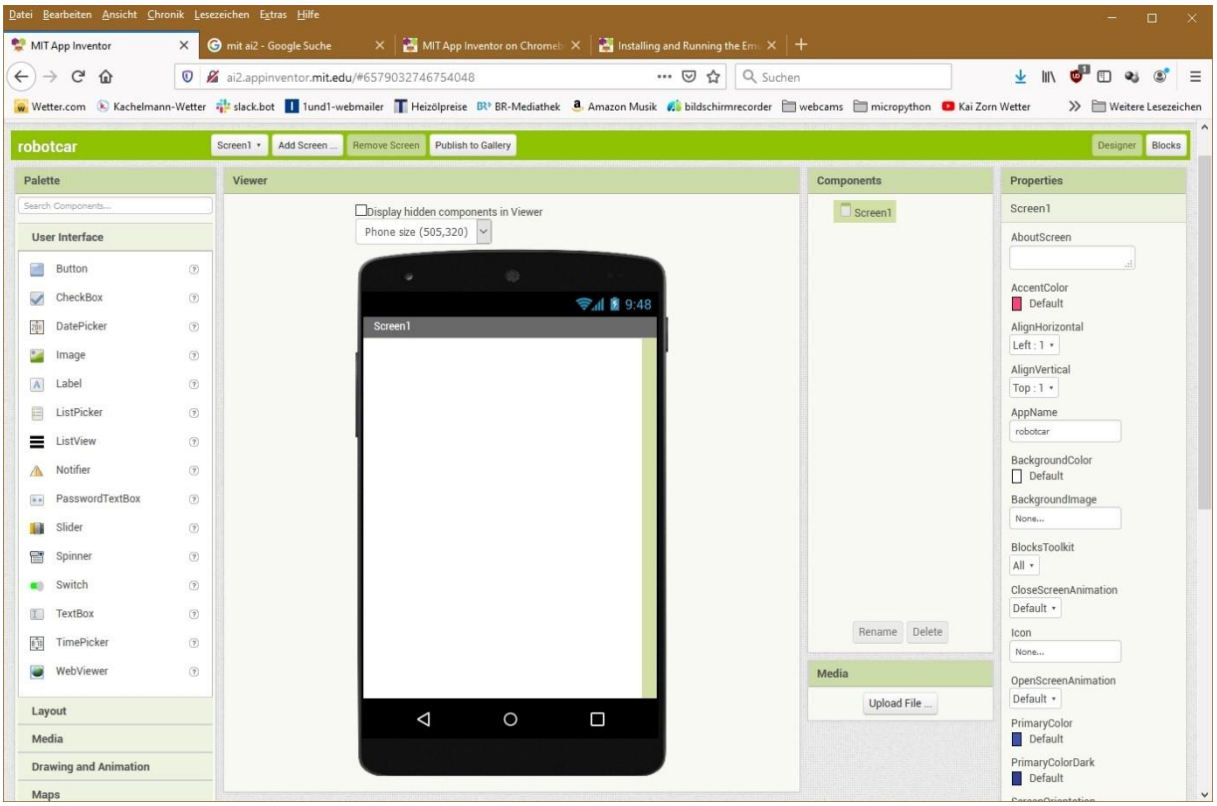

Arbeiten mit dem MIT App Inventor 2

Besorgen und Installation der Software

Die Hardware ist im Prinzip komplett und soll über eine Handy-App gesteuert werden. Was Sie jetzt und hier brauchen, sind demnach nur zwei Dinge, Ihren Projektaufbau und Ihr Android-Smartphone. Na ja, und dann ist da noch das bisschen Software. Um die freie Software vom MIT (Massachusetts Institute of Technology), das auch Lizenzinhaber ist, zu nutzen, gibt es drei Wege. Zwei davon führen über Softwareteile, die auf dem PC installiert werden müssen. Ich habe die dritte Möglichkeit gewählt, die auch von den Machern von App-Inventor 2 (AI2) vordringlich empfohlen wird. Auf dem PC kommt dazu nur ein Browser zum Einsatz und auf dem Handy die App **MIT AI2 Companion**. Auf dem PC tut's ein Browser wie **Firefox** oder **Chrome**. Mit Internet Explorer funktioniert es definitiv nicht, aber der ist eh tot. Ob Edge einsetzbar ist, habe ich nicht getestet, wozu auch? Den MIT AI2 Companion können Sie über den Google Playstore installieren. **Damit die hier beschriebene Vorgehensweise korrekt arbeitet, muss das Handy in einem lokalen Funknetz eingeloggt sein, nicht über das Internet**. Ein Zugriff über das Internet kann nicht funktionieren, weil zwar vom Handy aus beliebige Ziele erreichbar sind, umgekehrt das Handy aber vom Internet aus nicht direkt angesprochen werden kann. Es ist im lokalen Netz des Providers gekapselt und deshalb von außen nicht sichtbar. Das heißt, der nötige Zugriff des Browsers auf das Handy aus dem Internet ist nicht möglich, weil keine öffentliche IP und kein Routing existieren.

Das grundsätzliche Werkzeug steht nun bereit. Es gibt aber ein kleines Handicap, das Handy kann zwar perfekt den TCP-Dialekt, versteht aber erst einmal kein Wort UDP. Das ist für die Standardanwendungen auch nicht erforderlich, für unser Projekt

aber umso mehr. Das UDP-Protokoll ist schlank, schnell und unkompliziert, für MicroPython-Projekte somit bestens geeignet, vor allem wenn es um Steuerungsaufgaben geht. Also verpassen wir dem Handy via AI2 einen kleinen Sprachkurs. Den habe ich auf Ullis Roboterseite gefunden. Hier ist die Downloadadresse:

https://ullisroboterseite.de/android-AI2-UDP/UrsAI2UDP.zip

Entpacken Sie das Archiv zunächst in ein beliebiges Verzeichnis. Wechseln Sie dort hin und überzeugen Sie sich, dass die Datei **de.ullisroboterseite.ursai2udp.aix** aufgelistet ist. Sie enthält die Erweiterungen, die wir zur Erzeugung unserer Steuer-App in Kürze brauchen werden.

Zusammen mit dem Appinventor 2 bietet das MIT ein Tutorial und diverse Beispielprojekte in englischer Sprache an. Aus diesem Grund erscheint dieser Blogbeitrag auch nur in Deutsch.

App Inventor 2 erstmals starten

Öffnen Sie jetzt Chrome oder Firefox, falls noch kein Browser läuft und geben Sie folgende URL ein.

http://ai2.appinventor.mit.edu

Die Anmeldung bei Appinventor ist nur über ein **Googlekonto** möglich. Haben Sie bereits eines, dann klicken Sie auf **Weiter**, sonst auf **Konto erstellen**.

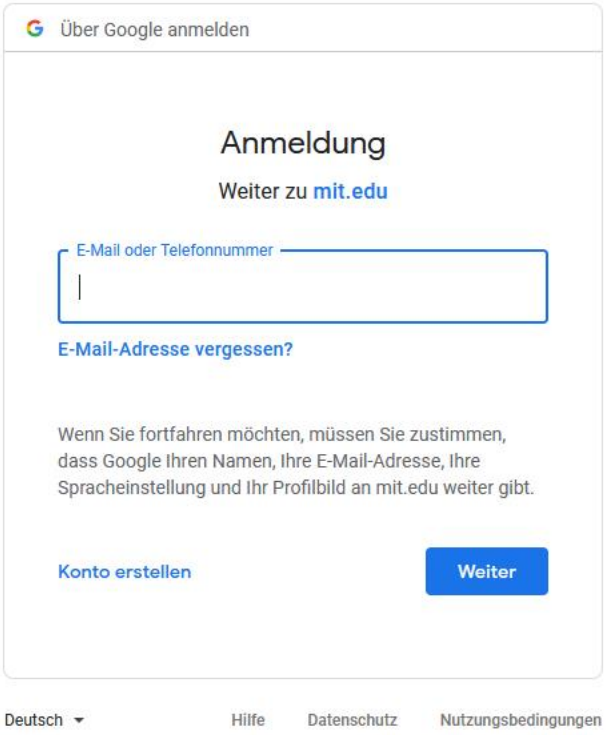

Abbildung: 1 Anmeldung über Google-Konto

Das Konto kann auch mit gefakten Daten angelegt werden. Das gilt für die erste und auch für die folgende Seite.

