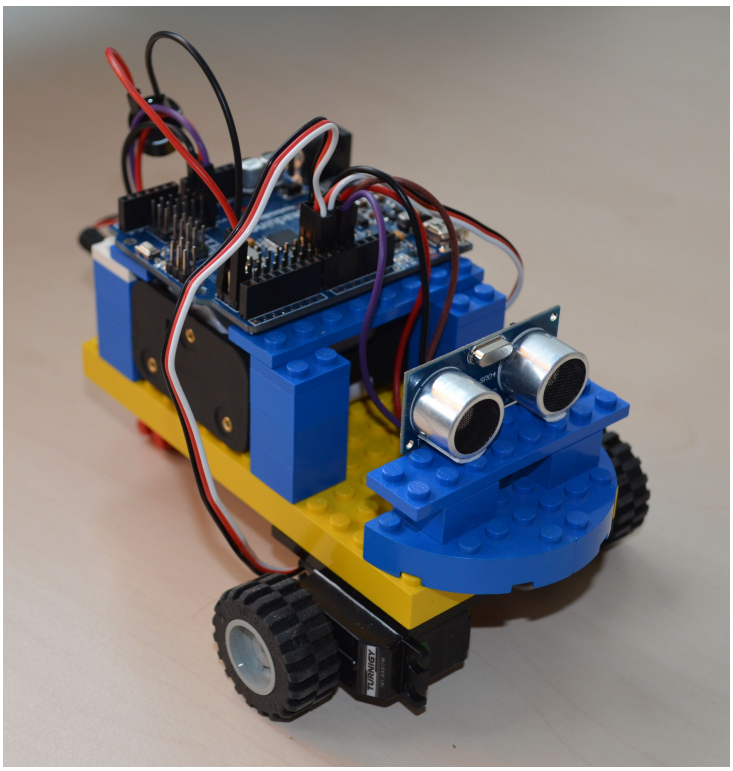
Hochschule Reutlingen
Reutlingen University

Fakultät Technik
Hochschule Reutlingen

Dokumentation Legoino-App



Sven Altenburg
Matrikel-Nr. 722534

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Legoino ist die Low-Cost Version der Hochschule Reutlingen zu dem Lego Mindstroms System. Er besteht aus Legobausteinen, Servomotoren einem Microcontroller und verschiedener Elektronik.
In dieser Dokumentation wird beschrieben, wie man eine App für den Legoino mit Hilfe des App Inventors programmiert.

2 App Inventor

App Inventor verwendet die Open-Blocks Java Bibliotheken zur Erstellung von grafischbasierten Programmiersprachen. Zum Zeitpunkt der Erstellung dieser Dokumentation ist die MIT App Inventor 2 (Beta) in der Version nb143e Verfügbar.

Abbildung 2.1: Logo App Inventor [1]

Der App Inventor ermöglicht es dem Programmierer, mit Hilfe der grafischen Schnittstelle per Drag and Drop, grafische Blöcke zu einer App für Smartphones (Android-System) zu erstellen.

Die Online basierende Programmierumgebung des App Inventors besteht aus 2 Ebenen:

Ebene	Beschreibung
Design-Editor	Hier wird das Layout der App gestaltet.
Block-Editor	Programmierung der im Design Editor selektierten Komponenten.

Da die Programmierumgebung online basiert, muss kein extra Programm heruntergeladen werden. Man benötigt jedoch ein Google Account, um eine eigene App zu programmieren.

2.1 Design-Editor

Wie schon erwähnt, wird mit dem Design Editor das Layout für die App erstellt. Darüber hinaus kann man hier auch die benötigten Sensoren (z.B. Beschleunigungssensor) aktivieren, die man für die eigene App benötigt. Der Design-Editor besteht aus 4 Bereichen.

