beginner mit anderer Programmiersprache. In diesem Fall mit dem aktuellen Robo Pro Coding.

Laufkarte für Schüler /Übersicht der Modelle

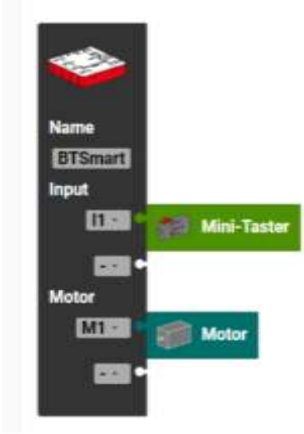
BT Smart Beginner

BT Smart Beginner Kasten von fischertechnik – u.a. mit RoboPro Light.

Die Bauanleitung und Begleitmaterial können kostenlos bei fischertechnik runtergeladen werden.

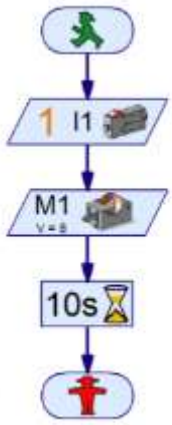
Schul-Kasten

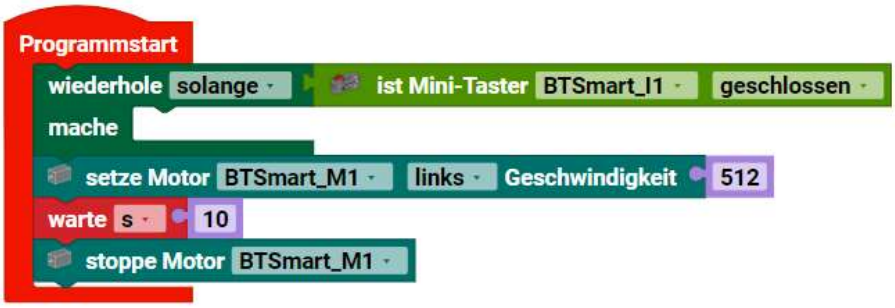
Karussell 3:



Modell Karussell BT Smart Controllerkonfiguration

RoboPro Light Programm: Robo Pro Coding Programm:





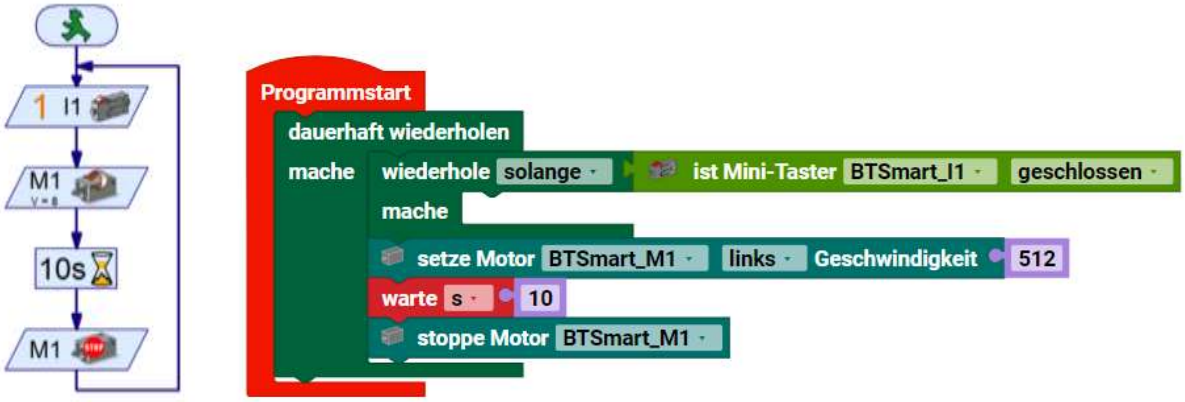
Blöcke kommen aus: Schleifen, Eingang, Motor und Util. Das Programm läuft einmal durch. Es startet und dann wartet es, bis der Taster geschlossen ist. Dann wird der Motor M1 auf volle Geschwindigkeit gesetzt, er läuft 10 Sekunden und wird dann wieder abgeschaltet.

Hinweis Fehlerteufel Wackelkontakt:

Erst dachte ich, ich habe mich vertan. Bei mir liefen meine eigenen Programme nicht. Ich habe viel versucht und ausprobiert. Es hat lange gedauert, aber als Übeltäter stellte sich ein Stecker raus, der vorne etwas schmal war und nur hin und wieder Kontakt mit der Buchse hatte. Etwas Aufbiegen und es lief alles…

Karussell 4:

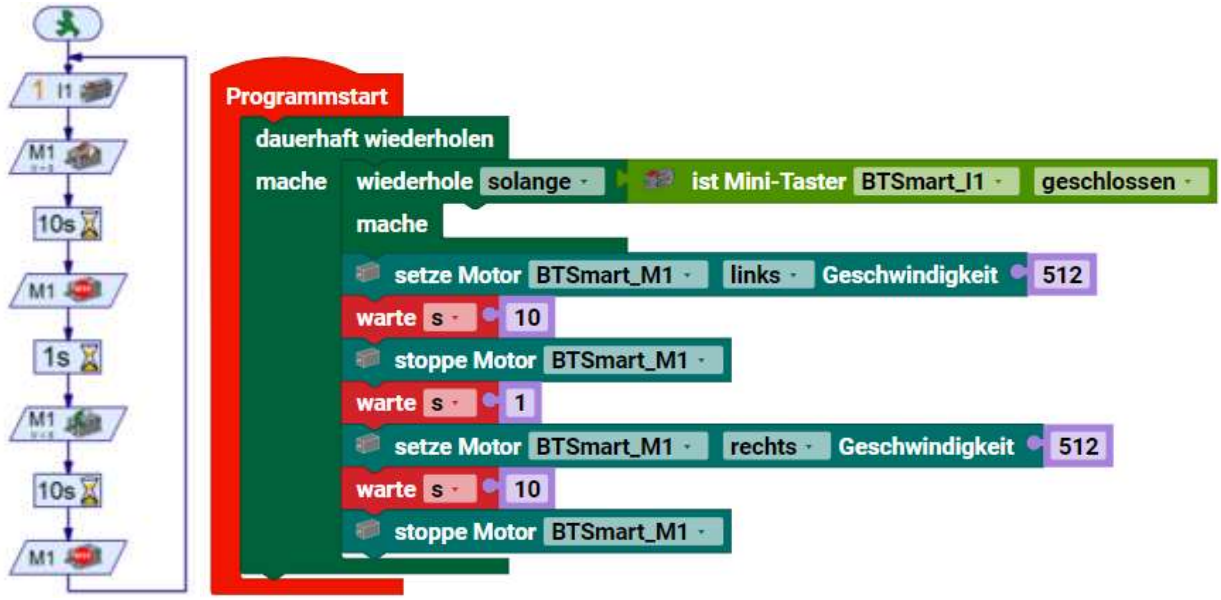
RoboPro Light Programm: Robo Pro Coding Programm:



Hier ist eine Endlosschleife. Nach dem Start wird auf den Taster gewartet, bis er offen ist. Dann wird der Motor gestartet und nach 10 Sekunden wieder abgeschaltet. Ein neuer Durchlauf startet.

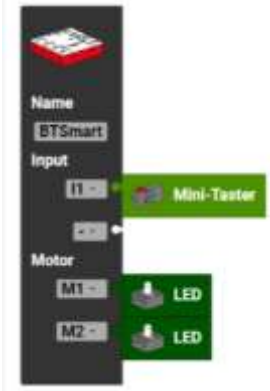
Karussell 5

RoboPro Light Programm: Robo Pro Coding Programm:



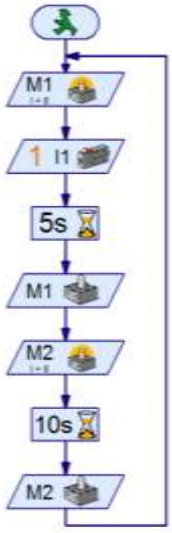
Auch hier ist eine Endlosschleife. Nach dem Start wird auf den Taster gewartet, bis er offen ist. Dann wird der Motor gestartet und nach 10 Sekunden wieder abgeschaltet. Der Motor steht für 1 Sekunde und wird mit anderer Drehrichtung gestartet. Nach 10 Sekunden wird er wieder abgeschaltet und ein neuer Durchlauf startet.

Fußgängerampel 1

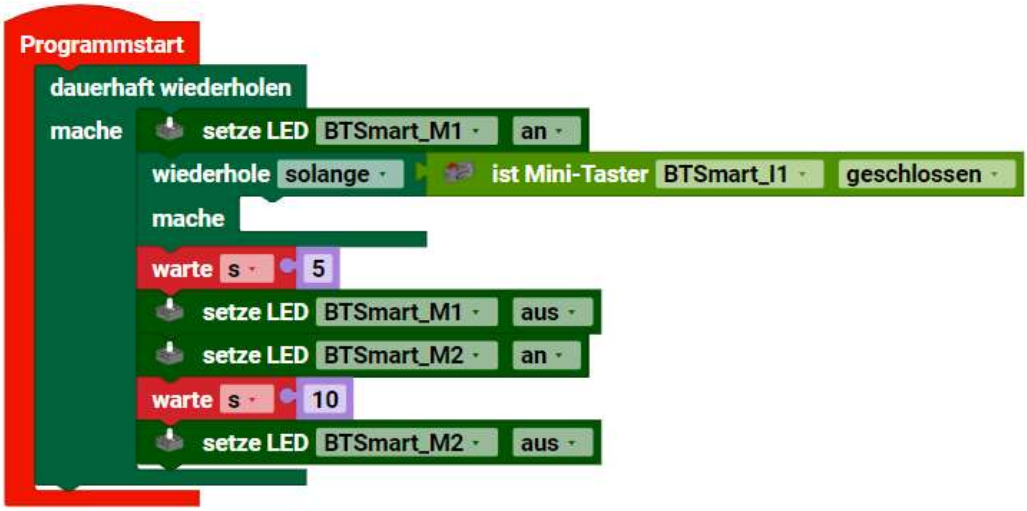


Modell Fußgängerampel BT Smart Controllerkonfiguration

RoboPro Light Programm:



Robo Pro Coding Programm:

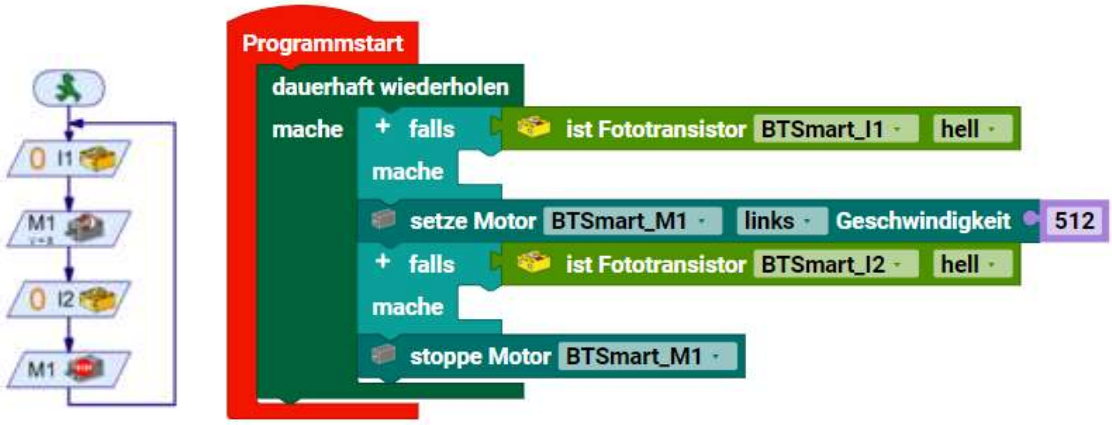


Das Programm wird in einer Endlosschleife gestartet. Die LED 1 (Rot) wird eingeschaltet und es wird auf den Taster gewartet, bis er gedrückt (geöffnet) wird. Nach 5 Sekunden wir die LED 1 aus und die LED 2 (Grün) eingeschaltet. Nach 10 Sekunden wird die LED 2 ausgeschaltet und die Schleife läuft wieder von vorne.

Förderband 1

Modell Förderband BT Smart Controller Controllerkonfiguration

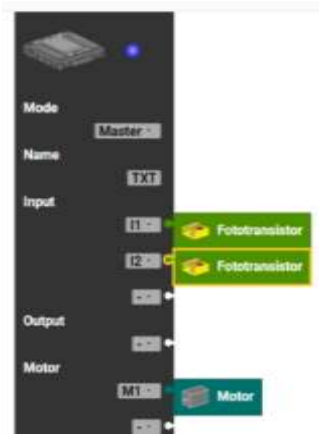
RoboPro Light Programm: Robo Pro Coding Programm:



Hauptprogramm

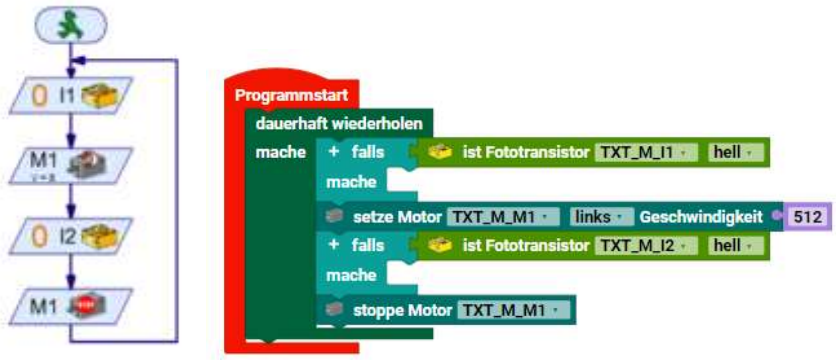
Das Programm läuft in einer Dauerschleife. Es wartet, bis ein Bauteil in der ersten Lichtschranke steht. Dann wird der Motor eingeschaltet und gewartet bis das Bauteil in der zweiten Lichtschranke ist. Dann wird der Motor abgeschaltet und es beginnt wieder von vorne.

Förderband 1 TXT 4.0 Controller statt BT Smart Controller
Hier für den TXT 4.0 Controller statt dem BT Smart Controller. Kann 1:1 für den BT Smart Controller übernommen werden. Es ändern sich nur die Bezeichnungen im Programm. Zum Beispiel wird aus TXT_M_I1 nun BT_Smart_I1. Beim Erstellen wird das automatisch eingefügt.



Förderband Controllerkonfiguration

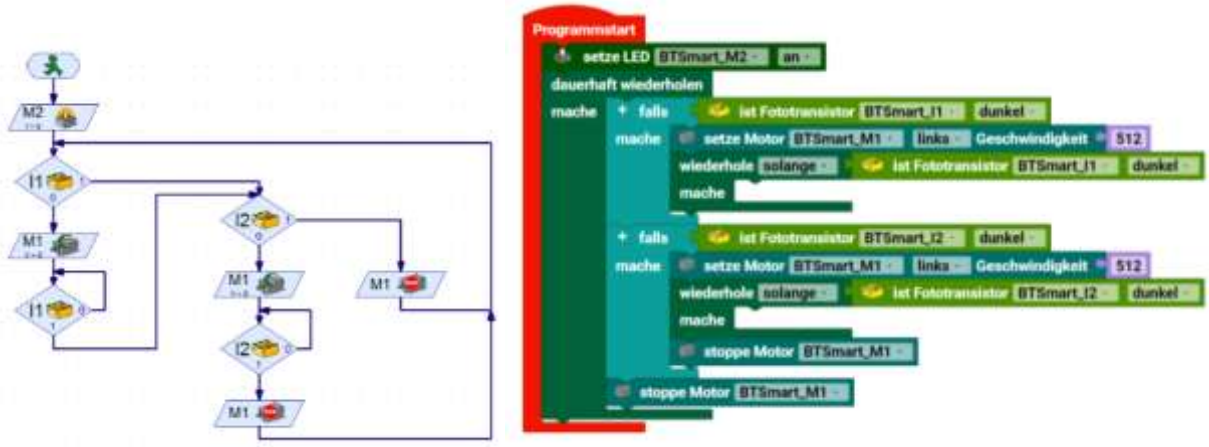
RoboPro Light Programm: Robo Pro Coding Programm:



Händetrockner

BT Smart Controller Controllerkonfiguration

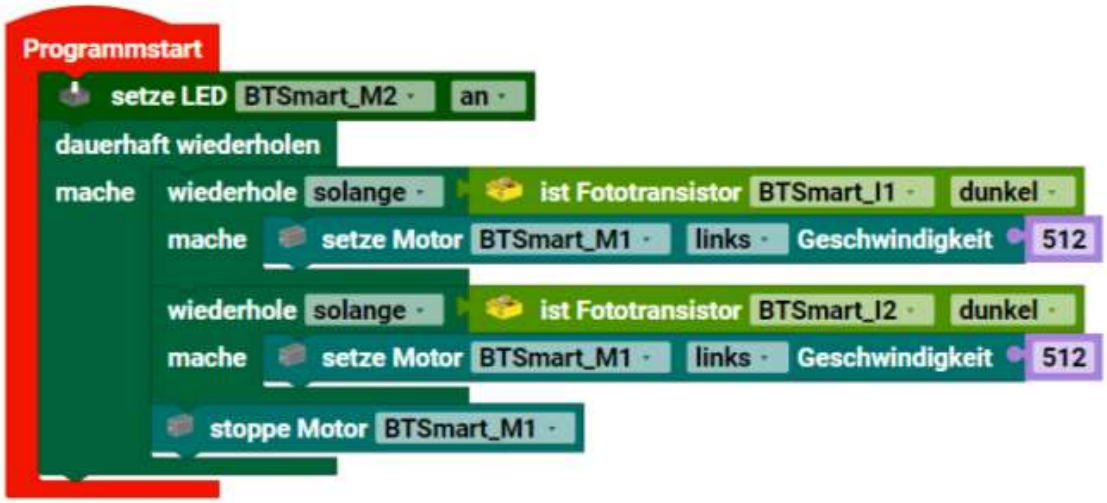
RoboPro Light Programm: Robo Pro Coding Programm



1:1 Umsetzung vom RoboPro Light Programm

Das Programm schaltet die LED an und fragt ab, ob die erste Lichtschranke unterbrochen ist oder nicht. Wenn sie unterbrochen ist, wird der Motor eingeschaltet und in einer Dauerschleife die erste Lichtschranke weiter abgefragt, ob sie immer noch unterbrochen ist. Wenn die erste Lichtschranke nicht unterbrochen ist, wird die zweite Lichtschranke abgefragt. Wenn sie unterbrochen ist, wird der Motor eingeschaltet und in einer Dauerschleife die zweite Lichtschranke so lange abgefragt, bis sie wieder frei ist und der Motor wird

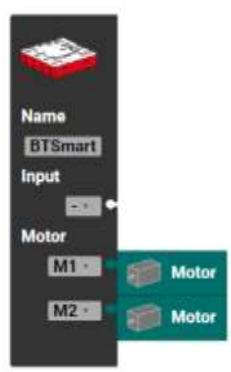
dann abgeschaltet. Sollte vorher auch die zweite Lichtschranke frei sein wird der Motor abgeschaltet.



Kürzere Alternative. Man könnte auf das Anschalten der LED verzichten, in dem man die LED an Plus vom BT Smart Controller anschließt.

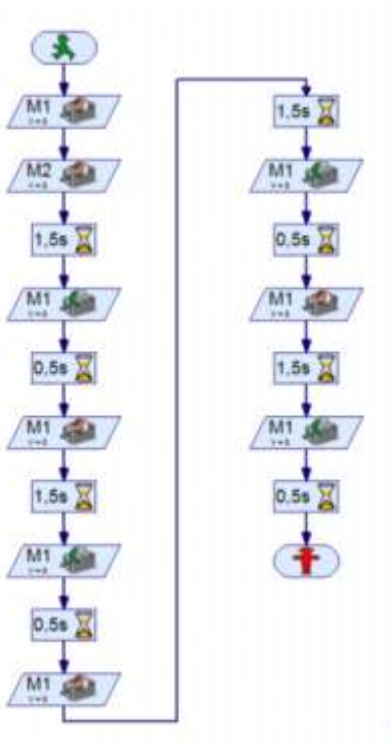
Nach dem Starten wird die LED eingeschaltet. Nun wird in einer Dauerschleife erst die erste Lichtschranke abgefragt. Solange die erste Lichtschranke unterbrochen ist, wird der Motor eingeschaltet. Nun wird die zweite Lichtschranke abgefragt. Solange die zweite Lichtschranke unterbrochen ist, wird der Motor eingeschaltet. Erst wenn auch die zweite Lichtschranke frei ist, wird der Motor abgeschaltet und ein neuer Durchlauf beginnt.

Fahrroboter 3



BT Smart Controllerkonfiguration

RoboPro Light Programm:

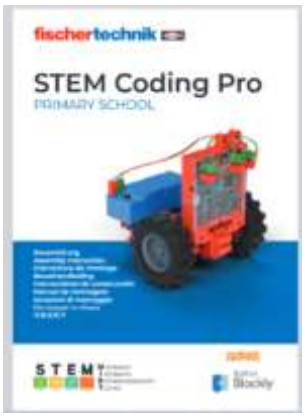


Robo Pro Coding Programm

Das Programm läuft einmal durch und endet dann. Als erstes werden beide Motoren eingeschaltet und 1,5 Sekunden laufen gelassen. Dann wird die Drehrichtung vom M1 umgekehrt und für 0,5 Sekunden laufen gelassen. Es gibt eine Abfolge der Drehrichtungsumkehr von M1 mit verschiedenen Zeiten. Danach endet das Programm.

Bemerkung. Der Motor wird nicht abgeschaltet. Erst durch das Programmende wird er abgeschaltet. Wenn man ein weiteres parallellaufendes Programm hat, bewegt sich der Roboter weiter. Man sollte also immer den Motor am Ende eines Programms abschalten.

