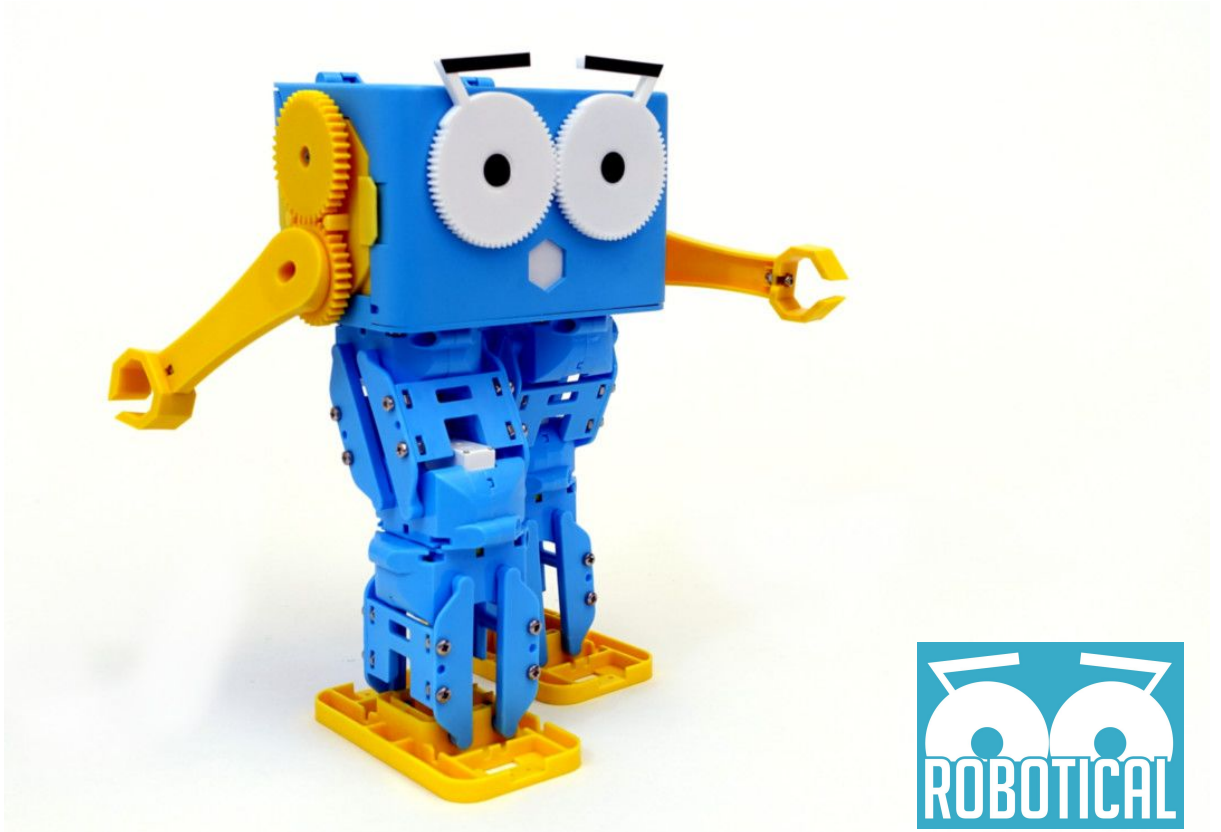


Lernen Sie mit Robotical & Generation Robots



MARTY ROBOTER EINFÜHRUNG & PROGRAMMIERUNG

Übersetzt von Génération Robots, 2018

Die Originalinhalte und Bilder wurden von Robotical erstellt, Quelle: <https://robotical.io/learn/>

Inhaltsverzeichnis

Die ersten Schritte mit deinem Marty	3
Verbinde deinen Marty mit dem WLAN und kalibriere ihn	3
Die wichtigsten Dinge zuerst	3
Marty App	4
Hilfe, ich habe ein Problem!	4
Was kommt als Nächstes?	5
Herzlichen Glückwunsch - du hast einen Marty!	7
Marty einschalten	7
Marty aufladen	7
Warnung bei niedrigem Akkustand	8
Ein- und Ausschalten von Motoren	8
Schutz bei Stürzen	8
Schutz vor Motorüberlastung	8
Schutz vor Zittern (es wird gemütlich)	9
Menschenähnliches Verhalten (es wird langweilig)	9
Benutze vorgefertigte Befehle	12
Installiere Python	14
MartyPy installieren	14
Eine Verbindung zu Marty über WLAN herstellen	17
Noch ein bisschen mehr machen	18
Weitere Beispiele...	18
Erste Schritte mit deiner Pi	19
Ein Pi Image installieren	20
Einen Pi selbst konfigurieren	20
Einen Raspberry Pi in Martys Kopf anschließen	20
Netzwerk Setup	22
SSH	22
Netzwerk Konfiguration	23
Die nächsten Schritte	23
Upgrades!	23
ROS: Das Robot Operating System	25
Okay, cool, aber was genau ist ROS?	25
Und wie funktioniert es?	25
Ich glaube, so langsam verstehe ich es...	26
Cool! Wo fange ich an?	27



GETTING STARTED WITH MARTY

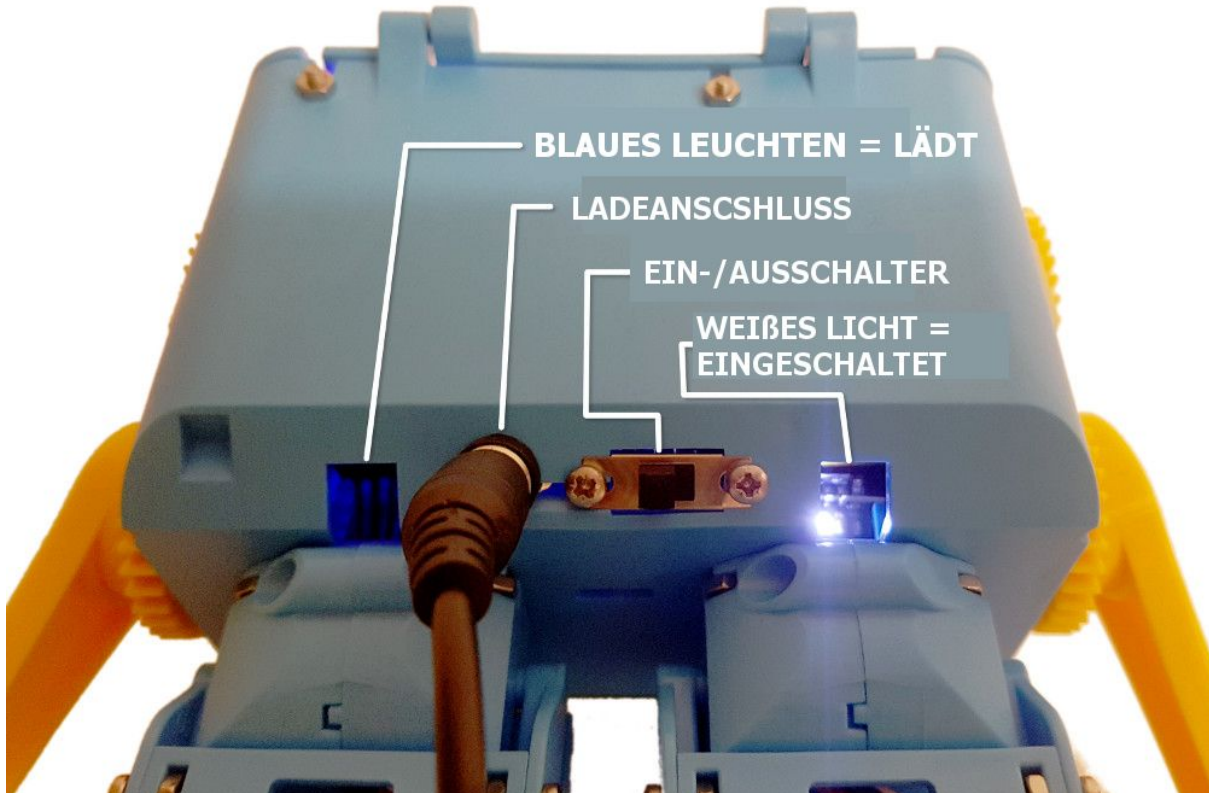
Die ersten Schritte mit deinem Marty

Verbinde deinen Marty mit dem WLAN und kalibriere ihn

Du hast deinen Roboter zusammengebaut - cool! Als nächstes musst du deinen Roboter mit dem WLAN verbinden und ihn kalibrieren. Folge einfach unserer Anleitung.