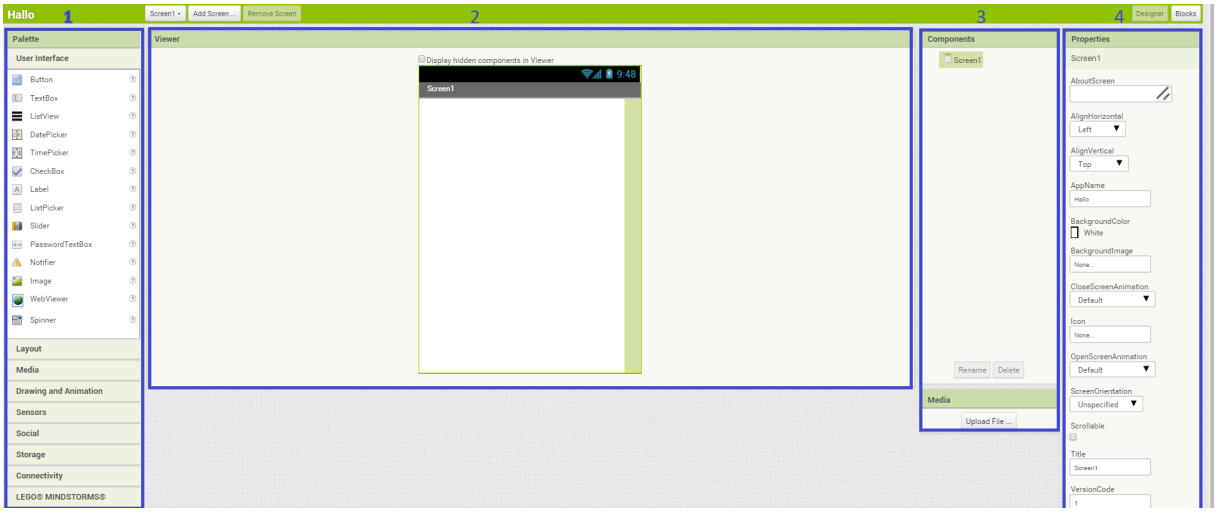


Abbildung 2.2: Design-Editor

Auf der linken Seite (1) in der Abbildung 2.2 , ist die Auswahl an Werkzeugen (Buttons, Label etc.), Layouts, Sensoren, die man für die eigene App zur Verfügung hat, ersichtlich. An Position 2 erstellt man mit der Auswahl, die man an Position 1 gewählt hat, das Layout für die App. Bei der Position 3 wiederrum sieht man alle Komponenten (Werkzeuge, Layouts, Sensoren), die man ausgewählt hat. Ganz rechts (4) in der Abbildung 2.2 sind die Eigenschaften der ausgewählten Komponenten ersichtlich, die man verändern kann.

2.2 Block-Editor

Wie erwähnt erstellt man hier mit Hilfe der im Design-Editor erstellten Layouts die eigene App. Dabei stehen dem Programmierer mehre Blöcke (z.B. Schleifen und Mathematische Funktionen) zur Verfügung.

Abbildung 2.3: Blocks

Jede Komponente die man ausgewählt hat, bekommt eigene Blöcke, die man mit den Standardblöcken kombinieren kann. In Abbildung 2.3 ist so ein Block zu sehen. Der Block bewirkt in einer App, dass beim Drücken des Button1, die Globale Variable *name* den Wert 10 zugeteilt wird.

3 Umsetzung

In den folgenden Kapitel wird beschrieben, wie man die App für den Legoino mit dem App Inventor erstellt.

3.1 Projekt erstellen

Nach der Anmeldung, wird man auf die Seite "Meine Projekte" weitergeleitet. Hier sind alle Projekte, die man mit Hilfe des App Inventors erstellt hat, zu sehen. Beim Erstellen eines neuen Projektes muss man den Button "Start new projekt" betätigen. Anschließend erscheint ein neues Fenster, in dem man den Namen des Projektes festlegen kann.
Wenn man sich für einen Namen entschieden hat, muss dieser mit dem Button "OK" bestätigt werden. Nach der Bestätigung wird das neue Projekt angelegt, und der Design-Editor erscheint am Bildschirm.

3.2 Layout erstellen

Als erstes sollte man sich bei der App-Entwicklung entscheiden, welche Orientierung das Display haben soll. Da der Legoino später mit den Beschleunigungssensoren gesteuert wird, ist es von Vorteil, die Orientierung "Landscape" zu wählen. Die Orientierung des Displays kann man in den Eigenschaften der Komponente "Screen1" einstellen. Dafür muss die Eigenschaft "ScreenOrientation" auf "Landscape" geändert werden.