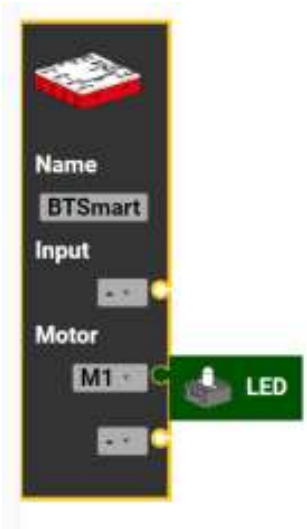
STEM Coding Pro



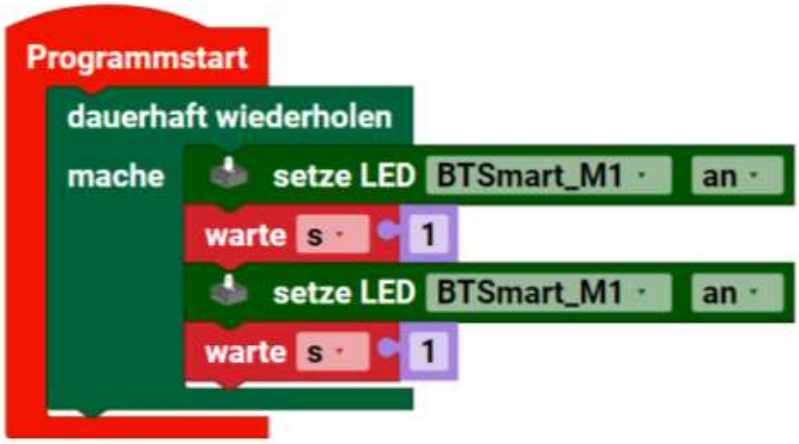
Bauanleitung lässt sich kostenlos runterladen. Es gibt didaktisches Begleitmaterial.
Die Programmiersprache ist Scratch.

Aufgabe 1: Blinklicht

fischertechnik Scratch Programm



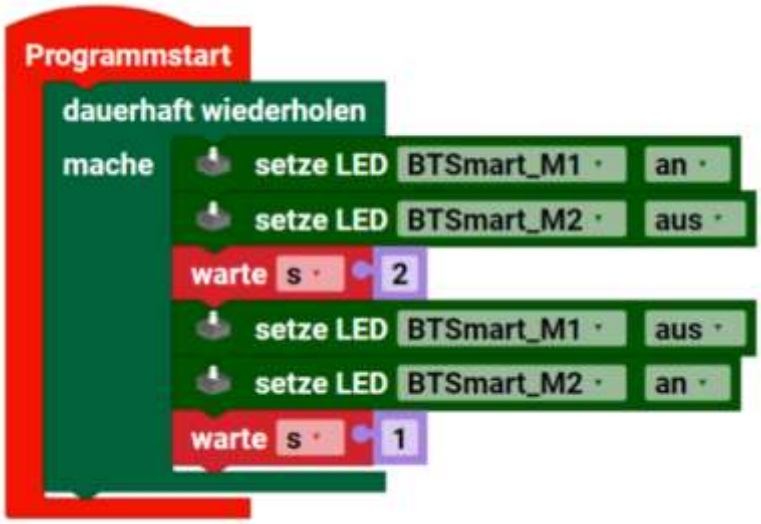
BT Smart Controllerkonfiguration



Eine Dauerschleife, wo die LED eingeschaltet wird, 1 Sekunde warten, die LED ausgeschaltet wird und wieder eine Sekunde gewartet wird. Ein neuer Durchlauf beginnt. Gestoppt wird das Programm oben links das „Quadrat“ Stopp. Neben dem „Dreieck“ Start, im blauen Feld.

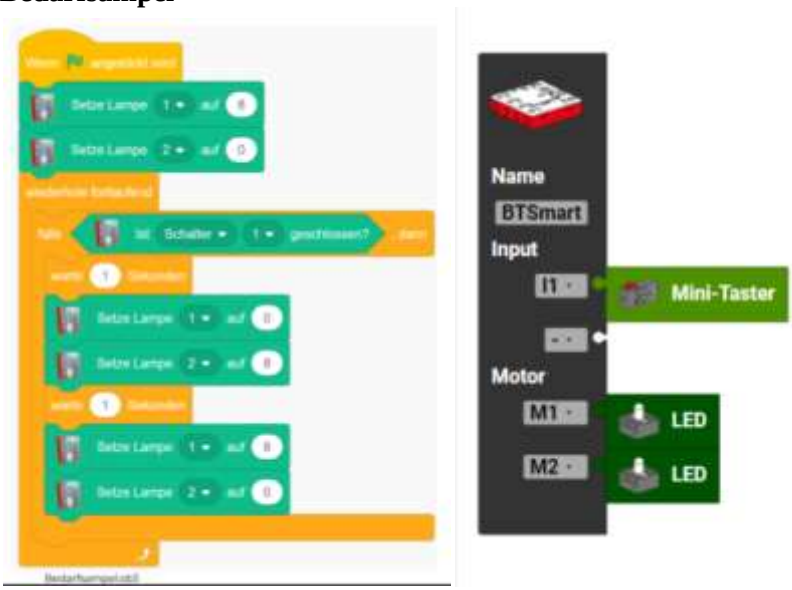
Blöcke sind aus Ausgang und Util

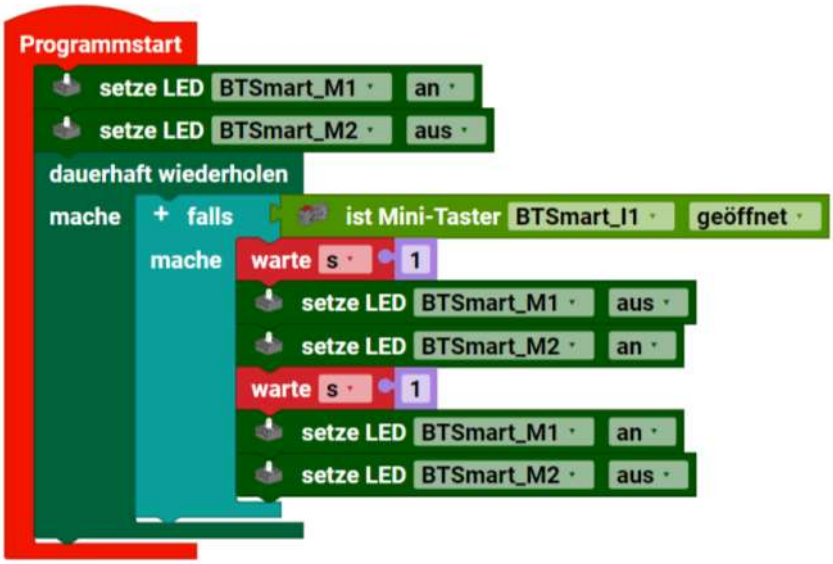
Fußgängerampel



Die Fußgängerampel hat zwei LEDs, die abwechselnd leuchten. Die Zeiten für Rot- und Grünfasen sind unterschiedlich. Blöcke sind aus Ausgang und Util

Bedarfsampel



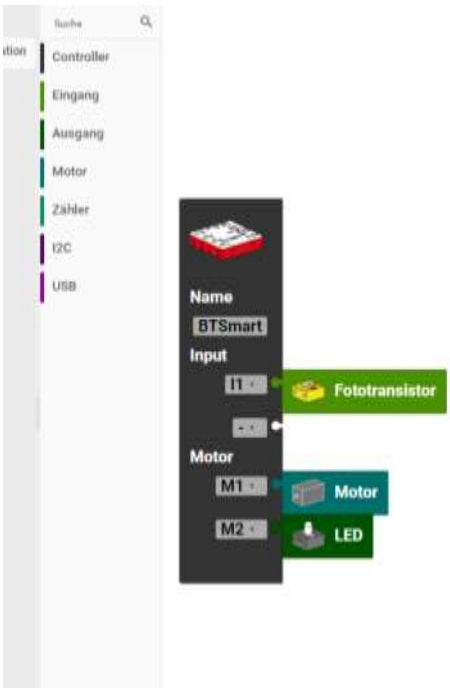


Blöcke sind aus Ausgang, Eingang, Logik und Util

Das Programm initialisiert als erstes das Leuchten der LEDs in eine bestimmte Weise. Es muss die rote LED leuchten und die grüne LED aus sein. Dann wird in einer Dauerschleife abgefragt, ob der Taster gedrückt wurde. Falls ja, wird eine Sekunde gewartet, die rote LED aus, die grüne LED angeschaltet. Nun wird wieder gewartet und die rote LED an und die grüne LED ausgeschaltet. Ein neuer Durchlauf beginnt.

Blindenampel
Die Blindenampel kann man so nicht mit dem BT Smart Controller und Robo Pro Coding machen, da kein Sound abgespielt werden kann.

Modell 3 Alarmanlage
Lichtschranke

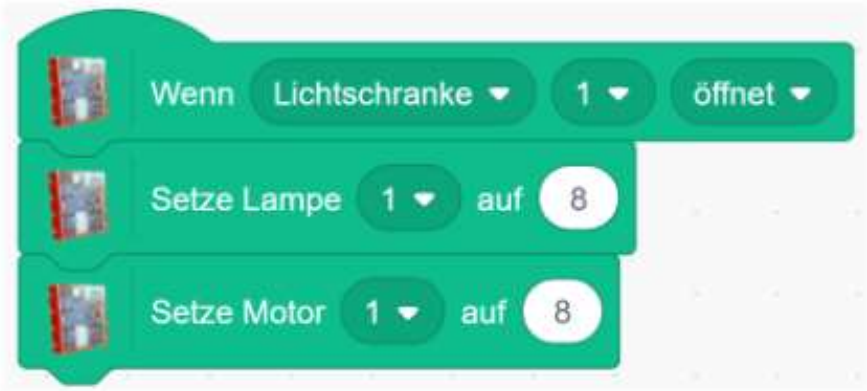


BT Smart Controllerkonfiguration fischertechnik Scratch-Programm

Hier muss man beim Fototransistor "dunkel" einstellen, da er ja von der Lampe die ganze Zeit beleuchtet wird und erst wenn man den Lichtstrahl unterbricht, das Signal ist, auf das man wartet.

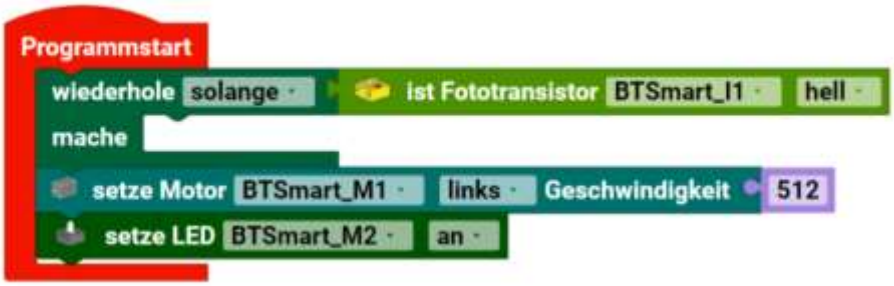
Einmal eingeschaltet, bleibt die LED bis zum Programmende an.

Alarmanlage



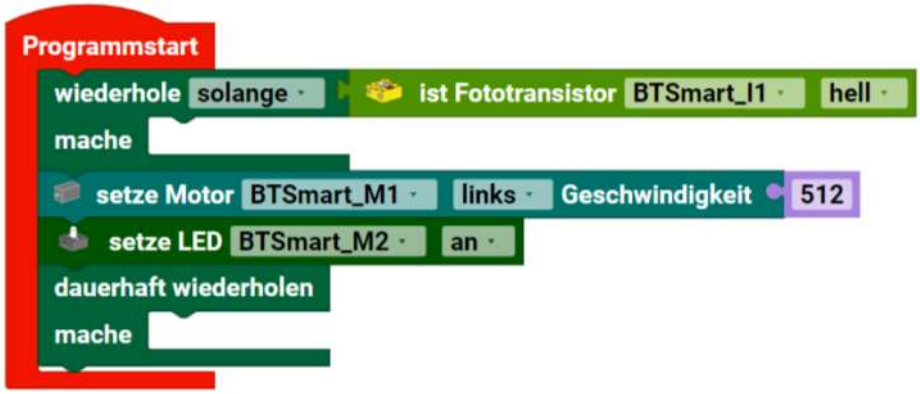
Alarmanlage.sb3

fischertechnik Scratch Programm



Robo Pro Coding Programm

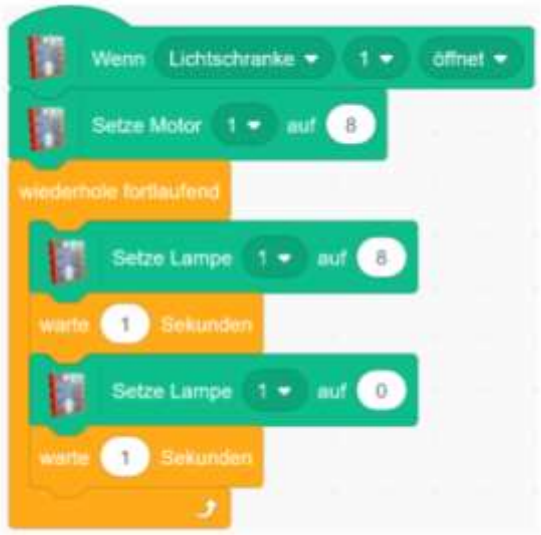
Das Programm wartet auf die Lichtschranke. Wenn sie unterbrochen wird, läuft das Programm weiter und schaltet den Motor und die Lampe ein. Aber nur kurz, da durch das Programmende alle Ausgänge abgeschaltet werden.



Wenn man den Block dauerhaft wiederholen daruntersetzt, bleibt der Motor und die LED an. Man beendet Das Programm durch Drücken von Stopp oben rechts.

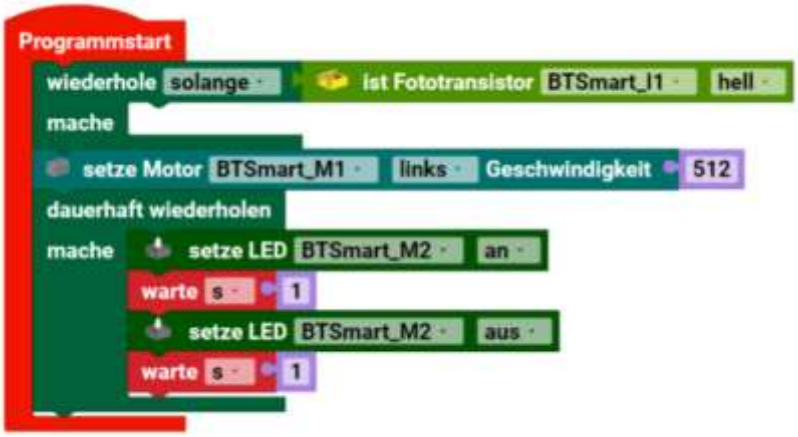
Hier wird zusätzlich der Motor über "falls" eingeschaltet. Auch dieser bleibt bis zum Programmende eingeschaltet. Es ist ja auch kein Stopp für den Motor im Programm.

Alarmanlage mit Blinklicht



Alarmanlage mit Blinklicht.sb3

fischertechnik Scratch Programm



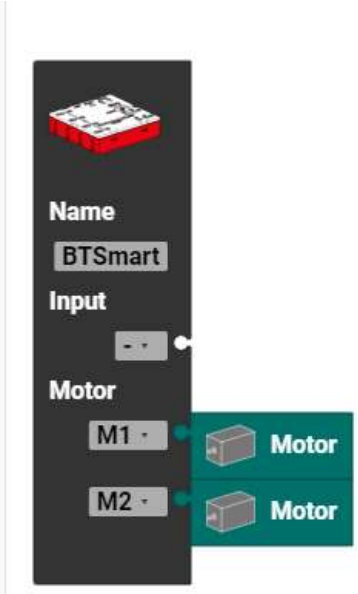
Statt das Programm, einfach mit einem leeren Block, in einer Dauerschleife weiter laufen zu lassen, kann man auch ein Blinken der Lampe machen.

Kein Sound auf den BT Smart Controller
Alarmanlage mit Sirene
Die Soundausgabe mit der Sirene läuft auf dem BT Smart Controller nicht, da er keinen Lautsprecher hat.
Info: Die Soundausgabe passiert auf dem Controller, nicht auf dem PC.

Fernsteuerung 1

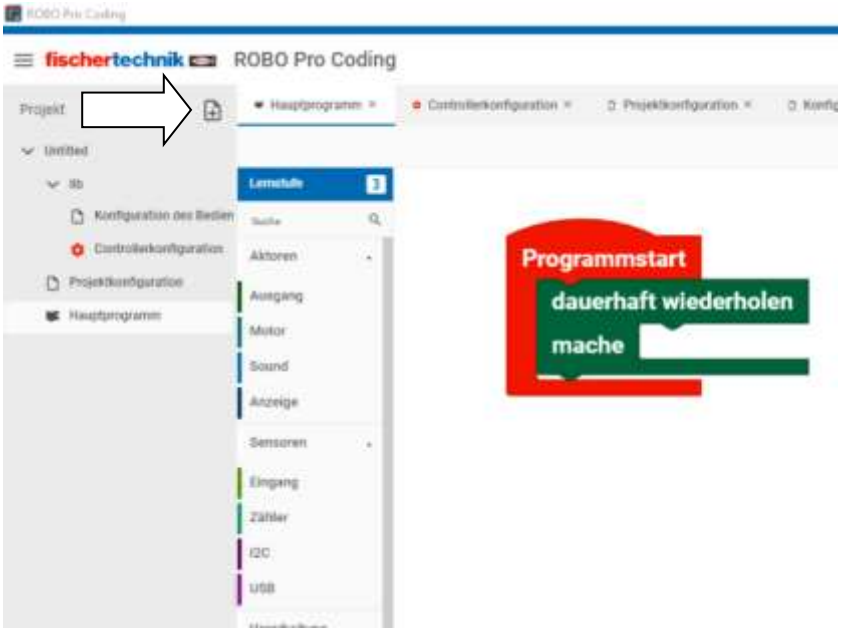
Die Fernsteuerung geht über die Pfeiltasten Tasten der PC Tastatur.

Fischertechnik Scratch Programm

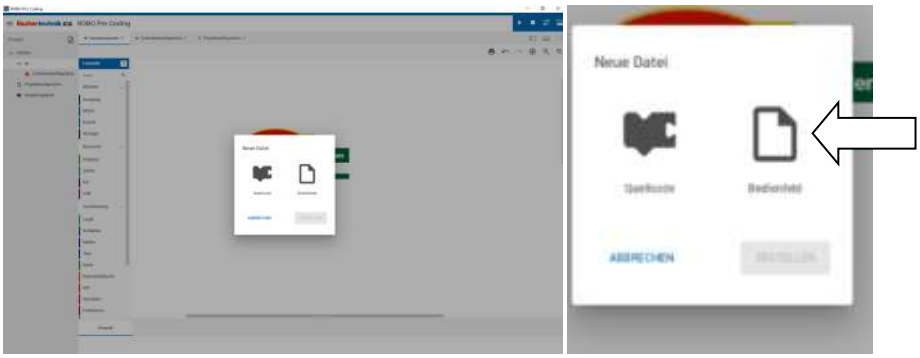


Bedienfeld

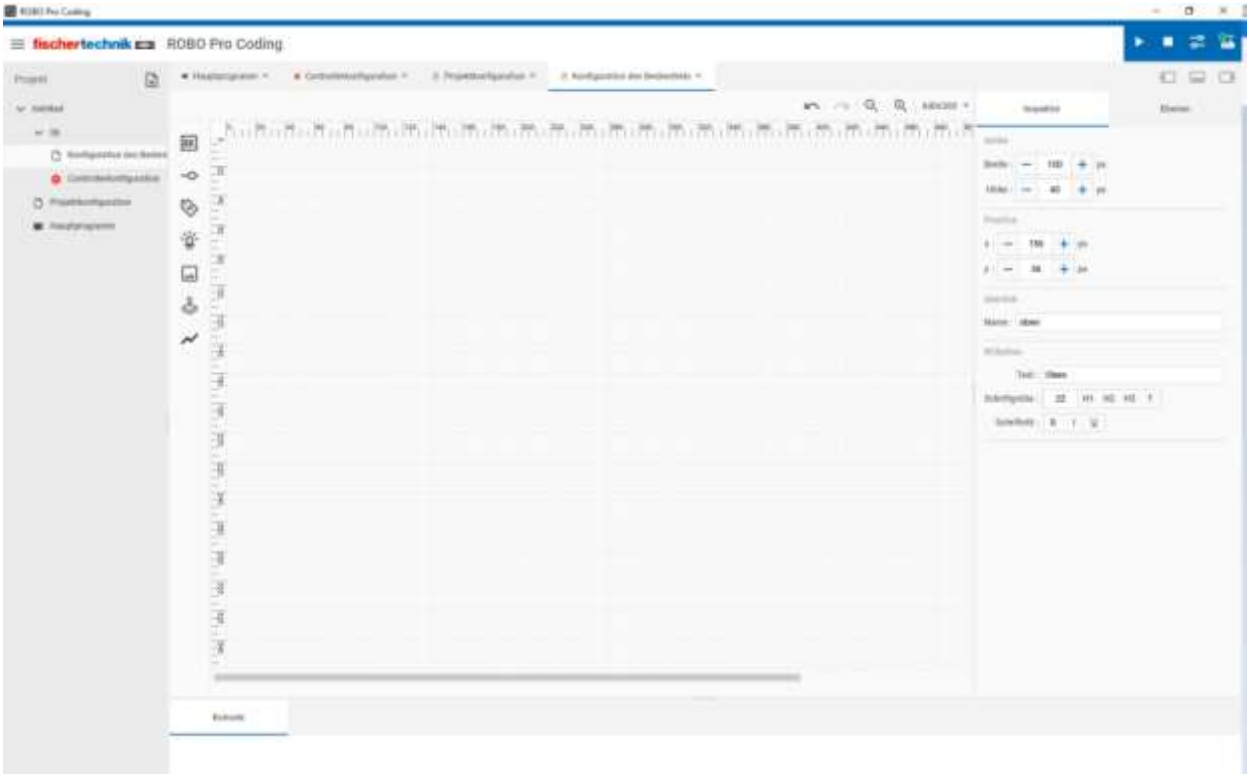
Ein Bedienfeld in Robo Pro Coding ist eine Art von Fernbedienung oder Steuerpult.