Die wichtigsten Dinge zuerst

Vergewissere dich, dass dein Marty eingeschaltet ist! Martys Ein- und Ausschalter befindet sich an seinem Rücken. Dort steckst du übrigens auch das Ladekabel ein:

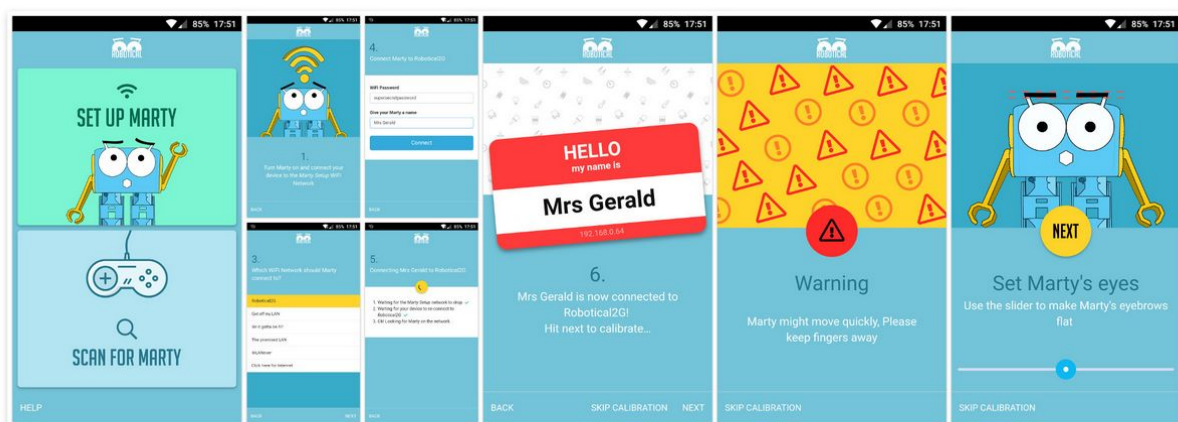


Marty App

Ein neuer Marty kann mit unserer App mit dem WLAN verbunden und kalibriert werden. Es dauert nur ein paar Minuten und wenn es einmal erledigt ist, kannst du die eingebaute Fernbedienung nutzen, um Martys Bewegungen zu kontrollieren und ihm sogar einen eigenen Namen geben.



Öffne die App, drücke auf das grüne "Set Up Marty" Bild und folge den Anweisungen auf dem Bildschirm.



Hilfe, ich habe ein Problem!

Hoffentlich hattest du überhaupt keine Schwierigkeiten, aber für den Fall der Fälle sind hier ein paar Hinweise:

- Die App findet meinen Marty nicht!
 - Stelle sicher, dass dein Marty mit dem WLAN verbunden ist. Wenn ein "Marty Setup" Netzwerk sichtbar ist, musst du zurück gehen und dein WLAN erneut einstellen.
 - Einige WLAN Repeater stellen nur ein anderes WLAN-Netzwerk mit einem ähnlichen Namen zu dem Hauptnetzwerk her. Marty und das Gerät, mit dem du ihn steuerst, müssen beide mit demselben Netzwerk verbunden sein!
 - Dein Netzwerk benutzt vielleicht andere IP-Adressen als üblich. Benutze unser Discovery Tool um deinen Marty zu finden (<http://docs.robotical.io/hardware/esp-socket-discovery/>).

- Mein Marty läuft nicht richtig!
 - Wenn du deinen Marty kalibrierst, achte darauf, dass er richtig steht (aufrecht, Arme an der Seite, Augenbrauen flach) bevor du deine Kalibrierung speicherst. Du musst deine Kalibrierung speichern, bevor du das Laufen ausprobierst.
 - Wenn Marty immer noch nicht richtig läuft, kontrolliere noch einmal, ob er korrekt zusammengebaut ist. Schauge, ob die jeweiligen Knöpfe die richtigen Gelenke bewegen - wenn nicht, hast du vielleicht die Motoren falsch angeschlossen. Das passiert schnell und ist auch schnell behoben - schau dir nochmal den Teil über die Elektronik in der Aufbauanleitung an und vergewissere dich, dass alles in den richtigen Anschlüssen steckt (beachte die Nummern).
 - Sind alle Teile der Beine richtig ausgerichtet? Schauge dir noch einmal die Grafik in der Anleitung an und überprüfe, ob dein Marty genauso aussieht.
- Mein Marty läuft, aber er ist etwas wackelig auf den Beinen...
 - Kalibriere deinen Marty nochmal neu und stelle seine Füße diesmal etwas mittiger.
- Mein Marty piept mich an!
 - Das klingt als hätte er bald keinen Akku mehr. Lade ihn auf!
- Martys Motoren machen gar nichts, wenn ich versuche, ihn zu kalibrieren!
 - Vergewissere dich, dass sie richtig angeschlossen sind - schau nochmal in den Teil über die Elektronik in der Anleitung
- Irgendwie funktioniert überhaupt nichts...
 - Besuche unsere Support-Seite und kontaktiere uns wenn nötig! (<https://robotical.io/support/>)

Was kommt als Nächstes?

Jetzt, da du mit dem Aufbau deines Marty fertig bist, gibt es eine Menge Arten, wie du mit dem Programmieren deines Roboters starten kannst! Keine Sorge, wir haben Anleitungen für komplette Anfänger und für erfahrene Experten.



Diese Anleitung behandelt Martys **Verhalten**, also was der Roboter wann macht und warum er es (wahrscheinlich) macht. Wenn du alles gelesen hast, wirst du einen guten Überblick über das gesamte System haben.



Scratch ist der einfachste Weg, deinen Marty zu programmieren, wenn du ein Anfänger bist. Es hat eine einfache, grafische Oberfläche und Elemente, die du kombinieren kannst, um ein Programm zu erstellen.

 **python™** **Beginne mit Python**
(Fortgeschritten)

Python ist ein eine vollwertige Programmiersprache, die du benutzen kannst, um Marty zu steuern und komplexere Programme zu entwickeln. Hier ist eine Einführung in unsere Python Library.

**Baue eine Raspberry Pi ein**
(Fortgeschritten)

Baue eine **Raspberry Pi** in Martys Kopf ein und du hast einen All-in-One Roboter! Das ermöglicht dir, Rick direkt über den **seriellen Port** zu steuern. Die Python Library *und* ROS unterstützen diesen Betriebsmodus.

 **ROS** **ROS und V-REP**
(Experte)

ROS (*Robot Operating System*) hat eine Menge großartiger Roboter-Tools, die sowie in der Industrie als auch in der akademischen Welt genutzt werden. Wir haben ROS Libraries für deinen Marty und die Möglichkeit, Marty im Coppelias Robotics V-REP Simulator darzustellen.

Einführung in Martys Verhalten



Herzlichen Glückwunsch - du hast einen Marty!

Du hast einen Roboterfreund fürs Leben gefunden!

Allerdings gibt es einige Dinge, die du über Marty wissen solltest, damit du dich gut um ihn kümmern kannst und ihr eine lange und glückliche Freundschaft habt.

Marty einschalten

Martys Einschalter ist an seinem Rücken. Wenn ein weißes Licht leuchtet, dann weißt du, dass er an ist. Wenn das Licht nicht leuchtet, kann es sein, dass der Akku leer ist oder nicht richtig angeschlossen ist.

Marty aufladen

Marty wird mit einem 9V-Klinkenstecker am Rücken aufgeladen. Du solltest ein spezielles Ladekabel mit deinem Marty-Set erhalten haben, mit dem du Marty von einem normalen USB-Anschluss aufladen kannst. Wenn Marty aufgeladen wird, siehst du ein blaues Leuchten, das aufhört, sobald der Akku vollständig geladen ist und das Ladegerät nicht mit Strom gespeist wird.

Marty kann während des Ladevorgangs benutzt werden - lass ihn aber nicht dauerhaft laden und gleichzeitig eingeschaltet sein. Behalte auch im Hinterkopf, dass dein USB-Ladegerät eventuell nicht schnell genug lädt, um starke Benutzung für längere Zeit zu ermöglichen - in diesem Fall ist es das Beste, den Roboter einfach auszuschalten und ein paar Stunden aufladen zu lassen.

ProTip

Das Ladegerät wird sich nach einer Weile aus Sicherheitsgründen automatisch abschalten. Wenn du also Marty für eine lange Zeit eingesteckt und eingeschaltet lässt, kann es passieren, dass nach etwa 8 Stunden der Ladevorgang aufhört. Dann wird Marty den Akku entladen bis er vollständig entladen ist. Du musst das Ladekabel ausstecken und dann wieder einstecken, um erneut aufzuladen.

Warnung bei niedrigem Akkustand

Wenn Marty's Akkustand niedrig ist, fängt er an zu piepsen. Du wirst es zuerst kurz hören, wenn Marty herum läuft. Es zeigt dir, dass dein Roboter anfängt, etwas müde zu sein und bald aufgeladen werden muss.

Wenn der Akkustand noch niedriger wird, piepst Marty dauerhaft. Nach etwa einer Minute gehen dann alle Motoren aus. Der Roboter reagiert trotzdem noch über WLAN und auch das Power-Licht leuchtet noch, aber er weigert sich, sich zu bewegen bis du den Akku aufgeladen hast.

Ein- und Ausschalten von Motoren

Wenn Marty eingeschaltet ist, kann jeder Motor individuell ein- und ausgeschaltet werden. Wenn ein Motor eingeschaltet ist, wird er Befehle befolgen, sich bewegen und Positionen halten. Wenn der Motor ausgeschaltet ist, kann er mit der Hand frei bewegt werden. Alle Betriebssysteme erlauben es dir, die Motoren ein- und auszuschalten.