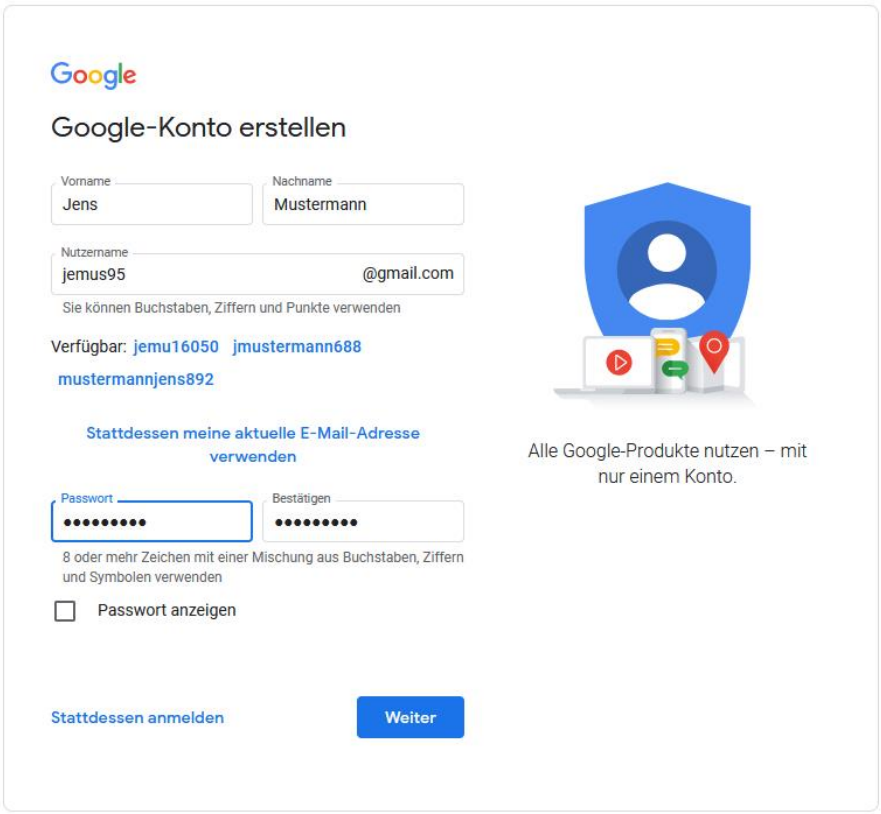

Abbildung: 2 Konto erstellen

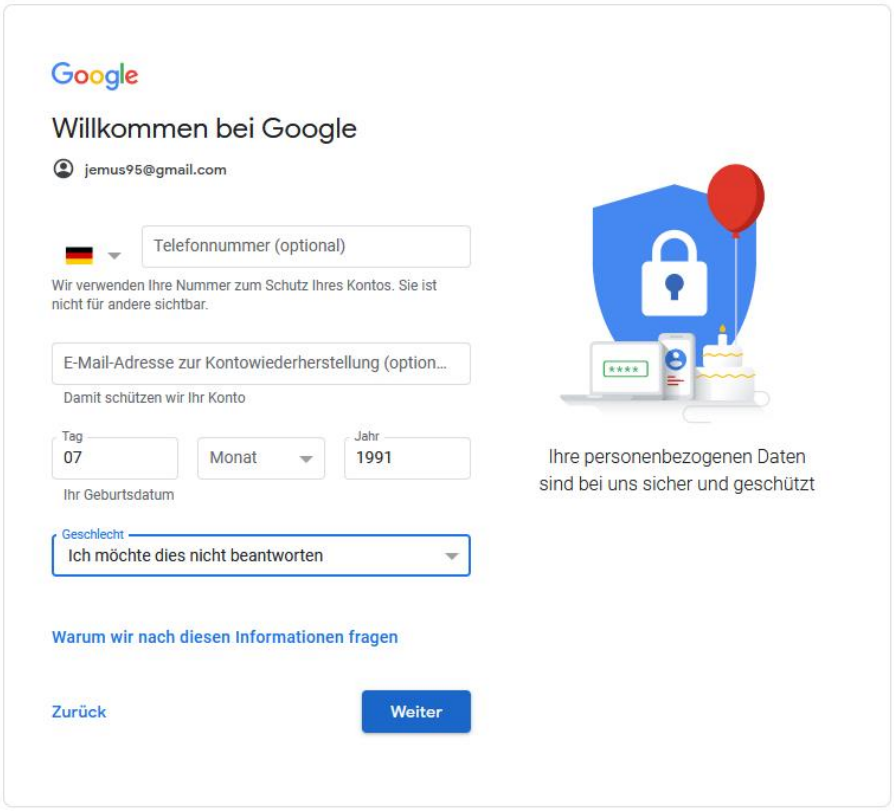

Abbildung: 3 Kontoerstellung abschließen

Nach den üblichen Einstellungen zur Privatsphäre landen Sie auf dieser Seite.

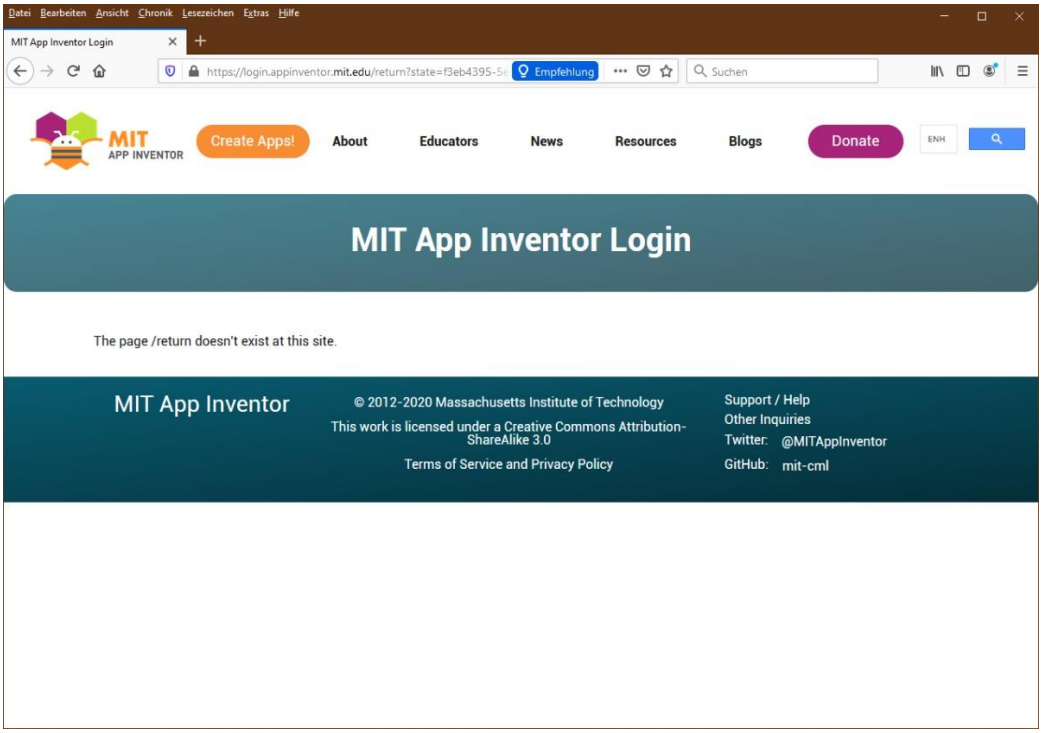

Abbildung: 4 Nach der Kontoerstellung

Der MIT APP INVENTOR -Link führt Sie zurück zum Login. Melden Sie sich mit Ihrem Googlekonto an und lassen Sie sich nicht von dem Riesenangebot an Tutorials und Beispielen verführen, sondern starten Sie einfach ein neues, leeres Projekt, Sie wollen doch Ihr Projekt mit einer eigenen App steuern, oder? Natürlich wollen Sie das!