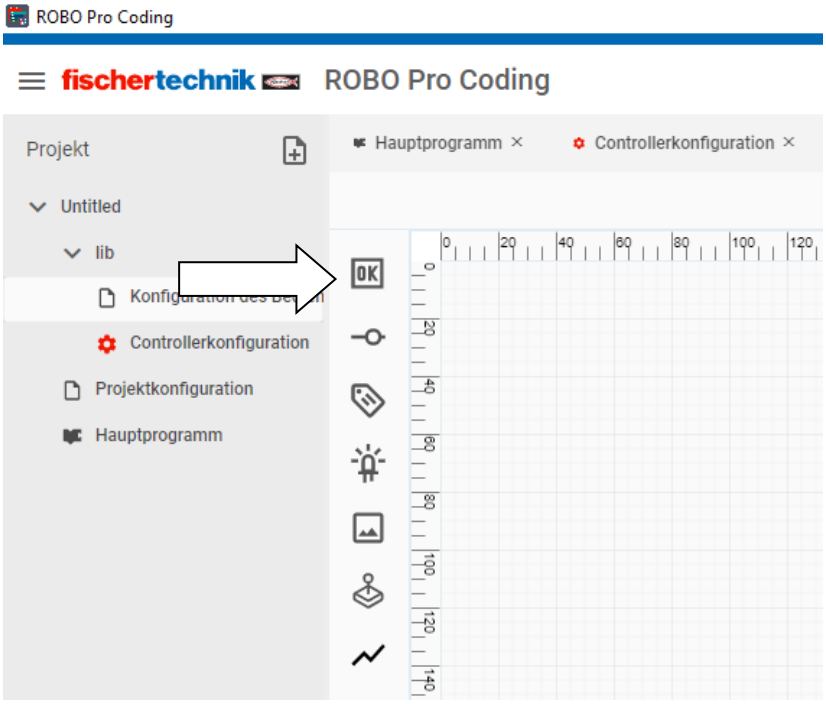
Um ein Bedienfeld anzulegen, muss man oben links auf das Plus klicken...



und Bedienfeld auswählen und auf Erstellen klicken.

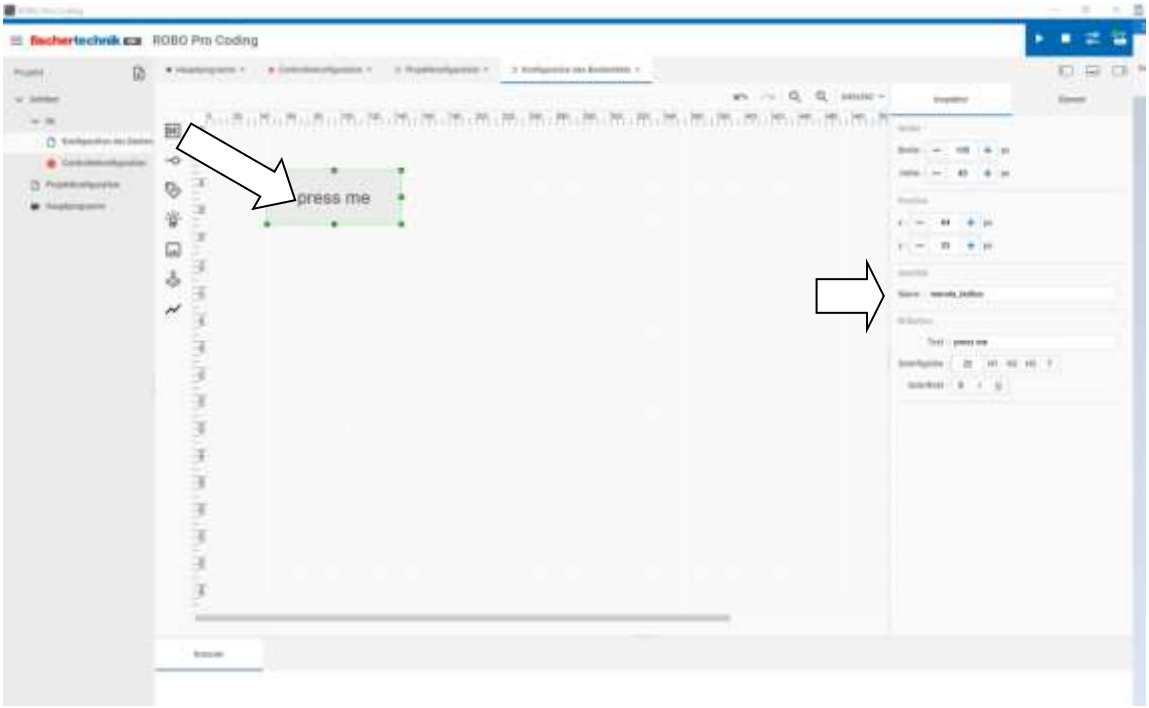


Es erscheint ein leeres Bedienfeld, in das man Bedienelemente oder Text „aufmalt“.

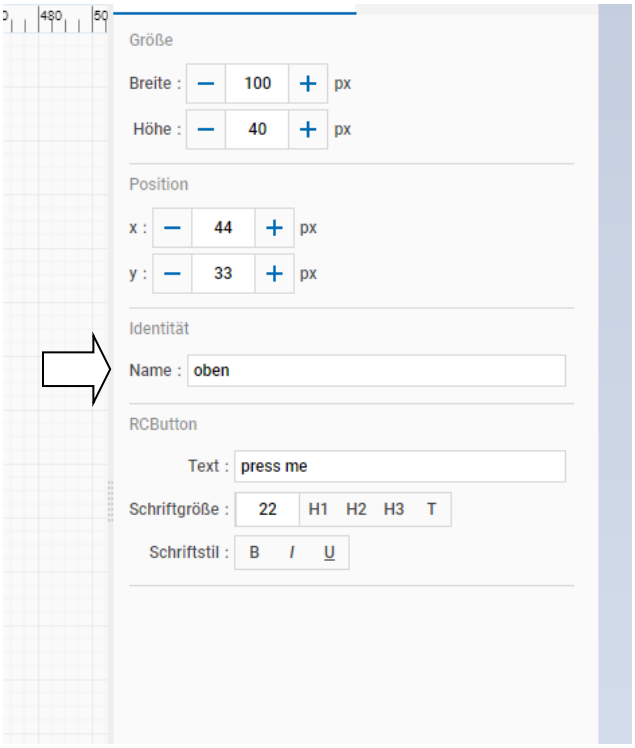


Neben der Skala oben rechts auf "OK" klicken und ziehen ihn in das Feld. Das "OK" ist ein Remote Controll Button, ein virtueller Taster. Ob dieser Taster gedrückt ist oder nicht kann man in dem Programm abfragen.

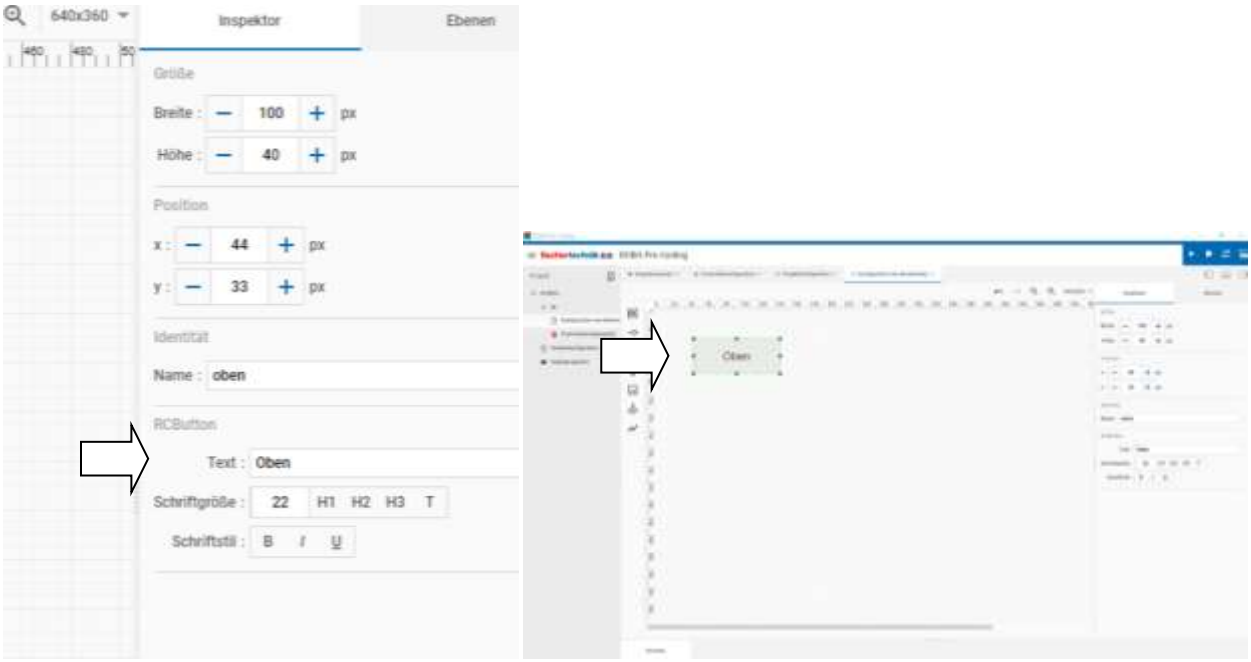
Wie in dem Scratch Programm erstellen wir so einen RCButton "Oben" für die Fernbedienung.



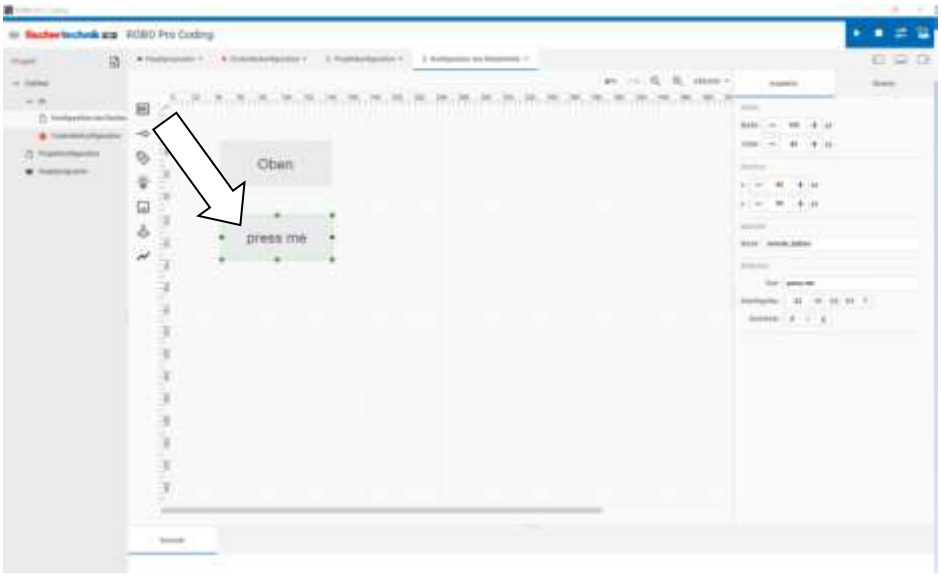
Wir ändern den Namen remote_button



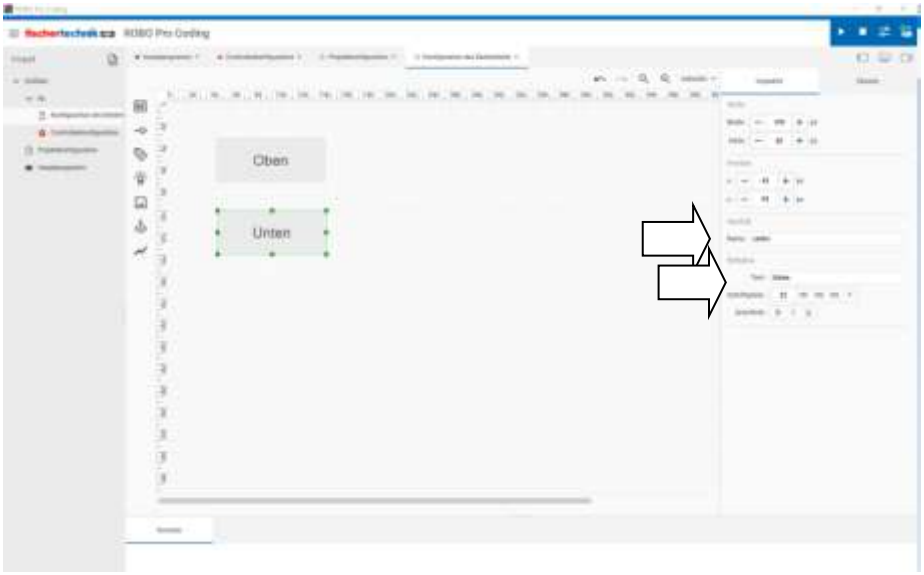
in „oben“. Man kann nur kleine Buchstaben verwenden. Große werden in kleine automatisch umgewandelt.



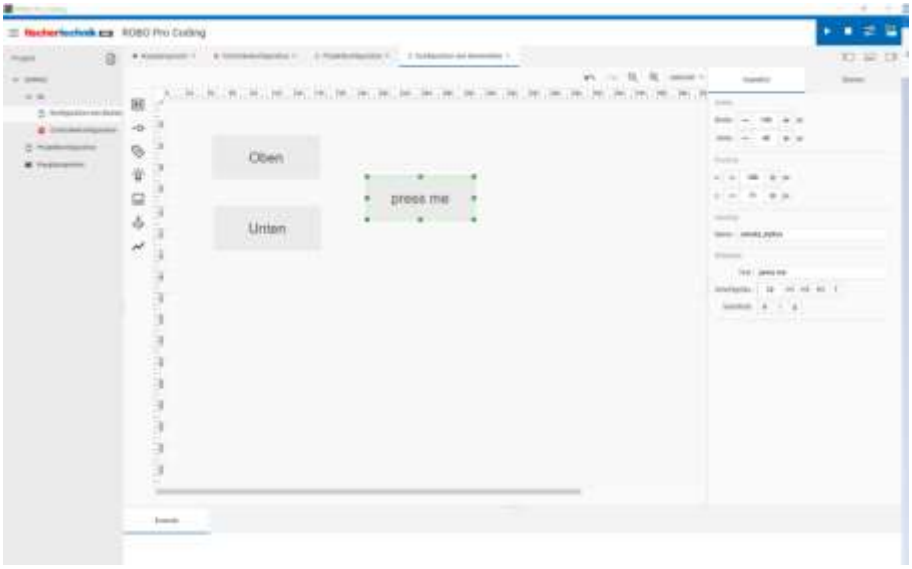
Wenn man den Text "press me" in "Oben" (mit großem Anfangsbuchstaben) eingibt, ändert sich die Beschriftung vom Taster.



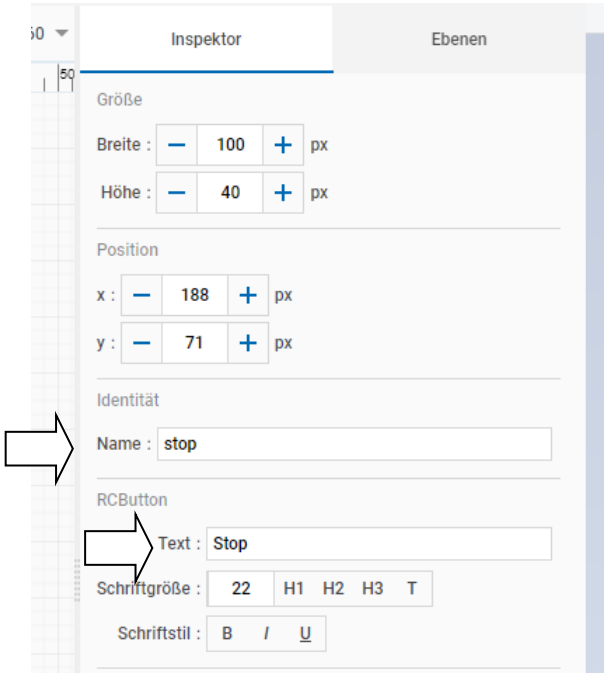
Den zweiten Taster ziehen wir unter den ersten Taster/Button.



Und ändern den Namen in "unter" und den Text in "Unten".

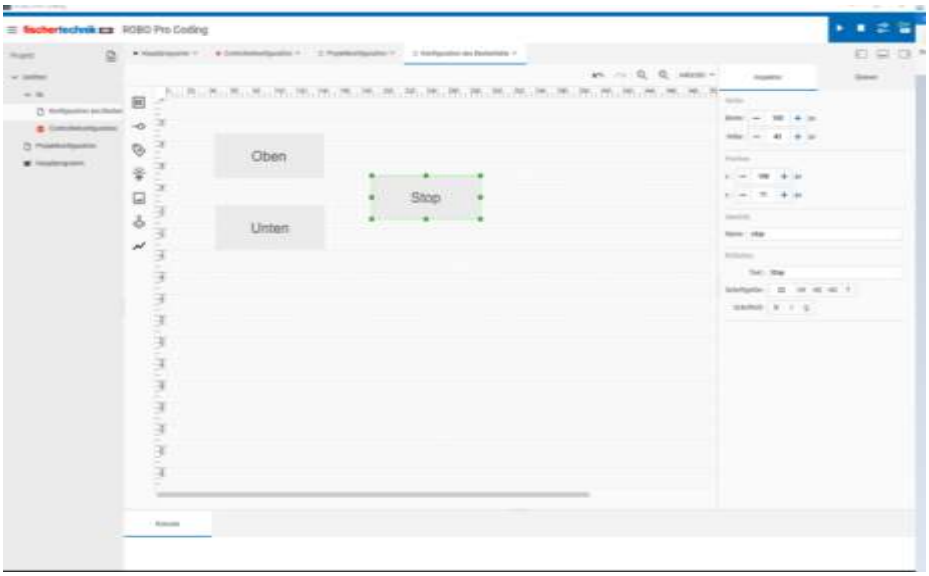


Den dritten Taster/Button setzen wir mittig, rechts daneben.



und ändern den Namen in "stop" und den Text in "Stop".

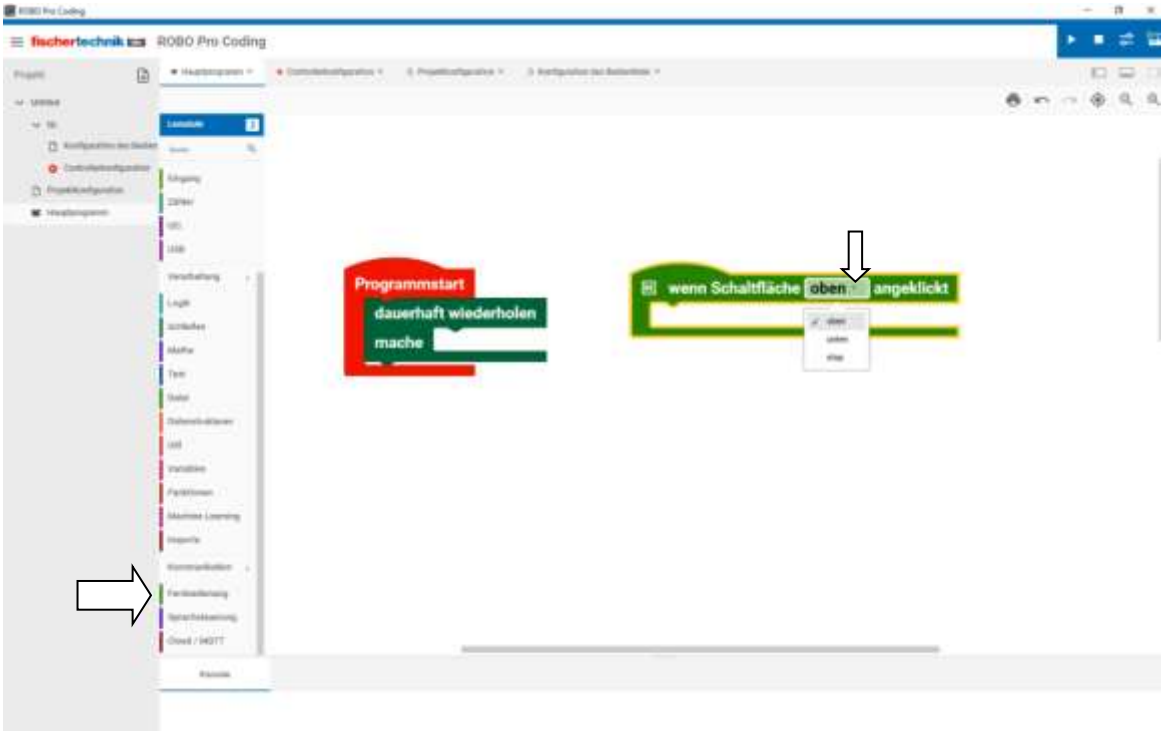
Wenn man auf einen Button drückt, erscheinen rechts die Daten des Buttons. Wenn man möchte, kann auch die Position, Schriftgröße und die Größe selbst ändern.