Anschließend wird das Layout der App mit den verschiedenen vordefinierten Layouts des App Inventors erstellt. Dazu muss der Reiter Layout ganz links bei ”Palette” angewählt werden. Für das Layout wird zuerst ein ”TableArrangement” in das Display gezogen und anschließend folgende Einstellungen geändert:

- Columns: 2
- Height: fill parent (passt das Layout an das Display des Smartphones an)
- Width: fill parent
- Rows: 1

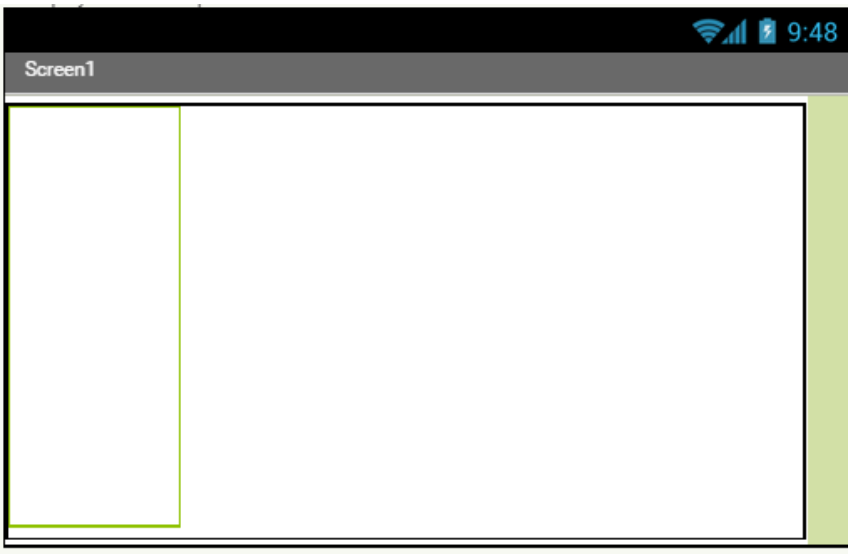


Abbildung 3.1: TableArrangement

Danach wird noch ein ”TableArrangement” gebraucht. Dieses Mal wird es in die linke Spalte des anderen Layouts gezogen. Auch hier muss man die Einstellungen ändern, dieses mal nur die Einstellung Row auf 6 (6 Spalten).

Wichtig: Es kann sein das die Einstellungen des 2 TableArrangement noch geändert werden muss. Das kann nur jeder selber einstellen, da es an das betreffende Display angepasst werden muss.

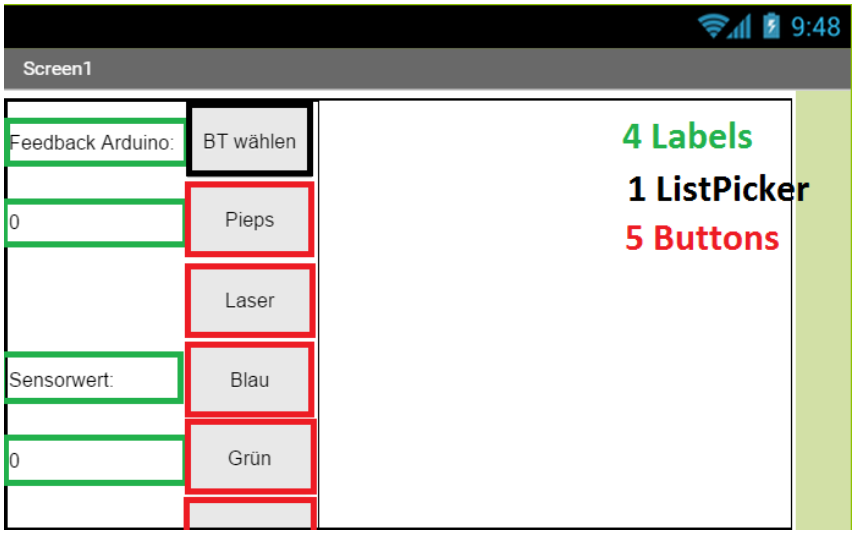


Abbildung 3.2: Layout mit Buttons

Für den nächsten Schritt benötigt man vier "Labels", fünf "Buttons" und ein "ListPicker". Wie in Abbildung 3.2 zu sehen, müssen diese angebracht werden. Dabei sind die vier "Labels" links angebracht, der "ListPicker" rechts oben und die fünf "Buttons" darunter. Den Text der Komponenten wird in den Einstellungen bei Text geändert.(siehe Abbildung 3.2)

Für den Block-Editor ist es von Vorteil, die Komponenten auch einen neuen Namen zu geben, um diese gut auseinander halten zu können. Dabei muss nur bei "Components" die entsprechende Komponente gewählt werden und der Button "Rename" ausgewählt werden. Von Vorteil wäre es, die Namen genauso zu wählen wie der Text der Komponenten.