Warnung

Beachte dass, je nachdem wie du Marty steuerst (ROS, Python, Scratch, ...), der Ausgangszustand der Motoren (eingeschaltet, ausgeschaltet) unterschiedlich sein kann. Dasselbe gilt für andere Schutzmaßnahmen und Sicherungen.

Schutz bei Stürzen

Marty stört es nicht, hinzufallen, aber möchte seine Gelenke nicht kaputt machen. Marty nutzt den eingebauten Beschleunigungsmesser, um feststellen, ob er gerade hinfällt. Wenn er merkt, dass er fällt, stellt er alle Motoren aus und hört sofort mit allem, was er macht auf.

Du kannst die Motoren wieder einschalten, damit Marty sich wieder bewegt.

Das kann übrigens auch passieren, wenn du Marty zu schnell hoch hebst!

Schutz vor Motorüberlastung

Marty's Motoren sind klein und können dementsprechend auch nur so viel Kraft erzeugen. Um sie also vor Schäden zu schützen, messen wir wie hoch die Belastung auf ihnen ist. Wenn diese zu hoch wird, schalten sich die Motoren aus. Entweder es ist zu viel Druck auf dem Motor oder eine dauerhafte Belastung würde den Motor überhitzen.

Der Schutz vor Motorüberlastung kann ein bisschen nervig sein, aber er ist wirklich nützlich, um Marty vor Schäden zu schützen. Du kannst diese Schutzfunktion zwar ausschalten, wenn du dir absolut sicher bist, was du tust, aber das können wir nicht empfehlen - einen Motor kaputt zu machen, ist nicht lustig!

Beachte: Es gibt *keine* Motorstromerkennung in den **Augenbrauen**, also bewege sie nur manuell, wenn der Motor ausgeschaltet ist!

Schutz vor Zittern (*es wird gemütlich*)

Wenn Marty aufhört, sich zu bewegen, messen wir die Spannung, die in seinen Motoren ist. Sollte noch ein bisschen Spannung zu messen sein, versucht Marty sein Gelenk zu bewegen und die Spannung zu entladen. Dadurch hören die Servomotoren auf zu zittern, wenn sie nicht ganz in einer "bequemen" Position zum Stehen kommen.

Das bedeutet, dass Marty sich noch ein bisschen auf eigene Faust bewegt, wenn ein Programm beendet ist. Dadurch endet er in einer Position, in der die Spannung im Motor reduziert ist. Das führt zu einer längeren Akkulaufzeit und glücklicheren Motoren. Yeah!

Menschenähnliches Verhalten (*es wird langweilig*)

Wenn Marty eingeschaltet wurde, aber für eine Weile keine Befehle erhält, wird ihm langweilig und er macht eine kleine Bewegung. Das kann zum Beispiel sein, dass er seine Arme hebt, mit dem Fuß aufstampft oder verärgert schaut.

Am Ende von jedem menschenähnlichen Verhalten bewegt Marty seine Augenbrauen, um den Akkustand zu zeigen. Je niedriger er ist, desto wütender schaut er!

Marty kann sich auch menschenähnlich verhalten, obwohl die Motoren ausgeschaltet sind. Sobald die Bewegung fertig ist, stellen sie sich aber sofort wieder aus. Motoren können mit allen Programmiersprachen (Scratch, Python, ROS, JS, ...) oder durch Hinlegen von Marty ausgeschaltet werden.

Beginne mit Scratch



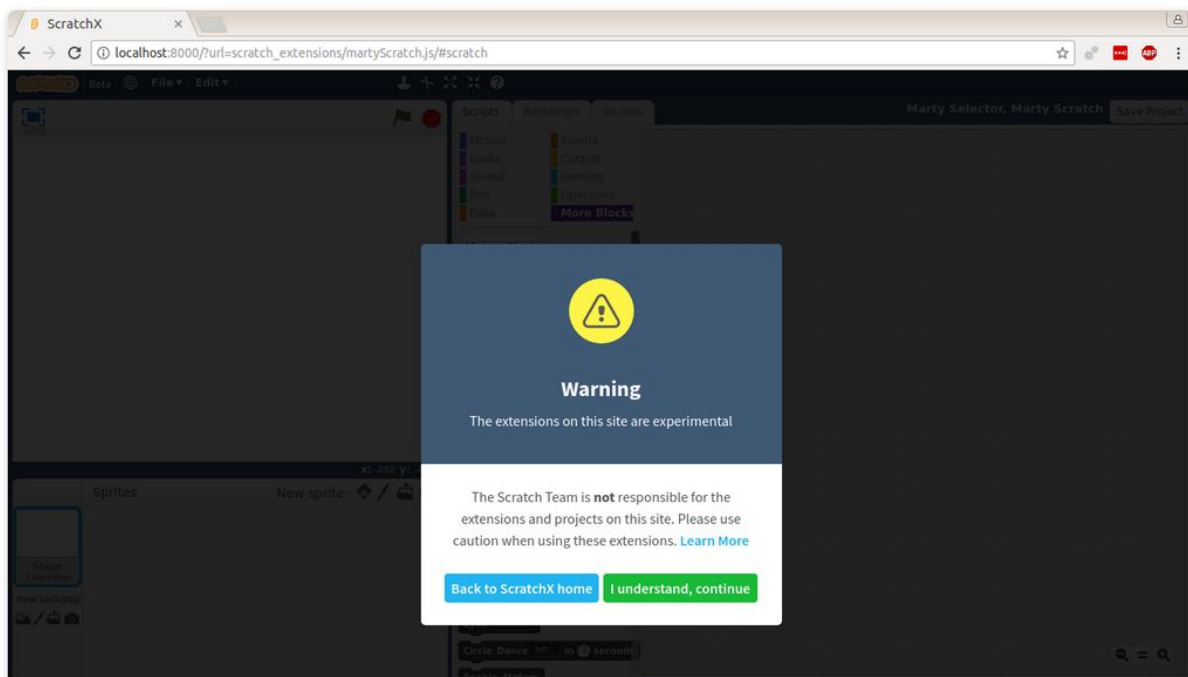
Egal, ob du noch nie vorher programmiert hast, ein Scratch Zauberer oder einfach ein erfahrener professioneller Entwickler bist, der Spaß haben möchte - es ist super einfach und lustig, Marty in Scratch zu programmieren.

Beachte: dein Marty muss mit dem WLAN verbunden sein, um per Scratch gesteuert zu werden.

Schalte zuerst deinen Marty an und öffne ScratchX:

www.scratchx.org

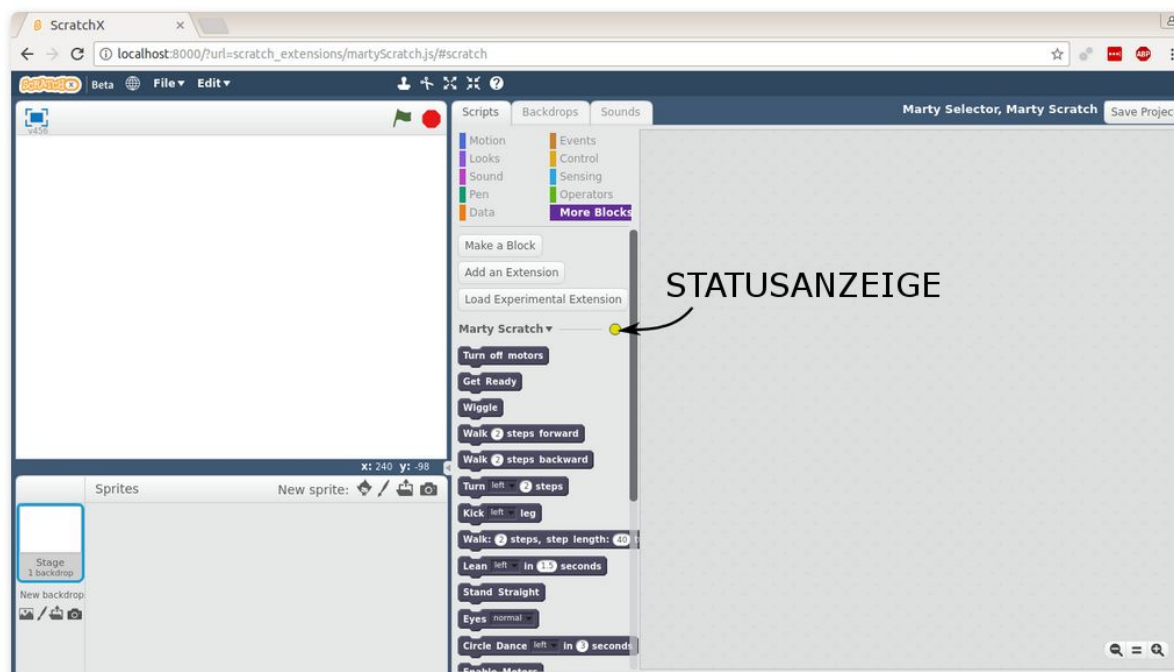
When Scratch lädt, siehst du folgende Warnung:



Die Warnung kommt, weil ScratchX noch in der Testphase ist. Du kannst einfach auf den grünen "I understand"-Button klicken, um weiterzumachen.