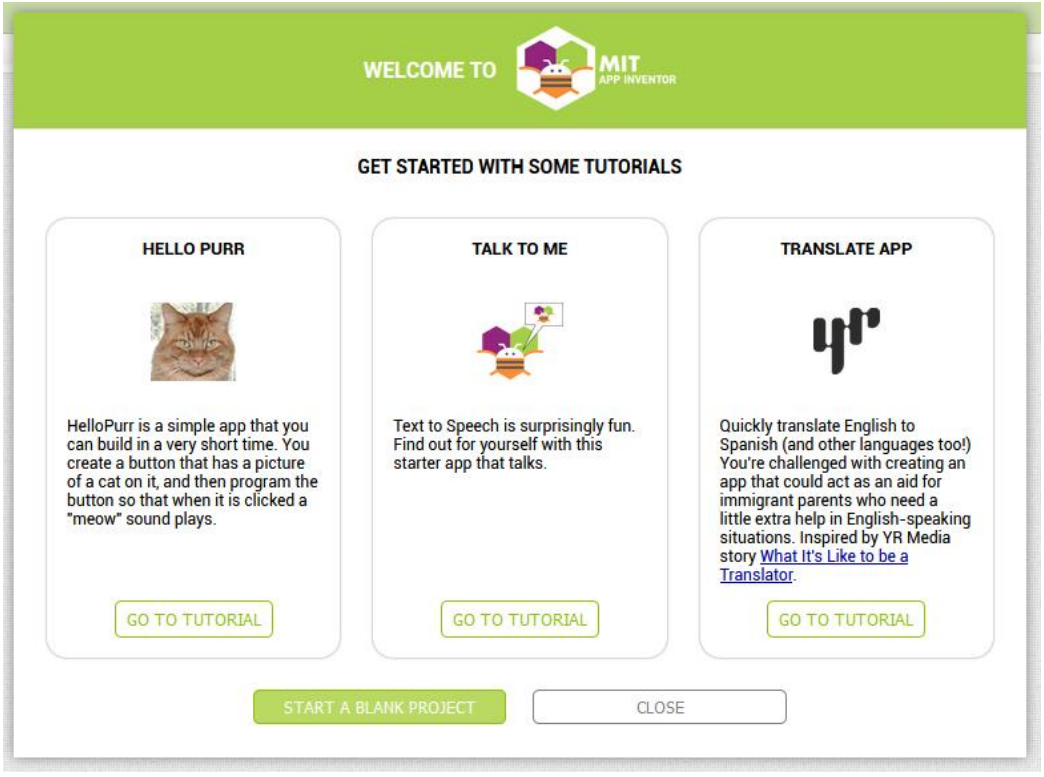

Abbildung: 5 Willkommen bei MIT App Inventor 2

Ich weiß aber auch, dass Sie später begierig alles aufsaugen werden, was AI2 Ihnen bietet - mir ging es jedenfalls so, das Angebot an Möglichkeiten ist einfach gigantisch.

Mit einem Fenster wie diesem starten wir in unser Projekt.

Abbildung: 6 NewProject

Einrichtung der Bildschirmobjekte

Werfen wir einen Blick auf die Seite auf der das Layout der App entworfen wird, den **Designer**. In dem Abschnitt **Palette** finden Sie die Zutaten. Der mittlere Bereich, der **Viewer**, ist Ihre Arbeitsfläche. Rechts davon in der Sektion **Components** wird die Hierarchie der verwendeten Komponenten dargestellt und rechts daneben, in **Properties**, die Eigenschaften der gerade ausgewählten Komponente. Das alles sehen Sie, wenn sie in der **grünen Zeile** über diesen Bereichen rechts außen **Designer** ausgewählt haben.

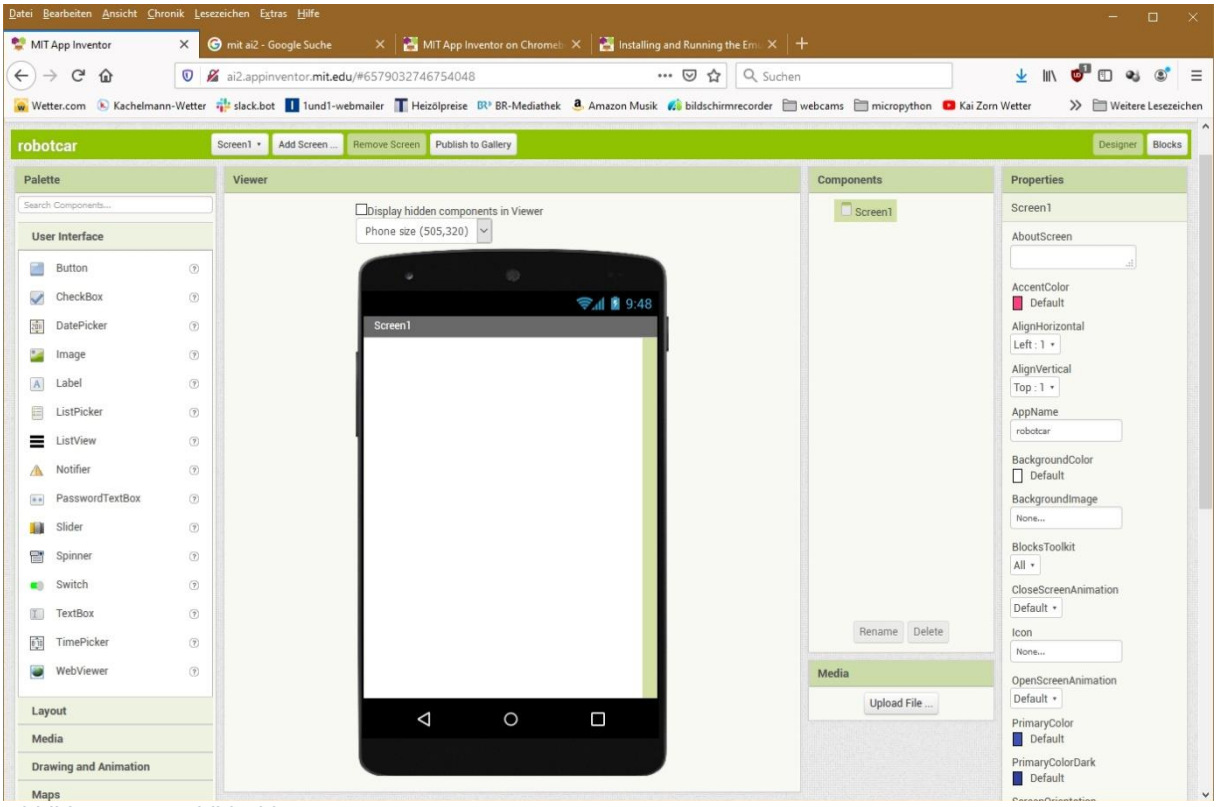

Abbildung: 7 Startbildschirm

Die Screenkomponente ist bereits aktiviert. Sie hat hier den Titel ROBOTCAR STEUERUNG. Scrollen Sie in Properties nach unten, bei **Title** können Sie Ihren Text für den Titel eingeben.

Abbildung: 8 Titelvergabe

Mit Arrangements aus dem Bereich **Layout** im **Palette**-Fenster können wir den Bildschirm (Screen) gliedern. In einem **HorizontalArrangement** werden die Elemente nebeneinander liegen, im **VerticalArrangement** übereinander. So gesehen ist der Screen auch ein vertikales Arrangement.

Über dem Ordner **Layout** liegt der Bereich **User Interface**. Darin finden wir die Elemente für Interaktionen wie Buttons, Labels, Eingabefelder und so weiter.

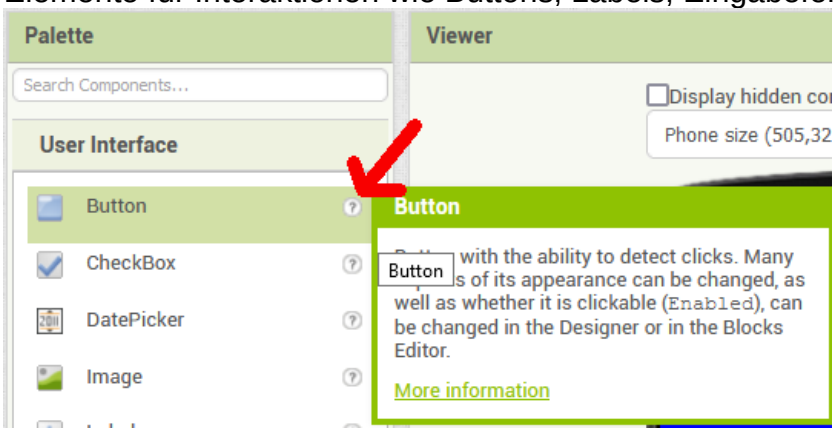

Abbildung: 9 Hilfe aufrufen

Klickt man auf das kleine Fragezeichen, erscheint ein Infofenster mit einem Link zu einer Dokumentationsseite. In Englisch sind dort weitere Erklärungen zum ausgewählten Element zu finden.