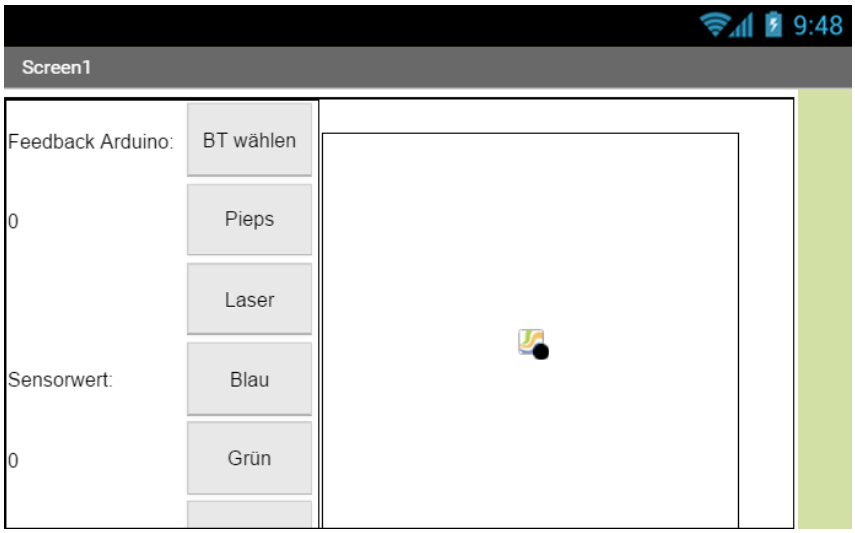


Abbildung 3.3: Fertiges Layout

Zum Schluss muss noch das optische Feedback für die Bewegung erzeugt werden. Hier muss zuerst in der Palette der Reiter "Drawing and Animation" gewählt werden. Danach zieht man die Komponente "Canvas" in das Display neben die Buttons. Hier muss bei den Einstellungen die Größe des Feldes geändert werden. Da es sehr viele verschiedene Display-Auflösungen auf dem Markt erhältlich sind, muss dieses je nach Model geändert werden. Bei dem Testgerät wurde die Größe auf 250*250 Pixel gewählt.
Als letztes wird noch ein Ball in die davor hinzugefügte Komponente eingefügt. Bei den Einstellungen wird x und y so gewählt, dass der Ball in der Mitte des Feldes liegt.
Zusätzlich zum Layout muss hier auch entschieden werden, welche Sensoren man für seine App benötigt. Deshalb muss im Reiter "Sensor" die Sensoren "Clock" und "AccelerometerSensor" ins Display gezogen werden.

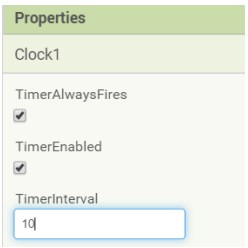


Abbildung 3.4: Clock Einstellung

Der Sensor "Clock" ist für die Zykluszeit (Programmlaufzeit) zuständig, deshalb wird bei ihm in den Einstellungen (TimerInterval) 10 gewählt. Die 10 bedeutet, dass alle 10 Millisekunden der "Clock" Block aufgerufen wird. Zykluszeit muss gewählt werden, sonst wird das Programm zu oft aufgerufen und dadurch die CPU überlastet. Später wird die Verbindung zwischen Legoino und Smartphone über Bluetooth stattfinden. Die Bluetooth Funktion findet man in dem Reiter "Connectivity" an erster Stelle. Diese muss wie die Sensoren in das Display gezogen werden.

3.3 Blocks erstellen

In den folgenden Kapiteln wird gezeigt, wie man die Funktionen in Blöcke erstellt.

3.3.1 Variable definieren

Abbildung 3.5: Variable Definition

Die Legoino App enthält 14 Variablen. Um die Variable zu definieren, muss bei Blocks "Variables" angeklickt werden und "initialize global name to in den Viewer" in das Fenster gezogen werden. Wie erwähnt muss dieses 14-mal gemacht werden. Danach können diese benannt und mit einem Anfangswert versehen werden.