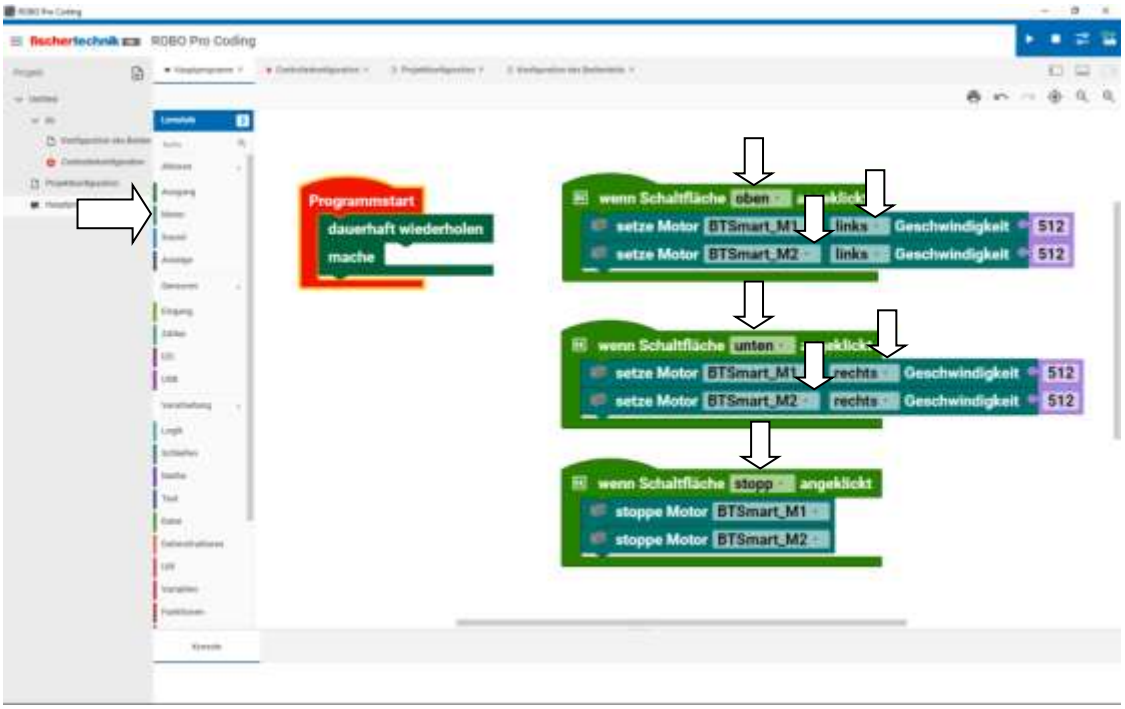
So sollte es nun aussehen.

Wir haben nun ein Bedienfeld mit drei Buttons/virtuellen Tastern

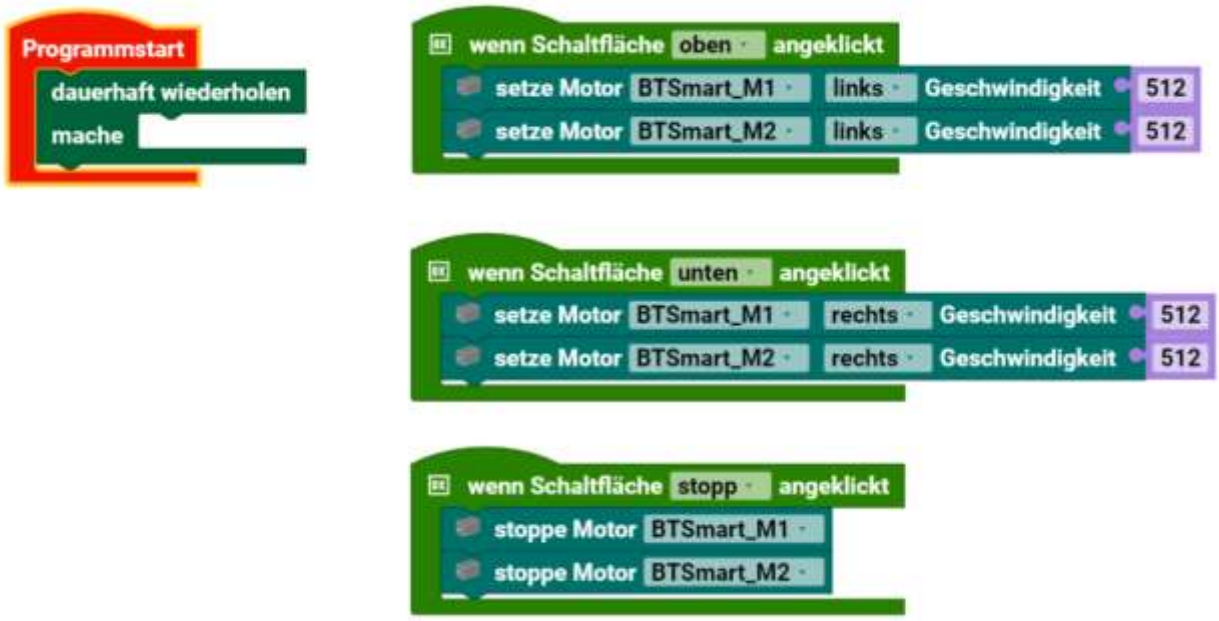
Hauptprogramm



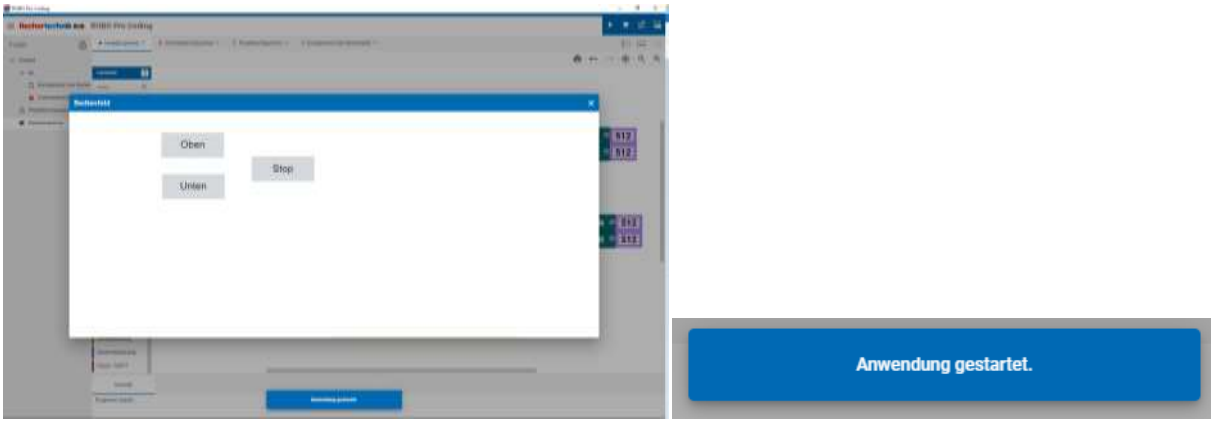
Im Hauptprogramm erzeugen wir drei Unterprogramme oben, unten und stopp. Dazu im Reiter "Fernbedienung" den Block "wenn Schaltfläche... angeklickt" 3 mal rüber ziehen. Wir benennen die jeweils in oben, unten und stopp um, indem man auf den kleinen Pfeil klickt.



Vom Reiter Motor ziehen wir "setze Motor...links Geschwindigkeit 512" rüber und duplizieren ihn 3 mal und ändern jeweils "links" und "rechts". In den dritten kommt der Block "stopp".



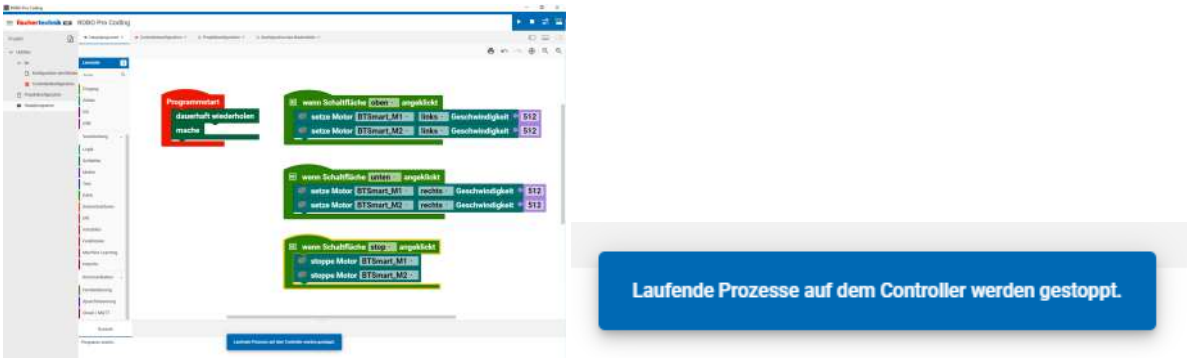
So sollte es nun aussehen.



Wenn wir das Programm starten, kommt das Bedienfeld. Kurz wird auch eine Meldung angezeigt.

Es startet das Hauptprogramm -und- die drei Unterprogramme.

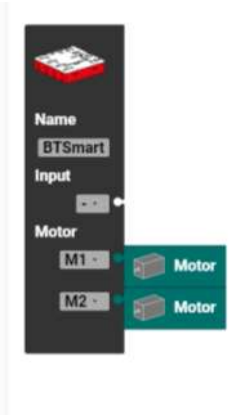
Durch das Drücken mit der Maus auf den jeweiligen Button, wird das jeweilige Unterprogramm ausgeführt. Es werden also die Motoren M1 und M2 mit entsprechender Richtung eingeschaltet oder beide ausgeschaltet.



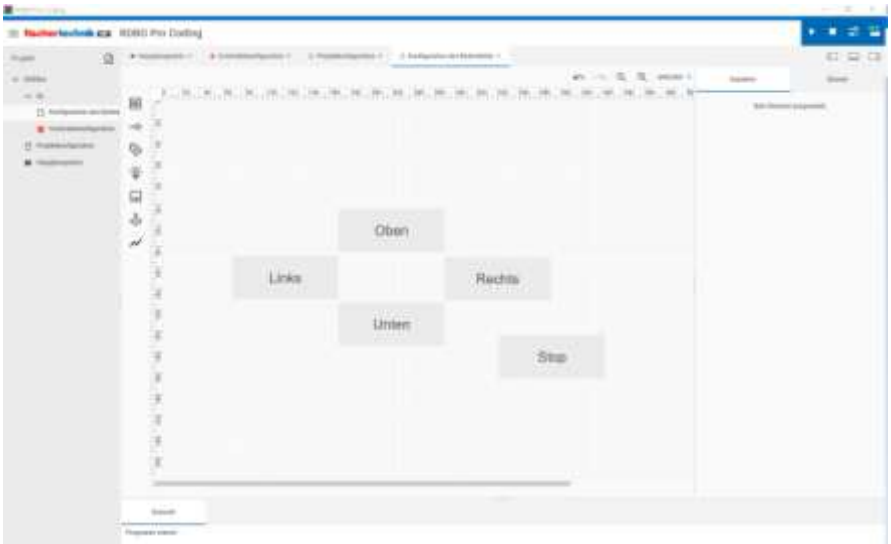
Wenn man das Bedienfeld über das X schließt oder oben auf Stopp klickt, wird das Programm beendet und kurz die Meldung angezeigt.

Fernsteuerung 2

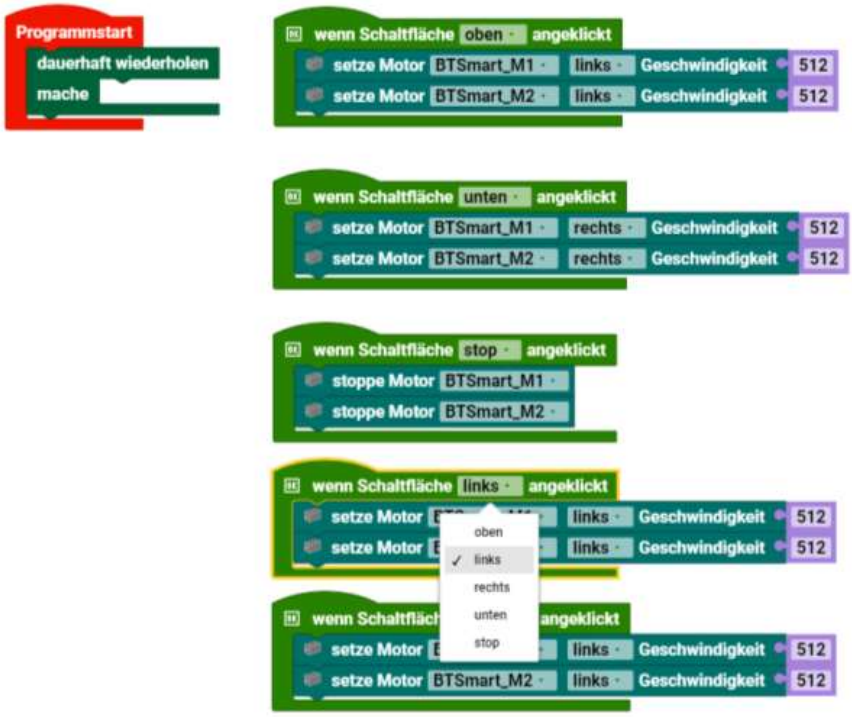
fischertechnik Scratch Programm



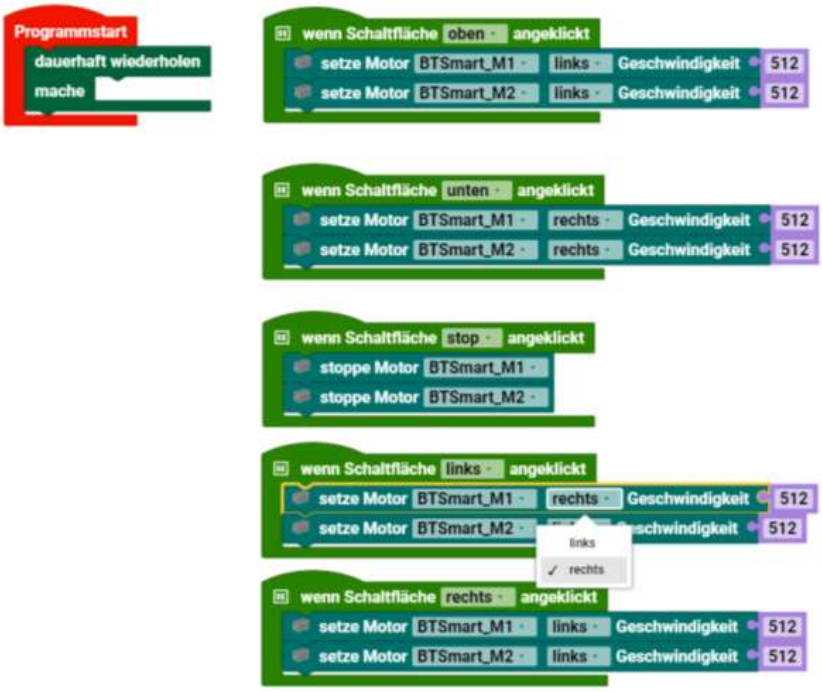
BT Smart Controllerkonfiguration



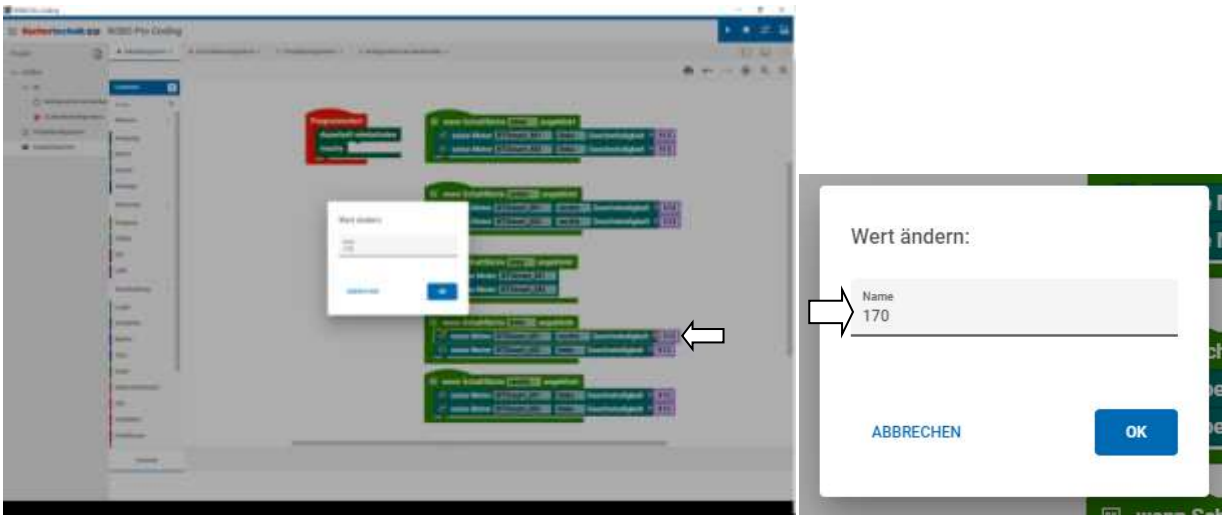
Hier kommen zum Programm Fernsteuerung 1 noch zwei weitere Buttons dazu. Dazu verschieben wir die Buttons. Wenn man hintereinander die Buttons neu rüber zieht, werden die Namen einfach mit fortlaufenden Zahlen markiert remote_button1, remote_button2... damit das Programm sie unterscheiden kenn. Diesen Namen und Texte einfach überschreiben. Es sollte nun so aussehen.



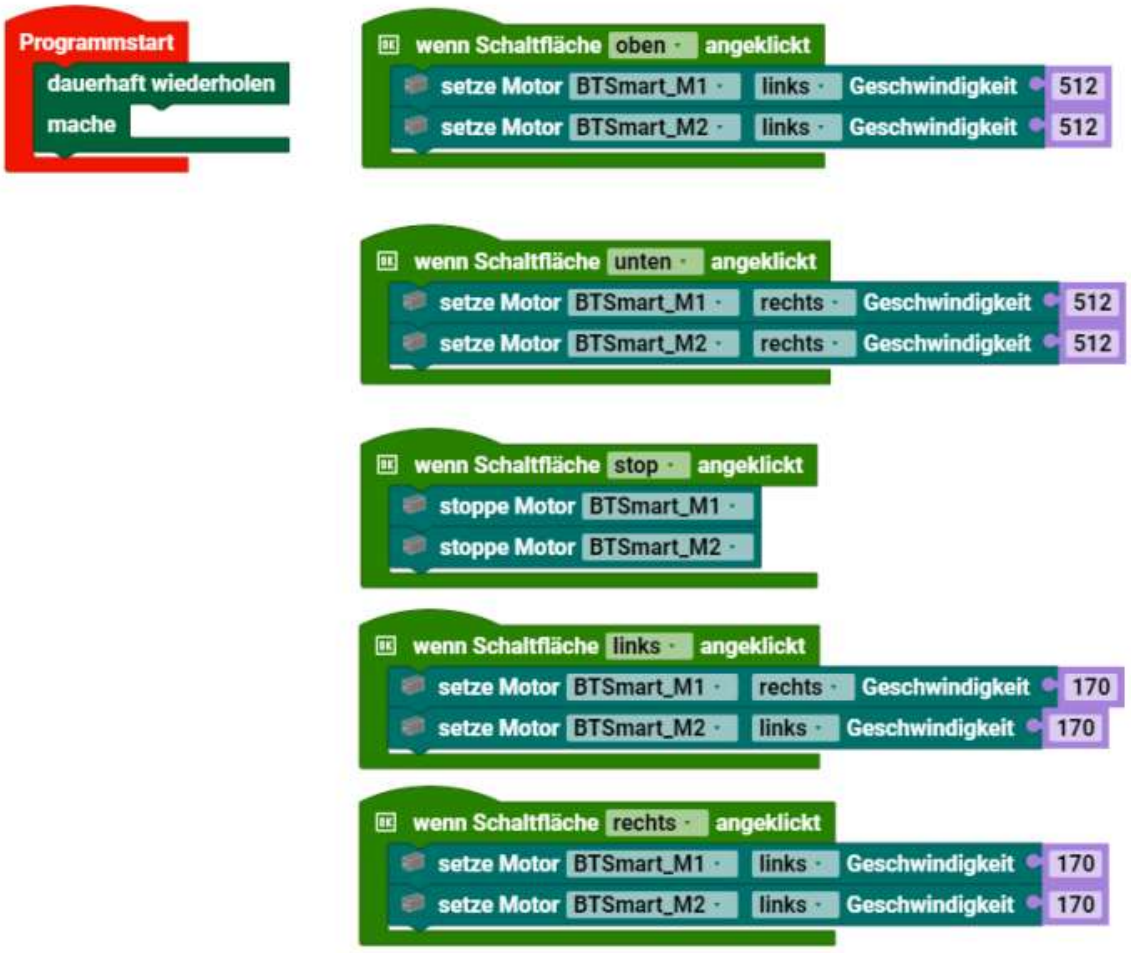
Wenn man den obersten Block zwei Mal dupliziert, kann man die nach unten ziehen und in "rechts" und "links" umbenennen.