Ein erstes einfaches Beispiel

Sollen ein Schalter und ein Textfenster auf dem Screen erscheinen, holen wir aus dem Ordner **User Interface** einen **Switch** und ein **Label** und ziehen sie die beiden auf die Handyfläche. Was, das Label will nicht neben dem Switch wohnen? Geben Sie den beiden halt ein ordentliches Zuhause, holen Sie sich aus dem Ordner **Layout** ein **HorizontalArrangement** und setzen Sie es über den Schalter. Jetzt können Sie die Elemente Switch1 und Label1 hineinsetzen und schauen Sie, jetzt nehmen sie auch friedlich neben einander Platz.

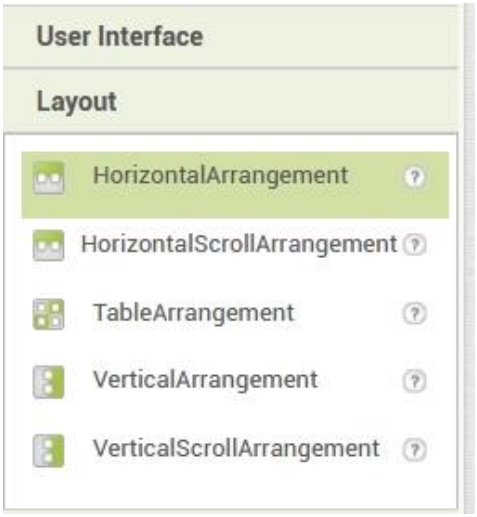

Abbildung: 10 HorizontalArrangement

Abbildung: 11 Erste Zeile

Verteilen wir nun die Eigenschaften! Wir markieren in **Components** den Eintrag **HorizontalArrangement1**. In **Properties** klicken wir auf das Feld von **Width**, **Fill Parent** und **OK**.

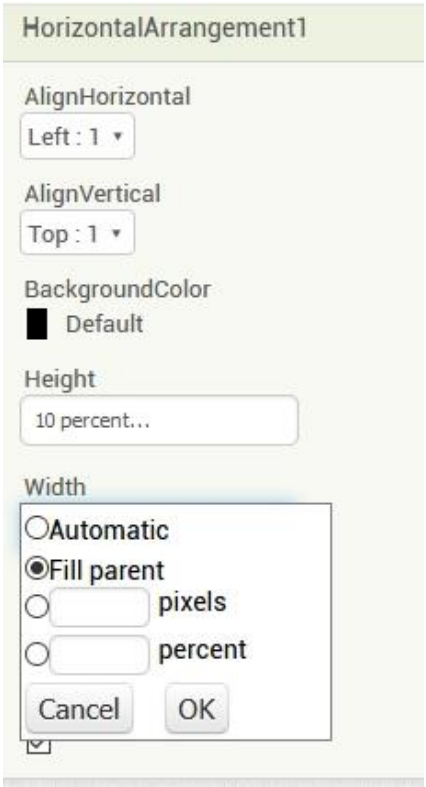

Abbildung: 12 Einige Eigenschaften von HorizontalArrangement

In **Components** markieren wir **Switch1**. Linksklick auf **Rename,** dann geben wir den neuen Namen **MainSwitch** ein und ändern die Eigenschaften wie folgt.

Abbildung: 13 Switch-Eigenschaften

Wir benennen **Label1** in **MyIP** um und stellen die Eigenschaften auf folgende Werte ein.

Abbildung: 14 myIP-Properties

Zwischen **MainSwitch** und **MyIP** fügen wir ein weiteres Label mit dem Namen **nameMyIP** und dem Text **Meine IP** ein.

Das nächste Arrangement heißt **ConfigIP**, ist vertikal, der Hintergrund gelb, die Höhe 20% des Displays und es nimmt die gesamte Bildschirmbreite ein. Ich denke, Sie kriegen das hin!?

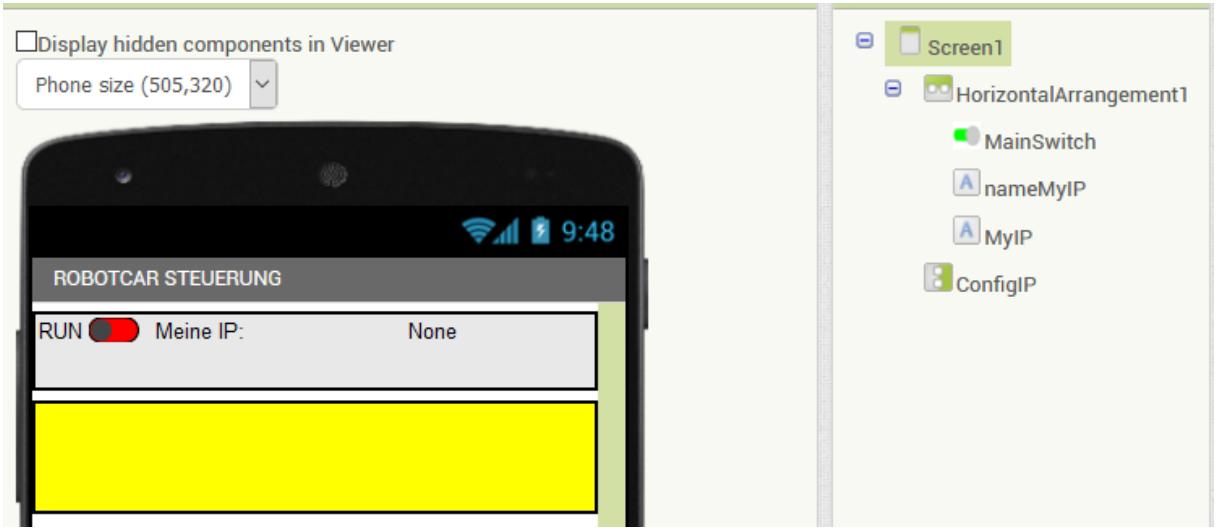

Abbildung: 15 Das vertikale Arrangement

Jetzt zur Innenausstattung des Arrangements. Ein **Spinner** ist eine Dropdownliste, aus der Einträge durch Touch ausgewählt werden können. Die Einträge werden durch eine Liste festgelegt. Die Begriffe sind durch Kommas getrennt. Beim Bildschirmaufbau ist der erste Begriff bereits markiert. Eine **Textbox** ist ein Eingabefeld, dessen Inhalt beim Verlassen übernommen wird.

Wir legen eine Liste von IP-Adressen fest, die mögliche Ziele des Projekts darstellen, WLAN-Accesspoint, ESP32-Accesspoint und PC zu Testzwecken. Die Liste sieht aus wie folgt: Klar, dass Sie die IP-Adressen an Ihre Netzumgebung anpassen müssen.

Default, 192.168.1.1, 192.168.1.8, 192.168.1.100

Die ausgewählte Adresse soll zur Laufzeit in der Textbox dargestellt werden, um die Auswahl mitzuteilen und /oder um Änderungen zu ermöglichen.

Die Eigenschaften des Spinners sind hier dargestellt.