Tabelle 1: Variable

Variablenamen	**Anfangswert**
Motorsteuerung	0
move_x	0
move_y	0
kippWinkelX	0
kippWinkelY	0
Feedback	create empty list
Aktuell_Min	-90
Aktuell_Max	90
Ziel_min	0
Ziel_max	250
servo_min	0
servo_max	180
seitenRuder	90
hoehenRuder	90

Alle Anfangswerte können mit dem ersten Block in "Math" vordefiniert werden. Einzig die Variable Feedback wird als Liste definiert. Dafür benötig man in dem Reiter "List", den Block mit dem Namen "create emty list". Ziel_max muss nach Fernster Größe der Komponente "Drawing and Animation" gewählt werden.

3.3.2 Bluetooth Funktion

Im Layout wurde ein "ListPicker" gewählt, dieser wird für die Bluetooth Verbindung genutzt.

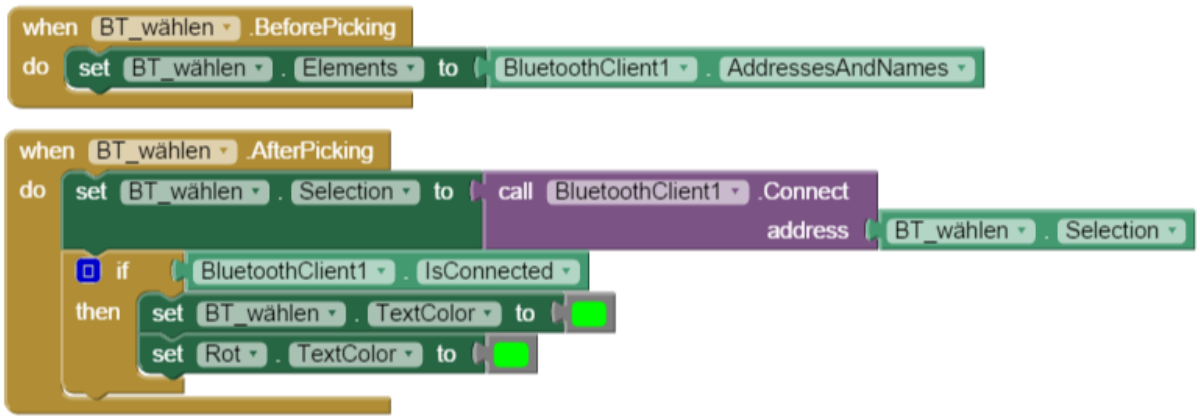


Abbildung 3.6: Bluetooth Funktion

In Abbildung 3.6 sind 2 Kontroll-Blöcke (gelb) vom "ListPicker" zu sehen. Die erste ist dafür da, dass nach dem Betätigen des "ListPickers" eine Liste aller möglichen Bluetooth Verbindungen angezeigt werden. Der 2. Block hat 2 Funktionen:

1. Verbindung herstellen mit der Auswahl die man getroffen hat
2. Falls Verbindung hergestellt wurde, zeige den Text auf dem "ListPicker" und dem "Button" (rot) grün an.

Alle hierfür benötigten Blöcke findet man in den Reitern "ListPicker", "Control", "Colors" und "BluetoothClient1".

3.3.3 Kippwinkel Berrechnung

Da die Bewegungssteuerung des Legoinos mit dem Kippen des Smartphones erfolgen soll, sind die Kippwinkel sehr wichtig.

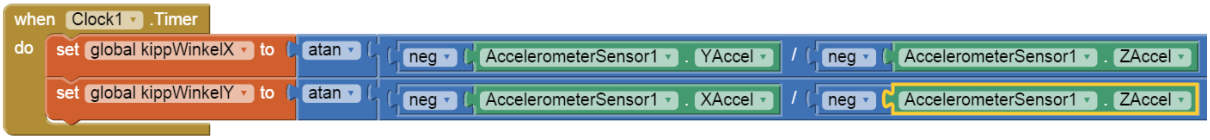


Abbildung 3.7: Kippwinkel

Die Berechnung erfolgt mit Hilfe des Beschleunigungssensors. Die Werte werden in den zuvor angelegten Variablen (KippWinkelX, KippWinkelY) gespeichert. Folgende Formeln wurden in der Abbildung 3.7 verwirklicht.