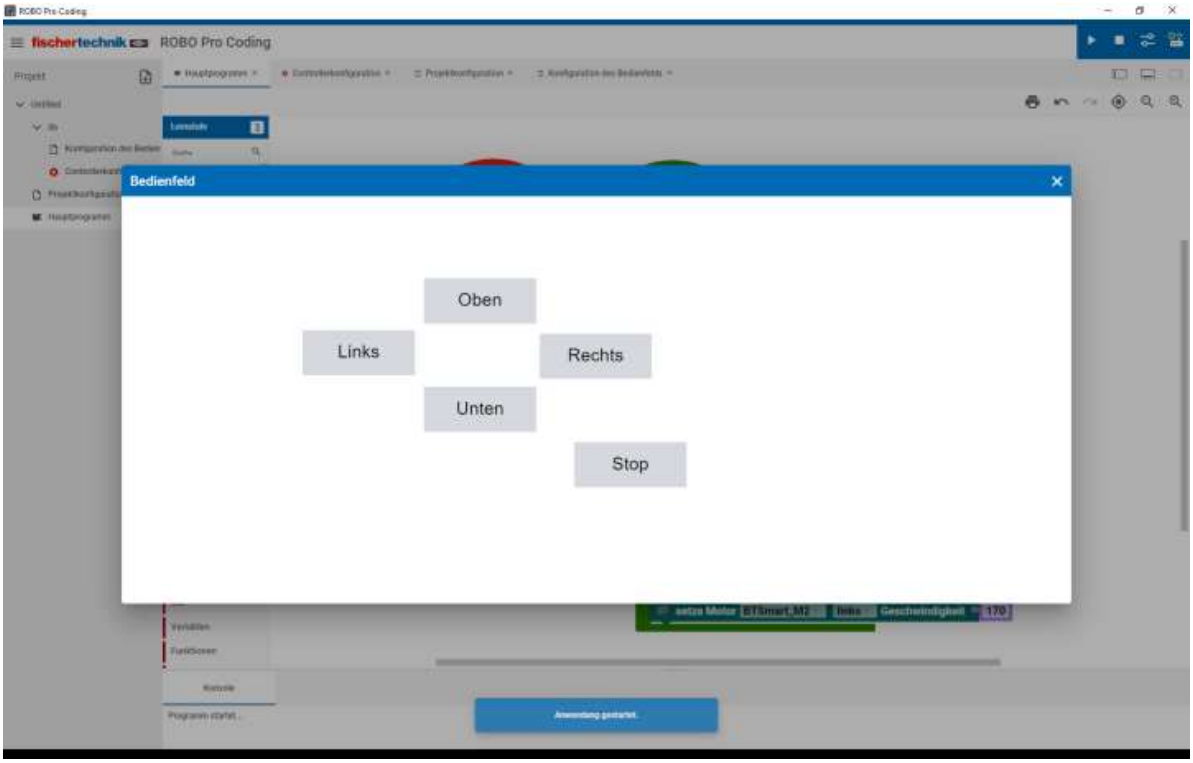
Nun die Drehrichtung ändern und…



die Geschwindigkeit auf 170 ändern.



So sollte es nun aussehen.

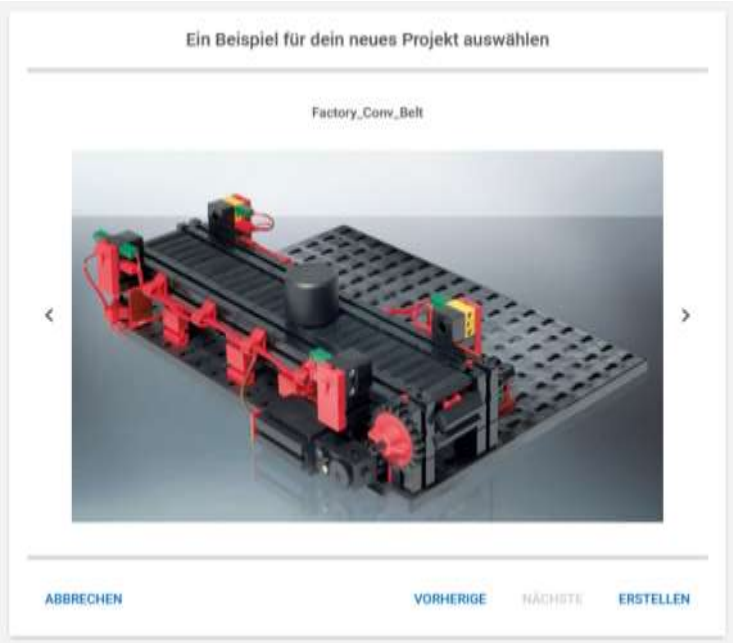


Beim Starten des Programms wir das Bedienfeld angezeigt.

Bemerkung: Die Bezeichnung Oben / Unten kommt vom Scratch Programm, weil dort die Pfeil-Tasten auf der Tastatur genommen wurden. Man kann auch Vorwärts und Rückwärts nehmen.

fischertechnik Industriemodell Factory_Conv_Belt 9V (Förderband)

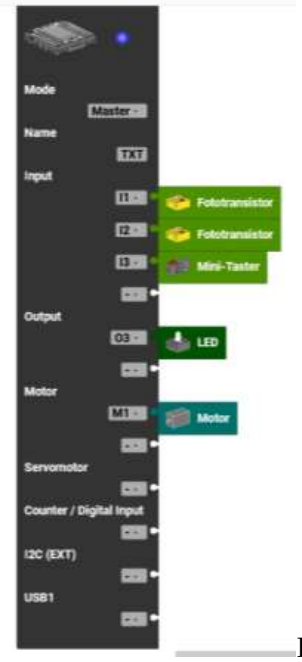
Transportband mit zwei Lichtschranken
Fertiges Beispiel unter: Projekt/Neu/ft-Beispiel/Training_models/Factory_Conv_Belt



Industriemodell (Fertigmodell) vom Fließband, hier in der 9V Bauweise. Bild ohne Controller.

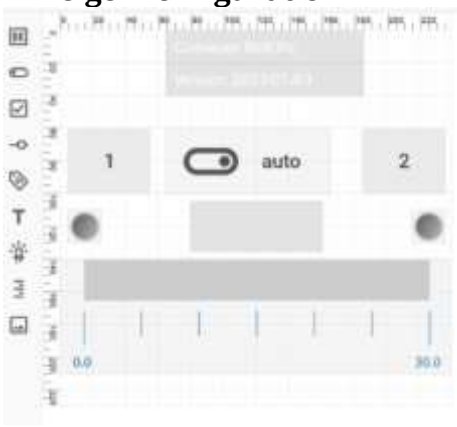
Der Minitaster sitzt auf der Rückseite und zählt die Impulse, wenn sich das Band bewegt. Zwei Fototransistoren (gelb) und gegenüberliegend LEDs die dauernd leuchten können.
Man kann das Modell auch mit anderen Controllern betreiben. Man sollte statt des Displays des TXT 4.0 mit der Anzeigenkonfiguration, dann aber das Bedienfeld auf dem PC nehmen.

Controllerconfiguration



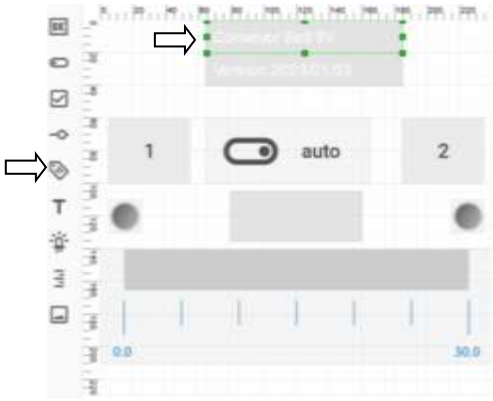
Die Controllerconfiguration des Programms.

Anzeigenkonfiguration

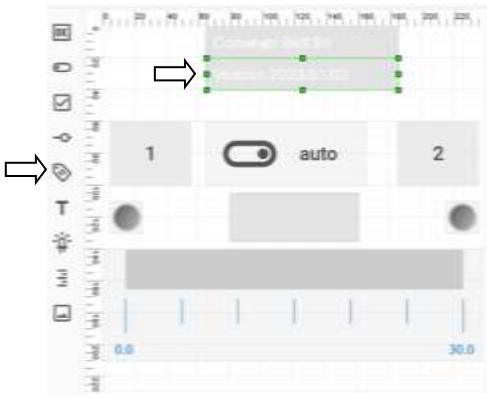


Gesamtansicht des Displays vom TXT 4.0

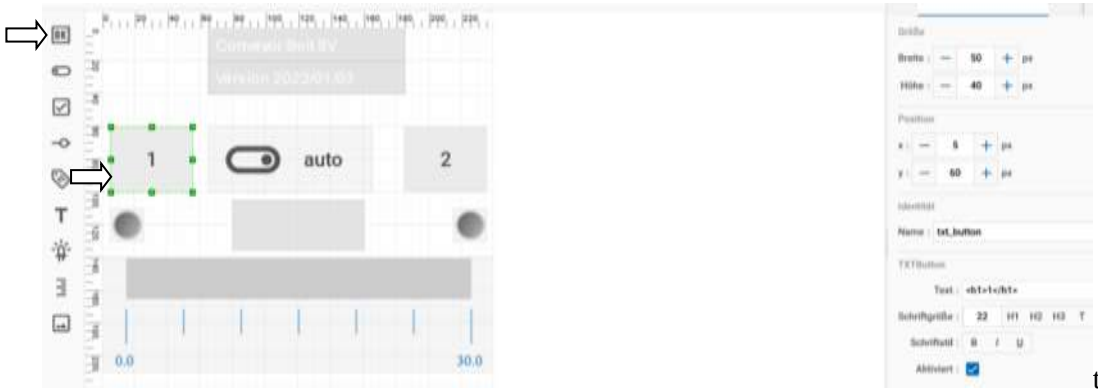
Anzeigenkonfiguration vom Display des TXT 4.0



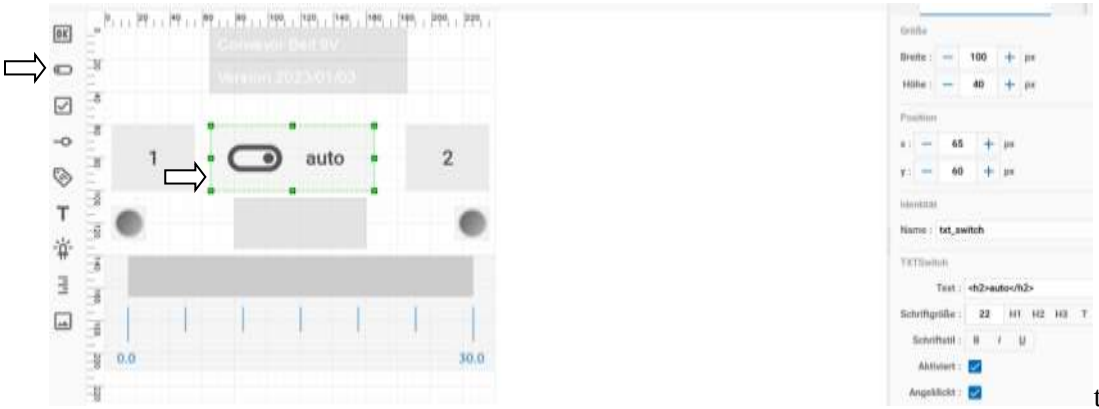
txt_label_name



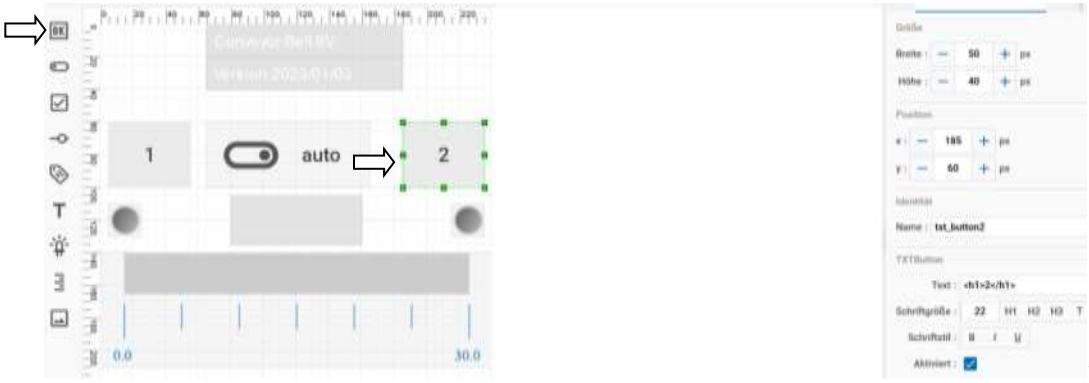
txt_label_version



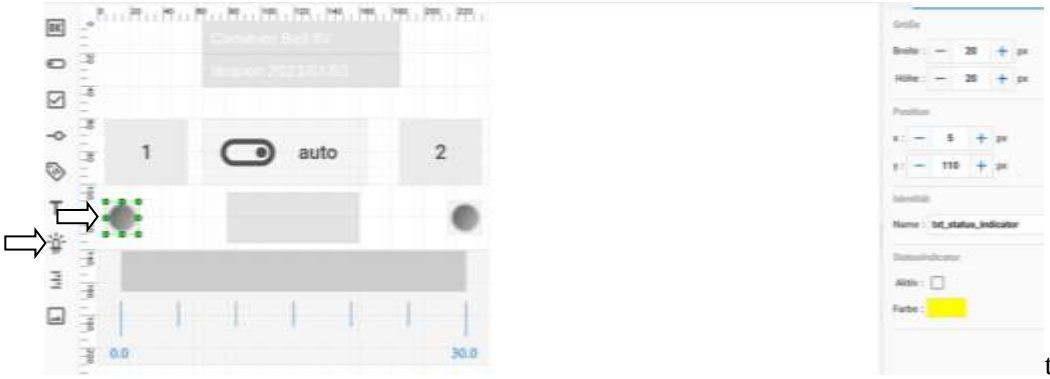
txt_button



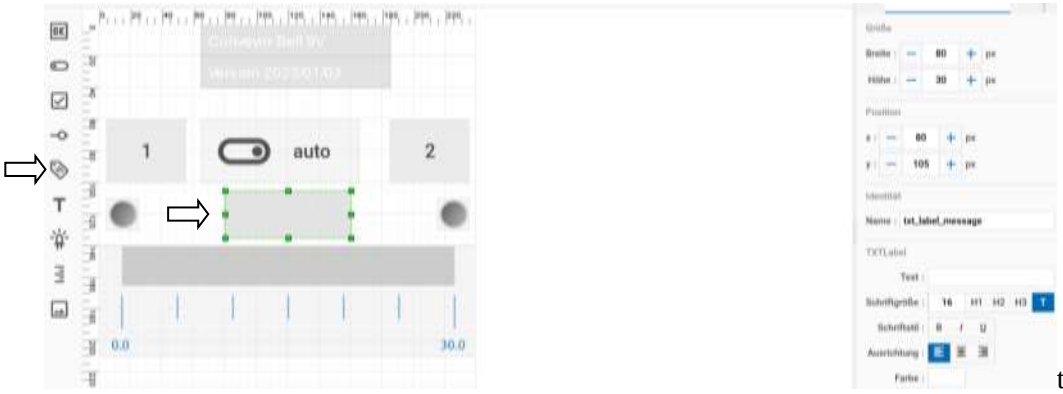
txt_switch



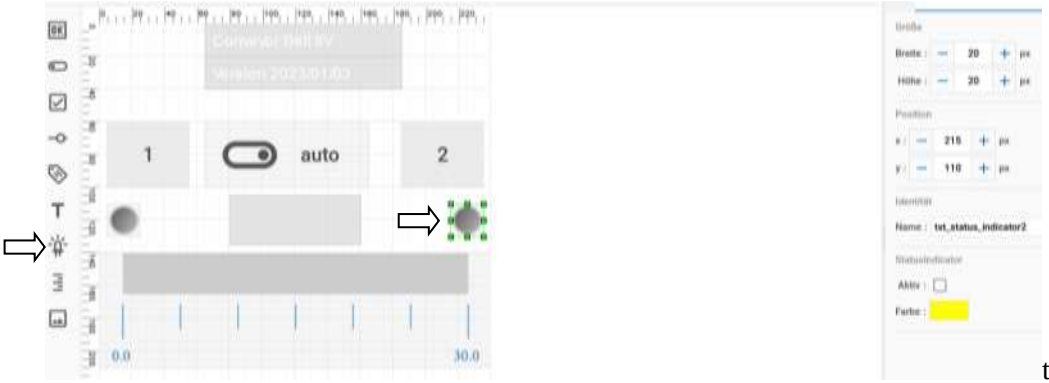
txt_button2



txt_status_indicator



txt_label_message

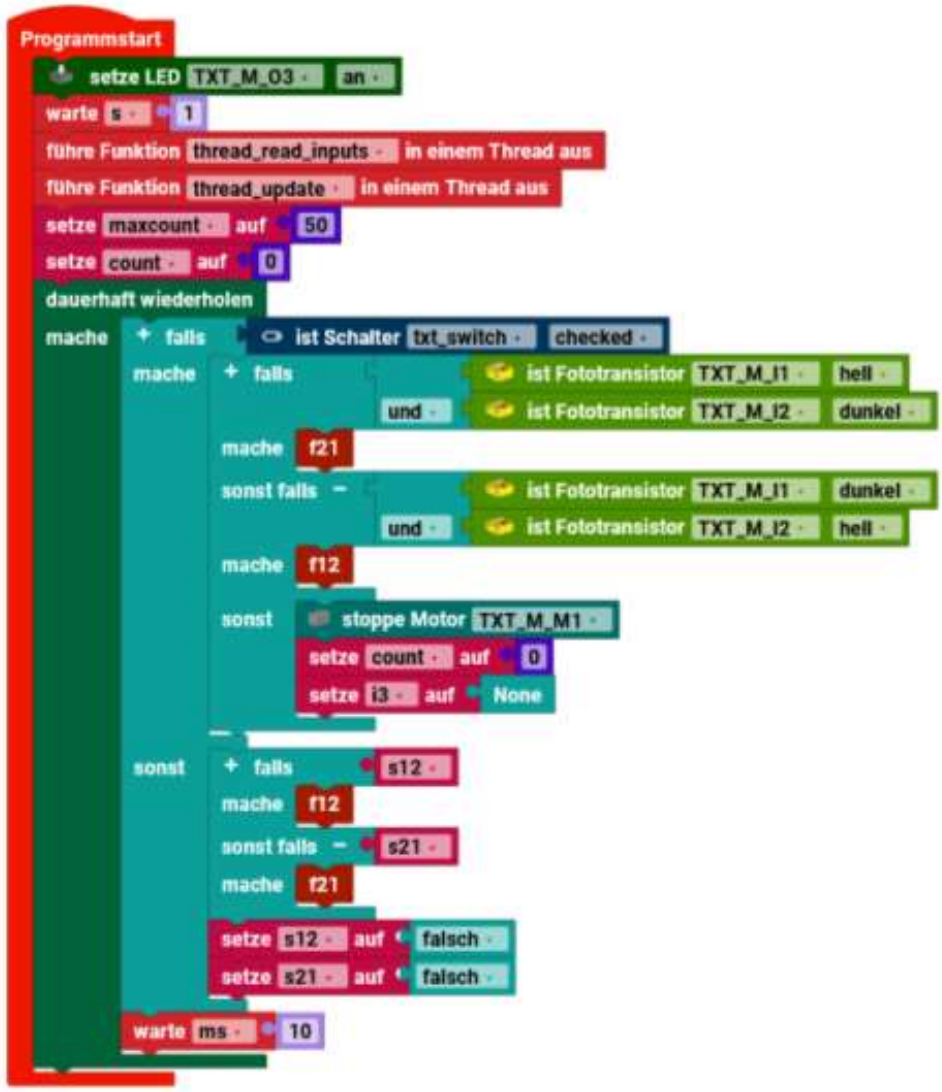


txt_status_indicator2

txt_gauge

Robo Pro Coding Programm

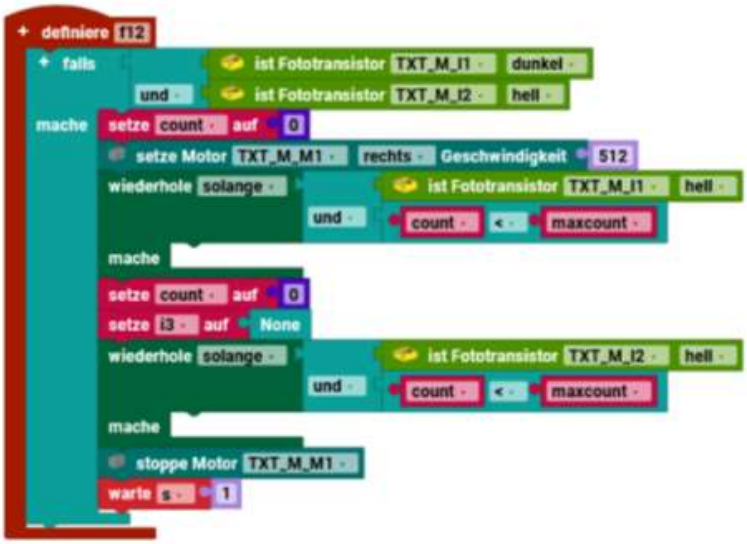
Hauptprogramm



Als erstes werden die LEDs eingeschaltet und 1 Sekunde gewartet. Dann werden die Funktionen „thread_read_inputs“ und „thread_update“ jeweils als eigenständiges Unterprogramm laufen gelassen. Die Variablen „maxcount“ und „count“ werden auf ihre Werte gesetzt.
In einer Endlosschleife wird nun der Schalter am Display (Mitte) des TXT 4.0 abgefragt. Falls er geschaltet ist und die eine Lichtschranke hell und die andere dunkel ist, wird die Funktion „f21“ ausgeführt. Wenn die erste Lichtschranke dunkel und die zweite hell ist, wird die Funktion „f12“ ausgeführt. Wenn nichts davon zutrifft, wird der Motor gestoppt, die Variable „count“ auf null und die Variable i3 auf „leer“.
Wenn der Schalter am TXT Display nicht geschaltet ist, wird abgefragt ob die Variable „s12“ wahr ist und dann wird die Funktion „f12“ ausgeführt. Wenn die Variable „s21“ wahr ist, wird die Funktion „f21“ ausgeführt. Danach werden beide Variablen „s12“ und „s21“ auf falsch gesetzt und nach 10 ms die Endlosschleife wiederholt.