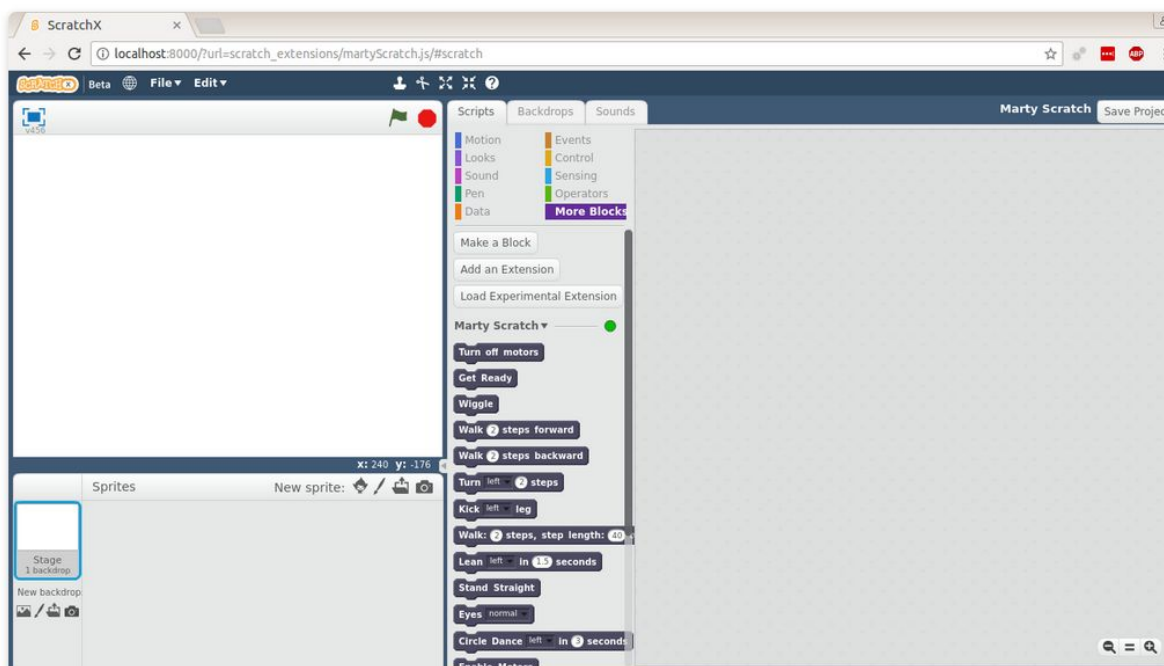
Scratch benutzt *Adobe Flash*, wenn du also nur die Scratch Homepage siehst, stelle sicher, dass dein Browser Flash unterstützt (du musst auch zulassen, dass Flash ausgeführt wird). Die Scratch Erweiterung wird nun nach Marty suchen.

Manchmal hindern Adblocker Scratch daran, deinen Marty zu finden. Falls du also Probleme damit haben solltest, schalte deinen Adblocker aus und versuche es noch einmal.

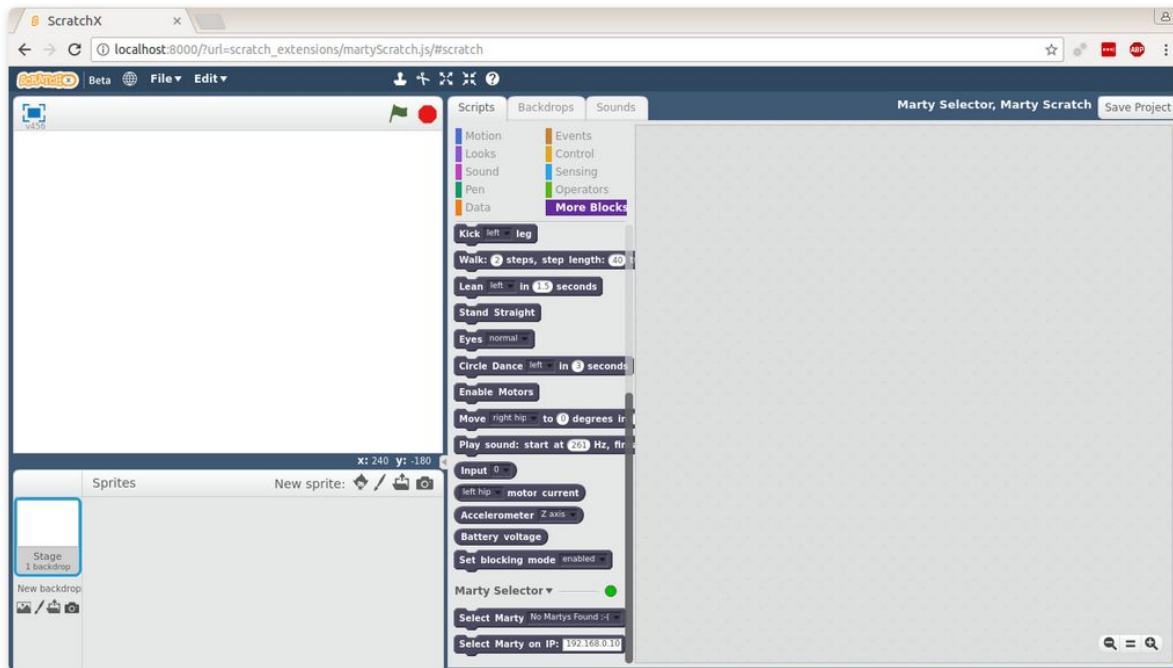


Es gibt einen kleinen bunten Kreis, der den Verbindungsstatus anzeigt (siehe Bild oben). Wenn er gelb ist, versucht Scratch gerade, sich mit deinem Marty zu verbinden.

Wenn alles klappt, ändert sich die Statusanzeige in einen grünen Kreis - das bedeutet, dass es deinen Marty gefunden hat. Yeah!



Wenn Scratch keinen oder mehrere Martys findet, erscheinen einige weitere Optionen unter dem Feld "Marty Selector":



Das gibt dir die Möglichkeit, auszuwählen, welchen Marty du benutzen willst oder, wenn keiner gefunden wurde, ihn über die IP-Adresse zu suchen.

Wenn ScratchX deinen Marty überhaupt nicht finden kann, vergewissere dich, dass er eingeschaltet und mit dem WLAN verbunden ist.

Benutze vorgefertigte Befehle

Jetzt, wo wir startklar sind, ist es Zeit, ein bisschen auszuprobieren!

Scratch ist super, weil du ganz einfach vorgefertigte Befehle per Drag-and-Drop-System zusammenbauen kannst.

Der *Get Ready* Befehl startet Martys Motoren, lässt Marty eine stehende Position einnehmen und mit den Augenbrauen zucken.

Die *walk*-Befehle lassen Marty laufen und der *wiggle*-Befehl lässt Marty wackeln.

Das ist alles nicht besonders schwierig.



Trotzdem gibt es einige Dinge über Marty, die man nicht vergessen sollte:

- Martys Motoren sind standardmäßig ausgeschaltet. Bevor du ihnen Befehle geben kannst, musst du sie einschalten. Benutze dazu entweder den *Get Ready* Befehl (der Marty auch gerade stehen und seine Augenbrauen zucken lässt) oder benutze den *Enable Motors* Befehl (der die Motoren nur bereit macht, Befehle auszuführen).
- Marty hat einige Schutzfunktionen, um ihn vor Schaden zu bewahren. Diese beinhalten Schutz bei Stürzen und Schutz bei Überlast:
 - **Schutz bei Stürzen:** Wenn Marty glaubt, dass er fällt, stellt er seine Motoren aus.
 - **Schutz bei Überlast:** Wenn zu viel Strom in einem Motor gemessen wird, geht der Motor aus.
- Sollte dein Marty jemals deine Befehle verweigern, dann versucht er vielleicht nur seine Motoren zu schützen!
- Marty mag es gern bequem. Am Ende jeder Befehlsfolge kann es sein, dass er seine Position ein bisschen ändert, um seine Motoren zu entlasten - sonst wird er müde. Marty wird es auch manchmal langweilig, weshalb er alle paar Minuten eine kleine Bewegung macht.

Versuche es einfach mal! Suche dir einige Befehle heraus und führe sie aus (Benutze einen *Event* Befehl oder mache einen Doppelklick darauf), damit Marty anfängt, sich zu bewegen!