$$\begin{aligned} KippWinkelX &= \arctan \frac{BeschleunigungswertY}{BeschleunigungswertZ} \\ KippWinkelY &= \arctan \frac{BeschleunigungswertX}{BeschleunigungswertZ} \end{aligned} \tag{1}$$

Die in Abbildung 3.7 verwendeten Blöcke sind in folgenden Registern zu finden:

- Clock1
- Variables
- Math
- AccelerometerSensor1

3.3.4 Ballbewegung

Der Ball dient als visuelles Feedback für den Benutzer, er bewegt sich je nach dem wie man das Smartphone kippt.

Abbildung 3.8: Ballbewegung

Dazu muss der Winkel in Pixel umgerechnet werden. Mit folgender Formel lässt sich dieses bewerkstelligen.

$$\begin{aligned} move\_x &= \frac{(KippWinkelX - Aktuell\_Min) * (Ziel\_max - Ziel\_min)}{(Aktuell\_Max - Akteull\_Min) + Ziel\_min} \\ move\_y &= \frac{(KippWinkelY - Aktuell\_Min) * (Ziel\_max - Ziel\_min)}{(Aktuell\_Max - Akteull\_Min) + Ziel\_min} \end{aligned} \tag{2}$$

Für die Formelumsetzung muss zuerst ein set-Block aus "Variables" unter die Blöcke aus Abschnitt 3.3.3 gezogen werden. Anschließend wird ein Dividierblock dahinter angelegt. Im Dividierblock wird in die erste Lücke ein Multiplizierblock gelegt. In die rechte Lücke wird ein Addierblock gezogen. Jetzt wird wie nach Formel (2) in beide Lücken des Multiplizierblocks und in die erste Lücke des Addierblocks jeweils ein Subtrahierblock hinein gelegt. Am Ende wird mit den get-Blocks aus "Variables" die Formel im App Inventor

abgeschlossen. Man muss nur noch die passenden Variablen in die get-Blocks eintragen. Die Ballbewegung selbst erzielt man mit dem lilafarbenen Block in Abbildung 3.8 und den errechneten Werten, die mit den get-Blöcken angefügt werden. Der lilafarbenen Block ist in dem Reiter "Ball1" zu finden.

3.3.5 Steuerung Legoino

Die Lenkbewegung/Vorwärts- und Rückwärtsfahren berechnet man mit derselben Formel wie in Abschnitt 3.3.4. Hier wird nur der Winkel nicht in Pixel umgewandelt, sondern in Servowinkel. Diese Blöcke befinden sich weiterhin in dem Clock-Kontrollblock.

Abbildung 3.9: Steuerung Legoino

Ein weiterer Unterschied ist, dass die errechneten Werte noch gerundet werden. Dafür muss nach dem set-Block ein round-Block eingefügt werden, anschließend kommen die gleichen Blöcke wie in Abschnitt 3.3.4 vor.
Wie erwähnt wird dieses Mal auf Servowinkel umgewandelt, deshalb ist die Formel (2) in folgenden Stellen anders.

Tabelle 2: Zu ersetzende Variablen

alte Variable	**neue Variable**
Ziel_min	servo_min
Ziel_max	servo_max

3.3.6 Button Funktion

Im Layout wurden 5 Buttons hinzugefügt, die alle unterschiedliche Funktionen am Legoino ausführen. Um diese Funktionen ausführen zu können, wird eine Variable bei jeder Buttonbetätigung geändert.