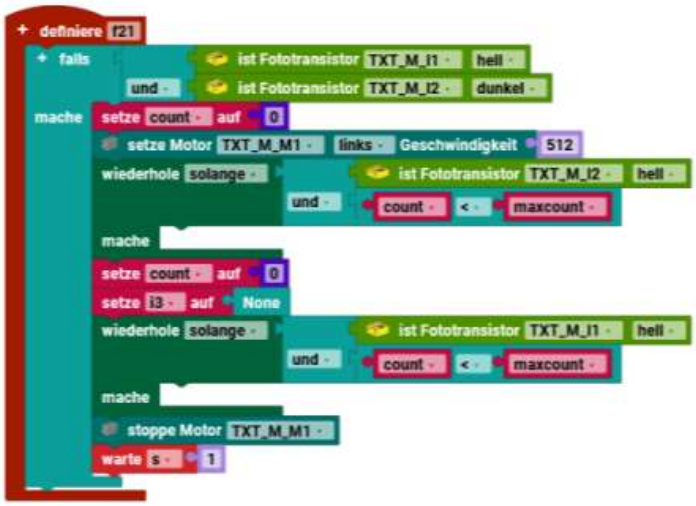
Zusammenfassung: Erst wird das Programm initialisiert, dann je nach Schalterstellung „Automatisch“ hin und her gefahren oder „Handsteuerung“ per Taster auf dem Display hin oder hergefahren. Gleichzeitig erfolgt eine Überwachung, ob das Werkstück angekommen ist oder nicht. Wenn beide Lichtschranken hell oder beide dunkel sein sollten, wird der Motor gestoppt. Die Flankenerkennung in „i3“ wird auf leer gesetzt.

Funktion „f12“

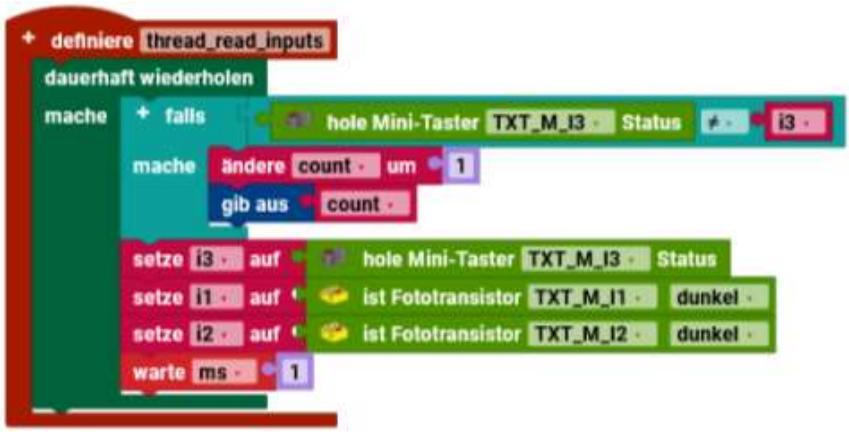


Hier wird abgefragt ob die Lichtschranken dunkel und hell sind. Wenn ja, wird die Variable „count“ auf null gesetzt und der Motor wird rechtsrum laufen gelassen. In einer Schleife wird abgefragt, ob der Fototransistor hell ist und ob die Bedingung count<maxcount erfüllt ist. Wenn eine der Bedingungen nicht erfüllt ist, wird „count“ auf 0 und i3 auf none gesetzt. Jetzt wird gewartet ob die Bedingungen Lichtschranke hell und count<maxcount ist und wenn eine davon nicht mehr wahr ist, wird der Motor gestoppt.

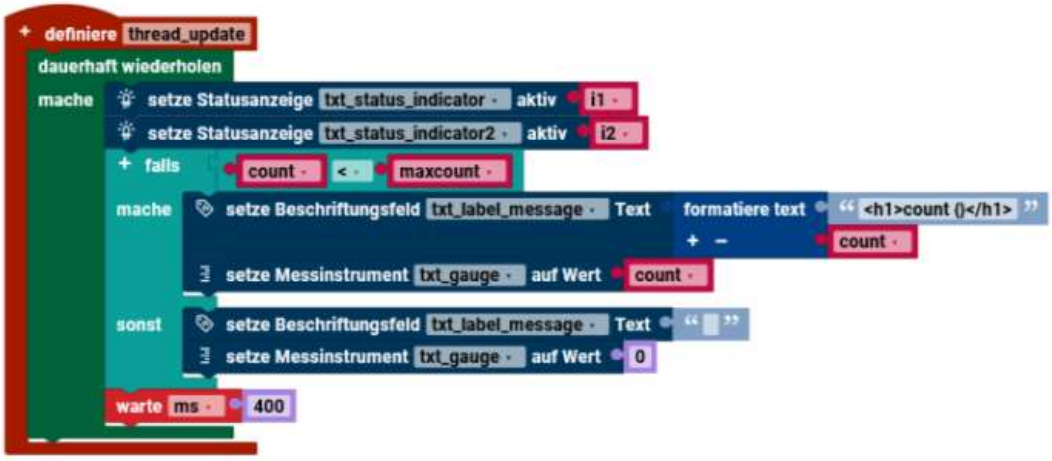
Zusammenfassung: Das Werkstück wird aus der Lichtschranke herausgefahren und ab da beginnt der Zähler zu zählen. Nun läuft das Werkstück in die zweite Lichtschranke oder es kommt nicht an. Der Motor wird gestoppt.



Dieselbe Erklärung wie oben, nur mit anderer Fahrtrichtung.



Unabhängig laufender Thread/Funktion. In einer Dauerschleife wird der Taster mit der Variablen i3 verglichen. Wenn die Bedingung wahr ist, wird die Variable „count“ um eins erhöht und auf der Konsole ausgegeben. Dann werden die Variablen gesetzt, 1ms gewartet und die Schleife wiederholt.
Kurzfassung: Es wird geprüft, ob eine Flanke (Wechsel des Schaltzustandes) des Tasters erfolgt ist. Wenn ja, wird der Zähler erhöht. Danach wird der neue Zustand des Tasters in „i3“ gespeichert, um eine neue Flanke zu erkennen. Der Zustand der Lichtschranken wird in den Variablen i1 und i2 gespeichert. Nun wird die Schleife wiederholt.



Die „thread_update“ Funktion läuft unabhängig vom Hauptprogramm. In einer Dauerschleife werden in der Statusanzeigen die Variablen i1 und i2 angezeigt. Wenn die Bedingung count<maxcount erfüllt ist wird „count“ und der Inhalt der Variablen „count“ im Display als Text angezeigt. Der Wert der Variablen „count“ wird auf dem Messinstrument angezeigt. Wenn die Bedingung nicht zutrifft, werden ein leeres Textfeld und ein Messinstrument mit 0 angezeigt. Es wird 400ms gewartet und dann die Schleife beginnt von vorne.
Zusammenfassung: Es wird der Zustand der Lichtschranken angezeigt. Dann wird der Zähler auf dem Display, einmal als Zahlenwert und auf einem Messinstrument ausgegeben. Wenn kein Wert da ist wird eine leere Anzeige angezeigt – und das alle 400ms von neuem.

Unterprogramme, die beide Taster auf dem Display des TXT 4.0 abfragt und wenn sie angeklickt werden, wird die entsprechende Variable auf wahr gesetzt.

Befehlsübersicht der Blöcke von Robo Pro Coding

Jeder Controller hat verschiedene Fähigkeiten und somit verschiedene Möglichkeiten sie zu programmieren. Der TXT 4.0 hat alle Fähigkeiten und somit alle Befehle und alle Blöcke.
Beim RX Controller und dem BT Smart Controller sind die fehlenden Blöcke ausgegraut.
Es kann aber sein, das in Zukunft noch andere Befehle dazukommen.

Hinweis:

Unterschiedliche Programmausführung
Es wurde berichtet, dass identische Programme, auf unterschiedlichen Controllern, unterschiedlich laufen. Im Rahmen dieses Buches, kann ich nur darauf hinweisen, dass dem eventuell so ist. Es gibt fortlaufend neue Versionen von Robo Pro Coding und der Firmware der Controllern. Ich kann nicht alles selber ausprobieren und dann im Buch aufnehmen. Es kann also sein, dass manche Sachen schon jetzt behoben sind und dann die Programme doch gleichlaufen.

Die Übersichten entsprechen den Reitern, mit deren Farben, in Robo Pro Coding.

Reihenfolge:

TXT 4.0 Controller / Alle Befehle

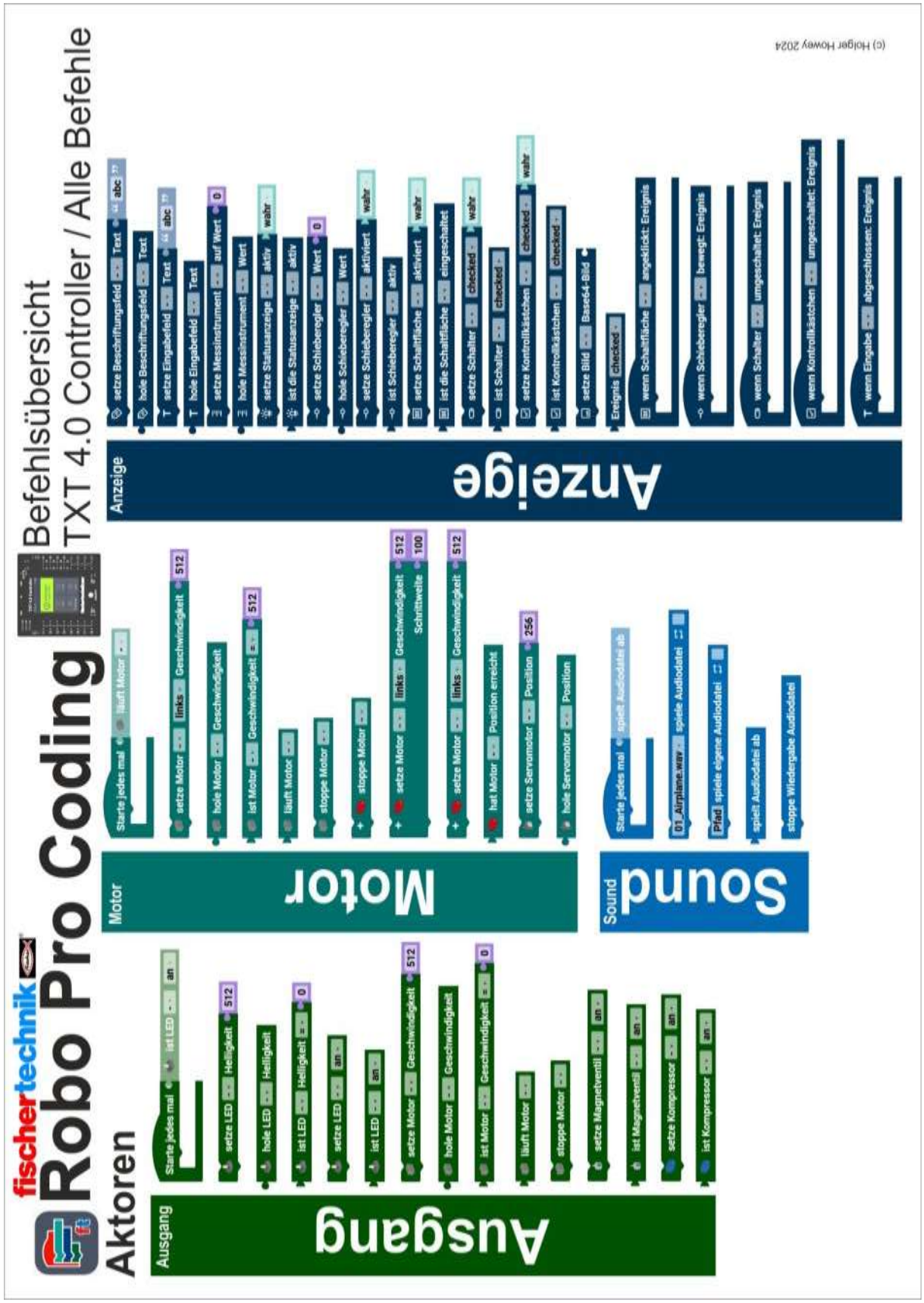
RX Controller

BT Smart Controller

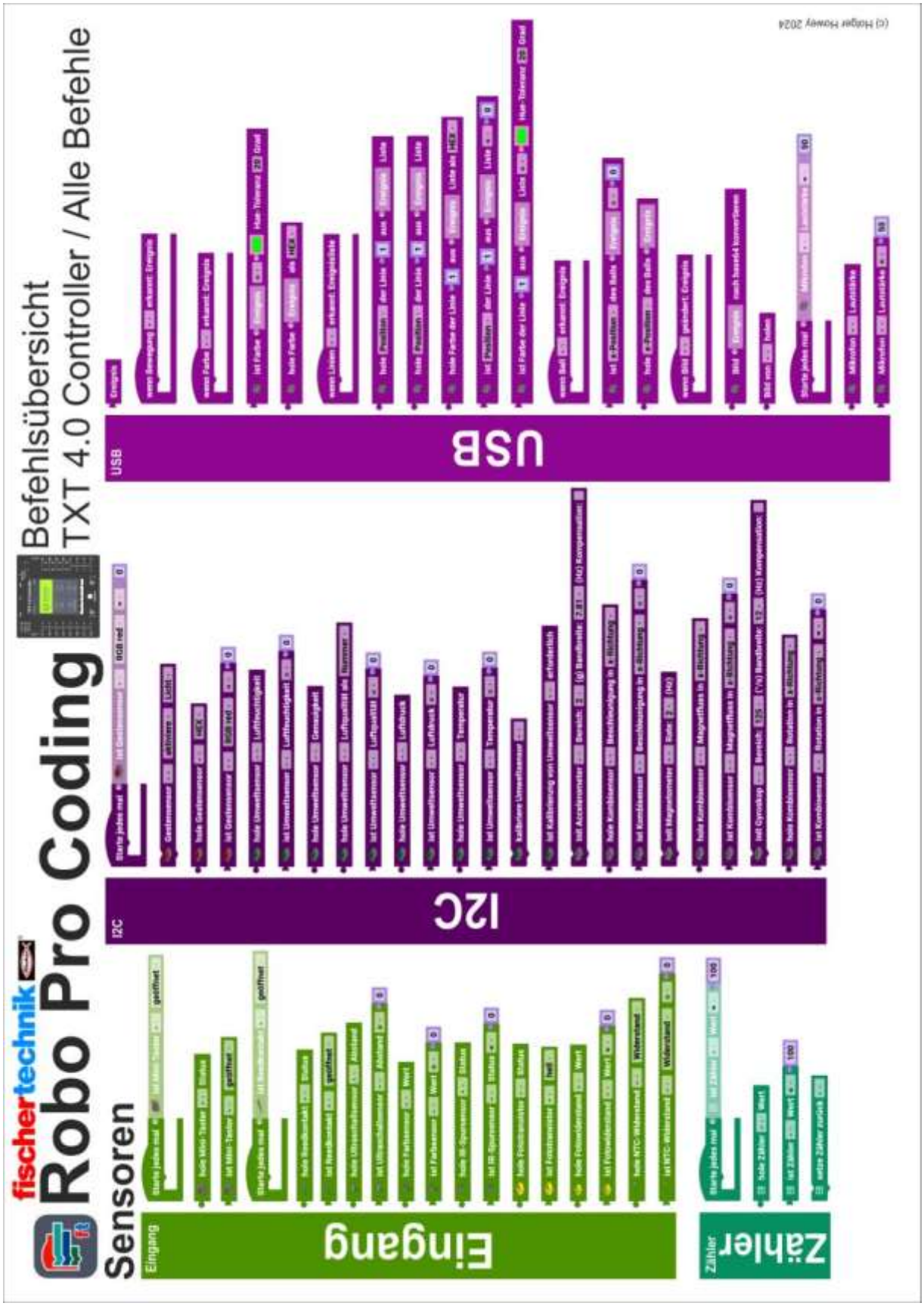
Befehlsübersicht TXT 4.0 / Alle Befehle

Der TXT 4.0 kann alle Befehle von Robo Pro Coding nutzen.

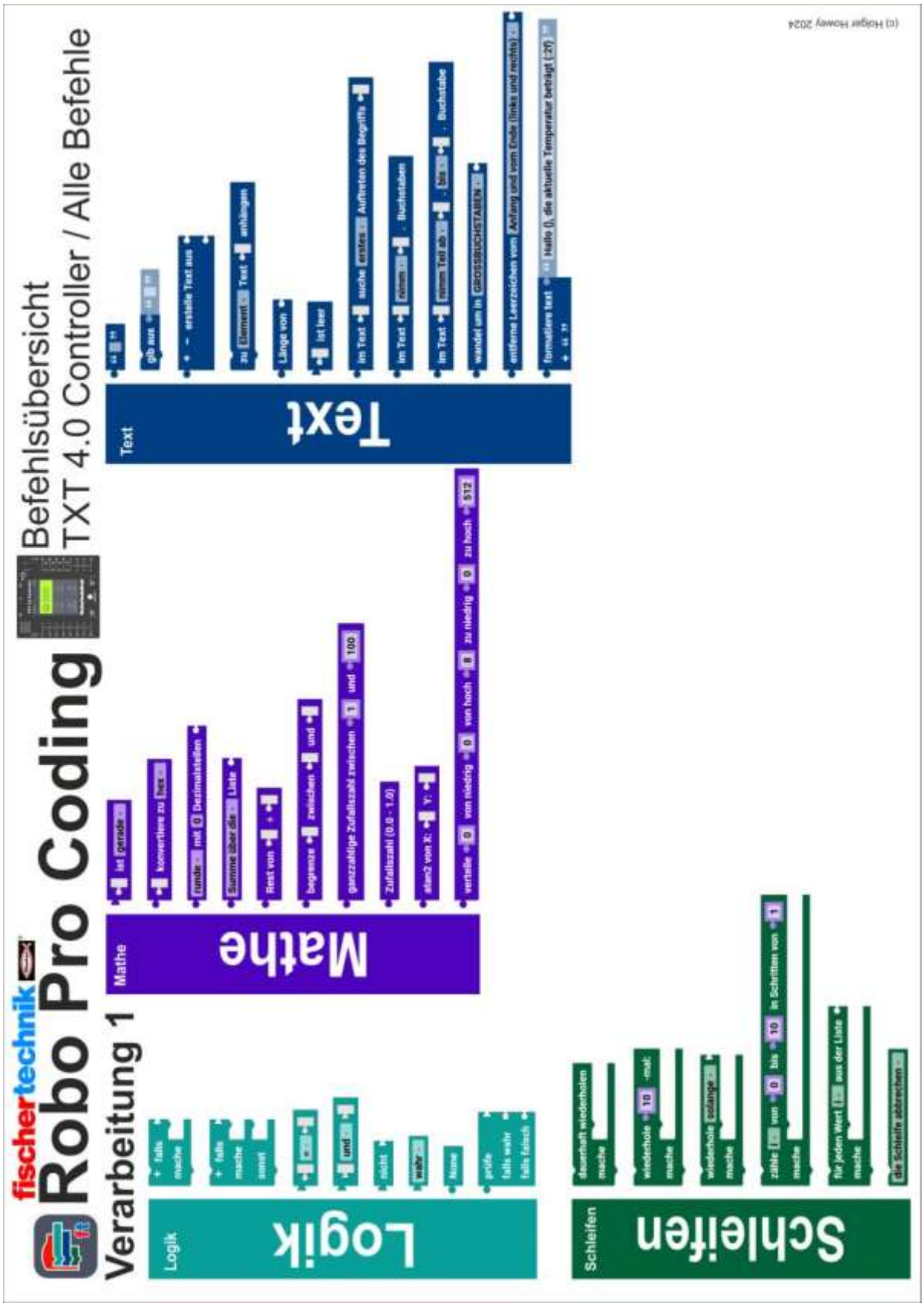
Aktoren


fischertechnik
Robo Pro Coding
Aktoren
Befehlsübersicht
TXT 4.0 Controller / Alle Befehle
Ausgang
Motor
Sound
Anzeige
(c) Holger Howey 2024

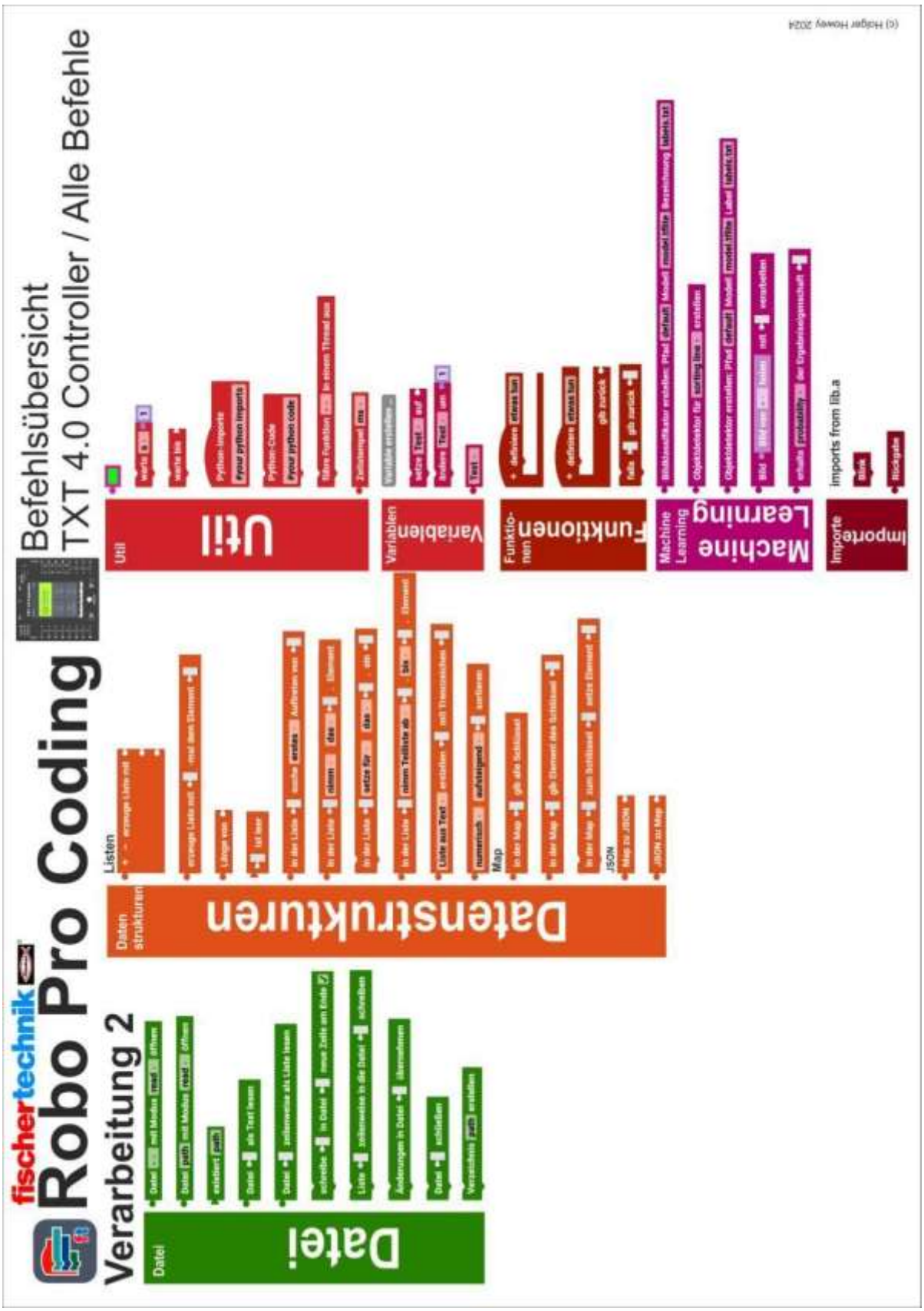
Sensoren


fischertechnik
Robo Pro Coding
Befehlsübersicht
TXT 4.0 Controller / Alle Befehle
Sensoren
Eingang
Zähler
I2C
USB