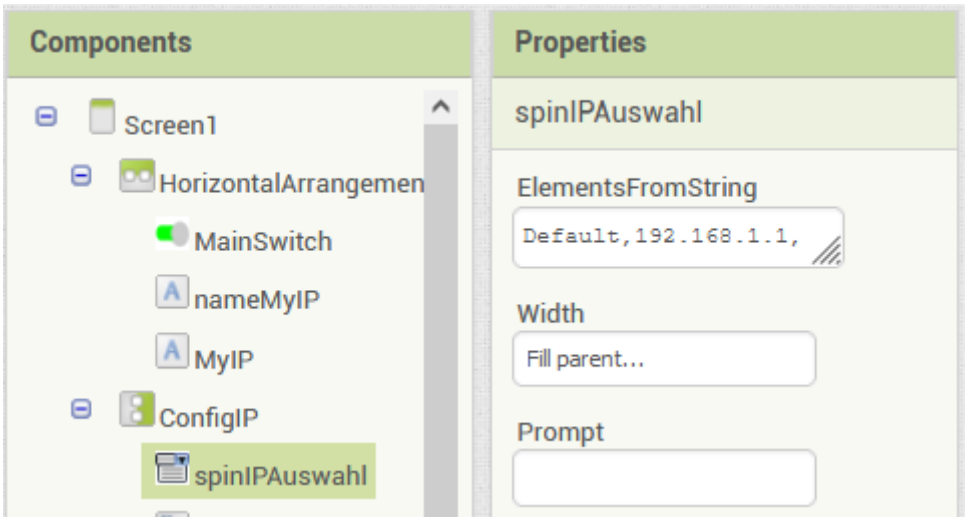

Abbildung: 16: Spinner Eigenschaften

Finden Sie die Eigenschaften der Textbox **txtFreeIP** heraus? Die Schriftgröße verrate ich Ihnen: 18.

Abbildung: 17 Textbox hinzugefügt

Haben Sie alles? BackgroundColor: Orange, FontSize: 18, Width: Fill Parent, Text: default, TextAlignment: center 1, TextColor: Blue.

Zum Abschluss ergänzen wir noch ein Eingabefeld mit dem Namen **txtInput**. Die Eigenschaften können Sie nach Ihrem Geschmack einstellen.

Unsichtbare Objekte

Es gibt auch eine Reihe von unsichtbaren Elementen. Eines davon ist das Accelerometer aus dem Palette-Ordner **Sensors**. Wie die anderen Elemente werden diese Komponenten auch vom Palette-Ordner auf die Handyfläche gezogen. Allerdings werden sie nicht dort, sondern unterhalb des Handybildes abgelegt.

Wissen Sie noch, in welchem Ordner sich die Datei **de.ullisroboterseite.ursai2udp.aix** befindet? Diese Datei brauchen wir als Nächstes. Sie liefert uns zwei weitere unsichtbare Elemente, einen UDP-Sender und einen Empfänger. Wir können damit den Sprachumfang von AI2 erweitern. Rollen wir in **Palette** ganz nach unten, klicken dort auf **Extension** und dann auf **Import extension**.

Abbildung: 18 Spracherweiterung für UDP installieren

Im folgenden Fenster navigieren wir nach Klick auf **Durchsuchen** zur aix-Datei.

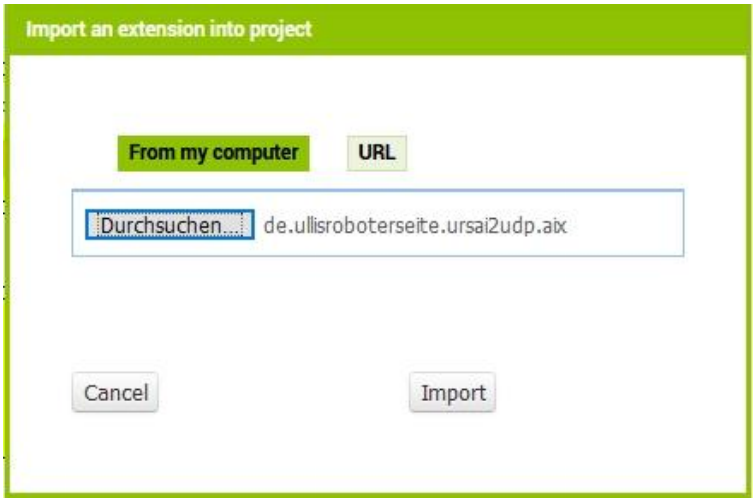

Abbildung: 19 Seite gefunden

Nach Klick auf **Import** werden nach ein paar Sekunden zwei Einträge angezeigt. Ziehen Sie **UDPXmitter** auf die Handyfläche. Jetzt beherrscht Ihr Handy auch den UDP-Dialekt und kann Datagramme an den Server auf unserem ESP32/ESP8266 senden. Wenn Sie dieses Feature auch in anderen Projekten nutzen möchten, müssen Sie den Importvorgang jedes Mal wiederholen, nachdem das neue Projekt angelegt wurde.

Verbindung mit dem Handy

Nun wird es Zeit, uns mit dem Handy zu verbinden, um unseren Entwurf real zu bewundern. Die AI-Companion-App ist auf dem Handy installiert? Dann los!

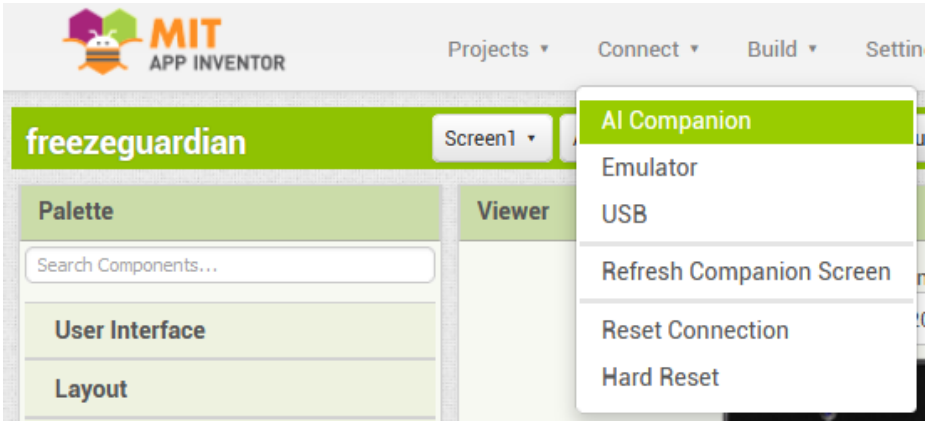

Abbildung: 20 Verbindung zum Handy aufbauen

Über das **Connect**-Menü rufen wir den **AI Companion** auf und landen in diesem Fenster mit QR-Code. Auf dem Handy starten wir die App **MIT AI2 Companion** und tippen auf **scan QR code**.

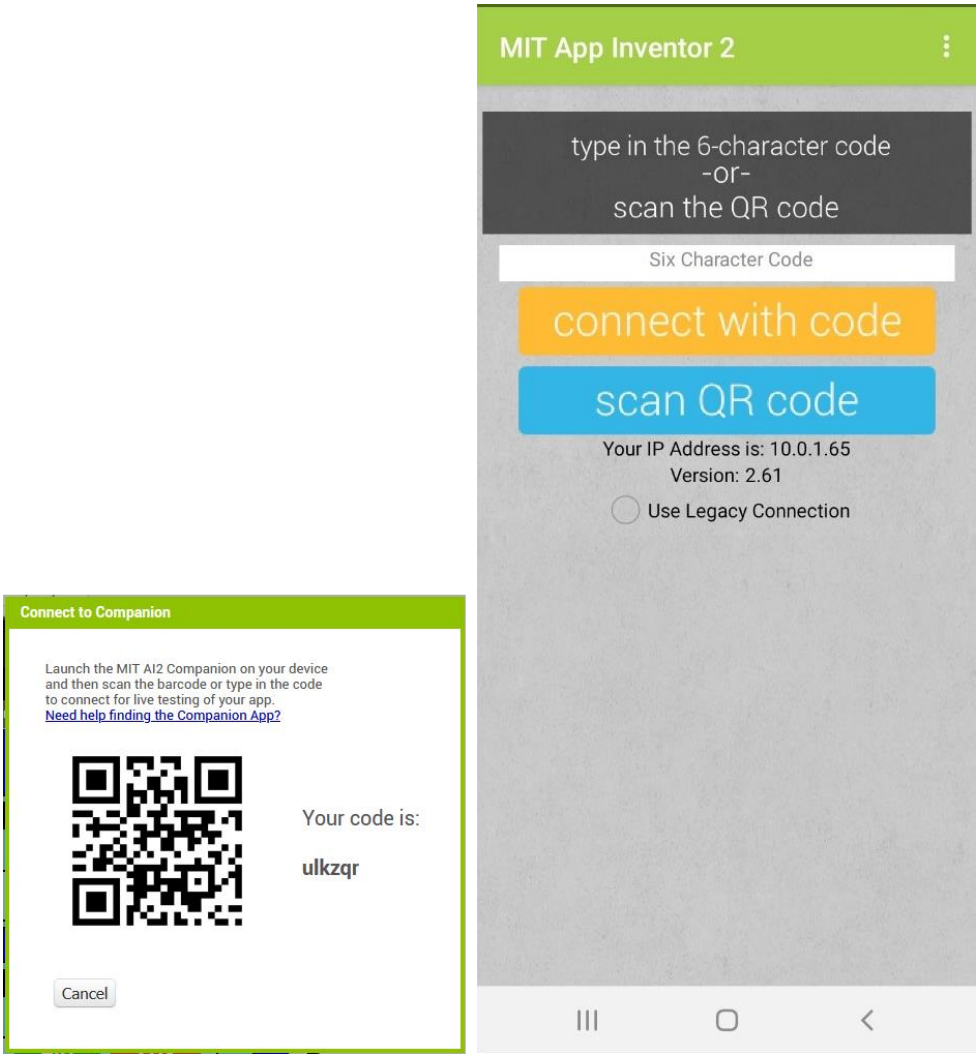

Abbildung: 21 Verbindung herstellen – links PC – rechts Handy

Der Progress-Bar im Browser zeigt den Fortschritt an.