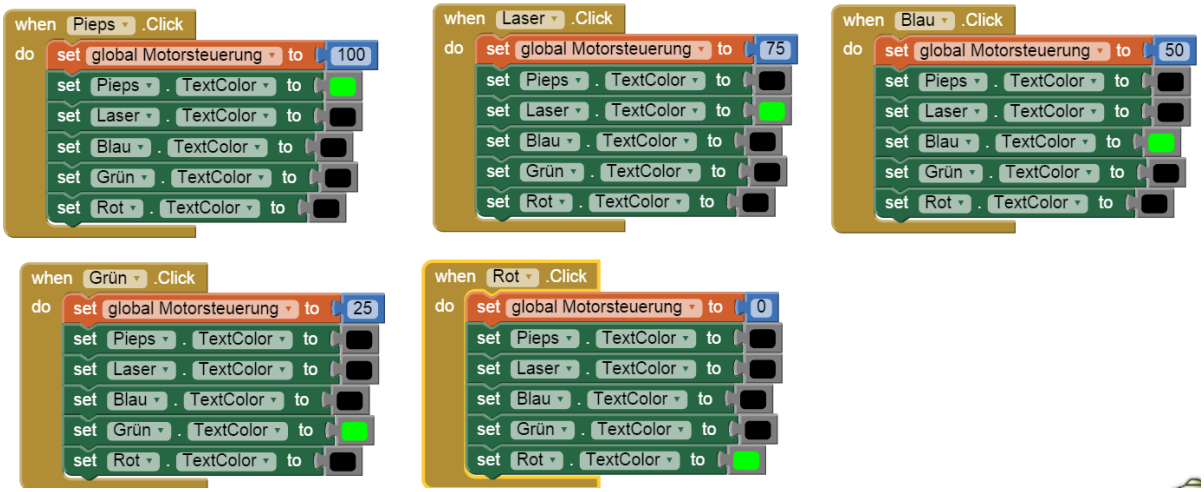


Abbildung 3.10: Button Funktion

Für ein optisches Feedback wird nach dem Klicken auf einen Button dessen Textfarbe auf Grün dargestellt. Alle Blöcke von Abbildung sind in folgenden Reitern zu finden.

- Button (Name der Buttons)
- Variable
- Math
- Colors

3.3.7 Senden von Daten über Bluetooth

Bisher konnte man den Legoino noch nicht steuern, da es noch keine Blocks gab, die mit ihm kommunizierten.

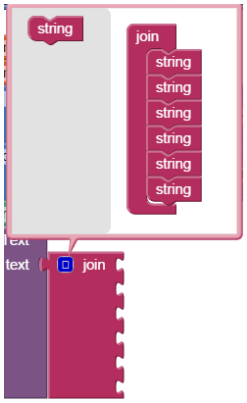


Abbildung 3.11: Sendeliste erstellen

Der Legoino erwartet einen String, in dem die Variablen seitenRuder, hoehenRuder und Motorsteuerung enthalten sind. Diese Variablen müssen mit einem Komma getrennt werden und der Abschluss des Strings muss mit \n enden.
Diese 6 einzelnen Strings werden zusammengefügt mit dem Block "join" aus dem Reiter "Text". In Abbildung 3.11 ist so ein Block zu sehen. Standartmäßig beinhaltet dieser nur 2 Strings. Deshalb muss auf das blaue Symbol geklickt und anschließend Strings in den joinblock geschoben werden.

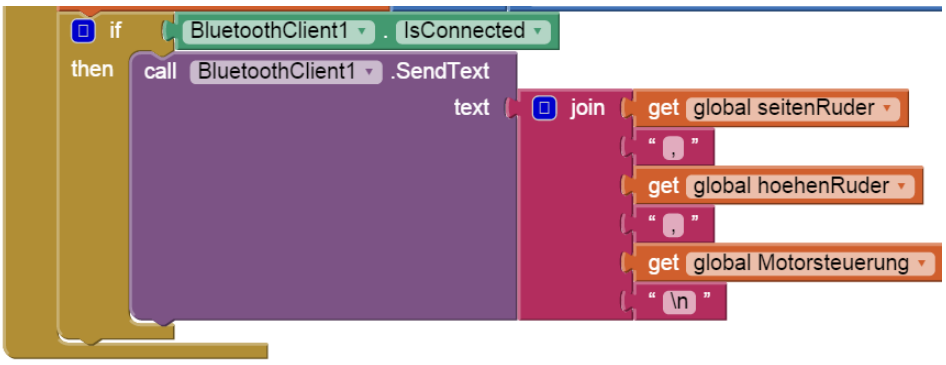


Abbildung 3.12: Daten Senden

Die Blöcke, die in Abbildung 3.12 zu sehen sind, werden unter die Blöcke aus Abschnitt 3.3.5 angebracht. Diese Blöcke findet man in den Reitern "Control", "BluetoothClient1", "Text und Variables".

3.3.8 Daten empfangen

Wenn die Verbindung zum Legoino hergestellt wurde, können auch Daten von ihm empfangen werden.