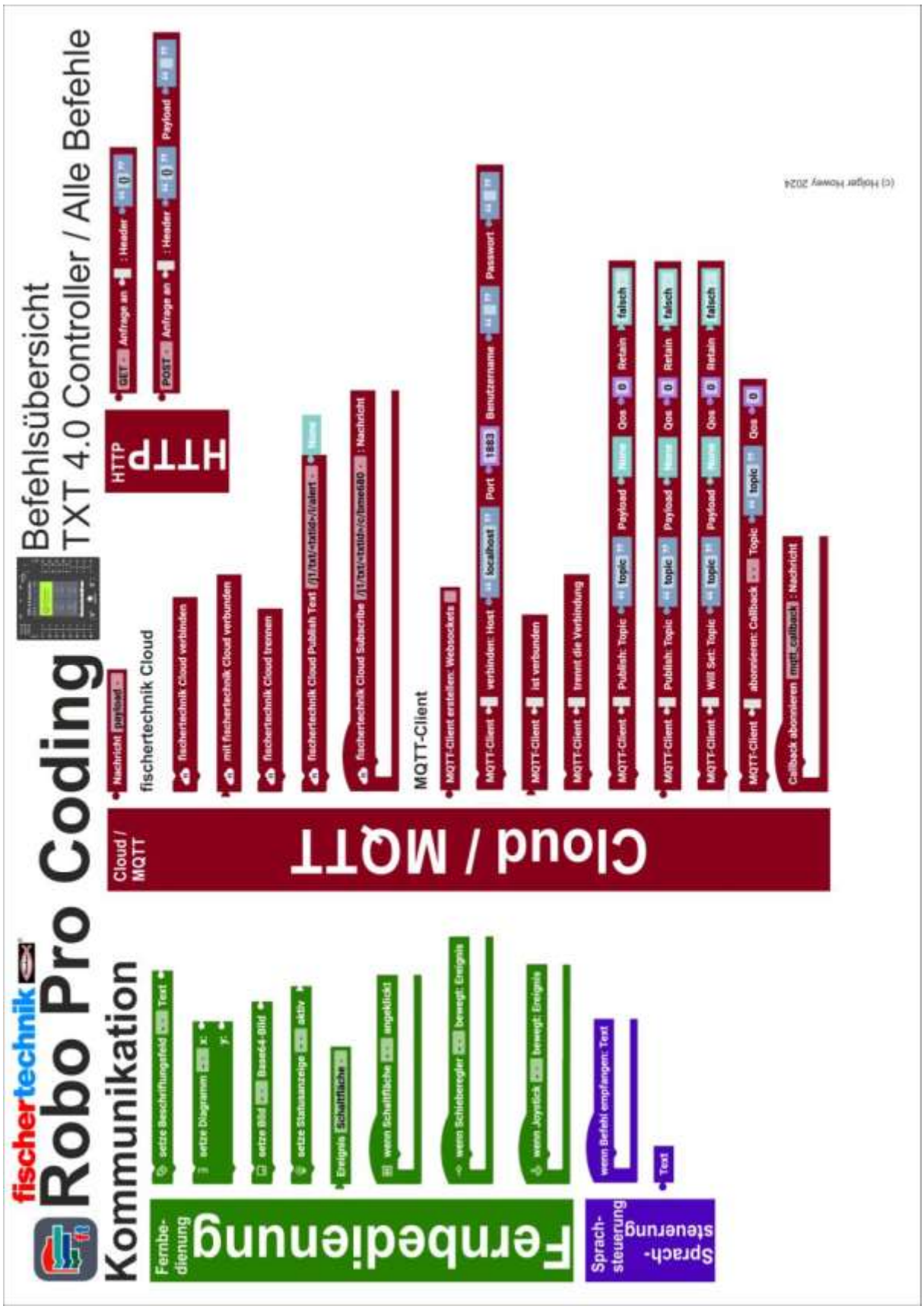
Verarbeitung 1



Verarbeitung 2



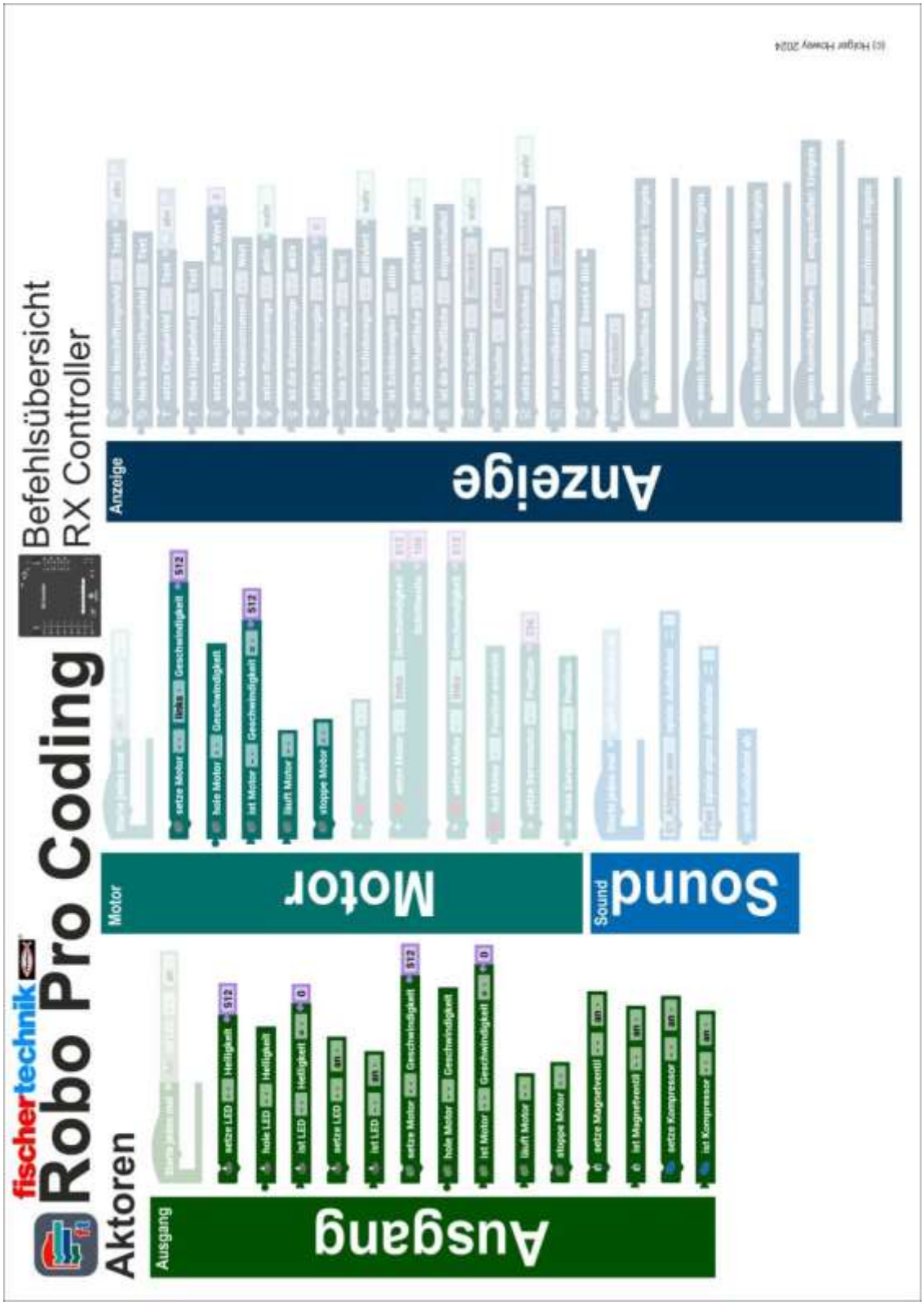
Kommunikation



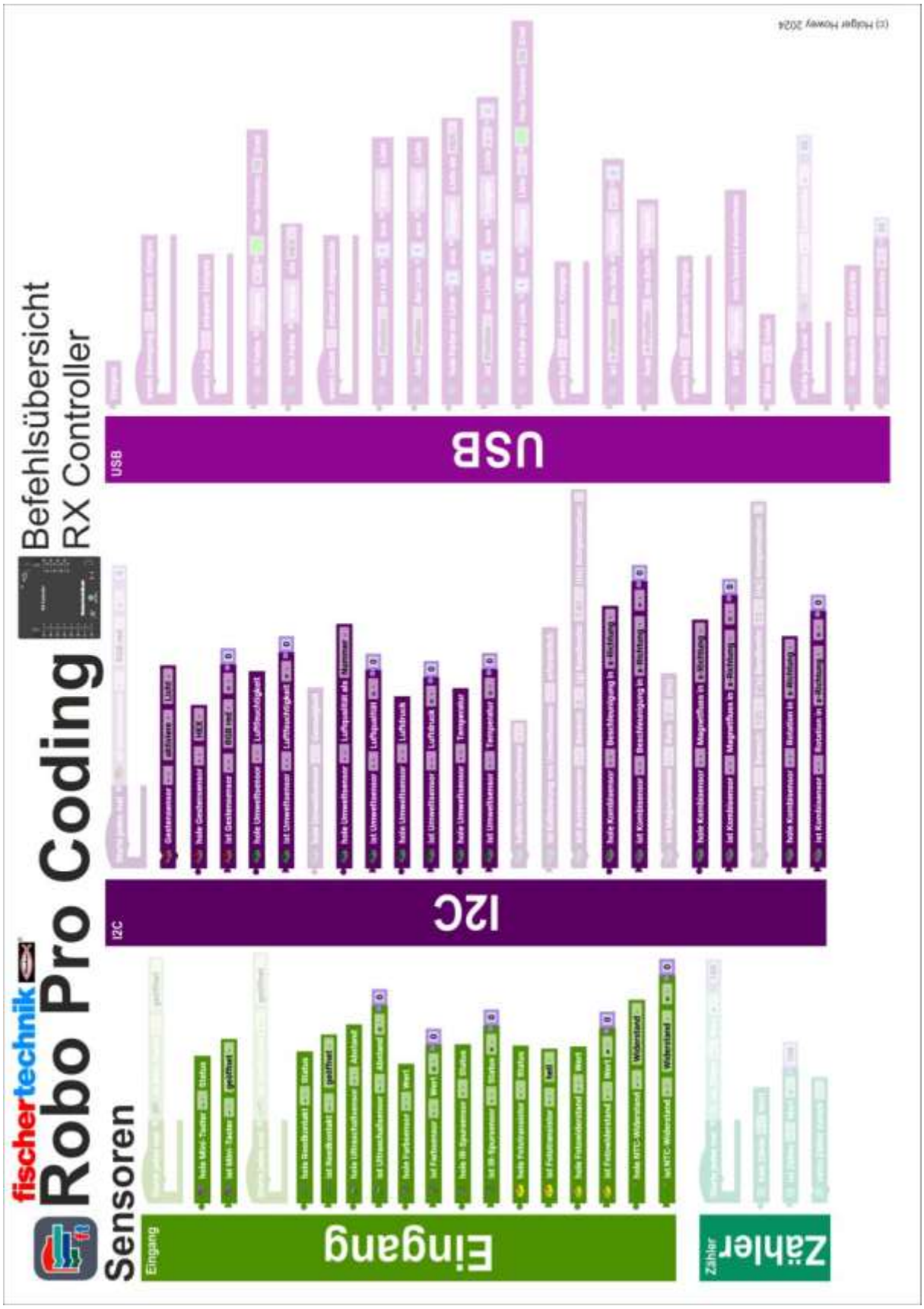
Befehlsübersicht RX Controller

Ausgegraute Blöcke sind nicht für den RX Controller nutzbar.

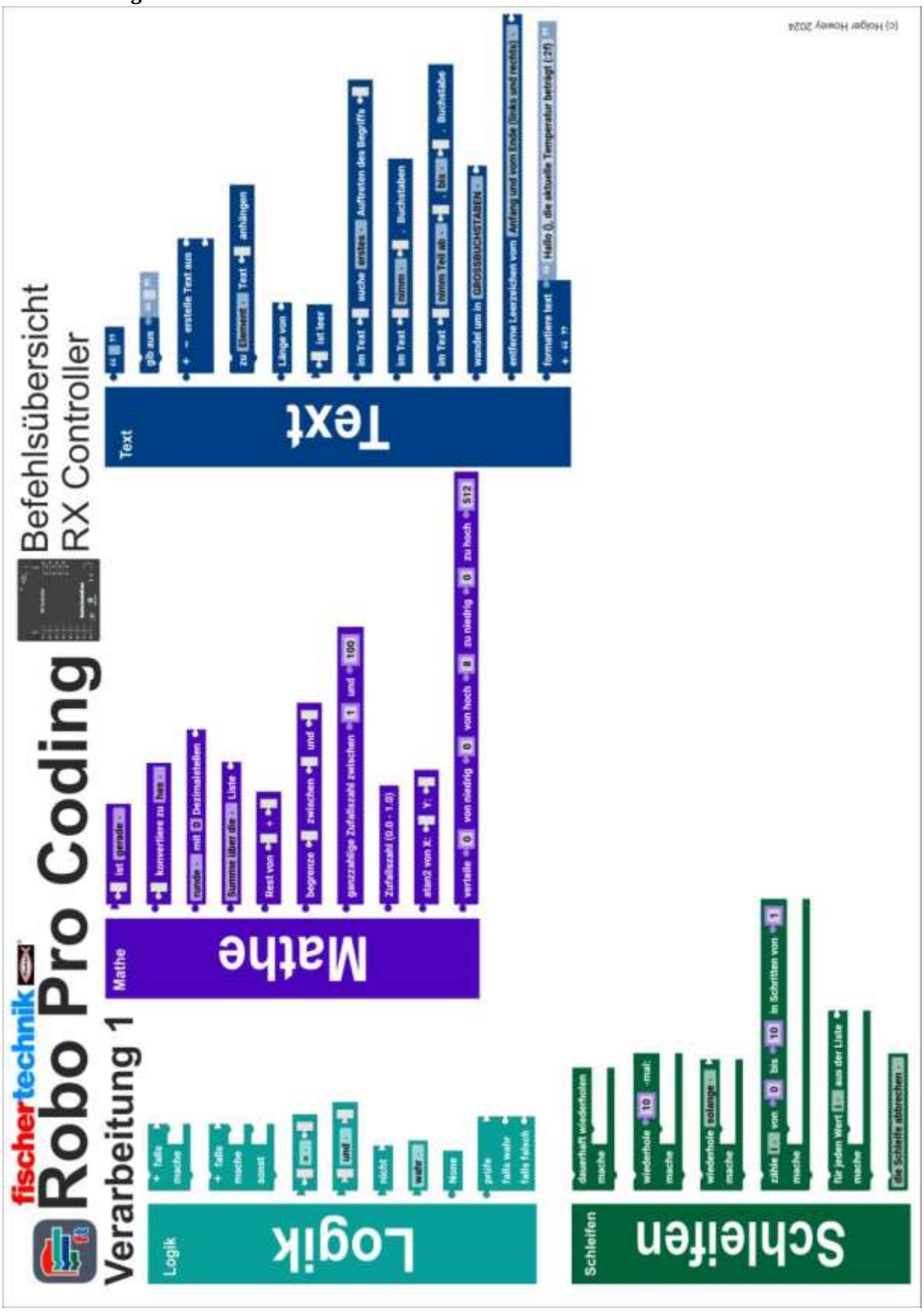
Aktoren



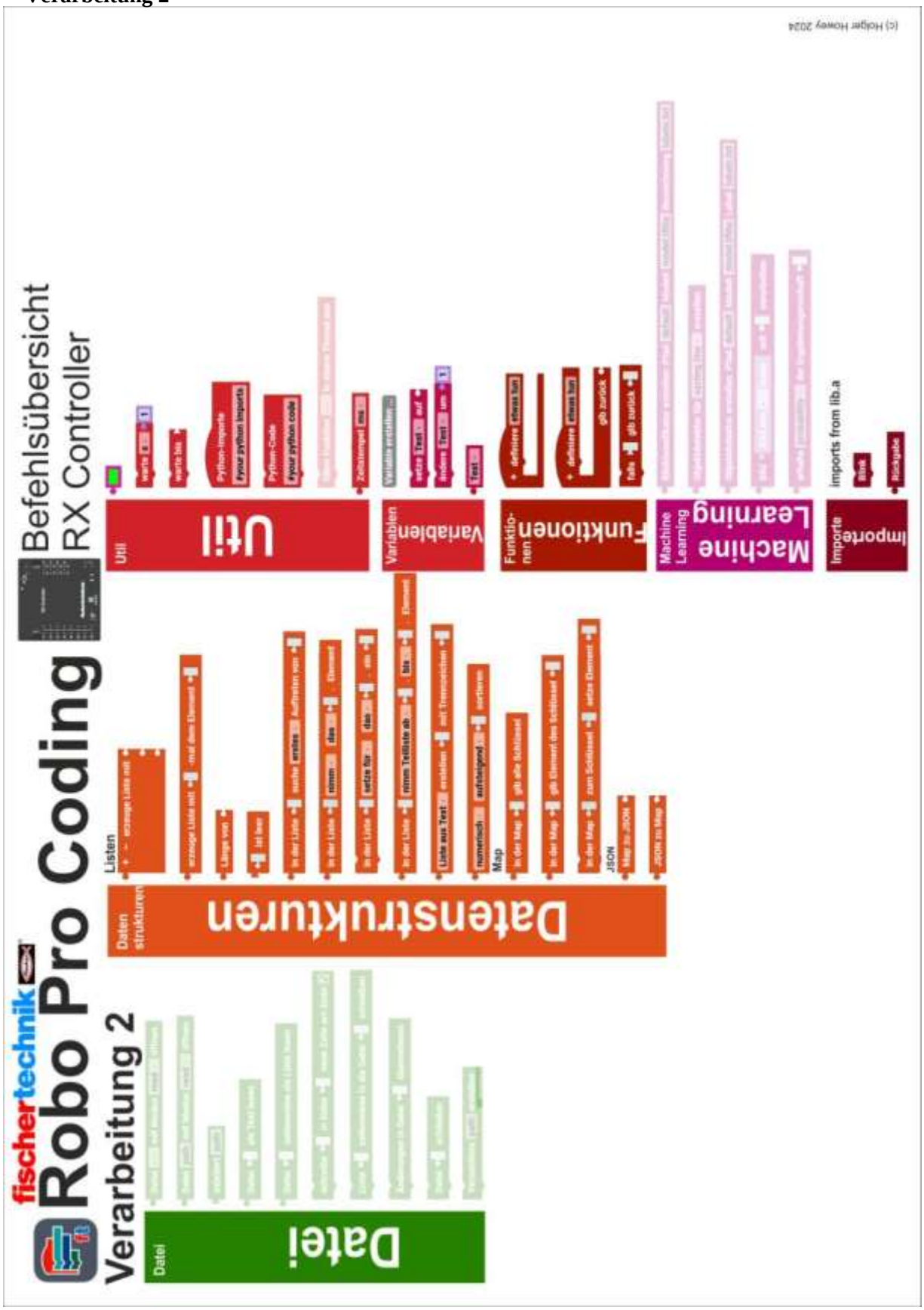
Sensoren


fischertechnik
Robo Pro Coding
Befehlsübersicht
RX Controller
Sensoren
Eingang
Zähler
I2C
USB
(c) Holger Howey 2024