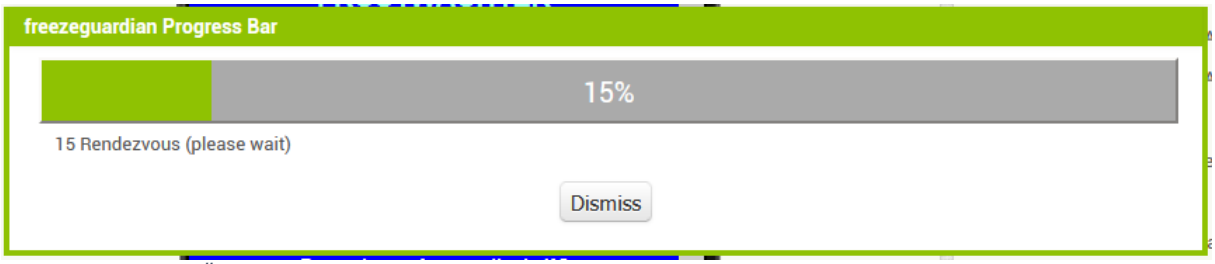

Abbildung: 22 Die Übertragung hat begonnen

Nach ein paar Sekunden erscheint unser Entwurf im Handydisplay. Alles, was jetzt am PC geändert wird, erscheint sofort auch auf dem Handy, Farben, Größenänderungen, Beschriftungen, …

Aber noch ist unsere App völlig handlungsunfähig. Es fehlt die Programmierung des Verhaltens bei eintretenden Ereignissen. Das finden wir alles, wenn wir den Designer verlassen, in der Abteilung **Blocks**, in die wir jetzt wechseln. Das Fenster **Palette** wird gegen **Blocks** ausgetauscht, die Fenster **Components** und **Properties** werden ausgeblendet. Im Viewer werden zunächst nur die Symbole für die Zwischenablage (Rucksack), den Mülleimer, Zoomen sowie Warnungen und Fehler angezeigt.

Jetzt stapeln wir Bausteine und erwecken damit die App zum Leben. Die Bausteine kommen Ihnen sicher bekannt vor, falls Sie bereits mit **Scratch** Sachen für den Arduino programmiert haben. Die Elemente, die im Designer platziert wurden, sind die Objekte, jetzt fügen wir die Eventhandler hinzu. Richtig, wir praktizieren OOP, Objekt-Orientierte-Programmierung.

Beginnen wir, wie bei jedem anderen Programm mit der Deklaration einiger Variablen. Die Bausteine werden aus dem Ordner **Variables** gezogen, Zahlen kommen aus **Math** und das Textelement aus **Text**.

Abbildung: 23 Angebote für Variablen

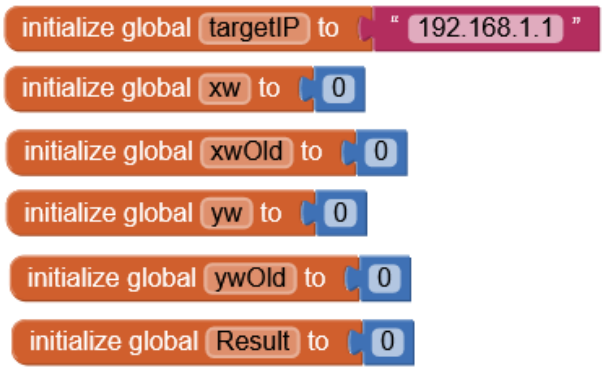

Abbildung: 24 Declaration der Variablen

Beim Aufbau der Oberfläche sollen die Felder mit den Startwerten initialisiert werden. Wir holen uns aus dem Ordner **Screen1** eine **when Screen1.initialize**- Klammer und füllen diese mit den Bausteinen zum Belegen der Eigenschaften aus dem Ordner des jeweiligen Elements. Die folgende Abbildung zeigt das Setzen der Ziel-IP-Adresse unseres ESP32.

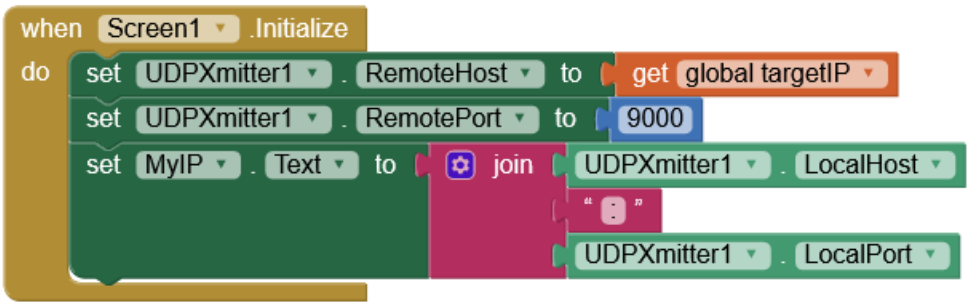

Abbildung: 25 Initialisierung von Variablen und Elementfeldern

Der Objektname steht stets als Erstes hinter dem Aktionsbezeichner (hier: **set**). Nach dem Punkt folgt die Eigenschaft und nach dem **to** werden die Eigenschaftswerte eingeklinkt. Es bleibt nur das haften, was semantisch sinnvoll ist. **get** ruft den Wert einer Variablen ab. **join** aus dem Ordner **Text** verbindet Strings. Nach dem Anklicken des blauen Radsymbols können durch Drag and Drop weitere Synapsen hinzugefügt werden.

Funktionen heißen im Appinventor-Kontext Procedures. Wir erstellen eine Prozedur, welche aus einem Eingabestring eine abgerundete, nicht negative Zahl macht. Dazu benutzen wir die lokale Variable temp. Das Ergebnis weisen wir der globalen Variablen Result zu.

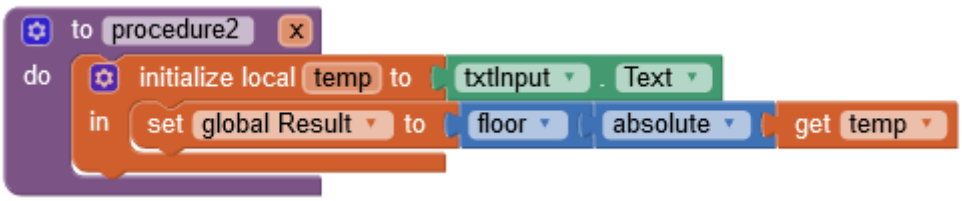

Abbildung: 26 Prozedurdefinition

Schließlich berechnen wir daraus die Quadratwurzel. Beim Einschalten des Schalters soll die Wurzel berechnet und im gleichen Textfeld angezeigt werden. Beim Ausschalten soll das Textfeld für die neue Eingabe gelöscht werden.