Beginne mit Python



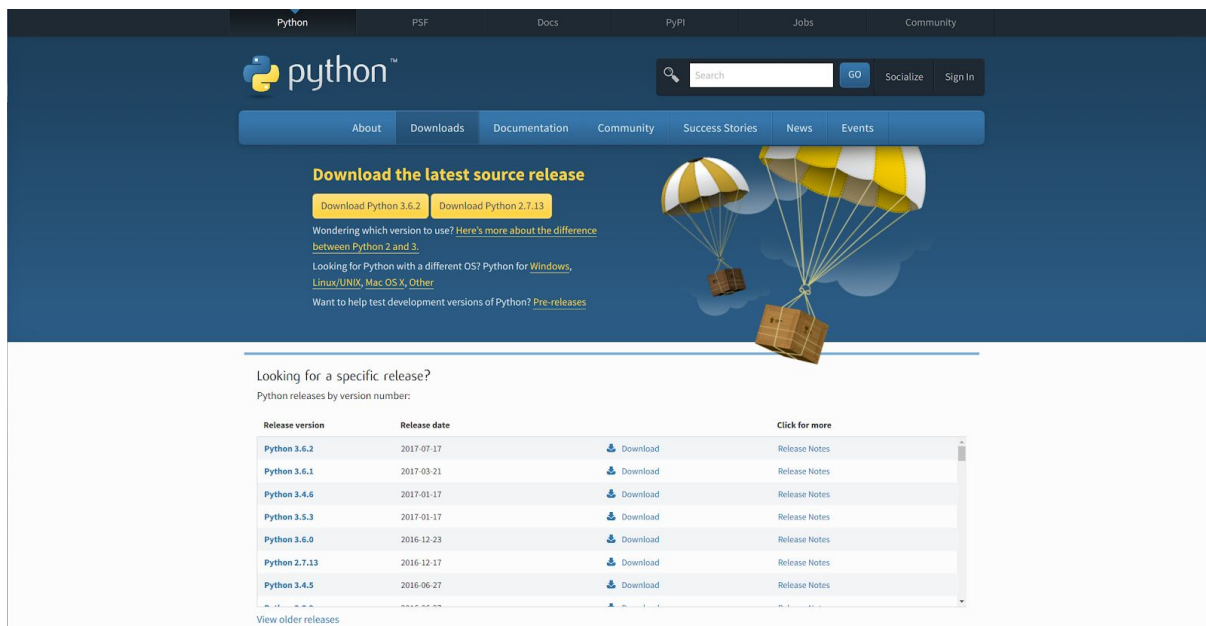
Installiere Python

Python ist eine Programmiersprache, die auf deinem Computer installiert werden muss bevor du sie benutzen kannst. Wie du das machst, hängt von dem Betriebssystem deines Computers ab. Hier sind einige gute Anleitungen, die dir zeigen, wie du Python auf deinem Betriebssystem installieren kannst oder wie du es benutzt, wenn es schon installiert ist (wie zum Beispiel auf einem RaspberryPi):

- Python - Raspberry Pi Setup
(<https://www.raspberrypi.org/documentation/usage/python/README.md>)
- Installing Python on Windows (How-To Geek)
(<https://www.howtogeek.com/197947/how-to-install-python-on-windows/>)
- Get Your Mac Ready for Python Programming (PyLadies)
<http://www.pyladies.com/blog/Get-Your-Mac-Ready-for-Python-Programming/>

MartyPy installieren

MartyPy ist eine Python Library für Marty. Libraries sind einfach nur Sammlungen von Codes, die wir geschrieben haben, damit der komplizierte Teil bereits erledigt ist und es einfacher wird, zu programmieren. Zum Beispiel enthält MartyPy die Definition der "walk"-Funktion, die Marty gehen lässt. Wenn du noch nie vorher mit Python gearbeitet hast, lohnt es sich, eine der vielen Anleitungen oder Videos online anzuschauen. Python ist eine der einfachsten Programmiersprachen und eine gute Herausforderung, nachdem man Scratch beherrscht.



The screenshot shows the Python.org website. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is a search bar and a 'GO' button. A main navigation bar contains links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a section titled 'Download the latest source release' with two buttons: 'Download Python 3.6.2' and 'Download Python 2.7.13'. Below these buttons are several links for more information, including 'Wondering which version to use? Here's more about the difference between Python 2 and 3.', 'Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other', and 'Want to help test development versions of Python? Pre-releases'. To the right of this text is an illustration of two parachutes with cargo boxes. Below the main content area, there is a section titled 'Looking for a specific release?' with the text 'Python releases by version number:'. This section contains a table with the following data:

Release version	Release date	Click for more
Python 3.6.2	2017-07-17	Download Release Notes
Python 3.6.1	2017-03-21	Download Release Notes
Python 3.4.6	2017-01-17	Download Release Notes
Python 3.5.3	2017-01-17	Download Release Notes
Python 3.6.0	2016-12-23	Download Release Notes
Python 2.7.13	2016-12-17	Download Release Notes
Python 3.4.5	2016-06-27	Download Release Notes

At the bottom of the table, there is a link 'View older releases'.

Du musst Python und Python Virtual Environments auf deinem Computer installiert haben. Python Virtual Environments oder venvs werden zusammen mit Python in SOE Systemen heruntergeladen. Ansonsten müssen sie ggf. in einem zweiten Schritt installiert werden.

Wir empfehlen, die aktuelle Version von Python 3 zu installieren. Wenn du Python 2 benutzt, was auch unterstützt wird, solltest du pyenv anstelle von virtualenv benutzen.

Wenn du ein Linux-System nutzt, benutze den Paketmanager, den du normalerweise nutzen würdest um Python und pyenv zu installieren. Allerdings wirst du höchstwahrscheinlich bereits eine Version von Python installiert haben.

Auf einem **Mac** ist die Installation etwas komplizierter, vorausgesetzt, du hast Homebrew und XCode noch nicht installiert. Auch hier empfehlen wir, eine Anleitung im Internet dafür zu finden und danach wieder zurückzukommen.

Windows-Nutzer können Python unter python.org herunterladen.

Kleiner Hinweis:

wenn du keine Lust hast, MartyPy "richtig" zu installieren, indem du ein virtualenv benutzt, kannst du einfach `pip install martypy` eintippen und es wird wahrscheinlich funktionieren. Wenn du Python für nichts anderes benutzt oder komplett neu auf dem Gebiet bist, dann ist es wahrscheinlich angenehmer für dich, den einfachen Weg zu wählen.

Öffne ein Terminal (Eingabeaufforderung). Wir möchten aus einem Projektordner arbeiten, also müssen wir zuerst ein leeres Verzeichnis erstellen und öffnen:

Wir erstellen einen Projektordner, den du nennen kannst, wie du willst - ich nehme `marty-py-tutorial`:

Linux/Mac/UNIX

```
you@computer:~$ mkdir marty-py-tutorial
```

Windows

```
C:\Users\Username>MKDIR marty-py-tutorial
```

Das Verzeichnis öffnen...

Linux/Mac/UNIX

```
you@computer:~$ cd marty-py-tutorial
```

Windows

```
C:\Users\Username>CHDIR marty-py-tutorial
```

Erstelle ein neue virtuelle Umgebung (VENV, Virtual Environment):

Linux/Mac/UNIX

```
you@computer:martypy-tutorial$ pyvenv VENV
```

Windows

```
C:\Users\Username\martypy-tutorial>pyvenv VENV
```

Und "aktiviere" es - Du musst dein Virtual Environment jedes Mal aktivieren, wenn deinen Code oder MartyPy benutzen möchtest und du das noch nicht in einem Fenster gemacht hast.

Linux/Mac/UNIX

```
you@computer:martypy-tutorial$ source ./VENV/bin/activate
```

Windows

```
C:\Users\Username\martypy-tutorial>source .\VENV\bin\activate
```

Jetzt können wir MartyPy installieren:

Linux/Mac/UNIX

```
(VENV) you@computer:martypy-tutorial$ pip install martypy
```

Windows

```
C:\Users\Username\martypy-tutorial>pip install martypy
```

Sobald der letzte Schritt erledigt ist, sollte MartyPy installiert sein. Um dies zu überprüfen, öffnen wir eine *Python Shell* und versuchen, die Library zu importieren.

```
$ python
Python x.y.z (default, Jan 1 1970, 00:00:01)
Type "help", "copyright", "credits" or "license" for more information.
>>> import martypy
>>>
```

Wenn keine Fehler angezeigt werden, hast du MartyPy erfolgreich installiert! Wenn du einen Fehler angezeigt bekommst (oder *Exception*), der so aussieht:

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named martypy
>>>
```

kann Python MartyPy nicht finden. Das liegt wahrscheinlich daran, dass entweder das Virtual Environment nicht richtig ausgeführt wurde oder es bei der Installation (`pip install martypy`) on MartyPy ein Problem gab.

Eine Verbindung zu Marty über WLAN herstellen

Die gewöhnlichste Art MartyPy zu benutzen ist, mit Marty über WLAN zu kommunizieren. Dafür musst du die **IP-Adresse** deines Roboters wissen. Du kannst dieses Online Tool benutzen, um Marty in deinem Netzwerk zu finden:

<http://docs.robotical.io/hardware/esp-socket-discovery/>

Jetzt, wo du die IP-Adresse deines Martys gefunden hast, können wir ein kurzes Script schreiben, um deinen Marty zu starten. Öffne ein Pythonscript in einem Editor deiner Wahl - wir empfehlen *Atom*, *Sublime Text*, *nano* oder *gedit*. Die letzteren beiden Editoren sind unter Linux/Ubuntu verfügbar. Du kannst aber auch ein IDE (Integrated Development Environment) für Python benutzen, z.B. *IDLE*, *Eclipse* oder *PyCharm*.

Kopiere und füge den folgenden Code in dein Dokument ein und ändere die IP-Adresse in die, die du mit unserem Online Tool gefunden hast

```
1 import martypy
2 mymarty = martypy.Marty('socket://192.168.0.x') # Ändere die IP
3 mymarty.hello() # Bewegt sich in die Ausgangsposition und zwinkert
```

Speichere das als `mymarty.py` in dem `marty-py-tutorial` Ordner, den wir vorhin erstellt haben. Jetzt können wir den Code ausführen! Wenn du ein IDE benutzt, gibt es wahrscheinlich einen Button dafür. Ansonsten öffne ein Terminal, gehe zum `marty-py-tutorial` Ordner, stelle sicher, dass du das Virtual Environment aktiviert hast und tippe dann folgendes und drücke anschließend die Enter-Taste.