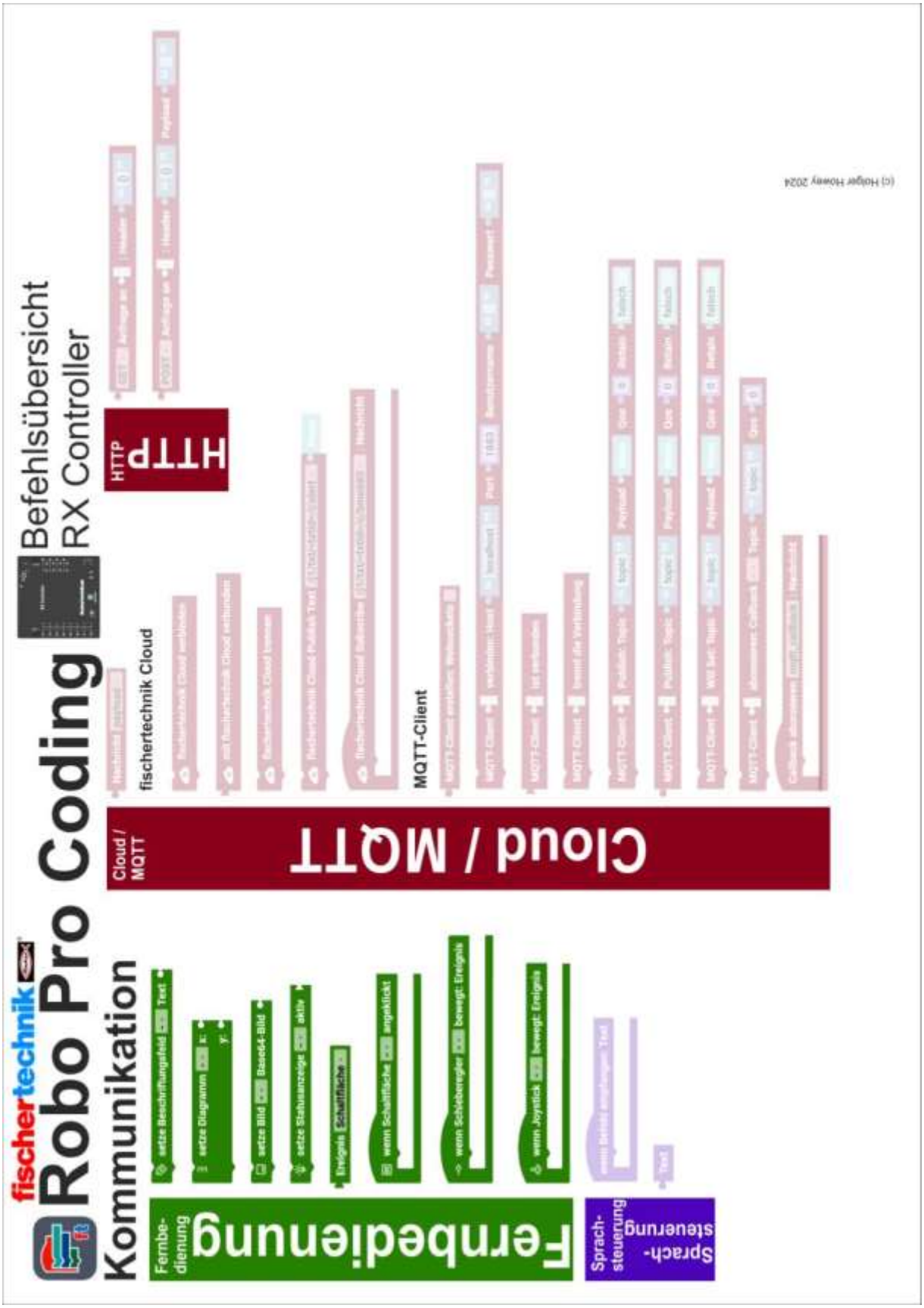
Verarbeitung 2



Verarbeitung 2



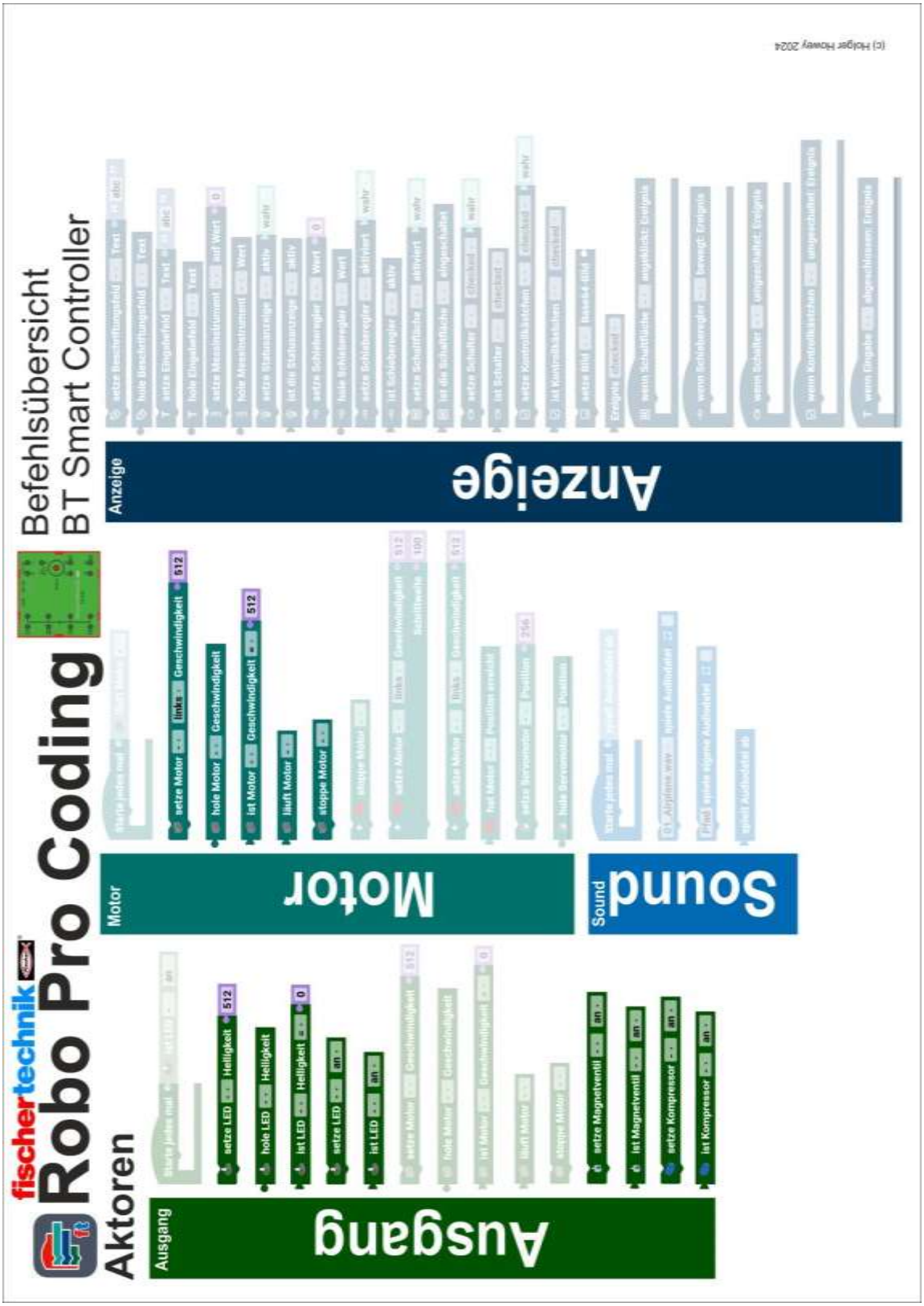
Kommunikation



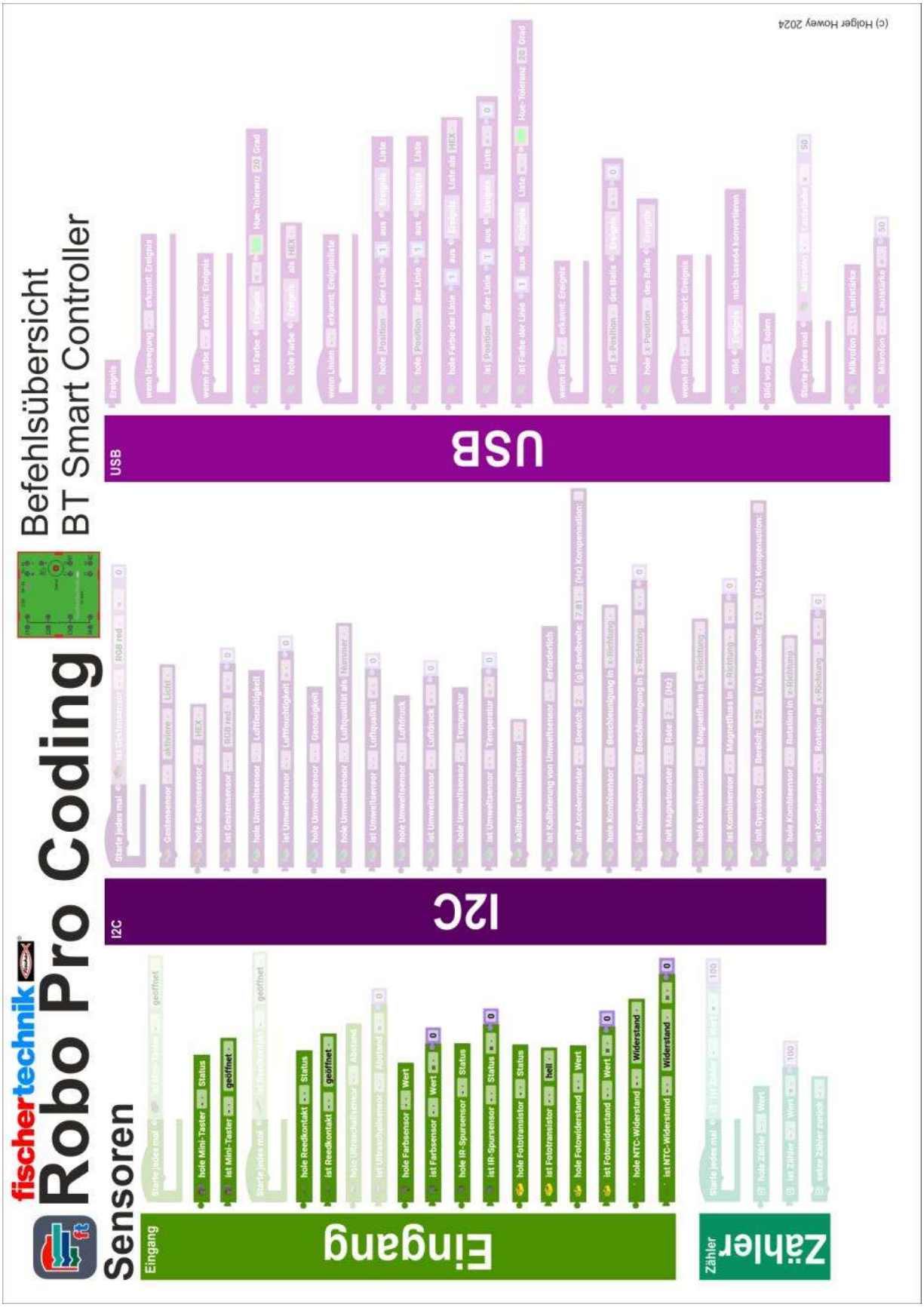
Befehlsübersicht BT Smart Controller

Ausgegraute Blöcke sind nicht für den BT Smart Controller nutzbar.

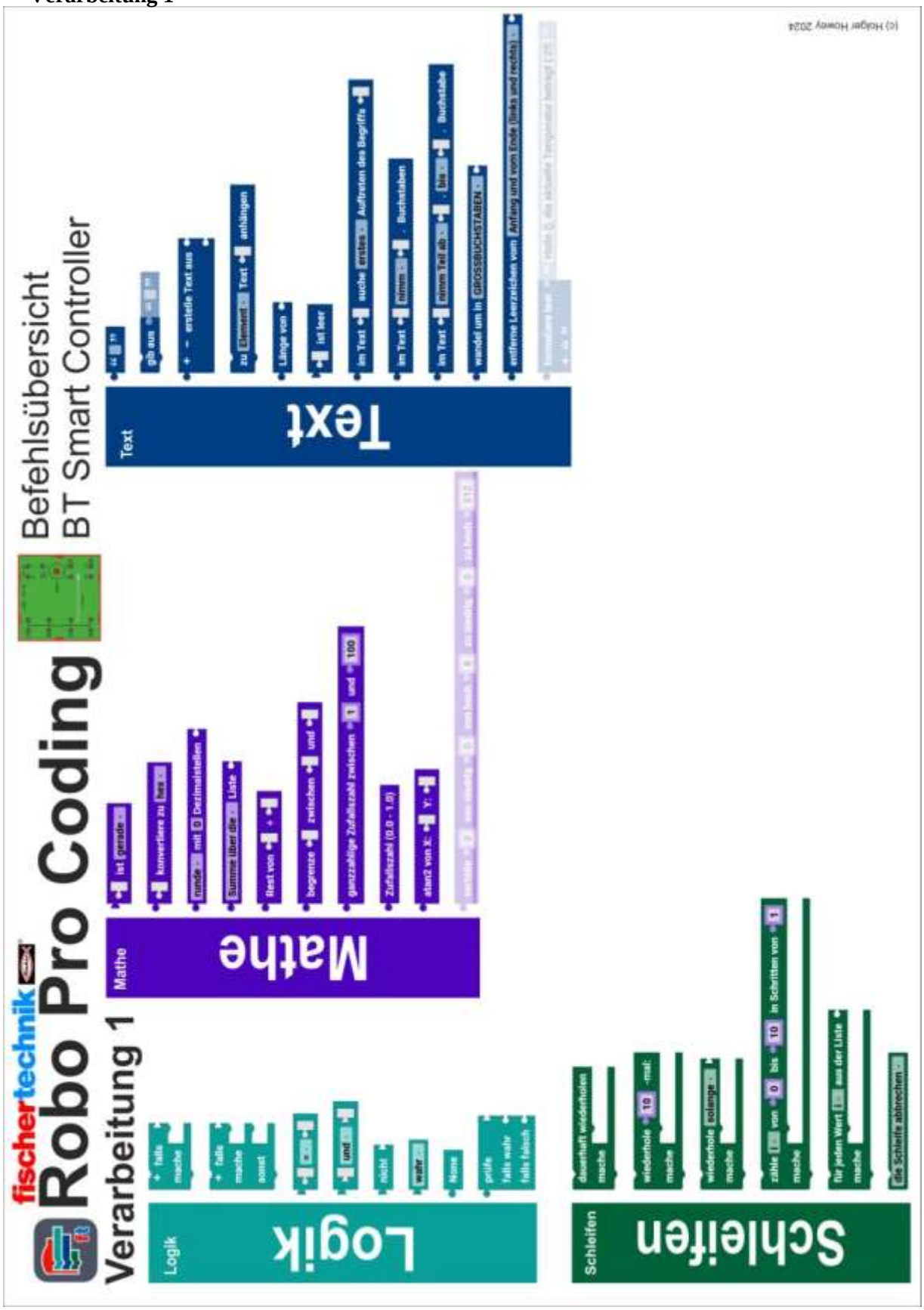
Aktoren



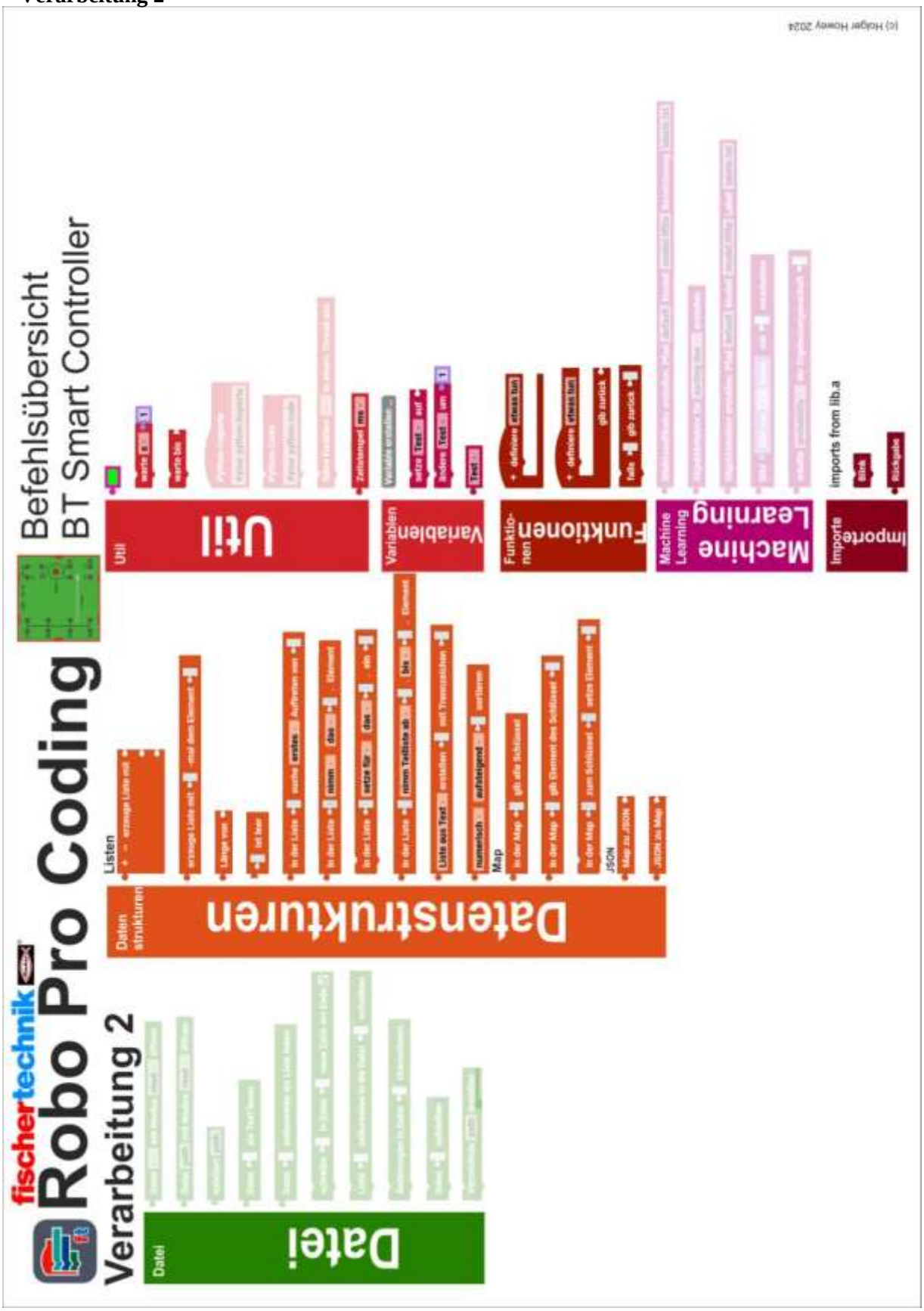
Sensoren


fischertechnik
Robo Pro Coding
Befehlsübersicht
BT Smart Controller
Sensoren
Eingang
Zähler
I2C
USB
(c) Holger Howey 2024

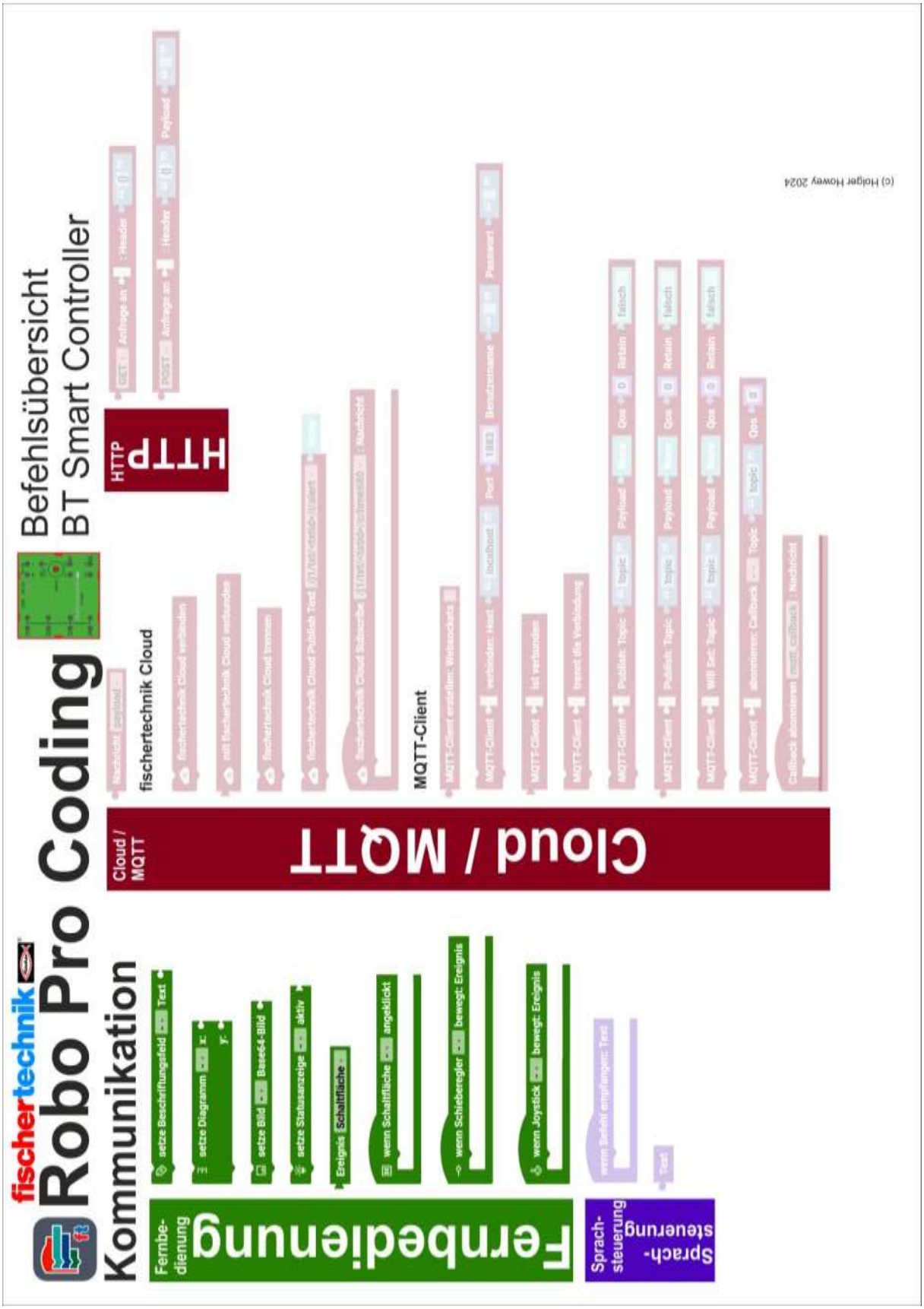
Verarbeitung 1



Verarbeitung 2



Kommunikation



Einstieg in Robo Pro Coding

Alle Befehle von Robo Pro Coding mit vielen Beispielen und zusätzlichen Informationen.

Hilfen und Übersichten zum Programmieren

Viele Informationen zu den Controllern:

- TXT 4.0
- RX-Controller
- BT Smart – Controller

Such in diesem PDF-Buch nach Stichworten. [CTRL] + F