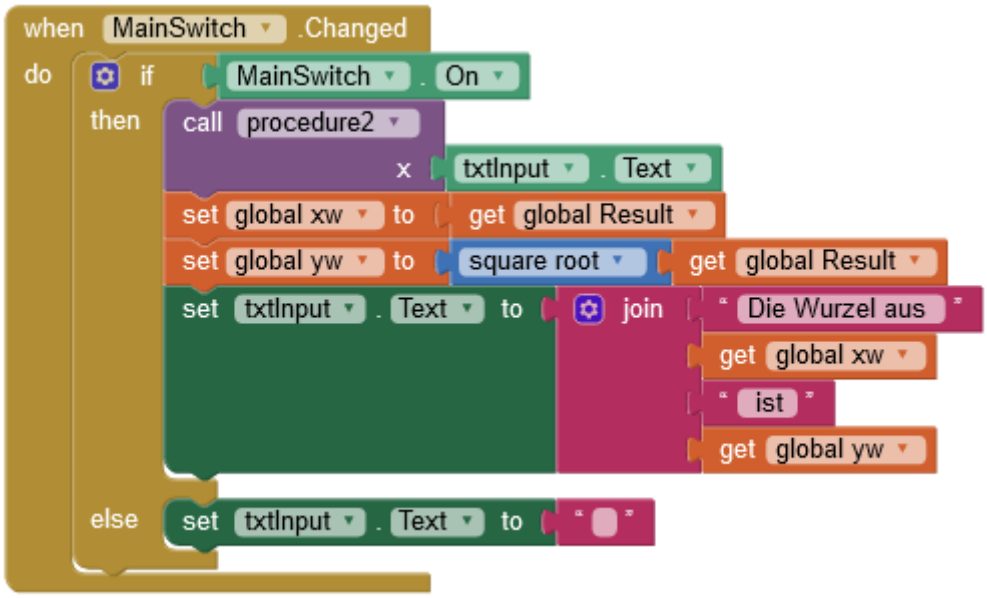

Abbildung: 27 Behandlung der Schalt-Events

Wir können feststellen, dass Zahlen implizit, also ohne unser spezielles Zutun, in Strings verwandelt werden und umgekehrt. Das ergibt sich automatisch aus dem Kontext.

Ich denke, das Beispiel erklärt sich ansonsten inzwischen selbst.

Für jede Klasse von Elementen gibt es eine kennzeichnende Färbung der Blocks. Innerhalb der Klasse kann man die Objekte austauschen. Linksklick auf das kleine Dreieck neben dem Objektnamen öffnet eine Liste mit weiteren Objekten derselben Klasse. Ein Objekt kann verschiedene Eigenschaften haben. Diese werden mit get abgerufen und mit set gesetzt. Diese Aktionen entsprechen dem Referenzieren und Belegen von Variablen in MicroPython, Lua, C und anderen Programmiersprachen. In der Objekt-Orientierten-Programmierung entsprechen diese Bausteine den Funktionen, welche Objekteigenschaften auslesen und setzen. Das direkte Ansprechen der Attribute, wie es in MicroPython möglich ist, sollte ja bekanntlich in der OOP vermieden werden, beziehungsweise ist ganz unmöglich.

Zum Abschluss noch ein Beispiel das den Einsatz des Spinners und des UDP-Senders zeigt. Auch dieses Beispiel erklärt sich wohl von selbst.

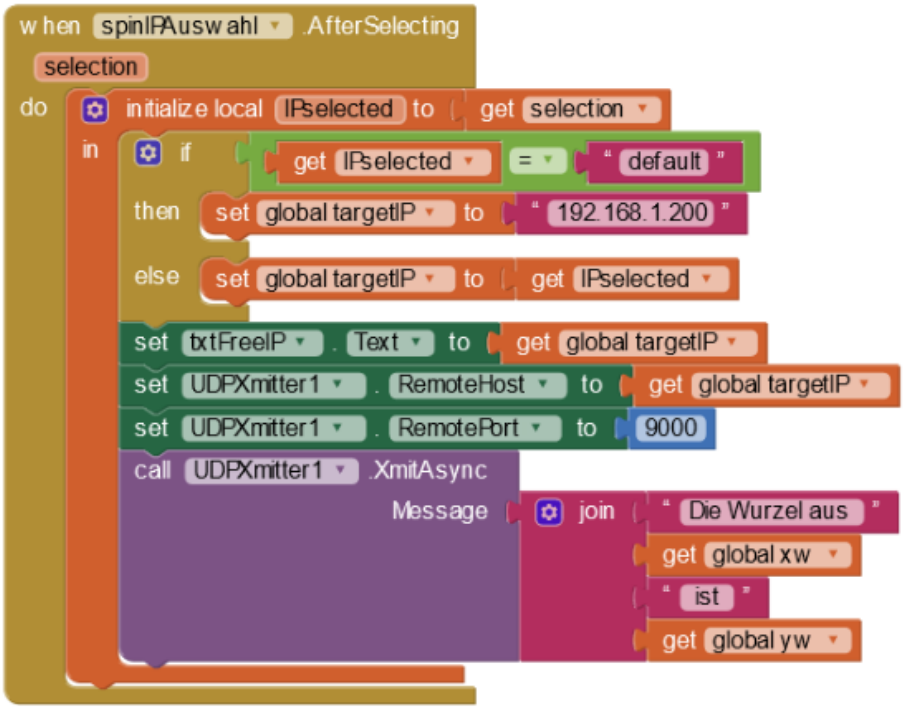

Abbildung: 28 Sendung über UDP

Den fertigen Entwurf können Sie hier als aia-Datei herunterladen und im Appinventor 2 importieren.

Wie kommt nun die App als Produktionssystem aufs Handy, denn bislang lief alles ja nur über die Appinventor 2-App und das auch nur, so lange, wie das Handy die Verbindung gehalten hat.

Wir starten den Download der apk-Datei durch Klick auf **Android App (.apk)** im Menü **Build**.

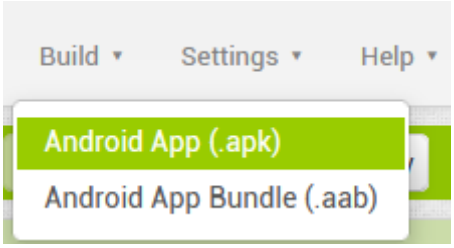

Abbildung: 29 Einleiten des Downloads

Nach dem Durchlaufen des Fortschrittsbalkens landen wir in dem Fenster der Abbildung 30.

Wenn auf dem Handy ein QR-Code-Leser installiert ist (zum Beispiel QR-Droid aus dem Google Playstore), dann scannen wir den Code und laden die Datei herunter. Sie landet im Ordner Downloads in den eigenen Dateien. Alternativ können wir über den linken Link die Datei auf den PC herunterladen, um sie dann anschließend, zum Beispiel über Bluetooth, zum Handy zu übertragen.

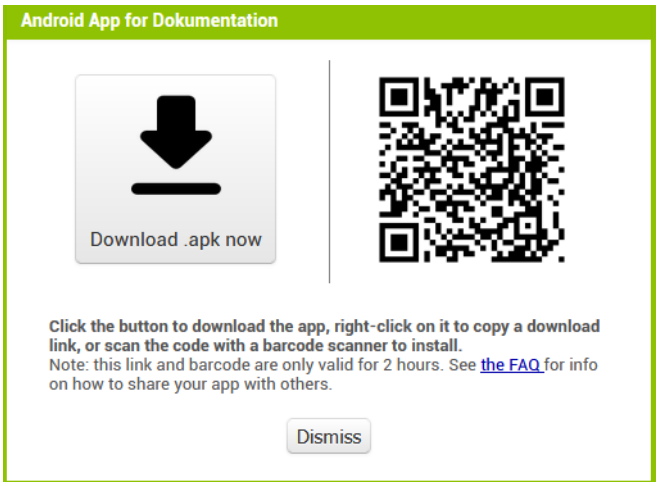

Abbildung: 30 Downloadlinks

Download auf das Handy

Ich zeige hier zuerst den Download via QR-Code. Nach dem Erkennen des Codes durch QR-Droid werden wir zum Download aufgefordert ---> **Herunterladen**.

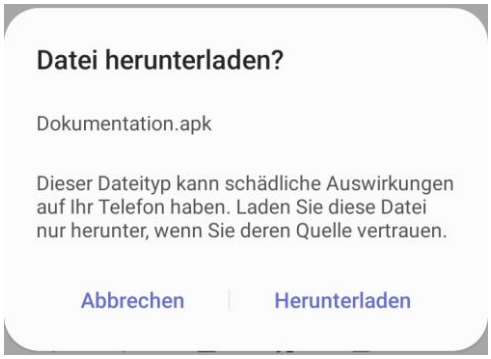

Abbildung: 31 Ins Handy herunterladen

Wir suchen die apk-Datei und tippen darauf. Nach Tipp auf **Installieren** steht die App nach einem kurzen Moment zur Verfügung und kann auch gleich gestartet werden.