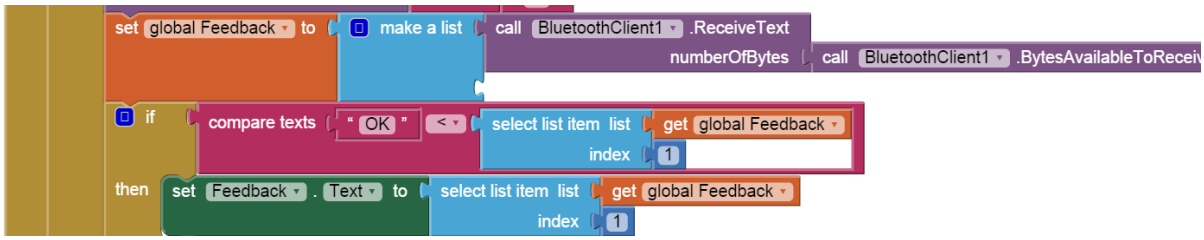


Abbildung 3.13: Daten empfangen

Es ist ein String mit dem Inhalt OK und den Ultraschallwerten. Nur wenn die Variable Feedback ein OK an erster Stelle steht wird das Label geändert.
Die in Abbildung 3.13 zu sehende Blöcke sind in den Registern ”Variables”,”Control”, ”Lists”, ”BluetoothClient1”, ”Text” und ”Button” (Feedback) zu finden.

4 Verbindung

In diesem Kapitel wird kurz die Möglichkeiten dargestellt, wie man den App Inventor mit dem Smartphone verbindet.

4.1 USB-Kabel

Wie die meisten Entwicklungsumgebungen für Apps, bietet auch der APP Inventor die Möglichkeit das Smartphone mit Hilfe eines USB-Kabels zu verbinden. Der App Inventor hat dabei 2 Möglichkeiten von Verbindungen.

1. App erstellen auf dem PC laden und auf das Smartphone übertragen. Hierfür muss der Reiter Build ausgewählt und App (save .apk to my computer) angeklickt werden. Anschließend kann man die apk auf das Smartphone übertragen. Zusätzlich muss im Smartphone die Option ”unbekannte Quellen” aktiviert sein.

2. Live Programmierung: App Inventor bietet auch die Möglichkeit ”Live zu sehen was man gerade programmiert”. Dafür muss zuerst auf folgender Seite ein Programm und die App MIT AI2 Companion geladen werden.
   Link: http://appinventor.mit.edu/explore/ai2/setup-device-usb.html
   Wenn das Programm installiert wurde und geöffnet ist, kann man Live Programmieren über das USB-Kabel.
   Hierfür muss nur der Reiter Connect ausgewählt und USB angeklickt werden.

Wichtig: Für diese Anwendungsmöglichkeit muss der Treiber des Smartphones installiert werden

4.2 WLAN-Verbindung

Der App Inventor bietet auch die Möglichkeit, über WLAN die App Live zu programmieren. Hierfür muss nur Reiter Connect ausgewählt und Al Companion angeklickt werden.

Wichtig: Auch hier benötigt das Smartphone die App MIT AI2 Companion, und der PC und das Smartphone müssen im selben WLAN-Netz sein.

Über WLAN hat man auch die Möglichkeit, die .apk herunterzuladen. Mithilfe des Reiters Build und APP(provide QR code for.apk) wird ein QR code angezeigt. Diesen kann man mit der App MIT AI2 Companion abscannen, und anschließend lädt die .apk automatisch auf das Smartphone.

4.3 Hotspot Smartphone

Wenn ein 2. Smartphone ein Hotspot startet, und der PC und das 1. Smartphone in diesem Hotspot WLAN sind, hat diese Methode dieselben Möglichkeiten wie Abschnitt 4.2 auch.

4.4 Mobiles Internet

Falls keine der oberen genannten Methoden verfügbar sind, kann man die App auch mit dem Mobilen Internet downloaden. Hierfür muss nur der QR code erstellt werden und mit Hilfe der MIT AI2 Companion gescannt werden.

5 Literaturverzeichnis

[1] **Logo App Inventor**
MIT App Inventor 2 (Beta)
http://btmdt5.mat.uni-bayreuth.de/?Projekte___Schuelerworkshops___Kleiner_Tag_der_Begabtenfoerderung_2015
Zuletzt aufgerufen: 08.06.2015