```
$ python ./mymarty.py
```

Marty sollte sich in seine Ausgangsposition bewegen und zwinkern. Das bedeutet, dass er aufrecht und gerade steht.

Noch ein bisschen mehr machen

Genauso wie in den letzten Schritten, öffne die `mymarty.py` Datei. Wir fügen jetzt ein bisschen mehr hinzu, was etwas interessanter als nur ein Augenzwinkern wird - Marty wird gleich einige Schritte laufen!

Der Befehl dazu heißt "walk". Wir fügen noch ein bisschen mehr Code hinzu, sodass Marty jedes Mal, wenn du die Enter-Taste drückst, einige Schritte vorwärts läuft.

```
1 import martypy
2 mymarty = martypy.Marty('socket://192.168.0.x') # Ändere die IP
3
4 print("Hit 'Enter' to walk two steps forward.")
5 print("Press Control + C to exit.")
6 while True:
7     input()
8     mymarty.walk()
```

Weitere Beispiele...

Lass Marty ein Geräusch machen:

```
1 import martypy
2 mymarty = martypy.Marty('socket://192.168.0.x') # Ändere die IP
3 mymarty.play_sound(8000, 200, 1000) # Mach ein Geräusch
```

```
1 import martypy
2 mymarty = martypy.Marty('socket://192.168.0.x') # Ändere die IP
3 mymarty.play_sound(8000, 200, 1000) # Mach ein Geräusch
```

Eine Raspberry Pi an Marty anschließen



Erste Schritte mit deiner Pi

Achtung: Diese Anleitung setzt ein oberflächliches Wissen über Linux voraus

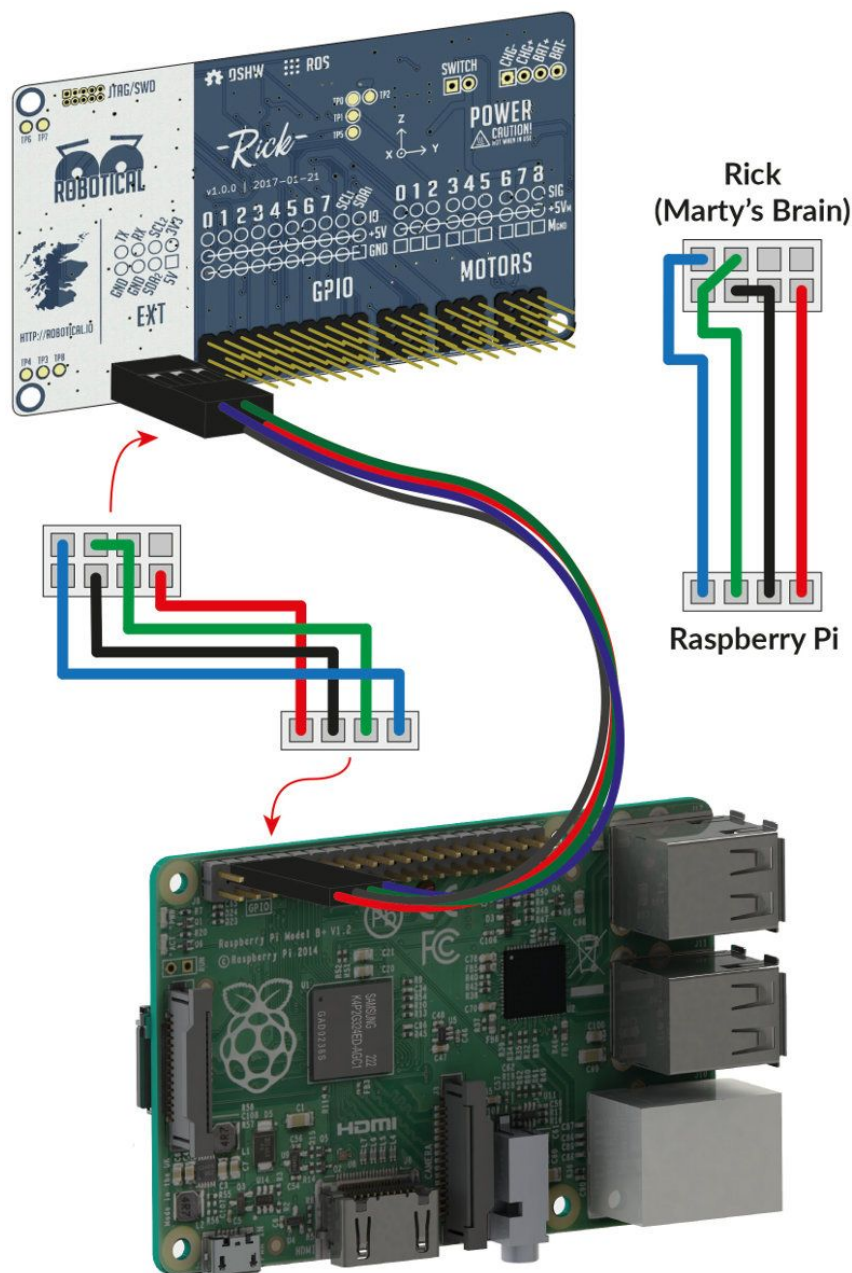


Schaubild für die Verbindung von Rick und einer RaspberryPi

Ein Raspberry Pi ist ein Einplatinenrechner, der ursprünglich entwickelt wurde, um das Programmieren zu erlernen und den Informatikunterricht zu unterstützen. Aufgrund der kleinen Größe, dem Preis und der Funktionalität kann ein Raspberry Pi die Fähigkeiten deines Roboters deutlich erweitern, wenn dieser an Marty angeschlossen wird. Tatsächlich wird Marty dadurch zu einem vollwertigen Computer, sodass er dadurch einige ziemlich komplexe Dinge machen kann.

Ein Pi Image installieren

Wir haben ein fertiges Image für deine RaspberryPi und eine Menge andere Programmier-Tools, die mit ROS funktionieren. Du kannst sie hier downloaden: <https://cdn.robotical.io/public/raspberry-pi-img/>

Für weitere Informationen, wie man ein Image auf eine SD-Karte schreibt, kannst du die offizielle Raspberry Pi-Website besuchen, die dieses Thema gut erklärt.

Einen Pi selbst konfigurieren

Wenn du unser offizielles Raspbian Image nicht benutzen möchtest, dann ist es möglich, alle Pakete selbst zu installieren. Allerdings ist das nur sinnvoll, wenn du ein erfahrener Nutzer bist.

Eine Schritt-für-Schritt-Anleitung gibt es hier nicht, allerdings sind die einzelnen Schritte gut im Internet zu finden. Die wichtigsten Schritte sind die Folgenden:

1. Installiere die aktuelle Version von Raspbian Jessie
2. Installiere ROS. Im Moment wird nur Indigo unterstützt, die Leistung von anderen ROS Distros kann variieren.
3. Installiere Marty-spezifische Abhängigkeiten auf deiner Pi

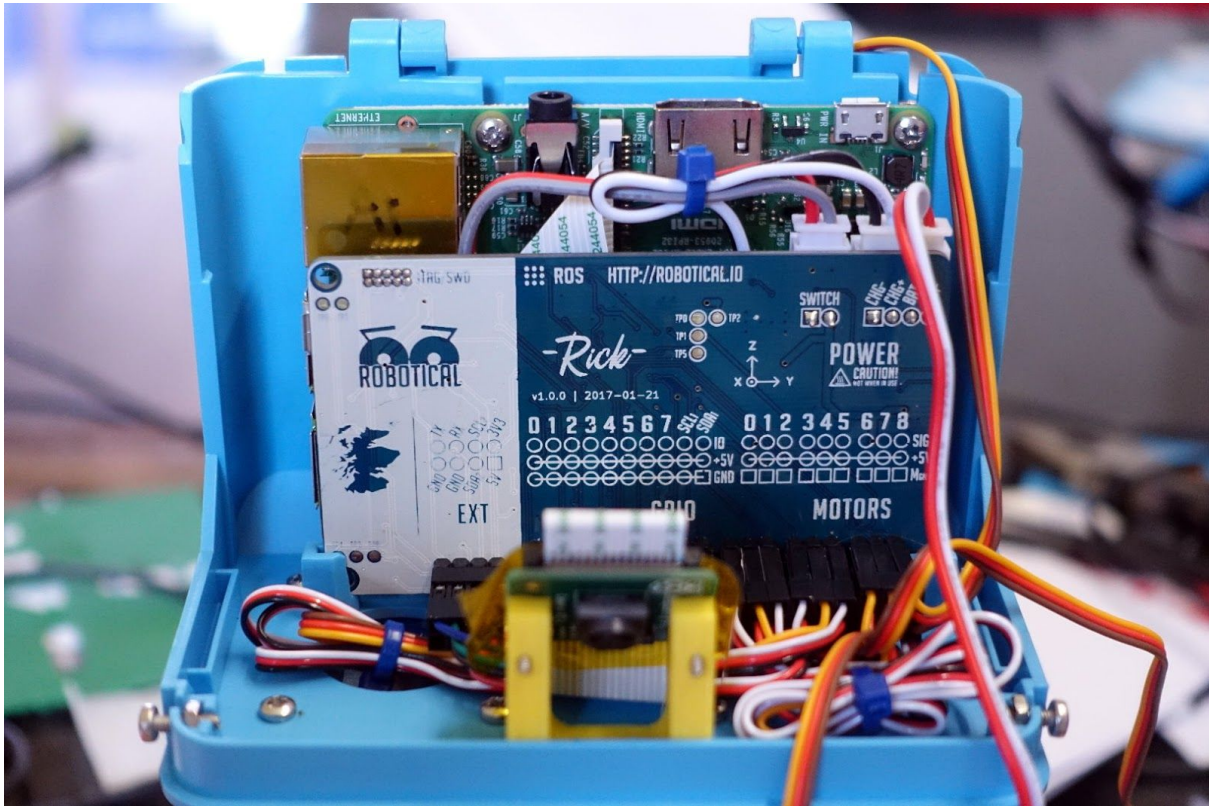
Einen Raspberry Pi in Martys Kopf anschließen

Martys Kopf hat Steckplätze und einen Slot hinter Rick, der dafür gedacht sind, einen Raspberry Pi (1, 2, 3 oder Zero) anzuschließen. Du kannst zwei M2.5 Schrauben und Muttern benutzen, um deinen Pi zu fixieren. Wenn du Zugang zu einem 3D-Drucker hast, haben wir auch einen Stecker für die Raspberry Pi Kamera designed, die aus dem Mund heraus schauen kann. Stecke das gefaltete Pi-Kamera-Kabel vorsichtig unter Rick, um sie zu verbinden.

Schau dir das Bild oben an, um zu wissen, wie man eine Pi an Rick anschließt. Die Pi braucht kein extra Mikro-USB-Kabel, da sie so auch mit dem Stromanschluss von Rick verbunden ist. Pass auf, dass du den Ethernet-Port gut isolierst, da der kleine JTAG-Port des Rick Boards einen Kurzschluss verursachen kann, wenn sie sich berühren. Weil der Raspberry Pi nun auch mit Strom versorgt wird, ist die Akkulaufzeit deines Marty kürzer. Du

kannst den Akku durch einen ersetzen, der eine größere Kapazität hat. Kontrolliere aber die Angaben genau, damit der andere Akku nicht gefährlich ist und deinen Marty beschädigt.

Je nachdem, was für eine Art Akku du auswählst, kann es sein, dass du noch ein anderes Ladegerät brauchst, da das Ladegerät für Rick zu schwach sein könnte.



Ein Raspberry Pi & Raspberry Pi Kamera in einem Marty

Netzwerk Setup

SSH

Um deinen Marty für den ersten Gebrauch zu konfigurieren, muss eine Verbindung hergestellt werden, um Befehle zu geben. Der einfachste Weg ist, eine USB-Tastatur und einen HDMI-Monitor an den Pi anzuschließen und die Befehle per Hand einzugeben. Für längere Arbeiten ist das jedoch unpraktisch, vor allem, wenn Marty sich bewegt. Eine bessere Lösung wäre es, auf die Kommandozeile von einem anderen Computer über ein Netzwerk zuzugreifen. Das klappt mit dem Secure Shell (SSH) Protokoll.

Die meisten Linux Distributionen und OSX Versionen haben SSH vorinstalliert, allerdings kann es sein, dass man für Windows einen Third-Party-Client wie PuTTY benötigt. Die nächsten Schritte gehen davon aus, dass du Linux benutzt (die Schritte sind für OSX sehr ähnlich). In Windows können sie anders sein, je nachdem, was für einen SSH Client du benutzt.

Verbinde deinen Computer mit dem WLAN, das Marty erstellt. Öffne ein Terminal und verbinde dich über SSH mit deinem Marty:

```
ssh pi@marty.local
```

Du wirst dann aufgefordert werden, ein Passwort einzugeben (marty). Eventuell musst du noch die Authentizität des Hosts bestätigen. Eine kurze Erklärung für diejenigen, für die SSH neu ist: Du hast gerade eine Verbindung von deinem Computer zur Raspberry Pi in Marty gemacht und hast somit volle Kontrolle. Du kannst das kontrollieren, indem du die Benutzereingabeaufforderung im Terminal kontrollierst (da sollte pi@marty stehen). Alle Befehle, die jetzt in das Terminal eingegeben werden, werden so behandelt, als ob sie in einem Terminal auf dem Pi selbst eingegeben worden wären.

Hinweis zum WLAN:

Wenn du einen Pi hast, die älter als ein Pi 3 Modell B ist, also *ohne* eingebautes WLAN, musst du den SSH Teil vorerst überspringen. Du kannst die Schritte später nachholen, aber du wirst deinen Pi an einen Monitor anschließen müssen und eine USB-Tastatur benutzen.

Das liegt daran, dass das Modell B eingebautes WLAN hat. Kombiniert mit einem nano USB-Dongle, kann ein Pi seinen eigenen Hotspot kreieren und sich mit einem anderen Netzwerk verbinden. Nachdem du die Netzwerk-Konfiguration abgeschlossen hast, kannst du dich über SSH auf deinen Pi verbinden, da dieser mit demselben Netzwerk wie dein Computer verbunden ist.

Netzwerk Konfiguration

Jetzt müssen wir deinem Pi die Netzwerk-Informationen geben. Gib folgendes ein:

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Das wird den Nano Text Editor in deinem Terminal öffnen. Hier solltest du deine Netzwerkinformationen in folgendem Format eingeben:

```
network={
  ssid="your-network"
  psk="your-network-password"
  key_mgmt=protocol
}
```

„your-network“ und „your-network-password“ sind der Name und das Passwort deines Netzwerks. **Die Anführungszeichen sind unbedingt notwendig.** „protocol“ ist der Sicherheitsschlüssel deines Netzwerks, oft ist das WPA-PSK. Stelle sicher, dass das Netzwerk dasselbe wie das ist, mit dem dein Computer verbunden ist. Wenn du fertig bist, **speichere**, schließe das Fenster und starte deinen Marty neu.

Beim Neustart sollte sich dein Marty automatisch mit dem Netzwerk verbinden.

Zum Testen kannst du ein IP Scanning Tool wie zum Beispiel „AngryIP“ benutzen, um zu sehen, ob dein Marty im Netzwerk ist.

Die nächsten Schritte

Jetzt, da du die Basics kennst, gibt es keine Grenzen mehr! Mit einer Pi in Marty hast du eine ganze Menge neuer Funktionen!

Wenn du Lust auf ein bisschen schwierigere Dinge hast, schaue dir an, wie du Marty mit ROS steuern kannst.

Upgrades!

Wenn du das große Glück hast und einen 3D-Drucker nutzen kannst, kannst du auch einen Mund mit Stecker für eine RPi Kamera drucken (unter www.robotical.io > 3D Print, am Ende der Seite).

Ein RaspberryPi im Kopf von Marty wird den Stromverbrauch deines Roboters erhöhen und somit die Akkulaufzeit verkürzen.

Allerdings kannst du deinen Akku upgraden - die meisten LiPo-Akkus (2 Zellen, 7,4V, ~2C oder größer) passen in deinen Marty.

Wenn du deinen Akku mit dem Ladegerät

Warnung:

Eine generelle Faustregel ist, dass je schwieriger und komplexer du deinen Marty benutzt und steuerst, desto höher wird auch das Risiko, dass etwas in deinem Marty kaputt geht. Bitte ändere Systeminstellungen deines Pis nur *vorsichtig* und passe auf, wenn du mit anderer Hardware arbeitest! Wenn nichts mehr geht, kannst du im Zweifelsfall immer noch ein neues Image laden.

aumlädst, das bei Marty dabei war, beachte, dass du die volle Verantwortung für die Sicherheit tragst und dass wir nicht garantieren konnen, dass das eingebaute Ladegerat fur deinen Akku funktioniert. Befolge immer die Sicherheitsrichtlinien des Herstellers.

Begebe dich mit ROS in die Tiefen von Marty



ROS: Das Robot Operating System

Mit einem Pi im Kopf von Marty und damit einem Boost von rechnerischen Fähigkeiten, kannst du komplexere Features ausprobieren. Eine der besten Wege, diese Leistung auszunutzen ist die Nutzung des Robot Operating Systems - ROS. Du kannst von der Community und der Website www.answers.ros.org profitieren und mehr über Roboter lernen und wie man coole Anwendungen für sie entwickelt.

Okay, cool, aber was genau *ist* ROS?

Im Wesentlichen ist ROS ein flexibles Framework, um die Entwicklung von Roboter-Software zu erleichtern. Es ist ein Pseudo-Operating-System, das aus vielen Software-Tools und Libraries besteht und somit Roboterentwicklern das tägliche Leben erleichtert. Eine der größten Vorteile ist, dass das Framework auf Open-Source basiert. Das wurde ursprünglich so entwickelt, damit mehrere Gruppen zusammen arbeiten können und das Endergebnis durch die unterschiedlichen Stärken und Fähigkeiten der Gruppen effektiver und stabiler ist. Dadurch können Einzelpersonen oder Gruppen ihre eigenen Tools und Libraries schaffen und sie zur Nutzung für eine größere ROS Community freigeben. Dieses Baukastenprinzip macht es angehenden Robotikern (was du wahrscheinlich bist, wenn du das hier liest) einfacher, da es dir erlaubt so viel von ROS zu benutzen, wie du magst. Du kannst einfach aussuchen, was am besten zu deiner Anwendung passt.

Und wie funktioniert es?

Zuerst kümmern wir uns zuerst um die Basics von ROS.

ROS basiert auf dem Prinzip des verteilten Rechnens und ist eine Art *Peer-to-Peer distributed System*. Ein ROS-System - nicht unbedingt nur eine einzelne Maschine - ist ein *Netzwerk* aus unabhängigen *Prozessen* (unten auch *node* genannt), die untereinander kommunizieren, indem sie mit einem Sender(Publisher)/Empfänger(Subscriber)-Modell *Nachrichten schicken*.

Ein *Prozesse* kann seine Daten über ein *Topic* senden, die dann von einem anderen *Prozess*, der sich dafür interessiert, aufgenommen werden kann. Ein Kameraprozess kann zum Beispiel eine Menge Bilder senden, die dann von einem anderen *Prozess* aufgenommen werden können, der die Bilder dann weiterverarbeitet. Es gibt keine Maximalanzahl an

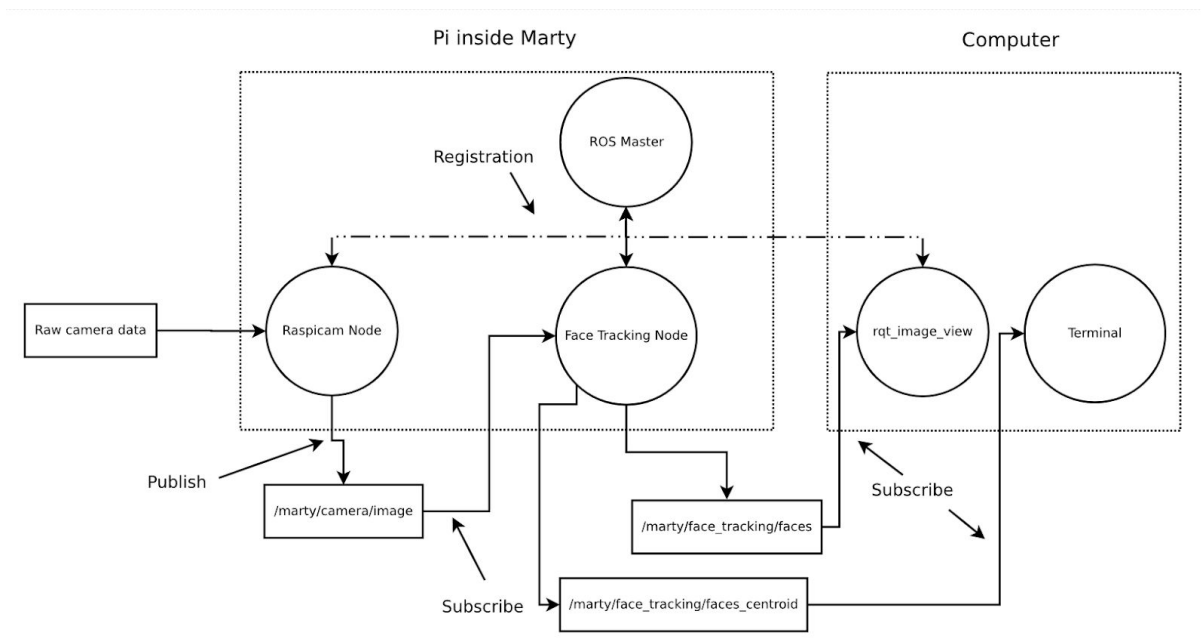
INFO:

Dir eine ausführliche Aufschlüsselung zu geben, was alles in ROS möglich ist, würde eine lange Zeit dauern und definitiv den Rahmen dieser kurzen Einführung sprengen. Du kannst dir auf www.ros.org/core-components/ einen Überblick über die Kern-Komponenten verschaffen. Im ROS Wiki (wiki.ros.org/) gibt es außerdem eine Reihe an Anleitungen und Aufzeichnungen.

Empfängern, die ein Topic haben kann, egal ob der Prozess auf demselben Computer oder dieselbe Programmiersprache verwendet. Dieses verteilte Modell ist super praktisch und kann leicht mehrere Geräte und Betriebssysteme umfassen und verbinden. Um Marty als Beispiel zu nehmen: Rick (eine Platine auf Arduino-Basis) kommuniziert mit der Pi durch serielle Kommunikation, indem er roserial benutzt.

Ich glaube, so langsam verstehe ich es...

Mache dir keine Sorgen, wenn das jetzt noch nicht alles Sinn für dich ergibt. Lass es uns mal bildlich darstellen. Wir benutzen Martys Gesichtsverfolger (Face Tracker) als Beispiel.



System Diagram für Martys Gesichtsverfolgerr

Es ist eine vereinfachte Grafik, aber relativ präzise. Lass uns noch ein bisschen vereinfachen, was hier passiert.

Der ROS Master steuert alles. Dieser Node(Prozess) ist essentiell, damit ein ROS-System funktioniert. Seine Aufgabe ist es, die anderen Nodes im System zu benennen und zu registrieren. Er passt auf die verschiedenen Publisher, Subscriber und Services auf, die im ganzen System laufen.

Wenn ein Node beim Master registriert wurde, kann er mit allen anderen Nodes kommunizieren, die beim Master registriert wurden.

Nun zu unserem Node "raspicam_node". Das ist der Node, der Bilder von der Raspberry Pi Kamera erhält und sie dann über das Topic "/marty/camera/image" für andere Nodes zugänglich macht.

Diese Daten werden dann vom Node "face_tracker" über das Topic "/marty/camera/image" empfangen und verarbeitet (zum Beispiel das Erkennen von Gesichtern, Augen und Lächeln).

Sobald dieser Schritt fertig ist, teilt dieser Knoten die Ergebnisse der verarbeiteten Bilder, wie z.B: erkannte Gesichter oder das geographische Zentrum eines Gesichtes. Diese Infos sind über die Topics: /marty/face_tracking/faces und /marty/face_tracking/faces_centroid) verfügbar

Das Ergebnis davon ist in diesem Fall ein einfaches Rechteck, was über die erkannten Merkmale gelegt wird. Das passiert auf dem Laptop, der für das ROS-System genutzt wird. Dieses Ergebnis kann man anzeigen, indem man rqt_image_view startet und das topic "/marty/face_tracking/faces_centroid" des "face_tracker" Nodes auswählt.

Es ist ein relativ einfaches Beispiel, aber es zeigt die Struktur einer typischen ROS-Anwendung.

Erwähnenswert ist das vorher genannte Baukastenprinzip von ROS. Es muss nicht unbedingt eine eingebaute Kamera sein, von der der face_tracker Node die Bilder empfängt, sondern es kann auch eine andere Kamera im Netzwerk sein (solange der Name des erhaltenen topic derselbe ist).

Außerdem ist es super einfach, Daten von diesem Node zu nutzen oder erweitern. Du musst einfach nur das benötigte topic empfangen und dann hast du die Daten. Zum Beispiel könntest du ein Script haben, das die zugehörigen Daten des /marty/face_tracking/faces_centroid topic empfängt und so Marty dazu bringt, zu laufen um dein Gesicht zu erfassen. Probiere es aus, wir freuen uns, es zu sehen!

Cool! Wo fange ich an?

Jetzt, da die Basics erklärt wurden, ist der nächste Schritt die Praxis.

Das offizielle Marty Pi Image hat bereits eine Version von ROS installiert, also kannst du den anstrengenden Part (das Installieren von ROS) überspringen.

Bevor du aber richtig loslegst, ist es wirklich sinnvoll, ROS noch auf einem anderen Computer zu installieren. Das wird das Entwickeln deutlich schneller und alles andere viel leichter machen. Wenn es darum geht, Roboter Anwendungen zu entwickeln, ist Linux am besten dazu geeignet. Es ist also quasi eine Voraussetzung, ein Betriebssystem wie zum Beispiel Ubuntu zu benutzen. Es ist eines der weit verbreitetsten Linux Distributionen und ist

offiziell von ROS unterstützt. Es gibt Versionen von ROS, die man zwar auch mit Windows und OSX "benutzen kann", aber die sind noch in der Testphase, fehlerhaft und nicht zu empfehlen.

Sobald du Ubuntu und ROS auf einem Computer deiner Wahl installiert hast, ist es am besten, wenn du dir zuerst die offiziellen ROS Anleitungen (wiki.ros.org/ROS/Tutorials) anschaust. Die werden dir alles erklären, was ein Anfänger wissen muss, damit Marty dann coole Sachen macht.