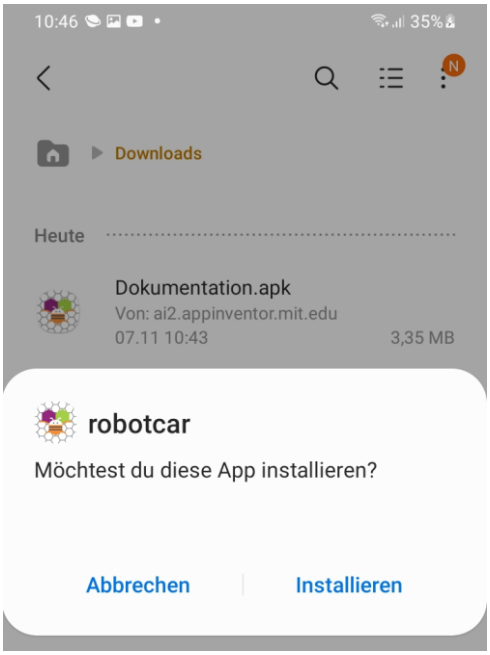

Abbildung: 32 Installation starten

Als aufmerksamer Leser haben Sie natürlich bemerkt, dass stellenweise der Name robotcar für die App auftaucht und andernorts Dokumentation. Nun die Erklärung ist einfach, ich habe die App zur Steuerung meines Robot Cars für die Erstellung dieser Dokumentation abgemagert, ein wenig verändert und umgetauft. Die Blogs zum Robot Car finden Sie übrigens in der folgenden Liste.

Teil1	deutsches PDF
	englisches PDF
Teil2	deutsches PDF
	englisches PDF
Teil3	deutsches PDF
	englisches PDF

Download auf den PC

Die zweite Option bietet nach der Übersetzung den Download auf Ihren PC an. Speichern Sie die apk-Datei in einem beliebigen Verzeichnis und transferieren Sie sie danach mit Bluetooth oder einer anderen Methode (e-Mail, WWW, …) auf Ihr Handy. **robotcar.apk** (apk = **A**ndroid **P**ac**K**age) landet vermutlich im Downloadordner und der liegt in 'Eigene Dateien'. Prüfen Sie zunächst, ob Setups aus diesem Ordner durchgeführt werden dürfen.

Einstellungen …

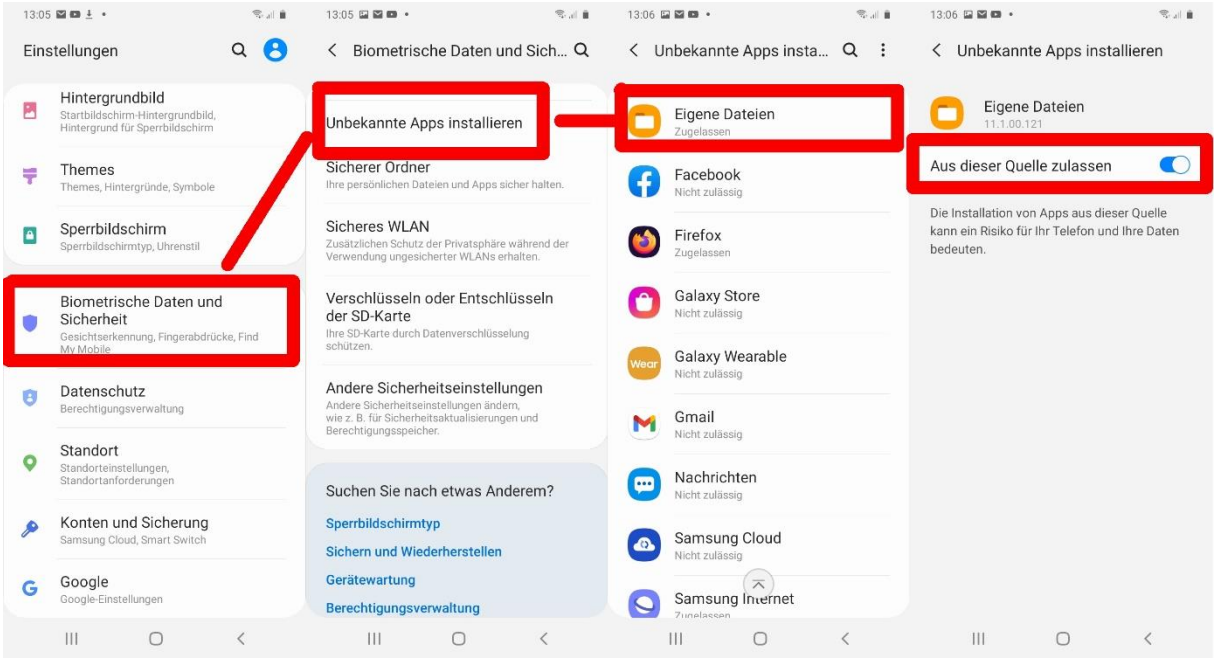

Abbildung: 33 Ändern der Sicherheitseinstellungen

Eigene Dateien …

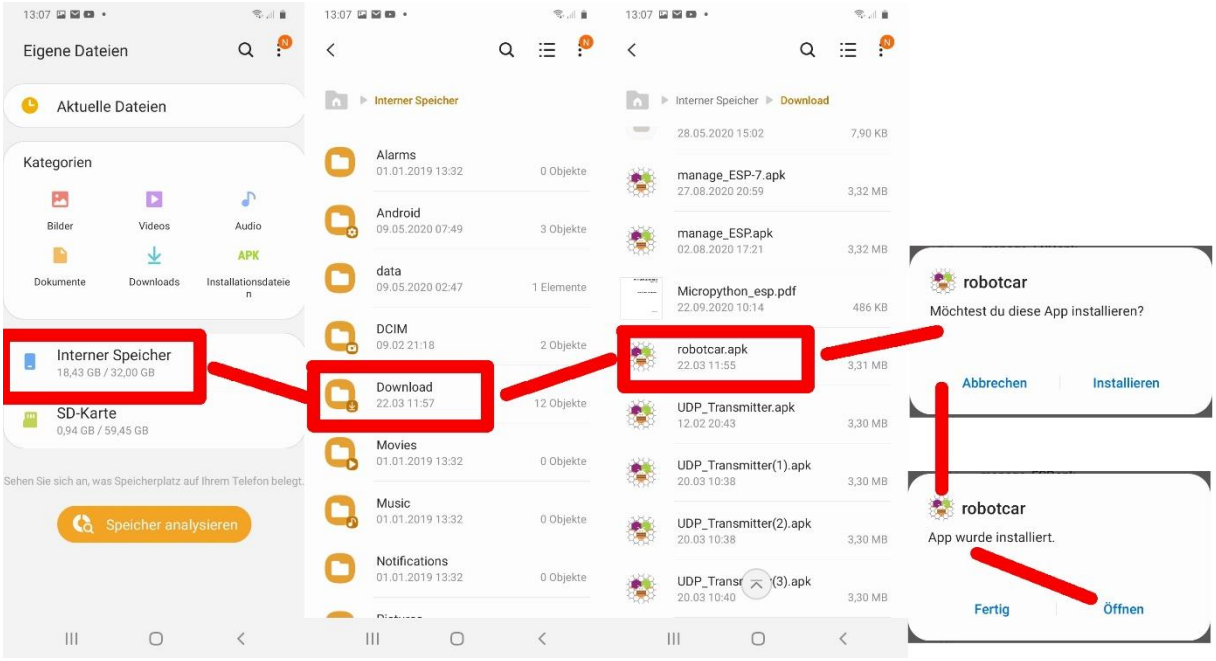

Abbildung: 34 Installation aus dem Ordner Eigene Dateien

Nach dem finalen Test der App sollten Sie die Erlaubnis, dass Apps aus dem Ordner 'Eigene Dateien' installiert werden dürfen, aus Sicherheitsgründen widerrufen.

Viel Spaß beim weiteren Entdecken der phantastischen Vielfalt der Möglichkeiten, die in Appinventor darauf warten, eingesetzt zu werden.